TD VI: Inteligencia Artificial

Trabajo Práctico I

Universidad Torcuato Di Tella

Participantes:

Catalina Brusco, Catalina Chab López y Belén Chen

Fecha: Abril del 2025

1. Introducción al problema: origen y variables principales del conjunto de datos

El conjunto de datos elegido proviene de Kaggle y se utiliza para predecir si un préstamo solicitado por una persona será aprobado o rechazado, basándose en distintas características del solicitante. El conjunto fue enriquecido con variables adicionales basadas en *Riesgo Financiero* para la Aprobación de Préstamos. Además, se aplicó *SMOTENC (Synthetic Minority Over-sampling Technique for Nominal and Continuous)* para generar nuevos puntos de datos y ampliar el conjunto de instancias. El conjunto de datos contiene 45,000 registros y 14 variables. A continuación se describen las variables clave:

Table 1: Resumen de los atributos del dataset

Columna	Descripción	Tipo.de.Dato
person_age	Edad de la persona	Float
person_gender	Género de la persona	Categórico
person_education	Nivel de educación más alto alcanzado	Categórico
person_income	Ingreso anual de la persona	Float
person_emp_exp	Años de experiencia laboral	Entero
person_home_ownership	Estado de propiedad de la vivienda (ej., alquiler, propia)	Categórico
loan_amnt	Monto del préstamo solicitado	Float
loan_intent	Propósito del préstamo	Categórico
loan_int_rate	Tasa de interés del préstamo	Float
loan_percent_income	Monto del préstamo como porcentaje del ingreso anual	Float
cb_person_cred_hist_length	Longitud del historial de crédito en años	Float
credit_score	Puntaje de crédito de la persona	Entero
previous_loan_defaults_on_file	Indicador de los anteriores incumplimientos de préstamo	Categórico
loan_status (objetivo)	Estado del préstamo: 1 = aprobado; 0 = rechazado	Entero (Binario)

Utilizar un árbol de decisión para modelar este problema es adecuado por varias razones. Primero, permite realizar tareas de clasificación binaria, como predecir si un préstamo será aprobado o rechazado, ya que divide los datos en función de las características más importantes. Segundo, maneja eficientemente datos mixtos, es decir, tanto numéricos (como ingresos y puntajes de crédito) como categóricos (como género y estado civil), sin necesidad de transformaciones complicadas. Tercero, la estructura de un árbol de decisión permite dividir los datos en subgrupos homogéneos, lo que ayuda a identificar clientes con características similares que pueden tener un comportamiento parecido, como los que tienen un puntaje de crédito bajo o altos ingresos. Esto es importante porque las relaciones entre características, como ingresos y puntaje de crédito, no siempre son lineales, y los árboles pueden manejar estas interacciones. Finalmente, su gran ventaja es la interpretabilidad; los árboles de decisión son fáciles de entender y explican claramente las razones detrás de cada clasificación, lo cual es fundamental en el ámbito bancario, donde se necesitan justificar las decisiones de aprobación o rechazo de préstamos.

2. Preparación de los datos

Es necesario preprocesar los datos, para lo cual realizaremos un análisis que incluya la verificación de las variables y su clasificación, la normalización o escalado de las variables, la detección de valores faltantes, la identificación de posibles anomalías y balanceado de datos.

Como se mencionó anteriormente, las variables están correctamente clasificadas según su tipo (categórico o numérico). Dado que vamos a utilizar árboles de decisión, no es necesario normalizar ni escalar los datos, ya que este modelo no depende de las magnitudes de las variables. Los árboles de decisión dividen los datos basándose en los valores específicos de las características, por lo que la escala de las variables no influye en

el desempeño del modelo. Asimismo, detallaremos que no existen valores faltantes en el conjunto de datos, como se puede observar en la Tabla 2.

Table 2: Resumen de valores nulos

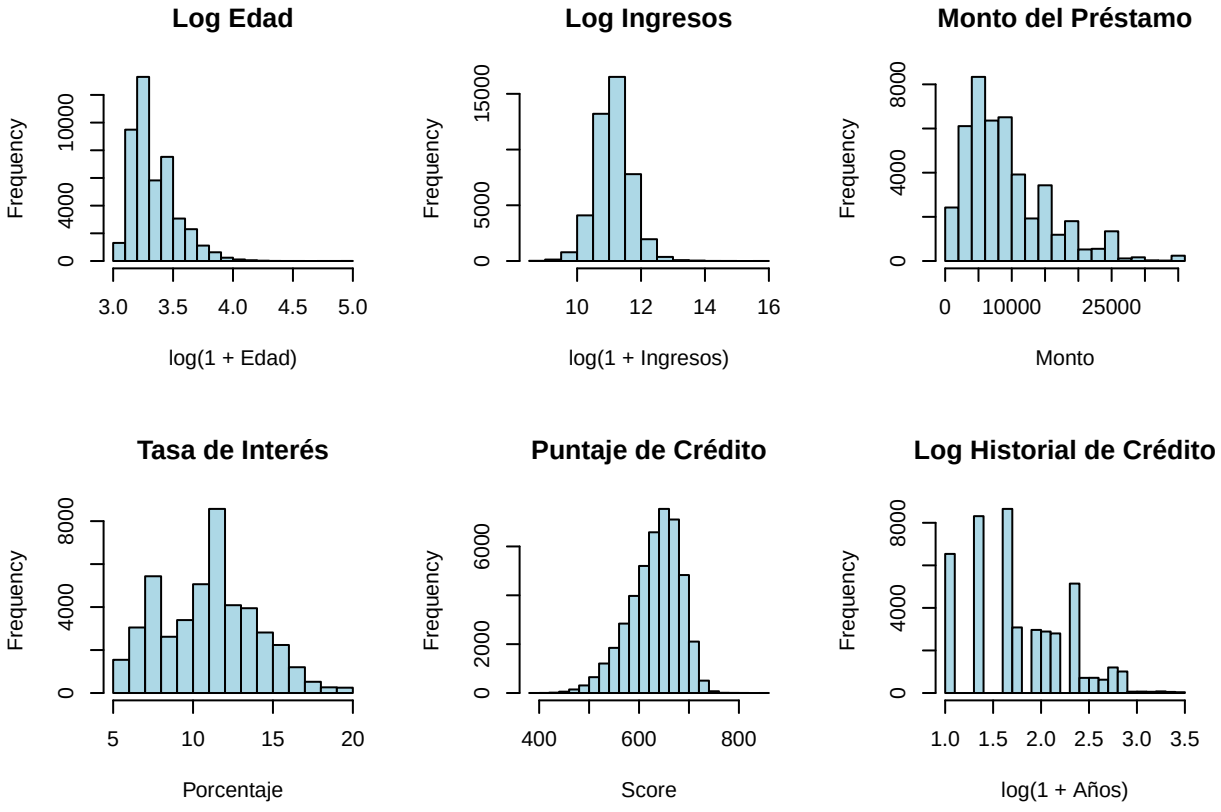
Cantidad.de.atributos.verificados	Atributos.con.0.nulos
14	14

También corroboramos que no huebiese una cantidad excesiva de filas repetidas, de hecho, nos dió cero.

Habiendo mencionado lo anterior, continuaremos con el análisis de las distribuciones de los distintos atributos. Como se puede observar en la Figura 1, se aplicó una transformación logarítmica a tres atributos clave: Edad, Ingresos e Historial de Crédito. Esta transformación es relevante porque, al ser creciente, no altera la distribución subyacente de los datos, pero sí comprime el rango de los valores grandes y expande el de los valores pequeños, lo que facilita su análisis.

Es importante destacar la relevancia de analizar la distribución de cada atributo. En este caso, observamos que atributos como Edad, Monto del Préstamo e Ingresos presentan una distribución sesgada a la derecha, mientras que el Puntaje de Crédito está sesgado a la izquierda. Esta asimetría podría generar predicciones erróneas para los valores extremos en cada uno de estos atributos. Además, notamos que el Historial de Crédito no sigue una distribución bien definida, lo que añade complejidad a su análisis.

Distribución de Variables Numéricas



Distribución de Variables Categóricas

Luego de analizar la distribución de las variables categóricas, no se identifican patrones relevantes ni desequilibrios significativos que requieran mención.

Anomalías

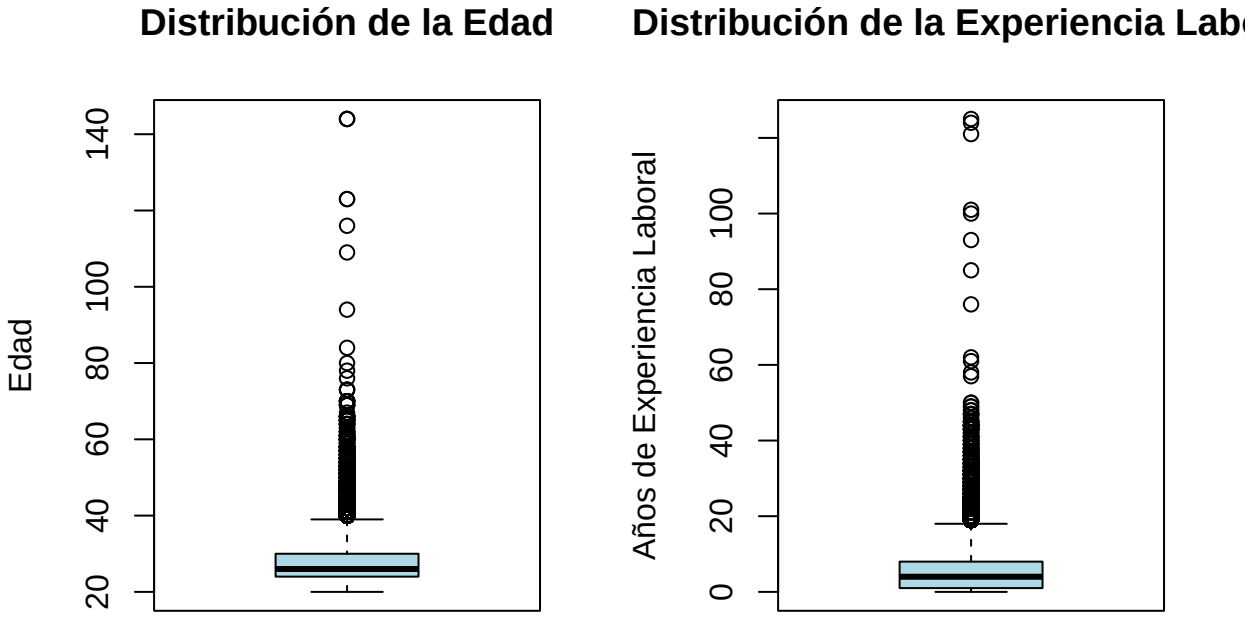


Figure 1: Boxplots de Experiencia Laboral y Edad

En el análisis preliminar de este conjunto de datos, se han identificado algunos valores atípicos (outliers) en los atributos de "Edad" (como personas con más de 120 años) y "Experiencia Laboral" (casos donde había más de 100 años de experiencia laboral). Sin embargo, su cantidad es chica (7 casos en un total de 45,000), lo que sugiere que no tendrán un impacto significativo en el entrenamiento del modelo. También, la distribución de la media en ambos Boxplots es bastante concentrada, lo que indica que la presencia de estos outliers no afecta la distribución general de los datos. En resumen, debido a que su número es tan pequeño, su impacto es prácticamente nulo, por lo que no es necesario eliminar estas filas.

Como resultado de los análisis realizados, no se eliminará ninguna información del conjunto de datos para evitar introducir sesgos en el proceso.

Chequeo de datos balanceados

Para evaluar el balance de nuestro conjunto de datos, analizamos las proporciones de las categorías de la variable de interés, en este caso, el estado del préstamo (aceptado o rechazado). A continuación, creamos un gráfico de barras que muestra la proporción de "Aceptado" y "Rechazado" en el conjunto de datos.

Al examinar el gráfico y las proporciones obtenidas, observamos que aproximadamente el 22% de los registros corresponden a "Aceptados" y el 77% a "Rechazados". Esto indica que el conjunto de datos no está completamente balanceado, aunque la desproporción no es extremadamente alta. La desproporción en el conjunto de datos podría generar un sesgo hacia la clase mayoritaria ("Rechazados"), lo que podría resultar en un mejor desempeño del modelo para predecir esta clase y un desempeño inferior para la clase minoritaria ("Aceptados").

Sin embargo, decidimos no intervenir en el conjunto de datos eliminando registros o utilizando técnicas de balanceo como sobremuestreo o submuestreo. Esto se debe a que tales técnicas podrían introducir sesgos

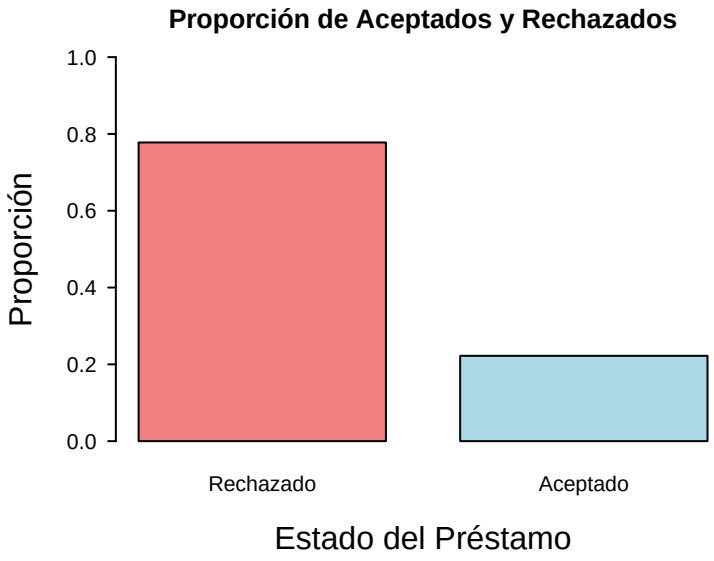


Figure 2: Proporción de Aceptados y Rechazados

adicionales o alterar la representatividad del conjunto de datos. En cambio, optamos por mantener la estructura original y tener en cuenta las proporciones de las clases al entrenar, validar y evaluar el modelo.

3. Construcción de árbol de decisión básico

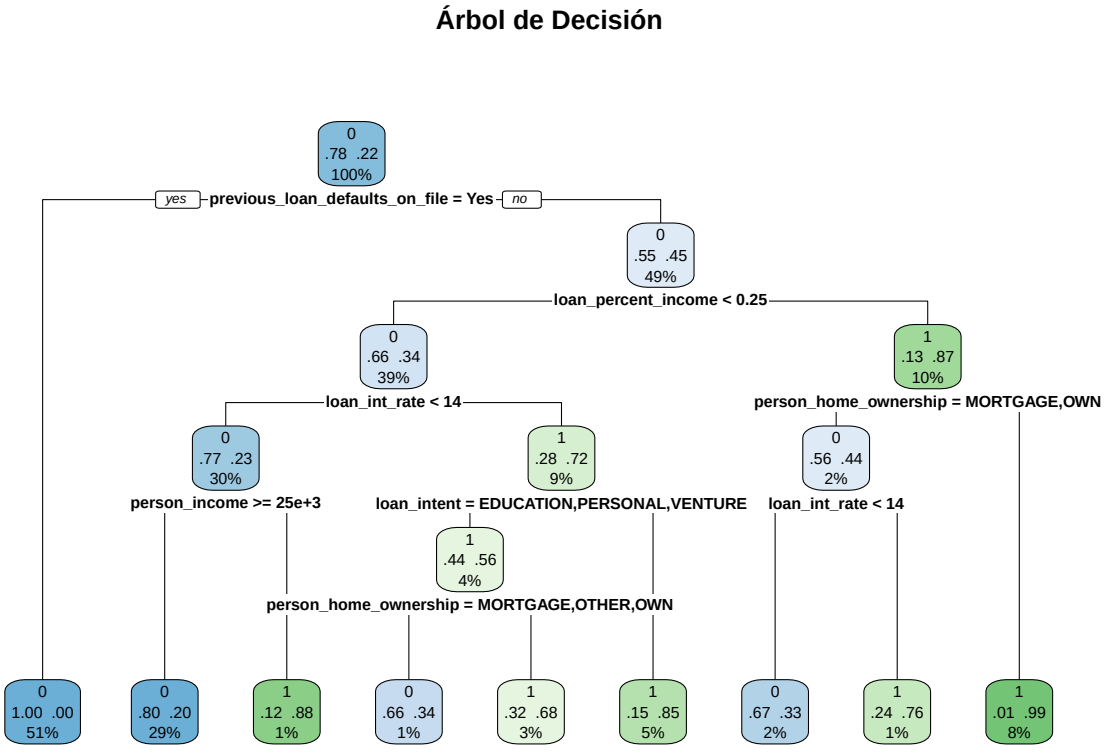


Figure 3: Árbol de Decisión del Modelo

Al realizar el análisis, mantuvimos la proporción de otorgamiento de préstamos (22%) y rechazo (78%) a lo largo de las etapas de entrenamiento, validación y testeo. En la descripción del árbol de decisión, observamos que cada nodo se compone de tres filas:

1.La primera fila indica la categoría de "no" (rechazo del préstamo).

2.La segunda fila muestra la proporción de casos dentro del subgrupo.

3.La tercera fila indica cuántos datos se concentran en ese nivel del árbol.

El árbol de decisión muestra que el primer factor que se utiliza para determinar si se le otorga un préstamo es la existencia de defaults previos (previous_loan_defaults_on_file), donde la presencia de antecedentes implica una altísima probabilidad de no pago. En ausencia de defaults, el siguiente factor relevante es la relación entre el monto solicitado y el ingreso (loan_percent_income), siendo los valores altos indicativos de mayor riesgo. Además, la tasa de interés del préstamo (loan_int_rate) cumple un rol clave: tasas superiores al 14% incrementan significativamente la probabilidad de incumplimiento. El nivel de ingresos (person_income) también influye, ya que a menor ingreso, el riesgo aumenta, especialmente combinado con tasas altas. La intención del préstamo (loan_intent), cuando es para educación, fines personales o emprendimientos, aparece asociada a mayor riesgo. Finalmente, la tenencia de vivienda propia o bajo hipoteca (person_home_ownership) contribuye a elevar la probabilidad de default en combinación con tasas altas y un elevado porcentaje préstamo/ingreso. En conjunto, estas variables permiten identificar perfiles de alto riesgo de manera clara.

Como conclusión, utilizando esta instancia del modelo entrenado con el conjunto de datos train se le rechazaría al 83% el préstamo y al resto se le otorgaría.

Hiperprámetros por defecto del árbol

A continuación se detallan los hiperparámetros utilizados por defecto en la construcción del árbol y el impacto que tiene cada uno en la estructura del modelo:

minsplit = 20: Define el tamaño mínimo de observaciones en un nodo para que el árbol considere realizar una partición. Un valor de 20 limita las divisiones a grupos con un tamaño suficientemente grande, evitando sobreajuste en nodos con pocos datos.

minbucket = 7: Indica el tamaño mínimo que pueden tener las hojas terminales. Esto asegura que cada hoja final contenga al menos 7 observaciones, promoviendo estabilidad en las predicciones.

cp = 0.01: Es el parámetro de complejidad que regula el proceso de poda. Solo se aceptan divisiones que logren mejorar la calidad del ajuste en al menos un 1%. Un valor de 0.01 representa un control intermedio, balanceando entre un árbol complejo y uno demasiado simple.

maxcompete = 4: Almacena hasta 4 splits "competidores" cercanos al mejor split en cada nodo. Esto es útil para analizar qué otras variables casi logran ser seleccionadas.

maxsurrogate = 5: Controla la cantidad máxima de variables sustitutas utilizadas cuando hay datos faltantes en la variable principal de división. La presencia de 5 surrogates mejora la capacidad del modelo para manejar datos incompletos. En nuestro caso, al no haber datos faltantes, este hiperparámetro pasa a ser irrelevante.

usesurrogate = 2: Define cómo se utilizan los surrogates. Con un valor de 2, el árbol emplea surrogate splits incluso si la variable principal está disponible, siempre que aporten mejora en el ajuste.

surrogatestyle = 0: Indica que la selección de variables sustitutas se realiza utilizando un índice de concordancia simple, priorizando velocidad y simplicidad.

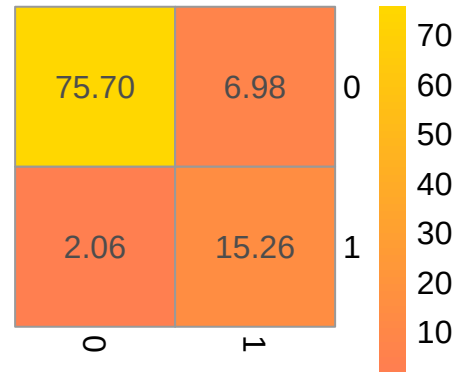
maxdepth = 30: Fija la profundidad máxima que puede alcanzar el árbol. Un valor alto de 30 permite que, si los datos y el pruning lo permiten, el árbol tenga una gran profundidad.

xval = 10: Determina que la validación cruzada interna para el proceso de poda se realice en 10 particiones, aportando mayor robustez a la selección del tamaño óptimo del árbol.

4. Evaluación del árbol de decisión básico

Se construye una matriz de confusión para analizar el rendimiento del modelo de clasificación. Inicialmente, se determina el número de predicciones positivas y negativas en el conjunto de prueba. A continuación, se define la función gen_m_c, que recibe como parámetros las etiquetas reales y las predicciones del modelo.Finalmente, los valores de la matriz se ajustan para representar la proporción de aciertos y errores, obteniendo así una versión normalizada de la matriz de confusión.

1. *TP* (Verdaderos Positivos): 75.7%
2. *TN* (Verdaderos Negativos): 15.26%
3. *FP* (Falsos Positivos): 6.98%
4. *FN* (Falsos Negativos): 2.06%



Los resultados obtenidos de las métricas de evaluación se detallan a continuación:

1. **Accuracy (Precisión Global)**: 0.9096
   - Esto significa que el modelo clasificó correctamente el 90.96% de todas las instancias en el conjunto de datos (tanto positivas como negativas).
2. **Precision (Precisión)**: 0.9156
   - Este valor indica que, de todas las predicciones que el modelo identificó como positivas (FP and TP), el 91.56% fueron efectivamente positivas. Como es un valor alto de precisión, esto significa que hay pocos falsos positivos.
3. **Recall (Sensibilidad o Exhaustividad)**: 0.9735
   - Esto significa que el modelo fue capaz de identificar correctamente el 97.35% de todas las instancias positivas reales (FN y TP) en el conjunto de datos. Como es un valor alto de recall, esto significa que el modelo tiene una buena capacidad para identificar casos positivos, aunque pueda tener algunos falsos positivos.
4. **F1-score**: 0.9437
   - El F1-score es 0.9437 y es el promedio ponderado de la precisión y el recall.Este valor es especialmente útil cuando hay un desbalance en las clases, ya que captura tanto la exactitud como la exhaustividad del modelo. Como es un F1-score alto, el modelo tiene un buen balance entre precisión y recall. Si precision y recall tienen valores similares, significa que el modelo no favorece en exceso una métrica sobre la otra, es decir, no sacrifica la detección de positivos (recall) ni la precisión en sus predicciones.
5. **AUC-ROC**: 0.9415
   - El AUC-ROC (Área bajo la curva ROC) es 0.9415, lo que indica un rendimiento muy bueno del modelo (es cercano a 1) ya que la curva ROC compara la tasa de verdaderos positivos con la tasa de falsos positivos.

En términos de la curva ROC que obtenemos, efectivamente mostramos que nuestro modelo cuenta con alta sensibilidad (True Positive Rate) y especificidad (False Positive Rate), corroborando el hecho de que la curva azul se aproxime a la esquina de la parte superior izquierda, donde el modelo clasificaría correctamente a muchos de los positivos y a los negativos.

Si dicha curva ocupase valores más próximos a la diagonal gris, significaría que el modelo no se comporta mejor que el de una clasificación aleatoria. En el caso ideal, donde el modelo es perfecto, tendríamos la curva pegada a los ejes, de forma que obtendríamos una AUC de 1, es decir, la curva podría ir hacia el destino óptimo de manera lineal.

Además, la curva ROC en los datos de test indica que el modelo mantiene un buen desempeño fuera de los datos de entrenamiento, lo que sugiere que no está sobreajustado. Si el modelo estuviera fuertemente sobreajustado, esperaríamos una alta performance en entrenamiento pero un deterioro significativo en test. Dado que la curva en test sigue mostrando una buena separación entre clases, se puede inferir que el modelo generaliza correctamente. En la siguiente parte, veremos cómo para mejorar el rendimiento de este modelo se podrían buscar mejores hiperparámetros que los default.

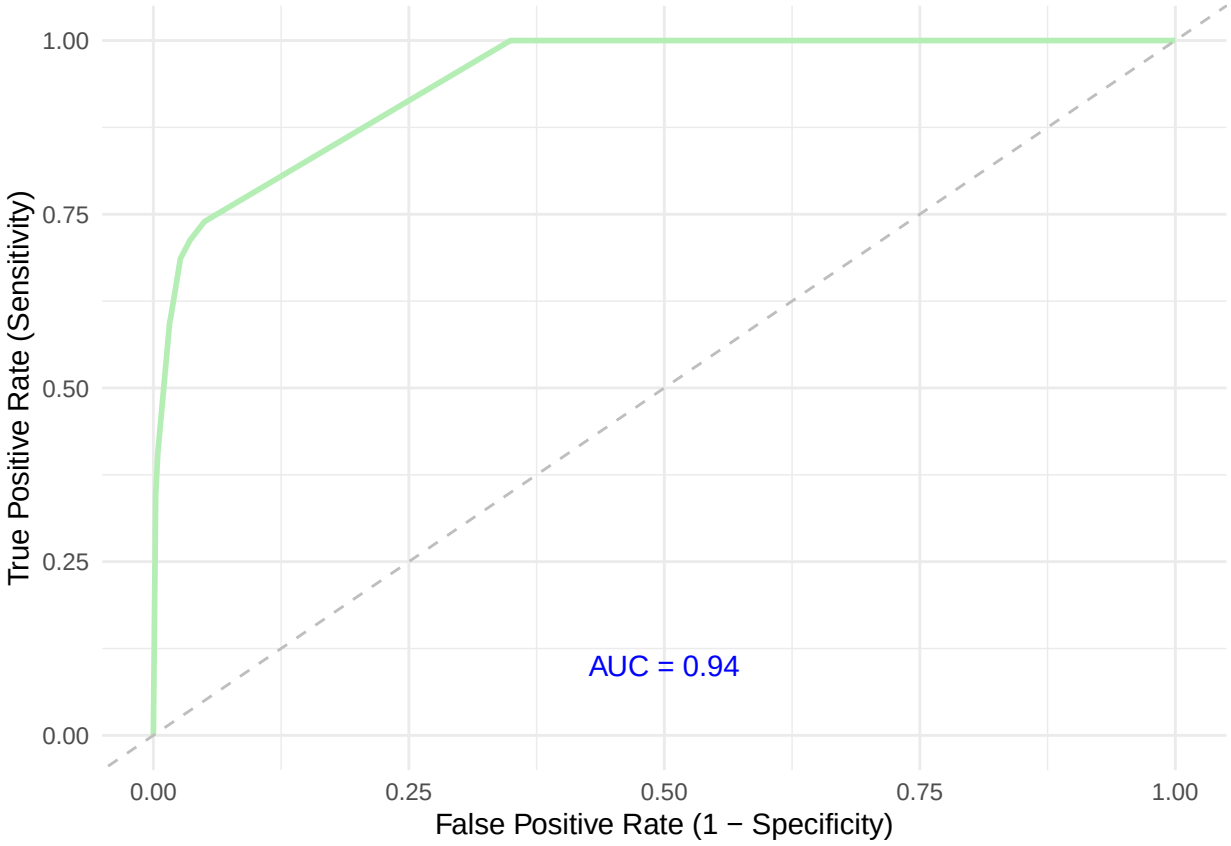


Figure 4: Curva ROC - Árbol de Decisión

5. Optimización del modelo

Mejores Hiperparámetros Los mejores hiperparámetros encontrados en la optimización son:

```
## max_depth = 17 , min_split = 29 , min_bucket = 21
```

Table 3: Comparativa de hiperparámetros entre los modelos 1 y 2

Parámetro	Modelo_Default	Modelo_Opt	Diferencia_Porcentual
maxdepth	30	17	-43.33333
minsplit	20	29	45.00000
minbucket	7	21	200.00000

Table 4: Comparativa de métricas entre los modelos 1 y 2

	Modelo_Default	Modelo_Opt	Diferencia
Accuracy	0.9096296	0.9145185	0.0048889
Precision	0.9156065	0.9316334	0.0160269
Recall	0.9735188	0.9605639	-0.0129548
F1	0.9436750	0.9458775	0.0022025

Inicialmente, el árbol por defecto tenía un AUC-ROC de 0.9415207. Para encontrar la mejor opción dentro de todas las exploradas, se minimizó el valor de AUC-ROC explorando los siguientes rangos de los parámetros: max_depth (1 a 30), min_split (1 a 30) y minbucket (1 a 30), estableciendo además cp = 0 y xval = 0, como indicaba la consigna. Como resultado, se obtuvo un árbol con un AUC-ROC de 0.9636. Esta mejora de 0.02207 se debió a la reducción de min_split de 20 a 16 y de max_depth de 30 a 17, mientras que minbucket fue el único parámetro que aumentó, pasando de 7 a 21 en comparación con el modelo por defecto.

En la Tabla 4 se observa que el nuevo modelo muestra una mejora en términos de accuracy, precisión y F1, aunque presenta una ligera disminución en recall. El F1 score experimenta una leve mejora, lo que sugiere un buen equilibrio entre precisión y recall. Sin embargo, el modelo pierde algo de capacidad para identificar los casos en los que sería adecuado otorgar un préstamo a una persona capaz de pagarlo. En otras palabras, esto podría traducirse en una pérdida de oportunidades y de ganancia para la empresa, ya que podría haberse beneficiado al otorgar el préstamo a una persona confiable. A pesar de la disminución en recall, el Modelo 2 sigue siendo generalmente superior, y dicha reducción no parece ser crítica dentro del contexto del problema.

6. Interpretación de resultados

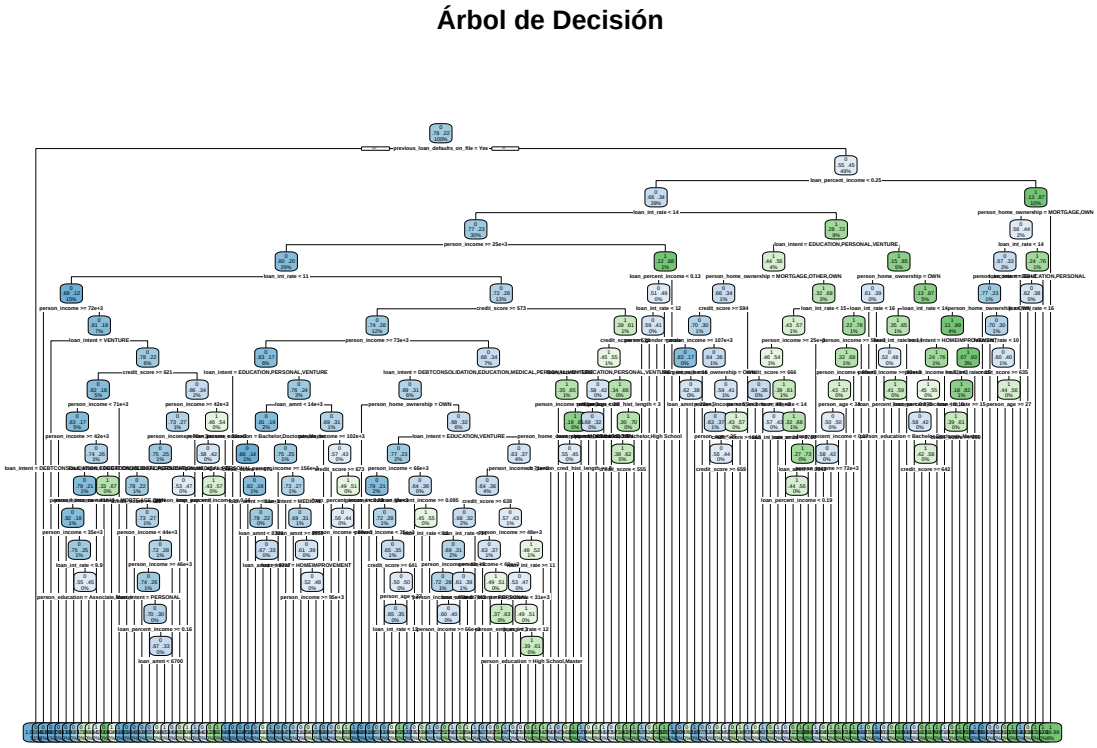


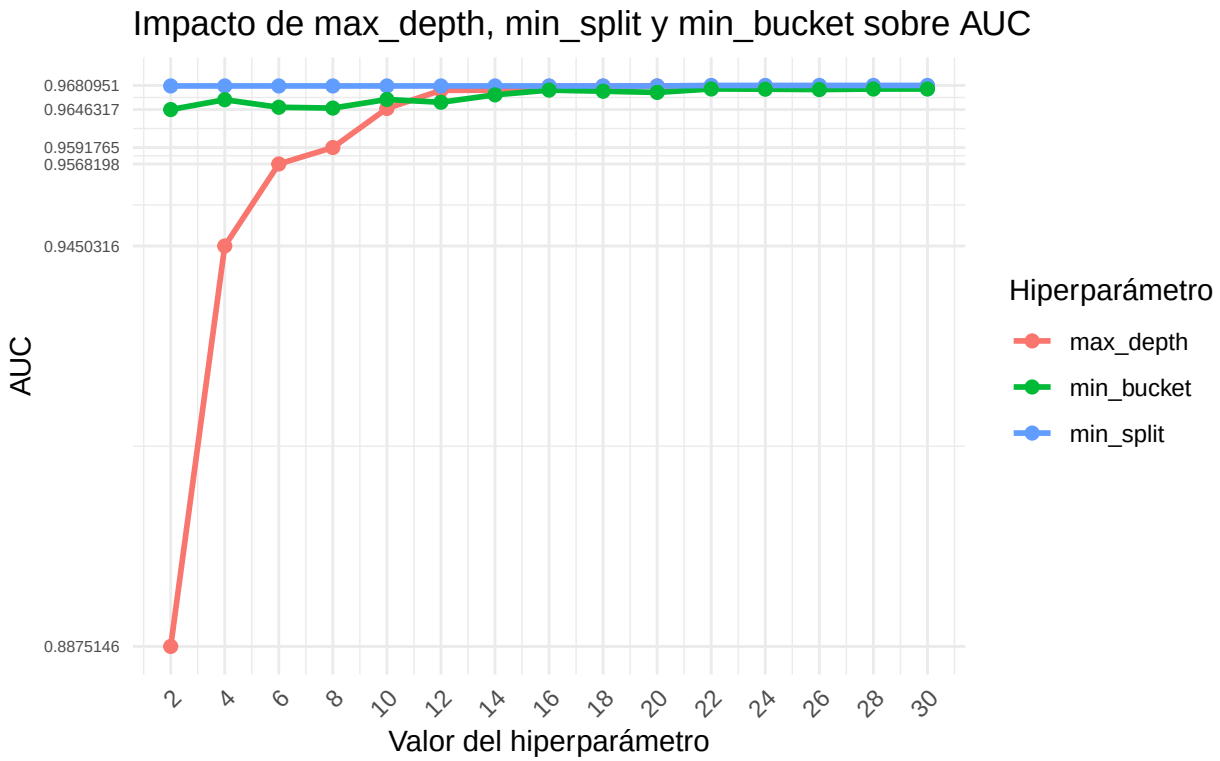
Figure 5: Árbol de Decisión del Modelo Optimizado

En el primer modelo, el valor predeterminado de minsplit era 20, lo que implicaba que, para hacer una partición, el modelo requería al menos 20 observaciones por nodo. Esto hacía que fuera más exigente y dificultaba la creación de particiones cuando había menos de 20 observaciones en un nodo. En el segundo modelo, esta exigencia se redujo a 15, lo que facilita la creación de particiones con menos observaciones, permitiendo una mayor granularidad y, por lo tanto, un modelo más detallado.

En cuanto a la profundidad del árbol (maxdepth), el valor predeterminado de 30 era bastante alto. Aunque, al hacer las particiones, no era necesario que el árbol fuera tan profundo, y en el peor de los casos, el árbol solo alcanzaba una profundidad de 6 niveles. Sin embargo, en el segundo modelo, la profundidad máxima llegó a ser de 17 niveles, lo que indica que el modelo ahora es mucho más profundo y tiene más capacidad para capturar complejidades en los datos.

Otro cambio relevante es el valor de minbucket, que en el modelo predeterminado estaba fijado en 7. Esto implicaba que cada hoja terminal del árbol debía tener al menos 7 observaciones. Este valor más alto resultaba en menos hojas, ya que el modelo agrupaba más observaciones en cada hoja. En el segundo modelo, al reducirse a 21, ahora es más probable que el modelo tenga más hojas, ya que cada hoja puede contener más divisiones con menos observaciones, lo que permite una mayor ramificación y más especificidad en las predicciones.Nos llama la atención que es un valor menor al de minsplit, eso no debería de suceder.

Lo que notamos en el árbol resultante es que este ha aumentado considerablemente en tamaño. El modelo optimizado ahora tiene un árbol mucho más ramificado, lo que significa que ha aumentado la capacidad de aprendizaje. Este tipo de árbol es capaz de capturar patrones más complejos y matices en los datos, lo cual puede mejorar la precisión del modelo, pero también puede llevar a un mayor riesgo de sobreajuste si no se gestiona adecuadamente.



Nota: Para cada línea se evalúa cómo afecta el hiperparámetro correspondiente al color, manteniendo los otros dos fijos. Además, se consideró que siempre se cumpla la restricción: min_split >= min_bucket.

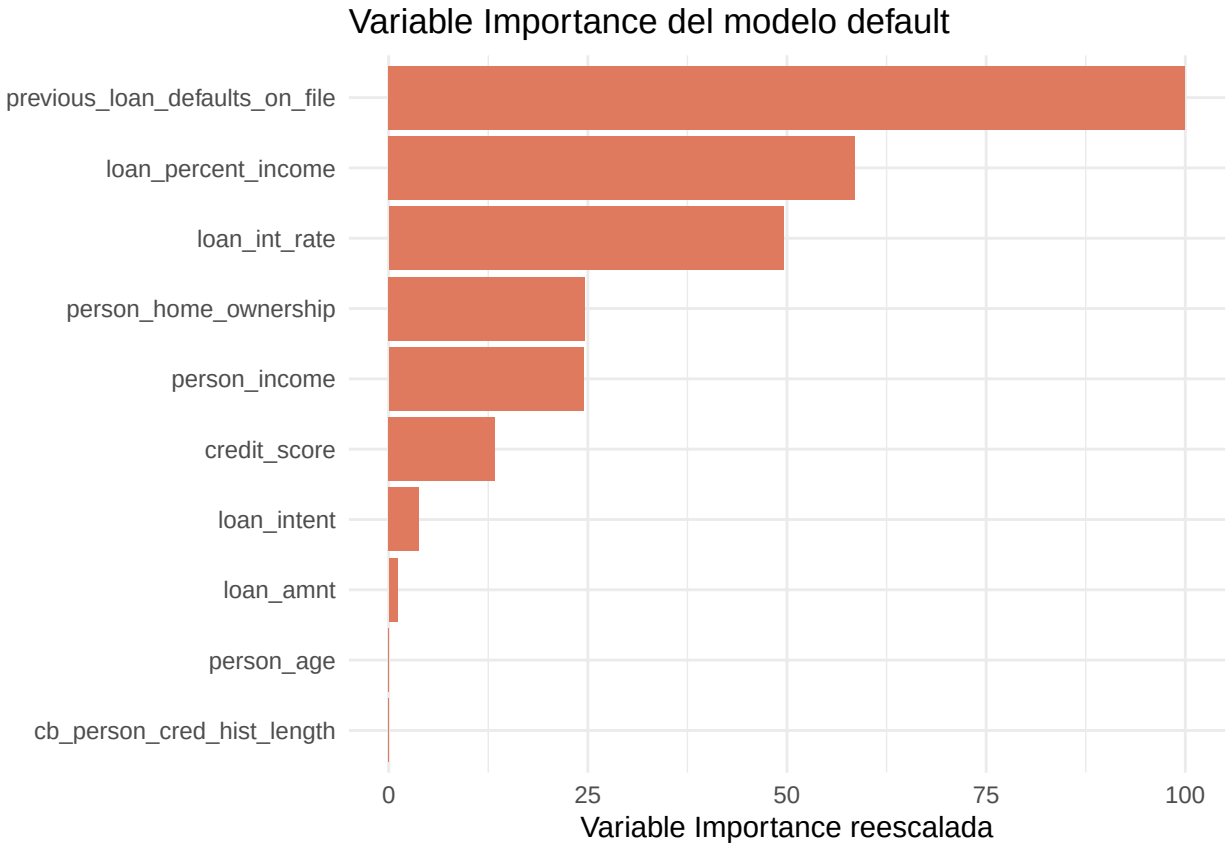
A partir del gráfico XX presentado, podemos analizar el impacto de los hiperparámetros max_depth, min_split y min_bucket en el desempeño del modelo, medido a través del área bajo la curva (AUC).

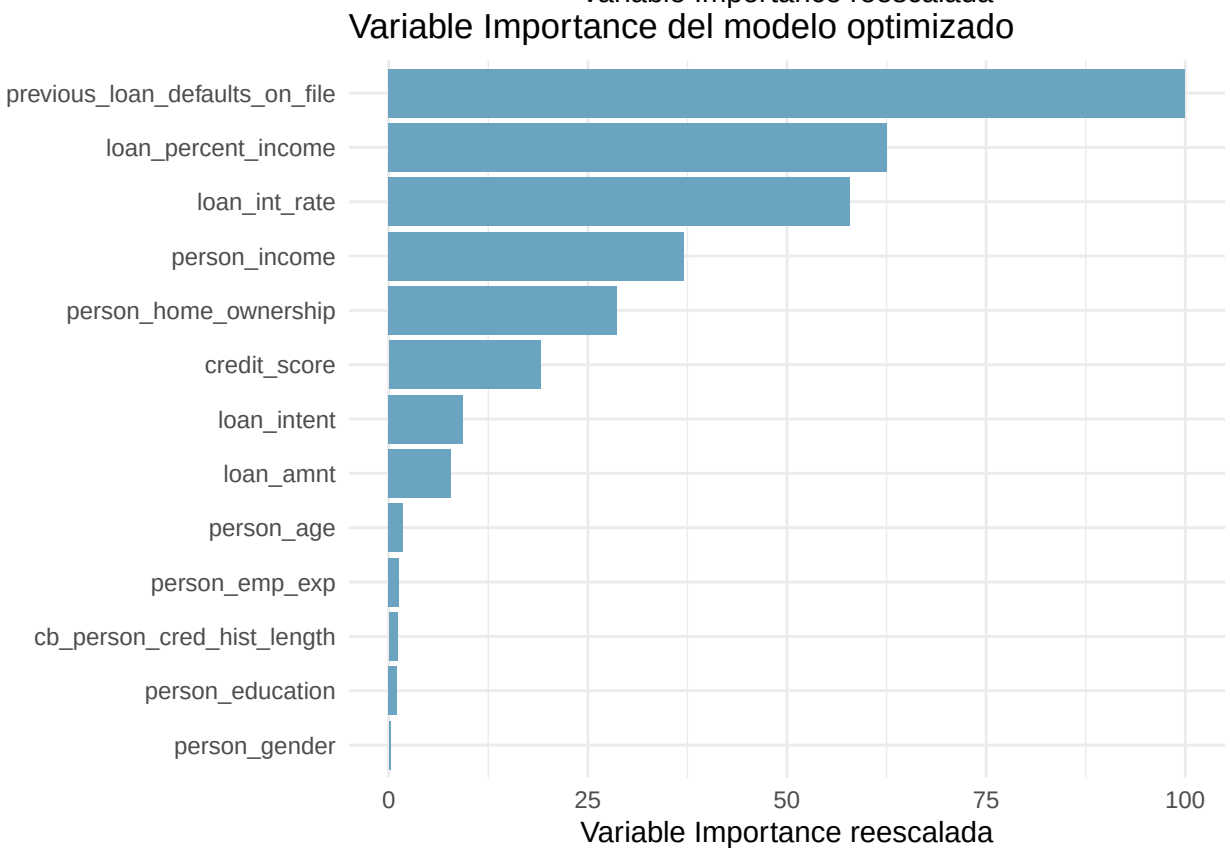
En primer lugar, observamos que el hiperparámetro con mayor influencia en el AUC es max_depth. A medida que aumenta desde 2 hasta aproximadamente 10, el AUC experimenta un crecimiento significativo, evidenciado por una pendiente pronunciada en la curva correspondiente. A partir de un valor de max_depth = 10, el AUC se estabiliza y alcanza un valor óptimo (0.9636), manteniéndose constante incluso con valores más altos de max_depth (>=12). Este comportamiento sugiere que más allá de cierto punto, aumentar la profundidad del árbol no aporta beneficios adicionales en términos de AUC y podría incluso derivar en un sobreajuste si no se controla adecuadamente.

Por otro lado, los hiperparámetros min_bucket y min_split muestran un comportamiento distinto. Desde valores bajos (por ejemplo, 2), estos ya presentan un AUC elevado, cercano a su valor óptimo. La curva de ambos parámetros se mantiene estable a lo largo de los distintos valores evaluados, lo que indica que su impacto en el rendimiento del modelo es menor en comparación con max_depth. Esto sugiere que una vez superado un umbral mínimo, estos parámetros no afectan significativamente la capacidad predictiva del modelo.

Un punto clave en el análisis ocurre alrededor de un valor de 16 para los tres hiperparámetros, donde el AUC se estabiliza completamente y toma un valor constante. Este comportamiento es crucial, ya que indica un punto de saturación en la optimización del modelo: más allá de este umbral, el ajuste de los hiperparámetros no genera mejoras en el desempeño.

En conclusión, el análisis sugiere que max_depth es el hiperparámetro con mayor influencia en la mejora del modelo, mientras que min_bucket y min_split tienen un impacto más limitado. Esto permite inferir que, en términos de optimización, es fundamental ajustar max_depth con mayor precisión, mientras que los otros dos hiperparámetros pueden establecerse en valores moderados sin una pérdida significativa en la calidad del modelo.





En estos gráficos observamos que, tanto en el modelo original con los hiperparámetros por defecto como

en el modelo optimizado, la importancia de las variables se mantiene igual. Destaca especialmente previous_loan_defaults_on_file, lo que indica que el historial de incumplimiento de préstamos anteriores es un factor clave en las predicciones. Además, la relación entre el monto del préstamo y los ingresos del solicitante, así como la tasa de interés, tienen un impacto significativo. Otros factores relevantes incluyen los ingresos personales, la propiedad de vivienda y el puntaje de crédito. Por otro lado, algunas variables, como el género y la educación, parecen tener poca influencia y podrían ser eliminadas para simplificar el modelo sin perder su precisión.

Table 5: Comparativa de métricas entre los modelos con valores faltantes

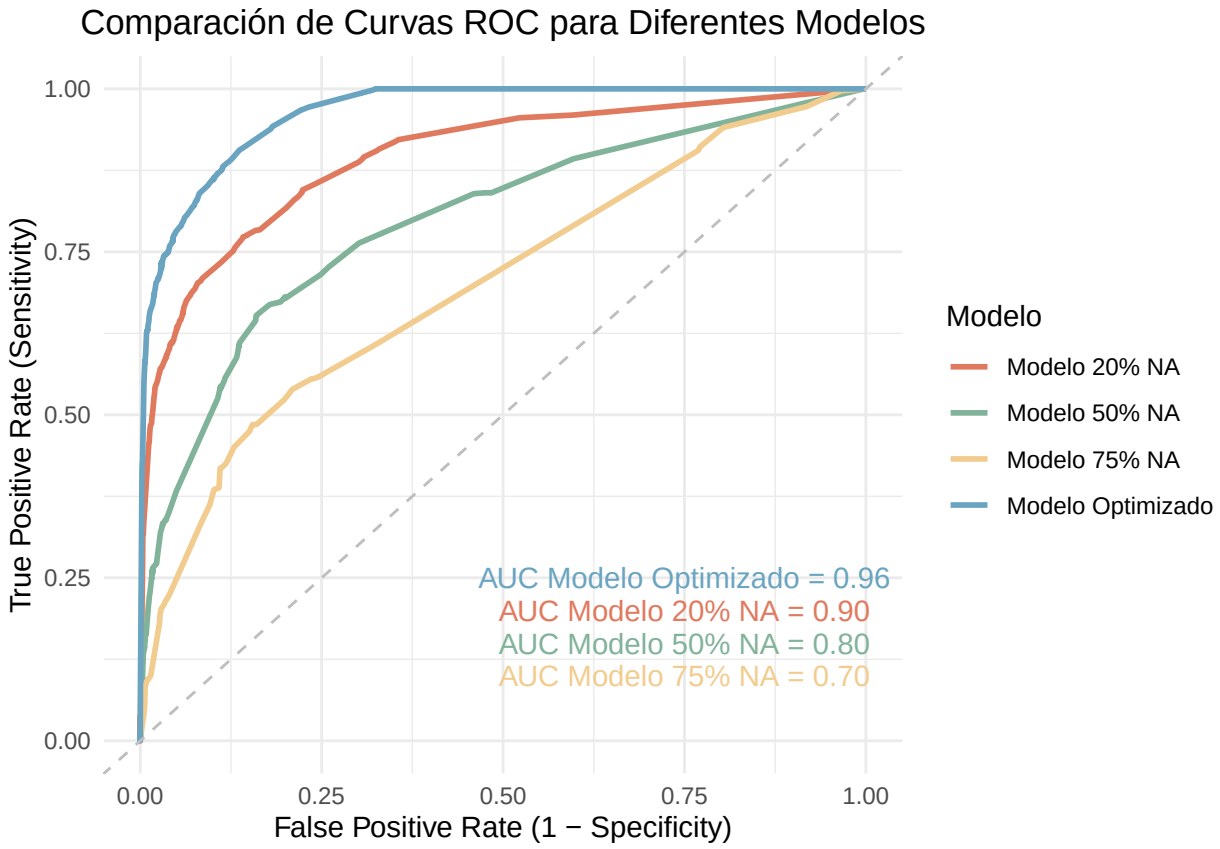
	Modelo_optimizado	Modelo_20	Modelo_50	Modelo_75
Accuracy	0.9145185	0.8792593	0.8225185	0.8003555
Precision	0.9316334	0.8966748	0.8330573	0.8132992
Recall	0.9605639	0.9550749	0.9636015	0.9658314
F1 Score	0.9458775	0.9249540	0.8935868	0.8830267
AUC	0.9636021	0.8961419	0.7973641	0.7000718

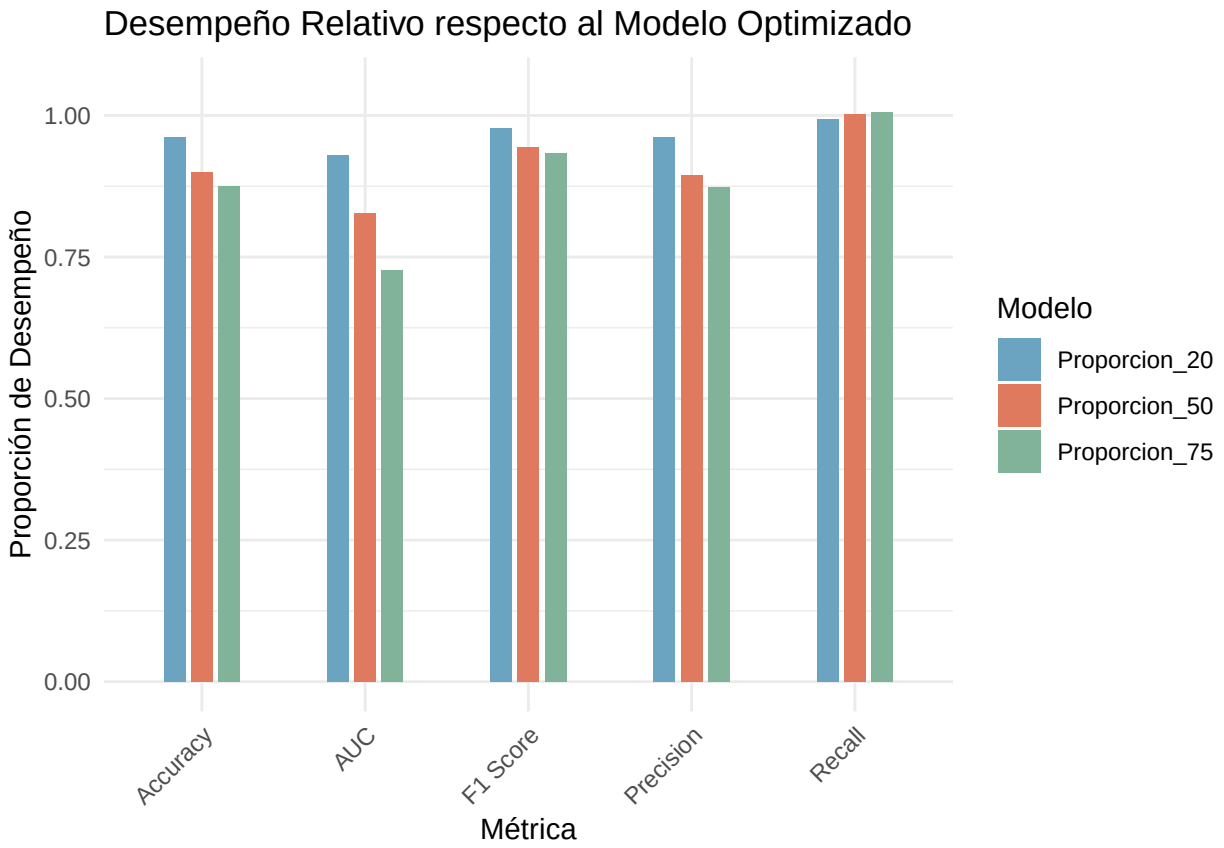
7. Análisis del impacto de los valores faltantes

En esta sección, se evalúa cómo la presencia de valores faltantes en las variables predictoras afecta el rendimiento de un modelo de clasificación basado en árboles de decisión. Para ello, tomamos el conjunto de datos original -que no tenía valores faltantes- e insertamos aleatoriamente valores faltantes para generar tres conjuntos de datos nuevos, cuyas proporciones de valores faltantes son: 20%, 50% y 75%. Estos conjuntos se aplicaron tanto al entrenamiento, validación como al testeo, manteniendo las mismas observaciones en cada versión para garantizar la consistencia del análisis.

Se procedió a optimizar los hiperparámetros del árbol de decisión en cada caso, maximizando el AUC en el conjunto de validación con un algoritmo de grid search. Finalmente, se comparó el desempeño de cada modelo con el árbol optimizado previamente obtenido en el punto 5, evaluando cómo el aumento en la proporción de datos faltantes impacta en las métricas de rendimiento. Este análisis se complementa con gráficos que ilustran visualmente las diferencias entre los modelos.

Grafico curvas ROC para los distintos modelos:





En el gráfico se compara el desempeño de los modelos con diferentes porcentajes de valores faltantes (20%, 50%, 75%) en relación con el modelo optimizado. La división entre los modelos refleja la proporción de desempeño de cada modelo en comparación con el optimizado, lo que permite observar cómo se ve afectado el rendimiento conforme aumenta el porcentaje de valores faltantes.

Un patrón claro es que todas las métricas, excepto recall, disminuyen conforme se incrementa el porcentaje de valores faltantes. Esto es consistente con lo que muestra el gráfico de AUC: a medida que aumenta el porcentaje de valores faltantes, el rendimiento de cada métrica (accuracy, precision, F1 score, AUC) empeora, siendo más pronunciada esta caída a medida que nos acercamos al 75% de valores faltantes. En cambio, recall muestra un comportamiento diferente: en el modelo optimizado, tiene un valor de 0.960, pero mejora en los modelos con 50% (0.9636) y 75% (0.9658) de valores faltantes. Este patrón sugiere que, aunque el modelo pierde precisión en otras métricas, se vuelve más efectivo para identificar los casos positivos cuando la proporción de valores faltantes aumenta. A partir del análisis del gráfico anterior, donde observamos que a mayor cantidad de valores faltantes (NA), el rendimiento del modelo disminuye, en este gráfico podemos ver de manera más detallada el impacto en cada métrica individual. Concluimos que accuracy, precision, F1 score y AUC disminuyen conforme aumenta el porcentaje de valores faltantes, lo que indica una pérdida en la capacidad del modelo para discriminar entre las clases. Sin embargo, recall muestra un comportamiento interesante: mejora en los modelos con 50% y 75% de valores faltantes. Esto sugiere que, aunque el modelo pierde precisión en otras métricas, sigue siendo eficaz para identificar los casos positivos, e incluso, en algunos casos, mejora en esta capacidad a pesar del aumento de los valores faltantes.