75.06 / 95.58

Organización de datos

Reporte TP N°1: Reservas de Hotel

Clasificación - Entrenamiento y Predicción

Checkpoint N°4: Redes Neuronales

Alen Davies Leccese: 107084
Luca Mauricio Lazcano: 107044

Mayo 2023

1 Introducción

Para la última etapa de este trabajo práctico, llegamos a uno de los conceptos más emocionantes de los últimos años, en el ámbito de la informática y la ciencia de datos. Continuaremos realizando entrenamientos y predicciones, esta vez con redes neuronales.

Los datasets a utilizar serán los mismos que en la entrega previa. También se realizan todas las importaciones de librerías y herramientas a utilizar.

Se normalizaron los datos con el `StandardScaler` y se splitearon en conjuntos de entrenamiento y prueba en una proporción 70/30, como en todas las entregas anteriores.

2 Diseño de modelos

Para encontrar el modelo que mejor se adapte a nuestras necesidades, creamos y probamos distintos modelos, variando algunos hiperparámetros y la arquitectura del modelo. Para la

creación de los modelos, variamos el hiperparámetro de `epochs`, los optimizadores y el `learning rate`.

Realizar un Search de hiperparámetros resultó demasiado pesado computacionalmente por lo que no se logró obtener resultados en un tiempo razonable.

Para todos los modelos, la arquitectura elegida consiste de 3 capas de neuronas: la de entrada, con 64 neuronas; una capa oculta de 32 neuronas, ambas con función de activación relu, y para la salida una única neurona con función de activación sigmoidea.

Como función de regularización utilizamos `EarlyStopping` para todos los modelos. Esto permite evitar el overfitting y dejar al modelo entrenado en un punto donde se minimiza el `loss`, configurado el parámetro `monitor='val_loss'`.

Como función de minimización de loss, se utilizó `BinaryCrossentropy` para todos los modelos.

El resumen de la arquitectura de los modelos es:

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_27 (Dense)            (None, 64)                10944

 dense_28 (Dense)            (None, 32)                2080

 dense_29 (Dense)            (None, 1)                 33

=================================================================
Total params: 13,057
Trainable params: 13,057
Non-trainable params: 0
_________________________________________________________________
```

2.1 Modelo 1

Para este modelo se utilizó el optimizador `Adam`, con un `learning rate = 0.001` y 256 `epochs`.

El `F1 score` resultante para este modelo es: **0,8495**.

La matriz de confusión resultante es:

En el notebook está graficado el `loss` en el entrenamiento y el test, vs cantidad de `epochs`.

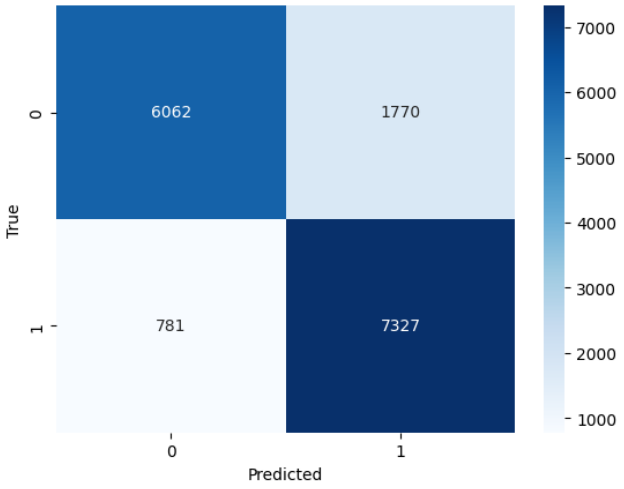

Figure 1: Matriz de confusión modelo 1.

2.2 Modelo 2

Para este modelo se utilizó el optimizador `SGD` (Stochastic Gradient Descent), con un `learning rate = 0.01` y 100 `epochs`.

El `F1 score` resultante para este modelo es: **0,8437**.

La matriz de confusión resultante es:

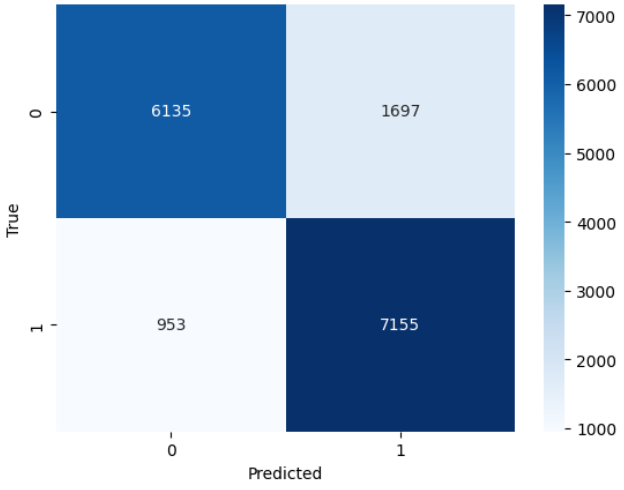

Figure 2: Matriz de confusión modelo 2.

En el notebook está graficado el `loss` en el entrenamiento y el test, vs cantidad de `epochs`.

2.3 Modelo 3

Para este modelo se utilizó el optimizador `RMSprop` (Root Mean Square Propagation), con un `learning rate = 0.001` y 50 `epochs`.

El `F1 score` resultante para este modelo es: **0,8516**.

La matriz de confusión resultante es:

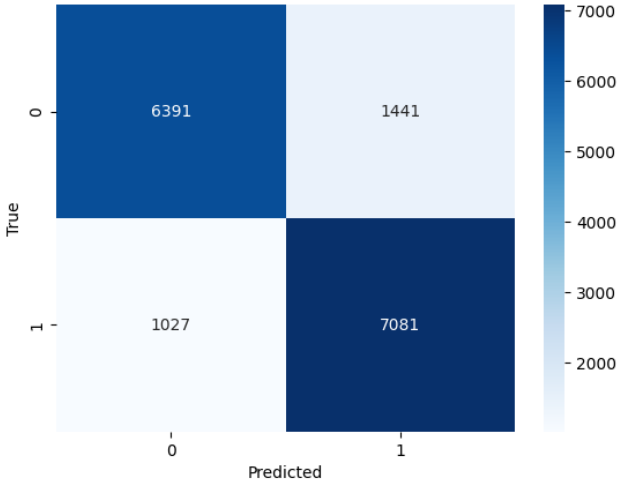

Figure 3: Matriz de confusión modelo 3.

En el notebook está graficado el `loss` en el entrenamiento y el test, vs cantidad de `epochs`.

3 Entrega y conclusión

Se eligió para la entrega el mejor modelo obtenido, es decir, el tercero. Para realizar la predicción binaria, se tomó como "1" (cancelado) los valores mayores a 0,5 y "0" los menores.

Este valor es cercano al recomendado de 0,4, y encontramos que el valor elegido proporciona mejores resultados.

Se guardó el archivo `.pickle` de este último modelo, y se realizó la entrega por Kaggle.

Se concluye que los modelos de redes neuronales proporcionan muy buenos resultados, sin la necesidad de explorar tantos parámetros como con otros modelos. De tener mayor capacidad de cómputo, se podría realizar una exploración más exhaustiva de hiperparámetros para encontrar el modelo perfecto.