



Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

*Generación de datos sintéticos para entrenamiento de sistemas
basados en aprendizaje automático supervisado*

Grado en Ingeniería Mecánica

ALUMNO: Marta Hidalgo Sánchez

DIRECTORES: Belén Barragáns Martínez
Pablo Sendín Raña

CURSO ACADÉMICO: 2024-2025

Universida_{de}Vigo



Centro Universitario de la Defensa en la Escuela Naval Militar

TRABAJO FIN DE GRADO

*Generación de datos sintéticos para entrenamiento de sistemas
basados en aprendizaje automático supervisado*

Grado en Ingeniería Mecánica
Intensificación en Tecnología Naval
Cuerpo General

Universida_{de}Vigo

RESUMEN

Este trabajo se centra en la generación de datos AIS (*Automatic Identification System*) estáticos de forma sintética para el entrenamiento de modelos de aprendizaje automático supervisado, en contextos donde los datos reales son escasos o están desbalanceados. En particular, los datos sintéticos generados tendrán como finalidad mejorar el entrenamiento de un sistema de predicción de tipo de buque.

Se realiza una revisión de diversas técnicas de generación de datos sintéticos y, considerando el tipo de dato que se desea generar, se seleccionan TVAE, *Gaussian Copula* y CTGAN. La generación se realiza tanto de forma global como segmentada por tipo de barco, con el objetivo de preservar las proporciones reales y las relaciones internas de los datos. Los datos sintéticos obtenidos se someten a una intensa fase de evaluación que incluye métricas estadísticas (KS-test, correlaciones, chi-cuadrado), visualizaciones (PCA, t-SNE, KDE), y pruebas de utilidad mediante el modelo de clasificación. Además, se discute el impacto del tamaño de muestra sobre las pruebas estadísticas y la influencia de los atributos en el entrenamiento de modelos de clasificación. El estudio concluye que la generación controlada por clase mejora la fidelidad de los datos sintéticos y puede ser clave en entornos con datos desequilibrados.

PALABRAS CLAVE

Datos sintéticos, Aprendizaje automático, AIS, TVAE, CTGAN

AGRADECIMIENTOS

Quiero comenzar estas palabras de agradecimiento dirigiéndome a mi familia que siempre ha sido mi apoyo firme. Gracias por acompañarme y aguantarme incluso desde la distancia durante estos cinco años. Especial mención a mi hermano por haberme enseñado este mundo, y por ser siempre un ejemplo a seguir para mí. Sin vosotros, no sería la persona que soy hoy.

Gracias, por supuesto, a todo el personal que ha contribuido a mi formación profesional y en valores: oficiales y mandos de la Armada, así como al profesorado del Centro Universitario de la Defensa. Especial mención merecen mis tutores, Belén y Pablo, por su gran experiencia y dedicación. Gracias a su paciencia e implicación, este trabajo ha salido adelante.

A mis amigas de siempre y a mi entrenador, que aún sabiendo poco de a qué me dedico desde hace cinco años, siguen manteniendo un vínculo incondicional, apoyándome en todas mis decisiones. Gracias por estar siempre ahí, no me equivoqué confiando en vosotros. A mis compañeros y amigos de la promoción 425-155, con los que he compartido muchas experiencias enriquecedoras, pues cinco años dan para mucho. Gracias, sin vosotros no hubiera sido igual.

No puedo dejar de agradecer al equipo de voleibol de Pontevedra y al voleibol como deporte, que ha sido siempre mi vía de escape y desconexión cuando el estrés de la escuela apretaba. Gracias por los partidos y las risas, y por recordarme la importancia de las pequeñas cosas.

Por último, gracias a todos los que han estado presentes, directa o indirectamente, en este viaje. Este trabajo marca el final de una etapa inolvidable en la Escuela Naval Militar, que recordaré siempre con gratitud y emoción.

Contenido

Índice de Figuras	5
Índice de Tablas.....	11
1 Introducción y objetivos	13
1.1 Contexto y motivación	13
1.2 Objetivos	14
1.3 Estructura de la memoria	14
2 Estado del arte	17
2.1 Inteligencia Artificial	17
2.1.1 Definición	17
2.1.2 Inteligencia Artificial Generativa	19
2.2 Aprendizaje Automático (<i>Machine Learning</i>)	19
2.2.1 Aprendizaje supervisado.....	20
2.2.2 Aprendizaje no supervisado.....	20
2.2.3 Aprendizaje Profundo (<i>Deep Learning</i>)	20
2.3 Algoritmos de aprendizaje automático	21
2.3.1 Árboles de decisión.....	21
2.3.2 Redes neuronales	21
2.4 Conocimiento del Entorno Marítimo	23
2.4.1 COVAM	23
2.4.2 Sistema de Identificación Automática (AIS)	24
2.4.3 Calidad y fiabilidad de los datos AIS	26
2.5 Datos sintéticos	27
2.5.1 Definición	27
2.5.2 Aplicaciones y uso de los datos sintéticos	28
2.5.3 Beneficios del uso de datos sintéticos	29
2.5.4 Retos y riesgos asociados a la generación de datos sintéticos.....	30
2.5.5 Datos sintéticos AIS para el entorno marítimo	31
2.6 Enfoques de la generación de datos sintéticos	31
2.6.1 Según la técnica	32
2.6.2 Según la estructura de los datos.....	32
2.6.3 Según el propósito de los datos	34
2.7 Técnicas de generación de datos sintéticos.....	34
2.7.1 Redes Generativas Adversarias (GANs)	35

2.7.2 Autocodificadores (AEs)	37
2.7.3 <i>Gaussian Copula</i>	39
2.7.4 <i>Synthetic Minority Oversampling Technique (SMOTE)</i>	39
2.8 Selección de metodología	40
2.9 Estudios relacionados.....	40
3 Desarrollo del TFG.....	43
3.1 Entorno de trabajo	43
3.1.1 <i>Jupyter Lab</i>	43
3.1.2 Bibliotecas en Python	44
3.2 Flujo de trabajo	45
3.3 Análisis de los datos disponibles	47
3.4 Generación de datos estáticos sintéticos	50
3.4.1 Sin agrupar por tipo de buque.....	50
3.4.2 Con agrupación por tipo de buque.....	51
3.5 <i>Data Augmentation</i>	57
4 Resultados obtenidos y validación	59
4.1 Métricas de calidad empleadas	59
4.1.1 <i>Kolmogorov-Smirnov (KS-Test)</i>	59
4.1.2 Chi-cuadrado.....	60
4.1.3 Estadísticos descriptivos.....	60
4.1.4 Gráficos de densidad.....	61
4.1.5 Matriz de correlación.....	61
4.1.6 PCA.....	61
4.1.7 t-SNE	61
4.2 Validación de utilidad	62
4.3 Análisis de resultados.....	64
4.3.1 Resultados de los datos sintéticos estáticos sin agrupar por tipo de buque	64
4.3.2 Resultados de los datos sintéticos estáticos por tipo de buque	74
4.3.3 Rendimiento en aprendizaje automático.....	92
4.3.4 Análisis de rendimientos de computación de cada técnica.....	97
4.3.5 Resultados de la prueba de <i>Data Augmentation</i>	99
5 Conclusiones y líneas futuras	103
5.1 Conclusiones	103
5.2 Líneas futuras	104
6 Bibliografía.....	105

Anexo I: Implicaciones sociales, y/o económicas, y/o ambientales.....	111
Anexo II: Reflexiones Éticas y Sociales	113
Anexo III: Enlace código completo en <i>github</i>	115
Anexo IV: Código para los datos sin agrupar por <i>shiptype</i>	117
Anexo V: Código para los datos agrupados por <i>shiptype</i>	119
Anexo VI: Código métricas de evaluación de los datos generados sin y con agrupación por <i>shiptype</i>	123
Anexo VII: Código para el cálculo de los atributos a partir de los datos sintéticos <i>basic</i> y métricas de análisis.....	131
Anexo VIII: Código para <i>Data Augmentation</i>	137
Anexo IX: Código para análisis de utilidad con <i>Random Forest</i>	141

ÍNDICE DE FIGURAS

Figura 2-1: Inteligencia Artificial, <i>Machine Learning</i> y <i>Deep Learning</i> (Fuente propia)	19
Figura 2-2: Arquitectura de una red neuronal (elaboración propia).....	22
Figura 2-3: Arquitectura de una red neuronal profunda (elaboración propia)	22
Figura 2-4: Tipos de datos sintéticos según proporción (fuente propia).....	28
Figura 2-5: Esquema de modelos generativos según el tipo de dato generado (tomada de [43])	33
Figura 2-6: Clasificación de las técnicas de generación de datos sintéticos (tomada de [39])	35
Figura 2-7: Taxonomía de GANs (fuente propia)	36
Figura 2-8: Arquitectura de un <i>autoencoder</i> (fuente propia)	37
Figura 2-9: Arquitectura de un VAE (fuente propia)	38
Figura 2-10: Ejemplo de iteración con SMOTE (fuente propia).....	40
Figura 3-1: Interfaz de <i>Jupyter Lab</i>	44
Figura 3-2: Logotipo de <i>Jupyter</i> [57]	44
Figura 3-3: Flujo de trabajo de la generación de datos sintéticos (tomada de [41])	46
Figura 3-4: Etapas de la generación de datos sintéticos (fuente propia)	46
Figura 3-5: Representación de los campos A, B, C y D [62]	48
Figura 3-6: Información de <i>final_static_data.csv</i>	49
Figura 3-7: Información de <i>final_static_data_basic.csv</i>	49
Figura 3-8: Cantidad de barcos por categoría	50
Figura 3-9: Proporción del campo <i>shiptype</i>	50
Figura 3-10: 5 primeras filas de <i>sinteticos_estaticos_TVAE_sin_distincion.csv</i>	51
Figura 3-11: 5 primeras filas de <i>sinteticos_estaticos_GaussianCopula_sin_distincion.csv</i>	51
Figura 3-12: 5 primeras filas de <i>sinteticos_estaticos_CTGAN_sin_distincion.csv</i>	51
Figura 3-13: Esquema de entrenamiento por <i>shiptype</i> (fuente propia)	52
Figura 3-14: Archivo <i>metadata_basic_shiptype_TVAE.json</i>	53
Figura 3-15: 5 primeras filas de <i>synthetic_static_basic_TVAE.csv</i>	54
Figura 3-16: 5 primeras filas de <i>synthetic_static_basic_GaussianCopula.csv</i>	54
Figura 3-17: 5 primeras filas de <i>synthetic_static_basic_CTGAN.csv</i>	54
Figura 3-18: 5 primeras filas de <i>synthetic_static_completo_TVAE.csv</i>	54
Figura 3-19: 5 primeras filas de <i>synthetic_static_completo_GaussianCopula.csv</i>	54
Figura 3-20: 5 primeras filas de <i>synthetic_static_completo_CTGAN.csv</i>	55
Figura 3-21: 5 primeras filas de <i>synthetic_static_atributos_TVAE.csv</i>	55
Figura 3-22: 5 primeras filas de <i>synthetic_static_atributos_GaussianCopula.csv</i>	55

Figura 3-23: 5 primeras filas de <i>synthetic_static_atributos_CTGAN.csv</i>	55
Figura 3-24: <i>Dataframe</i> original del archivo <i>prueba.csv</i>	56
Figura 3-25: Resultados de la prueba de suma de columnas para las tres técnicas de generación de datos sintéticos	56
Figura 3-26: Funciones de cálculo de los atributos estáticos	57
Figura 3-27: Atributos de <i>synthetic_static_basic_TVAE.csv</i>	57
Figura 3-28: Atributos de <i>synthetic_static_basic_GaussianCopula.csv</i>	57
Figura 3-29: Atributos de <i>synthetic_static_basic_CTGAN.csv</i>	57
Figura 3-30: Comparación de distribuciones tras aplicar <i>Data Augmentation</i>	58
Figura 4-1: Balance de clases para TVAE sin distinción	65
Figura 4-2: Balance de clases para <i>Gaussian Copula</i> sin distinción	65
Figura 4-3: Balance de clases para CTGAN sin distinción	65
Figura 4-4: Resultados del <i>KS-test</i> para TVAE sin distinción	65
Figura 4-5: Resultados <i>KS-test</i> para muestra de 100 datos con CTGAN sin distinción	66
Figura 4-6: Comparación % de valores únicos en los datos sintéticos sin distinción con los datos reales	66
Figura 4-7: Función densidad del parámetro <i>to_bow</i> para <i>cargo</i> sin agrupación con TVAE	67
Figura 4-8: Función densidad del parámetro <i>to_bow</i> para <i>passenger</i> sin agrupación con TVAE	67
Figura 4-9: Función densidad del parámetro <i>to_bow</i> para <i>cargo</i> sin agrupación con <i>Gaussian Copula</i>	67
Figura 4-10: Función densidad del parámetro <i>to_bow</i> para <i>passenger</i> sin agrupación con <i>Gaussian Copula</i>	67
Figura 4-11: Función densidad del parámetro <i>to_bow</i> para <i>cargo</i> sin agrupación con CTGAN	68
Figura 4-12: Función densidad del parámetro <i>to_bow</i> para <i>passenger</i> sin agrupación con CTGAN	68
Figura 4-13: Matriz de correlación para <i>cargo</i> de los datos reales	70
Figura 4-14: Matriz de correlación para <i>passenger</i> de los datos reales	70
Figura 4-15: Matriz de correlación para <i>cargo</i> de <i>basic</i> sin agrupación con TVAE	70
Figura 4-16: Matriz de correlación para <i>passenger</i> de <i>basic</i> sin agrupación con TVAE	70
Figura 4-17: Matriz de correlación para <i>cargo</i> de <i>basic</i> sin agrupación con <i>Gaussian Copula</i>	71
Figura 4-18: Matriz de correlación para <i>passenger</i> de <i>basic</i> sin agrupación con <i>Gaussian Copula</i>	71
Figura 4-19: Matriz de correlación para <i>cargo</i> de <i>basic</i> sin agrupación con CTGAN	71
Figura 4-20: Matriz de correlación para <i>passenger</i> de <i>basic</i> sin agrupación con CTGAN	71
Figura 4-21: Gráfico PCA de los datos <i>basic</i> sin agrupación con TVAE para <i>cargo</i>	72
Figura 4-22: Gráfico PCA de los datos <i>basic</i> sin agrupación con TVAE para <i>passenger</i>	72
Figura 4-23: Gráfico PCA de los datos <i>basic</i> sin agrupación con <i>Gaussian Copula</i> para <i>cargo</i>	72
Figura 4-24: Gráfico PCA de los datos <i>basic</i> sin agrupación con <i>Gaussian Copula</i> para <i>passenger</i>	72
Figura 4-25: Gráfico PCA de los datos <i>basic</i> sin agrupación con CTGAN para <i>cargo</i>	72

Figura 4-26: Gráfico PCA de los datos <i>basic</i> sin agrupación con CTGAN para <i>passenger</i>	72
Figura 4-27: Gráfico t-SNE de los datos <i>basic</i> sin agrupación con TVAE para <i>cargo</i>	73
Figura 4-28: Gráfico t-SNE de los datos <i>basic</i> sin agrupación con TVAE para <i>passenger</i>	73
Figura 4-29: Gráfico t-SNE de los datos <i>basic</i> sin agrupación con <i>Gaussian Copula</i> para <i>cargo</i>	74
Figura 4-30: Gráfico t-SNE de los datos <i>basic</i> sin agrupación con <i>Gaussian Copula</i> para <i>passenger</i>	74
Figura 4-31: Gráfico t-SNE de los datos <i>basic</i> sin agrupación con CTGAN para <i>cargo</i>	74
Figura 4-32: Gráfico t-SNE de los datos <i>basic</i> sin agrupación con CTGAN para <i>passenger</i>	74
Figura 4-33: Resultados <i>KS-test</i> para muestra de 100 datos con TVAE por <i>shiptype</i>	75
Figura 4-34: Comparación % de valores únicos en los datos sintéticos por <i>shiptype</i> con los datos reales.....	76
Figura 4-35: Función densidad del parámetro <i>to_bow</i> para <i>cargo</i> por <i>shiptype</i> con TVAE.....	76
Figura 4-36: Función densidad del parámetro <i>to_bow</i> para <i>passenger</i> por <i>shiptype</i> con TVAE.....	76
Figura 4-37: Función densidad del parámetro <i>to_bow</i> para <i>cargo</i> por <i>shiptype</i> con <i>Gaussian Copula</i>	77
Figura 4-38: Función densidad del parámetro <i>to_bow</i> para <i>passenger</i> por <i>shiptype</i> con <i>Gaussian Copula</i>	77
Figura 4-39: Función densidad del parámetro <i>to_bow</i> para <i>cargo</i> por <i>shiptype</i> con CTGAN	77
Figura 4-40: Función densidad del parámetro <i>to_bow</i> para <i>passenger</i> por <i>shiptype</i> con CTGAN....	77
Figura 4-41: Matriz de correlación para <i>cargo</i> de <i>basic</i> con agrupación para TVAE	79
Figura 4-42: Matriz de correlación para <i>passenger</i> de <i>basic</i> con agrupación para TVAE	79
Figura 4-43: Matriz de correlación para <i>cargo</i> de <i>basic</i> con agrupación para <i>Gaussian Copula</i>	79
Figura 4-44: Matriz de correlación para <i>passenger</i> de <i>basic</i> con agrupación para <i>Gaussian Copula</i>	79
Figura 4-45: Matriz de correlación para <i>cargo</i> de <i>basic</i> con agrupación para CTGAN	80
Figura 4-46: Matriz de correlación para <i>passenger</i> de <i>basic</i> con agrupación para CTGAN	80
Figura 4-47: Gráfico PCA de los datos <i>basic</i> con agrupación con TVAE para <i>cargo</i>	80
Figura 4-48: Gráfico PCA de los datos <i>basic</i> con agrupación con TVAE para <i>passenger</i>	80
Figura 4-49: Gráfico PCA de los datos <i>basic</i> con agrupación con <i>Gaussian Copula</i> para <i>cargo</i>	81
Figura 4-50: Gráfico PCA de los datos <i>basic</i> con agrupación con <i>Gaussian Copula</i> para <i>passenger</i>	81
Figura 4-51: Gráfico PCA de los datos <i>basic</i> con agrupación con CTGAN para <i>cargo</i>	81
Figura 4-52: Gráfico PCA de los datos <i>basic</i> con agrupación con CTGAN para <i>passenger</i>	81
Figura 4-53: Gráfico t-SNE de los datos <i>basic</i> con agrupación con TVAE para <i>cargo</i>	82
Figura 4-54: Gráfico t-SNE de los datos <i>basic</i> con agrupación con TVAE para <i>passenger</i>	82
Figura 4-55: Gráfico t-SNE de los datos <i>basic</i> con agrupación con <i>Gaussian Copula</i> para <i>cargo</i> ...	82
Figura 4-56: Gráfico t-SNE de los datos <i>basic</i> con agrupación con <i>Gaussian Copula</i> para <i>passenger</i>	82

Figura 4-57: Gráfico t-SNE de los datos <i>basic</i> con agrupación con CTGAN para <i>cargo</i>	82
Figura 4-58: Gráfico t-SNE de los datos <i>basic</i> con agrupación con CTGAN para <i>passenger</i>	82
Figura 4-59: Matriz de correlación de los datos reales del archivo <i>final_static_data.csv</i>	84
Figura 4-60: Matriz de correlación de los datos sintéticos completos con TVAE.....	84
Figura 4-61: Matriz de correlación de los datos sintéticos completos con <i>Gaussian Copula</i>	84
Figura 4-62: Matriz de correlación de los datos sintéticos completos con CTGAN.....	85
Figura 4-63: Comparación <i>len</i> vs (<i>to_bow</i> + <i>to_stern</i>) sintéticos en TVAE	85
Figura 4-64: Estadísticas del error en la relación <i>len</i>	86
Figura 4-65: Distribución del error en <i>len</i>	86
Figura 4-66: Funciones densidad del conjunto de datos con atributos calculados con TVAE	87
Figura 4-67: Funciones densidad del conjunto de datos con atributos calculados con <i>Gaussian Copula</i>	87
Figura 4-68: Funciones densidad del conjunto de datos con atributos calculados con CTGAN	87
Figura 4-69: Matriz de correlación del conjunto de datos con atributos calculados con TVAE	89
Figura 4-70: Matriz de correlación del conjunto de datos con atributos calculados con <i>Gaussian Copula</i>	89
Figura 4-71: Matriz de correlación del conjunto de datos con atributos calculados con CTGAN	89
Figura 4-72: Gráficos PCA y t-SNE del conjunto de datos sintéticos con atributos calculados para TVAE	90
Figura 4-73: Gráficos PCA y t-SNE del conjunto de datos sintéticos con atributos calculados para <i>Gaussian Copula</i>	90
Figura 4-74: Gráficos PCA y t-SNE del conjunto de datos sintéticos con atributos calculados para CTGAN	91
Figura 4-75: Comparación de <i>Accuracy</i> por técnica y <i>dataset</i>	94
Figura 4-76: Comparación de <i>F1-score</i> por técnica y <i>dataset</i>	94
Figura 4-77: Comparación de <i>Precision</i> por técnica y <i>dataset</i>	95
Figura 4-78: Comparación de <i>Recall</i> por técnica y <i>dataset</i>	95
Figura 4-79: Comparación de <i>AUC-ROC</i> por técnica y <i>dataset</i>	95
Figura 4-80: Matrices de confusión del entrenamiento con dos conjuntos de datos sintéticos diferentes	96
Figura 4-81: Importancia de las variables para dos conjuntos de datos del archivo <i>final_static_data_basic.csv</i>	97
Figura 4-82: Importancia de las variables para dos conjuntos de datos del archivo <i>final_static_data.csv</i>	97
Figura 4-83: Tiempos de ejecución por técnica y <i>shiptype</i> en la generación por tipo de buque.....	98
Figura 4-84: Gráfico de barras de tiempos de ejecución para los datos sintéticos <i>basic</i>	98
Figura 4-85: Gráfico de barras de tiempos de ejecución para los datos sintéticos completo.....	99
Figura 4-86: Diagrama de sectores del balance de clases tras la aplicación de SMOTE.....	100

Figura 4-87: Comparativa de media y desviación entre datos reales y datos sintéticos generados con SMOTE	100
Figura 4-88: Gráficos de densidad de <i>draught</i> y <i>to_bow</i> para <i>passenger</i> con <i>Data Augmentation</i> .100	
Figura 4-89: Gráficos de densidad de <i>to_stern</i> y <i>to_port</i> para <i>passenger</i> con <i>Data Augmentation</i> 101	
Figura 4-90: Gráficos de densidad de <i>to_starboard</i> para <i>passenger</i> con <i>Data Augmentation</i>	101
Figura 4-91: Comparación curva ROC clase <i>passenger</i> (real vs SMOTE)	102

ÍNDICE DE TABLAS

Tabla 2-1: Definiciones de Russell – Norvig de Inteligencia Artificial (tomada de [8]).....	18
Tabla 2-2: Arquitecturas de GANs	36
Tabla 3-1: Atributos estáticos del archivo <i>final_static_data.csv</i>	48
Tabla 4-1: Resumen de las métricas empleadas para la evaluación de calidad de los datos sintéticos (fuente propia).....	62
Tabla 4-2: Métricas de evaluación de utilidad (fuente propia)	64
Tabla 4-3: Resultados <i>chi-cuadrado</i> para los datos sintéticos sin agrupación por <i>shiptype</i>	64
Tabla 4-4: Resultados de las estadísticas descriptivas de <i>to_bow</i> para <i>cargo</i> sin distinción.....	69
Tabla 4-5: Resultados de las estadísticas descriptivas de <i>to_bow</i> para <i>passenger</i> sin distinción.....	69
Tabla 4-6: Resultados <i>chi-cuadrado</i> para los datos sintéticos por <i>shiptype</i>	75
Tabla 4-7: Resultados de las estadísticas descriptivas de <i>to_bow</i> para <i>cargo</i> con agrupación.....	78
Tabla 4-8: Resultados de las estadísticas descriptivas de <i>to_bow</i> para <i>passenger</i> con agrupación.....	78
Tabla 4-9: Resultados de las estadísticas descriptivas de <i>to_bow</i> para <i>cargo</i> con agrupación completo	83
Tabla 4-10: Resultados de las estadísticas descriptivas de <i>to_bow</i> para <i>passenger</i> con agrupación completo.....	83
Tabla 4-11: Resultados de las estadísticas descriptivas de <i>len</i> del conjunto de datos sintéticos con atributos.....	88
Tabla 4-12: Resultados de las estadísticas descriptivas de <i>ldivw</i> del conjunto de datos sintéticos con atributos.....	88
Tabla 4-13: Resumen de los resultados obtenidos del análisis de calidad de cada conjunto de datos ...	91
Tabla 4-14: Resultados de las métricas de utilidad.....	93
Tabla 4-15: Tiempos entrenamiento por técnica en la generación sin distinción	98
Tabla 4-16: Resultados métricas rendimiento <i>Random Forest</i> para <i>Data Augmentation</i>	101

1 INTRODUCCIÓN Y OBJETIVOS

1.1 Contexto y motivación

En la actualidad, la inteligencia artificial (IA) ha revolucionado numerosos sectores, permitiendo el desarrollo de sistemas avanzados capaces de analizar grandes volúmenes de datos, identificar patrones y generar predicciones precisas. Es una de las tecnologías más influyentes en la automatización y optimización de procesos que ha llevado a su adopción en sectores como la industria, la seguridad y el transporte marítimo, tanto en el ámbito civil como en la defensa.

Sin embargo, uno de los principales desafíos en la implementación de esta tecnología es la necesidad de contar con conjuntos de datos de alta calidad, lo que en muchas ocasiones representa una limitación debido a la escasez, restricciones de acceso o problemas de privacidad. El Sistema de Identificación Automática (AIS) es uno de los elementos fundamentales en la monitorización y seguridad marítima, pues es una fuente crucial de información. No obstante, estos datos presentan múltiples deficiencias, como datos incompletos, errores en la introducción manual y la ausencia de registros en ciertas áreas o tipos de embarcaciones. Estas limitaciones dificultan el entrenamiento efectivo de modelos de aprendizaje automático que puedan mejorar la vigilancia marítima, optimizar rutas de navegación y prevenir incidentes.

En este contexto, y visto que en la era de la información y la tecnología, los datos se han convertido en un recurso de gran valor, la generación de datos sintéticos se presenta como una solución innovadora para mitigar la escasez e incrementar la calidad de los datos AIS. La generación de datos sintéticos permite crear conjuntos de datos artificiales que mantienen las características estadísticas de los datos reales, preservando así la calidad y diversidad de la información sin comprometer la privacidad o depender de bases de datos limitadas.

La motivación de este trabajo proviene del Programa COINCIDENTE (Programa de Cooperación en Investigación Científica y Desarrollo en Tecnologías Estratégicas) incentivado por el Ministerio de Defensa de España cuyo objetivo es aprovechar tecnologías emergentes del ámbito civil que puedan satisfacer necesidades estratégicas de las Fuerzas Armadas, promoviendo la colaboración entre la industria, la comunidad científica y el sector tecnológico. Entre las convocatorias recientes se destacan la de 2023 [1] en la que se contempló la generación de datos sintéticos para el entrenamiento de algoritmos de IA en aplicaciones de defensa, y la convocatoria de 2024 [2] en la que se contempló la aplicación de la inteligencia artificial generativa a misiones militares. En este contexto, el presente trabajo muestra una línea de investigación alineada con los objetivos del Programa COINCIDENTE, ya que puede contribuir a la mejora del análisis de datos en el ámbito naval, optimizando la identificación y clasificación de embarcaciones mediante técnicas de inteligencia artificial.

Además, este trabajo se enmarca dentro de una línea de investigación previa realizada por el Centro Universitario de la Defensa en la Escuela Naval Militar (CUD-ENM), sobre la predicción del tipo de buque utilizando datos AIS y técnicas de inteligencia artificial. En particular, los Trabajos Fin de Grado (TFG) realizados por Gonzalo Rodríguez Casajús [3] y Jorge Roldán Calvo [4], exploraron el uso de modelos de aprendizaje automático para clasificar embarcaciones a partir de los datos AIS disponibles y de la representación de trayectorias, enfrentándose a limitaciones derivadas de la calidad y disponibilidad de los datos. En este contexto, el presente TFG complementa dichas investigaciones proporcionando una solución basada en la generación de datos sintéticos, lo que permite mejorar la cantidad y diversidad de datos disponibles para el entrenamiento de modelos de clasificación.

1.2 Objetivos

El principal objetivo de este Trabajo Fin de Grado es la aplicación y estudio de diferentes técnicas de generación de datos sintéticos para mejorar la disponibilidad y calidad de los datos AIS, utilizando modelos de aprendizaje automático. Se analizarán las técnicas de generación probabilísticas y generativas, evaluando su capacidad para crear datos sintéticos coherentes que guarden las relaciones existentes en los datos reales y su aplicabilidad en modelos de clasificación de buques.

De forma más específica, este TFG:

- ♦ Plantea el análisis de la literatura existente sobre la inteligencia artificial generativa, y enfoques de generación de datos sintéticos.
- ♦ Se analizarán los datos AIS disponibles en cuanto a la estructura, calidad y limitaciones de los datos AIS reales.
- ♦ Generación de datos sintéticos empleando las técnicas de generación de datos más apropiadas tras la revisión del estado del arte y teniendo en cuenta el tipo de dato a generar.
- ♦ Se pretende realizar una evaluación de calidad y coherencia de los datos generados, comparando los datos sintéticos con los datos reales mediante métricas estadísticas, análisis de correlaciones y visualizaciones.
- ♦ También se pretende llevar a cabo una validación de utilidad en modelos de aprendizaje automático consistente en entrenar modelos de clasificación con los datos sintéticos generados y evaluar su desempeño en comparación con datos reales.

De este modo, este trabajo no solo busca evaluar la viabilidad de diferentes técnicas de generación de datos sintéticos, sino que también pretende servir como herramienta de apoyo para investigaciones centradas en el análisis de datos AIS y la predicción de embarcaciones mediante inteligencia artificial, facilitando la robustez y precisión de los modelos predictivos.

1.3 Estructura de la memoria

En este apartado, tras establecer el contexto y los objetivos, se procede a explicar la estructura de la memoria. El documento se divide en cinco capítulos, seguidos por la bibliografía y los anexos, organizados de la siguiente manera:

- Capítulo 1: Introducción y Objetivos. Se introduce la motivación del estudio, los objetivos generales y específicos, y la estructura del documento.
- Capítulo 2: Estado del Arte. Se presenta una revisión del marco teórico y de los conceptos clave en los que se fundamenta este trabajo, incluyendo: inteligencia artificial, aprendizaje automático, y generación de datos sintéticos principalmente los enfoques probabilísticos y generativos aplicados a datos tabulares. También se justifican las técnicas de generación de datos sintéticos seleccionadas para el desarrollo del trabajo y se realiza una revisión bibliográfica de estudios previos relacionados con la predicción del tipo de buque a partir de datos AIS y la generación de datos sintéticos.
- Capítulo 3: Desarrollo del TFG. Se describen las herramientas empleadas; se detallan los *datasets* AIS empleados para explicar su estructura; y se describen los flujos de trabajo para la

implementación del proceso de generación, detallando los parámetros utilizados, la segmentación de datos por tipo de embarcación (*shiptype*) y la generación de atributos calculados. También se explica cómo se almacenan y estructuran los datos generados para su posterior análisis y validación.

- Capítulo 4: Resultados y Validación. Se describen las métricas de calidad y utilidad empleadas para el análisis de los datos sintéticos y se presentan los resultados obtenidos de las diferentes técnicas de generación de datos sintéticos empleadas. Se comparan los datos sintéticos con los datos reales, la preservación de relaciones matemáticas entre atributos y su coherencia, y se evalúa su impacto en modelos de aprendizaje automático. También se examina el impacto del *Data Augmentation* en la mejora de la representación de clases minoritarias dentro del conjunto de datos, y los tiempos de computación de cada técnica.
- Capítulo 5: Conclusiones y Líneas Futuras. Se exponen las principales conclusiones del estudio y se proponen posibles líneas de trabajo futuras para mejorar y ampliar la investigación.

Finalmente, se incluye la bibliografía con las fuentes utilizadas a lo largo del trabajo, seguida de una serie de anexos que apoyan el desarrollo del estudio y permiten una mejor comprensión de los métodos empleados.

2 ESTADO DEL ARTE

En este capítulo se presentan los fundamentos teóricos relevantes para la generación de datos sintéticos. Se inicia con una exploración del concepto de Inteligencia Artificial (IA), abordando sus definiciones, con especial énfasis en la Inteligencia Artificial Generativa (IAG), clave en la creación de datos sintéticos. Posteriormente, se analiza el aprendizaje automático (*Machine Learning*, ML) y sus distintos enfoques, con un énfasis en las redes neuronales dado que es la tecnología subyacente en los algoritmos de generación de datos sintéticos.

A continuación, se introduce el Conocimiento del Entorno Marítimo, explicando la relevancia del Sistema de Identificación Automática (AIS) y la problemática derivada de la escasez y calidad de estos datos en el ámbito naval. Se destaca cómo la generación de datos sintéticos puede ofrecer soluciones para mejorar la disponibilidad y fiabilidad de la información AIS.

El capítulo también profundiza en la naturaleza de los datos sintéticos, detallando su definición, aplicaciones, ventajas y los desafíos asociados a su uso. Se clasifican los enfoques de generación de datos sintéticos según su método de creación, según la estructura de los datos y según su propósito. Finalmente, se presentan las técnicas específicas para la generación de datos sintéticos tabulares, y se justifica la selección de las metodologías utilizadas en este trabajo. También se revisan estudios previos que han aplicado técnicas similares en diferentes contextos.

2.1 Inteligencia Artificial

2.1.1 Definición

En los últimos años, la Inteligencia Artificial ha evolucionado de ser un concepto lejano asociado a la ciencia ficción para consolidarse como una parte cada vez más integral de nuestra realidad cotidiana y presente en diversos ámbitos de nuestro día a día. Gracias a los avances tecnológicos y su acelerado desarrollo, el concepto de IA ha adquirido un sentido real. A pesar de eso, encontrar una definición exacta que abarque los diferentes enfoques de esta disciplina sigue siendo tarea difícil, dada la indeterminación del propio concepto de “inteligencia” y su constante y acelerado avance tecnológico que no responde a algo estático.

En términos generales, la Inteligencia Artificial es un campo de la informática cuyo objetivo es diseñar y crear sistemas capaces de realizar tareas que requieren procesos propios de inteligencia humana como la capacidad de aprendizaje, resolución de problemas y de razonamiento. Es decir, emplea algoritmos y modelos matemáticos para imitar el pensamiento humano a través del manejo de datos, la identificación de patrones y el empleo de predicciones.

La primera definición formal de IA fue propuesta en 1956 durante la Conferencia de Dartmouth, punto de partida oficial de este campo, por John McCarthy, miembro del departamento de Ciencias Informáticas de la Universidad de Stanford, a la que se refirió como “*ciencia e ingenio de hacer máquinas inteligentes*” [5]. Según la RAE, la Inteligencia Artificial es la “*disciplina científica que se ocupa de crear programas informáticos que ejecutan operaciones comparables a las que realiza la mente humana, como el aprendizaje o el razonamiento lógico*” [6]. Según el estándar ISO/IEC 22989:2022, norma que establece un marco conceptual para la Inteligencia Artificial, la define como “*un campo técnico y científico dedicado al sistema de ingeniería que genera resultados como contenido, previsiones, recomendaciones o decisiones para un conjunto determinado de objetivos definidos por el ser humano*” [7].

Prácticamente, cada autor que escribe sobre el tema propone su propia definición de Inteligencia Artificial. Tal vez uno de los criterios más ampliamente aceptado para definir este concepto se basa en los cuatro enfoques planteados por los científicos informáticos Stuart Russell y Peter Norvig en su obra *Artificial Intelligence: A Modern Approach* [8]. En este libro, los autores analizaron ocho definiciones propuestas de Inteligencia Artificial, resumidas en la Tabla 2-1, y clasificaron la Inteligencia Artificial en cuatro categorías principales: sistemas que piensan como humanos, sistemas que piensan de manera racional, sistemas que actúan como humanos y sistemas que actúan de forma racional.

Observando la Tabla 2-1, las definiciones en la parte superior se centran en los procesos de pensamiento y el razonamiento, mientras que las de la parte inferior abordan el comportamiento. A su vez, las definiciones a la izquierda evalúan la fidelidad al desempeño humano, mientras que las de la derecha miden en relación con una medida de desempeño ideal, conocida como racionalidad.

Sistemas que piensan como humanos	Sistemas que piensan racionalmente
«El nuevo y excitante esfuerzo de hacer que los computadores piensen... máquinas con mentes, en el más amplio sentido literal». (Haugeland, 1985) «[La automatización de] actividades que vinculamos con procesos de pensamiento humano, actividades como la toma de decisiones, resolución de problemas, aprendizaje...» (Bellman, 1978)	«El estudio de las facultades mentales mediante el uso de modelos computacionales». (Charniak y McDermott, 1985) «El estudio de los cálculos que hacen posible percibir, razonar y actuar». (Winston, 1992)
Sistemas que actúan como humanos	Sistemas que actúan racionalmente
«El arte de desarrollar máquinas con capacidad para realizar funciones que cuando son realizadas por personas requieren de inteligencia». (Kurzweil, 1990) «El estudio de cómo lograr que los computadores realicen tareas que, por el momento, los humanos hacen mejor». (Rich y Knight, 1991)	«La Inteligencia Computacional es el estudio del diseño de agentes inteligentes». (Poole et al., 1998) «IA... está relacionada con conductas inteligentes en artefactos». (Nilsson, 1998)

Tabla 2-1: Definiciones de Russell – Norvig de Inteligencia Artificial (tomada de [8])

El campo de la Inteligencia Artificial ha experimentado un crecimiento exponencial en los últimos años, con un uso cada vez más extendido de términos como aprendizaje automático o *Machine Learning*

(ML), y aprendizaje profundo o *Deep Learning* (DL). Estos conceptos están interrelacionados y pueden visualizarse como círculos concéntricos, como se muestra en la Figura 2-1, donde la IA representa el campo general, dentro del cual se encuentra el ML, y finalmente, el DL como una subcategoría avanzada del aprendizaje automático.

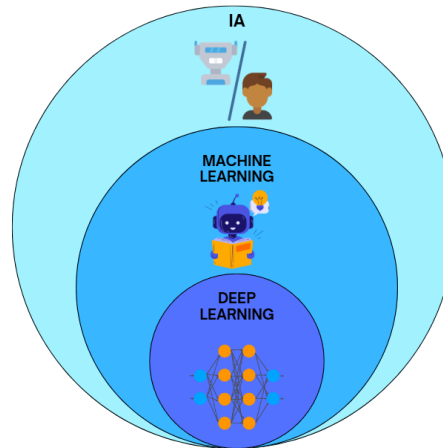


Figura 2-1: Inteligencia Artificial, Machine Learning y Deep Learning (Fuente propia)

El *Machine Learning* surgió como un enfoque dentro de la IA que permite a los sistemas aprender a partir de datos, en lugar de ser programados explícitamente para cada tarea. Posteriormente, el *Deep Learning* revolucionó este paradigma mediante el uso de redes neuronales profundas, que han demostrado un rendimiento sobresaliente en el procesamiento de datos complejos como imágenes, texto y voz. Hoy en día, ambos conceptos son los impulsores de la Inteligencia Artificial Generativa.

2.1.2 Inteligencia Artificial Generativa

La Inteligencia Artificial Generativa (IAG) es el nombre asignado al tipo de inteligencia artificial capaz de crear contenido nuevo de manera autónoma, emulando la creatividad humana en diversas áreas como la generación de texto, imágenes, audio y código, mediante el uso de algoritmos y modelos de aprendizaje automático (*Machine Learning*). Su auge reciente ha sido impulsado por avances en redes neuronales profundas, que han permitido el desarrollo de modelos altamente sofisticados capaces de generar información con coherencia y precisión. Entre las arquitecturas clave destacan las Redes Generativas Adversarias (GANs), los *Autoencoders Variacionales* (VAEs) y los *Transformers*, que se expondrán más adelante.

Aunque los fundamentos de la IAG no son nuevos, su impacto se ha visto aumentado por la accesibilidad de modelos pre-entrenados en grandes volúmenes de datos (*Big Data*) y la creciente capacidad computacional. La proliferación de herramientas como ChatGPT, DALL-E y GitHub Copilot ha permitido su integración en múltiples disciplinas, desde la investigación científica hasta la producción creativa [9].

2.2 Aprendizaje Automático (*Machine Learning*)

Como ya se ha mencionado antes, el aprendizaje automático ha sido el elemento clave que ha impulsado el desarrollo de la Inteligencia Artificial Generativa (IAG) y se basa en el uso de algoritmos y arquitecturas que aprenden patrones y características de datos, que les permiten realizar tareas de clasificación, predicción y generación, para las cuales no estaban inicialmente programados. La característica distintiva de estos algoritmos es su capacidad para adaptar su comportamiento en función de la información disponible y los resultados obtenidos en interacciones previas.

Los algoritmos de *Machine Learning* se dividen en cuatro categorías: aprendizaje supervisado, en el cual se entrena con datos etiquetados que permiten establecer relaciones entre entrada y salida;

aprendizaje no supervisado, que identifica patrones en datos sin etiquetas; aprendizaje semisupervisado que utiliza datos etiquetados y no etiquetados para mejorar las predicciones; y aprendizaje por refuerzo donde un agente interactúa con un entorno y aprende mediante un sistema de recompensas y penalizaciones [10]. A continuación, se explican con más detalle las dos primeras categorías.

2.2.1 Aprendizaje supervisado

El aprendizaje automático supervisado se caracteriza por el uso de conjuntos de datos etiquetados para su entrenamiento, lo que le permite clasificar datos similares o predecir resultados con precisión. El funcionamiento del aprendizaje supervisado se basa en un conjunto de entrenamiento que enseña a los modelos a producir la salida correcta. Este conjunto debe incluir tanto las entradas como las salidas correctas, lo que permite al modelo aprender con el tiempo. La precisión del algoritmo se evalúa mediante una función de pérdida, y se ajusta continuamente para minimizar el error. A medida que se introducen los datos en el modelo, este se ajusta hasta conseguir una adaptación efectiva.

Existen diferentes algoritmos de aprendizaje supervisado, que generalmente se implementan en lenguajes de programación como R o Python. Algunos de los métodos más destacados incluyen redes neuronales (que simulan la interconectividad del cerebro humano a través de capas de nodos), los algoritmos de Naïve Bayes (que se basan en el Teorema de Bayes y suponen independencia condicional entre clases), los algoritmos de regresión lineal y logística (empleados para identificar relaciones entre variables y realizar predicciones), los árboles de decisión (para la predicción y clasificación de datos que emplea secuencias de decisiones representadas en diagramas de árbol), *Random Forest* (que utiliza múltiples árboles de decisión para reducir la varianza y mejorar la precisión de las predicciones), y las máquinas de vectores de soporte (SVM) (que se utilizan tanto para clasificación como para regresión, creando un hiperplano que maximiza la distancia entre las clases [11]).

2.2.2 Aprendizaje no supervisado

Los algoritmos de aprendizaje automático no supervisado analizan y agrupan conjuntos de datos que no están etiquetados, formando subconjuntos conocidos como clústeres. Estos algoritmos son capaces de identificar patrones ocultos o agrupaciones en los datos sin necesidad de intervención humana.

Existen diversos algoritmos para abordar la agrupación de datos no etiquetados en clústeres, y se pueden clasificar en exclusivos o basado en centroides, superpuestos, jerárquicos o probabilísticos entre otros. La agrupación exclusiva, como el algoritmo de k-medias, asigna cada punto de datos a un solo clúster. En contraste, la agrupación superpuesta permite que un punto pertenezca a varios clústeres con diferentes grados de pertenencia. La agrupación jerárquica, o análisis de clústeres jerárquicos, se divide en aglomerativa y divisiva, empleando métodos como el enlace de Ward, enlace promedio, enlace completo y enlace único para medir la similitud entre clústeres [12]. Por último, la agrupación probabilística se fundamenta en la probabilidad de que un punto de datos pertenezca a una distribución específica, siendo el modelo de mezcla gaussiana un ejemplo común.

Los algoritmos de aprendizaje automático no supervisado utilizan mecanismos como reglas de asociación o reducciones de dimensionalidad. Las reglas de asociación identifican relaciones entre variables en un conjunto de datos y son generadas por algoritmos como "Apriori". Para reducir la cantidad de características en un modelo y optimizar el rendimiento del algoritmo, se recurre a la reducción de dimensionalidad con métodos tales como Análisis de Componentes Principales (PCA) o Descomposición en Valores Singulares (SVD).

2.2.3 Aprendizaje Profundo (Deep Learning)

Dado que el aprendizaje profundo y el aprendizaje automático a menudo se utilizan de manera intercambiable, es importante destacar sus diferencias. El aprendizaje automático, el aprendizaje profundo y las redes neuronales son subcampos de la inteligencia artificial. Sin embargo, ML es un enfoque más general que engloba diversas técnicas para entrenar modelos, mientras que DL es una especialización dentro de ML basada en redes neuronales profundas.

La distinción entre el aprendizaje profundo y el aprendizaje automático radica en la forma en que cada algoritmo aprende. Mientras que el aprendizaje automático tradicional depende en gran medida de datos estructurados y etiquetados, el aprendizaje profundo es capaz de procesar grandes volúmenes de datos no estructurados, como imágenes o texto, sin necesidad de intervención humana para extraer características. Esto lo convierte en un enfoque más autónomo y escalable. Además, a diferencia de los modelos supervisados convencionales, el aprendizaje profundo puede aplicar técnicas de aprendizaje no supervisado, permitiéndole identificar patrones y relaciones dentro de los datos sin necesidad de etiquetas previas. Su capacidad de ajustar y refinar sus predicciones de manera iterativa lo hace altamente eficiente en tareas complejas de reconocimiento, clasificación y generación de datos [13].

2.3 Algoritmos de aprendizaje automático

Como ya se ha mencionado, existen diversos algoritmos de aprendizaje automático que varían en su capacidad de generalización, interpretabilidad y eficiencia computacional. En este apartado se presentan los dos más relevantes para este trabajo: los árboles de decisión que permiten tareas de interpretabilidad y clasificación; y las redes neuronales que ofrecen una capacidad excepcional de modelado de patrones en datos de gran escala.

2.3.1 Árboles de decisión

Los árboles de decisión organizan el flujo de los datos en una estructura jerárquica en forma de árbol, en la que cada nodo representa una pregunta o prueba sobre una variable, mientras que las ramas representan posibles respuestas. El árbol inicia con un nodo raíz que representa el conjunto completo de datos. A partir de este nodo se divide en dos o más subnodos según una condición específica. Este proceso se repite en cada subnodo hasta alcanzar los nodos hoja, que representan las predicciones finales [14].

A pesar de que son fáciles de comprender, pues simulan el proceso de toma de decisiones humano, el principal desafío es seleccionar la característica adecuada para cada una de las divisiones. Utilizan criterios como Gini o Entropía para seleccionar la mejor división en cada nodo. Las ventajas de los árboles de decisión se manifiestan en que son altamente interpretables y permiten visualizar la importancia de las variables en la clasificación final, lo que facilita la comprensión del modelo y la toma de decisiones. Además, son robustos frente a la sobrecarga y el ruido en los datos, lo que los hace adecuados para conjuntos de datos complejos y variables [15].

Los árboles de decisión pueden extenderse a bosques aleatorios como *Random Forest*, que se basa en la combinación de múltiples árboles para mejorar la precisión y reducir el sobreajuste. Cada árbol se entrena con una muestra aleatoria del conjunto de datos original, reduciendo la correlación entre ellos y mejorando la precisión del modelo. Durante la predicción, cada árbol genera un resultado y la decisión final se obtiene mediante votación mayoritaria (para clasificación). *Random Forest* es ampliamente utilizado en aprendizaje automático y está disponible en la biblioteca *scikit-learn* de Python, facilitando su implementación en tareas de minería de datos y predicción [16].

2.3.2 Redes neuronales

Las redes neuronales artificiales (RNA) o redes neuronales simuladas (RNS) son modelos que simulan el funcionamiento de las neuronas en el cerebro humano [17]. Están formadas por capas de neuronas o nodos interconectados que transforman progresivamente los datos de entrada hasta obtener un resultado de salida. Cada neurona recibe entradas de las neuronas de la capa anterior, procesa esa información y transfiere el resultado a las neuronas de la capa siguiente, generando representaciones cada vez más sofisticadas [18]. Este proceso se repite a través de todas las capas de la red como se muestra en Figura 2-2.

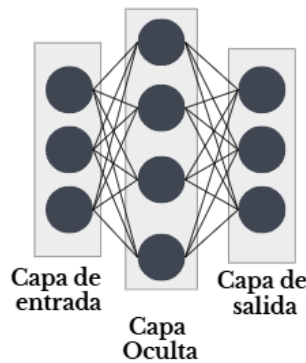


Figura 2-2: Arquitectura de una red neuronal (elaboración propia)

Una red neuronal típica presenta una organización estructural en tres tipos de capas principales. La capa de entrada recibe los datos en forma de vectores numéricos. A continuación, las capas ocultas llevan a cabo transformaciones sobre los datos a través de funciones de activación no lineales. Finalmente, la capa de salida genera la predicción o clasificación deseada. La mayoría de las redes neuronales son de avance, lo que significa que fluyen en una dirección, desde la entrada hasta la salida. Sin embargo, también se puede entrenar un modelo mediante retropropagación, es decir, moviéndose en la dirección opuesta, desde la salida hasta la entrada. Cada conexión entre neuronas está asociada a un peso que se ajusta durante el proceso de entrenamiento mediante la técnica retropropagación del error (*backpropagation*), que permite minimizar la diferencia entre la predicción obtenida y la salida esperada [19].

Existen diversas arquitecturas de redes neuronales diseñadas para abordar tipos específicos de problemas:

- Las Redes Neuronales Profundas (DNN) son una extensión de las redes neuronales tradicionales. La distinción entre el aprendizaje profundo y las redes neuronales se encuentra en el término "profundo", que hace referencia al número de capas en una red neuronal. Se considera que una red neuronal es un algoritmo de aprendizaje profundo cuando cuenta con más de tres capas, incluyendo las capas de entrada y salida, como se muestra en la Figura 2-3. Se utilizan comúnmente en el procesamiento de datos tabulares, texto e imágenes de menor escala.

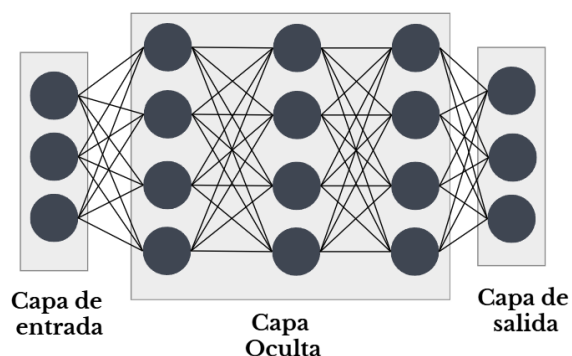


Figura 2-3: Arquitectura de una red neuronal profunda (elaboración propia)

- Las Redes Neuronales Convolucionales (CNN) están diseñadas para el procesamiento de imágenes y datos espaciales. Se caracterizan por la inclusión de capas de convolución que aplican filtros para detectar patrones, y capas de *pooling* que reducen la dimensionalidad de los datos manteniendo la información importante [20]. Las CNN han demostrado un rendimiento excepcional en tareas de reconocimiento de imágenes, detección de objetos y visión por computadora.

- Las Redes Neuronales Recurrentes (RNN) son modelos diseñados para trabajar con datos secuenciales, donde la información previa influye la interpretación de datos futuros. Cuentan con conexiones recurrentes en sus capas ocultas, lo que permite la conservación de información a lo largo del tiempo. Estas redes son utilizadas en aplicaciones como el procesamiento de lenguaje natural (NLP), análisis de series temporales y traducción automática. Un tipo avanzado de RNN es la *Long Short-Term Memory* (LSTM), que mitiga el problema de la desaparición del gradiente y facilita el aprendizaje de dependencias a largo plazo en las secuencias de datos [21].

2.4 Conocimiento del Entorno Marítimo

2.4.1 COVAM

El Centro de Operaciones y Vigilancia de Acción Marítima (COVAM) es una unidad operativa de la Armada Española, integrada dentro del Estado Mayor de la Fuerza de Acción Marítima. Su objetivo principal es analizar y proporcionar información precisa relativa a los espacios marítimos de interés en tiempo real, lo que se denomina como Conocimiento del Entorno Marítimo (CEM) o como *Recognized Maritime Picture* (RMP), según su designación en inglés [22]. Es clave para la seguridad y operatividad en el ámbito marítimo, especialmente para la defensa de los espacios marítimos de responsabilidad, soberanía e interés nacional; para el comercio internacional y para la gestión del tráfico marítimo. Se constituye como una herramienta fundamental que facilita la toma de decisiones efectiva tanto en el ámbito civil como militar.

Ubicado en Cartagena, en el edificio de Capitanía General, desempeña un papel crucial en la seguridad marítima nacional e internacional. A diferencia de los Centros de Operaciones Navales, que se centran en la vigilancia y respuesta ante amenazas militares, el COVAM actúa como un centro de fusión de información, integrando datos de múltiples fuentes para proporcionar una visión completa del entorno marítimo.

El COVAM tiene aplicaciones en diversas áreas del ámbito marítimo, incluyendo:

- Seguridad y defensa marítima para la identificación y seguimiento de embarcaciones sospechosas, prevención de actividades ilícitas y patrullaje en zonas de conflicto.
- Gestión del tráfico marítimo para el control de rutas comerciales, optimización del tránsito en puertos y prevención de colisiones.
- Protección medioambiental para la monitorización de vertidos de petróleo, control de zonas de pesca y preservación de ecosistemas marinos.
- Respuesta ante emergencias para la coordinación en operaciones de rescate y respuesta ante desastres naturales.

Además de su labor en la vigilancia y seguridad de los espacios marítimos, el COVAM tiene una función destacada en la cooperación con la comunidad marítima. Su misión no se limita únicamente a la supervisión de operaciones navales, sino que también sirve como un punto de contacto y colaboración con otras entidades nacionales e internacionales, facilitando la gestión del tráfico marítimo y la seguridad de las rutas comerciales. El Concepto de Seguridad Marítima en la Armada establece que la seguridad en el entorno marítimo no es responsabilidad exclusiva de una única entidad, sino que depende de la cooperación coordinada entre múltiples agencias que operan en el ámbito marítimo. Cada una de estas organizaciones contribuye dentro de su respectiva área de competencia, formando un sistema integral de vigilancia y protección [23].

El COVAM no solo opera dentro del territorio nacional, sino que también extiende su alcance a aguas de interés estratégico como el Golfo de Guinea, el Golfo Pérsico y el Océano Índico. Para ello, trabaja en conjunto con unidades de la Fuerza de Acción Marítima (FAM) y colabora estrechamente con agencias internacionales para la gestión de la seguridad en rutas comerciales clave. El Almirante de Acción Marítima (ALMART), en su rol de Comandante del Mando Operativo Marítimo (CMOM), es

responsable de coordinar las operaciones de vigilancia y seguridad en los espacios marítimos bajo soberanía española y en aquellas regiones de interés estratégico.

Para garantizar un manejo seguro y eficiente de la información, el COVAM opera bajo una estructura de tres niveles de seguridad. La primera es abierta para la gestión de información de conocimiento general sobre el entorno marítimo, utilizada para la cooperación con entidades civiles y organismos internacionales. La segunda capa tiene información sensible que contiene datos estratégicos que requieren restricciones de acceso, destinados a uso gubernamental y militar. La tercera capa clasificada, en ella se maneja información de alto nivel de seguridad, accesible únicamente para personal autorizado y utilizada en operaciones críticas. Esta división permite al COVAM analizar y compartir información de manera segura, garantizando la protección de datos sensibles y facilitando la interoperabilidad con otras instituciones.

El COVAM obtiene información de diversas fuentes, lo que le permite disponer de una imagen detallada del tráfico marítimo y de las condiciones de seguridad en las zonas de interés. Entre los principales sistemas de recopilación de datos se encuentran:

- El Sistema de Identificación Automática (AIS) que recibe señales procedentes de estaciones terrestres, buques, aeronaves y satélites, proporcionando datos sobre identidad, ruta, velocidad y destino de las embarcaciones.
- Sensores y sistemas de vigilancia a partir de los cuales el COVAM integra información de unidades militares nacionales e internacionales, datos de buques de pesca equipados con sistemas de balizamiento y otras fuentes adicionales.
- Bases de datos especializadas que incluyen historiales de buques, información sobre empresas del sector marítimo y registros de actividades pesqueras y comerciales.

Toda esta información es procesada y analizada mediante herramientas avanzadas de fusión de datos. El COVAM cuenta con el Sistema Integrado de Vigilancia y Conocimiento del Entorno Marítimo (SIVICEMAR), una plataforma que permite el análisis automático del comportamiento de los buques y la detección de actividades inusuales. Gracias a este sistema, se pueden identificar patrones de navegación sospechosos y activar los protocolos de respuesta adecuados.

El COVAM dispone de medios de Mando y Control (C2) tanto a nivel nacional como en el marco de la Unión Europea (UE) y la OTAN, lo que permite una coordinación efectiva en operaciones de seguridad marítima a nivel internacional. Su equipo de profesionales opera las 24 horas del día, los 365 días del año, supervisando continuamente el entorno marítimo y respondiendo de manera inmediata ante cualquier incidencia [22].

2.4.2 Sistema de Identificación Automática (AIS)

El sistema AIS es un sistema de ayuda a la navegación que fue desarrollado con el objetivo de mejorar la seguridad marítima. Permite a través de VHF a los barcos que se encuentran próximos intercambiar automáticamente información digital sobre su identidad, ubicación, rumbo, velocidad, destino, etc., con el fin de evitar colisiones en la mar. El AIS es un componente esencial del Sistema Mundial de Socorro y Seguridad Marítimos (GMDSS) y un requisito obligatorio desde 2002 de acuerdo con el convenio SOLAS (*Safety Of Life At Sea*) (Capítulo V – Regla 19) de la Organización Marítima Internacional (OMI), para todo buque en trayectos internacionales con más de 300 toneladas, o de 500 toneladas para trayectos nacionales, y para todos los buques de pasaje [24].

Un transpondedor AIS establece su posición, velocidad y rumbo en un determinado momento utilizando un receptor GPS integrado. Esta información se mezcla con otros datos de navegación relevantes y se transmiten automáticamente entre embarcaciones equipadas con AIS. Los transpondedores AIS de diferentes embarcaciones y estaciones costeras reciben los datos y los emplean para generar una representación gráfica en un *chartplotter* del tráfico marítimo en una zona. Además, el transpondedor AIS puede integrarse en diversas aplicaciones de navegación [25].

Las principales aplicaciones del AIS son:

- Fuente de información para autoridades marítimas.
- Herramienta de apoyo para los dispositivos de control del tráfico marítimo o *Vessel Traffic Service* (VTS) y para tareas de salvamento y rescate (SAR).
- Complementar el radar.
- Evitar abordajes en la mar facilitando el intercambio de información y ofreciendo cálculos cinemáticos como la distancia mínima de paso (CPA) o el tiempo hasta dicha distancia (TCPA).
- Identificación de balizas y marcas de navegación que ayudan a la navegación, AtoN (*Aids to Navigation*).

Existen dos tipos de transpondedores AIS: clase A y clase B. Los transpondedores AIS clase A son obligatorios para embarcaciones de más de 300 toneladas brutas en viajes internacionales, buques de pasaje y otras embarcaciones que deben cumplir con las regulaciones del convenio SOLAS. Ofrecen una transmisión de datos más completa y una tasa de actualización más frecuente, aunque de mayor coste y peso. En cambio, los transpondedores AIS clase B, más económicos y ligeros, son ideales para embarcaciones menores, no sujetas al convenio SOLAS, como las embarcaciones de recreo. Transmiten menos información y una tasa de actualización menor. También existen receptores AIS que solo reciben, pero no transmiten datos [26].

Los datos transmitidos y recibidos por el sistema AIS se organizan en: datos estáticos, datos dinámicos, datos del viaje y mensajes relativos a la seguridad para alertar de peligros.

- Los datos estáticos reflejan datos fijos del buque que no cambian normalmente con el tiempo, tan solo cuando se modifica alguno de los parámetros del buque. Se ingresan manualmente en el sistema AIS por el operador. Estos datos se transmiten cada 6 minutos:
 - o Número MMSI (*Maritime Mobile Service Identity*): código único de 9 dígitos de la estación transmisora que sirve para identificar la embarcación.
 - o Número IMO: número de 7 dígitos.
 - o Nombre y distintivo de llamada.
 - o Eslora y manga.
 - o Ubicación de la antena GPS.
 - o Tipo de buque.
- Los datos dinámicos son aquellos que cambian constantemente y reflejan la situación en tiempo real de la embarcación. Estos datos se generan a partir de sensores del barco y se transmiten automáticamente por el sistema AIS a intervalos de 2 a 10 segundos en función de la velocidad.
 - o Posición GPS (latitud y longitud),
 - o Hora de la posición del barco en UTC.
 - o Velocidad sobre fondo (*Speed Over Ground*, SOG).
 - o Rumbo sobre fondo (*Course Over Ground*, COG).
 - o Rumbo verdadero (*True Heading*): dirección en la que apunta la proa del buque.
 - o Tasa de cambio de rumbo (*Rate of Turn*, ROT): indica la velocidad a la que la embarcación cae.
 - o Estado de navegación: modo de operación del buque (en marcha, fondeado, maniobra restringida...).
- Los datos de viaje proporcionan información sobre la ruta y el destino del buque. Se introducen manualmente y se transmiten cada 6 minutos, pero se pueden actualizar durante el viaje si hay algún cambio.
 - o Calado.
 - o Cargamento.
 - o Plan de ruta donde se establece el puerto de salida y el puerto de destino, así como fecha y hora de la salida y de la recalada en el próximo puerto.
 - o Hora estimada de llegada (*Estimated Time of Arrival*, ETA).

- Mensajes de seguridad breves en formato libre para comunicaciones de seguridad entre buques cercanos.

2.4.3 Calidad y fiabilidad de los datos AIS

Como se ha mencionado anteriormente, el Sistema de Identificación Automática es una herramienta esencial en la navegación marítima, ya que proporciona información detallada sobre las embarcaciones, incluyendo datos estáticos, dinámicos y relacionados con el viaje. Sin embargo, la calidad y fiabilidad de estos datos pueden variar significativamente según el tipo de información y la clase de transpondedor AIS instalado en la embarcación. A continuación, se procede a estudiar la fiabilidad de los datos AIS en función de diferentes artículos publicados en esta materia.

Los datos dinámicos, que incluyen la posición, rumbo y velocidad de la embarcación, se generan automáticamente a partir de sistemas como el GPS, lo que los hace generalmente más fiables. Según el estudio de plausibilidad de los parámetros AIS relacionado con la navegación basado en series temporales de los autores Frank Heymann, Thoralf Noack y Pawel Banyś, del Centro Aeroespacial alemán eV [27]:

- La posición GPS transmitida por el AIS es altamente precisa, con un margen de error de pocas decenas de metros debido a la precisión del sistema GPS.
- El *Course Over Ground* (COG) tiene una fiabilidad del 95%, con un error máximo de ± 3 grados.
- El *Speed Over Ground* (SOG) muestra también una fiabilidad del 95%, con un margen de error de ± 0.3 nudos.

En contraste, los datos estáticos como el tipo de barco y dimensiones, y los datos relacionados con el viaje como el puerto de destino y el calado dependen de la entrada manual por parte de la tripulación, lo que reduce su fiabilidad. Según el estudio de las fortalezas e inconvenientes de los datos AIS realizado por Ties Emmens, Chintan Amrit, Asad Abdi y Mayukh Ghosh, de las Universidades de Amsterdam y Twente (Países Bajos) [28]:

- El 80% de los datos AIS contienen errores debido a ruido, valores ausentes, duplicados o información inexacta. En concreto, el 86% de los datos estáticos y el 48% de los datos de viaje presenta cuerpos vacíos.
- El 35% de los barcos no actualizan el puerto de destino durante su travesía.
- El 53% de los mensajes AIS carecen de un ETA en formato completo.
- El 30% de los barcos transmiten un valor nulo en el calado.
- La calidad de los datos AIS varía según el tipo de barco; los buques de pasajeros y de carga presentan las tasas más altas de inexactitud en la información introducida manualmente.
- Fallos en los dispositivos AIS y diferencias en la calidad de los equipos pueden generar inconsistencias en los datos transmitidos.

Según otro estudio de reconocimiento del tipo de barco basado en la trayectoria de navegación del barco empleando redes neuronales convolucionales realizado por Tianyu Yang, Xin Wang y Zhengjiang Liu de la Facultad de Navegación de la Universidad Marítima de Dalian, China [29] el 6% de los barcos analizados no tenía una etiqueta de tipo de barco aplicable, y otro 3% estaba etiquetado solo como "buque" sin una clasificación específica.

En conclusión, mientras que los datos dinámicos del AIS son generalmente precisos debido a su generación automática, los datos estáticos y relacionados con el viaje presentan problemáticas en términos de calidad y fiabilidad debido a la entrada manual y la variabilidad en la forma de cubrir los datos entre diferentes tipos de embarcaciones y clases de AIS. Estas limitaciones justifican el uso de datos sintéticos, los cuales pueden complementar la información real para mejorar la robustez de los modelos de análisis y predicción en el ámbito marítimo.

2.5 Datos sintéticos

2.5.1 Definición

La revolución digital lleva en desarrollo más de 50 años, pero los avances recientes en la capacidad de procesamiento han impulsado una nueva era de *Big Data*. La disponibilidad masiva de datos provenientes de redes sociales, registros administrativos electrónicos, transacciones comerciales y otras fuentes, tanto privadas como públicas, ha aumentado exponencialmente. Sin embargo, este crecimiento también ha incrementado los riesgos asociados a la divulgación de información sensible y la protección de la privacidad. El concepto de datos sintéticos surge como una solución a estos problemas y tiene sus raíces en la estadística y la simulación computacional, con antecedentes que se remontan a mediados del siglo XX.

Uno de los primeros enfoques en la generación de datos sintéticos fue desarrollado por Donald Rubin en 1993. En el contexto de la seguridad de los datos D. Rubin creó el concepto de los datos totalmente sintéticos e introdujo la idea de imputación múltiple sintética [30], que trata de generar valores posibles para datos no cubiertos en encuestas y estudios estadísticos. Ese mismo año, Little sugirió sintetizar parcialmente los datos en lugar de generarlos completamente sintéticos. En la actualidad, con el auge de la inteligencia artificial, y sobre todo del aprendizaje profundo, se han desarrollado técnicas más sofisticadas, como los Autocodificadores Variacionales (VAE) en 2013 y las Redes Generativas Adversarias (GANs) en 2014, que se detallarán más adelante.

Los datos sintéticos son conjuntos de datos generados artificialmente mediante algoritmos computacionales o simulaciones. En lugar de ser seleccionados a partir de sucesos del mundo real, los datos sintéticos pretenden replicar las características y patrones de los datos reales en entornos digitales. Se utilizan para probar conjuntos de datos operativos, validar modelos matemáticos y entrenar modelos de aprendizaje profundo. Los datos sintéticos ofrecen una calidad de datos mejorada evitando la vulnerabilidad del error humano, brindan una gran escalabilidad para satisfacer las necesidades de entrenamiento de modelos y permiten un control total en su organización, presentación y etiquetado [31].

Las principales razones por las que se requiere el uso de datos sintéticos incluyen preocupaciones de privacidad, pruebas de productos y sistemas más rápidos, y entrenamiento de algoritmos de inteligencia artificial. Actualmente, las leyes de privacidad de datos limitan cómo las empresas deben manejar la información sensible. Para ello, el uso de datos sintéticos puede minimizar riesgos asociados con la divulgación de información personal. En cuanto a la construcción de nuevos productos y servicios digitales, los datos reales pueden no estar disponibles, ser escasos o incompletos. Los datos sintéticos pueden generarse rápidamente, facilitando el desarrollo de modelos de aprendizaje automático confiables. Además, la anotación de datos por parte de personas es un proceso costoso y que requiere de mucho tiempo, que se puede evitar.

Antes de optar por el método más apropiado de generación de datos sintéticos, es necesario definir el tipo de datos sintéticos requerido. Los tres tipos principales de datos sintéticos son [32]:

- Datos estructurados: son aquellos con un formato predefinido en tablas (tabulares), bases de datos, u hojas de cálculo. Ejemplos son los registros de clientes, registros sanitarios, lecturas de sensores o los datos AIS.
- Datos no estructurados: aquellos que carecen de un formato predefinido y varían su presentación como las imágenes, videos, textos o archivos de voz.
- Datos secuenciales: aquellos que presentan una estructura temporal o dependen de un orden como registros de sucesos o trayectorias.

Según la proporción de los datos de salida existen tres categorías principales de datos sintéticos, que se pueden observar en la Figura 2-4:

- Datos completamente sintéticos: aquellos generados sin conexión directa a datos reales, pero que contienen todas las características necesarias y distribuciones estadísticas de los datos reales. Se suelen usar cuando la privacidad y confidencialidad impiden el uso de los datos originales.
- Datos parcialmente sintéticos: aquellos que contienen información del conjunto original de datos reales, excepto información sensible, lo que puede implicar que algunos valores verdaderos permanezcan en el conjunto de datos sintéticos. Estos datos ayudan a completar los conjuntos de datos existentes cuando falta cierta información o la cantidad es insuficiente. Un ejemplo puede ser una base de datos de pacientes donde se sustituyen datos personales, pero se retienen los valores reales de las mediciones clínicas [33].
- Datos híbridos: aquellos derivados de datos reales y sintéticos. A partir de la distribución de los originales se elige un registro de datos sintéticos y se unen en un conjunto de datos híbridos. A diferencia de los parcialmente sintéticos, no contienen registros totalmente reales.

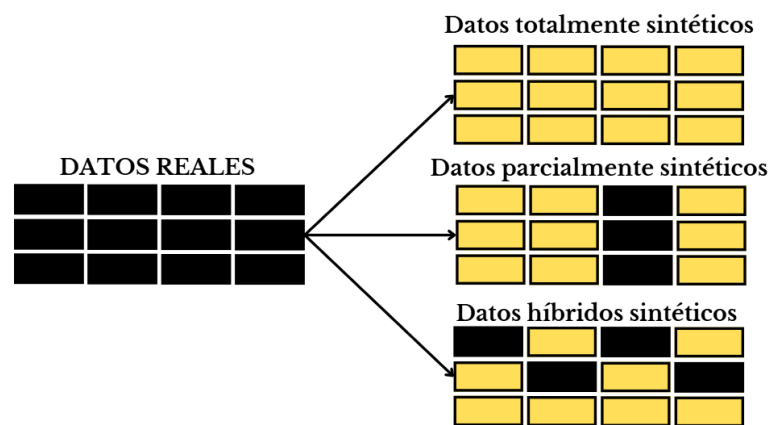


Figura 2-4: Tipos de datos sintéticos según proporción (fuente propia)

2.5.2 Aplicaciones y uso de los datos sintéticos

Los datos sintéticos ofrecen una multitud de ventajas y beneficios, siendo una opción muy atractiva para múltiples aplicaciones. Su versatilidad y utilidad en situaciones relacionadas con escasez de datos reales, protección de datos sensibles y en procesos de entrenamiento y prueba de modelos en escenarios específicos, los convierten en una herramienta imprescindible para la innovación tecnológica. A continuación, se exploran diferentes dominios en los cuales la generación de datos sintéticos aborda sustancialmente desafíos del mundo real [34].

En el campo de la visión por ordenador, que es la rama de la IA que trabaja con redes neuronales para otorgar a la tecnología la capacidad de interpretar el mundo como los humanos, el uso de datos sintéticos para la generación de imágenes de entrenamiento demuestra una precisión comparable a la de los datos reales. Además, los datos sintéticos para modelos de visión por ordenador reducen el coste de los servicios de etiquetado de imágenes. Ejemplos de uso de datos sintéticos en este campo [35] son la creación de imágenes para entrenar modelos de control de calidad de procesos, detección de enfermedades con imágenes sintéticas o entrenamiento de vehículos autónomos, entre otros.

La creación de voces sintéticas para diversas aplicaciones, como la producción de vídeo, los asistentes virtuales y los videojuegos se ha vuelto más sencilla y precisa, y con el avance de la tecnología se puede esperar que las voces sintéticas se vuelvan más frecuentes en nuestra vida diaria, asistiéndonos y enriqueciendo experiencias. Este campo incluye varias disciplinas de acústica, lingüística y procesamiento de señales. Los datos sintéticos también se pueden aplicar para la conversión de texto a voz (*Text-to-Speech*, TTS) logrando una naturalidad casi humana y para entrenar modelos de reconocimiento automático del habla (*Automatic Speech Recognition*, ASR).

En el campo del procesamiento del lenguaje natural (PLN), el creciente interés en los datos sintéticos ha impulsado el desarrollo de modelos generativos de aprendizaje para categorizar, filtrar y buscar información. El significado de las palabras y frases puede cambiar según su contexto. Para abordar esta problemática, los datos sintéticos sirven para desarrollar esquemas de entrenamiento que refuercen la capacidad del sistema de generalizar y procesar el lenguaje humano.

En el sector sanitario, los datos sintéticos son valiosos para garantizar la privacidad de los pacientes al mismo tiempo que se investigan enfermedades y se desarrollan nuevas tecnologías. Se pueden generar bases de datos sintéticas de registros clínicos sin exponer información sensible, lo que permite el desarrollo de sistemas de diagnóstico y modelos predictivos de enfermedades. Los datos sintéticos en el ámbito de la medicina también son útiles para simulaciones epidemiológicas, creación de imágenes para entrenamiento de sistemas de diagnóstico e incluso para el diseño de fármacos con datos sintéticos y modelos generativos que reproduzcan y aprendan las estructuras moleculares de los fármacos.

Los datos sintéticos en el mundo empresarial permiten a las organizaciones acceder a datos más rápidamente evitando los protocolos de privacidad y seguridad existentes en los datos reales y permitiendo el modelado de los datos y la generación de información para la toma de decisiones de las empresas.

En el sector financiero los datos sintéticos ofrecen beneficios significativos, constituyéndose como una herramienta clave para mejorar sus sistemas y garantizar la seguridad. Los datos sintéticos imitan los datos reales sin necesidad de exponer la información sensible del cliente, permitiendo cumplir con las leyes de privacidad y utilizar los conjuntos de datos para análisis y validación de modelos financieros. Por ejemplo, utilizando datos sintéticos de mercado se pueden evaluar escenarios extremos de crisis financieras o fluctuaciones del mercado que no estén presentes en los datos históricos. También permiten una evaluación más precisa de riesgos. Los datos sintéticos pueden replicar patrones de fraude y entrenar a los algoritmos de detección en la identificación de anomalías de manera más efectiva.

Al igual que en el resto de sectores, la industria marítima puede beneficiarse de los datos sintéticos para la simulación de rutas y para el entrenamiento de sistemas predictivos de identificación de patrones y tipos de barcos. En definitiva, para el entrenamiento de cualquier modelo de aprendizaje IA hay que disponer de un conjunto de datos diversos y de alta calidad que pueden ser generados sintéticamente.

2.5.3 Beneficios del uso de datos sintéticos

A lo largo del capítulo, se han mencionado ya algunas de las principales razones por las que utilizar datos sintéticos. Ahora, profundizaremos en sus principales ventajas en comparación con los datos reales [36].

- Mayor disponibilidad y accesibilidad de datos: Las herramientas de generación de datos sintéticos permiten una generación ilimitada que evita la escasez de datos. Se pueden producir datos bajo demanda y a gran escala de manera rentable. Además, se pueden generar datos ya etiquetados y categorizados sin la necesidad de procesar información sin formato desde cero. De la misma manera, los datos sintéticos pueden completar los datos existentes, aumentando y equilibrando el volumen total de datos reales disponibles para el entrenamiento de modelos. Además, proporcionan la flexibilidad necesaria para aliviar las restricciones de uso dentro de una organización, permitiendo utilizar los datos de manera más eficiente.
- Protección de la privacidad: La generación de datos sintéticos, que no contienen información sensible o personal, contribuye a mitigar riesgos de privacidad y a cumplir con las regulaciones en materia de protección de datos. En campos como la salud, finanzas o el sector legal son de gran ayuda ya que permiten el análisis e investigación imitando las características y patrones estadísticos de los datos reales sin exponer datos confidenciales.
- Reducción de costes y tiempos de adquisición: La recopilación de datos reales a menudo implica altos costes como el diseño de estudios de investigación o la obtención de permisos y

cumplimiento de normas de protección de datos. En cambio, la creación de datos sintéticos es más eficiente y económica al eliminarse las complejidades asociadas a la recolección y manejo de datos reales. Además, la posibilidad de trabajar con datos sintéticos acelera el desarrollo de proyectos basados en datos, al reducirse el tiempo necesario de procesamiento de los mismos.

- Mayor control sobre calidad de datos y eliminación de datos erróneos o sesgados: Los datos sintéticos permiten a las organizaciones un mayor control sobre la calidad de los datos utilizados. Permite la creación de conjuntos de datos ya “limpios” y homogéneos sin errores humanos, valores atípicos o inconsistencias de los datos reales. De igual modo, los conjuntos de datos reales a menudo están sujetos a sesgos por la forma en la que se recopilaron o por la representación desigual de los mismos, por lo que los datos sintéticos pueden diseñarse de manera que se eliminen dichos sesgos. Además, la creación de datos sintéticos bajo condiciones controladas permite asegurar y ajustar la calidad de los datos.
- Eficiencia en el desarrollo de modelos de IA: El entrenamiento de modelos de IA con datos sintéticos de alta calidad mejora el rendimiento y la capacidad de aprendizaje automático del modelo. Proporcionan una solución a los sobreajustes, permiten iteraciones rápidas dando lugar a entrenamientos de modelos en un tiempo reducido, y permiten adaptabilidad y escalabilidad para ajustarse a nuevas necesidades y diferentes escenarios.

En definitiva, el uso de datos sintéticos presenta múltiples ventajas significativas en comparación con los datos reales. Permiten mayor disponibilidad y accesibilidad, además de facilitar la protección de la privacidad. También reducen costes y tiempos de adquisición, simplificando el proceso de desarrollo de proyectos. Proporcionan un mayor control sobre la calidad de los datos, al eliminar errores y sesgos de los datos reales. Y su uso mejora la eficiencia en el entrenamiento de modelos de IA, optimizando el rendimiento y permitiendo una rápida adaptación a nuevas necesidades.

2.5.4 Retos y riesgos asociados a la generación de datos sintéticos

A pesar de las múltiples ventajas que proporcionan los datos sintéticos, su generación puede conllevar varios desafíos y limitaciones como la evaluación de calidad, el mal uso de los datos o la “contaminación” de los mismos [37].

Los datos sintéticos mal generados pueden conducir a modelos de IA inexactos y poco fiables, por ello la calidad y realismo de los datos sintéticos son cruciales para un entrenamiento efectivo de estos modelos. La integridad de los datos sintéticos dependerá de la metodología y herramientas empleadas para su generación; y de la cantidad de datos reales de aprendizaje del modelo, cuantos más datos se usen, mayor será la calidad y realismo de los datos sintéticos.

A pesar de que uno de los principales beneficios de los datos sintéticos es la protección de la privacidad, con ingeniería inversa se podría revelar información sobre los datos originales, planteando riesgos de seguridad. La re-identificación es un riesgo posible de los datos sintéticos cuando la metodología de generación está demasiado alineada con los datos de entrenamiento originales, a lo que se denomina sobreajuste. Si los datos reales contienen fuertes correlaciones entre atributos, es probable que estas correlaciones se repliquen en los datos sintéticos [38], permitiendo identificar a un individuo específico a partir del conjunto de datos sintéticos.

La introducción de sesgos artificiales, al generar datos sintéticos a partir de un conjunto de datos reales que ya contienen sesgos, puede amplificarse en lugar de corregirse. Además, si los datos generados no están bien equilibrados o no son representativos, pueden introducir nuevos sesgos que afecten el rendimiento del modelo de IA.

Determinar la calidad y eficacia de los datos sintéticos es un desafío importante, pues no es sencillo determinar si un conjunto de datos es suficientemente representativo. Este reto es aún más evidente cuando no hay datos reales con los que comparar. Es necesario utilizar métricas adecuadas como distribuciones estadísticas y pruebas con modelos de aprendizaje automático. La elección del método y técnicas correctas para generar datos sintéticos también puede resultar complicada en función de los

datos de partida, la computación disponible y el propósito de los datos sintéticos. Con frecuencia es necesario experimentar para determinar el enfoque más adecuado para cada caso particular. Además, algunas técnicas avanzadas de generación de datos sintéticos, como las redes generativas adversarias (GANs) o los autocodificadores variacionales (VAEs), requieren de mayores recursos computacionales que pueden limitar su aplicabilidad en entornos con restricciones de hardware o tiempo.

La generación de datos sintéticos también plantea desafíos éticos y legales, pues existe la posibilidad de usar las técnicas de generación de datos sintéticos para la creación de *deepfake*, contenido manipulado con fines malintencionados como fraudes o suplentación de identidad. Pueden generar riesgos de propiedad intelectual especialmente al generarse imágenes sintéticas a partir de fuentes artísticas.

Para mitigar los riesgos asociados a la generación de datos sintéticos en IA se recomienda comprender los objetivos y usos de los datos sintéticos requeridos, usar diversas fuentes de datos de alta calidad, emplear diversos modelos generativos, e identificar y mitigar sesgos de los datos originales y de los propios procesos de generación [36]. Es fundamental evaluar cuidadosamente los riesgos y desafíos asociados para garantizar que los datos sintéticos se utilicen de manera efectiva y ética en el desarrollo de soluciones basadas en IA, ver Anexo II: Reflexiones Éticas y Sociales.

2.5.5 Datos sintéticos AIS para el entorno marítimo

El Sistema de Identificación Automática proporciona información crucial sobre el tráfico marítimo, pero presenta diversas limitaciones, algunas ya mencionadas en el apartado 2.4.3, que afectan a su uso en el entrenamiento de modelos de aprendizaje automático. Estas limitaciones justifican la necesidad de generar datos sintéticos para mejorar la calidad y representatividad de los conjuntos de datos utilizados en modelos predictivos y de análisis de comportamiento de buques.

Uno de los problemas más relevantes en el uso de los datos AIS reales es la escasez de ciertos parámetros, especialmente en zonas con baja densidad de tráfico o para ciertos tipos de embarcaciones, como los pesqueros o los barcos militares, que pueden no transmitir datos AIS durante periodos de tiempo o incluso desactivarlo por razones de seguridad. Además, existe desequilibrio en la distribución de clases, puesto que los cargueros tienen una representación mucho mayor que otros menos frecuentes. También existen errores en la introducción manual de datos, ruido en las transmisiones, manipulación intencionada y, en algunos casos, los datos AIS históricos no están disponibles públicamente o su acceso es restringido debido a normativas de privacidad o costes de adquisición. Muchas bases de datos son privadas y requieren suscripción, lo que limita la capacidad de investigadores para acceder a información completa y de calidad. Además, algunos países aplican restricciones al almacenamiento y distribución de datos AIS, lo que dificulta la recopilación de conjuntos de datos globales y su uso en estudios a gran escala.

Por ello, los datos sintéticos AIS se presentan como una alternativa para el entrenamiento de modelos de IA al:

- Equilibrar la representación de diferentes tipos de embarcaciones.
- Completar o corregir los datos con campos vacíos debidos a errores o falta de cobertura.
- Asegurar la disponibilidad de datos históricos sin restricciones para análisis e investigación.
- Facilitar la simulación de escenarios extremos para el entrenamiento de modelos de detección de anomalías.

2.6 Enfoques de la generación de datos sintéticos

Existen muchos puntos de vista a la hora de generar datos sintéticos. Según el enfoque clásico se distinguen los modelos base, los modelos estadísticos y probabilísticos, y los modelos que emplean *Machine Learning* [39]. En cambio, algunos estudios dividen la generación de datos sintéticos en métodos estándar y en métodos que usan *Deep Learning* [40] y otros estudios lo analizan según el tipo de dato a generar [41]. En este trabajo se orientará la generación de datos sintéticos desde tres enfoques:

según el método de generación, según el tipo de dato sintético a generar y según el propósito de los datos sintéticos.

2.6.1 Según la técnica

En esta clasificación se agrupan los métodos según el enfoque de generación de los datos sintéticos y la técnica empleada para modelar la distribución subyacente en los datos reales. Se distinguen los modelos basados en inteligencia artificial, los modelos probabilísticos y los modelos basados en reglas. Esta clasificación sirve como base para el resto de enfoques.

Generación basada en modelos generativos de IA

Los modelos basados en IA permiten generar datos sintéticos aprendiendo patrones complejos de los datos reales mediante redes neuronales y algoritmos de IA. Existen innumerables variaciones de estos algoritmos y una amplia gama de arquitecturas específicas. Algunos ejemplos más populares incluyen: Redes Generativas Adversarias (GANs) que utilizan una red generadora y una red discriminadora en competencia para generar datos realistas; Autocodificadores (AEs) que aprenden una representación latente de los datos y generan nuevas muestras basadas en esa representación; modelos basados en *Transformers* que emplean un mecanismo único llamado *self-attention* (autoatención) para capturar dependencias y relaciones de largo alcance dentro de los datos de entrada de manera más eficaz y *Large Language Models* (LLMs) como GPT-3 para la generación de texto sintético [42].

Generación basada en modelos matemáticos o probabilísticos

Los modelos categorizados en este grupo generan datos sintéticos mediante modelos estadísticos y probabilísticos que intentan simular los datos reales. Generalmente capturan las estructuras de correlación entre atributos de los datos reales y las imitan en los datos sintéticos. Dentro de estos grupos se incluyen: *Gaussian Copula*, que modela dependencias entre variables mediante funciones de cópula; *Kernel Density Estimation* (KDE), que emplea métricas no paramétricas para estimar la distribución de probabilidad de una variable continua; método de Monte Carlo, que genera datos sintéticos a partir de un muestreo aleatorio; las redes bayesianas (*Bayesian Networks*, BN), que generan datos sintéticos al estimar la distribución conjunta de las variables y muestrear nuevas instancias a partir de ellas; o la técnica de *Synthetic Minority Over-sampling Technique* (SMOTE), que genera nuevas instancias interpolando valores.

Generación basada en reglas y simulaciones

Los modelos basados en reglas y simulaciones generan datos sintéticos estableciendo normas explícitas o modelos computacionales que imitan las características y relaciones de los datos reales [42]. Estas técnicas garantizan que los datos generados respeten restricciones lógicas y patrones definidos. Engloban simulaciones computacionales con *Agent Based Models* (ABM), simulaciones probabilísticas con la técnica de Monte Carlo, modelos de transformación o expansión de datos como *Synthetic Minority Over-sampling Technique* (SMOTE) y reglas de generación sintética (*Rule-Based Data Synthesis*) que se basan en la definición manual de reglas para generar datos en base a relaciones lógicas conocidas.

2.6.2 Según la estructura de los datos

La clasificación anterior se basó en la forma de generar los datos sintéticos. Sin embargo, dependiendo del tipo de datos disponibles, la generación de datos sintéticos puede enfocarse en las diferentes estructuras de los datos. La Figura 2-5 muestra una taxonomía detallada de los principales tipos de datos y los modelos generativos utilizados para su generación. Se distinguen los datos estructurados, como los tabulares y de series temporales, y los datos no estructurados, que incluyen imágenes, audio y vídeo.

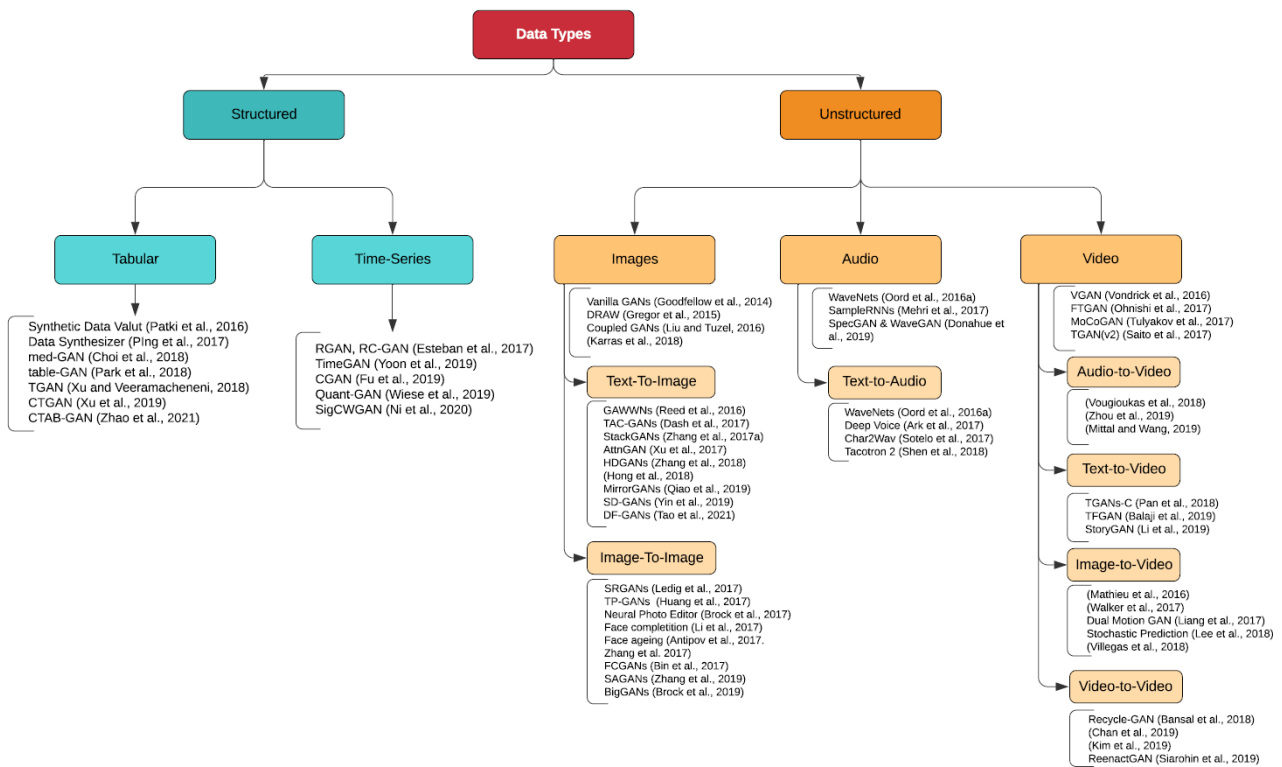


Figura 2-5: Esquema de modelos generativos según el tipo de dato generado (tomada de [43])

A continuación, se presentan las principales categorías de datos y las principales técnicas utilizadas para generarlos:

Datos tabulares

Aquellos datos que se presentan en formato estructurado con filas y columnas, de forma similar a bases de datos y hojas de cálculo. Entre los métodos actuales para generar datos tabulares sintéticos, como lo son los datos AIS, destacan los modelos estadísticos y probabilísticos como *Gaussian Copula*, y los modelos generativos como *Conditional Tabular GAN* (CTGAN) y *Tabular Variational Autoencoder* (TVAE), pero también se engloban en el resto de metodologías de generación.

Datos de series temporales

Las series temporales representan valores que evolucionan a lo largo del tiempo y su generación debe preservar las dependencias temporales. Algunos modelos generativos como las Redes Neuronales Recurrentes (RNNs) y *Transformers*, y modelos específicos de GAN como *TimeGAN* han demostrado ser efectivos en la generación de este tipo de datos. Estos modelos se aplican en sectores como finanzas, meteorología y transporte.

Datos espaciales o geográficos

Estos datos incluyen coordenadas geográficas y patrones de movilidad. En generación sintética de datos espaciales se utilizan enfoques basados en simulaciones y modelos probabilísticos para recrear rutas de tráfico marítimo o vehicular. Modelos generativos como *StyleGAN* para imágenes de mapas, y simulaciones probabilísticas como las técnicas de Monte Carlo, que pueden modelar rutas de tráfico marítimo o patrones de movilidad.

Datos de imagen o vídeo

El aprendizaje profundo ha permitido avances en la generación de datos sintéticos de imagen y vídeo. Las técnicas predominantes para la generación de imágenes o vídeos sintéticos son los modelos GAN como *CycleGAN*, *BigGAN* y los *Diffusion Models* que permiten generar datos visuales realistas.

2.6.3 Según el propósito de los datos

Dependiendo de la finalidad de los datos generados y de las necesidades del modelo de aprendizaje automático, los datos sintéticos pueden ser generados en función de diferentes objetivos que van desde la ampliación de un conjunto de datos hasta la protección de la privacidad de información sensible.

Para aumentar el tamaño del dataset (*Data Augmentation*)

El aumento de datos es una técnica utilizada para incrementar el tamaño de un conjunto de datos mediante transformaciones sobre los datos reales existentes en lugar de generar muestras desde cero. Destacan técnicas probabilísticas como SMOTE o mediante el uso de técnicas generativas como GAN y AE. A diferencia de la generación de datos sintéticos, que crea datos completamente nuevos a partir de un modelo que aprende la distribución de los datos reales, el *Data Augmentation* modifica y expande los datos existentes aplicando transformaciones como perturbaciones, combinaciones o interpolaciones para mejorar la representatividad del conjunto de datos sin alterar significativamente su significado [44]. Hasta la fecha, las técnicas de *Data Augmentation* han sido las más usadas y exitosas dentro del mundo de la generación de datos sintéticos [43].

Para eliminación de sesgos y mejorar representatividad

Los sesgos en los datos pueden originar errores en la posterior predicción de los modelos de aprendizaje automático. Para mitigarlos y mejorar la representatividad de los datos, se emplean técnicas generativas como *FairGAN* y *WGAN*, que generan datos sintéticos que no contienen los sesgos presentes en los datos originales. También se emplean modelos probabilísticos de reasignación de pesos (*reweighting*).

Para preservar la privacidad

La generación de datos sintéticos con el objetivo de garantizar la privacidad emplea diversas técnicas generativas como *Differentially Private Wasserstein Generative Adversarial Networks* DPWGAN [45]. Los métodos basados en privacidad diferencial generan datos sintéticos incorporando ruido aleatorio cuidadosamente calibrado a los datos originales, con el objetivo de proteger la información sensible [42].

Para simulación de escenarios o situaciones extremas

La generación de datos sintéticos puede estar enfocada para la simulación de escenarios extremos que no se dan en la vida real, lo que permite entrenar a modelos y evaluar la resiliencia y respuesta de los mismos. Los modelos de generación de datos sintéticos más empleados son los basados en reglas como las técnicas de Monte Carlo o los ABM, que implican el modelado de agentes individuales como células o programas informáticos para la interacción en un entorno virtual [46].

2.7 Técnicas de generación de datos sintéticos

Existe una amplia variedad de técnicas, algoritmos y modelos utilizados para la generación de datos sintéticos como se muestra en la Figura 2-6, pero para este trabajo se seguirá el enfoque de generación de datos según la metodología y se profundizará en las técnicas generativas y en las técnicas probabilísticas. Y dado que no sería posible enumerar todos los elementos de la literatura, se presenta una visión general de los modelos principales. Los dos modelos generativos más utilizados para la creación de datos sintéticos son las Redes Generativas Adversarias (GANs) y los Autocodificadores (AEs). Entre las diversas técnicas probabilísticas disponibles, en este trabajo se emplearán dos enfoques clave: *Gaussian Copula*, una técnica basada en cópulas que modela la dependencia entre variables, y SMOTE (*Synthetic Minority Over-sampling Technique*), un método diseñado para mejorar la representatividad de clases minoritarias en conjuntos de datos desbalanceados.

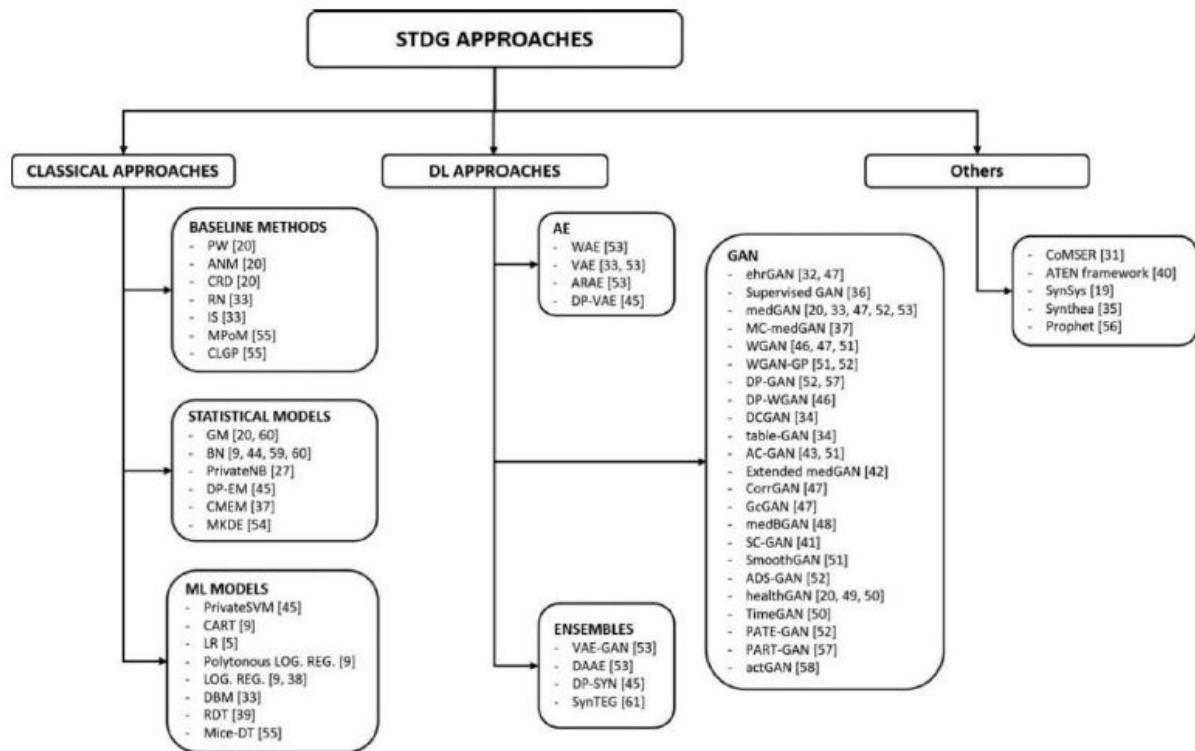


Figura 2-6: Clasificación de las técnicas de generación de datos sintéticos (tomada de [39])

2.7.1 Redes Generativas Adversarias (GANs)

Las Redes Generativas Adversarias (GANs) son un enfoque de aprendizaje profundo basado en un proceso competitivo entre dos redes neuronales, propuesto por Ian J. Goodfellow en 2014. A lo largo de los años, estas arquitecturas han evolucionado y mejorado significativamente, convirtiéndose en una herramienta avanzada para la generación de datos sintéticos. Actualmente, las GANs han demostrado gran capacidad para crear contenido en diversas áreas como imágenes, vídeo o texto.

Una GAN se compone de una red neuronal generadora y otra discriminadora que, como su nombre indica, son opuestas. La red generadora trata de generar muestras que siguen la distribución subyacente de los datos reales, mientras que la discriminadora trata de clasificar el conjunto de datos como falsos si detecta que son sintéticos. Es decir, compiten entre sí. Mientras que la red generadora (G) crea puntos de datos que imitan la distribución de los datos originales con el objetivo de engañar al discriminador, la red discriminadora (D) trabaja para distinguir entre las observaciones generadas y las reales. Este proceso competitivo impulsa la mejora continua de ambas redes: el generador perfecciona sus muestras sintéticas para hacerlas más realistas, mientras que el discriminador refina su capacidad de detección. A medida que el entrenamiento avanza, el generador logra producir datos cada vez más similares a los reales, dificultando la tarea del discriminador [40].

Para explicar cómo funcionan las GANs, se divide en el entrenamiento de la red generadora y en el del discriminador por otro lado.

- La red generadora produce datos sintéticos a partir de un vector de ruido aleatorio, y utiliza funciones de activación no lineales para transformar el ruido en una salida que imite la distribución de los datos reales.
- La red discriminadora evalúa si los datos proporcionados son reales o generados artificialmente comparándolos con los datos reales de entrenamiento. Al igual que el generador, emplea funciones de activación no lineales para analizar la autenticidad de los datos. El discriminador asigna a cada observación una probabilidad de 0 a 1, que indica la confianza en que dicha

observación provenga del conjunto real. Un valor cercano a 0 indica que el discriminador está casi seguro de que la muestra ha sido generada artificialmente, mientras que un valor próximo a 1 significa que el discriminador considera que la muestra proviene de los datos reales.

Ambas redes se entrenan simultáneamente en un proceso de retroalimentación constante, que utiliza funciones de pérdida para guiar la mejora, como se muestra en la Figura 2-7. La función de pérdida del discriminador mide la precisión del discriminador al clasificar los datos como reales o sintéticos, mientras que la función de pérdida del generador mide la capacidad del generador para engañar al discriminador. Si el discriminador identifica la muestra como falsa, envía una retroalimentación negativa al generador, que ajusta sus parámetros para mejorar la calidad de las muestras. Si, por el contrario, la acepta como real, ofrece retroalimentación positiva, mostrando que el generador se acercó a producir una salida convincente. Este proceso permite que el generador mejore en la creación de datos similares a los reales, mientras que el discriminador afina su capacidad para diferenciarlos.

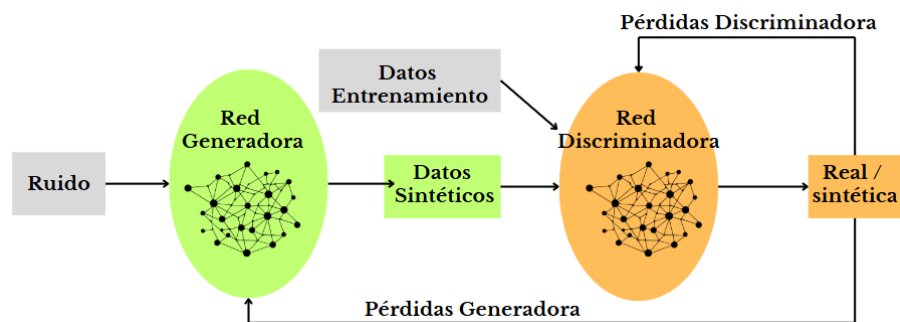


Figura 2-7: Taxonomía de GANs (fuente propia)

Aunque las GANs básicas representan una idea sólida para la generación de datos sintéticos, presentan diversas desventajas que dificultan su entrenamiento. Entre los principales desafíos se encuentran la inestabilidad en la pérdida, la dependencia de hiperparámetros críticos y la posibilidad de que el generador aprenda a engañar al discriminador con facilidad, afectando el equilibrio del modelo, a lo que se conoce como colapso modal. Para abordar estos problemas, se han desarrollado variantes de GANs más robustas y especializadas [40]. La Tabla 2-2 muestra un resumen de algunas arquitecturas GANs existentes.

Arquitectura GAN	Descripción
Conditional GAN (CGAN)	Introduce etiquetas de clase como entrada en el generador y discriminador para generar datos condicionados.
Deep Convolutional GAN (DCGAN)	Utiliza redes convolucionales profundas para mejorar la calidad de imágenes
InfoGAN	Introduce variables latentes interpretables para obtener representaciones significativas
Wasserstein GAN (WGAN)	Modifica la función de pérdida para mejorar la estabilidad y reducir el colapso modal
CycleGAN	Traducción de imágenes entre dominios sin necesidad de pares de imágenes alineadas
StackGAN	Generación de imágenes en dos etapas para mejorar la calidad y resolución
Self-Attention GAN (SAGAN)	Incorpora mecanismos de autoatención para mejorar los detalles en imágenes
BigGAN	Permite generar imágenes a gran escala con mayor calidad y diversidad
StyleGAN	Introduce un modelo de generación basado en mapeo latente para mejorar el control de las imágenes generadas

Tabla 2-2: Arquitecturas de GANs

Las GANs han demostrado ser muy eficaces en la generación de imágenes, sin embargo, su aplicación en datos tabulares presenta problemáticas debido a las propiedades únicas de este tipo de dato. Los datos tabulares incluyen tanto variables continuas como categóricas, lo que dificulta el uso de las arquitecturas GAN tradicionales diseñadas para imágenes. Además, es común encontrar distribuciones más complejas en datos tabulares que en imágenes. Y en muchos conjuntos de datos tabulares, como en el caso de este trabajo, las clases minoritarias en variables categóricas están altamente desbalanceadas, lo que puede llevar a un colapso modal. Para abordar estos problemas, de variables mixtas, distribuciones complejas y problemas de colapso modal, se desarrollaron arquitecturas GAN especializadas para datos tabulares como *TableGAN* (TGAN), *Conditional Tabular GAN* (CTGAN) y *TabFairGAN*.

TGAN fue una de las primeras arquitecturas GAN diseñadas específicamente para la generación de datos tabulares. Fue desarrollada con el propósito de manejar variables categóricas y continuas mediante muestreo condicional de manera eficiente. Permite capturar relaciones complejas entre características sin la necesidad de pre-procesamientos adicionales. Pero su desempeño en la generación de variables desbalanceadas es limitado. CTGAN es una mejora de TGAN, desarrollada por los mismos autores, con el objetivo de solucionar las limitaciones en la generación de variables continuas y categóricas. Su arquitectura incorpora un mecanismo de muestreo condicional adaptativo, permitiendo que el generador aprenda con mayor precisión la distribución de datos tabulares. Utiliza un mecanismo de normalización Gaussiana para mejorar la representación de variables continuas. Mejora la estabilidad de los entrenamientos y evita el colapso modal para asegurar una mejor representación de clases minoritarias. *TabFairGAN* es una evolución de CTGAN que mejora la equidad en la generación de datos tabulares y evita sesgos en los datos sintéticos [40].

2.7.2 Autocodificadores (AEs)

Un autocodificador es un tipo de red neuronal que consta de dos partes: una red codificadora (*encoder*) que comprime los datos de entrada de alta dimensión en una representación espacial de menor dimensión denominada código latente, que captura las características más relevantes de los datos originales, y una red decodificadora (*decoder*) que descomprime dicha representación comprimida en el dominio original con el objetivo de minimizar la diferencia entre la entrada y la salida (función de pérdida del *autoencoder*), asegurando que la reconstrucción sea lo más precisa posible. La Figura 2-8 representa el mecanismo de un *autoencoder*.

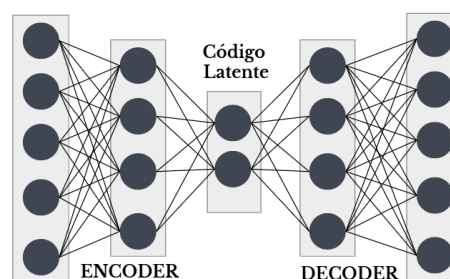


Figura 2-8: Arquitectura de un *autoencoder* (fuente propia)

En la parte izquierda, se muestra la estructura de un codificador, donde la capa de entrada posee más unidades que la capa intermedia, la cual reduce la dimensionalidad en el código latente. En la parte derecha, el decodificador utiliza el código para reconstruir la información en su dominio original, generando una salida que busca replicar la entrada inicial.

Desde un enfoque formal, el codificador se puede entender como una función $c = E(x)$ y el decodificador como $r = D(c)$, pero el propósito del autocodificador no es aprender a reproducir exactamente la entrada a través de la ecuación $D(E(x)) = x$, sino más bien capturar las características y patrones esenciales de los datos originales, para reconstruir con fidelidad datos de salida. Esta restricción fuerza al modelo a identificar y priorizar las características más relevantes de los datos, lo

que ha permitido que los autocodificadores sean ampliamente utilizados en tareas como aprendizaje de representaciones y reducción de dimensionalidad dentro del aprendizaje profundo [40].

Los autocodificadores son utilizados en diversas aplicaciones, como reducción de dimensionalidad, detección de anomalías, generación de datos sintéticos y eliminación de ruido en imágenes y señales. Los *autoencoders* presentan múltiples variantes según el problema a resolver. Los *vanilla autoencoders* son la versión más básica y se utilizan para reducción de dimensionalidad y eliminación de ruido. Los *sparse autoencoders* añaden una restricción de escasez en las activaciones, lo que permite extraer características más relevantes. Para el procesamiento de imágenes, los *autoencoders convolucionales* utilizan capas convolucionales para capturar estructuras espaciales. En datos secuenciales, los *sequence-to-sequence autoencoders* emplean redes recurrentes como *Long Short-Term Memory*, *LSTM* (Memoria a Largo y Corto Plazo) o *Gated Recurrent Unit*, *GRU* (Unidad Recurrente Con Puertas), que son arquitecturas de red neuronal recurrente (RNN), facilitando el análisis de series temporales y traducción automática. Los *denoising autoencoders* que introducen ruido durante el entrenamiento para mejorar la reconstrucción de datos limpios [47].

Los autocodificadores enfrentan ciertos retos en la generación de datos sintéticos. Uno de los principales inconvenientes es que la distribución de puntos en el espacio latente no está bien definida. Es decir, al muestrear un punto aleatorio y pasarlo por el decodificador, no hay garantía de que genere una muestra coherente con los datos originales. También puede existir falta de diversidad en las muestras generadas pues el *autoencoder* puede fijarse en reconstruir los patrones más frecuentes del conjunto original. Y, además, los puntos que representan a una misma clase en el espacio latente pueden estar dispersos, formando regiones separadas con vacíos significativos. Para mitigar estas limitaciones, se desarrollaron los Autocodificadores Variacionales (VAE), que incorporan un enfoque probabilístico en el espacio latente [40].

Los VAEs combinan los principios de los AEs con técnicas de inferencia variacional, lo que permite aproximar distribuciones complejas y generar nuevos datos que siguen la misma distribución. Es decir, a diferencia de los *autoencoders* clásicos, que simplemente comprimen y reconstruyen datos, los VAEs modelan la distribución subyacente de los datos en un espacio latente probabilístico, permitiendo generar nuevas muestras similares a las originales. El codificador no convierte la entrada en un solo punto del espacio latente, sino que genera dos vectores: media y desviación estándar. Estos parámetros definen una distribución gaussiana en el espacio latente, que permite muestrear diferentes puntos en lugar de obtener una única representación. El decodificador a partir de un punto muestreado de la distribución reconstruye la muestra con mayor diversidad en comparación con los *autoencoders* tradicionales. La Figura 2-9 muestra un esquema del funcionamiento de los VAEs.

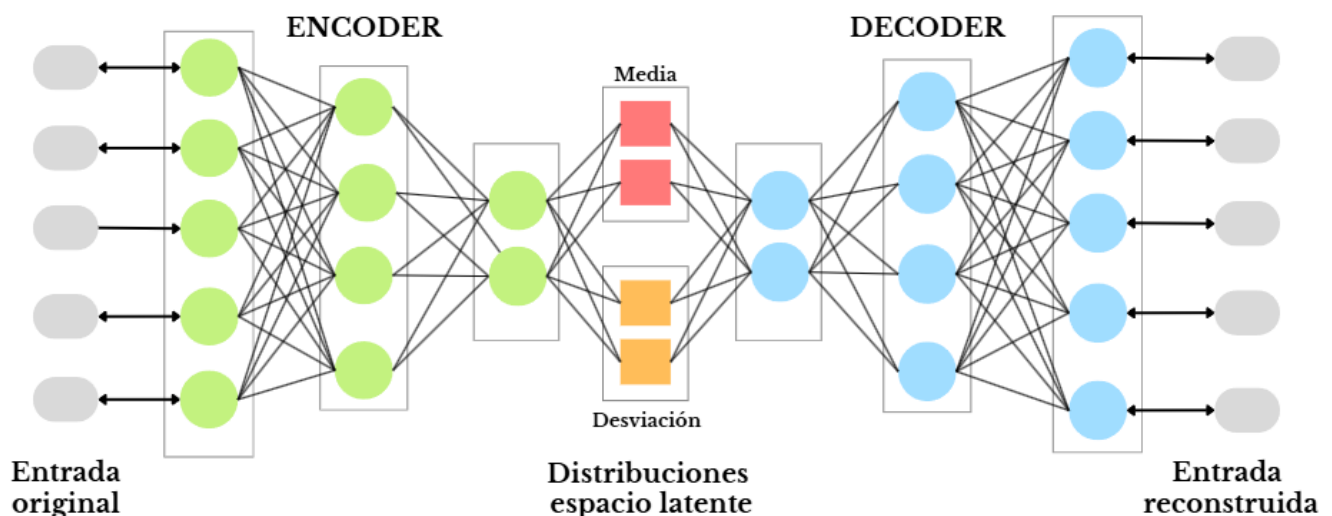


Figura 2-9: Arquitectura de un VAE (fuente propia)

Existen múltiples variantes de VAEs, diseñados para la mejora de la generación de datos sintéticos en diferentes dominios. Algunos ejemplos son: *Ladder Variational Autoencoder* (LVAE), que optimiza el entrenamiento de VAEs profundos mediante un enfoque recursivo, mejorando la calidad de la representación latente; *Vector Quantized-Variational AutoEncoder* (VQ-VAE), que introduce representaciones latentes discretas para evitar el colapso posterior, mejorando la generación de imágenes y audio; *Oblivious Variational Autoencoder* (OVAE), que combina VAEs con árboles de decisión diferenciables para generar datos tabulares sintéticos con alta fidelidad y privacidad o *Causal Recurrent Variational Autoencoder* (CR-VAE) para series temporales, que aprende gráficos de causalidad a partir de datos multivariados, permitiendo una mejor comprensión de patrones subyacentes. Para datos tabulares destaca el modelo *Tabular VAE* (TVAE) que maneja tanto datos continuos como categóricos. Adicionalmente, el modelo C-VAE (*Contrastive VAE*) que aplica VAEs para el sobremuestreo de datos tabulares desbalanceados, mejorando el rendimiento en tareas de aprendizaje supervisado [42].

2.7.3 Gaussian Copula

Otra técnica probabilística presente en la literatura es el uso de cópulas, funciones matemáticas que describen la relación entre dos o más variables aleatorias sin definir sus distribuciones marginales, es decir, sin la necesidad de asumir una distribución específica para cada variable. En particular, una cópula es una distribución de probabilidad multivariada en la que cada variable sigue una distribución uniforme. Dichas variables se definen a partir de la función de distribución acumulativa de un vector aleatorio. En esencia, una cópula es una técnica utilizada en estadística para describir cómo se relacionan las variables aleatorias, separando la dependencia entre ellas de sus distribuciones individuales.

Gaussian Copula es un tipo específico de cópula que se basa en la distribución normal multivariada. En términos sencillos, toma un conjunto de datos, los transforma en una forma donde siguen una distribución normal estándar, analiza la correlación entre las variables en este nuevo espacio y luego genera nuevos datos que respetan estas relaciones [48].

2.7.4 Synthetic Minority Oversampling Technique (SMOTE)

Synthetic Minority Oversampling Technique, propuesto en 2002, es uno de los algoritmos más antiguos, junto con *Random OverSampling* (ROS) y otros algoritmos tradicionales, que trata de replicar una distribución de datos.

Es una técnica ampliamente utilizada para aumentar la representación de clases minoritarias en conjuntos de datos desbalanceados. A diferencia de los enfoques tradicionales de sobremuestreo, que simplemente replican instancias existentes, SMOTE genera nuevas muestras sintéticas interpolando entre los datos originales de la clase minoritaria.

Dado un punto de datos de una clase minoritaria y su vecino más cercano de la misma clase, se determina la distancia entre ellos (la distancia se calcula como la diferencia entre ambos vectores de características). Esta distancia se multiplica por un número aleatorio entre 0 y 1 y se suma al punto de datos seleccionado. Esto hace que la nueva muestra se sitúe en el segmento de línea entre la muestra original y su vecino. El mismo proceso se repite hasta alcanzar el número deseado de muestras. La Figura 2-9 muestra un ejemplo de una iteración del algoritmo SMOTE [40].

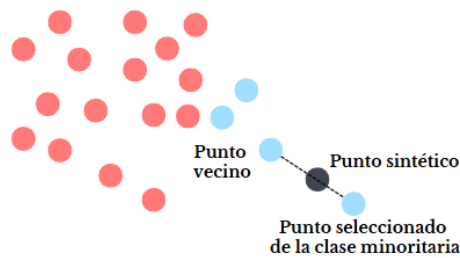


Figura 2-10: Ejemplo de iteración con SMOTE (fuente propia)

Al igual que las GANs, la técnica de SMOTE ha experimentado varias modificaciones, desde que se propuso inicialmente para resolver diversos problemas en distintos dominios.

2.8 Selección de metodología

Tal y como se ha presentado en el apartado anterior, actualmente hay una gran variedad de métodos, algoritmos y modelos para la generación de datos sintéticos, y cada uno presenta fortalezas y limitaciones adaptadas a cada tipo específico de dato. En este trabajo, se realizó una revisión bibliográfica para identificar las metodologías más relevantes y se ha optado por utilizar los modelos proporcionados por la biblioteca SDV (*Synthetic Data Vault*) [49].

SDV es una biblioteca de código abierto desarrollada originalmente en 2016 por el MIT Data to AI Lab, pero actualmente bajo el desarrollo de *DataCebo*. Se ha consolidado como una de las herramientas más avanzadas para la generación y evaluación de datos sintéticos, en especial para datos tabulares, aunque incluye una amplia variedad de modelos en distintos dominios como series temporales, datos multivariantes y estructuras relacionales. Además, permite modificar los parámetros de los modelos para adaptarlos a necesidades específicas e incluye herramientas de evaluación de los datos sintéticos. Gracias a estas características, SDV se presenta como una herramienta idónea para el desarrollo del presente trabajo.

En particular, se trabajará con tres modelos principales dentro de esta biblioteca: *Gaussian Copula* con su enfoque probabilístico basado en la modelación de dependencias entre variables mediante cópulas gaussianas, CTGAN, para el modelo generativo de datos tabulares mediante redes neuronales adversarias, y TVAE con su *autoencoder variacional*, diseñado para aprender representaciones latentes en datos tabulares y generar muestras sintéticas realistas.

Cada uno de estos modelos será evaluado en profundidad para determinar su desempeño en la generación de datos sintéticos AIS estáticos. Como resultado de esta comparación, se identificará la metodología más adecuada para el entrenamiento de sistemas basados en aprendizaje automático supervisado. Se evaluará la viabilidad del uso de datos sintéticos AIS estáticos para mejorar el rendimiento de un algoritmo de clasificación de tipos de buques.

2.9 Estudios relacionados

En este apartado, se presentan diferentes estudios que han servido como referencia para este TFG, y han permitido abordar adecuadamente ciertos aspectos. En primer lugar, se hará una breve revisión bibliográfica de estudios relacionados con la generación de datos sintéticos en diferentes ámbitos, como en la sanidad, y trabajos sobre la generación de datos sintéticos tabulares. Se continuará con una búsqueda de trabajos relacionados con la inteligencia artificial generativa aplicada al ámbito marítimo, centrada en el análisis de datos AIS y en sistemas de clasificación. Finalmente, se explicarán los TFGs predecesores a este de la misma línea de investigación desarrollada por el CUD-ENM.

En el primer bloque destacan trabajos como el de Nohemí Sánchez Medel titulado *Reconocimiento de células sanguíneas por medio de aprendizaje profundo y generación de datos sintéticos* [50], que aplica técnicas de *Data Augmentation* en el análisis de imágenes del ámbito médico, demostrando la

utilidad de los datos sintéticos para mejorar el rendimiento de modelos de clasificación. Aunque su enfoque se centra en el ámbito de la medicina, los principios de *Data Augmentation* pueden aplicarse en la generación de datos AIS para equilibrar la representación de distintas clases de embarcaciones. En el contexto de la generación de datos tabulares, Marc Nebot Moyano en su trabajo: *Métodos para la generación de datos artificiales tabulares* [51], desarrolló una librería en *PyTorch* para evaluar diferentes modelos generativos (GANs, VAEs, *Bit Diffusion*), mientras que Sergio Arroni del Riego, en su Trabajo Fin de Master: *Métodos para la generación de datos artificiales tabulares* [52], explora el uso de modelos de difusión y redes neuronales basadas en Transformers para la generación de datos tabulares sintéticos. Su estudio enfatiza la importancia de preservar la estructura estadística de los datos generados.

En segundo lugar, se incorporan estudios aplicados al ámbito marítimo. Uno de los más relevantes es el trabajo de Chris M. Ward, Josh Harguess y Cameron Hilton, titulado "*Ship Classification from Overhead Imagery Using Synthetic Data and Domain Adaptation*" [53]. En este estudio, los autores abordan el problema de la clasificación de buques a partir de imágenes aéreas mediante el uso de datos sintéticos y técnicas de adaptación de dominio. Utilizando el motor de videojuegos *Unity* y modelos 3D de barcos, generaron un conjunto de datos sintéticos de 200,000 imágenes, lo que permitió mejorar significativamente el rendimiento de las redes neuronales convolucionales en tareas de clasificación, especialmente en escenarios con escasez de datos reales. También destaca el proyecto desarrollado por HI-Iberia y la DGAM (Dirección General de Armamento y Material) en el marco del programa COINCIDENTE que dio lugar al sistema SEDA (Sistema de Explotación de Datos del Ambiente) que se utiliza para recoger, procesar, analizar y presentar información del entorno marítimo, procedente de sensores o sistemas como el AIS, utilizado por COVAM. En este contexto, el trabajo [54], cuyo objetivo es la detección de abarloomientos de buques mediante imágenes satelitales, recurre a la generación de datos sintéticos, combinando imágenes reales y fondos marinos artificiales para entrenar modelos de detección y clasificación de dichos abarloomientos. Este proyecto demuestra de forma práctica el valor de los datos sintéticos para entrenar modelos de IA cuando los datos reales son escasos o no están etiquetados.

Por último, se destacan los TFG predecesores desarrollados en el CUD-ENM. El trabajo de Gonzalo Rodríguez Casajús, titulado *Predicción de tipo de buque utilizando datos AIS y técnicas de inteligencia artificial* [3], que abordó la predicción del tipo de buque a partir de datos AIS estáticos y dinámicos mediante técnicas de aprendizaje automático supervisado basado en *Random Forest*. Por otro lado, el TFG de Jorge Roldán Calvo, titulado *Predicción de tipo de buque utilizando representaciones de trayectorias y técnicas de Inteligencia Artificial* [4], propuso un enfoque alternativo basado en la representación de trayectorias AIS para la clasificación de buques, demostrando que la inclusión de datos cinemáticos aporta mejoras significativas en el rendimiento de los modelos. Ambos trabajos evidencian la necesidad de conjuntos de datos más completos y diversos, lo que motiva directamente el enfoque seguido en el presente TFG mediante la generación de datos sintéticos AIS.

Estos estudios previos han proporcionado un marco teórico para la presente investigación. A diferencia de ellos, este TFG se enfoca en la generación de datos AIS estáticos sintéticos con el objetivo de evaluar la viabilidad de distintas técnicas generativas en la clasificación de buques, asegurando que los datos sintéticos conserven relaciones matemáticas clave y puedan integrarse en modelos de aprendizaje automático. No solo complementa estas líneas de trabajo, sino que aporta una posible solución para abordar la problemática de los campos no cubiertos en los mensajes AIS o permitir mayor representación de clases minoritarias.

3 DESARROLLO DEL TFG

Tras haberse analizado las distintas metodologías y técnicas de generación de datos sintéticos, junto con las ventajas que ofrecen en los diferentes ámbitos de aplicación, en este capítulo, se presenta en detalle cada solución propuesta para generar datos AIS estáticos de forma sintética. Para ello, se ha empleado el entorno *Jupyter* y el lenguaje de programación Python con sus bibliotecas especializadas en el tratamiento y modelado de datos.

Se documentan todas las pruebas realizadas, desde aquellas cuyo objetivo fue obtener una familiarización con los datos de partida, seguida de la generación de datos sintéticos estáticos sin distinción por tipo de barco, hasta la generación de datos sintéticos estáticos por tipo de barco. También se explica el proceso de cálculo de atributos y por último se describe una prueba de *Data Augmentation* de la clase *passenger* en los datos reales.

3.1 Entorno de trabajo

En este apartado se describe el entorno de trabajo y bibliotecas utilizadas para el desarrollo de este proyecto. Para llevar a cabo las tareas de programación se utilizó el entorno de trabajo *Jupyter Lab* y el lenguaje de programación empleado para la elaboración y evaluación de los datos sintéticos fue *Python*, debido principalmente a la existencia de múltiples librerías para la generación y evaluación de datos sintéticos, las cuales se presentarán a continuación.

3.1.1 *Jupyter Lab*

Jupyter Lab es una aplicación web interactiva de código abierto que permite la escritura y ejecución de código Python de manera eficiente, en celdas de forma modular, facilitando el desarrollo y visualización de los resultados de manera organizada [55]. Además, *Jupyter Lab* es una evolución de *Jupyter Notebooks*, lo que ofrece aún más ventajas como integrar texto explicativo, gráficos y visualizaciones directamente en el mismo entorno, facilitando la documentación del trabajo; y una interfaz más flexible y modular que permite trabajar con múltiples cuadernos y pestañas dentro del mismo entorno de trabajo, como se muestra en la Figura 3-1.

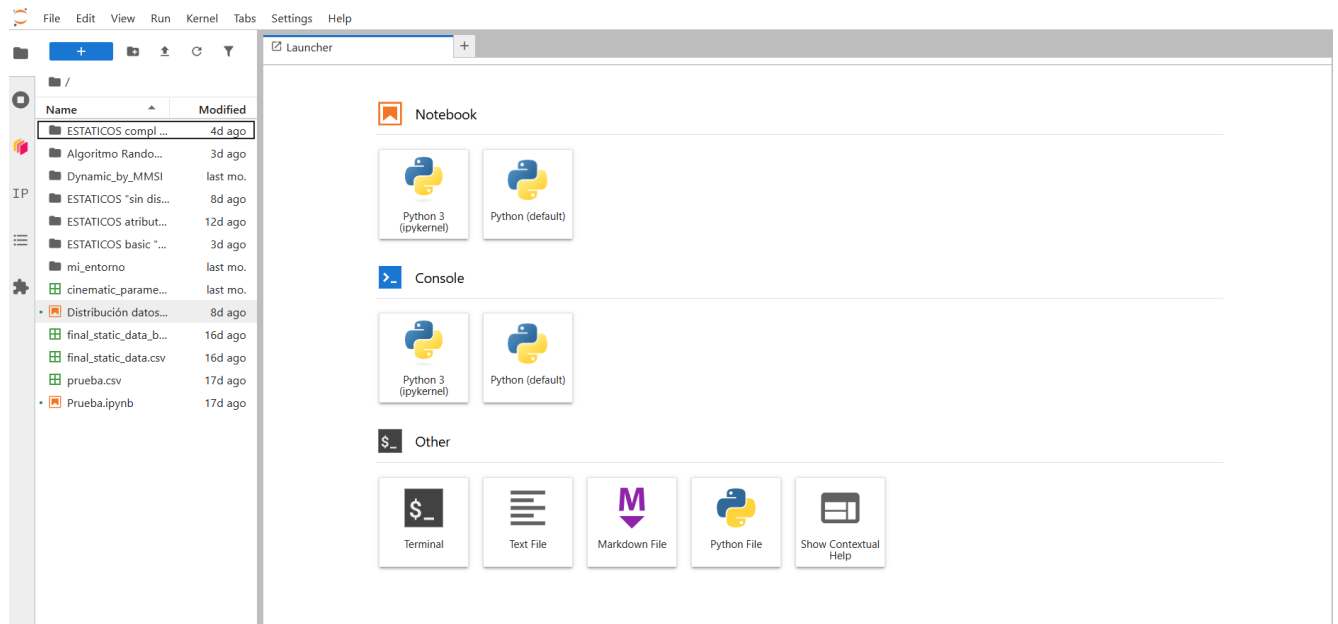


Figura 3-1: Interfaz de Jupyter Lab

Surgió en 2014, y su nombre proviene de la unión de los tres lenguajes de programación principales en los que se basó: Julia, Python y R, pero es compatible con más de 50 lenguajes de programación. Además, como se ve en la Figura 3-2, el logotipo es un homenaje a Galileo Galilei, quien documentó el descubrimiento de las lunas de Júpiter [56].



Figura 3-2: Logotipo de Jupyter [57]

El programa se ejecuta en un servidor del CUD-ENM y es accesible de forma remota en un equipo propio. También funciona desde la aplicación web para cualquier navegador, es decir, no requiere de instalación de aplicaciones adicionales en los equipos cliente, permitiendo trabajar de forma online. Además, aporta otras ventajas como diferentes opciones de formato a la hora de exportar y compartir los resultados en HTML, PDF, *Markdown*, *script* de *Python* o fichero *Notebook* [58]. Permite la colaboración entre usuarios con acceso simultáneo a cuadernos de *Jupyter Lab*, pero ejecutando en el servidor de forma independiente para cada uno de ellos. Estas características hacen que *Jupyter Lab* sea una herramienta ideal para este trabajo.

3.1.2 Bibliotecas en Python

Python es un lenguaje de programación de alto nivel, interpretado y multiparadigma, ampliamente utilizado en el desarrollo de aplicaciones científicas, programación web, análisis de datos e inteligencia artificial [59]. Su sintaxis clara y legible facilita su adopción tanto por principiantes como por programadores experimentados. Su amplia y activa comunidad contribuye con múltiples bibliotecas especializadas como *NumPy*, *Pandas* y *Scikit-learn*, esenciales en el análisis de datos, que amplían su funcionalidad, haciendo de Python una excelente opción para tareas de investigación y aprendizaje automático. Gracias a su capacidad de tipificación dinámica y soporte para múltiples paradigmas, Python se ha convertido en uno de los lenguajes más populares. En este trabajo, Python ha sido clave para la generación de datos sintéticos mediante técnicas probabilísticas y generativas, así como para la evaluación de la calidad de los datos sintéticos generados en comparación con los datos reales.

La versión de Python utilizada en este proyecto fue *Python 3.12.3* garantizando compatibilidad con las bibliotecas necesarias que se detallan a continuación:

Para la manipulación y procesamiento de datos:

- *Pandas*: es una biblioteca fundamental para el procesamiento de datos que proporciona estructuras de datos eficientes como *dataframes* (*bidimensional*), y *series* (*unidimensional*), la lectura de archivos CSV (Valores Separados por Comas) y funciones para limpieza, filtrado y análisis de datos.
- *Numpy*: esta biblioteca permite la computación numérica y análisis de datos de gran volumen, trabajando con matrices y *arrays*.

Para la generación de datos sintéticos:

- *SDV (Synthetic Data Vault)*: es una biblioteca de Python desarrollada por *MIT Data to AI Lab* en 2016, pero actualmente bajo el desarrollo de *DataCebo*, para la generación de datos sintéticos tabulares con dependencias de variables [49]. En particular, se emplearon *TVAESynthesizer*, *CTGANSynthesizer* y *GaussianCopulaSynthesizer* de SDV para la generación de los datos sintéticos AIS.

Para el análisis y validación de los datos sintéticos:

- *Matplotlib* y *seaborn*: son bibliotecas de visualización de datos que permiten crear gráficos de línea, barras, dispersión, histogramas y más.
- *Scikit-learn*: es una biblioteca de aprendizaje automático en Python que permite el análisis y validación de datos. Incluye una amplia variedad de algoritmos de aprendizaje supervisado y no supervisado, así como de herramientas para la evaluación de modelos.
- *Scipy*: es una biblioteca de herramientas avanzadas para cálculo científico, matemático y estadístico como optimización, álgebra lineal, integración, interpolación, procesamiento de señales y pruebas estadísticas como *KS-test* o Chi-cuadrado.

3.2 Flujo de trabajo

La Figura 3-3 representa el flujo de trabajo general que todo proceso de generación de datos sintéticos sigue, el cual consta de cuatro etapas:

- La adquisición de datos que involucra la recolección y gestión de los datos reales.
- La preparación y tratamiento de los datos que implica el procesado y análisis de los datos reales para eliminar inconsistencias como valores inverosímiles, valores atípicos o valores perdidos que pueden afectar negativamente al modelo de generación de datos [60].
- La generación de los datos que supone la aplicación de las diferentes metodologías y modelos de generación (generativos, probabilísticos, simulaciones...) para imitar las propiedades de los datos reales.
- Y la evaluación de calidad de los datos generados para asegurar que cumplen con los estándares requeridos de fidelidad y utilidad.

Para este trabajo se dispone de un conjunto de datos procesado y generado en el CUD – ENM a partir de tráfico real AIS, de dos meses, proporcionado por el COVAM.

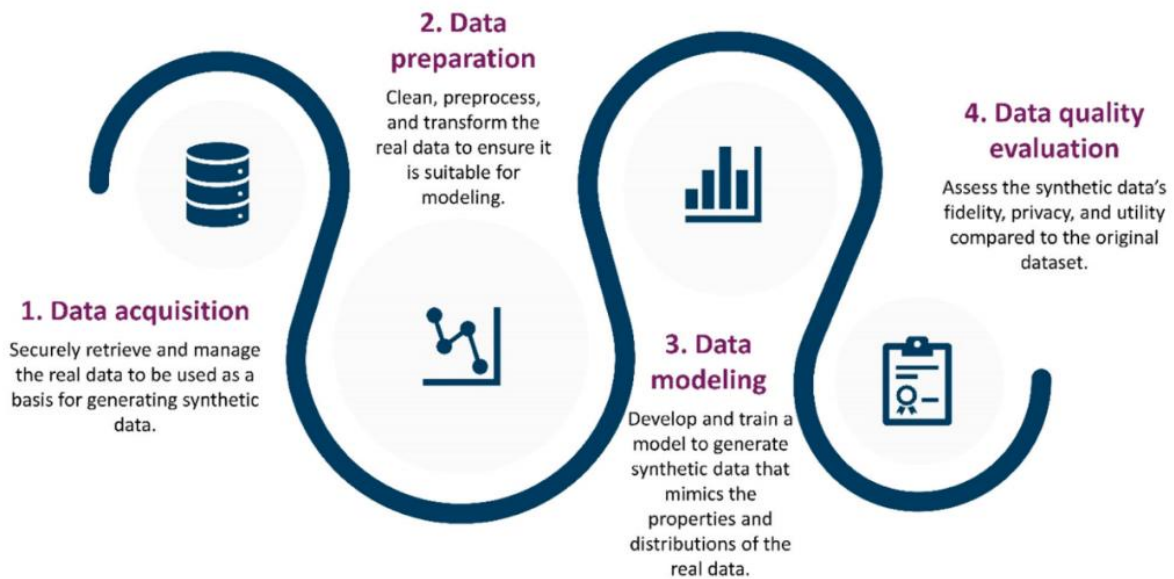


Figura 3-3: Flujo de trabajo de la generación de datos sintéticos (tomada de [41])

Para la implementación del proceso de generación de datos sintéticos se van a emplear diferentes sintetizadores de SDV, siguiendo las etapas que se muestran en Figura 3-4.

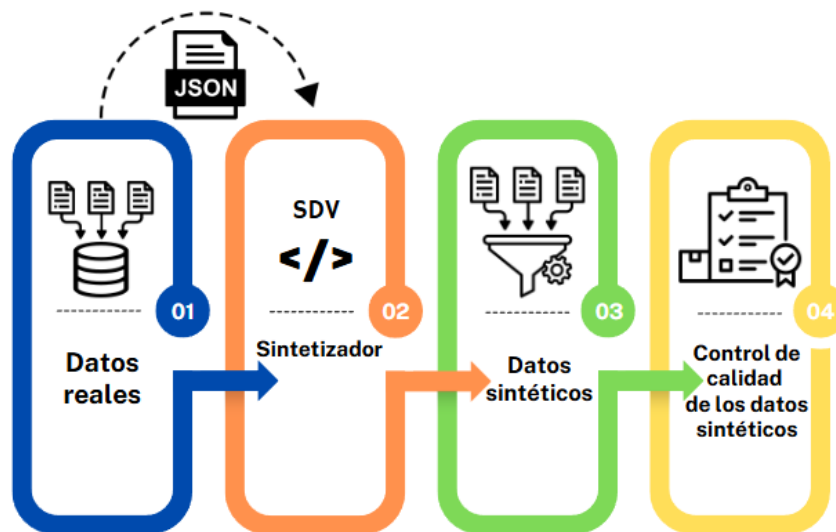


Figura 3-4: Etapas de la generación de datos sintéticos (fuente propia)

El punto de partida será el uso de los datos reales AIS a partir de los cuales se van a generar los nuevos datos sintéticos. Para generar dichos datos sintéticos es necesario utilizar metadatos. Los metadatos proporcionan la descripción del conjunto de datos originales almacenados en un *dataframe* (*df*) que es un *dataset* que está especialmente organizado en columnas, utilizado comúnmente en análisis de datos y programación. Es decir, los metadatos son una caracterización del contenido. En este caso, al tratarse de datos tabulares, describe el tipo de dato que contiene cada columna. Se recomienda guardar los metadatos en un archivo JSON para su futura reproducibilidad [60].

Una vez recopilados los datos reales junto con sus metadatos correspondientes, se procede a la creación y entrenamiento del sintetizador. El sintetizador es un programa diseñado para generar datos ficticios mediante técnicas de modelado probabilístico y generativo. Utiliza algoritmos que analizan la estructura y distribución de los datos originales, usados como datos de entrenamiento, para aprender sus

propiedades. Con los datos de entrenamiento, el sintetizador ajusta sus parámetros internos y, tras el proceso de entrenamiento, el sintetizador podrá generar datos sintéticos.

Como paso final, en el capítulo 4 se comprobará la calidad de los datos generados y se evaluará si mantienen las propiedades del conjunto de datos reales.

3.3 Análisis de los datos disponibles

Las bases de datos empleadas para este trabajo han sido generadas a partir de mensajes AIS reales transmitidos por los buques y recogidos por el COVAM durante los meses de enero y febrero de 2023. Existen diferentes tipos de mensajes AIS transmitidos por un buque, pero para este trabajo son de interés los mensajes AIS de tipo 5 que recogen los datos dimensionales de la plataforma. Para este trabajo, la generación de datos sintéticos se centra en los datos AIS estáticos y no en los dinámicos dado que son los más relevantes para los sistemas de predicción de tipo de buque, y dado que son menos fiables y de menor calidad en comparación con los dinámicos según los estudios [27] y [28] mencionados en el apartado 2.4.3 de la memoria.

Los sistemas AIS envían la información codificada mediante texto plano que se procesan con una librería específica: *pyais* [61], pero como ya se ha mencionado antes, estos datos han sido depurados de forma que sean adecuados para su uso en la generación de datos sintéticos. El procesado de los datos ha sido objeto de estudio en anteriores TFGs [3] y [4].

Para los datos estáticos, que caracterizan las dimensiones de la plataforma transmisora, se dispone de dos archivos CSV: *final_static_data.csv* y *final_static_data_basic.csv*. Los campos disponibles y comunes para ambos archivos son los siguientes:

1. **MMSI**: número de identificación del servicio móvil marítimo, es una serie de 9 dígitos que identifica inequívocamente a cada estación transmisora.
2. **To_bow**: es la distancia desde el transpondedor AIS hasta la proa en metros y se corresponde con el campo A de los mensajes AIS.
3. **To_stern**: es la distancia desde el transpondedor AIS hasta la popa en metros y se corresponde con el campo B de los mensajes AIS.
4. **To_port**: es la distancia desde el transpondedor AIS hasta la banda de babor en metros y se corresponde con el campo C de un mensaje AIS.
5. **To_starboard**: es la distancia desde el transpondedor AIS hasta la banda de estribor en metros y se corresponde con el campo D de un mensaje AIS.
6. **Draught**: es el calado del buque en metros.
7. **Shiptype**: indica el tipo de buque escrito explícitamente (*tug*, *passenger*, *cargo*, *fishing* y *tanker*)

En la Figura 3-5 se aporta una representación gráfica de los campos A, B, C y D de los mensajes AIS.

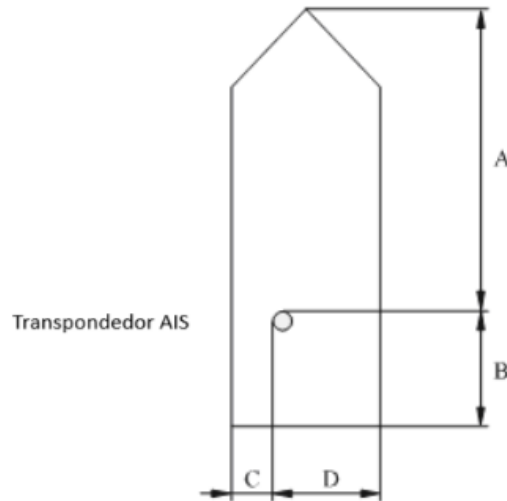


Figura 3-5: Representación de los campos A, B, C y D [62]

A partir de los datos estáticos del archivo *final_static_data_basic.csv*, anteriores TFGs [3] y [4] han propuesto la definición de los atributos de la Tabla 3-1, para mejorar la predicción de tipo de buque mediante técnicas de IA. Estos datos se almacenan en el archivo CSV denominado *final_static_data.csv*.

Atributo	Descripción	Fórmula
<i>len</i>	Eslora total	$A + B$
<i>wid</i>	Manga total	$C + D$
<i>ldivw</i>	Relación entre eslora y manga	len / wid
<i>ldivd</i>	Relación entre eslora y calado	$len / draught$
<i>wdivd</i>	Relación entre manga y calado	$wid / draught$
<i>area</i>	Área aproximada de la planta	$len \times wid$
<i>grith</i>	Suma de la eslora y la manga	$len + wid$
<i>aml</i>	Área longitudinal sumergida	$len \times draught$
<i>amt</i>	Área transversal sumergida	$wid \times draught$
<i>vs</i>	Volumen sumergido	$len \times wid \times draught$
<i>aol</i>	Tanto por uno que supone la distancia A sobre la eslora total del buque. Resultante de dividir A entre la eslora.	A / len

Tabla 3-1: Atributos estáticos del archivo *final_static_data.csv*

Lo primero que realizamos es la carga de los datos contenidos en los archivos y ejecutando el comando *DataFrame.info* obtenemos información sobre el contenido de los *dataframe* de cada archivo.

```
# Cargar el dataset completo
df = pd.read_csv("final_static_data.csv")
```

```
# Mostrar nombres de columnas y tipos de datos
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45367 entries, 0 to 45366
Data columns (total 18 columns):
#   Column          Non-Null Count  Dtype
---  -
0   MMSI             45367 non-null  int64
1   draught          45367 non-null  float64
2   to_bow           45367 non-null  float64
3   to_stern         45367 non-null  float64
4   to_port          45367 non-null  float64
5   to_starboard     45367 non-null  float64
6   len              45367 non-null  float64
7   wid              45367 non-null  float64
8   ldivw            45367 non-null  float64
9   ldivd            45367 non-null  float64
10  wdivd            45367 non-null  float64
11  area             45367 non-null  float64
12  grith            45367 non-null  float64
13  aml              45367 non-null  float64
14  amt              45367 non-null  float64
15  vs               45367 non-null  float64
16  aol              45367 non-null  float64
17  shiptype         45367 non-null  object
dtypes: float64(16), int64(1), object(1)
memory usage: 6.2+ MB
None
```

Figura 3-6: Información de *final_static_data.csv*

```
#Ver nºcolumnas, tipos de datos y valores nulos
print(df_real_limpio.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 45367 entries, 0 to 45366
Data columns (total 7 columns):
#   Column          Non-Null Count  Dtype
---  -
0   MMSI             45367 non-null  int64
1   draught          45367 non-null  float64
2   to_bow           45367 non-null  float64
3   to_stern         45367 non-null  float64
4   to_port          45367 non-null  float64
5   to_starboard     45367 non-null  float64
6   shiptype         45367 non-null  object
dtypes: float64(5), int64(1), object(1)
memory usage: 2.4+ MB
None
```

Figura 3-7: Información de *final_static_data_basic.csv*

Como muestra la Figura 3-6, el *dataframe* de *final_static_data.csv* contiene 45367 filas, 18 columnas, ningún valor nulo y ocupa 6,2 MB. En cambio, la Figura 3-7 muestra que el *dataframe* del archivo *final_static_data_basic.csv* también contiene 45367 filas, pero 7 columnas y ocupa 2,4 MB. En ambas se especifican los tipos de datos por columna siendo la columna MMSI de tipo *int* (número entero), la columna *shiptype* de tipo *object* (dato categórico), y el resto de tipo *float* (valores decimales).

La Figura 3-8 muestra el número de barcos por cada categoría y la Figura 3-9 muestra un diagrama de sectores en el que se representa la proporción de las categorías del campo *shiptype*.

```
#Distribucion de los tipos de barcos
print(df_real["shiptype"].value_counts())
```

```
shiptype
ShipType.Cargo      22097
ShipType.Tanker     8986
ShipType.Tug        5821
ShipType.Fishing    5551
ShipType.Passenger  2912
Name: count, dtype: int64
```

Figura 3-8: Cantidad de barcos por categoría

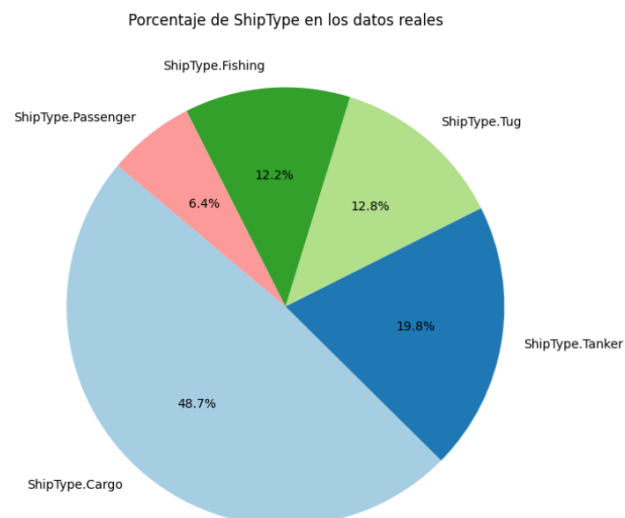


Figura 3-9: Proporción del campo *shiptype*

El *shiptype Cargo* (cargueros) es el más abundante de la muestra con 22097 barcos representando el 48,7% del total. El *shiptype Tanker* (petroleros) representa el 19,8% del total. La proporción de los *shiptypes Fishing* (pesqueros) y *Tug* (remolcadores) es parecida con un 12,2% y 12,8% del total, respectivamente, siendo el *shiptype Passenger* (cruceros) el menos abundante, con un 6,4% del total de datos.

3.4 Generación de datos estáticos sintéticos

En este apartado se pretende describir el proceso de generación de datos sintéticos a partir del conjunto de datos AIS estáticos. Se emplean tres técnicas diferentes (TVAE, *Gaussian Copula* y CTGAN) con dos enfoques principales o estrategias de entrenamiento: generación sin distinción de tipo de barco y generación específica por *shiptype*. Además, se emplean ambos archivos CSV para evaluar el impacto de un mayor número de parámetros originales en la calidad de los datos generados.

3.4.1 Sin agrupar por tipo de buque

En este enfoque, los datos sintéticos se generan sin diferenciar el tipo de buque. Es decir, el sintetizador se entrena con todos los mensajes, independientemente del tipo de buque, del archivo *final_static_data_basic.csv*. Esta prueba nos permitirá ver la capacidad de cada técnica para capturar la distribución global de los datos AIS reales. Se generan tres *scripts* por cada técnica que se pueden consultar en el Anexo IV: Código para los datos sin agrupar por *shiptype*.

Este enfoque permitió evaluar la capacidad de los modelos para capturar patrones generales en el conjunto de datos completo. Sin embargo, como se expondrá más adelante, esta estrategia encontró ciertos problemas debido a las diferencias significativas en los patrones y correlaciones de parámetros entre los distintos *shiptypes*.

Para esta primera prueba se configura el número de datos generados *num_datos_sinteticos* = 50000. Este ajuste se realiza con el objetivo de mantener una proporción cercana a la del conjunto original que consta de 45367 datos. El archivo *final_static_data_basic.csv* contiene una columna de numeración que se elimina para evitar su generación sintética. Para ello se le indican al modelo las columnas que se quieren generar: *columnas_estaticas*=["MMSI", "draught", "to_bow", "to_stern", "to_port", "to_starboard", "shiptype"], y se emplean los diferentes sintetizadores de la librería de SDV para la generación de los datos sintéticos.

Los datos generados por cada técnica se guardaron en archivos CSV con el nombre *sinteticos_estaticos_TECNICA_sin_distincion.csv* y también los metadatos asociados en archivos JSON con el nombre *metadata_TECNICA_sin_distincion.json*.

En la Figura 3-10, Figura 3-11 y Figura 3-12 se muestran las 5 primeras filas de los diferentes archivos sintéticos generados a partir del archivo *final_static_data_basic.csv*, entrenando al algoritmo sin diferenciación por tipo de buque.

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	5015007	11.890596	202.704251	43.645463	15.718198	22.368696	
1	8369770	1.719441	4.294888	6.524507	4.742052	2.597673	
2	13721166	4.320384	74.277407	21.502601	6.754399	9.189718	
3	9027652	6.835948	122.519316	15.916724	13.224334	8.833088	
4	9301115	2.083593	0.906677	7.435651	6.933366	1.731611	
shiptype							
0	ShipType.Tanker						
1	ShipType.Fishing						
2	ShipType.Cargo						
3	ShipType.Cargo						
4	ShipType.Fishing						

Figura 3-10: 5 primeras filas de *sinteticos_estaticos_TVAE_sin_distincion.csv*

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	5015007	7.802587	124.926580	65.816147	17.495746	18.214170	
1	8369770	6.373577	48.037359	21.694339	11.318186	12.406191	
2	13721166	8.545818	201.022579	28.363901	15.317246	15.989583	
3	9027652	9.461572	98.870790	15.846482	11.827223	8.793048	
4	9301115	14.307206	303.163356	54.195889	25.601123	32.087010	
shiptype							
0	ShipType.Cargo						
1	ShipType.Fishing						
2	ShipType.Cargo						
3	ShipType.Cargo						
4	ShipType.Cargo						

Figura 3-11: 5 primeras filas de *sinteticos_estaticos_GaussianCopula_sin_distincion.csv*

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	5015007	2.862068	14.883501	13.709884	3.175171	4.127156	
1	8369770	3.071866	19.723704	9.216781	1.518547	5.946649	
2	13721166	6.787577	199.630571	81.637254	19.489475	12.757548	
3	9027652	3.399968	9.657893	10.227369	4.358086	7.957244	
4	9301115	1.755785	37.370423	22.795802	3.483896	5.450436	
shiptype							
0	ShipType.Passenger						
1	ShipType.Fishing						
2	ShipType.Cargo						
3	ShipType.Tug						
4	ShipType.Passenger						

Figura 3-12: 5 primeras filas de *sinteticos_estaticos_CTGAN_sin_distincion.csv*

3.4.2 Con agrupación por tipo de buque

Dado que las diferentes categorías de buque presentan características estructurales distintas, se opta por filtrar los datos en función del campo *shiptype*. Se entrena cada modelo de forma individual para cada *shiptype* con el objetivo de permitir un aprendizaje más específico de los patrones y estructuras de cada categoría, para así conservar las relaciones internas de cada clase. La Figura 3-13 muestra un esquema del entrenamiento individual para cada tipo de barco.

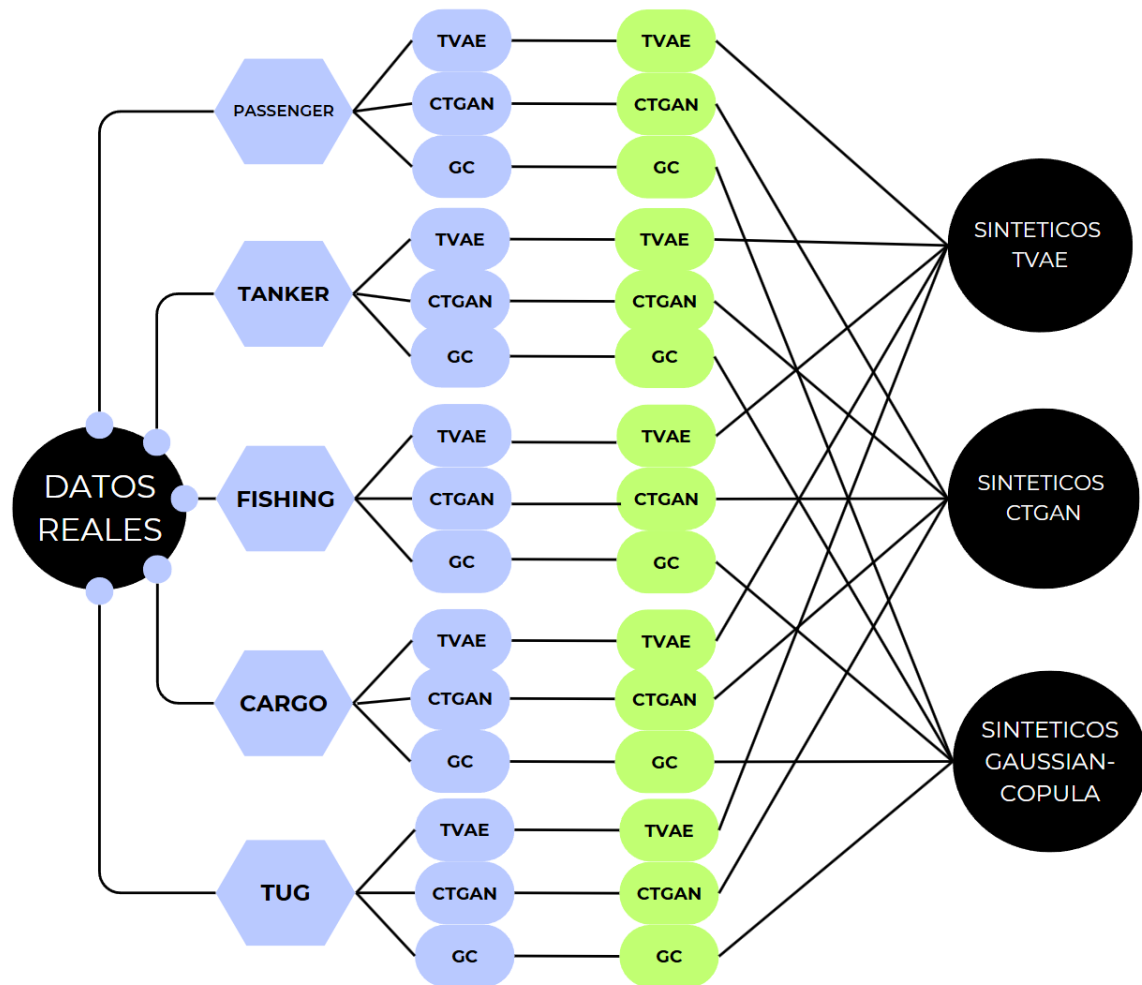


Figura 3-13: Esquema de entrenamiento por *shiptype* (fuente propia)

Cada conjunto de datos reales se divide en las cinco categorías de buques, se entrenan los tres modelos para generar datos sintéticos por cada *shiptype* y se reagrupan en un conjunto de datos completo para su posterior evaluación y comparación con los archivos originales.

El objetivo principal es evaluar la coherencia de los atributos generados a partir de los datos sintéticos mediante tres configuraciones distintas de los datos originales:

- Para la primera configuración, se generan datos sintéticos del archivo *final_static_data_basic.csv* y posteriormente se utilizan las variables sintéticas A, B, C, D y *draught* para calcular los atributos derivados mediante las fórmulas específicas presentadas en la Tabla 3-1.
- En la segunda configuración, se emplea el archivo *final_static_data.csv*, que incluye los atributos derivados por lo que se generan directamente de forma sintética, en lugar de calcularse a posteriori a partir de las fórmulas.
- Y el último planteamiento es la generación sintética de los atributos a partir del archivo *final_static_data.csv* pero excluyendo los campos A, B, C, D y *draught* de los que dependen. Este planteamiento se realiza debido a que estos atributos son los que después tienen más relevancia en las predicciones de tipo de buque como veremos en el capítulo 4.

Para esta prueba se generan tres *scripts* por cada archivo de datos reales. Se puede consultar en el Anexo V: Código para los datos agrupados por *shiptype*.

Al igual que en la generación sin agrupar por tipo de buque, se establece el número de datos generados: $num_total_synthetic = 50000$ en las tres configuraciones. Además, para garantizar una

representación fiel de los datos reales, se mantiene la proporción original de cada *shiptype*. Para ello, se ajusta mediante $num_samples = \text{int}(num_total_synthetic * proportion)$. Esta condición contribuye a una mayor coherencia en el análisis de calidad posterior.

Para la primera configuración, como se usa el archivo *final_static_data_basic.csv*, se definen las columnas: *MMSI*, *draught*, *to_bow*, *to_stern*, *to_port*, *to_starboard*, *shiptype* para evitar generar la columna extra que corresponde a la indexación en el archivo original. Para la segunda configuración se emplean todos los campos pues el archivo *final_static_data.csv* no contiene la columna de indexación. Y para la última configuración se definen las columnas a generar del archivo completo que se corresponden con los campos: *len*, *wid*, *ldivw*, *ldivd*, *wdivd*, *area*, *grith*, *aml*, *amt*, *vs*, *aol*, *shiptype*.

Dado que los datos se generan por lotes separados, como se explica en la Figura 3-13, se aplica una fase de mezcla aleatoria para evitar que en los archivos finales se guarden en bloques por *shiptype*. Es decir, como se generan los datos separados por tipo de buque, cada conjunto de datos sintéticos se genera de forma secuencial y se guardarían en bloques consecutivos si no se implementa antes una mezcla aleatoria utilizando: $sample(frac=1, random_state=42)$. El parámetro *random_state=42* asegura que la mezcla sea siempre la misma cada vez que se ejecute el código. Además, el almacenamiento de datos sin mezclar podría introducir patrones no deseados en los análisis posteriores. Algunos análisis como PCA o t-SNE detectarían patrones artificiales simplemente porque los datos están en bloques en lugar de mezclados. Al mezclar, se garantiza que las distribuciones sean más representativas en cualquier muestreo aleatorio y simula como estarían distribuidos en un *dataset* real.

Los datos generados por cada técnica y para cada conjunto de datos se guardaron en archivos CSV de la siguiente manera:

- *synthetic_static_basic_TECNICA.csv* para la primera configuración.
- *synthetic_static_completo_TECNICA.csv* para la segunda configuración.
- *synthetic_static_atributos_TECNICA.csv* para la última configuración.

También se guardaron los metadatos asociados en archivos JSON por cada técnica y para cada conjunto de datos empleado:

- *metadata_basic_shiptype_TECNICA.json*
- *metadata_completo_shiptype_TECNICA.json*
- *metadata_atributos_shiptype_TECNICA.json*

A modo de ejemplo, la Figura 3-14 muestra el contenido del archivo JSON *metadata_basic_shiptype_TVAE.json*.

```

root
├── tables
│   └── AIS_estaticos_TVAE
│       ├── columns
│       │   ├── MMSI
│       │   │   └── sdtype "id"
│       │   ├── draught
│       │   │   └── sdtype "numerical"
│       │   ├── to_bow
│       │   │   └── sdtype "numerical"
│       │   ├── to_stern
│       │   │   └── sdtype "numerical"
│       │   ├── to_port
│       │   │   └── sdtype "numerical"
│       │   ├── to_starboard
│       │   │   └── sdtype "numerical"
│       │   └── shiptype
│       │       └── sdtype "categorical"
│       └── primary_key "MMSI"
├── relationships 0 items
└── METADATA_SPEC_VERSION "V1"

```

Figura 3-14: Archivo *metadata_basic_shiptype_TVAE.json*

En la Figura 3-15, Figura 3-16 y Figura 3-17 se muestran las 5 primeras filas de los diferentes archivos sintéticos generados a partir del archivo *final_static_data_basic.csv* que se corresponden con la primera configuración.

Primeras 5 filas de TVAE:

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	14374624	9.819297	146.485939	34.891283	20.392690	10.544324	
1	15466187	5.290521	31.112901	59.467735	8.766642	12.818422	
2	14981267	9.243166	196.997671	31.122374	21.136936	11.278718	
3	8090997	12.842989	203.632909	27.649310	12.005997	22.157070	
4	5591053	4.181110	29.111132	28.783013	6.013622	2.019162	
shiptype							
0	ShipType.Tanker						
1	ShipType.Cargo						
2	ShipType.Cargo						
3	ShipType.Cargo						
4	ShipType.Fishing						

Figura 3-15: 5 primeras filas de *synthetic_static_basic_TVAE.csv*

Primeras 5 filas de GaussianCopula:

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	14374624	13.098885	279.648748	49.782706	35.394045	25.544530	
1	15466187	14.497094	256.266989	40.811324	29.478509	17.362116	
2	14981267	8.713863	135.246131	59.484439	13.153698	11.664804	
3	8090997	2.878784	131.404949	12.199003	4.455921	10.133496	
4	5591053	7.052607	18.299146	45.141388	10.781726	4.190420	
shiptype							
0	ShipType.Tanker						
1	ShipType.Cargo						
2	ShipType.Cargo						
3	ShipType.Cargo						
4	ShipType.Fishing						

Figura 3-16: 5 primeras filas de *synthetic_static_basic_GaussianCopula.csv*

Primeras 5 filas de CTGAN:

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	14374624	12.541259	185.632102	49.274023	19.895917	23.471877	
1	15466187	8.702465	15.533554	83.341413	30.033746	18.527647	
2	14981267	5.301816	17.380511	69.464140	10.168411	10.490118	
3	8090997	7.378139	92.360891	17.094739	5.050334	10.452327	
4	5591053	3.718687	9.174241	20.207645	1.921807	4.971802	
shiptype							
0	ShipType.Tanker						
1	ShipType.Cargo						
2	ShipType.Cargo						
3	ShipType.Cargo						
4	ShipType.Fishing						

Figura 3-17: 5 primeras filas de *synthetic_static_basic_CTGAN.csv*

En la Figura 3-18, Figura 3-19 y Figura 3-20 se muestran las 5 primeras filas de los diferentes archivos sintéticos generados a partir del archivo *final_static_data.csv* que se corresponde con la segunda configuración.

Primeras 5 filas de TVAE:

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	14374624	8.569754	146.556322	34.386109	21.220992	6.468825	
1	15466187	8.432990	169.514328	18.014104	22.999470	11.931708	
2	14981267	8.308049	119.097571	21.373788	10.128303	9.871043	
3	8090997	8.696925	32.868165	179.107409	20.016717	29.094309	
4	5591053	2.624793	12.923174	13.654256	3.025459	2.992804	
len							
0	182.010520	26.601793	5.776739	21.952104	3.539978	5224.851158	
1	191.565153	32.030762	6.151720	25.906746	3.933445	6375.747682	
2	172.354537	21.779436	6.571282	19.218253	3.137491	3583.483849	
3	122.459446	45.576002	3.515002	18.909471	3.459737	13473.865672	
4	27.260042	6.994532	4.232264	11.290253	2.753880	183.011771	
wid							
0	212.114106	1795.908052	319.844421	48502.057679	0.812159		
1	208.865711	1601.612861	258.957766	40483.210907	0.855368		
2	178.019680	1342.883292	135.461902	35599.255651	0.862297		
3	340.349562	3475.742702	531.856202	50880.446372	0.060628		
4	39.259670	91.080741	17.285732	339.604866	0.497846		
ldivw							
0	212.114106	1795.908052	319.844421	48502.057679	0.812159		
1	208.865711	1601.612861	258.957766	40483.210907	0.855368		
2	178.019680	1342.883292	135.461902	35599.255651	0.862297		
3	340.349562	3475.742702	531.856202	50880.446372	0.060628		
4	39.259670	91.080741	17.285732	339.604866	0.497846		
ldivd							
0	212.114106	1795.908052	319.844421	48502.057679	0.812159		
1	208.865711	1601.612861	258.957766	40483.210907	0.855368		
2	178.019680	1342.883292	135.461902	35599.255651	0.862297		
3	340.349562	3475.742702	531.856202	50880.446372	0.060628		
4	39.259670	91.080741	17.285732	339.604866	0.497846		
wdivd							
0	212.114106	1795.908052	319.844421	48502.057679	0.812159		
1	208.865711	1601.612861	258.957766	40483.210907	0.855368		
2	178.019680	1342.883292	135.461902	35599.255651	0.862297		
3	340.349562	3475.742702	531.856202	50880.446372	0.060628		
4	39.259670	91.080741	17.285732	339.604866	0.497846		
area							
0	212.114106	1795.908052	319.844421	48502.057679	0.812159		
1	208.865711	1601.612861	258.957766	40483.210907	0.855368		
2	178.019680	1342.883292	135.461902	35599.255651	0.862297		
3	340.349562	3475.742702	531.856202	50880.446372	0.060628		
4	39.259670	91.080741	17.285732	339.604866	0.497846		
shiptype							
0	ShipType.Tanker						
1	ShipType.Cargo						
2	ShipType.Cargo						
3	ShipType.Cargo						
4	ShipType.Fishing						

Figura 3-18: 5 primeras filas de *synthetic_static_completo_TVAE.csv*

Primeras 5 filas de GaussianCopula:

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	14374624	8.323372	123.230873	13.237009	9.320063	9.281483	
1	15466187	6.334969	36.754783	17.853757	0.083741	16.464459	
2	14981267	6.404947	149.464695	8.711072	13.491291	14.884409	
3	8090997	6.057472	123.662053	14.582472	11.085755	7.550511	
4	5591053	3.293867	10.105069	9.161949	2.783526	4.182690	
	len	wid	ldivw	ldivd	wdivd	area	\
0	120.265894	28.705430	5.143684	11.432138	2.411103	2740.943709	
1	64.548642	9.891399	6.926115	12.228812	1.723206	573.032915	
2	128.121164	30.038871	4.757444	22.032309	4.643087	3823.987179	
3	139.949351	18.350921	7.538707	27.089471	3.662188	2657.313089	
4	25.598809	7.033642	3.383315	7.296987	2.184267	248.564001	
	grith	aml	amt	vs	aol	\	
0	142.614727	906.217484	161.311640	19058.603327	0.930315		
1	73.991449	443.853987	66.794726	4018.129341	0.748037		
2	157.403559	820.001006	178.239758	23548.657775	0.959586		
3	159.463249	816.308392	104.913130	15737.562122	0.932111		
4	34.283234	79.510186	23.133164	677.125395	0.543017		
shiptype							
0	ShipType.Tanker						
1	ShipType.Cargo						
2	ShipType.Cargo						
3	ShipType.Cargo						
4	ShipType.Fishing						

```

Primeras 5 filas de CTGAN:
  MMSI draught to_bow to_stern to_port to_starboard \
0 14374624 9.877730 82.380764 41.885084 8.234937 11.333452
1 15466187 4.782890 113.411360 17.852154 17.167420 10.060951
2 14981267 9.301248 23.005537 65.264670 28.347441 16.432693
3 8090997 5.685180 56.676242 12.887145 5.621105 9.392071
4 5591053 5.453624 18.915574 24.190204 5.109782 5.955321

  len wid ldivw ldivd wdivd area \
0 76.559450 42.688777 4.117667 16.935879 3.922122 1169.991980
1 126.203614 16.245181 6.273664 18.297737 2.824493 1475.994510
2 243.503934 37.112302 3.150386 19.470356 4.276807 12224.130565
3 83.380632 16.783087 6.431012 16.971195 2.461767 493.377071
4 58.682351 7.972129 4.335290 8.363732 1.606269 357.293634

  grith aml amt vs aol \
0 76.761290 596.720737 43.800530 5062.023369 0.826770
1 83.304350 514.932656 81.677086 9747.715407 0.877502
2 306.208127 1996.848531 390.410505 71478.441733 0.177913
3 56.072993 372.734349 52.279147 8575.738665 0.787077
4 59.896728 299.532183 84.900379 892.011072 0.263952

  shiptype
0 ShipType.Tanker
1 ShipType.Cargo
2 ShipType.Cargo
3 ShipType.Cargo
4 ShipType.Fishing

```

Figura 3-20: 5 primeras filas de *synthetic_static_completo_CTGAN.csv*

En la Figura 3-21, Figura 3-22 y Figura 3-23 se muestran las 5 primeras filas de los diferentes archivos sintéticos generados a partir del archivo *final_static_data.csv* empleando únicamente los atributos.

```

Primeras 5 filas de TVAE:
  len wid ldivw ldivd wdivd area \
0 184.369248 31.901932 5.708980 26.545064 4.341453 5839.641208
1 186.299427 21.903074 6.395197 19.264728 3.241160 5547.249995
2 91.862681 19.553908 6.528522 26.952664 3.698073 2439.612387
3 116.425295 17.006886 5.825994 62.839318 4.885065 1650.902188
4 51.302549 8.988010 4.908818 8.581238 1.644091 564.196463

  grith aml amt vs aol \
0 216.131214 1580.444576 233.177671 58607.596309 0.833252
1 212.697907 1876.043134 303.790820 58407.585364 0.825908
2 116.986950 481.717390 37.221232 17036.902798 0.724023
3 104.631568 245.259632 56.521965 6317.730608 0.873632
4 66.789484 279.469260 60.277598 3315.358302 0.664914

  shiptype
0 ShipType.Tanker
1 ShipType.Cargo
2 ShipType.Cargo
3 ShipType.Cargo
4 ShipType.Fishing

```

Figura 3-21: 5 primeras filas de *synthetic_static_atributos_TVAE.csv*

```

Primeras 5 filas de GaussianCopula:
  len wid ldivw ldivd wdivd area \
0 149.585997 26.755388 6.231059 28.460950 4.354965 3771.745246
1 267.695037 37.991918 7.288719 16.314923 2.222991 9525.834628
2 251.397815 37.570584 7.014823 28.980800 4.003405 8941.000593
3 91.633212 14.984631 6.173858 23.954687 3.937052 1329.894308
4 90.492456 16.274030 5.133443 12.950288 2.686839 1387.551286

  grith aml amt vs aol \
0 174.655683 881.498147 148.056826 22470.846901 0.977917
1 304.361541 3264.414871 477.603665 112430.150188 0.774822
2 288.836667 2242.806011 324.400187 77500.082025 0.464754
3 104.916042 323.374587 54.367080 4819.871928 0.498296
4 103.225287 524.994674 93.541770 9076.334909 0.138905

  shiptype
0 ShipType.Tanker
1 ShipType.Cargo
2 ShipType.Cargo
3 ShipType.Cargo
4 ShipType.Fishing

```

Figura 3-22: 5 primeras filas de *synthetic_static_atributos_GaussianCopula.csv*

```

Primeras 5 filas de CTGAN:
  len wid ldivw ldivd wdivd area \
0 248.398294 42.475486 5.791456 34.749881 4.956806 9841.058697
1 80.360048 19.034869 5.955441 12.846975 2.707765 1962.057353
2 210.028150 32.022608 6.246398 21.276431 3.280331 6503.270237
3 150.468842 13.252447 5.691815 19.912264 2.724273 1364.626316
4 13.699641 5.947576 1.657050 13.121234 3.367496 62.057448

  grith aml amt vs aol \
0 291.585546 2911.252812 406.669345 115069.298549 0.832049
1 105.688955 361.508760 80.425051 16149.200791 0.860460
2 237.410034 1678.363589 298.258705 85968.918535 0.852022
3 131.248118 571.030718 122.409202 14230.165688 0.865701
4 14.340300 41.962096 8.639284 328.800888 0.717269

  shiptype
0 ShipType.Tanker
1 ShipType.Cargo
2 ShipType.Cargo
3 ShipType.Cargo
4 ShipType.Fishing

```

Figura 3-23: 5 primeras filas de *synthetic_static_atributos_CTGAN.csv*

Observando los resultados de la segunda configuración se aprecia que hay valores de *len* que no se corresponden con la suma de *to_bow* y *to_stern* como, por ejemplo, en la fila 2 de la Figura 3-20 que se corresponde con CTGAN. El valor de *len* generado es de 243,50 metros, mientras que la suma de *to_bow* y *to_stern* es 88,26 metros. Igual ocurre en los datos generados para la tercera configuración. Por ejemplo, el atributo *ldiw* en la fila 4 de la Figura 3-23 no se corresponde con la división de *len* entre *wid*, pues el valor de *ldiw* generado es de 1,657 cuando debería ser 2,30, resultado de la división de *len* = 13,699 metros entre *wid* = 5,947 metros.

Se realiza una prueba sencilla para comprobar si las técnicas son capaces de aprender relaciones matemáticas entre columnas sin que se les indique de forma explícita. Se crea un *dataframe* de 20 filas y tres columnas donde la tercera es la suma de las dos primeras, y se guarda en *prueba.csv* para generar a partir de dicho archivo datos sintéticos mediante TVAE, *Gaussian Copula* y CTGAN. La Figura 3-24 muestra el *dataframe* original y la Figura 3-25 muestra los resultados obtenidos con las tres técnicas. Se comprueba que ninguna de las tres es capaz de generar una tercera columna en la que todos los valores se correspondan a la suma de las dos primeras.

				TVAE			GAUSSIAN COPULA		
				Columna1	Columna2	Columna3	Columna1	Columna2	Columna3
0	1	5	6	0	1	4	0	3	8
1	1	3	4	1	4	1	1	1	8
2	3	2	5	2	3	6	2	1	12
3	4	0	4	3	5	2	3	8	8
4	5	1	6	4	1	2	4	1	3
5	7	2	9	5	4	2	5	7	10
6	3	4	7	6	4	5	6	2	5
7	5	4	9	7	6	5	7	3	9
8	2	6	8	8	6	5	8	2	9
9	1	10	11	9	4	1	9	1	7
10	3	3	6						
11	3	2	5						
12	4	7	11						
13	5	4	9						
14	6	9	15						
15	4	1	5						
16	9	7	16						
17	5	1	6						
18	3	3	6						
19	1	2	3						

				CTGAN		
				Columna1	Columna2	Columna3
0	7	4	13	0	7	4
1	3	4	9	1	3	4
2	5	1	9	2	5	1
3	3	1	9	3	3	1
4	4	5	13	4	4	5
5	3	4	7	5	3	4
6	3	4	10	6	3	4
7	8	0	11	7	8	0
8	6	9	16	8	6	9
9	4	2	5	9	4	2

Figura 3-24: Dataframe original del archivo prueba.csv

Figura 3-25: Resultados de la prueba de suma de columnas para las tres técnicas de generación de datos sintéticos

En base al resultado de esta prueba, en el capítulo 4 de la memoria no se analizarán con detalle los resultados de los datos generados en la segunda configuración, y se descarta el uso y análisis de los generados en la tercera configuración.

Por último, se procede a calcular los atributos *len*, *wid*, *ldiw*, *ldivd*, *wdivd*, *area*, *grith*, *aml*, *amt*, *vs* y *aol* a partir de cada archivo de datos sintéticos generados en la primera configuración. Para ello, se define la función que se muestra en la Figura 3-26 y se generan tres archivos CSV: *atributos_calculados_TVAE.csv*, *atributos_calculados_GaussianCopula.csv* y *atributos_calculados_CTGAN.csv* que contienen los datos sintéticos generados en la primera configuración con sus correspondientes atributos calculados.

```
# Definir la función para calcular atributos estáticos
def statics_processing(df):
    df['len'] = df.to_bow + df.to_stern # Eslora
    df['wid'] = df.to_port + df.to_starboard # Manga
    df['ldivw'] = df['len'] / df['wid'] # Cociente entre eslora y manga
    df['ldivd'] = df['len'] / df['draught'] # Cociente entre eslora y calado
    df['wdivd'] = df['wid'] / df['draught'] # Cociente entre manga y calado
    df['area'] = df.len * df.wid # Área de la cubierta aproximada
    df['grith'] = df.len + df.wid # Suma de eslora y manga
    df['aml'] = df.len * df.draught # Área longitudinal sumergida
    df['amt'] = df.wid * df.draught # Área transversal sumergida
    df['vs'] = df.len * df.draught * df.wid # Volumen sumergido
    df['aol'] = df['to_bow'] / df['len'] # Proporción que supone a sobre la eslora total
    return df
```

Figura 3-26: Funciones de cálculo de los atributos estáticos

La Figura 3-27, Figura 3-28 y Figura 3-29 muestran las 5 primeras filas de los atributos calculados a partir de cada archivo generado en la primera configuración.

Primeras filas de los atributos calculados a partir de los datos sintéticos TVAE:

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	14374624	9.819297	146.485939	34.891283	20.392690	10.544324	
1	15466187	5.290521	31.112901	59.467735	8.766642	12.818422	
2	14981267	9.243166	196.997671	31.122374	21.136936	11.278718	
3	8090997	12.842989	203.632909	27.649310	12.005997	22.157070	
4	5591053	4.181110	29.111132	28.783013	6.013622	2.019162	

	shiptype	len	wid	ldivw	ldivd	wdivd	\
0	ShipType.Tanker	181.377222	30.937014	5.862790	18.471508	3.150634	
1	ShipType.Cargo	90.580636	21.585063	4.196450	17.121308	4.079951	
2	ShipType.Cargo	228.120045	32.415654	7.037342	24.679861	3.506986	
3	ShipType.Cargo	231.282220	34.163067	6.769949	18.008442	2.600056	
4	ShipType.Fishing	57.894145	8.032784	7.207233	13.846598	1.921209	

	area	grith	aml	amt	vs	aol
0	5611.269565	212.314235	1780.996761	303.779715	55098.720849	0.807631
1	1955.188753	112.165699	479.218731	114.196224	10343.966609	0.343483
2	7394.660335	260.535698	2108.551351	299.623253	68350.069991	0.863570
3	7901.309945	265.445287	2970.354922	438.755880	101476.433913	0.880452
4	465.051135	65.926929	242.061763	33.585948	1944.429740	0.502834

Figura 3-27: Atributos de *synthetic_static_basic_TVAE.csv*

Primeras filas de los atributos calculados a partir de los datos sintéticos GaussianCopula:

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	14374624	13.098885	279.648748	49.782706	35.394045	25.544530	
1	15466187	14.497094	256.266989	40.811324	29.478509	17.362116	
2	14981267	8.713863	135.246131	59.484439	13.153698	11.664804	
3	8090997	2.878784	131.404949	12.199003	4.455921	10.133496	
4	5591053	7.052607	18.299146	45.141388	10.781726	4.190420	

	shiptype	len	wid	ldivw	ldivd	wdivd	\
0	ShipType.Tanker	329.431455	60.938574	5.405959	25.149579	4.652195	
1	ShipType.Cargo	297.078313	46.840624	6.342322	20.492266	3.231035	
2	ShipType.Cargo	194.730570	24.818502	7.846186	22.347214	2.848163	
3	ShipType.Cargo	143.603952	14.589417	9.843022	49.883550	5.067910	
4	ShipType.Fishing	63.440534	14.972146	4.237237	8.995331	2.122923	

	area	grith	aml	amt	vs	aol
0	20075.083190	390.370029	4315.184893	798.227406	262961.215487	0.848883
1	13915.333658	343.918937	4306.772248	679.052938	201731.901123	0.862624
2	4832.921037	219.549072	1696.855559	216.265032	42113.412974	0.694530
3	2095.097904	158.193368	413.404717	41.999775	6031.333720	0.915051
4	949.840916	78.412680	447.421174	105.592664	6698.854972	0.288446

Figura 3-28: Atributos de *synthetic_static_basic_GaussianCopula.csv*

Primeras filas de los atributos calculados a partir de los datos sintéticos CTGAN:

	MMSI	draught	to_bow	to_stern	to_port	to_starboard	\
0	14374624	12.541259	185.632102	49.274023	19.895917	23.471877	
1	15466187	8.702465	15.533554	83.341413	30.033746	18.527647	
2	14981267	5.301816	17.380511	69.464140	10.168411	10.490118	
3	8090997	7.378139	92.360891	17.094739	5.050334	10.452327	
4	5591053	3.718687	9.174241	20.207645	1.921807	4.971802	

	shiptype	len	wid	ldivw	ldivd	wdivd	\
0	ShipType.Tanker	234.906125	43.367794	5.416603	18.730665	3.458009	
1	ShipType.Cargo	98.874967	48.561393	2.036082	11.361720	5.580189	
2	ShipType.Cargo	86.844651	20.658528	4.203816	16.380170	3.896500	
3	ShipType.Cargo	109.455629	15.502661	7.060442	14.835128	2.101161	
4	ShipType.Fishing	29.381886	6.893609	4.262192	7.901145	1.853775	

	area	grith	aml	amt	vs	aol
0	10187.360395	278.273919	2946.018671	543.886755	127762.330133	0.790240
1	4801.506152	147.436360	860.455901	422.603807	41784.937381	0.157103
2	1794.082663	107.503179	460.434382	109.527720	9511.896632	0.200133
3	1696.853466	124.958290	807.578806	114.380778	12519.620074	0.843820
4	202.547229	36.275495	109.262034	25.635173	753.209734	0.312241

Figura 3-29: Atributos de *synthetic_static_basic_CTGAN.csv*

3.5 Data Augmentation

En este apartado se describe la aplicación de la técnica de *Data Augmentation* con el objetivo de mejorar la representación de la clase minoritaria *passenger* en el conjunto de datos reales. El propósito de esta prueba es determinar si la aplicación de *Data Augmentation* mejora las métricas de clasificación del modelo de aprendizaje automático.

Dado que *passenger* es la categoría menos representada en el conjunto de datos reales, se plantea aplicar una técnica que aumente la cantidad de ejemplos de esta clase sin alterar la distribución de las demás categorías. Como se explica en el estado del arte, existen diferentes enfoques de *Data Augmentation* para datos tabulares, pero en este trabajo se optará por SMOTE pues permite generar ejemplos sintéticos realistas sin modificar valores individuales y mantiene la estructura estadística de los datos originales, reduciendo el riesgo de generar valores fuera de rango.

Se aplica SMOTE a la categoría *passenger* del archivo real *final_static_data_basic.csv*, y se guardan los datos generados en un nuevo archivo *final_static_basic_augmented.csv* para su posterior análisis de utilidad en el capítulo 4. Se establece un aumento del 50% en el número de entradas de tipo *passenger* mediante $\text{num_passenger_augmented} = \text{int}(\text{num_passenger} * 1.5)$. El script de código generado se puede consultar en Anexo VIII: Código para *Data Augmentation*.

La Figura 3-30 muestra una comparación de la distribución de *shiptype* entre el archivo original y el archivo aumentado tras aplicar *Data Augmentation*. En total, se generan 1456 nuevos buques de la categoría *passenger*, lo que representa un incremento del 50% sobre el número original.

Distribución original de shiptype:		Distribución después de SMOTE:	
shiptype		shiptype	
ShipType.Cargo	22097	ShipType.Cargo	22097
ShipType.Tanker	8986	ShipType.Tanker	8986
ShipType.Tug	5821	ShipType.Tug	5821
ShipType.Fishing	5551	ShipType.Fishing	5551
ShipType.Passenger	2912	ShipType.Passenger	4368
Name: count, dtype: int64		Name: count, dtype: int64	

Figura 3-30: Comparación de distribuciones tras aplicar *Data Augmentation*

4 RESULTADOS OBTENIDOS Y VALIDACIÓN

En este capítulo se presentan el análisis de calidad y la validación de utilidad de los datos sintéticos obtenidos mediante las diferentes técnicas (TVAE, *Gaussian Copula* y CTGAN), comparándolos con los datos reales. El análisis se divide en dos enfoques principales: en primer lugar, se examinan los datos sintéticos estáticos generados sin agrupar por tipo de buque y, en segundo lugar, se examinan aquellos generados de manera específica para cada categoría de barco.

La evaluación de calidad se realiza a través de diversas métricas estadísticas y visualizaciones, que permiten identificar diferencias respecto a los datos reales. La validación de utilidad se lleva a cabo mediante el entrenamiento de un modelo de aprendizaje automático (*Random Forest*), para analizar el rendimiento en las tareas de clasificación de barcos en función de los diferentes conjuntos de datos sintéticos. También se incluye un análisis del impacto computacional de cada técnica y una prueba de *Data Augmentation* para mejorar la representación de la categoría minoritaria, en este caso *passenger*.

4.1 Métricas de calidad empleadas

Para la evaluación de las diferentes técnicas de generación de datos sintéticos, es crucial disponer de métricas precisas que permitan medir la efectividad de estos modelos a la hora de replicar la distribución de los datos reales, así como de visualizaciones que faciliten la comparación entre los datos reales y los datos generados. En esta sección, nos enfocaremos en las métricas y visualizaciones empleadas para evaluar la calidad de los datos sintéticos en función de las tres técnicas de generación.

Dos métricas muy utilizadas para la evaluación de calidad y precisión son la prueba de *Kolmogorov-Smirnov* (KS-test) y la prueba *Chi-cuadrado*. Para la visualización y comparación entre datos reales y sintéticos se emplearán gráficos de densidad, matrices de correlación, *Principal Components Analysis* (PCA) y *Distributed Stochastic Neighbor Embedded* (t-SNE). También se emplearán estadísticas descriptivas.

4.1.1 Kolmogorov-Smirnov (KS-Test)

La prueba de *Kolmogorov-Smirnov* es una herramienta estadística que compara dos muestras y determina si provienen de la misma distribución. Permite medir el grado de concordancia existente entre la distribución de un conjunto de datos y una distribución teórica específica [63]. Es útil en el contexto de la evaluación de la calidad de los datos, especialmente cuando queremos comparar la distribución de datos reales frente a datos sintéticos.

Mide la discrepancia entre la función de distribución empírica (EDF) de una muestra y la función de distribución acumulativa (CDF) de una distribución de referencia. La EDF es una función escalonada que muestra la proporción de observaciones que son menores o iguales a un valor dado. Y la CDF es una función continua que representa la probabilidad acumulada de que una variable aleatoria tome

valores menores o iguales a un punto específico [64]. El estadístico KS (*KS-statistics*) se calcula como la diferencia máxima entre estas dos funciones. Cuanto mayor sea el valor de *KS-statistics* (D), en el rango de 0 a 1, mayor es la discrepancia entre las distribuciones. Sin embargo, para determinar la significancia estadística de esta diferencia, se utiliza el valor p asociado al estadístico ' D '. Se formula una hipótesis nula que postula que las dos muestras provienen de la misma distribución, y una hipótesis alternativa que sostiene que las dos muestras provienen de distribuciones diferentes. La prueba devuelve un valor p que indica la probabilidad de observar una diferencia igual o mayor que la diferencia máxima entre las distribuciones, asumiendo que la hipótesis nula es cierta. Por lo que un valor p alto indica que la diferencia observada no es estadísticamente significativa, no hay suficientes evidencias para rechazar la hipótesis nula. Y un valor de p bajo (comúnmente menor de 0,05) indica que la diferencia observada es estadísticamente significativa por lo que se puede rechazar la hipótesis nula, pues existe una discrepancia entre las distribuciones [65]. En resumen, el valor p indica si esa diferencia es estadísticamente significativa o solo una variación aleatoria.

Una de las principales limitaciones del KS-test es su sensibilidad al tamaño de la muestra. Para muestras pequeñas, los resultados pueden no ser confiables, ya que la prueba puede no detectar diferencias reales entre distribuciones. Y para muestras grandes, incluso pequeñas discrepancias pueden ser estadísticamente significativas, lo que puede llevar al rechazo de la hipótesis nula por diferencias insignificantes [64].

En el presente trabajo, esta evaluación se ha llevado a cabo para comparar cada parámetro de los datos generados con su equivalente en los datos reales.

4.1.2 Chi-cuadrado

La prueba de *chi-cuadrado* es un test que sirve para evaluar los datos de tipo categórico. Se basa en la comparación de las frecuencias que se observan con las esperadas bajo la hipótesis nula. Permite realizar pruebas de bondad de ajuste, independencia y homogeneidad [66].

Matemáticamente, el estadístico de *chi-cuadrado* (X^2) se calcula como:

$$X^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

Ecuación 4-1: Estadístico de *chi-cuadrado*

Donde O_i son los valores observados en cada categoría como, por ejemplo, la cantidad de barcos de cada tipo en los datos sintéticos. Y E_i son los valores esperados según la distribución de datos reales. Cuanto mayor es X^2 , mayor es la diferencia entre los datos reales y sintéticos [67].

En este trabajo, *chi-cuadrado* se utiliza para evaluar si la distribución de tipos de barcos en los datos sintéticos refleja correctamente la distribución de los datos reales, es decir, se aplicó la bondad de ajuste. Se plantea como hipótesis nula que las distribuciones de los tipos de buques son iguales, y como hipótesis alternativa que las distribuciones son diferentes. El valor p indica la probabilidad de obtener un resultado igual o mayor que X^2 , asumiendo que la hipótesis nula es cierta. Por lo que si $p > 0,05$ no hay evidencias suficientes para rechazar la hipótesis nula y los datos reales y sintéticos tienen una distribución de valores categóricos similar.

4.1.3 Estadísticos descriptivos

Se emplean los estadísticos descriptivos como el número total de valores en las columnas (*count*), la media (*mean*), la desviación estándar (*std*), el valor mínimo (*min*), los cuartiles (25%, 50% y 75%) y el valor máximo (*max*) para comparar cada parámetro de los datos y evaluar si los datos sintéticos tienen características similares a los datos reales.

4.1.4 Gráficos de densidad

Los gráficos de estimación de densidad por núcleo (*Kernel Density Estimation*, KDE) permiten visualizar y comparar las distribuciones de variables numéricas de cada campo de los datos reales y sintéticos. Son útiles para detectar diferencias en dispersión y desviaciones, y para identificar sesgos en los datos sintéticos con respecto a los originales. Utilizan una estimación de la función de densidad de probabilidad de una variable para entender cómo se distribuyen los datos. Generalmente emplean un *kernel* gaussiano para suavizar la distribución [68].

4.1.5 Matriz de correlación

Las matrices de correlación analizan la relación entre variables en los datos reales y su preservación en los datos sintéticos. La matriz de correlación evalúa las relaciones entre variables en cada conjunto de datos. Se emplea el coeficiente de correlación de Pearson (para variables continuas), Spearman (si existen relaciones no lineales) y Kendall (para datos ordinales). El valor del coeficiente de correlación puede tomar valores entre -1 y 1. Un valor de correlación igual a 1 indica una correlación perfecta positiva, donde ambas variables aumentan. En cambio, un valor de correlación -1 indica una correlación perfecta negativa, al aumentar una variable, la otra disminuye. Y un valor de correlación igual a 0 indica que no hay correlación entre las variables [69]. Se compara la similitud entre las matrices de correlación de Pearson de los datos reales y sintéticos por cada *shiptype*.

4.1.6 PCA

El análisis de componentes principales (PCA) es una técnica de reducción de dimensionalidad en la cual se proyectan los datos en un espacio de menor dimensión y muestra las direcciones en las que los datos tienen la máxima variabilidad. Se utiliza para simplificar conjuntos de datos complejos con el objetivo de capturar la mayor variabilidad posible de los datos en menos dimensiones. Su objetivo es transformar un conjunto de variables correlacionadas en uno de variables no correlacionadas, conocidas como componentes principales. La primera componente principal (PC1) es la dirección en la que los puntos de datos muestran la mayor variabilidad. La segunda componente principal (PC2) refleja la segunda mayor variabilidad en el conjunto de datos y debe ser independiente de PC1. En términos geométricos, PC2 es perpendicular a PC1, lo que significa que no hay correlación entre ellas. Se utiliza un diagrama de dispersión para ilustrar la relación entre PC1 y PC2, donde ambos ejes son perpendiculares entre sí [70].

En este trabajo se aplica para evaluar la representatividad de los datos sintéticos en relación con los datos originales, proyectándolos en un espacio de dos dimensiones. La comparación de los datos reales y sintéticos en el espacio PCA proporciona información sobre la fidelidad con la que los modelos generativos han sido capaces de capturar la variabilidad del *dataset* original. Si los datos sintéticos se distribuyen de manera similar a los reales en la representación bidimensional obtenida con PCA, esto indica que los patrones subyacentes y las relaciones fundamentales entre las variables han sido correctamente aprendidos y replicados. Por el contrario, una distribución diferenciada puede señalar limitaciones en el proceso de generación, como la incapacidad de capturar ciertas dependencias o variaciones presentes en los datos originales.

4.1.7 t-SNE

El *t-Distributed Stochastic Neighbor Embedding* también es una técnica de reducción de dimensionalidad que agrupa datos similares en un espacio bidimensional. Permite evaluar si los *clusters* de los datos sintéticos reflejan las agrupaciones de los datos reales. A diferencia de PCA, que busca maximizar la varianza global y asume relaciones lineales, t-SNE preserva las relaciones locales no lineales de los datos reales [71]. Es útil para visualizar cómo los datos sintéticos se alinean con los datos reales en términos de agrupaciones y patrones. En la representación gráfica, el eje x se corresponde con t-SNE 1 que captura patrones de similitud en los datos, y el eje y se corresponde con t-SNE 2 que visualiza la relación entre grupos de datos.

Si los puntos sintéticos y reales están mezclados en la representación gráfica, entonces la generación ha sido precisa. En cambio, si hay separación entre los datos sintéticos y los reales, indica diferencias estructurales.

La Tabla 4-1 muestra un resumen del conjunto de métricas que se van a emplear para la evaluación de calidad de los datos sintéticos generados en las diferentes pruebas.

MÉTRICA	EVALUACIÓN DE	INTERPRETACIÓN
KS-Test	Similitud en la distribución de las variables	Valor p alto: no se rechaza que las distribuciones sean similares. Valor p bajo: existen diferencias significativas.
Chi-cuadrado	Balance de clases de barcos	Valor p alto indica una distribución de clases similar. Valor p bajo indica que los datos sintéticos no reflejan bien la composición original.
Estadísticos descriptivos	Tendencia central y dispersión	Si los diferentes estadísticos son similares en ambos conjuntos de datos, indica una buena aproximación. Diferencias sugieren sesgo o variabilidad incorrecta.
Gráficos de densidad (KDE)	Forma de la distribución de las variables	Si las curvas de KDE son similares, los datos generados replican la distribución real. Diferencias indican desajustes.
Matrices de correlación	Relación entre variables	Si las correlaciones son similares, las relaciones entre variables han sido preservadas en los datos sintéticos. Si el coeficiente de correlación es 1, entonces es perfecta y positiva. Si es 0, no existe relación entre las variables, Y si es -1, es perfecta negativa.
PCA	Preservación de la estructura global de los datos	Si los datos sintéticos se solapan con los reales en la visualización PCA, indica que se ha capturado bien la estructura de los datos.
t-SNE	Agrupamiento de datos en espacios de baja dimensión	Si los datos sintéticos y reales se agrupan de manera similar, indica que la estructura local ha sido bien replicada.

Tabla 4-1: Resumen de las métricas empleadas para la evaluación de calidad de los datos sintéticos (fuente propia)

4.2 Validación de utilidad

La validación de utilidad de los datos sintéticos es un paso fundamental para determinar si pueden ser utilizados en aplicaciones reales. Para este trabajo se exploraron diversas estrategias de validación utilizadas en la aplicabilidad de los datos sintéticos para modelos de aprendizaje automático. La correcta validación de estos datos es crucial para garantizar que los modelos entrenados con datos sintéticos puedan ser utilizados en escenarios del mundo real sin pérdida de precisión ni sesgo de los resultados.

A continuación, se describen las posibles estrategias para evaluar los datos empleando modelos de aprendizaje automático.

- *Train and Test on Real (TTR)*: Esta técnica consiste en entrenar y evaluar el modelo exclusivamente con datos reales para obtener una referencia sobre el rendimiento esperado en condiciones ideales y así usarlo como línea base para comparar con otras estrategias.
- *Train and Test on Synthetic (TTS)*: En esta estrategia, tanto el entrenamiento como la evaluación se realizan únicamente con datos sintéticos. Permite verificar la coherencia interna de los datos generados, pero no garantiza que los datos sintéticos puedan ser utilizados en escenarios reales, ya que no evalúa su capacidad de generalización a datos reales.
- *Train on Real, Test on Synthetic (TRTS)*: Este método entrena un modelo con datos reales y lo evalúa con datos sintéticos. Su objetivo es analizar en qué medida los datos sintéticos reflejan la estructura y distribución de los datos reales. Si la calidad de los datos sintéticos no es suficiente, es posible que el rendimiento del modelo disminuya significativamente en la fase de prueba, indicando que los datos generados no son fieles a los reales. Sin embargo, un bajo rendimiento no implica que los datos sintéticos no sean apropiados para el entrenamiento de modelos de IA, sino que reflejan diferencias que afectan a la compatibilidad.
- *Train on Synthetic, Test on Real (TSTR)*: Esta estrategia implica entrenar el modelo con datos sintéticos y evaluar el rendimiento con los datos reales. Este enfoque es valioso para determinar si los datos generados pueden reemplazar a los reales en el entrenamiento de modelos de aprendizaje automático. Es decir, se evalúa la capacidad de los datos sintéticos para ser utilizados en aplicaciones donde los datos reales son limitados, difíciles de obtener o presentan restricciones de privacidad. Un rendimiento satisfactorio en la evaluación con datos reales indicaría que los datos sintéticos contienen información relevante y estructuralmente similar a los datos originales.

Para este trabajo, TSTR representa la mejor opción para evaluar la utilidad de los datos sintéticos estáticos AIS generados. Con este enfoque se puede determinar en qué medida serían capaces de sustituir o complementar los datos reales en modelos de aprendizaje automático aplicados a la identificación y caracterización de barcos, y evaluar cuál de las tres técnicas de generación de datos sintéticos logra mejores resultados. El algoritmo utilizado para el modelo de aprendizaje automático es *Random Forest*. Las principales métricas empleadas para evaluar los resultados en aprendizaje automático se describen en la Tabla 4-2.

MÉTRICA	DESCRIPCIÓN
Exactitud (<i>Accuracy</i>)	Porcentaje de predicciones que fueron correctas sobre el total de predicciones realizadas por el modelo. Puede ser engañosa en conjuntos de datos desbalanceados.
Precisión (<i>Precision</i>)	Mide cuantas predicciones positivas fueron realmente positivas en cada clase. Es decir, es la proporción de predicciones correctas de una clase sobre el total de predicciones hechas para esa clase.
Exhaustividad o sensibilidad (<i>Recall</i>)	Proporción de elementos correctamente identificados de una clase sobre el total de elementos de esa clase.
Valor F1 (<i>F1 score</i>)	Balance entre precisión y exhaustividad que permite evaluar la calidad del modelo. Si un modelo tiene precisión del 100% pero exhaustividad del 50%, el Valor-F1 equilibra ambas métricas

AUC-ROC (Area Under the Curve – Receiver Operating Characteristic)	El área bajo la curva ROC (<i>Receiver Operating Characteristic</i>), mide la capacidad del modelo para distinguir entre dos clases.
--	--

Tabla 4-2: Métricas de evaluación de utilidad (fuente propia)

Aunque la métrica AUC-ROC es muy útil para evaluar la capacidad discriminativa de los modelos en problemas de clasificación, su aplicación directa en contextos de clasificación con múltiples clases, como el presente trabajo, puede resultar ambigua. Esta métrica está diseñada principalmente para problemas binarios, por lo que para aplicarla correctamente en escenarios multiclase es necesario transformar el problema a un enfoque *One-vs-Rest*, calculando un AUC individual por clase y luego calculando el promedio. Dado que el objetivo principal de este análisis es comparar el rendimiento global de diferentes conjuntos de datos sintéticos en modelos de clasificación multiclase de tipo de buque, se considerarán más las métricas de precisión, *recall*, *F1-score* y exactitud. Estas métricas permiten una comparación más clara y coherente entre técnicas, especialmente en un contexto con clases desbalanceadas. No obstante, a pesar de que se calcule el AUC-ROC global mediante el enfoque *One vs Rest*, también se podría considerar el uso de AUC-ROC para evaluar la capacidad del modelo para discernir la clase minoritaria *passenger* del resto de clases.

4.3 Análisis de resultados

4.3.1 Resultados de los datos sintéticos estáticos sin agrupar por tipo de buque

En la siguiente sección se examinan los datos sintéticos estáticos generados mediante TVAE, *Gaussian Copula* y CTGAN, sin agrupar por tipo de buque, a partir del archivo *final_static_data_basic.csv* con las métricas descritas en el apartado 4.1.

Se aplica el test *Chi-cuadrado* para evaluar si las técnicas de generación de datos sintéticos preservaban el balance de clases del campo categórico *shiptype*. Los resultados obtenidos se muestran en la Tabla 4-3.

	TVAE	<i>Gaussian Copula</i>	CTGAN
<i>Chi-cuadrado</i>	173.2877486363599	0.24010050549459644	8811.78260476053
<i>p</i>	2.0595154785443013e-36	0.9933455397151213	0.0

Tabla 4-3: Resultados *chi-cuadrado* para los datos sintéticos sin agrupación por *shiptype*

Para TVAE y CTGAN se rechaza la hipótesis nula pues $p < 0,05$ en ambos casos. Es decir, no siguen la distribución original de *shiptype*. En cambio, para *Gaussian Copula* $p > 0,05$ por lo que no se rechaza la hipótesis nula, es decir la distribución sintética del campo categórico *shiptype* es similar a la real. Esta conclusión debe interpretarse con cautela, ya que la prueba *chi-cuadrado* es especialmente sensible en muestras grandes como la utilizada en este trabajo (más de 45.000 registros), lo que puede llevar a rechazar la hipótesis nula incluso ante pequeñas desviaciones. Por ello, se generan los diagramas de sectores del balance de clases según la técnica utilizada. La Figura 4-1, Figura 4-2 y Figura 4-3 representan dichos diagramas para TVAE, *Gaussian Copula* y CTGAN, respectivamente.

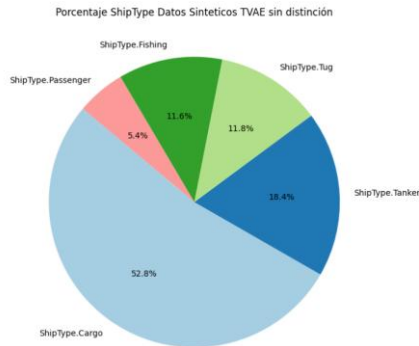


Figura 4-1: Balance de clases para TVAE sin distinción

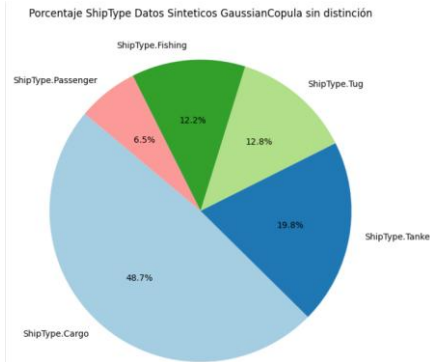


Figura 4-2: Balance de clases para Gaussian Copula sin distinción

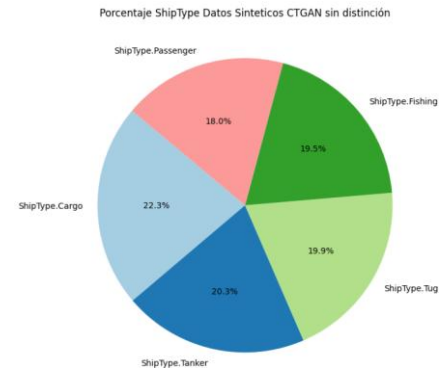


Figura 4-3: Balance de clases para CTGAN sin distinción

Según los diagramas, el balance de clases para TVAE sigue una proporción parecida a la real, pero de acuerdo con la prueba de *chi-cuadrado*, el valor de p indica diferencias estadísticamente significativas. La distribución de CTGAN, como ya indicaba el valor de $p=0.0$ en *chi-cuadrado*, no se aproxima a la distribución real pues genera distribuciones balanceadas entre todos los tipos de buque. En cambio, las proporciones de los *shiptypes* en *Gaussian Copula* sí se igualan al balance real tanto en *chi-cuadrado* como en la visualización en la Figura 4-2.

La prueba de *chi-cuadrado* detecta diferencias matemáticas, pero el impacto real sobre la distribución puede ser mínimo como ocurre en TVAE. Por tanto, aunque el resultado de la prueba sea significativo a nivel estadístico, no se considera suficiente para descartar la calidad de la técnica en el balance de clases de buque.

A continuación, se compara la similitud de las distribuciones real y sintética del resto de variables numéricas. Para ello, se aplica *KS-test* a cada variable y por cada tipo de buque. La Figura 4-4 muestra los resultados obtenidos para TVAE. El *KS-test* devuelve valores de $p < 0.05$ en todos los tipos de barco, por lo que se rechaza la hipótesis nula de que vienen de la misma distribución subyacente. Esto indica que existe discrepancia entre las distribuciones, es decir, ninguna técnica ha logrado replicar fielmente la distribución de los datos reales.

```

KS-Test para el tipo de barco: ShipType.Tug
draught: KS-Statistic=0.1004, P-Value=0.0000
to_bow: KS-Statistic=0.0913, P-Value=0.0000
to_stern: KS-Statistic=0.0766, P-Value=0.0000
to_port: KS-Statistic=0.2215, P-Value=0.0000
to_starboard: KS-Statistic=0.1721, P-Value=0.0000

KS-Test para el tipo de barco: ShipType.Fishing
draught: KS-Statistic=0.1467, P-Value=0.0000
to_bow: KS-Statistic=0.1267, P-Value=0.0000
to_stern: KS-Statistic=0.0923, P-Value=0.0000
to_port: KS-Statistic=0.2818, P-Value=0.0000
to_starboard: KS-Statistic=0.1571, P-Value=0.0000

KS-Test para el tipo de barco: ShipType.Cargo
draught: KS-Statistic=0.0628, P-Value=0.0000
to_bow: KS-Statistic=0.0745, P-Value=0.0000
to_stern: KS-Statistic=0.0862, P-Value=0.0000
to_port: KS-Statistic=0.1001, P-Value=0.0000
to_starboard: KS-Statistic=0.0664, P-Value=0.0000

KS-Test para el tipo de barco: ShipType.Passenger
draught: KS-Statistic=0.1035, P-Value=0.0000
to_bow: KS-Statistic=0.1226, P-Value=0.0000
to_stern: KS-Statistic=0.1079, P-Value=0.0000
to_port: KS-Statistic=0.0845, P-Value=0.0000
to_starboard: KS-Statistic=0.1288, P-Value=0.0000

KS-Test para el tipo de barco: ShipType.Tanker
draught: KS-Statistic=0.1306, P-Value=0.0000
to_bow: KS-Statistic=0.1275, P-Value=0.0000
to_stern: KS-Statistic=0.2348, P-Value=0.0000
to_port: KS-Statistic=0.1022, P-Value=0.0000
to_starboard: KS-Statistic=0.1672, P-Value=0.0000

```

Figura 4-4: Resultados del KS-test para TVAE sin distinción

El *KS-test* de los datos sintéticos generados con *Gaussian Copula* y CTGAN, también devolvió valores de p cercanos a 0 en todas las variables y para todos los *shiptype*. Esto no implica necesariamente que las técnicas de generación sean incorrectas, sino que el test detecta diferencias estadísticas entre las distribuciones reales y sintéticas debido a su elevada sensibilidad, especialmente con muestras grandes como las empleadas en este trabajo.

Para comprobar la sensibilidad se repitió el *test*, pero comparándose una muestra de 100 datos, en lugar de ambos conjuntos completos. Se observó que, al reducir el tamaño de datos, los valores *p* aumentaban, lo que indica que el *KS-test* es muy sensible con tamaños de muestra grandes. Los valores de *p* aumentaron concretamente para ciertas variables de CTGAN como muestra la Figura 4-5, pero para el resto de técnicas la mayoría de variables continuaron siendo próximos a 0.

	shiptype	variable	ks_statistic	p_value	sample_size	resultado
1	ShipType.Cargo	draught	0.11665610716386843	0.12135985226624328	100	Similares
2	ShipType.Cargo	to_bow	0.1241150382404852	0.08419632331428514	100	Similares
3	ShipType.Cargo	to_stern	0.12505272208897128	0.08028459706771635	100	Similares
4	ShipType.Cargo	to_port	0.08007783862062723	0.5169744419040552	100	Similares
5	ShipType.Cargo	to_starboard	0.2925537403267412	4.458576689598348e-08	100	Diferentes
6	ShipType.Fishing	draught	0.22435056746532156	7.95892997590637e-05	100	Diferentes
7	ShipType.Fishing	to_bow	0.06441902359935146	0.7848269839601497	100	Similares
8	ShipType.Fishing	to_stern	0.14300486398847054	0.03235551427024853	100	Diferentes
9	ShipType.Fishing	to_port	0.2865249504593767	.2470513479426463e-07	100	Diferentes
10	ShipType.Fishing	to_starboard	0.21285534137993153	0.0002198267990133775	100	Diferentes
11	ShipType.Passenger	draught	0.19042582417582418	0.0015193028749813774	100	Diferentes
12	ShipType.Passenger	to_bow	0.11427197802197803	0.1477779139891867	100	Similares
13	ShipType.Passenger	to_stern	0.2091620879120879	0.000344769305959496	100	Diferentes
14	ShipType.Passenger	to_port	0.24785714285714286	.0210480574940923e-05	100	Diferentes
15	ShipType.Passenger	to_starboard	0.1939423076923077	0.0011627418101966853	100	Diferentes
16	ShipType.Tanker	draught	0.11831738259514801	0.11552548934672147	100	Similares
17	ShipType.Tanker	to_bow	0.12977298019140887	0.06517633718520366	100	Similares
18	ShipType.Tanker	to_stern	0.119158691297574	0.11096990753674223	100	Similares
19	ShipType.Tanker	to_port	0.1497529490318273	0.021149228363671495	100	Diferentes
20	ShipType.Tanker	to_starboard	0.19332962385933675	0.0010318641817410208	100	Diferentes
21	ShipType.Tug	draught	0.1612145679436523	0.010624164017926721	100	Diferentes
22	ShipType.Tug	to_bow	0.10291874248410926	0.2319910163328472	100	Similares
23	ShipType.Tug	to_stern	0.14236385500773063	0.03345895255822283	100	Diferentes
24	ShipType.Tug	to_port	0.2736041917196358	5.404565291116647e-07	100	Diferentes

Figura 4-5: Resultados *KS-test* para muestra de 100 datos con CTGAN sin distinción

Con el objetivo de entender por qué *KS-test* da valores *p* cercanos a 0, se realiza un análisis del número de valores únicos por variable y tipo de buque en el conjunto de datos reales y en los diferentes conjuntos de datos sintéticos generados con TVAE, *Gaussian Copula* y CTGAN. La Figura 4-6 muestra el porcentaje de valores únicos por variable para los datos sintéticos y los datos reales.

shiptype	variable	%_unique_TVAE_sin_distincion	%_unique_GC_sin_distincion	%_unique_CTGAN_sin_distincion	%_unique_Real
ShipType.Cargo	draught	100.00	100.00	100.00	77.17
ShipType.Cargo	to_bow	100.00	100.00	100.00	9.21
ShipType.Cargo	to_stern	99.85	99.91	99.99	8.75
ShipType.Cargo	to_port	99.98	99.94	99.98	14.10
ShipType.Cargo	to_starboard	99.99	99.96	99.98	14.11
ShipType.Fishing	draught	100.00	100.00	100.00	7.94
ShipType.Fishing	to_bow	99.80	100.00	99.66	8.67
ShipType.Fishing	to_stern	98.91	99.97	99.83	8.97
ShipType.Fishing	to_port	99.88	99.84	98.85	10.65
ShipType.Fishing	to_starboard	99.88	99.89	99.90	10.50
ShipType.Passenger	draught	99.89	100.00	99.99	14.49
ShipType.Passenger	to_bow	99.86	100.00	99.76	13.67
ShipType.Passenger	to_stern	99.78	99.97	100.00	18.82
ShipType.Passenger	to_port	100.00	100.00	99.70	14.97
ShipType.Passenger	to_starboard	100.00	100.00	100.00	14.94
ShipType.Tanker	draught	100.00	100.00	100.00	77.00
ShipType.Tanker	to_bow	100.00	100.00	100.00	9.63
ShipType.Tanker	to_stern	99.97	99.97	99.98	7.95
ShipType.Tanker	to_port	99.94	100.00	100.00	12.04
ShipType.Tanker	to_starboard	100.00	99.98	99.96	11.99
ShipType.Tug	draught	100.00	100.00	100.00	12.76
ShipType.Tug	to_bow	99.71	100.00	99.11	10.32
ShipType.Tug	to_stern	99.78	99.72	100.00	10.07
ShipType.Tug	to_port	99.97	99.73	99.14	11.91
ShipType.Tug	to_starboard	100.00	99.70	100.00	11.87

Figura 4-6: Comparación % de valores únicos en los datos sintéticos sin distinción con los datos reales

Los datos sintéticos presentan un número de valores únicos muy elevado, incluso cercano al 100%, en comparación con los reales. Es decir, prácticamente cada valor es diferente, lo que indica una excesiva variabilidad en comparación con los datos reales. Esto puede provocar que la función de distribución acumulada (CDF) de los datos sintéticos y la de los datos reales difieran más bruscamente, pues la CDF de los sintéticos puede ser más suave y continua, respecto a la CDF de los datos reales (que puede tener valores repetidos o regiones más planas), y que *KS-test* detecte esas diferencias como significativas, aunque no lo sean a nivel visual en un gráfico KDE. Por ello, se complementa el análisis con otras herramientas visuales y estadísticas. Además, aunque el valor de p indique que son diferentes si la estadística *KS-statistic* es baja, como es el caso, el dato sintético puede ser válido.

Para visualizar mejor estas diferencias se generan los gráficos KDE. Se comparan los *shiptype cargo*, por ser el más abundante, y *passenger*, por ser el menos representado. Se analiza el parámetro *to_bow* en ambos buques para cada técnica de generación de datos empleada.

La Figura 4-7, Figura 4-9 y la Figura 4-11 muestran las distribuciones de densidad de *to_bow* para los buques de tipo *cargo* y la Figura 4-8, Figura 4-10 y la Figura 4-12 muestran las de los *shiptype passenger*.

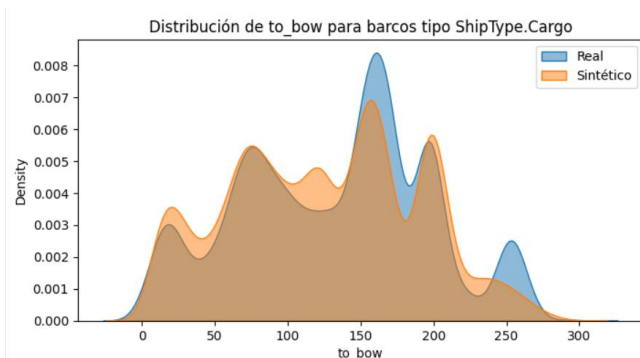


Figura 4-7: Función densidad del parámetro *to_bow* para *cargo* sin agrupación con TVAE

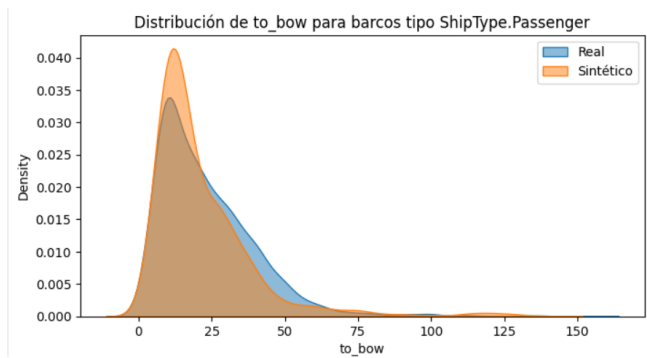


Figura 4-8: Función densidad del parámetro *to_bow* para *passenger* sin agrupación con TVAE

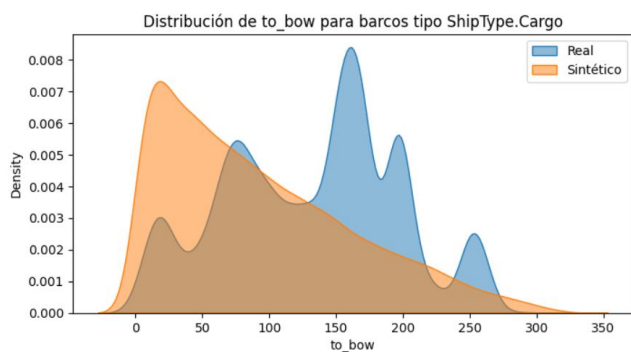


Figura 4-9: Función densidad del parámetro *to_bow* para *cargo* sin agrupación con Gaussian Copula

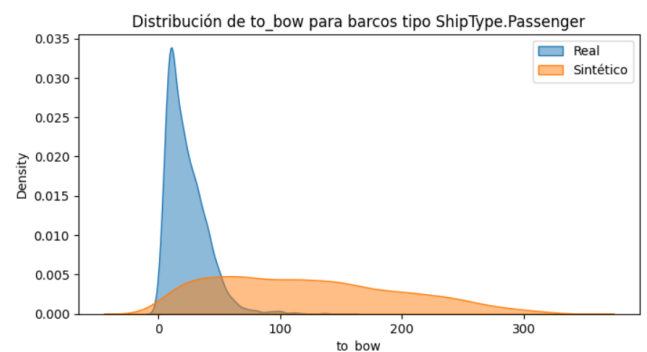


Figura 4-10: Función densidad del parámetro *to_bow* para *passenger* sin agrupación con Gaussian Copula

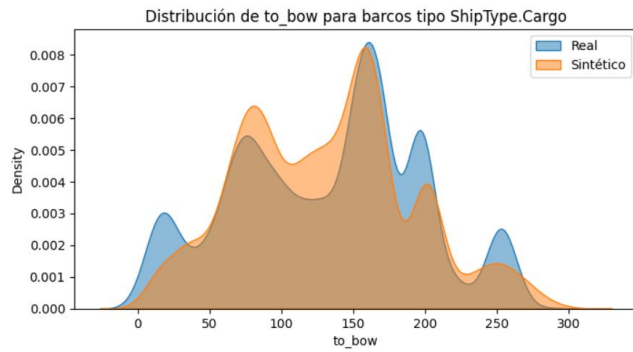


Figura 4-11: Función densidad del parámetro *to_bow* para *cargo* sin agrupación con CTGAN

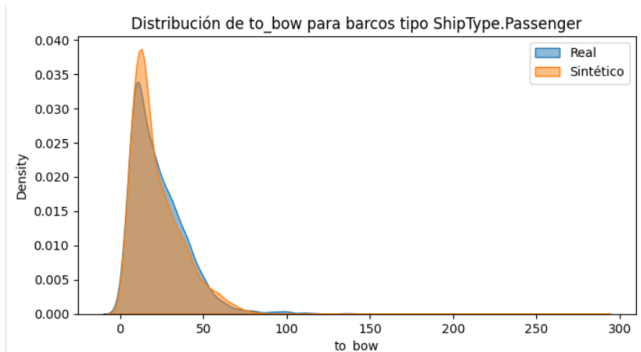


Figura 4-12: Función densidad del parámetro *to_bow* para *passenger* sin agrupación con CTGAN

La Figura 4-7 y la Figura 4-8 se corresponden con la generación mediante TVAE en la cual:

- La distribución sintética para la categoría *cargo* se aproxima de manera significativa a la forma de la distribución real, sin embargo, presenta menor concentración en algunos picos en comparación con los reales.
- La distribución sintética de la categoría *passenger* también presenta una distribución similar a la real, aunque con una diferencia de densidad en el pico mayor que los reales. Además, en la cola de la distribución, donde se encuentran los valores extremos, la densidad de probabilidad es menor.

La Figura 4-9 y la Figura 4-10 se corresponden con la generación mediante *Gaussian Copula* donde:

- La distribución sintética para la categoría *cargo* no sigue la forma general de la distribución real. Tiende a generar datos de forma suavizada, reduciendo la variabilidad, lo que se refleja en un desplazamiento de la distribución hacia valores más bajos.
- En el caso de la categoría *passenger*, la discrepancia también es notable. Los datos reales presentan una concentración en un pico mientras que los sintéticos exhiben un rango más amplio y están desplazados hacia valores superiores. Esta observación indica que el modelo de *Gaussian Copula* no ha logrado capturar adecuadamente la distribución de los datos reales para este tipo de embarcación.

La Figura 4-11 y la Figura 4-12 se corresponden con la generación mediante CTGAN donde:

- Para la distribución sintética de *cargo*, CTGAN muestra mayor capacidad para replicar la forma de la distribución real. Sin embargo, las colas de las distribuciones son diferentes. Hay menor número de valores en los extremos, lo que indica una menor variabilidad en los datos generados.
- En el caso de *passenger*, se aprecia mejora en comparación con TVAE, pues hay una mayor similitud de forma, pero hay un sobreajuste considerable en el pico de la gráfica.

Para el análisis por gráficos de densidad, el modelo TVAE parece captar de forma adecuada la distribución de *to_bow* para *cargo*, y para *passenger*, aunque presenta alguna deficiencia. Por otra parte, el modelo de *Gaussian Copula* tiende a suavizar excesivamente los datos, lo que resulta en una distribución de densidad errónea, incluso sin asemejarse a una *gaussiana*. Y, por último, el modelo CTGAN muestra mejores aproximaciones de la forma de la distribución de densidad, pero con tendencias al sobreajuste de ciertos valores.

A continuación, la Tabla 4-4 y Tabla 4-5 muestran las estadísticas descriptivas del parámetro *to_bow* de ambos tipos de buques.

		<i>count</i>	<i>mean</i>	<i>std</i>	<i>min</i>	25%	50%	75%	<i>max</i>
Cargo	Real	22097	131.954	64.9193	0	79.0000	144.00	173.00	300.000
	TVAE	26404	125.327	64.0308	0	74.2497	125.920	170.617	294.398
	<i>Gaussian Copula</i>	24362	91.718	70.5921	0.0040	32.9232	76.1851	137.325	325.354
	CTGAN	11162	131.706	59.5537	1.2439	83.8469	133.321	166.874	302.673

Tabla 4-4: Resultados de las estadísticas descriptivas de *to_bow* para *cargo* sin distinción

		<i>count</i>	<i>mean</i>	<i>std</i>	<i>min</i>	25%	50%	75%	<i>max</i>
Passenger	Real	2912	23.281	16.7964	0	10.00	20.00	32.00	154.00
	TVAE	2707	21.711	18.138	0	10.6444	16.1507	27.7284	140.699
	<i>Gaussian Copula</i>	3244	21.784	75.489	0.0982	59.4939	113.320	176.276	329.018
	CTGAN	9021	22.439	17.330	0	11.0163	17.4017	30.2975	286.037

Tabla 4-5: Resultados de las estadísticas descriptivas de *to_bow* para *passenger* sin distinción

Para *cargo* se observa que TVAE genera una media próxima a la real manteniendo la variabilidad, pues su desviación estándar es similar a la de los datos reales. *Gaussian Copula* genera una media mucho menor pues, como muestra el gráfico, los valores sintéticos se agrupaban entre 0 y 100. Además, su desviación estándar es mayor, lo que sugiere mayor dispersión de los datos. En cambio, CTGAN mantiene una media muy similar a la real, aunque con menor variación.

Para *passenger* se observa que TVAE tiene una media cercana a la real y con mayor variabilidad. *Gaussian Copula* también mantiene una media próxima, pero sobreestima la desviación estándar considerablemente, generando valores mayores y más dispersos. CTGAN logra una media y desviación estándar más próximos a la real, lo que indica que mantiene una variabilidad relativamente bien representada.

En general, TVAE parece la mejor técnica a la hora de preservar distribuciones generales de los datos. *Gaussian Copula* presenta mayores problemas en la representación de los datos sintéticos. Y CTGAN, aunque muestra medias cercanas a las reales, para conjuntos de datos grandes, como el caso de *cargo*, reduce la variabilidad, ajustándose mejor para conjuntos de menor tamaño como *passenger*. Aunque esto puede deberse a que el número de datos generados por CTGAN para *cargo* es aproximadamente un 50% menor en comparación con el resto de técnicas, y el número generado para *passenger* triplica el del resto de técnicas.

Uno de los aspectos fundamentales en la evaluación de la calidad de los datos sintéticos es la capacidad de las técnicas de preservar las relaciones entre variables. Para ello, se han comparado las matrices de correlación de los datos reales representadas en la Figura 4-13 para *cargo* y en la Figura 4-14 para *passenger*, con los generados sintéticamente representadas en la Figura 4-15 y Figura 4-16 para TVAE, en la Figura 4-17 y Figura 4-18 para *Gaussian Copula*, y en la Figura 4-19 y Figura 4-20 para CTGAN.

En los datos reales, las correlaciones entre variables muestran una estructura clara con relaciones bien definidas entre los distintos atributos del barco. Se destacan correlaciones fuertes entre los parámetros *to_bow*, *to_stern*, *to_port*, *to_starboard* y *draught* lo cual es esperable ya que estas variables están relacionadas con la geometría de la embarcación.

El análisis de las matrices de correlación presenta TVAE como la mejor técnica en la conservación de correlaciones para ambos tipos de buque, aunque con ligeras pérdidas de dependencia en algunas variables sobre todo para la categoría *passenger*. *Gaussian Copula* se comporta de manera aceptable para las relaciones del *shiptype cargo*, pero muestra desviaciones importantes en *passenger*. Y CTGAN presenta las pérdidas más significativas en las correlaciones, lo que indica que su capacidad para mantener la estructura de relaciones de los datos reales es limitada.

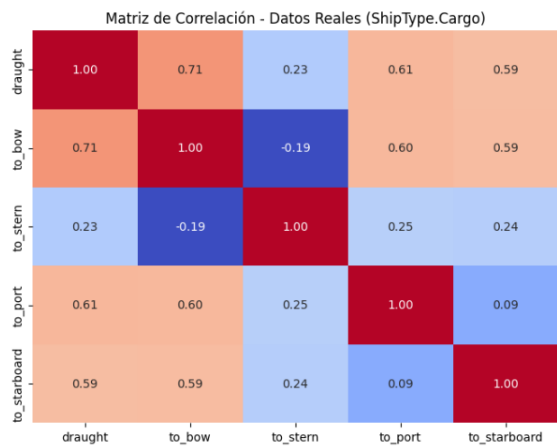


Figura 4-13: Matriz de correlación para *cargo* de los datos reales

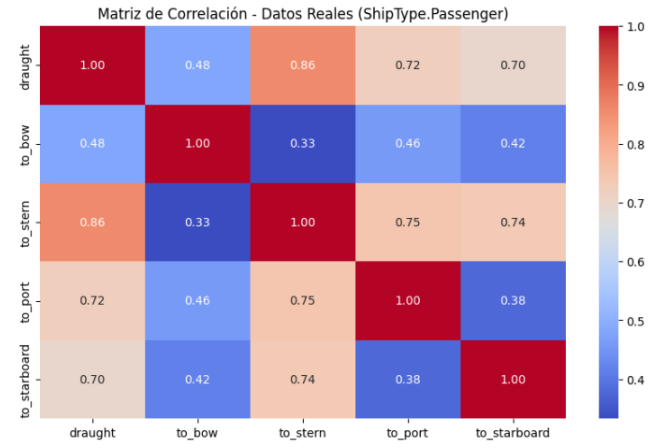


Figura 4-14: Matriz de correlación para *passenger* de los datos reales

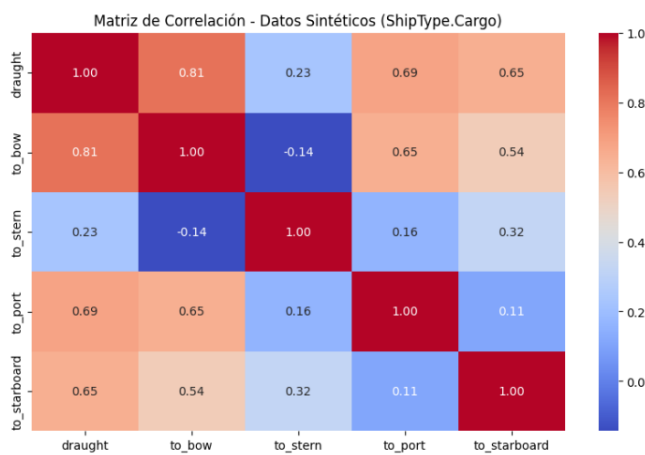


Figura 4-15: Matriz de correlación para *cargo* de *basic* sin agrupación con TVAE

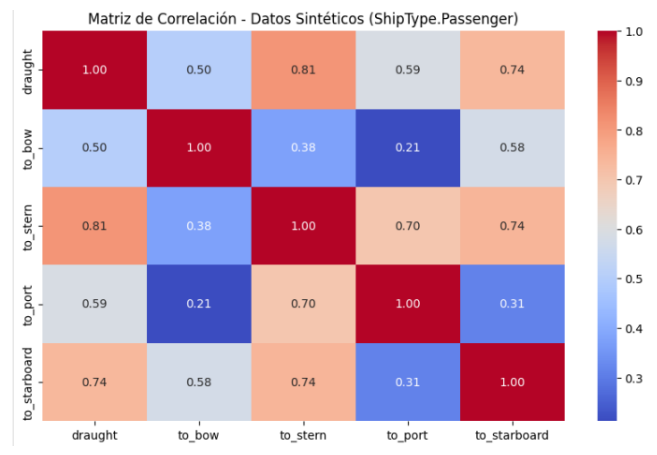


Figura 4-16: Matriz de correlación para *passenger* de *basic* sin agrupación con TVAE

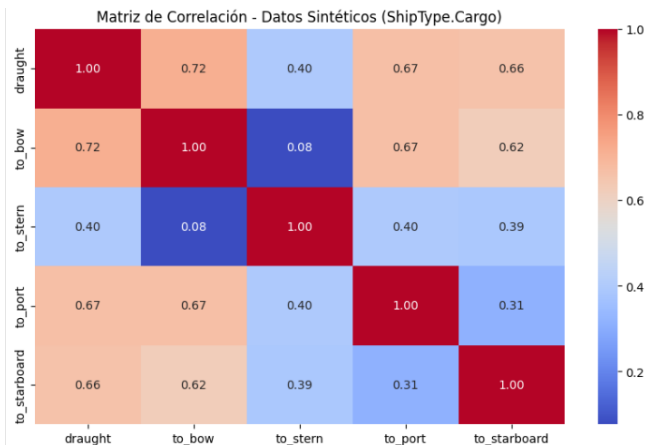


Figura 4-17: Matriz de correlación para *cargo* de *basic* sin agrupación con *Gaussian Copula*

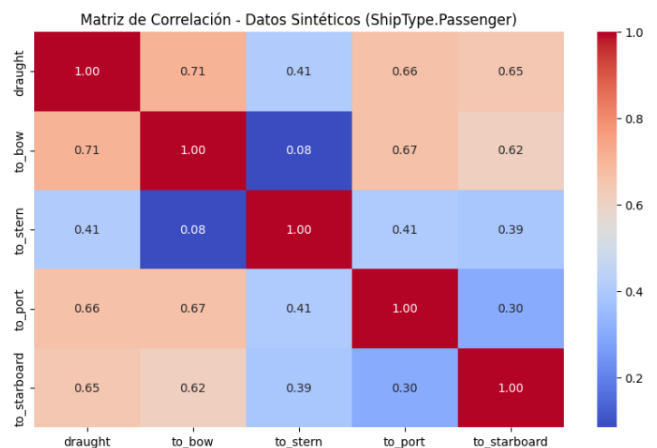


Figura 4-18: Matriz de correlación para *passenger* de *basic* sin agrupación con *Gaussian Copula*

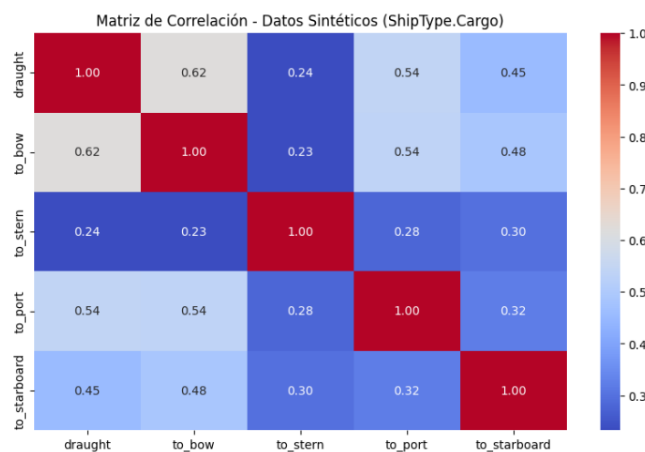


Figura 4-19: Matriz de correlación para *cargo* de *basic* sin agrupación con CTGAN

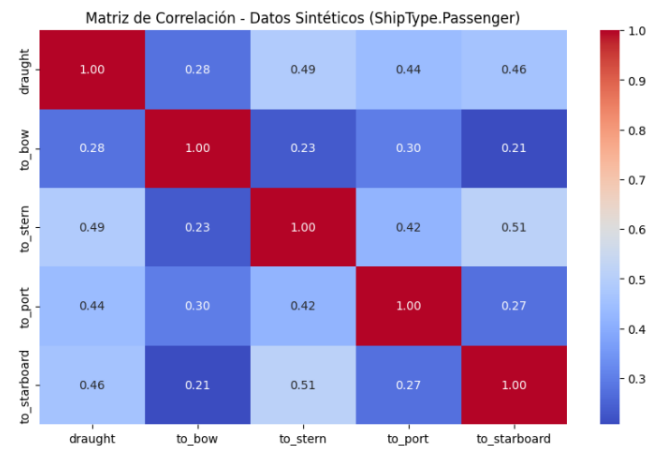


Figura 4-20: Matriz de correlación para *passenger* de *basic* sin agrupación con CTGAN

Para visualizar si los datos generados por TVAE, *Gaussian Copula* y CTGAN conservan la estructura de los datos reales, se aplican los análisis de PCA y t-SNE, para los *shiptype cargo* y *passenger*.

La Figura 4-21 y la Figura 4-22 muestran los gráficos PCA de los datos generados con TVAE para *cargo* y *passenger*, respectivamente, la Figura 4-23 y la Figura 4-24 para *Gaussian Copula*, y la Figura 4-25 y la Figura 4-26 para CTGAN.

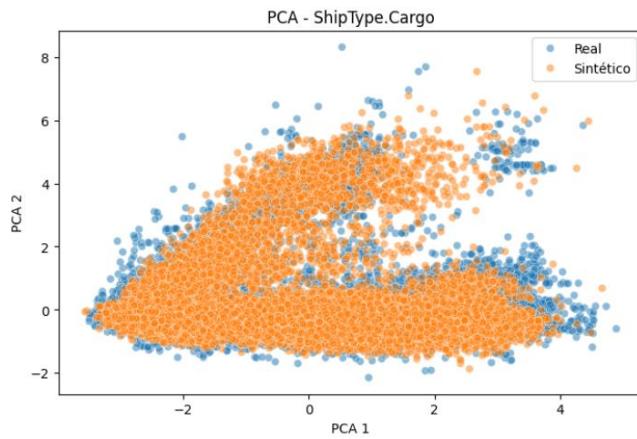


Figura 4-21: Gráfico PCA de los datos *basic* sin agrupación con TVAE para *cargo*



Figura 4-22: Gráfico PCA de los datos *basic* sin agrupación con TVAE para *passenger*

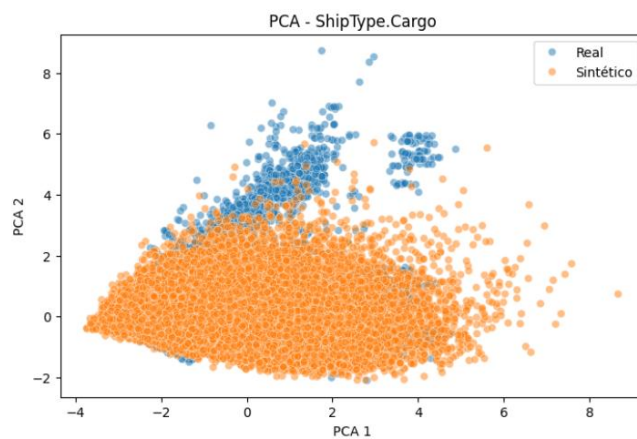


Figura 4-23: Gráfico PCA de los datos *basic* sin agrupación con *Gaussian Copula* para *cargo*

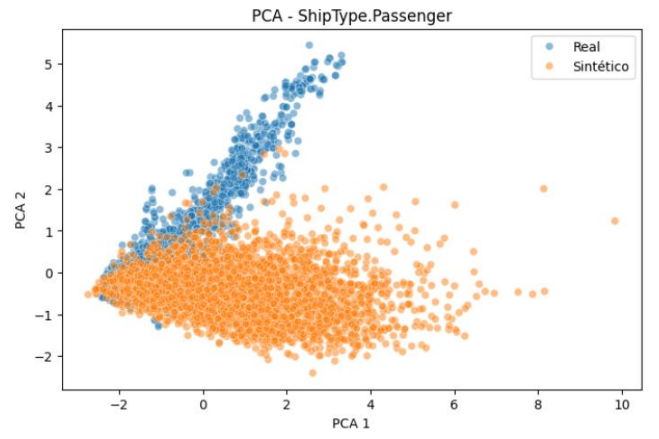


Figura 4-24: Gráfico PCA de los datos *basic* sin agrupación con *Gaussian Copula* para *passenger*

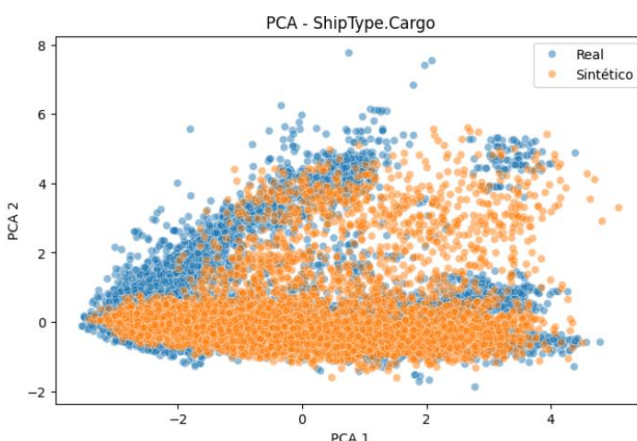


Figura 4-25: Gráfico PCA de los datos *basic* sin agrupación con CTGAN para *cargo*

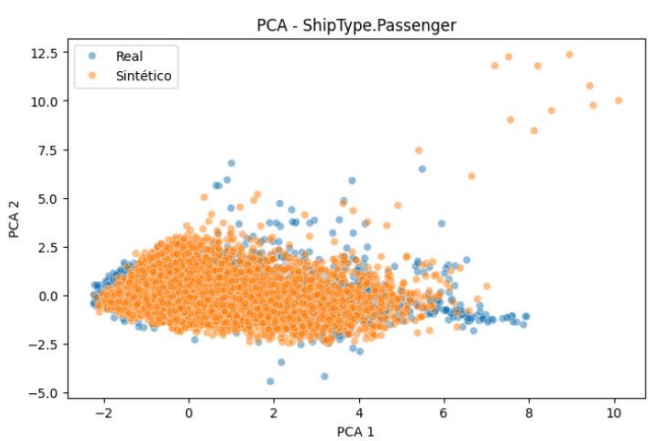


Figura 4-26: Gráfico PCA de los datos *basic* sin agrupación con CTGAN para *passenger*

El análisis de las componentes principales (PCA) para el *shiptype cargo*, muestra que el modelo TVAE genera datos sintéticos alineados con los datos reales, aunque con pequeñas diferencias en la dispersión. En *Gaussian Copula* se observa una menor alineación con los datos reales con una concentración de los sintéticos en la parte inferior, lo que sugiere que la técnica no logra captar

adecuadamente la variabilidad de estos datos. CTGAN muestra una buena correspondencia con los datos reales, pero con mayor variabilidad en algunas regiones, lo que indica que algunos patrones pueden haber sido distorsionados.

Para los buques tipo *passenger*, TVAE muestra ligeros desplazamientos que pueden indicar un sesgo en la generación. En *Gaussian Copula* se observa una mayor separación entre los datos sintéticos y reales, por lo que esta técnica muestra dificultades para capturar la estructura de los datos originales. Y para CTGAN se observa una mejor alineación con los datos reales en comparación con *cargo*, debido posiblemente al mayor número de datos de este tipo de buque para CTGAN.

En el análisis de t-SNE, que permite visualizar la agrupación local y la similitud de los datos sintéticos y reales, los modelos TVAE y *Gaussian Copula* para los buques *cargo*, representados en la Figura 4-27 y la Figura 4-29 respectivamente, muestran una buena superposición de los datos, por lo que se puede concluir que estas técnicas capturan bien las relaciones locales. Pero para CTGAN, en la Figura 4-31, los datos sintéticos están parcialmente superpuestos con los reales, pues hay áreas en las que la separación es evidente, lo que sugiere que esta técnica no captura adecuadamente la estructura de los datos para los buques tipo *cargo*.

En cambio, para los *shiptype passenger*, las técnicas TVAE y *Gaussian Copula*, representados en la Figura 4-28 y en la Figura 4-30, muestran menor alineación en comparación con *cargo*. Se observan claros agrupamientos separados entre los datos sintéticos y los reales, lo que indica que estas técnicas generan datos con menor diversidad estructural para conjuntos reducidos. Y CTGAN en la Figura 4-32, parece generar datos con una superposición aceptable entre reales y sintéticos, siendo previsible al triplicar el número de datos sintéticos en comparación con las otras dos técnicas.

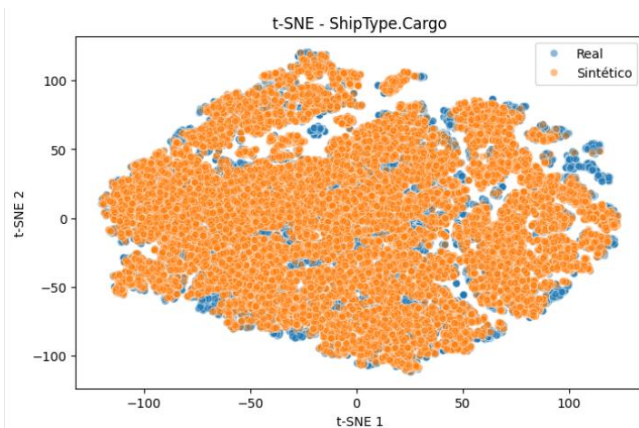


Figura 4-27: Gráfico t-SNE de los datos *basic* sin agrupación con TVAE para *cargo*

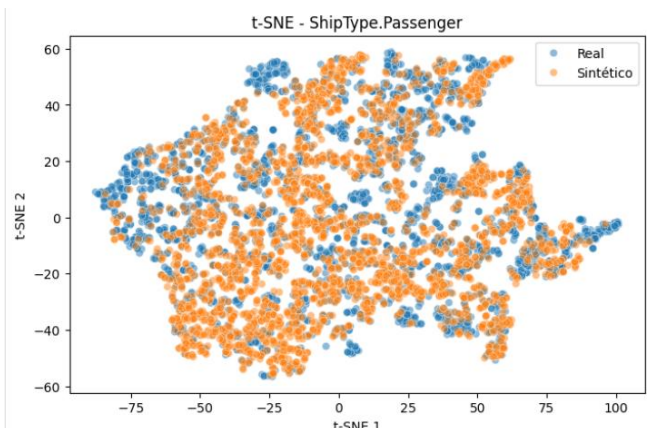


Figura 4-28: Gráfico t-SNE de los datos *basic* sin agrupación con TVAE para *passenger*

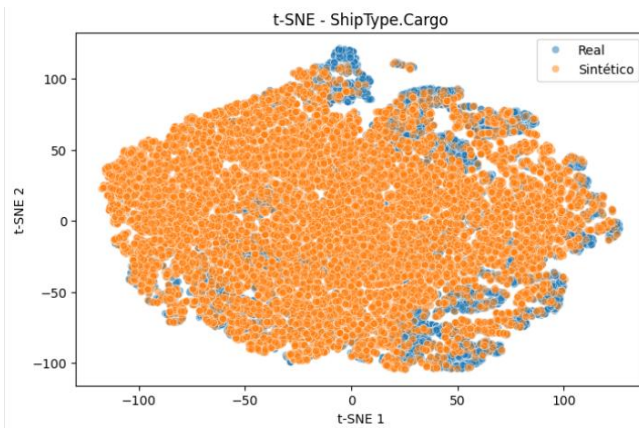


Figura 4-29: Gráfico t-SNE de los datos *basic* sin agrupación con *Gaussian Copula* para *cargo*

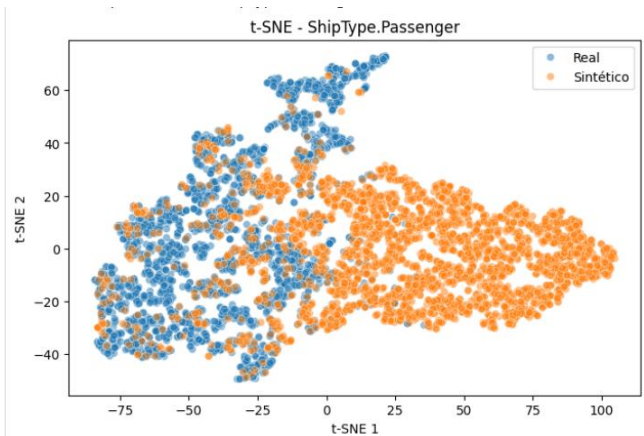


Figura 4-30: Gráfico t-SNE de los datos *basic* sin agrupación con *Gaussian Copula* para *passenger*

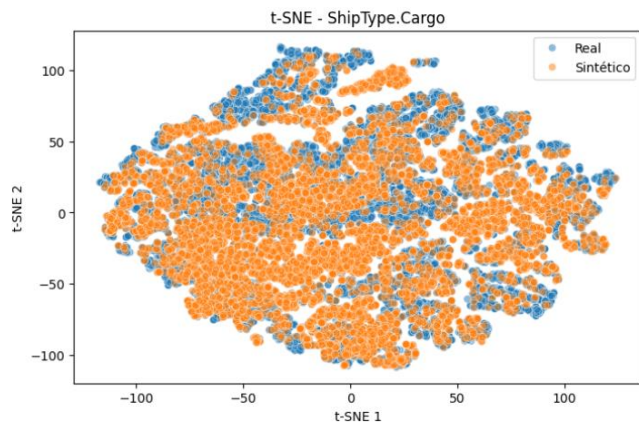


Figura 4-31: Gráfico t-SNE de los datos *basic* sin agrupación con CTGAN para *cargo*

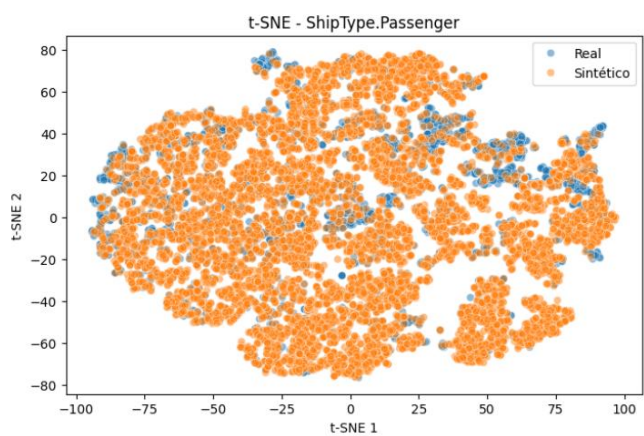


Figura 4-32: Gráfico t-SNE de los datos *basic* sin agrupación con CTGAN para *passenger*

A partir de los resultados obtenidos en los análisis de reducción de dimensionalidad (PCA y t-SNE), puede parecer que CTGAN conserva mejor la estructura de los datos sintéticos para la clase *passenger*, mientras que TVAE lo hace para la clase *cargo*. Sin embargo, esta conclusión debe tomarse con cautela, ya que CTGAN no respetó la proporción original de clases y generó una cantidad desproporcionada de barcos para *passenger* y reducida para *cargo*, lo que puede haber condicionado la representación gráfica en estos análisis. Por tanto, no puede extraerse una conclusión válida únicamente a partir de estas visualizaciones. En cambio, en el resto de métricas y análisis como el balance de clases, las correlaciones entre parámetros y las distribuciones de densidad, TVAE muestra un comportamiento más coherente y robusto, independientemente del tamaño del conjunto de datos por clase.

4.3.2 Resultados de los datos sintéticos estáticos por tipo de buque

A continuación, se compara la calidad de los datos sintéticos estáticos generados con las tres técnicas del archivo *final_static_basic.csv*, pero agrupándolos por *shiptype*. Además, se recurre a generar datos sintéticos con proporciones controladas, para permitir un análisis más justo entre las técnicas.

Dado que se aseguró mediante $\text{num_samples} = \text{int}(\text{num_total_synthetic} * \text{proportion})$ que la proporción de buques de cada categoría se mantuviera en la generación de los datos sintéticos, el test de *chi-cuadrado* muestra resultados de $p > 0,05$ e iguales para las tres técnicas, pues se genera el mismo

número de datos, ver Tabla 4-6. Por lo que ahora las tres técnicas tienen una distribución del campo categórico *shiptype* similar que permita realizar un análisis entre técnicas más acertado.

	TVAE	<i>Gaussian Copula</i>	CTGAN
<i>Chi-cuadrado</i>	4.0988078529712404e-05	4.0988078529712404e-05	4.0988078529712404e-05
<i>p</i>	0.9999999997900001	0.9999999997900001	0.9999999997900001

Tabla 4-6: Resultados *chi-cuadrado* para los datos sintéticos por *shiptype*

La prueba de *Kolmogorov-Smirnov (KS-test)* para esta configuración, aplicada a cada variable y categoría de buque vuelve a arrojar valores de *p* cercanos a cero para las tres técnicas. Esto implica, desde el punto de vista estadístico, el rechazo de la hipótesis nula, es decir, que los datos reales y sintéticos no provienen de la misma distribución. Sin embargo, como ya se mencionó en el análisis anterior, estos resultados deben interpretarse con cautela.

Se vuelve a repetir la prueba sobre un subconjunto reducido de los datos de 100 muestras, con el objetivo de evaluar las diferencias con el anterior conjunto de datos sintéticos sin distinción por tipo de buque, en el cual CTGAN fue la que arrojó un mayor número de variables con valores $p > 0,05$. Sin embargo, en esta configuración, TVAE pasa a ser la técnica que presenta más variables con $p > 0,05$ como se muestra en Figura 4-33.

	shiptype	variable	ks_statistic	p_value	sample_size	resultado
1	ShipType.Cargo	draught	0.06936054668054492	0.6954214372558456	100	Similares
2	ShipType.Cargo	to_bow	0.07660813685115625	0.5735376747818848	100	Similares
3	ShipType.Cargo	to_stern	0.08218400687876182	0.48378510553774223	100	Similares
4	ShipType.Cargo	to_port	0.09951848667239896	0.25765865155238316	100	Similares
5	ShipType.Cargo	to_starboard	0.1338792596280038	0.050407248343872016	100	Similares
6	ShipType.Fishing	draught	0.10802017654476671	0.1874947927515765	100	Similares
7	ShipType.Fishing	to_bow	0.17790308052603135	0.0034299088655982243	100	Diferentes
8	ShipType.Fishing	to_stern	0.0715078364258692	0.6691693512981641	100	Similares
9	ShipType.Fishing	to_port	0.12506935687263557	0.08469479441193091	100	Similares
10	ShipType.Fishing	to_starboard	0.16721671770852098	0.007181995379355268	100	Diferentes
11	ShipType.Passenger	draught	0.10160714285714285	0.2529062024147041	100	Similares
12	ShipType.Passenger	to_bow	0.1429120879120879	0.03463702633059804	100	Diferentes
13	ShipType.Passenger	to_stern	0.0865521978021978	0.438464039364906	100	Similares
14	ShipType.Passenger	to_port	0.07740384615384616	0.5806225260200947	100	Similares
15	ShipType.Passenger	to_starboard	0.12652472527472527	0.08266570108818236	100	Similares
16	ShipType.Tanker	draught	0.10770309370131316	0.18721369627308906	100	Similares
17	ShipType.Tanker	to_bow	0.09034943245047852	0.37228554295680993	100	Similares
18	ShipType.Tanker	to_stern	0.10212107723124861	0.23687225924921299	100	Similares
19	ShipType.Tanker	to_port	0.08885154685065658	0.392551935155565	100	Similares
20	ShipType.Tanker	to_starboard	0.12388159359002894	0.08806711666216065	100	Similares
21	ShipType.Tug	draught	0.1202044322281395	0.10728410547744996	100	Similares
22	ShipType.Tug	to_bow	0.23211303899673597	1.8551931970034394e-05	100	Diferentes
23	ShipType.Tug	to_stern	0.11909809311114929	0.11312920596425503	100	Similares
24	ShipType.Tug	to_port	0.18760178663459887	0.001672321528695853	100	Diferentes

Figura 4-33: Resultados *KS-test* para muestra de 100 datos con TVAE por *shiptype*

Al igual que en el análisis previo, la Figura 4-34 muestra que los datos sintéticos generados contienen un mayor número de valores únicos en comparación con los reales, lo que puede explicar, en parte, los valores de *p* próximos a cero, debido a esa mayor variabilidad.

shiptype	variable	%_unique_TVAE_shiptype	%_unique_GaussianCopula_shiptype	%_unique_CTGAN_shiptype	%_unique_Real
ShipType.Cargo	draught	99.91	99.53	99.86	77.17
ShipType.Cargo	to_bow	100.00	100.00	98.70	9.21
ShipType.Cargo	to_stern	99.96	99.94	99.97	8.75
ShipType.Cargo	to_port	99.98	99.92	98.94	14.10
ShipType.Cargo	to_starboard	100.00	99.95	99.90	14.11
ShipType.Fishing	draught	99.97	99.77	98.92	7.94
ShipType.Fishing	to_bow	99.97	99.92	99.80	8.67
ShipType.Fishing	to_stern	99.84	99.77	97.52	8.97
ShipType.Fishing	to_port	100.00	100.00	100.00	10.65
ShipType.Fishing	to_starboard	99.93	100.00	100.00	10.50
ShipType.Passenger	draught	99.94	100.00	99.03	14.49
ShipType.Passenger	to_bow	100.00	100.00	96.01	13.67
ShipType.Passenger	to_stern	99.72	99.81	97.20	18.82
ShipType.Passenger	to_port	99.75	99.69	99.69	14.97
ShipType.Passenger	to_starboard	99.94	99.78	99.50	14.94
ShipType.Tanker	draught	99.93	99.59	99.37	77.00
ShipType.Tanker	to_bow	100.00	100.00	99.96	9.63
ShipType.Tanker	to_stern	99.90	99.69	98.93	7.95
ShipType.Tanker	to_port	99.96	99.92	99.96	12.04
ShipType.Tanker	to_starboard	99.96	99.85	99.74	11.99
ShipType.Tug	draught	99.95	99.92	99.55	12.76
ShipType.Tug	to_bow	100.00	100.00	99.97	10.32
ShipType.Tug	to_stern	99.89	100.00	99.78	10.07
ShipType.Tug	to_port	100.00	99.94	99.95	11.91
ShipType.Tug	to_starboard	99.91	99.98	100.00	11.87

Figura 4-34: Comparación % de valores únicos en los datos sintéticos por *shiptype* con los datos reales

El *KS-test* es especialmente sensible a las diferencias en la forma de las funciones de distribución acumulada, y la presencia de muchos valores únicos en los datos sintéticos genera una curva más escalonada o irregular, lo que incrementa la discrepancia con respecto a la curva de los datos reales. Por tanto, incluso si la forma general de la distribución parece similar a nivel visual, esta alta dispersión puede conducir a un rechazo de la hipótesis nula.

A continuación, se procede con el análisis de las distribuciones KDE de la variable *to_bow* para *cargo* a la izquierda y *passenger* a la derecha, para las tres técnicas: TVAE en la Figura 4-35 y en la Figura 4-36; *Gaussian Copula* en la Figura 4-37 y en la Figura 4-38; y CTGAN en la Figura 4-39 y en la Figura 4-40, comparando los datos reales y sintéticos.

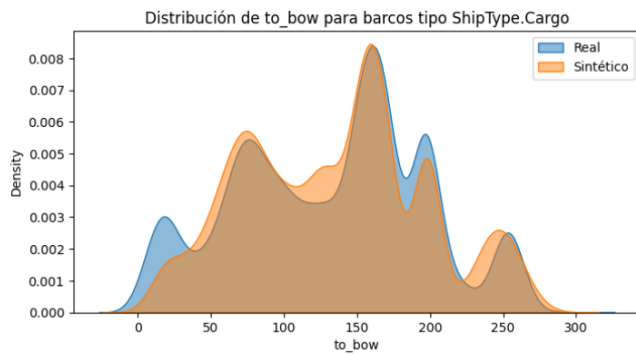


Figura 4-35: Función densidad del parámetro *to_bow* para *cargo* por *shiptype* con TVAE

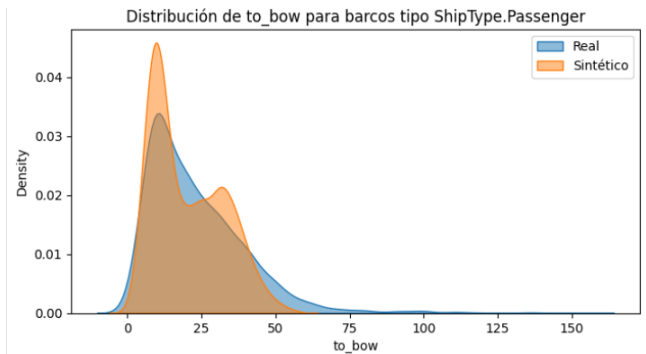


Figura 4-36: Función densidad del parámetro *to_bow* para *passenger* por *shiptype* con TVAE

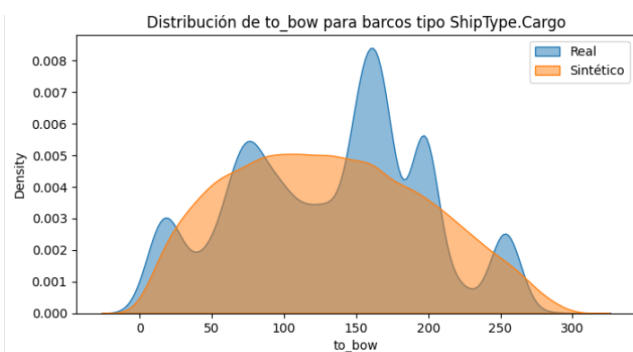


Figura 4-37: Función densidad del parámetro *to_bow* para cargo por *shiptype* con *Gaussian Copula*

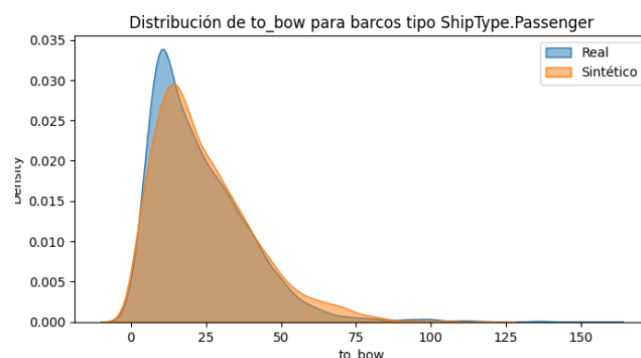


Figura 4-38: Función densidad del parámetro *to_bow* para passenger por *shiptype* con *Gaussian Copula*

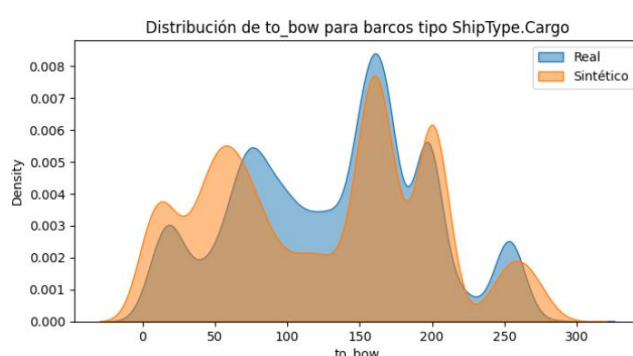


Figura 4-39: Función densidad del parámetro *to_bow* para cargo por *shiptype* con CTGAN

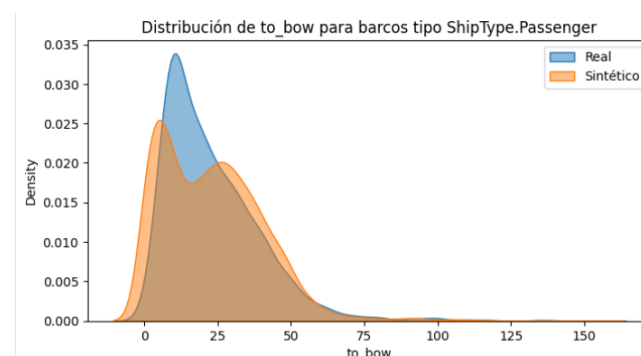


Figura 4-40: Función densidad del parámetro *to_bow* para passenger por *shiptype* con CTGAN

Para *cargo*, la técnica TVAE muestra que la distribución sintética de la variable *to_bow* se alinea mejor con los datos reales en comparación con la versión sin distinción, aunque sigue habiendo ligeras diferencias de densidad en los extremos. En *Gaussian Copula* sigue presentando una forma más suavizada en comparación con los datos reales, aunque parece haber mejorado la representación de la variabilidad en comparación con la versión sin distinción. Y para CTGAN mantiene la forma aproximándose a la real, con una forma de la distribución en extremos más fiel que en la versión anterior. No obstante, sigue existiendo una tendencia a la sobre-representación en los extremos y una sub-representación en ciertos intervalos intermedios.

Para *passenger*, la técnica TVAE ha capturado bien el pico principal, pero genera un segundo pico entre 25 y 50 que no se muestra en los datos reales, lo que sugiere una sobre-representación en este rango. La cola de la distribución presenta una caída más abrupta que indica una reducción mayor de la variabilidad en comparación con la gráfica anterior. En *Gaussian Copula* mejora considerablemente la forma y variabilidad ajustándose a la distribución real en comparación con la generación sin agrupación por *shiptype*. En CTGAN la distribución sintética es más uniforme en comparación con la generación sin agrupar por *shiptype*, donde se concentraban los valores en un pico muy marcado, aunque se observa una concentración de menor magnitud y ligeramente desplazada hacia la izquierda, lo que indica que el modelo ha generado valores en este rango con menor densidad de la esperada. Además, al igual que TVAE, aparecen dos picos que indican un sobreajuste en los valores en torno a 5 metros y 30 metros.

En el análisis de las distribuciones de densidad, *Gaussian Copula* mejora de forma significativa especialmente para *passenger* en comparación con la generación sin agrupar por tipo de buque, pero continúa suavizando la forma de la distribución para *cargo*. TVAE logra un mayor ajuste de la forma de la distribución para *cargo*, pero empeora para *passenger* en comparación con las anteriores gráficas. Y

CTGAN mejora la forma de la distribución para *cargo* en comparación con la generación sin agrupación, pero introduce una mayor dispersión de los datos.

A continuación, se muestran las estadísticas descriptivas del parámetro *to_bow* de ambos tipos de buques: *cargo* en la Tabla 4-7 y *passenger* en la Tabla 4-8.

		<i>count</i>	<i>mean</i>	<i>std</i>	<i>min</i>	25%	50%	75%	<i>max</i>
<i>Cargo</i>	Real	22097	131.954	64.9193	0.0	79.0000	144.00	173.00	300.000
	TVAE	24353	135.404	62.2397	0.0	82.6152	140.203	172.325	291.365
	<i>Gaussian Copula</i>	24353	130.564	66.4555	0.1175	77.5864	127.519	180.606	299.219
	CTGAN	24353	124.616	74.2399	0.0	57.8491	142.4834	181.6321	291.255

Tabla 4-7: Resultados de las estadísticas descriptivas de *to_bow* para *cargo* con agrupación

		<i>count</i>	<i>mean</i>	<i>std</i>	<i>min</i>	25%	50%	75%	<i>max</i>
<i>Passenger</i>	Real	2912	23.2819	16.7964	0.0	10.00	20.00	32.00	154.00
	TVAE	3209	20.3197	12.1254	1.0409	9.9305	17.1224	30.4926	57.219
	<i>Gaussian Copula</i>	3209	24.7007	16.7744	0.0784	12.2129	20.8524	33.7718	118.15
	CTGAN	3209	23.1235	17.7931	0.0	7.9242	21.7722	34.2880	153.20

Tabla 4-8: Resultados de las estadísticas descriptivas de *to_bow* para *passenger* con agrupación

Para *cargo*, TVAE genera una media (135.404 m) más próxima a la real (131,954 m) que la generada sin agrupación por *shiptype* (125,32), aunque la desviación estándar es ligeramente menor evidenciando menor variabilidad, sobre todo en los extremos. *Gaussian Copula* mejora considerablemente la media ajustándose más a la real en comparación con el caso anterior, pero la desviación estándar sigue siendo mayor que la real. Para CTGAN las estadísticas descriptivas empeoran en relación con la generación sin agrupar por *shiptype*. La media es ligeramente inferior y la desviación estándar aumenta, debido a los sobre ajustes y a la sub-representación de ciertos rangos.

Para *passenger*, TVAE genera una media inferior y una pérdida de variabilidad pues la desviación estándar es considerablemente menor también (12,12 frente a 16,79). Además, genera un valor máximo muy inferior al valor máximo real, es decir no se ajusta a los valores extremos. *Gaussian Copula* mejora la media y la desviación estándar, aproximándose a los valores reales. Y CTGAN, en este caso mejora su media siendo la más próxima y con mejor relación de valores máximos.

En comparación con la generación sin distinción por tipos de buque, TVAE se mantiene como la mejor técnica para conjuntos de datos extensos como *cargo*, aunque presenta diferencias en los valores extremos. *Gaussian Copula* mejora considerablemente su precisión con la generación por *shiptype*, pero sigue suavizando la forma de la distribución de densidad. Y CTGAN mantiene valores cercanos a los reales, pero con sobre-representación en ciertas regiones. Se ajusta mejor para la categoría *passenger*, es decir, para conjuntos de datos menores.

En cuanto a las relaciones entre variables, la Figura 4-41 y la Figura 4-42 muestran las matrices de correlación de los datos generados con TVAE, la Figura 4-43 y la Figura 4-44 las de los generados con *Gaussian Copula* y la Figura 4-45 y la Figura 4-46 las de los generados con CTGAN, para *cargo* y

passenger respectivamente. Las matrices de los datos reales se pueden ver, en la Figura 4-13 y en la Figura 4-14, presentadas en el primer análisis.

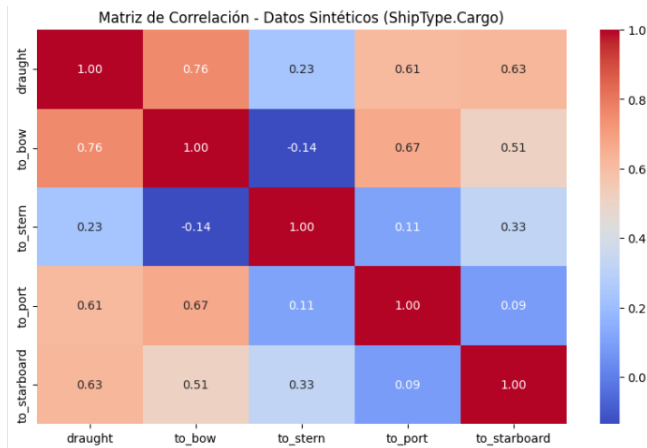


Figura 4-41: Matriz de correlación para *cargo* de *basic* con agrupación para TVAE

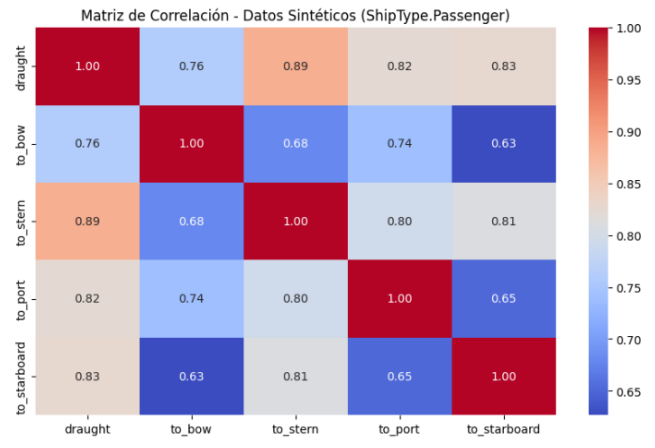


Figura 4-42: Matriz de correlación para *passenger* de *basic* con agrupación para TVAE

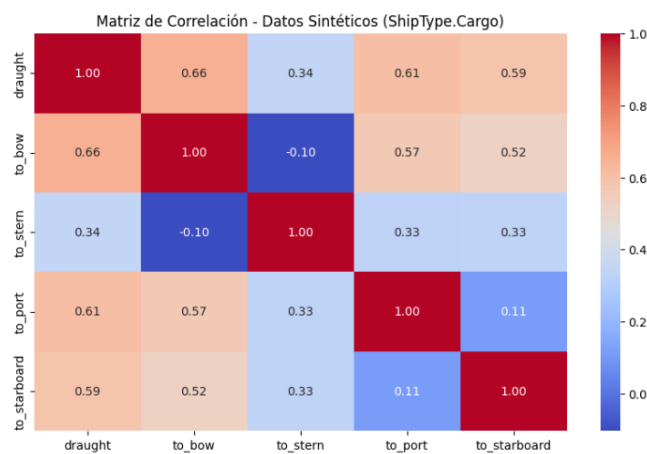


Figura 4-43: Matriz de correlación para *cargo* de *basic* con agrupación para Gaussian Copula

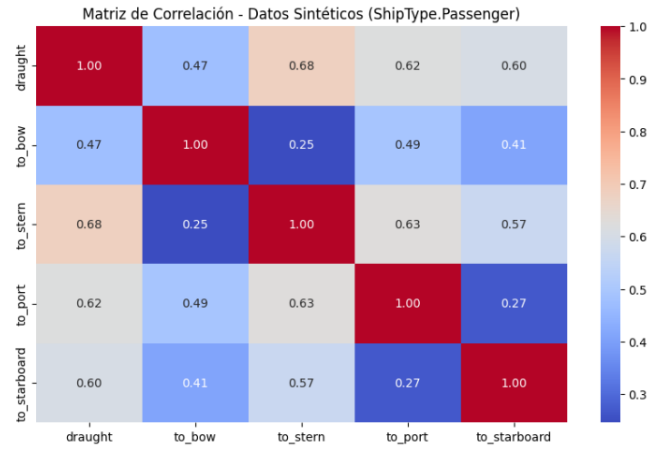


Figura 4-44: Matriz de correlación para *passenger* de *basic* con agrupación para Gaussian Copula

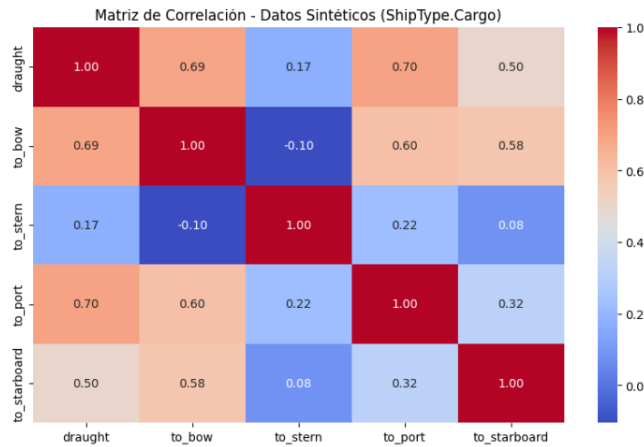


Figura 4-45: Matriz de correlación para *cargo* de *basic* con agrupación para CTGAN

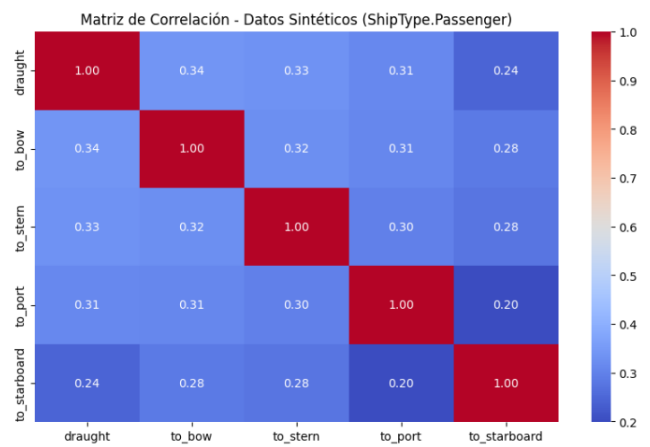


Figura 4-46: Matriz de correlación para *passenger* de *basic* con agrupación para CTGAN

TVAE se presenta como la técnica que mejor guarda las relaciones entre variables para *cargo*, si bien reduce la relación entre *to_port* y *to_stern*, pero para *passenger* varía las relaciones entre varios parámetros. *Gaussian Copula* suaviza ligeramente la intensidad de las correlaciones, pero en general, las mantiene para ambas categorías de buques. CTGAN preserva las relaciones para *cargo*, aunque muestra alguna reducción en varios parámetros, pero para *passenger* muestra una disminución considerable de las relaciones, por lo que la técnica no ha logrado capturar las dependencias entre variables.

La efectividad de cada modelo para capturar la variabilidad y las relaciones estructurales de los datos reales se representan en los gráficos PCA para *cargo* y *passenger* que se muestran de la Figura 4-47 a la Figura 4-52.

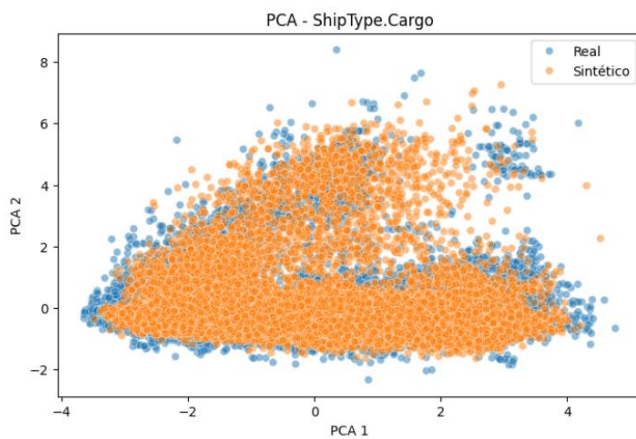


Figura 4-47: Gráfico PCA de los datos *basic* con agrupación con TVAE para *cargo*

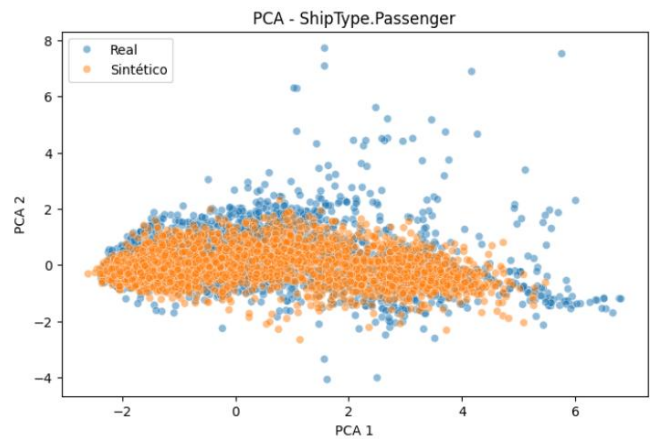


Figura 4-48: Gráfico PCA de los datos *basic* con agrupación con TVAE para *passenger*

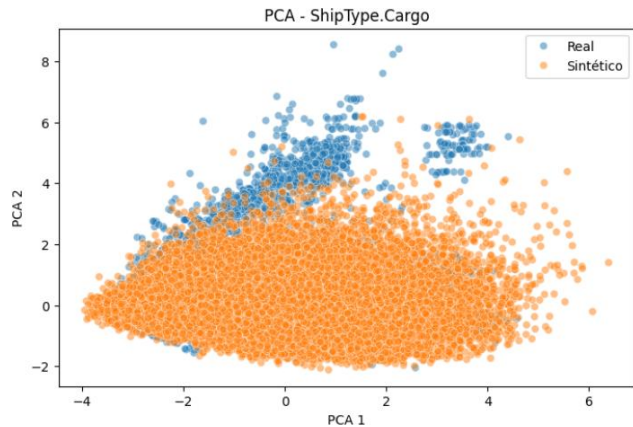


Figura 4-49: Gráfico PCA de los datos *basic* con agrupación con *Gaussian Copula* para *cargo*



Figura 4-50: Gráfico PCA de los datos *basic* con agrupación con *Gaussian Copula* para *passenger*

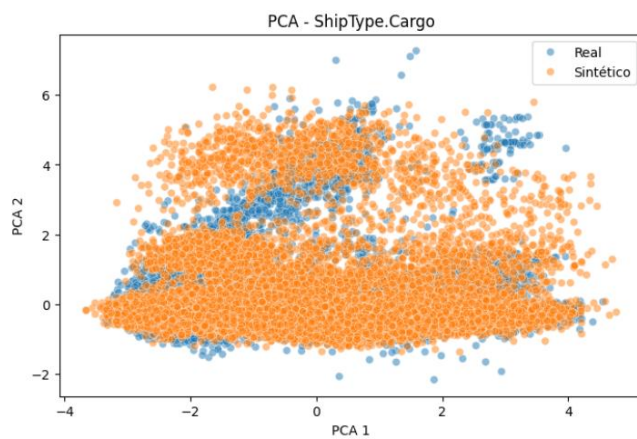


Figura 4-51: Gráfico PCA de los datos *basic* con agrupación con CTGAN para *cargo*



Figura 4-52: Gráfico PCA de los datos *basic* con agrupación con CTGAN para *passenger*

Para *cargo* las tres técnicas mejoran su representación, pero vuelve a destacar como la mejor técnica TVAE, como se muestra en la Figura 4-47, pues es la que mejor cobertura de la estructura real mantiene. *Gaussian Copula*, como se muestra en la Figura 4-49, muestra el peor desempeño frente a TVAE y CTGAN, se revela un sesgo claro y pérdida de variabilidad al observarse una separación clara entre los datos reales y sintéticos, y al encontrarse los sintéticos agrupados en la parte inferior. CTGAN, mostrada en la Figura 4-51, presenta buena superposición, pero con ligeras diferencias en la dispersión en PCA2. En algunas zonas de mayor densidad, la distribución sintética parece más dispersa que la real, lo que sugiere cierta sobre-representación en ciertas regiones.

Para *passenger*, TVAE, como se ilustra en la Figura 4-48, mejora considerablemente su representación siendo la técnica más efectiva a diferencia de la generación sin agrupación por *shiptype* donde CTGAN mostraba mejores resultados. Buena alineación en PCA1, lo que indica que el modelo ha capturado bien la principal fuente de variabilidad. En cambio, para *Gaussian Copula*, ver Figura 4-50, y CTGAN, ver Figura 4-52, se nota que los datos sintéticos están más concentrados en ciertas zonas que los reales, lo que puede indicar diferencias de variabilidad en los extremos de las distribuciones.

En cuanto a las proyecciones t-SNE de los datos reales y sintéticos generados por TVAE, *Gaussian Copula* y CTGAN, respectivamente en la Figura 4-53, Figura 4-55 y Figura 4-57 para *cargo* y en la Figura 4-54, Figura 4-56 y Figura 4-58 para *passenger*, todas las técnicas mejoran significativamente en comparación con la representación sin agrupamiento, pero en especial *Gaussian Copula*. Aun así, TVAE

y CTGAN siguen siendo las más efectivas pues los datos sintéticos se distribuyen de manera similar a los reales, lo que indica una buena representación de las estructuras locales.

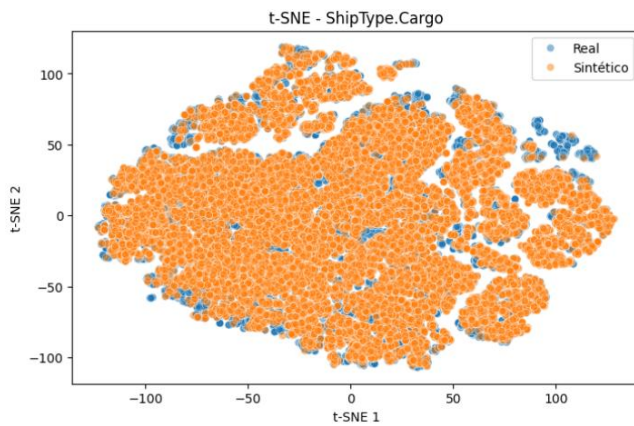


Figura 4-53: Gráfico t-SNE de los datos *basic* con agrupación con TVAE para *cargo*

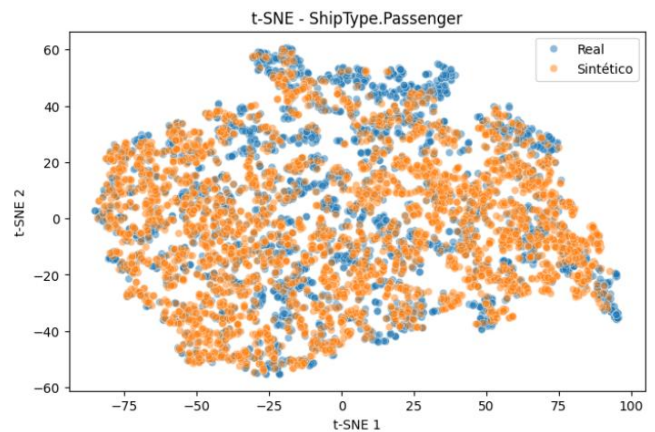


Figura 4-54: Gráfico t-SNE de los datos *basic* con agrupación con TVAE para *passenger*

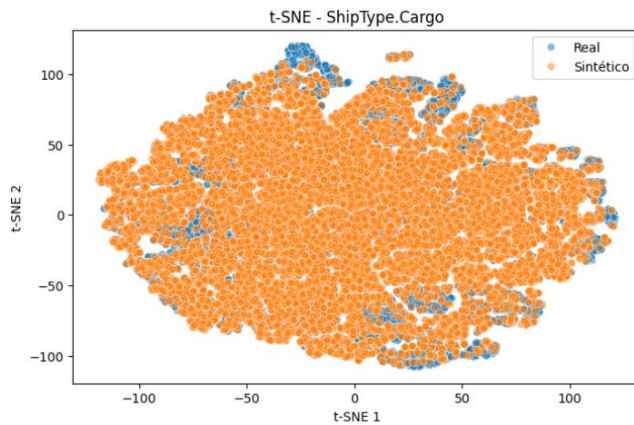


Figura 4-55: Gráfico t-SNE de los datos *basic* con agrupación con *Gaussian Copula* para *cargo*

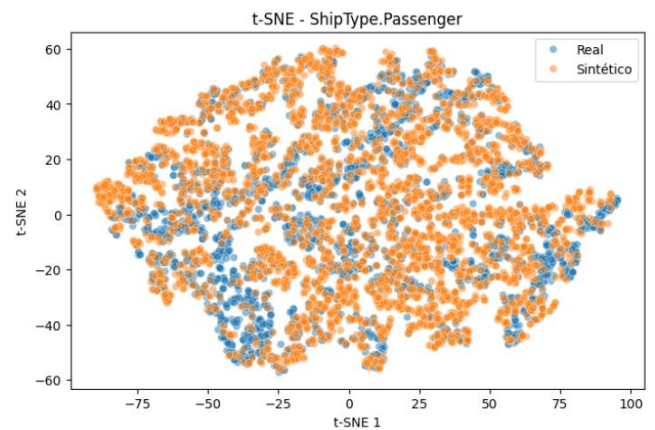


Figura 4-56: Gráfico t-SNE de los datos *basic* con agrupación con *Gaussian Copula* para *passenger*

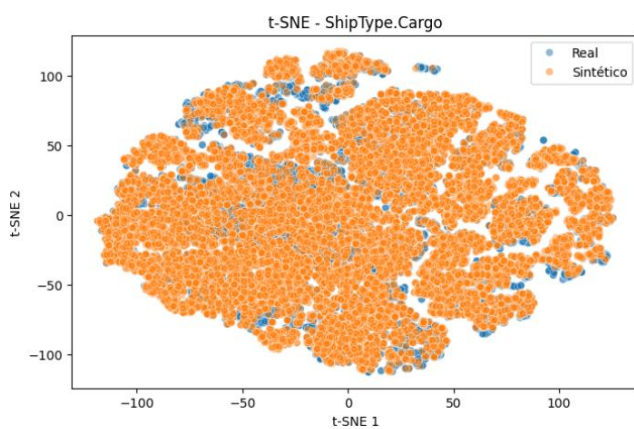


Figura 4-57: Gráfico t-SNE de los datos *basic* con agrupación con CTGAN para *cargo*

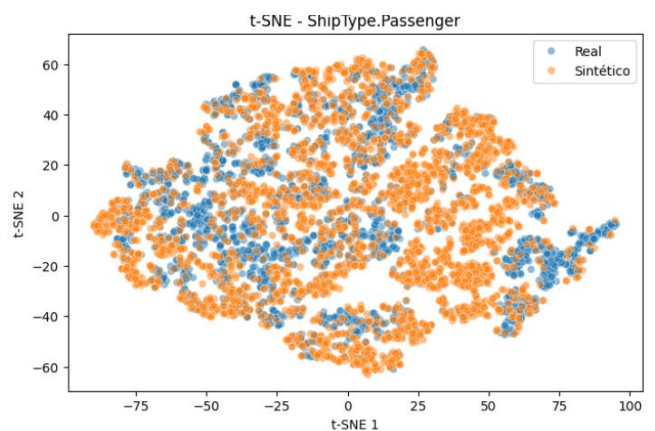


Figura 4-58: Gráfico t-SNE de los datos *basic* con agrupación con CTGAN para *passenger*

Tras analizar las métricas y visualizaciones para la generación de datos sintéticos agrupados por *shiptype* del *archivo final_static_data_basic.csv*, la técnica que mejor desempeño global muestra es

TVAE, pues preserva la distribución de los datos reales (aunque no para conjuntos de datos pequeños) muestra mayor alineación estructural en PCA y t-SNE, y conserva mejor las correlaciones entre variables. CTGAN y *Gaussian Copula* también mejoran su eficacia en comparación con la generación sin distinción por tipo de buque.

A continuación, se analizan los resultados obtenidos en la segunda configuración, donde se emplea el archivo *final_static_data.csv* que contiene tanto las variables dimensionales como los atributos asociados, para las tres técnicas de generación de datos sintéticos. Como se explicó en el apartado 3.4.2 de la memoria, ninguna de las tres técnicas empleadas en este trabajo es capaz de aprender relaciones matemáticas entre columnas, por ello el estudio de este conjunto de datos se limita a analizar las estadísticas descriptivas y las matrices de correlación.

La Tabla 4-9 muestra las estadísticas descriptivas del parámetro *to_bow* para la categoría *cargo*. TVAE y CTGAN presentan valores muy cercanos a los reales, pero CTGAN tiene menor dispersión en comparación con los reales. *Gaussian Copula* presenta una desviación estándar mayor a la real, introduciendo más variabilidad en los datos sintéticos. En la Tabla 4-10 para la categoría *passenger*, *Gaussian Copula* presenta una media y desviación estándar muy cercanos a los datos reales, pero con una reducción en la dispersión de los valores máximos. CTGAN conserva mejor los valores extremos, pero presenta mayor dispersión al poseer una desviación estándar mayor. Y TVAE mantiene valores similares a los reales con una desviación estándar próxima a la real.

		<i>count</i>	<i>mean</i>	<i>std</i>	<i>min</i>	25%	50%	75%	<i>max</i>
Cargo	Real	22097	131.954	64.9193	0.0	79.0000	144.00	173.00	300.000
	TVAE	24353	124.954	63.9327	0.0	75.5650	132.851	168.162	297.407
	<i>Gaussian Copula</i>	24353	129.8965	66.4262	0.3500	77.0142	125.6203	179.594	298.179
	CTGAN	24353	131.2989	62.2000	0.3393	80.8393	142.9809	173.557	276.622

Tabla 4-9: Resultados de las estadísticas descriptivas de *to_bow* para *cargo* con agrupación completo

		<i>count</i>	<i>mean</i>	<i>std</i>	<i>min</i>	25%	50%	75%	<i>max</i>
Passenger	Real	2912	23.2819	16.7964	0.0	10.00	20.00	32.00	154.00
	TVAE	3209	21.1515	15.7150	0.0	8.4826	16.6816	31.6706	131.13
	<i>Gaussian Copula</i>	3209	24.4427	16.5282	0.1027	11.8799	21.1336	33.3717	112.09
	CTGAN	3209	24.9206	20.8657	0.0	9.7024	18.5418	36.8663	154.00

Tabla 4-10: Resultados de las estadísticas descriptivas de *to_bow* para *passenger* con agrupación completo

Comparando estas estadísticas descriptivas con las de los resultados anteriores, se observa que las tres técnicas mantienen e incluso mejoran los valores. Esto indica que un mayor número de parámetros en el conjunto de datos original puede favorecer la precisión de las técnicas en la generación de datos sintéticos, pero sin que guarden las relaciones entre ellos como se demuestra en las matrices de correlación.

Las siguientes matrices muestran las correlaciones entre las variables de los datos sintéticos generados por TVAE en la Figura 4-60, *Gaussian Copula* en la Figura 4-61 y CTGAN en la Figura 4-62, para los barcos tipo *cargo*, a la izquierda, y *passenger*, a la derecha, comparándolas con las matrices de los datos reales de la Figura 4-59.

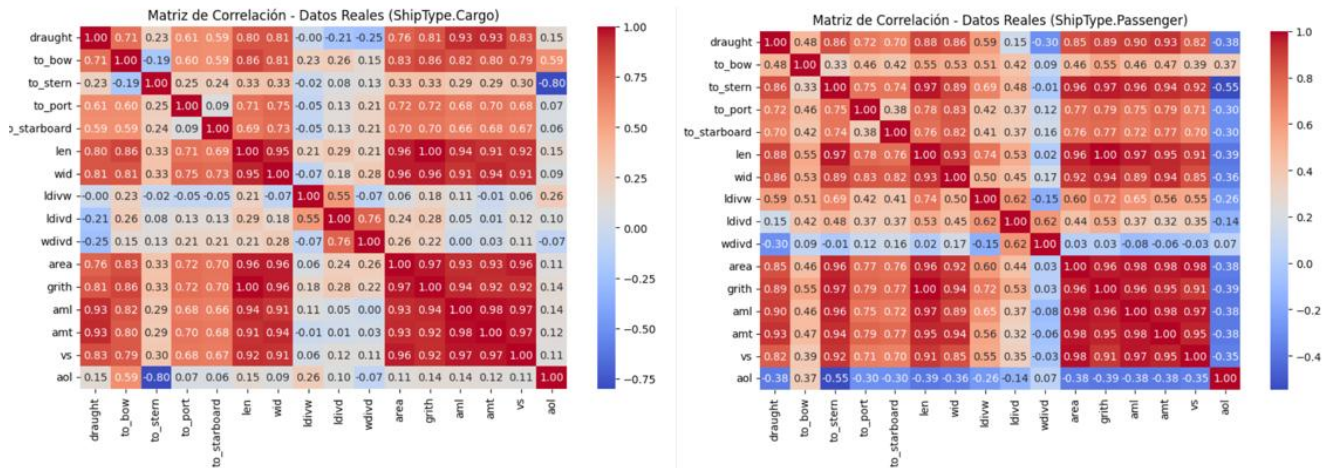


Figura 4-59: Matriz de correlación de los datos reales del archivo *final_static_data.csv*

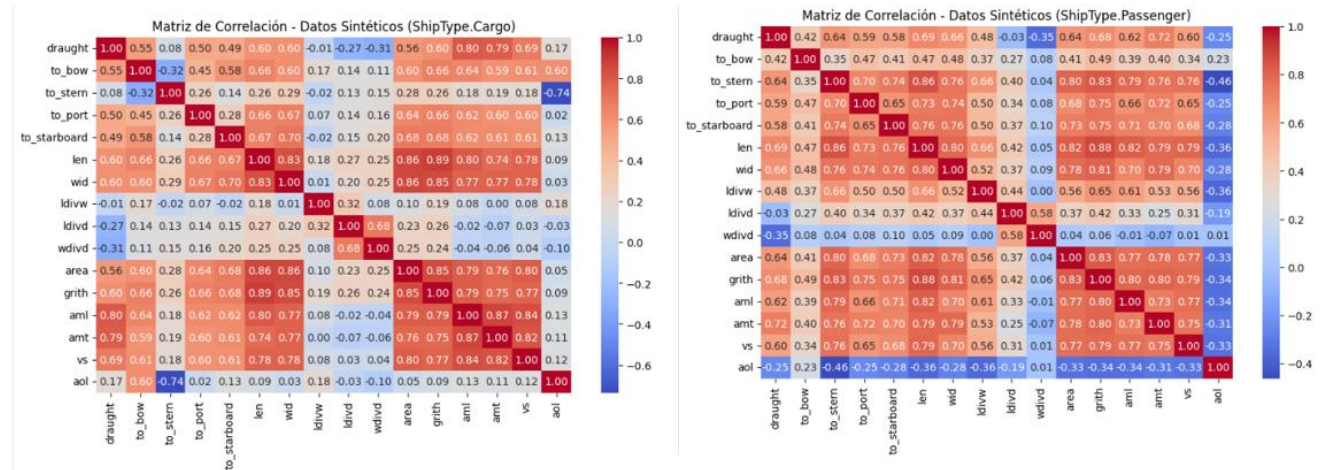


Figura 4-60: Matriz de correlación de los datos sintéticos completos con TVAE

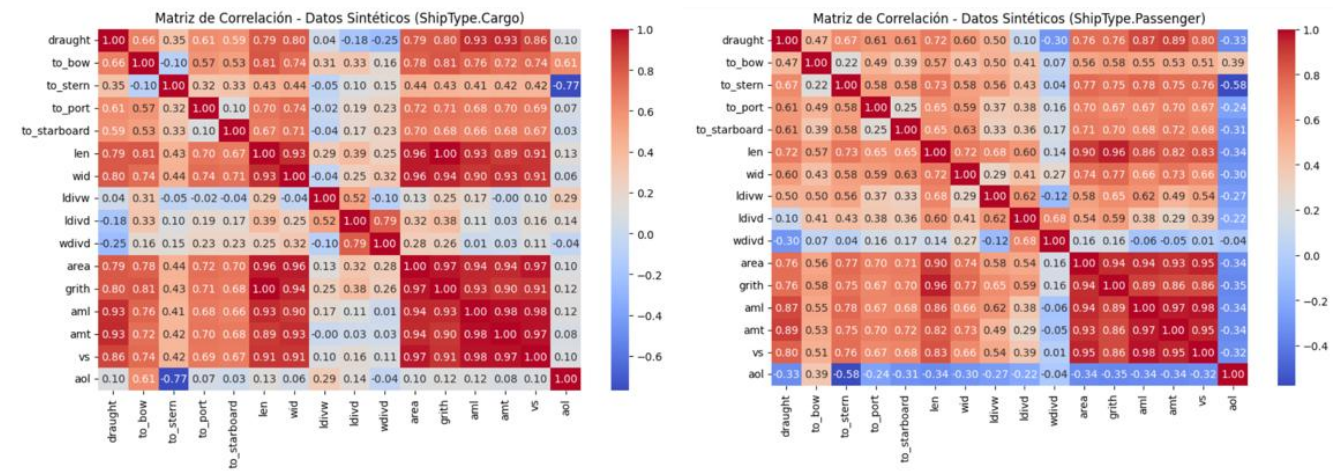


Figura 4-61: Matriz de correlación de los datos sintéticos completos con *Gaussian Copula*

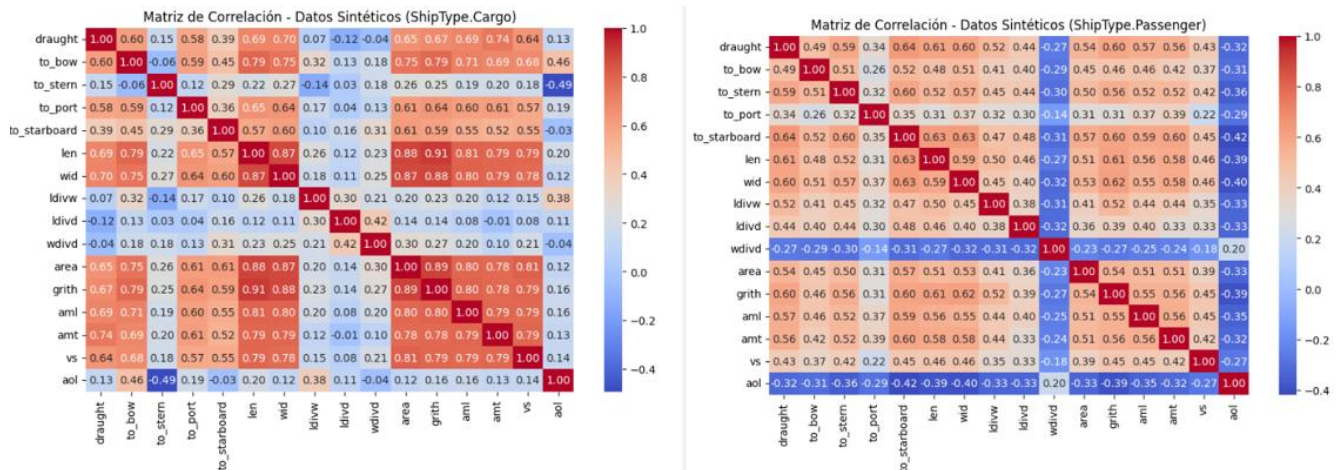


Figura 4-62: Matriz de correlación de los datos sintéticos completos con CTGAN

Las matrices de correlación de los datos reales muestran relaciones estructurales marcadas con valores en torno a 0,9 entre los atributos y las variables dimensionales. Algunas de las relaciones fuertes incluyen *len*, *wid*, *area*, *grith* y *draught*, así como la correlación negativa entre *to_bow* y *to_stern*. En cambio, las tres técnicas de generación de datos sintéticos disminuyen significativamente los valores. TVAE disminuye en 0,1 la mayoría de correlaciones. En cambio, *Gaussian Copula* mantiene en mejor medida la intensidad de las correlaciones y CTGAN disminuye hasta a la mitad la magnitud de las correlaciones.

También se comprueba en qué medida TVAE ha sido capaz de mantener la relación $len = to_bow + to_stern$ en los datos sintéticos mediante la comparación de los valores *len* generados y los valores *len* esperados de la suma de *to_bow* y *to_stern* sintéticos. En la Figura 4-63 se visualiza el *scatterplot* (gráfico de dispersión) de dicha comparación.

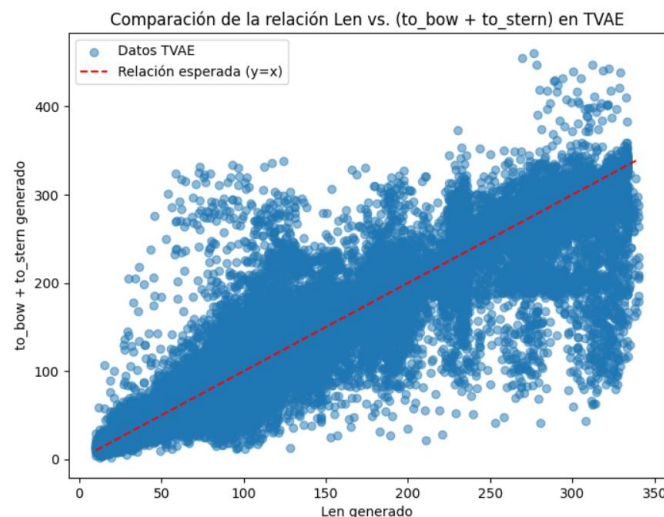


Figura 4-63: Comparación *len* vs (*to_bow* + *to_stern*) sintéticos en TVAE

Si la técnica de generación de datos hubiera aprendido correctamente la relación, todos los puntos estarían alineados sobre la línea roja, pero la gran dispersión de los datos sintéticos indica que TVAE no mantiene la relación matemática esperada. Además, se observan casos en los que $to_bow + to_stern$ es mucho mayor que *len* y viceversa.

La Figura 4-64 muestra la estadística del error definido como: $error = len - (to_bow + to_stern)$. Se observan valores de $to_bow + to_stern$ de hasta 266 unidades menor o 264 unidades mayor que *len* y una desviación estándar significativa que indica una variabilidad muy alta del error.

```

Estadísticas del error en la relación Len:
count    49997.000000
mean      2.660431
std       32.610961
min       -266.048986
25%       -7.873888
50%        0.870572
75%       11.089211
max       264.242748
Name: error_len, dtype: float64

```

Figura 4-64: Estadísticas del error en la relación *len*

En la Figura 4-65 se observa el histograma de errores donde la mayoría de errores se concentran cerca de 0, pero hay una gran cantidad de valores en los extremos. Se confirma que, aunque en algunos casos, haya podido mantener la relación, en muchos otros los valores generados por TVAE no respetan la coherencia matemática de los datos reales.

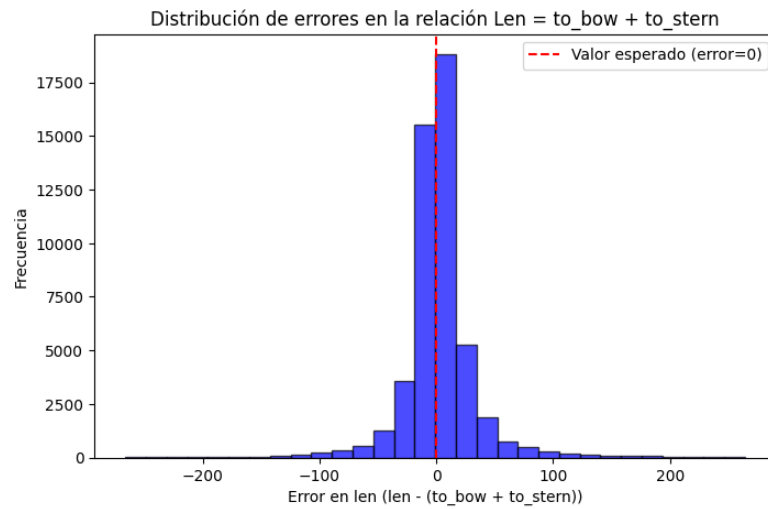


Figura 4-65: Distribución del error en *len*

Esta prueba demuestra que TVAE no es capaz de capturar las relaciones matemáticas entre columnas, generando valores inconsistentes. Dado que las otras técnicas de generación de datos sintéticos tampoco fueron capaces de aprender las relaciones matemáticas al realizarse la misma prueba, se concluye que la generación de datos sintéticos a partir del archivo *final_static_data.csv* debe ser descartada. Por el mismo motivo, no se analizarán los resultados de la tercera configuración donde se generaban solo los atributos a partir del archivo *fnal_static_data.csv*, pues tampoco guardan las relaciones entre columnas.

Para garantizar datos sintéticos coherentes, la alternativa viable es calcular *len* a partir de *to_bow* y *to_stern* después de la generación de datos sintéticos, en lugar de intentar que las técnicas de generación lo aprendan. Esto asegurará que las relaciones estructurales se mantengan de manera correcta.

Se emplean las mismas métricas y estadísticas para el análisis de los atributos calculados a partir de los datos sintéticos generados por agrupación de tipo de barco del archivo *final_static_data_basic.csv*. Se evalúan los atributos *len*, al ser un atributo base para el cálculo de otros, y *ldivw* que es una relación clave entre eslora y manga, para el conjunto de datos sintéticos en cada técnica.

Las siguientes gráficas muestran la comparación entre las distribuciones de densidad de los datos reales y los datos sintéticos generados por TVAE en la Figura 4-66, *Gaussian Copula* en la Figura 4-67 y CTGAN en la Figura 4-68 para las variables *len* a la izquierda y *ldivw* a la derecha.

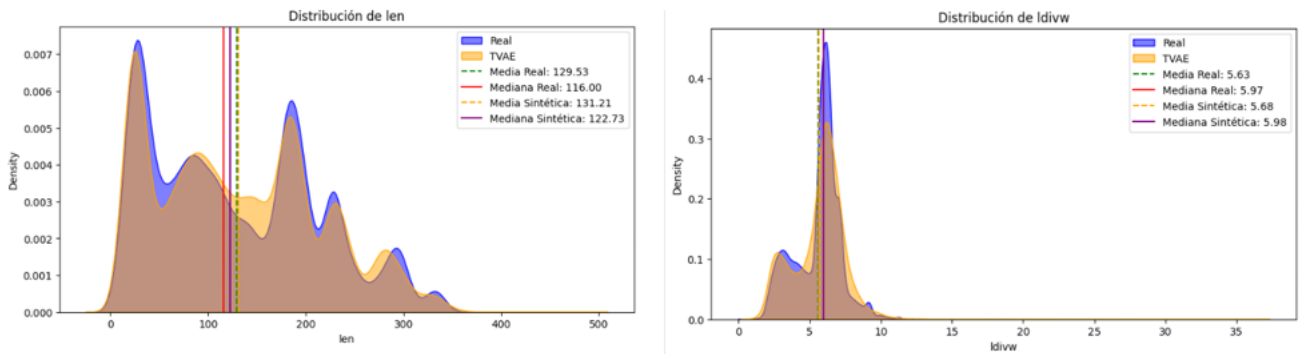


Figura 4-66: Funciones densidad del conjunto de datos con atributos calculados con TVAE

TVAE muestra una distribución sintética de *len* que se asemeja bastante a la real, aunque con ligeras desviaciones en la densidad de los valores intermedios. Para *ldivw*, la forma de la distribución es similar, aunque con menor densidad en el pico.

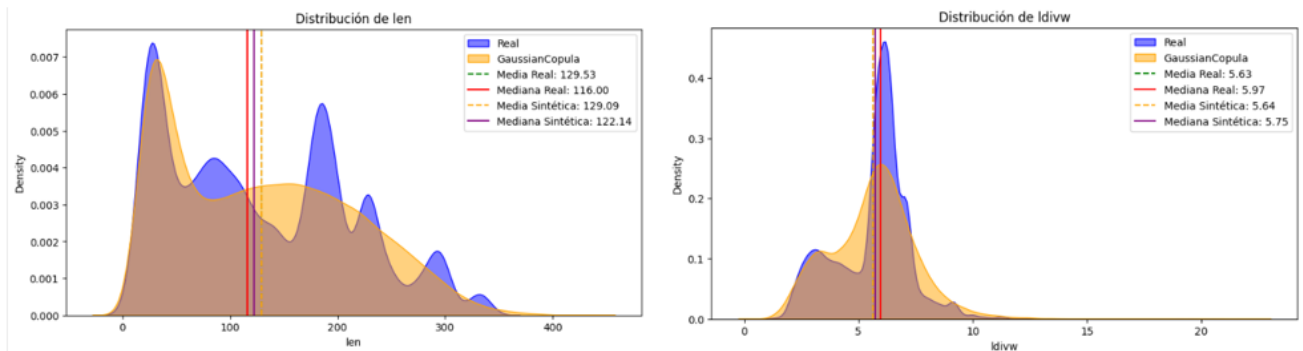


Figura 4-67: Funciones densidad del conjunto de datos con atributos calculados con *Gaussian Copula*

En *Gaussian Copula*, ambas distribuciones presentan diferencias mínimas entre media y mediana, pero la forma de la distribución sintética muestra mayor dispersión en valores altos para *len* y mayor desviación en los extremos para *ldivw*.

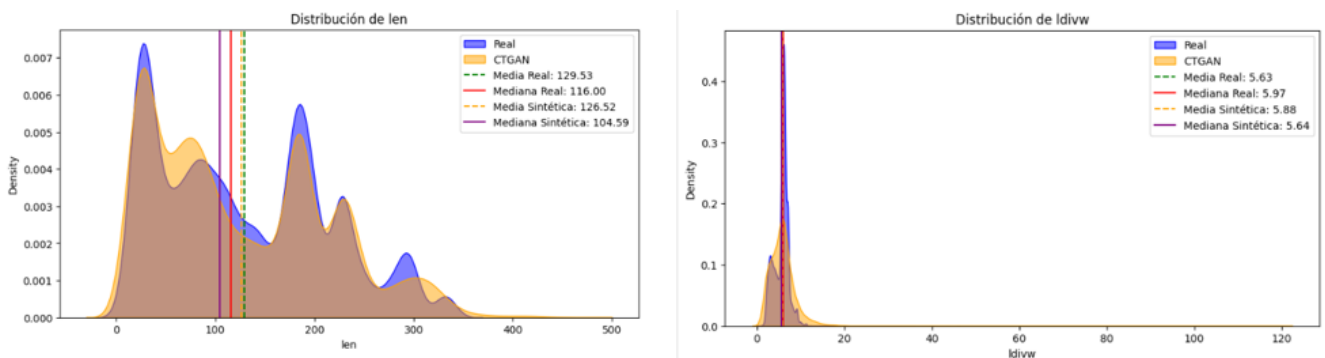


Figura 4-68: Funciones densidad del conjunto de datos con atributos calculados con CTGAN

Con CTGAN, la distribución sintética de *len* no sigue fielmente la distribución real, mostrando una mayor concentración en ciertos valores específicos. Para *ldivw*, CTGAN no sigue la forma de la distribución real generando mayor densidad en valores diferentes a la distribución real y mayor dispersión de los datos.

En función de los gráficos de distribuciones, TVAE se presenta como la técnica que mejor preserva las distribuciones originales para ambas variables. *Gaussian Copula* logra una aproximación aceptable, pero suavizada, como en anteriores configuraciones. Y CTGAN presenta las mayores desviaciones en ambas variables.

La Tabla 4-11 y la Tabla 4-12 muestran la comparación de las métricas de las estadísticas clave entre los datos reales y los datos sintéticos generados por las tres técnicas para ambas variables.

		Media	Mediana	std	min	max
len	Real	129.53022	116.000	84.7356	10.00	340.00
	TVAE	131.2105	122.7306	83.8866	3.6074	480.3380
	<i>Gaussian Copula</i>	129.0889	122.1365	84.4774	0.8573	429.1964
	CTGAN	126.5234	104.5908	88.3463	0.1830	469.8302

Tabla 4-11: Resultados de las estadísticas descriptivas de *len* del conjunto de datos sintéticos con atributos

Para *len*, TVAE y *Gaussian Copula* generan valores de media y mediana cercanos a los reales. En cambio, CTGAN muestra mayor diferencia en la media y mediana, lo que demuestra alteraciones en la distribución de los valores. Además, CTGAN muestra mayor variabilidad pues es la técnica con mayor desviación estándar. En cuanto a los valores extremos, las tres técnicas generan valores máximos muy altos en comparación con los reales, pero CTGAN genera un rango significativamente más amplio del rango real con valores máximos de 469 metros frente a 340 metros y valores mínimos de 0,18 metros frente a 10 metros de los datos originales.

		Media	Mediana	std	min	max
ldivw	Real	5.6306	5.9666	1.6596	0.5405	19.9946
	TVAE	5.6763	5.9776	1.8532	0.7607	36.6944
	<i>Gaussian Copula</i>	5.6406	5.7468	1.8972	0.3828	22.3895
	CTGAN	5.8826	5.6355	3.2091	0.0254	121.4309

Tabla 4-12: Resultados de las estadísticas descriptivas de *ldivw* del conjunto de datos sintéticos con atributos

Para *ldivw*, las tres técnicas generan valores de media y mediana muy ajustados a los valores reales, pero CTGAN tiene la mayor desviación estándar, indicando que los valores generados tienen mayor dispersión. Los valores máximos y mínimos evidencian la mayor dispersión de CTGAN con valores extremos anómalos con un mínimo en 0,025 metros frente a los 0.54 metros reales y un máximo en 121.43 metros frente a los 19.99 metros reales. En cambio, TVAE y *Gaussian Copula* mantienen una dispersión más similar con rangos más cercanos a los datos reales.

Las estadísticas descriptivas coinciden con los análisis de los gráficos de densidad en que la técnica que mejor ajusta los valores de media, mediana y desviación estándar con los datos originales es TVAE. *Gaussian Copula* también ofrece una buena aproximación, pero con ligeras variaciones en la dispersión. Y CTGAN es la técnica que más valores atípicos genera y mayor dispersión.

A continuación, se analizan las matrices de correlación para cada técnica, representándose a la izquierda la matriz de los datos reales y a la derecha la matriz de cada técnica, donde la Figura 4-69 muestra la matriz de TVAE, la Figura 4-70 la matriz generada por *Gaussian Copula* y la Figura 4-71 la de los datos generados con CTGAN.

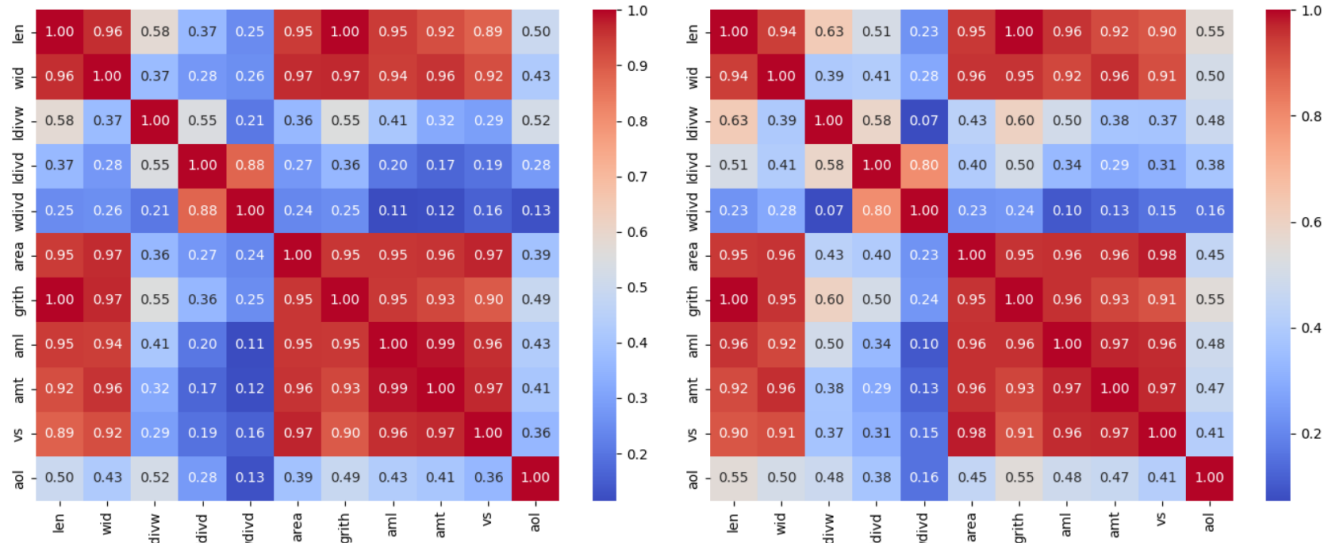


Figura 4-69: Matriz de correlación del conjunto de datos con atributos calculados con TVAE

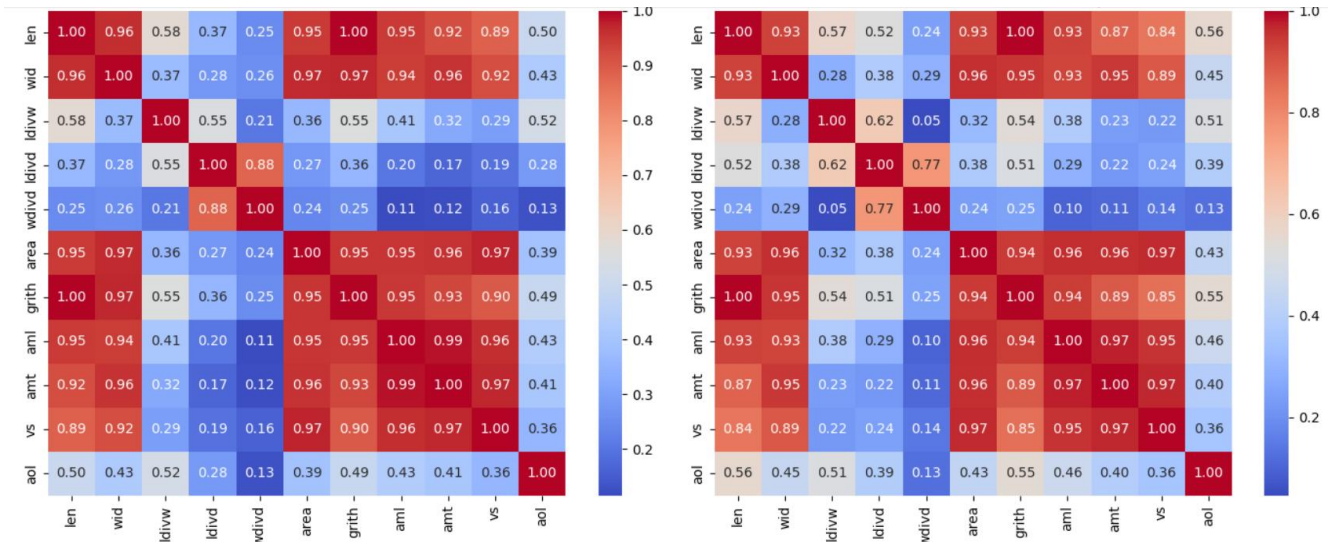


Figura 4-70: Matriz de correlación del conjunto de datos con atributos calculados con *Gaussian Copula*

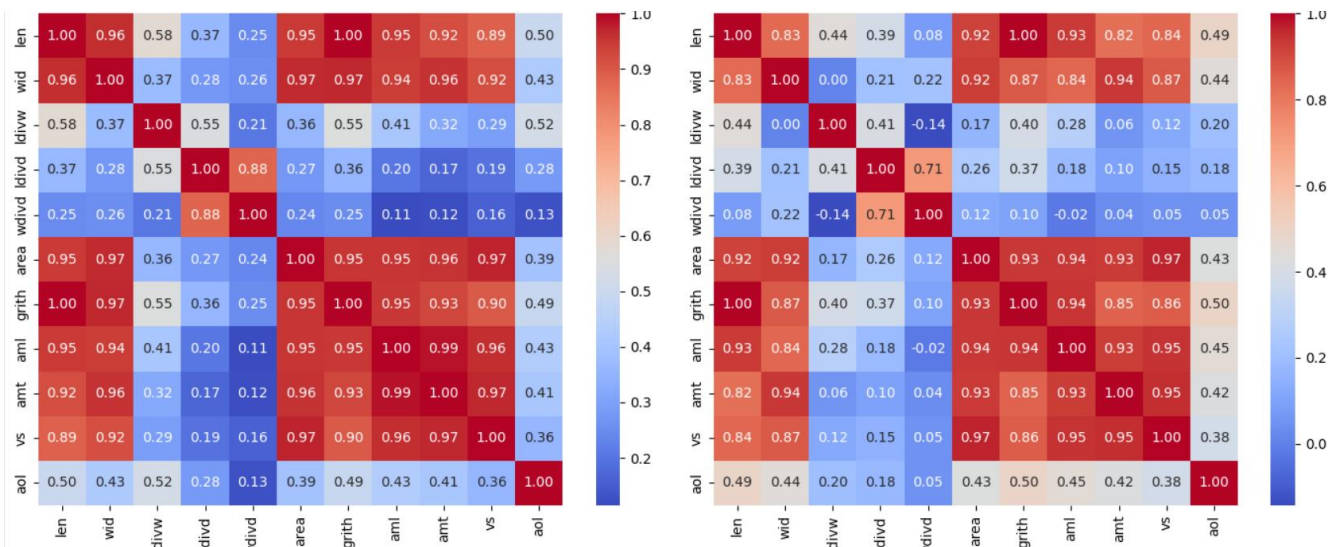


Figura 4-71: Matriz de correlación del conjunto de datos con atributos calculados con CTGAN

TVAE es la técnica que mejor preserva la estructura de correlación de los datos reales. Mantiene relaciones similares en términos de intensidad y dirección, especialmente en *len*, *wid*, *area* y *grith*. *Gaussian Copula* también logra mantener las relaciones entre variables, aunque con variaciones de intensidad en variables secundarias. CTGAN es la técnica que peor conserva las relaciones con correlaciones inesperadas como 0.0 entre *ldivw* y *wid* o incluso negativas entre *ldivw* y *wdivd*.

En cuanto al análisis de reducción de dimensionalidad con PCA y t-SNE, la Figura 4-72, la Figura 4-73 y la Figura 4-74 muestran los gráficos PCA a la izquierda y los gráficos t-SNE a la derecha para TVAe, *Gaussian Copula* y CTGAN respectivamente.

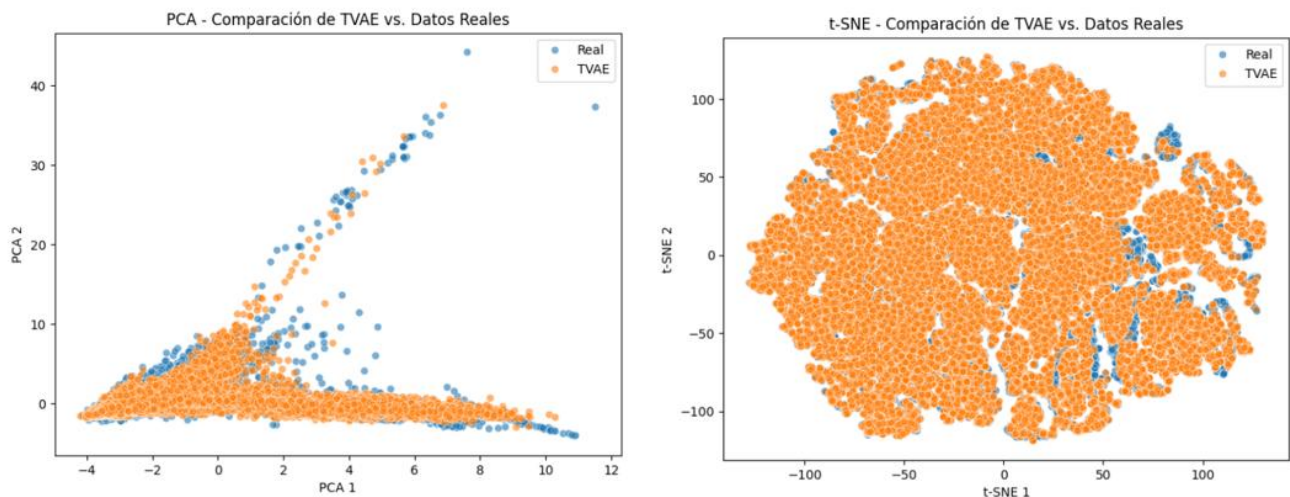


Figura 4-72: Gráficos PCA y t-SNE del conjunto de datos sintéticos con atributos calculados para TVAe

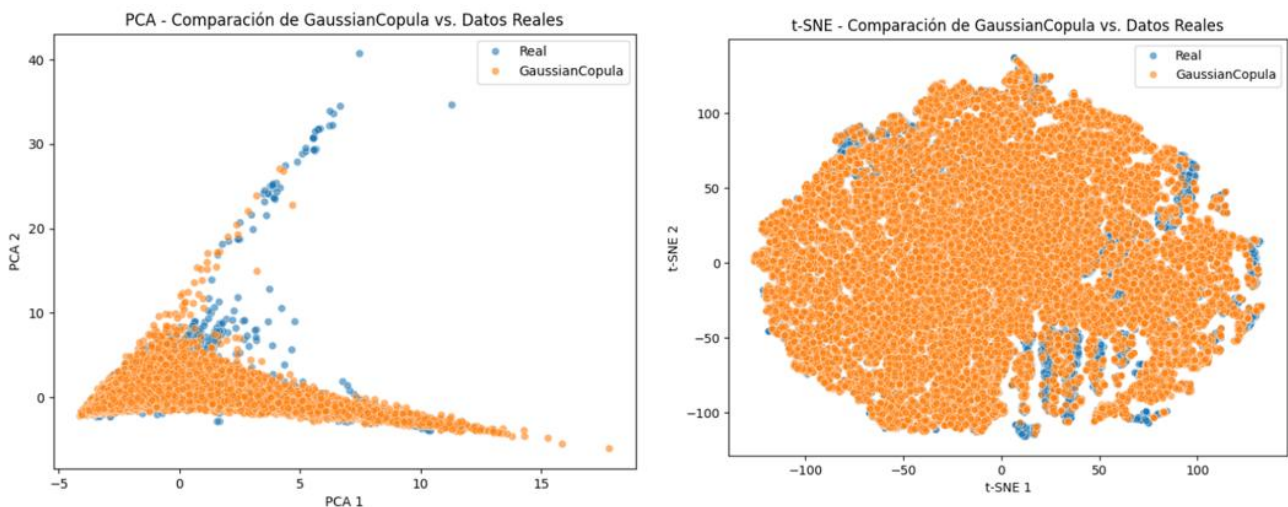


Figura 4-73: Gráficos PCA y t-SNE del conjunto de datos sintéticos con atributos calculados para *Gaussian Copula*

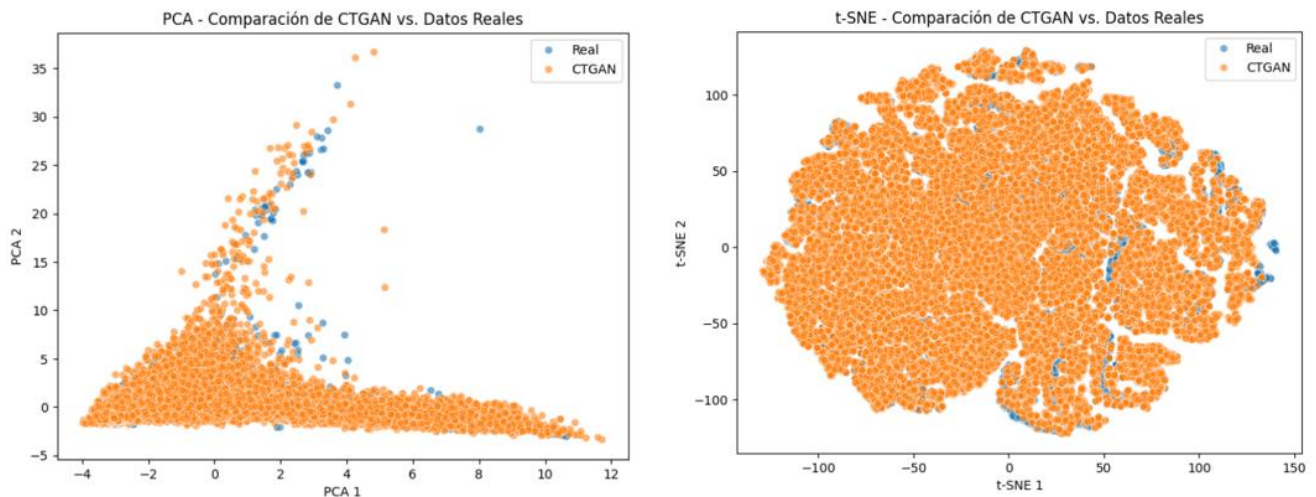


Figura 4-74: Gráficos PCA y t-SNE del conjunto de datos sintéticos con atributos calculados para CTGAN

El Análisis de Componentes Principales muestra que TVAE es la técnica que mejor conserva la variabilidad de los datos originales, con un solapamiento significativo entre ambas distribuciones. *Gaussian Copula* mantiene una estructura similar, pero con variaciones en los extremos. Y CTGAN introduce mayor dispersión en PCA2. Las gráficas t-SNE confirman que TVAE logra la mejor representación global, con una alta coincidencia entre los valores sintéticos y los reales. *Gaussian Copula* mantiene un buen solapamiento, pero con zonas de mayor concentración. Y CTGAN presenta un comportamiento similar a *Gaussian Copula*, con una alta densidad de puntos sintéticos en ciertas zonas.

La Tabla 4-13 muestra un resumen de la mejor técnica en función del análisis realizado y el conjunto de datos examinado, para aclarar los resultados obtenidos en las diferentes configuraciones.

	KDE	Estadísticas descriptivas	Matrices correlación	PCA	t-SNE	Chi-cuadrado	KS-test
Basic sin agrupar por shiptype	TVAE CTGAN	TVAE	TVAE	TVAE cargo CTGAN passenger	TVAE cargo CTGAN passenger	<i>Gaussian Copula</i>	CTGAN
Basic por shiptype	TVAE cargo CTGAN passenger	TVAE cargo CTGAN passenger	TVAE <i>Gaussian Copula</i>	TVAE CTGAN	TVAE	TVAE <i>Gaussian Copula</i> CTGAN	TVAE
Completo	-	TVAE	<i>Gaussian Copula</i>	-	-	TVAE <i>Gaussian Copula</i> CTGAN	X
Basic por shiptype con atributos calculados	TVAE	TVAE <i>Gaussian Copula</i>	TVAE	TVAE	TVAE	TVAE <i>Gaussian Copula</i> CTGAN	X

Tabla 4-13: Resumen de los resultados obtenidos del análisis de calidad de cada conjunto de datos

El análisis de la Tabla 4-13 muestra que TVAE es la técnica más consistente, destacando en la mayoría de métricas: KDE, estadísticas descriptivas, PCA, *t-SNE* y correlación. CTGAN también obtiene buenos resultados, aunque su rendimiento es más variable según el conjunto de datos. *Gaussian Copula*, por el contrario, sobresale solo en las pruebas de correlación del conjunto de datos sintéticos completo y en la prueba de Chi-cuadrado con el conjunto de datos sintéticos *basic* sin agrupar por tipo de buque, no obtiene buenos resultados en las métricas basadas en reducción dimensional, ni en las estadísticas.

Respecto a los conjuntos de datos, los que se generaron por *shiptype* (especialmente los del grupo *Basic* por *shiptype* y *Basic* con atributos calculados) muestran un rendimiento general superior, con más técnicas destacando en múltiples métricas. Por tanto, la segmentación por *shiptype* y el cálculo posterior de atributos parecen estrategias clave para mejorar la calidad sintética.

4.3.3 Rendimiento en aprendizaje automático

En esta sección se muestran los resultados obtenidos en la validación de la utilidad de los diferentes conjuntos de datos sintéticos generados por las tres técnicas TVAE, *Gaussian Copula* y CTGAN. Se contrastan los resultados obtenidos tras entrenar el algoritmo de *Random Forest* con los datos originales como punto de referencia (TTR), con los resultados obtenidos al emplear los diferentes conjuntos de datos sintéticos generados con las tres técnicas, como datos de entrenamiento. Se evalúan las diversas métricas: *Accuracy*, *F1-score*, *Precision*, *Recall* y AUC-ROC, explicadas en la Tabla 4-2, para determinar la efectividad de cada técnica y la influencia del tipo de *dataset* sintético utilizado en la eficiencia del modelo de aprendizaje automático.

Para el *Train and Test on Real* (TTR), se emplea un 75% de los datos reales para entrenamiento y un 25% para prueba. Para el *Train on Synthetic, Test on Real* (TSTR), se emplea el 100% de los datos sintéticos para el entrenamiento y el 25% de los reales para test. La Tabla 4-14 muestra las métricas obtenidas de los diferentes entrenamientos del algoritmo: TTR y TSTR para cada técnica de generación de datos sintéticos y para cada *dataset* generado. Se comparan los dos *datasets* de datos reales: el conjunto básico (*final_static_basic.csv*) y el extendido que incluye los atributos (*final_static_data.csv*). Y se comparan los datos sintéticos generados a partir de dichos archivos para cada técnica de generación de datos sintéticos (TVAE, CTGAN y *Gaussian Copula*):

- Sin agrupar: los datos sintéticos generados del archivo *final_static_data_basic.csv* sin distinguir clases.
- Por *shiptype*: conjunto sintético a partir del archivo *final_static_data_basic.csv*, pero separado para cada tipo de buque.
- Completo: conjunto de datos sintéticos a partir del archivo *final_static_data.csv*, es decir el que contiene atributos. A pesar de que se demostró que este conjunto de datos sintéticos no respeta las relaciones entre columnas se emplea en el entrenamiento del modelo para comparar resultados con el *dataset* sintético *basic* con atributos calculados.
- *Basic* + atributos: se compara el uso de atributos derivados frente al uso de datos completamente generados.

Origen Datos	Técnica	Dataset generado	Accuracy	F1-score	Precision	Recall	AUC-ROC
Final_static_data_basic.csv	Reales		0.8345	0.8325	0.8337	0.8345	0.9695
	TVAE	Sin agrupar	0.7061	0.6731	0.7121	0.7061	0.8999
		Por shiptype	0.7339	0.7346	0.7358	0.7339	0.9332
	Gaussian Copula	Sin agrupar	0.4815	0.4102	0.4183	0.4815	0.6804
		Por shiptype	0.6451	0.6486	0.6555	0.6451	0.8982
	CTGAN	Sin agrupar	0.6222	0.6303	0.6579	0.6222	0.8962
		Por shiptype	0.7075	0.7075	0.7057	0.7075	0.9109
Final_static_data.csv	Reales		0.85179	0.8506	0.8513	0.8517	0.9739
	TVAE	Completo	0.7459	0.7484	0.7556	0.7459	0.9394
		Basic + atributos	0.7496	0.7504	0.7526	0.7496	0.9384
	Gaussian Copula	Completo	0.6963	0.6735	0.6866	0.6963	0.9121
		Basic + atributos	0.6621	0.6664	0.6768	0.6621	0.9071
	CTGAN	Completo	0.7624	0.7568	0.7573	0.7624	0.9374
		Basic + atributos	0.7147	0.7116	0.7124	0.7147	0.9179

Tabla 4-14: Resultados de las métricas de utilidad

El rendimiento del modelo entrenado con los datos reales de *final_static_data.csv*, que contiene las variables dimensionales y los atributos, logra métricas ligeramente superiores en comparación con el entrenamiento con el archivo *final_static_data_basic.csv*, que solo contiene las variables dimensionales. Como es de esperar, usar más atributos mejora la clasificación. El rendimiento del modelo con datos reales sirve de línea base, ya que refleja el rendimiento ideal que el modelo es capaz de alcanzar.

Los resultados obtenidos tras entrenar con los *dataset* generados sin agrupar por tipo de buque y los generados por *shiptype* con cada técnica TVAE, *Gaussian Copula* y CTGAN del archivo *final_static_data_basic.csv*, muestran que agrupar por *shiptype* mejora significativamente el rendimiento del modelo en todas las técnicas.

- TVAE mejora el valor de *Accuracy* de 0,7061 sin agrupación a 0.7339 con agrupación y con un AUC-ROC que mejora de 0.8999 a 0.9332.
- CTGAN mejora de 0.6222 a 0.7075 en *Accuracy*, con un incremento en AUC-ROC de 0.8962 a 0.9109.
- *Gaussian Copula* mejora *Accuracy* de 0,4815 a 0,6451 pero presenta el peor rendimiento en comparación con las otras dos técnicas.

Estos resultados sugieren que TVAE es la mejor técnica de generación de datos sintéticos cuando se agrupan por *shiptype*.

En cuanto a los resultados obtenidos de la comparación entre el dataset sintético completo y el basic con atributos calculados generados por *shiptype* a partir del archivo *final_static_data.csv* con las tres técnicas, muestran que:

- TVAE obtiene un rendimiento similar para ambos conjuntos de datos,
- *Gaussian Copula* vuelve a presentar los peores rendimientos para ambos conjuntos de datos, siendo similares entre ambos *datasets*.
- CTGAN muestra un rendimiento mayor de las predicciones para el conjunto de datos sintéticos completo.

Conforme a los resultados, para el conjunto de datos sintéticos completo, CTGAN obtiene el mejor rendimiento superando a TVAE. En cambio, para el conjunto de datos sintéticos *basic* con atributos TVAE muestra mejores resultados que CTGAN.

Para una mejor comprensión visual, la Figura 4-75, la Figura 4-76, la Figura 4-77, la Figura 4-78 y la Figura 4-79 muestran los gráficos de barras comparativos del *Accuracy*, *F1-score*, *Precision*, *Recall* y AUC-ROC, respectivamente, para cada técnica y conjunto de datos sintéticos empleado en el entrenamiento.

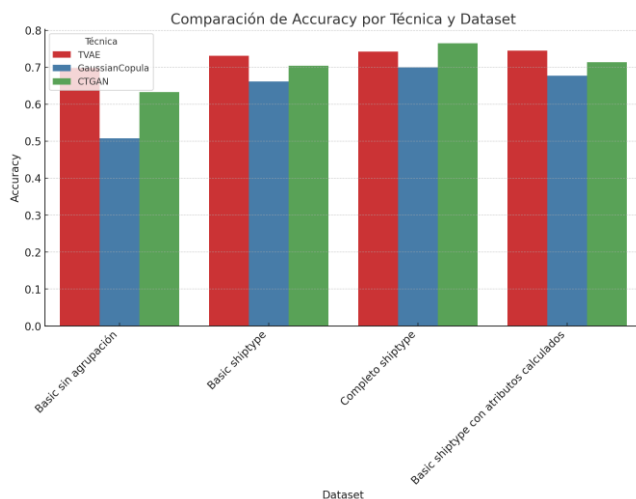


Figura 4-75: Comparación de *Accuracy* por técnica y *dataset*

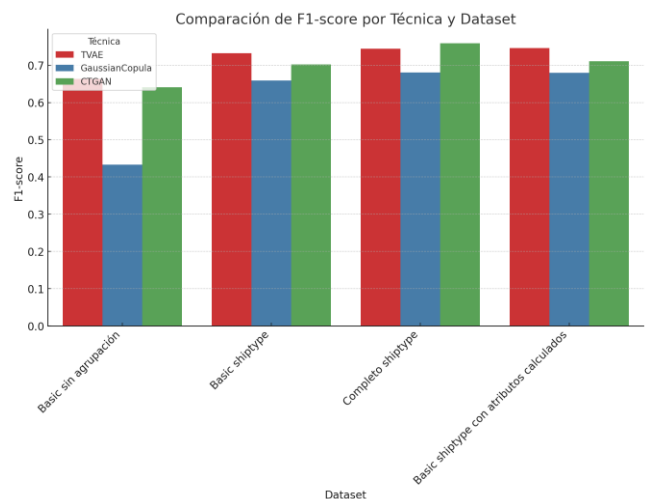


Figura 4-76: Comparación de *F1-score* por técnica y *dataset*

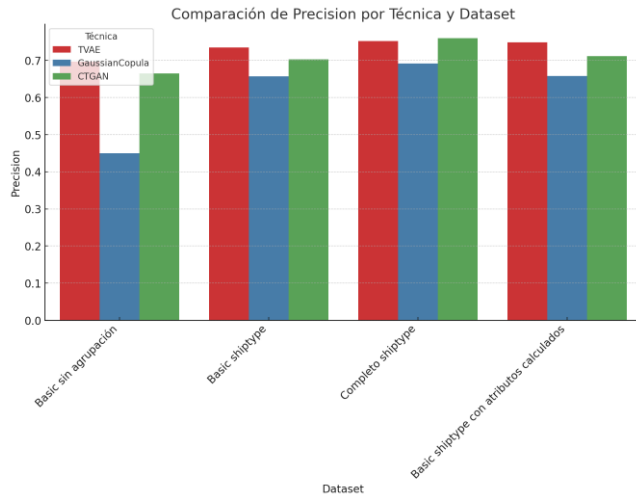


Figura 4-77: Comparación de *Precision* por técnica y *dataset*



Figura 4-78: Comparación de *Recall* por técnica y *dataset*

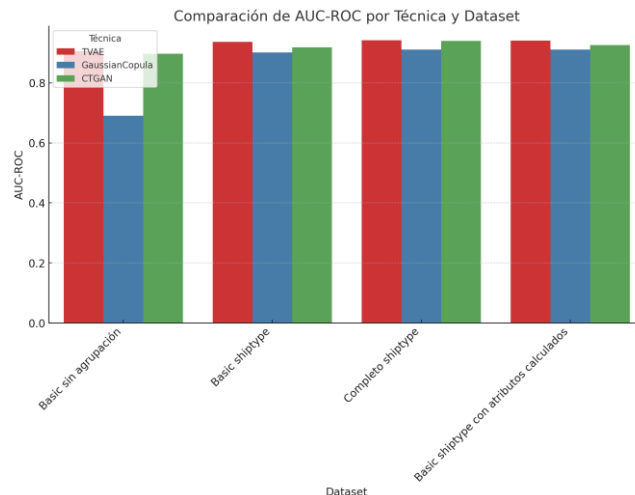


Figura 4-79: Comparación de *AUC-ROC* por técnica y *dataset*

Al comparar entre *datasets*, se destaca que:

- Agrupar por *shiptype* mejora significativamente el rendimiento del modelo de predicción.
- El *dataset* completo obtiene mejores resultados en todas las métricas.

Al comparar entre técnicas de generación, se observa que:

- TVAE es el más estable en todos los *datasets*.
- *Gaussian Copula* tiene el peor rendimiento, especialmente en *basic* sin agrupación.
- CTGAN obtiene las mejores métricas en completo.

A pesar de que el conjunto de datos sintéticos completo genera los mejores resultados en términos de rendimiento del algoritmo de *Random Forest* para la técnica CTGAN, este *dataset* no conserva las relaciones matemáticas entre los parámetros dimensionales y los atributos derivados, lo que puede generar inconsistencias en los datos sintéticos. Por lo tanto, si la prioridad es garantizar la coherencia de los datos generados sobre la precisión del modelo de clasificación, el *dataset basic* con atributos calculados es el más conveniente con la técnica de generación TVAE, ya que muestra las métricas más equilibradas.

La Figura 4-80 muestra la matriz de confusión del conjunto de datos completo generado con CTGAN, a la izquierda, y la del conjunto de datos con atributos calculados generados con TVAE, a la derecha. Estas matrices representan los errores y aciertos del algoritmo al predecir el tipo de barco. Las filas representan las clases reales, las columnas representan las predicciones de *Random Forest* y la diagonal (de arriba izquierda a abajo derecha) las predicciones correctas.



Figura 4-80: Matrices de confusión del entrenamiento con dos conjuntos de datos sintéticos diferentes

La matriz de la izquierda muestra valores altos en la diagonal lo que explica el alto valor de *Accuracy* del modelo para el conjunto de datos completos generado con CTGAN. Sin embargo, dado que este *dataset* no conserva las relaciones matemáticas entre los parámetros dimensionales y los atributos derivados, es posible que existan inconsistencias estructurales en los datos generados. Además, la matriz de la derecha, que representa la matriz de confusión del algoritmo para el *dataset* con atributos generados con TVAE, muestra menos aciertos, pero una distribución de errores más equilibrada, lo que sugiere que el algoritmo usa mejor los atributos calculados para diferenciar entre tipos de barcos. CTGAN logra más aciertos en *cargo* (4732 frente a 4445 de TVAE) pero a costa de clasificar erróneamente 801 *tankers* como *cargos*. Esto confirma la necesidad de elegir cuidadosamente el método de generación en función del objetivo final: si se prioriza la precisión en clasificación, CTGAN con el *dataset* completo es la mejor opción; si se busca coherencia, TVAE con el *dataset* con atributos calculados es una alternativa más adecuada.

Para comprender qué variables influyen más en las predicciones del algoritmo, se ha analizado la importancia de los parámetros utilizados en cada configuración de los datos sintéticos. La Figura 4-81 representa los gráficos de barras con la importancia de las variables para la clasificación con el conjunto de datos *basic* sin agrupación de CTGAN, a la izquierda, y el conjunto de datos *basic* por *shiptype* de TVAE, a la derecha. Y la Figura 4-82, los gráficos para el conjunto de datos completos de CTGAN, a la izquierda, y el conjunto con atributos de TVAE, a la derecha.

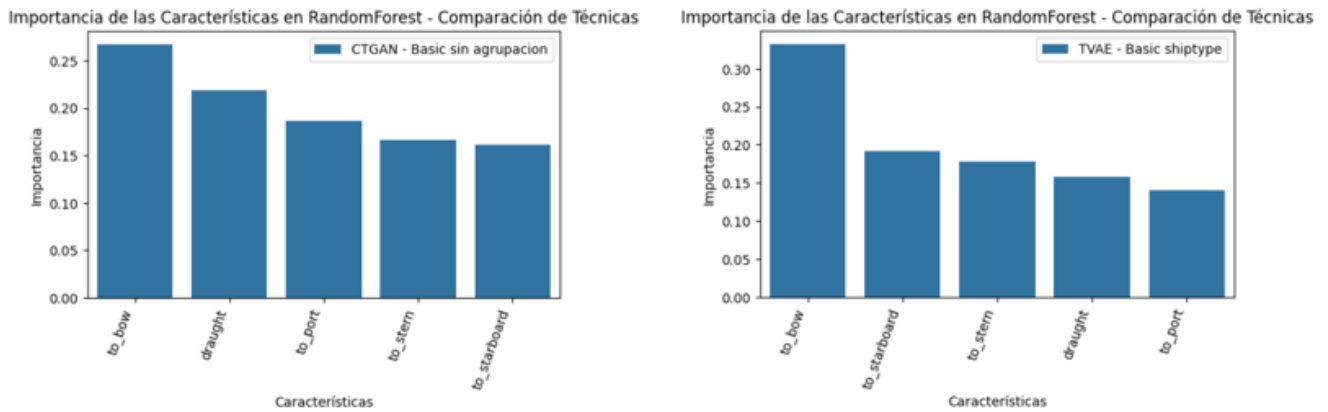


Figura 4-81: Importancia de las variables para dos conjuntos de datos del archivo *final_static_data_basic.csv*

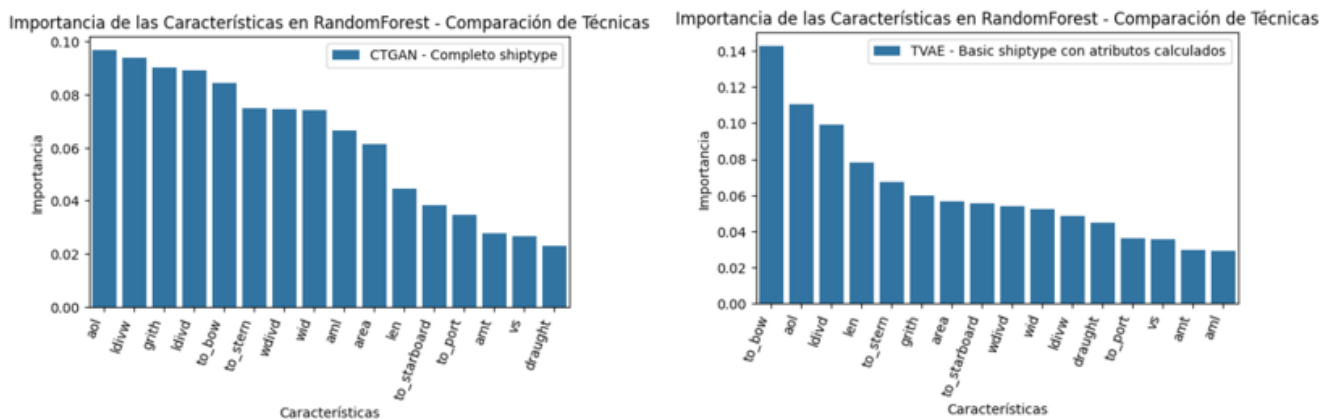


Figura 4-82: Importancia de las variables para dos conjuntos de datos del archivo *final_static_data.csv*

El análisis de la importancia de las características para el algoritmo *Random Forest* entrenado con los datos sintéticos revela que *to_bow* es la variable más influyente para la clasificación en la mayoría de configuraciones, independientemente de la técnica de generación utilizada. Este resultado coincide con la variable más influyente al entrenar el modelo con el archivo real.

Para el entrenamiento con los conjuntos de datos *basic* por *shiptype* con los atributos calculados, la variable *to_bow* continúa siendo la más relevante, seguida por *aol* y *ldivd* para las tres técnicas de generación de datos sintéticos. En cambio, al utilizar los datos sintéticos completos, la importancia de las características varía según la técnica de generación clasificándose *grith*, *ldivw*, *to_bow*, *len*, *aol* y *ldivd* como las más influyentes, a pesar de que para el archivo original fueron *ldivw* y *ldivd*.

Estos resultados refuerzan la idea de que un conjunto de datos más completo aporta variables relevantes para el algoritmo, pero su coherencia interna es crucial para preservar la fidelidad respecto a los datos originales. Si las relaciones matemáticas entre variables no se mantienen correctamente en los datos sintéticos, la importancia de las características puede verse alterada, afectando la interpretación del modelo y la calidad de las predicciones.

4.3.4 Análisis de rendimientos de computación de cada técnica

En este apartado se estudia el rendimiento computacional de la generación de datos sintéticos, pues es otro factor determinante, aparte de la calidad de los datos, a la hora de seleccionar una técnica de generación de datos sintéticos. Se comparará el rendimiento de cada técnica para cada conjunto de datos utilizado. Para ello, se midió el tiempo de entrenamiento de los sintetizadores en ambos escenarios: entrenando al sintetizador con el conjunto completo de datos sin agrupación por tipo de buque, y entrenando por tipo de buque.

La Tabla 4-15 muestra los tiempos de CPU del entrenamiento de cada técnica cuando se generaron los datos sin agrupar por tipo de buque.

	TVAE	<i>Gaussian Copula</i>	CTGAN
Tiempo de CPU	12 min 31 s	2,97 s	14 min 46 s

Tabla 4-15: Tiempos de entrenamiento por técnica en la generación sin distinción

La Figura 4-83 muestra los tiempos de entrenamiento en la generación por *shiptype* de los datos sintéticos *basic* (a la izquierda) y los datos sintéticos completo (a la derecha) para cada tipo de técnica.

Tiempos de computación por método y tipo de barco (basic):	Tiempos de computación por método y tipo de barco (completo):
TVAE - ShipType.Cargo: 95.00 segundos	TVAE - ShipType.Cargo: 182.51 segundos
TVAE - ShipType.Tanker: 30.40 segundos	TVAE - ShipType.Tanker: 54.56 segundos
TVAE - ShipType.Tug: 19.67 segundos	TVAE - ShipType.Tug: 36.19 segundos
TVAE - ShipType.Fishing: 18.59 segundos	TVAE - ShipType.Fishing: 35.41 segundos
TVAE - ShipType.Passenger: 11.53 segundos	TVAE - ShipType.Passenger: 18.53 segundos
GaussianCopula - ShipType.Cargo: 1.42 segundos	GaussianCopula - ShipType.Cargo: 3.72 segundos
GaussianCopula - ShipType.Tanker: 0.62 segundos	GaussianCopula - ShipType.Tanker: 1.93 segundos
GaussianCopula - ShipType.Tug: 0.51 segundos	GaussianCopula - ShipType.Tug: 1.61 segundos
GaussianCopula - ShipType.Fishing: 0.53 segundos	GaussianCopula - ShipType.Fishing: 1.56 segundos
GaussianCopula - ShipType.Passenger: 0.43 segundos	GaussianCopula - ShipType.Passenger: 1.21 segundos
CTGAN - ShipType.Cargo: 160.86 segundos	CTGAN - ShipType.Cargo: 290.13 segundos
CTGAN - ShipType.Tanker: 61.41 segundos	CTGAN - ShipType.Tanker: 87.16 segundos
CTGAN - ShipType.Tug: 38.68 segundos	CTGAN - ShipType.Tug: 57.59 segundos
CTGAN - ShipType.Fishing: 40.30 segundos	CTGAN - ShipType.Fishing: 55.09 segundos
CTGAN - ShipType.Passenger: 18.80 segundos	CTGAN - ShipType.Passenger: 26.30 segundos

Figura 4-83: Tiempos de ejecución por técnica y *shiptype* en la generación por tipo de buque

Para visualizar mejor estos resultados y poder comparar entre técnicas, la Figura 4-84 y la Figura 4-85 muestran los gráficos de barras de los tiempos de ejecución por tipo de buque y técnica para el conjunto de datos sintéticos *basic* y completo, respectivamente.

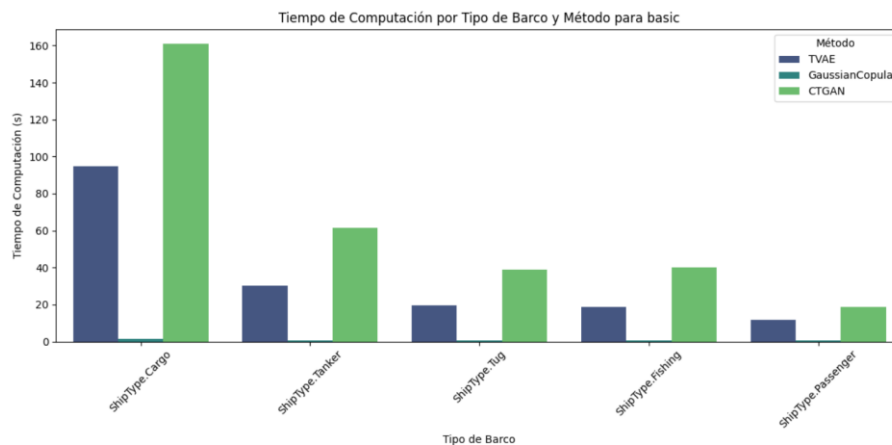


Figura 4-84: Gráfico de barras de tiempos de ejecución para los datos sintéticos *basic*

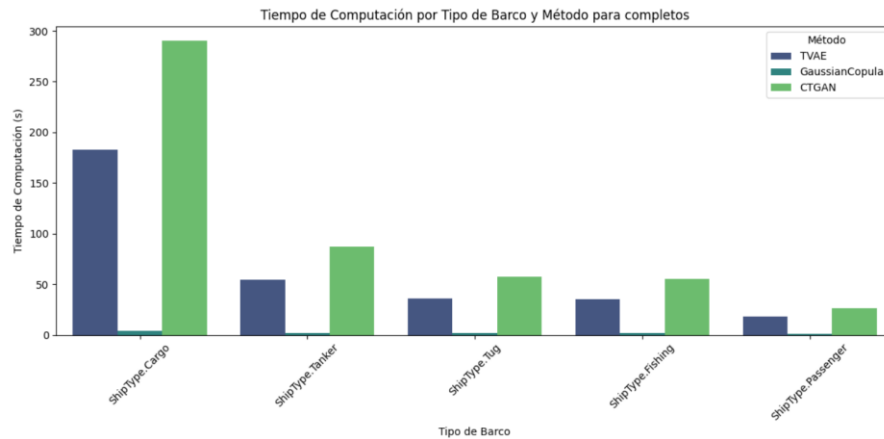


Figura 4-85: Gráfico de barras de tiempos de ejecución para los datos sintéticos completo

Analizando las gráficas se observa que CTGAN muestra los mayores tiempos de ejecución en ambos conjuntos de datos, siendo mayor para el conjunto de datos completo y especialmente para cargo debido al mayor volumen de datos de esta clase. TVAE tienen tiempos moderados en todos los tipos de barcos, pero también aumenta en el conjunto de datos completos. *Gaussian Copula* muestra tiempos mucho menores y más estables, con diferencias mínimas entre tipos de buques y entre conjunto de datos. En definitiva, la diferencia de tiempos entre el conjunto *basic* y el completo es clara, pues todas las técnicas requieren más tiempo al trabajar con los datos completos que poseen un mayor número de atributos. Al contrastar estos resultados con los obtenidos en la generación sin agrupar por tipo de buque, se observa que ambas coinciden en que CTGAN es la técnica que más tiempo requiere. Pero sorprendentemente, el tiempo total de ejecución al aplicar la generación por tipo de buque es menor que en la generación sin agrupar. Esta diferencia se debe a que, al trabajar con subconjuntos más pequeños (por *shiptype*), los modelos se entrenan más rápido y con menor complejidad, en lugar de un único entrenamiento con muchas muestras y una alta variedad de estructuras internas.

Por tanto, el enfoque segmentado por clase no solo mejora la calidad de los datos sintéticos generados, al preservar las relaciones internas propias de cada tipo de barco, sino que también es más eficiente tiempo de ejecución total, lo cual representa una ventaja significativa frente a la generación sin agrupar por tipo de buque. Aunque *Gaussian Copula* se muestre como la técnica más rápida en todos los escenarios, TVAE representa un equilibrio ideal entre calidad y rendimiento computacional, con tiempos medios razonables y una mayor capacidad para captar la estructura de los datos originales.

4.3.5 Resultados de la prueba de Data Augmentation

En este apartado se evalúa la aplicación de la técnica SMOTE sobre los datos reales para mejorar la representación de la clase *passenger*. Con esta técnica se generaron nuevas instancias interpoladas a partir de los datos vecinos más cercanos de los datos reales de la clase *passenger*. Como puede observarse en la Figura 4-86 la clase *passenger* ahora representa un 9,3% de los datos totales en comparación con el 6,4% del conjunto original.

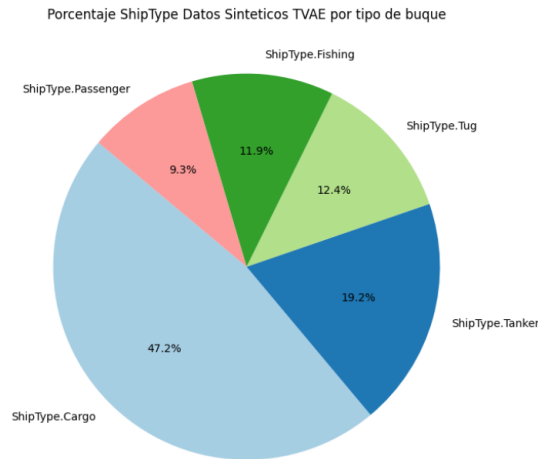


Figura 4-86: Diagrama de sectores del balance de clases tras la aplicación de SMOTE

A continuación, se realiza una comparativa estadística de las variables de *passenger* entre los datos reales y los aumentados con SMOTE. La Figura 4-87 muestra la comparación entre la media y la desviación estándar de cada variable.

Tabla comparativa Passenger (real vs SMOTE):

	Variable	Media Real	Media SMOTE	Std Real	Std SMOTE
0	draught	3.293	3.307	1.853	1.796
1	to_bow	23.282	23.414	16.796	16.234
2	to_stern	55.680	55.531	57.485	55.598
3	to_port	7.224	7.208	5.044	4.808
4	to_starboard	7.169	7.186	4.899	4.732

Figura 4-87: Comparativa de media y desviación entre datos reales y datos sintéticos generados con SMOTE

Los valores obtenidos muestran gran coherencia entre ambas versiones, con diferencias mínimas, lo que sugiere que la técnica SMOTE preserve adecuadamente las características estadísticas de los datos originales.

Para completar el análisis, se generaron las curvas de densidad para comparar las distribuciones de las variables entre los datos reales y los generados por SMOTE para la categoría *passenger* (Figura 4-88 a Figura 4-90).

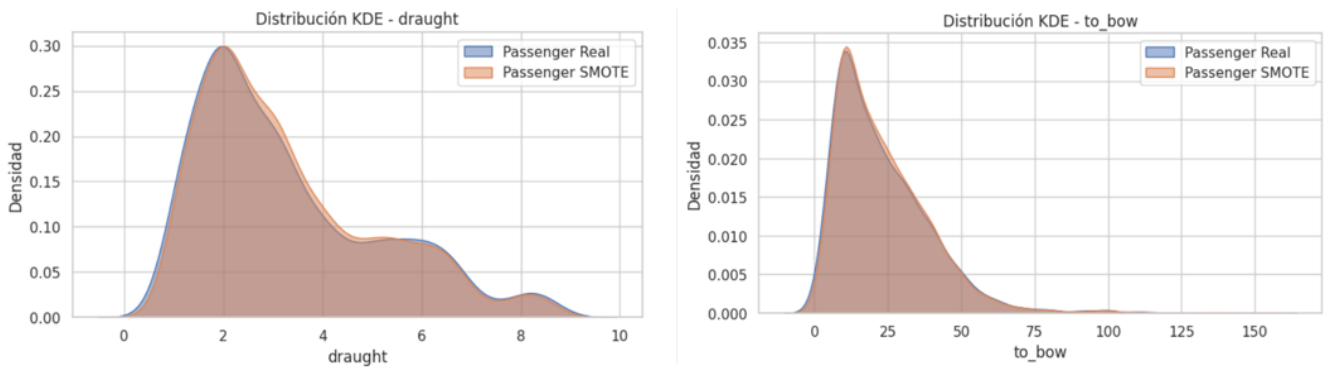


Figura 4-88: Gráficos de densidad de *draught* y *to_bow* para *passenger* con Data Augmentation

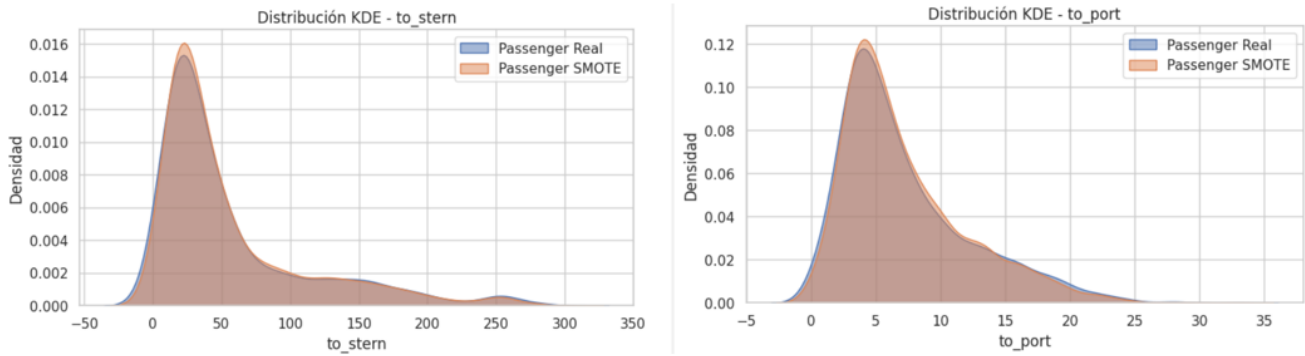


Figura 4-89: Gráficos de densidad de *to_stern* y *to_port* para *passenger* con *Data Augmentation*

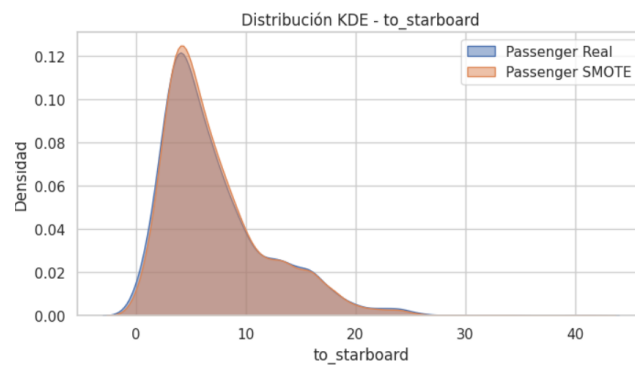


Figura 4-90: Gráficos de densidad de *to_starboard* para *passenger* con *Data Augmentation*

Para todas las variables, las curvas de los datos reales y los datos reales con SMOTE se solapan de forma muy significativa siendo casi idénticas, por lo que las interpolaciones no generaron valores anómalos ni atípicos.

Además del análisis estadístico y visual, se evalúa el impacto de la técnica de *Data Augmentation* sobre el rendimiento del modelo de clasificación basado en *Random Forest*. Se compara el rendimiento del modelo entrenado con los datos reales originales y el rendimiento del modelo entrenado con los datos aumentados mediante SMOTE para la clase *passenger*. Los resultados se muestran en la Tabla 4-16.

	<i>Accuracy</i>	<i>F1-score</i>	<i>Precision</i>	<i>Recall</i>	<i>AUC-ROC</i>
Datos reales	0.834597	0.832538	0.833731	0.834597	0.969525
Datos reales aumentados	0.837519	0.835737	0.836846	0.837519	0.971008

Tabla 4-16: Resultados de las métricas de rendimiento de *Random Forest* para *Data Augmentation*

Ambos conjuntos de datos muestran valores muy similares de *accuracy*, *F1-score*, *precisión* y *recall*, siendo ligeramente superiores en el modelo entrenado con los datos aumentados. Dado que el objetivo principal del *Data Augmentation* es mejorar la detección de la clase *passenger*, se calcula el AUC-ROC de forma binaria exclusivamente para *passenger*, en vez del que se muestra en la Tabla 4-16, que es un promedio. Se obtuvo un valor de AUC-ROC específico para *passenger* de 0,9687 para los datos reales y de 0,9756 para los aumentados. La Figura 4-91 muestra la comparación de las curvas ROC obtenidas en ambas situaciones.

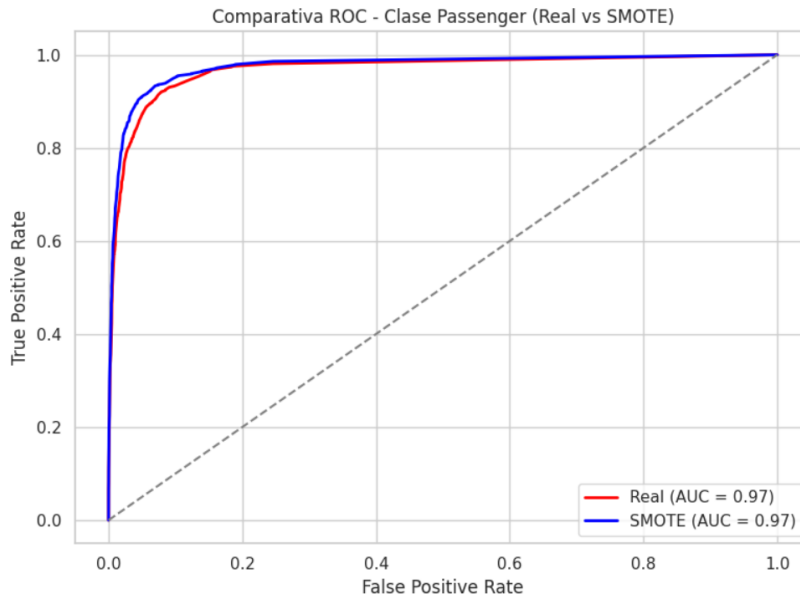


Figura 4-91: Comparación curva ROC clase *passenger* (real vs SMOTE)

Se observa que el modelo entrenado con SMOTE (curva azul) presenta una ligera mejora en la capacidad de discriminación de la clase *passenger* respecto al modelo entrenado con los datos reales (curva roja). El aumento de la clase *passenger* ha tenido un impacto positivo, aunque sutil, en la detección de esta clase minoritaria.

5 CONCLUSIONES Y LÍNEAS FUTURAS

5.1 Conclusiones

En este trabajo se ha explorado la viabilidad de generar datos sintéticos a partir de registros AIS estáticos como solución a la escasez y desequilibrio en el tamaño de las clases, ambos problemas presentes en los conjuntos de datos reales empleados en modelos de aprendizaje automático supervisado. Para ello, se implementaron y evaluaron tres técnicas de generación de datos sintéticos: TVAE, *Gaussian Copula* y CTGAN, aplicándolas tanto de manera global sobre el conjunto de datos completo como de forma segmentada por tipo de buque. Dicha evaluación incluyó métricas estadísticas, análisis visuales de distribuciones y correlaciones, así como una validación de utilidad mediante un modelo de aprendizaje automático supervisado. Las principales conclusiones derivadas de este estudio se detallan en los siguientes párrafos.

En primer lugar, TVAE ha destacado como la técnica generativa más eficaz en términos generales. Por su parte, CTGAN mostró un rendimiento competitivo especialmente cuando el tamaño de los datos de entrenamiento es reducido, lo cual se observó al generar datos segmentados por tipo de buque. En contraste, *Gaussian Copula* tendió a suavizar las distribuciones, siendo la que peor replicó los datos originales, perdiendo detalle en las colas y picos característicos. En definitiva, las técnicas basadas en redes neuronales demostraron mayor capacidad para reproducir distribuciones complejas que un enfoque puramente estadístico de cópulas, siendo TVAE la opción más robusta de las evaluadas.

En segundo lugar, se comprueba que la generación de datos de forma segmentada por tipo de buque mejora la calidad y fidelidad de los datos sintéticos. Al entrenar modelos generativos por separado para cada categoría de *shiptype*, las distribuciones específicas de cada tipo de buque se preservan mejor. No obstante, su implementación requiere un volumen suficiente de datos por clase. En el caso de tener categorías con datos escasos es necesario explorar enfoques como *Data Augmentation*. En este TFG, la técnica de *data augmentation* (SMOTE) permitió aumentar la proporción de muestras de la clase minoritaria *passenger* en el conjunto de datos originales, con el objetivo de mejorar su representación y el rendimiento del modelo de clasificación. Esta prueba ha demostrado ser una herramienta eficaz para mitigar problemas de desequilibrio de clases.

En tercer lugar, se identificó una limitación importante en las tres técnicas de generación de datos sintéticos utilizadas: no infieren ni mantienen por sí mismas ciertas relaciones matemáticas existentes entre algunos atributos. Como se explicó con el atributo *len*, que es la suma de las entradas *to_bow* y *to_stern*, en la mayoría de instancias se generaron valores que no respetaban la suma esperada. Por ello, las relaciones matemáticas se incorporaron manualmente tras la generación sintética. De esta forma, el

conjunto de datos sintéticos más adecuado para usos posteriores es aquel en el que se generaron solo los parámetros básicos, calculando a posteriori los atributos derivados mediante funciones. Y se opta por excluir el conjunto de datos completo, que contiene los atributos generados sintéticamente, a pesar de que en la validación de utilidad mostró mejores rendimientos.

En resumen, este TFG logra generar datos estáticos AIS de forma sintética que reproducen razonablemente bien las características de los datos reales. Como aportaciones principales: se identifica TVAE como la técnica más eficaz para este trabajo, se resalta la importancia de la segmentación por tipo de buque y se demuestra que los datos sintéticos con técnicas de *data augmentation* pueden servir como sustituto o complemento de los datos reales en tareas de aprendizaje automático. Sin embargo, conviene reconocer las limitaciones encontradas: las técnicas no garantizan por sí solas la preservación de todas las relaciones presentes en los datos originales, la calidad del dato sintético depende de la representatividad del conjunto de entrenamiento real (si este contiene sesgos o insuficiencias, los datos sintéticos los reflejarán), y el alcance del estudio se limitó a datos estáticos AIS y a un único caso de uso (clasificación de *shiptype*).

5.2 Líneas futuras

En base a las conclusiones del apartado anterior, se proponen a continuación varias líneas futuras para ampliar y continuar los avances realizados en este trabajo:

- Estudio de otras técnicas de generación de datos sintéticos como variantes avanzadas de GANs o VAEs, que puedan crear datos con mejores resultados en cuanto a la calidad y a su uso en modelos de aprendizaje automático supervisado. También se podría estudiar la posibilidad de combinar modelos y experimentar con generadores condicionales que incluyan restricciones como *Diffusion Models* o *TabTransformers* para tratar de preservar las relaciones matemáticas entre columnas.
- Mejoras y expansión del análisis de *Data Augmentation*. Se podría diseñar estrategias híbridas de aumento de datos que combinen enfoques generativos y enfoques de sobremuestreo clásico.
- Ampliar el estudio a la generación sintética de trayectorias para mejorar modelos de predicción e identificación de patrones de navegación sospechosos. Dado que el sistema AIS genera también datos dinámicos, se podría abordar la generación de estos sintéticamente. Un estudio en esta dirección validaría si los enfoques y hallazgos de este TFG como, por ejemplo, la necesidad de segmentación o de preservar relaciones, se mantienen en datos espacio temporales, contribuyendo a una herramienta más general de generación de datos sintéticos AIS tanto estáticos como dinámicos.

6 BIBLIOGRAFÍA

- [1] Ministerio de Defensa, «Resolución 320/38174/2023, de 21 de abril, de la Dirección General de Armamento y Material, por la que se convoca proceso de selección de proyectos de I+D de interés para Defensa susceptibles de ser incluidos en el ámbito del Programa Coincidente,» Boletín Oficial del Estado, Madrid, 2023.
- [2] Ministerio de Defensa, «Resolución 320/38147/2024, de 17 de abril, de la Dirección General de Armamento y Material, por la que se convoca proceso de selección de proyectos de I+D de interés para Defensa susceptibles de ser incluidos en el ámbito del Programa Coincidente,» Boletín Oficial del Estado, Madrid, 2024.
- [3] G. Rodríguez Casajús, B. Barragáns Martínez y P. Sendín Raña, «Predicción de tipo de buque utilizando datos AIS y técnicas de inteligencia artificial,» Trabajo Fin de Grado, 2022.
- [4] J. Roldán Calvo, B. Barragáns Martínez y P. Sendín Raña, «Predicción de tipo de buque utilizando representaciones de trayectorias y técnicas de Inteligencia Artificial,» Trabajo Fin de Grado, 2024.
- [5] C. Manning, «*Stanford University*,» Septiembre 2020. [En línea]. Available: <https://hai.stanford.edu/sites/default/files/2020-09/AI-Definitions-HAI.pdf>. [Último acceso: 13 Enero 2025].
- [6] Diccionario de la Real Academia Española, «rae: inteligencia,» Real Academia Española, 2014. [En línea]. Available: <https://dle.rae.es/inteligencia?m=form#2DxmhCT>.
- [7] *International Organization for Standardization*, «ISO,» 2022. [En línea]. Available: <https://hai.stanford.edu/sites/default/files/2020-09/AI-Definitions-HAI.pdf>. [Último acceso: 14 Enero 2025].
- [8] S. Russell y P. Norvig, *Artificial Intelligence: A Modern Approach*, Third Edition ed., New Jersey: Prentice Hall, 1995.
- [9] J. R. Casar Corredera, «Inteligencia artificial generativa,» *Anales de la Real Academia de Doctores de España*, vol. 8, nº 3, pp. 475-489, 2023.
- [10] IBM, «*What is machine learning?*,» [En línea]. Available: <https://www.ibm.com/think/topics/machine-learning>. [Último acceso: 15 marzo 2025].

- [11] Equipo de comunicacion de Telefónica, «Aprendizaje supervisado: definición y aplicaciones,» Telefónica, 10 junio 2024. [En línea]. Available: <https://www.telefonica.com/es/sala-comunicacion/blog/aprendizaje-supervisado-definicion-aplicaciones/>. [Último acceso: 17 marzo 2025].
- [12] IBM, «¿Qué es el *clustering*?,» IBM, 21 febrero 2024. [En línea]. Available: <https://www.ibm.com/es-es/think/topics/clustering>. [Último acceso: 16 marzo 2025].
- [13] IBM, «*What is deep learning*?,» [En línea]. Available: <https://www.ibm.com/think/topics/deep-learning>. [Último acceso: 15 marzo 2025].
- [14] La universidad en Internet, «Árboles de decisión: en qué consisten y aplicación en Big Data,» UNIR, [En línea]. Available: <https://www.unir.net/revista/ingenieria/arboles-de-decision/>. [Último acceso: 16 marzo 2025].
- [15] R. Martínez-Martínez y F. Daverio, «Arboles de decisiones,» CIDE: Centro de Investigación y Docencia Económica, [En línea]. Available: <https://rafneta.github.io/CienciaDatosPythonCIDE/Laboratorios/Lab6/Arboles.html>. [Último acceso: 16 marzo 2025].
- [16] GeeksforGeeks, «*Random Forest Algorithm in Machine Learning*,» GeeksforGeeks, [En línea]. Available: <https://www.geeksforgeeks.org/random-forest-algorithm-in-machine-learning/>. [Último acceso: 16 marzo 2025].
- [17] Equipo de comunicacion de Telefónica, «¿Qué es el *machine learning*?,» Telefónica, 15 octubre 2024. [En línea]. Available: <https://www.telefonica.com/es/sala-comunicacion/blog/que-es-y-como-funciona-el-machine-learning/>. [Último acceso: 15 marzo 2025].
- [18] Ó. Garcia-Olalla Olivera, «Redes Neuronales artificiales: Qué son y cómo se entrenan,» Xeridia, septiembre 2019. [En línea]. Available: <https://www.xeridia.com/blog/redes-neuronales-artificiales-que-son-y-como-se-entrenan-parte-i>. [Último acceso: 16 marzo 2025].
- [19] IBM, «¿Qué son las redes neuronales?,» [En línea]. Available: <https://www.ibm.com/es-es/topics/neural-networks>. [Último acceso: 16 marzo 2025].
- [20] J. Dabass, «*Friendly Introduction to Deep Learning Architectures (CNN, RNN, GAN, Transformers, Encoder-Decoder Architectures)*,» Python in Plain English, 1 abril 2024. [En línea]. Available: <https://python.plainenglish.io/friendly-introduction-to-deep-learning-architectures-cnn-rnn-gan-transformers-encoder-decoder-b11334e4cdf7>. [Último acceso: 16 marzo 2025].
- [21] Amazon Web Services, «¿Qué es una RNN (red neuronal recurrente)?,» AWS, [En línea]. Available: <https://aws.amazon.com/es/what-is/recurrent-neural-network/>. [Último acceso: 16 marzo 2025].
- [22] J. Maiz, «Centro de operaciones y vigilancia de acción marítima de la Armada,» *Revista Defensa*, nº 420, 8 septiembre 2017.
- [23] M. de la Gándara y García, «El COVAM de la Armada al servicio de la comunidad marítima,» 27 julio 2011. [En línea]. Available: <https://armada.defensa.gob.es/ArmadaPortal/ShowPropertyServlet?nodePath=/BEA%20Repository/Desktops/Portal/ArmadaEspannola/Pages/conocenosespeciales/05actividades/segu>

- ridadmaritima/02COVAMIEE/02COVAMIEE-es/argCOVAMIEEES. [Último acceso: 27 febrero 2025].
- [24] Raymarine, «Sistema de Identificación Automática,» 22 agosto 2022. [En línea]. Available: <https://www.raymarine.com/es-es/aprendizaje/guias-en-linea/automatic-identification-system>. [Último acceso: 10 febrero 2025].
- [25] Comar systems, «What is AIS?,» [En línea]. Available: <https://comarsystems.com/what-is-ais/>. [Último acceso: 10 febrero 2025].
- [26] SVB, «SVB-marine: Todo lo que debe saber sobre AIS,» [En línea]. Available: <https://www.svb-marine.es/es/guia/todo-lo-que-debe-saber-sobre-ais.html?srsId=AfmBOopCSt5GbY8V9u9YHzGuD-2aUjiKWefkwZaidXqES26oLNS93cg3>. [Último acceso: 12 febrero 2025].
- [27] F. Heymann, T. Noack y P. Banys, «*Plausibility analysis of navigation related AIS parameter based on time series*,» de *European Navigation Conference*, Neustrelitz, Germany, 2013.
- [28] T. Emmens, C. Amrit, A. Abdi y M. Ghosh, «*The promises and perils of Automatic Identification System data*,» *Expert Systems With Applications*, vol. 1178, 15 septiembre 2021.
- [29] T. Yang, X. Wang y Z. Liu, «*Ship Type Recognition Based on Ship Navigating Trajectory and Convolutional Neural Network*,» *Journal of Marine Science and Engineering*, vol. 10, n° 84, 2022.
- [30] D. B. Rubin, «*Discussion Statistical Disclosure Limitation*,» *Journal of Oficial Statistics*, vol. 9, n° 2, pp. 461-468, 1993.
- [31] Turing, «*Synthetic data generation techniques*,» [En línea]. Available: <https://www.turing.com/kb/synthetic-data-generation-techniques>. [Último acceso: 23 Enero 2025].
- [32] M. All, «*synthetic data generation*,» Datacamp, 22 octubre 2024. [En línea]. Available: <https://www.datacamp.com/es/tutorial/synthetic-data-generation>. [Último acceso: 25 enero 2025].
- [33] C. Ortega, «Datos sintéticos,» QuestionPro, [En línea]. Available: <https://www.questionpro.com/blog/es/datos-sinteticos/>. [Último acceso: 24 enero 2025].
- [34] Y. Lu, M. Shen, H. Wang, X. Wang, C. v. Rechem, T. Fu y W. Wei, «*Machine Learning for Synthetic Data Generation: A Review*,» *Arxiv*, vol. 14, n° 8, 2021.
- [35] R. R. Páez Yrigoyen, «Los datos sintéticos son el gran avance en Computer Vision,» *Forbes*, 2022.
- [36] C. Ortega, «QuestionPro,» [En línea]. Available: <https://www.questionpro.com/blog/es/ventajas-de-los-datos-sinteticos/>. [Último acceso: 6 febrero 2025].
- [37] T. Marwala, E. Fournier-Tombs y S. Stinckwich, «*The Use of Synthetic Data to Train AI Models: Opportunities and Risks for Sustainable Development*,» *Arxiv*, 31 Agosto 2023.
- [38] E. D. Cristofaro, «Synthetic Data: Methods, Use Cases, and Risks,» *Arxiv*, 27 febrero 2024.

- [39] M. Hernandez, G. Epelde, A. Alberdi, R. Cilla y D. Rankin, «*Synthetic data generation for tabular health records: A systematic review*,» *Neurocomputing*, vol. 493, pp. 28-45, Julio 2022.
- [40] A. Figueira y B. Vaz, «*Survey on Synthetic Data Generation, Evaluation Methods and GANs*,» *Mathematics MDPI*, vol. 2733, nº 10, 2 agosto 2022.
- [41] V. C. Pezoulas, D. I. Zaridis, E. Mylona, C. Androutsos, K. Apostolidis, N. S. Tachos y D. I. Fotiadis, «*Synthetic data generation methods in healthcare: A review on open-source tools and methods*,» *Computational and Structural Biotechnology Journal*, vol. 23, pp. 2892-2910, Diciembre 2024.
- [42] M. Goyal y Q. H. Mahmoud, «*A Systematic Review of Synthetic Data Generation Techniques Using Generative AI*,» *Electronics*, vol. 13, nº 3509, 2024.
- [43] J. Jordon, F. Houssiau, M. Bottarelli, L. Szpruch, G. Cherubin, C. Maple, S. N. Cohen y A. Weller, «*Synthetic Data - what, why and how?*,» The Royal Society, Reino Unido, 2022.
- [44] A. Ali Awan, «*A Complete Guide to Data Augmentation*,» Datacamp, 9 diciembre 2024. [En línea]. Available: <https://www.datacamp.com/tutorial/complete-guide-data-augmentation>. [Último acceso: 15 marzo 2025].
- [45] J. Huang, Q. Huang, G. Mou y C. Wu, «*DPWGAN: High-Quality Load Profiles Synthesis With Differential Privacy Guarantees*,» *IEEE Transactions on Smart Grid*, vol. 14, nº 4, pp. 3283 - 3295, 2023.
- [46] C. Ortega, «Generación de datos sintéticos: Técnicas y consideraciones,» QuestionPro, [En línea]. Available: <https://www.questionpro.com/blog/es/generacion-de-datos-sinteticos/>. [Último acceso: 15 marzo 2025].
- [47] J. Villacampa Porta, «Generación sintética de datos para modelos de machine learning orientados a la caracterización dieléctrica de líquidos mediante un resonador dieléctrico,» Trabajo Fin de Grado, Comillas, 2023.
- [48] F. Benali, D. Bodénès, N. Labroche y C. d. Runz, «*MTCopula: Synthetic Complex Data Generation Using Copula*,» 23rd International Workshop on Design, Optimization, Languages, Nicosia, Cyprus, 2021.
- [49] DataCebo, «*Welcome to the SDV!*,» DataCebo, [En línea]. Available: <https://docs.sdv.dev/sdv>. [Último acceso: 15 marzo 2025].
- [50] N. Sánchez Medel y R. Díaz Hernández, «Reconocimiento de células sanguíneas por medio de aprendizaje profundo y generación de datos sintéticos,» INAOE: Instituto Nacional de Astrofísica, Óptica y Electrónica, Puebla, México, 2023.
- [51] M. Nebot Moyano y J. Béjar Alonso, «Métodos para la generación de datos artificiales tabulares,» Trabajo Fin de Grado, Facultad de Informatica de Barcelona, Barcelona, 2024.
- [52] S. Arroni del Riego y D. Manrique Gamo, «Generador de datos sintéticos tabulares basado en modelos de difusión,» Trabajo de Fin de Máster, Universidad Politécnica de Madrid, 2024.
- [53] C. M. Ward, J. Harguess y C. Hilton, «*Ship classification from overhead imagery using synthetic data and domain adaptation*,» *Space and Naval Warfare Systems Center, Pacific (SSC PAC)*, San Diego, CA, United States, 2018.

- [54] M. Naranjo Martínez, L. Ciruelo Sanz, J. M. Moreu Gamazo, E. Díez Benito, V. Chaparro Parra y R. Gómez-Espinosa Martín, «Implementación de un modelo de *Deep Learning* para la detección de abarreamientos en imágenes satelitales en SEDA,» HI-Iberia , Madrid.
- [55] Project Jupyter, «*The JupyterLab Interface*,» Jupyterlab, 15 marzo 2025. [En línea]. Available: <https://jupyterlab.readthedocs.io/en/stable/user/interface.html>.
- [56] P. Recuerdo, «Python para todos (2): ¿Qué son los Jupyter Notebooks?,» Wordpress, 17 Octubre 2021. [En línea]. Available: <https://palomarecuero.wordpress.com/2021/10/17/python-para-todos-2-que-son-los-jupyter-notebooks/>. [Último acceso: 15 Marzo 2025].
- [57] Project Jupyter, «*Jupyter*,» [En línea]. Available: <https://jupyter.org/>. [Último acceso: 15 marzo 2025].
- [58] IONOS, «*Jupyter Notebook*: documentos web para análisis de datos, código en vivo y mucho más,» IONOS Digital Guide, 28 Febrero 2019. [En línea]. Available: <https://www.ionos.es/digitalguide/paginas-web/desarrollo-web/jupyter-notebook/>. [Último acceso: 15 marzo 2025].
- [59] Coursera Staff, «¿Qué es Python y para qué se usa? Guía para principiantes,» Coursera, 29 noviembre 2023. [En línea]. Available: <https://www.coursera.org/mx/articles/what-is-python-used-for-a-beginners-guide-to-using-python>. [Último acceso: 15 marzo 2025].
- [60] datos.gob, «datos.gob: Cómo preparar un conjunto de datos para machine learning y análisis,» 28 Abril 2023. [En línea]. Available: <https://datos.gob.es/es/blog/como-preparar-un-conjunto-de-datos-para-machine-learning-y-analisis>. [Último acceso: 20 Febrero 2025].
- [61] Pyais 2.1.1 documentation, «*Welcome to Pyais's documentation!*,» [En línea]. Available: <https://pyais.readthedocs.io/en/latest/>. [Último acceso: 23 febrero 2025].
- [62] United States Coast Guards Navigation Center, «*AIS class a ship static and voyage related data (Message 5)*,» U.S. Department of Homeland Security, [En línea]. Available: <https://www.navcen.uscg.gov/ais-class-a-static-voyage-message-5>.
- [63] R. García Bellido, J. González Such y J. y Jornet Meliá, «*SPSS: KOLMOGOROV SMIRNOV*,» 2010. [En línea]. Available: https://www.uv.es/innomide/spss/SPSS/SPSS_0802A.pdf. [Último acceso: 4 marzo 2025].
- [64] learnstatisticseasily, «Qué es: Prueba de *Kolmogorov-Smirnov*,» [En línea]. Available: <https://es.statisticseasily.com/glossario/what-is-kolmogorov-smirnov-test/#:~:text=Limitaciones%20de%20la%20prueba%20Kolmogorov%2DSmirnov&text=Una%20limitaci%C3%B3n%20importante%20es%20su,nula%20incluso%20para%20diferencias%20triviales..> [Último acceso: 17 marzo 2025].
- [65] StudySmarter, «Prueba de *Kolmogorov-Smirnov*,» [En línea]. Available: <https://www.studysmarter.es/resumenes/matematicas/estadistica-y-probabilidad/prueba-de-kolmogorov-smirnov/>. [Último acceso: 4 marzo 2025].
- [66] M. Narvaez, «Prueba de chi-cuadrado: ¿Qué es y cómo se realiza?,» QuestionPro, [En línea]. Available: <https://www.questionpro.com/blog/es/prueba-de-chi-cuadrado-de-pearson/>. [Último acceso: 4 marzo 2025].
- [67] DATAtab Team (2025), «DATAtab: Prueba Chi-cuadrado,» [En línea]. Available: <https://datatab.es/tutorial/chi-square-test>. [Último acceso: 4 marzo 2025].

- [68] M. Waskom, «seaborn.kdeplot,» [En línea]. Available: <https://seaborn.pydata.org/generated/seaborn.kdeplot.html>. [Último acceso: 4 marzo 2025].
- [69] A. Thevapalan, «Matriz de correlaciones en Excel: Guía completa para crear e interpretar,» datacamp, 16 enero 2025. [En línea]. Available: <https://www.datacamp.com/es/tutorial/correlation-matrix-excel>. [Último acceso: 4 marzo 2025].
- [70] IBM, «¿Qué es el análisis de componentes principales (PCA)?,» 8 diciembre 2023. [En línea]. Available: <https://www.ibm.com/es-es/topics/principal-component-analysis>. [Último acceso: 4 marzo 2025].
- [71] A. Ali Awan, «Introducción al t-SNE,» datacamp, 21 agosto 2024. [En línea]. Available: <https://www.datacamp.com/es/tutorial/introduction-t-sne>. [Último acceso: 5 marzo 2025].
- [72] T. E. Raghunathan, «*Synthetic Data*,» *Annual Review of Statistics and Its Application*, vol. 8, pp. 129-140, 2021.

ANEXO I: IMPLICACIONES SOCIALES, Y/O ECONÓMICAS, Y/O AMBIENTALES

La generación de datos sintéticos mediante técnicas de IA está teniendo un impacto creciente en múltiples sectores. Desde una perspectiva general, los datos sintéticos permiten paliar la escasez de datos reales, facilitan la realización de investigaciones, pueden preservar la privacidad y reducen costes asociados a la obtención de datos reales.

En el plano social, los datos sintéticos permiten generalizar el acceso a conjunto de datos que están limitados en su origen, o que contienen información sensible. En el ámbito marítimo, por ejemplo, permitirían entrenar modelos de aprendizaje automático sin necesidad de compartir información operativa clasificada como la ruta de algún buque de interés nacional. En el ámbito de la medicina los datos sintéticos están permitiendo avances en investigaciones de enfermedades de las que no se dispone muchas fuentes de datos.

Desde el punto de vista económico, el uso de datos sintéticos puede suponer un importante ahorro de recursos, dado que evitan campañas de recopilación y etiquetado de datos. En el contexto de este TFG, los datos AIS sintéticos pueden aumentar la cantidad de información disponible para entrenar modelos de clasificación, detectar anomalías o prever trayectorias, lo que a su vez optimiza los recursos técnicos y humanos utilizados en las actividades de vigilancia y análisis marítimo.

Por último, la generación de datos sintéticos puede disminuir la necesidad de campañas de recolección de datos, reduciendo así el impacto ambiental, especialmente en entornos sensibles como el marítimo. Sin embargo, no se puede obviar que el entrenamiento de modelos de IA conllevan un consumo significativo de energía, generando un reto en términos de sostenibilidad digital. Es crucial desarrollar modelos más eficientes energéticamente y centrarse en datasets de tamaño moderado. Esto ayuda a equilibrar la innovación tecnológica con la responsabilidad ambiental. En resumen, avanzar en IA debe hacerse considerando su huella energética y buscando un equilibrio entre progreso y sostenibilidad.

ANEXO II: REFLEXIONES ÉTICAS Y SOCIALES

El uso de la IA y, en particular, la generación de datos sintéticos, plantea numerosos retos éticos que deben considerarse para el diseño de cualquier proyecto que emplee estas técnicas.

Los datos sintéticos plantean desafíos éticos y sociales relacionados con la privacidad. Aunque la generación de datos sintéticos busca proteger la confidencialidad, no elimina completamente los riesgos de privacidad, pues existe la posibilidad de reidentificación de individuos si los modelos sobreajustan los patrones de los datos originales, lo que puede filtrar información sensible.

Otro aspecto a considerar es la veracidad y fiabilidad de los datos generados. Aunque las técnicas aprenden a imitar distribuciones de los datos reales no garantizan la conservación de todas las relaciones, como se mostró en este trabajo. Por ello, los datos sintéticos deben usarse con cautela, especialmente si van a alimentar sistemas de decisión automáticos en contextos sensibles, como en defensa o sanidad, que podrían conducir a decisiones inexactas o equivocadas si no se evalúan con rigor.

Además de la calidad de los datos, es crucial abordar la transparencia de los mismos. Desde una perspectiva social, es vital informar cuándo se utilizan datos sintéticos. Si una agencia gubernamental presenta estadísticas, debe especificar si se basan en datos reales, sintéticos o una combinación. No hacerlo podría ser considerado engañoso. En publicaciones científicas o informes de inteligencia, ocultar que los datos son sintéticos puede dañar la confianza y credibilidad una vez que se descubra. La veracidad también está relacionada con los riesgos de desinformación, ya que la misma tecnología que genera datos sintéticos puede usarse para la creación de *deepfakes* y contenido engañoso. El uso malintencionado de datos sintéticos representa un peligro ético. Por ello, deben establecerse protocolos para verificar la autenticidad de los datos.

En definitiva, mejorar la calidad de los datos y mantener las responsabilidades ético-sociales es crucial para el avance en innovaciones tecnológicas y en investigaciones científicas. Se debe evitar la falsa seguridad de que los datos sintéticos son siempre seguros, y se deben desarrollar técnicas de control de privacidad y desarrollar políticas de uso de los datos sintéticos.

ANEXO III: ENLACE CÓDIGO COMPLETO EN *GITHUB*

El código completo implementado en este trabajo está disponible en un repositorio de GitHub, que proporciona una visión detallada de todas las técnicas empleadas y de todos los resultados obtenidos. Se puede acceder al repositorio en el siguiente enlace:

<https://github.com/MartaHS11/Codigo-TFG-Generacion-de-datos-sinteticos-AIS/tree/main>

ANEXO IV: CÓDIGO PARA LOS DATOS SIN AGRUPAR POR *SHIPTYPE*

Se muestra un ejemplo del código utilizado para la **generación** de los datos sintéticos *basic* sin agrupar por tipo de buque. Este *script* se corresponde con la técnica TVAE, para el resto de técnicas se emplea el mismo código, pero cambiando el sintetizador (*GaussianCopulaSynthesizer* y *CTGANSynthesizer*) y guardándolo con el formato que se explicó en el capítulo 3.

```
#Importar Librerías
import pandas as pd
from sdv.metadata import Metadata
from sdv.single_table import TVAESynthesizer

# Cargar Los datos reales
df = pd.read_csv("final_static_data_basic.csv")

#Seleccionar columnas relevantes y generar dataframe
columnas_estaticas=["MMSI", "draught", "to_bow", "to_stern", "to_port", "to_starboard",
"shiptype"]
df_estaticos=df[columnas_estaticas]

#Convertir Los datos categoricos a string para que Los Lea mejor el modelo
df_estaticos["shiptype"]=df_estaticos["shiptype"].astype(str)

#Detectar la metadata a partir del dataframe
metadata=Metadata.detect_from_dataframe(
    data=df_estaticos,
    table_name= 'AIS_estaticos'
)
# Crear el modelo VAE para generación de datos sintéticos o GaussianCopulaSynthesizer /
CTGANSynthesizer
synthesizer = TVAESynthesizer(metadata)

%%time
# Entrenar el modelo con Los datos reales
synthesizer.fit(df_estaticos)

%%time
# Generar datos sintéticos
num_synthetic_rows = 50000
df_sinteticos = synthesizer.sample(num_rows=num_synthetic_rows)

#Guardar metadatos en json
metadata.save_to_json('metadata_TVAE_sin_distincion.json')

# Guardar Los datos sintéticos en un CSV
df_sinteticos.to_csv("sinteticos_estaticos_TVAE_sin_distincion.csv", index=False)
```


ANEXO V: CÓDIGO PARA LOS DATOS AGRUPADOS POR *SHIPTYPE*

A continuación, se muestra el código empleado para la generación de los datos sintéticos agrupados por *shiptype* para las tres técnicas. El ejemplo se corresponde con los conjuntos de datos *basic* por *shiptype* pues se carga el archivo *final_static_data_basic.csv*, pero el código es el mismo para generar los datos sintéticos completos si se carga el archivo *final_static_data.csv* en su lugar.

```
#Importar Librerías
import pandas as pd
import time
from sdv.metadata import Metadata
from sdv.metadata import SingleTableMetadata
from sdv.single_table import TVAESynthesizer, CTGANSynthesizer, GaussianCopulaSynthesizer

# Cargar los datos reales (cambiar archivo para la generación de los datos completos)
df = pd.read_csv("final_static_data_basic.csv")

#Seleccionar las columnas y guardar en un dataframe
columnas_estaticas=["MMSI", "draught", "to_bow", "to_stern", "to_port", "to_starboard",
"shiptype"]
df_estaticos=df[columnas_estaticas]

# Obtener los tipos de barco únicos
ship_types = df_estaticos["shiptype"].unique()

# Diccionario para almacenar los datos generados
synthetic_data_all = {"TVAE": [], "GaussianCopula": [], "CTGAN": []}
computation_times = {"TVAE": {}, "GaussianCopula": {}, "CTGAN": {}}

# Obtener las proporciones reales de cada tipo de barco
shiptype_counts = df_estaticos["shiptype"].value_counts(normalize=True)

# Número total de datos sintéticos a generar
num_total_synthetic = 50000

#Calculo de las proporciones de cada tipo
for shiptype, proportion in shiptype_counts.items():
    num_samples = int(num_total_synthetic * proportion) # Ajustar según la proporción real
    (multiplicando la proporción por el total de datos sintéticos)
    print(f"\n Generando {num_samples} datos para tipo de barco: {shiptype} ({proportion:.2%}
del total)")

# Filtrar los datos del tipo de barco actual
df_ship = df_estaticos[df_estaticos["shiptype"] == shiptype]

# Definir metadatos del dataset
metadata = SingleTableMetadata()
metadata.detect_from_dataframe(df_ship)

# Crear sintetizadores
synthesizers = {
    "TVAE": TVAESynthesizer(metadata),
```

```
"GaussianCopula": GaussianCopulaSynthesizer(metadata),
"CTGAN": CTGANSynthesizer(metadata)
}
#Entrenamiento de cada modelo y medicion del tiempo de computación
for method, synthesizer in synthesizers.items():
    print(f"Entrenando {method} para {shiptype}...")
    start_time = time.time()
    synthesizer.fit(df_ship)
    end_time = time.time()
    computation_times[method][shiptype] = end_time - start_time # Guardar tiempo por barco

    # Generar datos sintéticos según la proporción original
    synthetic_data_ship = synthesizer.sample(num_rows=num_samples)
    synthetic_data_ship["shiptype"] = shiptype # Restaurar la categoría
    synthetic_data_all[method].append(synthetic_data_ship)
# Crear un diccionario para almacenar cada metadata
metadata_all = {
    'TVAE': Metadata.detect_from_dataframe(data=df_estaticos, table_name='AIS_estaticos_TVAE'),
    'GaussianCopula': Metadata.detect_from_dataframe(data=df_estaticos,
table_name='AIS_estaticos_Gaussian'),
    'CTGAN': Metadata.detect_from_dataframe(data=df_estaticos, table_name='AIS_estaticos_CTGAN')
}
# Guardar la metadata a un archivo JSON
for method, data in synthetic_data_all.items():
    metadata_all[method].save_to_json(f"metadata_basic_shiptype_{method}.json")
    print(f"Metadatos de {method} guardados en 'metadata_basic_shiptype_{method}.json'.")

# Guardar Los datos generados en CSVs
for method, data in synthetic_data_all.items():
    synthetic_static_basic= pd.concat(data, ignore_index=True)
    synthetic_static_basic = synthetic_static_basic.sample(frac=1,
random_state=42).reset_index(drop=True) # Mezclar los datos
    synthetic_static_basic.to_csv(f"synthetic_static_basic_{method}.csv", index=False)
    print(f"Datos sintéticos generados con {method} guardados en
'synthetic_static_basic_{method}.csv'.")

for method in ["TVAE", "GaussianCopula", "CTGAN"]:
    # Cargar el CSV generado
    df_synthetic = pd.read_csv(f"synthetic_static_basic_{method}.csv")

    # Mostrar 5 filas
    print(f"Primeras 5 filas de {method}:")
    print(df_synthetic.head(n=5)) # Muestra aleatoria

    print("\n" + "-"*50 + "\n") # Separador entre salidas

# Convertir Los tiempos en un DataFrame para análisis posterior
data_plot = []
for method, times in computation_times.items():
    for shiptype, time_taken in times.items():
        data_plot.append({"Método": method, "Tipo de Barco": shiptype, "Tiempo (s)":
time_taken})

df_times = pd.DataFrame(data_plot)

# Guardar Los tiempos en un CSV
df_times.to_csv("computation_times_basic.csv", index=False)
```

```
# Mostrar tiempos de computación
print("\n Tiempos de computación por método y tipo de barco (basic):")
for method, times in computation_times.items():
    for shiptype, time_taken in times.items():
        print(f"{method} - {shiptype}: {time_taken:.2f} segundos")

# Importación de librerías para representación de tiempos
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

# Cargar el archivo CSV que contiene los tiempos de computación
archivo_csv = "computation_times_basic.csv"
df_times_plot = pd.read_csv(archivo_csv)

# Verificar los primeros registros para asegurarse de que se cargó correctamente
print(df_times_plot.head())

# Obtener los métodos únicos en el DataFrame
metodos_unicos = df_times_plot['Método'].unique()

# Crear gráficos para cada método
for metodo in metodos_unicos:
    plt.figure(figsize=(10, 6))
    sns.barplot(data=df_times_plot[df_times_plot['Método'] == metodo],
                x='Tipo de Barco',
                y='Tiempo (s)',
                palette='viridis')

    plt.title(f'Tiempo de Computación para {metodo}')
    plt.xlabel('Tipo de Barco')
    plt.ylabel('Tiempo de Computación (s)')
    plt.xticks(rotation=45)
    plt.tight_layout() # Ajustar el layout
    plt.show()

# Crear gráfico de barras en conjunto para comparar
plt.figure(figsize=(12, 6))
sns.barplot(data=df_times_plot, x='Tipo de Barco', y='Tiempo (s)', hue='Método',
            palette='viridis')

# Personalización del gráfico
plt.title("Tiempo de Computación por Tipo de Barco y Método para basic")
plt.xlabel('Tipo de Barco')
plt.ylabel('Tiempo de Computación (s)')
plt.legend(title='Método')
plt.xticks(rotation=45) # Rotar las etiquetas del eje x para mejor legibilidad
plt.tight_layout() # Ajustar el layout para que no se solapen los textos

# Mostrar el gráfico
plt.show()
```


ANEXO VI: CÓDIGO MÉTRICAS DE EVALUACIÓN DE LOS DATOS GENERADOS SIN Y CON AGRUPACIÓN POR *SHIPTYPE*

A continuación, se muestra el código para el análisis de calidad empleando las métricas (chi-cuadrado, *ks-test*, distribuciones KDE, estadísticas descriptivas, matrices de correlación, PCA y t-SNE). El ejemplo se corresponde para el archivo de datos sintéticos generados sin distinción por TVAE. Pero se emplea el mismo código es el mismo para todas las técnicas en las dos configuraciones con y sin distinción.

```
#Importar Librerias
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
from sklearn.preprocessing import StandardScaler
from scipy.stats import ks_2samp
import scipy.stats as stats

#Cargar Los datos reales y sinteticos
df_real = pd.read_csv('final_static_data_basic.csv', index_col=0)
df_synth = pd.read_csv('sinteticos_estaticos_TVAE_sin_distincion.csv')

#Comprobar que coinciden las columnas de ambos archivos
print(df_real.columns)
print(df_synth.columns)

# Eliminar las columnas 'MMSI' para la evaluación de calidad y volver a mostrar columnas
df_real_limpio = df_real.drop(columns=['MMSI'])
df_synth_limpio = df_synth.drop(columns=['MMSI'])
print(df_real_limpio.columns)
print(df_synth_limpio.columns)

#Comprobar si se han generado todas las clases
print("Tipos de barco en datos reales:", df_real_limpio["shiptype"].unique())
print("Tipos de barco en datos sintéticos:", df_synth_limpio["shiptype"].unique())
#Porcentaje de cada shiptype
shiptype_percentage = df_synth_limpio["shiptype"].value_counts(normalize=True) * 100
# Crear el gráfico de pastel
plt.figure(figsize=(8, 8))
plt.pie(shiptype_percentage, labels=shiptype_percentage.index, autopct="%1.1f%%",
startangle=140, colors=plt.cm.Paired.colors)
plt.title("Porcentaje ShipType Datos Sinteticos TVAE sin distinción")
plt.show()

# Obtener los tipos de barco únicos
tipos_barco = df_real_limpio["shiptype"].unique()

#Comparacion de distribuciones de cada parametro para cada tipo de barco
for shiptype in tipos_barco:
    print(f"Procesando {shiptype}...") # Esto imprimirá cada barco que se está procesando
    df_real_tipo = df_real_limpio[df_real["shiptype"] == shiptype]
    df_synth_tipo = df_synth_limpio[df_synth["shiptype"] == shiptype]
```

```
for column in df_real_tipo.select_dtypes(include=[np.number]).columns:
    print(f"    - Generando gráfico para {column}") # Para ver qué columnas está procesando
    plt.figure(figsize=(8, 4))
    sns.kdeplot(df_real_tipo[column], label='Real', fill=True, alpha=0.5)

    if not df_synth_tipo.empty:
        sns.kdeplot(df_synth_tipo[column], label='Sintético', fill=True, alpha=0.5)

    plt.title(f'Distribución de {column} para barcos tipo {shiptype}')
    plt.legend()
    plt.show()

# Estadísticas descriptivas
# Agrupar por shiptype
stats_real = df_real_limpio.groupby("shiptype").describe()
stats_synth = df_synth_limpio.groupby("shiptype").describe()
# Mostrar estadísticas de cada tipo de barco
for shiptype in df_synth_limpio["shiptype"].unique():
    print(f"\nEstadísticas para {shiptype} (Datos Sintéticos):\n")
    print(stats_synth.loc[shiptype])

# Graficar las matrices de correlación
for ship in ship_types_synth:
    df_ship_synth = df_synth_limpio[df_synth["shiptype"] ==
    ship].select_dtypes(include=[np.number])

    if not df_ship_synth.empty:
        plt.figure(figsize=(10, 6))
        sns.heatmap(df_ship_synth.corr(), cmap="coolwarm", annot=True, fmt=".2f")
        plt.title(f'Matriz de Correlación - Datos Sintéticos ({ship})')
        plt.show()

#PCA
#Iterar sobre los tipos de barco
for shiptype in tipos_barco:
    print(f"Evaluando tipo de barco: {shiptype}")

    # Filtrar los datos reales y sintéticos
    df_real_tipo = df_real_limpio[df_real["shiptype"] ==
    shiptype].select_dtypes(include=[np.number])
    df_synth_tipo = df_synth_limpio[df_synth["shiptype"] ==
    shiptype].select_dtypes(include=[np.number])

    # Unir los datos reales y sintéticos para PCA
    df_real_tipo["label"] = "Real"
    df_synth_tipo["label"] = "Sintético"
    df_combined = pd.concat([df_real_tipo, df_synth_tipo], ignore_index=True)

    # Normalizar los datos
    scaler = StandardScaler()
    df_scaled = scaler.fit_transform(df_combined.drop(columns=["label"],
    errors="ignore"))

    # Aplicar PCA
    pca = PCA(n_components=2)
    pca_result = pca.fit_transform(df_scaled)

    # Agregar los resultados al DataFrame combinado
```

```

df_combined["PCA1"] = pca_result[:, 0]
df_combined["PCA2"] = pca_result[:, 1]

# Graficar PCA diferenciando reales y sintéticos
plt.figure(figsize=(8, 5))
sns.scatterplot(data=df_combined, x="PCA1", y="PCA2", hue="label", alpha=0.5)
plt.xlabel("PCA 1")
plt.ylabel("PCA 2")
plt.title(f"PCA - {shiptype}")
plt.legend()
plt.show()

# t-SNE para visualización más detallada
# Iterar sobre los tipos de barco
for shiptype in tipos_barco:
    print(f"Evaluando tipo de barco: {shiptype}")

    # Filtrar los datos reales y sintéticos
    df_real_tipo = df_real_limpio[df_real["shiptype"] ==
shiptype].select_dtypes(include=[np.number])
    df_synth_tipo = df_synth_limpio[df_synth["shiptype"] ==
shiptype].select_dtypes(include=[np.number])

    # Normalizar los datos
    scaler = StandardScaler()
    df_scaled_real = scaler.fit_transform(df_real_tipo) # Normalizar los datos reales
    df_scaled_synth = scaler.transform(df_synth_tipo) # Normalizar los datos sintéticos

    # Concatenar los datos para aplicar t-SNE
    df_combined = np.vstack((df_scaled_real, df_scaled_synth)) # Combinar matrices
    labels = ['Real'] * df_scaled_real.shape[0] + ['Sintético'] * df_scaled_synth.shape[0] #
Crear etiquetas

    # Aplicar t-SNE
    tsne = TSNE(n_components=2, random_state=42)
    tsne_result = tsne.fit_transform(df_combined)

    # Crear un DataFrame con los resultados de t-SNE
    df_tsne = pd.DataFrame(tsne_result, columns=['tSNE1', 'tSNE2'])
    df_tsne['label'] = labels # Añadir las etiquetas

    # Graficar t-SNE diferenciando reales y sintéticos
    plt.figure(figsize=(8, 5))
    sns.scatterplot(data=df_tsne, x='tSNE1', y='tSNE2', hue='label', alpha=0.5)
    plt.xlabel("t-SNE 1")
    plt.ylabel("t-SNE 2")
    plt.title(f't-SNE - {shiptype}')
    plt.legend()
    plt.show()

#CHI-CUADRADO
# Contar ocurrencias de cada shiptype en los datos reales y sintéticos
freq_real = df_real_limpio['shiptype'].value_counts().sort_index()
freq_synth = df_synth_limpio['shiptype'].value_counts().sort_index()

# Unir ambas distribuciones en un DataFrame
freq_df = pd.DataFrame({'Real': freq_real, 'Sintético': freq_synth}).fillna(0)

```

```
# Aplicar la prueba de chi-cuadrado
chi2_stat, p_value, dof, expected = stats.chi2_contingency(freq_df)

# Mostrar resultados
print("Estadístico Chi-cuadrado:", chi2_stat)
print("Grados de libertad:", dof)
print("Valor p:", p_value)

# Interpretación
alpha = 0.05 # Nivel de significancia
if p_value < alpha:
    print("Rechazamos la hipótesis nula: hay diferencias significativas entre las distribuciones.")
else:
    print("No se rechaza la hipótesis nula: las distribuciones son similares.")

#KS-Test
# Diccionario para almacenar resultados
ks_results = {}

# Aplicar KS-Test a cada tipo de barco y cada variable numérica
for shiptype in tipos_barco:
    print(f"\nKS-Test para el tipo de barco: {shiptype}\n")

    df_real_tipo = df_real_limpio[df_real_limpio["shiptype"] == shiptype]
    df_synth_tipo = df_synth_limpio[df_synth_limpio["shiptype"] == shiptype]

    # Asegurar que solo se usen columnas numéricas
    columnas_numericas = df_real_tipo.select_dtypes(include=[np.number]).columns

    for column in columnas_numericas:
        if column in df_synth_tipo.columns: # Evitar errores si la columna no existe en los sintéticos
            ks_stat, p_value = ks_2samp(df_real_tipo[column], df_synth_tipo[column])
            print(f"{column}: KS-Statistic={ks_stat:.4f}, P-Value={p_value:.4f}")
            ks_results[(shiptype, column)] = {"KS-Statistic": ks_stat, "P-Value": p_value}
```

Dado que esta última métrica (ks-test) aplicada al conjunto completo de datos no mostraba resultados efectivos, como se explicó en el capítulo 4, se realiza la misma prueba con una muestra de 100 datos, empleando el siguiente código, el cual se emplea para el resto de conjuntos de datos cambiando los archivos:

```
import pandas as pd
import numpy as np
from scipy.stats import ks_2samp

# Cargar los datasets
real_data = pd.read_csv("final_static_data_basic.csv", index_col=0)
synthetic_data = pd.read_csv("sinteticos_estaticos_TVAE_sin_distincion.csv")

# Tamaño de muestra de los datos sintéticos
tamano_muestra = 100 # Puedes modificar este valor

# Columnas numéricas comunes (descarta MMSI y shiptype)
variables_numericas = [
    col for col in real_data.select_dtypes(include=[np.number]).columns
    if col not in ['MMSI', 'shiptype']
]
```

```
# Lista de resultados
resultados = []

# Shiptypes comunes
shiptypes = sorted(set(real_data['shiptype']).intersection(set(synthetic_data['shiptype'])))

for tipo in shiptypes:
    real_tipo = real_data[real_data['shiptype'] == tipo]
    synthetic_tipo = synthetic_data[synthetic_data['shiptype'] == tipo]

    for columna in variables_numericas:
        real_col = real_tipo[columna].dropna()
        synthetic_col = synthetic_tipo[columna].dropna()

        #Muestras aleatorias de tamaño igual
        muestra_real = real_col.sample(n=tamano_muestra, random_state=0)
        muestra_sintetica = synthetic_col.sample(n=tamano_muestra, random_state=42)

        # KS-Test con muestras de igual tamaño
        stat, p = ks_2samp(real_col, muestra_sintetica)
        interpretacion = "Similares" if p > 0.05 else "Diferentes"

        resultados.append({
            'shiptype': tipo,
            'variable': columna,
            'ks_statistic': stat,
            'p_value': p,
            'sample_size': tamano_muestra,
            'resultado': interpretacion
        })

# Guardar resultados
df_resultados = pd.DataFrame(resultados)
df_resultados.to_csv("ks_test_por_muestra_TVAE_sin_distincion.csv", index=False)
```

A continuación, se detalla el código utilizado para realizar las pruebas de preservación de relaciones entre columnas del conjunto de datos sintético completo generado a partir del archivo *final_static_data.csv*.

PRUEBA SENCILLA DE SUMA DE COLUMNAS

```
import pandas as pd
from sdv.single_table import CTGANSynthesizer
from sdv.metadata import Metadata
from sdv.metadata import SingleTableMetadata

#Crear Dtaframe de 20 filas con las columnas ejemplo:
data={
    'Columna1': [1, 1, 3, 4, 5, 7, 3, 5, 2, 1, 3, 3, 4, 5, 6, 4, 9, 5, 3, 1],
    'Columna2': [5, 3, 2, 0, 1, 2, 4, 4, 6, 10, 3, 2, 7, 4, 9, 1, 7, 1, 3, 2]
}

df = pd.DataFrame(data)

# Calcular la tercera columna como la suma de las dos anteriores
df['Columna3'] = df['Columna1'] + df['Columna2']
```

```
# Guardar el DataFrame en un archivo CSV
df.to_csv('prueba.csv', index=False)

print(df.head(20))

archivo_csv="prueba.csv"

#PRUEBA PARA CTGAN
df_CTGAN= pd.read_csv('prueba.csv')
# Definir metadatos del dataset
metadata_CTGAN = SingleTableMetadata()
metadata_CTGAN.detect_from_dataframe(df_CTGAN)
%%time
synthesizer = CTGANSynthesizer(metadata_CTGAN)
synthesizer.fit(df_CTGAN)
num_datos_sinteticos_CTGAN = 20
df_sinteticos_CTGAN = synthesizer.sample(num_datos_sinteticos_CTGAN)
print(df_sinteticos_CTGAN.head(10))

#PRUEBA PARA TVAE
# Cargar archivo
df_TVAE = pd.read_csv("prueba.csv")
# Definir metadatos del dataset
metadata_TVAE = SingleTableMetadata()
metadata_TVAE.detect_from_dataframe(df_TVAE)
# Crear y entrenar el modelo VAE para generación de datos sintéticos
synthesizer = TVAESynthesizer(metadata_TVAE)
synthesizer.fit(df_TVAE)
%%time
# Generar datos sintéticos
num_synthetic_rows = 20
df_sinteticos_TVAE = synthesizer.sample(num_rows=num_synthetic_rows)
print(df_sinteticos_TVAE.head(10))

#PRUEBA PARA GAUSSIAN COPULA
# Cargar archivo
df_GC = pd.read_csv("prueba.csv")
# Definir metadatos del dataset
metadata_GC = SingleTableMetadata()
metadata_GC.detect_from_dataframe(df_GC)
#Crear el sintetizador GaussianCopula
synthesizer = GaussianCopulaSynthesizer(metadata_GC)
#Entrenar el modelo con los datos reales
synthesizer.fit(df_GC)
%%time
#Generar los datos sintéticos
num_datos_sinteticos = 20
df_sinteticos_GC = synthesizer.sample(num_datos_sinteticos)
#Mostrar los primeros registros de datos sintéticos
print(df_sinteticos_GC.head(10))
```

CÁLCULO DEL VALOR *LEN* Y ERROR

```
import pandas as pd
import matplotlib.pyplot as plt

# Cargar el archivo generado por TVAE para el Archivo Final
file_path = "synthetic_static_completo_TVAE.csv"
df_TVAE_final = pd.read_csv(file_path)
```

```
# Asegurar que las columnas necesarias están presentes
required_columns = ["len", "to_bow", "to_stern"]
missing_columns = [col for col in required_columns if col not in df_TVAE_final.columns]

if missing_columns:
    raise ValueError(f"Faltan las siguientes columnas en el archivo: {missing_columns}")

# Calcular la diferencia entre 'len' y 'to_bow + to_stern'
df_TVAE_final["error_len"] = df_TVAE_final["len"] - (df_TVAE_final["to_bow"] +
df_TVAE_final["to_stern"])

# Generar scatterplot para visualizar la relación esperada vs. generada
plt.figure(figsize=(8, 6))
plt.scatter(df_TVAE_final["len"], df_TVAE_final["to_bow"] + df_TVAE_final["to_stern"],
alpha=0.5, label="Datos TVAЕ")
plt.plot([df_TVAE_final["len"].min(), df_TVAE_final["len"].max()],
        [df_TVAE_final["len"].min(), df_TVAE_final["len"].max()],
        color='red', linestyle='dashed', label="Relación esperada (y=x)")
plt.xlabel("Len generado")
plt.ylabel("to_bow + to_stern generado")
plt.title("Comparación de la relación Len vs. (to_bow + to_stern) en TVAЕ")
plt.legend()
plt.show()

# Mostrar estadísticas de los errores
error_stats = df_TVAE_final["error_len"].describe()
print("\n Estadísticas del error en la relación Len:")
print(error_stats)

# Mostrar distribución del error
plt.figure(figsize=(8, 5))
plt.hist(df_ctgan_final["error_len"], bins=30, alpha=0.7, color='blue', edgecolor='black')
plt.axvline(0, color='red', linestyle='dashed', label="Valor esperado (error=0)")
plt.xlabel("Error en len (len - (to_bow + to_stern))")
plt.ylabel("Frecuencia")
plt.title("Distribución de errores en la relación Len = to_bow + to_stern")
plt.legend()
plt.show()
```


ANEXO VII: CÓDIGO PARA EL CÁLCULO DE LOS ATRIBUTOS A PARTIR DE LOS DATOS SINTÉTICOS BASIC Y MÉTRICAS DE ANÁLISIS

Código para el cálculo de todos los atributos tras la generación de los datos sintéticos *basic* por *shiptype* para cada técnica.

```
import pandas as pd

# Definir la función para calcular atributos estáticos
def statics_processing(df):
    df['len'] = df.to_bow + df.to_stern # Eslora
    df['wid'] = df.to_port + df.to_starboard # Manga
    df['ldivw'] = df['len'] / df['wid'] # Cociente entre eslora y manga
    df['ldivd'] = df['len'] / df['draught'] # Cociente entre eslora y calado
    df['wdivd'] = df['wid'] / df['draught'] # Cociente entre manga y calado
    df['area'] = df.len * df.wid # Área de la cubierta aproximada
    df['grith'] = df.len + df.wid # Suma de eslora y manga
    df['aml'] = df.len * df.draught # Área Longitudinal sumergida
    df['amt'] = df.wid * df.draught # Área transversal sumergida
    df['vs'] = df.len * df.draught * df.wid # Volumen sumergido
    df['aol'] = df['to_bow'] / df['len'] # Proporción que supone a sobre la eslora total
    return df

# Rutas de archivos de datos sintéticos
archivos_sinteticos = {
    'TVAE': 'synthetic_static_basic_TVAE.csv',
    'GaussianCopula': 'synthetic_static_basic_GaussianCopula.csv',
    'CTGAN': 'synthetic_static_basic_CTGAN.csv'
}

# Procesar cada archivo y guardar los resultados
for metodo, archivo in archivos_sinteticos.items():
    # Cargar los datos sintéticos
    df_sinteticos = pd.read_csv(archivo)

    # Aplicar la función para calcular atributos estáticos
    df_sinteticos_procesados = statics_processing(df_sinteticos)

    # Guardar los datos procesados en un nuevo archivo CSV
    nombre_archivo_salida = f"atributos_calculados_{metodo}.csv"
    df_sinteticos_procesados.to_csv(nombre_archivo_salida, index=False)

    # Imprimir las primeras 5 filas del DataFrame procesado
    print(f"Primeras filas de los atributos calculados a partir de los datos sinteticos {metodo}:")
    print(df_sinteticos_procesados.head(5))
    print("\n" + "-"*50 + "\n") # Separador para mejor lectura
```

A continuación, se muestra el código generado para la evaluación de calidad de los archivos *basic* con atributos calculados:

```
import pandas as pd
import numpy as np
import scipy.stats as stats
from scipy.stats import ks_2samp
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
from scipy.spatial.distance import jensenshannon
from sklearn.preprocessing import StandardScaler

# Cargar los datasets reales y calculados para cada técnica
real_data = pd.read_csv("final_static_data.csv")
tvae_data = pd.read_csv("atributos_calculados_TVAE.csv")
gaussian_data = pd.read_csv("atributos_calculados_GaussianCopula.csv")
ctgan_data = pd.read_csv("atributos_calculados_CTGAN.csv")

# Función para analizar valores NaN e Infinitos
def analizar_valores_problema(df, nombre):
    # Seleccionar solo columnas numéricas
    df_numeric = df.select_dtypes(include=[np.number])

    # Contar valores NaN e Infinitos
    nan_counts = df_numeric.isna().sum()
    inf_counts = np.isinf(df_numeric).sum()

    # Crear DataFrame con los resultados
    df_issues = pd.DataFrame({
        "Columnas": df_numeric.columns,
        "Valores NaN": nan_counts.values,
        "Valores Infinitos": inf_counts.values
    })

    # Mostrar solo columnas con problemas
    print(f"\nResumen de valores NaN e Infinitos en {nombre}:")

    nan_problems = df_issues[df_issues["Valores NaN"] > 0]
    inf_problems = df_issues[df_issues["Valores Infinitos"] > 0]

    if not nan_problems.empty:
        print("\nColumnas con NaN:")
        print(nan_problems)

    if not inf_problems.empty:
        print("\nColumnas con valores Infinitos:")
        print(inf_problems)

    if nan_problems.empty and inf_problems.empty:
        print("No se encontraron valores NaN ni Infinitos.")

# Ejecutar la función en los tres archivos
analizar_valores_problema(tvae_data, "TVAE")
analizar_valores_problema(gaussian_data, "Gaussian Copula")
analizar_valores_problema(ctgan_data, "CTGAN")
```

```
# Función para comparar cada conjunto sintético con los datos reales
def comparar_con_reales(real_data, synthetic_data, nombre):
    print(f"\nComparación de {nombre} con datos reales:\n")

    # Selección de atributos
    atributos = ["len", "wid", "ldivw", "ldivd", "wdivd", "area", "grith", "aml", "amt", "vs",
"aol"]

    # Limpieza de datos sintéticos: Eliminar NaN en 'aol' y Infinitos en 'ldivw'
    antes = synthetic_data.shape[0]
    synthetic_data = synthetic_data.dropna(subset=["aol"]) # Elimina NaN en 'aol'
    synthetic_data = synthetic_data[~np.isinf(synthetic_data["ldivw"])] # Elimina Infinitos en
'ldivw'
    despues = synthetic_data.shape[0]

    print(f" Eliminadas {antes - despues} filas con NaN o Infinitos en {nombre}.")

    # Limpieza de datos reales: Eliminar cualquier NaN si los hay
    real_data = real_data.dropna(subset=atributos)

    # Comparación de estadísticas descriptivas
    stats_df = []
    for attr in atributos:
        for dataset, dataset_name in zip([real_data, synthetic_data], ["Real", nombre]):
            stats_df.append({
                'Atributo': attr,
                'Dataset': dataset_name,
                'Media': dataset[attr].mean(),
                'Mediana': dataset[attr].median(),
                'Desviación Estándar': dataset[attr].std(),
                'Asimetría': stats.skew(dataset[attr]),
                'Min': dataset[attr].min(),
                'Max': dataset[attr].max()
            })
    stats_df = pd.DataFrame(stats_df)
    print(stats_df)

    # Comparación de distribuciones con KS-Test
    ks_results = []
    for attr in atributos:
        ks_stat, p_value = ks_2samp(real_data[attr], synthetic_data[attr])
        ks_results.append({'Atributo': attr, 'KS-Statistic': ks_stat, 'P-Value': p_value})
    ks_results = pd.DataFrame(ks_results)
    print("\nResultados KS-Test:")
    print(ks_results)

    # Visualización: Histogramas KDE con medias y medianas
    for attr in atributos:
        plt.figure(figsize=(10, 5))
        sns.kdeplot(real_data[attr], label='Real', fill=True, alpha=0.5, color="blue")
        sns.kdeplot(synthetic_data[attr], label=nombre, fill=True, alpha=0.5, color="orange")

    # Calcular medias y medianas
    mean_real, median_real = real_data[attr].mean(), real_data[attr].median()
    mean_synth, median_synth = synthetic_data[attr].mean(), synthetic_data[attr].median()

    # Añadir líneas verticales de media y mediana
```

```
plt.axvline(mean_real, color="green", linestyle="dashed", label=f"Media Real:
{mean_real:.2f}")
plt.axvline(median_real, color="red", linestyle="solid", label=f"Mediana Real:
{median_real:.2f}")
plt.axvline(mean_synth, color="orange", linestyle="dashed", label=f"Media Sintética:
{mean_synth:.2f}")
plt.axvline(median_synth, color="purple", linestyle="solid", label=f"Mediana Sintética:
{median_synth:.2f}")

plt.title(f'Distribución de {attr}')
plt.legend()
plt.show()

# Comparación de correlaciones
correlation_matrices = {
    "Real": real_data[atributos].corr(),
    nombre: synthetic_data[atributos].corr()
}

# Visualización de matrices de correlación
fig, axes = plt.subplots(1, 2, figsize=(14, 6))
for ax, (dataset_name, matrix) in zip(axes.flatten(), correlation_matrices.items()):
    sns.heatmap(matrix, ax=ax, cmap='coolwarm', annot=True, fmt=".2f")
    ax.set_title(f'Matriz de correlación - {dataset_name}')
plt.tight_layout()
plt.show()

# Gestión de valores NaN antes de PCA y t-SNE
real_data_clean = real_data[atributos].dropna()
synthetic_data_clean = synthetic_data[atributos].dropna()

# Verificación antes de PCA y t-SNE
if real_data_clean.empty or synthetic_data_clean.empty:
    print("Error: No hay suficientes datos limpios para PCA/t-SNE. Omitiendo este
análisis.")
    return

# Escalar los datos
scaler = StandardScaler()
combined_data = pd.concat([real_data_clean, synthetic_data_clean])
combined_scaled = scaler.fit_transform(combined_data)

# PCA
pca = PCA(n_components=2)
X_pca = pca.fit_transform(combined_scaled)

df_pca = pd.DataFrame(X_pca, columns=["PCA 1", "PCA 2"])
df_pca["Tipo"] = ["Real"] * len(real_data_clean) + [nombre] * len(synthetic_data_clean)

plt.figure(figsize=(8, 6))
sns.scatterplot(data=df_pca, x="PCA 1", y="PCA 2", hue="Tipo", alpha=0.6)
plt.title(f"PCA - Comparación de {nombre} vs. Datos Reales")
plt.legend()
plt.show()

# t-SNE
tsne = TSNE(n_components=2, perplexity=30, random_state=42)
X_tsne = tsne.fit_transform(combined_scaled)
```

```
df_tsne = pd.DataFrame(X_tsne, columns=["t-SNE 1", "t-SNE 2"])
df_tsne["Tipo"] = ["Real"] * len(real_data_clean) + [nombre] * len(synthetic_data_clean)

plt.figure(figsize=(8, 6))
sns.scatterplot(data=df_tsne, x="t-SNE 1", y="t-SNE 2", hue="Tipo", alpha=0.6)
plt.title(f"t-SNE - Comparación de {nombre} vs. Datos Reales")
plt.legend()
plt.show()

print("\nComparación finalizada.\n")

# Comparaciones independientes para cada técnica
comparar_con_reales(real_data, tvae_data, "TVAE")
comparar_con_reales(real_data, gaussian_data, "GaussianCopula")
comparar_con_reales(real_data, ctgan_data, "CTGAN")
```


ANEXO VIII: CÓDIGO PARA *DATA AUGMENTATION*

```
import pandas as pd
from imblearn.over_sampling import SMOTE

# Cargar el dataset
file_path = "final_static_data_basic.csv"
df = pd.read_csv(file_path)

# Verificar las categorías en 'shiptype'
print("Distribución original de shiptype:")
print(df['shiptype'].value_counts())

# Separar características (X) y variable objetivo (y)
X = df.drop(columns=['shiptype']) # Eliminar la variable objetivo para aplicar SMOTE
y = df['shiptype'] # Variable objetivo

# Calcular el 50% más para la clase Passenger
num_passenger = df[df['shiptype'] == 'ShipType.Passenger'].shape[0]
num_passenger_augmented = int(num_passenger * 1.5)

# Aplicar SMOTE con incremento del 50% en Passenger
smote = SMOTE(sampling_strategy={'ShipType.Passenger': num_passenger_augmented},
random_state=42)
X_resampled, y_resampled = smote.fit_resample(X, y)

# Crear un nuevo DataFrame con los datos aumentados
df_augmented = pd.concat([X_resampled, y_resampled], axis=1)

# Verificar la nueva distribución de shiptype
print("\nDistribución después de SMOTE:")
print(df_augmented['shiptype'].value_counts())

# Imprimir las cinco primeras filas del dataset generado
print("\nPrimeras 5 filas del dataset aumentado:")
print(df_augmented.head())

# Guardar el dataset aumentado
df_augmented.to_csv("final_static_basic_augmented.csv", index=False)
print("\nArchivo con SMOTE guardado correctamente como final_static_basic_augmented.csv")
```

Se presenta también en este anexo, el código generado para evaluar la similitud de los datos passenger generados con los reales.

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.decomposition import PCA
from sklearn.manifold import TSNE
from sklearn.preprocessing import StandardScaler
from scipy.stats import ks_2samp
import scipy.stats as stats
```

```
#Cargar los datos reales y sinteticos
df_real = pd.read_csv('final_static_data_basic.csv', index_col=0)
df_synth = pd.read_csv('final_static_basic_augmented.csv', index_col=0)

print(df_real.columns)
print(df_synth.columns)

print("Tipos de barco en datos reales:", df_real["shiptype"].unique())
print("Tipos de barco en datos sintéticos:", df_synth["shiptype"].unique())

#Porcentaje de cada shiptype
shiptype_percentage = df_synth["shiptype"].value_counts(normalize=True) * 100

import matplotlib.pyplot as plt
# Crear el gráfico de pastel
plt.figure(figsize=(8, 8))
plt.pie(shiptype_percentage, labels=shiptype_percentage.index, autopct="%1.1f%%",
startangle=140, colors=plt.cm.Paired.colors)
plt.title("Porcentaje ShipType Datos Sinteticos TVAE por tipo de buque")
plt.show()

# Ver cuántos barcos 'Passenger' hay en cada uno
print("Barcos Passenger (reales):", (df_real['shiptype'] == 'ShipType.Passenger').sum())
print("Barcos Passenger (aumentados):", (df_synth['shiptype'] == 'ShipType.Passenger').sum())

# Limpiar el campo 'shiptype' extrayendo solo el nombre (después del punto)
df_real['shiptype'] = df_real['shiptype'].astype(str).str.split('.').str[-1]
df_synth['shiptype'] = df_synth['shiptype'].astype(str).str.split('.').str[-1]

#Filtrar clase Passenger
real_passenger = df_real[df_real['shiptype'] == 'Passenger']
augmented_passenger = df_synth[df_synth['shiptype'] == 'Passenger']

# Variables numéricas comunes
variables_numericas = ['draught', 'to_bow', 'to_stern', 'to_port', 'to_starboard']

# Tabla estadística comparativa

resumen = []

for var in variables_numericas:
    real_vals = real_passenger[var].dropna()
    smote_vals = augmented_passenger[var].dropna()

    if real_vals.nunique() > 1 and smote_vals.nunique() > 1:
        resumen.append({
            'Variable': var,
            'Media Real': real_vals.mean(),
            'Media SMOTE': smote_vals.mean(),
            'Std Real': real_vals.std(),
            'Std SMOTE': smote_vals.std(),
        })

tabla_estadisticas = pd.DataFrame(resumen)
print("\n Tabla comparativa Passenger (real vs SMOTE):")
print(tabla_estadisticas.round(3))
```

```
# Graficar curvas KDE
sns.set(style="whitegrid")

for var in variables_numericas:
    real_vals = real_passenger[var].dropna()
    smote_vals = augmented_passenger[var].dropna()

    if real_vals.nunique() > 1 and smote_vals.nunique() > 1:
        plt.figure(figsize=(8, 4))
        sns.kdeplot(real_vals, label='Passenger Real', fill=True, alpha=0.5)
        sns.kdeplot(smote_vals, label='Passenger SMOTE', fill=True, alpha=0.5)
        plt.title(f'Distribución KDE - {var}')
        plt.xlabel(var)
        plt.ylabel('Densidad')
        plt.legend()
        plt.show()
```


ANEXO IX: CÓDIGO PARA ANÁLISIS DE UTILIDAD CON *RANDOM FOREST*

Se muestra el código generado para realizar la prueba TTR. El ejemplo se corresponde con el archivo *final_static_data_basic.csv*, pero es el mismo código para el archivo *final_static_data.csv*.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler, label_binarize
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report, roc_auc_score, confusion_matrix

# Cargar los datos reales o los generados tras Data Augmentation y eliminar columnas
innecesarias
real_data = pd.read_csv("final_static_data_basic.csv").drop(columns=["Unnamed: 0", "MMSI"],
errors='ignore')

# Separar datos en train y test
df_train, df_test = train_test_split(real_data, random_state=43)
x_train = df_train.drop(columns=['shiptype'])
y_train = df_train['shiptype']
x_test = df_test.drop(columns=['shiptype'])
y_test = df_test['shiptype']

# Binarizar etiquetas para calcular AUC-ROC
classes = sorted(y_test.unique())
y_test_bin = label_binarize(y_test, classes=classes)

# Crear modelo con pipeline
pipe = Pipeline([
    ('scaler', MinMaxScaler()),
    ('classifier', RandomForestClassifier(n_estimators=100, random_state=42))
])

# Entrenar el modelo con los datos reales
model = pipe.fit(x_train, y_train)
y_pred = model.predict(x_test)

# Calcular matriz de confusión
conf_matrix = confusion_matrix(y_test, y_pred)

# Verificar si el clasificador soporta predict_proba
if hasattr(model.named_steps['classifier'], "predict_proba"):
    y_pred_proba = model.predict_proba(x_test)
    # Calcular AUC-ROC
    try:
        if len(classes) > 2:
            auc_roc = roc_auc_score(y_test_bin, y_pred_proba, multi_class='ovr')
        else:
            auc_roc = roc_auc_score(y_test_bin.ravel(), y_pred_proba[:, 1])
```

```
except ValueError:
    print("No se pudo calcular AUC-ROC para los datos reales")
    auc_roc = np.nan
else:
    auc_roc = np.nan

# Obtener métricas
report = classification_report(y_test, y_pred, zero_division=0, output_dict=True)
accuracy = report['accuracy']
f1 = report['weighted avg']['f1-score']
precision = report['weighted avg']['precision']
recall = report['weighted avg']['recall']

# Mostrar resultados
results = pd.DataFrame({
    "Dataset": ["Datos Reales"],
    "Accuracy": [accuracy],
    "F1-score": [f1],
    "Precision": [precision],
    "Recall": [recall],
    "AUC-ROC": [auc_roc],
})
print(results)

# Graficar matriz de confusión
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix, annot=True, fmt='d', cmap='Blues', xticklabels=classes,
            yticklabels=classes)
plt.xlabel("Predicción")
plt.ylabel("Real")
plt.title("Matriz de Confusión - Datos Reales")
plt.show()

# Obtener importancia de características
feature_importances = model.named_steps['classifier'].feature_importances_
feature_importance_df = pd.DataFrame({
    'Característica': x_train.columns,
    'Importancia': feature_importances
}).sort_values(by='Importancia', ascending=False)

# Graficar importancia de características
plt.figure(figsize=(12, 6))
sns.barplot(data=feature_importance_df, x='Característica', y='Importancia', palette='viridis')
plt.xticks(rotation=70, ha='right')
plt.xlabel("Características")
plt.ylabel("Importancia")
plt.title("Importancia de las Características en RandomForest - Datos Reales")
plt.show()
```

Para la prueba TSTR se generan dos códigos iguales uno para los datos sintéticos generados a partir del archivo *final_static_data_basic.csv* y otro para los generados a partir del archivo *final_static_data.csv*, pero cargando los archivos correspondientes a cada configuración.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import MinMaxScaler, label_binarize
from sklearn.pipeline import Pipeline
from sklearn.metrics import classification_report, roc_auc_score, confusion_matrix

# Cargar los datos reales
real_data = pd.read_csv("final_static_data.csv" o "final_static_data_basic.csv")
.drop(columns=["MMSI"], errors="ignore")

# Reemplazar valores infinitos por NaN y luego eliminarlos
real_data.replace([np.inf, -np.inf], np.nan, inplace=True)
real_data.dropna(inplace=True)

# Datasets sintéticos por técnica
techniques = ["TVAE", "GaussianCopula", "CTGAN"]
dataset_variants = [
    "Completo shiptype", "Basic shiptype con atributos calculados"
]

synthetic_data_dict = {}
for tech in techniques:
    synthetic_data_dict[tech] = {
        "Completo shiptype":
pd.read_csv(f"synthetic_static_completo_{tech}.csv").drop(columns=["MMSI"],
errors="ignore").replace([np.inf, -np.inf], np.nan).dropna(),
        "Basic shiptype con atributos calculados":
pd.read_csv(f"atributos_calculados_{tech}.csv").drop(columns=["MMSI"],
errors="ignore").replace([np.inf, -np.inf], np.nan).dropna(),
    }

# o cargar los sintéticos que se corresponden con el archivo basic
synthetic_data_dict = {}
for tech in techniques:
    synthetic_data_dict[tech] = {
        "Basic sin agrupacion":
pd.read_csv(f"sinteticos_estaticos_{tech}_sin_distincion.csv").drop(columns=["MMSI"],
errors="ignore").replace([np.inf, -np.inf], np.nan).dropna(),
        "Basic shiptype":
pd.read_csv(f"synthetic_static_basic_{tech}.csv").drop(columns=["MMSI"],
errors="ignore").replace([np.inf, -np.inf], np.nan).dropna(),
    }

# Separar datos reales en train y test
df_train, df_test = train_test_split(real_data, random_state=43)
x_test = df_test.drop(columns=['shiptype'])
y_test = df_test['shiptype']

# Binarizar etiquetas para calcular AUC-ROC
```

```
classes = sorted(y_test.unique())
y_test_bin = label_binarize(y_test, classes=classes)

# Almacenar resultados
results = []
feature_importances = {}
conf_matrices = {}

for tech, datasets in synthetic_data_dict.items():
    for name, syn_data in datasets.items():
        print(f"Evaluando: {tech} - {name}")

        # Separar características y etiquetas en los datos sintéticos
        x_train = syn_data.drop(columns=['shiptype'])
        y_train = syn_data['shiptype']

        # Asegurar que las características coincidan con x_test
        x_train, x_test = x_train.align(x_test, join='inner', axis=1)

        # Crear modelo con pipeline
        pipe = Pipeline([
            ('scaler', MinMaxScaler()),
            ('classifier', RandomForestClassifier(n_estimators=100, random_state=42))
        ])

        model = pipe.fit(x_train, y_train)
        y_pred = model.predict(x_test)

        # Calcular matriz de confusión
        conf_matrices[f"{tech} - {name}"] = confusion_matrix(y_test, y_pred)

        # Verificar si el clasificador soporta predict_proba
        if hasattr(model.named_steps['classifier'], "predict_proba"):
            y_pred_proba = model.predict_proba(x_test)
            # Calcular AUC-ROC
            try:
                if len(classes) > 2:
                    auc_roc = roc_auc_score(y_test_bin, y_pred_proba, multi_class='ovr')
                else:
                    auc_roc = roc_auc_score(y_test_bin.ravel(), y_pred_proba[:, 1])
            except ValueError:
                print(f"No se pudo calcular AUC-ROC para {tech} - {name}")
                auc_roc = np.nan
        else:
            auc_roc = np.nan

        # Obtener métricas
        report = classification_report(y_test, y_pred, zero_division=0, output_dict=True)
        accuracy = report['accuracy']
        f1 = report['weighted avg']['f1-score']
        precision = report['weighted avg']['precision']
        recall = report['weighted avg']['recall']

        results.append({
            "Técnica": tech,
            "Dataset": name,
            "Accuracy": accuracy,
            "F1-score": f1,
```

```

        "Precision": precision,
        "Recall": recall,
        "AUC-ROC": auc_roc,
    })

    # Obtener importancia de características
    forest = model.named_steps['classifier']
    importances = forest.feature_importances_
    feature_importances[f"{tech} - {name}"] = pd.Series(importances,
index=x_train.columns).sort_values(ascending=False)

# Mostrar resultados en tabla
results_df = pd.DataFrame(results)
print(results_df)

# Graficar importancia de características para cada dataset sintético
plt.figure(figsize=(20, 6))
for key, fi in feature_importances.items():
    fi = fi.reset_index()
    fi.columns = ['att', 'importance']
    sns.barplot(data=fi, x='att', y='importance', label=key)
    plt.xticks(rotation=70, ha='right')
    plt.xlabel("Características")
    plt.ylabel("Importancia")
    plt.title("Importancia de las Características en RandomForest - Comparación de Técnicas")
    plt.legend()
    plt.subplots_adjust(bottom=0.3)
    plt.show()

# Graficar matrices de confusión
for key, cm in conf_matrices.items():
    plt.figure(figsize=(8, 6))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=classes, yticklabels=classes)
    plt.xlabel("Predicción")
    plt.ylabel("Real")
    plt.title(f"Matriz de Confusión - {key}")
    plt.show()

```

También se incluye en este anexo el código para evaluar el impacto del *Data Augmentation* en el modelo de aprendizaje automático, con el análisis de la curva AUC-ROC específico para *passenger*:

```

from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import MinMaxScaler
from sklearn.pipeline import Pipeline
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_curve, auc

# Separar train y test desde df_real
df_train, df_test = train_test_split(df_real, random_state=43)

x_train = df_train.drop(columns=['shiptype', 'MMSI'])
y_train = (df_train['shiptype'] == 'Passenger').astype(int)

x_test = df_test.drop(columns=['shiptype', 'MMSI'])
y_test = (df_test['shiptype'] == 'Passenger').astype(int)

```

```
# Pipeline: escalado + RandomForest
pipe = Pipeline([
    ('scaler', MinMaxScaler()),
    ('classifier', RandomForestClassifier(n_estimators=100, random_state=42))
])

# Entrenar y predecir
model = pipe.fit(x_train, y_train)
y_score = model.predict_proba(x_test)[:, 1]

# Calcular curva ROC y AUC
fpr, tpr, thresholds = roc_curve(y_test, y_score)
auc_passenger = auc(fpr, tpr)

print(f"\n AUC-ROC binario para clase Passenger: {auc_passenger:.4f}")

# Separar train y test desde df_synth (con SMOTE)
df_train_s, df_test_s = train_test_split(df_synth, random_state=43)

x_train_s = df_train_s.drop(columns=['shiptype', 'MMSI'])
y_train_s = (df_train_s['shiptype'] == 'Passenger').astype(int)

x_test_s = df_test_s.drop(columns=['shiptype', 'MMSI'])
y_test_s = (df_test_s['shiptype'] == 'Passenger').astype(int)

# Pipeline
pipe_s = Pipeline([
    ('scaler', MinMaxScaler()),
    ('classifier', RandomForestClassifier(n_estimators=100, random_state=42))
])

# Entrenar modelo
model_s = pipe_s.fit(x_train_s, y_train_s)
y_score_s = model_s.predict_proba(x_test_s)[:, 1]

# Calcular curva ROC y AUC
fpr_s, tpr_s, _ = roc_curve(y_test_s, y_score_s)
auc_passenger_s = auc(fpr_s, tpr_s)

print(f"\n AUC-ROC binario para clase Passenger (SMOTE): {auc_passenger_s:.4f}")

#Grafica comparativa de curvas AUC ROC para passenger entre Los datos reales y con SMOTE
plt.figure(figsize=(8, 6))
plt.plot(fpr, tpr, color='red', lw=2, label=f'Real (AUC = {auc_passenger_r:.2f})')
plt.plot(fpr_s, tpr_s, color='blue', lw=2, label=f'SMOTE (AUC = {auc_passenger_s:.2f})')
plt.plot([0, 1], [0, 1], color='gray', linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Comparativa ROC - Clase Passenger (Real vs SMOTE)')
plt.legend(loc="lower right")
plt.grid(True)
plt.tight_layout()
plt.show()
```