MACHINE LEARNING PROJECT

TRABAJO CIENCIA DE DATOS DIABETES

Udemy

GRADO EN MATEMÁTICAS

ALUMNA: Siria Catherine Íñiguez Brito

Enero de 2025

Resumen

Enfermedades como la diabetes son una de las principales causas de muerte en el mundo, afectando a más de 1.6 millones de personas según los datos de la OMS. Por lo tanto, su detección temprana juega un papel significativamente importante para prevenir complicaciones graves como enfermedades cardiovasculares, daños renales o derrames cerebrales. Este proyecto busca utilizar técnicas de *machine learning* para crear un modelo predictivo que, basado en informes médicos de una población, pueda clasificar de manera eficiente y precisa si un paciente tiene riesgo de padecer diabetes, ayudando así a los especialistas a tomar decisiones informadas y mejorar resultados en la salud pública.

Palabras Clave: diabetes, aprendizaje automatizado, modelos predictivos, análisis de datos.

‘

Índice

Índice de figuras

Índice de tablas

1 INTRODUCCIÓN

En esta nueva era de las tecnologías han florecido una infinidad de algoritmos tales como la regresión linel, regresión logística, árboles de decisión, redes neuronales enfocados en resolver problemas del mundo real utilizando técnicas relacionadas con el aprendizaje automatizado.

En particular, el aprendizaje automatizado tiene múltiples aplicaciones en el ámbito de la medicina, como: diagnósticos más precisos y personalizados, reducción de errores médicos, alerta de riesgos e identificación temprana de enfermedades, entre muchas otras. Estas aplicaciones no solo tienen un impacto positivo en la calidad de vida de los pacientes, sino también en la eficiencia del sistema de salud.

En el presente trabajo, se busca, utilizando técnicas de *machine learning*, desarrollar un modelo que en base a unas determinadas características, sea capaz de clasificar si un paciente padecerá de diabetes.

1.1 Objetivos

Este projecto tiene como objetivo principal aplicar los conocimientos adquiridos en el curso online por Udemy: **'Máster de especialista en Ciencia de Datos con Python'**. Dichos conocimientos se enfocarán en resolver un problema concreto de clasificación que consiste en predecir si un paciente padecerá diabetes.

1.2 Metodología

La metodología seguida en este proyecto consta de las siguientes etapas:

- **Información:** primeramente se toma como base principal el curso de 'Máster de especialista en Ciencia de Datos con Python' ofrecido por Udemy. A partir de este curso, se estructurará el poyecto, se seleccionará el conjunto de datos con el que vamos a trabajar y se indentificarán las herramientas de *machine learning* necesarias para aplicar el modelo que mejor se ajusta al problema.

- **Análisis:** esta fase está efocada en aplicar las herrmaientas aprendidas durante la fase previa incuyendo: análisis exploratorio de los datos, fase de procesamiento de los datos, fase de tratamiento de los datos. Esto permitirá elegir un modelo adecuado para el problema.

- **Implementación:** en esta etapa se buscará el algoritmo que pueda predecir con mayor preción nuestra variable dependiente, ajustándose mejor a las necesidades del problema planteado.

- **Valoración:** para poder realizar una valoración del algoritmos elegido, se utilizaran varias técnicas de *machine learning*. En caso de un juicio totalmente negativo, se evaluará la posibilidad de revisar y ajustar las etapa previas. En el caso de resultados aceptables, se encauzará el camino para la realización de trabajos futuros.

2 ANÁLISIS EXPLORATORIO DE LOS DATOS

En este proyecto se elegirá un modelo, aplicando técnicas de machine learning que nos permita predecir el valor de nuestra variable respuesta. En este caso particular, se trata de un problema de clasificación binaria, cuyo objetivo es predecir en base a una serie de características, si un paciente padece o no de diabetes. Para llevar a cabo el estudio de *machine learning*, se utilizará como lenguaje de programación Python, adjuntando además el archivo de nombre *Diabetes.py* en el repositorio https://github.com/Siria-Catherine-Iniguez-Brito/Diabetes-Prediction-Using-Data-Science-Project.git.

El conjuto de datos utilizado proviene de https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database/data que contiene información mádica de mujeres indias. Este conjunto consta de 768 observaciones y 9 variables descritas de la siguiente manera:

- **Preg:** número de veces que la paciente estuvo embarazada.
- **Plas:** concentración plasmática de glucosa a las 2 horas en una prueba de tolerancia a la glucosa oral .
- **Pres:** presión arterial diastólica (mm Hg).
- **Skin:** grosor del pliegue cutáneo del tríceps (mm).
- **Test:** insulina sérica de 2 horas (muU/ml).
- **Mass:** índice de masa corporal (peso en kg/(altura en m^2) .
- **Pedi:** función de pedigrí de diabetes .
- **Age:** edad de la paciente en años.
- **Class:** padece o no diabetes.

	preg	plas	pres	skin	test	mass	pedi	age	class
0	6	148	72	35	0	33.6	627.00	50	1
1	1	85	66	29	0	26.6	351.00	31	0
2	8	183	64	0	0	23.3	672.00	32	1
3	1	89	66	23	94	28.1	167.00	21	0
4	0	137	40	35	168	43.1	2288.00	33	1
..	...	...	...	...	...	...	...	...	...
763	10	101	76	48	180	32.9	171.00	63	0
764	2	122	70	27	0	36.8	0.34	27	0
765	5	121	72	23	112	26.2	245.00	30	0
766	1	126	60	0	0	30.1	349.00	47	1
767	1	93	70	31	0	30.4	315.00	23	0

[768 rows x 9 columns]

Figura 1: Data set.

En este caso se toma como variable respuesta 'class' que toma dos valores: **0** (negativo, el paciente no tiene diabetes) y **1** (positivo, el paciente tiene diabetes) indicándonos así, si el paciente tiene o no diabetes.

Con el objetivo de obtener algún conocimiento previo sobre las variables, utilizamos la función def describe( ) que aporta información importante acerca de: el nº de observaciones de cada variable, la media, la desviación típica, el mínimo, los percentiles 25, 50 y 75, y el máximo como se observa en la siguiente tabla.

	preg	plas	pres	skin	...	mass	pedi	age	class
count	768.000	768.000	768.000	768.000	...	768.000	768.000	768.000	768.000
mean	3.845	120.895	69.105	20.536	...	31.993	428.235	33.241	0.349
std	3.370	31.973	19.356	15.952	...	7.884	340.486	11.760	0.477
min	0.000	0.000	0.000	0.000	...	0.000	0.100	21.000	0.000
25%	1.000	99.000	62.000	0.000	...	27.300	205.000	24.000	0.000
50%	3.000	117.000	72.000	23.000	...	32.000	337.000	29.000	0.000
75%	6.000	140.250	80.000	32.000	...	36.600	591.500	41.000	1.000
max	17.000	199.000	122.000	99.000	...	67.100	2329.000	81.000	1.000

Figura 2: Función describe.

Para comenzar, se muestra un análisis exploratorio de los datos, obteniendo el tipo de las variables, la correlación, el sesgo y un histograma de estas de forma complementaria para tener una vista preliminar.

2.1 Tipo de variables

Usando la función def typeof( ), se identifican los tipos de variables presentes.

```
preg       int64
plas       int64
pres       int64
skin       int64
test       int64
mass     float64
pedi     float64
age        int64
class      int64
dtype: object
```

Figura 3: Tipo de las variables.

Al analizar las Figuras 2 y 3, se observa que todas las variables explicativas son numéricas, y la variable respuesta a pesar de adoptar valores enteros se trata de una variable categórica. Al aplicar la función def groupby('class').size() se aprecia que el modelo está desbalanceado con 500 registros correspondientes a pacientes sin diabetes y 268 a pacientes con diabetes. Este desbalance en las clases podría influir en la fiabilidad del algoritmo de *machine learning* arrojando resultados sesgados hacia la clase mayoritaria, por lo que en etapas posteriores se deberá considerar este un factor importante.

2.2 Correlación

De manera orientativa e informativa se analizará la correlación de cada par de variables. El objetivo es conseguir que en el modelo cada predictor esté relacionado con la variable respuesta y reducir el riesgo de añadir variables que nos aporten la misma información, para ello, usamos la función def corr (method ='pearson').

Matriz de correlación

	preg	plas	pres	skin	test	mass	pedi	age	class
preg	1.000	0.129	0.141	-0.082	-0.074	0.018	-0.026	0.544	0.222
plas	0.129	1.000	0.153	0.057	0.331	0.221	0.133	0.264	0.467
pres	0.141	0.153	1.000	0.207	0.089	0.282	0.051	0.240	0.065
skin	-0.082	0.057	0.207	1.000	0.437	0.393	0.154	-0.114	0.075
test	-0.074	0.331	0.089	0.437	1.000	0.198	0.185	-0.042	0.131
mass	0.018	0.221	0.282	0.393	0.198	1.000	0.104	0.036	0.293
pedi	-0.026	0.133	0.051	0.154	0.185	0.104	1.000	0.018	0.177
age	0.544	0.264	0.240	-0.114	-0.042	0.036	0.018	1.000	0.238
class	0.222	0.467	0.065	0.075	0.131	0.293	0.177	0.238	1.000

Figura 4: Matriz de correlación.

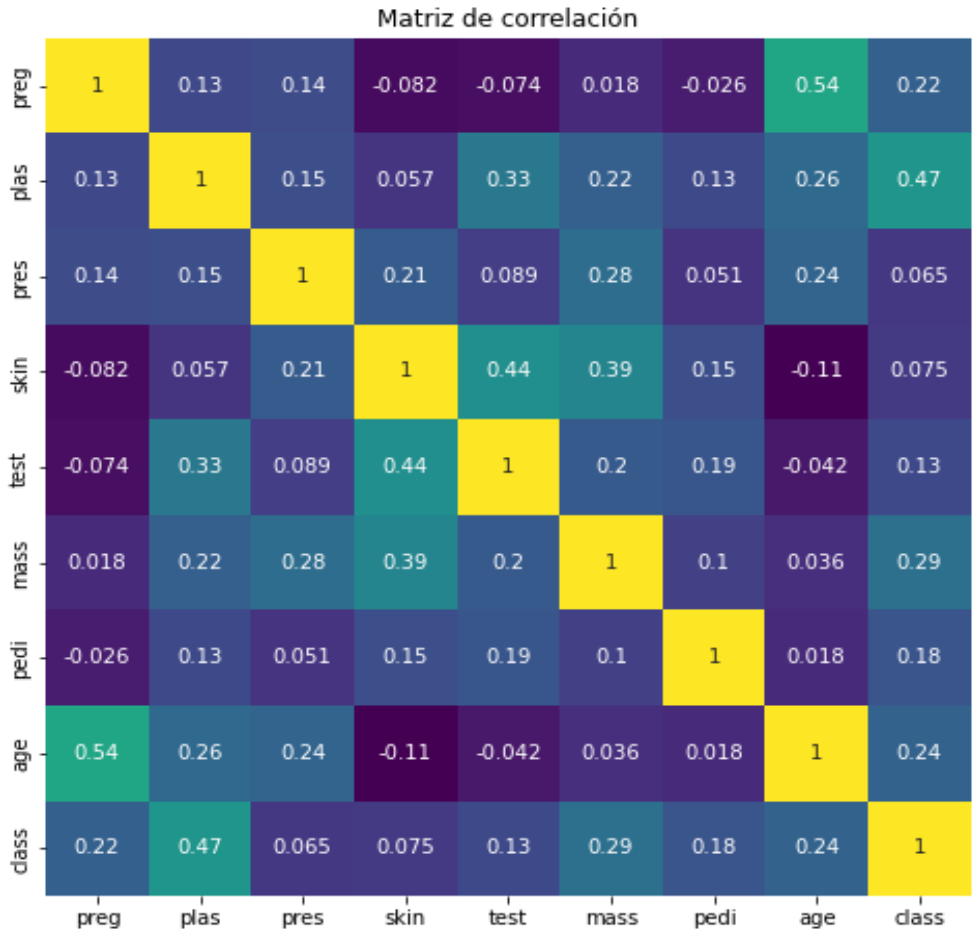

Figura 5: Visualización de la matriz de correlación.

- **Correlación entre atributos:** según esta, se puede medir si dos variables son dependientes o independientes. Idealmente la correlación entre características debe situarse en el rango (-0.75, 0.75). Por ejemplo las variable 'preg' y 'skin' tienen una baja correlación, en este caso concreto de -0.082.

- **Correlación entre atributos y clase:** es preferible que la correlación entre las características y la variable respuesta se situe en (-1, -0.75) y (0.75, 1) indicándo así una dependencia lineal entre ellas y por lo tanto un comportamiento más óptimo del algoritmo. La correlación más alta entre 'class' y alguna de las características es con la variable 'plas' con un valor de 0.467.

2.3 Historiograma, sesgo y densidad

Se elaboraron histogramas para cada variable con el propósito de mostrar la repetición de los datos que hay en ellas. Para la creación de los histogramas, se han importado las librerías matplotlib.pyplot y seaborn. A partir del historiograma podemos detectar valores corruptos como por ejemplo en la variable 'pres', que incluye valores de cero, lo cual es médicamente imposible a menos que el paciente haya fallecido, pero para este estudio no nos interesa. Por otro lado, el sesgo de las variables se calculó utilizando la función def skew( ) que confirmó las observaciones de los historiogramas. Por ejemplo, la variable 'preg' muestra un sesgo hacia la derecha con un valor de 0.902 (Véase Figura 7), lo que se puede corregir con trasformaciones estadísticas.

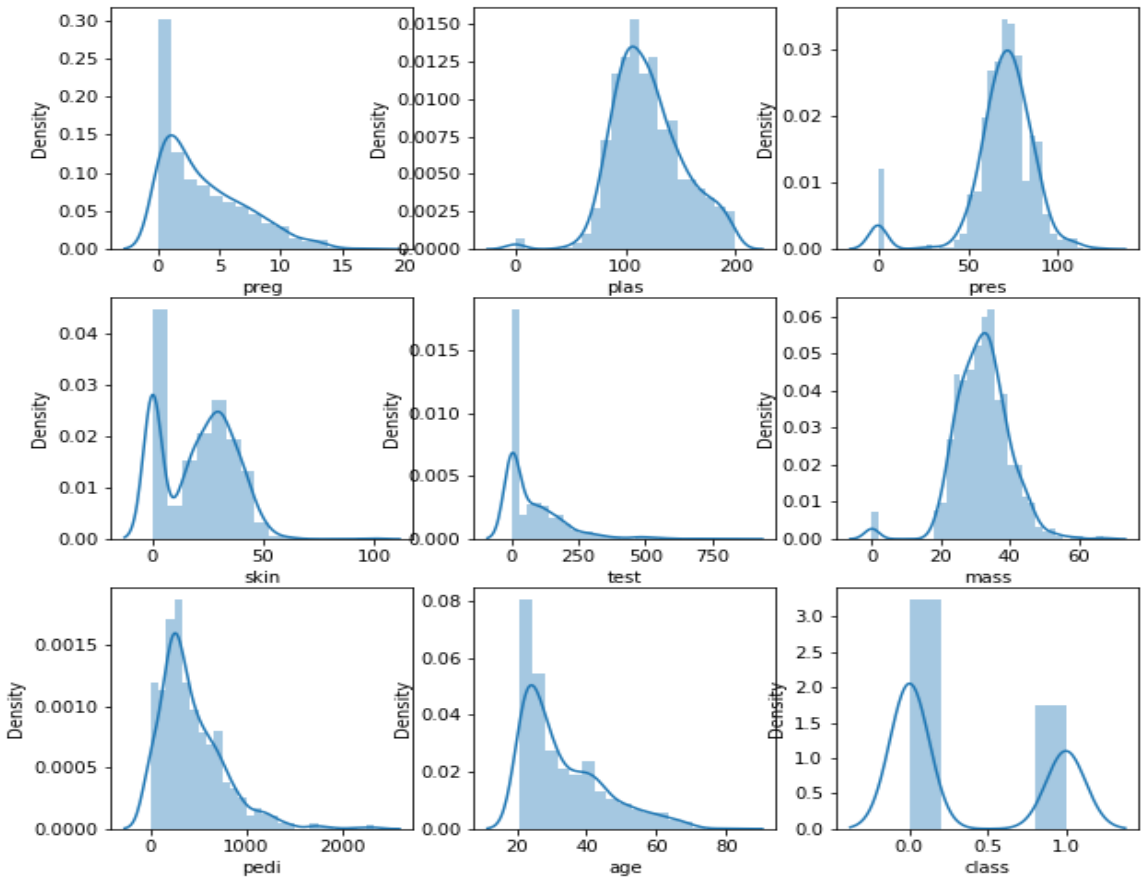

Figura 6: Historiograma y densidad.

```
preg     0.902
plas     0.174
pres    -1.844
skin     0.109
test     2.272
mass    -0.429
pedi     1.562
age      1.130
class    0.635
```

Figura 7: Sesgo de las variables.

En la visualización univariable de los datos adjuntada en la Figura 6 se obtiene bastante información acerca de la distribución de densidad que siguen las variables, se adjunta además evidencia propia de estas distribuciones en la Figura 8. Por ejemplo, la variable 'age' parece seguir una distribución exponencial, en cambio 'mass' se asemeja más a una campana de gauss, lo cual es importante pues muchos de los algoritmos de *machine learning* asumen que los datos siguen una distribución normal, aunque siempre se pueden corregir con escalamiento o transformaciones que se verán más adelante.

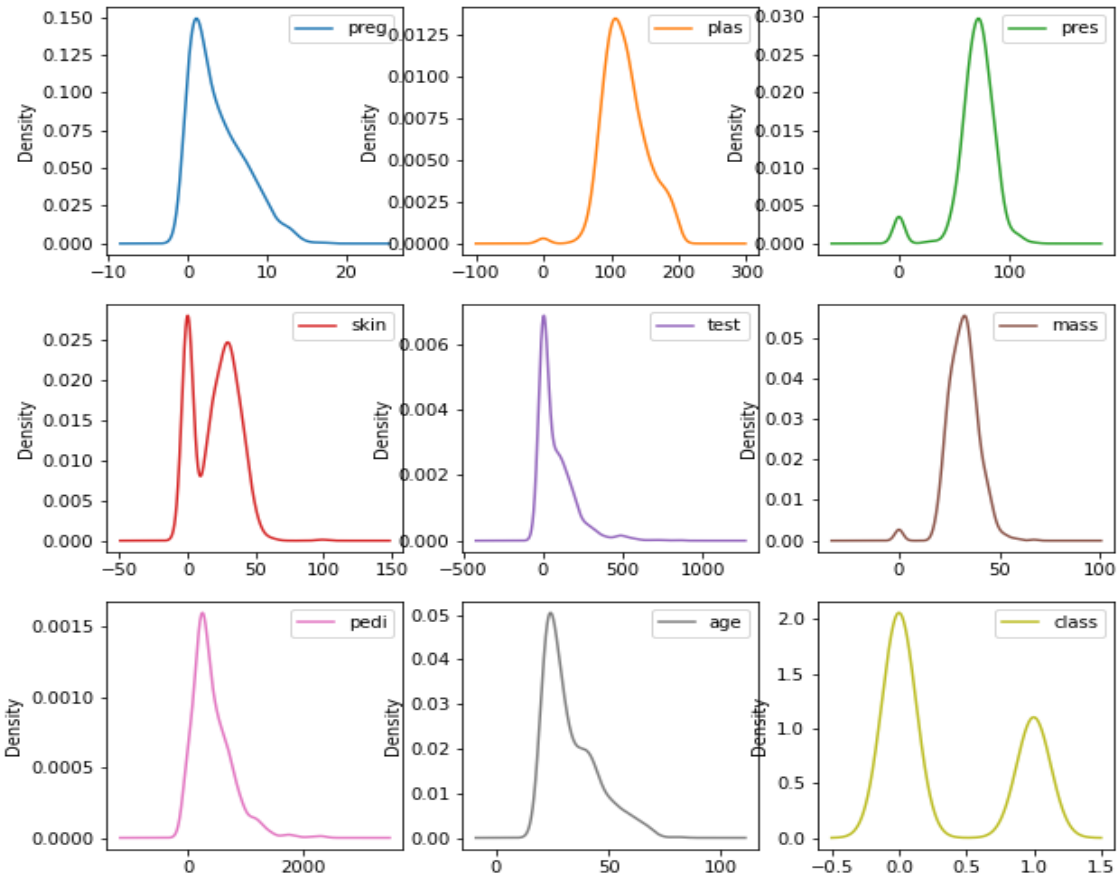

Figura 8: Gráficos de densidad.

2.4 Boxplot

Para la detección temprana de valores atípicos y una visualización de los datos se aplicó diagramas de cajas (*boxplot*) obteniendo un resumen de los datos en cinco medidas descriptivas, además de intuir su morfología y simetría.

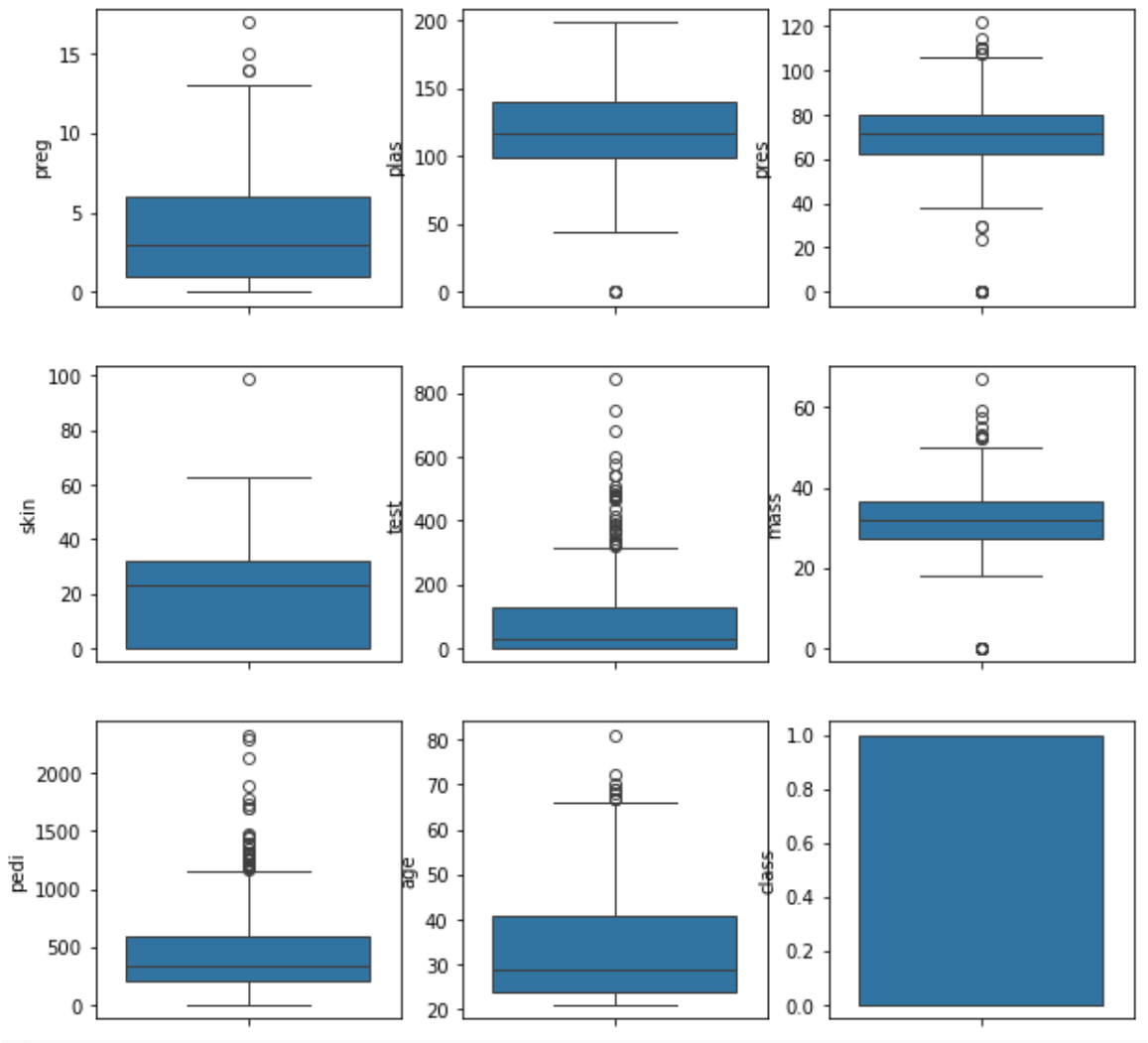

Figura 9: Boxplot.

En concreto, se observa que 'pres' presenta un valor atípico, puesto que toma el valor cero, incosistente en este contexto médico. Este análisis confirmó la necesidad de eliminar o transformar ciertas observaciones antes de la modelización.

2.5 Dispersión

En esta sección se va a comprobar visualmente si los atributos tienen una tendencia lineal, de clustering o correlación en la interacción entre ellos. Para ello, se contruyen gráficos de dispersión entre las diferentes variables involucradas incluyendo la variable respuesta.

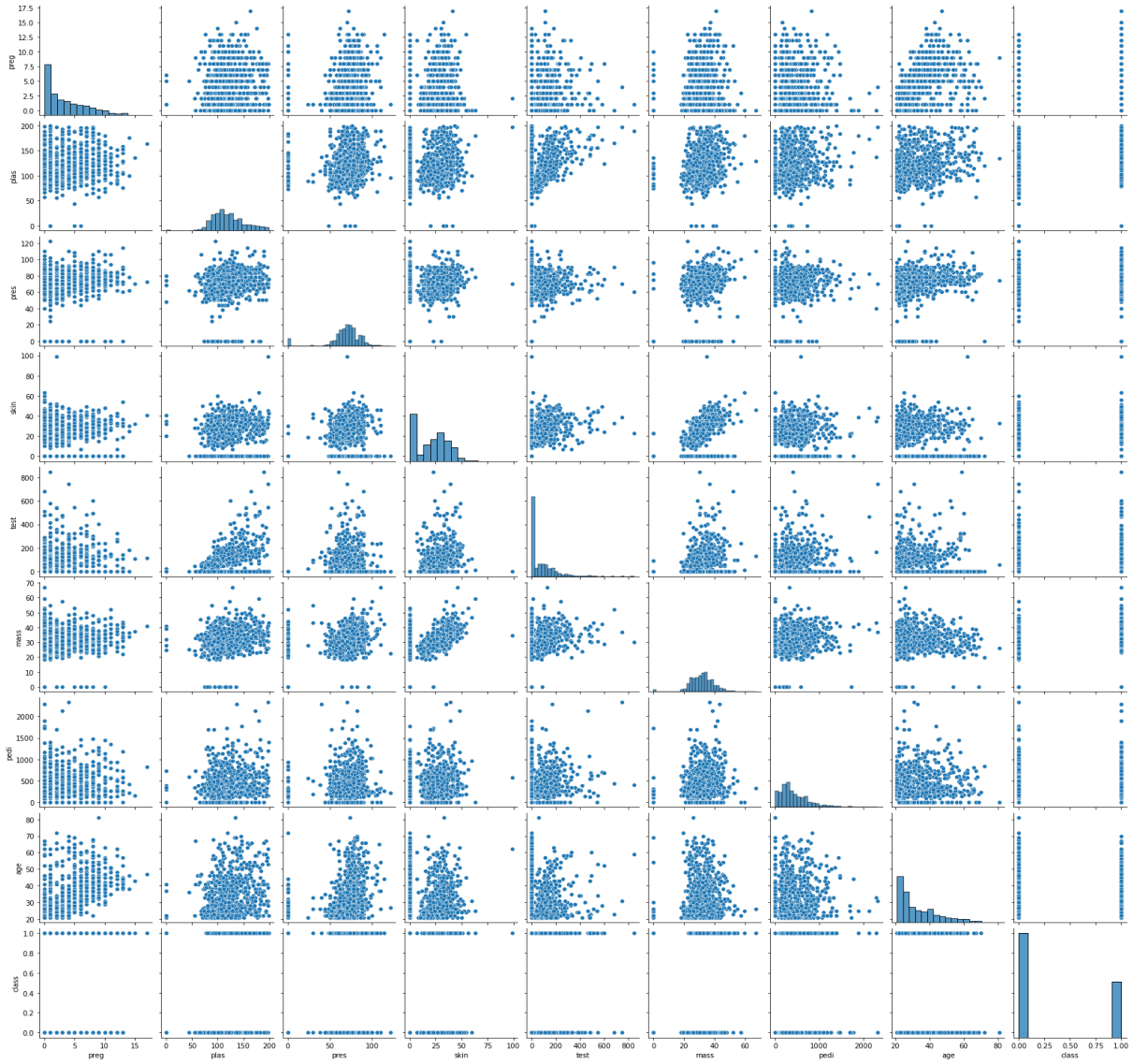

Figura 10: Gráficos de dispersión

Observando la figura 10, en la mayoría de los casos ninguna variable muestra tener una interacción lineal con respecto a otra, excepto 'plas' con 'test' o 'mass' y 'skin' aunque no queda del todo claro, lo que podría requerir técnicas adicionales de procesamiento o selección de carcaterísticas.

3 PROCESAMIENTO DE LOS DATOS

En esta sección se trasformarán y/o modificarán los datos en caso de que sea necesario para garantizar que los algoritmos de predicción funcionen de manera adecuada. Este proceso incluye la detección de valores incosistentes, el tratamiento de sesgos y la preparacion de los datos para la modelización.

3.1 Eliminación de valores incosistentes

Gracias a los boxplots generados en la etapa anterior (Figura 9), se detectan algunos valores corruptos en la base de datos. Por ejemplo, para la variable 'pres' referida a la presión toma valores cero, los cuales son incosistentes, pues indicaría la muerte del paciente. Esta observaciones se eliminaron utilizando el siguiente comando data.drop(indices1), siendo $indices1 = data[data["pres"] <= 0].index$. De manera similar, se eliminan los registros con valores de masa corporal iguales o inferiores a cero, ya que no tiene sentido practico en este contexto.

3.2 Corrección del sesgo

Gracias a la Figura 8 se puede observar que algunas variables como 'pres' y 'age' mostraron un sesgo importante en su distribución. En este caso como los datos son todos positivos podemos aplicar method = 'box-cox'. Como se observa en las Figuras 11 y 12 la transformación de Box-cox, permitió que la distribución de estas variables se asemeja más a una distribución normal.

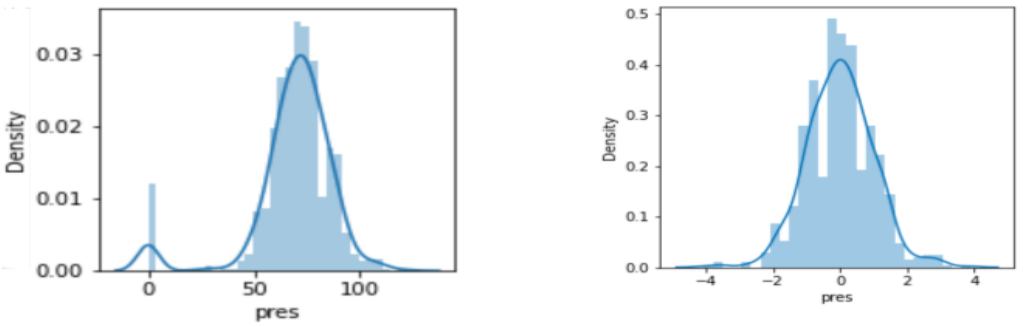

Figura 11: Transformación variable pres.

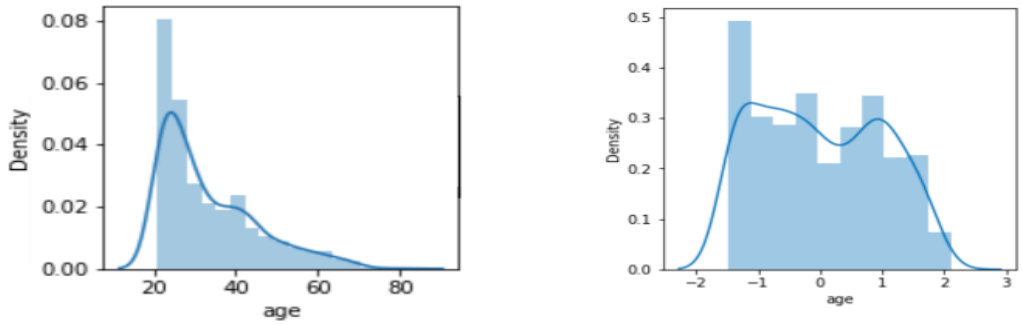

Figura 12: Trasformación variable age.

En cambio, las varaibles ‘pedi’,‘test’ también muestran sesgo, sin embargo, en este caso presentan valores negativos. Para su transformación utilizaremos method = ‘Yeo-Johnson’ de la librería PowerTransformer.

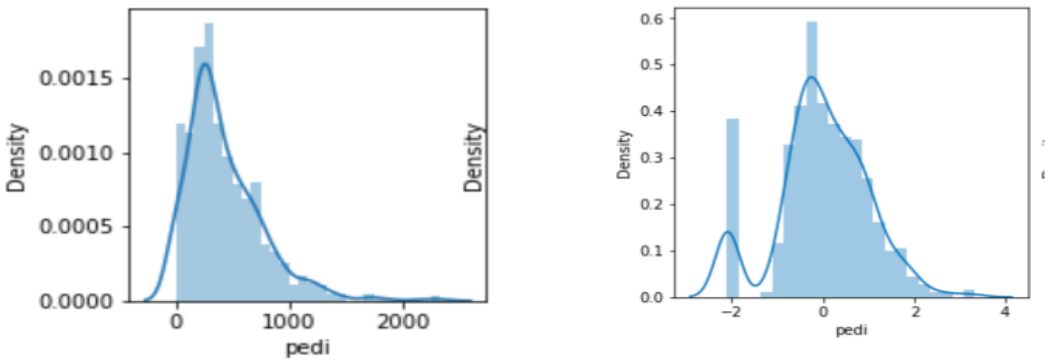

Figura 13: Trasformación variable pedi.

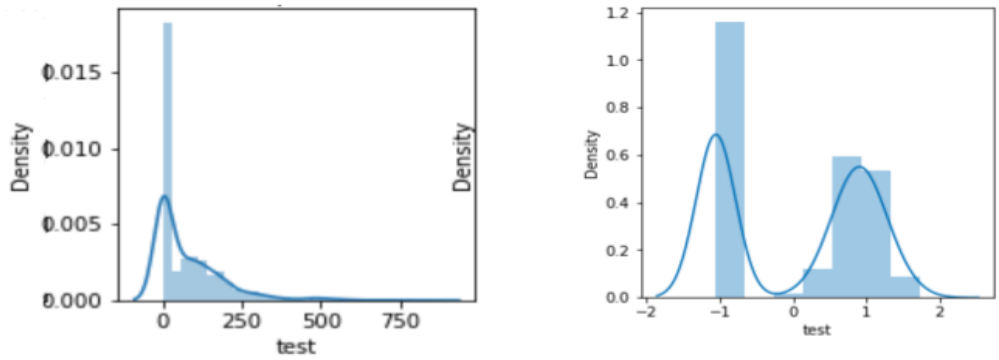

Figura 14: Transformación variable test.

Como se observa en las Figuras 11, 12, 13 y 14 tras la transformación de Box-cox y Yeo-Johnson, la distribución de estas variables se asemeja más a una distribución normal.

3.3 Preparación de conjuntos de datos

A partir de los datos trasformados, se obtienen 3 conjuntos de datos distintos tras aplicar una estandarización, normalización y reducción de dimensiones con PCA.

- **Estandarización**: los datos fueron escalados para tener una media de 0 y una desviación estándar de 1.
- **Normalización**: los valores se ajustaron al rango [0,1].
- **Reducción de dimensiones con PCA**: se aplicó analisis de componentes principales para reducir el número de variables explicativas, manteniendo solo las 5 más relevantes. Esto se aplicó como un experimento adicional, aunque el conjunto de datos original no era particularmente grande.

El resultado final se muestra en las Figuras 15, 16 y 17 respectivamente.

	pres	age	test	...	skin	mass	class
0	-0.020729	1.355278	-1.056490	...	0.860048	0.164241	1.0
1	-0.507694	0.115478	-1.056490	...	0.477824	-0.853145	0.0
2	-0.671017	0.218686	-1.056490	...	-1.369593	-1.332770	1.0
3	-0.507694	-1.478394	0.790066	...	0.095600	-0.635134	0.0
4	-2.678735	0.315457	1.032973	...	0.860048	1.544980	1.0
..	...	...	...	...	...	...	...
724	0.301565	1.763557	1.062044	...	1.688200	0.062503	0.0
725	-0.182566	-0.376344	-1.056490	...	0.350416	0.629332	0.0
726	-0.020729	0.005176	0.863009	...	0.095600	-0.911282	0.0
727	-0.999268	1.227906	-1.056490	...	-1.369593	-0.344452	1.0
728	-0.182566	-1.045204	-1.056490	...	0.605232	-0.300850	0.0

[729 rows x 9 columns]

Figura 15: Conjunto de datos tras la estandarización.

	pres	age	test	...	skin	mass	class
0	-0.000133	0.008695	-0.006778	...	0.224537	0.215556	1.0
1	-0.005419	0.001233	-0.011278	...	0.309564	0.283945	0.0
2	-0.003634	0.001184	-0.005721	...	0.000000	0.126180	1.0
3	-0.005280	-0.015376	0.008217	...	0.239218	0.292262	0.0
4	-0.018113	0.002133	0.006985	...	0.236666	0.291438	1.0
..	...	...	...	...	...	...	...
724	0.002577	0.015071	0.009076	...	0.410209	0.281164	0.0
725	-0.001401	-0.002888	-0.008108	...	0.207221	0.282435	0.0
726	-0.000164	0.000041	0.006848	...	0.182503	0.207895	0.0
727	-0.007713	0.009477	-0.008154	...	0.000000	0.232319	1.0
728	-0.001778	-0.010182	-0.010292	...	0.301991	0.296146	0.0

[729 rows x 9 columns]

Figura 16: Conjunto de datos tras la normalización.

	P1	P2	P3	P4	P5	class
0	-27.514727	12.069997	-2.507616	2.353980	0.125691	1
1	36.011450	7.932084	-5.824932	-1.791159	0.625136	0
2	-60.597979	-25.891594	-7.150458	2.778990	-0.949397	1
3	32.187472	2.229763	-3.425536	-2.442488	0.173094	0
4	-16.905573	14.675568	6.844058	-4.132332	-3.665901	1
..	...	...	...	...	...	...
724	18.849042	26.847006	-3.728850	7.445738	0.603422	0
725	-1.331212	6.179075	3.072518	-1.815853	2.208098	0
726	0.273251	0.245939	-6.409390	1.346375	0.079052	0
727	-3.951277	-21.718994	1.592015	-3.266733	0.013977	1
728	27.774377	10.260818	-2.830056	-2.166587	0.659469	0

[729 rows x 6 columns]

Figura 17: Conjunto de datos tras la reducción de dimensiones con PCA.

4 MODELIZACIÓN

En esta sección, el enfoque principal es encontrar el algoritmo de *machine learning* que mejor prediga la salida para el problema en estudio. Para ello, se evaluaron múltiples algoritmos aplicados a diferentes versiones del conjunto de datos y se evaluó su rendimiento utilizando 'accuracy' y 'ROC' como métricas.

4.1 Algoritmos evaluados

Los algoritmos probados para la evaluación fueron los siguientes: DecisionTreeClassifier( ), LogisticRegression( ), KNeighborsClassifier( ), SVC( ), GaussianNB( ), LinearDiscriminantAnalysis( ). Estos algoritmos se aplicaron a las tres versiones del conjunto de datos preparados en la sección anterior.

4.2 Evaluación del rendimiento

Para evaluar el rendimiento de cada modelo, se utilizaron dos métricas 'accuracy' y 'ROC'. Los resultados obtenido se muestran de forma numérica como gráfica para así comprender mejor los resultados (ver Figuras 18 y 19).

```
LoR: 77.37% (1.56%)     cart: 69.28% (4.28%)
LoR: 65.57% (0.27%)     cart: 64.07% (5.34%)
LoR: 76.68% (1.54%)     cart: 65.84% (1.51%)
LDA: 76.68% (1.79%)     NB: 74.63% (2.88%)
LDA: 67.08% (2.55%)     NB: 70.24% (3.86%)
LDA: 76.82% (1.44%)     NB: 76.68% (0.97%)
k-NN: 73.52% (2.57%)    SVM: 77.78% (0.79%)
k-NN: 66.80% (1.96%)    SVM: 65.57% (0.27%)
k-NN: 72.00% (5.36%)    SVM: 76.82% (1.17%)
```

Figura 18: Rendimiento de los modelos con la métrica accuracy.

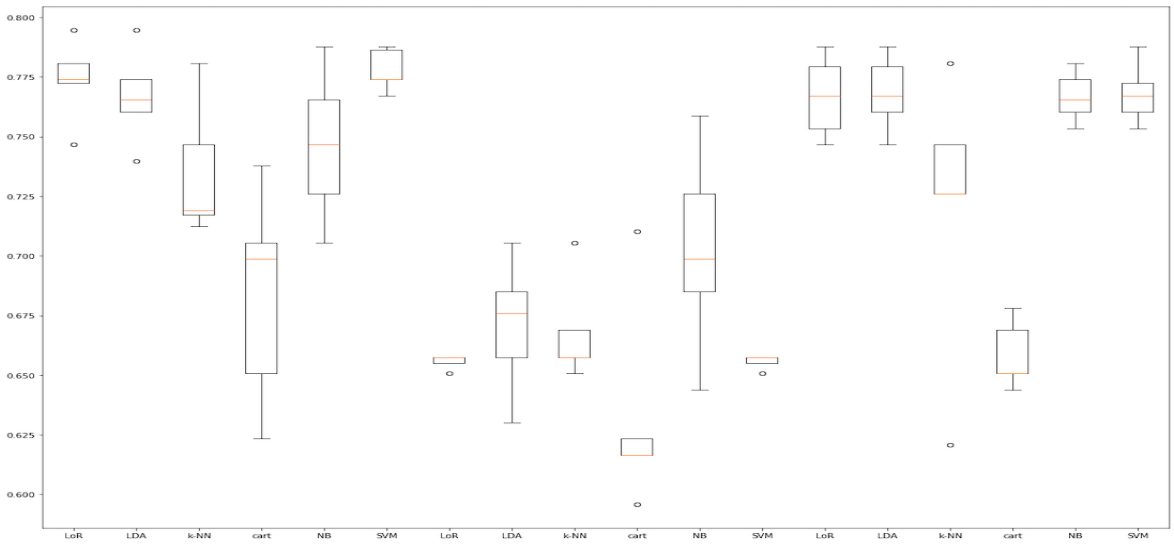

Figura 19: Rendimiento de los diferente de forma gráfica .

```
LoR: 0.84 (0.024)
LoR: 0.63 (0.026)
LoR: 0.83 (0.018)
LDA: 0.84 (0.023)
LDA: 0.72 (0.03)
LDA: 0.83 (0.018)
k-NN: 0.79 (0.024)
k-NN: 0.63 (0.0093)
k-NN: 0.75 (0.039)
cart: 0.65 (0.056)
cart: 0.60 (0.042)
cart: 0.62 (0.011)
NB: 0.83 (0.02)
NB: 0.74 (0.034)
NB: 0.82 (0.018)
SVM: 0.84 (0.024)
SVM: 0.72 (0.03)
SVM: 0.83 (0.017)
```

Figura 20: Rendimiento de los modelos con la métrica accuracy.

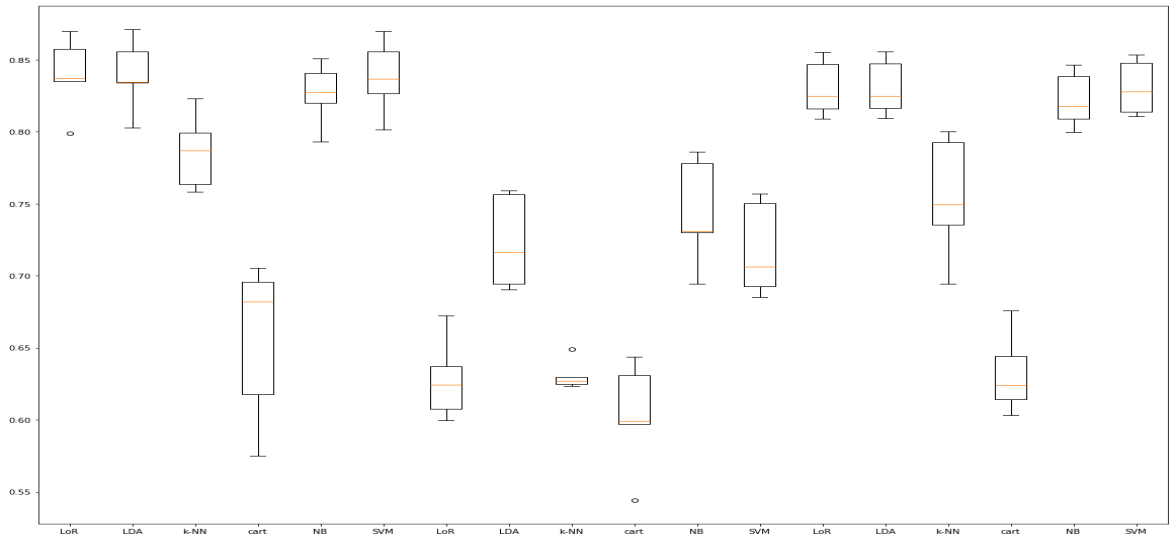

Figura 21: Rendimiento de los diferente de forma gráfica .

4.3 Selección del modelo final

Para seleccionar el modelo más fiable, se elabora primero una tabla (Cuadro 1) donde para cada modelo se elige el conjunto de datos {1,2,3} correspondientes a la estandarización, normalización y PCA que mejor se comporta atentiendo a cada métrica, seleccionando aquel con una media más alta y menor desviación estándar de acuerdo con los resultados mostrados en las Figuras 18 y 20.

MODELO	ACCURACY	ROC
LOR	**1** (77.37 %, 1.56 %)	**1** (0.84, 0.024)
LDA	**3** (67.08 %, 1.44 %)	**1** (0.84, 0.023) ó **3** (0.83, 0.018)
k-NN	**1** (73.52 %, 2.57 %)	**1** (0.79, 0.024)
cart	**3** (65.84 %, 1.51 %)	**1** (0.65, 0.056)
NB	**3** (76.68 %, 0.97 %)	**1** (0.83, 0.02) ó **3** (0.82, 0.018)
SVM	**1** (77.78 %, 0.79 %)	**1** (0.84, 0.024) ó **3** (0.83, 0.017)

Cuadro 1: Elección del mejor modelo.

Aparentemente el conjunto de datos que mejor se comporta con los diferentes algoritmos es aquel con los datos estandarizados, apareciendo un total de 9 veces. De acuerdo con los resultados, véase Cuadro 1, el modelo que obtuvo mejor desempeño usando como métrica el *accuracy* fue SVM aplicado al conjunto de datos estandarizado, ya que es el modelo mostró el accuracy más alto en media (77.78 %), sumado a una baja desviación estándar (0.79 %) indicando así una mayor consistencia en sus predicciones. Por otro lado, los modelos que tuvieron un mejor desempeño usando la métrica *roc* fueron 'LDA' con los datos estandarizados (0.84, 0.023) y 'SVM' con la reducción de dimensiones (0.83, 0.017).

Debido a que la muestra está desbalanceada, se procede a obtener un reporte de clasificación para estos modelos respectivamente, para ello dividimos los conjuntos de datos en datos de entrenamiento y datos de test desde from sklearn.model_selection import train_test_split y $test\_size = 0{,}33$:

	precision	recall	f1-score	support
0.0	0.78	0.84	0.81	166
1.0	0.56	0.47	0.51	75
accuracy			0.72	241
macro avg	0.67	0.65	0.66	241
weighted avg	0.71	0.72	0.71	241

Figura 22: Reporte de clasificación (SVM y estandarización).

	precision	recall	f1-score	support
0.0	0.79	0.85	0.82	166
1.0	0.60	0.49	0.54	75
accuracy			0.74	241
macro avg	0.69	0.67	0.68	241
weighted avg	0.73	0.74	0.73	241

Figura 23: Reporte clasificación (LDA y estandarización)

	precision	recall	f1-score	support
0.0	0.79	0.87	0.83	166
1.0	0.63	0.48	0.55	75
accuracy			0.75	241
macro avg	0.71	0.68	0.69	241
weighted avg	0.74	0.75	0.74	241

Figura 24: Reporte de clasificación (SVM y PCA).

En base a los resultados obtenidos en las diferentes Tablas 22, 23 y 24 el modelo aplicado en la Tabla 24 parece ser el más equilibrado pues tiene el mayor *F1-score* para la clase 1, lo que refleja un balance adecuado entre *pecision* y *recall*. Además tiene la mejor precisión promedio (*weighted avg*) y el mayor *accuracy* (0.71). El modelo de la tercera tabla es competitivo, pero tiene un *F1-score* ligeramente más bajo para la clase 1. En conlusión, el modelo que mejor se adapta al problema el modelo SVM aplicando una reducción de dimensiones.

5 CONCLUSIONES

Para finalizar este trabajo, es importante destacar la relevancia de las técnicas aprendidas en el curso 'Máster de especialista en Ciencia de Datos con Python' de Udemy, los cuales se aplicaron con éxito al problema de claficación planteado. Gracias a estas técnicas, se logran los siguientes objetivos.

- **Desarrollo del modelo predictivo**: se implementó un modelo confiable y robusto capaz de predecir con cierta precisión si un paciente padece diabetes, utilizando datos médicos específicos.
- **Exploración de los datos**: se realizó un analisis exhaustivo de las variables, detectando patrones, relaciones y valores atípicos y correlaciones que afectaban a la calidad de los datos.
- **Transformacion y preparación de los datos**: se corrigieron sesgos y se ajustaron las distribuciones de las variables para optimizar el rendimiento de los algoritmos.
- **Evaluación de los algoritmos**: se compararon múltiples modelos de *machine learning*, seleccionando finalmente el SVM como mejor opción debido a su alto desempeño y consitencia.

5.1 Limitaciones y trabajos fuuros

No obstante, a pesar de haber obtenido resultados stisfactorios, este proyecto presenta algunas limitaciones:

- **Desbalance de clases:** aunque se aplicaron técnicas para solventar el problema del desbalance en las clases, se podrían explorar en trabajos futuros métodos más sofisticados como sobremuestreo(*oversampling*), submuestreo (*undersampling*) o ajuste de pesos en los algoritmos.
- **Conjunto de datos:** en nuestro problema específico, los datos provienen de una población concreta, sin embargo, es recomendable trabajar con conjuntos más diversos y más extensos.

Este proyecto se adentra en el mundo de la ciencia de datos de manera exploratoria y práctica, desarrollando habilidades clave en el análisis y modelado de datos, así como en el uso de herramientas avanzadas de *machine learning*.

Como primer paso hacia proyectos más complejos, se considera este trabajo una experiencia enriquecedora que sienta las bases para futuras investigaciones y aplicaciones en el fascinante campo de la ciencia de datos.

6 BIBLIOGRAFÍA

[1] https://www.who.int/es/news-room/fact-sheets/detail/diabetes#:~:text=En%202021%2C%201a%20diabetes%20fue,por%20causa%20cardiovascular%20(1).

[2] Máster de especialista en Ciencia de Datos con Python. Udemy https://www.udemy.com/course/master-en-ciencia-de-datos-con-python/?couponCode=KEEPLEARNING

[3] https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database/data