|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | Imagen relacionada | |  | | MAESTRÍA EN DATA MINING & KNOWLEDGE DISCOVERY  Fundamentos del Aprendizaje Automático | | Trabajo Práctico Final  2019 | | | |
|  | | |
| Profesor: |  | Hernán Merlino |
| |  |  | | --- | --- | | Integrantes del grupo | | | Nombre | E-Mail | | Uboldi, Maximiliano | maxiuboldi@gmail.com | | Villegas, Marcelo | allekinda@gmail.com | | | |

Tabla de contenido

[1. Conjunto de datos y análisis exploratorio 3](#_Toc20853199)

[2. Preprocesamiento y selección del modelo 4](#_Toc20853200)

[3. Implementación 5](#_Toc20853201)

[4. Próximas tareas 9](#_Toc20853202)

# Conjunto de datos y análisis exploratorio

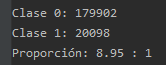
A los fines de la realización del presente trabajo se obtuvo el dataset de una competencia de Kaggle alojada en el siguiente enlace:

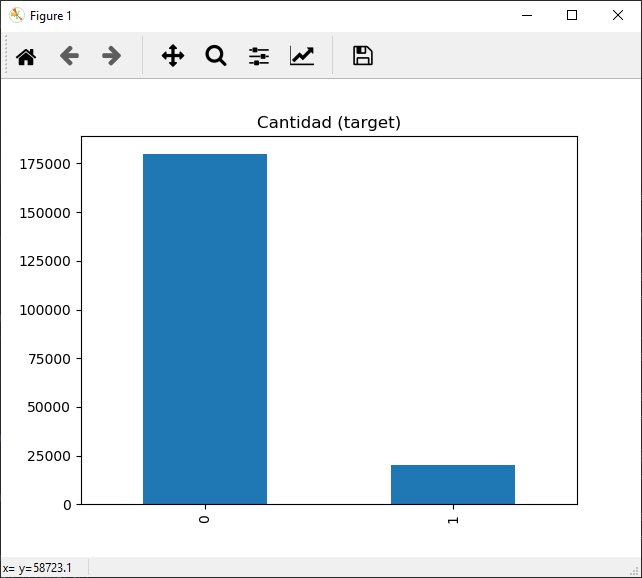
<https://www.kaggle.com/c/santander-customer-transaction-prediction>

En la misma se detallaba que a partir del conjunto de datos suministrado se debía identificar aquellos clientes que harían una transacción específica en el futuro, independientemente del monto de dinero transaccionado.

Los datos consisten en 200.000 registros y 200 variables numéricas, todas anonimizadas, consignando un target binario.

Del análisis exploratorio del conjunto de entrenamiento se observa que no existen correlaciones entre las features y las clases se encuentran desbalanceadas en la siguiente proporción:



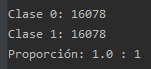


El dataset no presenta valores ausentes, y si bien se detectan algunos valores atípicos, los mismos no fueron trabajados particularmente. Se adjunta en archivo la **descripción** de las variables asociadas (“resultados\_santander.xlsx”).

# Preprocesamiento y selección del modelo

A los fines de realizar una primera aproximación a la resolución del problema, se particionó el dataset en entrenamiento y test, en forma estratificada, con un 80% de datos para entrenar y un 20% para testear y se realizó una búsqueda randomizada sobre un espacio limitado de hiperparámetros considerando los clasificadores disponibles en los algoritmos XGBoost y LightGBM. Considerando la métrica del área bajo la curva ROC, dichos algoritmos obtuvieron un promedio de 0.88 en entrenamiento, pero se degradaron hasta un 0.60 en test, con un accuracy de 0.80.

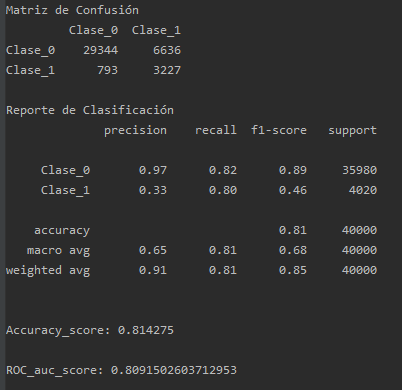
Dado este escenario, se decidió balancear las clases en entrenamiento utilizando la técnica random under-sampling, quedando el conjunto de entrenamiento distribuido de la siguiente forma:



Este conjunto de entrenamiento se trabajó utilizando la herramienta de Automated Machine Learning TPOT, considerando un tiempo máximo de ejecución de 1.080 minutos, con un tiempo máximo de evaluación de 20 minutos, optimizando bajo la métrica “roc\_auc”, con 5 fold cross validation. El resultado fue el siguiente pipeline:

StackingEstimator(estimator=GaussianNB()),  
ExtraTreesClassifier(bootstrap=True, criterion="gini", max\_features=0.6500000000000001, min\_samples\_leaf=20, min\_samples\_split=19, n\_estimators=100)

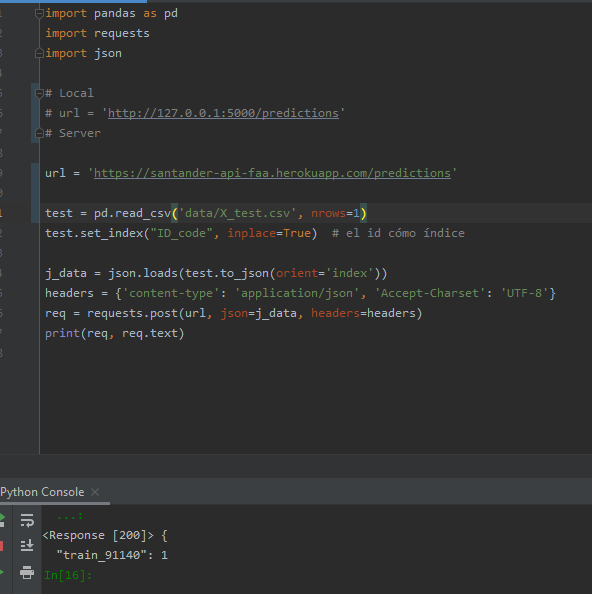
El promedio de score en entrenamiento fue de 0.88. A continuación se exponen las métricas en testing:

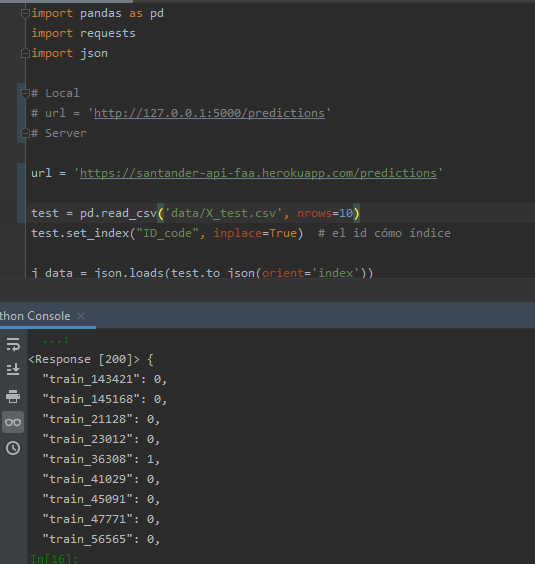


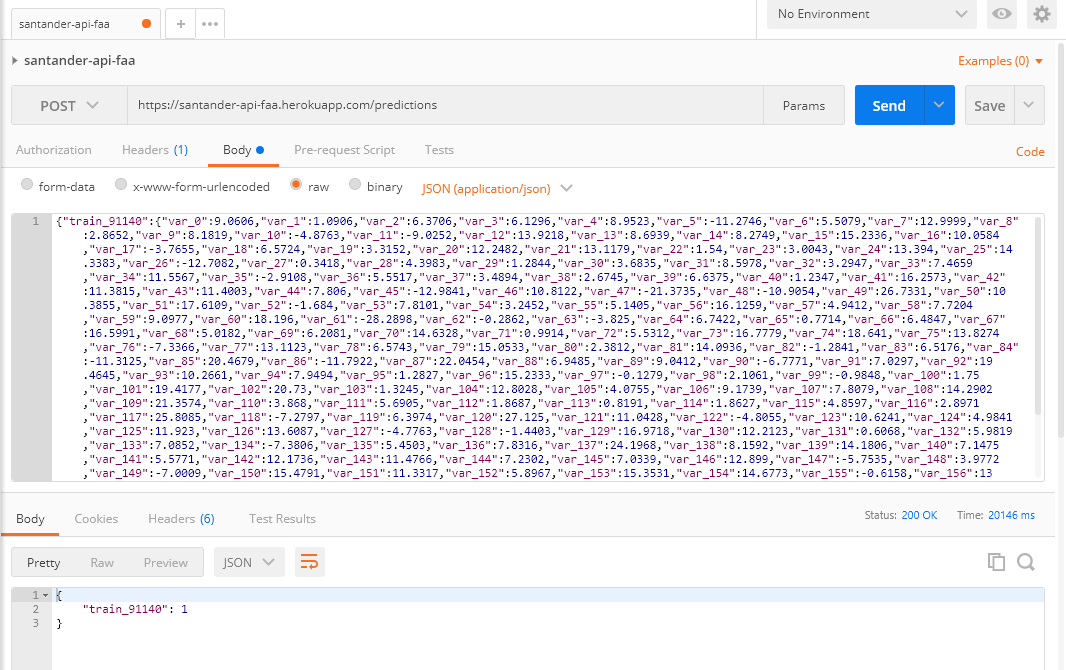
Se adjunta en archivo el detalle de las 251 iteraciones que llegó a realizar la herramienta (“resultados\_santander.xlsx”).

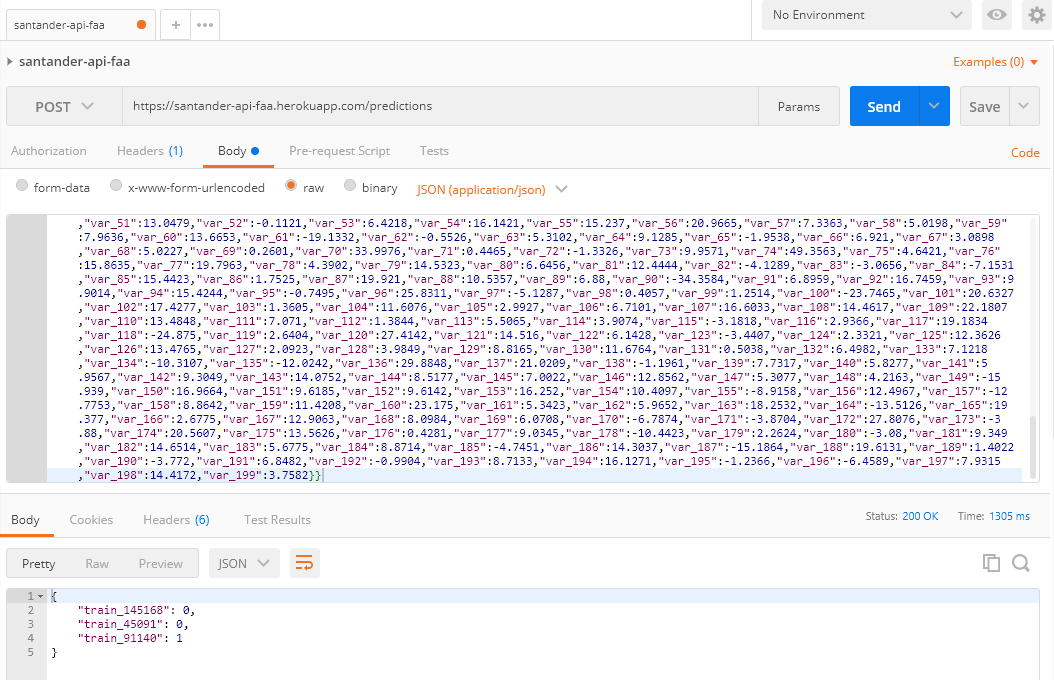
# Implementación

El modelo fue implementando a través de una API alojada en la plataforma HEROKU. La API acepta únicamente el método POST que debe enviar un JSON con al menos un identificador (id de registro) y las variables asociadas y responderá con el identificador y la predicción asociada. Se detallan a continuación ejemplos de request sobre la URL (<https://santander-api-faa.herokuapp.com/predictions>) para un único caso y para varios casos, tanto por código como por herramientas gráficas:









# Próximas tareas

Respecto del modelo desarrollado, se podría evaluar con mayor detalle las variables involucradas, buscando generar features nuevas que permitan separar mejor las clases en cuestión. Asimismo, se podría optimizar el entrenamiento permitiendo a la herramienta TPOT un mayor tiempo de ejecución, pudiéndose hallar mejores pipelines que el encontrado actualmente.

La implementación de la API podría mejorarse almacenando los datos recibidos en el POST en una base de datos, así como las predicciones asociadas a fin de evaluar la estabilidad del modelo a través del tiempo y colaborar con la mejora del mismo. También se podría trabajar en una mejora en las respuestas, pudiendo discriminar bajo pares de clave valor los identificadores y las predicciones, ayudando de esta forma en la integración de los consumidores del producto.