Twitter Sentiment Analysis

Akshaykumar Kanani, Student, IIT Jodhpur, Anish Anand, Student, IIT Jodhpur,
Jyani Akshay Jagdishbhai, Student, IIT Jodhpur

Abstract: In today's world tweeting has become an important way of communication. So, the sentiment analysis of tweets has become a necessity of today's world. So, in this project we have used a dataset containing 1.6 million tweets, on which we have performed preprocessing, data cleansing, and printed the word cloud. After that, we splitted the dataset into train and test, and made a TF-IDF matrix, then we applied different classifiers to predict the accuracy.

Index Terms: Sentiment Analysis, Data Cleansing, Classifiers, Feature Selection, Cross Validation

----------❖----------

1. Introduction

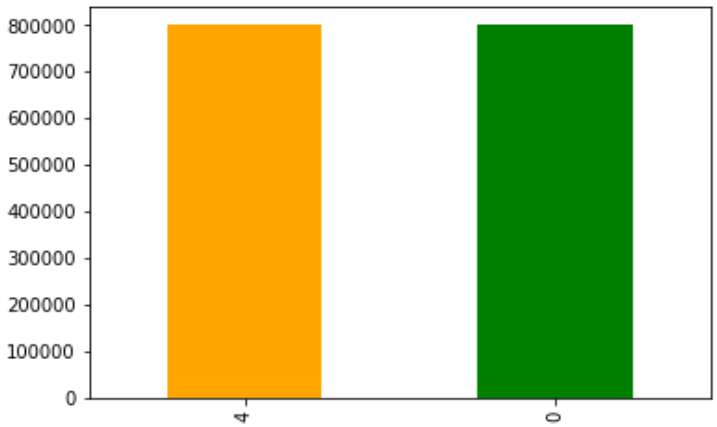
This project has been done as a part of our course of Machine Learning and Pattern Recognition at Indian Institute of Technology Jodhpur, under the supervision of Dr. Richa Singh. In this project we have analysed the mood of a dataset of 1.6 million tweets, applied different classifiers and predicted the accuracies according to them.

2. Working of the Code

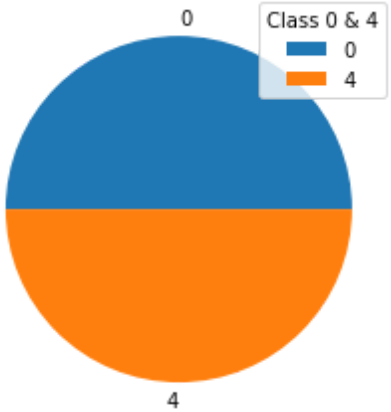
Under this section, we will discuss the concepts involved and the working of the code.

A. Libraries and preprocessing of data

First we have imported all the necessary libraries such as pandas, numpy, matplotlib.pyplot and all the libraries required to run different classifiers and all. After that we have uploaded the dataset containing 1.6 million tweets and printed the first five rows of the dataset and the shape of the dataset which came out to be (1600000, 6) as expected. Then we assigned the names to the different columns of the dataset. After that we have printed the plot for number of tweets and target and got the following plot:

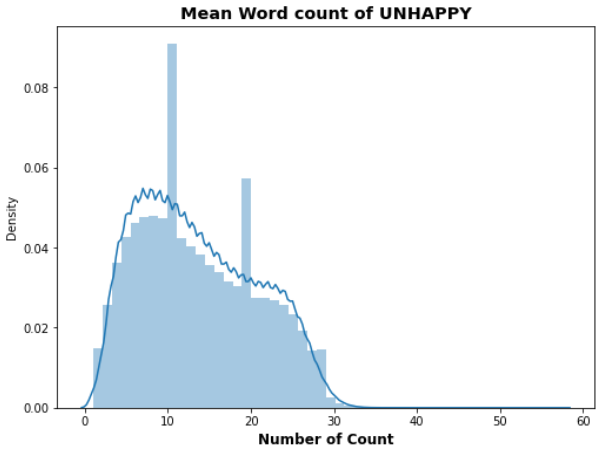


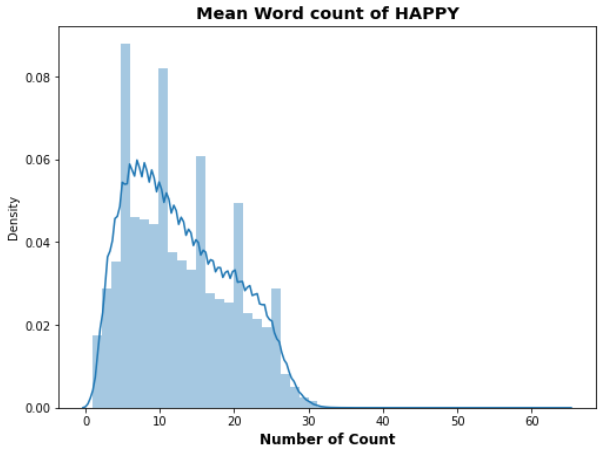
Then we counted the number of tweets in each target and got that we have 80,000 tweets with 4 as target and another 80,000 tweets with target 0, and plotted a pie chart for the same:



B. Finding relation between length of tweets

First we find the average length of the tweets which comes out to be 74.09011125. Then we find the mean of fake class and real class which come out to be 13.58198375 and 12.7703175 respectively, these means are not very much different so we didn't find any relation between length of tweets and class also we can see from the plot that these are almost the same:





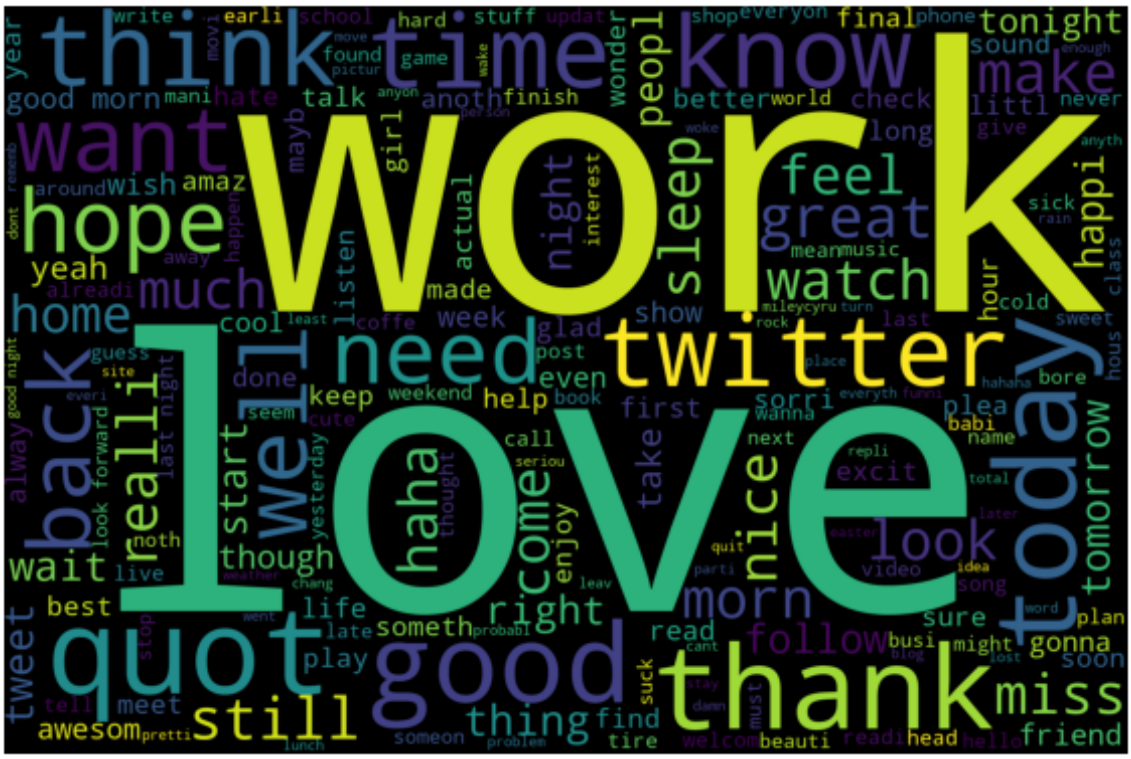
Then we dropped all the columns of the dataset except for the target variable and the tweet text and printed the head of the remaining data. Then we select the first 15000 entries of positive tweets and first 5000 tweets of negative tweets and add them together to form a new dataset named tweetdata on which we will apply all things. We have taken the subset of the given dataset so as to ease the computation. After that we printed the shape and head of the obtained dataset 'tweetdata'.

C. Data Cleaning

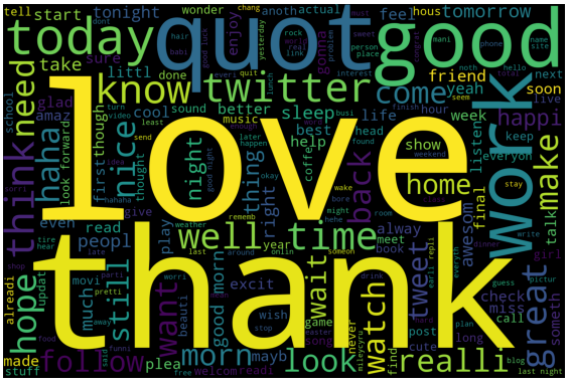
The given dataset of tweets contains many elements which do not have any impact on sentiment analysis, so we are required to clean the dataset for proper analysis of the tweet. First we search for any null value in the target variable and the tweet text column, and find out that there aren't any null values. Then, we try to remove all the stopwords using the nltk library. After that we remove all the handles i.e., starting with @. Then we remove all the links, punctuation and special characters, then we remove the stop words, then we split the words and tokenize it, then applied stemmers, then stitched the back words, and finally we removed all the small words of length less than 3, all the above things, which are removed doesn't affect the sentiment of the tweet, so we done it to improve the accuracy of the result. Then we printed the head of the modified dataset. After that, we installed the spell checker module for correcting the spelling.

D. Word Cloud

Word cloud is an example of Exploratory Data Analysis. It is a way of visualizing, summarizing and interpreting the information that is hidden in rows and column format. In a word cloud the word, appearing the most number of time in the dataset, will appear the biggest in the word cloud image, and the word appearing second most number of time will appear smaller that the first and so on, the bigness will be decided on the basis of frequency of the word in the dataset. First we printed the word cloud for all the tweets in the dataset, which come out to be:

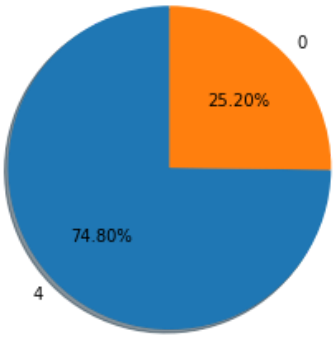


After that we printed word clouds for each target that is for positive sentiments with target value of 4 and for negative sentiment with target value of 0. And the image are as follows:



E. Model Training

Using train_test_split library we split the dataset into test and train, with test size of 0.3 and random state of 42. After that we created a pie chart of train data having values of target as 0 and 4 and got the following pie chart:



After that we printed the first 10 tweets of the train elements.

F. Term Frequency Inverse Document Frequency (TF-IDF)

Term Frequency Inverse Document Frequency (TF-IDF) is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus. It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling. We printed the shape and the head of the TF-IDF matrix. The shape of the TF-IDF matrix comes out to be (14000, 17155).

G. Classifiers

- **Random Forest Classifier:** Random forests are an ensemble learning method for classification, regression and other tasks that operates by constructing a multitude of decision trees at training time and outputting the class that is the mode of the classes (classification) or mean/average prediction (regression) of the individual trees. In this we have used n_estimators=1000 and random_state=42

- **Logistic Regression:** Logistic regression is a statistical model that in its basic form uses a logistic function to model a binary dependent variable, although many more complex extensions exist. In regression analysis, logistic regression is estimating the parameters of a logistic model.

- **Decision Tree Classifier:** Decision Tree Classifier is a simple and widely used classification technique. It applies a straightforward idea to solve the classification problem. Decision Tree Classifier poses a series of carefully crafted questions about the attributes of the test record. Each time it receives an answer, a follow-up question is asked until a conclusion about the class label of the record is reached.

- **Multinomial Naive Bayes:** Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of

email or newspaper article. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

- **MLPClassifier:** MLPClassifier stands for Multi-layer Perceptron classifier which in the name itself connects to a Neural Network. Unlike other classification algorithms such as Support Vectors or Naive Bayes Classifier, MLPClassifier relies on an underlying Neural Network to perform the task of classification. In this we have used random_state=1 and max_iter=200

- **SVM:** SVM or Support Vector Machine is a linear model for classification and regression problems. It can solve linear and non-linear problems and work well for many practical problems. The idea of SVM is simple: The algorithm creates a line or a hyperplane which separates the data into classes.

- **KNN:** K-Nearest Neighbors (KNN) is one of the simplest algorithms used in Machine Learning for regression and classification problems. KNN algorithms use data and classify new data points based on similarity measures (e.g. distance function). Classification is done by a majority vote to its neighbors. In this we have used n_neighbors = 5

- **XGBoost:** XGBoost stands for "Extreme Gradient Boosting". XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements Machine Learning algorithms under the Gradient Boosting framework. It provides a parallel tree boosting to solve many data science problems in a fast and accurate way. In this we have used max_depth=6, n_estimators=1000 and nthread= 3

3. Conclusion

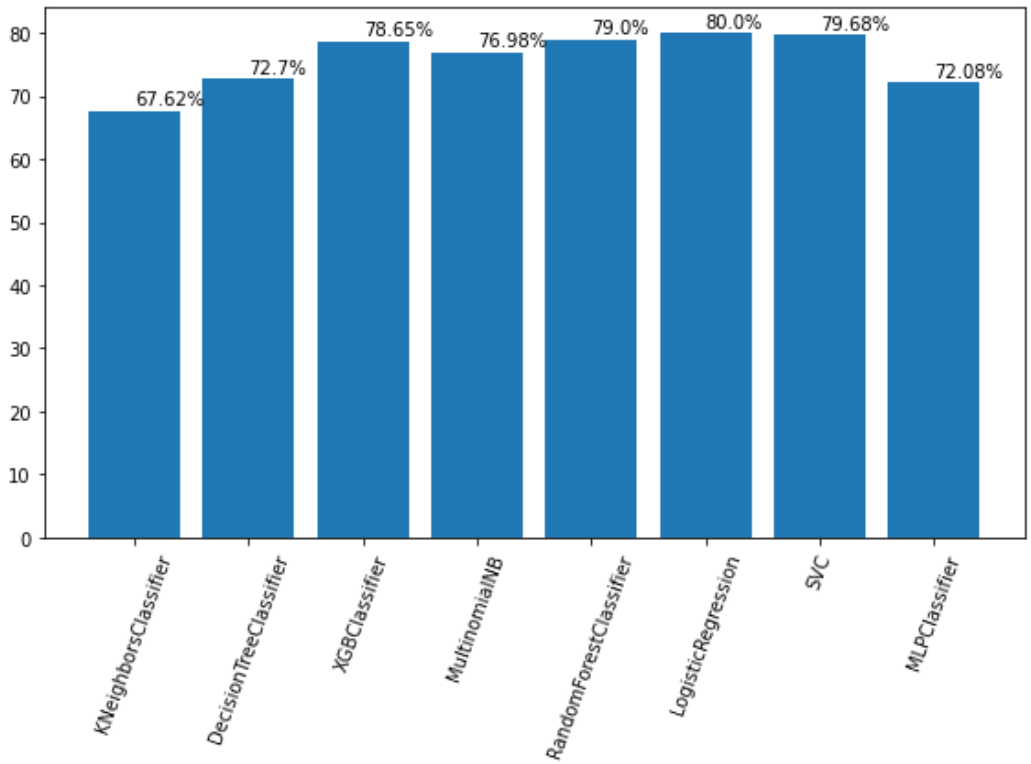
After applying eight different classification algorithms we got the following results as shown in the table:

Classifier	Precision		Recall		F1 Score		Accuracy
	Target 0	Target 4	Target 0	Target 4	Target 0	Target 4	
KNN	0.28	0.80	0.32	0.78	0.30	0.79	67.62
Decision Tree	0.45	0.82	0.44	0.82	0.45	0.82	72.7
XGBoost	0.35	0.93	0.61	0.81	0.45	0.87	78.65

Multinomial Naive Bayes	0.07	1.00	0.88	0.77	0.13	0.87	76.98
Random Forest	0.39	0.92	0.61	0.82	0.48	0.87	79.0
Logistic Regression	0.33	0.95	0.69	0.81	0.45	0.88	80.0
SVC	0.30	0.96	0.70	0.81	0.42	0.88	79.68
MLP Classifier	0.55	0.78	0.44	0.84	0.49	0.81	72.08

	KNN	**Decision Tree**	**XG Boost**	**Multinomial Naive Bayes**	**SVC**	**MLP**	**Random Forest**	**Logistic Reg.**
Weighted Mean of F1 Score	0.68	0.73	0.81	0.85	0.83	0.71	0.81	0.83

Then we plotted the graph of accuracies of different classifiers used:



From the above table and plot we can see that Logistic Regression has the highest accuracy score of 80.0% and K Nearest Neighbour classifier has the least accuracy score of 67.62%. And the mean accuracy score of all the classifiers used is 75.84%.

Acknowledgement

We would like to express our gratitude to our course instructor Dr Richa Singh for her immense guidance and support throughout the course of Pattern Recognition and Machine Learning. We would also like to thank our lab instructors Dr Romi Banerjee and Dr Yashaswi Verma for their guidance and support throughout the lab sessions of the course and any other time. We would also like to express our gratitude towards all the teaching assistants involved in this course for helping us whenever we needed. And last but not the least, we would like to thank our batchmates for their immense help and support. Without their support and guidance this project would not have been completed.

References

- https://www.kaggle.com/kazanova/sentiment140
- https://www.kaggle.com/c/twitter-sentiment-analysis2
- https://www.analyticsvidhya.com/blog/2018/07/hands-on-sentiment-analysis-dataset-python/
- https://towardsdatascience.com/twitter-sentiment-analysis-classification-using-nltk-python-fa912578614c
- https://www.geeksforgeeks.org/twitter-sentiment-analysis-using-p

Contribution of Each Member

Akshaykumar Kanani (B19EE008) has written the code for Data cleaning[remove handle,remove links,remove punctuations and special characters,remove stop words,split text and tokenize,apply stemmer,stitch back words,remove small words,and spelling checking], KNN, Decision Tree Classifier, XGBoost and Multinomial Naive Bayes classification

Anish Anand (B19EE010) has pre-processed the data and, tried to find the relation between length of the target classes,and class and plot the graph of relations, created the word cloud for positive and negative target classes. and prepared the report for the project and created a Readme file for the same.

Jyani Akshay Jagdishbhai (B19EE041) has written the code for model training and testing, TF-IDF(Term Documentary), Random Forest Classifier, Logistic Regression, SVM, MLPclassification using best parameters, also try to improve the accuracy of models by running them with better parameters or by applying the dimensionality reduction technique and getting the final result.And also use the feature selection method(PCA) to get the result in less time.

Code debugging and finding the references were done by all the members of the team.