**Sentiment Analysis of Movie review dataset**

**Project overview:**

The project domains background around the area of Sentiment analysis. Sentiment analysis or Opinion mining is a significant task in the field of Natural Language Processing also in machine learning and Data science. It is used to understand the sentiment in social media, in political analysis and in survey responses. In general the main aim of this is to determine the attitude of speaker with positive, neutral and negative polarity.

**Problem Statement:**

The project that the proposal infers to is called “Movie Review Sentiment Analysis”. The main goal is to classify the sentiment of reviews from the ***Rotten Tomatoes dataset.*** The Rotten Tomatoes movie review dataset is a corpus of movie reviews used for sentiment analysis. The main task is to label phrases on a scale of five values: negative, somewhat negative, neutral, somewhat positive, positive. There are many obstacles such as sentence negation, sarcasm, language ambiguity, and many others make the sentiment prediction more difficult. In general, this particular Sentiment Analysis is a multiclass classification task to be faced.

**Data exploration:**

The dataset contains tab-separated files with phrases from the Rotten Tomatoes dataset. The train/test split has been preserved to benchmark, but the sentences have been shuffled from their original order. Each Sentence has been parsed into many phrases by the Stanford parser. Each phrase has a PhraseId. Each sentence has a SentenceId. Phrases that are repeated (such as short/common words) are only included once in the data.

**The Train Set has 4 columns and 156060 rows of data. Its features are the following:**

1. **PhraseId**, is a unique Phrase identifier per phrase. Multiple phrases originate from the same sentence and its data type is “numeric”. We have 156060 unique PhraseIds in the entire train set.

2. **SentenceId**, is a unique sentence. In the trainset we have 8543 unique Sentences in the train dataset.

3. **Phrase**, it is type of “string” and it stems from the Sentence that is referenced by SentenceId. In total they are 156060 unique Phrases and each phrase is the result from a unique split to the Sentence that belongs to.

4. **Sentiment**, it is the Sentiment Labels and the target feature that must be predicted in the Test Set. Its labels are the following: 0 – negative, 1 - somewhat negative, 2 – neutral, 3 - somewhat positive, 4 – positive.

**The Test Set has 3 columns and they are the following:**

1. **PhraseId**, is a unique Phrase identifier per phrase. Multiple phrases originate from the same sentence and its datatype is “numeric”. We have 66292 unique PhraseIds in the test set.

2. **SentenceId**, is a unique Sentence. In the trainset we have 3310 unique Sentences/reviews in the test set.

3. **Phrase**, it is type of “string” and it stems from the Sentence that is referenced by SentenceId. In total they are 156060 unique Phrases in the test set and each phrase is the result from a unique split to the Sentence that belongs to.

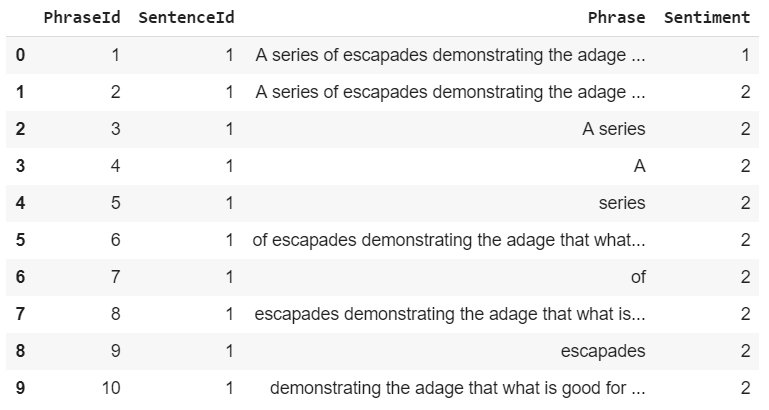
**Independent variables:**

* PhraseId
* Sentence Id
* Phrase

**Dependent variables:**

* Sentiment

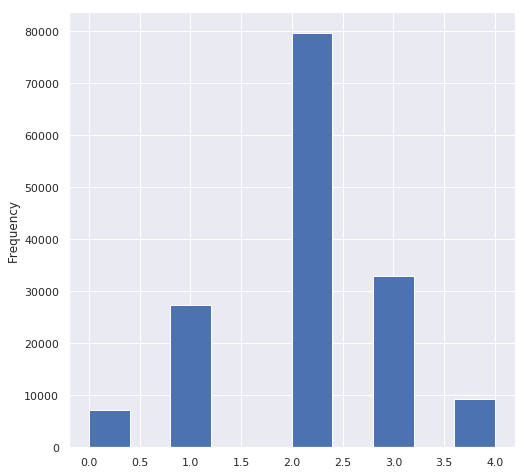
**The following figure demonstrates how the cases look like from the train set:**



**Fig:** Observations of train data

During the training of the Machine Learning models the **PhraseId** and **SentenceId** will not be used as they are just Id incremental numbers and they do not have any predictive ability during training. However, the **Phrases** will definitely be used to train to predict sentiment during the project.

Furthermore, the dataset is unbalanced, which means that the train set does not provide almost equal number of cases for all the different types of sentiment that must be predicted. The following figure depicts the distribution of the sentiment at the train set:



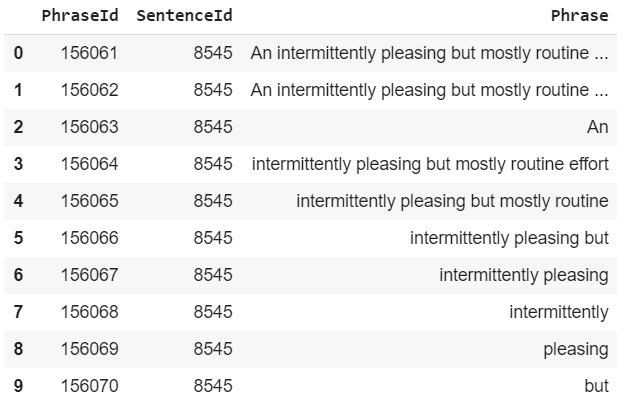
**Fig:** Sentiment distribution from train data

|  |  |
| --- | --- |
| **sentiment** | **count** |
| 1. negative | 7072 |
| 1. somewhat negative | 27273 |
| 1. Neutral | 79582 |
| 1. somewhat positive | 32927 |
| 1. positive | 9206 |

**Table:** Table shows sentiment distribution in train data

It shows that the sentiment “2 - Neutral” is the dominant one. Having an unbalanced dataset may lead us to classifiers that cannot identify and classify cases that belong to positive or negative sentiments and they may have a chance to misclassify them.

**The following figure demonstrates how the cases look like from the test set:**



**Fig:** Observation of test data

**Methodology:**

**Exploratory Data Analysis:**

**1. Exploratory Data Analysis**

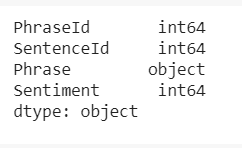
Data exploration is a crucial stage for any predictive model. The quality of the input decides the quality of the output and if there is any outliers or anomalies it affects the output of model.

**Train Data:** The predictive model is always built on train data set. An intuitive way to identify the train data is, that it always has the target variable included. In this the model will be trained and it extracts the features from the dataset.

**Test Data:** Once the model is built, it’s accuracy is tested on test data. This data always contains less number of observations than train data set. Also, it does not include target variable.

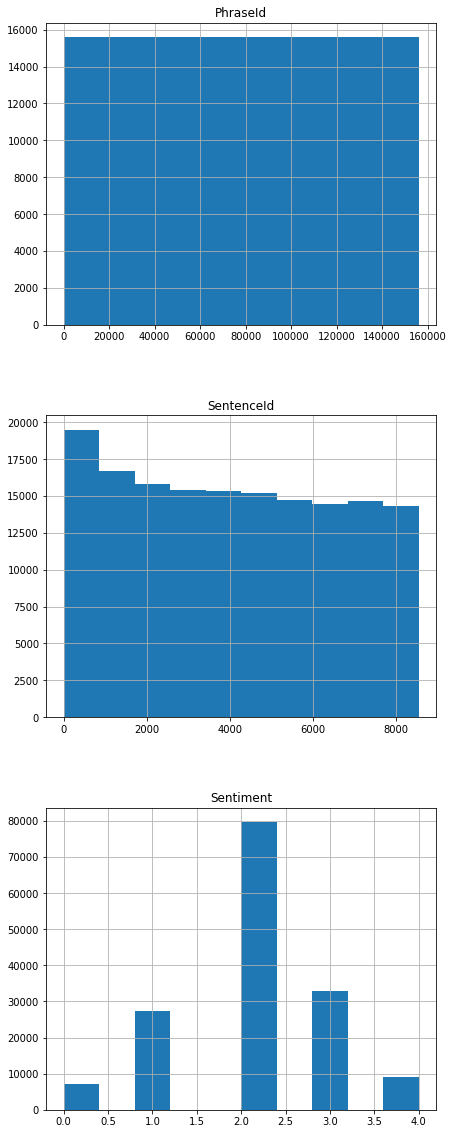
* 1. **. Variable type Identification**

If we see the structure of the data set then we can observe there are different kinds of datatypes of variable some of them are numerical and some are object. So we have to treat different kinds of variables in different ways and we can decide which variables are required for training the model.



**1.2. Distribution of numerical variables:**

Here we visualize the numerical variables present in our train dataset. Although Sentiment is categorized as numerical, it has only five unique values (0, 1, 2, 3, 4). So we can treat it as categorical variable. Now, I’ll test and plot a histogram for each numerical variables and analyze the distribution.



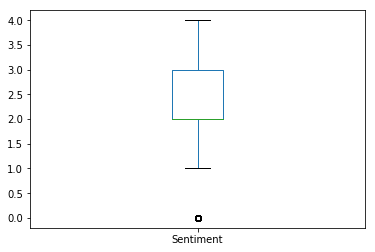
**Fig:** Distribution of numeric variables

By looking at this histograms we can tell that PhraseId and SentenceId are not required for our analysis as it does not provide any precitive modelling. So, we use only Phrase and Sentiment for our modelling.

**1.3. Outlier analysis:**

Outliers are extreme values that deviate from other observations on data , they may indicate a variability in a measurement, experimental errors or a novelty. In other words, an outlier is an observation that diverges from an overall pattern on a sample. If we don’t treat them properly then it will affect our overall predictions.

We have used boxplot method to visualize the outliers and find the missing values. Let’s have a look of outlier effects on Sentiment variable.



**Fig:** Outlier analysis of Sentiment variable

From this visualisation we cannot find any outlier from our Sentiment variable and we can use it for our model building without any replacing of missing values.

**1.4. Data Preprocessing:**

The dataset of this rotten tomatoes turned to have some unique features. we have only phrases as data. And a phrase can contain a single word. And one punctuation mark can change the meaning and cause phrase to receive a different sentiment. Also assigned sentiments can be strange. This means several things:

* using stopwords can be a bad idea, especially when phrases contain one single stopword.
* puntuation could be important, so it should be used.
* ngrams are necessary to get the most info from data.

As you can see sentence id denotes a single review with the phrase column having the entire review text as an input instance followed by random suffixes of the same sentence to form multiple phrases with subsequent phrase ids. This repeats for every single new sentence id (or new review per se). The sentiment is coded with 5 values 0= Very negative to 4=Very positive and everything else in between.

The workflow will show the complete building of model:

1. Prepare Problem

* Load required libraries
* Load train and test dataset

2. Summarize Data

* Descriptive statistics
* Data visualizations

3. Prepare Data

* Recognising anomalies
* Data Transforms

4. Evaluate Algorithms

* Split-out validation dataset
* Test options and evaluation metric
* Spot Check Algorithms
* Compare Algorithms

5. Improve Accuracy

* Algorithm Tuning
* Ensembles

6. Finalize Model

* Predictions on the validation dataset
* Create a standalone model on the entire training dataset

**Results:**

**Model Evaluation and Validation:**

The trainset was split in the ratio 80:20 train and validation set respectively. In every execution the textual data was transformed in TF – IDF matrix.

**Machine Learning models and TF – IDF as feature extraction:**

TF - IDF is an information retrieval technique that weighs a term’s frequency (TF) and its inverse document frequency (IDF).

The trainset and the test set were converted via the TF – IDF vectorizer from sklearn. we can see that due to the fact the ngram\_range is from 1 to 3 the columns of the TF - IDF matrix vectorizer is extremely huge. After applying TF-IDF vectorizer the shape of train set is increased to 301627. This may lead us to slow down the Machine Learning models to fit the data.

For all the ML models the random state will be set to 42 in order to the models be reproducable and create the same results in every run.

**Algorithms and Techniques:**

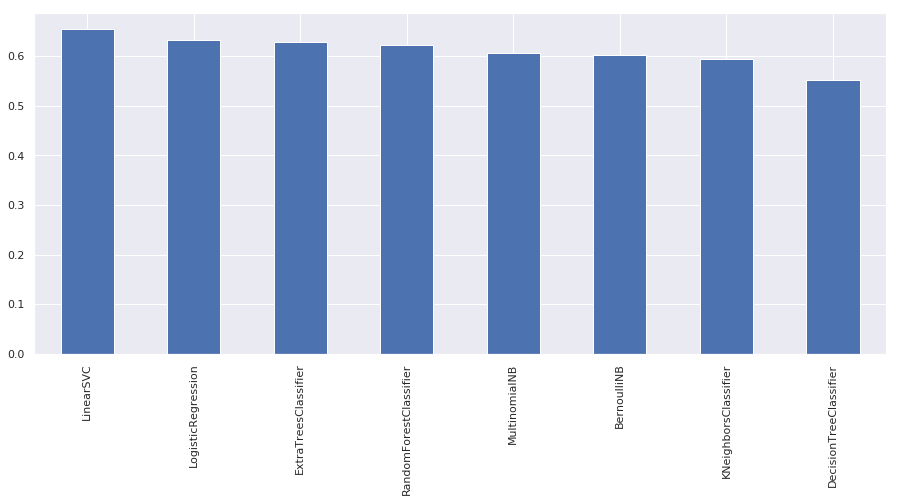
The machine learning techniques that are used are:

* Logistic Regression
* Decision Tree Classifier
* Extra Tree Classifier
* Random Forest Classifier
* Linear SVC
* Bernoulli NB
* Multinomial NB
* K Neighbours Classifier

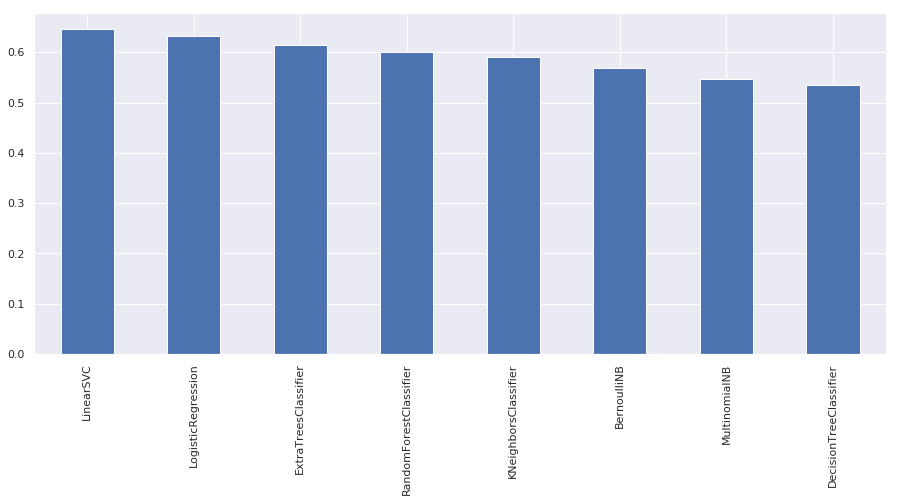
The performance results of accuracy and F1 score over validation set are:

|  |  |  |
| --- | --- | --- |
| **Model** | **Accuracy** | **F1 score** |
| Linear SVC | 0.654 | 0.645 |
| Logistic Regression | 0.631 | 0.631 |
| Extra Tree Classifier | 0.627 | 0.615 |
| Random Forest Classifier | 0.621 | 0.6 |
| Multinomial NB | 0.606 | 0.546 |
| Bernoulli NB | 0.602 | 0.568 |
| K Neighbours Classifier | 0.593 | 0.591 |
| Decision Tree Classifier | 0.551 | 0.535 |

**Visualising** **ML Classifiers based on their accuracy with TF - IDF as Feature Extraction:**

****

**Visualising** **ML Classifiers based on their F1-score with TF - IDF as Feature Extraction**

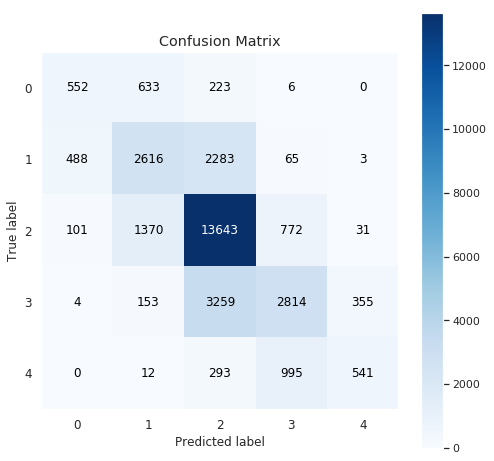


Based on the bar plots above the top performed ML models based on both accuracy and F1-score are the following:

* LinearSVC (SVM Classifier)
* Logistic Regression Classifier
* ExtraTreesClassifier
* RandomForestClassfier

They perform very well in both accuracy and F1-score. So I will move on my ensembling their predictions using the statistical mode to judge what class should be used for my ensemble prediction in my validation set and it results in accuracy of 0.65.

**Confusion matrix of ensembling top 4 models over validation test:**



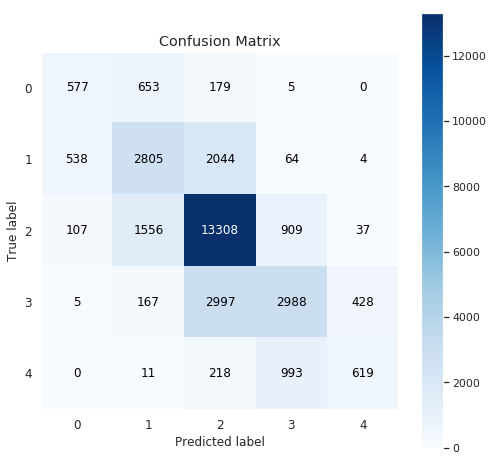
**Tuning the models:**

Since the top 4 Machine Learning models performed really well and evaluated in both accuracy and F1-score metric using their default parameters as they are, the next step is to tune them in the same train - test split and see if they perform even better. After tuning and ensemble the above top performed ML models the performance results over the Validation and Test Set are the following:

|  |  |  |
| --- | --- | --- |
| **Model** | **Accuracy** | **F1 score** |
| Linear SVC | 0.654 | 0.645 |
| Logistic Regression | 0.658 | 0.648 |
| Extra Tree Classifier | 0.628 | 0.616 |
| Random Forest Classifier | 0.625 | 0.604 |

Even ensembling the tuned predictions over the validation set using the statistical mode, we can not overcome and improve the accuracy over 0.65 and the same happens to F1-score.

**Confusion matrix of ensembling top 4 models after tuning the model:**



**Conclusion**:

After tuning the top 4 models we saw only minor improvements from the accuracy and F1 score and we consider the top models that performed well among 8 techniques and produced predictions with Linear SVC, Logistic Regression, Extra trees and Random forest classifiers over test data.