Universidad de La Laguna

Práctica 1

Extracción de Conocimiento de Base de Datos

JORGE CABRERA RODRÍGUEZ
10-14-2023

Introducción al *dataset*

Este proyecto trabaja con una base de datos llamada *homeLoanAproval.csv*, que contiene información detallada sobre los créditos bancarios concedidos a 614 usuarios de una entidad bancaria desconocida. Esta base de datos cuenta con valores como puedan ser, entre otros:

- El estado civil del usuario (**Married**)
- EL nivel educativo del solicitante (**Education**)
- El importe del crédito solicitado (**LoanAmount**)
- El número de personas que dependan del solicitante (**Dependents**)
- El estado del crédito (**Loan_Status**) (Aprobado o denegado)

Con esta base de datos se busca construir un sistema que sea capaz de ayudar en la tarea de concesión de créditos bancarios para futuros usuarios, de tal forma que, para unos datos de entrada desconocidos para el sistema, este sea capaz de predecir si el crédito será aprobado o denegado.

Desarrollo del Proyecto

Visualización de los datos

La primera tarea que realizar con la base de datos es realizar una visualización inicial, para observar los datos con los que trabajamos, así como sus deficiencias y posibles problemas que puedan surgir.

	Loan_ID	Gender	Married	Dependents	Education	SelfEmployed	ApplicantIncome	CoapplicantIncome	LoanAmount	LoanAmountTerm	PropertyArea	LoanStatus
0	LP001002	Male	No	0	Graduate	No	5849	0	NaN	360.0	Urban	Y
1	LP001003	Male	Yes	1	Graduate	No	4583	1508	128.0	360.0	Rural	N
2	LP001005	Male	Yes	0	Graduate	Yes	3000	0	66.0	360.0	Urban	Y
3	LP001006	Male	Yes	0	Not Graduate	No	2583	2358	120.0	360.0	Urban	Y
4	LP001008	Male	No	0	Graduate	No	6000	0	141.0	360.0	Urban	Y

Ilustración 1. Visualización inicial de los datos

El *dataset* cuenta con 614 registros y 12 variables por cada registro. De estas 12 variables, 11 son variables de entrada y 1 es la variable de salida, que es la que se pretende predecir (**LoanStatus**).

La distribución de valores nulos registrados es la siguiente:

	Tipo de dato	Valores faltantes (NA)
Loan_ID	object	0
Gender	object	13
Married	object	3
Dependents	object	15
Education	object	0
SelfEmployed	object	32
ApplicantIncome	int64	0
CoapplicantIncome	object	0
LoanAmount	float64	22
LoanAmountTerm	float64	14
PropertyArea	object	0
LoanStatus	object	0

Ilustración 2. Recuento de valores nulos

Además de los valores nulos, también es necesario analizar los valores atípicos (*outliers*) que puedan existir en la base de datos. Para ello, se ha realizado un análisis de los valores atípicos de las variables numéricas, que son las siguientes:

- **ApplicantIncome**
- **LoanAmount**
- **LoanAmountTerm**

Para este análisis, se ha utilizado el diagrama de cajas (*boxplot*), que nos permite observar los valores atípicos de una variable de forma visual.

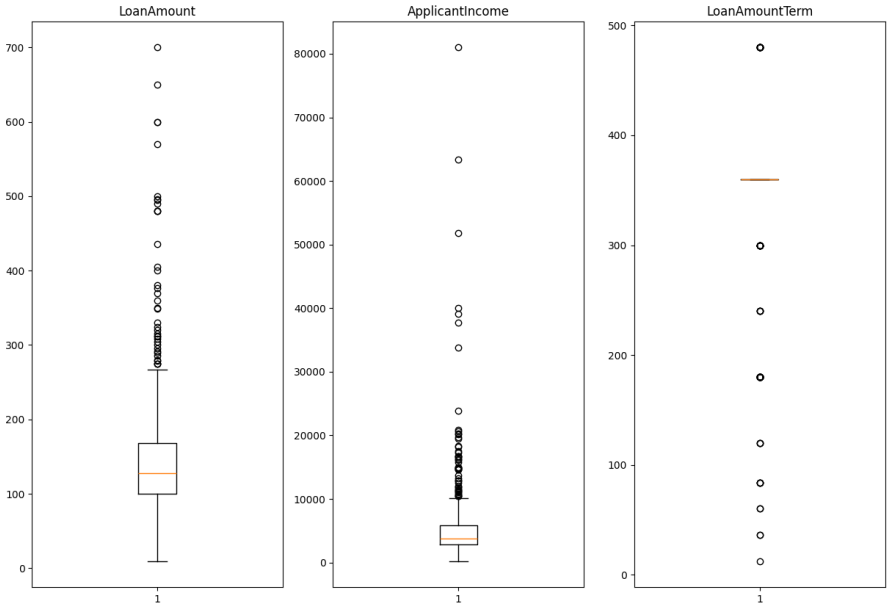

Ilustración 3. Diagrama de cajas 1

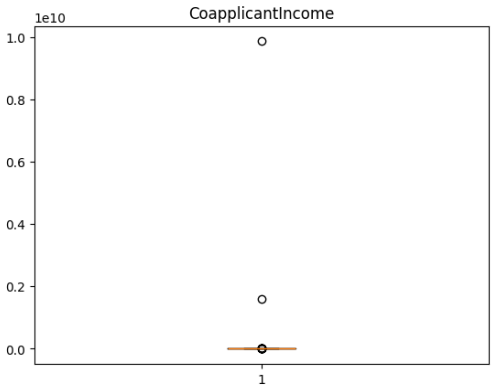

Ilustración 4. Diagrama de cajas 2

Como último paso de visualización de datos, puede ser interesante analizar el balance de clases de la variable de salida **LoanStatus**, para observar si existe un desbalance que pueda afectar al modelo de clasificación.

	Registros	Porcentaje (%)
LoanStatus		
Y	422	68.73
N	192	31.27

Ilustración 5. Balance de clases

Podemos observar un desbalance de clases moderado, pues casi un 70% de las entradas pertenecen a la clase de créditos aprobados (**Y**), mientras que el resto pertenecen a la clase de créditos denegados (**N**).

Siendo un desbalance no tan marcado, es probable que no influya demasiado en el modelo de clasificación. Sin embargo, sería interesante más adelante realizar un entrenamiento del modelo con y sin balanceo de clases, para observar si existe alguna mejora en el rendimiento de este.

Preprocesado de datos

Eliminación de variables innecesarias

El primer paso de la fase de preprocesado será la eliminación de aquellas variables que no aporten información al modelo, o sean irrelevantes por su propia naturaleza.

En este caso en particular, todas las variables aportan información sobre los clientes que puede ser relevante; todas menos la variable **LoanID**, pues es un valor serial que se genera de forma automática para cada registro, y no aporta ninguna información útil para un modelo de clasificación.

Corrección de valores faltantes

Tras haber eliminado las variables innecesarias, se procederá a corregir los valores faltantes de las variables restantes. Para esto, generaremos dos conjuntos de datos, uno con los valores faltantes eliminados, y otro con los valores faltantes completados mediante técnicas tales como:

- Media de los valores de la variable
- Mediana de los valores de la variable
- Moda de los valores de la variable
- Media de los valores dependiendo de otra variable

El primer paso en esta fase es la eliminación de los valores faltantes, guardando los resultados en un primer conjunto de datos *'Trimmed Dataset'*.

El siguiente paso es rellenar todos los campos faltantes, dependiendo del tipo de dato y de su naturaleza, y guardar los resultados en un conjunto de datos *'Filled Dataset'*:

- Los datos no numéricos se pueden rellenar según la moda de sus valores, pero no según la media ni la mediana, pues no podemos realizar los cálculos de promedio sobre ellos.
- Los datos numéricos se pueden rellenar tanto con la media como con la moda o la mediana.
    - Sin embargo, si contienen valores anómalos (outliers), es mejor rellenar con la mediana, pues esta se ve menos afectada por los valores anómalos.

Campo a rellenar	Tipo de dato	Transformación realizada
Gender	Cualitativo	Sustitución por moda
Married	Cualitativo	Sustitución por moda
Dependents	Cualitativo	Sustitución por moda
SelfEmployed	Cualitativo	Sustitución por moda
LoanAmount	Cuantitativo	Sustitución por media
LoanAmountTerm	Cuantitativo	Sustitución por media

Ilustración 6. Sustitución de variables nulas

Etiquetado

Para aquellas variables cualitativas que no sean binarias, se realizará un etiquetado de los valores en ambos conjuntos de datos, de tal forma que se conviertan en valores numéricos que puedan ser utilizados por el modelo de clasificación.

	Gender	Married	Dependents	Education	SelfEmployed	ApplicantIncome	CoapplicantIncome	LoanAmount	LoanAmountTerm	PropertyArea	LoanStatus
0	1	0	0	0	0	5849	0	128.0	360.0	2	1
1	1	1	1	0	0	4583	1508	128.0	360.0	0	0
2	1	1	0	0	1	3000	0	66.0	360.0	2	1
3	1	1	0	1	0	2583	2358	120.0	360.0	2	1
4	1	0	0	0	0	6000	0	141.0	360.0	2	1

Ilustración 7. Ejemplo de dataset tras etiquetado

Corrección de *outliers*

Para la corrección de *outliers* voy a eliminar todos aquellos valores por encima y por debajo el valor de la mediana más 1.5 veces el rango intercuartílico.

En esta corrección, se eliminarán todos los valores atípicos menos los relacionados con **LoanAmountTerm**, pues al ser un campo tan concentrado alrededor del valor 360 (como se observa en el apartado de visualización de datos), nos quedaría una columna con únicamente un valor, lo que no aportaría información alguna al problema.

Tras realizar la corrección se obtienen los siguientes resultados:

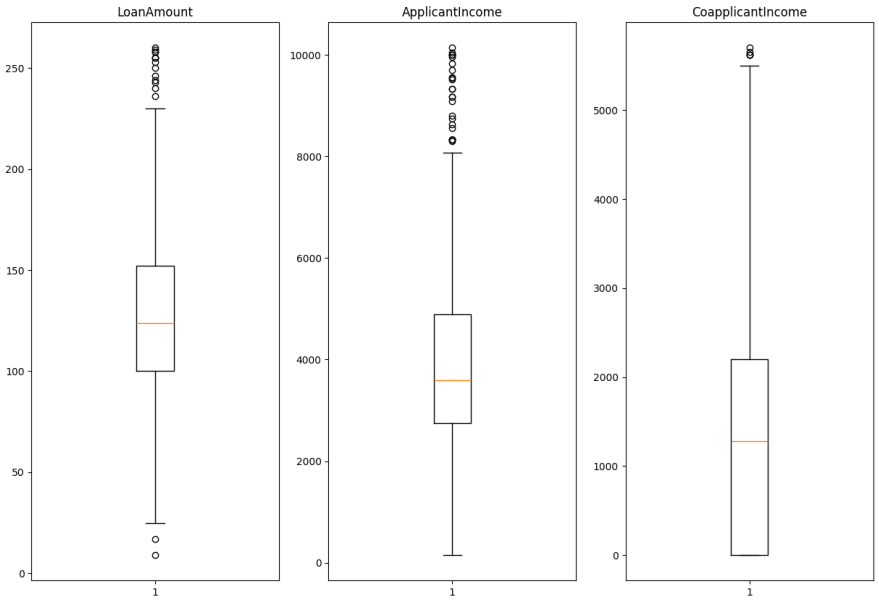

Ilustración 8. Distribución de valores tras corrección de outliers

Como se puede observar, se han eliminado gran parte de los registros que contenían valores *outliers* a partir de los umbrales calculados. Los *outliers* actuales son aquellos puntos que originalmente se contaban como valores ordinarios, pero por el hecho de haber eliminado otros valores, ahora se reconsideran como *outliers*.

En este proyecto se conservará esta "segunda generación" de *outliers*, pues ya se ha realizado suficiente limpieza.

Normalización de los datos

Una vez preprocesados, formateados y limpiados los datos, se procederá a normalizarlos en ambos *datasets* (**trimmed** y **filled**), para que todos los valores se encuentren en un rango similar y no haya valores que puedan afectar al modelo de clasificación.

	Gender	Married	Dependents	Education	SelfEmployed	ApplicantIncome	CoapplicantIncome	LoanAmount	LoanAmountTerm	PropertyArea	LoanStatus
0	1	0	0	0	0	0.570528	0.000000	0.474104	0.74359	2	1
1	1	1	1	0	0	0.443788	0.264515	0.474104	0.74359	0	0
2	1	1	0	0	1	0.285314	0.000000	0.227092	0.74359	2	1
3	1	1	0	1	0	0.243568	0.413612	0.442231	0.74359	2	1
4	1	0	0	0	0	0.585644	0.000000	0.525896	0.74359	2	1

*Ilustración 9. Normalizado de **filled dataset***

	Gender	Married	Dependents	Education	SelfEmployed	ApplicantIncome	CoapplicantIncome	LoanAmount	LoanAmountTerm	PropertyArea	LoanStatus
1	1	1	1	0	0	0.054830	1.529722e-07	0.185647	0.74359	0	0
2	1	1	0	0	1	0.035250	0.000000e+00	0.088924	0.74359	2	1
3	1	1	0	1	0	0.030093	2.391966e-07	0.173167	0.74359	2	1
4	1	0	0	0	0	0.072356	0.000000e+00	0.205928	0.74359	2	1
5	1	1	2	0	1	0.065145	4.256442e-07	0.402496	0.74359	2	1

*Ilustración 10. Normalizado de **trimmed datasetw***

Balance de clases

Para solventar el desbalance de clases que se observó en la fase de visualización de datos, se utilizará la técnica de *undersampling* sobre la clase mayoritaria una vez sea generado el subconjunto de entrenamiento.

Para ello, dividiremos los dos conjuntos de datos de los que disponemos en:

- Entrenamiento (70%)
- Test (30%)

Posteriormente, realizaremos un proceso de *undersampling* sobre el subconjunto de entrenamiento para obtener un balance correcto de clases, resultando en la siguiente proporción para "**filled**" y "**trimmed**" respectivamente.

	count
LoanStatus	
0	124
1	124

	count
LoanStatus	
0	122
1	122

Ilustración 11. Balance de clases tras balanceo

Con esto obtendremos 4 posibles conjuntos de datos a analizar:

- **Trimmed** balanceado
- **Trimmed** sin balancear
- **Filled** balanceado
- **Filled** sin balancear

Definición de modelos

Cada conjunto de datos expuesto anteriormente se utilizará con los siguientes algoritmos de clasificación:

- Clasificador KNN
- Árbol de clasificación
- Clasificador *Naive Bayes*

Resultados extraídos

Trimmed balanceado

	algorithm	accuracy	precision_n	precision_y	recall_n	recall_y
0	KNeighborsClassifier	0.590476	0.404255	0.741379	0.558824	0.605634
1	DecisionTreeClassifier	0.495238	0.313725	0.666667	0.470588	0.507042
2	GaussianNB	0.323810	0.316832	0.500000	0.941176	0.028169

*Ilustración 12. Resultados de **trimmed** balanceado*

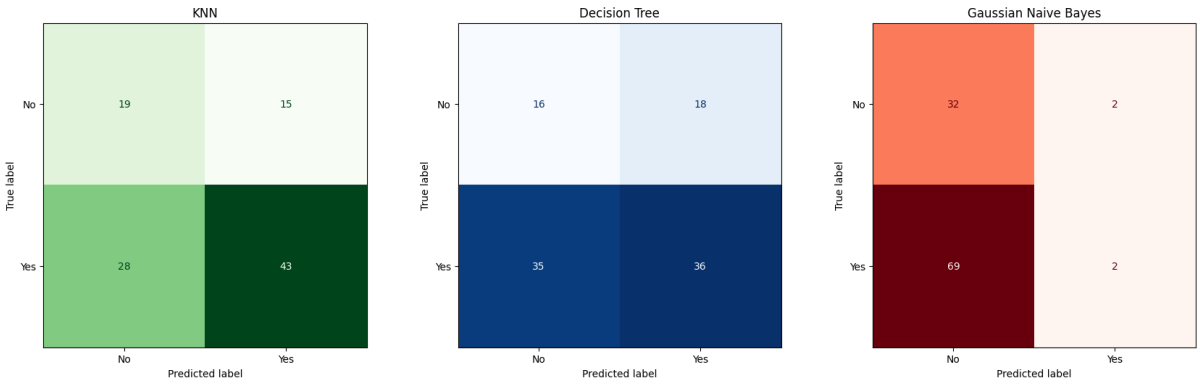

*Ilustración 13. Matrices de confusión de **trimmed** balanceado*

Trimmed sin balancear

	algorithm	accuracy	precision_n	precision_y	recall_n	recall_y
0	KNeighborsClassifier	0.638095	0.300000	0.673684	0.088235	0.901408
1	DecisionTreeClassifier	0.638095	0.433333	0.720000	0.382353	0.760563
2	GaussianNB	0.314286	0.317308	0.000000	0.970588	0.000000

*Ilustración 14. Resultados **trimmed** sin balancear*

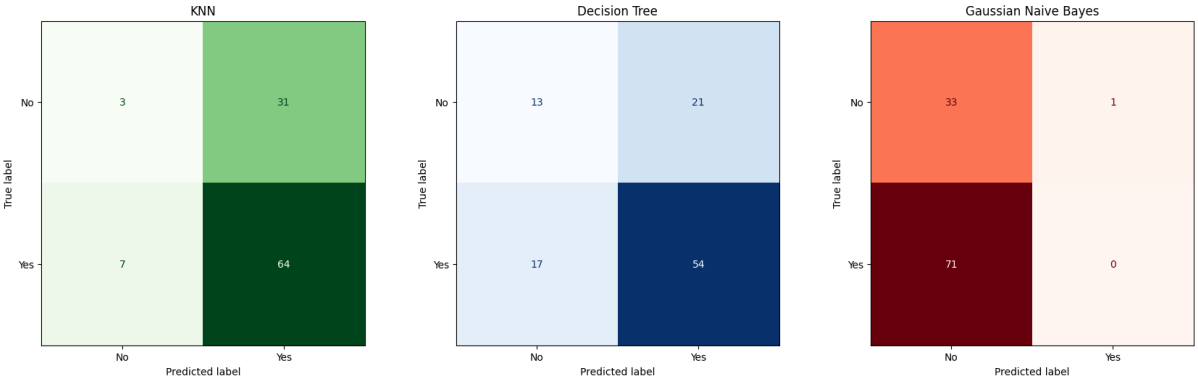

*Ilustración 15. Matrices de confusión de **trimmed** sin balancear*

Filled balanceado

	algorithm	accuracy	precision_n	precision_y	recall_n	recall_y
0	KNeighborsClassifier	0.590476	0.404255	0.741379	0.558824	0.605634
1	DecisionTreeClassifier	0.495238	0.313725	0.666667	0.470588	0.507042
2	GaussianNB	0.323810	0.316832	0.500000	0.941176	0.028169

*Ilustración 16. Resultados de **filled** balanceado*

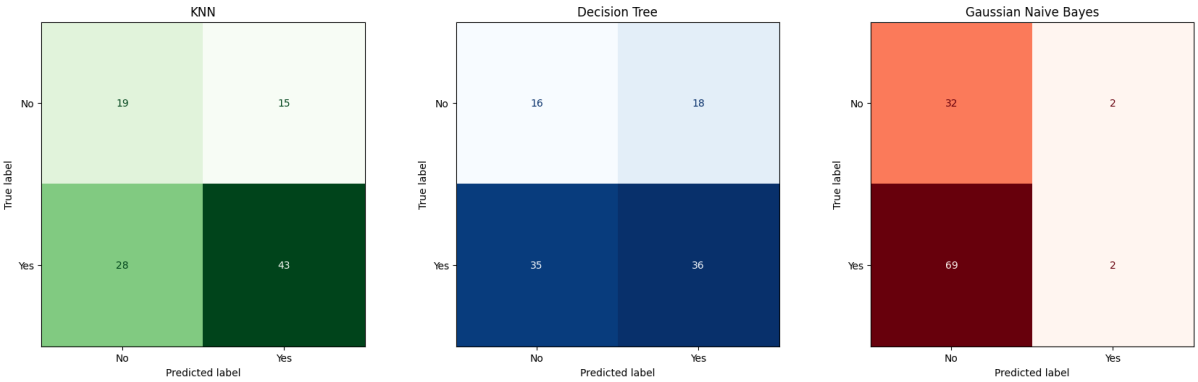

*Ilustración 17. Matrices de confusión de **filled** balanceado*

Filled sin balancear

	algorithm	accuracy	precision_n	precision_y	recall_n	recall_y
0	KNeighborsClassifier	0.626168	0.444444	0.642857	0.102564	0.926471
1	DecisionTreeClassifier	0.579439	0.411765	0.657534	0.358974	0.705882
2	GaussianNB	0.626168	0.428571	0.640000	0.076923	0.941176

*Ilustración 18. Resultados de **filled** sin balancear*

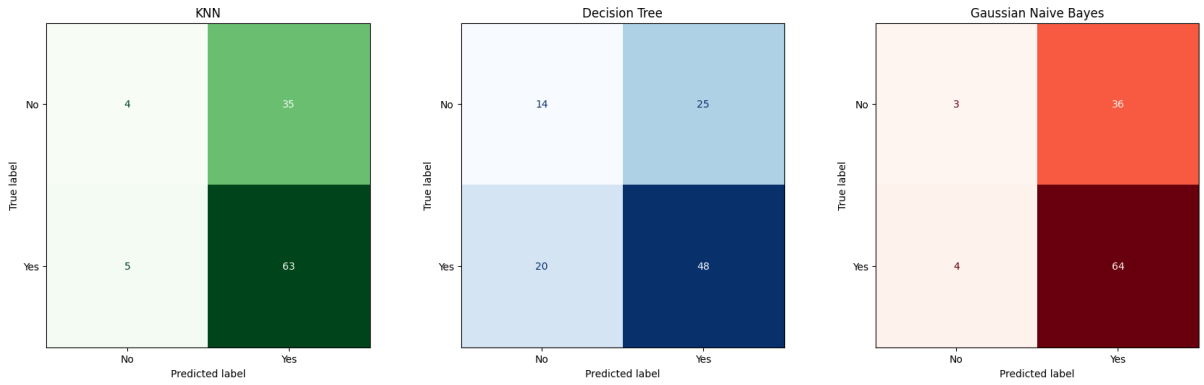

*Ilustración 19. Matrices de confusión de **filled** sin balancear*

Conclusiones extraídas

Los resultados obtenidos en el proyecto son bastante precarios, pues ninguno de los clasificadores entrenados ha obtenido exactitudes *(accuracy)* superiores al 65%, lo que puede indicar que el modelo no es capaz de generalizar bien los datos.

Analizando los resultados por conjunto, podemos observar que aquellos conjuntos con información sin procesar (**trimmed_dataset** y su versión balanceada **trimmed_dataset_balanced**) obtienen mejores *accuracies* que aquellos conjuntos con información preprocesada (**filled_dataset** y su versión balanceada **filled_dataset_bal**). Sin embargo, los valores de *recall* y *precision* tienden a ser más polares en los conjuntos sin procesar que en los procesados, que son más estables.

Observando las matrices de confusión, podemos percibir resultados muy similares entre los conjuntos sin procesar y los procesados, sobre todo con los algoritmos **KNN** y **Árbol de clasificación**. Sin embargo, el algoritmo **Naive Bayes** obtiene predicciones opuestas entre ambos conjuntos. No se puede distinguir ninguna diagonal marcada en ninguna matriz de confusión, por lo que incluso sin analizar los valores de *accuracy*, se puede observar que el modelo no es capaz de generalizar bien los datos.

Como futuras mejoras al modelo para buscar mejores resultados, se podría analizar qué variables son las que más influyen en la predicción, y eliminar aquellas que no aporten información relevante al modelo. Además, se podrían generar variables *dummy* para aquellas variables cualitativas que no sean binarias, para que el modelo pueda trabajar con ellas de forma más eficiente. También podría ser interesante el análisis de los *outliers* para ver si los datos eliminados fueran realmente *outliers*, o si por el contrario se han eliminado valores válidos para el modelo.