

Review prediction CLASSIFIER

Model



Submitted by:

Umesh Kumar Gupta

Acknowledgement

Firstly, I would like to express my indebtedness appreciation to my SME Khushboo Garg and data trained for giving me this opportunity. Their constant guidance and advice played a vital role in making the execution of the project. Data Trained gave me useful suggestions that were crucial in making this report as flawless as possible. I’d also like to show my gratitude to live Data trained support.

Introduction:

We have a client who has a website where people write different reviews for technical products. Now they are adding a new feature to their website i.e. The reviewer will have to add stars(rating) as well with the review. The rating is out 5 stars and it only has 5 options available 1 star, 2 stars, 3 stars, 4 stars, 5 stars. Now they want to predict ratings for the reviews which were written in the past and they don’t have rating. So, we have to build a model which can predict the rating by seeing the review.

Objective:

Nowadays, a massive number of reviews is available online. Besides offering a valuable source of information, these informational contents generated by users, also called User Generated Contents (UGC) strongly impact the purchase decision of customers. As a matter of fact, a recent survey (Hinckley, 2015) revealed that 67.7% of consumers are effectively influenced by online reviews when making their purchase decisions. More precisely, 54.7% recognized that these reviews were either fairly, very or absolutely important in their purchase decision making. Relying on online reviews has thus become a second nature for consumers.

Review of Literature:

The rapid development of Web 2.0 and e-commerce has led to a proliferation in the number of online user reviews. Online reviews contain a wealth of sentiment information that is important for many decision-making processes, such as personal consumption decisions, commodity quality monitoring, and social opinion mining. Mining the sentiment and opinions that are contained in online reviews has become an important topic in natural language processing, machine learning, and Web mining.

Motivation:

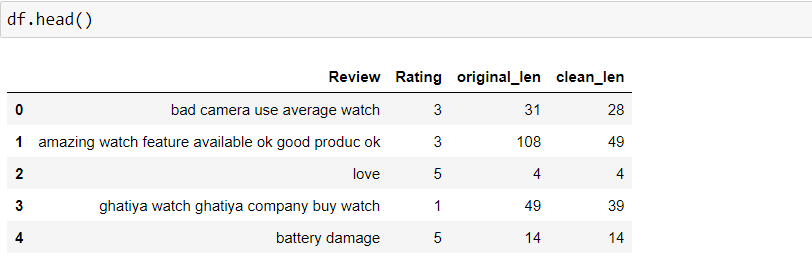
Many product reviews are not accompanied by a scale rating system, consisting only of a textual evaluation. In this case, it becomes problematic and time-consuming to compare different products in order to eventually make a choice between them. Therefore, models able to predict the user rating from the text review are critically important. Getting an overall sense of a textual review could in turn improve consumer experience.

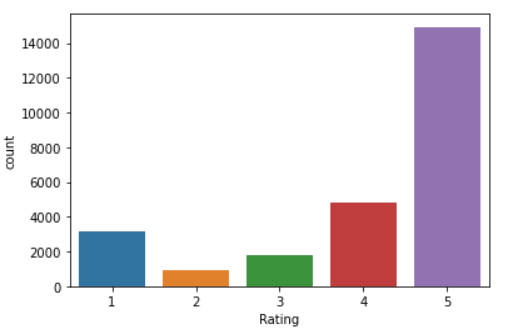
Scrapping & Description of dataset

I have used Flipkart website to extract reviews from ['laptops', 'Phones', 'Headphones', 'smart watches', 'Professional Cameras', 'Printers', 'monitors', 'Home theatre', 'router']. There are in total 25544 rows and 2 columns of ratings and reviews are present in our dataset. -

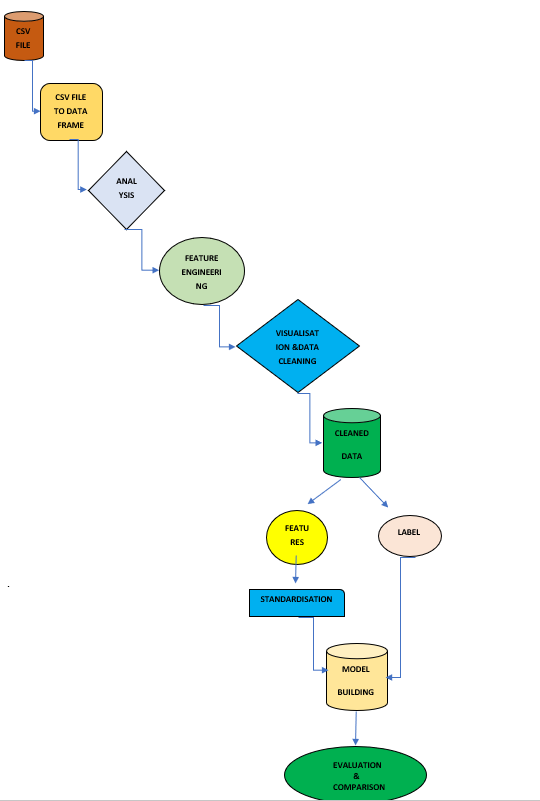
* **Ratings**: It is the Label column, which includes ratings in the form of integers from 1 to 5.
* **Full review**: It contains text data on the basis of which we have to build a model to predict ratings.

We found the occurrence of ratings ratio as shown below. We can observe that our dataset is quite imbalanced. Maximum, 14924 number of ratings present is of 5 star and minimum, 3128 is of 1 star. We then create two more columns length and clean length on the basis of the lengths of the text before and after cleaning for our analysis purpose.





Architecture of dataset

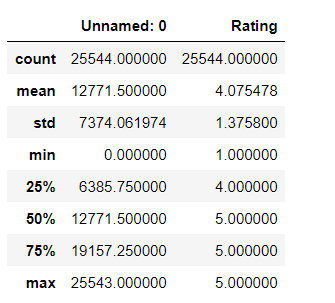


Data pre processing

Data processing and feature engineering are crucial in machine learning to build a prediction model. Furthermore, a model cannot be made without some data processing. For instance, as shown in the experiment, the model could not be trained before handling the missing values and converting the text in the dataset into numerical values. Hence, from the experiment, we saw that pre-processing the data does improve the prediction accuracy.

**info**: it is used to give info about not null value and datatype of features. here datatype of comments is (object) which is been taken care where as label have int datatype.

* **Describe**: The describe method is used for calculating some statistical data like **percentile, mean** and **std** of the numerical values of the Series or Data Frame. It analyses both numeric and object series and also the Data Frame column sets of mixed data types.it also give info about distribution of data.



**Converting upper case to lower case**: It converts all the upper-case text in the comment to lower case, it is an important technique as it helps in cleaning the data. Quite recently, one of readers found that different variation in input capitalization (e.g., ‘Canada’ vs. ‘Canada’) gave him different types of output or no output at all. This was probably happening because the dataset had mixed-case occurrences of the word ‘Canada’ and there was insufficient evidence for the neural-network to effectively learn the weights for the less common version. This type of issue is bound to happen when your dataset is fairly small, and lowercasing is a great way to deal with sparsity issues.

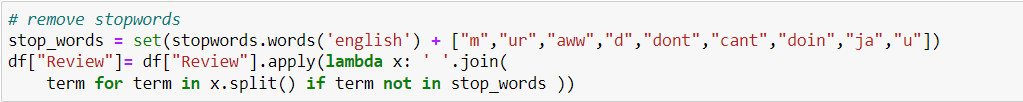


**Text Normalisation**: As I was now certain that there are no missing records in my data, I decided to start with data pre-processing. Firstly, I decided to normalize the text data since comments from online forums usually contain inconsistent language, use of special characters in place of letters (e.g., @rgument), as well as the use of numbers to represent letters (e.g., n0t). To tackle such inconsistencies in data, I decided to use ***Regex.*** The text normalization steps that I performed are listed below: -

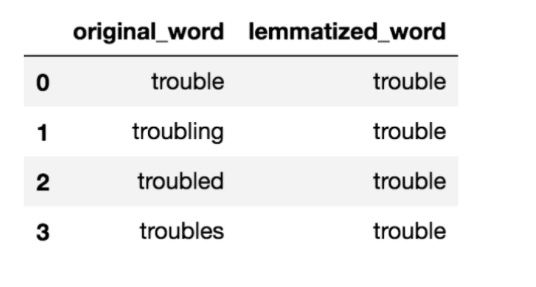
* Removing Characters in between Text.
* Removing Repeated Characters.
* Converting data to lower-case.
* Removing Punctuation.
* Removing unnecessary white spaces in between words.
* Removing “\n”.
* Removing Non-English characters.

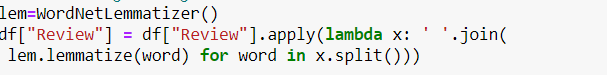


**Stop word Removal:** Stop words are those words that are frequently used in both written and verbal communication and thereby do not have either a positive/negative impact on our statement. Stop words are a set of commonly used words in a language. Examples of stop words in English are “a”, “the”, “is”, “are” and etc. The intuition behind using stop words is that, by removing low information words from text, we can focus on the important words instead. For example, in the context of a search system, if your search query is “what is text processing?”, you want the search system to focus on surfacing documents that talk about text pre-processing over documents that talk about what is. This can be done by preventing all words from your stop word list from being analysed. Stop words are commonly applied in search systems, text classification applications, topic modelling, topic extraction and others.

****

**Lemmatization:** Lemmatization on the surface is very similar to stemming, where the goal is to remove inflections and map a word to its root form. The only difference is that, lemmatization tries to do it the proper way. It doesn’t just chop things off, it actually transforms words to the actual root. For example, the word “better” would map too “good”. It may use a dictionary such as word net for mapping or some special rule-based approaches. Here is an example of lemmatization in action using a WordNet-based approach:

****

****

**Tf-Idf vectorization:** In NLP, tf-idf is an important measure and is used by algorithms like cosine similarity to find documents that are similar to a given search query.

* What is Term Frequency (tf): tf is the number of times a term appears in a particular document. So, it’s specific to a document. A few of the ways to calculate tf is given below: -

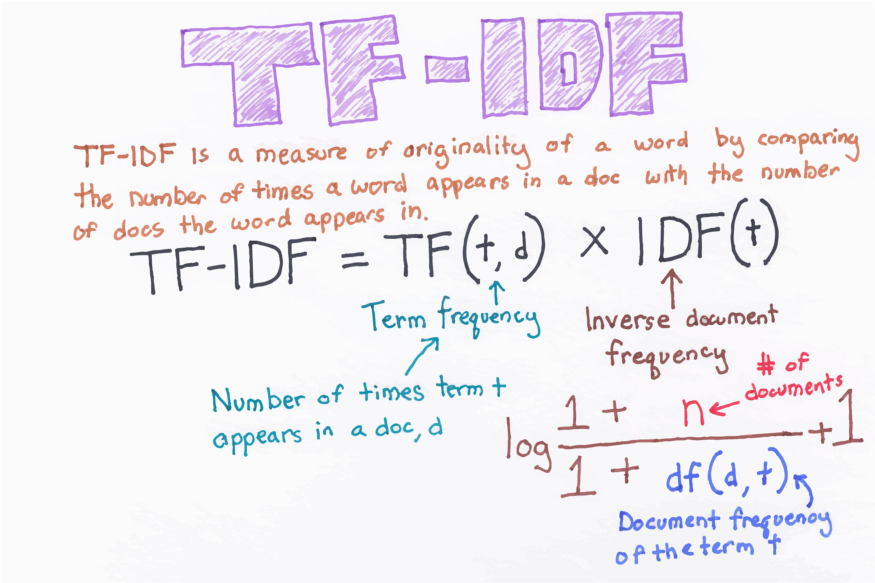
tf(t) = No. of times term ‘t’ occurs in a document.

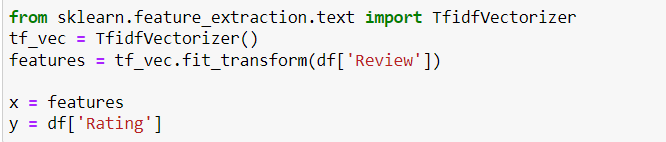
* Inverse Document Frequency (idf):idf is a measure of how common or rare a term is across the entire corpus of documents. So, the point to note is that it’s common to all the documents. If the word is common and appears in many documents, the idf value (normalized) will approach 0 or else approach 1 if it’s rare. A few of the ways we can calculate idf value for a term is given below:

idf(t) = log e [ (1+n) / ( 1 + df(t) ) ] + 1 (default i: e smooth\_idf = True)

and

idf(t) = log e [ n / df(t) ] + 1 (when smooth\_idf = False)



****

Model development and evaluation

HYPER PARAMETER

TUNING

DATA

CLENING

DATA

PREPARATION

DATA

UNDERSTANDING

CROSS VALIDATION SCORE

MODEL

BUILDING

Vectorization(tf-idf)

MODELING

VISUALISATION

DATA MINING

EDA

PROBLEM

FRAMING

Converting upper case to lower case

Text normalisation

Removing stop words & lemmatization

Accuracy score

Above is the architecture which shows the important steps involve in data pre-processing and model building. Firstly, I analyse the dataset and there are only two columns in it one is review and other is rating. As it is a nlp model it involves various data cleaning steps which is needed to clean our data. All the steps involve in data cleaning is explained above. As our machine leaning model will not understand text so tfidf vectorizer is used which is converting text to vector which is easily understood by our model. Finally comes model building, I have used 6 classifier algorithm () which fits best in this dataset and predict good accuracy. After finding my best model with the help of accuracy and cross validation score comes hyper parameter tuning for best fit model.

**Hardware** **requirement**: -

* Minimum core i5 or higher
* Minimum 8gb of ram

Software and tools required:

* Python is widely used in scientific and numeric computing:
* SciPy is a collection of packages for mathematics, science, and engineering.
* Pandas is a data analysis and modelling library.

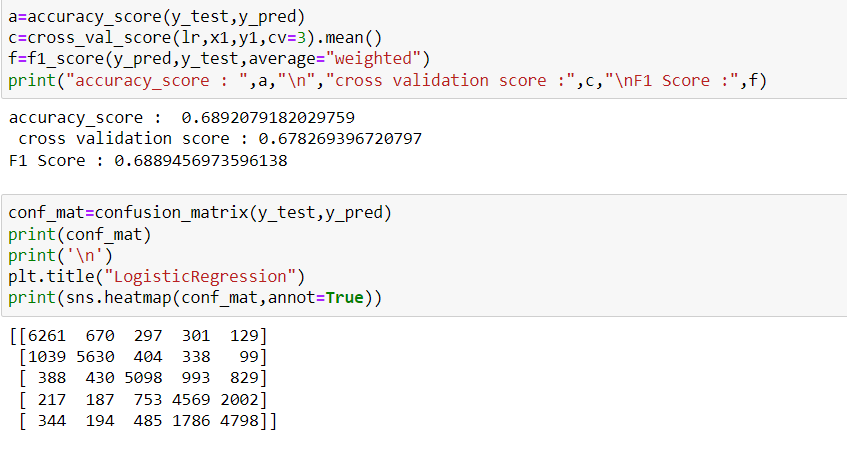
Modules or library required for project data analysis and visualization:

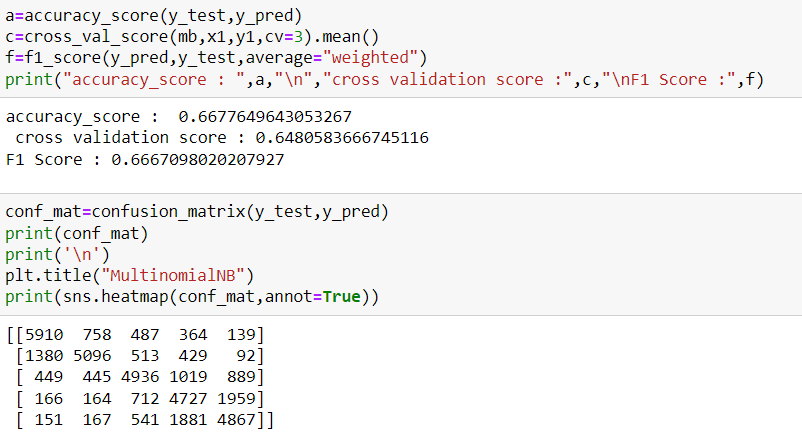
* **Pandas** for data analysis and import
* **NumPy** to perform Mathematical operation
* **Seaborn** and **matplotlib** to data visualization
* **Scikit-learn:** All the models, metrics and feature selection etc are present inside of that module. We import from this library according to our need.
* **Train-Test split**: There are two primary phases in the system:

1. **Training phase:** The system is trained by using the data in the data set and fits a model (line/curve) based on the algorithm chosen accordingly.

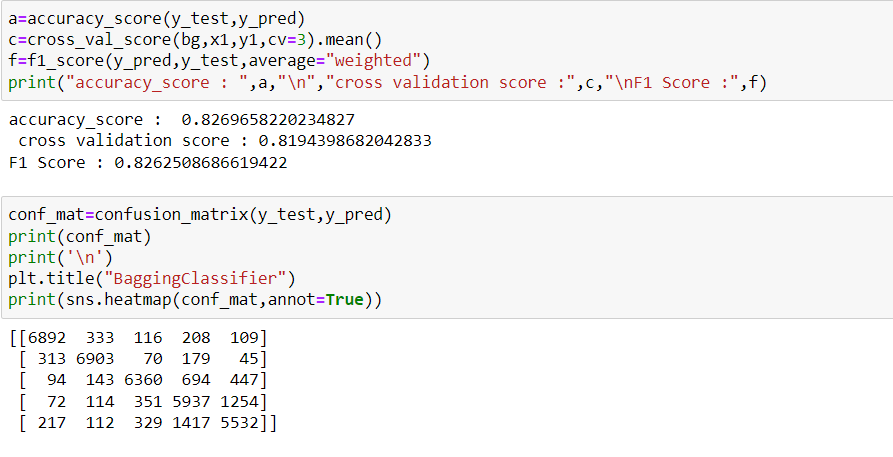
2. **Testing phase:** the system is provided with the testing data, and it is tested for its working. The accuracy is checked. And therefore, the data that is used to train the model or test it, must be appropriate. The system is designed to detect and predict price of used car and hence appropriate algorithms must be used to do the two different tasks

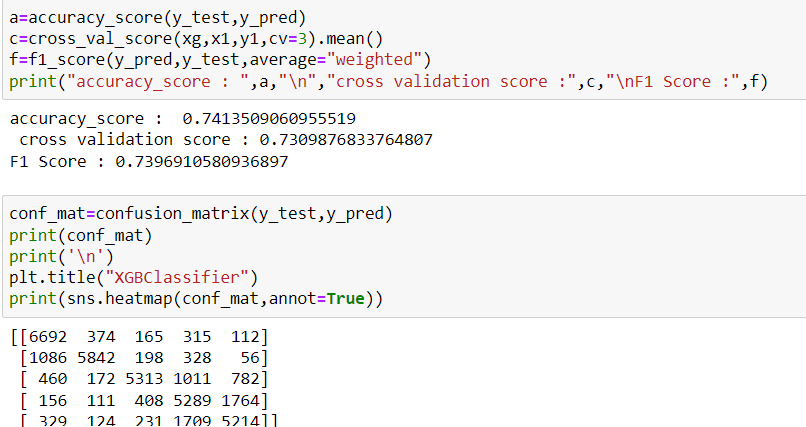
Model Building: I use 4 different algorithms for model building:

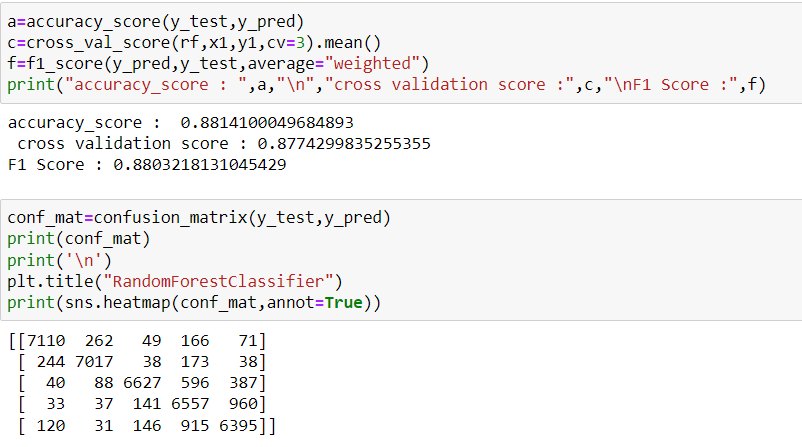
Logistic Regression: 

MultinomialNB: 

Decision Tree Classifier: 

Bagging Classifier: 

XGB Classifier: 

RandomForest Clasifier: 

Accuracy score:

Accuracy is one metric for evaluating classification models. Informally, accuracy is the fraction of predictions our model got right. Formally, accuracy has the following definition:

Accuracy=Number of correct predictions /Total number of predictions

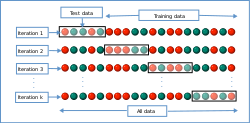
For binary classification, accuracy can also be calculated in terms of positives and negatives as follows:

Accuracy=(TP+TN)/(TP+TN+FP+FN)

Where *TP* = True Positives, *TN* = True Negatives, *FP* = False Positives, and *FN* = False Negatives

Cross Validation Score:

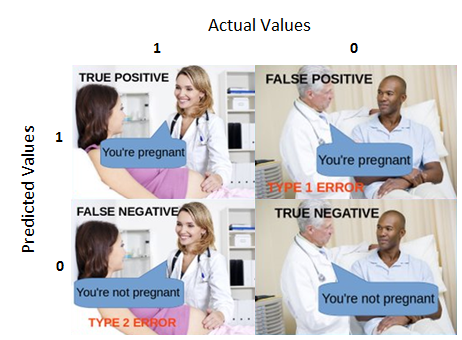
Cross-validation is a statistical method used to estimate the skill of machine learning models. It is commonly used in applied machine learning to compare and select a model for a given predictive modelling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods. The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation.

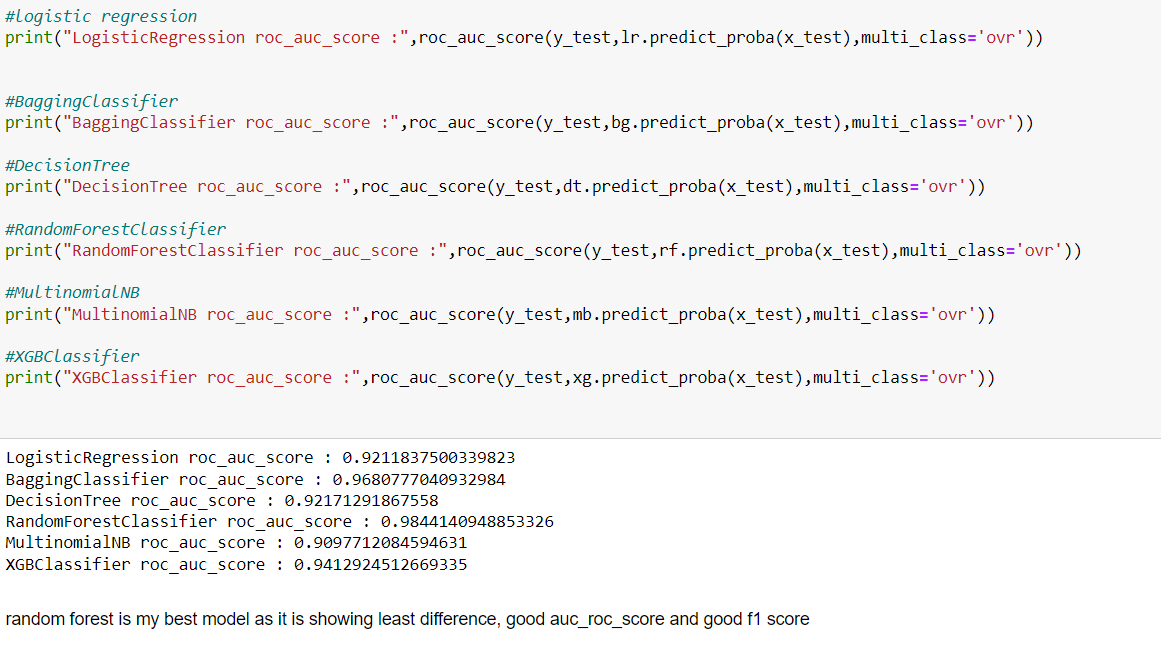


Confusion matrix:

A confusion matrix 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. The confusion matrix itself is relatively simple to understand, but the related terminology can be confusing. Well, it is a performance measurement for machine learning classification problem where output can be two or more classes. It is a table with 4 different combinations of predicted and actual values. It is extremely useful for measuring Recall, Precision, Specificity, Accuracy, and most importantly AUC-ROC curves.

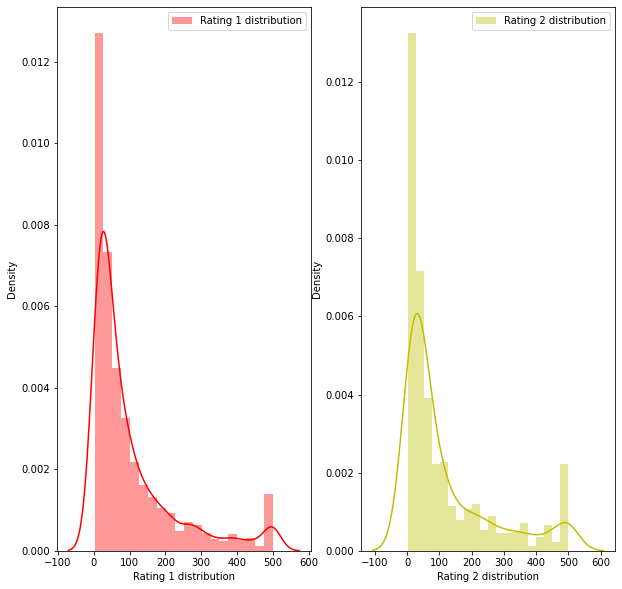
Let’s understand TP, FP, FN, TN in terms of pregnancy analogy.

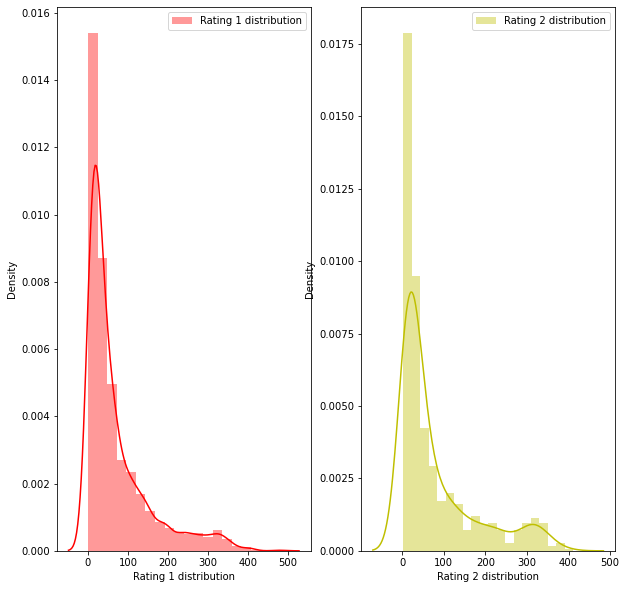


AUC-ROC SCORE: AUC - ROC curve is a performance measurement for the classification problems at various threshold settings. ROC is a probability curve and AUC represents the degree or measure of separability. It tells how much the model is capable of distinguishing between classes. Higher the AUC, the better the model is at predicting 0 classes as 0 and 1 classes as 1. 

Visualisation:

Data visualization is the discipline of trying to understand data by placing it in a visual context so that patterns, trends and correlations that might not otherwise be detected can be exposed.





The above distribution plot shows the review distribution before cleaning and review distribution after cleaning. As we can see and analyze from the graph that distribution is close to normal distribution which is good for model building as our model shows good accuracy on normally distributed dataset.

Word Cloud:

Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud. Word clouds are widely used for analyzing data from social network websites.



RESULT:

Many machine learning algorithms are used to predict. However, the prediction accuracy of these algorithms depends heavily on the given data when training the model. If the data is in bad shape, the model will be overfitted and inefficient, which means that data pre-processing is an important part of this experiment and will affect the final results. Thus, multiple combinations of pre-processing methods need to be tested before getting the data ready to be used in training. After analyzing every model Random Forest Classifier shows good accuracy and cv with least difference and on doing hyper parameter tuning it accuracy reaches to 88.

CONCLUSION:

In this project we have tried to detect the Ratings in commercial websites on a scale of 1 to 5 on the basis of the reviews given by the users. We made use of natural language processing and machine learning algorithms in order to do so. We interpreted that Random Forest classifier model is giving us best results. Through this project we were able to learn various Natural language processing techniques like lemmatization, stemming, removal of stop words. This project has demonstrated the importance of sampling effectively, modelling and predicting data. Through different powerful tools of visualization, we were able to analyze and interpret different hidden insights about the data. The few challenges while working on this project where: -

• Imbalanced dataset

• Lots of text data

The dataset was highly imbalanced so we balanced the dataset using smote technique. We converted text data into vectors with the help of tfidf vectorizer.

FUTURE WORK:

While we couldn’t reach out goal of maximum accuracy in Ratings prediction project, we did end up creating a system that can with some improvement and deep learning algorithms get very close to that goal. This model can further be improved with the addition of more algorithms into it. However, the output of these algorithms needs to be in the same format as the others. Once that condition is satisfied, the modules are easy to add as done in the code. We can also improve our model accuracy by fetching same data for all the class in rating. This provides a great degree of modularity and versatility to the project.

REFRENCE:

<https://machinelearningmastery.com/multi-class-imbalanced-classification/>.

<https://scholar.smu.edu/cgi/viewcontent.cgi?article=1134&context=datasciencereview>.