CENG 3548-WEB MINING
FINAL REPORT

170709061-Fatma Karadağ,
170709050-Gizem Pesen
Topic : Twitter Sentiment Analysis

Sunday 30th May, 2021

Contents

Abstract

It is a Natural Language Processing Problem where Sentiment Analysis is done by Classifying the positive tweets and negative tweets by machine learning models with classification, text mining, text analysis, data analysis and data visualization

1 Introduction

Twitter is a social media environment with 330 million monthly active users. It enables many people of different religions, races and geographies to stay in interaction.It is a 'microblogging' system that allows you to send and receive short posts called tweets. Tweets can be up to 140 characters long and can include links to relevant websites and resources.[1].

In this project assignment, we tried to examine and analyze the data of twitter (train.tweet and test.tweet). While analyzing, we tried to get more accurate values by using multiple different classification methods. Generally, we gave importance to visualize and explain more clearly while introducing the data. The tweets we analyze are usually sentimental tweets (positive or negative words).

2 Used Technologies

- Python 3

  Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. It can be downloaded in referance [2]

- Jupyter Notebook

  It is opened with **Anaconda** .

- Latex

3 Libraries

3.1 nltk

NLTK stands for natural language toolkit. Natural Language Toolkit; It is an open source library created with over 50 corpus and lexical resources developed and being developed with Python programming language to work with human language data. There are also a number of modules in this library, these modules are packages that we will use when preprocessing our data, using machine learning algorithms, making transactions with Twitter API, etc. For example; We can give the operations of separating the words in a sentence (Tokenization), removing the suffixes in the word and finding the root (Stemming).[3] .

Listing 1: nltk

```
#Natural Language Toolkit(NLTK)
!pip install nltk

import nltk

nltk.download('stopwords')
from nltk.corpus import stopwords
from nltk.stem.porter import PorterStemmer
```

3.2 numpy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays[4]. [5].

Listing 2: numpy

```
import numpy as np
```

In addition to this , numpy can be found on github. [6].

3.3 pandas

Pandas is a Python library used for working with data sets.It has functions for analyzing, cleaning, exploring, and manipulating data.The name ”Pandas” has a reference to both ”Panel Data”, and ”Python Data Analysis” and was created by Wes McKinney in 2008. [7].

Listing 3: pandas

```
import pandas as pd
```

3.4 matplotlib and seaborn

Matplotlib produces publication-quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shell, web application servers, and various graphical user interface toolkits.[7].

Listing 4: matplotlib

```
import pandas as pd
```

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.[8].

Listing 5: seaborn

```
import pandas as pd
```

3.5 re

A regular expression (or RE) specifies a set of strings that matches it; the functions in this module let you check if a particular string matches a given regular expression (or if a given regular expression matches a particular string, which comes down to the same thing).[9].

Listing 6: re

```
import re
```

3.6 wordcloud

Gizem noticed that the wordcloud library was used in all the examples that examined the tweeter data and suggested using it.

Word clouds (also known as text clouds or tag clouds) work in a simple way: the more a specific word appears in a source of textual data (such as a speech, blog post, or database), the bigger and bolder it appears in the word cloud.

A word cloud is a collection, or cluster, of words depicted in different sizes. The bigger and bolder the word appears, the more often it's mentioned within a given text and the more important it is.

Also known as tag clouds or text clouds, these are ideal ways to pull out the most pertinent parts of textual data, from blog posts to databases. They can also help business users compare and contrast two different pieces of text to find the wording similarities between the two.[10].

Listing 7: wordcloud

```
from wordcloud import WordCloud
```

In our project worldcloud used as;

Listing 8: wordcloud

```
wordcloud = WordCloud(background_color = 'white', width = 1000, height =
    1000).generate_from_frequencies(dict(words_frequency))

plt.figure(figsize=(10,8))
plt.imshow(wordcloud)
plt.title("WordCloud - Vocabulary from Reviews", fontsize = 22)
```

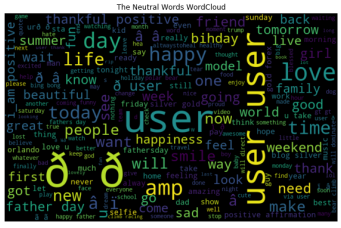

Figure 1: An example wordcloud output from our project

4 Code Analysis and Outputs

4.0.1 Data and Head

All group members opened the csv files from the location on their computer, they used jupyter as an ide and zoom as a communication tool. Fatma printed the head and the first 10 columns train part of the data, and Gizem printed the test part.

By pd.read csv (), the file is read by entering the datasets inside.

Listing 9: data

```
train =
    pd.read_csv(r'C:\Users\pesen\OneDrive\Desktop\Twitter-Sentiment-Analysis-master\train_tweet.csv')
test =
    pd.read_csv(r'C:\Users\pesen\OneDrive\Desktop\Twitter-Sentiment-Analysis-master\test_tweets.csv')
```

With **head ()**, data belonging to a few lines is taken from the dataset and displayed on the screen. The desired number can be written here and the data can be drawn. For example, head (10) is drawn the first 10 lines.

Listing 10: head

```
train.head()
test.head()
```

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation

Figure 2: Train head output

	id	tweet
0	31963	#studiolife #aislife #requires #passion #dedic...
1	31964	@user #white #supremacists want everyone to s...
2	31965	safe ways to heal your #acne!! #altwaystohe...
3	31966	is the hp and the cursed child book up for res...
4	31967	3rd #bihday to my amazing, hilarious #nephew...

Figure 3: Test head output

4.0.2 Check Data With Isnull Method

isnull (). Any () method, Fatma suggested to check if there is a gap in the data with the isnull method , we checked if there is a gap and got a false output and continued our way with the result we got from this output.

Listing 11: isnull

```
train.isnull().any() #Checked with isnull() method and it return false
test.isnull().any()
```

```
id       False
tweet    False
dtype: bool
```

Figure 4: There is no null values

.

4.0.3 Negative and Positive Words

Gizem looked at the examples and compared the tweets as 1 and 0 in binaries . With this line of code, we assigned the label value to 0 for negative words. We assigned a value of 1 to the label, which is positive. Thus, we have defined it as **binary**.

Listing 12: label 0

```
# to check the negative comments on the train set
train[train['label'] == 0].head(10)
```

Positive and negative results were visualized. (Plot is used for visualization.) You can see clearly negative tweets are more than positive tweets with this graph.

Listing 13: 0 and 1

```
#We can see clearly negative comments are more than positive comments
train['label'].value_counts().plot.bar(color = 'pink', figsize = (6, 4))
```

.

	id	label	tweet
0	1	0	@user when a father is dysfunctional and is s...
1	2	0	@user @user thanks for #lyft credit i can't us...
2	3	0	bihday your majesty
3	4	0	#model i love u take with u all the time in ...
4	5	0	factsguide: society now #motivation
5	6	0	[2/2] huge fan fare and big talking before the...
6	7	0	@user camping tomorrow @user @user @user @use...
7	8	0	the next school year is the year for exams.ð□□...
8	9	0	we won!!! love the land!!! #allin #cavs #champ...
9	10	0	@user @user welcome here ! i'm it's so #gr...

Figure 5: Label negative

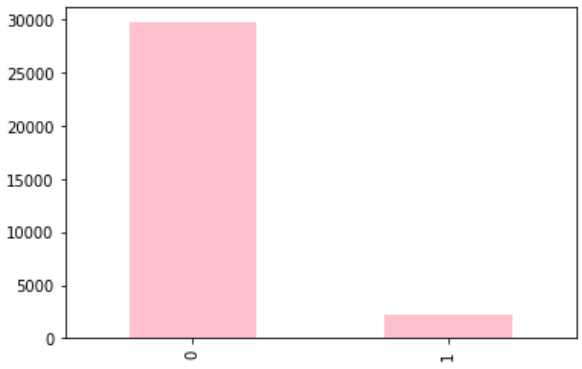

Figure 6: Comparing positive and negative tweets

4.0.4 Train and Test Data

Fatma compared her train and test data with the graph using her favorite colors of orange and pink. we stated that there is more train data from here.

Listing 14: label 0

```
# adding a column to represent the length of the tweet

train['len'] = train['tweet'].str.len()
test['len'] = test['tweet'].str.len()

train.head(10)
```

.

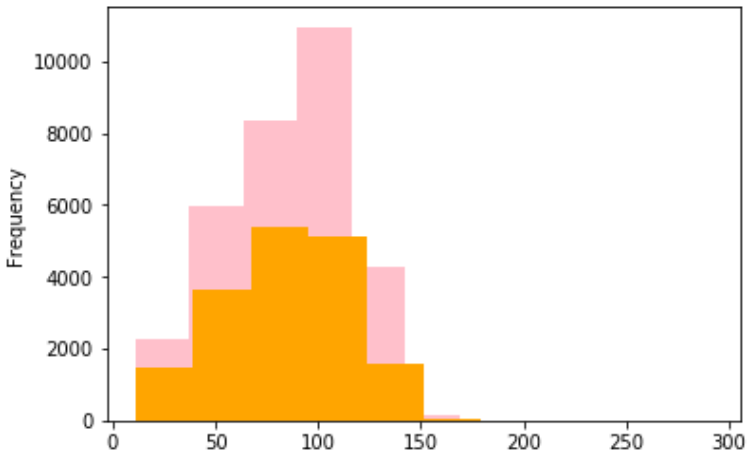

Figure 7: Train and Test

4.0.5 Length

Gizem added the data column as we needed the length of the words.

Listing 15: label 0

```
# adding a column to represent the length of the tweet

train['len'] = train['tweet'].str.len()
test['len'] = test['tweet'].str.len()

train.head(10)
```

	id	label	tweet	len
0	1	0	@user when a father is dysfunctional and is s...	102
1	2	0	@user @user thanks for #lyft credit i can't us...	122
2	3	0	bihday your majesty	21
3	4	0	#model i love u take with u all the time in ...	86
4	5	0	factsguide: society now #motivation	39
5	6	0	[2/2] huge fan fare and big talking before the...	116
6	7	0	@user camping tomorrow @user @user @user @use...	74
7	8	0	the next school year is the year for exams.ð□□...	143
8	9	0	we won!!! love the land!!! #allin #cavs #champ...	87
9	10	0	@user @user welcome here ! i'm it's so #gr...	50

Figure 8: Adding lenght

4.0.6 Group Data

Gizem grouped them as negative, positive tweets. With methods, also adding a column to represent the length of the tweet. train.groupby ('label'). With the describe () method, we grouped negative and positive labels (labels defined as 0 and 1) in the train dataset and displayed them in a single table.

Listing 16: group

```
train.groupby('label').describe()
```

	id								len							
	count	mean	std	min	25%	50%	75%	max	count	mean	std	min	25%	50%	75%	max
label																
0	29720.0	15974.454441	9223.783469	1.0	7981.75	15971.5	23965.25	31962.0	29720.0	84.328634	29.566484	11.0	62.0	88.0	107.0	274.0
1	2242.0	16074.896075	9267.955758	14.0	8075.25	16095.0	24022.00	31961.0	2242.0	90.187779	27.375502	12.0	69.0	96.0	111.0	152.0

Figure 9: Groupby

4.0.7 Variation of Lenght

Fatma plotted the variation of length and found the relationship between frequency and length. The variety of the train dataset was shown with the methods.

Listing 17: group

```
train.groupby('len').mean()['label'].plot.hist(color = 'black', figsize = (6, 4),)
plt.title('variation of length')
plt.xlabel('Length')
plt.show()
```

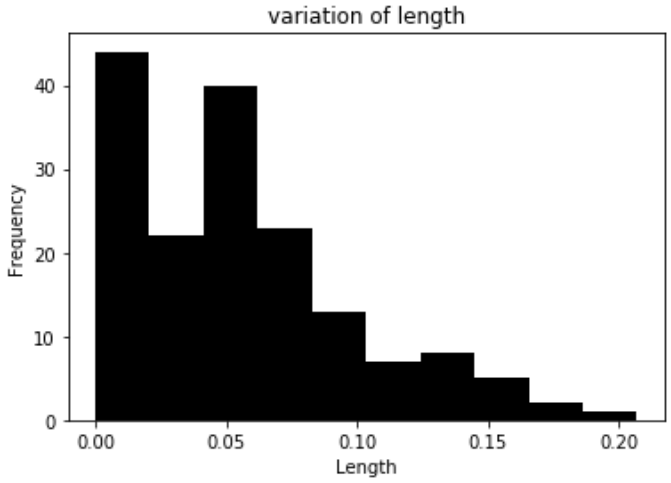

Figure 10: Variation of lenght

4.0.8 Most Frequently Occuring Words

Both group members did the import and plotting of the Countvectorizer part together, and they had difficulty in this process.The frequency of the words in English was calculated using CountVectorizer from the Sklearn library, and the following graphic was created in order from high to low.

Listing 18: group

```
from sklearn.feature_extraction.text import CountVectorizer
count_vector = CountVectorizer(stop_words = 'english') #
words = count_vector.fit_transform(train.tweet)
sum_words = words.sum(axis=0)

words_frequency = [(word, sum_words[0, i]) for word, i in
    count_vector.vocabulary_.items()]
# oktan  aza giden sort edildi.
words_frequency = sorted(words_frequency, key = lambda x: x[1], reverse = True)
```

```
Text(0.5, 1.0, 'Most Frequently Occuring Words - Top 30')
```

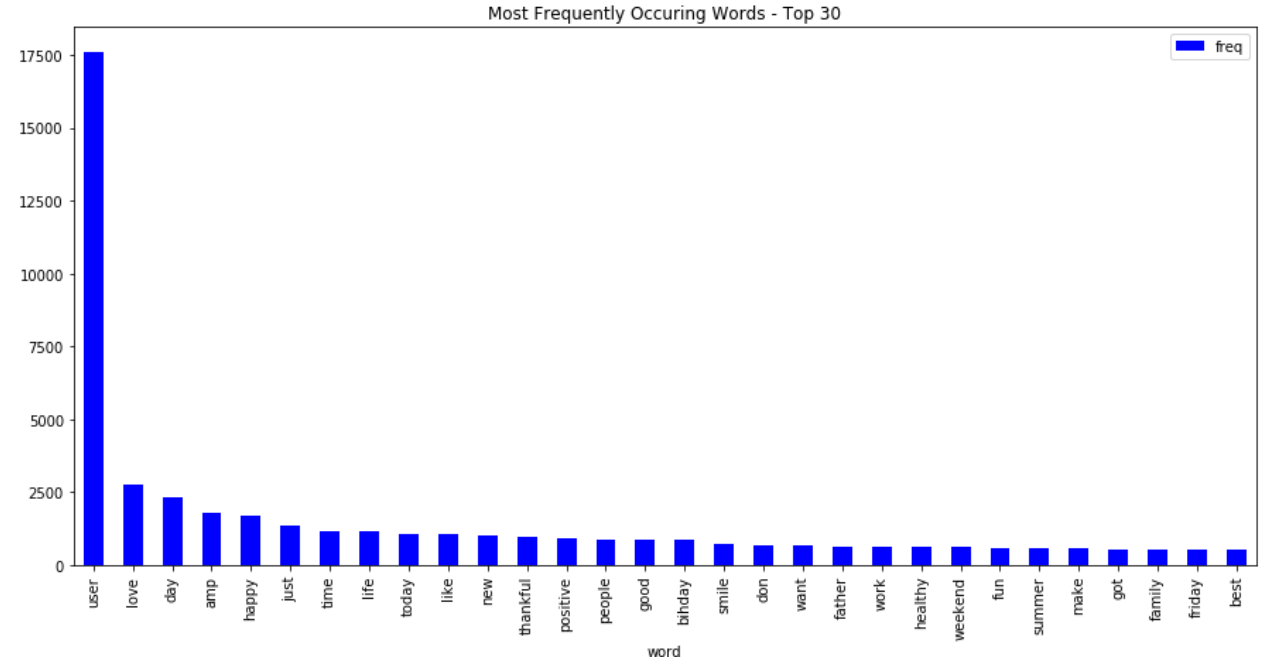

Figure 11

5 Predictions

5.1 StandardScaler

Most machine learning algorithms use the Euclidean formula, which measures the distance between two data points, one of the distance metrics, in their calculations. To ensure that all values contribute equally, the values must be brought to the same unit. The data, which varies in terms of size, will increase the variance and will not have an equal effect in distance calculations, as they will outweigh the low-value features in weight calculations. Data is scaled by feature standardization or Z-score normalization method. For this process, StandardScaler method of sklearn.preprocessing library is imported and used in python.

5.2 RandomForestClassifier

The Random Forest algorithm can be used in both classification and regression problems such as decision trees. Working logic creates more than one decision tree. When it will produce a result, the average value in these decision trees is taken and the result is produced.

5.2.1 RandomForestClassifier Code Analysis

we first included the libraries required for classification.

Listing 19: RandomForestClassifier

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix
from sklearn.metrics import f1_score
```

Then we assign the RandomForestClassifier() classification to the variable named model. And in the other classification methods we have made, we define the model variable and assign the classification methods to the model

Listing 20: model

```
model = RandomForestClassifier()
```

With the model fit () method, the information contained in the datasets is retrieved or in other words, the information is kept.

Listing 21: model.fit

```
model.fit(x_train, y_train)
```

We get predictions about the test data using the model.predict () function. Here we compare the basic accuracy values for the x valid sample with the predictions in our model.

Listing 22: prediction

```
y_pred = model.predict(x_valid)
```

We did print to print model.score on the screen.

Listing 23: prediction

```
print("Training Accuracy :", model.score(x_train, y_train))
print("Validation Accuracy :", model.score(x_valid, y_valid))
```

calculating the f1 score for the validation set

Listing 24: prediction

```
print("F1 score :", f1_score(y_valid, y_pred))
```

The confusion matrix allows us to compare test data with predicted values and measure the performance of our model. TP (True Positive) and TN (True Negative) values give the correct number of values. FP (False Positive) and FN (False Negative) values give the number of false values. The ratio of the number of correct values to all the number of values indicates the Correct Prediction Ratio. The ratio of the number of false values to all the number of values indicates the False Prediction Rate.

Listing 25: cm

```
cm = confusion_matrix(y_valid, y_pred)
print(cm)
```

Output

Listing 26: Accuracy

```
Training Accuracy : 0.9944933461265696
Validation Accuracy : 0.9481917156801402
F1 score : 0.5775510204081632
```

5.3 LogisticRegression

Logistic Regression is a regression method for classification. It is used to classify categorical or numerical data. It works only if the dependent variable, ie the result, can take 2 different values. (Yes / No, Male / Female, Fat / Thin etc.)

Output

Listing 27: Accuracy

```
Training Accuracy : 0.984773267698469
Validation Accuracy : 0.9410586910274058
f1 score : 0.5915004336513443
```

5.4 DecisionTreeClassifier

The decision tree is a recursively process, as the name suggests, a tree structure is used. A tree structure is created by starting with a single node and branching into new results. When the algorithm runs, the entered value moves on a certain path by looking at the nodes and gives a result. There are 3 types of knots.

1. Chance Node: It is indicated with a circle. Indicates multiple possible paths.

2. Decision Node: It is indicated with a rectangle. Indicates that a decision will be made.

3. End Node: It is indicated with a triangle. Indicates a result.

Output

Listing 28: Accuracy

```
Training Accuracy : 0.9991656585040257
Validation Accuracy : 0.9325491177574772
f1 score : 0.5404944586530265
```

5.5 SVC

It can classify or regress data by making use of gaps between data. However, it is generally used for classification. It can be applied in clustering or outlier detection. Basically, it allows us to distinguish between data with the help of SVM vectors.

Output

Listing 29: Accuracy

```
Training Accuracy : 0.978181969880272
Validation Accuracy : 0.9521962207483419
f1 score : 0.4986876640419947
```

5.6 SGDClassifier

Output

Listing 30: Accuracy

```
Training Accuracy : 0.9820199407617538
Validation Accuracy : 0.9499436866474784
f1 score : 0.5815899581589958
```

6 Conclusion

We tried to explain the libraries and classification methods we used while doing this project homework step by step above, and we supported the results of these analyzes with screenshots.

Since the scores we obtain from the classification methods we use are usually very close to 1 (between 0.94-0.98), it shows that the classification methods we use are the right choice for the data set we analyze. Sgd Classifier and Decision Tree Classifier calculated by Fatma ; RandomForestClassifier and SVC calculated by Gizem. The best prediction is 0.9991656585040257 with Decision Tree Classifier. In the few articles we have read, your score for your data set close to 1 has been written as proof that analyzing the data you have is the right choice. Taking this as a reference, we think our choices are correct.

7 References

[] https://www.veribilimiokulu.com/kernel-support-vector-machine-svm-ile-siniflandirma-python-ile
[] https://www.websitehostingrating.com/tr/twitter-statistics/
[] https://ai.yemreak.com/makine-ogrenimi/scikit-learn