Inteligencia Artificial para las Ciencias e Ingenierías Proyecto de semestre - Informe final

Elaborado por:

María Isabel López Gómez, CC: 1000445792, Ingeniería Civil
Jhon Alexander Valenzuela Benavides, CC: 1214733689, Ingeniería Industrial
Juan Diego Sarria Montenegro, CC:1000549369 , Ingeniería Sanitaria

Medellín, Antioquia

2023

INTRODUCCION

Descripción del problema

La principal empresa de información financiera en Taiwán, la Taiwan Economic Journal, ofrece a las empresas los datos esenciales para realizar un análisis fundamental del mercado financiero. Sus servicios y soluciones de consultoría abarcan la gestión del riesgo crediticio y de mercado, la valoración de activos y el análisis de inversiones. Utilizando datos recopilados por esta empresa entre 1999 y 2009, se busca desarrollar un modelo predictivo para determinar el posible momento de quiebra de una empresa. La definición de quiebra se basa en las regulaciones comerciales de la bolsa de valores de Taiwán. Con el propósito de crear este modelo, se emplearán las bases de datos proporcionadas en la competencia de Kaggle. Estas bases de datos serán analizadas y organizadas adecuadamente para aplicar las métricas necesarias, con el objetivo de obtener una alta precisión que permita prever si una empresa podría enfrentar dificultades financieras.

Dataset

El dataset empleado es Company Bankruptcy Prediction https://www.kaggle.com/datasets/fedesoriano/company-bankruptcy-prediction?datasetId=1111894&sortBy=voteCount el cual consta de 6.819 instancias y 96 columnas.

Métricas de desempeño

Una vez que se entiende que el problema a resolver es un problema de clasificación binaria, se toman en cuenta las métricas, para comprender el desempeño del modelo. Las métricas definidas para medir la efectividad del modelo son las siguientes:

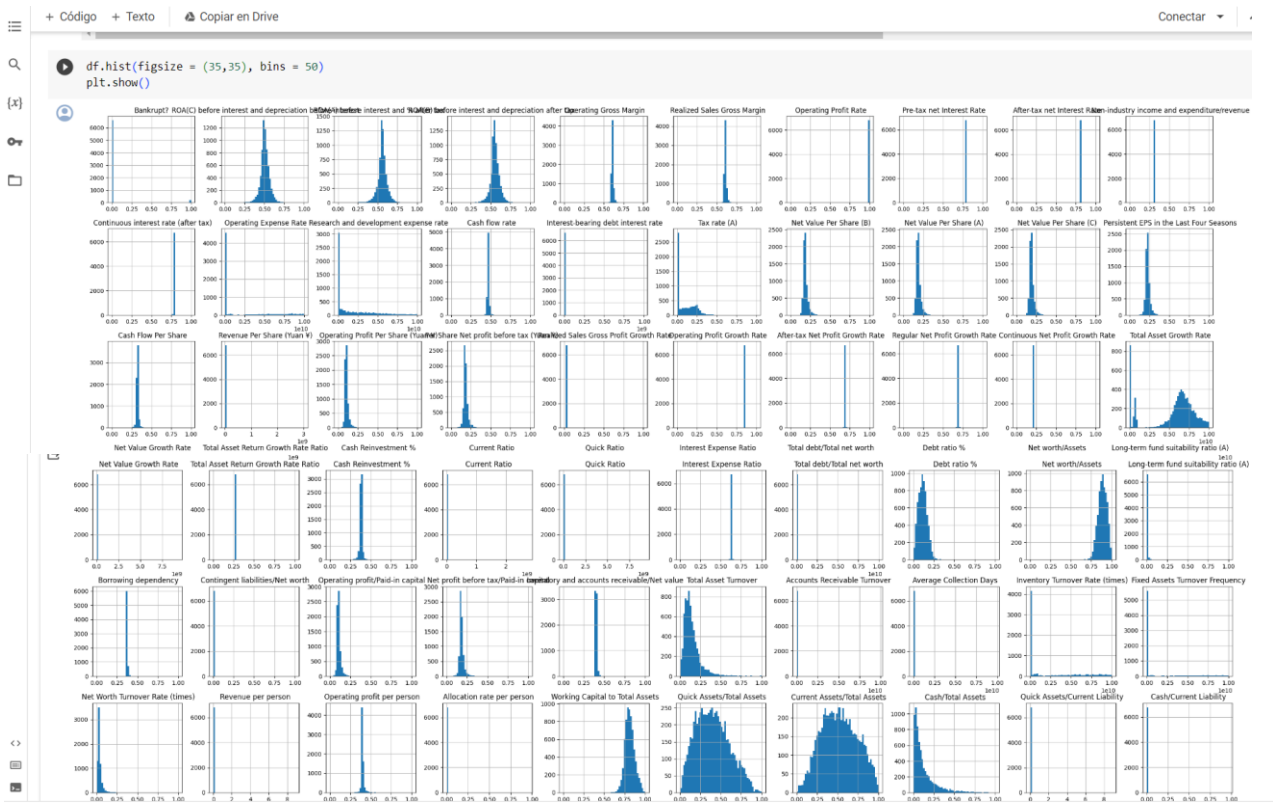
- Exactitud (Accuracy)
- Matriz de confusión
- Precisión (Precision)
- Recall (Recall)
- Puntuación F1 (F1-Score)
- Curvas ROC (ROC)

Métricas del negocio

La insolvencia de una empresa tiene repercusiones tanto a nivel empresarial como en la economía global. Predecir este estado es crucial en actividades vinculadas a entidades financieras. El propósito es anticipar la posible bancarrota de una empresa o negocio. Mejorar la efectividad y eficiencia en el análisis predictivo de la insolvencia empresarial conlleva a una toma de decisiones más precisa en cuanto a préstamos por parte de la entidad.

EXPLORACIÓN DESCRIPTIVA DEL DATASET

En el archivo 01 - exploración de datos y preprocesado.ipynb es posible evidenciar las distribuciones que presentan los datos (se agregaran algunas gráficas, las demás se encuentran en el archivo mencionado):

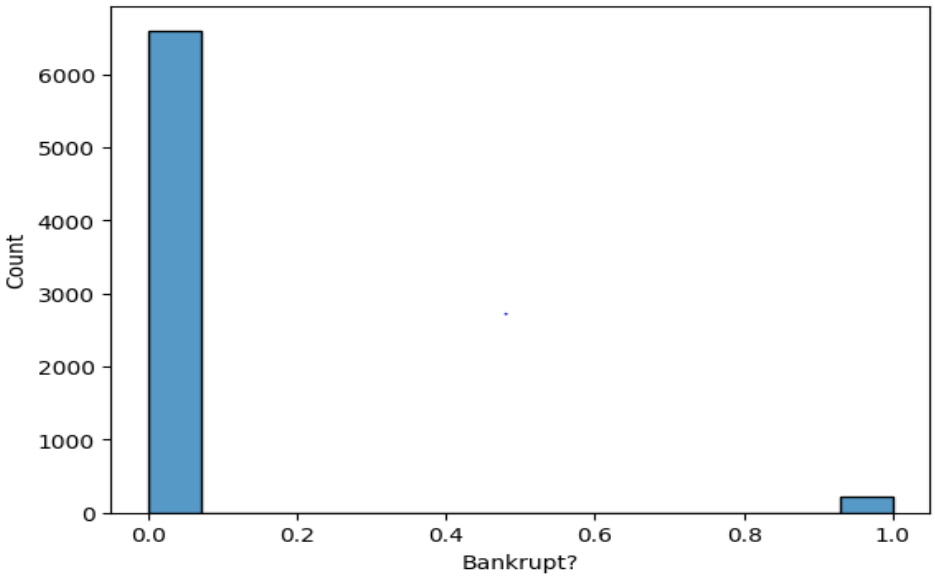


Mediante la realización del trabajo es posible evidenciar que a partir del archivo *dataset* se realiza una exploración de datos para conocer los datos, identificar patrones en los datos y graficarlos, y se encontró lo siguiente:

- **Tamaño del dataset:** El tamaño del dataset es (6819, 96).

- **Valores nulos**: Las columnas Operating Gross Margin, Total Asset Return Growth Rate Ratio, Accounts Receivable Turnover tienen 501 valores nulos.

- **Variable objetivo:** Se grafica variable de salida, se encuentra que el valor 0 (empresa no entrará en bancarrota) es el más presente en el dataset.



- **Tipos de datos:** Los tipos de datos que están presentes son float64 en su mayoría, algunos int64 y otros object en las variables que se convirtieron a categóricas.

● **Inspección de datos numéricos**: Se calcula el promedio, la desviación estándar, valor mínimo, máximo y percentiles de las columnas numéricas además, se graficó la matriz de correlaciones.

● **Inspección de variables categóricas:** En las variables transformadas a categóricas se realizó un conteo de las filas correspondientes a cada categoría (High, Medium y Low). Se incluyen también histogramas para encontrar relaciones entre la variable de salida (Bankrupt?) y cada una de las variables categóricas.

Iteraciones de desarrollo

Todas las iteraciones realizadas se hicieron con base en el dataset*. parquet,* archivo de salida generado tras haber realizado un preprocesamiento de datos previo que consistió en lo siguiente:

- **Llenado de datos faltantes:** Debido a que las 3 columnas con datos faltantes son numéricas, el proceso se realiza con la media. Esto a través de la función fillna().

- **Conversión de variables categóricas a numéricas:** Se utiliza la estrategia One Hot Encoding, la cual consiste en la creación de una columna para cada valor distinto que exista en la característica que estamos codificando y, para cada registro se marca con un 1 la columna a la que pertenezca dicho registro y se dejan las demás con un valor de 0. Esto es posible gracias a la función get_dummies().

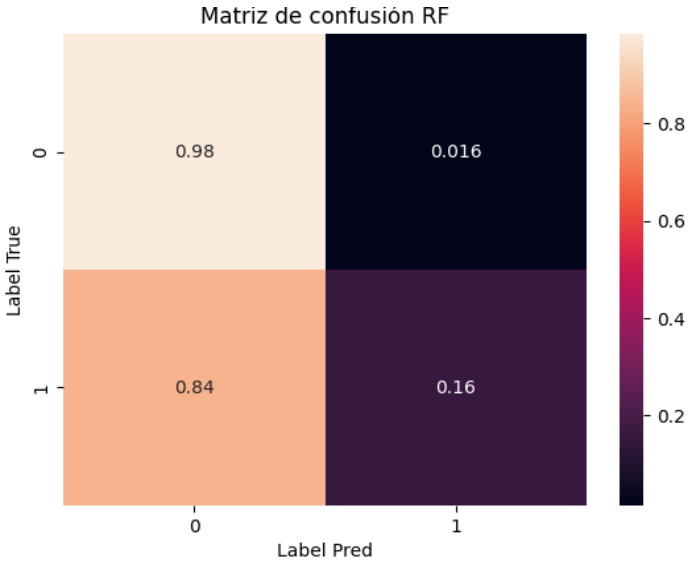
Teniendo esto en cuenta, se enuncian las distintas iteraciones:

Iteración 1: Modelos sin SMOTE

En la primera iteración se llevaron a cabo experimentos dividiendo los datos mediante StratifiedKFold, esto con el fin de que los experimentos se realicen conservando la estratificación de los datos, lo cual es posible visualizarlo en el archivo 02-entrenamiento-modelo.ipynb. Los modelos probados fueron los siguientes:

RF (Random Forest)

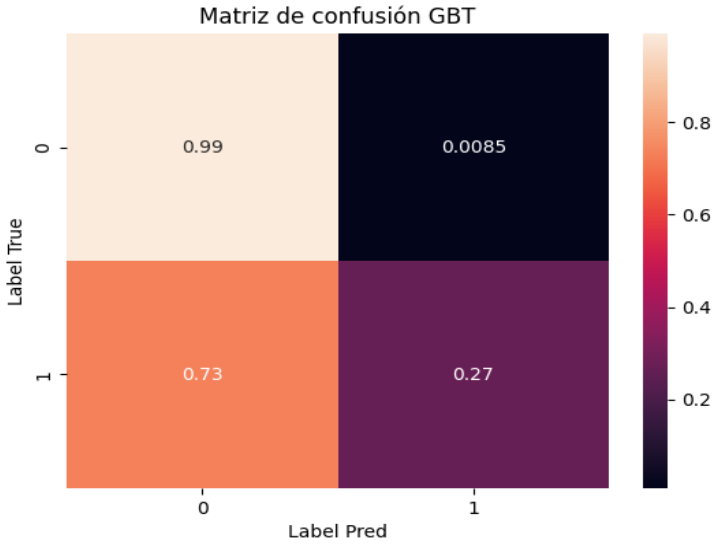
Para este modelo, se realizaron corridas con árboles [5,10,20,50,100, 150] y variables para la selección del mejor umbral [5,10,15,20,25] a través de combinaciones entre sí. Por ejemplo, con un número de árboles 50 y variables para selección de mejor umbral 5 se obtuvo una eficiencia (accuracy) de prueba de 95.63%, y en cuanto a la matriz de confusión se obtuvo lo siguiente:



Para la matriz de confusión se emplean porcentajes para visualizar mejor los resultados de las predicciones del modelo. En cuanto a los verdaderos negativos (TN) se observa que el modelo clasifica correctamente la clase 0 (la empresa no entrará en bancarrota) con el 98% de los datos pertenecientes a esa clase. A nivel de verdaderos positivos (TP) no se obtiene un resultado muy favorable, ya que se observa que el modelo predice correctamente el 16% de las veces cuando una empresa sí entrará en quiebra. En los falsos negativos se obtiene una tasa del 84% mientras que en los falsos positivos se visualiza una tasa del 1.6%.

Gradient Boosting

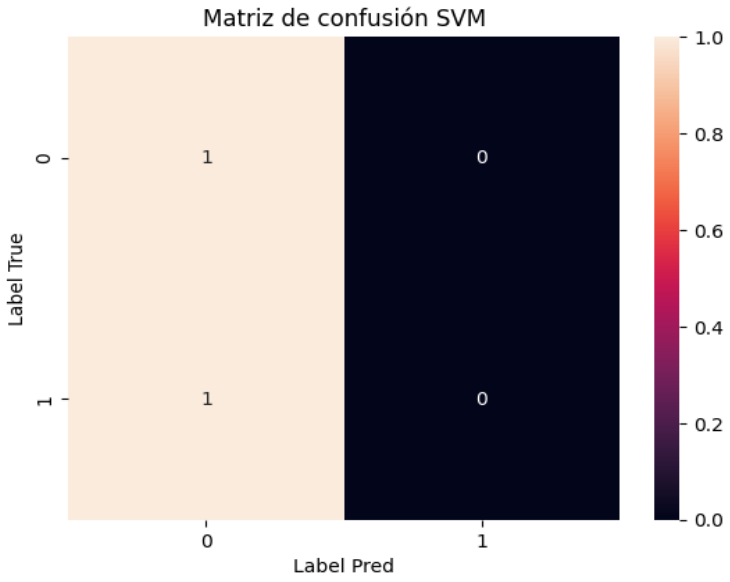
Con la función GradientBoostingClassifier de sklearn se varía el n_estimators entre los valores [20,50,100,200,300], por ejemplo, empleando un valor de n_estimators en 300, obtenemos una eficiencia de prueba de 96.5184% y con relación a la matriz de confusión se observa lo siguiente:



En cuanto a los verdaderos negativos (TN) se observa que el modelo clasifica correctamente la clase 0 (la empresa no entrará en bancarrota) con el 99% de los datos pertenecientes a esa clase. A nivel de verdaderos positivos (TP) no se obtiene un resultado muy favorable, ya que se observa que el modelo predice correctamente el 27% de las veces cuando una empresa sí entrará en quiebra. En los falsos negativos se obtiene una tasa del 73% mientras que en los falsos positivos se visualiza una tasa del 0.85%.

SVM (Support Vector Machine)

Para este caso se realizó un experimento con kernel 'rbf', gamma en 0.01 y C en 0.01 y se obtuvo una eficiencia de prueba de 96.7737% y la siguiente matriz de confusión:

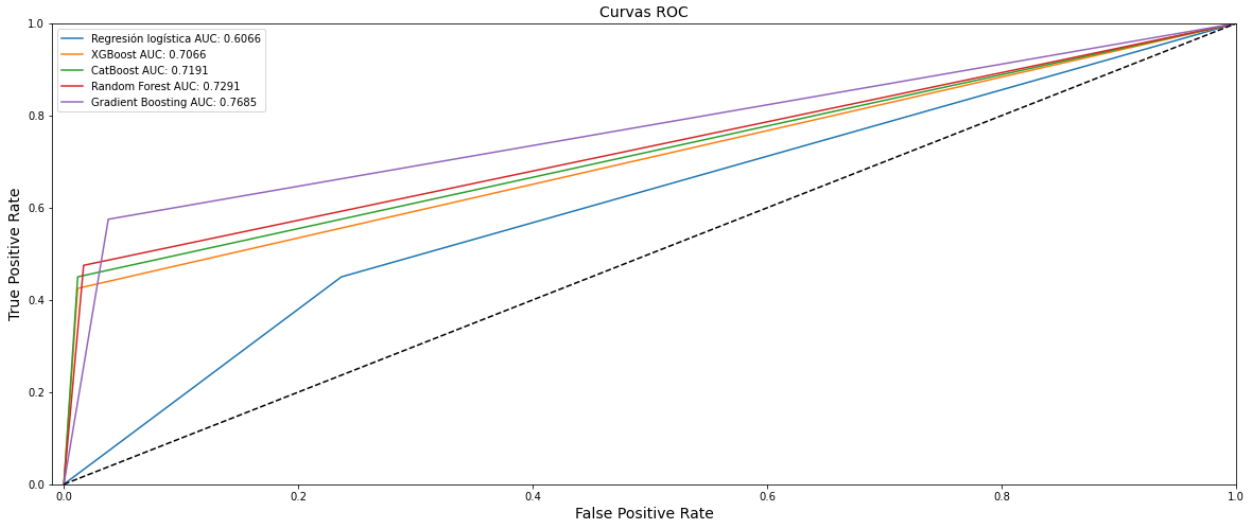


Para este caso, la matriz de confusión muestra que el 100% el modelo realizó predicciones con el valor de 0 (empresa no entrará en bancarrota), un resultado no muy esperado.

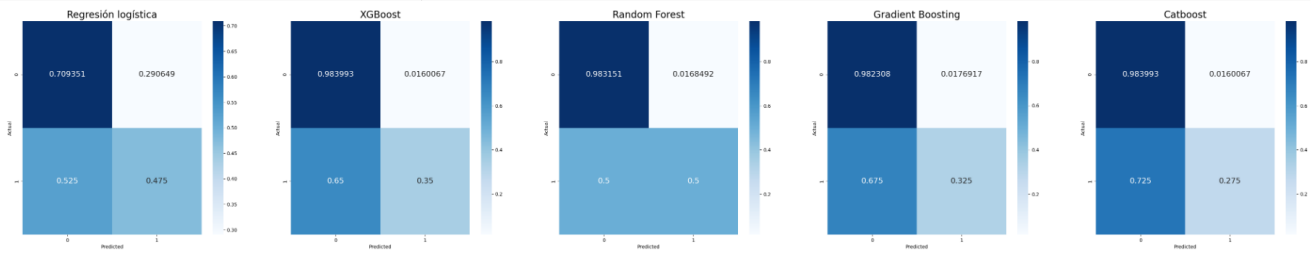
Iteración 2: Modelos con SMOTE

Esta iteración puede ser observada en el notebook 02-entrenamiento-modelo.ipynb en el cual inicialmente se llevó a cabo una división de datos de train y de test. Además, para esta se buscaba mejorar el desempeño de los modelos a través de nuevas técnicas como SMOTE la cual es una técnica estadística de sobremuestreo de minorías sintéticas para aumentar el número de casos de un conjunto de datos de forma equilibrada. Además de esto también se dividieron los datos mediante StratifiedKFold (n_splits en 5), para que los experimentos se realicen conservando la estratificación de los datos.

Posterior a esto se realizaron experimentos con los modelos regresión logística, XGBoost, Random Forest, Gradient Boosting y Catboost haciendo uso de la función RandomizedSearchCV para encontrar los parámetros óptimos de cada modelo mencionado. A continuación, se observan las curvas ROC de los cinco modelos con parámetros óptimos:

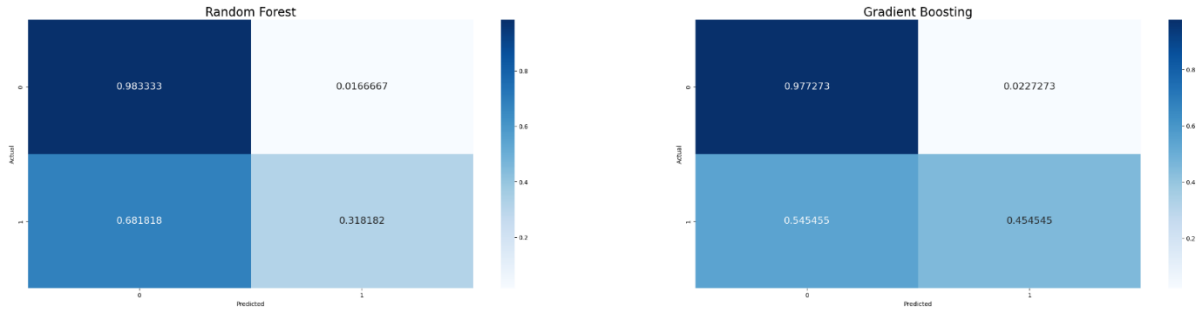


Es posible evidenciar el modelo con mejor AUC es Gradient Boosting Tree, y de segundo se encuentra Random Forest con un valor de 0.7291. Además de esto se graficaron las matrices de confusión a través del parámetro ‘normalize’ en true, para así observar los porcentajes y poder comparar con los resultados de la anterior iteración:



Lo que se buscaba mejorar para esta iteración eran los verdaderos positivos (TP) ya que esta era una falencia para nosotros en la anterior iteración, por lo cual, optamos por elegir los modelos Random Forest en el que se observa que el modelo predice correctamente el 50% de las veces cuando una empresa sí entrará en quiebra además del modelo Gradient Boosting con el cual se evidencia que el modelo predice correctamente el 32.5% de las veces cuando una empresa sí entrará en bancarrota.

Con esta elección, se emplean los datos de testing separados inicialmente para validar estos dos modelos, obteniéndose así las matrices de confusión:



Por último, se hace un reporte de las métricas de ambos:

Metricas de Random Forest

```
[ ]  print(classification_report(y_test, test_pred_rfc, target_names=label))
```

	precision	recall	f1-score	support
Fin.Stable	0.98	0.98	0.98	660
Fin.Unstable	0.39	0.32	0.35	22
accuracy			0.96	682
macro avg	0.68	0.65	0.67	682
weighted avg	0.96	0.96	0.96	682

Metricas de Gradient Boosting

```
print(classification_report(y_test, test_pred_gbc, target_names=label))
```

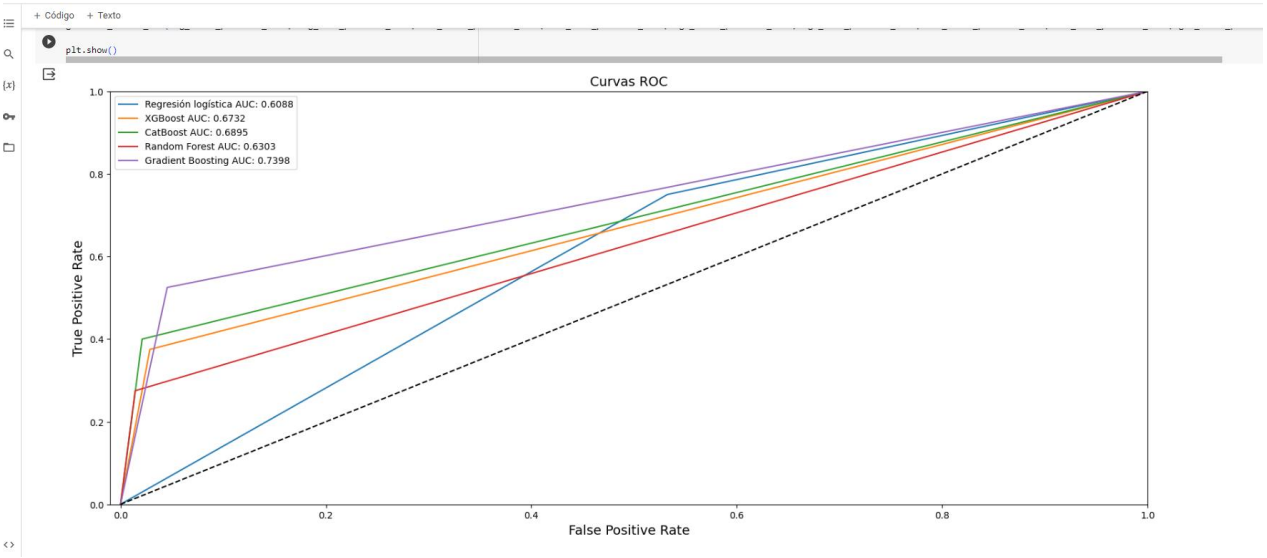
	precision	recall	f1-score	support
Fin.Stable	0.98	0.98	0.98	660
Fin.Unstable	0.40	0.45	0.43	22
accuracy			0.96	682
macro avg	0.69	0.72	0.70	682
weighted avg	0.96	0.96	0.96	682

Iteración 3: Modelos con SMOTE y PCA

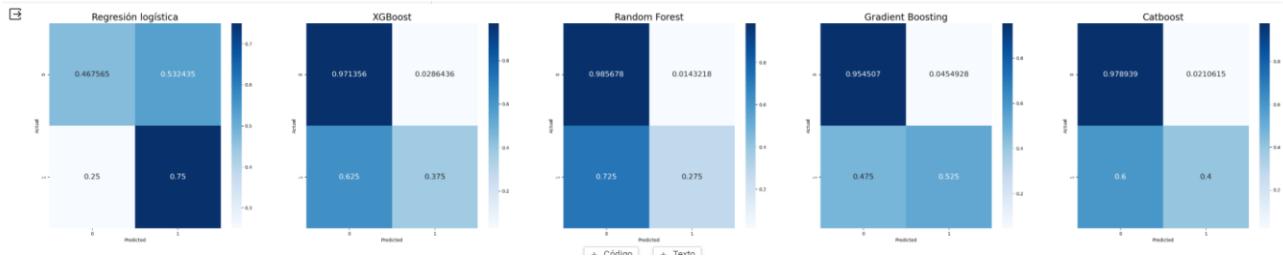
La presente iteración puede ser observada en el notebook 03 - Entrenamiento-modelo-parte2.ipynb, la iteración se realizó de la misma manera que en el notebook 02-entrenamiento-modelo.ipynb pero incluyéndose PCA (Análisis de componentes principales), el cual es un método estadístico que permite simplificar la complejidad de espacios muestrales con muchas dimensiones a la vez que conserva su información. Internamente se realizaron varias corridas con diversos números de componentes (n_components) y se encontró que el número óptimo de componentes es 62 por lo cual en el notebook se observa la corrida con este número.

Se llevaron a cabo también experimentos con los modelos regresión logística, XGBoost, Random Forest, Gradient Boosting y Catboost haciendo uso de la función RandomizedSearchCV para encontrar los parámetros óptimos de cada modelo

mencionado. A continuación, se observan las curvas ROC de los cinco modelos con parámetros óptimos:

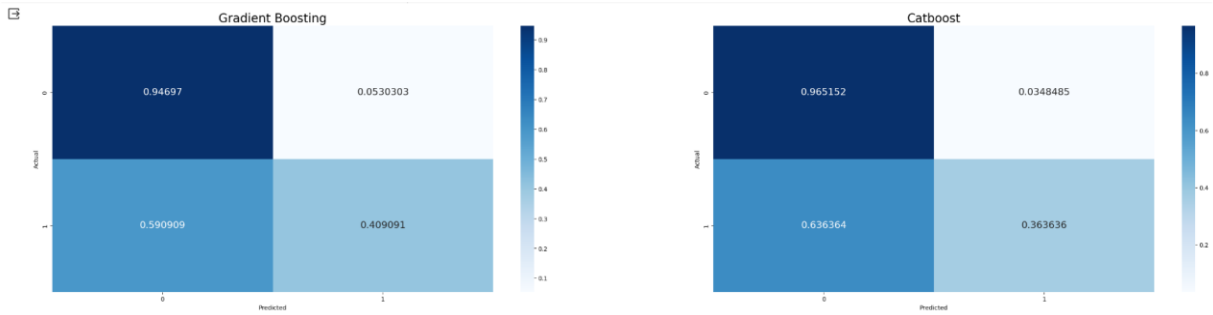


Se puede ver que el modelo con mejor AUC es CatBoost con un valor de 0.6895, y de segundo se encuentra Gradient Boosting Tree con un valor de 0.7398. A continuación, se muestran las matrices de confusión:



Se opta por elegir los modelos CatBoost en el que se observa que el modelo predice correctamente el 40% de las veces cuando una empresa sí entrará en quiebra además del modelo Gradient Boosting con el cual se encuentra que el modelo predice correctamente el 52.5% de las veces cuando una empresa sí entrará en bancarrota.

Con esta elección, se emplean los datos de testing separados inicialmente para validar estos dos modelos, obteniéndose así las matrices de confusión:



Por último, se hace un reporte de las métricas de ambos:

Metricas de CatBoost

```
print(classification_report(y_test, test_pred_gbc, target_names=label))
```

```
              precision    recall  f1-score   support

   Fin.Stable       0.98      0.95      0.96       660
 Fin.Unstable       0.20      0.41      0.27        22

     accuracy                           0.93       682
    macro avg       0.59      0.68      0.62       682
 weighted avg       0.95      0.93      0.94       682
```

Metricas de Gradient Boosting

```
[ ] print(classification_report(y_test, test_pred_cbt, target_names=label))
```

```
              precision    recall  f1-score   support

   Fin.Stable       0.98      0.97      0.97       660
 Fin.Unstable       0.26      0.36      0.30        22

     accuracy                           0.95       682
    macro avg       0.62      0.66      0.64       682
 weighted avg       0.96      0.95      0.95       682
```

Iteración 4: Redes neuronales

En la cuarta iteración, que se puede observar en el notebook 04-Entrenamiento-Modelo-Parte3.ipynb se tiene una red neuronal con la siguiente configuración:

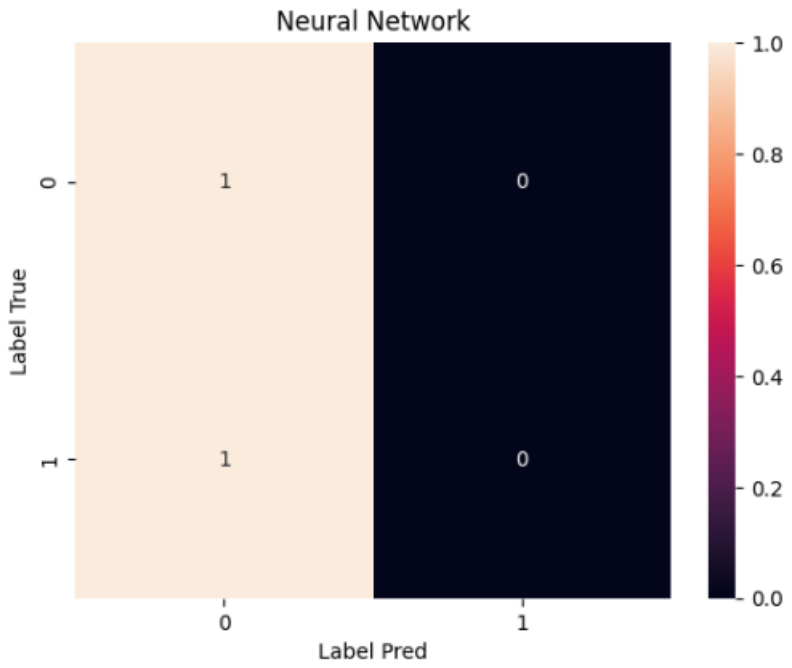
- Una capa densa de 128 neuronas y activación ReLu
- Una capa de Dropout de 0.3
- Una capa densa de 64 neuronas y activación ReLu
- Una capa densa de 32 neuronas y activación ReLu
- Una capa densa de 16 neuronas y activación ReLu
- Una capa de Dropout de 0.3
- Una capa densa de salida con 1 neurona y activación sigmoide.

Para esta configuración se utilizó el optimizador Nadam con un learning rate de 0.001, se utilizó como función de pérdida Categorical Cross Entropy. El entrenamiento se realizó con un batch size de 256 durante 30 épocas y un split de validación del 20%.

El resultado de la matriz de confusión es el siguiente:

```
calcularMatrizCF(conf_matrix,"Neural Network")
```

```
[[1 0]
 [1 0]]
```



Se observa un gran sobreajuste debido al desbalance de los datos, esto dando a entender que todos los datos se han clasificado en la clase mayoritaria (clase 0), mientras que en la clase minoritaria (clase 1) no se ha clasificado nada.

RETOS Y CONSIDERACIÓN DEL DESPLIEGUE

Los desafíos al inicio del proyecto en términos de desempeño del modelo fueron:

- La tasa de éxito es del 85%. Valor de sensibilidad superior al 80%.
- El valor de especificidad es superior al 90%.

La tasa de éxito es superior al 85% porque los modelos clasifican como desequilibrados, y la clase mayoritaria es la que tiene un rendimiento muy superior al 85%, por lo que no es una métrica que tenga mucho efecto en la implementación del modelo. Todos los modelos tienen valores de sensibilidad inferiores al 80%, por lo que no se cumple el requisito de esta métrica de mostrar la proporción de verdaderos positivos clasificados. Todos los modelos tienen valores de especificidad inferiores al 90%, por lo que no se cumple el requisito de esta métrica de mostrar la proporción de verdaderos negativos. La conclusión es que los modelos no se pueden implementar porque no se han cumplido los requisitos mínimos para un buen rendimiento a nivel de producción.

CONCLUSIONES

- La manera más efectiva de construir un modelo de machine learning de calidad implica la aplicación de una variedad de algoritmos predictivos y la exploración de combinaciones de hiperparámetros. Esto es especialmente relevante en el contexto de clasificación binaria utilizando el mismo conjunto de datos, ya que los algoritmos convencionales, como la regresión logística, no garantizan siempre los resultados óptimos. Asimismo, se sugiere evaluar minuciosamente el rendimiento de cada modelo mediante distintas métricas de validación. En este sentido, es crucial no basar las decisiones únicamente en la precisión (accuracy), sino también considerar otras métricas como el área bajo la curva (AUC), la curva ROC, la precisión y la matriz de confusión.

- Mediante técnicas de reducción de dimensionalidad como PCA, es viable lograr modelos de machine learning más simplificados que, además, pueden presentar un rendimiento mejorado tanto en términos de resultados como en el tiempo de ejecución. En nuestro caso particular, partíamos inicialmente de 114 variables de entrada, las cuales fueron reducidas a 65 componentes mediante este proceso.

- El data cleaning es una etapa que ocupa una considerable cantidad de tiempo en un proyecto de machine learning y representa un paso fundamental para lograr modelos con resultados satisfactorios.

- El tipo de red neuronal empleada no se ajusta adecuadamente a los datos, lo que la hace ineficaz para resolver el problema específico. Esta situación posiblemente se debe a que el modelo es demasiado complejo para la naturaleza de los datos empleados.

- En conjuntos de datos desequilibrados, como el que se utilizó en este proyecto, es de gran importancia aplicar técnicas de sobremuestreo como SMOTE para obtener resultados más efectivos. Durante el desarrollo del proyecto, se observó que, en términos de verdaderos positivos (TP), el modelo Random Forest tuvo una precisión del 16% al predecir correctamente las empresas en quiebra sin utilizar SMOTE. Sin embargo, al aplicar la técnica SMOTE, hubo una mejora sustancial en los verdaderos positivos (TP) utilizando el modelo CatBoost, alcanzando una precisión del 57.5% al predecir acertadamente las empresas que entrarían en bancarrota.

REFERENCIAS

- *Manuel, J., & Resumen, N. (s/f). Evaluación de estrategias de sobremuestreo utilizando SMOTE para mejorar problemas de clasificación supervisada. Edu.ar. Recuperado el 25 de noviembre de 2023, de http://www.labredes.unlu.edu.ar/sites/www.labredes.unlu.edu.ar/files/site/data/bdm/TPF-Natello-2022.pdf*

- *Caja García, O. (2020). Librería Python para el aprendizaje y la implementación de redes neuronales. Universitat Politècnica de València.*