# ***PREVENCIÓN DE IMPAGOS EN PRÉSTAMOS BANCARIOS MEDIANTE TÉCNICAS DE APRENDIZAJE AUTOMÁTICO***

**Autor:** Illana Guerra, Fátima

## 1. INTRODUCCIÓN

El Aprendizaje Supervisado forma parte de la rama de Aprendizaje Automático. Consiste en entrenar un modelo utilizando un conjunto de datos de entrenamiento con el objetivo de que dicho modelo aprenda automáticamente la relación que existe entre las entradas y las salidas, pudiendo realizar predicciones posteriores sobre datos no clasificados.

En nuestro caso concreto utilizaremos Aprendizaje Supervisado orientado a clasificación binaria, es decir, dada una variable categórica con dos valores posibles, trataremos de predecir el valor de dicha etiqueta en una nueva instancia.

## 2. DESCRIPCIÓN DEL PROBLEMA

La base de datos que hemos escogido para nuestro estudio se llama 'Statlog (German Credit Data)' [1] y consiste en un conjunto de datos que tiene como objetivo determinar si una persona es apta o no para recibir un préstamo, concretamente en la economía alemana.

En total tenemos 5 variables ordinales, 7 variables numéricas y 8 variables categóricas, de entre las cuales 2 son binarias:

1. Cuenta de cheques- Ordinal- 4 valores posibles: [A14 < A11 < A12 < A13]
2. Duración del préstamo- Numérica- Unidad: meses
3. Historial de préstamos- Ordinal- 5 valores posibles: [A34 < A33 < A32 < A31 < A30]
4. Propósito del crédito- Categórica- 11 valores posibles
5. Cantidad del préstamo- Numérica- Unidad: Deutsche Mark
6. Cuenta de Ahorros- Ordinal- 5 valores posibles: [A65 < A61 < A62 < A63 < A64]
7. Tiempo Laboral- Ordinal- 5 valores posibles: [A71 < A72 < A73 < A74 < A75]
8. Tasa Cuota Mensual/Ingresos- Numérica- Unidad: porcentaje
9. Estado personal y sexo- Categórica- 5 valores posibles
10. Presencia de segundo deudor/garante- Categórica- 3 valores posibles
11. Antigüedad en hogar- Numérica- Unidad: años
12. Propiedades- Categórica- 4 valores posibles
13. Edad- Numérica- Unidad: años
14. Presencia de otros pagos a plazos- Categórica- 3 valores posibles
15. Propiedad del hogar- Categórica- 3 valores posibles
16. Número de créditos en este banco- Numérica- Unidad: enteros
17. Estado de profesión- Ordinal- 4 valores posibles: [A171 < A172 < A173 < A174]
18. Número de personas dependientes del préstamo- Numérica- Unidades: enteros
19. Estado del registro del número de teléfono- Binaria [0, 1]
20. Trabajador extranjero- Binaria [0, 1]

La última variable es la variable de predicción 'class' que determina si una persona es o no apta, con una etiqueta '1' y '2' respectivamente.

*(Para más información detallada sobre las variables consultar documentación, referencia[1])*

Sabemos por la documentación de los datos que existe una versión completamente numérica. No obstante, dicha versión no cuenta con ningún tipo de documentación adicional ni explicación de su preprocesado haciendo imposible interpretar el significado de los datos. Por ello, preprocesamos la base de datos original.

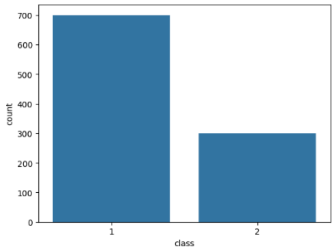
Nos damos cuenta de que tenemos 2 variables, 'Número de teléfono' y 'Propiedad del hogar', que son redundantes y no nos proporcionan ninguna información acerca de la aptitud de una persona para recibir el préstamo. Por lo tanto, las eliminamos.

También eliminamos la variable que involucra el sexo de una persona y su situación personal ya que su distribución es completamente irregular y podría generar predicciones sesgadas.

Una vez preprocesados, estos serán los datos que utilizaremos para nuestro estudio.

| ... | Cantidad | Ahorros | AntiguedadLaboral | TasaPrestamo | AntiguedadHogar | Edad | NCreditos | Profesion | NDependencias | Extranjero |
|---|---|---|---|---|---|---|---|---|---|---|
| ... | 0.050567 | 0.0 | 4.0 | 4.0 | 4.0 | 0.857143 | 2.0 | 2.0 | 1.0 | 1.0 |
| ... | 0.313690 | 1.0 | 2.0 | 2.0 | 2.0 | 0.053571 | 1.0 | 2.0 | 1.0 | 1.0 |
| ... | 0.101574 | 1.0 | 3.0 | 2.0 | 3.0 | 0.535714 | 1.0 | 1.0 | 2.0 | 1.0 |
| ... | 0.419941 | 1.0 | 3.0 | 2.0 | 4.0 | 0.464286 | 1.0 | 2.0 | 2.0 | 1.0 |
| ... | 0.254209 | 1.0 | 2.0 | 3.0 | 4.0 | 0.607143 | 2.0 | 2.0 | 2.0 | 1.0 |

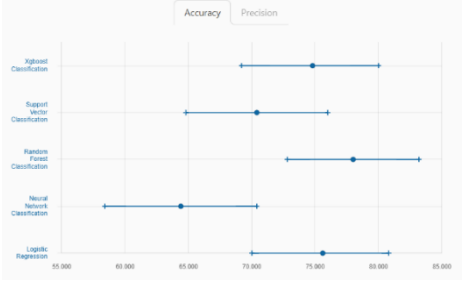
Por otro lado, antes de comenzar con el proceso de modelado, comprobamos el balance del conjunto de datos:

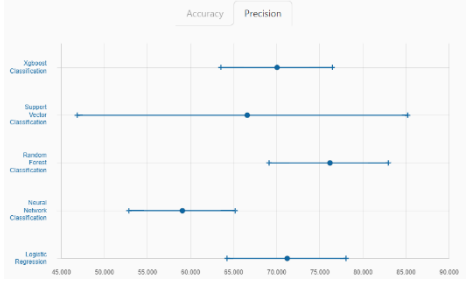



Como podemos observar se nos plantea un problema de clases desbalanceadas, al haber 700 instancias clasificadas como 'Aptos' frente a las 200 clasificadas como 'No Aptas'. Al tratarse de una diferencia considerable nuestros modelos podrían quedar sesgados a favor de la clase dominante. Para solucionarlo, tendremos que llevar a cabo técnicas como el ajuste de datos, en aquellos algoritmos que lo permitan.

Además, tenemos concretamente que el 70% de los datos pertenecen a la clase 1 y que el 30% de los datos pertenecen a la clase 2. Por ello sabemos que partimos de una precisión base de 0.7 ya que, si clasificáramos todas las instancias como 'Apto' tendríamos una precisión del 70%. Si nuestros modelos no pueden superar esa precisión realmente no nos estarán proporcionando ninguna información útil en la predicción.

Concretamente, según la documentación de los datos, partimos de las siguientes métricas de rendimiento:



Ahora, estudiaremos los criterios de evaluación y la información que nos proporcionan en nuestro caso de estudio concreto:

**Precisión**: esta medida determina la proporción de instancias que, clasificadas como positivas, son correctas. En nuestro caso esta métrica es importante ya que, si la precisión es pequeña, significa que hay muchos falsos positivos, es decir, mucha gente que no es apta, pero se le clasifica como apta. Esto, en relación con la economía del banco, puede llegar a suponer grandes pérdidas por lo que se trata de una métrica muy importante a tener en cuenta.

**Recall/Recuperación**: esta medida determina la proporción de valores positivos clasificados correctamente. En nuestro caso esta métrica es importante a la hora de atender a los casos donde se clasifica un potencial cliente como 'No Apto'. Esto puede ser importante si tenemos en cuenta la pérdida de oportunidad para el prestamista.

**F1-Score**: esta métrica combina la precisión y la recuperación. Útil si queremos buscar un equilibrio entre ambas métricas.

**Exactitud/Accuracy**: esta medida determina la proporción de valores clasificados correctamente, sean positivos o negativos. Mide el rendimiento del modelo y es la métrica en la que nos fijaremos para compararla con la precisión base antes explicada.

**Macro avg (Promedio Macro)**: nos proporciona la media de todas las métricas por clase, dando la misma importancia a ambas, sin tener en cuenta el desbalance. En nuestro caso, al tener los datos tan desproporcionados, no nos aporta demasiada información real sobre el rendimiento del modelo.

**Weighted avg (Promedio Ponderado)**: es útil cuando hay un desequilibrio entre las clases por lo que, en nuestro caso, refleja mejor el rendimiento general del modelo en todo el conjunto de datos.

**Validación Cruzada Estratificada**: esta métrica evalúa el rendimiento general de un modelo, dividiendo los datos en subconjuntos y utilizándolos para entrenar el modelo en una parte y evaluarlo en otra, repitiendo el proceso varias veces y obteniendo la media de los resultados. Concretamente la Validación Cruzada K-Fold divide los datos en K conjuntos (folds) y entrena el modelo K veces, utilizando cada pliegue como conjunto de prueba una única vez. Se asegura que la proporción de clases en cada capa sea similar a la de los datos originales por lo que nos es útil en nuestros datos, con distribución desequilibrada.

## 3. METODOLOGÍA

Una vez tenemos los datos listos, los dividimos en datos de entrenamiento y de prueba, para entrenar y evaluar los modelos, y creamos una instancia StratifiedKFold, con 10 particiones en concreto, para hallar la mejor combinación de parámetros en cada modelo. S

```
XTrain, XTest, yTrain, yTest = train_test_split(X, y, test_size=0.20, random_state=42)
skf = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
```

Ahora creamos 4 modelos distintos, uno por cada técnica:

MODELO 1: K-NN

Para el primer modelo utilizaremos el algoritmo K-NN. Este consiste en representar el número de vecinos más cercanos que se debe considerar al realizar una predicción. Para cada punto se

calculan aquellos vecinos que estén más cerca según las distancias entre puntos obtenidas, asignándole la etiqueta de clasificación más común entre dichos vecinos.

Primero utilizaremos la función GridSearchCV para crear un clasificador 'KNeighborsClassifier' con el mejor valor K posible. En nuestro caso obtenemos que K = 16.

```
clf_KNN = GridSearchCV(KNeighborsClassifier(metric='euclidean'), param_grid, cv=skf, scoring='accuracy')
```

Una vez tenemos el modelo óptimo, lo ajustamos a los datos de entrenamiento y realizamos las predicciones sobre el conjunto de entrenamiento y de test.

```
model_KNN = KNeighborsClassifier (n_neighbors = k, metric='euclidean')
model_KNN.fit(XTrain, yTrain)
yhatTrain = model_KNN.predict(XTrain)
yhatTest = model_KNN.predict(XTest)
```

## MODELO 2: ÁRBOLES DE DECISIÓN

Para el segundo modelo utilizaremos la técnica de Árboles de Decisión. Este algoritmo divide los datos en una estructura jerárquica donde los datos se agrupan en subconjuntos en base a sus características. Cada nodo representa una pregunta sobre una característica en concreto y las ramas presentan las posibles respuestas. Al final, las hojas del árbol representan las predicciones del modelo.

De nuevo, establecemos un conjunto de valores posibles para los distintos parámetros del algoritmo. De entre ellos, escogemos valores pequeños para la profundidad máxima del árbol y establecemos la posibilidad de un 'ccp_alpha' distinto de 0 para poder aplicar una poda.

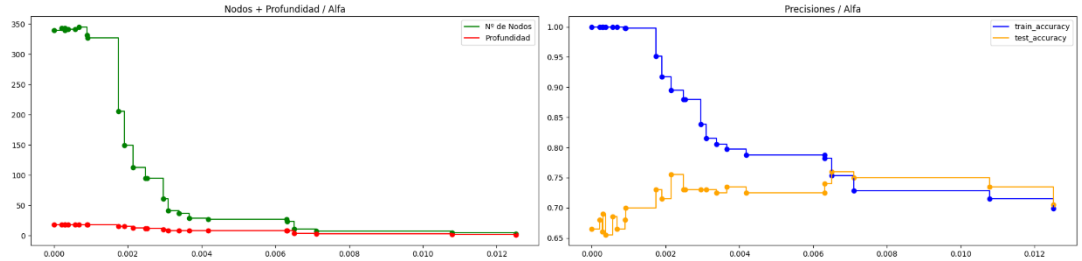
```
param_grid = {
    'min_samples_leaf': [5,10],
    'min_samples_split':[5, 10, 20],
    'max_depth': np.arange(2,11),
    'criterion': ['entropy', 'gini'],
    'ccp_alpha': [0, 0.01, 0.02, 0.05]
}
```

```
clf_ADD = GridSearchCV(DecisionTreeClassifier(random_state=42), param_grid, cv=skf)
clf_ADD.fit(XTrain , yTrain)

model_ADD = clf_ADD.best_estimator_
print("Best estimator found by grid search: ", model_ADD)

Best estimator found by grid search:  DecisionTreeClassifier(ccp_alpha=0, max_depth=6, min_samples_leaf=5,
                       min_samples_split=20, random_state=42)
```

```
The train accuracy is: 0.8025
The test accuracy is: 0.745
```

Utilizando Validación Cruzada obtenemos que el valor óptimo para alfa es 0, es decir, no aplicar una poda al árbol. No obstante, si ajustamos el modelo a los datos de entrenamiento y realizamos las predicciones observamos que obtenemos un modelo sobreajustado.

Por ello realizamos una búsqueda manual [2] del mejor valor para alfa, obteniendo una pila de posibles valores y creando un modelo para cada uno. Estudiamos entonces el número de nodos, profundidad del árbol, precisión de entrenamiento y precisión de test para cada caso:




Vemos que con un valor 0 de alfa los datos de entrenamiento y test difieren considerablemente entre sí. Cogemos entonces el mejor valor que, según la gráfica nos presenta una relación equilibrada entre el número de nodos y la profundidad máxima, y entre las precisiones.

Volvemos a aplicar Validación Cruzada para buscar la mejor combinación de valores ya que, al cambiar alfa, ésta también cambiará. El mejor modelo es:

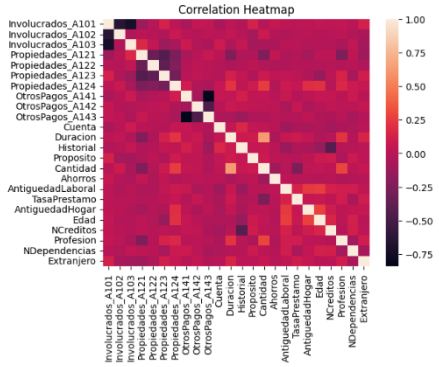
```
Best estimator found by grid search:  DecisionTreeClassifier(ccp_alpha=0.006506107804891942, max_depth=4,
                      min_samples_leaf=5, min_samples_split=5,
                      random_state=42)
```

## MODELO 3: REGRESIÓN LOGÍSTICA

Para el tercer modelo utilizaremos Regresión Logística, un algoritmo que permite clasificar observaciones en función de clases categóricas (discretas). Consiste en implementar una ecuación lineal con las variables explicativas independientes del tipo: z = β0 + β1x1 + β2x2 + ... + βnxn donde los coeficientes βi son los parámetros del modelo.

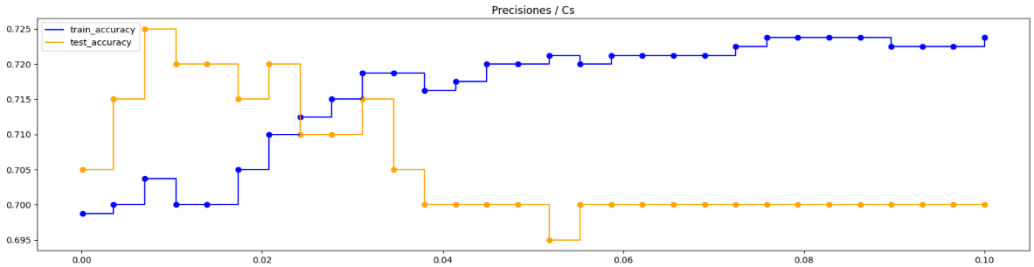
El valor z obtenido por la recta se introduce después como entrada de una función sigmoide que lo expresa como una probabilidad, en rango [0, 1]. En nuestro caso, como se trata de una decisión binaria, la frontera de decisión se establece en 0.5 generalmente donde aquellos puntos con p>=0.5 se clasificarán como 'Aptos' y aquellos con p<=0.5 como 'No Aptos'.

Por ello, primero observamos la correlación entre las variables para asegurarnos de que no existe multicolinealidad:




Después, el parámetro del algoritmo más importante a determinar es el valor de 'C'. Este parámetro ayuda a prevenir el sobreajuste del modelo al penalizar los coeficientes más grandes. Es el inverso del parámetro de regularización por lo que, cuanto mayor sea C menor será la regularización. Nos interesa entonces tener valores de C bajos.

En vez de utilizar Validación Cruzada, al igual que en el árbol de decisión, buscamos 'manualmente' el valor óptimo de C fijándonos en las precisiones de entrenamiento y test:




Si aplicamos Validación Cruzada obtenemos el siguiente valor:

```
Best estimator found by grid search:  LogisticRegression(C=0.05521724137931035, random_state=42)
```

Según la gráfica, con dicho valor de C obtendríamos, de nuevo, un modelo sobreajustado, de forma que escogemos el que nos indica mayor equilibrio según la gráfica y nos queda el siguiente modelo:

```
Best estimator found:  LogisticRegression(C=0.031103448275862068, random_state=42)
```

MODELO 4: SVM (Support Vector Machines)

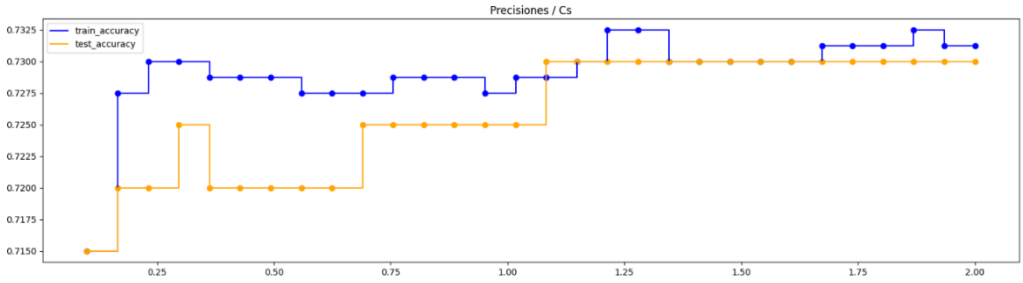
Para el cuarto y último modelo utilizaremos el algoritmo SVM (Support Vector Machines), que consiste en buscar un hiperplano que maximice el margen entre ambas clases, en el caso de clasificación binaria.

De nuevo tenemos que encontrar el valor óptimo del parámetro de regularización 'C'. Para ello, en este caso, primero empleamos Validación Cruzada para encontrar un valor aproximado del rango en el que debemos buscar con exactitud. Probamos con valores de tamaños distintos:

```
param_grid = {'C': [0.001, 0.01, 0.1, 1, 10]}
clf_SVC = GridSearchCV(svm.SVC(kernel="linear", random_state=42), param_grid, cv=skf)
clf_SVC.fit(XTrain, yTrain)
model_SVC = clf_SVC.best_estimator_
print("Best estimator found by grid search: ", model_SVC)
```

```
Best estimator found by grid search:  SVC(C=1, kernel='linear', random_state=42)
```

Realizamos una búsqueda manual en torno a dicho valor:




Fijándonos en la gráfica obtenemos el siguiente modelo:

```
Best estimator found:  SVC(C=1.1482758620689655, kernel='linear')
```

# 3. RESULTADOS

MODELO 1: K-NN

```
----------Train Confusion Matrix----------
[[523  36]
 [163  78]]

              precision    recall  f1-score   support

           1       0.76      0.94      0.84       559
           2       0.68      0.32      0.44       241

    accuracy                           0.75       800
   macro avg       0.72      0.63      0.64       800
weighted avg       0.74      0.75      0.72       800
```

```
----------Test Confusion Matrix----------
[[130  11]
 [ 47  12]]

              precision    recall  f1-score   support

           1       0.73      0.92      0.82       141
           2       0.52      0.20      0.29        59

    accuracy                           0.71       200
   macro avg       0.63      0.56      0.56       200
weighted avg       0.67      0.71      0.66       200
```

| Data | Accuracy | Precision | Recall | F1 | Error | Std Error |
|---|---|---|---|---|---|---|
| Train | 0.75 | 0.76 | 0.94 | 0.84 | 0.25 | 0.43 |
| Test | 0.71 | 0.73 | 0.92 | 0.82 | 0.34 | 0.47 |

```
CV-Validation Accuracy 0.73
```

Lo primero en lo que nos fijamos es que, tanto en el conjunto de entrenamiento como de test hay más falsos negativos (FN) que falsos positivos (FP). Como explicamos anteriormente, en nuestro caso, este es el caso ideal ya que para el banco supone más pérdidas clasificar como 'Apto' a un individuo realmente 'No Apto'. Además, en general, parece que el modelo muestra mayor exactitud en todas las métricas para la 1º etiqueta, lo que sugiere que tiene un mejor rendimiento para predecir la clase 1 tal y como esperábamos debido al desbalance de los datos.

Por otro lado, según la precisión de entrenamiento, el 76% de los individuos clasificados como 'Apto' realmente lo eran y, en el caso del conjunto de test, el 73%. Mientras tanto, según la medida de recuperación (Recall) el 94% de los individuos fueron clasificados como Aptos correctamente en el conjunto de entrenamiento y en el de test un 92%. Además, según la exactitud, en general, el 75% de instancias fueron clasificadas correctamente mientras que en el conjunto de test baja al 71%. Por último, según la precisión de Validación Cruzada el rendimiento general del modelo es del 73%.

## MODELO 2: ÁRBOLES DE DECISIÓN

```
----------Train Confusion Matrix----------
[[478  81]
 [116 125]]

              precision    recall  f1-score   support

           1       0.80      0.86      0.83       559
           2       0.61      0.52      0.56       241

    accuracy                           0.75       800
   macro avg       0.71      0.69      0.69       800
weighted avg       0.75      0.75      0.75       800
```

```
----------Test Confusion Matrix----------
[[122  19]
 [ 29  30]]

              precision    recall  f1-score   support

           1       0.81      0.87      0.84       141
           2       0.61      0.51      0.56        59

    accuracy                           0.76       200
   macro avg       0.71      0.69      0.70       200
weighted avg       0.75      0.76      0.75       200
```

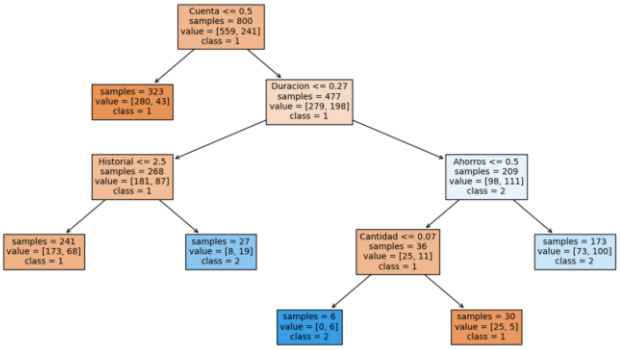
| Data | Accuracy | Precision | Recall | F1 | Error | Std Error |
|---|---|---|---|---|---|---|
| Train | 0.75 | 0.80 | 0.86 | 0.83 | 0.25 | 0.43 |
| Test | 0.76 | 0.81 | 0.87 | 0.84 | 0.40 | 0.49 |

```
CV-Validation Accuracy:  0.7237500000000001
```

Al igual que antes tanto en el conjunto de entrenamiento como de test hay más falsos negativos (FN) que falsos positivos (FP). No obstante, en este caso la diferencia entre ambos no es tan grande. Además, de nuevo, el modelo muestra mayor exactitud en todas las métricas para la 1º etiqueta, pero también con menos diferencia entre ambas, lo que sugiere que este modelo tiene un mejor rendimiento que el anterior sobre la predicción de la segunda etiqueta. Tiene mayor capacidad de generalización.

Por otro lado, según la precisión de entrenamiento, el 80% de los individuos clasificados como 'Apto' realmente lo eran y, en el caso del conjunto de test, el 81%. Mientras tanto, según la medida de recuperación (Recall) el 86% de los individuos fueron clasificados como Aptos correctamente en el conjunto de entrenamiento y en el de test un 87%. Además, según la exactitud, en general, el 75% de instancias fueron clasificadas correctamente mientras que en el conjunto de test baja al 76%. Por último, según la precisión de Validación Cruzada el rendimiento general del modelo es del 72%.

```
|--- Cuenta <= 0.50
|   |--- class: 1
|--- Cuenta >  0.50
|   |--- Duracion <= 0.27
|   |   |--- Historial <= 2.50
|   |   |   |--- class: 1
|   |   |--- Historial >  2.50
|   |   |   |--- class: 2
|   |--- Duracion >  0.27
|   |   |--- Ahorros <= 0.50
|   |   |   |--- Cantidad <= 0.07
|   |   |   |   |--- class: 2
|   |   |   |--- Cantidad >  0.07
|   |   |   |   |--- class: 1
|   |   |--- Ahorros >  0.50
|   |   |   |--- class: 2
```




Una de las ventajas del Árbol de Decisión es que se puede representar visualmente el 'hilo de pensamiento' que sigue para llegar a una conclusión. En nuestro caso, basándonos en el árbol obtenido y su estructura llegamos a las siguientes conclusiones:

- A una persona que tiene más de 200 DM en su cuenta se le suele conceder el préstamo.
- A una persona que no tiene cuenta o cuya cuenta tiene menos de 0 DM, que pide un préstamo durante unos pocos meses y cuyo historial muestra otros créditos pagados debidamente se le suele conceder el préstamo.
- A una persona que no tiene cuenta o cuya cuenta tiene menos de 0 DM, que pide un préstamo durante unos pocos meses y cuyo historial muestra otros préstamos existentes o retraso en otros pagos no se le suele conceder el préstamo.
- A una persona que no tiene cuenta o cuya cuenta tiene menos de 0 DM y pide un préstamo durante varios meses, que tiene bastantes ahorros y pide una cantidad pequeña no se le suele conceder el préstamo.
- A una persona que no tiene cuenta o cuya cuenta tiene menos de 0 DM y pide un préstamo durante varios meses, que tiene bastantes ahorros y pide una cantidad grande se le suele conceder le préstamo.

- A una persona que no tiene cuenta o cuya cuenta tiene menos de 0 DM, que pide un préstamo durante varios meses y que tiene pocos ahorros no se le suele conceder le préstamo.

Vemos que el Árbol de Decisión toma decisiones principalmente en torno a las variables de 'Cuenta', 'Duración', 'Historial', 'Ahorros' y 'Cantidad'. Algunas categorías ordinales nos las convierte a decimal, dividiendo el conjunto de valores posibles en dos para la ramificación.

## MODELO 3: REGRESIÓN LOGÍSTICA

```
----------Train Confusion Matrix----------
[[531  28]
 [197  44]]

              precision    recall  f1-score   support

           1       0.73      0.95      0.83       559
           2       0.61      0.18      0.28       241

    accuracy                           0.72       800
   macro avg       0.67      0.57      0.55       800
weighted avg       0.69      0.72      0.66       800
```

```
----------Test Confusion Matrix----------
[[133   8]
 [ 49  10]]

              precision    recall  f1-score   support

           1       0.73      0.94      0.82       141
           2       0.56      0.17      0.26        59

    accuracy                           0.71       200
   macro avg       0.64      0.56      0.54       200
weighted avg       0.68      0.71      0.66       200
```

| Data | Accuracy | Precision | Recall | F1 | Error | Std Error |
|---|---|---|---|---|---|---|
| Train | 0.72 | 0.73 | 0.95 | 0.83 | 0.28 | 0.45 |
| Test | 0.72 | 0.73 | 0.94 | 0.82 | 0.33 | 0.47 |

```
CV-Validation Accuracy:  0.7070000000000001
```

En este caso, tanto en el conjunto de entrenamiento como de test, la diferencia entre falsos negativos (FN) y falsos positivos (FP) es mayor que en el modelo anterior. Además, de nuevo, el modelo muestra mucha mayor exactitud en todas las métricas para la 1º etiqueta. La precisión de la 2º etiqueta es la más cercana a la de la 1º, lo que indica que, de las pocas instancias con dicha etiqueta, más de la mitad se clasifican de forma correcta. Recall tiene un valor muy bajo.

Por otro lado, según la precisión de entrenamiento y test, el 73% de los individuos clasificados como Apto realmente lo eran. Mientras tanto, según la medida de recuperación (Recall) el 95% de los individuos fueron clasificados como Aptos correctamente en el conjunto de entrenamiento y en el de test un 94%. Además, según la exactitud, en general, el 72%. Por último, según la precisión de Validación Cruzada el rendimiento general del modelo es del 71%.

## MODELO 4: SVM (Support Vector Machines)

```
----------Train Confusion Matrix----------
[[538  21]
 [195  46]]

              precision    recall  f1-score   support

           1       0.73      0.96      0.83       559
           2       0.69      0.19      0.30       241

    accuracy                           0.73       800
   macro avg       0.71      0.58      0.57       800
weighted avg       0.72      0.73      0.67       800
```

```
----------Test Confusion Matrix----------
[[137   4]
 [ 50   9]]

              precision    recall  f1-score   support

           1       0.73      0.97      0.84       141
           2       0.69      0.15      0.25        59

    accuracy                           0.73       200
   macro avg       0.71      0.56      0.54       200
weighted avg       0.72      0.73      0.66       200
```

| Data | Accuracy | Precision | Recall | F1 | Error | Std Error |
|---|---|---|---|---|---|---|
| Train | 0.73 | 0.73 | 0.96 | 0.83 | 0.27 | 0.44 |
| Test | 0.73 | 0.73 | 0.97 | 0.84 | 0.32 | 0.47 |

```
CV-Validation Accuracy:  0.713
```

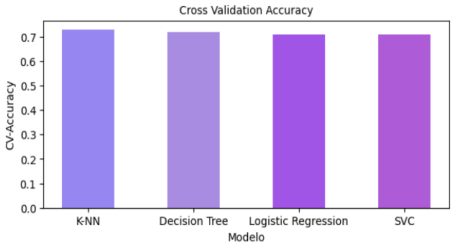
Tanto en el conjunto de entrenamiento como de test, la diferencia entre falsos negativos (FN) y falsos positivos (FP) es considerable. Además, de nuevo, el modelo muestra mucha mayor exactitud en todas las métricas para la 1º etiqueta. La precisión de la 2º etiqueta es la más cercana a la de la 1º, lo que indica que, de las pocas instancias con dicha etiqueta, más de la mitad se clasifican de forma correcta. Recall, de nuevo, tiene un valor muy bajo.

Por otro lado, según la precisión de entrenamiento y test, el 73% de los individuos clasificados como Apto realmente lo eran. Mientras tanto, según la medida de recuperación (Recall) el 96% de los individuos fueron clasificados como Aptos correctamente en el conjunto de entrenamiento y en el de test un 97%. Además, según la exactitud, en general, el 73%. Por último, según la precisión de Validación Cruzada el rendimiento general del modelo es del 71%.

## 4. DISCUSIÓN – COMPARACIÓN DE MODELOS

### PRECISIÓN VALIDACIÓN CRUZADA

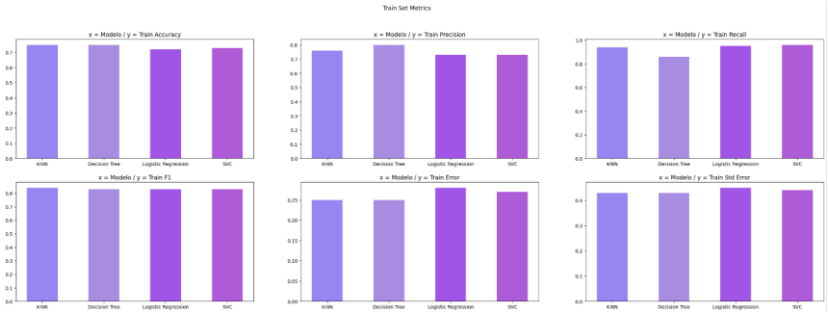
| Modelo | CV-Accuracy |
|---|---|
| K-NN | 0.73 |
| Decision Tree | 0.72 |
| Logistic Regression | 0.71 |
| SVC | 0.71 |




Por lo general, según los valores de Validación Cruzada obtenidos, llegamos a la conclusión de que el modelo que mejor rendimiento tiene parece ser el 1º, ajustado mediante el algoritmo K-NN. No obstante, los valores son muy parecidos por lo que tampoco podemos asegurar, únicamente mediante esta métrica, cuál es mejor.

### MÉTRICAS DEL CONJUNTO DE ENTRENAMIENTO

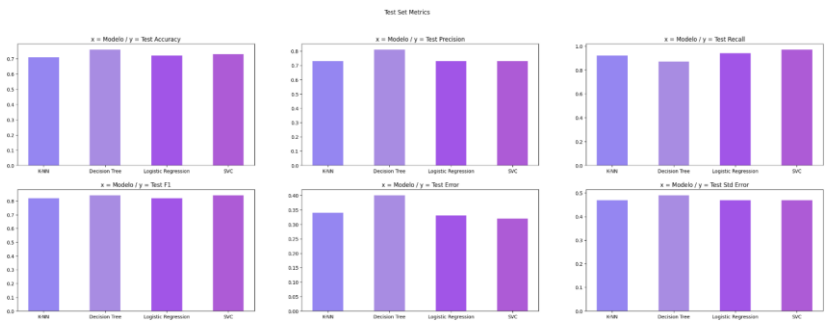
| Modelo | Train Accuracy | Train Precision | Train Recall | Train F1 | Train Error | Train Standard Error |
|---|---|---|---|---|---|---|
| K-NN | 0.75 | 0.76 | 0.94 | 0.84 | 0.25 | 0.43 |
| Decision Tree | 0.75 | 0.80 | 0.86 | 0.83 | 0.25 | 0.43 |
| Logistic Regression | 0.72 | 0.73 | 0.95 | 0.83 | 0.28 | 0.45 |
| SVC | 0.73 | 0.73 | 0.96 | 0.83 | 0.27 | 0.44 |




Ahora, si nos fijamos en la evaluación del conjunto de entrenamiento vemos que el modelo que mayor precisión tiene es el Árbol de Decisión, pero es el que menor Recall muestra, aunque no de forma significativa. No obstante, vemos que tiene un valor F1 considerable, lo cual significa que la precisión y la recuperación están equilibradas. La segunda mejor opción podría ser el 2º modelo, SVM, ya que muestra un alto Recall y F1. El modelo de Regresión Logístia muestra valores similares. El modelo K-NN también es una buena opción si nos fijamos más concretamente en las precisiones y en la exactitud.

### MÉTRICAS DEL CONJUNTO DE TEST

| Modelo | Test Accuracy | Test Precision | Test Recall | Test F1 | Test Error | Test Standard Error |
|---|---|---|---|---|---|---|
| K-NN | 0.71 | 0.73 | 0.92 | 0.82 | 0.34 | 0.47 |
| Decision Tree | 0.76 | 0.81 | 0.87 | 0.84 | 0.40 | 0.49 |
| Logistic Regression | 0.72 | 0.73 | 0.94 | 0.82 | 0.33 | 0.47 |
| SVC | 0.73 | 0.73 | 0.97 | 0.84 | 0.32 | 0.47 |



Y, por último, si nos fijamos en los datos de test, el que muestra mayor precisión y exactitud es, de nuevo, el Árbol de Decisión, a excepción del Recall donde, al igual que antes, tiene el valor más bajo, aunque no de forma significativa. No obstante, vemos que tiene un valor F1 considerable, lo cual significa que la precisión y la recuperación están equilibradas. Los otros 3 modelos muestran valores similares.

## 5. CONCLUSIONES

En conclusión, como no podemos utilizar Validación Cruzada para determinar qué modelo es mejor nos fijaremos en las métricas de evaluación del conjunto de entrenamiento y de test. Tras estudiarlas detenidamente, determinamos que el modelo con el que hemos obtenido un mejor rendimiento para nuestros datos es el Árbol de Decisión.

Como explicamos al principio, en nuestro caso de estudio concreto la precisión es una medida de gran importancia. Nos asegura que el número de falsos positivos sea mínimo, es decir, que el número de personas clasificadas como 'Aptas' sin serlo sea lo más bajo posible. Es más costoso realizar un préstamo que no se va a poder pagar que rechazar unos pocos casos viables. Es decir, considerando la economía del banco, la precisión es mucho más importante que, incluso, la métrica de recuperación, Recall, que se centra en aquellos casos 'Aptos' calificados como 'No Aptos'.

El 2º modelo tiene mayor precisión y exactitud que el resto, tanto en el conjunto de test como en el de entrenamiento, y el mayor valor de F1, es decir que, aunque Recall sea el más bajo, es el modelo que mejor equilibrio consigue entre ambas métricas.

Además, como ya hemos comentado, al tener una mayor proporción de datos con la etiqueta 1 es normal que modelos como SVM tengan la métrica de Recall tan alta, al haber más datos que se clasifiquen como positivos. Es decir, podría significar que el modelo está sobreajustado. Por ello podemos concluir que el modelo 2 también es el que mayor capacidad de generalización tiene, óptimo para el desbalance de nuestros datos.

Sin embargo, aunque es cierto que el mejor modelo es el Árbol de Decisión en general no hemos obtenido una mejora significativa con respecto a la precisión base de la que partíamos. Para mejorar nuestro modelo podríamos:

- Utilizar muestras ponderadas para manejar el desequilibrio de clases.
- Hacer un estudio más profundo de los hiperparámetros óptimos.
- Realizar selección de variables, ya que hemos visto que el árbol se basa solo en unas pocas.
- Utilizar métodos como Random Forest que genera múltiples árboles y, como hemos visto en las precisiones base, es el algoritmo que mejor resultados obtiene.

## 6. REFERENCIAS

[1] DATA-SET: https://archive.ics.uci.edu/dataset/144/statlog+german+credit+data

[2] BÚSQUEDA MANUAL DE CCP: https://www.kaggle.com/code/arunmohan003/pruning-decision-trees-tutorial#1.-Pre-pruning-techniques