Maestría en Explotación de Datos y Descubrimiento del Conocimiento

Trabajo Practico N°1

Materia: Aprendizaje Automático

Integrantes:

* Martin Llada
* Rodrigo Mauriño
* Juan Jose Iguaran

Nro. de grupo:

Fecha de entrega: 27/05/2019

Índice

1. Resumen 3
2. Introducción 3
3. Datos 4
4. Metodología 6
5. Resultados 8
6. Conclusiones 12
7. Resumen

En este trabajo se implementa un modelo de Árbol de decisión con el objeto de predecir el otorgamiento (o no) de préstamos hipotecarios en el Estado de Nueva York para el año 2015. A partir de entrenar diferentes modelos -combinando diferentes niveles de altura y valores para los hiperparámetros- se arriba a que el árbol con una profundidad de 6 y que utiliza, como criterio de selección de atributo el método Gini gain reporta los mejores resultados relativos en término de la performance. Esto se evidencia al computar las medidas de accuracy (en promedio, 82%) y ROC AUC (en promedio, 78%). Por otro lado, se encuentra que este árbol reduce levemente su capacidad de predicción cuando el atributo más relevante en el conjunto de datos, posee datos faltantes o ruido. Por último, con el objeto de comparar este método de clasificación con otra técnica, se comparó la performance del árbol de decisión mencionado con el método Naive Bayes con distribución de probabilidad Bernoulli. Si bien, ambos modelos presentan buenas medidas de desempeño (utilizando ROC AUC), se evidencia que para el conjunto de datos de validación el algoritmo de Naive Bayes posee una media más elevada y un menor desvío estándar.

1. Introducción

El objetivo del trabajo es predecir y analizar otorgamientos de préstamos hipotecarios en el estado de Nueva York en 2015 utilizando herramientas en Python.

Se utilizaron algoritmos predictivos de Arboles de Decisión y Naive Bayes para clasificar. Para cada modelo se analizó el desempeño frente a la ocurrencia de datos faltantes generados y tolerancia al ruido.

El Dataset utilizado en el proyecto proviene del portal Kaggle[[1]](#footnote-1) y es de dominio público. Es una fracción del conjunto de datos original de créditos hipotecarios otorgados desde el 2007 hasta la actualidad para todos los estados de Estados Unidos[[2]](#footnote-2). Estos reportes son confeccionados por la entidad Consumer Finance Protection Board (CFPB).

Se tomaron en cuenta distintas variables relacionadas con el solicitante del préstamo como la situación económica, género, etnicidad, estado habitacional del inmueble, objetivo de la hipoteca y ubicación además de datos respecto a la agencia inmobiliaria, entre otros atributos. Si bien inicialmente se contaron con 6 resultados distintos que clasificaban la decisión tomada para cada caso, el análisis se hizo sobre el agrupamiento de estos 6 resultados en 2 posibles clasificaciones: Otorgado y No Otorgado.

1. Datos

El Dataset original para el estudio posee 439.655 casos, 77 atributos y 1 atributo clasificatorio que puede tomar 6 valores distintos que representan la acción a tomar respecto al crédito hipotecario. Con el objetivo de simplificar el análisis se substrajo al azar mediante una distribución uniforme un subset menor al 1% del total de casos, dejando solamente 3478 casos para estudiar.

Inicialmente se aplicó un pre filtro para mejorar la calidad de datos y preservar únicamente los atributos más relevantes para el estudio. Con el pre filtro se eliminaron atributos redundantes, atributos constantes, número identificador de cada caso y columnas que contenían principalmente datos faltantes. Se presentan en la **Tabla 1** los 28 atributos utilizados en el análisis.

|  |  |  |  |
| --- | --- | --- | --- |
| **Columna** | **Descripción** | **Tipo de Dato** | **Clasificación de datos** |
| agency\_code | Agencia inmobiliaria | Numérico | Categórico |
| applicant\_ethnicity | Etnicidad de solicitante | Numérico | Categórico |
| applicant\_income\_000s | Ingresos de solicitante en miles | Numérico | Continuo |
| applicant\_race\_1 | Raza de solicitante | Numérico | Categórico |
| applicant\_sex | Genero de solicitante | Numérico | Categórico |
| application\_date\_indicator | Indicador fecha de aplicación | Numérico | Categórico |
| census\_tract\_number | Número de Censo | Numérico | Categórico |
| co\_applicant\_ethnicity | Etnicidad de co-solicitante | Numérico | Categórico |
| co\_applicant\_race\_1 | Raza de co-solicitante | Numérico | Categórico |
| co\_applicant\_sex | Genero de co-solicitante | Numérico | Categórico |
| county\_name | Nombre de condado | String | Categórico |
| hoepa\_status | Estado hoepa | Numérico | Categórico |
| lien\_status | Estado de derecho de retención | Numérico | Categórico |
| loan\_purpose | Propósito de préstamo | Numérico | Categórico |
| loan\_type | Tipo de préstamo | Numérico | Categórico |
| msamd | msamd | Numérico | Categórico |
| owner\_occupancy | Ocupación de solicitante | Numérico | Categórico |
| preapproval | Pre aprobación | Numérico | Categórico |
| property\_type | Tipo de Propiedad | Numérico | Categórico |
| purchaser\_type | Tipo de comprador | Numérico | Categórico |
| sequence\_number | Número secuencial | Numérico | Categórico |
| hud\_median\_family\_income | Ingreso medio de familia | Numérico | Continuo |
| loan\_amount\_000s | Valor de préstamo en miles | Numérico | Continuo |
| number\_of\_1\_to\_4\_family\_units | Unidades familiares | Numérico | Continuo |
| number\_of\_owner\_occupied\_units | Unidades ocupadas | Numérico | Continuo |
| minority\_population | Población minoritaria | Numérico | Continuo |
| population | Población | Numérico | Continuo |
| tract\_to\_msamd\_income | Ingreso msamd | Numérico | Continuo |

(Tabla 1: Atributos estudiados)

Como se puede observar el Dataset cuenta con una amplia variación datos de distinta naturaleza: 8 datos continuos y 20 datos categóricos, uno de los cuales se presenta como Tipo de dato String, county\_name. Debido a los procesos clasificatorios que vamos a someter a este Dataset es conveniente transformar la columna county\_name en un dato Numérico. Por lo cual se procedió a realizar un Label Encoder en la columna county\_name y se transformó cada valor distinto en String a un valor numérico utilizando números enteros consecutivos.

La columna del atributo clasificatorio se llama action\_taken y presenta valores categóricos numéricos, a continuación se presentan en la Tabla 2:

|  |  |  |  |
| --- | --- | --- | --- |
| **action\_taken** | **Descripción** | **Cantidad** | **Proporción** |
| 1 | Loan originated | 1951 | 56.10% |
| 2 | Application approved but not accepted | 129 | 3.71% |
| 3 | Application denied by financial institution | 630 | 18.11% |
| 4 | Application withdrawn by applicant | 353 | 10.15% |
| 5 | File closed for incompleteness | 142 | 4.08% |
| 6 | Loan purchased by the institution | 273 | 7.85% |

(Tabla 2: Clases originales)

Hay una claro desbalance entre las 6 posibles clases, donde la primera clase tiene casi la misma cantidad de casos que la suma del resto. Nuevamente por la naturaleza de los modelos predictivos que vamos a usar en el análisis de los datos debemos transformar la clase action\_taken en valores binarios. Se tomó el criterio de clasificar los códigos 1,5 y 6 como prestamos aprobado y el resto de los casos representan el caso contrario.

|  |  |  |  |
| --- | --- | --- | --- |
| **action\_taken** | **Descripción** | **Cantidad** | **Proporción** |
| 0 | Aprobado | 2366 | 68.03% |
| 1 | No Aprobado | 1112 | 31.97% |

(Tabla 3: Clases usadas para el estudio)

Una vez aplicada la transformación vemos ahora un leve desbalance hacia la clase “Aprobado”.

1. Metodología

Para realizar el estudio predictivo se realizó una partición del Dataset en datos de entrenamiento y datos de testing en un 80% y 20% respectivamente. En los datos de entrenamiento realizaremos la mayoría de pruebas y ajustes de los métodos predictivos así como el análisis de la tolerancia de los modelos a datos faltantes y datos ruidosos. El ajuste de los algoritmos consiste principalmente en la variación de sus hiperparametros y la evaluación de distintos métricas para encontrar la mejor configuración del algoritmo a este problema. Mientras que el subset de testing se usara para verificar la consistencia de dichos modelos con datos nuevos que jamás se vieron durante el entrenamiento. El objetivo de este análisis de consistencia es verificar que el modelo no está sobre ajustando las predicciones sobre el set de entrenamiento. En el caso que nos encontremos con un caso de sobre ajuste podremos observar una diferencia apreciable entre los score obtenidos al evaluar los subsets de entrenamiento y testing.

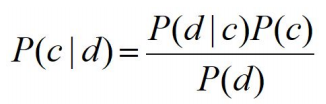
Al mismo tiempo el set de entrenamiento se dividirá en segmentos de validación para aplicar un 5-fold cross validation. Este método consiste en tomar 5 copias del set de entrenamiento y para caso tomar un set de validación distinto y disjuntos entre sí, en el actual trabajo se tomó un 20% de la copia de enteramiento. En cada copia el algoritmo se entrenara con el conjunto restante del set y se evaluara sobre el validation. Finalmente se tomara el promedio del score obtenido de los 5 sets de validation y se analizará del desvió estándar en cada caso para evaluar la consistencia. El objetivo final de este proceso es poder analizar el desempeño de distintas configuraciones del algoritmo sin tener que utilizar datos del set de testing.

A continuación se describen los algoritmos predictivos utilizados en el proyecto: Arboles de decisión y Naive Bayes.

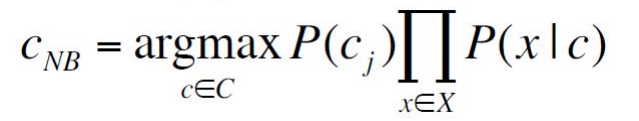
Arboles de decisión es un algoritmo basado en la disyunción de conjunciones, aplicando reglas if-then, de los posibles casos que nos podemos encontrar en un Dataset, Es un método de Machine Learninng predictivo apto tanto para casos clasificatorios y regresivos y es característico por plantear soluciones fáciles de interpretar. Este algoritmo es ideal para proyectos de estas características dado que es robusto al ruido, datos faltantes y acepta datos continuos. Para aceptar datos continuos es necesario discretizar las variables en rangos discretos mediante la aplicación de un umbral, con el que criterio que la partición de la variables maximice la reducción de la impureza. Un árbol de decisión construido posee un atributo raíz, de máxima reducción impureza, ramas que se unen a otros nodos representados por otros algoritmos y hojas al final donde se encuentra la decisión final de cada camino. Las desventajas de este algoritmo están relacionadas con su simpleza. Al analizar independientemente cada nivel del árbol para seleccionar los atributos de los nodos, no se toman en cuenta elecciones tomadas en otros niveles por lo que el árbol solo es capaz de encontrar óptimos locales pero no globales. Además, como en la mayoría de los modelos predictivos posee un sesgo inductivo.

Los hiperparametros de este algoritmo más relevantes para este estudio son el criterio de selección de atributo para elegir los nodos y la altura máxima del árbol. El criterio para elegir los nodos se basa en algoritmos que analizan la impureza de cada variable frente a la cantidad clases, Aprobado y No Aprobado. Siendo un atributo con igual cantidad de tipos de casos considerados de máxima impureza, el árbol prefiere la selección de atributos de baja impureza cerca de la raíz. Los criterios más usados son Information Gain y Gini Gain, y en el trabajo se analizara la ventaja de utilizar uno u el otro. En cuanto a la altura máxima del árbol, es un hiperparametro usado para limitar el crecimiento excesivo y es la herramienta principal de este algoritmo para evitar el sobreajuste. Al limitar al crecimiento del árbol se puede evitar la sobre especificación del entrenamiento y mantener una generalización suficiente para que pueda evaluar exitosamente casos que no se vieron el entrenamiento.

El algoritmo de Naive-Bayes es un método predictivo probalístico utilizado para problemas clasificatorios. Consiste en calcular las probabilidades marginales de cada atributo en cada clase y analizar para que clase el caso dado tiene mayor probabilidad de pertenecer. La mayor desventaja de este proceso es que estamos asumiendo independencia entre los atributos. Por ejemplo dado la clase c y el atributo D con valores d:



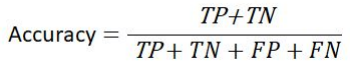
Dado un caso con varios atributos y varias clases podemos resumir la ecuación como:



Donde el resultado sería la clase del atributo X.

Una vez definidos la metodología de entrenamiento y los modelos disponibles es necesario definir las métricas que vamos utilizar en el proyecto. Vamos a utilizar la métrica estándar Accuraccy y la métrica ROC-AUC y evaluar diferencias.

La métrica Acuraccy consiste en calcular el porcentaje de Clasificaciones positivas, tanto falsas como verdaderas, por sobre el total de clasificaciones realizadas.

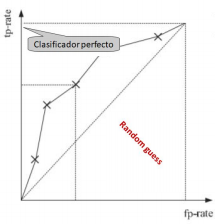


Es una métrica simple y popular para realizar los tanteos iniciales para modelos predictivos. La desventaja que posee es que no da ninguna información de los errores tipo que podríamos estar cometiendo dado que no distingue entre verdaderos positivos y falsos positivos.

Por otro lado la ROC-AUC es una métrica más compleja que toma en consideración tanto los verdaderos positivos como los falsos positivos. La métrica consiste en primero calcular los scores FPR y TPR aplicando un umbral de aceptación variables entre 0% y 100%.



Los resultados para cada umbral aplicado se grafican con ambos scores como ejes y se dibuja una curva.



Finalmente se calcula el área debajo de la curva como la métrica. El objetivo de la métrica es considerar un valor que represente un compromiso entre ambos scores mencionados.

Para el análisis de datos faltantes vamos a generar al azar este fenómeno en los atributos para luego imputarles valores mediante la moda y la moda por clase. Respecto a la tolerancia al ruido procederemos de forma similar, generando valores ruidos que varian desde un 0% a un 35 del valor inicial modificado y analizaremos como impacta en el estudio.

1. Resultados

Análisis del 5-fold cross validation del Árbol de decisión con máxima altura = 3 y criterio de selección Gini Gain para ambas métricas.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Acurracy** | | **ROC** | |
|  | **Training** | **Validation** | **Training** | **Validation** |
|  | 79.40% | 80.80% | 74.20% | 71.00% |
|  | 78.80% | 78.10% | 78.50% | 80.90% |
|  | 78.20% | 77.90% | 78.60% | 74.50% |
|  | 80.40% | 77.20% | 78.20% | 78.90% |
|  | 78.70% | 78.60% | 74.40% | 69.50% |
| **Promedio** | **79.10%** | **78.52%** | **76.78%** | **74.96%** |
| **Desviación** | 0.008 | 0.012 | 0.02 | 0.044 |

(Tabla 3)

Comparación del análisis del 5-fold cross validation del Árbol de decisión con máxima altura = 3 y criterio de selección Information Gain y Gini Gain para ambas métricas.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Max. Altura = 3 y Gini Gain | | | | Max. Altura = 3 y Information Gain | | | |
| **Acurracy** | | **ROC** | | **Acurracy** | | **ROC** | |
| **Training** | **Validation** | **Training** | **Validation** | **Training** | **Validation** | **Training** | **Validation** |
| 79.40% | 80.80% | 74.20% | 71.00% | 78.70% | 76.10% | 78.40% | 76.20% |
| 78.80% | 78.10% | 78.50% | 80.90% | 78.30% | 80.80% | 79.50% | 77.70% |
| 78.20% | 77.90% | 78.60% | 74.50% | 79.00% | 78.10% | 73.30% | 73.20% |
| 80.40% | 77.20% | 78.20% | 78.90% | 80.20% | 77.70% | 79.10% | 77.50% |
| 78.70% | 78.60% | 74.40% | 69.50% | 78.90% | 78.60% | 79.60% | 78.50% |

(Tabla 4)

Comparación del análisis del 5-fold cross validation del Árbol de decisión con máxima altura = 6 y criterio de selección Information Gain y Gini Gain para ambas métricas.

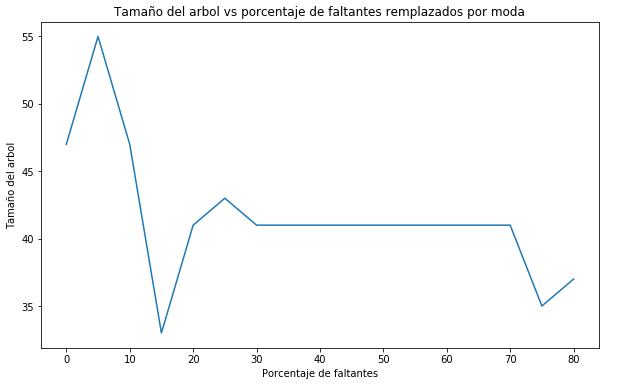
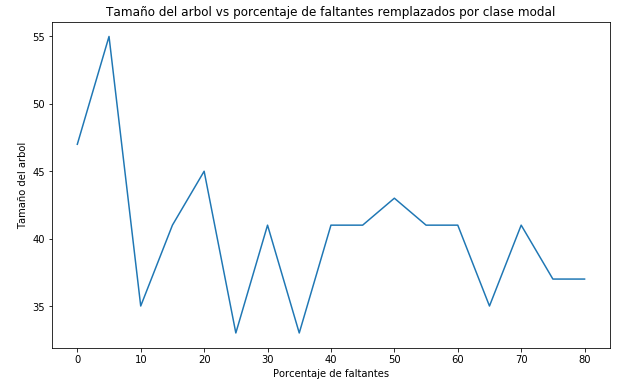
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Max. Altura = 6 y Gini Gain | | | | Max. Altura = 6 y Information Gain | | | |
| **Acurracy** | | **ROC** | | **Acurracy** | | **ROC** | |
| **Training** | **Validation** | **Training** | **Validation** | **Training** | **Validation** | **Training** | **Validation** |
| 85.10% | 81.50% | 83.70% | 77.70% | 84.60% | 81.10% | 81.10% | 76.10% |
| 84.80% | 83.50% | 84.30% | 79.30% | 85.70% | 82.00% | 83.00% | 79.90% |
| 84.90% | 81.80% | 83.10% | 74.60% | 84.40% | 84.50% | 82.60% | 77.10% |
| 86.60% | 79.90% | 82.80% | 81.00% | 85.60% | 80.60% | 81.70% | 76.70% |
| 85.30% | 81.50% | 83.10% | 77.70% | 84.60% | 79.50% | 80.80% | 79.10% |

(Tabla 5)

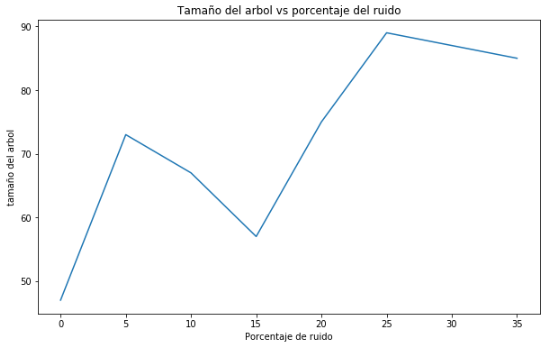
Comparación del análisis del 5-fold cross validation del Árbol de decisión con máxima altura sin límite y criterio de selección Information Gain y Gini Gain para ambas métricas.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Gini Gain | | | | Information Gain | | | |
| **Acurracy** | | **ROC** | | **Acurracy** | | **ROC** | |
| **Training** | **Validation** | **Training** | **Validation** | **Training** | **Validation** | **Training** | **Validation** |
| 100.00% | 78.20% | 100.00% | 72.80% | 100.00% | 78.40% | 100.00% | 74.00% |
| 100.00% | 79.30% | 100.00% | 73.10% | 100.00% | 76.10% | 100.00% | 73.40% |
| 100.00% | 75.20% | 100.00% | 75.00% | 100.00% | 78.80% | 100.00% | 73.50% |
| 100.00% | 76.80% | 100.00% | 71.00% | 100.00% | 77.90% | 100.00% | 71.00% |
| 100.00% | 76.80% | 100.00% | 72.60% | 100.00% | 76.80% | 100.00% | 74.20% |

(Tabla 6)

Análisis del impacto de datos faltantes imputados por moda y clase modal en el tamaño del árbol:

(Gráfico 1) (Gráfico 2)

Análisis del impacto de datos faltantes imputados por moda y clase modal en el tamaño del árbol en el score con un árbol de Máxima profundidad = 6 y criterio Gini Gain.

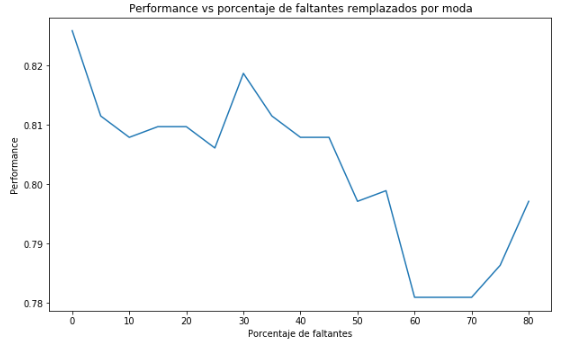
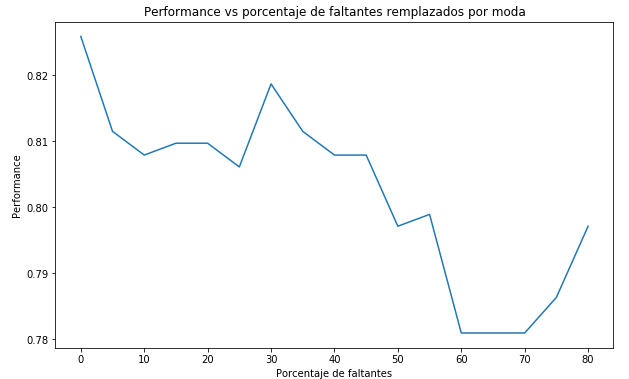
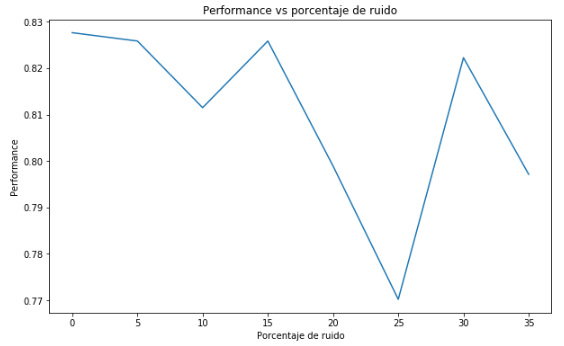
(Gráfico 3) (Gráfico 4)

Gráfico del tamaño del árbol máximo en función de la cantidad de datos ruidos introducidos en el Dataset



(Gráfico 5)

Gráfico performance medida en accuracy de un árbol con máxima altura= 6 y Gini Gain



(Gráfico 6)

Evaluación de 5-fold cross validation utilizando Naive-Bayes con métrica ROC

|  |  |  |
| --- | --- | --- |
|  | **Training** | **Validation** |
|  | 82.60% | 80.10% |
|  | 82.20% | 81.40% |
|  | 82.30% | 81.00% |
|  | 81.70% | 83.50% |
|  | 81.50% | 84.30% |
| **Promedio** | **82.06%** | **82.06%** |
| **Desviacion** | 0.005 | 0.018 |

(Tabla 7)

Comparación de Scores obtenidas con el algoritmo de Naive Bayes y Arboles de decisión (Máxima altura =6, criterio = Information gain). Primero se entrenaron en el Training y se evaluó en Validation, luego se entrenó en Desarrollo y se evaluó en Test.

|  |  |  |
| --- | --- | --- |
|  | Árbol de Decisión | Naive Bayes |
| Training | 0.828 | 0.7914 |
| Validation | 0.8204 | 0.8204 |
| Test | 0.806 | 0.815 |

(Tabla 8)

1. Conclusiones

Partiendo del análisis inicial para la primera configuración de hiperparametros, **Tabla 3,** se puede apreciar que para ambas métricas, la media es más alta y la desviación estándar es más baja para los datos de Training. Esto parece indicar -y de manera consistente con lo esperado- que el modelo se comporta mejor para el conjunto de datos con los cuales fue entrenado. Por otro lado, también se evidencia que la medida de acucracy es significativamente mejor en todos los casos que el ROC-AUC. Esta diferencia no necesariamente indica que la Accuracy es mejor que la otra métrica, para llegar a una conclusión sobre que métrica es más conveniente para este estudio habría que analizar la naturaleza del problema.

Analizando las **Tablas 3-6** podemos ver como varían los resultados en función de los hiperparametros que hemos utilizado: Máximo Altura y Criterio. En cuanto al criterio utilizado no podemos observar una diferencia apreciable entre el uso de Information Gain o Gini Gain, por lo que se concluye que no existe gran diferencia en cuanto a criterio de selección. Lo opuesto ocurre con la Altura Máxima del árbol, para una Máxima altura de 3 obtenemos valores alrededor del 75% lo que no nos indica que podríamos estar en un caso de underfitting. Si vemos el otro extremo de no poner límite a la altura del árbol caemos en un claro caso de overfitting, con Scores de 100% en los datos de entrenamiento y Scores muy bajos, alrededor de 70% en el Validation. El caso medio que se analizó, con altura de 6, nos muestra que efectivamente el primer análisis de la altura estaba underfiteado porque el score de entrenamiento y validation mejoró considerablemente. Aunque acá tampoco podemos saber si esta altura es la mejor o todavía se puede mejorar, utilizando herramientas de Python para encontrar la altura perfecta para este estudio, encontramos con una altura máxima de 6 es la altura de máximo score.

Las gráficas de datos faltantes, **Gráfico 1- 2** nos demuestra que a pesar de solo apreciar leves diferencias entre las gráficas, se puede derivar una clara relación entre ambos casos: a mayor porcentaje de faltantes, disminuye el tamaño del árbol. Si ahora analizamos la performance del árbol frente a este fenómeno, **Gráfico 3-4,** vemos que a mayor porcentaje de faltantes, disminuye la performance independientemente de la estrategia de relleno. Teniendo en cuenta que la altura máxima que se uso fue de 6, de alta performance, podría significar que al introducir datos faltantes deberíamos buscar una nueva altura de máxima performance para mantener calidad de la predicción.

En los **Gráficos 5-6**, se ve que conforme aumenta el ruido en el atributo más importante, aumenta el tamaño del árbol denotando una relación directa entre estas dos variables. A sí mismo, se puede apreciar que existe una leve relación inversa entre el porcentaje del ruido y la performance del modelo. Incluso cuando el porcentaje del ruido en el atributo más importante se incrementa, la performance del algoritmo, y por lo tanto, su capacidad de predicción, se mantiene en niveles del orden del 77% (con un porcentaje de ruido igual a 25%) y 80% (cuando el porcentaje del ruido alcanza el 35%).

Al utilizar Naive Bayes para el entrenamiento, **Tabla 7,** obtenemos una mejor performance en general que los valores obtenidos para el mejor árbol pero para hacer una comparación relevante hay que evaluar ambos métodos en Testing, **Tabla 8.** Como se esperaba siendo la performance obtenida a partir del modelo Naive Bayes es mejor que aquella que se deriva del árbol de decisión.

1. <https://www.kaggle.com/jboysen/ny-home-mortgage> [↑](#footnote-ref-1)
2. <https://www.consumerfinance.gov/data-research/hmda/explore> [↑](#footnote-ref-2)