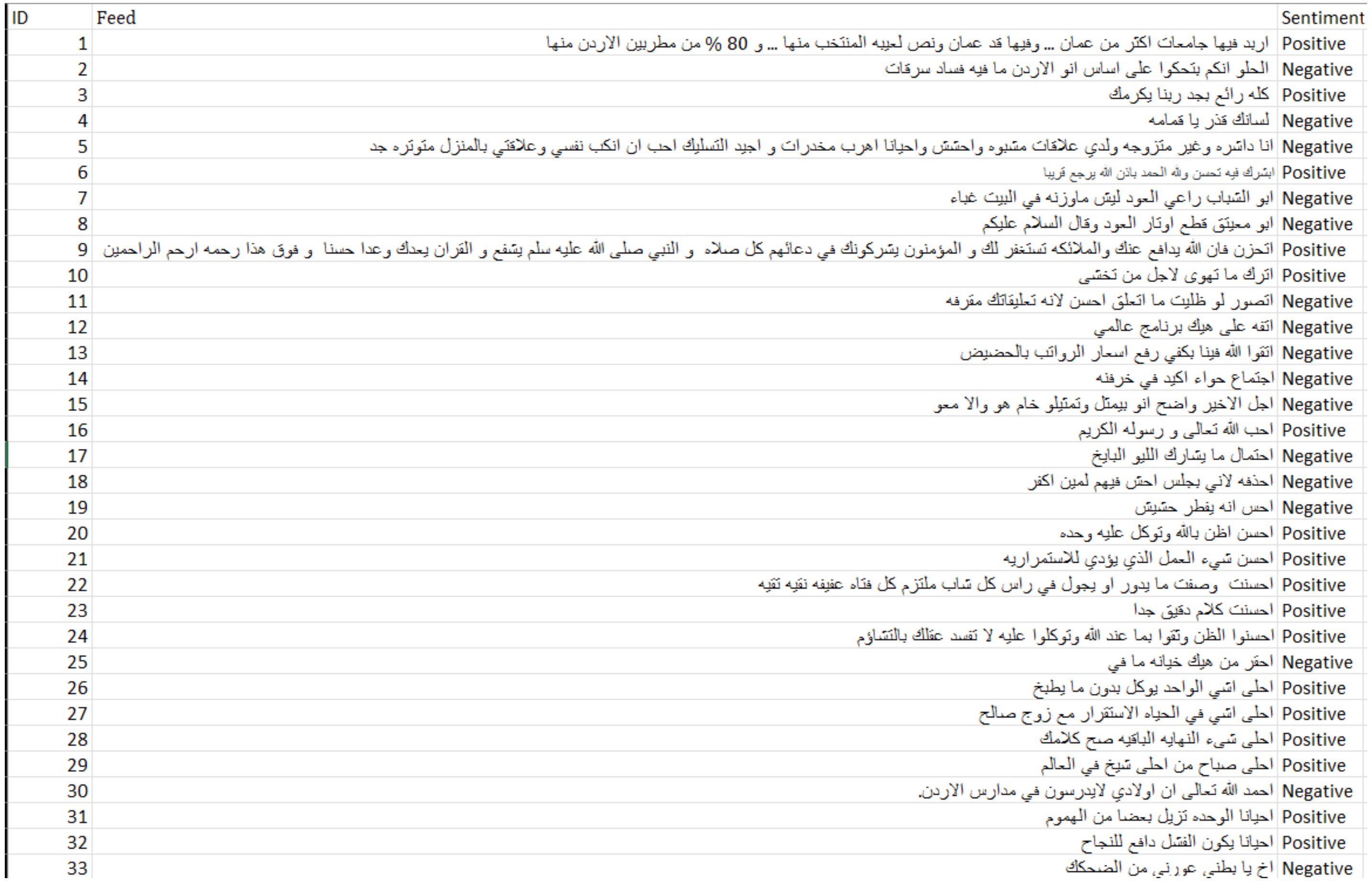
Sentiment Analysis (Arabic Dataset

Osama ALDARABSEH 190201955

Sentiment Analysis(Arabic Dataset)

- A Sentiment Analysis project with an Arabic dataset involves analysing and classifying text data written in the Arabic language based on the sentiment expressed. The goal is to determine whether the sentiment conveyed in the text is positive, negative

The Dataset



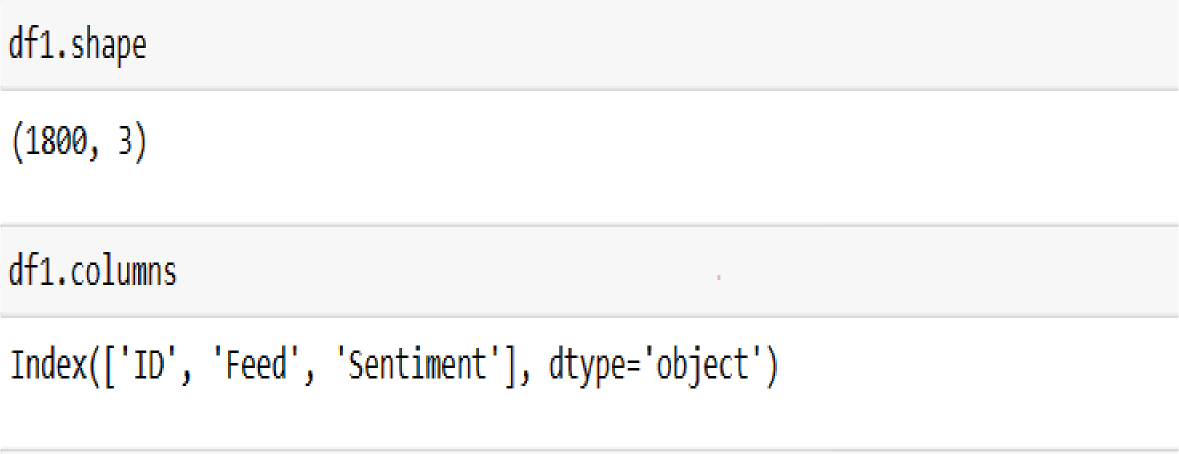
ID	Feed	Sentiment
1	اريد فيها جامعات اكثر من عمان ... وفيها قد عمان ونص لعيبه المنتخب منها ... و 80 % من مطربين الاردن منها	Positive
2	الحلو انكم بتحكوا على اساس انو الاردن ما فيه فساد سرقات	Negative
3	كله رائع بجد ربنا يكرمك	Positive
4	لسانك قذر يا قمامه	Negative
5	انا داشره وغير متزوجه ولدي علاقات مشبوه واحشش واحيانا اهرب مخدرات و اجيد التسليك احب ان انكب نفسي وعلاقتي بالمنزل متوتره جد	Negative
6	ابشرك فيه تحسن ولله الحمد باذن الله يرجع قريبا	Positive
7	ابو الشباب راعي العود ليش ماوزنه في البيت غباء	Negative
8	ابو معيتق قطع اوتار العود وقال السلام عليكم	Negative
9	اتحزن فان الله يدافع عنك والملائكه تستغفر لك و المؤمنون يشركونك في دعائهم كل صلاه و النبي صلى الله عليه سلم يشفع و القران يحدك وعدا حسنا و فوق هذا رحمه ارحم الراحمين	Positive
10	اترك ما تهوى لاجل من تخشى	Positive
11	اتصور لو ظليت ما اتعلق احسن لانه تعليقاتك مقرفه	Negative
12	اتفه على هيك برنامج عالمي	Negative
13	اتقوا الله فينا بكفي رفع اسعار الرواتب بالحضيض	Negative
14	اجتماع حواء اكيد في خرفنه	Negative
15	اجل الاخير واضح انو بيمثل وتمثيلو خام هو والا معو	Negative
16	احب الله تعالى و رسوله الكريم	Positive
17	احتمال ما يشارك اللبو البايخ	Negative
18	احذفه لاني بجلس احش فيهم لمين اكفر	Negative
19	احس انه يفطر حشيش	Negative
20	احسن اظن بالله وتوكل عليه وحده	Positive
21	احسن شيء العمل الذي يؤدي للاستمراريه	Positive
22	احسنت وصفت ما يدور او يجول في راس كل شاب ملتزم كل فتاه عفيفه نقيه تقيه	Positive
23	احسنت كلام دقيق جدا	Positive
24	احسنوا الظن وثقوا بما عند الله وتوكلوا عليه لا تفسد عقلك بالتشاؤم	Positive
25	احقر من هيك خيانه ما في	Negative
26	احلى اشي الواحد يوكل بدون ما يطبخ	Positive
27	احلى اشي في الحياه الاستقرار مع زوج صالح	Positive
28	احلى شيء النهايه الباقيه صح كلامك	Positive
29	احلى صباح من احلى شيخ في العالم	Positive
30	احمد الله تعالى ان اولادي لايدرسون في مدارس الاردن.	Negative
31	احيانا الوحده تزيل بعضا من الهموم	Positive
32	احيانا يكون الفشل دافع للنجاح	Positive
33	اخ يا بطني عورني من الضحكك	Negative

https://metatext.io/datasets/arabic-jordanian-general-tweets-(ajgt)

Used Algorithm

- Recurrent Neural Network (RNN) that is effective in modeling sequential data and has been successful in capturing context and long-term dependencies, which are important in sentiment analysis tasks.

Data Analysis



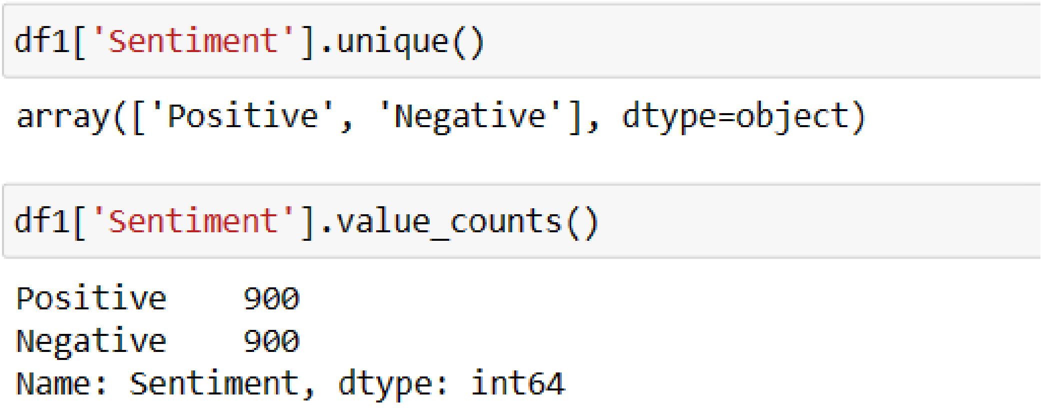
```
df1.shape
```

```
(1800, 3)
```

```
df1.columns
```

```
Index(['ID', 'Feed', 'Sentiment'], dtype='object')
```

Data Analysis



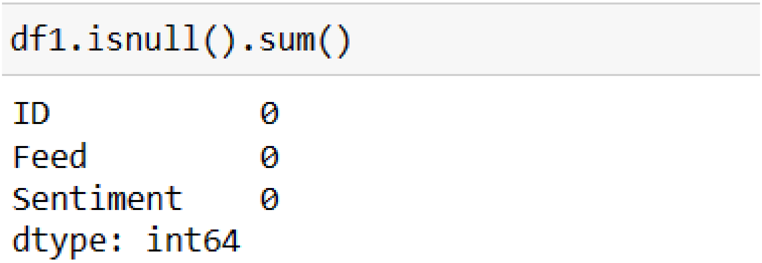
```
df1['Sentiment'].unique()
```

```
array(['Positive', 'Negative'], dtype=object)
```

```
df1['Sentiment'].value_counts()
```

```
Positive    900
Negative    900
Name: Sentiment, dtype: int64
```

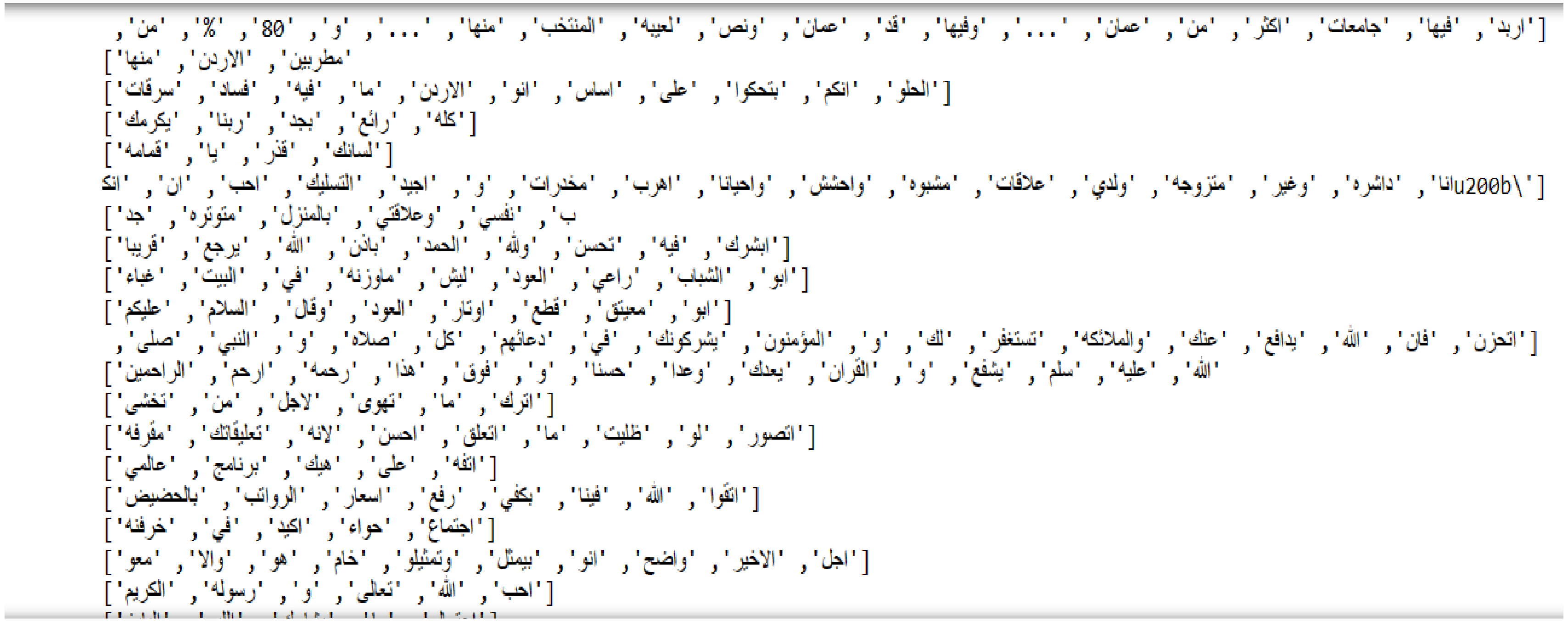
Data Analysis



```
df1.isnull().sum()
```

```
ID           0
Feed         0
Sentiment    0
dtype: int64
```

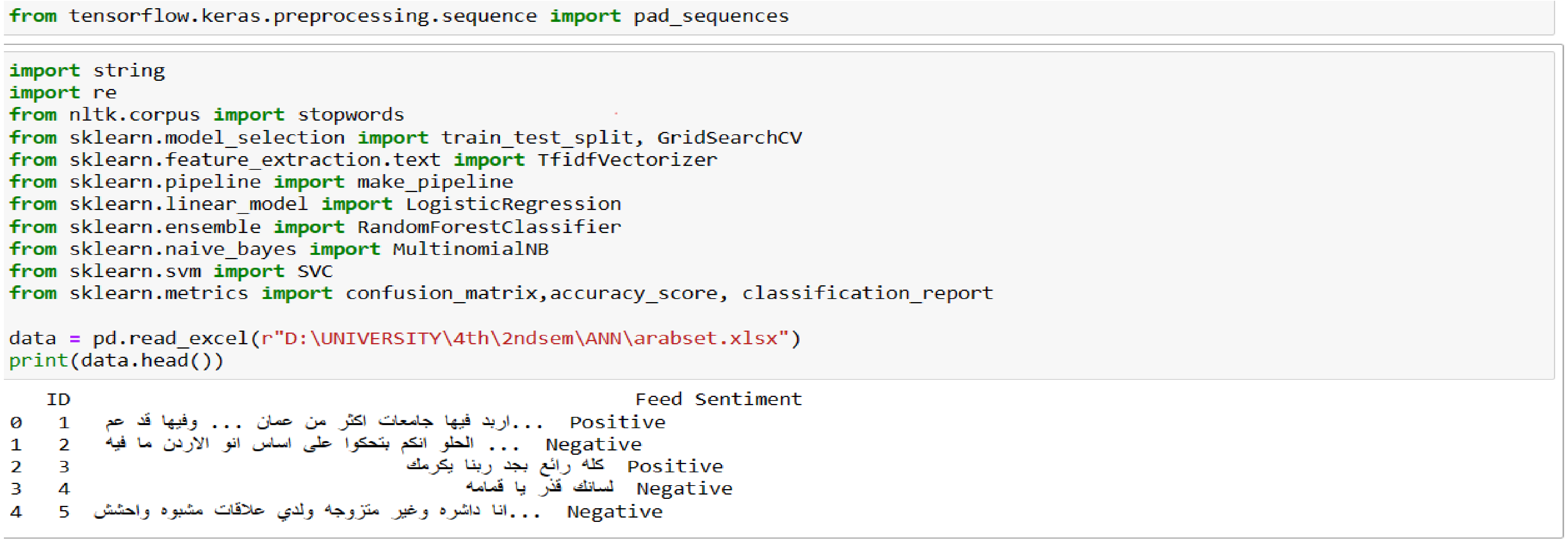
Dataset Tokenization



```
['اربد', 'فيها', 'جامعات', 'اكثر', 'من', 'عمان', '...', 'وفيها', 'قد', 'عمان', 'ونص', 'لعيبه', 'المنتخب', 'منها', '...', 'و', '80', '%', 'من',
'مطربين', 'الاردن', 'منها']
['الحلو', 'انكم', 'بتحكوا', 'على', 'اساس', 'انو', 'الاردن', 'ما', 'فيه', 'فساد', 'سرقات']
['كله', 'رائع', 'بجد', 'ربنا', 'يكرمك']
['لسانك', 'قذر', 'يا', 'قمامه']
['\u200bانا', 'داشره', 'وغير', 'متزوجه', 'ولدي', 'علاقات', 'مشبوه', 'واحشش', 'واحيانا', 'اهرب', 'مخدرات', 'و', 'اجيد', 'التسليك', 'احب', 'ان', 'انك
ب', 'نفسي', 'وعلاقتي', 'بالمنزل', 'متوتره', 'جد']
['ابشرك', 'فيه', 'تحسن', 'ولله', 'الحمد', 'باذن', 'الله', 'يرجع', 'قريبا']
['ابو', 'الشباب', 'راعي', 'العود', 'ليش', 'ماوزنه', 'في', 'البيت', 'غباء']
['ابو', 'معيتق', 'قطع', 'اوتار', 'العود', 'وقال', 'السلام', 'عليكم']
['اتحزن', 'فان', 'الله', 'يدافع', 'عنك', 'والملائكه', 'تستغفر', 'لك', 'و', 'المؤمنون', 'يشركونك', 'في', 'دعائهم', 'كل', 'صلاه', 'و', 'النبي', 'صلى',
'الله', 'عليه', 'سلم', 'يشفع', 'و', 'القران', 'يعدك', 'وعدا', 'حسنا', 'و', 'فوق', 'هذا', 'رحمه', 'ارحم', 'الراحمين']
['اترك', 'ما', 'تهوى', 'لاجل', 'من', 'تخشى']
['اتصور', 'لو', 'ظليت', 'ما', 'اتعلق', 'احسن', 'لانه', 'تعليقاتك', 'مقرفه']
['اتفه', 'على', 'هيك', 'برنامج', 'عالمي']
['اتقوا', 'الله', 'فينا', 'بكفي', 'رفع', 'اسعار', 'الرواتب', 'بالحضيض']
['اجتماع', 'حواء', 'اكيد', 'في', 'خرفنه']
['اجل', 'الاخير', 'واضح', 'انو', 'بيمثل', 'وتمثيلو', 'خام', 'هو', 'والا', 'معو']
['احب', 'الله', 'تعالى', 'و', 'رسوله', 'الكريم']
```

Pre-processing

We utilized a library for the pre-processing step in our project, which greatly facilitated our data preparation process.



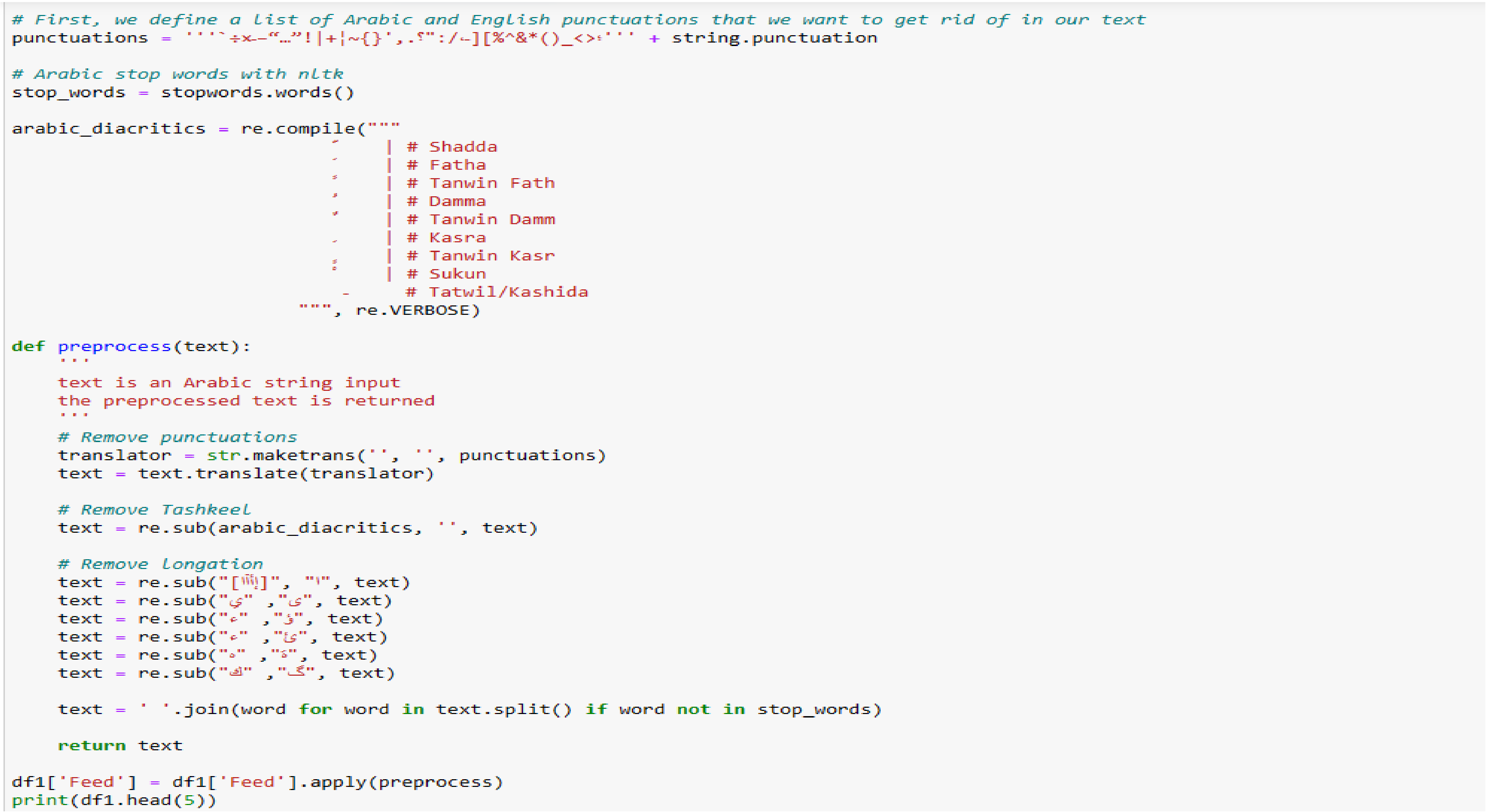
```
from tensorflow.keras.preprocessing.sequence import pad_sequences
```

```
import string
import re
from nltk.corpus import stopwords
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.pipeline import make_pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix,accuracy_score, classification_report

data = pd.read_excel(r"D:\UNIVERSITY\4th\2ndsem\ANN\arabset.xlsx")
print(data.head())
```

```
   ID                                               Feed Sentiment
0   1  اربد فيها جامعات اكثر من عمان ... وفيها قد عم...  Positive
1   2  الحلو انكم بتحكوا على اساس انو الاردن ما فيه ...  Negative
2   3                            كله رائع بجد ربنا يكرمك  Positive
3   4                                لسانك قذر يا قمامه  Negative
4   5  انا داشره وغير متزوجه ولدي علاقات مشبوه واحشش...  Negative
```

Cleaning The Text

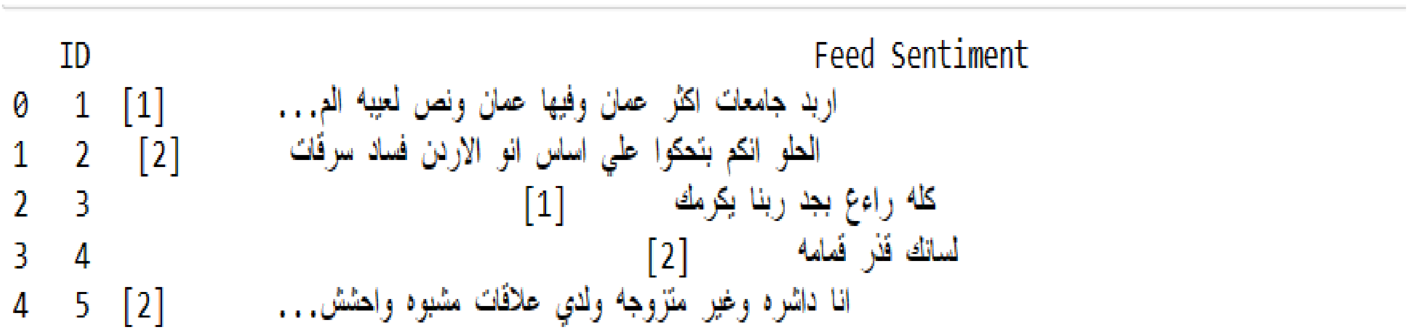
- with this code for cleaning texts in Arabic. The steps basically involve removing punctuation, Arabic diacritics (short vowels and other harakahs), elongation, and stopwords (which is available in NLTK corpus).



```
# First, we define a list of Arabic and English punctuations that we want to get rid of in our text
punctuations = '''`÷×؛<>_()*&^%][ـ،/:"؟.,'{}~¦+|!”…“–ـ''' + string.punctuation

# Arabic stop words with nltk
stop_words = stopwords.words()

arabic_diacritics = re.compile("""
                             ّ    | # Shadda
                             َ    | # Fatha
                             ً    | # Tanwin Fath
                             ُ    | # Damma
                             ٌ    | # Tanwin Damm
                             ِ    | # Kasra
                             ٍ    | # Tanwin Kasr
                             ْ    | # Sukun
                             ـ     # Tatwil/Kashida
                         """, re.VERBOSE)

def preprocess(text):
    '''
    text is an Arabic string input
    the preprocessed text is returned
    '''
    # Remove punctuations
    translator = str.maketrans('', '', punctuations)
    text = text.translate(translator)

    # Remove Tashkeel
    text = re.sub(arabic_diacritics, '', text)

    # Remove Longation
    text = re.sub("[إأآا]", "ا", text)
    text = re.sub("ى", "ي", text)
    text = re.sub("ؤ", "ء", text)
    text = re.sub("ئ", "ء", text)
    text = re.sub("ة", "ه", text)
    text = re.sub("گ", "ك", text)

    text = ' '.join(word for word in text.split() if word not in stop_words)

    return text

df1['Feed'] = df1['Feed'].apply(preprocess)
print(df1.head(5))
```

The Cleaned Text

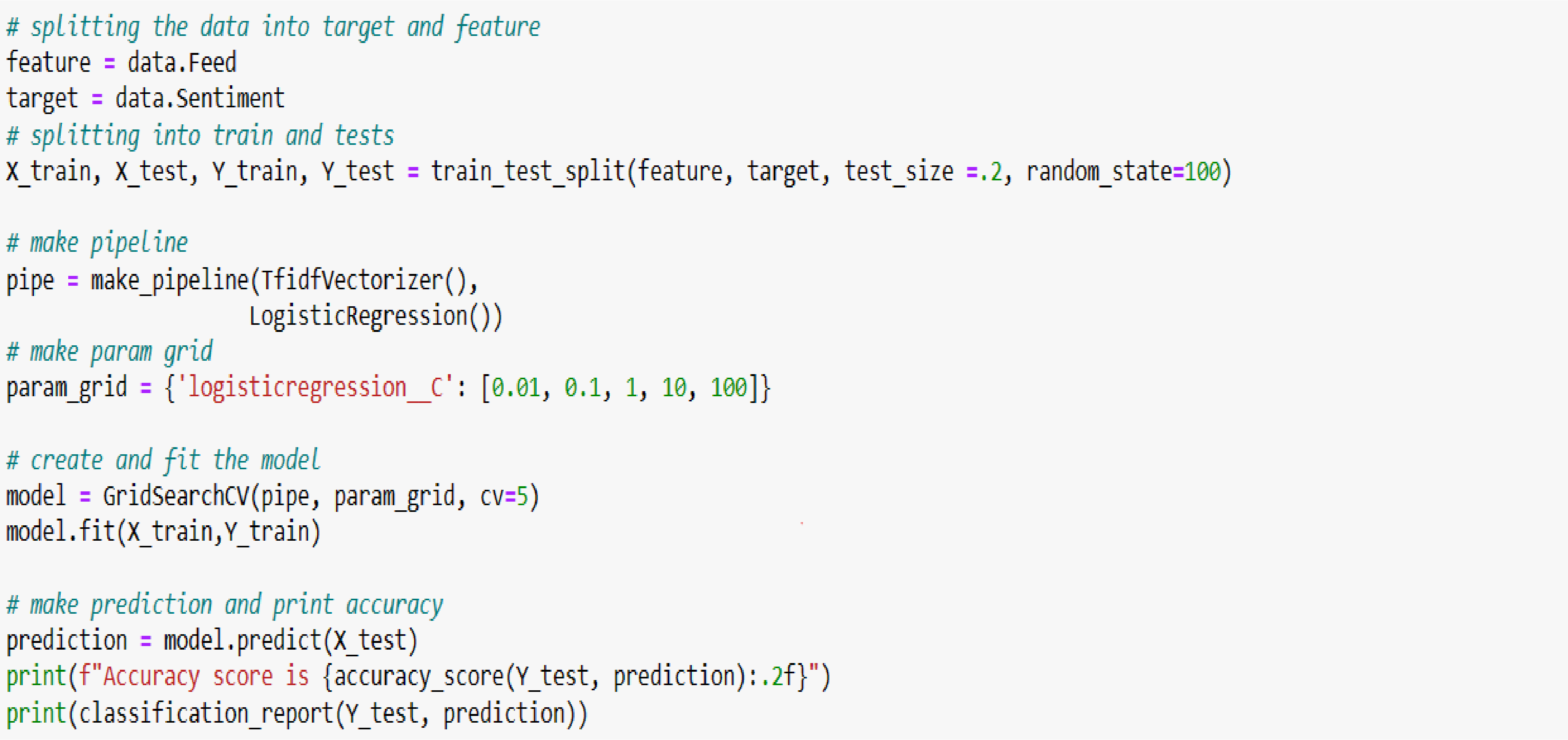
We can see the text in the picture after getting processed.



```
   ID                                    Feed Sentiment
0   1  [1]      اربد جامعات اكثر عمان وفيها عمان ونص لعيبه الم...
1   2   [2]      الحلو انكم بتحكوا علي اساس انو الاردن فساد سرقات
2   3                          [1]        كله راءع بجد ربنا يكرمك
3   4                              [2]        لسانك قذر قمامه
4   5  [2]      انا داشره وغير متزوجه ولدي علاقات مشبوه واحشش...
```

Building The Model

Logistic Regression is a very common classification algorithm. It is simple to implement and can serve as a baseline algorithm for classification tasks. In order to make the code shorter, Pipeline class in Scilkit-Learn which combines vectorization, transformation, gridsearch and classification is used.

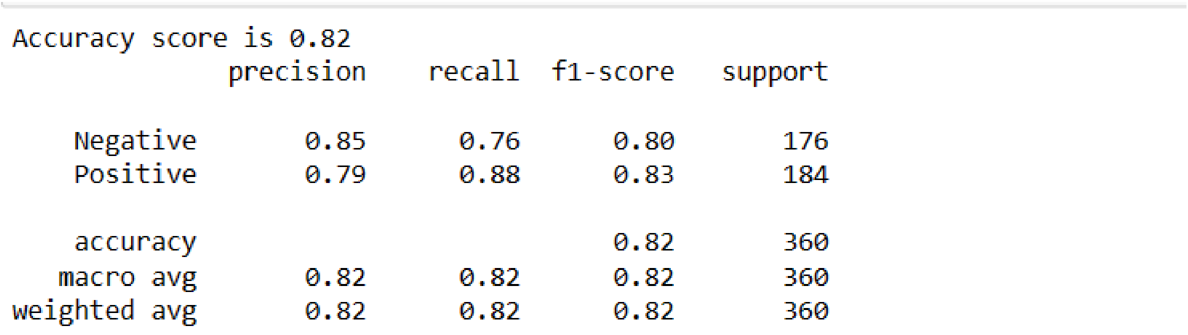


```
# splitting the data into target and feature
feature = data.Feed
target = data.Sentiment
# splitting into train and tests
X_train, X_test, Y_train, Y_test = train_test_split(feature, target, test_size =.2, random_state=100)

# make pipeline
pipe = make_pipeline(TfidfVectorizer(),
                    LogisticRegression())
# make param grid
param_grid = {'logisticregression__C': [0.01, 0.1, 1, 10, 100]}

# create and fit the model
model = GridSearchCV(pipe, param_grid, cv=5)
model.fit(X_train,Y_train)

# make prediction and print accuracy
prediction = model.predict(X_test)
print(f"Accuracy score is {accuracy_score(Y_test, prediction):.2f}")
print(classification_report(Y_test, prediction))
```

Logistic Regression Results

After we built the first layer of the model with the Logistic Regression Algorithm, the results produced results as shown in the picture.

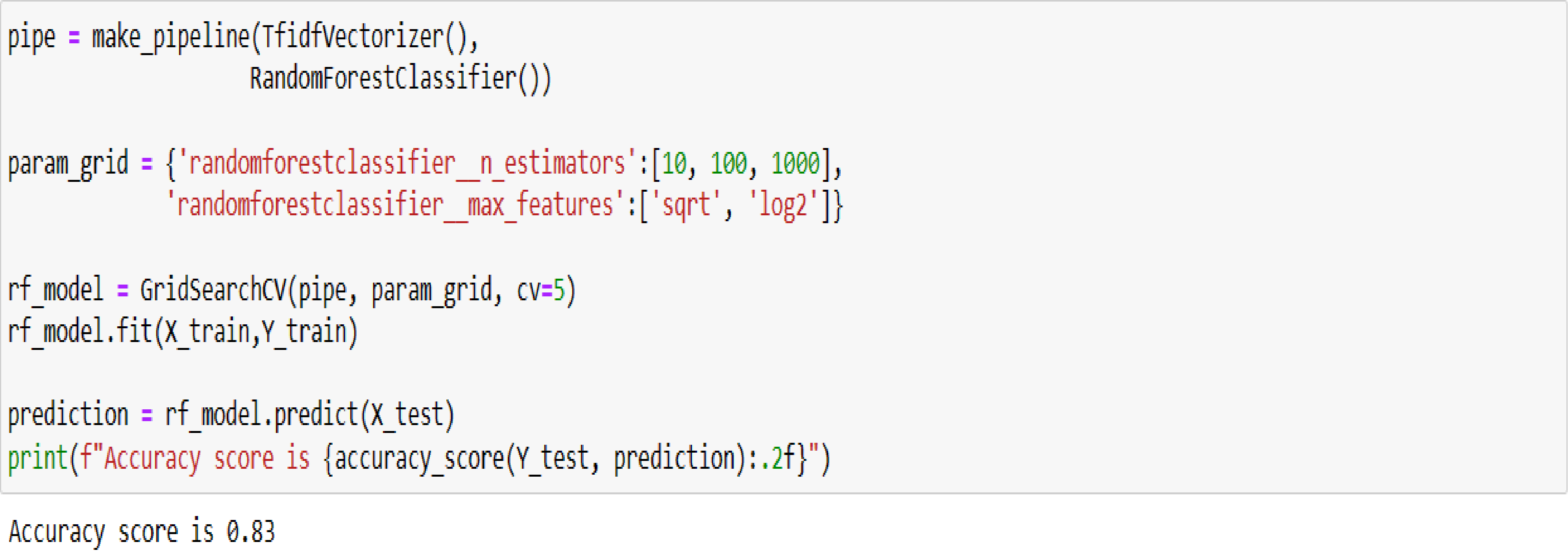
The model we have trained achieved an accuracy of 82% on the test set. The precision, recall, and F1-scores for both the "Negative" and "Positive" classes are relatively balanced, indicating a reasonable performance overall. The precision indicates the percentage of correct predictions for each class, while recall represents the percentage of instances correctly identified. The F1-score is a balanced measure that considers both precision and recall. The support values indicate the number of instances in each class.



```
Accuracy score is 0.82
              precision    recall  f1-score   support

    Negative       0.85      0.76      0.80       176
    Positive       0.79      0.88      0.83       184

    accuracy                           0.82       360
   macro avg       0.82      0.82      0.82       360
weighted avg       0.82      0.82      0.82       360
```

Random Forest Classifier

After using the Random Forest Classifier, which is an ensemble learning method that constructs multiple decision trees and combines their predictions to make a final prediction. That was the second layer of the Model, and it gave an accuracy result of 0.83

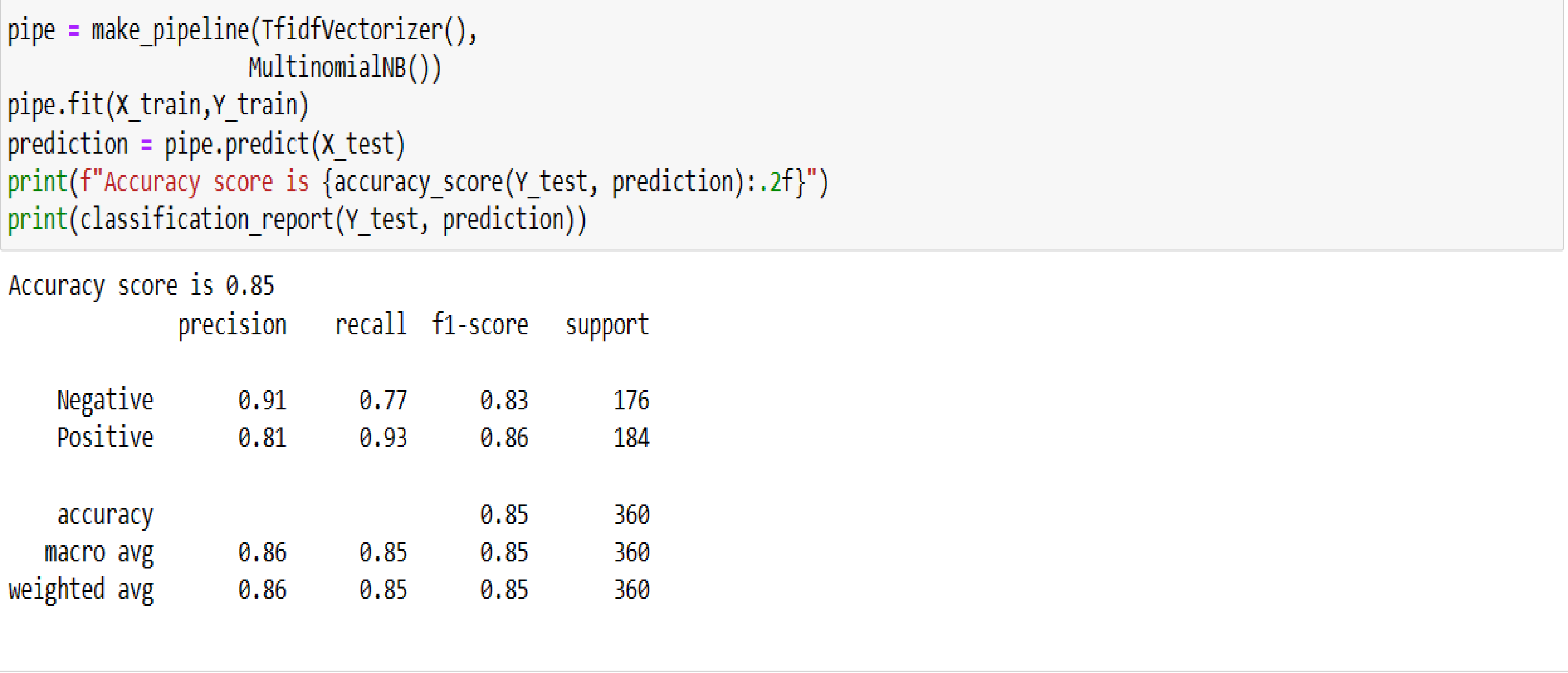


```
pipe = make_pipeline(TfidfVectorizer(),
                     RandomForestClassifier())

param_grid = {'randomforestclassifier__n_estimators':[10, 100, 1000],
             'randomforestclassifier__max_features':['sqrt', 'log2']}

rf_model = GridSearchCV(pipe, param_grid, cv=5)
rf_model.fit(X_train,Y_train)

prediction = rf_model.predict(X_test)
print(f"Accuracy score is {accuracy_score(Y_test, prediction):.2f}")
```

```
Accuracy score is 0.83
```

Naive Bayes Classifier (Multinomial)

While we were trying to improve the accuracy, we have used the Multinomial Classifier and that improved the accuracy by 0.02

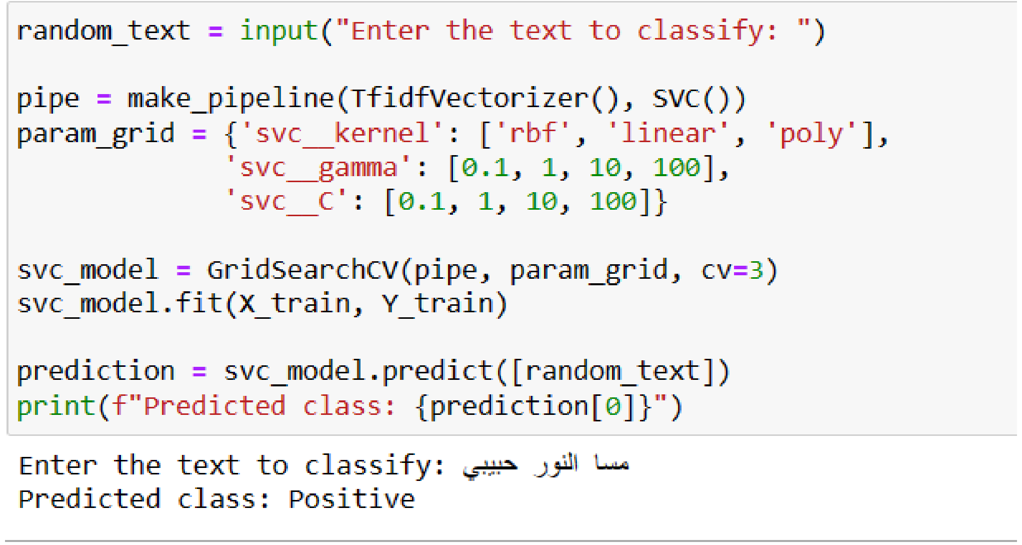


```
pipe = make_pipeline(TfidfVectorizer(),
                     MultinomialNB())
pipe.fit(X_train,Y_train)
prediction = pipe.predict(X_test)
print(f"Accuracy score is {accuracy_score(Y_test, prediction):.2f}")
print(classification_report(Y_test, prediction))
```

```
Accuracy score is 0.85
              precision    recall  f1-score   support

    Negative       0.91      0.77      0.83       176
    Positive       0.81      0.93      0.86       184

    accuracy                           0.85       360
   macro avg       0.86      0.85      0.85       360
weighted avg       0.86      0.85      0.85       360
```

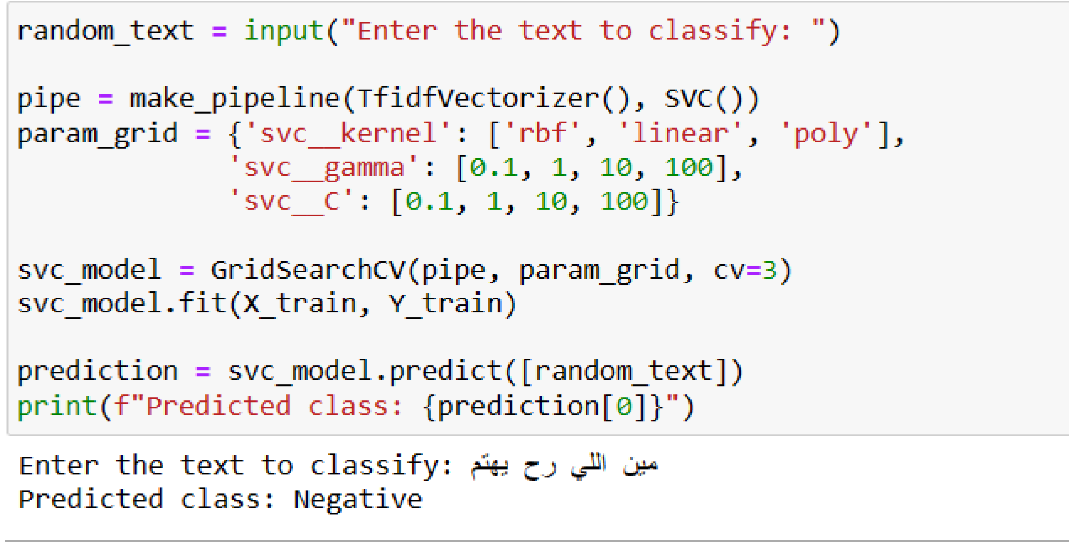
Results: Support Vector Machine (SVM)

Finally, after we applied the SVM algorithm we have tested the model with random texts and the results were correct for these samples as shown below



```
random_text = input("Enter the text to classify: ")

pipe = make_pipeline(TfidfVectorizer(), SVC())
param_grid = {'svc__kernel': ['rbf', 'linear', 'poly'],
              'svc__gamma': [0.1, 1, 10, 100],
              'svc__C': [0.1, 1, 10, 100]}

svc_model = GridSearchCV(pipe, param_grid, cv=3)
svc_model.fit(X_train, Y_train)

prediction = svc_model.predict([random_text])
print(f"Predicted class: {prediction[0]}")
```

```
Enter the text to classify: مسا النور حبيبي
Predicted class: Positive
```



```
random_text = input("Enter the text to classify: ")

pipe = make_pipeline(TfidfVectorizer(), SVC())
param_grid = {'svc__kernel': ['rbf', 'linear', 'poly'],
              'svc__gamma': [0.1, 1, 10, 100],
              'svc__C': [0.1, 1, 10, 100]}

svc_model = GridSearchCV(pipe, param_grid, cv=3)
svc_model.fit(X_train, Y_train)

prediction = svc_model.predict([random_text])
print(f"Predicted class: {prediction[0]}")
```

```
Enter the text to classify: مين اللي رح يهتم
Predicted class: Negative
```