UNIVERSIDAD TORCUATO DI TELLA

Análisis de Riesgo de Crédito

Trabajo Final

Maestría en Finanzas

Universidad Torcuato Di Tella (UTDT)

Prof. Roccatagliata Pablo

Basabe Paloma

Mérida Ezequiel

Parrotta Agustin

Octubre de 2021

1. Introducción

Es importante que los bancos puedan evaluar adecuadamente el riesgo de cada cliente. Además, los tiempos de respuesta para estas evaluaciones son cada vez más exigentes.

El dataset utilizado contiene información de 150 mil clientes de una entidad financiera. La variable target 'SeriousDlqIn2yrs' tiene un 7% de casos positivos y un 93% de negativos, por lo que se encuentra fuertemente desbalanceada.

Se realiza un análisis exploratorio del dataset utilizando el lenguaje de programación Python. A continuación, se implementan los modelos de Random Forests, KNN y Logistic Regression. Finalmente, se ajusta el umbral (threshold) del modelo de Random Forests asignándole costos a la matriz de confusión.

2. Análisis Exploratorio

A continuación, se analizan las principales variables presentes en el dataset. A partir de esto, se realiza un proceso de limpieza a los datos.

El dataset presenta 150 mil registros de clientes de una entidad financiera. Posee 11 columnas (1 label y 11 features):

- **SeriousDlqin2yrs**: Es el label o target del dataset. Presenta valores 0 o 1, en la cual 1 indica que la persona presenta más de 90 días de atraso en el pago (default).
- **RevolvingUtilizationOfUnsecuredLines**: Saldo total en tarjetas de crédito y líneas de crédito personales, excepto bienes raíces y sin deudas a plazos (como préstamos para automóviles), dividido por la suma de los límites de crédito.
- **Age**: Edad del prestatario en años.
- **DebtRatio**: Pagos mensuales de deuda, pensión alimenticia y costo de vida, dividido por el ingreso bruto mensual.
- **MonthlyIncome**: Ingreso mensual.
- **NumberOfOpenCreditLinesAndLoans**: Número de préstamos abiertos y líneas de crédito.
- **NumberRealEstateLoansOrLines**: Número de préstamos hipotecarios e inmobiliarios, incluidas las líneas de crédito con garantía hipotecaria.
- **NumberOfTime30-59DaysPastDueNotWorse**: Número de veces que el prestatario ha estado atrasado entre 30 y 59 días como máximo en los últimos 2 años.
- **NumberOfTime60-89DaysPastDueNotWorse**: Número de veces que el prestatario ha estado atrasado entre 60 y 89 días como máximo en los últimos 2 años.

- **NumberOfTimes90DaysLate**: Número de veces que el prestatario ha estado atrasado 90 días o más.
- **NumberOfDependents**: número de dependientes en la familia excluyéndose a sí mismos (cónyuge, hijos, etc.).

Variable SeriousDlqIn2yrs

El primer análisis a realizar es En la Figura 1 se observa el histograma del label.

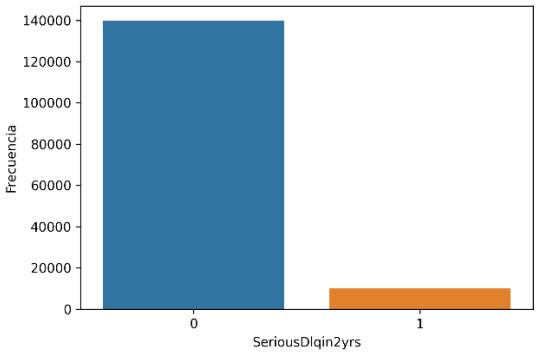

Figura 1.

Puede verse que el dataset se encuentra fuertemente desbalanceado, con un 6,7% de casos positivos (default) y un 93,3% de negativos.

Existen algunas soluciones para este tipo de problemas. Una de las más utilizadas es el Resampling, el cual se encuentra a su vez en dos versiones:

- Una es agregar copias de registros de la clase minoritaria (over-sampling)

- La otra consiste en eliminar registros de la clase mayoritaria (under-sampling)

Otra de las técnicas más conocidas es la denominada SMOTE (Synthetic Minority Over-sampling Technique), la cual consiste en sintetizar nuevos registros de la clase minoritaria.

Análisis General

En esta sección se realiza un análisis general del dataset.

En primer lugar, se observa que el mismo presenta valores nulos sólo en 2 columnas:

- **MonthlyIncome**: 29731 registros nulos (19.8%).
- **NumberOfDependents**: 3924 registros nulos (2.6%).

Debido a que este número es elevado en la variable MonthlyIncome, no es conveniente borrar los registros. En cambio, deberían reemplazarse con algún valor representativo, como la media o mediana. Esto se realiza más adelante.

Otro de los aspectos a analizar es la correlación entre las variables. En la Tabla 1 se expone la correlación lineal entre los features y el target. No se observa una dependencia lineal fuerte, ya que los valores se aproximan a 0.

Variable	**Coeficiente de correlación lineal**
NumberOfTime30-59DaysPastDueNotWorse	0.126
NumberOfTimes90DaysLate	0.117
NumberOfTime60-89DaysPastDueNotWorse	0.102
NumberOfDependents	0.046
RevolvingUtilizationOfUnsecuredLines	-0.002
NumberRealEstateLoansOrLines	-0.007
DebtRatio	-0.008
MonthlyIncome	-0.020
NumberOfOpenCreditLinesAndLoans	-0.030
Age	-0.115

Tabla 1.

Mas aún, en la Figura 2 se observa la matriz de correlación lineal, teniendo en cuenta todas las variables.

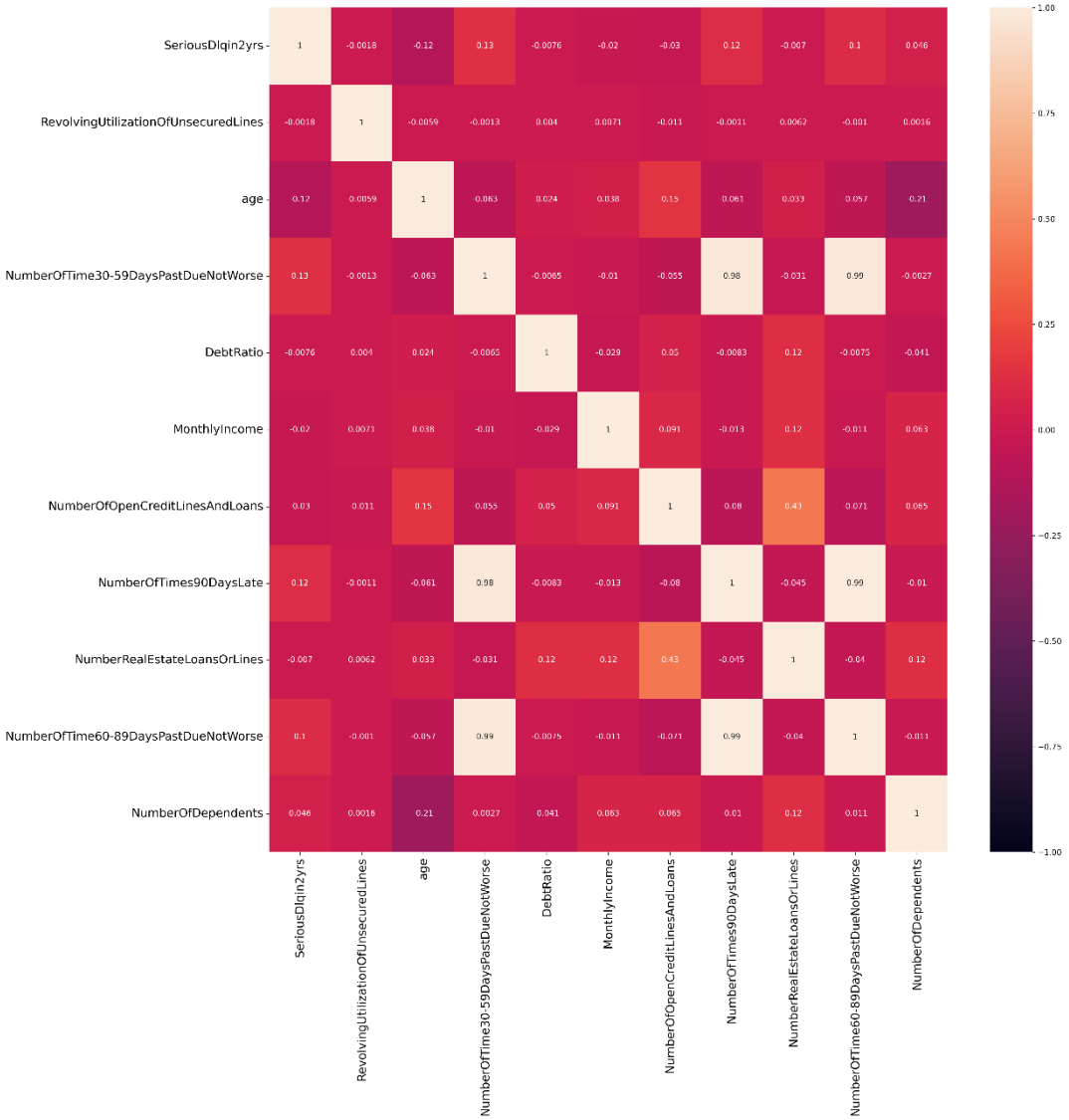

Figura 2.

Se observa que NumberOfTime30-59DaysPastDueNotWorse, NumberOfTime60-89DaysPastDueNotWorse y NumberOfTimes90DaysLate presentan una correlación lineal positiva casi perfecta. Esto tiene sentido, teniendo en cuenta que describen situaciones similares (número de veces que el cliente defaulteó). Por ende, la información es redundante, siendo una opción viable seleccionar solo una de las tres features. Esto se analiza nuevamente más adelante.

Variable MonthlyIncome

En la Figura 3 se observa el histograma de esta variable

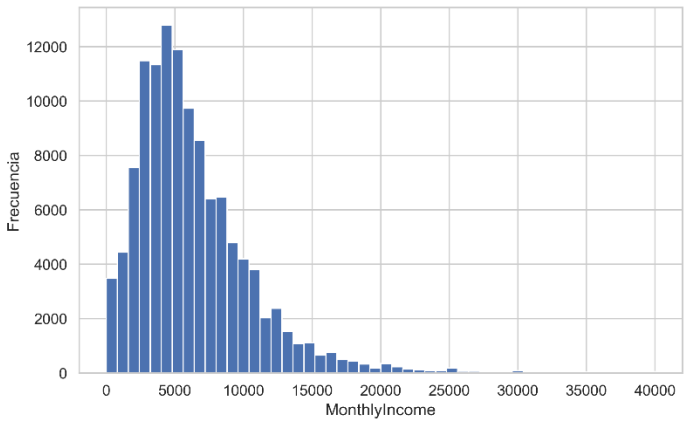

Figura 3.

La media y mediana son 6670 y 5400, respectivamente. Por un lado, se observa que la curva presenta una asimetría positiva, lo cual tiene sentido con la realidad debido a que es una distribución de ingresos. Debido a esto, para reemplazar los valores nulos es conveniente utilizar la mediana que la media.

Variable NumberOfDependents

En la Figura 4 se observa el histograma de esta variable.

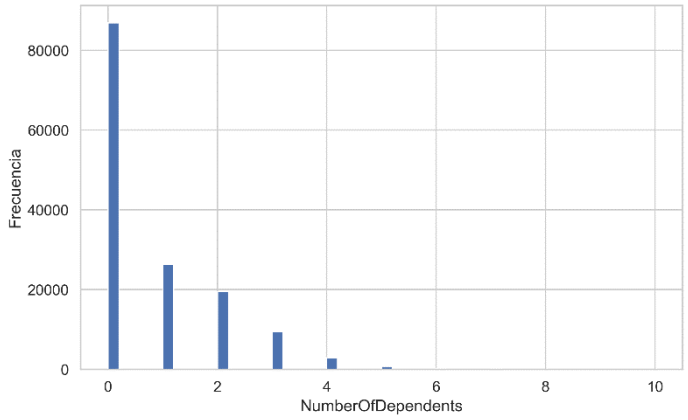

Figura 4.

La mayoría de las observaciones se encuentran cercanas 0 (solo 12 registros presentan valores por encima de 9). A su vez, es posible que el prestatario, si no tiene familiares a cargo, no rellene este dato, lo cual podría explicar la existencia de valores nulos. Debido a esto, se decide reemplazar los valores nulos con el valor 0.

Variables NumberOfTime30-59DaysPastDueNotWorse, NumberOfTime60-89DaysPastDueNotWorse y NumberOfTimes90DaysLate

Al intentar graficar el histograma de estas variables se observó que la mayoría de las observaciones se encuentran entre 0 y 20, aproximadamente. Sin embargo, existe una serie de registros (269) en las que los valores son 96 y 98 simultáneamente para las tres variables, lo cual es erróneo. Se decidió eliminar estas observaciones.

En la sección anterior se observó que la correlación era aproximadamente 1. En la Figura 5 se vuelve a graficar la matriz de correlación, luego de eliminar los datos erróneos. Se observa que la correlación lineal bajó considerablemente, por lo que no se eliminará ninguna variable.

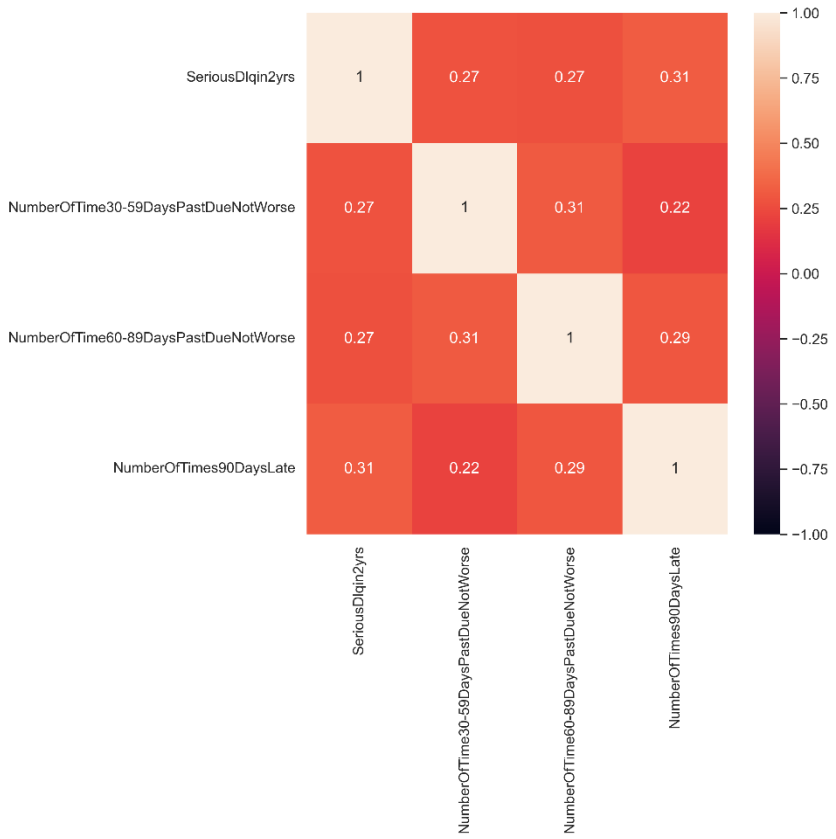

Figura 5.

Variable DebtRatio

Para esta variable se observa que el percentil75 es 0.87 y su valor máximo es 329664. Esto da indicios de que existen outliers que se deberían eliminar. Siendo más específico, se observa que para valores donde DebtRatio es mayor al percentil90 solo existen 845 de los 14971 registros valores donde MonthlyInome tiene valores no nulos, lo cual indica la posibilidad de un error en estos datos. Más aun, los valores de MonthlyIncome son 0 o 1. Se decide, por ende, eliminar esta serie de observaciones.

Variable RevolvingUtilizationOfUnsecuredLines

Para esta variable se observa que el percentil75 es 0.55 y su valor máximo es 50708. Esto, al igual que para DebtRatio, puede indicar que existen errores en los datos. En este caso, se observa que para valores donde la variable en discusión es mayor al percentil90, MonthlyIncome no presenta muchos valores nulos. Sin embargo, se observa que el porcentaje de default es del 22.6%. Por ende, se decide profundizar más sobre esto. La variable indica el saldo que se adeuda en cada línea de crédito dividido la cantidad de líneas de crédito. Ergo, tiene sentido que, a mayor sea este valor, la probabilidad de default se incrementa. Sin embargo, a partir del percentil99, esta probabilidad comienza a descender. Se decide entonces eliminar esta última serie.

Separación del Dataset

Luego de aplicar todos los cambios al dataset, el número de observaciones resulta en 133406. Antes de continuar con el proceso, es necesario separar los datos en train y test. Se utiliza un 20% de los mismos para conformar el test set. De esta forma los train y test sets poseen 106724 y 26682 registros, respectivamente. Es importante destacar que la separación se realiza de manera estratificada respecto a la variable target, debido a que el dataset se encuentra desbalanceado.

3. Preproceso

Mucho de los modelos de Machine Learning precisan normalizar o estandarizar los datos, debido a que son sensibles a las escalas. Sin embargo, modelos como Decision Tree o Random Forests no requieren esto, ya que sus algoritmos no comparan features entre sí. Aún así, realizar este procesamiento es una buena práctica. Además se utilizarán modelos que sí lo requieren. Por ende, se opta por estandarizar los datos.

4. Entrenamiento

4.1. Modelos Baseline

Se seleccionan los siguientes modelos a utilizar:

- Regresión logística
- K-Nearest Neighbors (KNN)
- Random Forests

A su vez, se utiliza el área bajo la curva como principal métrica de performance. Se calculan también Accuracy, Precision, Recall y F1.

Regresión Logística

En primer lugar, se entrena un modelo de regresión logística. La curva ROC de este modelo se expone en la Figura 6. Su AUC es de 0.843.

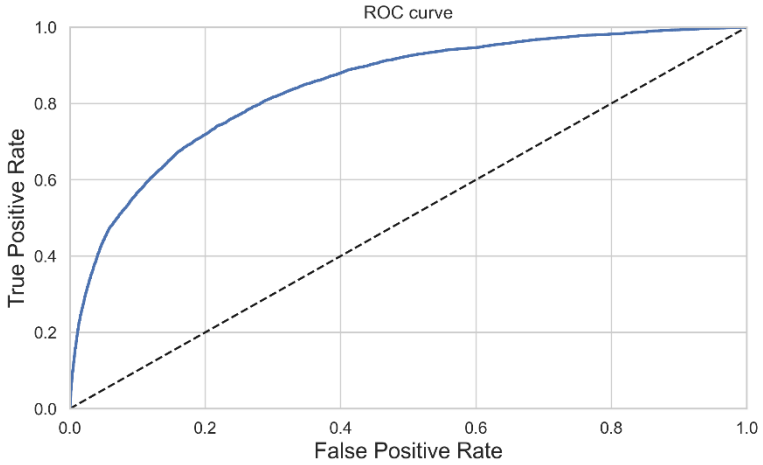

Figura 6.

KNN

Luego se entrena el modelo KNN. La curva ROC se expone en la Figura 7. El AUC obtenido es de 0.725, encontrándose por debajo de la Regresión Logística.

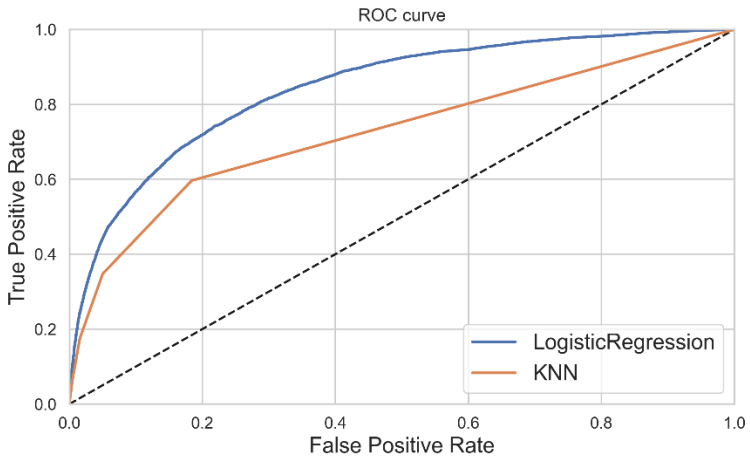

Figura 7.

Random Forests

Luego se entrena el modelo Random Forests. La curva ROC se muestra en la Figura 8. Su AUC es de 0.827, un poco menor a la Regresión Logística pero mayor al modelo KNN.

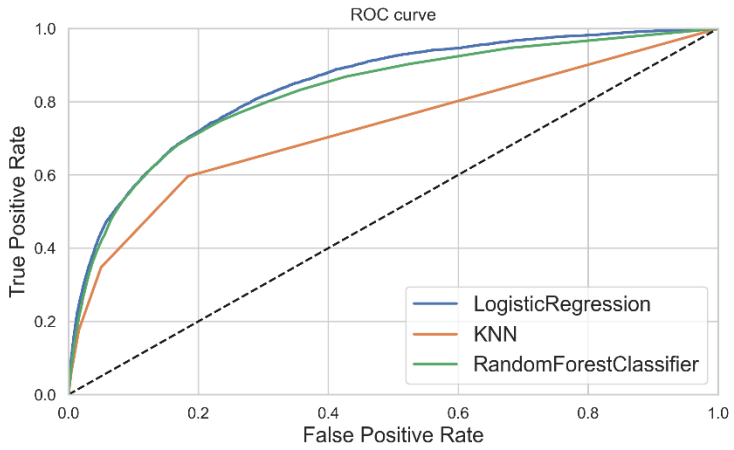

Figura 8.

Si bien la Regresión Logística presentó el mejor resultado, el modelo de RandomForests suele aplicar bien a este tipo de problemas, siempre y cuando se seleccionen correctamente sus hiperparámetros [1].

4.2. Optimización de hiperparámetros

Los principales hiperparámetros del modelo Random Forests son:

1. Número de Decision Trees a entrenar.
2. Número de features a incluir (de forma aleatoria) en cada nodo.
3. Máxima profundidad de Decision Trees (número de nodos). Este es un hiperparámetro que puede sobreajustar mucho a los datos de entrenamiento (a mayor profundidar, mayor ajuste).
4. La mínima cantidad de muestras necesarias para separar un nodo. Este hiperparámetro también es importante a la hora de regularizar el modelo.
5. Utilizar o no la técnica de Bootstrap.

A continuación, se realiza un tuneo de los hiperparámetros. En particular, se seleccionan el primero y el tercero. Los valores elegidos son:

- Número de Decision Trees: 100, 325, 550, 775, 1000.
- Máxima profundidad: 10, 32, 55, 77, 100, sin límite.

Se realizó validación cruzada con cada uno de los modelos. El modelo con 1000 Decision Trees y máxima profundidad de 10 fue el que arrojó mejor AUC, 0.856 precisamente. Nótese que los valores son los extremos dentro del rango elegido, por lo que se podría seguir profundizando en el tuneo, explorando sobre rangos, es decir, mayor número de Decision Trees y profundidades menores. Especialmente, este último es más importante teniendo en cuenta el riesgo de overfitting.

En la Figura 9 se presenta la curva ROC del modelo optimizado final, con AUC de 0.856.

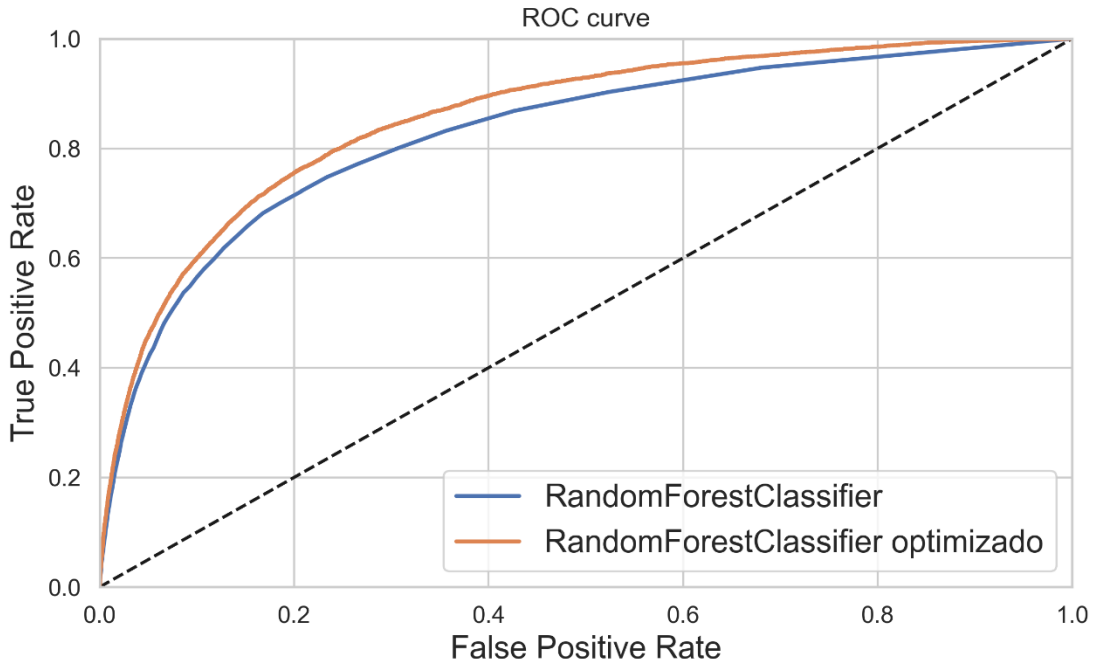

Figura 9.

5. Estimación de Costos

En la Figura 10 se muestra la matriz de confusión resultante.

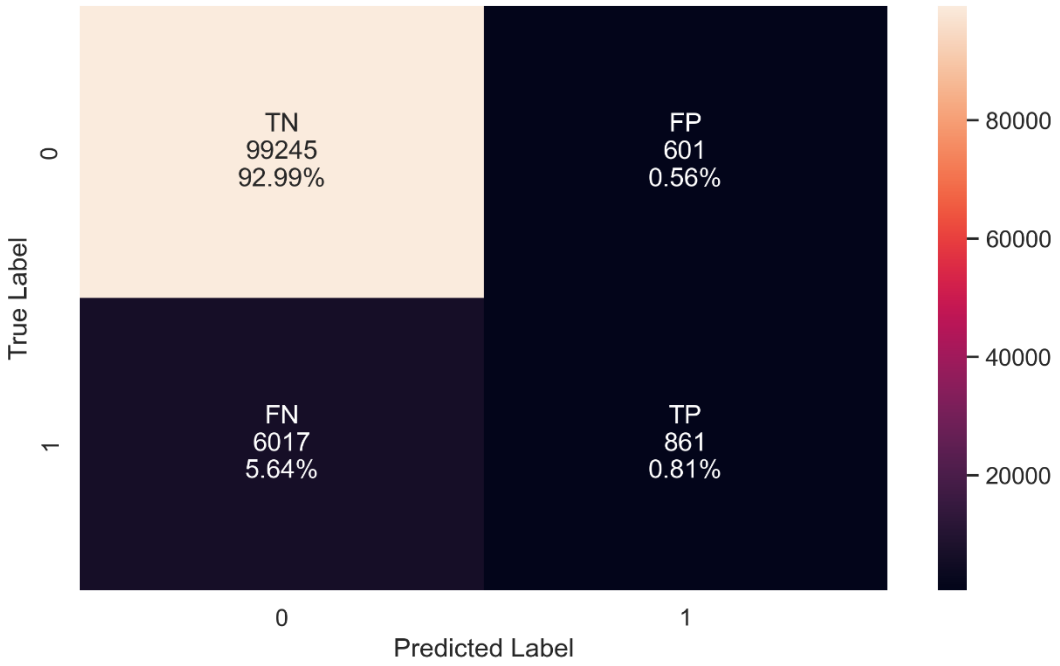

Figura 10.

Se exponen a continuación algunas métricas relevantes:

- Precision: 58.89%
- Recall: 12.51%
- Accuracy: 93.79%
- F1: 20.65%

Para elegir el valor del umbral de crédito (threshold) apropiado, es necesario establecer una matriz de costos de clasificación. Correa presenta una propuesta para realizar esto [2], con algunas simplificaciones opcionales. En su forma más compleja, la estrategia consiste en asignar costos específicos a cada cliente según su nivel de deuda o la potencial pérdida de ganancia.

Sin embargo, una forma de simplificar el armado de esta matriz de costos es asignar valores constantes. Así, tiene sentido realizar una donde:

- El costo de los True Positive y True Negative sean nulos, ya que si el modelo predice correctamente, ya que solo se le estaría dando un crédito a clientes que no defaultean.
- El costo de False Negatives es mucho más severo que el de False Positive, debido a que no detectar un posible cliente que defaultee es mucho más costoso que la potencial pérdida de ganancia al no otorgarle un préstamo a un cliente. Dado esto, se le otorga una relación FP/FN de 1/10.

De esta manera, se debe minimizar la siguiente función:

$$C = C_{TP}\,TP + C_{TN}\,TN + C_{FP}\,FP + C_{FN}\,FN = FP + 10\,FN$$

Con esto, utilizando la herramienta Solver de Excel, se varía el threshold con el objetivo de minimizar esta función. Se observa que el resultado es threshold = 0.08, menor al valor por default de 0.5. Esto tiene sentido, ya que se está penalizando mucho más un falso negativo, por lo que se prefiere que el modelo detecte como positivo con una menor probabilidad. Con esto, la matriz de confusión resultante se observa en la Figura 11.

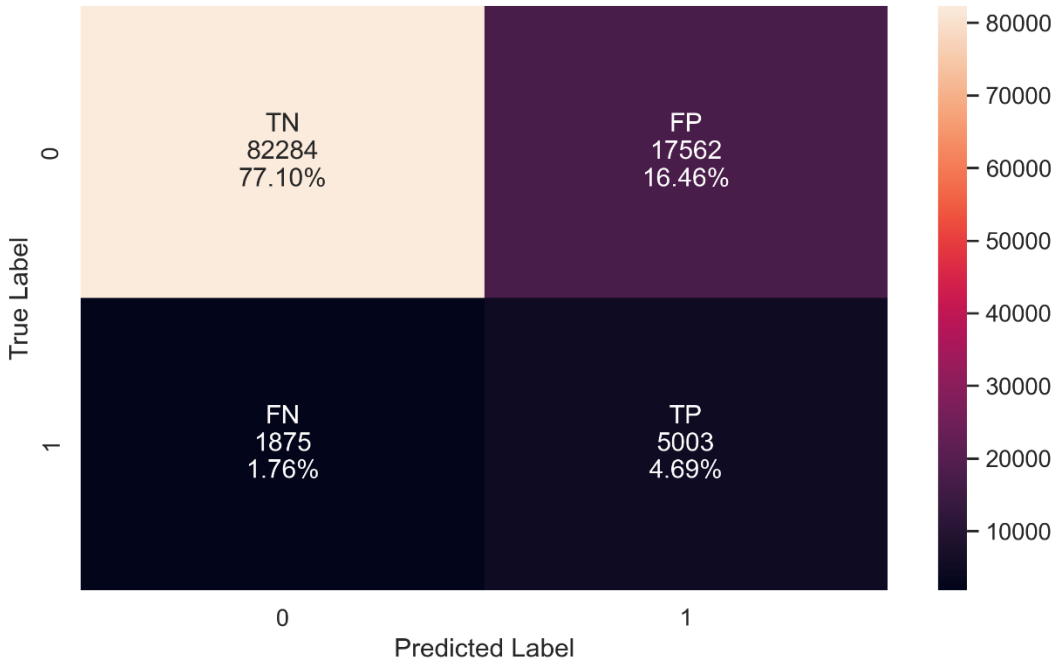

Figura 11.

Se exponen a continuación las métricas de performance vistas anteriormente:

- Precision: 22.17%
- Recall: 72.74%
- Accuracy: 81.78%
- F1: 33.98%

Obsérvese del gráfico que el número de FN disminuyó a menos de 2%, a costo de que FP aumentó un 16%. Esto es análogo a que el Recall aumentó en detrimento de la Precision.

Por su parte, se observa que el Accuracy disminuyó. Esto se debe a que el threshold, al disminuir, asigna como positivos a registros con baja probabilidad (mucho de las observaciones que antes se catalogaban como TN se convierten en FP). Es por esto también que el Accuracy no es una buene métrica de performances para este tipo de datos.

Conceptualmente, el threshold influye decisivamente en las predicciones del modelo, por lo que se puede considerar como un hiperparámetro del mismo. Asimismo, este se podría optimizar utilizando la técnica de validación cruzada. Para eso, se pueden definir ciertos valores de threshold (al igual que como se realiza con el resto de hiperparámetros) y aplicarlos a las predicciones antes de calcular la métrica de evaluación.

6. Evaluación

Finalmente, se entrena el modelo final con el train set completo y se evalúa el mismo utilizando el test set (el cual es estandarizado al igual que el train set). La matriz de confusión se muestra en la Figura 12.

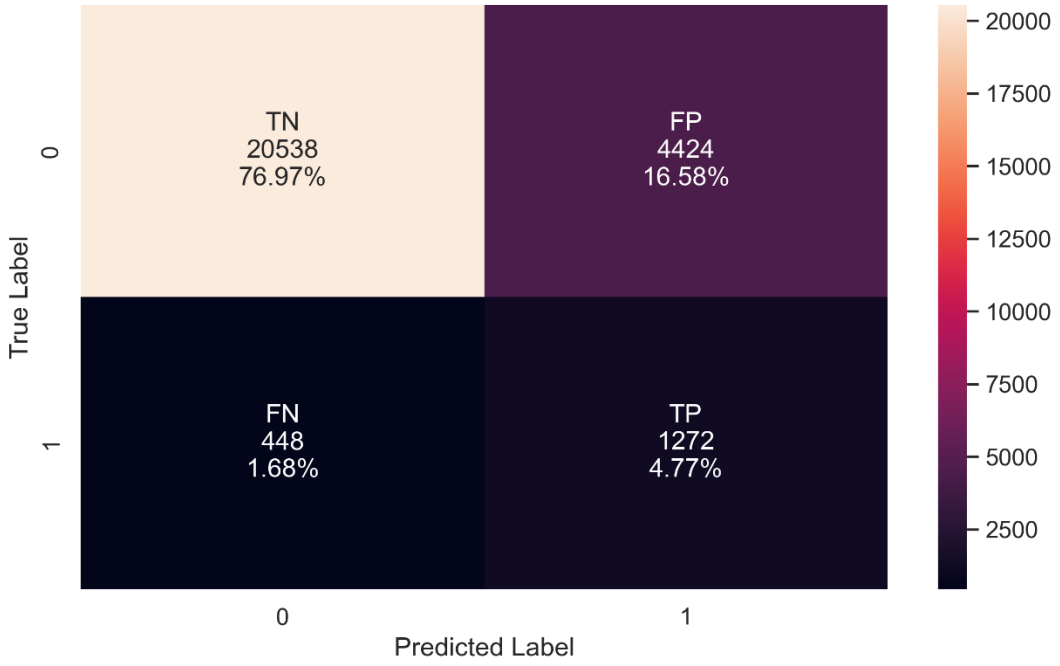

Figura 12.

Se exponen a continuación las métricas de performance:

- Precision: 22.33%
- Recall: 73.95%
- Accuracy: 81.74%
- F1: 34.30%

Puede observarse que las métricas no difieren de los valores calculados con el train set. Esto indica que el modelo no presenta overfitting y ajusta de la misma manera a los datos de testeo.

7. Conclusión

El presente trabajo consistió en realizar un análisis de riesgo de crédito, a partir de una base de datos con información crediticia de 150 mil clientes. En primer lugar, se realizó un análisis exploratorio del dataset, observando las distintas variables involucradas y realizando una limpieza y procesamiento de los datos. Se removieron datos erróneos, se reemplazaron valores nulos por valores representativos del dataset y se estandarizaron los datos. A su vez, la variable target, la cual indicaba si el cliente defaulteó o no, se

encontraba fuertemente desbalanceada (7% de casos positivos), por lo cual se tuvo especial cuidado en la evaluación de los modelos utilizados.

Luego se ajustaron los datos utilizando técnicas de Machine Learning. En particular, se utilizaron la Regresión Logística, k-Nearest Neighbors y Random Forests. Los primeros dos se ajustaron a modo de comparación, mientras que para el tercero se realizó un análisis más profundo, optimizando algunos de sus hiperparámetros, como el número de árboles y la máxima profundidad. Otro aspecto a mencionar es que la métrica de performance empleada fue el área bajo la curva AUC, muy utilizada en modelos de clasificación. Además, se usó también la técnica de validación cruzada sobre el set de entrenamiento.

Luego se implementó un tuneo del umbral de crédito a partir de una matriz de costos. En particular, se le asignaron costos nulos a los valores True Positive y True Negative, mientras que se le asignó un valor de 1 a los False Positive y de 10 a los False Negative. Luego, se ajustó el threshold del modelo de Random Forests para minimizar el costo, dando un resultado de 0.08.

Finalmente, se calcularon las principales métricas de performance sobre el set de test (20% del dataset original), obteniéndose:

- Precision: 22.33%
- Recall: 73.95%
- Accuracy: 81.74%
- F1: 34.30%

8. Bibliografía

[1] Géron A, *Hands-on Machine Learning with Scikit-Learn. Keras & TensorFlor,* 2019. O´Reilly.

[2] Correa A, *Tutorial: Example-Dependent Cost-Sensitive Credit Scoring using CostCla*, 2015. https://nbviewer.org/github/albahnsen/CostSensitiveClassification/blob/master/doc/tutorials/tutorial_edcs_credit_scoring.ipynb