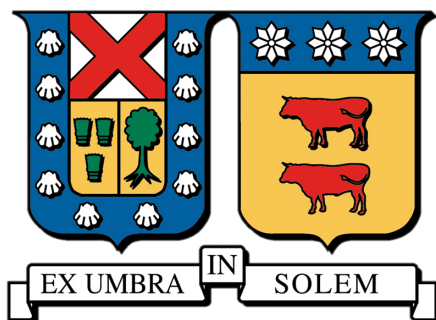


**UNIVERSIDAD TÉCNICA FEDERICO SANTA MARÍA**  
**DEPARTAMENTO DE ELECTRÓNICA**  
**VALPARAÍSO - CHILE**



**“Aplicación de modelos generativos en salud para la generación de mamografías y registros clínicos sintéticos.”**

**NICOLÁS IGNACIO VENEROS GALLARDO**

**MEMORIA DE TITULACIÓN PARA OPTAR AL TÍTULO DE INGENIERO CIVIL  
ELECTRÓNICO, MENCIÓN COMPUTADORES**

**PROFESOR GUIA:                    MARCOS ZUÑIGA**

**PROFESORES CORREFERENTES:   MAURICIO ARAYA**

**ENERO - 2023**

# Abstract

Para la investigación y desarrollo de sistemas de aprendizaje de máquinas es necesario tener una base de datos en común que permita entrenar y comparar distintos modelos. Sin embargo, en el área de salud existen barreras legales que al buscar proteger la integridad y privacidad de los pacientes prohíbe la publicación de dicha información, obstruyendo las labores de investigación.

Como solución al problema anteriormente planteado se propone el uso de modelos generativos que a partir de un dataset inicial generen información sintética que mantenga las características estadísticas originales, así aumentando el volumen de datos útiles disponibles para investigación.

En este informe se presenta el estudio de las *Generative Adversarial Networks* (GANs) como alternativa para generación de mamografías sintéticas, así como el desarrollo y aplicación de un software basado en la biblioteca *Synthetic Data Vault* (SDV) para generación de tablas sintéticas a partir archivos '.csv' o '.xlsx'.

# Índice general

|                                                             |          |
|-------------------------------------------------------------|----------|
| <b>1. Introducción</b>                                      | <b>1</b> |
| 1.1. Problema . . . . .                                     | 1        |
| 1.2. Acercamiento a la solución . . . . .                   | 1        |
| 1.3. Hipótesis . . . . .                                    | 2        |
| 1.4. Objetivos . . . . .                                    | 2        |
| 1.5. Estructura de la memoria . . . . .                     | 3        |
| <b>2. Estado del arte</b>                                   | <b>4</b> |
| 2.1. Soluciones existentes . . . . .                        | 4        |
| 2.2. Técnicas relevantes . . . . .                          | 5        |
| 2.2.1. GAN . . . . .                                        | 5        |
| 2.2.2. DCGAN [Radford et al., 2015] . . . . .               | 6        |
| 2.2.3. WGAN [Arjovsky et al., 2017] . . . . .               | 6        |
| 2.2.4. ProGAN [Karras et al., 2017] . . . . .               | 6        |
| 2.2.5. The Synthetic Data Vault . . . . .                   | 7        |
| 2.3. Discusión . . . . .                                    | 7        |
| <b>3. Desarrollo de la solución</b>                         | <b>9</b> |
| 3.1. Modelo ProGAN para generación de mamografías . . . . . | 9        |
| 3.2. Generador de tablas sintéticas . . . . .               | 12       |
| 3.2.1. Entradas . . . . .                                   | 13       |
| 3.2.2. Módulos . . . . .                                    | 18       |
| 3.2.3. Código . . . . .                                     | 21       |

|                                                                   |           |
|-------------------------------------------------------------------|-----------|
| <b>4. Validación</b>                                              | <b>27</b> |
| 4.1. Resultados tablas sintéticas: Módulo de evaluación . . . . . | 29        |
| 4.1.1. <i>Single table data</i> . . . . .                         | 29        |
| 4.1.2. TMVA . . . . .                                             | 32        |
| 4.1.3. HEM-FROT-HEM1 . . . . .                                    | 34        |
| 4.1.4. HEM-BAC . . . . .                                          | 36        |
| 4.1.5. <i>Relational table data</i> . . . . .                     | 41        |
| 4.2. Resultados mamografías sintéticas: Encuesta . . . . .        | 51        |
| <b>5. Conclusiones</b>                                            | <b>55</b> |

# Índice de cuadros

|                                                                                                                                                                                                        |    |
|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 3.1. Resultados de rendimiento para modelos generativos correspondientes a modalidad <i>single table data</i> . . . . .                                                                                | 12 |
| 3.2. Tipos de restricción y sus respectivos argumentos de entrada. . . . .                                                                                                                             | 14 |
| 3.3. Funciones desarrolladas para su implementación en el software. . . . .                                                                                                                            | 21 |
| 3.4. Variables de configuración contenidas en CONFIG.py . . . . .                                                                                                                                      | 22 |
| 4.1. Estructura de la tabla <i>laboratorio.csv</i> . . . . .                                                                                                                                           | 29 |
| 4.2. Reporte Column Shapes para la tabla <i>laboratorio.csv</i> . . . . .                                                                                                                              | 30 |
| 4.3. Reporte Column Shapes para fragmento correspondiente a la categoría <i>TMVA</i> de la tabla <i>laboratorio.csv</i> al ejecutar el programa con submódulo table divider activado. . . . .          | 32 |
| 4.4. Reporte Column Shapes para fragmento correspondiente a la categoría <i>HEM-FROT-HEM1</i> de la tabla <i>laboratorio.csv</i> al ejecutar el programa con submódulo table divider activado. . . . . | 34 |
| 4.5. Reporte Column Shapes para fragmento correspondiente a la categoría <i>HEM-BAC</i> de la tabla <i>laboratorio.csv</i> al ejecutar el programa con submódulo table divider activado. . . . .       | 36 |
| 4.6. Estructura de la hoja <i>pacientes</i> contenida en el libro excel <i>real.xlsx</i> . . . . .                                                                                                     | 41 |
| 4.7. Estructura de la hoja <i>tratamientos</i> contenida en el libro excel <i>real.xlsx</i> . . . . .                                                                                                  | 41 |
| 4.8. Primeras 8 filas de las columnas <i>fecha_defuncion</i> y <i>lugar_fallecido</i> de la tabla <i>pacientes</i> de los datos reales. . . . .                                                        | 44 |
| 4.9. Primeras 8 filas de las columnas <i>fecha_defuncion</i> y <i>lugar_fallecido</i> de la tabla <i>pacientes</i> de los datos reales transformados. . . . .                                          | 44 |
| 4.10. Primeras 8 filas de las columnas <i>fecha_defuncion</i> y <i>lugar_fallecido</i> de la tabla <i>pacientes</i> de los datos sintéticos. . . . .                                                   | 48 |
| 4.11. Respuestas de la segunda sección de la encuesta del encuestado sin formación. . . . .                                                                                                            | 51 |

|                                                                                                                           |    |
|---------------------------------------------------------------------------------------------------------------------------|----|
| 4.12. Respuestas de la tercera sección de la encuesta del encuestado sin formación. . . . .                               | 51 |
| 4.13. Respuestas de la segunda sección de la encuesta del primer encuestado con formación<br>no especializada. . . . .    | 52 |
| 4.14. Respuestas de la tercera sección de la encuesta del primer encuestado con formación<br>no especializada. . . . .    | 52 |
| 4.15. Respuestas de la segunda sección de la encuesta del segundo encuestado con forma-<br>ción no especializada. . . . . | 52 |
| 4.16. Respuestas de la tercera sección de la encuesta del segundo encuestado con formación<br>no especializada. . . . .   | 53 |

# Índice de figuras

|                                                                                                                                                                                                                  |    |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 2.1. Arquitectura de sistema GAN. . . . .                                                                                                                                                                        | 5  |
| 3.1. Primeros resultados: DCGAN y WGAN . . . . .                                                                                                                                                                 | 9  |
| 3.2. Primeros resultados: ProGAN . . . . .                                                                                                                                                                       | 11 |
| 3.3. Resultados finales: ProGAN . . . . .                                                                                                                                                                        | 11 |
| 3.4. Esquema de la estructura del software generador de tablas. . . . .                                                                                                                                          | 13 |
| 4.1. Representación gráfica del reporte Column Shapes para <i>laboratorio.csv</i> . . . . .                                                                                                                      | 30 |
| 4.2. Detalles de la distribución de la columna <i>DFED_Resultado</i> . . . . .                                                                                                                                   | 31 |
| 4.3. Detalles de la distribución de la columna <i>DFED_CodigoTest</i> . . . . .                                                                                                                                  | 31 |
| 4.4. Distribución de la columna <i>DFED_CodigoTest</i> al ejecutar el programa con el sub-<br>módulo table divider activado para el fragmento correspondiente a la categoría <i>TMVA</i> . . . . .               | 33 |
| 4.5. Distribución de la columna <i>DFED_Resultado</i> al ejecutar el programa con el sub-módulo<br>table divider activado para el fragmento correspondiente a la categoría <i>TMVA</i> . . . . .                 | 33 |
| 4.6. Distribución de la columna <i>DFED_ResultadoNumerico</i> al ejecutar el programa con el<br>sub-módulo table divider activado para el fragmento correspondiente a la categoría<br><i>TMVA</i> . . . . .      | 34 |
| 4.7. Distribución de la columna <i>DFED_CodigoTest</i> al ejecutar el programa con el sub-<br>módulo table divider activado para el fragmento correspondiente a la categoría <i>HEM-<br/>FROT-HEM1</i> . . . . . | 35 |
| 4.8. Distribución de la columna <i>DFED_Resultado</i> al ejecutar el programa con el sub-módulo<br>table divider activado para el fragmento correspondiente a la categoría <i>HEM-FROT-<br/>HEM1</i> . . . . .   | 35 |

|                                                                                                                                                                                                                |    |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----|
| 4.9. Distribución de la columna <i>DFED_ResultadoNumerico</i> al ejecutar el programa con el sub-módulo table divider activado para el fragmento correspondiente a la categoría <i>HEM-FROT-HEM1</i> . . . . . | 36 |
| 4.10. Distribución de la columna <i>DFED_CodigoTest</i> al ejecutar el programa con el sub-módulo table divider activado para el fragmento correspondiente a la categoría <i>HEM-BAC</i> . . . . .             | 37 |
| 4.11. Distribución de la columna <i>DFED_Resultado</i> al ejecutar el programa con el sub-módulo table divider activado para el fragmento correspondiente a la categoría <i>HEM-BAC</i> . . . . .              | 37 |
| 4.12. Distribución de la columna <i>DFED_ResultadoNumerico</i> al ejecutar el programa con el sub-módulo table divider activado para el fragmento correspondiente a la categoría <i>HEM-BAC</i> . . . . .      | 38 |
| 4.13. Representación gráfica del reporte Column Pair trends para tabla <i>laboratorio.csv</i> con sub-módulo table divider desactivado. . . . .                                                                | 39 |
| 4.14. Representación gráfica del reporte Column Pair trends para tabla <i>laboratorio.csv</i> con sub-módulo table divider activado. . . . .                                                                   | 39 |
| 4.15. Representación gráfica del reporte Column Pair trends correspondiente al fragmento <i>TMVA</i> para la tabla <i>laboratorio.csv</i> con sub-módulo table divider activado. . . . .                       | 40 |
| 4.16. Representación gráfica del reporte Column Shapes para tabla <i>pacientes</i> . . . . .                                                                                                                   | 42 |
| 4.17. Representación gráfica del reporte Parent Child Relationships para la tabla <i>pacientes</i> . . . . .                                                                                                   | 42 |
| 4.18. Detalle de la distribución de la columna <i>COMUNA</i> de la tabla <i>pacientes</i> . . . . .                                                                                                            | 43 |
| 4.19. Detalle de la distribución de la columna <i>SEX</i> de la tabla <i>pacientes</i> . . . . .                                                                                                               | 44 |
| 4.20. Detalle de la distribución de la columna <i>fecha_defuncion</i> de la tabla <i>pacientes</i> . . . . .                                                                                                   | 45 |
| 4.21. Detalle de la distribución de la columna <i>FECHA_NACIMIENTO</i> de la tabla <i>pacientes</i> . . . . .                                                                                                  | 45 |
| 4.22. Detalle de la distribución generada para el par de columnas <i>COMUNA-REGION</i> de la tabla <i>pacientes</i> . . . . .                                                                                  | 46 |
| 4.23. Detalle de la distribución generada para el par de columnas <i>COMUNA-REGION</i> de la tabla <i>pacientes</i> al implementar restricción del tipo <b>FixedC</b> . . . . .                                | 47 |
| 4.24. Detalle de la distribución generada para el par de columnas <i>FECHA_INICIO-FECHA_FIN</i> de la tabla <i>pacientes</i> . . . . .                                                                         | 49 |
| 4.25. Detalle de la distribución generada para el par de columnas <i>FECHA_INICIO-FECHA_FIN</i> de la tabla <i>pacientes</i> al implementar restricción del tipo <b>Inequality</b> . . . . .                   | 49 |

# Capítulo 1

## Introducción

### 1.1. Problema

En el contexto de la salud existe escasez de datos públicos que sean de alta calidad y útiles para investigación, dado que toda la información relacionada a pacientes que se guarda en hospitales, clínicas y otras instalaciones relacionadas a la medicina está sujeta a estrictas regulaciones que prohíben o restringen su publicación con el fin de proteger la privacidad del paciente.

### 1.2. Acercamiento a la solución

Una solución a este problema es la de-identificación o anonimización de los datos [Yoon et al., 2020], en donde se desprenden todos los identificadores personales de los documentos. Sin embargo, esto suele conducir al desorden y pérdida de coherencia de la información.

Alternativamente existe la generación sintética de datos a través de algoritmos de aprendizaje de máquinas usando como referencia datos reales, tal como se discute en [Abufadda and Mansour, 2021]. Esto tiene por objetivo producir información de alta calidad cuya publicación no tenga impedimentos, permitiendo aumentar el volumen de datos útil disponible para investigación. Un ejemplo de aplicación es en situaciones donde se dispone de escasa información, como casos de enfermedades raras.

En este trabajo se aborda esta alternativa en torno a 2 tipos de datos que cumplen un rol fundamental en la medicina moderna: imágenes y tablas de datos clínicos. Las imágenes médicas tienden a ser una pieza clave en la identificación de anomalías o enfermedades en el paciente, especialmente cuando estas existen al interior del cuerpo. Por otro lado, las tablas sirven como registros de información sobre los pacientes y/o actividades de importancia, tales como el número de sesiones, tipo de

enfermedad, procedimientos realizados, etc.

Dada la vasta diversidad de imágenes médicas existentes se decide realizar un estudio enfocado a mamografías con y sin presencia de cáncer de mama. Esto debido a la importancia del tema a nivel global, siendo este cáncer uno de los más comunes junto al cáncer de piel.

El modelo generativo a implementar para las imágenes se denomina GAN, una herramienta de aprendizaje de máquinas que se ha vuelto relevante en los últimos años gracias a su capacidad de crear imágenes sintéticas que conservan las características estadísticas y visuales del dataset usado en su entrenamiento. Así, se pretende evaluar las limitaciones actuales de esta herramienta para la creación de imágenes sintéticas de mamografías con y sin presencia de cáncer.

Por otro lado, las tablas de datos clínicos pueden tener una gran variabilidad en cuanto a los datos que almacenan, por lo que para generar de forma genérica datos sintéticos es necesario un programa que sea capaz de identificar los elementos de cada columna y las relaciones existentes entre estas, para cada set de datos que se desee aleatorizar y reproducir. Para esta parte del trabajo se tiene por objetivo desarrollar dicho software utilizando de base la biblioteca *Synthetic Data Vault* (SDV), donde se le facilita al usuario su uso mediante una interfaz.

### 1.3. Hipótesis

Los modelos generativos de vanguardia pueden generar información sintética de calidad en el ámbito oncológico. ¿Qué tan realistas pueden ser las imágenes generadas por técnicas modernas de machine learning? ¿Se pueden obtener resultados con las características necesarias para poder ser usados en estudios médicos?

### 1.4. Objetivos

**Objetivo general:** Determinar el alcance de los modelos generativos para la generación de información sintética en el contexto de la salud, específicamente en torno a mamografías y registros clínicos.

**Objetivos específicos:**

- Desarrollar un software que permita sintetizar registros clínicos coherentes y realistas a partir de tablas reales.

- Evaluar el desempeño de técnicas generativas en cuanto a su capacidad de generar imágenes sintéticas de mamografías.

## 1.5. Estructura de la memoria

En el próximo capítulo, *Estado del arte*, se expone un resumen sobre las soluciones propuestas por investigadores para el problema planteado, las técnicas modernas estudiadas en la actualidad en lo relevante a la generación de datos sintéticos y una breve discusión sobre los aportes del trabajo realizado a la investigación en materia de sintetización de datos en salud.

Luego, en el capítulo de *Desarrollo de la solución* se presenta el trabajo desarrollado en torno a los 2 focos de interés en generación sintética de datos, tablas e imágenes, explicándose en profundidad el software elaborado a través de la expansión de la biblioteca SDV para la sintetización de tablas de datos y los resultados obtenidos para las distintas técnicas probadas en la sintetización de mamografías.

A continuación, en el capítulo *Validación*, se presentan las métricas desarrolladas para la evaluación del trabajo, los resultados obtenidos para las métricas evaluadas y una breve discusión sobre el cumplimiento de la hipótesis en contraste con los resultados presentados.

Finalmente, en *Conclusiones*, se discute el cumplimiento de los objetivos, el alcance del trabajo realizado y los posibles trabajos que pueden continuar este estudio.

## Capítulo 2

# Estado del arte

### 2.1. Soluciones existentes

La aplicación de tecnologías de aprendizaje de máquinas en el área de salud no es un tema innovador. Desde que se inventaron este tipo de algoritmos se han buscado activamente formas de incorporarlos a diversas áreas, en que la medicina y salud no son excepción. Sin embargo, los modelos de aprendizaje de máquinas requieren de un alto volumen de datos para converger a una solución y es precisamente en la salud que el manejo de dichos datos está sujeto a varias complicaciones, entre ellas las barreras legales discutidas anteriormente.

Para abordar esta problemática la comunidad científica diseñó, en primera instancia, algoritmos capaces de anonimizar la información. Estos deben eliminar todo elemento que pueda enlazar los datos con los pacientes, siendo su objetivo hacer imposible el rastreo de las personas involucradas a partir de la información médica. Las dificultades que conllevaba eliminar solo los segmentos de información pertinentes en distintos formatos y lugares dependientes de un dataset limitado, en conjunto con la posible desorganización del dataset resultante, fueron factores que dieron paso a la aplicación de algoritmos de generación sintética.

Conforme estas tecnologías fueron desarrollándose gran parte de la investigación se dirigió hacia *Variational Auto Encoders* (VAEs) y *Generative Adversarial Networks*, ambas técnicas conformadas por 2 redes neuronales que interactúan entre sí con el objetivo de generar información cuyas características estadísticas sean similares a las del dataset de entrenamiento.

En [Abufadda and Mansour, 2021] se discuten numerosos artículos cuyo propósito es estudiar modelos generativos en el ámbito de la salud. Es notable que en su mayoría dichos artículos utilizan variantes de GAN, ya sean las más conocidas o específicamente diseñadas para ese estudio en particular.

En general, los resultados demuestran que los datos sintéticos pueden mejorar significativamente el desempeño en algunos algoritmos de aprendizaje, aunque esto varía dependiendo del tipo de información, área de aplicación, etc.

Pese a que no se tiene conocimiento de artículos que discutan exactamente los mismos elementos que en este documento, sí los hay que tratan materias similares, como [Sedigh et al., 2019]. En este trabajo se trata la generación de información sintética en torno a cáncer de piel para la mejora en el desempeño en algoritmos de detección, similar a uno de los objetivos planteados en este trabajo para cáncer de mama.

## 2.2. Técnicas relevantes

### 2.2.1. GAN

Una *Generative Adversarial Network* [Goodfellow et al., 2014] o *GAN*, se refiere a un sistema caracterizado por la presencia de dos redes neuronales que compiten mutuamente, como se muestra en la figura 2.1. La *red generativa* ( $G(z; \theta_g)$ ) produce muestras artificiales  $x_f$  a partir de un vector aleatorio  $z$  considerado como entrada, mientras que la *red discriminativa* ( $D(x; \theta_d)$ ) intenta distinguir entre muestras reales  $x_r$  provenientes del set de entrenamiento y valores falsos obtenidos desde el generador, emitiendo como salida una predicción binaria sobre si la muestra es real o no.

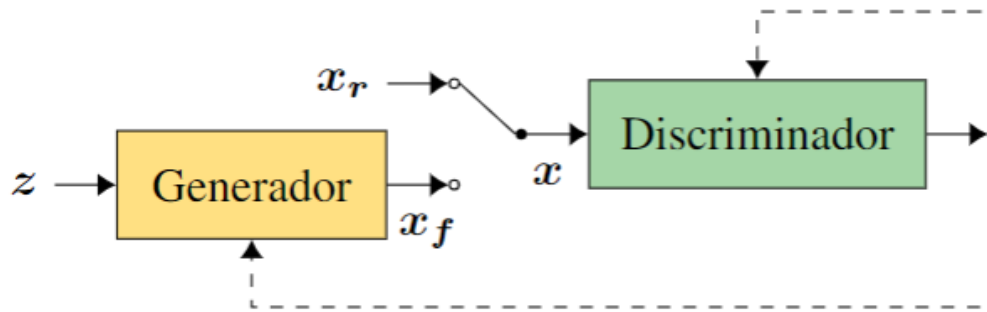


Figura 2.1: Arquitectura de sistema GAN.

Conforme las redes se entrenan  $G(z; \theta_g)$  mejora la calidad de información generada, mientras que  $D(x; \theta_d)$  mejora su capacidad de diferenciar entre información real e información generada.

### 2.2.2. DCGAN [Radford et al., 2015]

DCGAN es una variante de GANs en la cual se usan *deep convolutional networks* tanto en la red discriminatoria como en la generativa.

Los autores del modelo argumentan que las modificaciones que hacen, entre las que se encuentran el uso de normalización por batch y la eliminación de las capas *fully connected*, logran un entrenamiento más estable para la red generativa.

### 2.2.3. WGAN [Arjovsky et al., 2017]

La contribución principal de esta variante es una modificación en el proceso de entrenamiento del discriminador, ahora llamado *crítico*, al actualizar los parámetros de este más veces en relación a los del generador.

Además, la salida del discriminador deja de ser binaria para otorgar un valor entre 0 y 1 que representa la predicción de que tan reales son las imágenes de entrada; Los pesos están acotados a valores pequeños y la función para el entrenamiento de las redes es la diferencia entre la predicción del crítico para las imágenes reales y la predicción para las imágenes falsas.

Los autores afirman que este modelo tiene un entrenamiento más estable y la característica única de que la función *loss* es un criterio para saber la calidad de las imágenes creadas.

### 2.2.4. ProGAN [Karras et al., 2017]

ProGAN, también conocido como PGGAN, adiciona una serie de elementos claves a WGAN con el fin de mejorar la calidad de las imágenes generadas:

1. *crecimiento progresivo*: Consiste en primero entrenar una WGAN para imágenes 4x4 y posteriormente agregar nuevas etapas de re-escalado a ambas redes, cada determinado número de *epochs*, hasta alcanzar la resolución deseada. Esta mecánica sigue la lógica de aprendizaje de los organismos vivos, en que las primeras etapas de aprendizaje consisten en la identificación de las características más importantes, para luego ir añadiendo detalles progresivamente.
2. La sustitución de las capas de *convolución traspuesta* por sobreescalados lineales que no adicionan pesos a la red.
3. El concepto de *ecualización de tasa de aprendizaje*, que permite una mejor competencia entre crítico y generador al escalar los pesos de las capas convolucionales de acuerdo a su número de

coeficientes.

4. Normalización de píxel, para evitar inestabilidades causadas por valores elevados dentro de algún vector de características.

### **2.2.5. The Synthetic Data Vault**

La SDV es un ecosistema de bibliotecas diseñadas para facilitar la generación de datos sintéticos a los usuarios a través de diversos modelos generativos para crear información completamente nueva que conserve las características estadísticas y el mismo formato que el conjunto de datos original. Los modelos son agrupados en torno a 3 formas de trabajo: *Single Table Data*, para tablas individuales; *Relational Data*, para múltiples tablas que estén relacionadas entre sí y *Time Series*, para tablas que registren la evolución de datos a lo largo del tiempo.

En el repositorio de Github de la SDV [Patki et al., 2016], se encuentran todas las especificaciones sobre el funcionamiento de la herramienta, en conjunto con tutoriales, códigos de ejemplo, detalles sobre implementación de restricciones, manipulación de parámetros, etc.

Así mismo, también cuenta con un conjunto de métricas de evaluación referidas como SDMetrics que utiliza para evaluar los resultados obtenidos con un valor entre 0 y 1, siendo 1 la mejor evaluación para un conjunto de datos sintéticos generados, y contando con la posibilidad de seleccionar cuál o cuáles métricas se desean utilizar.

## **2.3. Discusión**

Si bien la aplicación de algoritmos de aprendizaje en materia de salud se ha vuelto un tema recurrente en investigación médica, en este trabajo se plantea evaluar el desempeño de las nuevas tecnologías específicamente en materia de generación de data sintética en formato de tablas de registros clínicos e imágenes de mamografías.

En relación al trabajo realizado con tablas se espera contribuir con un software dedicado a facilitar la generación de información sintética en este formato, así permitiendo que individuos ajenos al campo de la programación sean capaces de producir e utilizar información sintética, a su vez ampliando la investigación en este campo.

Por otro lado, los resultados obtenidos en torno a las imágenes de mamografías esperan comprobar qué tan cerca están las técnicas modernas de producir información lo suficientemente realista para suplir, sino reemplazar, datos reales de estas características para el entrenamiento de algoritmos de aprendizaje, o bien para el apoyo en procedimientos médicos.

## Capítulo 3

# Desarrollo de la solución

### 3.1. Modelo ProGAN para generación de mamografías

#### Testing:

Para decidir en base a qué algoritmo se realizaría el trabajo, hubo un periodo de testing en que se probaron distintas variantes de GAN en torno al dataset *mini-MIAS* de la *Mammographic Image Analysis Society*, disponible en <http://peipa.essex.ac.uk/info/mias.html>.

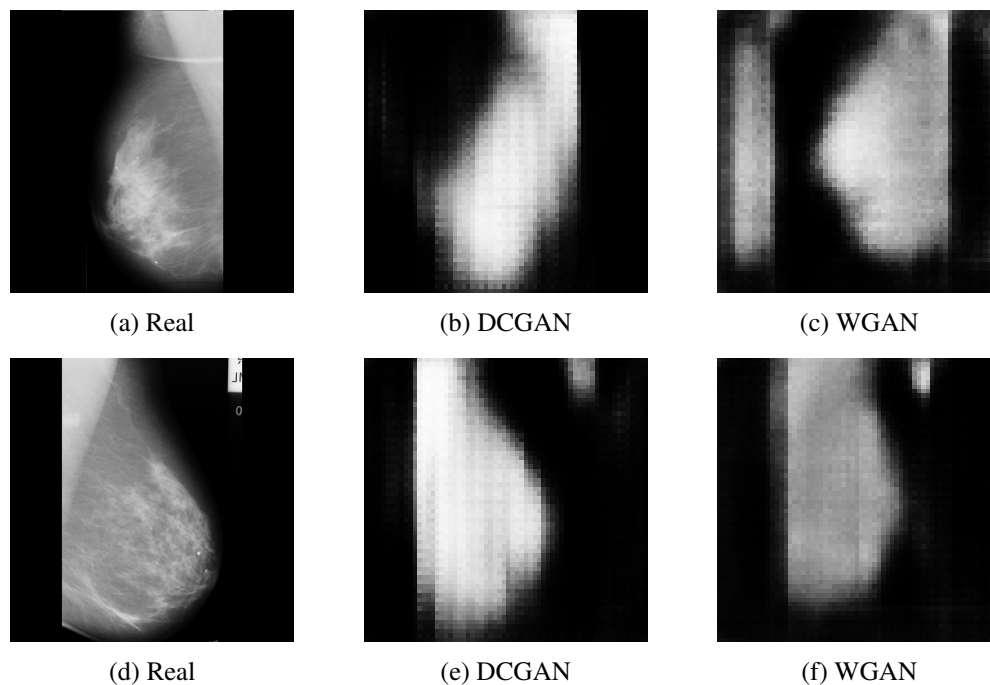


Figura 3.1: Primeros resultados: DCGAN y WGAN

Este dataset está conformado por 322 imágenes de mamografías en formato PGM, correspondien-

tes a 161 pacientes, con una imagen por lado. Las imágenes están normalizadas a una resolución de 1024x1024, pero, con propósito de reducir los recursos computacionales necesarios, estas fueron reescaladas a 64x64.

Durante esta fase se probaron DCGAN y WGAN, teniendo por conclusión que ambos sistemas tienen problemas intentando replicar la textura característica de una mamografía, tal como se observa en la figura 3.1.

Cabe destacar que pese a no obtenerse resultados con la calidad mínima esperada, estos algoritmos sí son capaces de generar imágenes con forma y color similares a las del dataset real.

### **ProGAN:**

Tras esto se decidió implementar un modelo del tipo ProGAN haciendo uso de un dataset con mayor volumen de datos: Breast Cancer Screening Digital Breast Tomosynthesis (BCS-DBT). Este es un dataset de 22.032 mamografías en formato DICOM correspondiente a 5060 pacientes, disponible en <https://wiki.cancerimagingarchive.net/pages/viewpage.action?pageId=64685580> y que cuenta con documentos .csv que contienen información sobre la presencia del cáncer en cada imagen.

Para el entrenamiento de las redes del modelo ProGAN se consideraron 30 epochs para cada resolución de imagen, empezando por resolución de 128x128 píxeles, learning rate de  $1e-3$  y tamaño de batch de 32.

Es importante destacar que, dados los recursos computacionales disponibles, se decidió reescalar las imágenes a un tamaño de 256x256 píxeles en formato .png. Pese a esto, sólo fue posible completar el entrenamiento para la primera resolución de imagen (128x128), para un total de 1000 imágenes, cuyos resultados se pueden observar en la figura 3.2.

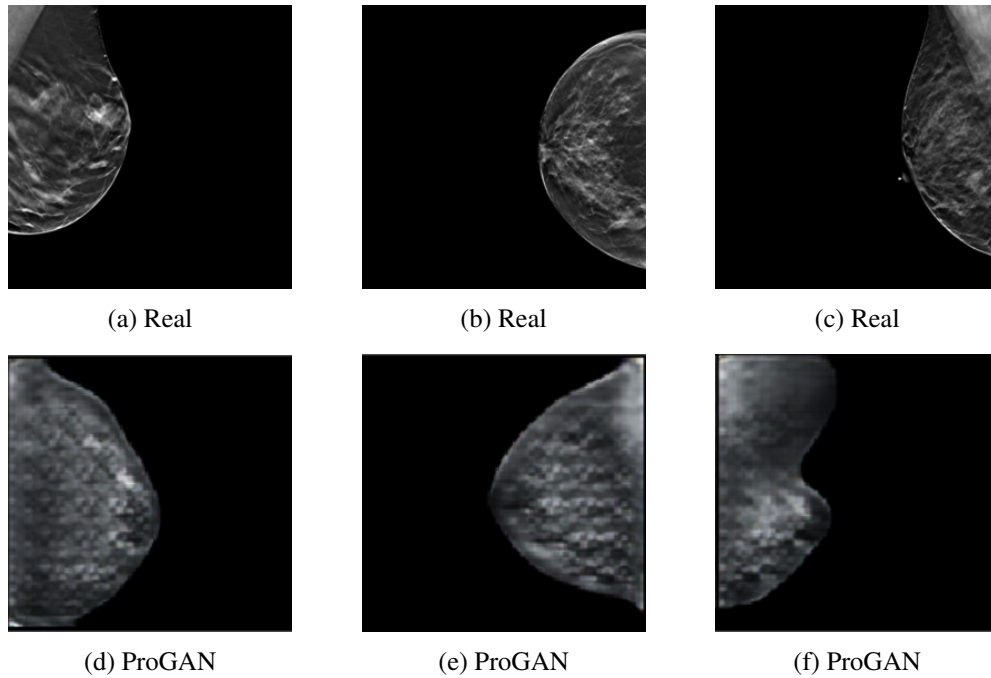


Figura 3.2: Primeros resultados: ProGAN

Al tener disponibilidad de mejores recursos computacionales se procedió a realizar el entrenamiento nuevamente con un total de 17.000 imágenes pertenecientes al dataset BCS-DBT. Sin embargo, sólo fue posible llegar a un tamaño de 512x512 píxeles debido a limitaciones de hardware. Con el fin de mejorar la calidad de las imágenes este modelo fue re-entrenado con el dataset una segunda vez, observándose los resultados presentados en la figura 3.3.

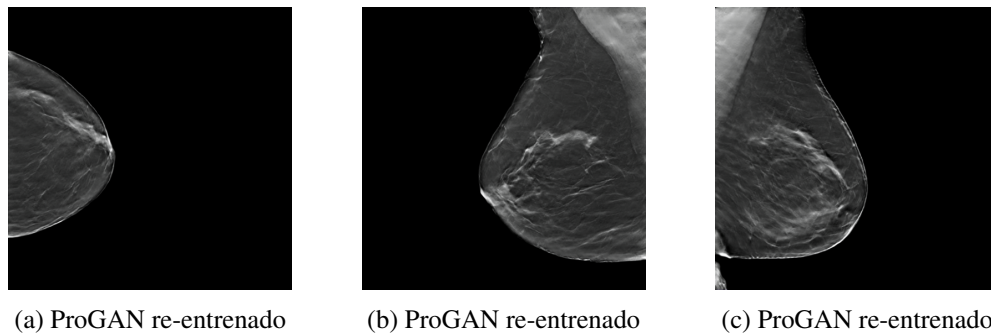


Figura 3.3: Resultados finales: ProGAN

Se evidencia que al aumentar el volumen de datos utilizados para el entrenamiento la calidad general de las imágenes aumenta, puesto que se observa una mejoría en torno a los detalles, formas y distribución de colores en la imagen. Así mismo, es notable como los límites que distinguen los músculos del tejido mamario son mucho más claros que en los resultados previos.

## 3.2. Generador de tablas sintéticas

Para comprobar el alcance que los modelos generativos podrían tener en materia de registros clínicos se plantea la elaboración de un software capaz de inferir las características estadísticas de un conjunto de datos reales para luego, a partir de estos, generar información sintética que las conserve y las exprese de forma coherente con la realidad.

Como se menciona anteriormente, este programa está basado en la biblioteca SDV, en su versión 0.17.2, la cual facilita herramientas dedicadas a la generación de data sintética en formato de tablas y diseñado en torno al lenguaje de programación *Python*.

Con el propósito de familiarizarse con la biblioteca hubo un periodo de testing en torno a un dataset público, obtenido en la web, durante el cual se probaron determinados elementos propios de esta, tales como la modalidad de trabajo, implementación de restricciones, tipos de modelos generativos disponibles, etc. Cabe mencionar que al momento de realizarse dichas pruebas la biblioteca se encontraba en su versión 0.14.2.

| Motor     | Tiempo de procesamiento [seg] | Puntaje (Métricas del SDV) |
|-----------|-------------------------------|----------------------------|
| COPULA    | 0.09939350000000058           | 0.7684313398168513         |
| GAN       | 0.08650009999999941           | 0.767246108218466          |
| GANCOPULA | 0.31047250000000004           | 0.7241461928255801         |
| TVAE      | 0.11171440000000032           | 0.7011022390012613         |

Tabla 3.1: Resultados de rendimiento para modelos generativos correspondientes a modalidad *single table data*.

En la tabla 3.1 se pueden ver los tiempos de procesamiento, en segundos, correspondientes a cada uno de los 4 modelos generativos de la SDV disponibles para el trabajo con tablas únicas. Observando los resultados es posible decir que, en general, el tiempo total para entrenamiento y generación de datos sintéticos es pequeño. Además, el puntaje otorgado por la propia métrica de evaluación de la biblioteca, el cual entrega valores entre 0 y 1 donde 1 es la mejor evaluación, muestra que los modelos tienen relativamente el mismo rendimiento.

Posteriormente, el trabajo se enfocó en evaluar qué elementos de la SDV serían fundamentales para el trabajo con registros clínicos, para luego combinarlos en un software que los implementa de forma semi asistida para su uso en tablas con distintos formatos.

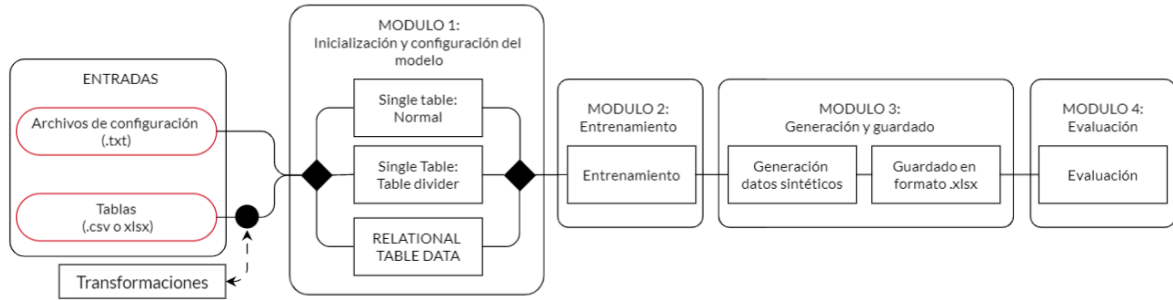


Figura 3.4: Esquema de la estructura del software generador de tablas.

En la figura 3.4 se gráfica un esquema que detalla el funcionamiento del software generador de tablas y las interacciones entre las entradas y los 4 módulos que lo componen, los cuales se describirán a detalle en las siguientes secciones.

### 3.2.1. Entradas

#### Tablas

El software generador de tablas utiliza la biblioteca *pandas* para leer y trabajar un único archivo *.csv* o un libro excel *.xlsx* con una o múltiples hojas. Estos serán utilizados por el programa como la información original desde donde extraer las características estadísticas y representativas que servirán de base para las tablas sintéticas generadas.

#### Archivos de configuración

Los archivos de configuración son 4 archivos de texto (*.txt*) donde que cada línea, en formato JSON para lenguaje *Python*, corresponde a las especificaciones que el modelo debe considerar para cada tabla.

Para el caso en que no se configuren especificaciones para alguna tabla, la línea correspondiente es reemplazada por un diccionario vacío (“{ }”).

1. **table\_names.txt:** Siempre está conformado por una única línea que denota una lista [*list*] contenedora de los nombres de las tablas que ingresan al programa.

Este archivo es responsable de informar sobre las tablas existentes y el orden en que deben ser incorporadas al modelo.

2. **constraints.txt:** Cada línea corresponde a un diccionario [*dict*] de 2 llaves: *constraints* y *specifications*.

Ambas llaves tienen por valor una lista, en donde cada elemento de la primera es un *string* asociado a un tipo de restricción (r) y cada elemento en la misma posición de la segunda corresponde a los argumentos de entrada de esta (r\_input).

Cada conjunto de entradas se agrupa como elementos en una lista y su contenido depende de la restricción a la que estén asociados, lo que se detalla en la tabla 3.2

| Restricción<br>(r) | Argumentos de entrada<br>(r_input)                                  | Tipo                                      |
|--------------------|---------------------------------------------------------------------|-------------------------------------------|
| Unique             | column_names                                                        | list[str]                                 |
| FixedC             | column_names                                                        | list[str]                                 |
| Positive           | column_name<br>strict                                               | str<br>bool                               |
| Negative           | column_name<br>strict                                               | str<br>bool                               |
| Inequality         | low_column_name<br>high_column_name<br>strict                       | str<br>str<br>bool                        |
| ScalarInequality   | column_name<br>relation<br>value                                    | str<br>str<br>float o datetime            |
| Range              | low_column_name<br>middle_column_name<br>high_column_name<br>strict | str<br>str<br>str<br>bool                 |
| ScalarRange        | column_name<br>low_value<br>high_value<br>strict                    | str<br>int o float<br>int o float<br>bool |
| OneHot             | column_names                                                        | list[str]                                 |

Tabla 3.2: Tipos de restricción y sus respectivos argumentos de entrada.

Así, cada línea de *constraints.txt* tiene el siguiente formato:

$\{ \text{"constraints"} : [ \text{"r1"}, \text{"r2"}, \dots, \text{"rn"} ], \text{"specifications"} : [ [r1\_input], [r2\_input], \dots, [rn\_input] ] \}$

Este archivo .txt permite al usuario forzar características en la información sintética en caso de que el modelo no sea capaz de inferirlas automáticamente o se desee realizar ese cambio a la información original.

Los detalles sobre que efecto tiene cada restricción y que valores deben ingresarse para cada argumento de entrada se exponen a continuación:

- **Unique:** Fuerza a que cada valor de una columna o grupo de columnas sean únicos.
  - a) *column\_names*: Lista con los nombres de las columnas sobre las que se desea aplicar la restricción.
- **FixedC:** Se asegura que las posibles combinaciones de un grupo de columnas permanezca igual al de la tabla original.
  - a) *column\_names*: Lista con los nombres de las columnas sobre las que se desea aplicar la restricción.
- **Positive:** Se asegura que los valores de una columna sean positivos.
  - a) *column\_name*: nombre de las columna sobre la que se desea aplicar la restricción.
  - b) *strict*: Valor booleano [*bool*] expresado como entero, que dicta si la restricción es estricta, donde:
$$1 \longrightarrow > 0$$
$$0 \longrightarrow \geq 0$$
- **Negative:** Se asegura que los valores de una columna sean negativos.
  - a) *col\_name*: nombre de las columna sobre la que se desea aplicar la restricción.
  - b) *strict*: valor booleano [*bool*] expresado como entero, que dicta si la restricción es estricta, donde:
$$1 \longrightarrow < 0$$
$$0 \longrightarrow \leq 0$$
- **Inequality:** Asegura que una columna sea siempre mayor que otra.
  - a) *low\_column\_name*: Nombre de la columna con valores inferiores.

- b) *high\_column\_name*: Nombre de la columna con valores superiores.
  - c) *strict*: valor booleano [*bool*], expresado como entero, que dicta si la restricción es estricta, donde:
    - $1 \longrightarrow > low\_col$
    - $0 \longrightarrow \geq low\_col$ .
- **ScalarInequality**: Asegura que una columna cumpla con una restricción relacional con respecto a un número.
  - a) *column\_name*: Nombre de la columna objetivo.
  - b) *relation*: Describe la relación entre la columna y el valor, teniendo por posibles relaciones: “<”, “<=”, “>”, “>=”.
  - c) *value*: valor numérico o fecha en torno a la cual se satisface la desigualdad.
- **Range**: Asegura que la columna objetivo siempre esté entre los valores de otras 2 columnas.
  - a) *low\_column\_name*: Nombre de la columna que impone el límite inferior.
  - b) *middle\_column\_name*: Nombre de la columna objetivo.
  - c) *high\_column\_name*: Nombre de la columna que impone el límite superior.
  - d) *strict*: valor booleano [*bool*], expresado como entero, que dicta si la restricción es estricta, donde:
    - $1 \longrightarrow low < x < high$
    - $0 \longrightarrow low \leq x \leq high$
- **ScalarRange**: Asegura que la columna objetivo siempre esté entre 2 valores numéricos o de tipo *datetime*.
  - a) *column\_name*: Nombre de la columna objetivo.
  - b) *low\_value*: límite inferior.
  - c) *high\_value*: límite superior.
  - d) *strict*: valor booleano [*bool*], expresado como entero, que dicta si la restricción es estricta, donde:
    - $1 \longrightarrow low < x < high$
    - $0 \longrightarrow low \leq x \leq high$
- **OneHot**: Asegura que las columnas objetivo cumplan con codificación OneHot.

a) **column\_names**: Lista con los nombres de las columnas sobre las que aplicar la restricción.

3. **date\_time.txt**: Cada línea es un diccionario [dict] en que las llaves son el nombre de una columna y el valor el formato de fecha y hora en que está dicha columna.

Este archivo cumple el propósito de informar al programa que columnas de cada tabla contienen fechas.

4. **relationships.txt**: Cada línea es una lista [list] con 2 valores, donde el primero es un nombre de tabla y el segundo un nombre de columna.

Este archivo cumple el propósito de informar al programa de las relaciones existentes entre tablas, siendo los valores anteriormente mencionados la tabla a la que se asocia la relación y el nombre de la columna involucrada en dicha relación respectivamente.

### 3.2.2. Módulos

El software puede ser dividido en 4 etapas, cada una representada por uno de los 4 módulos dispuestos a continuación.

#### Módulo 1: Creación y configuración del modelo

El módulo 1 es la base del funcionamiento del software. En este se toman los nombres contenidos en *table\_names.txt* para determinar el número de tablas que ingresarán al programa y el orden en que deben ser incorporadas.

##### **Single table data: Normal**

Este submódulo actúa en caso de ser una única tabla, donde se procede a inicializar una instancia y configurarla con las restricciones contenidas en el archivo *constraints.txt*.

##### **Single table data: Table divider**

Funciona de la misma manera que el submódulo *Single table data: Normal*, con la diferencia que se le pide al software fragmentar la tabla acorde a las categorías contenidas dentro de una columna específica, donde cada fragmento corresponde a todas las filas pertenecientes a esa categoría. Esto tiene el propósito de facilitar la inferencia de características en tablas con mucha variabilidad entre filas.

Un ejemplo de aplicación es en tablas de resultados de pruebas de laboratorio, en donde el tipo de prueba incide directamente en el rango de valores de los resultados. Al descomponer la tabla por tipo de prueba, el rango de valores que el programa lee a la vez son más acotados.

##### **Relational table data**

Este submódulo se utiliza cuando los datos reales están contenidos en más de una tabla.

Tras crear una instancia, se utiliza el archivo *table\_names.txt* para establecer un orden en la lectura de información. Luego, una tabla a la vez, se incorporan a una instancia de clase *MetaData*: la tabla, sus restricciones pertinentes, contenidas en *constraints.txt* y la relación que pueda tener con otra de las tablas, contenida en *relationships.txt*.

Finalmente, se le entregan al modelo un diccionario, cuyas llaves son los nombres de las tablas y su valor las tablas como un *PandasDataframe*, y el *MetaData*.

#### Módulo 2: Entrenamiento

Este módulo recibe el modelo configurado y lo entrena con la información real.

### Módulo 3: Generación y guardado de datos

Este módulo recibe el modelo entrenado y genera la información sintética pertinente que luego guarda en un libro excel (.xlsx).

### Módulo 4: Evaluación

Este módulo recibe los datos reales, los datos sintéticos y la metadata correspondiente para generar una serie de reportes y figuras obtenidas a través de SDMetrics, la biblioteca de evaluación ya incorporada en la SDV. Su activación es opcional y es escogida por el usuario a través de la variable *EVALUATE* (tabla 3.4).

Los reportes consisten en un conjunto de 4 archivos .xlsx en que cada hoja corresponde a la evaluación de los elementos de una tabla:

- **Column\_Shapes.xlsx**: Evalúa la similitud en cuanto a la distribución de valores para cada columna.
- **Column\_Pair\_Trends.xlsx**: Evalúa la similitud de la correlación entre pares de columnas de una misma tabla.
- **Parent\_Child\_Relationships.xlsx**: Evalúa la similitud en cuanto al número de filas hijas para cada tabla padre.

Este reporte sólo se genera para el caso en que se ingrese al sub-módulo *relational table data*.

- **Evaluation\_method.xlsx**: Se genera a partir de un segundo método de evaluación, en que se utilizan las métricas *Inverted Kolmogorov-Smirnov D statistic* y *Chi-squared*.

En todos los reportes, el resultado de la evaluación se representa como un valor entre 0 y 1, donde 1 es el mejor puntaje posible.

En adición a lo anterior, se generan imágenes que representan los puntajes obtenidos a través de matrices de correlación y gráficos, las cuales se dividen en 2 grupos:

- **Column\_scores**: Representación gráfica de los reportes **Column\_Shapes.xlsx**, **Column\_Pair\_Trends.xlsx** y **Parent\_Child\_Relationships.xlsx**.
- **Column\_score\_details**: Comparación gráfica de la distribución de una columna o par de columnas específicas entre los datos reales y los sintéticos.

En este caso, las columnas o par de columnas a evaluar son especificadas por el usuario a través de las variables *E\_COLS* y *E\_PAIR\_COLS* (tabla 3.4) respectivamente y la representación gráfica se ajusta dependiendo del tipo de dato asociado a las columnas evaluadas (numérico/datetime o categórico).

Si el módulo Table Divider se encuentra activo, se dispone de una variable que controla los fragmentos que se desean evaluar, **CATEGORIES\_TO\_EVALUATE** (tabla 3.4), la cual tiene 3 posibles valores:

- **“All”**: Se procede a evaluar todas las categorías.
- **Lista vacía ([])**: Se evalúa la tabla en su totalidad, sin tomar en consideración las distintas categorías.
- **Lista con los nombres de las categorías**: Se procede a evaluar sólo las categorías contenidas en la lista, cada una por separado.

## Transformaciones

Al realizar pruebas con la biblioteca SDV se notó que para la generación sintética es necesario que las tablas no contengan campos vacíos (*nan*), así como para la evaluación, una misma columna no puede contener fechas de distintos formatos ni combinadas con otro tipo de dato.

Para esto se consideraron 2 transformaciones:

- Convertir todos los valores *nan* a un espacio “ ”.
- Normalizar los valores de fecha de las columnas pertinentes, indicadas en el archivo *date\_time.txt*.  
Para el caso de las columnas de fecha que tengan valores en espacio (“ ”) tras la primera transformación, se reemplaza ese valor por un string vacío “”.

### 3.2.3. Código

El programa generador de tablas se compone a partir de 3 archivos: UTILITIES.py, el cual almacena todas las funciones desarrolladas para el software, detalladas en la tabla 3.3; CONFIG.py, el cual almacena las variables que controlan el módulo de evaluación y el sub-módulo table divider, detalladas en la tabla 3.4; y main.py, donde se ejecuta el programa apoyándose en los contenidos de los 2 archivos .py mencionados anteriormente.

| Función                | Entrada(s)                                                                             | Funcionalidad                                                                                                                                                                                                         |
|------------------------|----------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| constraints_reader()   | [1] Diccionario correspondiente a línea de constraints.txt                             | Retorna una lista con objetos de tipo sdv.constraints o None si [1] esta vacío.                                                                                                                                       |
| relationships_reader() | [1] Lista correspondiente a línea de relationships.txt                                 | Retorna par 'nombre tabla padre' y 'nombre columna de enlace' o par [None, None] si [1] está vacío.                                                                                                                   |
| convert_date()         | [1] Tabla como dataframe<br>[2] Diccionario correspondiente a línea de date_time.txt   | Aplica conversion() sobre los valores de las columnas de [1] que están señaladas en [2].                                                                                                                              |
| conversion()           | [1] Valor<br>[2] formato de fecha como string                                          | Normaliza [1] a tipo datetime y, en caso de ser “ ”, convierte a un bloque vacío “”.                                                                                                                                  |
| table_divider()        | [1] Tabla como dataframe<br>[2] Nombre de columna<br>[3] Mínimo número de repeticiones | Retorna un diccionario de dataframes donde cada uno corresponde a un fragmento de [1].<br>Cada fragmento contiene todas las filas con uno de los posibles valores de [2], siempre que este se repita [3] o más veces. |
| tables_to_xlsx()       | [1] Diccionario de dataframes<br>[2] Nombre del archivo                                | Guarda un archivo .xlsx con nombre [2], cuyas hojas son los dataframes de [1] con el nombre de su respectiva llave.                                                                                                   |

Tabla 3.3: Funciones desarrolladas para su implementación en el software.

En adición a las funciones previamente presentadas, se desarrolló la función **evaluate\_data()**, la cual es profundizada en el algoritmo 4 y tiene por función incorporar el módulo de evaluación al software.

| Variable               | Contenido                                                                                                                                                                                                                                                                                                               |
|------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| TABLE_DIVIDER          | Valor booleano que determina si el sub-módulo table divider está activado.                                                                                                                                                                                                                                              |
| TD_COL                 | Columna a utilizar para el sub-módulo table divider.                                                                                                                                                                                                                                                                    |
| MIN_REPS               | Mínimo de repeticiones que debe tener una categoría para ser considerada en la sintetización.                                                                                                                                                                                                                           |
| CATEGORIES_TO_EVALUATE | <p>Lista con los nombres de las categorías que se consideraran en la evaluación.</p> <p>Si la lista está vacía la evaluación considera la tabla sintetizada en su totalidad.</p> <p>Si en vez de ser una lista es un <i>string</i> de valor “All”, se realiza la evaluación para todas las categorías consideradas.</p> |
| primary_key            | Nombre de la columna cuyos valores serán usados como identificador. Ej. ID del paciente.                                                                                                                                                                                                                                |
| EVALUATE               | Valor booleano que determina si el módulo de evaluación está activado.                                                                                                                                                                                                                                                  |
| E_COLS                 | Lista con los nombres de las columnas de las que se generarán imágenes.                                                                                                                                                                                                                                                 |
| E_PAIR_COLS            | Lista con los pares de nombres de columnas, emparejados dentro de una lista, de los que se generarán imágenes.                                                                                                                                                                                                          |

Tabla 3.4: Variables de configuración contenidas en CONFIG.py

A continuación se presentan los pseudocódigos sobre los cuales se basa el esquema de la figura 3.4.

---

**Algorithm 1** Software generador de tablas (main.py)

---

**LECTURA DE ENTRADAS**

*table\_names* ← Línea de *table\_names.txt*

*cons* ← *constraints.txt*

*rel* ← *relationships.txt*

*time\_spc* ← *date\_time.txt*

*real\_dict* = {}

▷ Diccionario vacío

**SINGLE TABLE DATA**

**if** *len(table\_names)* == 1 **then**

*table* ← *table\_names*[0]

*table\_df* ← **pandas.read\_csv**(*table*)

▷ Tabla real como dataframe

*Num\_rows* ← Número de filas en *table\_df*

*Constraints\_list* ← **constraint\_reader**(Línea de *cons*)

**Transformaciones**

*table\_df.fillnan*(" ")

▷ Conversión de *nan* a " "

*time\_line* ← Línea de *time\_spc*

*real\_dict*[*table*] ← **convert\_date**(*table\_df*, *time\_line*)

▷ Tabla 3.3

**if** *TABLE\_DIVIDER* == *False* **then**

        SINGLE TABLE DATA: Normal

▷ Algorithm 2

**else**

        SINGLE TABLE DATA: Table divider

▷ Algorithm 3

**end if**

**RELATIONAL TABLE DATA**

**else**

**Inicializa instancia de metadata**

**for** *index*, *table* **in** *table\_names* **do**

*table\_df* ← **pandas.read\_excel**(*tables.xlsx*, *table*)

▷ Tabla real *index* como dataframe

**Transformaciones**

*table\_df.fillnan*(" ")

▷ Conversión de *nan* a " "

*time\_line* ← Línea de *time\_spc*

*real\_dict*[*table*] ← **convert\_date**(*table\_df*, *time\_line*)

▷ Tabla 3.3

*Constraints\_list* ← **constraint\_reader**(Línea *index* de *cons*)

▷ Tabla 3.3

*parent*, *foreign\_key* ← **relationship\_reader**(Línea *index* de *rel*)

▷ Tabla 3.3

*table*, *table\_df*, *primary\_key*, *Constraint\_list*, *parent*, *foreign\_key* → *metadata*

**end for**

**Inicializa modelo** *sdv.HMA1* **con** *metadata*

*sdv.fit*(*real\_dic*)

▷ Entrenamiento

*synthetic\_dic* ← *sdv.samples*()

▷ Generación

**tables\_to\_excel**(*synthetic\_dic*, "synthetic.xlsx")

▷ Tabla 3.3

**if** *EVALUATE* **then**

*evaluate\_data*(...)

▷ Algoritmo 4

**end if**

**end if**

---

---

**Algorithm 2** SINGLE TABLE DATA: normal

---

**Inicializa modelo** *sdv.GaussianCopula* **con** *Constraint\_list*

*sdv.fit(table\_df)*

▷ Entrenamiento

*synthetic\_dic* = {*table* : *sdv.samples(Num\_rows)*}

▷ Generación

**tables\_to\_xlsx**(*synthetic\_dic*, “synthetic.xlsx”)

▷ **Tabla 3.3**

**if** EVALUATE **then**

*metadata* = *Metadata()*

▷ Inicializa instancia de *metadata*

*table*, *table\_df*, *primary\_key*, *Constraint\_list*  $\longrightarrow$  *metadata*

**evaluate\_data**(...)

▷ Algoritmo 4

**end if**

---

---

**Algorithm 3** SINGLE TABLE DATA: Table divider

---

*final\_table*  $\leftarrow$  dataframe vacío

*tables*  $\leftarrow$  **table\_divider**(*table\_df*, *TD\_COL*, *MIN\_REPS*)

▷ **Tabla 3.3**

**for** *key*, *value* **in** *tables* **do**

**Inicializa modelo** *sdv.GaussianCopula* **con** *Constraint\_list*

*sdv.fit(value)*

▷ Entrenamiento

*samples*  $\leftarrow$  *sdv.samples*(*len(value)*)

▷ Generación

*final\_table*  $\leftarrow$  *pandas.concat*(*final\_table*, *samples*)

▷ Concatena tablas sintéticas

**end for**

*synthetic\_dic* = {*table* : *final\_table*}

**tables\_to\_xlsx**(*synthetic\_dic*, “synthetic.xlsx”)

▷ **Tabla 3.3**

**if** EVALUATE **then**

**if** (CATEGORIES\_TO\_EVALUATE == []) **then**

*metadata* = *Metadata*()

▷ Inicializa instancia de *metadata*

*table*, *table\_df*, *primary\_key*, *Constraint\_list*  $\longrightarrow$  *metadata*

**evaluate\_data**(...)

▷ Algoritmo 4

**else**

**if** (CATEGORIES\_TO\_EVALUATE == “All”) **then**

*categories\_to\_evaluate* = Lista con **todas** las categorías en *TD\_COL*

**else**

*categories\_to\_evaluate* = CATEGORIES\_TO\_EVALUATE

**end if**

**for** *category* **in** *categories\_to\_evaluate* **do**

*category\_r\_dict[table]* = Fragmento de la tabla real correspondiente a *category*

*category\_s\_dict[table]* = Fragmento de la tabla sintética correspondiente a *category*

*category\_metadata* = *Metadata*()

▷ Inicializa instancia de *metadata*

*table*, *category\_r\_dic[table]*, *primary\_key*, *Constraint\_list*  $\longrightarrow$  *category\_metadata*

**evaluate\_data**(... , *TD* = True, *category\_name* = *category*)

▷ Algoritmo 4

**end for**

**end if**

**end if**

---

---

**Algorithm 4** Modulo evaluación:

evaluate\_data(real\_dict, synthetic\_dict, metadata\_dict, E\_COLS, E\_PAIR\_COLS, table\_names, TD = False, category\_name = None)

---

**\*Si TD=True, en los nombres de los directorios generados se incluye un directorio extra con nombre “category\_name”**

*report* = instancia de **QualityReport()**

*report.generate(real\_dict, synthetic\_dict, metadata\_dict)* ▷ Genera reporte general

**for** *table* **in** *table\_names* **do**

**Consigue detalles:**

- *Column Shapes*
- *Column Pair Trends*
- *Parent Child Relationships*

**end for**

**Genera directorios**

**Guarda detalles .xlsx y *Column\_scores***

**for** *table* **in** *metadata* **do**

**for** *column* **in** *E\_COLS* **do**

Genera gráfico para la columna *column*

Guarda gráfico como .png en *Column\_score\_details*

**end for**

**for** *column\_pair* **in** *E\_PAIR\_COLS* **do**

Genera gráfico para par de columnas *column\_pair*

Guarda gráfico como .png en *Column\_score\_details*

**end for**

**end for**

---

## Capítulo 4

# Validación

Para la validación del trabajo realizado es necesario elaborar métricas de evaluación que se ajusten al problema abordado en cada uno de sus enfoques, es decir, tablas e imágenes.

En primer lugar, para evaluar la calidad de las tablas sintetizadas se requiere tomar en consideración los siguientes factores:

- La información generada no debe ser una copia de los datos utilizados para el entrenamiento.
- La distribución de valores correspondiente a cada columna del dataset sintético debe ser similar a la distribución del dataset original.
- El dataset sintético debe mostrar características coherentes con la realidad.

Para evaluar los elementos previamente listados se decide hacer uso del módulo de evaluación descrito en la sección 3.2.2. Con el fin de evaluar los sub-módulos *single table data: normal* y *single table data: table divider* se hace uso de una tabla correspondiente a pruebas de laboratorio, mientras que para la evaluación del sub-módulo *relational table data* se hace uso de un grupo de 5 tablas correspondientes a una serie de registros clínicos relacionados a través del número identificador del paciente.

En segundo lugar, para obtener resultados cuantitativos que nos permitan estimar la calidad de las imágenes sintetizadas de forma realista, se considera que el método más adecuado dado el contexto del problema es una encuesta a profesionales, tanto especializados como generales, que permita evaluar la similitud del dataset sintético con las imágenes originales.

Para ello, la encuesta, realizada a través de la plataforma **survio**, se divide en 3 secciones:

1. **Sección 1** (1 pregunta): Se pregunta sobre el grado de especialización del encuestado en torno a la lectura de mamografías, admitiendo como respuesta una de las siguientes 3 opciones:

- No cuento con formación.
- Cuento con formación, mas no especializada.
- Cuento con formación especializada.

2. **Sección 2** (5 preguntas): En torno a una imagen sintética se pide evaluar, en la escala del 1 al 7, 3 características de la mamografía:

- La forma de la mama
- La textura de los tejidos internos de la mama
- La disposición de los tejidos internos de la mama

Luego se pide evaluar en la escala del 0 al 10 la calidad general de la imagen.

Una de las preguntas de esta sección utiliza una mamografía real, y los resultados obtenidos para esta son utilizados a modo de control.

3. **Sección 3** (10 preguntas): Se exponen dos mamografías, una real y una sintética, sin especificar cuál es auténtica. Primero se le pide al encuestado seleccionar la que este crea es real y luego seleccionar, dentro del mismo grupo de características de la sección anterior, cuál o cuáles de estas determinaron su respuesta.

En este ultimo segmento el encuestado puede agregar una característica de la mamografía en caso de ser necesario.

## 4.1. Resultados tablas sintéticas: Módulo de evaluación

### 4.1.1. *Single table data*

La tabla utilizada para evaluar el funcionamiento del programa generador de tablas en su modalidad *single table data* es *laboratorio.csv*, cuya estructura se detalla en la tabla 4.1.

| Nombre de la columna   | Tipo          |
|------------------------|---------------|
| PID                    | Identificador |
| DF_FechaCreacion       | Datetime      |
| DF_HoraCreacion        | Datetime      |
| DFED_CodigoTest        | Categorico    |
| CE_Abreviado           | Categorico    |
| CET_Abreviado          | Categorico    |
| DFED_Resultado         | Categorico    |
| DFED_ResultadoNumerico | Numérico      |
| CP_Descripcion         | Categorico    |
| DFE_TipoMuestra        | Categorico    |

Tabla 4.1: Estructura de la tabla *laboratorio.csv*

Las columnas de interés para la evaluación en este caso son *DFED.CodigoTest*, la cual contiene el código correspondiente a la prueba realizada al paciente; *DFED.Resultado*, la cual contiene el resultado directo de la prueba que puede ser numérico o categorico y *DFED.ResultadoNumerico*, la cual contiene los resultados de la prueba como valor numérico.

Cabe mencionar que la columna *DFED.CodigoTest* está estrictamente relacionada a las columnas *CE\_Abreviado*, *CET\_Abreviado* y *CP\_Descripcion*, dado que cada una esta asociada al código de la prueba.

#### *Single table data: Normal*

Al ejecutar el programa este ingresa automáticamente al sub-módulo *Single table data: Normal*, tras lo cual se generan los datos sintéticos y luego los reportes y figuras del módulo de evaluación discutidos en la sección 3.2.2.

En la tabla 4.2 se muestra el reporte **Column Shapes**, cuya representación gráfica es observable

en la figura 4.1.

| Column                 | Metric       | Quality Score |
|------------------------|--------------|---------------|
| DF_FechaCreacion       | KSComplement | 0,613183      |
| DF_HoraCreacion        | KSComplement | 0,8346        |
| DFED_CodigoTest        | TVComplement | 0,928609      |
| CE_Abreviado           | TVComplement | 0,970313      |
| CET_Abreviado          | TVComplement | 0,941863      |
| DFED_Resultado         | TVComplement | 0,840431      |
| DFED_ResultadoNumerico | TVComplement | 0,865701      |
| CP_Descripcion         | TVComplement | 0,920304      |
| DFE_TipoMuestra        | TVComplement | 0,972963      |

Tabla 4.2: Reporte Column Shapes para la tabla *laboratorio.csv*.

Data Quality: Column Shapes (Average Score=0.88)

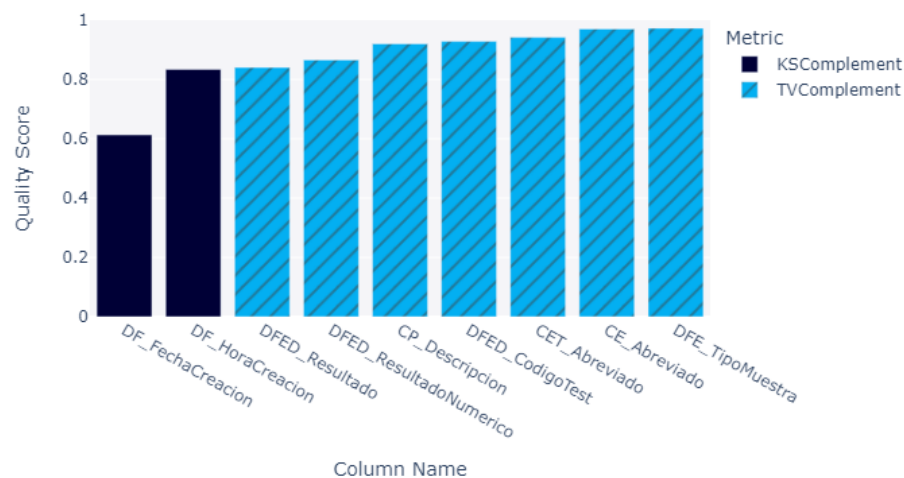


Figura 4.1: Representación gráfica del reporte Column Shapes para *laboratorio.csv*.

Es notable que la gran mayoría de las columnas obtiene una puntuación mayor a 0.8. Sin embargo, al ver el gráfico que muestra los detalles para la columna *DFED\_Resultado*, presentado en las figura 4.2, se observa que se manejan una gran cantidad de datos interpretados de forma categórica, lo que implica que se utilizan los mismos datos presentes en la tabla original, sin generarse nueva

información. Además, no existe una relación estricta entre los resultados y las pruebas a las que están asociados en los datos reales, visibles en la figura 4.3.

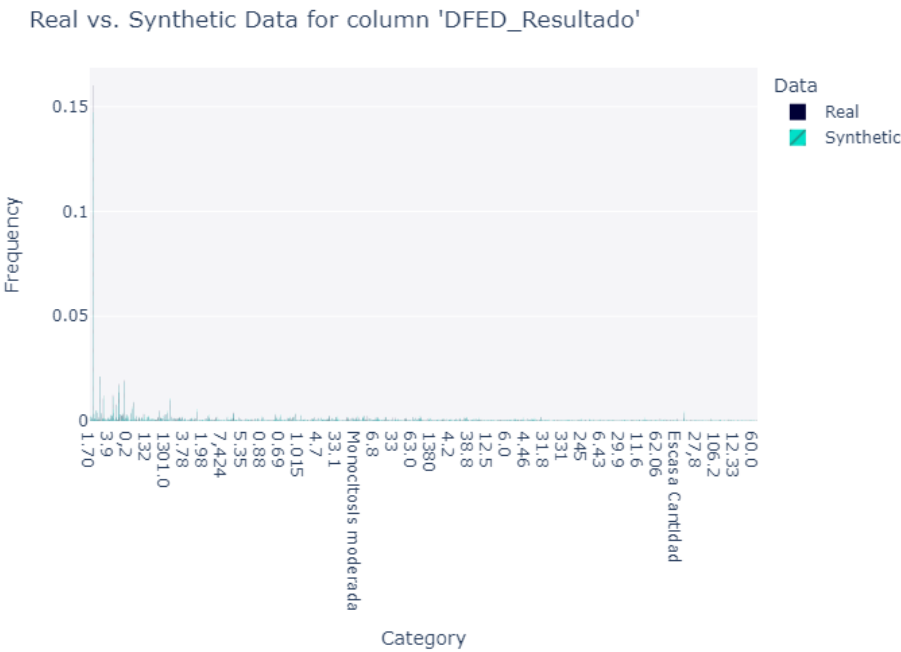


Figura 4.2: Detalles de la distribución de la columna DFED.Resultado.

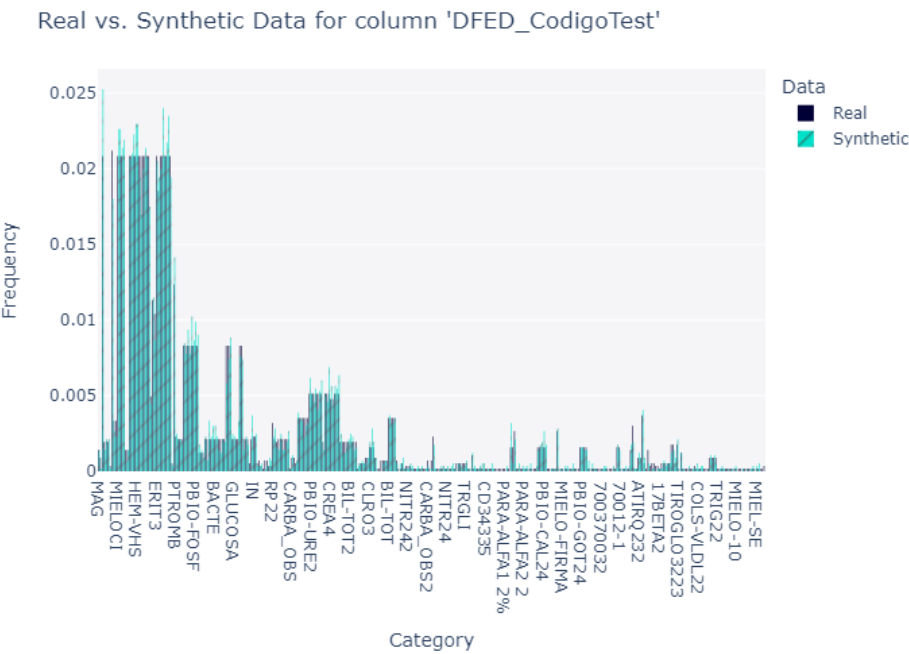


Figura 4.3: Detalles de la distribución de la columna DFED.CodigoTest.

### ***Single table data: Table Divider***

Por otro lado, al activar el módulo table divider y setear *SAMPLES\_TO\_EVALUATE* para las 3 categorías de *DFED\_CodigoTest* con mayor número de filas, es decir *TMVA* (120 valores), *HEM-FROT-HEMI* (118 valores) y *HEM-BAC* (118 valores), se generan los resultados presentados en las siguientes secciones.

Al comparar los gráficos correspondientes a la distribución de la columna *DFED\_CodigoTest* (figuras 4.3, 4.4, 4.7 y 4.10) es evidente que el sub-módulo *table divider* funciona correctamente, dado que fragmenta exitosamente la tabla por cada categoría en la columna escogida.

Acorde a esto, un elemento a evaluar es si la fragmentación efectivamente ayuda al modelo en la inferencia de características. Para ello, basta con observar el cambio en los reportes *column shapes* (tablas 4.2, 4.3, 4.4 y 4.5), los cuales muestran una mejor evaluación general cuando se modela una categoría a la vez.

#### **4.1.2. TMVA**

| Column                 | Metric       | Quality Score |
|------------------------|--------------|---------------|
| DF_FechaCreacion       | KSComplement | 0,566667      |
| DF_HoraCreacion        | KSComplement | 0,791667      |
| DFED_CodigoTest        | TVComplement | 1             |
| CE_Abreviado           | TVComplement | 1             |
| CET_Abreviado          | TVComplement | 1             |
| DFED_Resultado         | TVComplement | 1             |
| DFED_ResultadoNumerico | TVComplement | 1             |
| CP_Descripcion         | TVComplement | 0,991667      |
| DFE_TipoMuestra        | TVComplement | 1             |

Tabla 4.3: Reporte Column Shapes para fragmento correspondiente a la categoría *TMVA* de la tabla *laboratorio.csv* al ejecutar el programa con submódulo table divider activado.

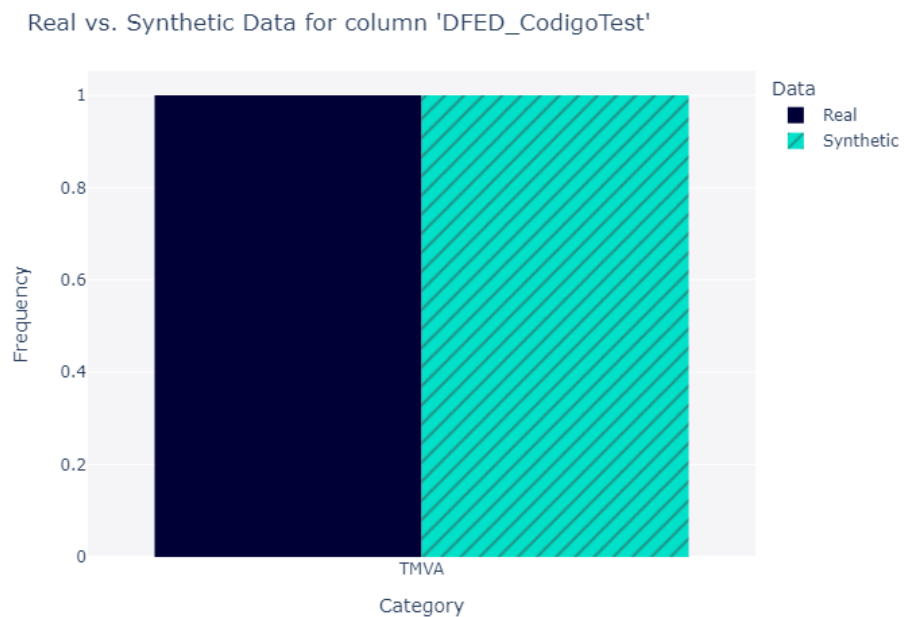


Figura 4.4: Distribución de la columna *DFED\_CodigoTest* al ejecutar el programa con el sub-módulo table divider activado para el fragmento correspondiente a la categoría *TMVA*.



Figura 4.5: Distribución de la columna *DFED\_Resultado* al ejecutar el programa con el sub-módulo table divider activado para el fragmento correspondiente a la categoría *TMVA*.

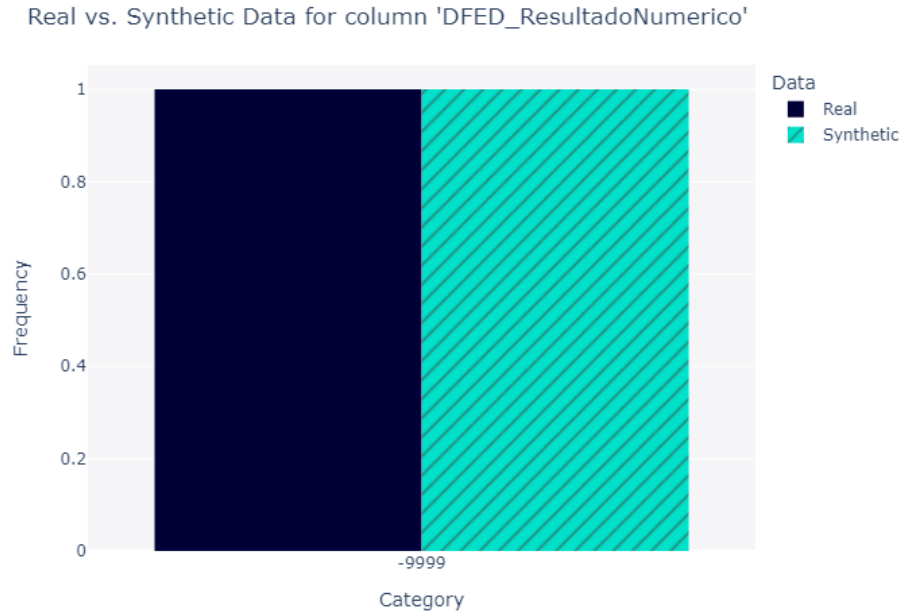


Figura 4.6: Distribución de la columna *DFED\_ResultadoNumerico* al ejecutar el programa con el submódulo table divider activado para el fragmento correspondiente a la categoría *TMVA*.

#### 4.1.3. HEM-FROT-HEM1

| Column                 | Metric       | Quality Score |
|------------------------|--------------|---------------|
| DF_FechaCreacion       | KSComplement | 0,576271      |
| DF_HoraCreacion        | KSComplement | 0,771186      |
| DFED_CodigoTest        | TVComplement | 1             |
| CE_Abreviado           | TVComplement | 1             |
| CET_Abreviado          | TVComplement | 1             |
| DFED_Resultado         | TVComplement | 0,864407      |
| DFED_ResultadoNumerico | TVComplement | 1             |
| CP_Descripcion         | TVComplement | 0,915254      |
| DFE_TipoMuestra        | TVComplement | 0,966102      |

Tabla 4.4: Reporte Column Shapes para fragmento correspondiente a la categoría *HEM-FROT-HEM1* de la tabla *laboratorio.csv* al ejecutar el programa con submódulo table divider activado.



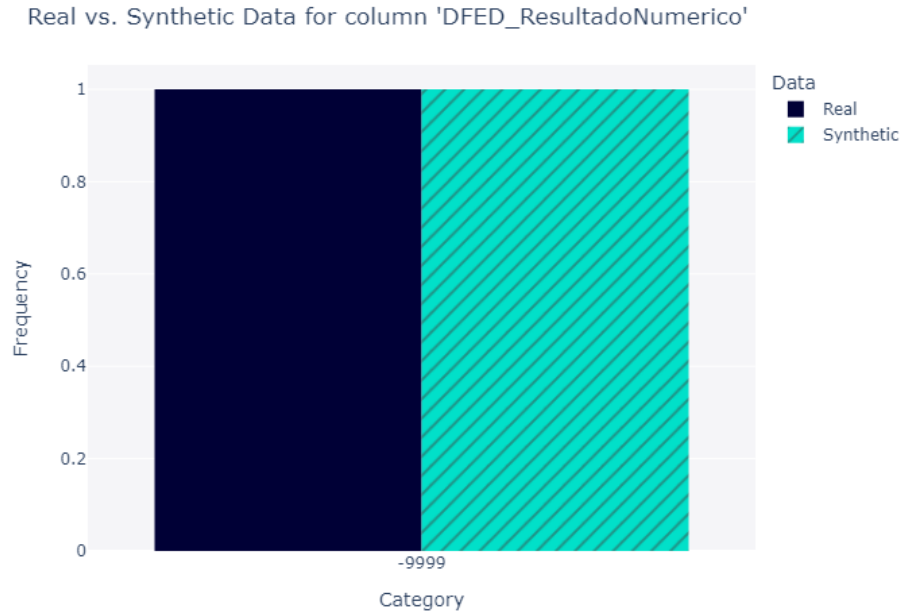


Figura 4.9: Distribución de la columna *DFED\_ResultadoNumerico* al ejecutar el programa con el submódulo table divider activado para el fragmento correspondiente a la categoría *HEM-FROT-HEM1*.

#### 4.1.4. HEM-BAC

| Column                 | Metric       | Quality Score |
|------------------------|--------------|---------------|
| DF_FechaCreacion       | KSComplement | 0,601695      |
| DF_HoraCreacion        | KSComplement | 0,779661      |
| DFED_CodigoTest        | TVComplement | 1             |
| CE_Abreviado           | TVComplement | 1             |
| CET_Abreviado          | TVComplement | 1             |
| DFED_Resultado         | TVComplement | 0,923729      |
| DFED_ResultadoNumerico | TVComplement | 0,915254      |
| CP_Descripcion         | TVComplement | 0,932203      |
| DFE_TipoMuestra        | TVComplement | 0,983051      |

Tabla 4.5: Reporte Column Shapes para fragmento correspondiente a la categoría *HEM-BAC* de la tabla *laboratorio.csv* al ejecutar el programa con submódulo table divider activado.

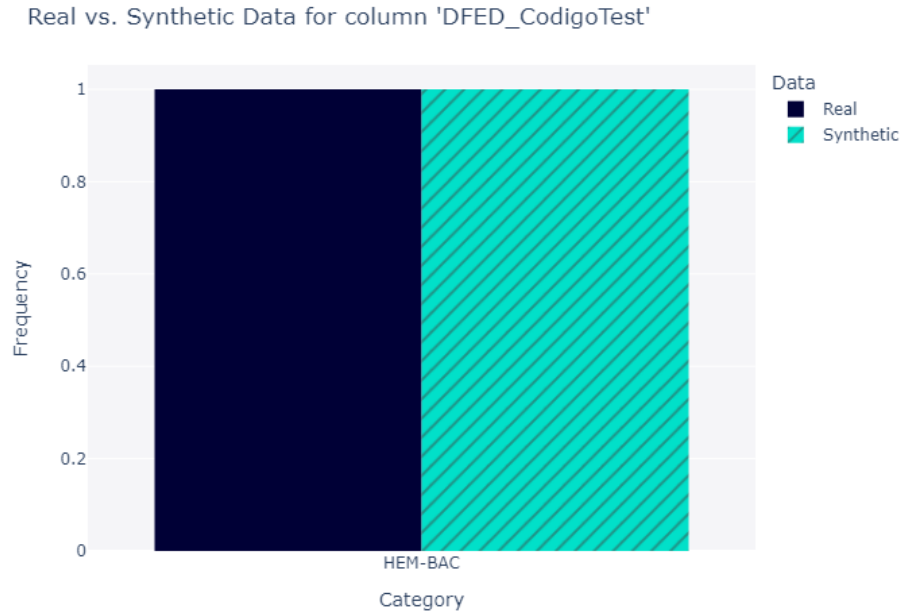


Figura 4.10: Distribución de la columna *DFED\_CodigoTest* al ejecutar el programa con el sub-módulo table divider activado para el fragmento correspondiente a la categoría *HEM-BAC*.

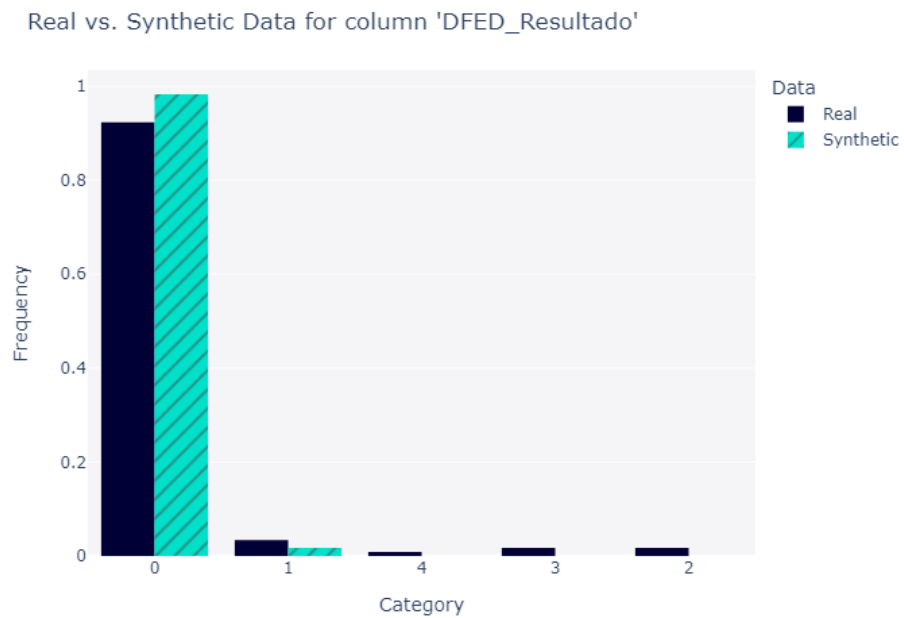


Figura 4.11: Distribución de la columna *DFED\_Resultado* al ejecutar el programa con el sub-módulo table divider activado para el fragmento correspondiente a la categoría *HEM-BAC*.

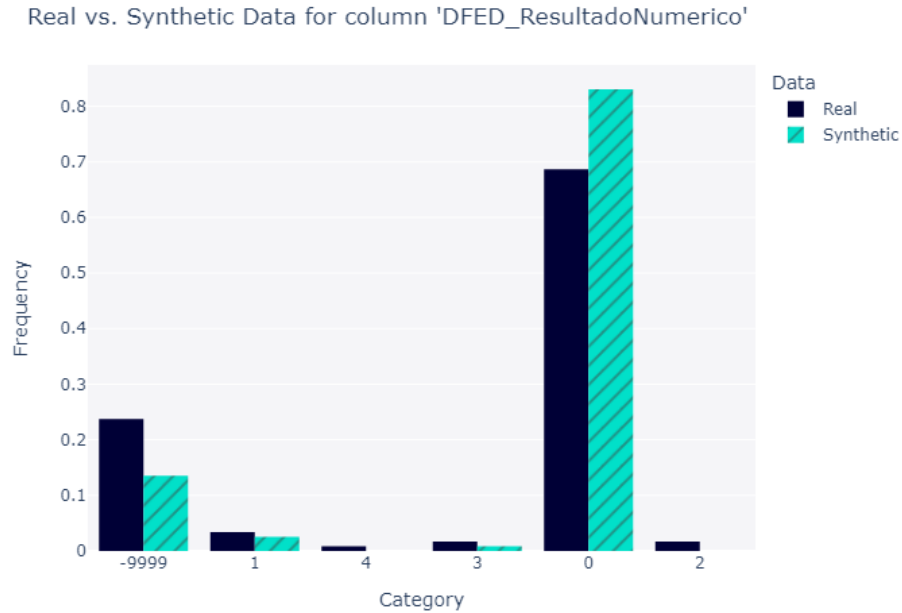


Figura 4.12: Distribución de la columna *DFED\_ResultadoNumerico* al ejecutar el programa con el sub-módulo table divider activado para el fragmento correspondiente a la categoría *HEM-BAC*.

Con respecto de las distribuciones de las columnas *DFED\_Resultado* y *DFED\_ResultadoNumerico* para TMVA (figuras 4.5 y 4.6), HEM-FROT-HEM1 (figuras 4.8 y 4.9) y HEM-BAC (figuras 4.11 y 4.12), se confirma al observar los gráficos que los posibles valores para cada prueba están estrictamente acotados.

Por otro lado, al ver la representación gráfica del reporte **Column Pair Trends** para la tabla sintética antes y después de activar el sub-módulo table divider, presentado en las figuras 4.13 y 4.14 respectivamente, es posible apreciar que se pasa de un puntaje promedio de 0.58 a uno de 0.71. Estos puntajes representan la relación entre columnas, entendiéndose por relación la variación de una al cambiar el valor de otra.

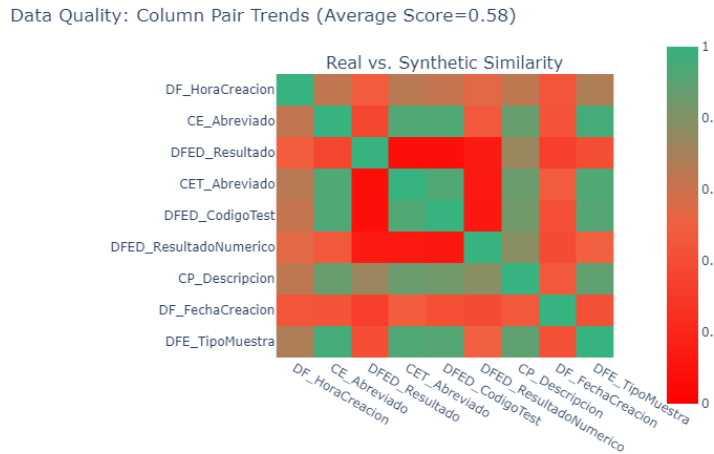


Figura 4.13: Representación gráfica del reporte Column Pair trends para tabla *laboratorio.csv* con sub-módulo table divider desactivado.

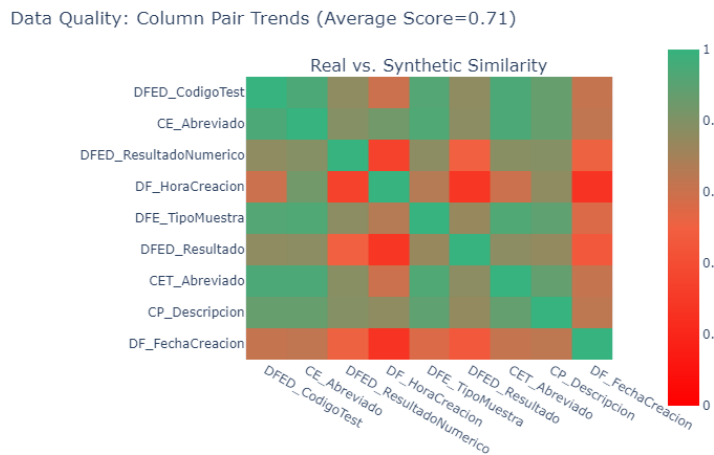


Figura 4.14: Representación gráfica del reporte Column Pair trends para tabla *laboratorio.csv* con sub-módulo table divider activado.

Con el fin de confirmar que al activar la fragmentación de la tabla la relación entre columnas aumenta, se presenta el gráfico de la figura 4.15, en la que se observa el gráfico de correlación entre columnas solo para el fragmento correspondiente a la categoría *TMVA*. En este, las únicas columnas que no conservan relación son aquellas que contienen la fecha y hora del tratamiento, mientras que las demás se encuentran estrictamente relacionadas (puntaje máximo de 1).

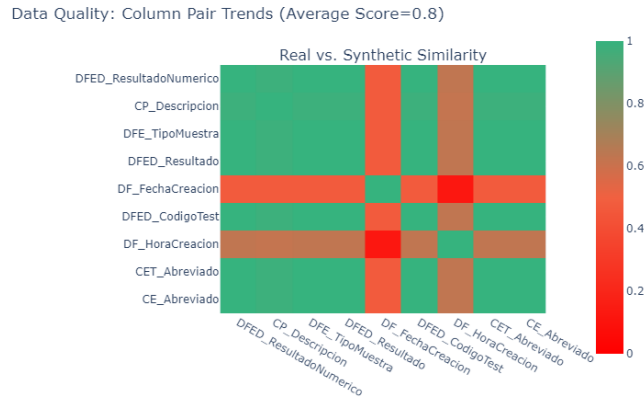


Figura 4.15: Representación gráfica del reporte Column Pair trends correspondiente al fragmento *TM-VA* para la tabla *laboratorio.csv* con sub-módulo table divider activado.

Por último, un elemento que se considera importante destacar es que al fragmentar la tabla, el metadata, es decir la información sobre el tipo de dato (*dtype*) contenido en cada columna del dataframe, no se actualiza durante la fragmentación. De haber diversos tipos de datos en una columna de la tabla original, a dicha columna se le asigna un *dtype* igual a *Objeto* y consecuencia de esto, tanto en la generación de datos como en la evaluación del fragmento, esta se trabaja de forma categórica aún si se expresa de forma numérica. Un ejemplo de esto es visible en los resultados para *HEM-BAC* (figura 4.12), donde los valores de la columna *DFED\_ResultadoNumerico* están expresados de forma numérica, mas el modelo los interpreta y evalúa como valores categóricos.

Se intentó hacer uso de los métodos disponibles en la biblioteca *Pandas* para actualizar automáticamente la información *dtype* en la creación de los fragmentos, pero estos no fueron efectivos. Por otro lado, al modificar manualmente el archivo *laboratorio.csv* para que la columna *DFED\_ResultadoNumerico* contenga únicamente valores numéricos, el modelo sí fue capaz de inferir correctamente el tipo de información en la columna.

#### 4.1.5. Relational table data

Para la evaluación del sub-módulo *relational table data* se hizo uso del archivo *real.xlsx*, libro excel en que cada hoja contiene una de las siguientes tablas: *pacientes*, *imagenologia*, *tratamientos*, *biopsias* y *diagnosticos*. Todas las tablas están enlazadas a la tabla padre *pacientes* través de la columna *PID*, columna contenedora del ID de los pacientes.

Dado que las tablas sólo tienen valores categóricos y datetime, se considera suficiente mostrar los resultados en torno a *pacientes* y *tratamientos*, cuyas estructuras se detallan en las tablas 4.6 y 4.7.

| Nombre de la columna | Tipo          |
|----------------------|---------------|
| PID                  | Identificador |
| SEX                  | Categórico    |
| FECHA_NACIMIENTO     | Datetime      |
| COMUNA               | Categórico    |
| REGION               | Categórico    |
| PREVISION            | Categórico    |
| NACIONALIDAD         | Categórico    |
| fecha_defuncion      | Datetime      |
| lugar_fallecido      | Categórico    |

Tabla 4.6: Estructura de la hoja *pacientes* contenida en el libro excel *real.xlsx*.

| Nombre de la columna   | Tipo          |
|------------------------|---------------|
| PID                    | Identificador |
| FECHA_INICIO           | Datetime      |
| FECHA_FIN              | Datetime      |
| CODIGO_PRESTACION      | Categórico    |
| DESCRIPCION_PRESTACION | Categórico    |
| TIPO_TRATAMIENTO       | Categórico    |
| CONVENIO_ONCOLOGICO    | Categórico    |
| PREVISION              | Categórico    |
| DIAGNOSTICO            | Categórico    |
| CIE10                  | Categórico    |

Tabla 4.7: Estructura de la hoja *tratamientos* contenida en el libro excel *real.xlsx*.

Al ejecutar el programa con *real.xlsx* como archivo de entrada se accede automáticamente a la modalidad *relational table data*, donde se procede a modelar el grupo de tablas y generar la información sintética.

#### *pacientes*

Esta tabla contiene la información relacionada a los pacientes referenciados en cada una de las otras 4 tablas, siendo algunos de los datos disponibles el sexo, lugar de procedencia, fecha de defunción (si existe) y lugar de defunción (si existe).

En la figuras 4.16 se observan los puntajes obtenidos para cada columna de la tabla *pacientes*, mientras que la figura 4.17 muestra el puntaje de la tabla en lo relativo al número de filas generado para cada tabla hija.

Data Quality: Column Shapes (Average Score=0.78)

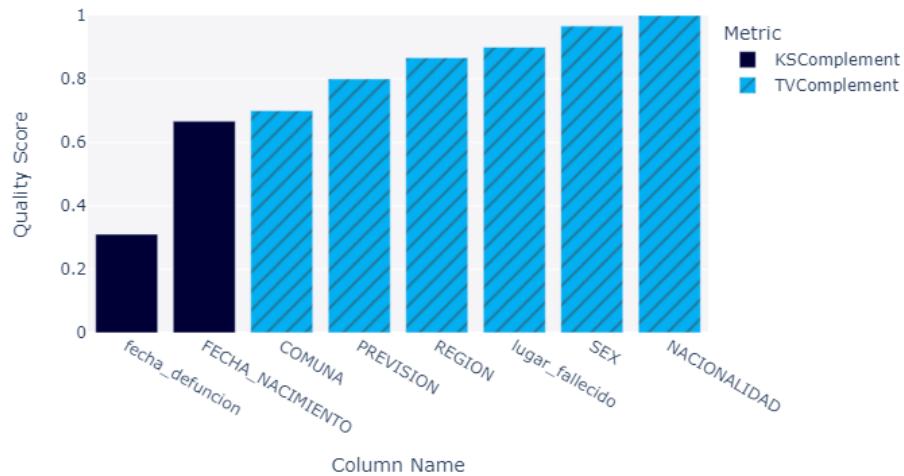


Figura 4.16: Representación gráfica del reporte Column Shapes para tabla *pacientes*

Data Quality: Table Relationships (Average Score=0.82)

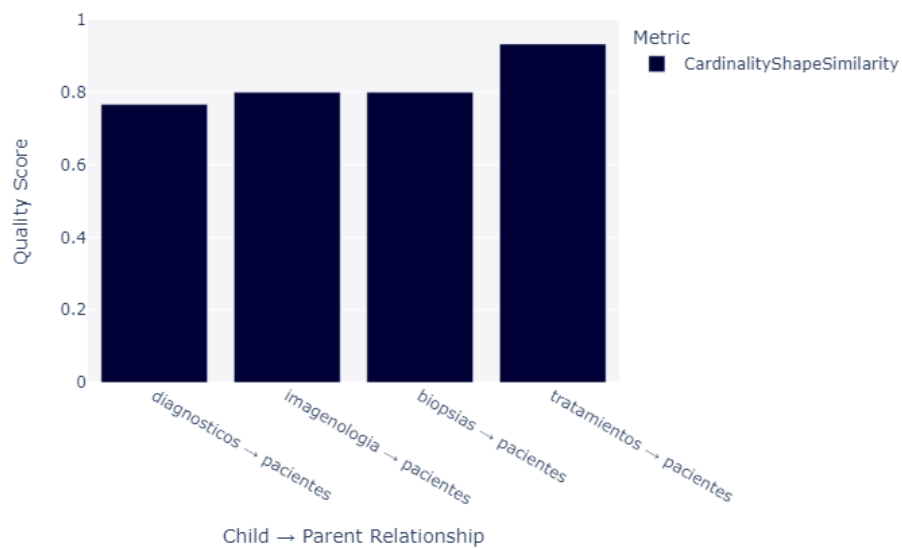


Figura 4.17: Representación gráfica del reporte Parent Child Relationships para la tabla *pacientes*.

Las columnas con peor puntaje en la figura 4.16 son *fecha\_defuncion*, para la métrica *KSComplement* y *COMUNA*, para la métrica *TVComplement*. Mientras que los mejores puntajes se dan a *FECHA\_NACIMIENTO*, para la métrica *KSComplement* y *NACIONALIDAD* y *SEX*, para la métrica

*TVComplement*.

El caso de la columna *NACIONALIDAD* puede ser explicado al destacar que sólo tiene una categoría, correspondiente al valor “CHILENA”. Al igual que todas las columnas que sólo tienen una categoría, la información en dicha columna será siempre 100 % igual a la de la tabla original y por tanto siempre obtendrá la máxima evaluación.

Por otro lado, al observar los gráficos que detallan la distribución de las columnas *COMUNA* y *SEX*, mostradas en las figuras 4.18 y 4.19 respectivamente, se visualiza que en el caso de la columna *COMUNA*, la cual contiene múltiples categorías, se descartan algunas totalmente (Ej. *SAN BERNARDO*) mientras que se aumenta la frecuencia de aparición de otras en aproximadamente un 30 % (Ej. *MELIPILLA*).

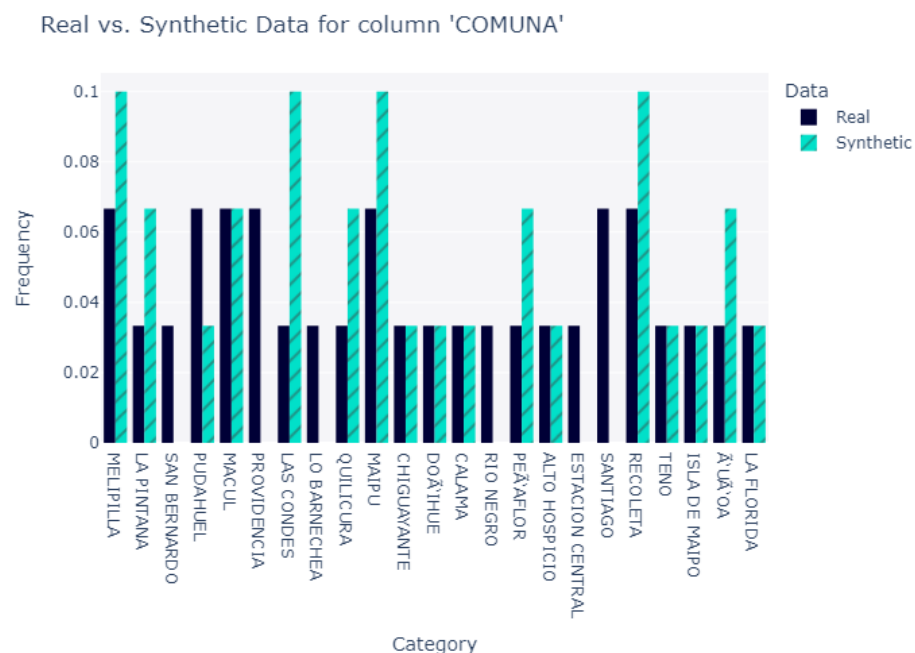


Figura 4.18: Detalle de la distribución de la columna *COMUNA* de la tabla *pacientes*.

En contraste, en la columna de 2 categorías *SEX* se realizan cambios equitativos en la frecuencia de aparición, los cuales no superan el margen del 5 %.

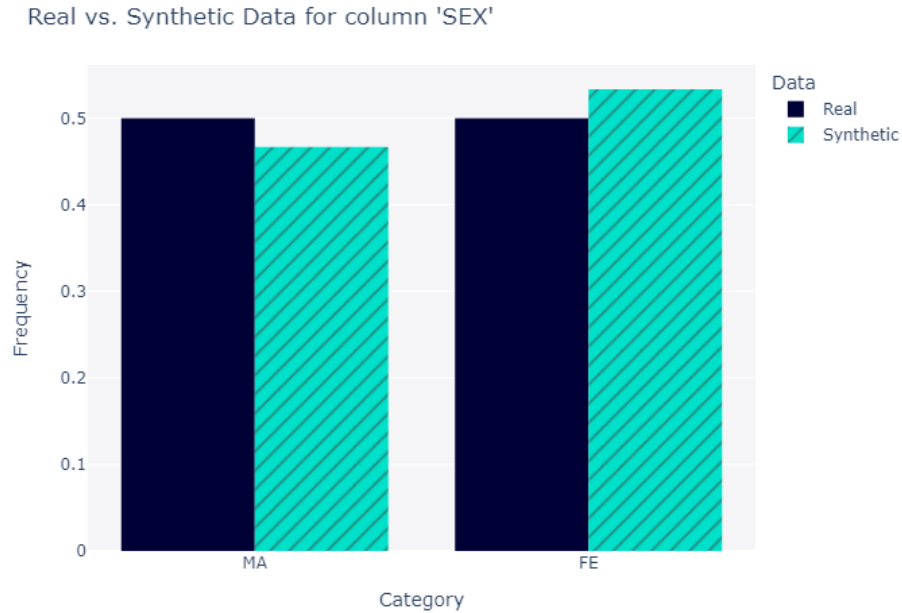


Figura 4.19: Detalle de la distribución de la columna *SEX* de la tabla *pacientes*.

Por otro lado, para el análisis de los gráficos de las columnas *fecha\_defuncion* y *FECHA\_NACIMIENTO*, presentados en las figuras 4.20 y 4.21 respectivamente, es necesario recordar que los resultados se generan a partir de una transformación de las tablas reales que implica la manipulación de las columnas de datos tipo Datetime, descrita en la sección 3.2.2 y mostrada en las tablas 4.8 y 4.9.

| real.xlsx       |                 |
|-----------------|-----------------|
| fecha_defuncion | lugar_fallecido |
|                 |                 |
| 11-12-2020      | FUERA DE FALP   |
|                 |                 |
| 13-01-2020      | FUERA DE FALP   |
|                 |                 |
|                 |                 |
|                 |                 |
|                 |                 |

Tabla 4.8: Primeras 8 filas de las columnas *fecha\_defuncion* y *lugar\_fallecido* de la tabla *pacientes* de los datos reales.

| real.xlsx tras las transformaciones |                 |
|-------------------------------------|-----------------|
| fecha_defuncion                     | lugar_fallecido |
|                                     |                 |
| 2020-12-11 00:00:00                 | FUERA DE FALP   |
|                                     |                 |
| 2020-01-13 00:00:00                 | FUERA DE FALP   |
|                                     |                 |
|                                     |                 |
|                                     |                 |
|                                     |                 |

Tabla 4.9: Primeras 8 filas de las columnas *fecha\_defuncion* y *lugar\_fallecido* de la tabla *pacientes* de los datos reales transformados.

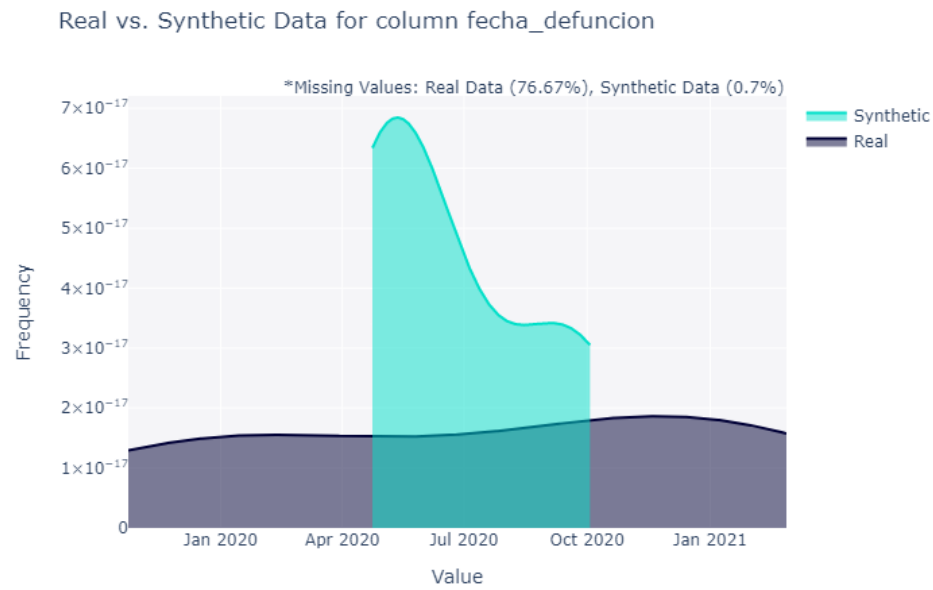


Figura 4.20: Detalle de la distribución de la columna *fecha\_defuncion* de la tabla *pacientes*.

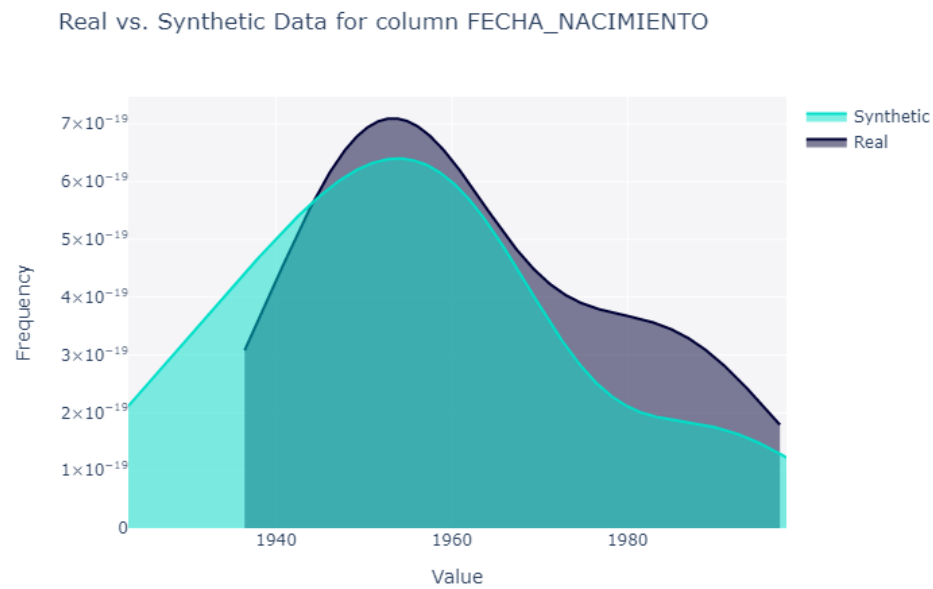


Figura 4.21: Detalle de la distribución de la columna *FECHA\_NACIMIENTO* de la tabla *pacientes*.

Al comparar los gráficos de las columnas *fecha\_defuncion* (contenedora de valores *nan*) y *FECHA\_NACIMIENTO*, es posible decir que a pesar de que en el primero no se logró la misma distribución de valores, estando los datos reales entre finales del 2019 y principios del 2021 y los sintéticos

sólo en el 2020, los resultados siguen manteniéndose dentro de límites realistas. Así mismo, la distribución generada para la segunda columna se asemeja en mayor medida a la de los datos reales, razón de la alta puntuación obtenida.

Al observar los resultados presentados hasta el momento, es posible decir que a rasgos generales el software generador de tablas obtiene resultados aceptables. Sin embargo, si se realiza un análisis más profundo de la tabla *pacientes*, existen 2 condiciones que debe tener la información sintética para ser coherente con la realidad:

- Las columnas *COMUNA* y *REGION* están estrictamente relacionadas, dado que representan la comuna y la región de procedencia del paciente respectivamente. Por lo tanto, cada categoría de *REGION* sólo puede estar asociada a una categoría de *COMUNA*.
- Las columnas *fecha\_defuncion* y *lugar\_fallecido* están relacionadas, dado que de no existir una fecha de defunción tampoco puede existir un lugar de defunción.

En lo que respecta al primer ítem, se pueden observar las distribuciones del par de columnas *COMUNA* y *REGION* antes y después de incorporar una restricción del tipo **FixedC** en las figuras 4.22 y 4.23 respectivamente.

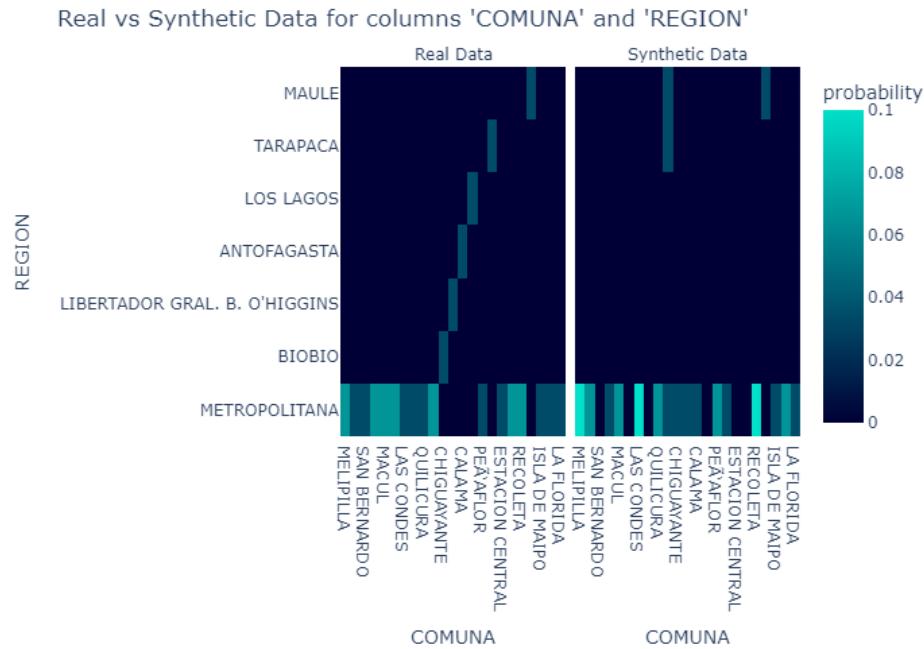


Figura 4.22: Detalle de la distribución generada para el par de columnas *COMUNA-REGION* de la tabla *pacientes*.

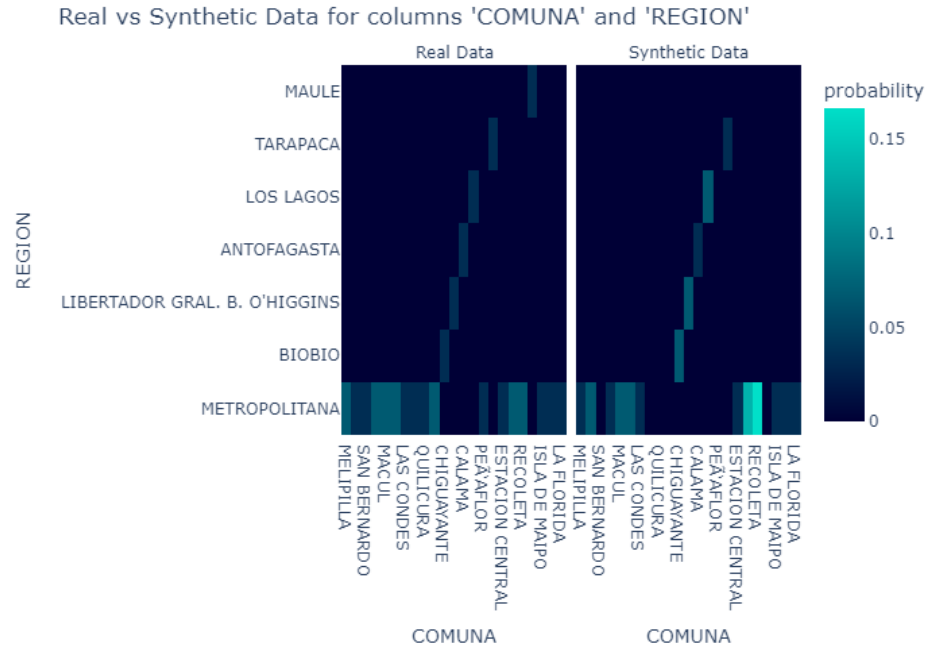


Figura 4.23: Detalle de la distribución generada para el par de columnas *COMUNA-REGION* de la tabla *pacientes* al implementar restricción del tipo **FixedC**.

Previo a incluir la restricción, se aprecia que a algunas regiones se les asocian con comunas que no pertenecen a esa categoría, mientras que al incluir la restricción, es claro que los valores de la columna *COMUNA* se vuelven dependientes del valor de la columna *REGION*.

En lo que respecta al segundo ítem, no es factible incorporar una restricción, dado que de forzar-se las combinaciones entre las columnas involucradas los valores de *fecha\_defuncion* pasarían a ser categóricos, y por lo tanto no se generaría nueva información datetime.

Al observar el extracto de las columnas *fecha\_defuncion* y *lugar\_defuncion* presentado en la tabla 4.10 es notable como el modelo no es capaz de inferir la característica deseada por completo, incorporando fechas que no tienen lugar de defunción o viceversa.

| synthetic.xlsx      |                 |
|---------------------|-----------------|
| fecha_defuncion     | lugar_fallecido |
|                     |                 |
| 2020-04-27 05:00:18 |                 |
| 2020-05-06 21:04:56 | FUERA DE FALP   |
|                     |                 |
|                     |                 |
| 2020-06-07 08:30:28 | FUERA DE FALP   |
| 2020-05-01 02:20:27 | FUERA DE FALP   |
| 2020-08-15 01:15:37 | FUERA DE FALP   |

Tabla 4.10: Primeras 8 filas de las columnas *fecha\_defuncion* y *lugar\_fallecido* de la tabla *pacientes* de los datos sintéticos.

### ***tratamientos***

Esta tabla contiene características del o de los tratamientos que un paciente recibió, siendo parte de la información la fecha de inicio y la fecha de término del tratamiento, correspondientes a las columnas *FECHA\_INICIO* y *FECHA\_FIN* respectivamente.

Similarmente a las columnas *COMUNA* y *REGION* de la tabla *pacientes*, es necesario incorporar una restricción que fuerce a los valores de la columna *FECHA\_FIN* a ser siempre iguales o mayores que los de la columna *FECHA\_INICIO*.

En las figuras 4.22 y 4.23 se pueden observar las distribuciones del par de columnas *FECHA\_INICIO* y *FECHA\_FIN* antes y después de incorporar una restricción del tipo **Inequality** respectivamente.

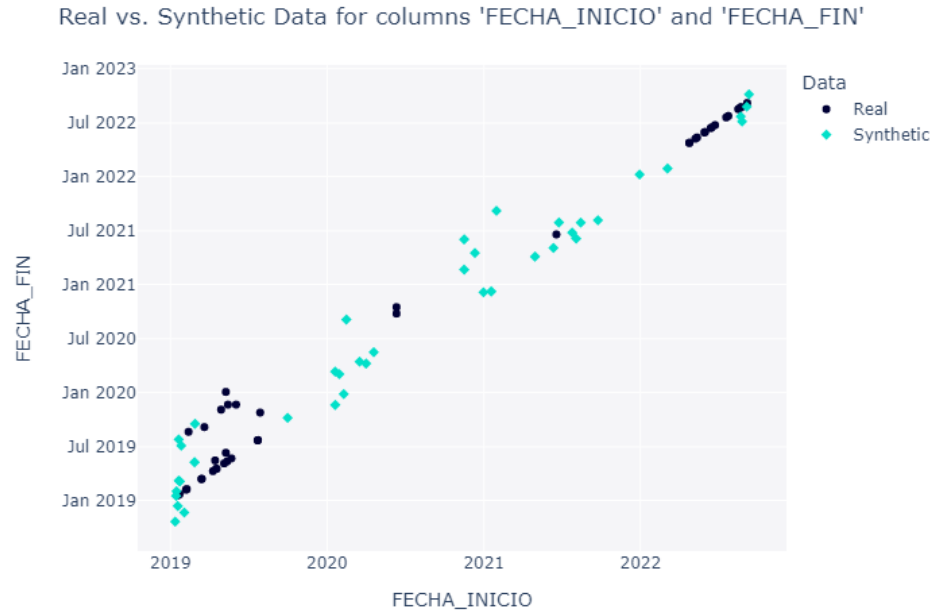


Figura 4.24: Detalle de la distribución generada para el par de columnas *FECHA\_INICIO*-*FECHA\_FIN* de la tabla *pacientes*.

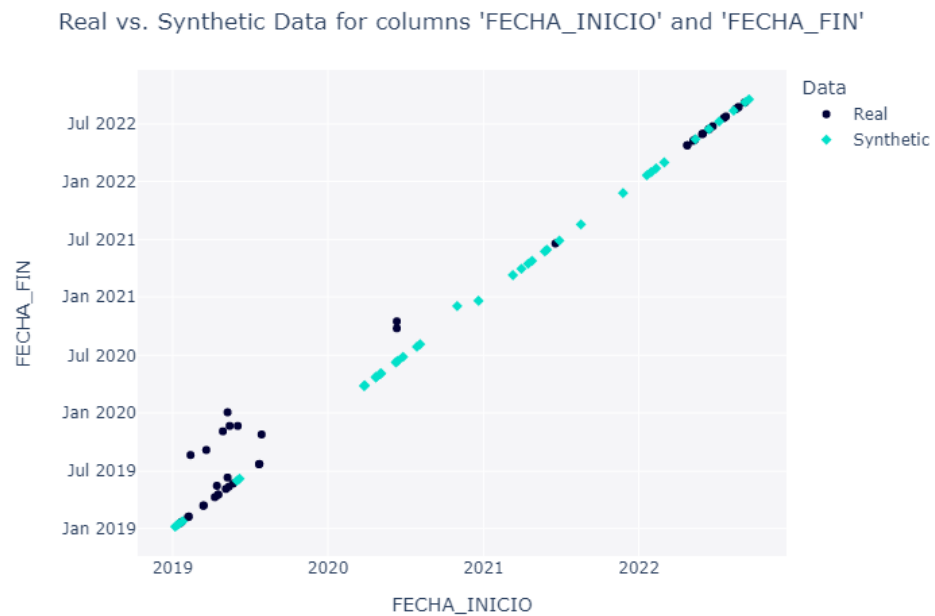


Figura 4.25: Detalle de la distribución generada para el par de columnas *FECHA\_INICIO*-*FECHA\_FIN* de la tabla *pacientes* al implementar restricción del tipo **Inequality**.

Es claro que, tras agregar la restricción, la distribución de las columnas en cuestión se adaptó

para satisfacer que la columna *FECHA\_FIN* no pueda tener valores menores a los de la columna *FECHA\_INICIO*.

## Discusión

Con respecto de las métricas establecidas al comienzo de la sección para las tablas sintéticas, es posible decir que:

- Fuera de las columnas de una sola categoría, la información generada en ningún caso fue exactamente igual a la información real.
- La distribución de los valores correspondientes a cada columna siempre se vio acotada dentro de los límites establecidos por los datos reales.

Sin embargo, tanto el modelo para tablas singulares como para tablas relacionales omitieron algunos de los valores categóricos. Al ejecutar el programa nuevamente con los mismos archivos de configuración fue posible notar que en cada oportunidad existían cambios en la distribuciones. Sin embargo, en la mayoría de los casos se seguían omitiendo algunas de las categorías con menor frecuencia de aparición.

- El modelo es capaz de inferir la mayoría de las características implícitas en los datos reales. Sin embargo, la implementación de estas sobre los datos sintéticos no son necesariamente estrictas. Esto se evidencia con la omisión de categorías en algunas ejecuciones, así como con el caso discutido de las fechas de defunción y el lugar de defunción.

Por otro lado, para el caso de características que el modelo es incapaz de inferir, estas pueden ser forzadas con las restricciones disponibles. La única excepción de la que se tiene conocimiento es el caso en que el valor numérico de una columna está asociado al valor categórico de otra, dado que el implementar una restricción puede disminuir la calidad de la información sintética.

Tomando en cuenta lo anteriormente planteado es posible decir que, dependiendo de los datos que se quiera sintetizar y que tan exigentes sean las necesidades del estudio, los resultados obtenidos podrían ser suficientes para ser usados en la investigación.

Un elemento de alto impacto en esto es el formato de la información. Es evidente que los bloques vacíos tienen un efecto negativo en la generación de datos, así como las columnas que mezclan tipos de datos.

## 4.2. Resultados mamografías sintéticas: Encuesta

En lo que respecta a la encuesta realizada para evaluar la calidad de las imágenes sintéticas, profundizada al comienzo de esta sección y accesible a través del link <https://www.surveio.com/survey/d/M1J7H1Z2F9Z8U7M2T?preview=1>, fue posible recaudar 3 respuestas. Una de una persona sin formación en la lectura de mamografías y dos de personas con formación no especializada (médicos generales o con especialización en otra área).

Las respuestas de los encuestados son presentadas a continuación como tablas correspondientes a las secciones 2 y 3 de la encuesta diseñada. Las tablas 4.11, 4.13 y 4.15 muestran los resultados para las 5 preguntas de evaluación de características (4 sintéticas y 1 de control), para cada una de las 3 características de interés, la evaluación general de la imagen y sus puntajes promediados. Por otro lado, las tablas 4.12, 4.14 y 4.16 muestran los resultados para las preguntas de selección, detallando el número de aciertos y el número de veces que cada característica ayudó en la decisión de selección.

| Sin formación          | Preguntas de evaluación |   |   |   | Promedio | Control |
|------------------------|-------------------------|---|---|---|----------|---------|
| Pregunta               | 1                       | 2 | 3 | 4 | -        | 5       |
| Forma (1-7)            | 5                       | 4 | 2 | 4 | 3,75     | 6       |
| Textura (1-7)          | 5                       | 3 | 4 | 4 | 4        | 2       |
| Disposición (1-7)      | 7                       | 3 | 2 | 6 | 4,5      | 4       |
| Calidad general (1-10) | 7                       | 5 | 3 | 7 | 5,5      | 4       |

Tabla 4.11: Respuestas de la segunda sección de la encuesta del encuestado sin formación.

| Sin formación       | Preguntas de selección |   |   |   |   |   |   |   |   |    | Aciertos / Veces seleccionada |
|---------------------|------------------------|---|---|---|---|---|---|---|---|----|-------------------------------|
| Pregunta            | 1                      | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |                               |
| Imagen seleccionada | 1                      | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 2  |                               |
| Imagen verdadera    | 1                      | 1 | 2 | 1 | 2 | 2 | 2 | 1 | 1 | 1  | 6 de 10                       |
| Forma               |                        |   | x |   |   |   |   | x | x |    | 3                             |
| Textura             |                        | x |   |   | x |   |   |   |   | x  | 3                             |
| Disposición         | x                      |   |   | x | x | x | x |   |   |    | 5                             |

Tabla 4.12: Respuestas de la tercera sección de la encuesta del encuestado sin formación.

| Médico 1               | Preguntas de evaluación |    |    |   | Promedio | Control |
|------------------------|-------------------------|----|----|---|----------|---------|
| Pregunta               | 1                       | 2  | 3  | 4 | -        | 5       |
| Forma (1-7)            | 6                       | 7  | 7  | 7 | 6,75     | 4       |
| Textura (1-7)          | 7                       | 6  | 7  | 4 | 6        | 6       |
| Disposición (1-7)      | 6                       | 7  | 7  | 5 | 6,25     | 7       |
| Calidad general (1-10) | 10                      | 10 | 10 | 8 | 9,5      | 10      |

Tabla 4.13: Respuestas de la segunda sección de la encuesta del primer encuestado con formación no especializada.

| Médico 1            | Preguntas de selección |   |   |   |   |   |   |   |   |    | Aciertos / Veces seleccionada |
|---------------------|------------------------|---|---|---|---|---|---|---|---|----|-------------------------------|
| Pregunta            | 1                      | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |                               |
| Imagen seleccionada | 1                      | 1 | 2 | 1 | 2 | 2 | 2 | 2 | 2 | 1  |                               |
| Imagen verdadera    | 1                      | 1 | 2 | 1 | 2 | 2 | 2 | 1 | 1 | 1  | 8 de 10                       |
| Forma               |                        |   |   |   |   |   |   |   |   |    | 0                             |
| Textura             | x                      | x | x | x | x | x | x | x | x | x  | 10                            |
| Disposición         |                        |   |   |   |   |   |   |   |   |    | 0                             |

Tabla 4.14: Respuestas de la tercera sección de la encuesta del primer encuestado con formación no especializada.

| Médico 2               | Preguntas de evaluación |   |   |   | Promedio | Control |
|------------------------|-------------------------|---|---|---|----------|---------|
| Pregunta               | 1                       | 2 | 3 | 4 | -        | 5       |
| Silueta (1-7)          | 5                       | 3 | 6 | 5 | 4,75     | 6       |
| Textura (1-7)          | 4                       | 5 | 4 | 5 | 4,5      | 4       |
| Disposición (1-7)      | 6                       | 5 | 4 | 5 | 5        | 5       |
| Calidad general (1-10) | 6                       | 5 | 6 | 6 | 5,75     | 6       |

Tabla 4.15: Respuestas de la segunda sección de la encuesta del segundo encuestado con formación no especializada.

| Médico 2            | Preguntas de selección |   |   |   |   |   |   |   |   |    | Aciertos / Veces seleccionada |
|---------------------|------------------------|---|---|---|---|---|---|---|---|----|-------------------------------|
| Pregunta            | 1                      | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |                               |
| Imagen seleccionada | 2                      | 2 | 2 | 1 | 2 | 2 | 2 | 2 | 1 | 2  |                               |
| Imagen verdadera    | 1                      | 1 | 2 | 1 | 2 | 2 | 2 | 1 | 1 | 1  | 8 de 10                       |
| Forma               |                        |   | x |   |   |   |   |   | x |    | 2                             |
| Textura             |                        | x |   |   | x | x | x | x |   | x  | 6                             |
| Disposición         | x                      |   |   | x |   |   |   |   |   |    | 2                             |

Tabla 4.16: Respuestas de la tercera sección de la encuesta del segundo encuestado con formación no especializada.

Tomando en cuenta los puntajes máximos para cada pregunta de la segunda sección, es posible normalizar entre un valor del 0 al 1 la evaluación total del set de imágenes utilizadas en la encuesta a través de la ecuación 4.1:

$$\frac{P - 4}{27} \quad (4.1)$$

Donde P es la suma de los promedios para cada uno de los 4 puntos evaluados: textura, forma, disposición y calidad general.

Así, es posible llegar a una evaluación de 0.51, 0.91 y 0.59 a partir de las tablas 4.11, 4.13 y 4.15 respectivamente. De la misma forma, al hacer el cálculo para la pregunta de control se llegan a los valores 0.44, 0.85 y 0.63 respectivamente.

En lo que respecta a la sección 3 de la encuesta, las características textura, forma y disposición fueron escogidas 5, 19 y 7 veces respectivamente de un total de 30. Así mismo, las imágenes sintéticas fueron escogidas sobre las reales solo 8 veces de un total de 30.

## Discusión

Observando los resultados previamente presentados es posible decir que la característica más problemática para la sintetización es la textura de los tejidos internos de la mama, ya que fue tanto la peor evaluada en la sección 2 de la encuesta como la más seleccionada en la sección 3. Así mismo, a pesar de tener puntajes aceptables es evidente que incluso un ojo no entrenado tiende a distinguir entre las imágenes sintéticas y las reales.

Por otro lado, la pregunta de control da cuenta de que al asegurar que la imagen visualizada

es sintética los encuestados tendieron a evaluar las características de interés con bajos puntajes. Se estima que esto tiene relación con el contexto del problema, dado que el cuerpo humano puede variar mucho de persona en persona es complejo establecer límites que permitan definir claramente una mama “normal”.

Además, una observación importante es que para la evaluación de las imágenes es necesario tener en consideración que existen muchos factores que afectan a la respuestas recopiladas. Un factor determinante en esto es el dispositivo sobre el cual se contesta la encuesta, puesto que el tamaño de las imágenes cambia de contestarse en un celular, un notebook o un monitor grande. Similarmente, las 14 imágenes sintéticas utilizadas en la encuesta fueron seleccionadas de forma arbitraria, por lo que al utilizar otro set de imágenes los resultados podrían variar.

## Capítulo 5

# Conclusiones

El trabajo realizado abarcó la generación de datos sintéticos en torno a registros clínicos, pruebas de laboratorio y mamografías. Tras una investigación sobre el estado del arte, se hizo uso de las técnicas escogidas para desarrollar un modelo generativo GAN para la sintetización de mamografías y un programa para la sintetización de tablas, con los cuales fue posible obtener resultados para cada uno de los 3 tipos de datos previamente mencionados.

Al analizar los resultados presentados en este trabajo es posible decir que, con respecto a la generación de imágenes, la calidad de las mamografías generadas pocas veces está al nivel de las mamografías reales.

Cabe destacar que las pruebas realizadas dan cuenta que la calidad de las imágenes es estrictamente dependiente de los recursos computacionales disponibles y del volumen de datos utilizados para el entrenamiento, por lo que los resultados mejorarían de utilizarse un mayor volumen de información.

Además, un elemento importante a tomar en consideración es la capacidad de los modelos generativos de modelar la presencia de cáncer. Sin embargo, existen 2 motivos por los cuales las GAN son una herramienta inadecuada para esta tarea. En primer lugar, la presencia de cáncer en mamografías siempre es evaluada por profesionales a través de la interpretación de dos imágenes, las cuales corresponden a un corte horizontal y vertical de la mama que en conjunto otorgan una percepción 3D del interior de esta. A pesar de que los resultados demuestran la capacidad que tienen los modelos GAN para sintetizar mamografías, no es posible para el modelo inferir características entre imágenes, y por lo tanto no es posible replicarlas en la sintetización. En segundo lugar, aún si omitimos la necesidad de generar imágenes en pares, es posible forzar rasgos si se dispone de suficientes datos que los muestren claramente. Tras el entrenamiento, el modelo podría generar imágenes con las características deseadas. Sin embargo, esto sería producto de intentar imitar la distribución probabilística de características

del dataset original y no por un reconocimiento directo de ese rasgo. En adición a lo anterior, es sabido que el cáncer puede presentarse en distintas formas como anomalías en el tejido, mas la presencia de esas anomalías no implican necesariamente que sea cáncer (razón por la que una interpretación 3D es necesaria). Por este motivo, se dificulta aún más el desarrollo de un modelo que sea capaz de controlar la presencia de cáncer en imágenes.

Dicho esto, es posible concluir en cuanto a modelos generativos para sintetización de imágenes, que estos tienen la capacidad de generar mamografías que, en pocas ocasiones, son capaces de engañar al ojo humano, tanto profesional como no. Consecuencia de esto, dependiendo del uso que se le quiera dar a esta herramienta, hay muchos factores que hacen que sea insuficiente para su uso en investigación.

Por otro lado, el análisis efectuado en torno a los resultados obtenidos para la sintetización de registros clínicos y pruebas de laboratorio, permite decir que en este ámbito los modelos generativos tienen un alcance mucho mayor que con imágenes. Sin embargo, de requerirse que se cumplan condiciones que escapan de la capacidad de inferencia del modelo y de las características que pueden forzarse con las restricciones incluidas en el programa, los resultados obtenidos podrían ser insuficientes.

Además, se considera necesario destacar que, tras modelar una tabla, cada ejecución del programa generará distintos resultados. Esta característica, propia de la SDV, puede obstruir algunas labores de investigación pues los resultados obtenidos no serán 100 % replicables. Este problema puede solucionarse al fijar la semilla utilizada para la generación de la biblioteca SDV, mas no fue posible su modificación durante el desarrollo de este trabajo.

Como trabajo posterior, se debe mejorar las capacidades de inferencia del modelo, especialmente en la implementación del sub-módulo table divider, puesto que en el tipo de trabajo para el que está diseñado será normal encontrar distintos tipos de datos dentro de una misma columna. Dado que los métodos disponibles para la modificación automática de *dtypes* en variables de tipo *Dataframe* no fueron efectivas, se estima que será necesario buscar métodos para forzar una re-lectura de cada fragmento que se trabaje.

A modo de cierre, es posible decir que los objetivos, tanto el general como los específicos, fueron cumplidos satisfactoriamente, pues fue posible establecer el alcance de los distintos modelos generativos en torno a las 2 áreas estudiadas, así como el diseño de un programa capaz de sintetizar tablas realistas haciendo uso de datos reales.

# Bibliografía

- [Abufadda and Mansour, 2021] Abufadda, M. and Mansour, K. (2021). A survey of synthetic data generation for machine learning. In *2021 22nd International Arab Conference on Information Technology (ACIT)*, pages 1–7.
- [Arjovsky et al., 2017] Arjovsky, M., Chintala, S., and Bottou, L. (2017). Wasserstein gan.
- [Goodfellow et al., 2014] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial nets. In Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N., and Weinberger, K., editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc.
- [Karras et al., 2017] Karras, T., Aila, T., Laine, S., and Lehtinen, J. (2017). Progressive growing of gans for improved quality, stability, and variation.
- [Patki et al., 2016] Patki, N., Wedge, R., and Veeramachaneni, K. (2016). The synthetic data vault. In *2016 IEEE International Conference on Data Science and Advanced Analytics (DSAA)*, pages 399–410.
- [Radford et al., 2015] Radford, A., Metz, L., and Chintala, S. (2015). Unsupervised representation learning with deep convolutional generative adversarial networks.
- [Sedigh et al., 2019] Sedigh, P., Sadeghian, R., and Masouleh, M. T. (2019). Generating synthetic medical images by using gan to improve cnn performance in skin cancer classification. In *2019 7th International Conference on Robotics and Mechatronics (ICRoM)*, pages 497–502.
- [Yoon et al., 2020] Yoon, J., Drumright, L. N., and van der Schaar, M. (2020). Anonymization through data synthesis using generative adversarial networks (ads-gan). *IEEE Journal of Biomedical and Health Informatics*, 24(8):2378–2388.