Amazon Review Sentiment Analysis

Submitted in partial fulfillment of the requirements of the degree

BACHELOR OF ENGINEERING IN COMPUTER ENGINEERING

By

Paras Patil / 41

Vishesh Mittal/ 35

Kapish Madhwani/ 32

Kaustubh Keny/ 26

Supervisor
Prof. Vidya Zope

Department of Computer Engineering

Vivekanand Education Society's Institute of Technology

HAMC, Collector's Colony, Chembur,

Mumbai-400074

University of Mumbai

(AY 2020-21)

CERTIFICATE

This is to certify that the Mini Project entitled "Amazon Review Sentiment Analysis" is a bonafide work of Paras Patil (41), Vishesh Mittal(35), Kaustubh Keny(26), Kapish Madhwani(32) submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of "Bachelor of Engineering" in "Computer Engineering" .

(Prof. Vidya Zope)

Supervisor

(Dr. Nupur Giri) **(Dr. J.M. Nair)**

Head of Department **Principal**

Mini Project Approval

This Mini Project entitled “**Amazon Review Sentiment Analysis**” by Paras Patil(41), Vishesh Mittal(35), Kaustubh Keny(26), Kapish Madhwani(32) is approved for the degree of Bachelor of Engineering in Computer Engineering.

Examiners

1………………………………………

(Internal Examiner Name & Sign)

2………………………………………

(External Examiner name & Sign)

Date:

Place:

Contents

Abstract :

Acknowledgements

Firstly, we would like to thank the Computer department of VESIT for providing a platform to showcase our skills. We would also like to offer our sincere gratitude to our mentor and our supervisor for this project , **Prof . Vidya Zope** ma’am. Her enthusiasm , knowledge and exacting attention to the details of this project have been an inspiration and kept our work on track right from the beginning of the Literature Survey uptill the writing of this report paper.

We are also grateful to the various authors of the research papers which we reviewed . Their generosity and expertise of one and all have improved this project in numerous ways and have saved us from notable errors and those that inevitably remain are entirely our own responsibility.

List of Abbreviations:

Sr. No.	Abbreviations	Description
1.	ML	Machine Learning
2.	VADER	Valence Aware Dictionary for Sentiment Reasoning
3.	TF-IDF	Term Frequency Inverse Document Frequency
4.	LSA	Latent Semantic Analysis
5.	LDA	Linear Discriminant Analysis
6.	CV	Count Vectorization
7.	SVM	Support vector Machines
8.	NB	Naive Bayes
9.	DT	Decision Trees

List of Figures :

Introduction

1.1 Introduction

Our project is centered upon the Sentiment Analysis of Amazon Customer reviews. **Sentiment analysis** (or **opinion mining**) uses natural language processing and machine learning to interpret and classify emotions in subjective data. **Sentiment analysis** is often used in businesses to detect sentiments in social data, gauge brand reputation, and for understanding customers.

We therefore attempt to implement the same right from cleaning unstructured text data to proper reviews. Our project furthermore, focuses on classifying the polarities in user reviews using various Machine Learning Algorithms.

1.2 Motivation for the project

The domain of **Natural Language processing** and its application is phenomenally huge and there is an enormous amount of data available on web applications and various other platforms where people share and get acquainted with information and express their views, therefore for building a brand around them we need to make use of the techniques available for harnessing useful information while respecting user privacy.

Machine learning is a booming industry and this motivated our group to explore something beyond the curriculum. Data Science and ML jobs are sought to be one of the best jobs in the 21st century as per renowned data experts.

Our Group shares a mutual interest in the Machine Learning and App Development domain, so we decided to combine these two.

1.3 Problem Statement & Objectives

We are performing the Sentimental analysis on **Amazon product reviews** after classification of text and then using various Machine Learning algorithms.

We will then comply with testing and comparing a number of supervised and unsupervised algorithms in order to determine which is the best suited for our purpose.

As of this review, we are planning on implementing algorithms such as **Random Forest** and **SVM** as well as different types of **Naive Bayes** in supervised learning domain.

Our objectives would be,

To understand various aspects and working of different algorithms and thus, compare them qualitatively as well as quantitatively.

To perform sentiment analysis on Amazon Customer reviews and determine the overall consumer polarity over the products.

To deploy an application to demonstrate Sentiment Analysis.

2)Literature Survey:

1) Textual Dissection Of Live Twitter Reviews Using Naive Bayes :-

By:

1) Sourav.Kunal
2) Arijit Saha
3) Aman.Varma
4) Varma Vivek Tiwari

Summary:

This paper proposes the use of Tweepy and TextBlob as a python library to access and classify Tweets using Naïve Bayes. They developed an algorithm that takes the query as the person’s name for whom the user wants to calculate the percentage of positive & negative tweets.

Advantages:

1) The live data capture is an excellent feature which takes care that all the data is up to date.
2) Using the pos_tagger makes it easier to use the lemmatizer.
3) TextBlob has a MIT License, is built on the top of NLTK and is very easily accessible. TextBlob is used for fast prototyping. On Comparing the Code Quality as calculated and provided by Lumnify, TextBlob is Level 3 and that of NLTK is Level 2.

Disadvantages:

1) If there is a foreign category in the test dataset, it will be assigned 0 probability. This can be rectified using smoothing techniques.
2) Multinomial Naive Bayes can handle and analyse this data with higher accuracy and efficiency.
3) They also used a uni-gram method.

2) Sentimental Analysis of Twitter Data with respect to General Elections in India :-

By:

1) Ankita Sharma
2) Udayan Ghose

Summary:

In this paper, an efficient approach is proposed to analyze the sentiments of the twitter data.Lexicon Based Approach for Sentiment Analysis it classifies words into 3 categories positive,negative and neutral.NRC Dictionary Based Approach it classify words in 6 categories.They used Rapidminer.

Advantages:

1) Lexicon based approach is nice for small datasets and it can be fast as well as effective.
2) Since data is sentence sized, the positive and negative sentiment scores aren't averaged to zero.

Disadvantages:

1) RapidMiner accuracy is less compared to **VADER**.
2) They used uni-grams which are less effective to catch sarcasm.

3) Sentiment analysis of social media Twitter with case of Anti-LGBT campaign in Indonesia using Naive Bayes, Decision Tree & Random Firest Algorithm:-

By:

1) Veny Amilia Fitri
2) Rachmadita Andreswari
3) Muhammad Azani Hasibuan

Summary:

This paper presents the usage of different classifier algorithms to analyse the sentiment associated with Anti-LGBT tweets. It used twitter as a source of data to feed and train the models. Based on parameters like recall, precision, F1-measure, it was concluded that Naive Bayes had the highest accuracy (86.43%) compared to rest (~82%). After analysis of data, the resultant major tendency for comments was found to be neutral.

Advantages:

1) A detailed comparative study between Random Forest, Naive Bayes and Decision Tree was made.
2) Naive Bayes can easily provide high accuracy in case of sentence sized data like tweets.

Disadvantages:

1) The number of models compared is low .
2) Naive Bayes assumes all features to be independent. This will reduce accuracy when collinearity is to be considered.

4)Text Classification using Different Feature Extraction Approaches:

By:

1) Robert Dzisevic
2) Dmitrij Sesok

Summary:

In this paper, three different text feature extraction dimensionality reduction approaches based on TF-IDF were applied for classification using keras. The TF-IDF LSA approach outperformed the plain TF-IDF by 1% for small datasets while the plain TF-IDF gave higher accuracy (91%) for larger datasets. The TF-IDF LDA approach failed to reduce the noise in data in both cases which ended up reducing the accuracy.

Advantages:

1) Overfitting of the TF-IDF model is prevented with LSA and LDA enhancements which can become very apparent in very large data-sets.
2) The calculations involved in TF-IDF are very easy.

Disadvantages:

1) If a word occurs in every element of the dataset, TF-IDF will assign it a 0 weightage thus, those words become irrelevant to other corpus terms.
2) The extraction time with TF-IDF increases exponentially with increase in document size.

5) Sentimental Short Sentences Classification by Using CNN Deep Learning Model with Fine Tuned Word2Vec

By:

1) Amit Kumar Sharma,
2) Sandeep Chaurasia
3) Devesh Kumar Srivastava

Summary:

This paper provides sentimental summarization of short sentences. Movie review corpus(imdb) was used. Their search is giving a better accurate result for feature extraction through Word2Vec and CNN methods for small sentences of movie review corpus. The proposed model is providing 99.07% accuracy for training samples and 82.19% for testing samples.

Advantages:

1) The CNN model provides better and more accurate results as compared to NB and ELM models.
2) The proposed model is better for learning local features from phrases or words

Disadvantages:

1) Occurence of multilingual content in the dataset leads to low sentimental results.
2) The model lacks in accuracy for sequential data (long sentences)
3) Social media databases are big and susceptible to noisy, incomplete, and inconsistent text due to their origin from different people and sources.

6) Constructing a heterogeneous training dataset for Emotion Classification

By :

1) Anchal Gupta
2) Satish Mahadevan Srinivasan

Summary :

This paper presents a dataset for twitter sentiment analysis by using 6 models and compares their F1 Accuracy . In this paper they have used lexicon based NRC classifiers for different emotions such as joyful , sad , angry and surprise.

Advantages :

1) Deep Neural networks showed a mediocre performance compared to other models , but these models did not overfit.

Disadvantages:

1) Models like RF performed well on testing data but performed poorly on training data sets due to overfitting.

7) String-based Multinomial Naïve Bayes for Emotion Detection among Facebook Diabetes Community.

By :

1) Vimala Balakrishnan
2) Wandeep Kaur

Summary :

This paper determined the emotions among the online Diabetes community on Facebook. The emotions were classified according to pultchik's wheel of emotions , comprising of 8 emotions fear, joy ,anger , sadness, surprise , trust , anticipation and disgust. It used 4 classifiers and their accuracy was compared .

Advantages:

1) String - based Multinomial Naive Bayes classifier outperformed every other classifier used for a particular data set.

Disadvantages:

1) The biggest drawback of MNB is that if it comes across any category that didn't exist in the training data set , it will not be able to predict the output, as the probability of that category was assigned to zero.

Proposed System:

1) Introduction:

Text classification can be done two different ways: manual or automatic classification. In the former, a human annotator interprets the content of text and categorizes it accordingly. This method can usually provide quality results but it’s time-consuming and expensive. The latter applies machine learning, natural language processing (NLP), and other AI-guided techniques to automatically classify text in a faster, more cost-effective, and more accurate manner.

There are many approaches to automatic NLP text classification, which fall into into three types of systems:

- Rule-based systems
- Machine learning-based systems
- Hybrid systems

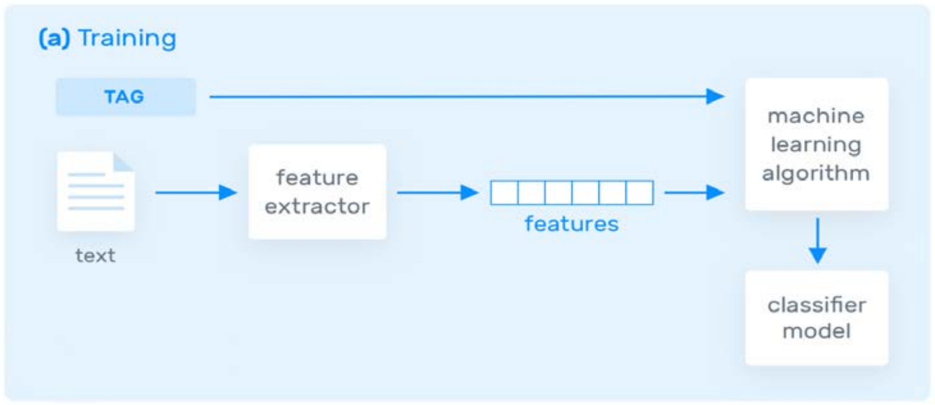
Our system is focused on Machine Learning approach:

Instead of relying on manually crafted rules, machine learning text classification learns to make classifications based on past observations. By using pre-labeled examples as training data, machine learning algorithms can learn the different associations between pieces of text, and that a particular output (i.e., tags) is expected for a particular input (i.e., text). A “tag” is the predetermined classification or category that any given text could fall into.

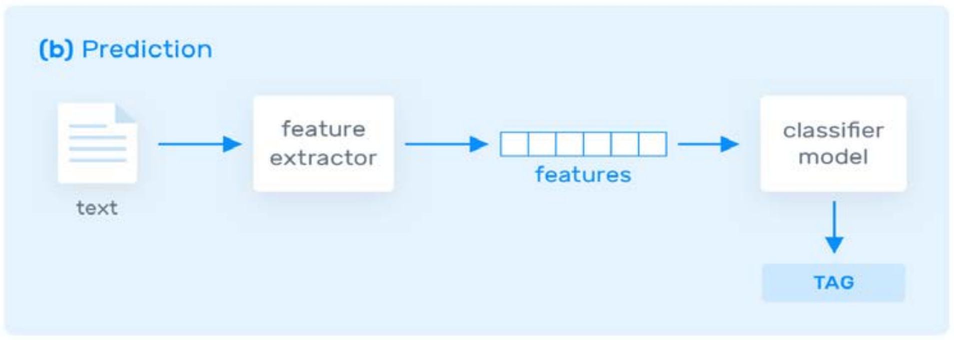
The first step towards training a machine learning NLP classifier is feature extraction: a method is used to transform each text into a numerical representation in the form of a vector. One of the most frequently used approaches is bag of words, where a vector represents the frequency of a word in a predefined dictionary of words.

For example, if we have defined our dictionary to have the following words {*This*, *is*, *the*, *not*, *awesome*, *bad*, *basketball*}, and we wanted to vectorize the text *"This is awesome,"* we would have the following vector representation of that text: (1, 1, 0, 0, 1, 0, 0).

Then, the machine learning algorithm is fed with training data that consists of pairs of feature sets (vectors for each text example) and tags (e.g. *sports*, *politics*) to produce a classification model:

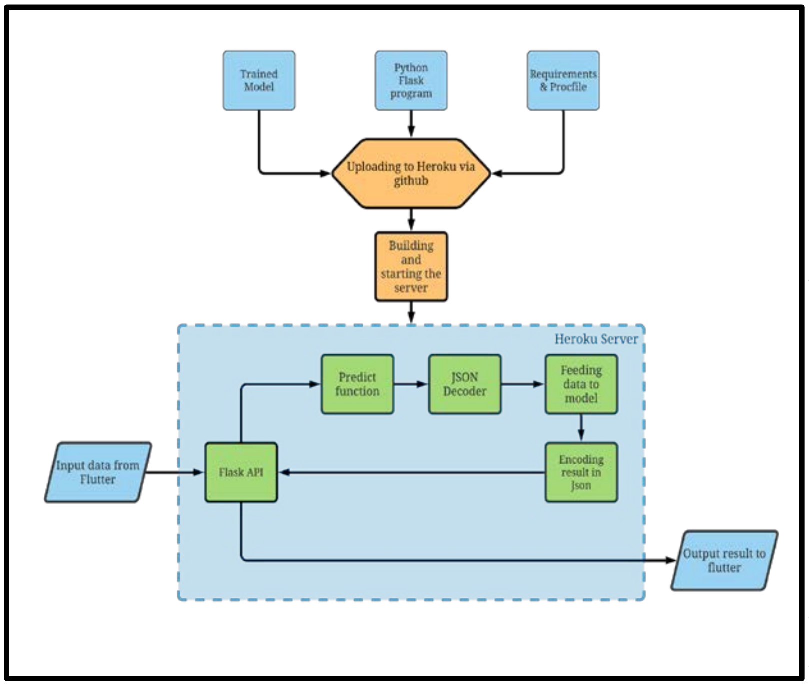


Once it's trained with enough training samples, the machine learning model can begin to make accurate predictions. The same feature extractor is used to transform unseen text to feature sets which can be fed into the classification model to get predictions on tags (e.g. *positive*, *negative*).



2)Architecture / Framework:

In order to perform the live testing of the models as well as deployment, an application was developed named '**ARSA**'. **Flask API** forms the back-bone of the integrating processes between ML models (backend) & flutter (frontend). Flask is a framework of Python that allows us to build up web-applications. Flask is based on WSGI (Web Server Gateway Interface) toolkit and Jinja2 template engine.



The flask python program and ML models are uploaded to Heroku. **Heroku** is a platform which provides a cloud space for running applications. The files are uploaded via GitHub.

Within the server, the flask API waits for a POST request which will be sent by flutter when the user presses 'PREDICT' button. This sends the data in json format to the API and also invokes the '**/predict**' route where all the analysis takes place.

The dropdown box within the flutter application allows selection of models which gets sent along with the text to be analysed. The text is then feeded to a selected model for prediction and the result obtained is sent back to the flutter application in json format where it is again reconverted and displayed.

3)Algorithm & Process Design:

Machine Learning Text Classification Algorithms

Some of the most popular machine learning algorithms for creating text classification models include the Naive Bayes family of algorithms, support vector machines (SVM), and deep learning(future scope).

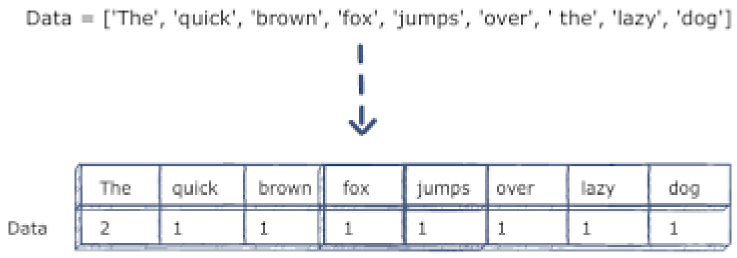
There are two ways to make binary/boolean features or to vectorize.

1. **True/False approach:**

   In this approach, the presence of certain words (features) in training data is tested and fitted to the resultant sentiment. Each feature is given a sentiment probability based on the number of times one classification is made with the presence of each feature being true (is present) or false (is absent). Here, it only considers the occurrence of the feature and not the frequency.

2. **Count Vectorization approach:**

   In this approach, the words (features) are converted into a vector (matrix) storing the number of occurrences of each feature i.e. its frequency.



```
Data = ['The', 'quick', 'brown', 'fox', 'jumps', 'over', ' the', 'lazy', 'dog']
```

	The	quick	brown	fox	jumps	over	lazy	dog
Data	2	1	1	1	1	1	1	1

Common cleaning methods:

1. **Removing stopwords**. These are common words that don't really add anything to the classification, such as a, able, either, else, ever and so on. So for our purposes, *The election was over* would be *election over* and *a very close game* would be a very *close game.*
2. **Lemmatizing words**. This is grouping together different inflections of the same word. So election, elections, elected, and so on would be grouped together and counted as more appearances of the same word.

We are going to use following classifier's:

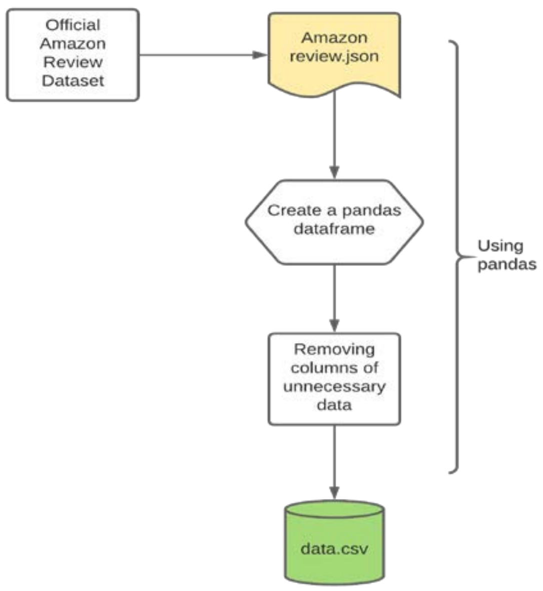
From NLTK:(True/False based)

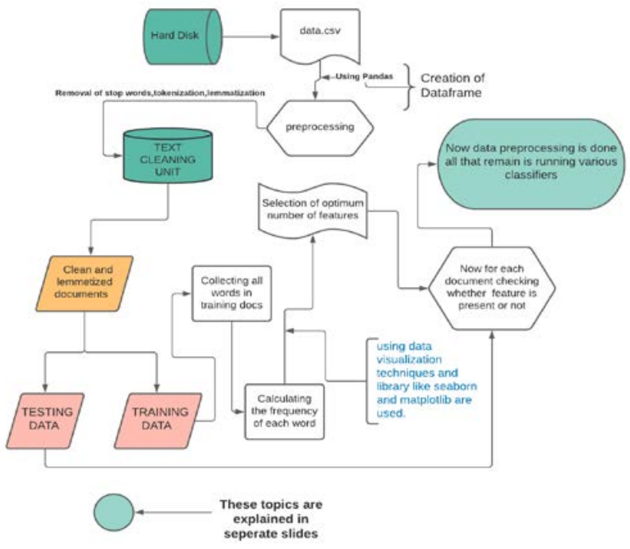
1. Bernoulli Naive Bayes

From SKLEARN:(Count Vectorization i.e. frequency based)

1. Multinomial Naive Bayes
2. Support Vector Machines
3. Decision Tree

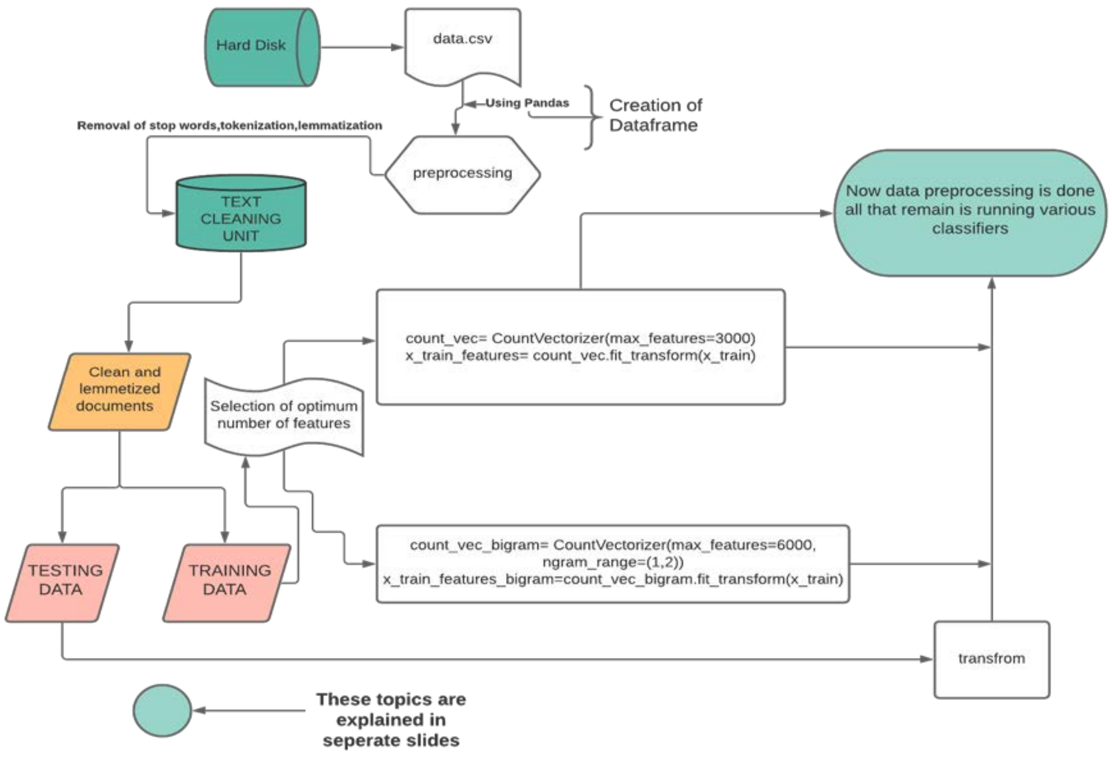
NOTE: Both unigram and bigram forms of the given SKLEARN models were implemented.


Official Amazon Review Dataset
Amazon review.json
Create a pandas dataframe
Removing columns of unnecessary data
data.csv
Using pandas


Hard Disk
data.csv
Using Pandas
Creation of Dataframe
Removal of stop words,tokenization,lemmatization
preprocessing
TEXT CLEANING UNIT
Now data preprocessing is done all that remain is running various classifiers
Selection of optimum number of features
Clean and lemmetized documents
Collecting all words in training docs
Now for each document checking whether feature is present or not
using data visualization techniques and library like seaborn and matplotlib are used.
TESTING DATA
TRAINING DATA
Calculating the frequency of each word
These topics are explained in seperate slides

Cleaning unit for Bernoulli naive bayes:

Using n-grams. Instead of counting single words as we did here, we could count sequences of words, like “clean match” and “close election”.We are performing bigrams for all sklearn classifiers.


Hard Disk
data.csv
Using Pandas
Creation of Dataframe
Removal of stop words,tokenization,lemmatization
preprocessing
TEXT CLEANING UNIT
Now data preprocessing is done all that remain is running various classifiers
count_vec= CountVectorizer(max_features=3000)
x_train_features= count_vec.fit_transform(x_train)
Clean and lemmetized documents
Selection of optimum number of features
count_vec_bigram= CountVectorizer(max_features=6000, ngram_range=(1,2))
x_train_features_bigram=count_vec_bigram.fit_transform(x_train)
TESTING DATA
TRAINING DATA
transfrom
These topics are explained in seperate slides

CLASSIFIERS:

1) Naive Bayes:

The Naive Bayes family of statistical algorithms are some of the most used algorithms in text classification and text analysis, overall.

One of the members of that family is Multinomial Naive Bayes (MNB) with a huge advantage, that you can get really good results even when your dataset isn’t very large (~ a couple of thousand tagged samples in our case 40k) and computational resources are scarce.

Naive Bayes is based on Bayes’s Theorem, which helps us compute the conditional probabilities of the occurrence of two events, based on the probabilities of the occurrence of each individual event. So we’re calculating the probability of each tag for a given text, and then outputting the tag with the highest probability.

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$$

The probability of A, if B is true, is equal to the probability of B, if A is true, times the probability of A being true, divided by the probability of B being true.

This means that any vector that represents a text will have to contain information about the probabilities of the appearance of certain words within the texts of a given category, so that the algorithm can compute the likelihood of that text’s belonging to the category.

A. MultinomialNB:

Implements the naive Bayes algorithm for multinomial distributed data, and is one of the two classic naive Bayes variants used in text classification (where the data are typically represented as word vector counts, although tf-idf vectors are also known to work well in practice). The distribution is parametrized by vectors $\theta_y=(\theta_{y1},\ldots,\theta_{yn})$

for each class y, where n is the number of features (in text classification, the size of the vocabulary) and θ_{yi} is the probability $P(x_i|y)$ of feature i appearing in a sample belonging to class y.The parameters θ_y is estimated by a smoothed version of maximum likelihood, i.e. relative frequency counting:

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n}$$

where $N_{yi}=\sum_{x \in T} x_i$ is the number of times feature i appears in a sample of class Y in the training set T, and $N_y=\sum_{i=1}^{n} N_{yi}$ is the total count of all features for class y.The smoothing priors $\alpha \geq 0$ accounts for features not present in the learning samples and prevents zero probabilities in further computations. Setting $\alpha=1$ is called Laplace smoothing, while $\alpha<1$ is called Lidstone smoothing.

B. Bernoulli NB:

BernoulliNB implements the naive Bayes training and classification algorithms for data that is distributed according to multivariate Bernoulli distributions; i.e., there may be multiple features but each one is assumed to be a binary-valued (Bernoulli, boolean) variable. Therefore, this class requires samples to be represented as binary-valued feature vectors; if handed any other kind of data, a BernoulliNB instance may binarize its input (depending on the binarize parameter).

The decision rule for Bernoulli naive Bayes is based on:

$$P(x_i \mid y) = P(i \mid y)x_i + (1 - P(i \mid y))(1 - x_i)$$

which differs from multinomial NB’s rule in that it explicitly penalizes the non-occurrence of a feature i that is an indicator for class y, where the multinomial variant would simply ignore a non-occurring feature.

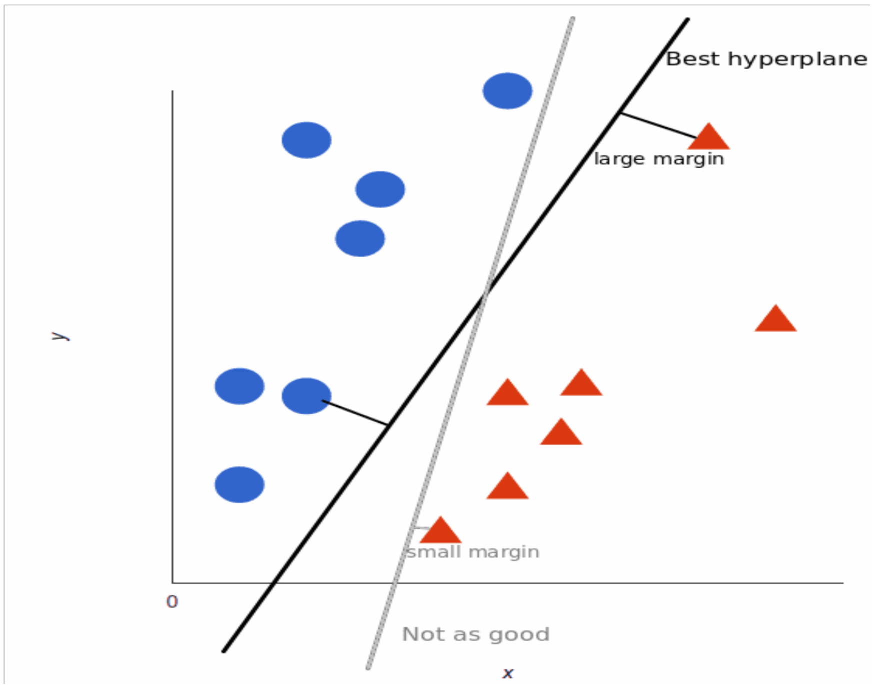
In the case of text classification, word occurrence vectors (rather than word count vectors) may be used to train and use this classifier. BernoulliNB might perform better on some datasets, especially those with shorter documents. It is advisable to evaluate both models, if time permits.

2) Support Vector Machines:

Support Vector Machines (SVM) is another powerful text classification machine learning algorithm, becauseike Naive Bayes, SVM doesn’t need much training data to start providing accurate results. SVM does, however, require more computational resources than Naive Bayes, but the results are even faster and more accurate.

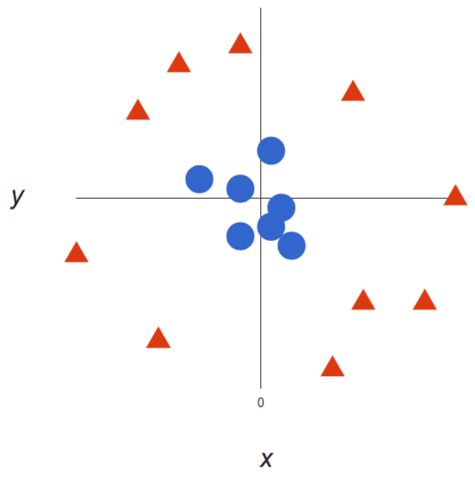
In short, SVM draws a line or “hyperplane” that divides a space into two subspaces. One subspace contains vectors (tags) that belong to a group, and another subspace contains vectors that do not belong to that group.

The optimal hyperplane is the one with the largest distance between each tag. In two dimensions it looks like this:



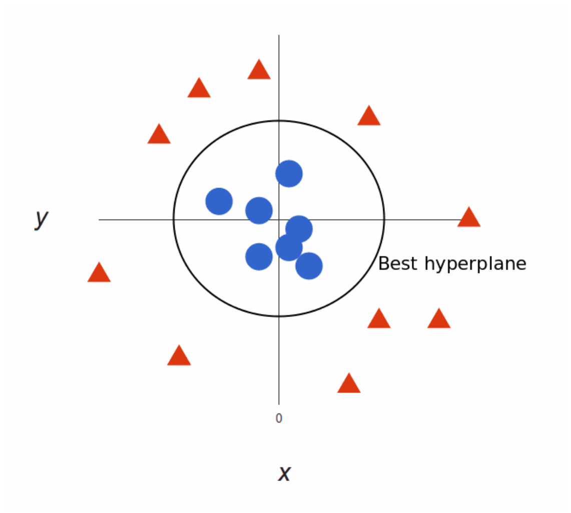
Those vectors are representations of your training texts, and a group is a tag you have tagged your texts with.

As data gets more complex, it may not be possible to classify vectors/tags into only two categories. So, it looks like this:



But that's the great thing about SVM algorithms – they're "multi-dimensional." So, the more complex the data, the more accurate the results will be. Imagine the above in three dimensions, with an added Z-axis, to create a circle.

Mapped back to two dimensions the ideal hyperplane looks like this:



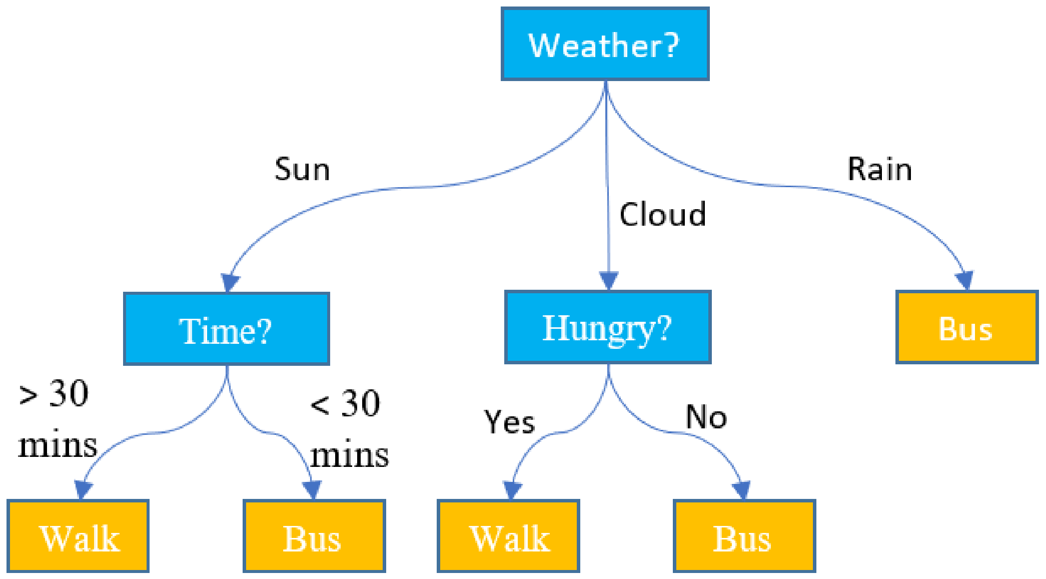
3) Decision tree:

Decision Trees (DTs) are a non-parametric supervised learning method used for classification and regression. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.

For instance, in the example below, decision trees learn from data to approximate a sine curve with a set of if-then-else decision rules. The deeper the tree, the more complex the decision rules and the fitter the model.

Some advantages of decision trees are:

1. Simple to understand and to interpret. Trees can be visualised.Requires little data preparation. Other techniques often require data normalisation, dummy variables need to be created and blank values to be removed. Note however that this module does not support missing values.
2. The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
3. Able to handle both numerical and categorical data. Other techniques are usually specialised in analysing datasets that have only one type of variable. See algorithms for more information.



4. Able to handle multi-output problems.

4) Details of hardware & software:

Hardware:

- HP Pavilion-15 laptop with an Intel core i5 9th gen processor upto 4.1 GHz and GTX 1050 ti mobile graphics card.
- HP Pavilion-15 laptop with a Ryzen 5 3rd gen processor upto 4.2 GHz and GTX 1650 mobile graphics card.

Software:

- Jupyter Notebook, Python, Anaconda & its libraries.
- Github.
- Android Studio, flutter.
- Microsoft VS Code.

4)Experiment & Result:

a) Classification report:

Accuracy: the percentage of texts that were categorized with the correct tag.

Precision: the percentage of examples the classifier got right out of the total number of examples that it predicted for a given tag.

Recall: the percentage of examples the classifier predicted for a given tag out of the total number of examples it should have predicted for that given tag.

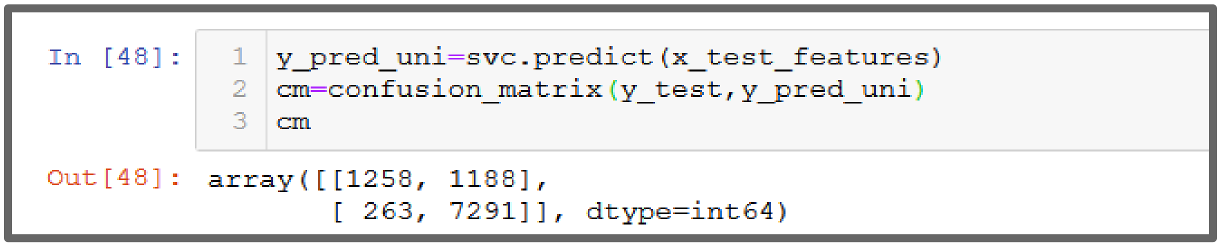
F1 Score: the harmonic mean of precision and recall.

Model		**Precision**	**Recall**	**F1-score**	**Support**
SVM (uni-grams)	**neg**	0.83	0.51	0.63	2446
	pos	0.86	0.97	0.91	7554
	accuracy		*0.85*		10000
	macro avg	0.84	0.74	0.77	10000
	weighted avg	0.85	0.85	0.84	10000
SVM (bi-grams)	**neg**	0.83	0.53	0.65	2446
	pos	0.86	0.97	0.91	7554
	accuracy		*0.86*		10000
	macro avg	0.85	0.75	0.78	10000
	weighted avg	0.86	0.86	0.85	10000
DT (uni-grams)	**neg**	0.56	0.54	0.55	2446
	pos	0.85	0.86	0.86	7554
	accuracy		*0.78*		10000
	macro avg	0.70	0.70	0.70	10000

	weighted avg	0.78	0.78	0.78	10000
DT (bi-grams)	**neg**	0.56	0.54	0.55	2446
	pos	0.85	0.86	0.86	7554
	accuracy	*0.79*			10000
	macro avg	0.71	0.70	0.70	10000
	weighted avg	0.78	0.79	0.78	10000
MNB (uni-grams)	**neg**	0.70	0.66	0.68	2352
	pos	0.90	0.91	0.90	7648
	accuracy	*0.85*			10000
	macro avg	0.80	0.79	0.79	10000
	weighted avg	0.85	0.85	0.85	10000
MNB (bi-grams)	**neg**	0.68	0.73	0.70	2352
	pos	0.90	0.91	0.90	7648
	accuracy	*0.85*			10000
	macro avg	0.79	0.81	0.80	10000

	weighted avg	0.86	0.85	0.85	10000

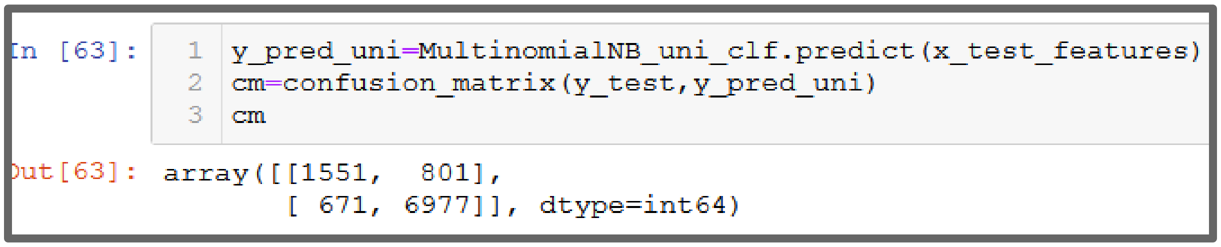
b.)Confusion matrices:



```
In [48]:  y_pred_uni=svc.predict(x_test_features)
          cm=confusion_matrix(y_test,y_pred_uni)
          cm

Out[48]: array([[1258, 1188],
                [ 263, 7291]], dtype=int64)
```

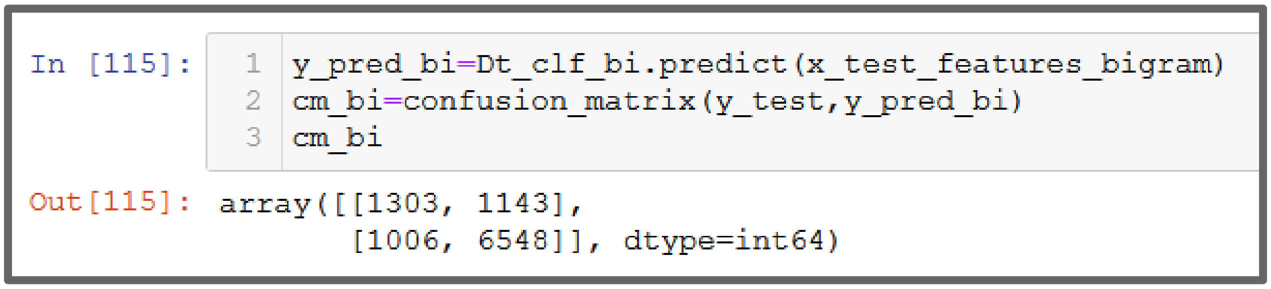
SVC(uni-grams)



```
In [63]:  y_pred_uni=MultinomialNB_uni_clf.predict(x_test_features)
          cm=confusion_matrix(y_test,y_pred_uni)
          cm

Out[63]: array([[1551,  801],
                [ 671, 6977]], dtype=int64)
```

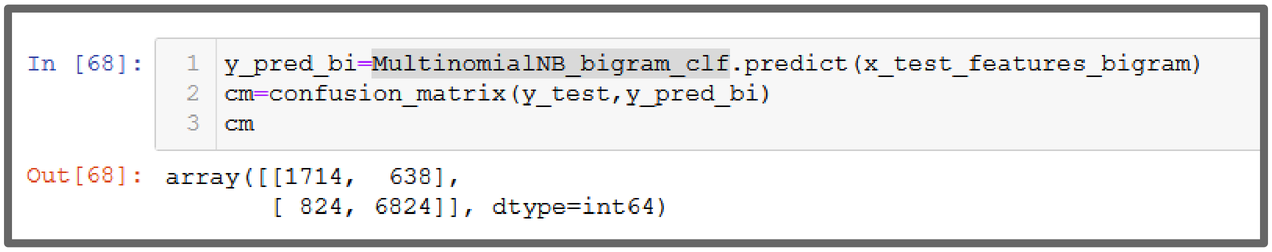
MNB(uni-grams)



```
In [115]:  y_pred_bi=Dt_clf_bi.predict(x_test_features_bigram)
           cm_bi=confusion_matrix(y_test,y_pred_bi)
           cm_bi

Out[115]: array([[1303, 1143],
                 [1006, 6548]], dtype=int64)
```

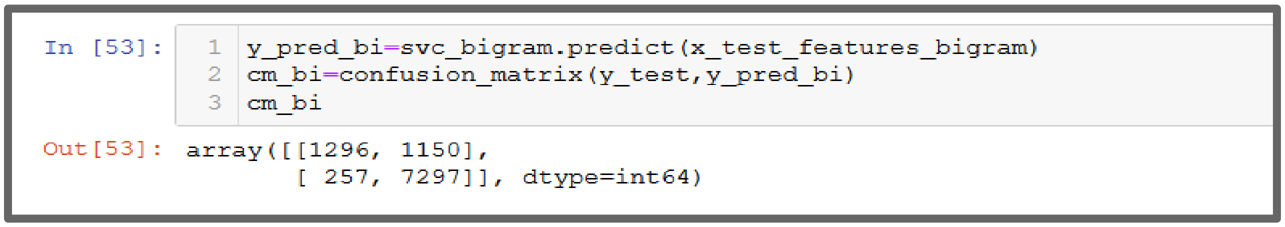
DT(bi-grams)



```
In [68]:  y_pred_bi=MultinomialNB_bigram_clf.predict(x_test_features_bigram)
          cm=confusion_matrix(y_test,y_pred_bi)
          cm

Out[68]: array([[1714,  638],
                [ 824, 6824]], dtype=int64)
```

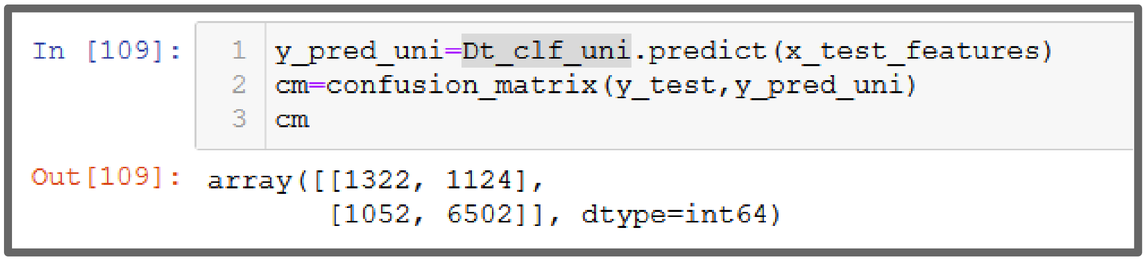
MNB(bi-grams)



```
In [53]:  y_pred_bi=svc_bigram.predict(x_test_features_bigram)
          cm_bi=confusion_matrix(y_test,y_pred_bi)
          cm_bi

Out[53]: array([[1296, 1150],
                [ 257, 7297]], dtype=int64)
```

SVC(bi-grams)

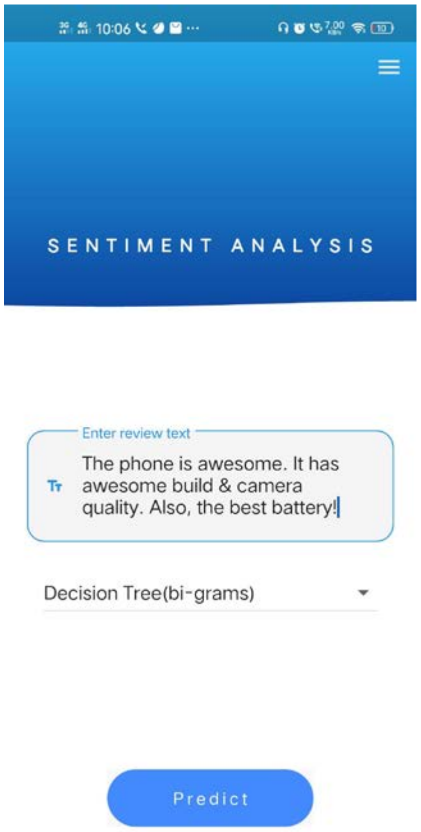


```
In [109]:  y_pred_uni=Dt_clf_uni.predict(x_test_features)
           cm=confusion_matrix(y_test,y_pred_uni)
           cm

Out[109]: array([[1322, 1124],
                 [1052, 6502]], dtype=int64)
```

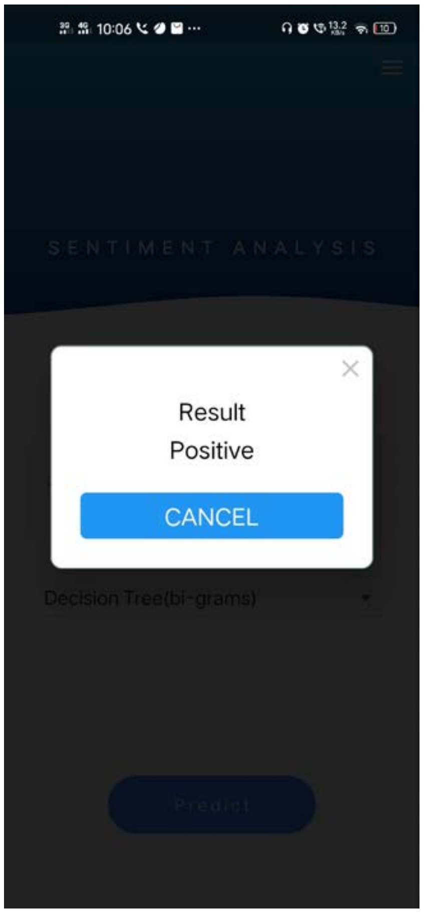
DT(uni-grams)

App design



The phone is awesome. It has awesome build & camera quality. Also, the best battery!

Decision Tree(bi-grams)

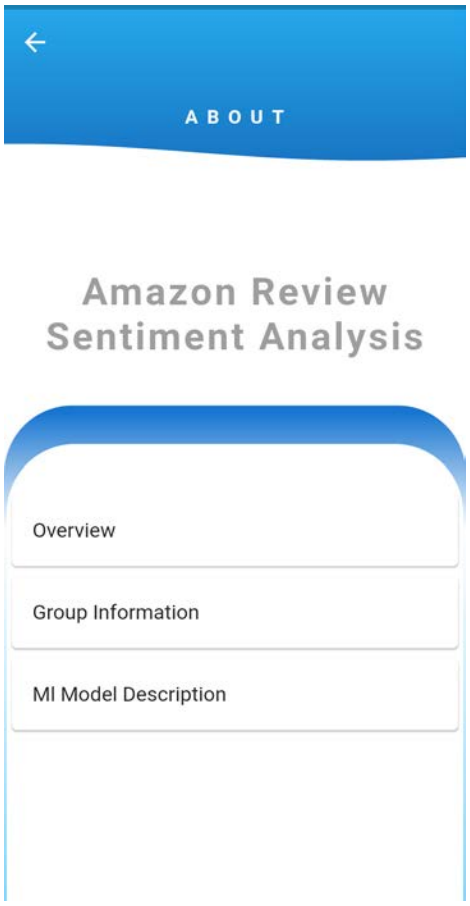


Result

Positive

Decision Tree(bi-grams)

Loading Page Home Page Result

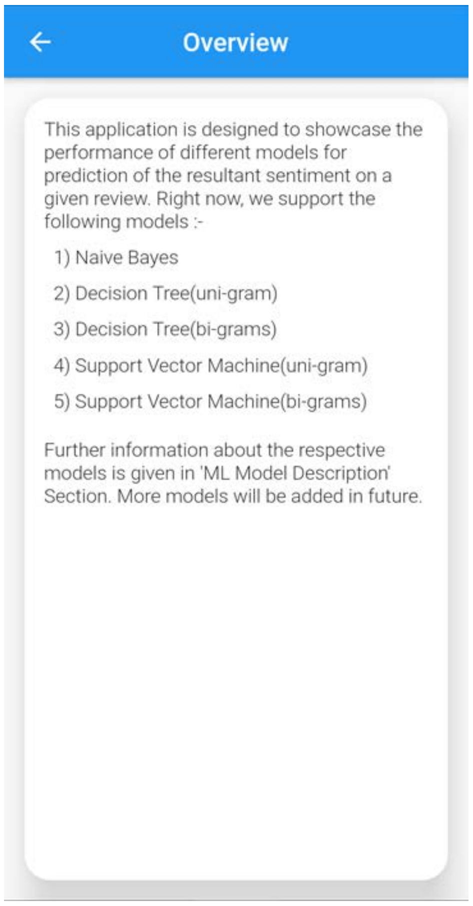


Amazon Review Sentiment Analysis

Overview

Group Information

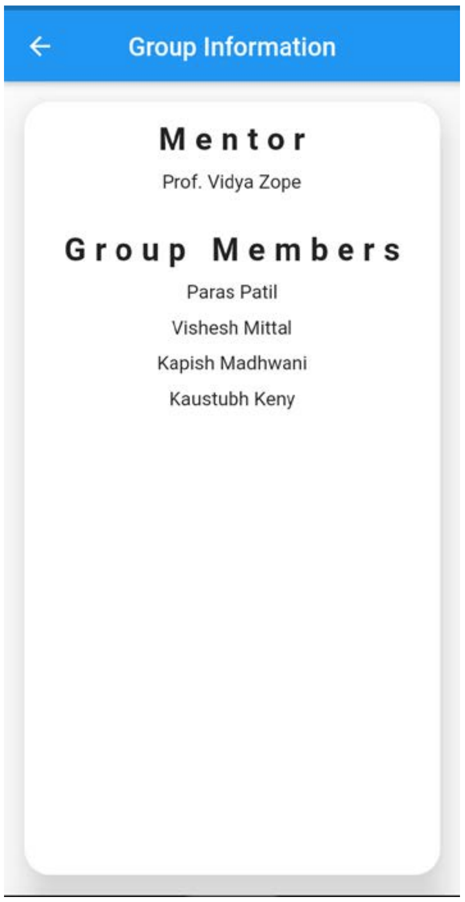
Ml Model Description


Overview

This application is designed to showcase the performance of different models for prediction of the resultant sentiment on a given review. Right now, we support the following models :-

1) Naive Bayes
2) Decision Tree(uni-gram)
3) Decision Tree(bi-grams)
4) Support Vector Machine(uni-gram)
5) Support Vector Machine(bi-grams)

Further information about the respective models is given in 'ML Model Description' Section. More models will be added in future.


Group Information

Mentor

Prof. Vidya Zope

Group Members

Paras Patil

Vishesh Mittal

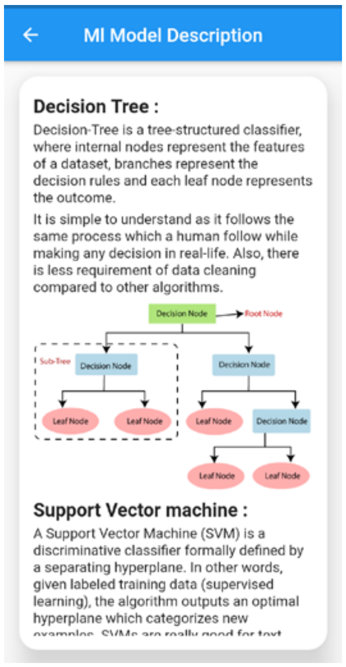
Kapish Madhwani

Kaustubh Keny

About Page

Overview Page

Group Information



Decision Tree :

Decision-Tree is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.

It is simple to understand as it follows the same process which a human follow while making any decision in real-life. Also, there is less requirement of data cleaning compared to other algorithms.

Support Vector machine :

A Support Vector Machine (SVM) is a discriminative classifier formally defined by a separating hyperplane. In other words, given labeled training data (supervised learning), the algorithm outputs an optimal hyperplane which categorizes new

Ml Model Description

5) Conclusion:

During our project work, we came with a few conclusions:

With the classification report, it was observed that DT has the lowest scores (0.78) while the rest of the models had a similar accuracy (0.86 - 0.87). However, in live testing it was seen that DT performs on par with SVM while the NLTK based Bernoulli's NB displayed the worst results. This may be because of the CV process involved in the data cleaning of SKLEARN based models. CV provided better weightages to the features which ended up improving performance.

Selection of appropriate Machine Learning models is essential since different models are suitable for different formats of data around the web in order to achieve the best results for our analysis.Tedious data cleaning processes can be automated and optimised using ML models, feature reduction, etc.

Python can be integrated along with flutter using Flask API which makes it easy to deploy ML models for public use via mobile applications.

6) FutureWork:

For the future scope of this project we sought to implement Hidden Markov Models algos. HMM are probabilistic models which have varied applications.We want to implement this technique for creating a PartOfSpeech tagger for our data cleaning unit. With that we would completely distance ourselves from the standard library used for Text and Speech cleaning NLTK.

Linking the application with data-base to store new reviews to implement reinforcement learning techniques to the model.

We are using LSTM,Neural Network and CNN model which is inside keras which is a high level API and is integrated as a part of Tensorflow 2.0

7) References:

[1] Vimala Balakrishnan & Wandeep Kaur , 23rd International Conference on Knowledge-Based and Intelligent Information & Engineering Systems , "**String-based Multinomial Naïve Bayes for Emotion Detection among Facebook Diabetes Community.**" Procedia Computer science , 159 , pp.30-37 (2019).

[2] Anchal Gupta and Satish Mahadevan Srinivasan , The Fifth Information Systems International Conference 2019 "**Constructing a heterogeneous training dataset for Emotion Classification.**" Procedia Computer Science , 161 , pp. 765-772 (2019)

[3] Amit Sharma ,Sandeep Chaurasia and Devesh Kumar Shrivastava, International Conference on Computational Intelligence and Data Science (ICCIDS 2019), "**Sentimental Short Sentences Classification by Using CNN DeepLearning Model with Fine Tuned Word2Vec**.",Procedia Computer Science 167, pp.1139-1147,2020.

[4] Ankita Sharma, Udayan Ghose ,International Conference on Smart Sustainable Intelligent Computing and Applications under ICITETM2020 , **"Sentimental Analysis of Twitter Data with respect to General Elections in India"** Procedia Computer Science Volume 173 , Pages 325-334 , 2020

[5] Sourav Kunal, Arijit Saha, Aman Varma, Vivek Tiwari , International Conference on Computational Intelligence and Data Science (ICCIDS 2018), **"Textual Dissection Of Live Twitter Reviews Using Naive Bayes"**,Procedia Computer Science Volume 132, Pages 307-313,2018

[6] Veny Amilia Fitri, Rachmadita Andreshwari and Muhammad Azani Hasibuan, "**Sentiment analysis of social media Twitter with case of Anti-LGBT campaign in Indonesia using Naive Bayes, Decision Tree & Random Firest Algorithm**," The Fifth Information Systems International Conference 2019, Procedia Computer Science, vol 161, pp. 765–772, 2019.

[7] Robert Dzisevič and Dmitrij Šešok, " **Text Classification using Different Feature Extraction Approaches,**" 2019 Open Conference of Electrical, Electronic and Information Sciences, 25-25 April 2019.

[8] Anton Borg and Martin Boldt., "**Using VADER Sentiment and SVM for Predicting Customer Response Sentiment**," Expert Systems with Applications, 2020.