Taller 3 - Análisis de Sentimientos

Daniel Alejandro Chimbi León

Edna Valentina Henao Barrera

Daniel Felipe Sanchez Mogollón

Facultad de Ingeniería, Universidad El Bosque

Big Data Analytics

Fabian Camilo Peña Lozano

28 de noviembre de 2021

PARTE 1 - Para construir un mejor modelo se requieren más datos etiquetados

Para aumentar el número de datos etiquetados, se aumenta el dato numérico de la variable text-size el cual nos permite subir de 100 en 100. Seguido de cada imagen en el que se incrementa la variable, se puede observar los datos Recall y F1 en los que se puede detallar el aumento de los mismos.

Para 100

```
X_train, X_test, y_train, y_test = train_test_split(tweets_labeled_df['full_text'], tweets_labeled_df['sentiment'], test_size = 0.235,
```

```
X_train.shape
```

```
(324,)
```

```
pd.Series(y_train).value_counts(normalize = True)
```

```
0.0    0.635802
1.0    0.364198
Name: sentiment, dtype: float64
```

```
X_test.shape
```

```
(100,)
```

```
print('Precision:', precision_score(y_test, y_test_tfidf_predict))
print('Recall:', recall_score(y_test, y_test_tfidf_predict))
print('F1:', f1_score(y_test, y_test_tfidf_predict))
```

```
Precision: 0.5
Recall: 0.19444444444444445
F1: 0.28
```

Para 200

```
X_train, X_test, y_train, y_test = train_test_split(tweets_labeled_df['full_text'], tweets_labeled_df['sentiment'], test_size = 0.47
```

```
X_train.shape
```

```
(224,)
```

```
pd.Series(y_train).value_counts(normalize = True)
```

```
0.0    0.638393
1.0    0.361607
Name: sentiment, dtype: float64
```

```
X_test.shape
```

```
(200,)
```

```
pd.Series(y_test).value_counts(normalize = True)
```

```
0.0    0.635
1.0    0.365
Name: sentiment, dtype: float64
```

```
print('Precision:', precision_score(y_test, y_test_bow_predict))
print('Recall:', recall_score(y_test, y_test_bow_predict))
print('F1:', f1_score(y_test, y_test_bow_predict))
```

```
Precision: 0.41304347826086957
Recall: 0.2602739726027397
F1: 0.31932773109243695
```

Para 300

```
X_train, X_test, y_train, y_test = train_test_split(tweets_labeled_df['full_text'], tweets_labeled_df['sentiment'], test_size = 0.706
```

```
X_train.shape
```

```
(124,)
```

```
pd.Series(y_train).value_counts(normalize = True)
```

```
0.0    0.637097
1.0    0.362903
Name: sentiment, dtype: float64
```

```
X_test.shape
```

```
(300,)
```

```
pd.Series(y_test).value_counts(normalize = True)
```

```
0.0    0.636667
1.0    0.363333
Name: sentiment, dtype: float64
```

```
print('Precision:', precision_score(y_test, y_test_tfidf_predict))
print('Recall:', recall_score(y_test, y_test_tfidf_predict))
print('F1:', f1_score(y_test, y_test_tfidf_predict))
```

```
Precision: 0.4
Recall: 0.07339449541284404
F1: 0.12403100775193798
```

PARTE 2 - El entrenamiento se está realizando con datos mal etiquetados

BOW ANTES

Anteriormente la variable bow mostraba una matriz de confusión más pequeña y la precisión es más baja en comparación.

[51]:
```
y_train_bow_predict = logistic_model.predict(X_bow)
y_test_bow_predict = logistic_model.predict(bow.transform(X_test))
```

[52]:
```
confusion_matrix(y_train, y_train_bow_predict)
```

[52]:
```
array([[1838,    5],
       [  79, 1089]], dtype=int64)
```

[53]:
```
confusion_matrix(y_test, y_test_bow_predict)
```

[53]:
```
array([[1389,  453],
       [ 833,  336]], dtype=int64)
```

[54]:
```
print('Precision:', precision_score(y_test, y_test_bow_predict))
print('Recall:', recall_score(y_test, y_test_bow_predict))
print('F1:', f1_score(y_test, y_test_bow_predict))
```

```
Precision: 0.42585551330798477
Recall: 0.2874251497005988
F1: 0.3432073544433095
```

BOW DESPUÉS

Al finalizar, el modelo maneja una matriz de confusión más grande y una precisión mayor, mejorando el resultado de las métricas.

Training and evaluating a model using BOW

```
2]: logistic_model = LogisticRegression(random_state = 2)

3]: logistic_model.fit(X_bow, y_train)

3]: LogisticRegression(random_state=2)

4]: y_train_bow_predict = logistic_model.predict(X_bow)
    y_test_bow_predict = logistic_model.predict(bow.transform(

5]: confusion_matrix(y_train, y_train_bow_predict)

5]: array([[216,   0],
           [  0, 123]], dtype=int64)

6]: confusion_matrix(y_test, y_test_bow_predict)

6]: array([[44, 10],
           [23,  8]], dtype=int64)

7]: print('Precision:', precision_score(y_test, y_test_bow_pre
    print('Recall:', recall_score(y_test, y_test_bow_predict))
    print('F1:', f1_score(y_test, y_test_bow_predict))

    Precision: 0.4444444444444444
    Recall: 0.25806451612903225
    F1: 0.32653061224489793
```

TF-IDF ANTES

Anteriormente el modelo TF-IDF mostraba una matriz de confusión más pequeña y la precisión es más baja en comparación.

Training and evaluating a model using TF-IDF

```
logistic_model = LogisticRegression(random_state = 2)
```

```
logistic_model.fit(X_tfidf, y_train)
```

```
LogisticRegression(random_state=2)
```

```
y_train_tfidf_predict = logistic_model.predict(X_tfidf)
y_test_tfidf_predict = logistic_model.predict(bow.transform(X_test))
```

```
confusion_matrix(y_train, y_train_tfidf_predict)
```

```
array([[1839,    4],
       [ 643,  525]], dtype=int64)
```

```
confusion_matrix(y_test, y_test_tfidf_predict)
```

```
array([[1566,  276],
       [ 952,  217]], dtype=int64)
```

```
print('Precision:', precision_score(y_test, y_test_tfidf_predict))
print('Recall:', recall_score(y_test, y_test_tfidf_predict))
print('F1:', f1_score(y_test, y_test_tfidf_predict))
```

```
Precision: 0.44016227180527384
Recall: 0.18562874251497005
F1: 0.2611311672683514
```

TF-IDF DESPUÉS

Al finalizar, el modelo maneja una matriz de confusión más grande y una precisión mayor, mejorando el resultado de las métricas.

Training and evaluating a model using TF-IDF

```
[28]: logistic_model = LogisticRegression(random_state = 2)

[29]: logistic_model.fit(X_tfidf, y_train)

[29]: LogisticRegression(random_state=2)

[30]: y_train_tfidf_predict = logistic_model.predict(X_tfidf)
      y_test_tfidf_predict = logistic_model.predict(bow.transform(X_test))

[31]: confusion_matrix(y_train, y_train_tfidf_predict)

[31]: array([[216,   0],
             [ 67,  56]], dtype=int64)

[32]: confusion_matrix(y_test, y_test_tfidf_predict)

[32]: array([[51,  3],
             [25,  6]], dtype=int64)

[33]: print('Precision:', precision_score(y_test, y_test_tfidf_predict))
      print('Recall:', recall_score(y_test, y_test_tfidf_predict))
      print('F1:', f1_score(y_test, y_test_tfidf_predict))

      Precision: 0.6666666666666666
      Recall: 0.1935483870967742
      F1: 0.30000000000000004
```

LINK DE YOUTUBE: https://youtu.be/53HglE4RNbY