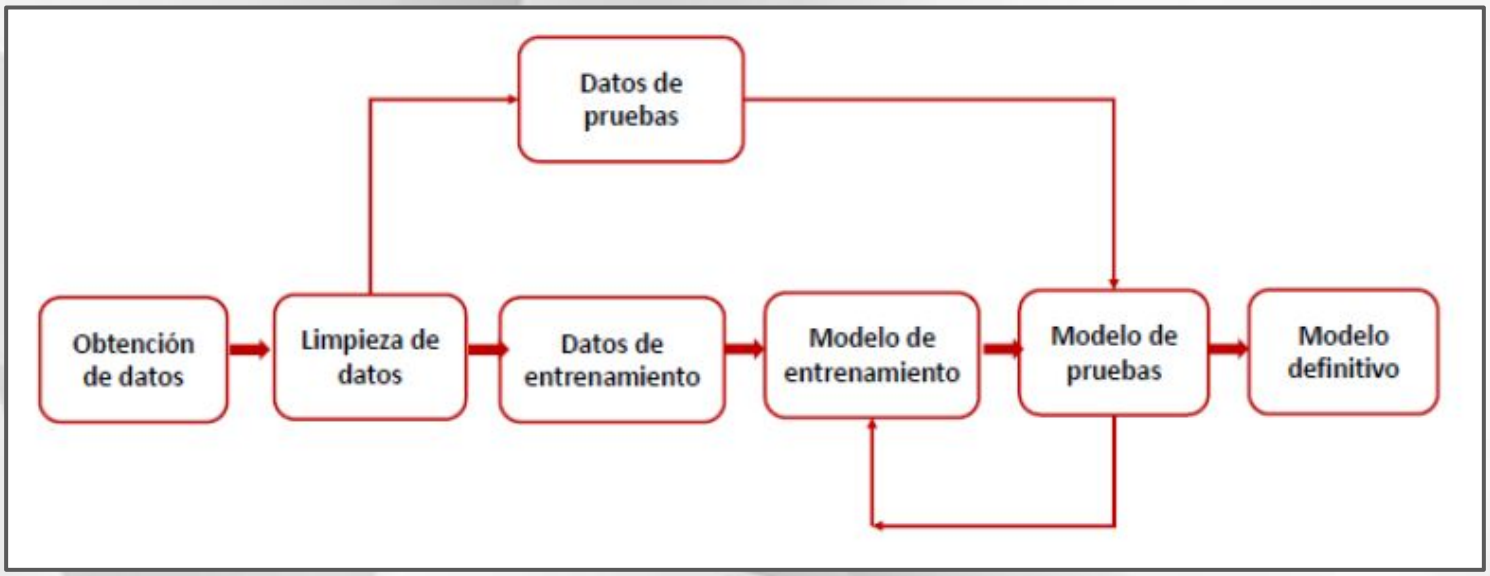
Grupo 5 - MLCP

Integrantes:

- Miguel Zambrano
- Ray Cori Illanes
- Renzo Guerrero Huayta

1. DISEÑO METODOLÓGICO

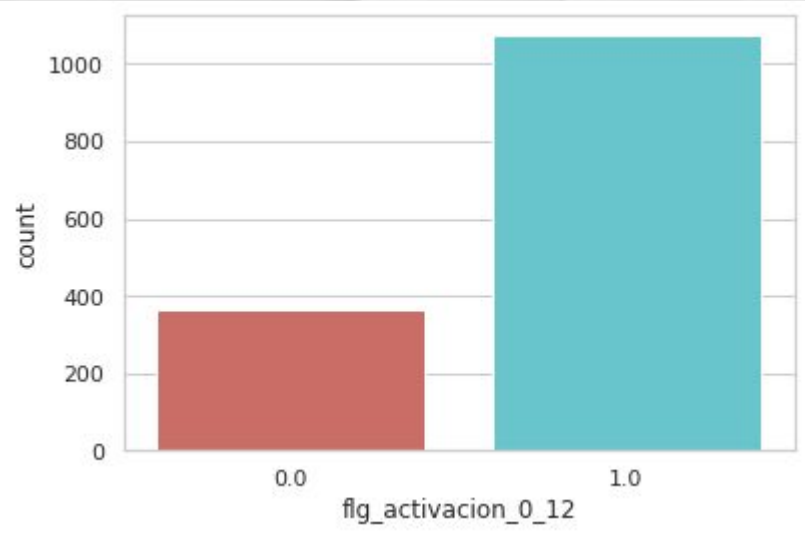


(Se trabajará con Datos de entrenamiento y pruebas con una proporción de 30 y 70 % respectivamente)

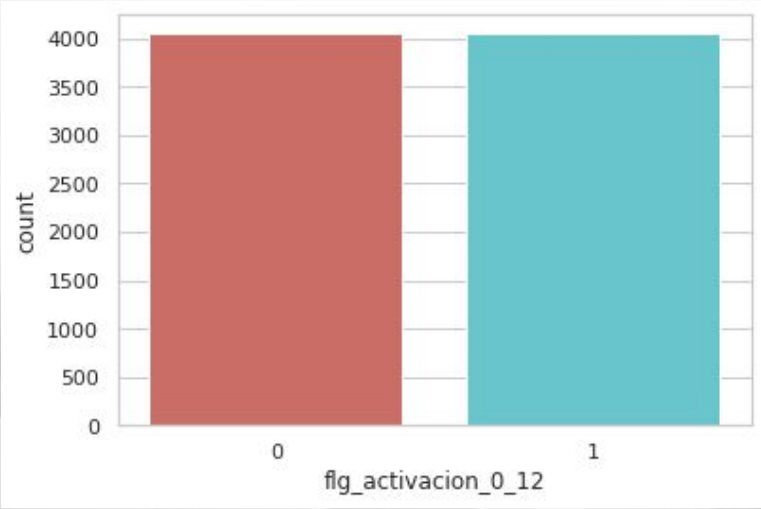
1.1. Obtención de datos : 397 var. y 7700 obs.

1.2. Limpieza de datos

→ **Desbalance de clases: aplicamos oversampling (flg_activacion_0_12)**



Inicial



Post-sampling

Tratamiento de variables específicas: Ejm : Caso var. “cat_zona1” (Uso de proporciones)

```
# Reemplazando [cat_zona1] con sus ocurrencias
cat_zona1 = (X_train["cat_zona1"].value_counts()/X_train["cat_zona1"].count()).to_dict()
X_train.replace(cat_zona1, inplace=True)
```

→ **Eliminación de variables referidas a ID**

→ **Creación de dummies :**

“tipo_cliente”

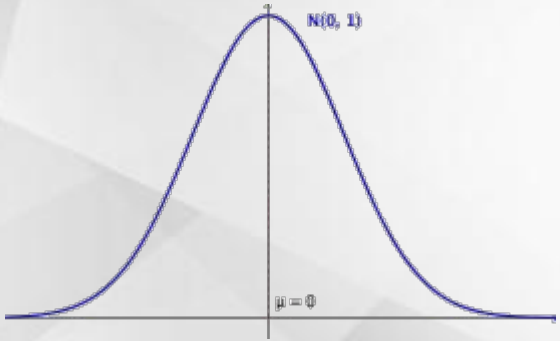
A

B

C

→ Estandarización de variables :

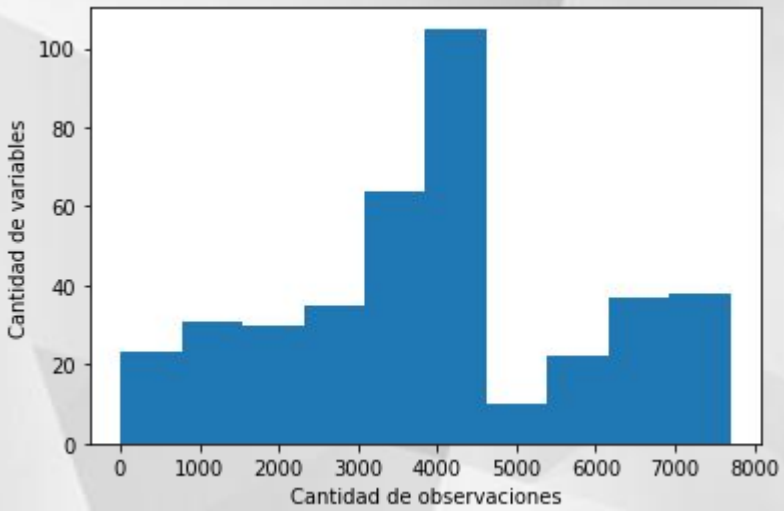
```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler().fit_transform(X_train_prepared.values)
X_train_prepared = pd.DataFrame(sc, index=X_train_prepared.index, columns=X_train_prepared.columns)
```



Razón : Con pocas excepciones (Decision Tree y Random Forest), los algoritmos de Machine Learning no funcionan bien cuando los atributos numéricos tienen escalas muy diferentes,

→ Intentos fallidos en limpieza y preprocesamiento :

a) **Eliminación de variables de acuerdo a missings (Mejora muy baja en modelamiento)**



-Bajos cambios en los scores al hacer los recortes en variables que tenían menos del 50%,40% o 30% del total de observaciones.

- Entre inputar media, moda y mediana, se optó por mediana, por arrojar mejores resultados.

b) Inputación de missings: Elecciones entre media, moda y **mediana**

```
#X_train_prepared = X_train.fillna(X_train.mode().iloc[0])
X_train_prepared = X_train.fillna(X_train.median())
#X_train_prepared = X_train.fillna(X_train.mean())
```

2. TÉCNICA ESTADÍSTICA UTILIZADA

Se harán competir los modelos para su elección y la toma de decisión :

1. Árbol de decisión
2. Random Forest
3. Regresión Logística
4. SVM Classifier
5. XG Boosting
6. Redes Neuronales (Tensorflow)

Criterios en la elección del modelo

		Predicted	
		Negative	**Positive**
Actual	**Negative**	True Negative	False Positive
	Positive	False Negative	True Positive

Precision = $\frac{\text{True Positive}}{\text{Actual Results}}$ or $\frac{\text{True Positive}}{\text{True Positive + False Positive}}$

Recall = $\frac{\text{True Positive}}{\text{Predicted Results}}$ or $\frac{\text{True Positive}}{\text{True Positive + False Negative}}$

Accuracy = $\frac{\text{True Positive + True Negative}}{\text{Total}}$

Especificity = $\frac{\text{True Negatives}}{\text{True Negatives + False Positives}}$

Problema del negocio

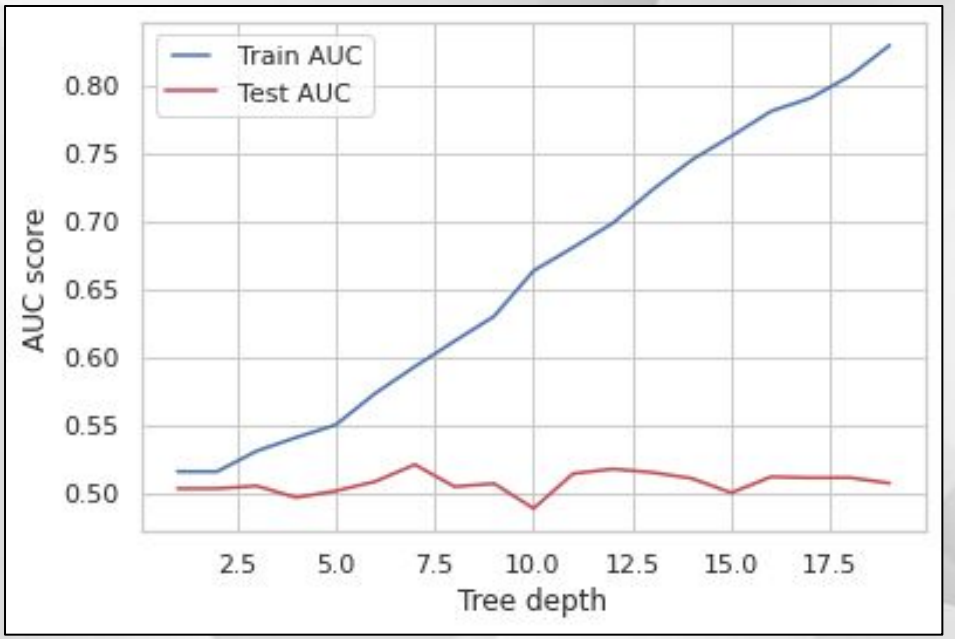
¿

Debo priorizar reducir los falsos positivos (tarjetas que el modelo dice que se entreguen, pero no se utilizarán)

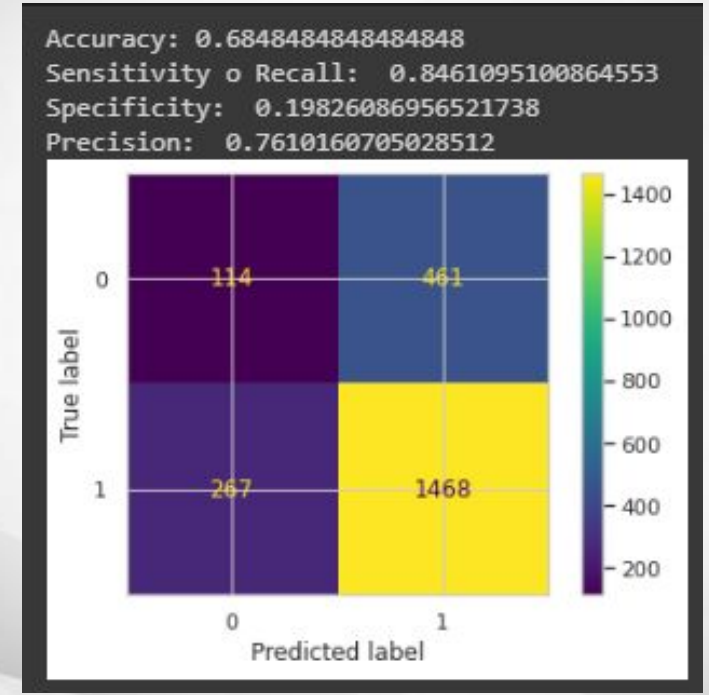
?

Cuando la base está desbalanceada el ***Accuracy*** va fallar, debemos entonces optimizar *Especificidad* y/o *Precisión* dado que en nuestro caso un Falso Negativo es más aceptable que un Falso Positivo.

2.1. Árbol de decisión

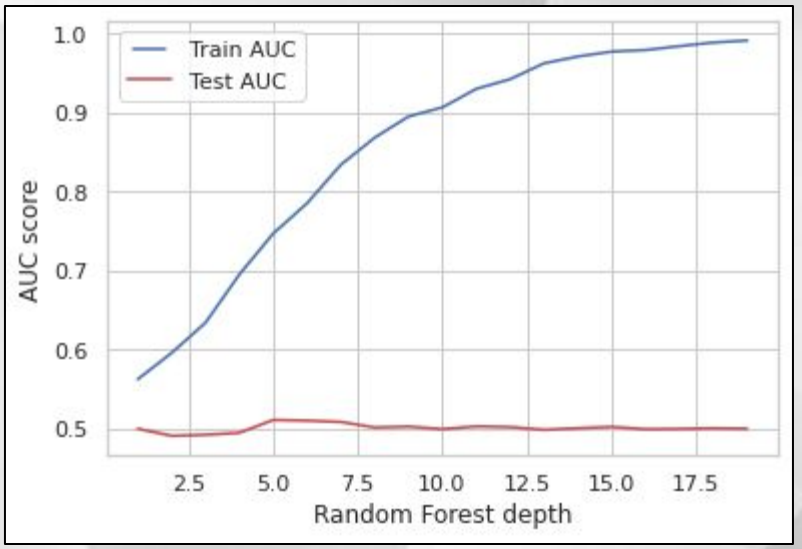


Profundidad del árbol

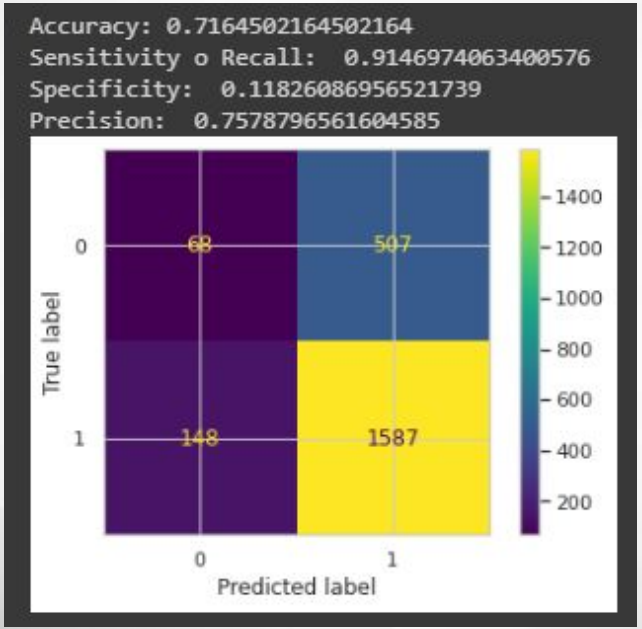


Resultados

2.2. Random Forest Classifier

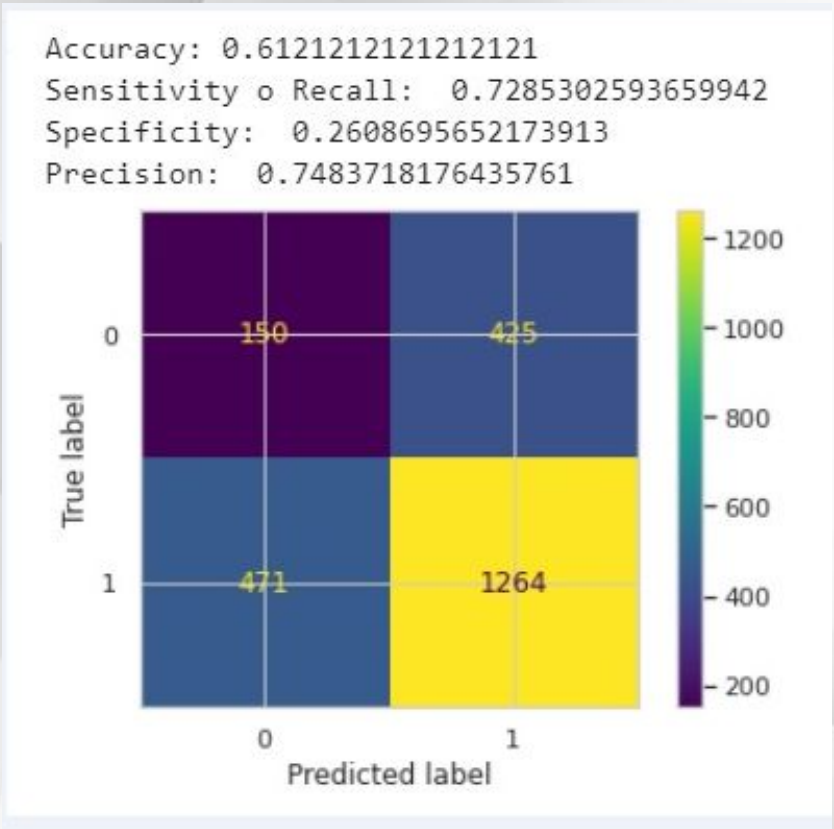


Evolución en la profundidad



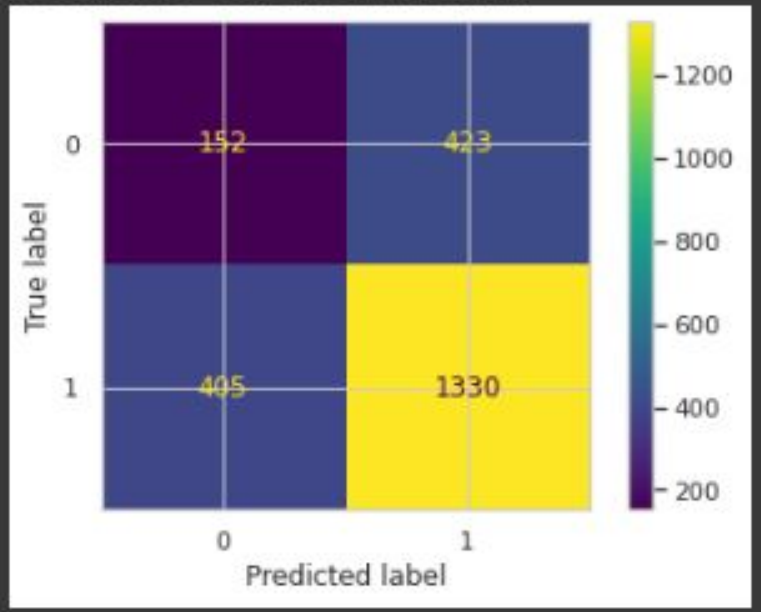
Resultados

2.3. Regresión Logística



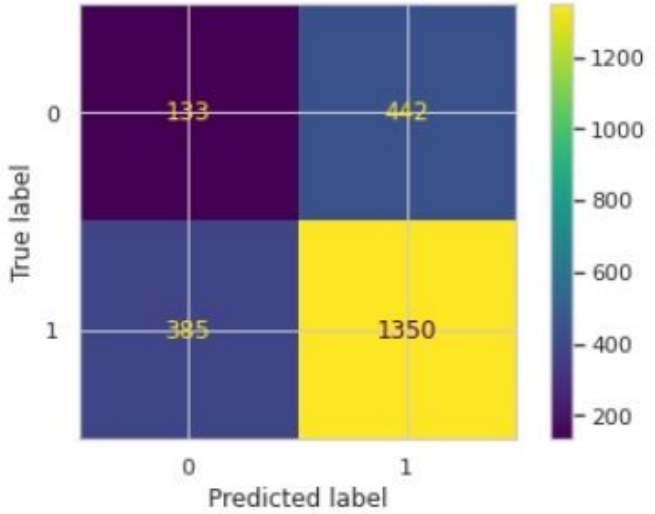
2.4. SVM

Accuracy: 0.6415584415584416
Sensitivity o Recall: 0.7665706051873199
Specificity: 0.2643478260869565
Precision: 0.7586993725042784

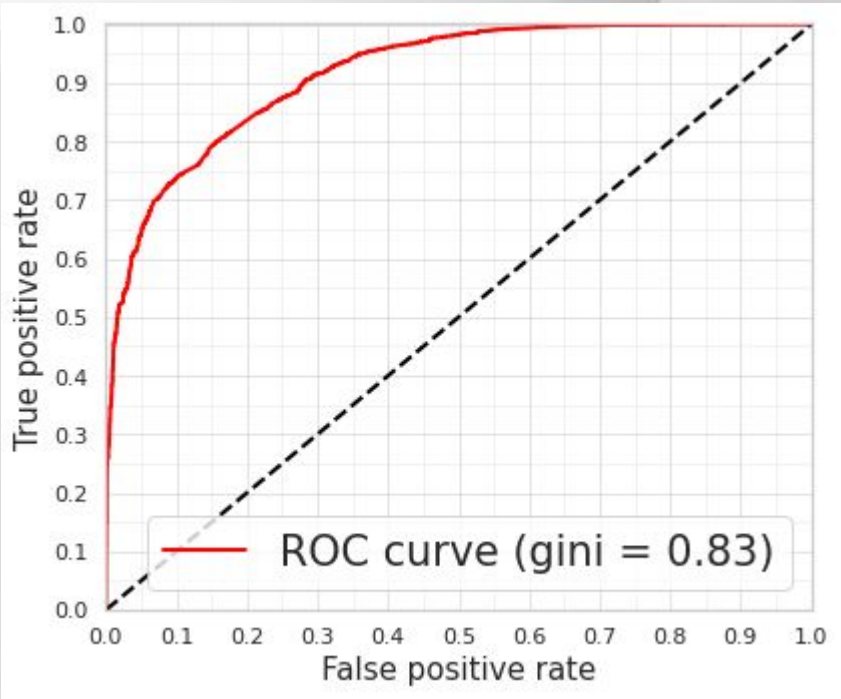


2.5. XG Boosting

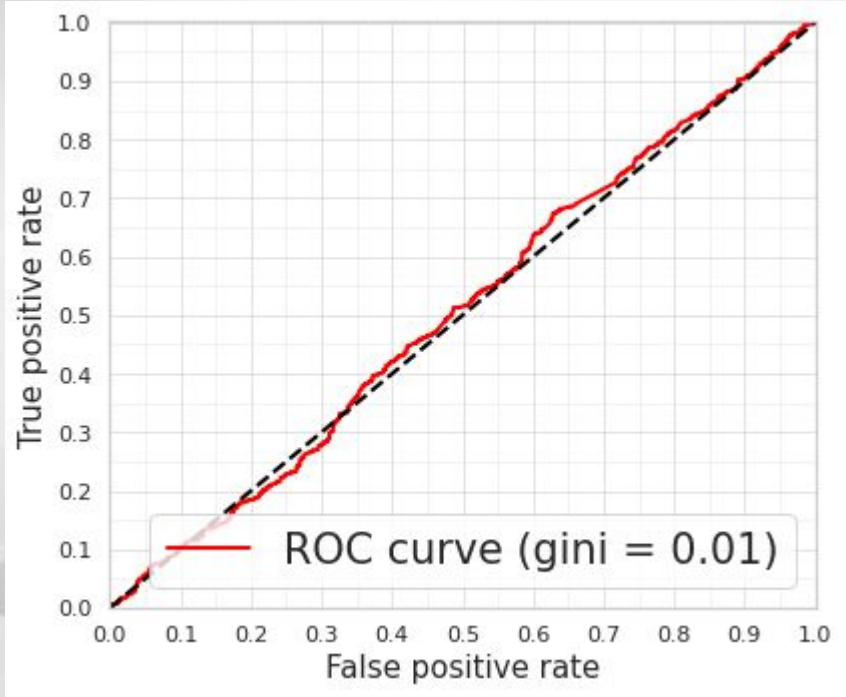
Accuracy: 0.641991341991342
Sensitivity o Recall: 0.7780979827089337
Specificity: 0.23130434782608697
Precision: 0.7533482142857143



Evolución de la curva ROC - XG Boosting

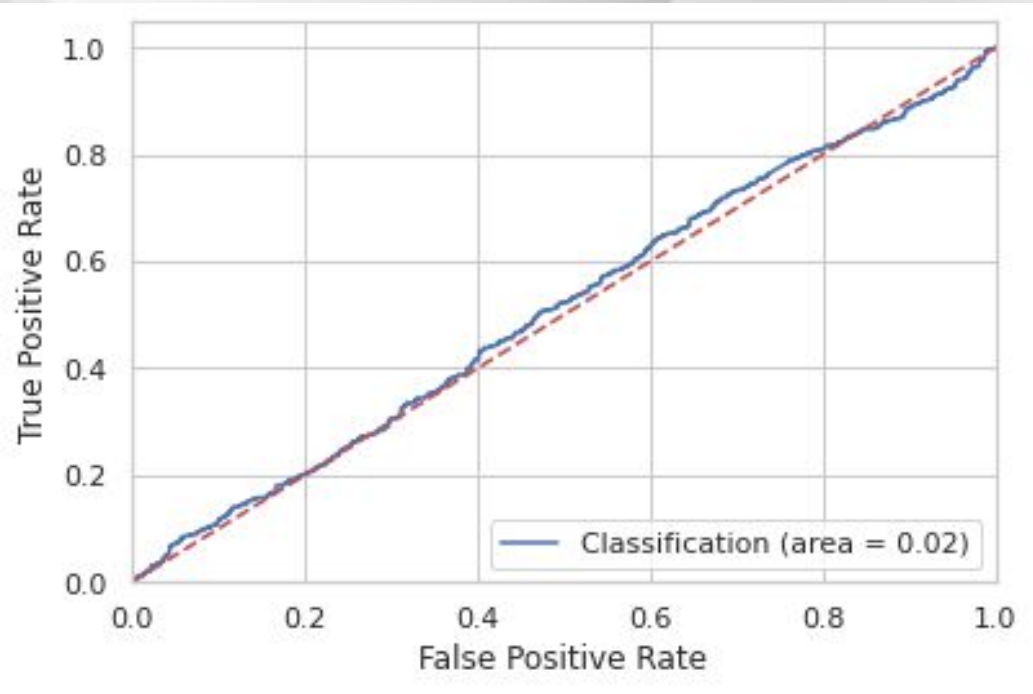


TRAIN



TEST

2.6. Redes Neuronales (Tensorflow)




```
Train: 0.044114980652294 02
 Test: 0.022268888610449622
```

3. RESULTADOS DEL MODELO

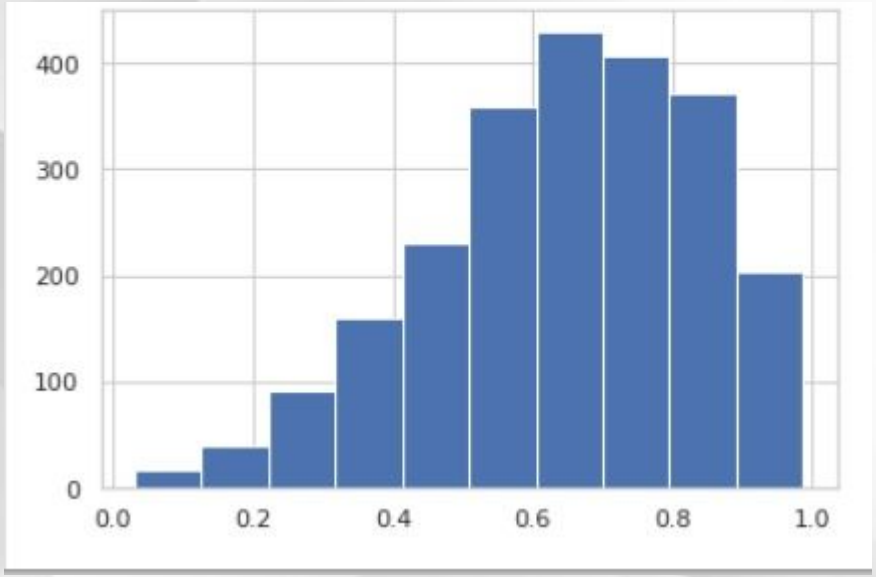
3.1. Modelo escogido : XG Boosting

- Mejor score respecto a las métricas seleccionadas :

- Capacidad computacional (el caso de SVM)

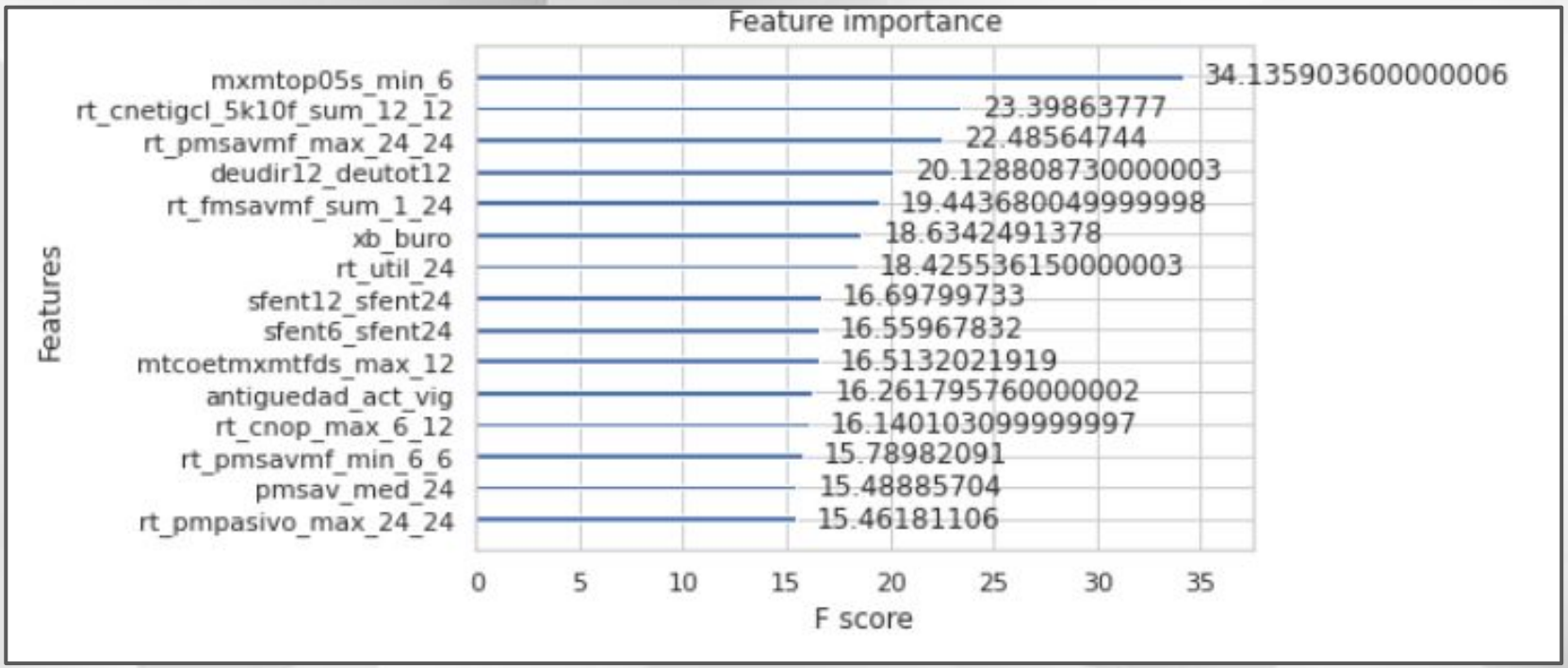
- Fácil implementación

Histograma predict_proba para XG Boosting (test)



Tomando en cuenta que el valor medio se encuentra alrededor de 0.7

3.2. Variables escogidas (Lista completa en el código presentado) :



4. PROPUESTAS DE POSIBLES USOS DEL MODELO

Compararemos el costo actual que se tiene cuando no se usa una tarjeta vs costo de no usar una tarjeta tomando en consideración las probabilidades que arroja el xgBoost.

Tipo de cliente	Cantidad	Costo unitario	Costo total
A	2620	160	2620*160*(1-P)
B	3049	160	3049*160*(1-P)
C	2031	160	2031*160*(1-P)

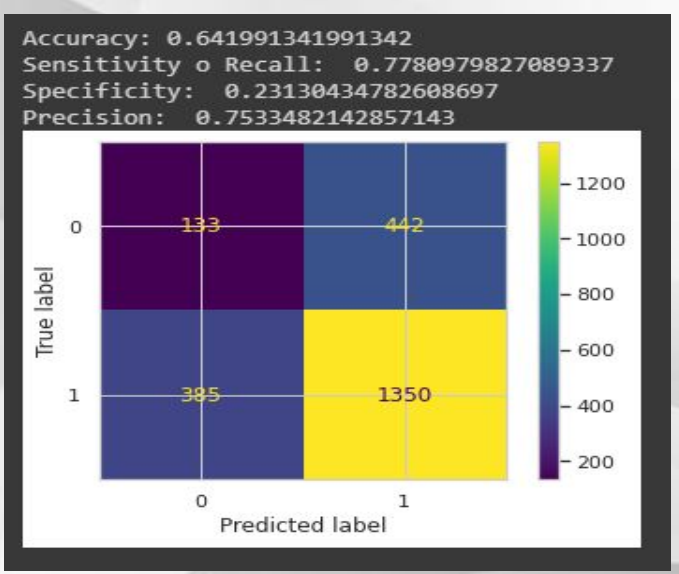
P: % de tarjetas que el banco vende a los cliente que terminan usándose

Costo base=7700*160*(1-60%)=492800

El nuevo P que usaremos se obtendrá de la matriz de confusión del xgBoost.

1-P= Falsos negativos(se le da la tarjeta pero no la usa)

Costo con el modelo=7700*160*FN=197120



CUESTIONES FINALES

Sobre la metodología

- ¿Es necesario para el gerente priorizar la capacidad de interpretabilidad de las variables sobre el mejor score ? **Depende**
- ¿Cambiarán las variables relevantes si aumentamos el número de observaciones? **Podría ser, teniendo en cuenta la data**

Sobre el tratamiento de las observaciones

- ¿Fue realmente efectivo añadir observaciones para compensar la baja proporción de la variable explicada? **Sí**
- ¿Necesitamos todavía más observaciones para mejorar el poder de modelamiento? **Es posible, tomar en cuenta el oversampling**