MLND Capstone

Rakshit Pratap Singh

Definition

Project Overview

Sentiment Analysis is a fundamental task in Natural Language Processing (NLP). Its uses are many: from analysing political sentiment on social media , gathering insight from user-generated product reviews or even for financial purposes, such as developing trading strategies based on market sentiment . The goal of most sentiment classification tasks is to identify the overall sentiment polarity of the documents in question, i.e. is the sentiment of the document positive or negative.

Sentiment analysis is critical because it helps us to see what customers like and dislike about a particular brand.
Customer feedback—from social media, websites , call center agents, or any other source—contains a treasure trove of useful business information. But, it isn't enough to know what customers are talking about. We must also know how they feel. Sentiment analysis is one way to uncover those feelings.

Sometimes known as "opinion mining," sentiment analysis can let us know if there has been a change in public opinion toward any aspect of our business. Peaks or valleys in sentiment scores give you a place to start if we want to make product improvements, train sales or customer care agents, or create new marketing campaigns.

Sentiment analysis is not a once and done effort. By reviewing customer's feedback on a business regularly you can be more proactive regarding the changing dynamics in the market place.

In this project we are doing the sentiment analysis of amazon fine food review dataset. Our task is to label the reviews as either positive or negative.

Related academic research and academic work:-

https://gist.github.com/abhigrover101/dff3ebd06a0c30c7155f

http://aics2017.dit.ie/papers/AICS2017_paper_21.pdf

Problem Statement

Our goal is to correctly label the amazon fine food reviews as either positive or negative. This is a supervised learning task so we will be employing various supervised learning techniques.First we will draw the insights from the amazon fine food reviews dataset and extracting the features from it so that it can be fed to the supervied learner classifier then we will select the algorithm which gives the highest metric score.

Metrics

Evaluation is done by Confusion Metrix which is a table that is often used to describe the performance of a classification model (or "classifier") on a set of test data for which the true values are known.

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

It is often convenient to combine precision and recall into a single metric called the F1 score, in particular if you need a simple way to compare two classifiers. The F1 score is the harmonic mean of precision and recall.

```
F1 = 2 * (precision * recall) / (precision + recall)
```

In the multi-class and multi-label case, this is the weighted average of the F1 score of each class.

In sports and daily activity dataset labels are uniformly distributed and f1_score is best to use when our class labels are uniformly distributed.

The **accuracy_score** function computes the accuracy, either the fraction (default) or the count (normalize=False) of correct predictions.

Analysis

Data Exploration

For this project, the dataset is obtained from Kaggle (https://www.kaggle.com/snap/amazon-fine-food-reviews/data).
The Amazon Fine Food Reviews dataset is approx. ~300 MB large dataset which consists of around 568k reviews about amazon food products written by reviewers between 1999 and 2012. Each review has the following 10 features:

- Id
- ProductId - unique identifier for the product
- UserId - unqiue identifier for the user
- ProfileName
- HelpfulnessNumerator - number of users who found the review helpful
- HelpfulnessDenominator - number of users who indicated whether they found the review helpful
- Score - rating between 1 and 5
- Time - timestamp for the review
- Summary - brief summary of the review
- Text - text of the review

Here, for each review we have score rating between 1 and 5.for simplification we will label the review as negative if it has a score of 1-3 and positive for a score of 4 or 5. There are many features that are provided in the dataset. However we will use 'Text' as an input and 'Score' as an output

Exploratory visualization

- First I plotted the bar graph to see the frequency of each rating score in a dataset.

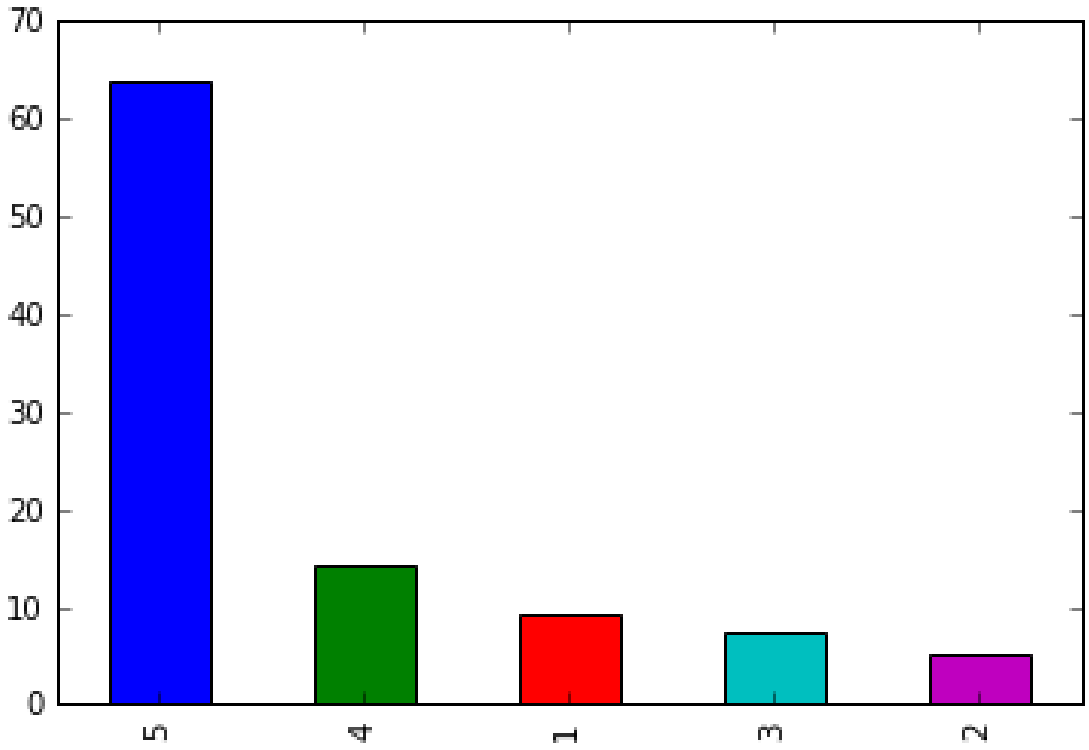

Figure1. Frequency of each rating

We can clearly see from the plotted graph that 5 rating has the highest frequency followed by 4,1,3 and 2.
Percentage of occurrence of each rating is depicted below:-

```
In [12]: print rating_count_perentage
5    63.878871
4    14.188483
1     9.194763
3     7.501047
2     5.236835
Name: Score, dtype: float64
```

- Then I added a new column to our dataset named “Helpfulness” which gives insight on the usefulness or helpfulness of the review.
  I divided the reviews in 4 categories as:-
  1. Not even voted
  2. 0 - 25%helpful
  3. 25 - 75% helpful
  4. 75 - 100% helpful

  Then I plotted various bar plots to extract the insights

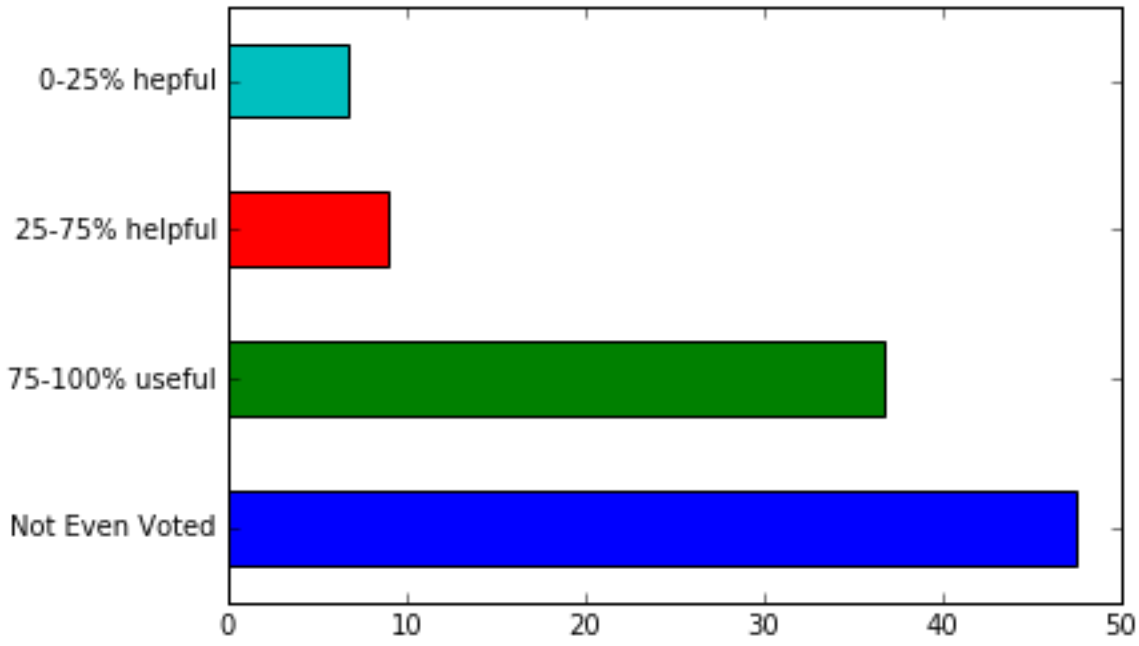

Figure2. Percentage of usefulness of ratings

From the above plot we conclude that there are almost 47% reviews that are not even voted, approx. 36% reviews are 75-100%useful ,9% reviews are 25 – 75%useful and 6.7% reviews are least useful.

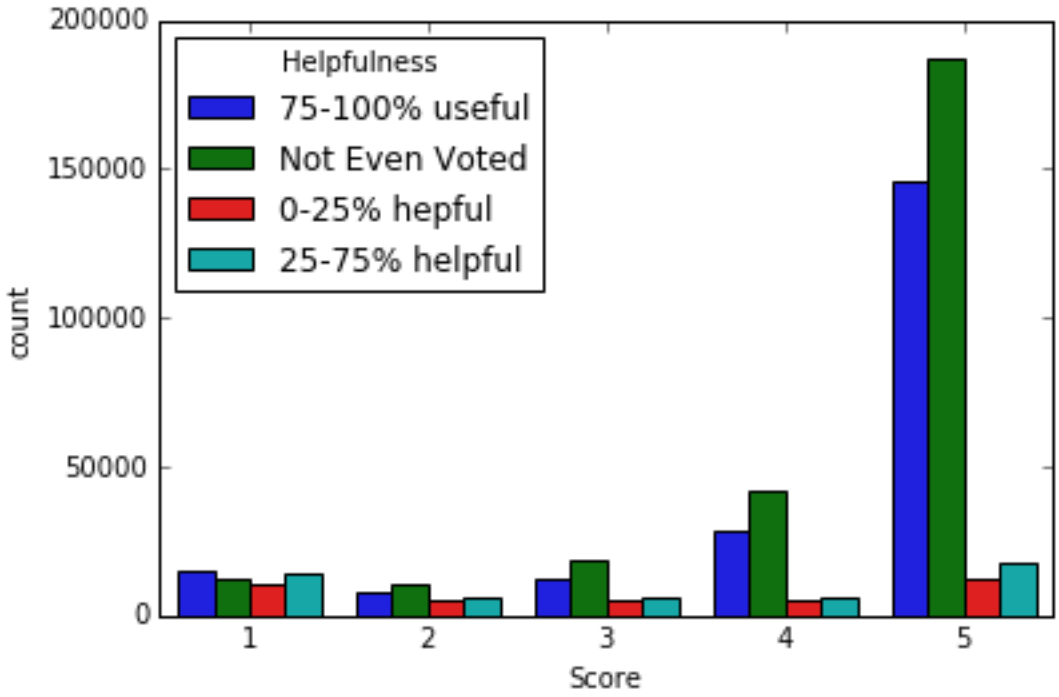

Figure3. Counting of each rating on the basis of their helpfulness

From the above graph it is clear that 5 rating reviews are more helpful to users followed by 4,3,2 and 1. 1 nad 2 rating are least helpful to users.

- Then I calculated the length of each review and plotted a box plot according to the sentiment of review(Positive or Negative).

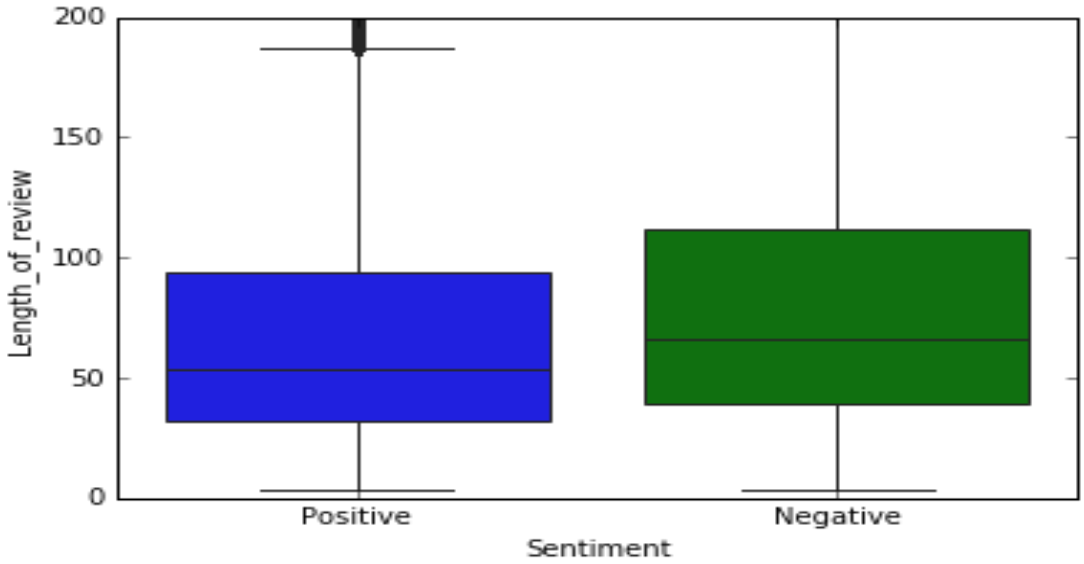

Figure4. Average length for positive and negative sentiment reviews

Here we can see that the positive reviews are shorter in length than the negative reviews as average median from the box plot of positive reviews is less than that of negative reviews.

- Then I plotted the word cloud for the positive and negative reviews.

For negative reviews

Algorithm and Results

Supervised algorithms used for the classification of the review are as follows:-

1.Multinomial Naïve Bayes:- Multinomial Naive Bayes is a specialized version of Naive Bayes that is designed more for text documents. Whereas simple naive Bayes would model a document as the presence and absence of particular words, multinomial naive bayes explicitly models the word counts and adjusts the underlying calculations to deal with in. It estimates the conditional probability of a particular word given a class as the relative frequency of term t in documents belonging to class(c). The variation takes into account the number of occurrences of term t in training documents from class (c),including multiple occurrences.

2. Logistic Regression:-
Logistic regression falls under the category of supervised learning; it measures the relationship between the categorical dependent variable and one or more independent variables by estimating probabilities using a logistic/sigmoid function. In spite of the name '*logistic regression*', this is not used for regression problem where the task is to predict the real-valued output. It is a classification problem which is used to predict a binary outcome (1/0, -1/1, True/False) given a set of independent variables.

3. RandomForestClassifier:-

In random forests, each tree in the ensemble is built from a sample drawn with replacement (i.e., a bootstrap sample) from the training set. In addition, when splitting a node during the construction of the tree, the split that is chosen is no longer the best split among all features. Instead, the split that is picked is the best split among a random subset of the features.As a result of this randomness, the bias of the forest usually slightly increases (with respect to the bias of a single non-random tree) but, due to averaging, its variance also decreases, usually more than compensating for the increase in bias, hence yielding an overall better model.

4. SGD Classifier:-

Stochastic Gradient Descent (SGD) is a simple yet very efficient approach to discriminative learning of linear classifiers under convex loss functions such as (linear) Support Vector Machines and Logistic Regression. Even though SGD has been around in the machine learning community for a long time, it has received a considerable amount of attention just recently in the context of large-scale learning.

SGD has been successfully applied to large-scale and sparse machine learning problems often encountered in text classification and natural language processing. Given that the data is sparse, the classifiers in this module easily scale to problems with more than 10^5 training examples and more than 10^5 features.

5. MLP Classifer(Multi-layer perceptron):-

Multi-layer Perceptron (MLP) is a supervised learning algorithm that learns a function $f(\cdot) : R^m \rightarrow R^o$ by training on a dataset, where m is the number of dimensions for input and o is the number of dimensions for output. Given a set of features $X = x_1, x_2, ..., x_m$ and a target y, it can learn a non-linear function approximator for either classification or regression. It is different from logistic regression, in that between the input and the output layer, there can be one or more non-linear layers, called hidden layers. Figure 1 shows a one hidden layer MLP with scalar output.

The leftmost layer, known as the input layer, consists of a set of neurons $\{x_i | x_1, x_2, ..., x_m\}$ representing the input features. Each neuron in the hidden layer transforms the values from the previous layer with a weighted linear summation $w_1x_1 + w_2x_2 + ... + w_mx_m$, followed by a non-linear activation function $g(\cdot) : R \rightarrow R$ - like the hyperbolic tan function. The output layer receives the values from the last hidden layer and transforms them into output values.

The module contains the public attributes `coefs_` and `intercepts_`. `coefs_` is a list of weight matrices, where weight matrix at index i represents the weights

between layer i and layer $i + 1$. intercepts_ is a list of bias vectors, where the vector at index i represents the bias values added to layer $i + 1$.

Benchmark

Benchmarking of our model is done by comparing our results by other kagglers. Kaggler ‘’amandeep” is able to get an accuracy of 89% for naïve bayes and 92% for logistic regression. A link for his notebook is:- https://www.kaggle.com/amanai/amazon-fine-food-review-sentiment-analysis

Methodology

Data Preprocessing

We can not feed the raw text data directly to any classification model.we need to convert the raw text data in to some sort of numbers so that they can be used as an input or output features for the classification model.
Further we need to clean our text data as it will reduce the size of the text and removes the features that are unnecessary and does not contain any valuable information.

All the steps of text cleaning and feature extraction from text are done on the feature “Text” from the dataset which contains the reviews of users in textual form and this is our input feature.. The steps are as follows:-

1. Tokenization:- Tokenization is defined as the process of splitting the text into smaller parts called tokens.I used the NLTK library in sckit – learn to tokenized the text.

2. Then I remove the punctuations from text as they don't contain any valuable information and also convert the text in to lower case.

3. Stop words filtration:- "Stop words" usually refers to the most common words in a language. Like 'the', 'is', 'a'.these words contain very little information.so including these words in our text will not prove to be useful. I used the NLTK corpus of stop words for the filteration of these words from the text.

4. Stemming:- Stemming is a technique to remove affixes from a word, ending up with the stem.It is a process of reducing a word to its root form. For example, the stem of tasting is taste, so stemming algorithm recognizes the ing suffix and removes it.Stemming makes our training data more dense and reduces the size of number of words used in the text. I used the PorterStemmer algorithm in NLTK for this purpose.

Implementation

- After all the text cleaning steps we get the clean data of customers reviews which is our input feature. For the output feature we have labeled the 'score' feature from dataset as positive if score is greater than 3 and negative otherwise.
- Then we split the data into training and testing sets. But still our input and output feature is in textual form so we need to convert them into numbers. For our output we have simply assigned 1 to the scores greater then 3 and 0 otherwise but our input feature is a long text so to vectorize it we have used the TF-IDF vectorizer.

TF-IDF vectorizer :- TFIDF, short for term frequency–inverse document frequency, is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.[1] It is often used as a weighting factor in searches of information retrieval, text mining, and user modeling. The tf-idf value increases proportionally to the number of times a word appears in the document and is offset by the frequency of the word in the corpus, which helps to adjust for the fact that some words appear more frequently in general.

The tf–idf is the product of two statistics, term frequency and inverse document frequency.

Term frequency is the number of times the term occurs in document.

The inverse document frequency is a measure of how much information the word provides, that is, whether the term is common or rare across all documents. It is the logarithmically scaled inverse fraction of the documents that contain the word, obtained by dividing the total number of documents by the number of documents containing the term, and then taking the logarithm of that quotient.

- After vectorizing the input feature now we can finally feed into supervised learning models.

Various supervised algorihms as state abvove in Algorithm and techniques sections were used. Accuracy and f1 score is calculated for each algorithms prediction.It is stated below in a table:-

Table 1

Algorithms	F1_Score	Accuracy
Multinomial NB	0.897223555443	0.822199079612
Random Forest classifier	0.936170766854	0.896519695456
SGD classifier	0.920983773681	0.86910508465
Logistic Regression	0.932163819825	0.891432230463

Results

Model evaluation and validation

Since the model is evaluated on the basis of accuracy and f1_score so the model with highest accuracy and f1_score will be choosen.For our case MLP classifier has got highest accuracy and f1_score.so I choose the MLP classifier for this project. In order to validate the model i.e to see if the model is able to generalize well on the unseen data, i choose to plot the ROC curve. In a Receiver Operating Characteristic (ROC) curve the true positive rate (Sensitivity) is plotted in function of the false positive rate (100-Specificity) for different cut-off points. A test with perfect discrimination (no overlap in the two distributions) has a ROC curve that passes through the upper left corner (100% sensitivity, 100% specificity). Therefore the closer the ROC curve is to the upper left corner, the higher the overall accuracy of the test. ROC curve plotted for the MLP classifier that we have used is :-

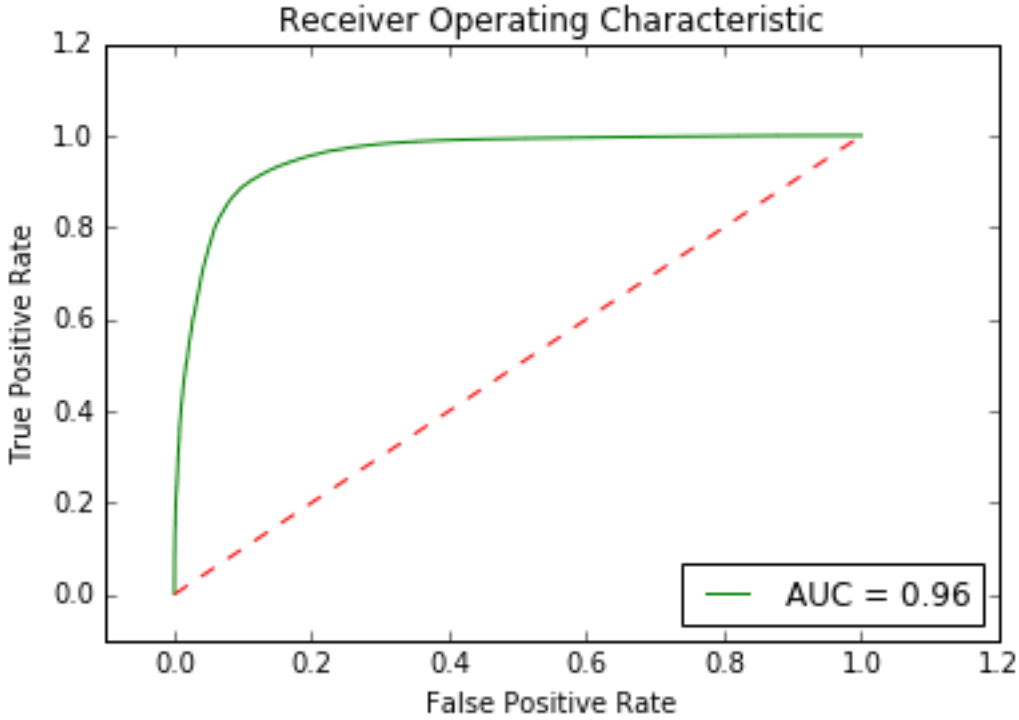

Figure5. ROC curve for MLP classifier

We can observe from the above plotted ROC curve that the area under the curve (AUC) is equal to 0.96. since we know that maximum AUC score is equal to 1, we can conclude that our model generalizes well on the unseen data and it has done a pretty decent job in the prediction of unseen data(reviews).Hence, we can say the final model and solution is robust enough to be trust and can be used by others needed to solve the same type of problem.

Justification

According to benchmark model best results achieved is 92.73% of accuracy by using logistic Regression. I have got the best of accuracy 92.13% using MLP Classifier. Better results can be achieved using more hidden layers and parameter tuning.

Conclusion

A decent job is being done in predicting the sentiment of reviews as the algorithm has got a good f1_score and accuracy.

Free Form Visualization

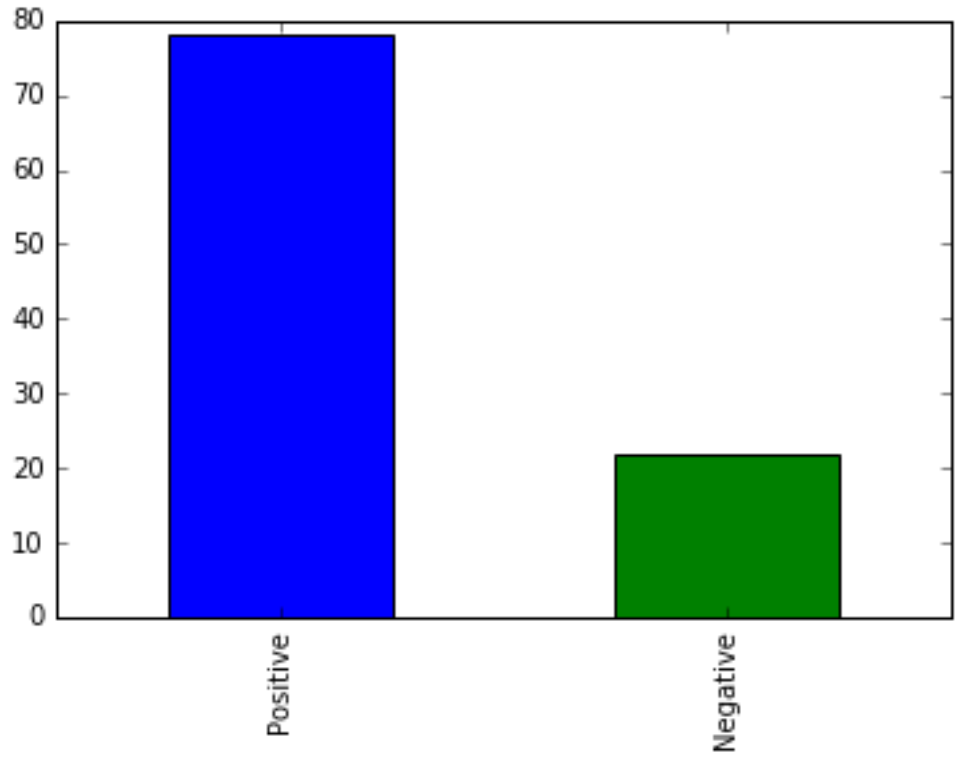

Figure6. Percentage of positive and negative reviews in dataset

Here we can see that positive sentiment in data occurs almost 78% of the time while negative sentiment occurs in data only 22% of the time. So our dataset is highly skewed towards positive sentiment. Moreover if we create naïve predictor which only gives positive prediction then its accuracy will be 78% so I think we should use f1_score metric in order to evaluate the performance of the model.

Reflection

Overall it was a very good experience for me to work on this project and we got a model that is good enough to predict the sentiment of reviews. The most interesting and challenging part for me was the exploratory visualization as I got the interesting and valuable insights from the dataset and I think we can explore

the data further to generate more useful insights. Cleaning the data was also the challenge and for me it was a very time consuming in terms of computation. Training the MLP classifier was also very expensive in terms of time consuming and computation, it took approx 2 to 3 hours(without the use of GPU) to fully train the classifier.

Improvement

Better results can be achieved by using more hidden layers and more complex architecture in MLP classifier.

RandomForestClassifier has also done a good job as it got a f1_score = 93.6 and accuracy = 89.6%. Using GridSearchCV could increase the results.