Tecnológico de Monterrey

Evidencia 1. Reporte de AUC obtenido

María Fernanda Ramírez Barragán	A01750879	ITC
Melissa Garduño Ruiz	A01748945	ITC

Fecha: 24/04/23

Instituto Tecnológico y de Estudios Superiores de Monterrey

Desarrollo de aplicaciones avanzadas

```
from sklearn.metrics import roc_auc_score
from check_plagarism import read_documents, suspicious_files,
original_files

# Manually define the labels names
labels_name = ['FID-01', 'FID-02', 'FID-03', 'FID-04',
'FID-05', 'FID-06', 'FID-07', 'FID-08', 'FID-09', 'FID-10',
'FID-11', 'FID-12', 'FID-13', 'FID-14','FID-15']
# Manually define the true labels (0 for original, 1 for
suspicious)
true_labels = [1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1]

# Function that checks if similarity scores is > than 0.30, if
so, append the plagarisim file name
def check_files():
    # Get the similarity matrix between all suspicious and
original documents
    similarities = read_documents()
    plag_docs_pred = []
    # Output the documents with similarity scores above 0.30
    for i, plagiarized_file in enumerate(suspicious_files):
        for j, original_file in enumerate(original_files):
            if similarities[i, j] > 0.30:
                # If so, append plagiarized_file name to
plag_docs_pred
                if plagiarized_file.split('.')[0] not in
plag_docs_pred:
                    # If plagiarized_file name is already in
plag_docs_pred, it doesn't append it twice

plag_docs_pred.append(plagiarized_file.split('.')[0])
    return plag_docs_pred

#Function that gets the predictions given plag_docs_pred of
"check_files()"
def get_predictions():
    predictions = []
    plag_docs_pred = check_files()
```

```
    # Check if the plag_docs_pred names are in labels_name
    for name in labels_name:
        # If so, append 1 to predictions empty list
        if name in plag_docs_pred:
            predictions.append(1)
        # If not, append 0 to predictions empty list
        else:
            predictions.append(0)
    # Returns the list with the predictions
    return predictions

predictions = get_predictions()
# Give AUC score
auc = roc_auc_score(true_labels, predictions)
print('-----------------------------------------')
print(f"El Area bajo la curva AUC es: {auc:.2f}")
print('-----------------------------------------')
```

El AUC (Area Under the Curve) es una medida de evaluación comúnmente utilizada en problemas de clasificación y es útil para evaluar la calidad de un modelo de clasificación.El valor del AUC oscila entre 0 y 1, donde un valor de 0,5 indica que el modelo es tan bueno como una clasificación aleatoria, mientras que un valor de 1 indica que el modelo es perfecto. En general, cuanto mayor sea el valor del AUC, mejor será la capacidad del modelo para clasificar correctamente los ejemplos en cada clase.
Para poder obtener el AUC de nuestro modelo, nos apoyamos en la función ‘roc_auc_score’ de la biblioteca de sklearn y dos funciones implementadas:

- check_files: Función que regresa una lista con los nombres (sin la extensión) de los archivos que el modelo evalúa como plagiados (similitud de coseno mayor a 0.3)
- get_predictions: Función que revisa si los nombres de la lista obtenida por ‘check_files’ se encuentran en la lista ‘labels_name’ (esta lista contiene todos los nombres de los archivos dados como sospechosos), de ser así, se agrega un 1 en la lista vacía ‘predictions’, de lo contrario, se agrega un 0.
- roc_auc_score: Función que devuelve el AUC dada la lista de true_labels (que se indica de manera manual ya que nosotros conocemos cuales archivos son plagio y cuales originales) y la lista obtenida predictions.

Resultados obtenidos:

Para la primera prueba del modelo, utilizamos 3 n-gramas (que anteriormente y gracias a las pruebas unitarias se había corroborado que era el valor que nos arrojaba los mejores resultados) obteniendo un valor de AUC = 1, esto quiere decir que las predicciones del modelo son exactamente iguales a las que reportamos como correctas:
Conocemos que los documento plagiados dentro de los documentos sospechosos brindados por los profesores son 'FID-01', 'FID-02', 'FID-05', 'FID-06', 'FID-15', por lo que, al pasar estos a una lista representada por 0 y 1 se mira de la siguiente manera:

```
true_labels = [1, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1]
```

Conociendo este dato, solo nos resta observar los resultados de las predicciones del modelo, obtenemos lo siguiente:

```
['FID-01', 'FID-02', 'FID-05', 'FID-06', 'FID-15']
```

Finalmente, convertimos esta lista que contiene los nombres de los archivos plagiados (según nuestro modelo) al mismo formato que contiene 'true labels'.
Como se puede observar, los documentos reportados por los profesores son exactamente los mismos que reporta nuestro modelo, es decir, que para esta primera entrega, y para los requisitos solicitados, el modelo tiene un 100% de precisión al usar 3 n-gramas.

```
----------------------------------------
El Area bajo la curva AUC es: 1.00
----------------------------------------
```

Sin embargo, al hacer la prueba con n-gramas = 2, obtenemos de nuestro modelo que los archivos plagiados son:

```
['FID-01', 'FID-02', 'FID-03', 'FID-04', 'FID-05', 'FID-06', 'FID-07', 'FID-08', 'FID-0
9', 'FID-10', 'FID-11', 'FID-12', 'FID-13', 'FID-14', 'FID-15']
```

obteniendo asi un AUC de:

```
----------------------------------------
El Area bajo la curva AUC es: 0.50
----------------------------------------
```

Que como anteriormente se mencionó, indica que el modelo es tan bueno como una clasificación aleatoria.