AcrossTheBorders

SYRIAN REFUGEE (IN TURKEY) TWEETS SENTIMENT ANALYSIS AND FAKE JOB DETECTION

Presented By:

TEAM SIGMA

Ishita Gupta (211472)

Shambhavi Singh (211343)

Vanshi Goyal (211364)

Jatin Rajput (211399)

PROBLEM STATEMENT

Due to the ongoing conflict in Syria, many Syrians continue to seek refuge in Turkey. This immigration has resulted in development of hatred in Turkey for the refugees leading to global resentment. This hatred has especially been in the form of social backlash especially via Twitter. Also, many Syrian refugees are manipulated by Turkish criminals by offering fraudulent jobs.

PROPOSED IDEA

In order to counter the problem of online hatred via Twitter, we decided to build a Machine Learning model for performing sentiment analysis of tweets regarding Syrian refugees in Turkey and segregate them depending on their sentiment as positive,negative or neutral.

This analysed data can be used by the Turkish government to sue the potential defaulters

In order to track the fraudulent job tweets, we decide to employ another ML model.

TWITTER SENTIMENT ANALYSIS

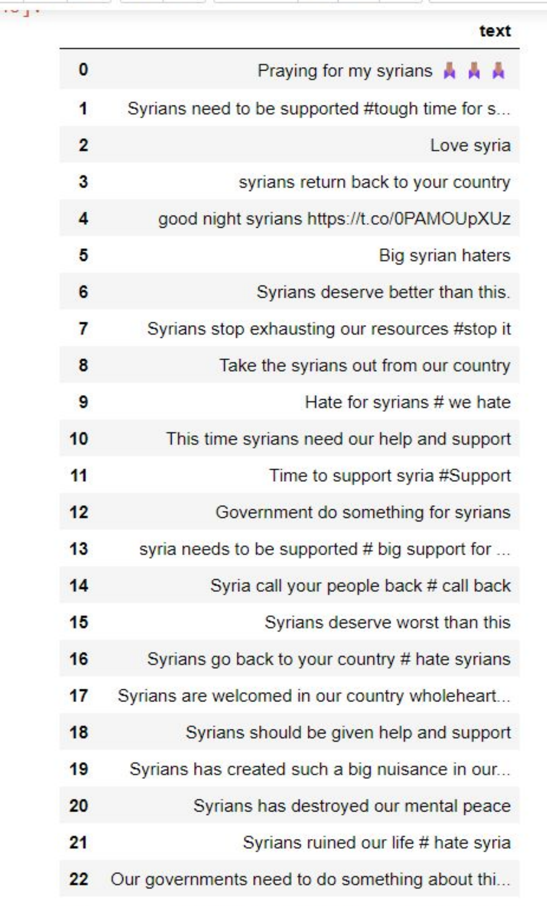
In this project we use Natural Language Toolkit (NLTK)
for analysing the sentiments of tweets (Syrian refugees).
We use the LogisticRegression model for the analysis and prediction of data.
The following steps are followed for the same:

1. Importing Necessary Modules and dataset
2. Data Preprocessing and Visualisation
3. Model Building and Training
4. Obtaining the classification report and displaying the confusion matrix.

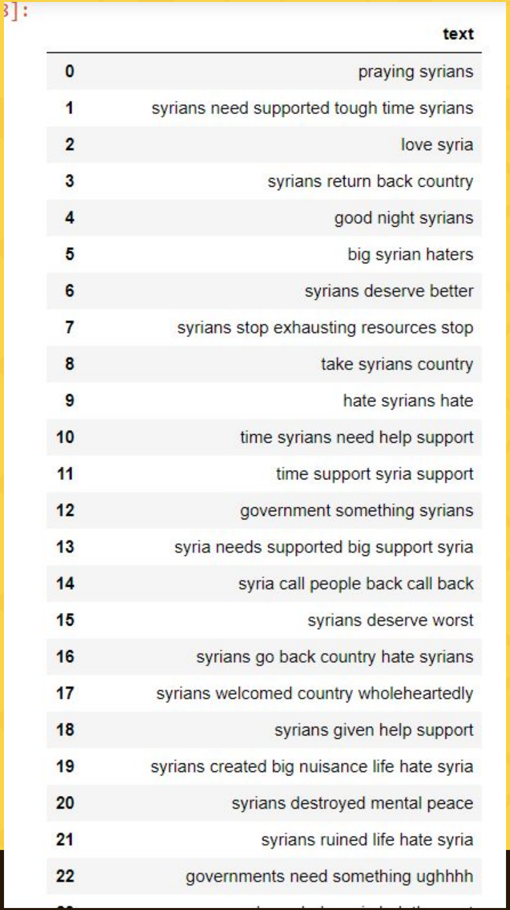
1. IMPORTING NECESSARY LIBRARIES AND DATASET

1. pandas (for data analysis and basic operations like DataFrames and Series)
2. numpy as np (for data analysis and mathematical calculations)
3. Re (for regex)
4. seaborn (for data visualisation)
5. matplotlib.pyplot (for data visualisation)
6. TextBlob (process the textual data)
7. word_tokenize (for tokenization)
8. PorterStemmer (for stemming)
9. stopwords (to remove stopwords)
10. CountVectorizer (to vectorize the text document)
11. train_test_split (to split the data into training and testing data)
12. LogisticRegression (to perform logistic regression)
13. classification_report, accuracy_score, confusion_matrix, ConfusionMatrixDisplay

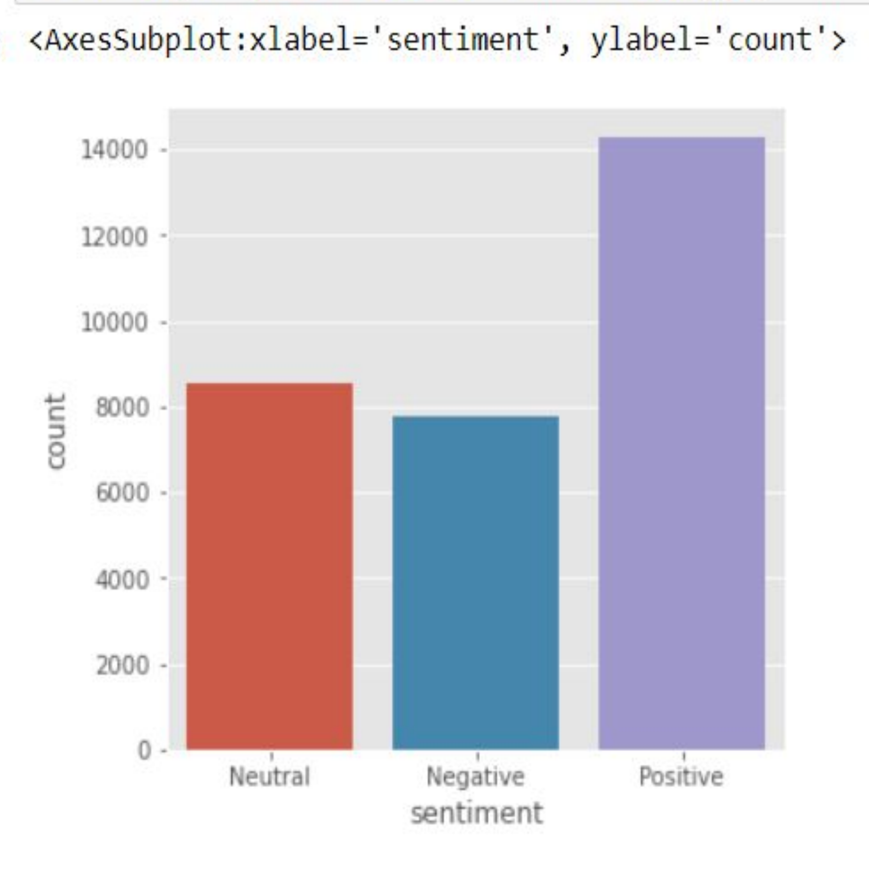
2. DATA PREPROCESSING AND VISUALISATION



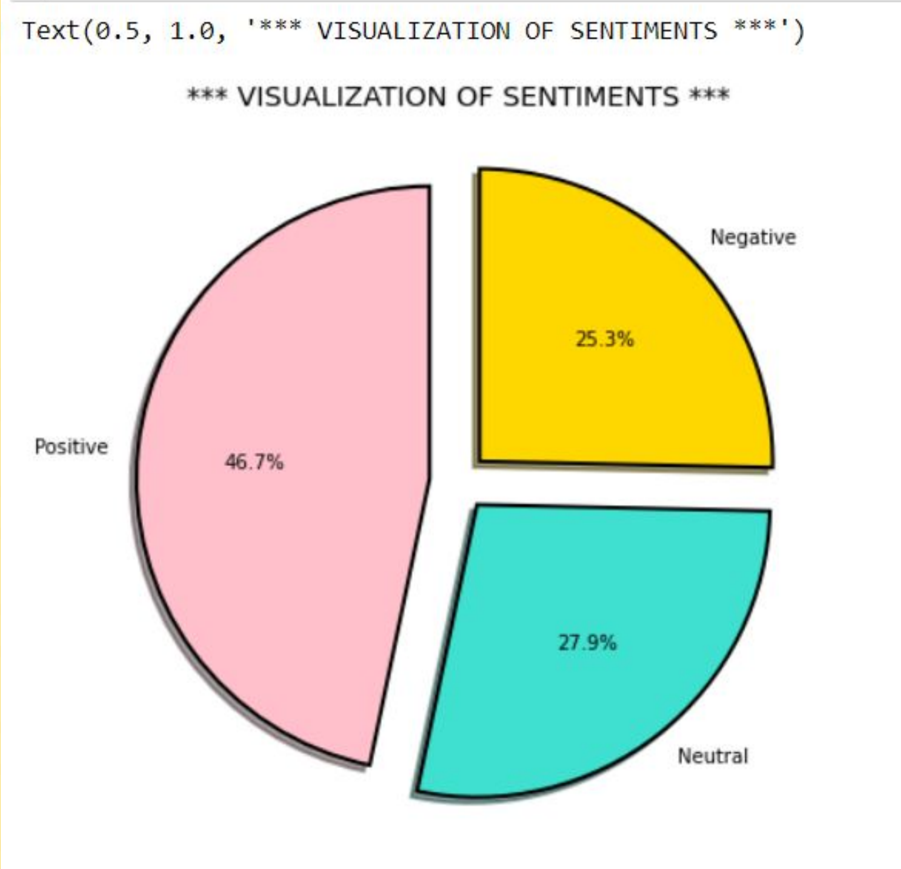
	text
0	Praying for my syrians 🙏🙏🙏
1	Syrians need to be supported #tough time for s...
2	Love syria
3	syrians return back to your country
4	good night syrians https://t.co/0PAMOUpXUz
5	Big syrian haters
6	Syrians deserve better than this.
7	Syrians stop exhausting our resources #stop it
8	Take the syrians out from our country
9	Hate for syrians # we hate
10	This time syrians need our help and support
11	Time to support syria #Support
12	Government do something for syrians
13	syria needs to be supported # big support for ...
14	Syria call your people back # call back
15	Syrians deserve worst than this
16	Syrians go back to your country # hate syrians
17	Syrians are welcomed in our country wholeheart...
18	Syrians should be given help and support
19	Syrians has created such a big nuisance in our...
20	Syrians has destroyed our mental peace
21	Syrians ruined our life # hate syria
22	Our governments need to do something about thi...



	text
0	praying syrians
1	syrians need supported tough time syrians
2	love syria
3	syrians return back country
4	good night syrians
5	big syrian haters
6	syrians deserve better
7	syrians stop exhausting resources stop
8	take syrians country
9	hate syrians hate
10	time syrians need help support
11	time support syria support
12	government something syrians
13	syria needs supported big support syria
14	syria call people back call back
15	syrians deserve worst
16	syrians go back country hate syrians
17	syrians welcomed country wholeheartedly
18	syrians given help support
19	syrians created big nuisance life hate syria
20	syrians destroyed mental peace
21	syrians ruined life hate syria
22	governments need something ughhhh



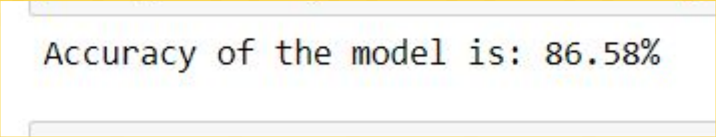
```
<AxesSubplot:xlabel='sentiment', ylabel='count'>
```



```
Text(0.5, 1.0, '*** VISUALIZATION OF SENTIMENTS ***')
```

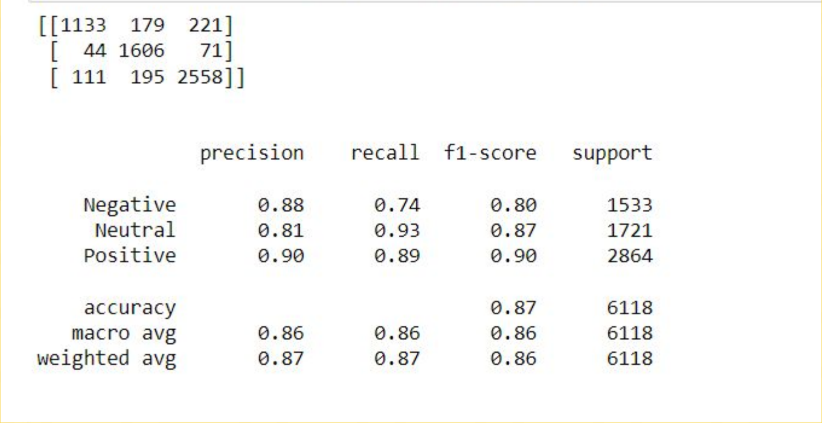
3. MODEL BUILDING AND TRAINING

1. Count Vectorization
2. Splitting of data into testing and training data
3. Train the data on LogisticRegression Model
4. Predict the value for test data
5. Calculating accuracy

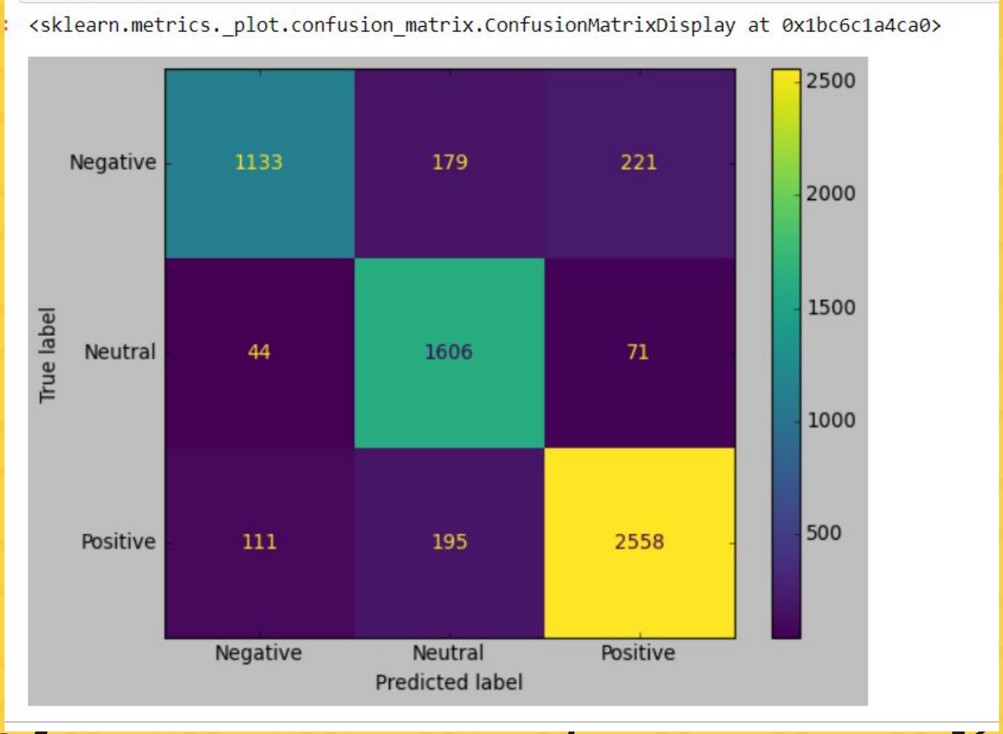


```
Accuracy of the model is: 86.58%
```

4. OBTAINING THE CLASSIFICATION REPORT AND CONFUSION MATRIX

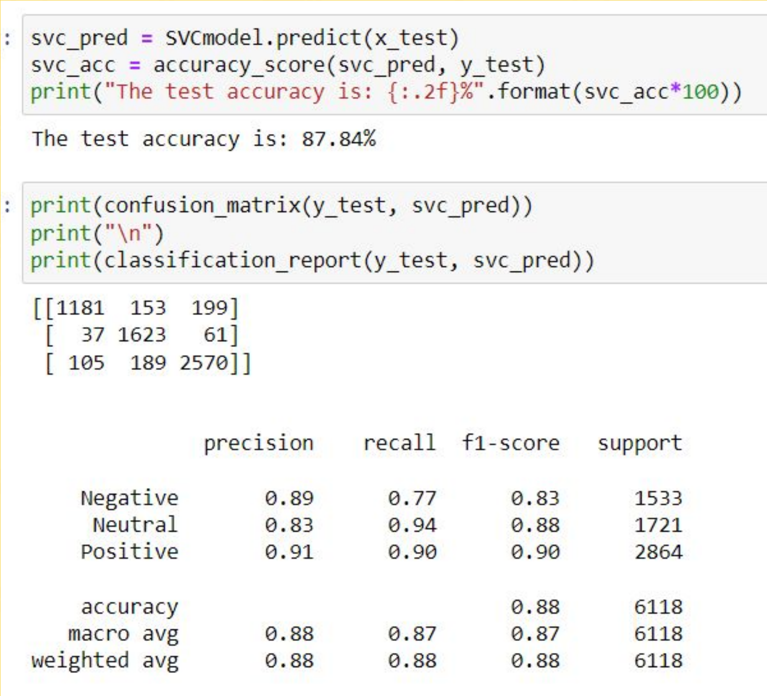


```
[[1133  179  221]
 [  44 1606   71]
 [ 111  195 2558]]


              precision    recall  f1-score   support

    Negative       0.88      0.74      0.80      1533
     Neutral       0.81      0.93      0.87      1721
    Positive       0.90      0.89      0.90      2864

    accuracy                           0.87      6118
   macro avg       0.86      0.86      0.86      6118
weighted avg       0.87      0.87      0.86      6118
```



```
: <sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x1bc6c1a4ca0>
```

ACCURACY USING SVC MODEL



```
svc_pred = SVCmodel.predict(x_test)
svc_acc = accuracy_score(svc_pred, y_test)
print("The test accuracy is: {:.2f}%".format(svc_acc*100))
```

```
The test accuracy is: 87.84%
```

```
print(confusion_matrix(y_test, svc_pred))
print("\n")
print(classification_report(y_test, svc_pred))
```

```
[[1181  153  199]
 [  37 1623   61]
 [ 105  189 2570]]


              precision    recall  f1-score   support

    Negative       0.89      0.77      0.83      1533
     Neutral       0.83      0.94      0.88      1721
    Positive       0.91      0.90      0.90      2864

    accuracy                           0.88      6118
   macro avg       0.88      0.87      0.87      6118
weighted avg       0.88      0.88      0.88      6118
```

FRAUD JOB DETECTION MODEL

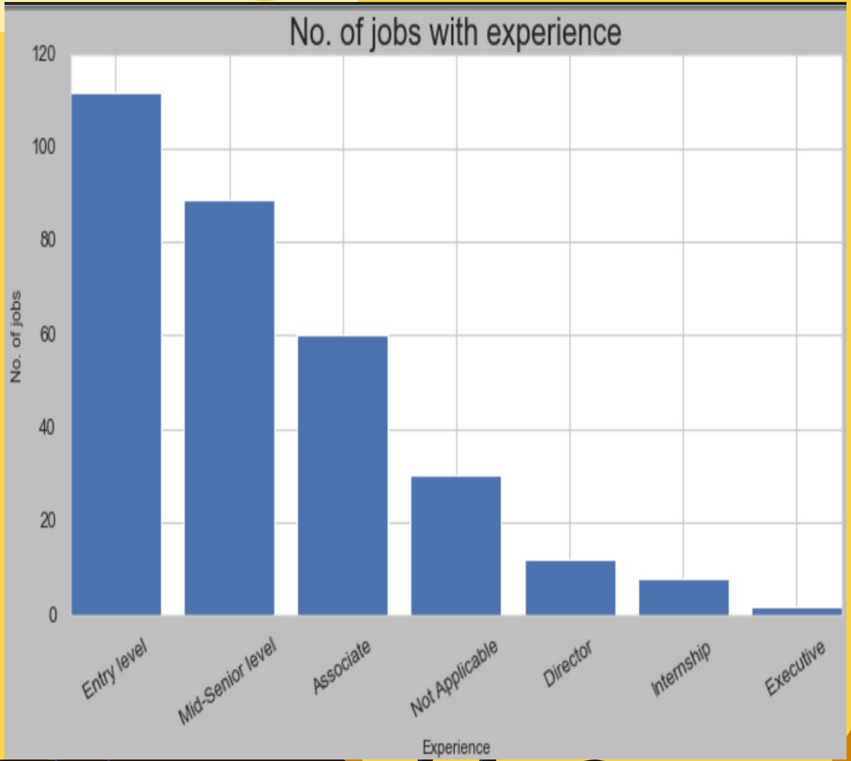
In this part of the project we use Spacy
for detection of fake job tweets.
We use the RandomForest model for the analysis and prediction of data.
The following steps are followed for the same:

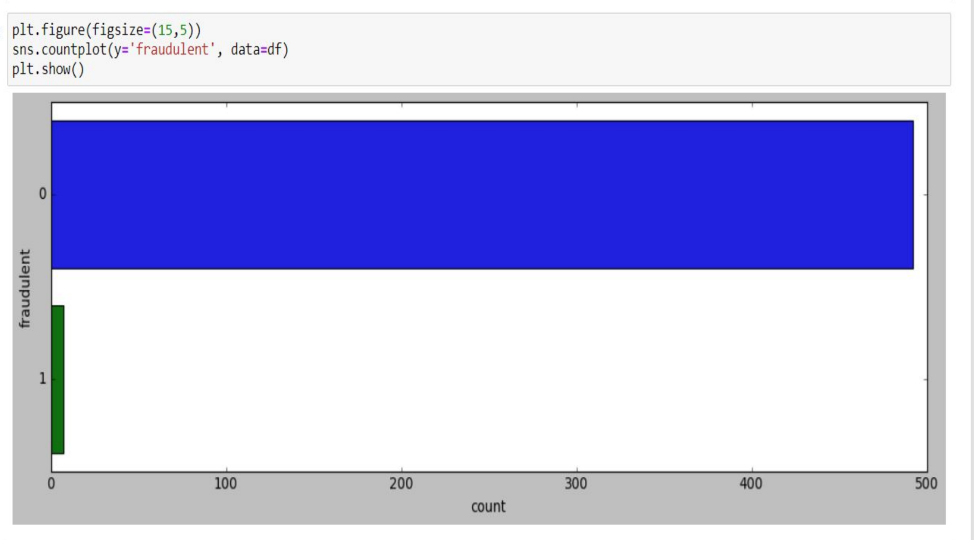
1. Importing Necessary Modules and dataset
2. Analysis of data using bar charts
3. Tokenizing and data cleaning
4. Feature Engineering
5. Training the model
6. To calculate accuracy and display the confusion matrix

1. IMPORTING NECESSARY LIBRARIES AND DATASET

1. string
2. random
3. TfidfVectorizer, CountVectorizer
4. Pipeline
5. TransformerMixin
6. ispacy
7. STOP_WORDS
8. English

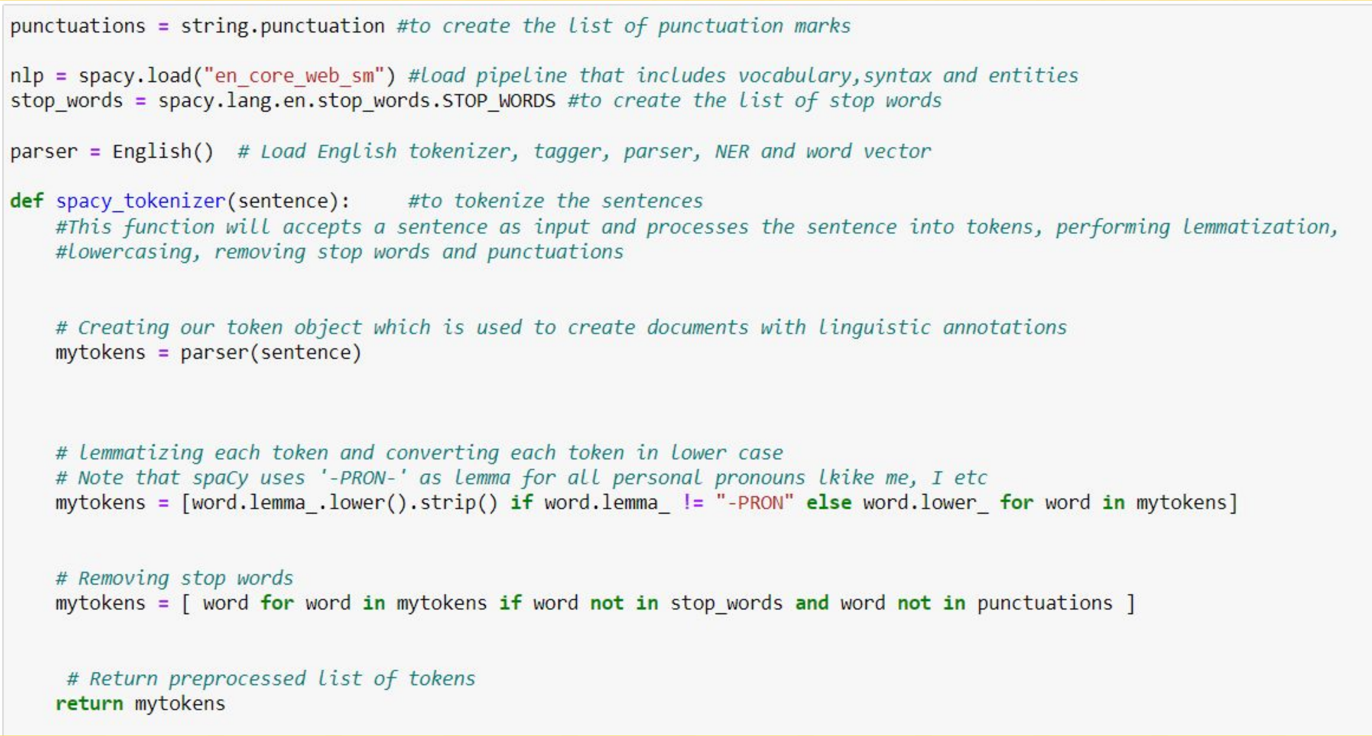
2. ANALYSIS OF DATA USING BAR CHARTS



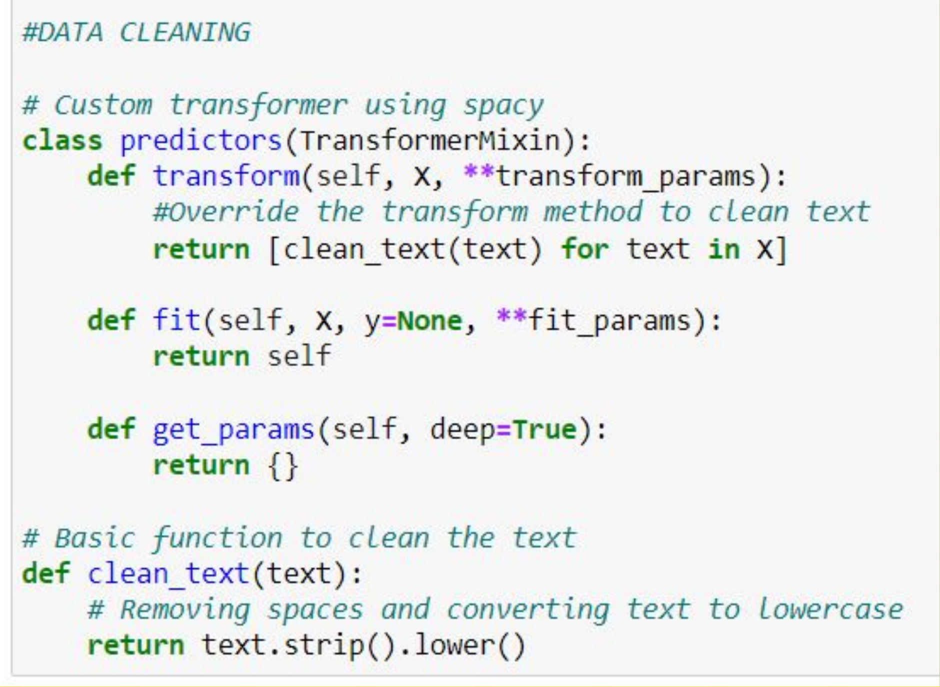


```
plt.figure(figsize=(15,5))
sns.countplot(y='fraudulent', data=df)
plt.show()
```

3. TOKENIZING AND DATA CLEANING

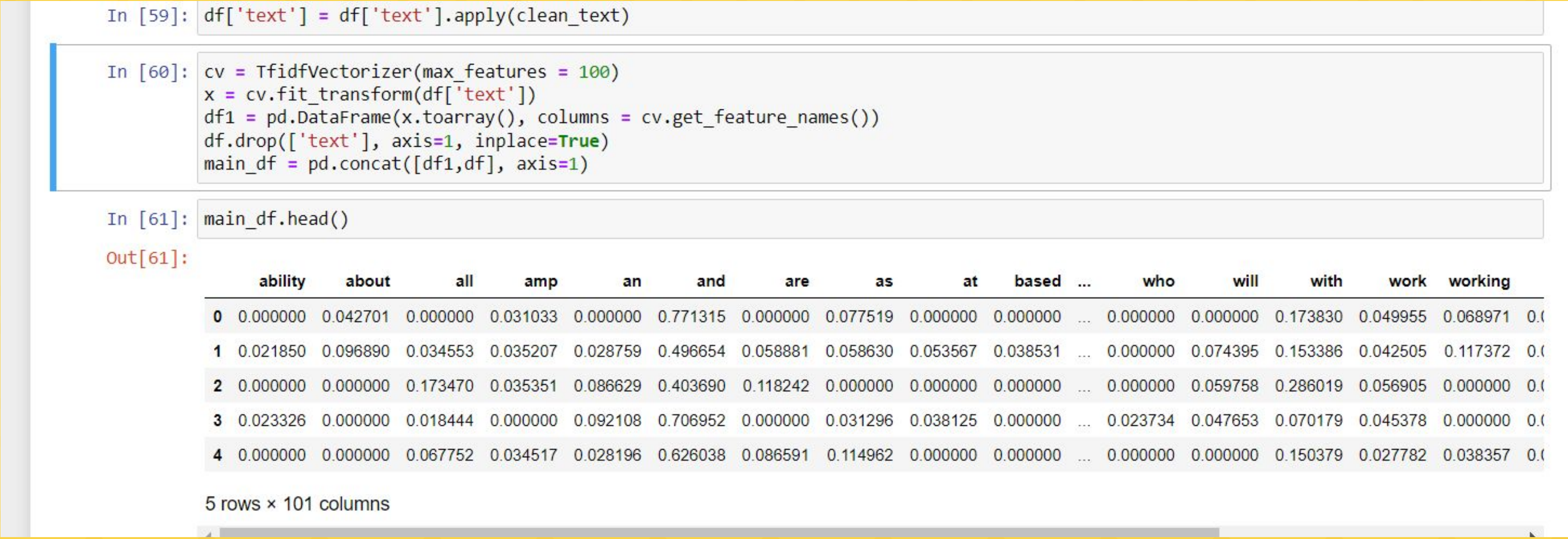


```
punctuations = string.punctuation #to create the list of punctuation marks

nlp = spacy.load("en_core_web_sm") #load pipeline that includes vocabulary,syntax and entities
stop_words = spacy.lang.en.stop_words.STOP_WORDS #to create the list of stop words

parser = English()  # Load English tokenizer, tagger, parser, NER and word vector

def spacy_tokenizer(sentence):     #to tokenize the sentences
    #This function will accepts a sentence as input and processes the sentence into tokens, performing lemmatization,
    #lowercasing, removing stop words and punctuations


    # Creating our token object which is used to create documents with linguistic annotations
    mytokens = parser(sentence)



    # lemmatizing each token and converting each token in lower case
    # Note that spaCy uses '-PRON-' as lemma for all personal pronouns lkike me, I etc
    mytokens = [word.lemma_.lower().strip() if word.lemma_ != "-PRON" else word.lower_ for word in mytokens]


    # Removing stop words
    mytokens = [ word for word in mytokens if word not in stop_words and word not in punctuations ]


     # Return preprocessed list of tokens
    return mytokens
```



```
#DATA CLEANING

# Custom transformer using spacy
class predictors(TransformerMixin):
    def transform(self, X, **transform_params):
        #Override the transform method to clean text
        return [clean_text(text) for text in X]

    def fit(self, X, y=None, **fit_params):
        return self

    def get_params(self, deep=True):
        return {}

# Basic function to clean the text
def clean_text(text):
    # Removing spaces and converting text to lowercase
    return text.strip().lower()
```

4.FEATURE ENGINEERING



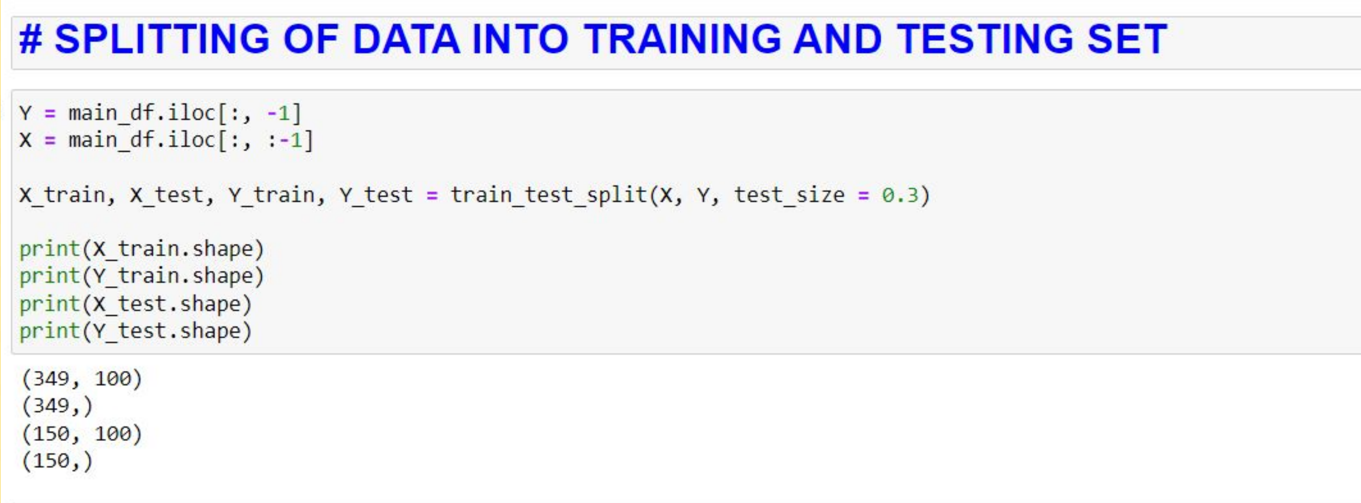
```
In [59]: df['text'] = df['text'].apply(clean_text)
```

```
In [60]: cv = TfidfVectorizer(max_features = 100)
         x = cv.fit_transform(df['text'])
         df1 = pd.DataFrame(x.toarray(), columns = cv.get_feature_names())
         df.drop(['text'], axis=1, inplace=True)
         main_df = pd.concat([df1,df], axis=1)
```

```
In [61]: main_df.head()
```

```
Out[61]:
      ability     about       all       amp        an       and       are        as        at     based  ...       who      will      with      work   working
   0 0.000000  0.042701  0.000000  0.031033  0.000000  0.771315  0.000000  0.077519  0.000000  0.000000  ...  0.000000  0.000000  0.173830  0.049955  0.068971  0.(
   1 0.021850  0.096890  0.034553  0.035207  0.028759  0.496654  0.058881  0.058630  0.053567  0.038531  ...  0.000000  0.074395  0.153386  0.042505  0.117372  0.(
   2 0.000000  0.000000  0.173470  0.035351  0.086629  0.403690  0.118242  0.000000  0.000000  0.000000  ...  0.000000  0.059758  0.286019  0.056905  0.000000  0.(
   3 0.023326  0.000000  0.018444  0.000000  0.092108  0.706952  0.000000  0.031296  0.038125  0.000000  ...  0.023734  0.047653  0.070179  0.045378  0.000000  0.(
   4 0.000000  0.000000  0.067752  0.034517  0.028196  0.626038  0.086591  0.114962  0.000000  0.000000  ...  0.000000  0.000000  0.150379  0.027782  0.038357  0.(

5 rows × 101 columns
```

5. TRAINING THE MODEL

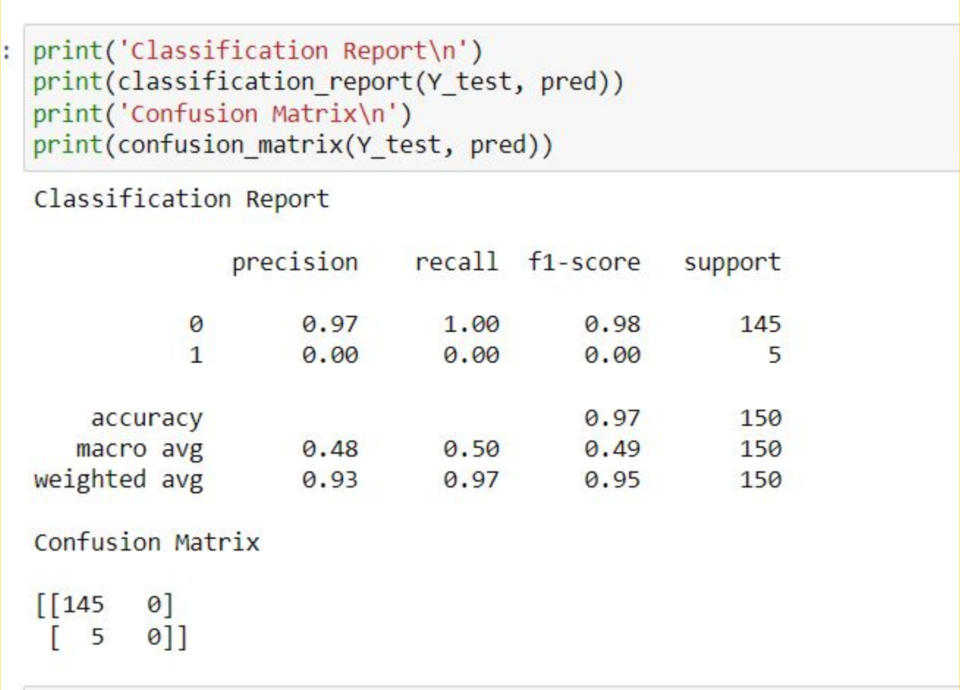


SPLITTING OF DATA INTO TRAINING AND TESTING SET

```
Y = main_df.iloc[:, -1]
X = main_df.iloc[:, :-1]

X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size = 0.3)

print(X_train.shape)
print(Y_train.shape)
print(X_test.shape)
print(Y_test.shape)
```

```
(349, 100)
(349,)
(150, 100)
(150,)
```

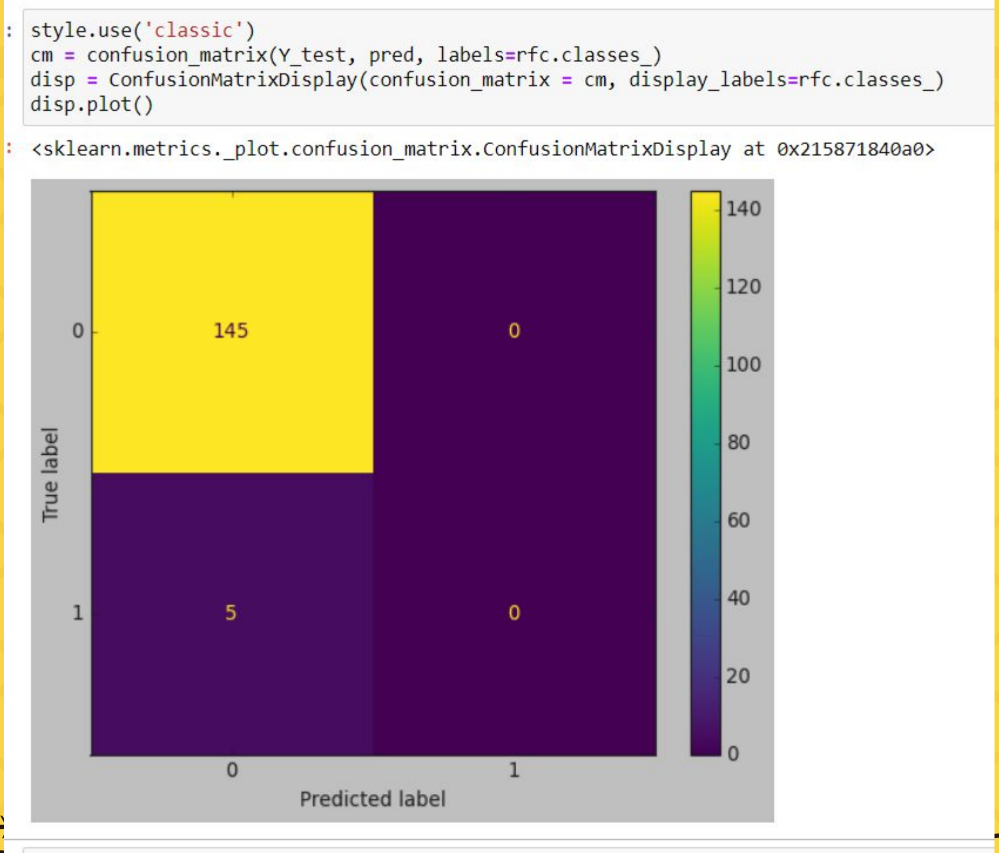
6. TO CALCULATE ACCURACY AND DISPLAY THE CONFUSION MATRIX



```
print('Classification Report\n')
print(classification_report(Y_test, pred))
print('Confusion Matrix\n')
print(confusion_matrix(Y_test, pred))
```

```
Classification Report

              precision    recall  f1-score   support

           0       0.97      1.00      0.98       145
           1       0.00      0.00      0.00         5

    accuracy                           0.97       150
   macro avg       0.48      0.50      0.49       150
weighted avg       0.93      0.97      0.95       150

Confusion Matrix

[[145   0]
 [  5   0]]
```

Confusion Matrix



```
style.use('classic')
cm = confusion_matrix(Y_test, pred, labels=rfc.classes_)
disp = ConfusionMatrixDisplay(confusion_matrix = cm, display_labels=rfc.classes_)
disp.plot()
```

```
<sklearn.metrics._plot.confusion_matrix.ConfusionMatrixDisplay at 0x215871840a0>
```

REFERENCES

https://www.kaggle.com/code/satishgunjal/tutorial-text-classification-using-spacy

https://medium.com/swlh/confusion-matrix-and-classification-report-88105288d48f

http://repository.bilkent.edu.tr/bitstream/handle/11693/52835/Twitter_sentiment_analysis_3_way_classification_positive_negative_or_neutral.pdf?sequence=1

THANK YOU