Universidad Politécnica de Cataluña

Aperendizage automático

Pima Indians Diabetes

Autora:

Míriam Méndez

Q1 curso 2021/2022

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

FIB

Índice

1. Introducción

En este proyecto vamos a estudiar el comportamiento de la diabetes de la población pima, bueno, concretamente sólo de las mujeres con al menos 21 años con esta herencia, que son los datos que nos proporciona el dataset.

Este conjunto de datos nos ha sido proporcionados por el National Institute of Diabetes and Digestive and Kidney Diseases, con el objetivo de construir un modelo de aprendizaje supervisado para predecir con precisión si el pacientes de este conjunto de datos padecerá diabetes o no, ¿seremos capaces?

Para ello tendremos que tener en cuenta ciertas medidas médicas como: el nivel de insulina, el nivel de glucosa, la presión sanguínea, el IMC, etc. Es decir, antes de empezar, necesitaremos un conocimiento básico de esta información, para poder entenderla y estudiar nuestros resultados, que obtendremos a partir de diferentes métodos de clasificación.

Por esta razón primero empezaremos con una introducción de la diabetes, después haremos una ojeada a los diferentes resultados que se han obtenido con este dateset para predecir la diabetes. Con esta información empezaremos el pre-processing, pero primero haremos una visualización de datos para preparar los datos de manera que sean adecuados para entrenar nuestros modelos. Los modelos que entrenaremos serán tres lineales/-cuadráticos y dos no lineales.

2. La diabetes

La diabetes es una enfermedad que aparece cuando el páncreas no produce suficiente insulina o el organismo no utiliza eficazmente la insulina que produce. La insulina es la hormona que regula el azúcar en la sangre. Esto hace que los pacientes tengan niveles altos de glucosa en sangre.

Existen 4 tipos de diabetes:

- Tipo I (DM1): se presenta con mayor frecuencia en la gente joven. Suele aparecer como consecuencia de la destrucción irreversible de las células productoras de insulina (células β), es decir, la cantidad de insulina es cero. La causa de esta destrucción es el proceso de autoinmunidad. Se trata de un error en el organismo que hace que el mecanismo de immunidad, que normalmente sirve para defender el cuerpo delante elementos extraños, provoque la autodestrucción de las células β del páncreas.

- Tipo II (DM2): es el tipo más frecuente, aproximadamente el 90 % de todos los casos. Se presenta en mayores de 40 años aunque cada vez hay más jóvenes que también la presentan, a causa de la obesidad. En la diabetes tipo II las alteraciones de glucemia tiene doble causa:
  - Existencia de un déficit de producción de la insulina por una disminución orgánica del numero de células β. Defecto parecido a la diabetes tipo I, pero con 2 diferencias importantes:
    - No se trata de una destrucción masiva de células sino de una disminución moderada, es decir, la cantidad de insulina es insuficiente en relación con los niveles altos de de glucemia.
    - No intervienen factores relacionados con la autoinmunidad.
  - Resistencia a la insulina. Esto es una situación en que la capacidad de actividades hormonales de la insulina queda devaluada: una determinada dosis d'insulina no consigue disminuir la glucemia con la misma intensidad con la que lo haria normalmente. Por tanto, la insulina pierde parte de su eficacia.
- Gestacional (DMG): se incluyen todos los tipos de diabetes iniciados o reconocidos por primera vez en el transcurso del embarazo. Se presenta generalmente en mujeres mayores. En la mayoría de casos se trata de una perturbación metabólica transitoria que desaparece luego del parto, pero tiene un cierto riesgo de padecer diabetes tipos II en un futuro. Pero el autentico riesgo es para el feto, que aunque este totalmente sano , ya que la anormalidad metabólica no es de el sino de la madre, puede provocar que el cuerpo del feto se haga anormalmente grande, dando lugar un aumento de riesgo durante el parto.
- Específica rara: Representa todos los tipos de diabetes que no han sido clasificada en los tres tipos previos.

3. Trabajos relacionados

Existen varios trabajos realizados con Pima Indians Diabetes Databese, que usan diferentes estrategias, en la misma página de Kaggle, ya podemos encontrar unos cuantos: **Shruti_Iyyer** [6] el más votado en Kaggle, actualmente, logró una precisión del 77,21 % mediante la combinación de *cross validation - 5 folds* y *KNN*. **Vincent Lugat** [7] consiguió una precisión del 87,3 % aplicando *LightGM* y *KNN* con un cros validation de 5 folds, también.

Por otra parte, tambien encontramos publicaciones que informan sobre sus métodos, como por ejemplo **Choubey** [8] que propone una cmobinación de *Genetic Algorithm* (GA) y *Multilayer Perceptron Neural Network* (MLP NN) para la tarea de clasificación. Este modelo obtuvo un 79,1 % de precisión. **Kumari** [9] aplica *SVM* e investiga varios kernels en su experimento, que alcanzó una precisión del 75,50 % usando *RBF kernel* y el método de *cross validation* para ajustar los híperparametros.

4. Exploración de datos

El dataset tiene un total de 768 instancias con 9 variables, todas numéricas menos el target que es binaria. Antes de empezar con el pre-processing, tenemos que tener en cuenta las siguientes observaciones:

- **Dataset desbalanceado**, solo hay un 34.9 % de pacientes que tienen diabetes, que corresponden a 268 de 768. Por tanto, si queremos realizar de forma correcta la clasificación, el número de datos ha de ser similar para cada clase. De no ser así los algoritmos tenderán a predecir que el paciente tendrá diabetes, ya que el modelo tendrá mas dificultades para identificar que caracteriza el paciente que no tiene diabetes.

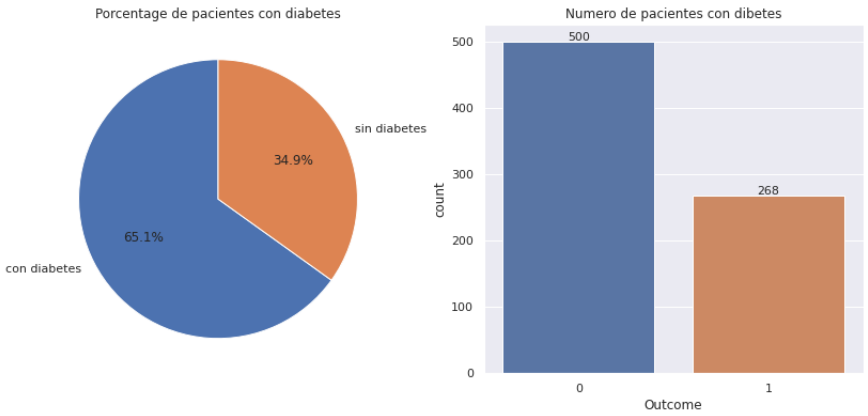


Figura 1: Distribución del target

- **Las variables no siguen una distribución normal**, como se aprecia en la figura (2) podemos comprobar con la estimación de densidad de kernel, que las variables que no siguen la campana de Gauss. No obstante, en muchos casos la media (rosa) y la mediana (marrón) son bastante similares y algunas se encuentran en el medio de los valores.

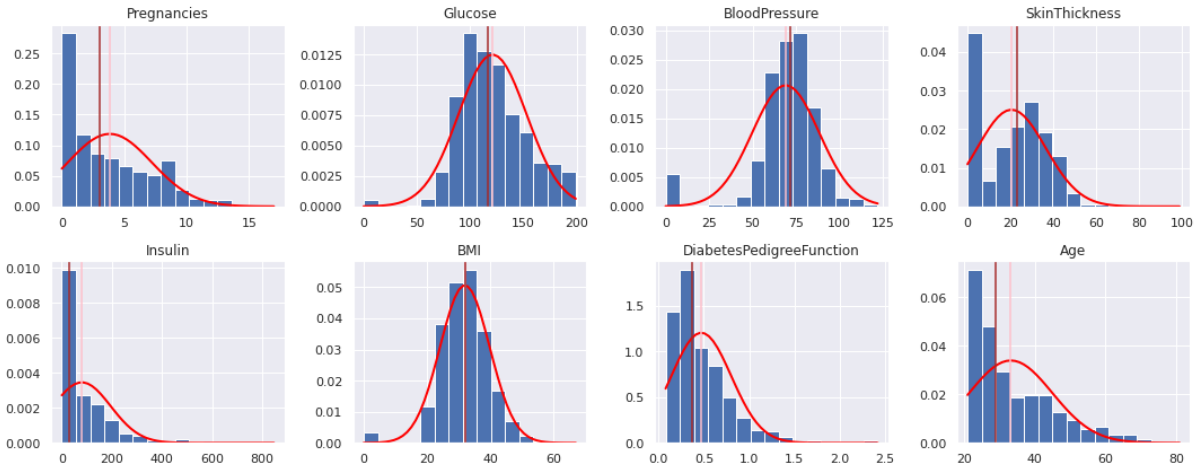


Figura 2: Distribución variables numéricas

- **Valores incoherentes**, ya en los histogramas nos hemos podido dar cuenta de estos valores, pero vamos a interpretarlos con la tabla (1), para justificar su incoherencia.

  - **Pregnancies:** número de embarazos que ha tenido, este dato no lo vamos a catalogar de incoherente. Aunque el máximo es muy exagerado, pero podría ser cierto.
  - **Glucose** [3]**:** concentración de glucosa en plasma, realizada con el examen de tolerancia oral a la glucosa de dos horas. Esta prueba es usada para la detección de diabetes tipo II y la gestacional. Los resultados según la referencia son clasificados del siguiente modo:
    - inferior a 140 mg/dl, nivel de glucosa normal
    - valor entre 140 y 199 mg/dl, trastorno de tolerancia a la glucosa o prediabetes
    - superior a 200 mg/dl, presencia de diabetes

    Pero en ningún caso la glucosa en sangre puede valer 0, ya que las células del cerbero se morirían. De hecho si el nivel ya se encuentra por debajo de 54 mg/dl requiere de acción inmediata.
  - **Blood Pressure:** [4] presión sanguínea en sangre (mm Hg), tampoco puede valer 0, ya que sino la paciente estaría muerta. De hecho la mayoría de personas que padecen diabetes II suelen tener una presión alta (mayor a 80 mm Hg). Aunque si esta embarazada es probable que la presión arterial baje, ya que el sistema circulatorio se expande rápidamente durante el embarazo, pero una vez ha dado a luz, vuelve a su nivel anterior.
  - **Skin Thickness:** [5] grosor del pliegue de la piel del tríceps (mm). El grosor da información sobre las reservas de grasa del cuerpo. Este valor tampoco puede ser 0, ya que entonces estaríamos diciendo que la paciente no tiene piel.

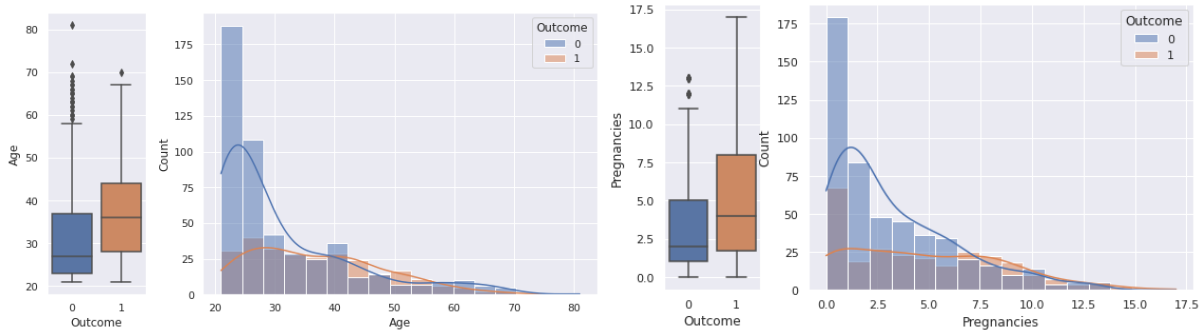
- **Insulin:** insulina sérica de 2 horas (mu U/ml). Puede valer cero pero solo si la paciente padece diabetes tipo I, como hemos visto en el apartado 2.
- **BMI:** índice de masa corporal (peso (kg)) /(altura (m))^2. Indica si la paciente tiene sobrepeso o es obesa. Este valor no puede ser cero, ya que por aritmética implica pesar 0kg. El máximo parece muy exagerado pero es posible
- **Diabetes Pedigree Function:** historial de diabetes en familiares y la relación genética de los familiares con el paciente. Relacionado con la diabetes Tipo I que tiene un componente genético importante.
- **Age:** Edad (años), es correcta se corresponde con la precondición del dataset: mujeres mayores de 21 años

Observación, como es un dataset realizado sólo a mujeres mayores de 21 años, hubiera sido interesante saber si la paciente esta embarazada o no, haciéndoles un test de embarazo. Creo que esta variable nos aportaría información sobre si la paciente tiene diabetes o no, pero para afirmarlo tendríamos que observar su influencia con el target.

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
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	79.799	31.993	0.472	33.241
std	3.370	31.973	19.356	15.952	115.244	7.884	0.331	11.760
min	0.000	0.000	0.000	0.000	0.000	0.000	0.078	21.000
25 %	1.000	99.000	62.000	0.000	0.000	27.300	0.244	24.000
50 %	3.000	117.000	72.000	23.000	30.500	32.000	0.372	29.000
75 %	6.000	140.250	80.000	32.000	127.250	36.600	0.626	41.000
max	17.000	199.000	122.000	99.000	846.000	67.100	2.420	81.000

Cuadro 1: Resumen estadístico de las variables numéricas

- **Relación variables y target** Analizando la relación de cada variable con el target, podemos empezar a extraer ideas sobre qué variables están más relacionadas con la diabetes y de que forma. Al comparar las distribuciones con el target, observamos que hay notables diferencias en la distribución. En particular de Pregnancies y Edad.
  Si observamos el gráfico de la edad, en las personas no diabéticas hay una disminución considerable en función de la edad. En cambio en las personas no diabéticas la disminución no se aprecia mucho. Podría existir la tendencia de que, a medida que las personas envejecen, es probable que se vuelvan diabéticas.
  En el caso de los embarazos, nos encontramos también con una distribución muy similar en el target, así podemos replicar la misma hipótesis, contra mayor sea el número de embarazos, más probabilidad tiene de ser diabética. Pero igual que en edad, la diabetes en si misma, no parece que tenga ninguna influencia ni en el número de embarazos ni en la edad.



(a) Boxplot y histograma de edad (b) Boxplot y histograma de embarazos

Figura 3: Relación con la variable respuesta diabtes (Outcome)

4.1. Pre-processing

Missing values: Al ejecutar la función de eliminar NaN's, no se ha visto ningún cambio. Porque están escondidos bajo el valor cero, por este motivo habíamos visto tantos ceros cuando interpretamos nuestras variables.

A causa de no saber con seguridad si el valor es realmente un cero o un NaN, generará imprecisión en nuestros valores. Nosotros para determinar si ese valor es missing o no, lo haremos a partir de las justificaciones hechas en valores incoherentes. [1]

El siguiente gráfico representa el número de missing values totales del dataset. [2]

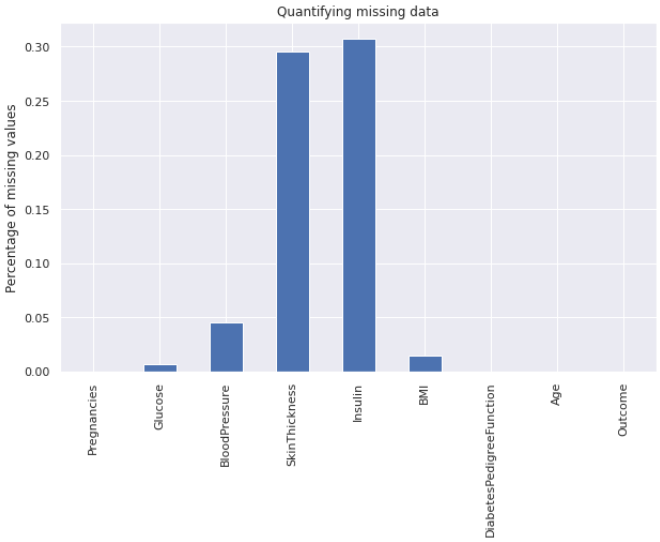


Figura 4: Missing values

[1] Los valores cero de Glucose, BloodPressure, SkinThickness se han marcado como NaN y los de Insuline solo si no tenía diabetes, dado que en la diabetes tipo I la cantidad de insulina es nula

[2] Marcando los ceros de Insuline como NaN, marginando la población con diabetes tipo I, la cantidad de missings hubiera sido un 48.7 %, en vez del 30.7 %. En otras palabras, no podemos saber con precisión si este 18 % del conjunto de datos son valores nulos o ceros.

Eliminar todos aquellas filas que contengan elementos faltantes nos supondría perder aproximadamente la mitad de la información, y eliminar las variables con mayor número de missings, SkinThickness e Insuline, perder dos variables importantes para determinar si es diabética o no la paciente. Pero por otra parte en la imputación de variables, corremos un riesgo muy alto de que el valor impuesto sea erróneo.

En vista de estas tres opciones, he optado por eliminar solo los datos con más de dos missing values y imputar el resto. Las estrategias de imputación que he usado han sido knn, media y mediana.

- Glucose: Aplicamos la imputación con la media a 3 missings.
- BloodPressure: Aplicamos la imputación con la media a 1 missing.
- BMI: Aplicamos la imputación con la mediana a 1 missing.
- Insulin: Aplicamos la imputación con knn a 93 missings.
- SkinThickness: Aplicamos la imputación con knn a 72 missings.

Para elegir la técnica me he basado en boxplots (5) y distribution plots (6), si la variable estaba sesgada, entonces elegía mediana y si seguía una distribución casi simétrica, la media. Pero en el caso de que el número de missings fuera elevado, se ha usado el 1knn: para cada missing value en Insulin y SkinThickness, busacremos el valor más similar y sustituiremos el missing value por este valor.

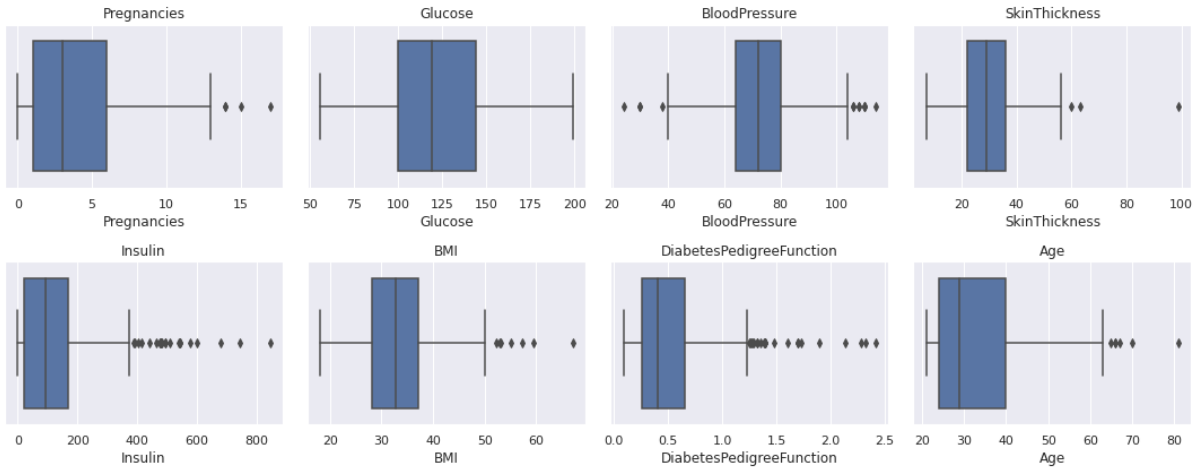


Figura 5: Boxplots

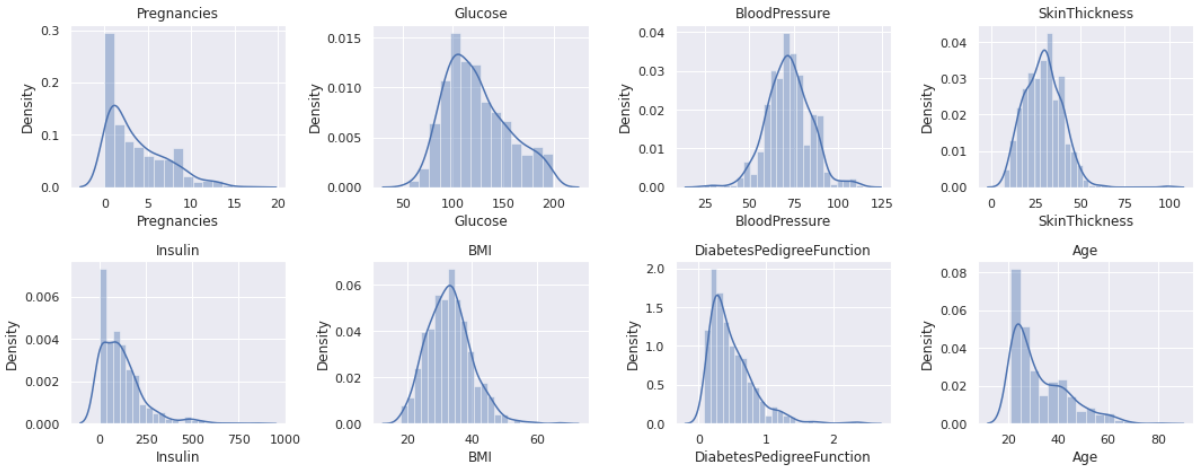


Figura 6: Distplots

El resultado final ha sido un dataset de 609 datos sin ningún valor nulo . Además este tratamiento nos ha favorecido, ya que ahora el target se encuentra más equilibrado (58.6 % con diabetes y 41.4 % sin diabetes).

Outliers: Luego de haber realizado el tratamiento de missing values, seguimos teniendo variables con outliers. Las más destacadas son Insuline y DaibetesPredigneeFunction que estan sesgadas a la derecha. Aunque todas tienen outliers, a excepción de la Glucosa.

Se ha eliminado el skinThckness con valor 99, ya que es un caso muy atípico, que creo que ha surgido de un error de codificación en la entrada, porque no es normal que una persona con un IMC de 34.7 (obesidad tipo I), tenga un pliegue de 99mm, mientras que uno de obesidad mayor, el pliegue sea mucho menor.

Los otros casos los he dejado, ya que en nuestro contexto de análisis estos valores podrían ser ciertos y por tanto aunque puedan distorsionar el comportamiento de los contrastes estadísticos, he preferido no eliminarlos.

Incoherent values: han sido tratados durante el proceso de missing values.

Codification of variables: sólo tenemos una variable binaria, que ya venía representada como numérica. Por tanto no se ha realizado one-hot-encode, ya que al final, el resultado de la variable tiene que terminar siendo numérico, para poder entrenar el modelo.

Feature elimination: He estado apunto de eliminar la edad, ya que en un principio había pensado que era independiente de si la paciente pudiera padecer diabetes o no, ya que la diabetes la padecen personas jóvenes, adultas y embarazadas independientemente de su edad, por tanto, sería una información extra que no ayudaría a predecir si tiene o no diabetes. No obstante al final he decidido dejarla, porque es cierto que contra más mayor se es, existe un mayor riesgo en padecer la diabetes II.

Normalizaton: Cada una de las variables tienen diferentes rangos, como se puede ver en la figura (7), tenemos valores más grandes que otros y una varianza muy distinta entre las variables.

Esto implicara normalizar y estandarizar los datos para poder entrenar determinados modelos.

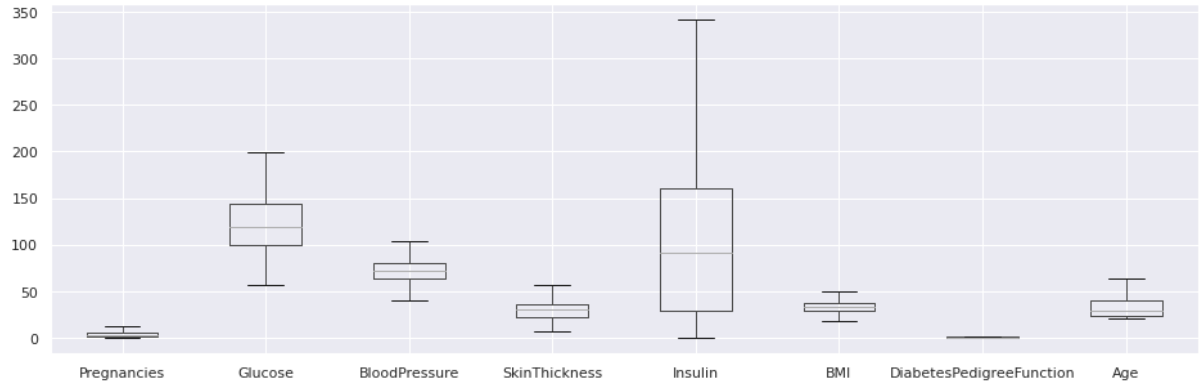


Figura 7: Boxplots

Feature transformation: Es evidente que existe asimetría, sobretodo en insulina, pero no la he corregido, porque si la corrigiera estaría falsificando información de nuestro contexto.

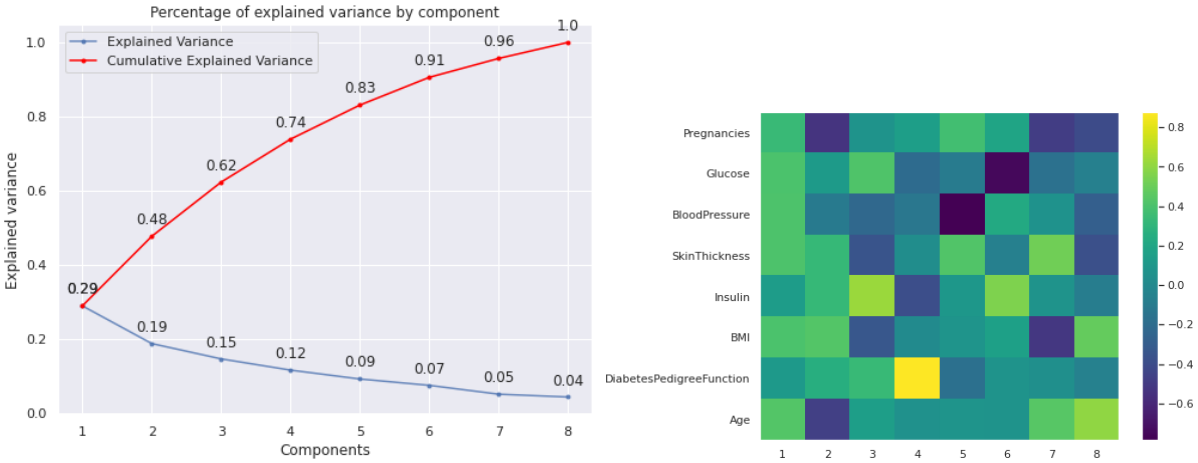
4.2. Visualización

Para hacer una mejor visualización de nuestros datos he usado el PCA.

El PCA al estar basado en la matriz de covarianza, hemos necesitado estandarizar nuestros datos previamente. Esto hará que los valores de las variables cambien a una media de 0 y una desviación estándar de 1.

En esta visualización (Figura 8) se puede observar que con 6 componentes se pude explicar el 91 % de la varianza de nuestros datos. Es decir, podríamos reducir nuestras 8 dimensiones a 6. Este método es útil cuando se quiere reducir la dimensionalidad.

Además una vez que se han calculado las componentes principales, podemos analizar la influencia de las variables en cada componente con el gráfico de heatmap. Por ejemplo, en la sexta componente recoge mayoritariamente información de Glucose.



(a) Porcentaje de varianza explicada acumulada (b) Heatmap de los componentes

Figura 8: PCA

Esta misma influencia se puede visualizar en el scatter plot, donde se han visualizado las 2 primeras componentes.

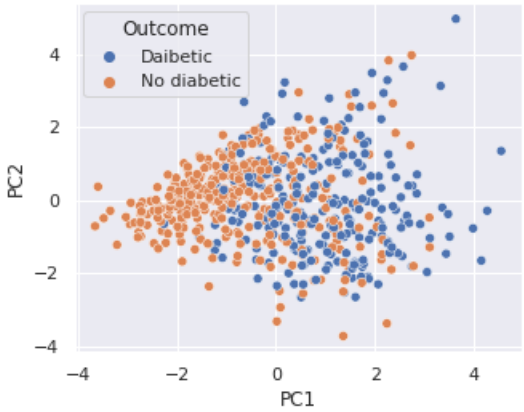


Figura 9: Scatter plot de PC1 y PC2

5. Modeling

5.1. Resampling protocol

La partición del dataset realizada ha sido 1/3 (203 instancias) para el test y 2/3 (406 instancias) para el entrenamiento. A este último conjunto se le ha aplicado *5-fold cross validation* para seleccionar los hiperparámetros óptimos.

El procedimiento para ajustar los hiperparámetros se ha efectuado con *RandomizedSearch*, menos en KNN que se ha usado *GridSearch*, ya que el espacio de los hiper-

parámetros no era demasiado grande. La diferencia entre estos dos es que *GridSearchCV* considera exhaustivamente todas las combinaciones de parámetros, mientras que *RandomizedSearchCV* funciona probando un número determinado de parámetros dados que pueden ser listas o distribuciones. Estas distribuciones nos pueden ser muy útiles, por ejemplo cuando se calcula el parámetro de regularización, el cual es usado para contrarrestar cualquier sesgo en la muestra y ayudar a que el modelo se generalice de forma correcta, evitando el sobreajustar el modelo a los datos de entrenamiento.

En esta búsqueda se ha intentado ajustar el número máximo de hiperparámetros y dejar el mínimo posible por defecto por motivos de experimentación. De todas formas, en los casos donde otros hiperparámetros acumulan muchas posibles combinaciones se ha optado por reducir la granularidad.

5.2. Modelos

Hay varios algoritmos que podemos usar para entrenar el modelo, no obstante en este proyecto se han usado tres métodos lineales/cuadráticos y dos no lineales de clasificación.

Para los modelos lineales/cuadráticos he elegido los siguientes tres:

- **Logisitc regression:** Es un análisis adecuado para llevar a cabo cuando la variable target es dicotómica. Los hiperparámetros óptimos que se han encontrado son: una fuerza de regularización moderada baja con una penalización de tipo l1, una optimización basada en el problema primal (no usa problema dual) y el solver saga.
- **K-nearest-neighbours:** Es un método no-paramétrico adecuado para problemas de clasificación. Los hiperparámetros óptimos que se han encontrado son: 23 vecinos, el algoritmo de vecinos más cercanos, asignación de pesos proporcionales a la inversa de la distancia desde el punto de consulta.
- **Linear SVM:** Consiste en un SVM con un kernel lineal, y por tanto, es adecuado para problemas de clasificación con un target binario. Los hiperparámetros óptimos que se han encontrado son: una fuerza de regularización baja con una penalización tipo l1, una optimización basada en el problema primal (no usa problema dual) y una función de penalización squared_hinge.

Para los modelos no lineales he elegido estos dos:

- **SVM with RBF kernel:** SVM (es un modelo adecuado para problemas de clasificación binaria) con un kernel RBF (un kernel no lineal muy común para SVM).

Los hiperparámetros óptimos encontrados han sido: una penalización squred l2 con una fuerza de regularización moderada y un parámetro gamma (para el kernel RBF de $\frac{1}{\#atributos}$).

- **Random Forest:** Es un modelo no paramétrico adecuado para la clasificación flexible y fácil de usar. Además, tiene en cuenta que los árboles de decisión individuales tienden a sobre ajustarse a los datos de entrenamiento, de modo que utiliza el promedio para mejorar la precisión predictiva y controlar el sobreajuste. Los hiperparámetros óptimos encontrados han sido: 889 árboles, profundidad máxima 80, un mínimo de 10 muestras para dividir el nodo con un mínimo de 2 muestras para las hojas, número de predicadores a la raíz cuadrada por cada división, control del tamaño de las sub_muestras con max_samples (bootstrap = True).

5.3. Modelo elegido

En la evaluación del modelo nos hemos basado en diferentes métricas, las primeras que se ejecutaron fueron: train accuracy, cross validation accuracy y test accuracy para tener una idea de la proporción que predice correctamente el modelo y comprobar que no haya ninguna anomalía, por ejemplo que los valores obtenidos no sean train accuracy $\geq$ CV accuracy $\geq$ test accuracy o train accuraccy sea mucho más alta que test accuracy (posible sobreajuste).

Modelos	Train accuracy	5-Fold CV	Test accuracy
Logistic regression	0.813	0.813	0.798
k-nearest-neighbours	1	0.788	0.773
Linear SVM	0.815	0.813	0.798
SVM with RBF kernel	0.828	0.795	0.788
Random Forest	0.933	0.835	0.813

Cuadro 2: CV, Train y Test accuracy de los modelos

En la Tabla (2) el valor que nos interesa más es el **test accuracy**, ya que la optimización del modelo no ha sido realizada en este conjunto. Así que **en accuracy, la mejor opción es Random Forest**, no obstante se podría dar el caso que el clasificador no identifique correctamente los pacientes con diabetes, ya que solo nos informa de la proporción de casos que se predijeron correctamente.

Así pues, también se han usado las métricas **precision** (precisión) y **recall** (exhaustividad). No se necesitan analizar más métricas porque la variable target es binaria y se encuentra balanceada.

La precisión nos informará de todos los pacientes que el modelo predijo que son diabéticos, ¿cuántos son realmente diabéticos?, y la exhaustividad nos informará de todos los pacientes que en realidad son diabéticos, ¿cuántos identificó el modelo?

Modelos	Precision	Recall	Accuracy
Logistic regression	0.764	0.696	0.798
k-nearest-neighbours	0.709	0.709	0.773
Linear SVM	0.764	0.696	0.798
SVM with RBF kernel	0.714	0.759	0.788
Random Forest	0.722	0.823	0.813

Cuadro 3: Precision, Recall y Accuracy de los modelos

La mayor precisión es de Logistic regression y Linear SVM, pero la mayor exhaustividad se ha obtenido con el algoritmo de Random Forest.

Según los datos obtenidos de la Tabla (3) no hay una gran diferencia entre los modelos, todas las métricas se encuentran entre el 0,7 y el 0,8 aproximadamente y además parece una clasificación buena, no tenemos una mala predicción y ningún modelo parece sobreajustado.

En relación tiempo-resultados, no considero que RandomForest sea el mejor algoritmo; sin embargo, si no tuviera un tiempo de ejecución tan alto y fuera tan costoso, hubiera elegido este sin dudarlo, pero no por el accuracy sino por el recall, ya que es el que tiene el valor más alto. El recall es la métrica que más nos interesa a nosotros, ya que es la que evita los falsos negativos, es decir, que una paciente diabética no sea identificada como no diabética.

Por tanto, con recall como métrica prioritaria evitamos que los pacientes con diabetes tengan consecuencias letales por no ser tratados, y con un tiempo de ejecución inferior a 15 segundos que es prácticamente nulo frente los 35 minutos de ejecución del RandomForest, he elegido el siguiente con mayor recall que es **SVM with RBF kernel**.

6. Conclusión

A estos resultados obtenidos le tendríamos que sumar el error que se comentó en el apartado de preprocessing cuando se realizó la imputación, y también las desviaciones generadas al entrenar los modelos, si hubiera tenido más tiempo me hubiera gustado dedicar más tiempo en el estudio de estos errores.

Referente al resultado se ha elegido el modelo de SVM with RBF kernel cuya accuracy es

del 78,8 % que en mi opinión es un resultado bastante bueno, a pesar de que los resultados de RandomForest son mejores, y el accuracy es un 3 % superior.

Respecto el proyecto, ha sido bastante grato poder aplicar muchos de los conceptos y ejercicios realizados en clase de teoría y laboratorio, ya que trasladar conocimiento académico en un problema real no siempre se puede y, por otra parte, me ha ayudado a entender mejor la exploración de datos, el tratamiento, la visualización con PCA y la aplicación de diferentes algoritmos para solucionar el problema, analizando el comportamiento de cada uno.

De hecho, este trabajo me ha hecho darme cuenta de que me gustan los datos, sobre todo la exploración de ellos, a lo mejor es tarde ya para hacer el trabajo de fin de grado orientado en machine learning, pero no descarto hacer el máster de ciencia de datos, ya que combina dos ámbitos que me gustan mucho la investigación y las matemáticas. Así pues, me alegro de haberlo descubierto.

Referencias

[1] Kaggle. *Pima Indians Diabetes Database* [En línea] [Consulta: 22 de febrero del 2022]. Disponible en: https://www.kaggle.com/uciml/pima-indians-diabetes-database

[2] Wikipedia. *Diabetes mellitus* [en línea] [Consulta: 22 de febrero 2022]. Disponible en: https://es.wikipedia.org/wiki/Diabetes_mellitus

[3] Wikipedia. *Glucose tolerance test* [en línea] [Consulta: 23 de febrero 2022]. Disponible en: https://www.mayoclinic.org/es-es/tests-procedures/glucose-tolerance-test/about/pac-20394296

[4] Wikipedia. *Presión sanguínea* [en línea] [Consulta: 23 de febrero 2022]. Disponible en: https://www.heart.org/en/health-topics/high-blood-pressure/understanding-blood-pressure-readings

[5] HaskellWiki. *Yampa/game engine* [en línea] [Consulta: 24 de febrero 2022]. Disponible en: https://www.brooklyn.cuny.edu/bc/ahp/LAD/C4d/C4d_skin.html

[6] Kaggle. *Step by Step Diabetes Classification-KNN-detailed* [En línea] [Consulta: 24 de febrero del 2022]. Disponible en: https://www.kaggle.com/shrutimechlearn/step-by-step-diabetes-classification-knn-detailed

[7] Kaggle. *Pima Indians Diabetes Database* [En línea] [Consulta: 24 de febrero del 2022]. Disponible en: https://www.kaggle.com/uciml/pima-indians-diabetes-database

[8] Choubey, Dilip Kumar., Paul, Sanchita. (2016) 'GA_MLP NN: A Hybrid Intelligent System for Diabetes Disease Diagnosis', International Journal of Intelligent Systems and Applications (IJISA), MECS, ISSN: 2074–904X (Print), ISSN: 2074–9058. (Online), Vol. 8, No. 1, pp.49–59.

[9] Jegan, Chitra. (2013). Classification Of Diabetes Disease Using Support Vector Machine. International Journal of Engineering Research and Applications. 3. 1797 - 1801.

[10] scikit-learn 1.0.2 documentation. *grid_search* [en línea] [Consulta: 24 de febrero 2022]. Disponible en: https://scikit-learn.org/stable/modules/grid_search.html

[11] scikit-learn 1.0.2 documentation. *sklearn.linear_model.LogisticRegression* [en línea] [Consulta: 24 de febrero 2022]. Disponible en: https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

[12] scikit-learn 1.0.2 documentation. *sklearn.gaussian_process.kernels.RBF* [en línea] [Consulta: 24 de febrero 2022]. https://scikit-learn.org/stable/modules/generated/sklearn.gaussian_process.kernels.RBF.html

[13] scikit-learn 1.0.2 documentation. *sklearn.svm.LinearSVC* [en línea] [Consulta: 24 de febrero 2022]. https://scikit-learn.org/stable/modules/generated/sklearn.svm.LinearSVC.html

[14] packtub. *Quantifying missing values* [en línea] [Consulta: 24 de febrero 2022]. https://subscription.packtpub.com/book/data/9781789806311/1/ch01lvl1sec04/quantifying-missing-data