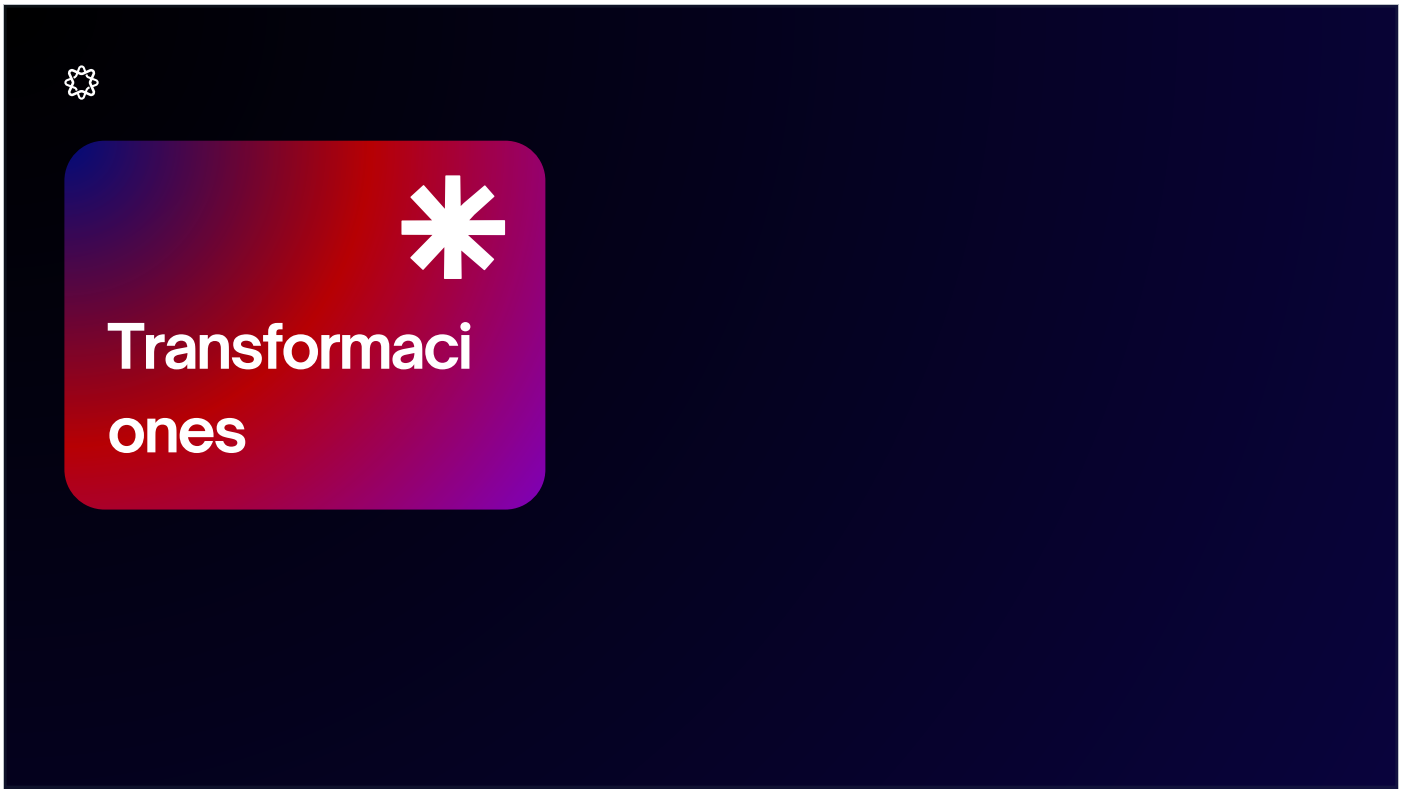
Redes neuronales secuenciales: Credit Risk

Martín Quijano - Martina Coletto

las columnas 11 y 14 no las consideramos ya que no eran relevantes


Transformaci
ones

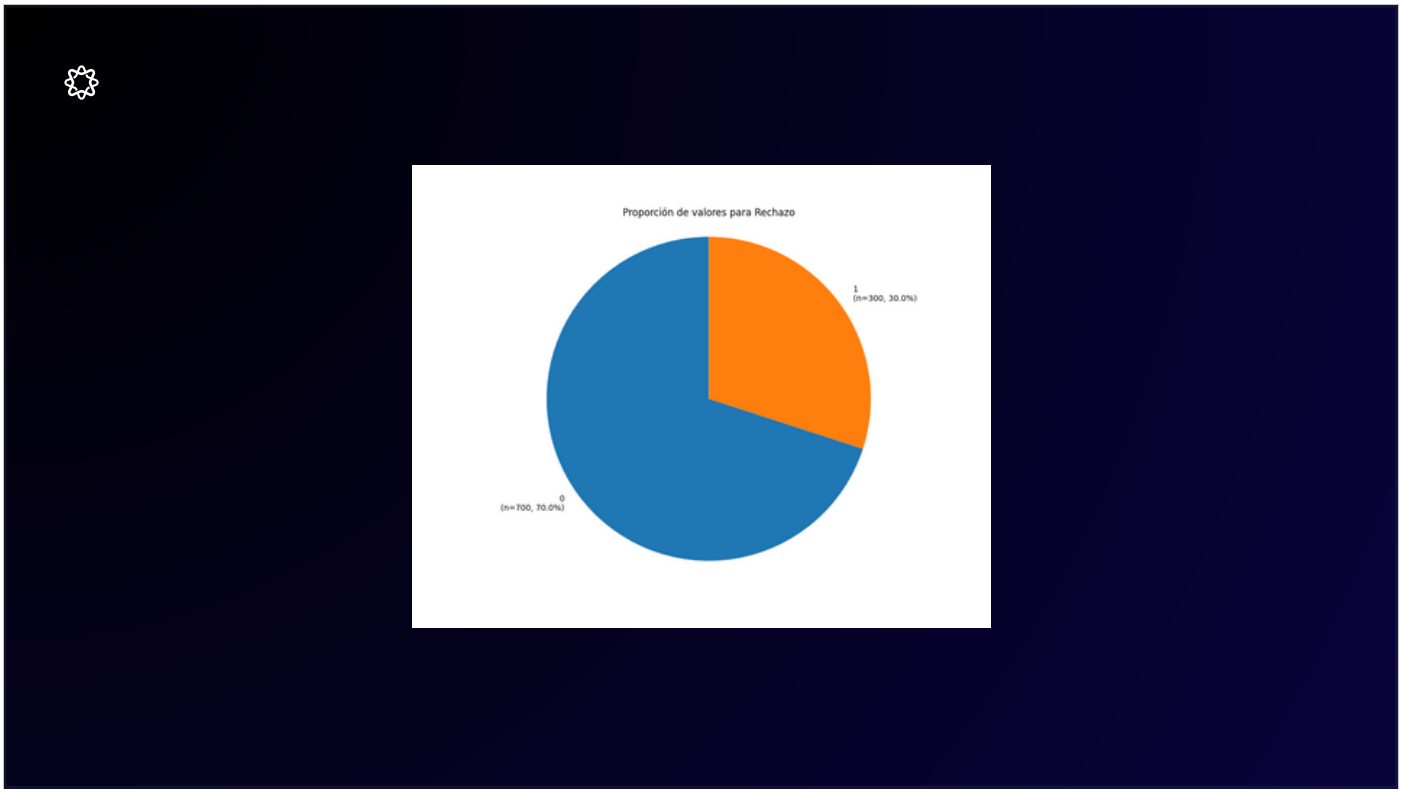
status-cuenta	A14 = 2, A11 = 1, A12 = 3, A13 = 4
duracion-meses	≤10 = 1, ≤20 = 2, ≤30 = 3, ≤40 = 4, >40 = 5
credit-history	A30 = 1, A31 = 2, A32 = 3, A33 = 4, A34 = 5
credit-purpose	A49 = 1, A48 = 2, A47 = 3, A46 = 4, A45 = 5, A44 = 6, A43 = 7, A42 = 8, A41 = 9, A40 = 10, A410 = 11
credit-amount	Sin transformar
saving-account-amount	A65 = 1, A61 = 2, A62 = 3, A63 = 2, A64 = 1
antigüedad-trabajo	A75 = 1, A74 = 2, A73 = 3, A72 = 4, A71 = 5
tasa-interés	Sin transformar
estado-civil	A91 = 1, A92 = 2, A93 = 3, A94 = 4, A95 = 5
garante	A101 = 3, A102 = 2, A103 = 1
propiedades	A124 = 4, A123 = 3, A122 = 2, A121 = 1

pasamos las categorias a numeros, manteniendo los ordenes numericos donde se podia (por ejemplo, en edad los dejamos de menor a mayor)

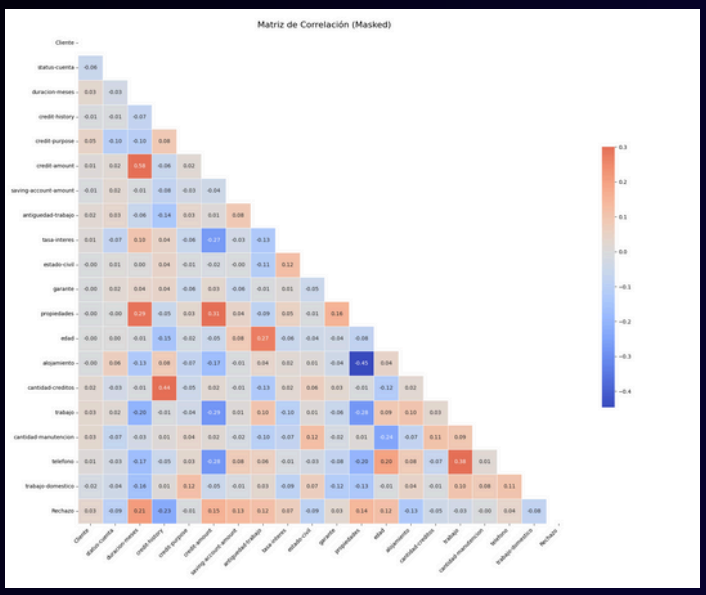
el objetivo de esto era hacer graficos que tengan labels que se puedan leer y entender mas facil

edad	Menor a 30 = 1, Mayor o igual a 30 = 0
alojamiento	A153 = 1, A151 = 2, A152 = 3
cantidad-créditos	Sin transformar
trabajo	A171 = 4, A172 = 3, A173 = 2, A174 = 1
cantidad-manutención	Sin transformar
teléfono	A191 = 1, A192 = 0
trabajo-doméstico	A201 = 0, A202 = 1
Variable objetivo (Rechazo)	1 = 0, 2 = 1

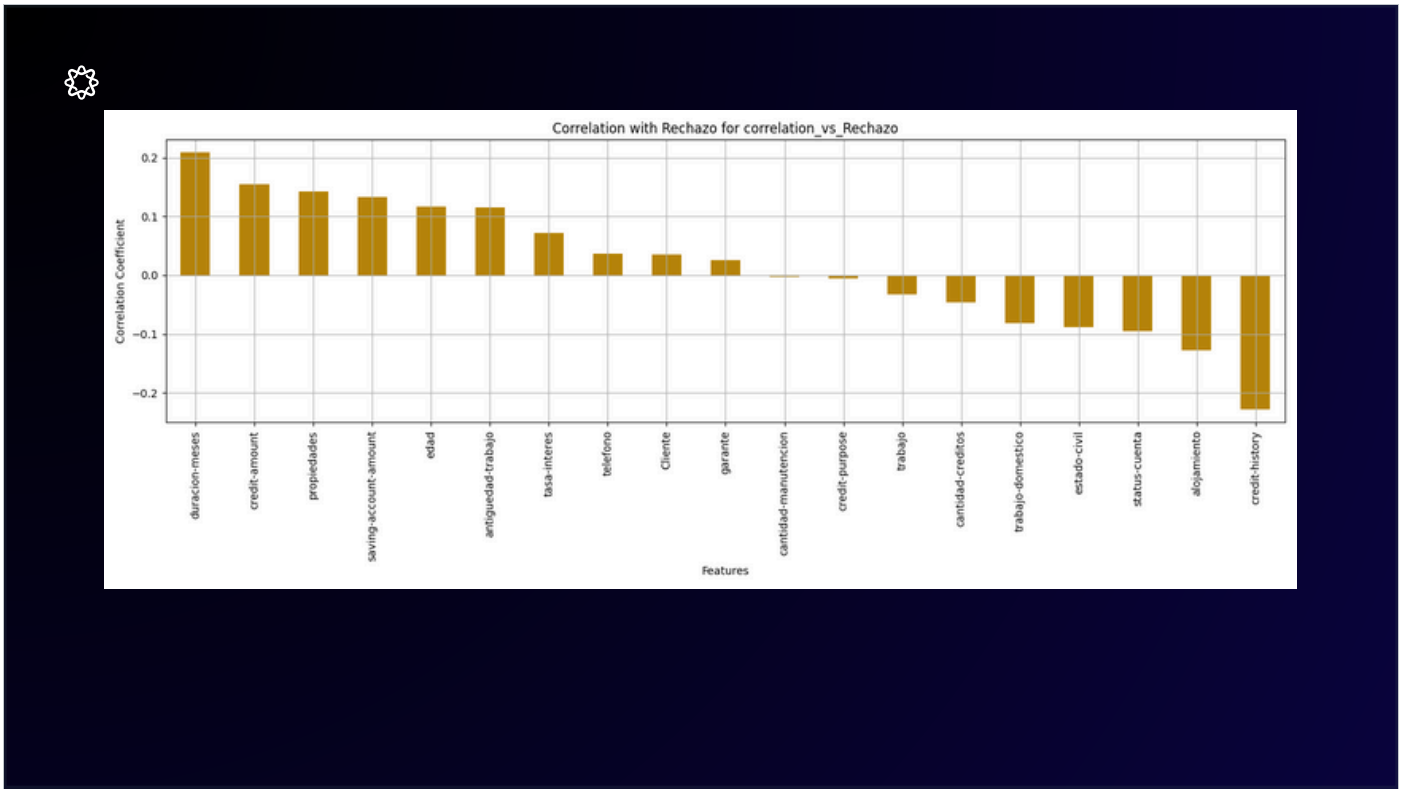

Análisis
exploratorio



Vemos que la muestra esta bastante desbalanceada, siendo 0 no rechazar y 1 es rechazar



Aca vemos como algunas de las variables tienen muy poca immportancia



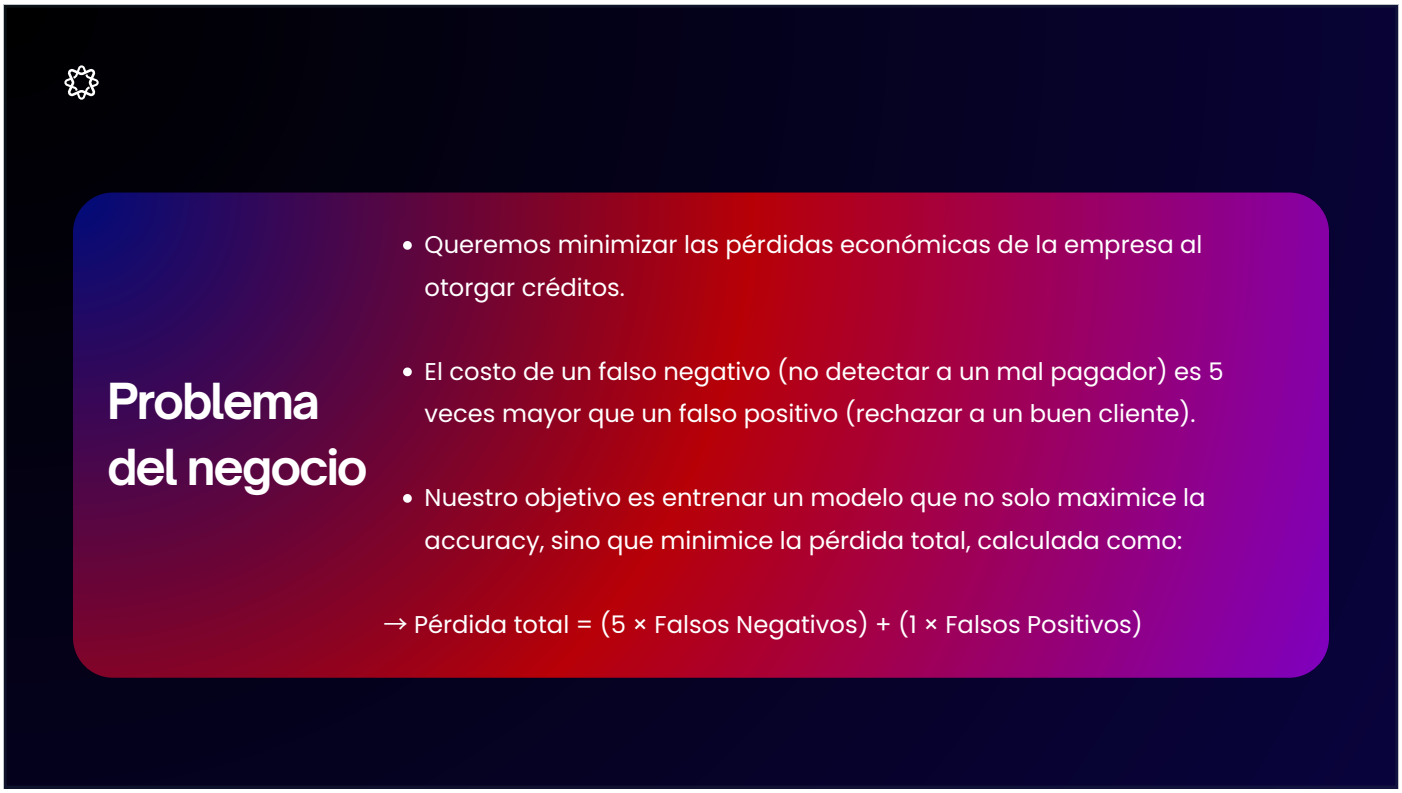
Aca vemos como algunas de las variables tienen muy poca immportancia

Information Value (IV) of Features (Target: Rechazo) - Rechazo

Feature	IV
status-cuenta	0.6660
credit-history	0.2932
duracion-meses	0.2218
credit-purpose	0.1692
saving-account-amount	0.1441
credit-amount	0.1136
propiedades	0.1126
antiguedad-trabajo	0.0864
alojamiento	0.0833
edad	0.0630
Cliente	0.0481
estado-civil	0.0447
trabajo-domestico	0.0439
garante	0.0320
tasa-interes	0.0263
cantidad-creditos	0.0133
trabajo	0.0088
telefono	0.0064
cantidad-manutencion	0.0000

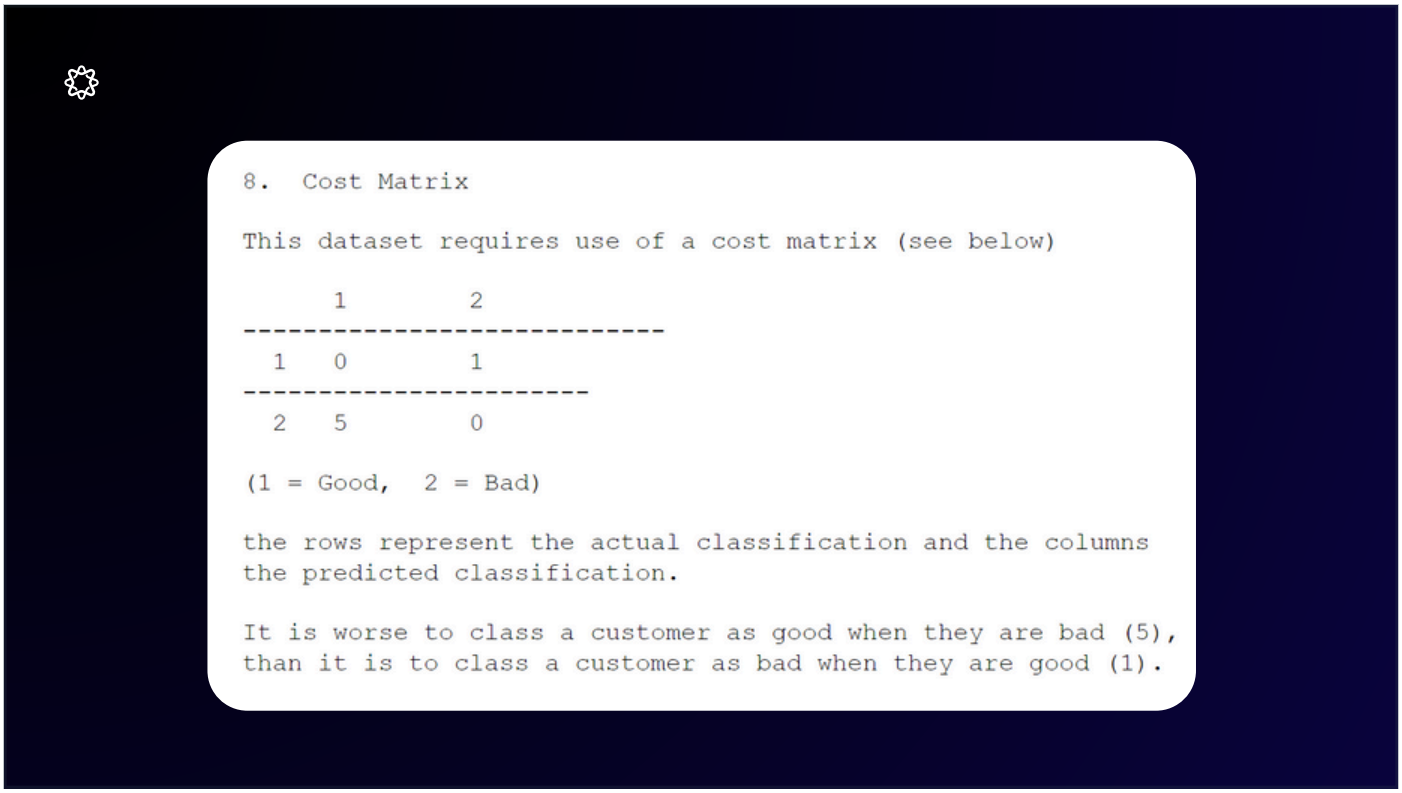
Aca vemos como algunas de las variables tienen muy poca immportancia

hacemos one hot encoding sobre las variables categoricas, ya que no queriamos que el modelo interprete relaciones lineales entre los valores de las variables



El objetivo de este modelo es minimizar el riesgo.

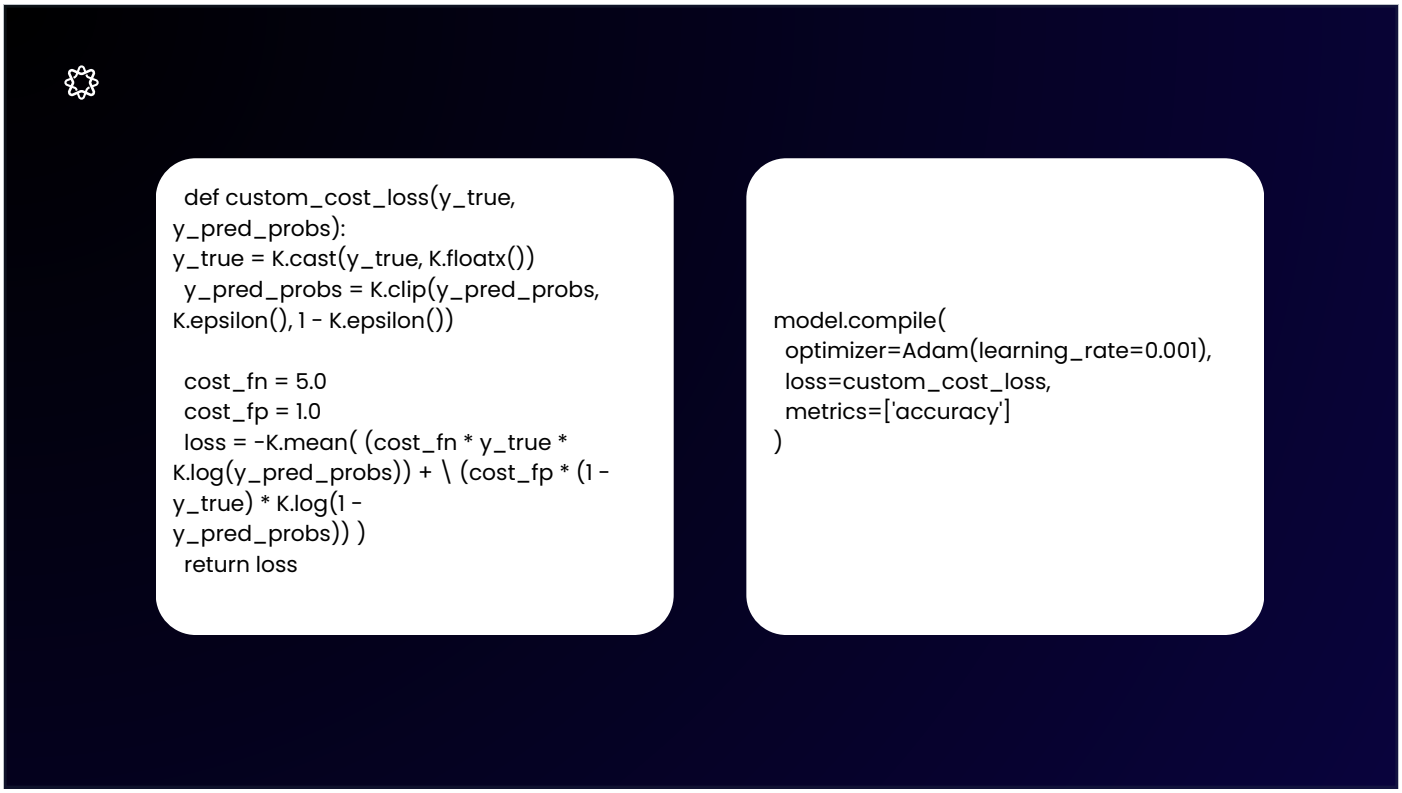
El caso decia que la perdida de un falso negativo (darle un credito a un mal cliente) era de 5, mientras que la perdida de un falso positivo (no darle un credito a un buen cliente) es de 1.



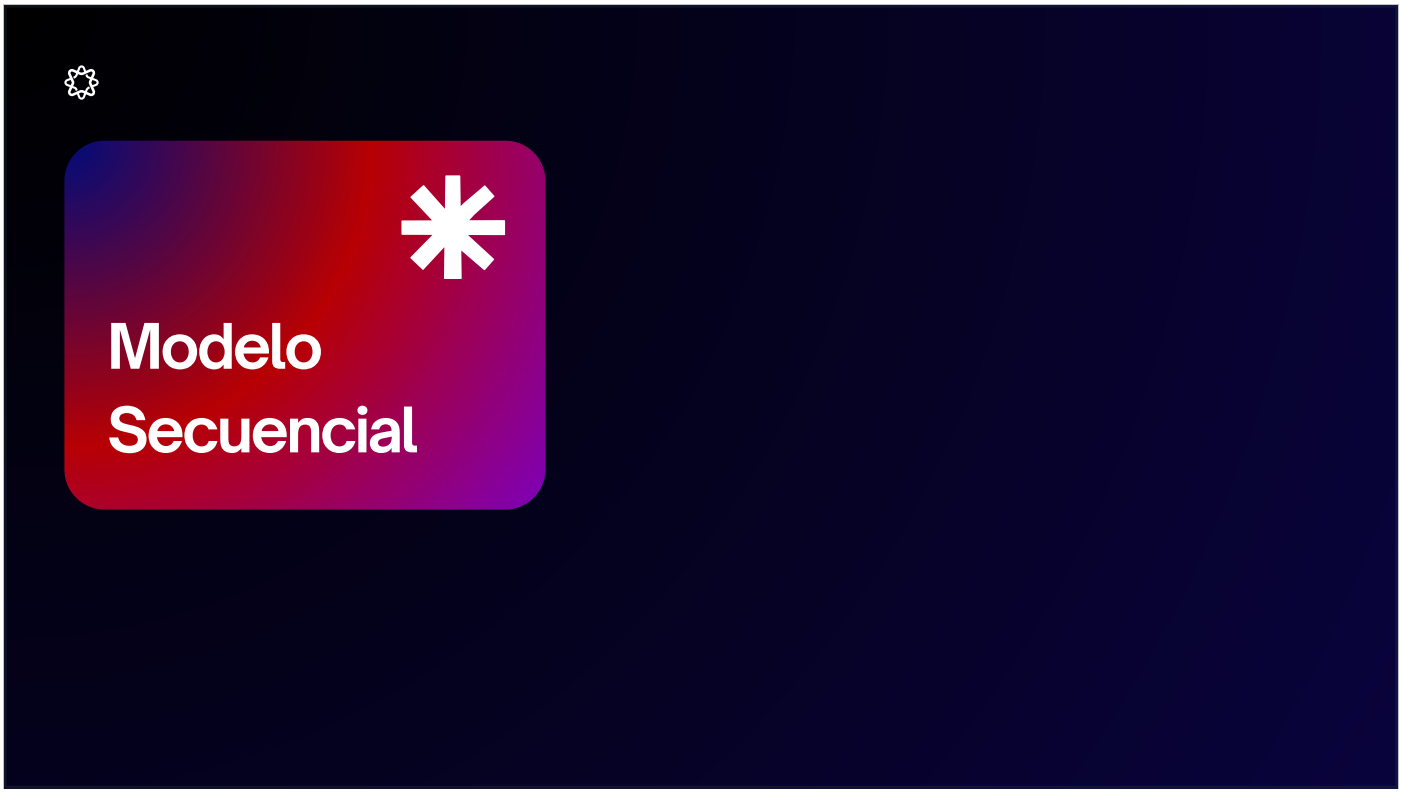
Lo que hicimos fue considerar la cost matrix que usa el banco.

Esto implica que cada persona a la que le damos un credito pero no es capaz de pagarlo, lo consideramos como un costo 5.
Y a la gente que no le damos un credito cuando deberiamos haberle dado un credito es un costo de 1.

Esto implica que el mejor modelo es un modelo que minimiza esta funcion de perdida.



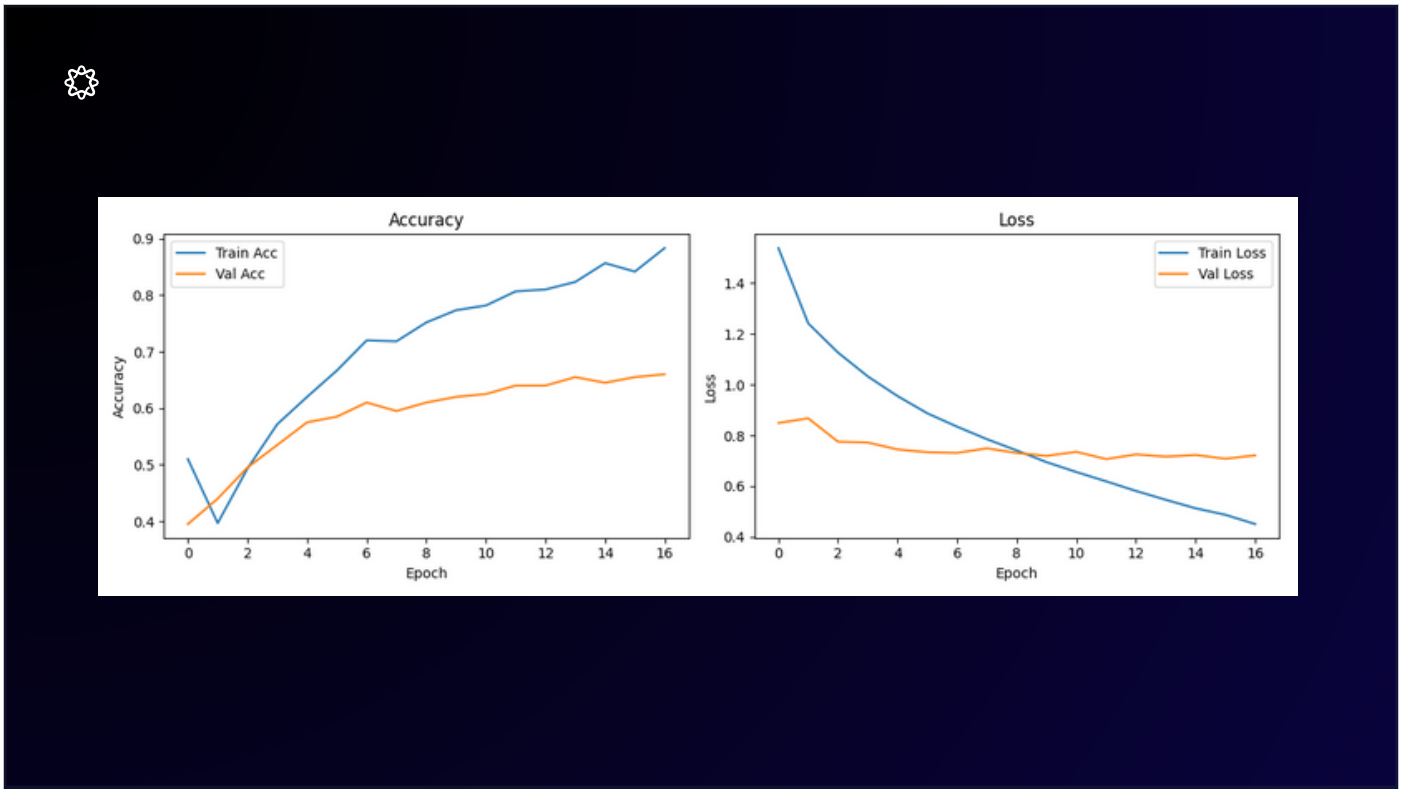
En base a eso, definimos nuestra funcion de loss y hacemos que el modelo la considere en el compile

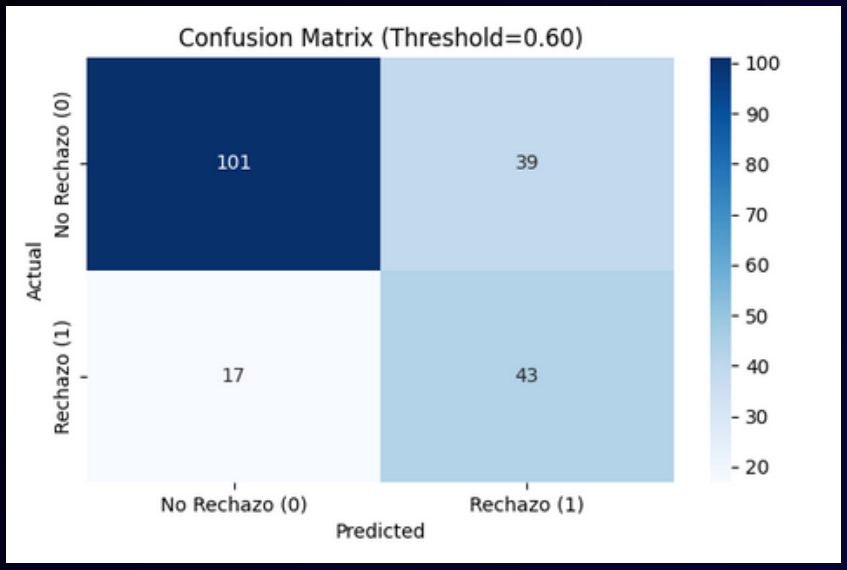

Modelo
Secuencial

Arquitectura de la red neuronal

Arquitectura detallada

capa	neuronas	Activacion
entrada	-	
capas ocultas	64	Relu
capa oculta	32	Relu
capa de salida	1	Sigmoid

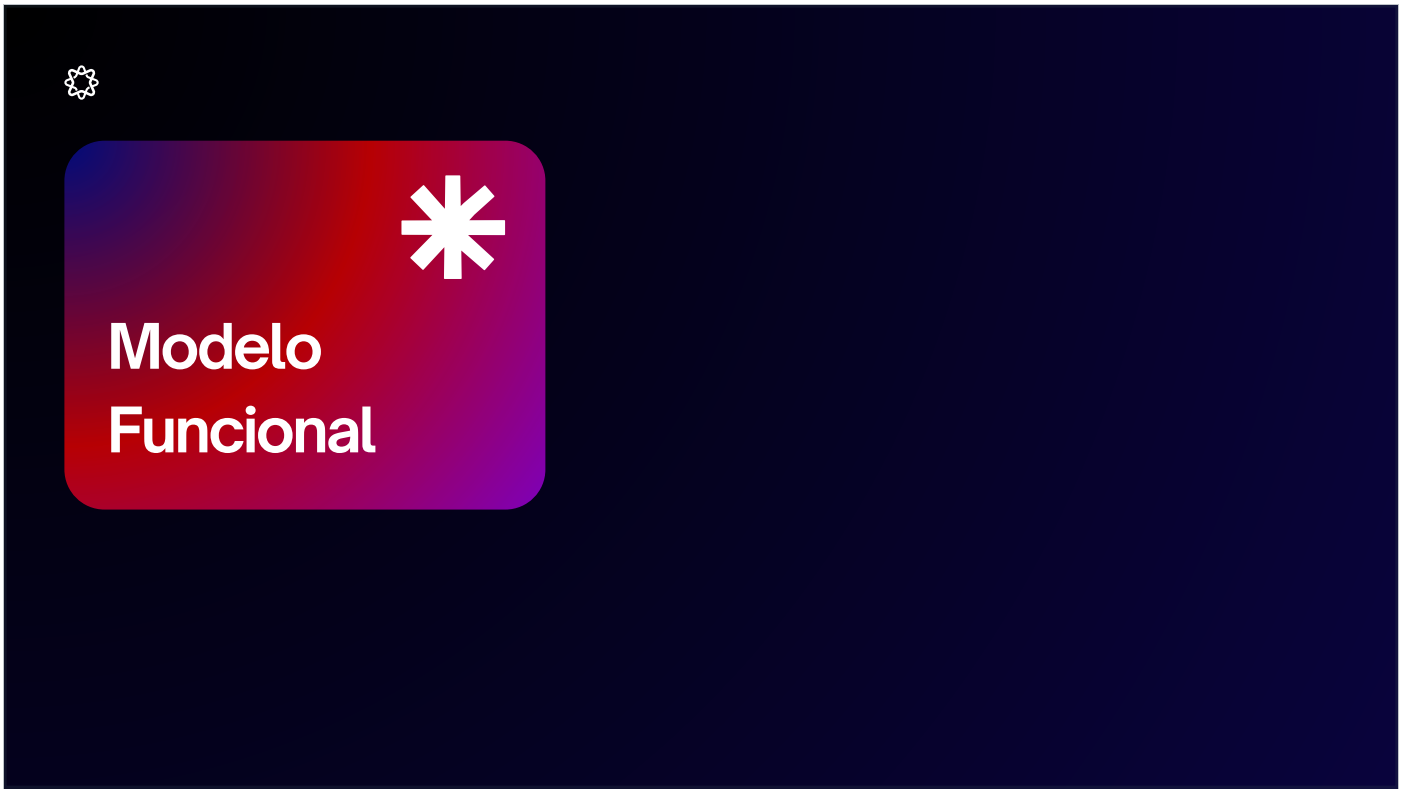

Accuracy
Train Acc
Val Acc
Accuracy
Epoch
Loss
Train Loss
Val Loss
Loss
Epoch

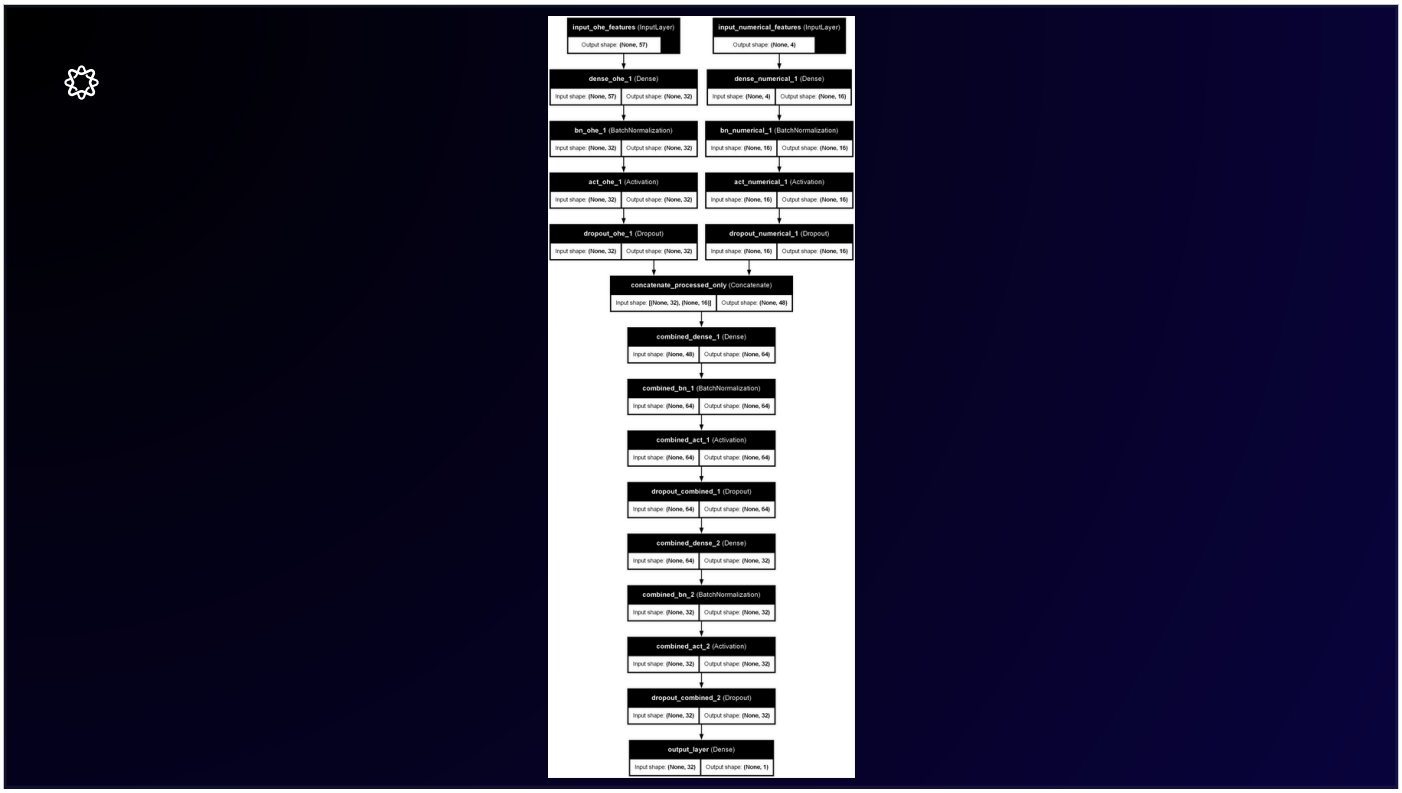

Confusion Matrix (Threshold=0.60)
Actual
No Rechazo (0)
Rechazo (1)
101
39
17
43
No Rechazo (0)
Rechazo (1)
Predicted
100
90
80
70
60
50
40
30
20

Classification Report (Threshold=0.60)

	precision	recall	f1-score	support
No Rechazo (0)	0.856	0.721	0.783	140.0
Rechazo (1)	0.524	0.717	0.606	60.0
accuracy	nan	nan	0.72	200.0
macro avg	0.69	0.719	0.694	200.0
weighted avg	0.756	0.72	0.73	200.0

vemos que el accuracy es de 0.72, pero el recall para rechazo (1) es relativamente bajo


Modelo
Funcional

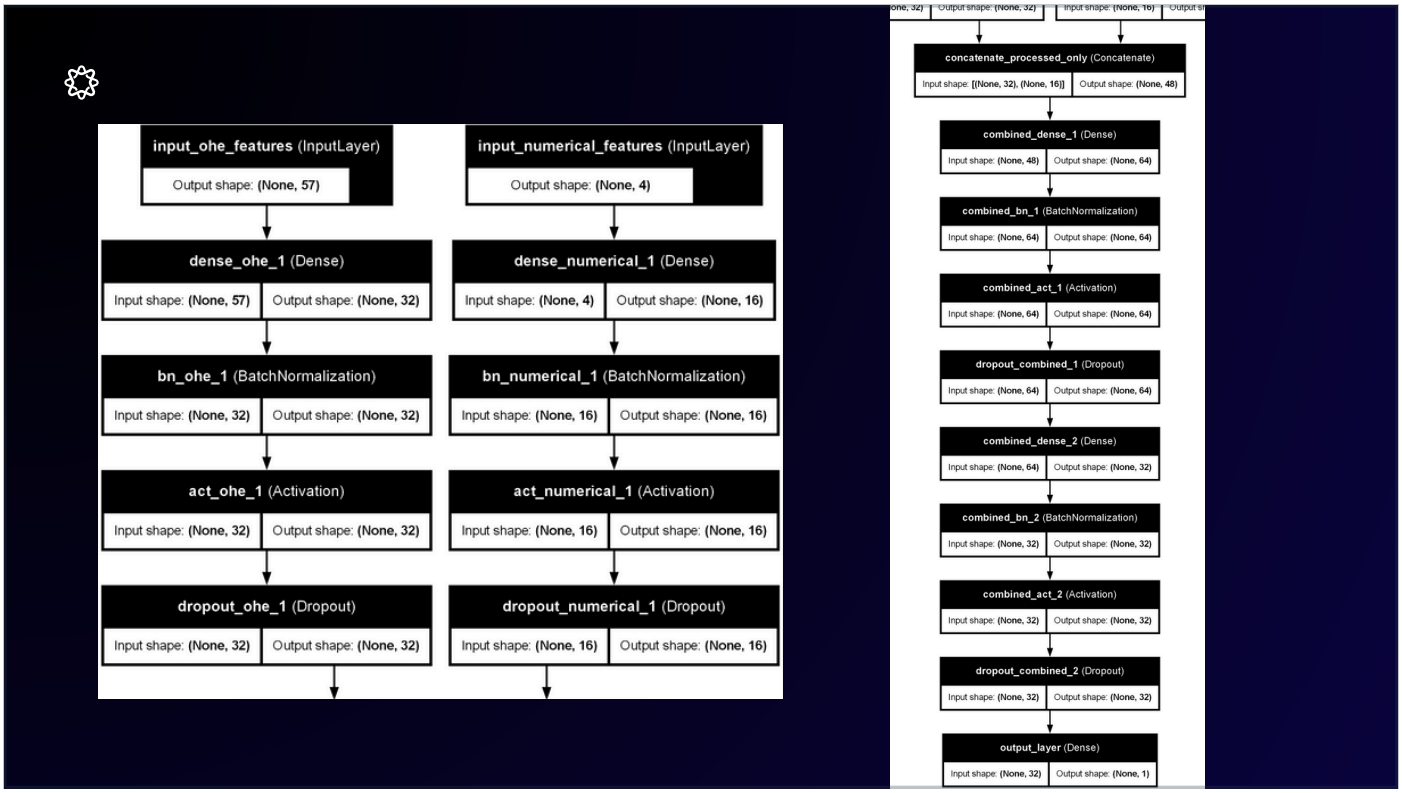


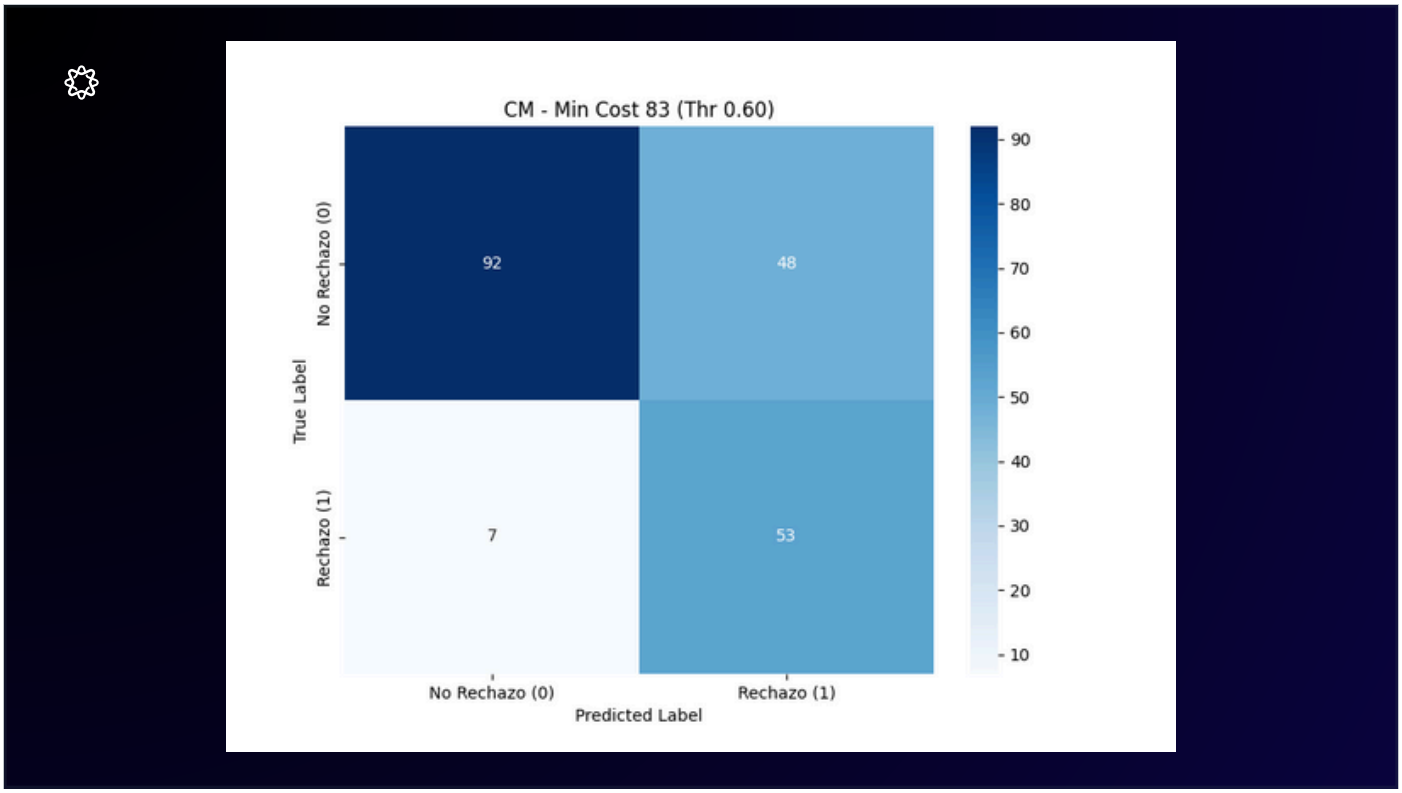
Lo que hicimos fue dividir los inputs entre variables numericas y las categoricas. Como las categoricas tienen one hot encoding, vemos que son 57 variables del one hot contra 4 variables numericas.

Cada input layer despues le hacemos bacth normalization, activacion y dropout. Una vez que tenemos eso, concatenamos ambas.

Una vez que tenemos la capa concatenada, tenemos dos capas ocultas. El circuito es capa oculta --> batch normalization --> activacion --> dropout para ambas capas.

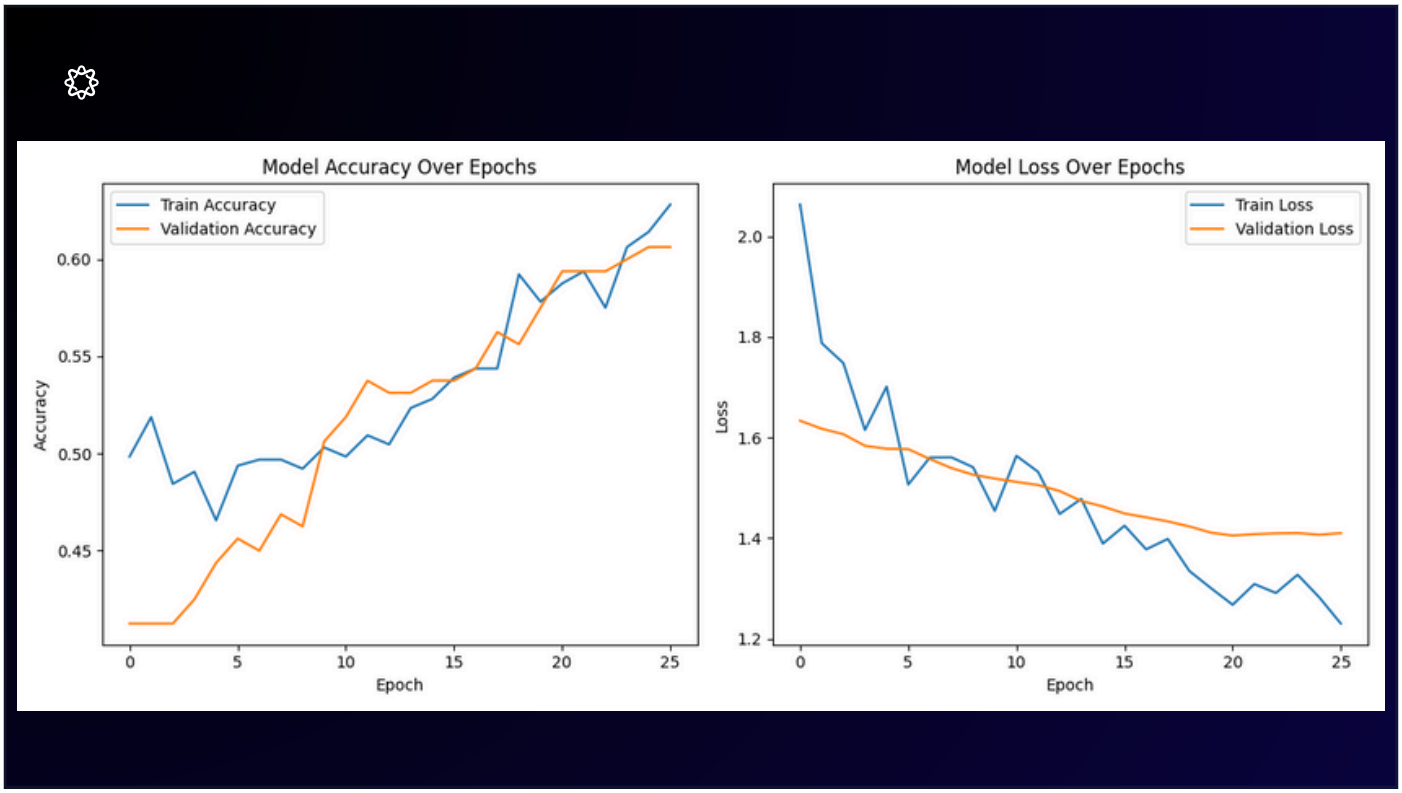
El modelo termina con una capa sigmoid que predice la probabilidad de rechazo = 1


input_ohe_features (InputLayer)
Output shape: (None, 57)
input_numerical_features (InputLayer)
Output shape: (None, 4)
dense_ohe_1 (Dense)
Input shape: (None, 57)
Output shape: (None, 32)
dense_numerical_1 (Dense)
Input shape: (None, 4)
Output shape: (None, 16)
bn_ohe_1 (BatchNormalization)
Input shape: (None, 32)
Output shape: (None, 32)
bn_numerical_1 (BatchNormalization)
Input shape: (None, 16)
Output shape: (None, 16)
act_ohe_1 (Activation)
Input shape: (None, 32)
Output shape: (None, 32)
act_numerical_1 (Activation)
Input shape: (None, 16)
Output shape: (None, 16)
dropout_ohe_1 (Dropout)
Input shape: (None, 32)
Output shape: (None, 32)
dropout_numerical_1 (Dropout)
Input shape: (None, 16)
Output shape: (None, 16)
concatenate_processed_only (Concatenate)
Input shape: [(None, 32), (None, 16)]
Output shape: (None, 48)
combined_dense_1 (Dense)
Input shape: (None, 48)
Output shape: (None, 64)
combined_bn_1 (BatchNormalization)
Input shape: (None, 64)
Output shape: (None, 64)
combined_act_1 (Activation)
Input shape: (None, 64)
Output shape: (None, 64)
dropout_combined_1 (Dropout)
Input shape: (None, 64)
Output shape: (None, 64)
combined_dense_2 (Dense)
Input shape: (None, 64)
Output shape: (None, 32)
combined_bn_2 (BatchNormalization)
Input shape: (None, 32)
Output shape: (None, 32)
combined_act_2 (Activation)
Input shape: (None, 32)
Output shape: (None, 32)
dropout_combined_2 (Dropout)
Input shape: (None, 32)
Output shape: (None, 32)
output_layer (Dense)
Input shape: (None, 32)
Output shape: (None, 1)



Vemos que hay muy pocos falsos negativos, y que hay una cantidad significativa de verdaderos positivos y de verdaderos negativos.

El modelo no va a buscar hacer un rechazo de todo porque no darle un prestamo a alguien que se lo merece tiene un costo.

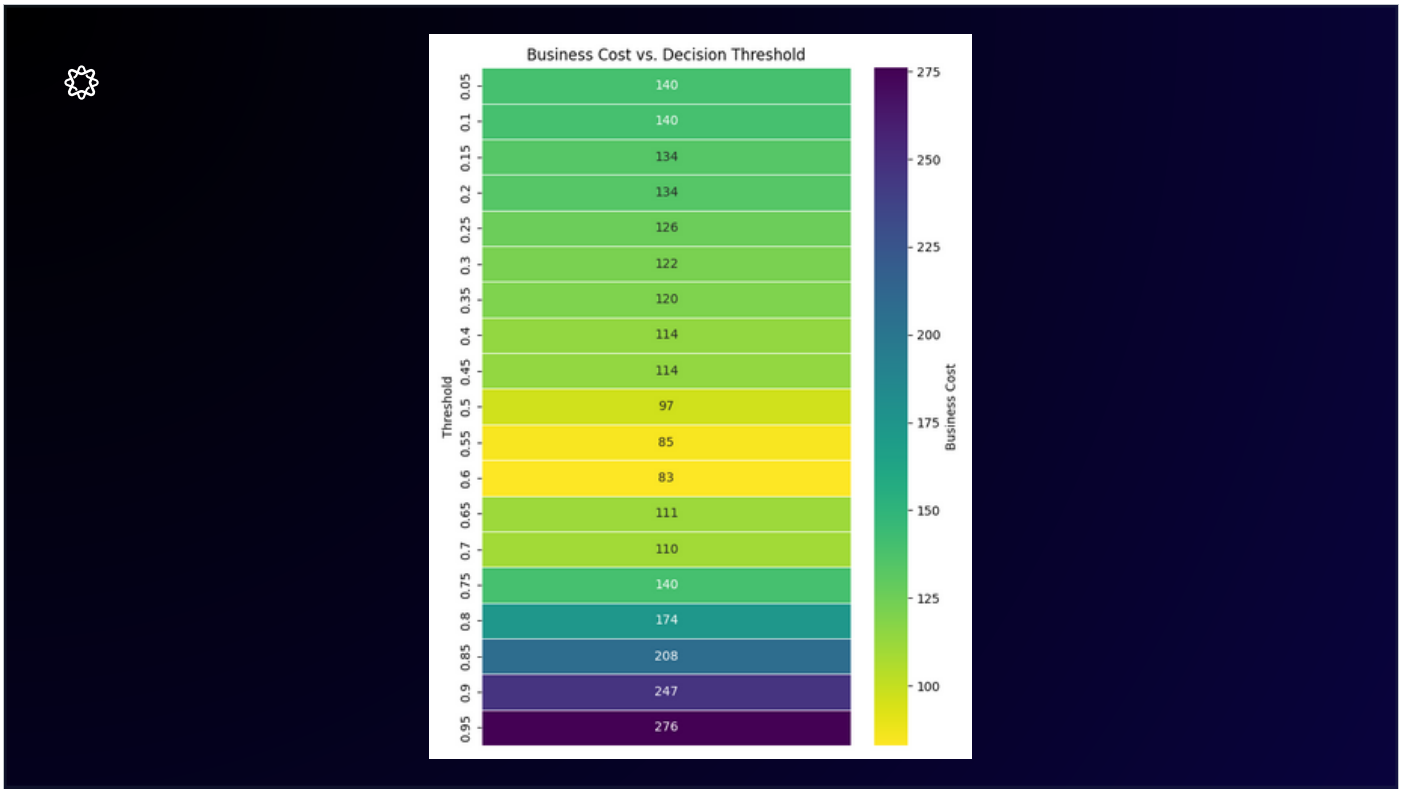

Model Accuracy Over Epochs
Train Accuracy
Validation Accuracy
Accuracy
Epoch
Model Loss Over Epochs
Train Loss
Validation Loss
Loss
Epoch

Best CR NN (Min Cost 83) - Thr 0.60

	precision	recall	f1-score	support
No Rechazo (0)	0.929	0.657	0.770	140.0
Rechazo (1)	0.525	0.883	0.658	60.0
macro avg	0.727	0.770	0.714	200.0
weighted avg	0.808	0.725	0.736	200.0
accuracy	nan	nan	0.725	200.0
macro avg	0.727	0.770	0.714	200.0
weighted avg	0.808	0.725	0.736	200.0

vemos que el recall para rechazo es 0.883, mientras que el accuracy es de 0.725.

Nuestro foco no fue priorizar ni recall de 1 o accuracy, sino reducir la funcion de loss



en el codigo iteramos el threshold para decidir si consideramos rechazo o no. Elegimos el valor que minimiza la funcion de loss. En este caso fue 0.6

Aca vemos los valores para todos los valores del threshold.

Como vemos, para 0.6 el valor de recall para rechazo (1) no es el mas alto, para 0.6 es 0.88 pero vemos que si bajasemos el threshold habria mas recall para rechazo (1), pero empeora la funcion de loss

Lo que vemos es que el modelo funcional da mejores metricas, por lo que hacer este modelo fue mejor.

El modelo funcional mejoro el recall de 1 (de 0.717 a 0.883), pero no mejoro el accuracy.

Si bien esto es algo positivo, lo que nos interesaba era minimizar la funcion de perdida, llevandola de 124 en el modelo secuencial a 83 en el modelo funcional

Gracias

Martín Quijano - Martina Coletto