Taller 1

Luis Santiago Jaramillo Espinosa, Parra Martínez Julio Alberto

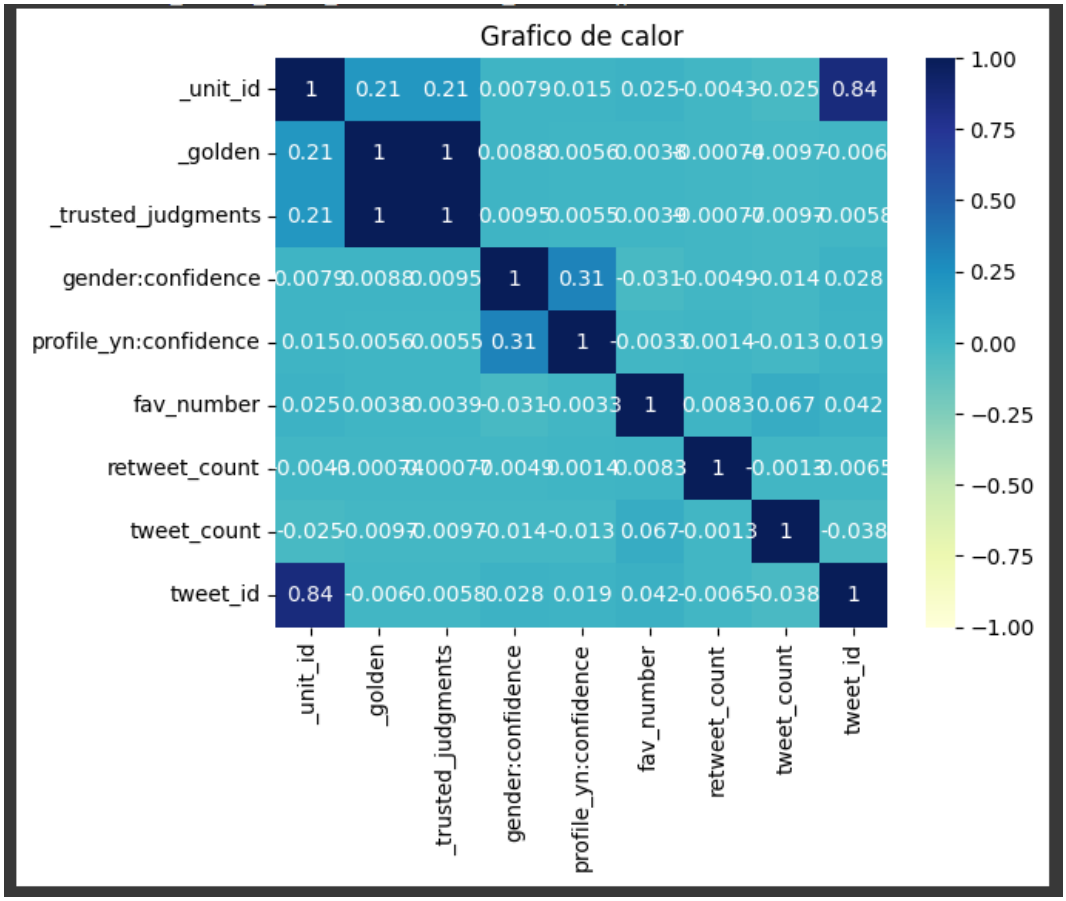
Facultad de ingeniería, Pontificia Universidad Javeriana de Bogotá

1. Comprensión del dataset

1.1 ¿Qué información presenta el dataset?

El dataset consta de 16 columnas y 20.050 datos. Algunas de las variables son categóricas por lo que más adelante serán tratadas. Así también se también se tienen datos de gran aporte como lo son **description** que pueden ser tratadas con NLP para de este modo poder ser mas precisos en la predicción de **gender**.

Se presenta el grafico de correlaciones.



2. Limpieza de los datos

Dado que el dataset tiene alrededor de aproximadamente 14.000 campos vacíos, no podemos eliminar todos aquellos campos ya que afectaría la precisión del modelo. Se opta por borrar algunos campos vacíos de algunas columnas, así como de eliminar columnas como lo son:

'_unit_id','tweet_id', 'user_timezone', 'tweet_coord', 'description', 'profileimage', 'text','link_color', 'tweet_location', 'name', '_last_judgment_at', 'created','tweet_created'. Así también se revisa el campo gender y se identifica que tiene 4 posibles valores: male, female, Brand y unknown. Se eliminan los Brand y los Unknowns ya que no aportan valor a la predicción de genero.

Finalmente de 26 columnas pasamos a tener solo 16: ['_unit_state', '_trusted_judgments', 'gender', 'gender:confidence', 'profile_yn', 'profile_yn:confidence', 'fav_number', 'gender_gold', 'profile_yn_gold', 'retweet_count', 'sidebar_color', 'tweet_count']

3. Variables Categoricas

Posteriormente se transforman male y female, donde male equivale a 1 y female a 0. Así también en las columnas 'profile_yn' y 'profile_yn_gold' se reemplaza Yes por 1. A la columna de color 'sidebar_color' se le asignan también valores numéricos.

4. Construcción del dataset

4.1 ¿Qué diferencia hay en usar un conjunto de validación?

Muchos modelos son propensos a un problema llamado sobreajuste que es cuando el modelo se ajusta tan bien a los datos de entrenamiento que no se generaliza bien a los nuevos datos. El signo revelador del sobreajuste es un modelo que funciona extremadamente bien con los datos de entrenamiento y su rendimiento es inferior con nuevos datos. La forma de tener esto en cuenta es dividir el conjunto de datos en varios conjuntos: un conjunto de entrenamiento para entrenar el modelo, un conjunto de validación para comparar el rendimiento de diferentes modelos y un conjunto de prueba final para comprobar cómo funcionará el modelo en el mundo real. [1]

4.2 ¿Mejora los resultados en la construcción del modelo usar un conjunto de validación?

No mejora los resultados, pero nos permite hacer ajustes al modelo como la tasa de aprendizaje o el número de iteraciones de esta forma optimizando su rendimiento. [2]

4.3 ¿Qué información provee el uso de un conjunto de validación?

El conjunto de validación afecta a un modelo, pero sólo indirectamente por lo que no provee información.

5. Elaboración del Modelo

5.1 Perceptrón.

```
1  from sklearn.linear_model import Perceptron
2  from sklearn.metrics import confusion_matrix, accuracy_score,
3  precision_score, recall_score, f1_score
4  from sklearn.metrics import confusion_matrix,
5  ConfusionMatrixDisplay
6  import matplotlib.pyplot as plt
7
8  #Distribucion de data recalcular,
9  #cambiar manejod e data comparaciion
10
11 # Entrenar el perceptrón
```

```
perceptron = Perceptron(max_iter=1000, eta0=0.01)
perceptron.fit(X, y)

# Predecir las etiquetas
y_pred_1 = perceptron.predict(X)

# Calcular la matriz de confusión
matriz_confusion_1 = confusion_matrix(y, y_pred_1)

# Calcular las métricas de evaluación
precision_1 = precision_score(y, y_pred_1)
recall_1 = recall_score(y, y_pred_1)
f1_1 = f1_score(y, y_pred_1)
```

5.2 Red Neuronal con una capa oculta con un numero de neurona igual al número de entradas.

```
from sklearn.neural_network import MLPClassifier

# Entrenar la red neuronal, activation default='relu'
mlp = MLPClassifier(hidden_layer_sizes=(X.shape[1],),
max_iter=1000, solver='adam')
mlp.fit(X, y)

# Predecir las etiquetas
y_pred_2 = mlp.predict(X)

# Calcular la matriz de confusión
matriz_confusion_2 = confusion_matrix(y, y_pred_2)

# Calcular las métricas de evaluación
precision_2 = precision_score(y, y_pred_2)
recall_2 = recall_score(y, y_pred_2)
f1_2 = f1_score(y, y_pred_2)
```

5.3 Red Neuronal con dos capas ocultas, la primera con las mitades de las entradas y la segunda con la misma cantidad de la capa oculta, la tercera capa será la de salida.

```

```

```
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split

from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense

# Dividir los datos en entrenamiento y prueba
X_train, X_test, y_train, y_test_3 = train_test_split(X,
y,train_size=0.6,test_size=0.4, random_state=42)

# Definir la arquitectura de la red neuronal
model = Sequential()

# Primera capa oculta
model.add(Dense(X_train.shape[1] // 2, activation='relu',
input_shape=(X_train.shape[1],)))

# Segunda capa oculta
model.add(Dense(X_train.shape[1] // 2, activation='relu'))

# Capa de salida
model.add(Dense(1, activation='sigmoid'))

# Compilar el modelo
model.compile(optimizer='adam', loss='binary_crossentropy',
metrics=['accuracy'])

# Entrenar el modelo
model.fit(X_train, y_train, epochs=100)

# Predecir las etiquetas
y_pred_3 = model.predict(X_test)

# Calcular la matriz de confusión
matriz_confusion_3 = confusion_matrix(y_test_3, y_pred_3)
```

```
# Calcular las métricas de evaluación
precision_3 = precision_score(y_test_3, y_pred_3)
recall_3 = recall_score(y_test_3, y_pred_3)
f1_3 = f1_score(y_test_3, y_pred_3)
```

6. Análisis de Resultados

Se obtuvo una mejora en precisión tras aplicar regularizaciones L1 incrementando su valor en 0.1

7. Ajustes

8. Referencias

[1] "Cómo preparar un conjunto de datos para machine learning y análisis", en *datos.gob.es, 2023*. Consultado: el 18 de febrero de 2023. [En línea]. Disponible en: https://datos.gob.es/es/blog/como-preparar-un-conjunto-de-datos-para-machine-learning-y-analisis

[2] Shah Tarang, "About Train, Validation and Test Sets in Machine Learning", en Medium 2017. Consultado: el 18 de febrero de 2023. [En línea]. Disponible en: https://towardsdatascience.com/train-validation-and-test-sets-72cb40cba9e7