Capstone Project

Machine Learning Engineer Nanodegree

Ibrahim Munther

### Sentiment Analysis of Arabic Text Data (Tweets)

**Abstract**

This project addresses the problem of sentiment analysis on Twitter. The goal of this project was to predict sentiment for the given Twitter post using Python. Sentiment analysis can predict many different emotions attached to the text, but in this report, only 2 major were considered: positive and negative. The training dataset was small (just over 45000 examples) and the data within it was highly skewed, which greatly impacted on the difficulty of building a good classifier. After creating a lot of custom features, utilizing bag-of-words representations and applying the logistic regression algorithm, the classification accuracy at the level of 75% was achieved.

# Introduction

### Domain Background

This dataset we collected in April 2019 by an company. It contains 58K Arabic tweets annotated in positive and negative labels, The company collected this dataset to provide Arabic sentiment corpus for the research the comapny doing to investigate deep learning approaches for Arabic sentiment analysis. The dataset is balanced and collected using positive and negative emojis lexicon.

The ability to understand the public sentiment in social media is increasingly considered as an important tool for market understanding and customer segmentation.

So, from this project, I receive an opportunity to work in sentiment analysis field. Also, it will definitely be beneficial for my startup. Because while dealing with the reviews of customers, we want to interpret what user tends to portray so that we can give him best recommended results.

Apart from this, Sentiment analysis has been an interesting field of study. This is still an evolving subject, it has functions that are too complicated to understand by the machines such as sarcasm, negative emotions, hyperbole etc.

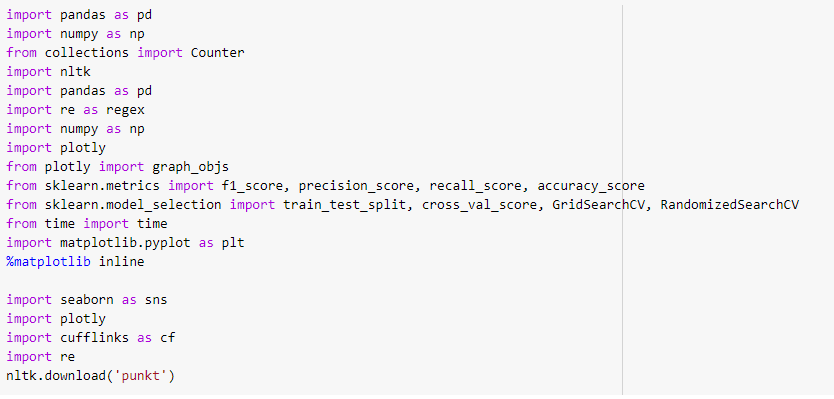
Because I am part of the industry, I know the potential in sentiment analysis. It adds a lot of value to the industry. Sentiment analysis bases its results on factors that are so inherently humane, it is bound to become one the major drivers of many business decisions in future.

### Problem Statement

The company collected this dataset to provide Arabic sentiment corpus for the research the company doing to investigate deep learning approaches for Arabic sentiment analysis.

# Used Python Libraries

Data was pre-processed using *pandas*, *gensim* and *numpy* libraries and the learning/validating process was built with *scikit-learn*. Plots were created using *Seaborn*.



# Datasets And Input

The input data consisted of two CSV files:

1. train.csv (45000 tweets)
2. test.csv (45000 tweets)

One for training and one for testing. The format of the data was the following (test data contain Class column):

The dataset contains the following attributes:

### Train data:

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 45275 entries, 0 to 45274

Data columns (total 2 columns):

# Column Non-Null Count Dtype

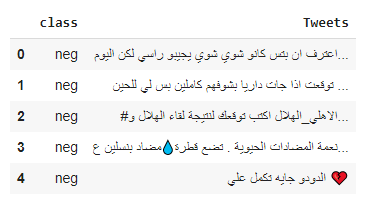
--- ------ -------------- -----

0 class 45275 non-null object

1 Tweets 45275 non-null object

dtypes: object(2)

memory usage: 707.5+ KB



Here, the’ ***class***’ is the target class, given the ‘***Tweets***’ column, ‘***class***’ defines whether the given user tweet is *positive or negative.*

### Text Data:

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 11520 entries, 0 to 11519

Data columns (total 2 columns):

# Column Non-Null Count Dtype

--- ------ -------------- -----

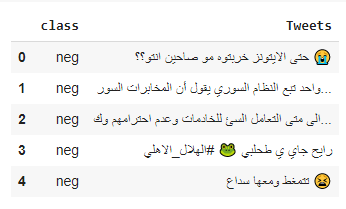
0 class 11520 non-null object

1 Tweets 11520 non-null object

dtypes: object(2)

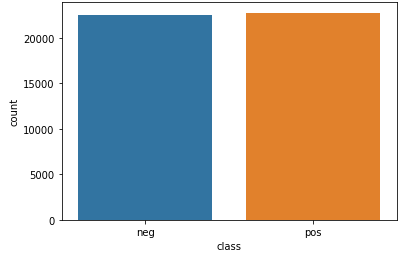
memory usage: 180.1+ KB

As the test.csv file was full of empty entries, they will be removed.



# Data Distibution

The below figure shows how the target class is dirtibuted.



# Preprocessing

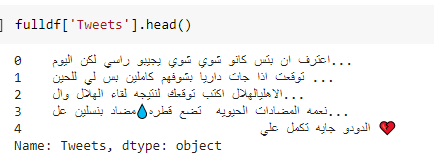
Data preprocessing is a data mining technique that involves transforming raw data into an understandable format. Real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. Data preprocessing is a proven method of resolving such issues. Data preprocessing prepares raw data for further processing.

The aim of the following preprocessing is to create representation of the data. The steps will execute as follows:

1. *Cleaning*
   * Remove URLs
   * Remove punctuatuions.
   * Remove longation.
   * Remove usernames (mentions)
   * Remove special characters
   * Remove numbers
2. *Text processing*
   * Tokenize
   * Stemming

### Data Cleaning:

Data cleaning is one of the crucial part to prepare the data for Bag-of-word representation. After Cleaning:



**Tokenization & stemming**

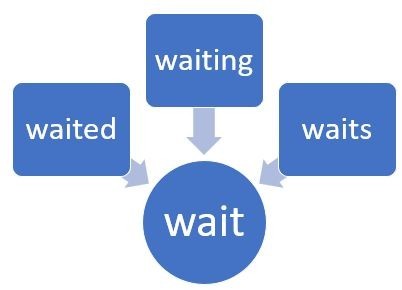
Tokenization consists of splitting the text into words, and words in a context that is with spaces, punctuation signs, cases, accents, diacritics, and so on — into standardized words. This step is critical for the whole flow accuracy

Stemming consists of preparing words expressions to find their stems. The process relies on a suffix dictionary making it possible to extract stems after analyzing the morphology of the word.

According to the inflected forms identified and the defined language, it computes the most relevant stems from the grammar and syntax rules of the language.

Stemming offers two main benefits:

* As it focuses on words stems, the process is quite tolerant of spelling mistakes.
* It only needs words to stem without requiring their context of use. Below feagure[10] is the stemming example:



For the text processing, nltk library is used. First, the tweets are tokenized using nlkt.word\_tokenize and then, stemming is done using **PorterStemmer** as the tweets are 100% in arabic.



# Wordlist

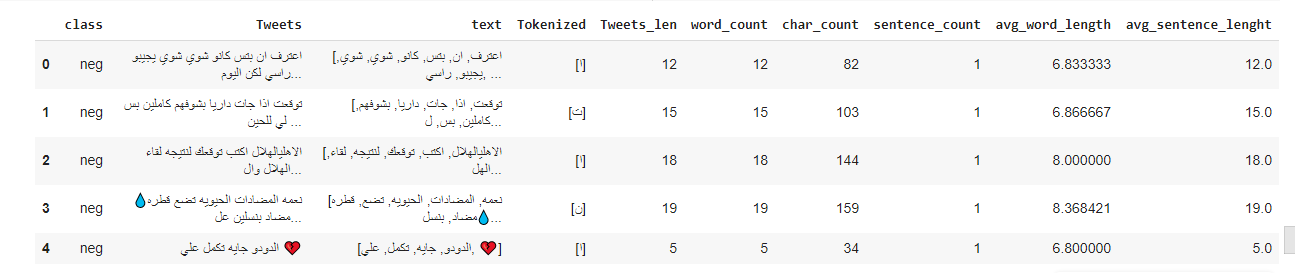
A bag-of-words model, or BoW for short, is a way of extracting features from the text for use in modeling, such as with machine learning algorithms.

The approach is very simple and flexible and can be used in a myriad of ways for extracting features from documents.

The wordlist (dictionary) is build by simple count of occurences of every unique word across all of the training dataset.

# Additional features

In order to **not** push any other algorithm to the limit on the current data model, let's try to add some features that might help to classify tweets.



**Feature name**

1. Length of tweets
2. .Number of words count
3. Number of char
4. Number of sentences
5. Average words length
6. Average sentence length.

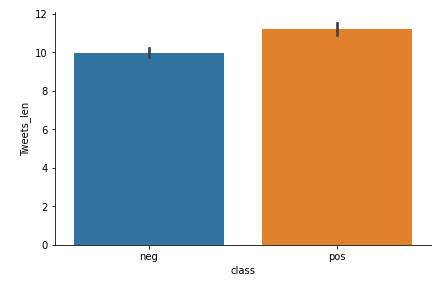
**Explanation**

1. Length of words in every tweet
2. Number of counts of every word in the tweets
3. Number of characaters in every tweet.
4. Number of sentences in every tweet.
5. The average of length of words in tweets.
6. The average of length of sentences in tweets

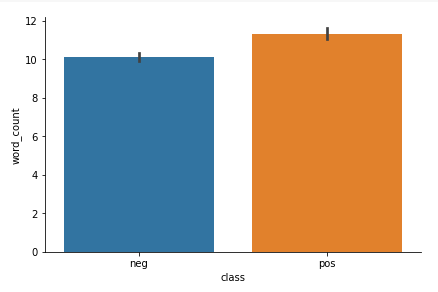
### Why to use additional features (Graph representaion)

Let's see how (some) of the extra features distribute the data set. Some of them, i.e number exclamation marks, number of pos/neg emoticons do this surprisingly well. Not with standing the good severance, those features seldom transpire only on a small subset of the training dataset.

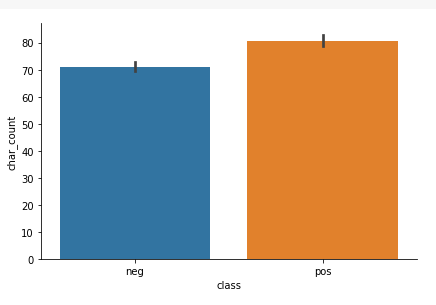
Length of tweets:



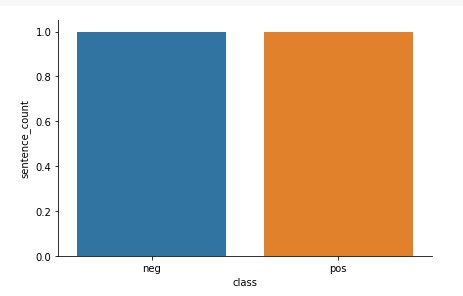
Number of words count:



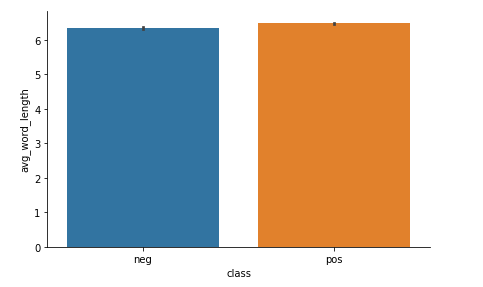
Number of char



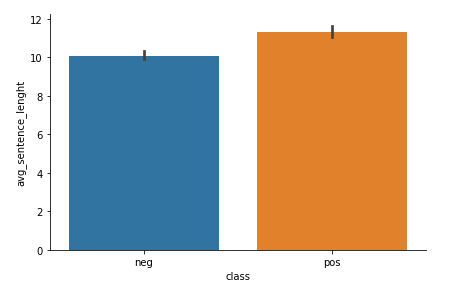
Number of sentences



Average words length:



Average sentence length:

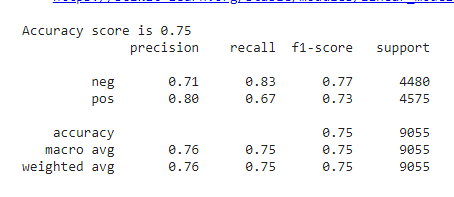


# Results

## Experiment 1: Logistic regression

It is nice to see what kind of results we might get from such simple model. Logistic Regression Classifier seems like a nice algorithm to start the experiments.

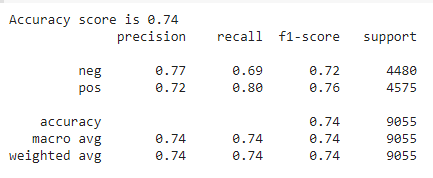
The experiment will be based on 7:3 train:test stratified split.



# Experiment 2: Naive bayes Classifier

As a second attempt on the classification the **Naïve Bayes Classifier** will be used.

It Doesn’t look better, And it’s not better than Logistic regression classifier.



We can observe a low recall level of the Logistic regression classifier for the negative class, which may be caused by the data skewness.

From the above experiments we can conclude that Logistic Regression classifier gives better result than other models hence we the Test Data will be classified on Logistic Regression.

## Test data classification

After finding the best classifier, load the test data and predict sentiment for them. The data will be exported to CSV file in format containing two columns: Class, Tweets. There are 45000 test samples with known distribution of the sentiment labels to compare between ur results and the results given.

# Conclusion

The increase of microblogging sites like Twitter offers an unparalleled opening to form and employ approaches & technologies that search and mine for sentiments. The work presented in this paper specifies an approach for sentiment analysis on Arabic Twitter data. To unseal the sentiment, we extracted the relevant data from the tweets, added the features.

The overall tweet sentiment was then calculated using a model that presented in this report. This work is exploratory in nature and the prototype evaluated is a preliminary prototype.

The models showed that prediction of text sentiment is a non-trivial task for machine learning. A lot of preprocessing is needed just to be able to run an algorithm. The main problem for sentiment analysis is to craft the machine representation of the text. Alot of additional features were created basing on common sense (length of the words, number of the characters, number of sentences etc). I think that a slight improvement in classification accuracy for the given training dataset could be developed, but since it included highly skewed data (small number of negative cases), the difference will be probably in the order of a few percents. The thing that could possibly enhance classification outcomes will be to add a lot of additional examples (increase training dataset), because given 45275 examples clearly do not include all sequence of words used, further - a lot of emotion-expressing information certainly is missing.

# Future Work

In the classification, I have covered most of the features but emotion expressing information is missing.

Furthermore, it can be useful to analyze the viewpoints of user reviews, whether it is a good or bad review. The above method can be used in hotel, flights reviews to recommend other users about the services. I will try to consolidate the above system into my recommendation system to provide users with trustworthy suggestions regarding the flights and hotels.

# Reference

Myportfolio: <https://github.com/ibrahimmun96>