

# LRTC LAB

IIIT HYDERABAD

Project Report

## SENTIMENT ANALYSIS

On English Language



INTERNATIONAL INSTITUTE OF  
INFORMATION TECHNOLOGY

HYDERABAD

Ponaganti Likitha Sri - S170414  
Gandipadala Sunil Kumar - S170904  
Baipalli Snehalatha - S170458  
Noolu Prasanna - S170191

# Abstract

*Sentiment Analysis also known as Opinion Mining refers to the use of natural language processing, text analysis to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine.*

*When we decide to purchase a product, we are likely to get the opinions from friends or relatives and do some surveys before we purchase the product. Hence, opinions are undeniably the key influencer of our behavior as well as the central to nearly all of the activities. Within the opinions, we often find the neutral, positive and negative polarities in the sentences. Based on the sentiment analysis taxonomy, it has opinion mining to have the opinion polarity classification, subjectivity detection, opinion spam detection, opinion summarization and argument expression detection. On the other hand, emotion mining has the emotion polarity classification, emotion detection, emotion cause detection and emotion classification. If it is based on granularity level, it has sentence level, document level and aspect/entity level of sentiment analysis. As for the machine learning approaches, it has semi-supervised learning, unsupervised learning and supervised learning of sentiment analysis.*

# Motivation

*The motivation for Sentiment Analysis is two-fold. Both consumers and producers highly value “customer’s opinion” about products and services. Thus, Sentiment Analysis has seen a considerable effort from industry as well as academia.*

## **The Consumer’s Perspective:**

While making a decision it is very important for us to know the opinion of the people around us. Earlier this group used to be small, with a few trusted friends and family members. But, now with the advent of the Internet we see people expressing their opinions in blogs and forums. These are now actively read by people who seek an opinion about a particular entity (product, movie etc.). Thus, there is a plethora of opinions available on the Internet. From a consumers’ point of view, extracting opinions about a particular entity is very important. Trying to go through such a vast amount of information to understand the general opinion is impossible for users just by the sheer volume of this data. Hence, the need for a system that differentiates between good reviews and bad reviews. Further, labeling these documents with their sentiment would provide a succinct summary to the readers about the general opinion regarding an entity.

## **The Producer’s Perspective:**

With the explosion of Web 2.0 platforms such as blogs, discussion forums, etc., consumers have at their disposal, a platform to share their brand experiences and opinions, positive or negative regarding any product or service. According to Pang and Lee (2008) these consumer voices can wield enormous influence in shaping the opinions of other consumers and, ultimately, their brand loyalties, their purchase decisions, and their own brand advocacy. Since the consumers have started using the power of the Internet to expand their horizons, there has been a surge of review sites and blogs, where users can perceive a product’s or service’s advantages and faults. These opinions thus shape the future of the product or the service. The vendors need a system that can identify trends in customer reviews and use them to improve their product or service and also identify the requirements of the future.

## **The Societies’ Perspective:**

Recently, certain events, which affected the Government, have been triggered using the Internet. The social networks are being used to bring together people so as to organize mass gatherings and oppose oppression. On the darker side, the social networks are being used to insinuate people against an ethnic group or class of people, which has resulted in a serious loss of life. Thus, there is a need for Sentiment Analysis systems that can identify such phenomena and curtail them if needed.

## Acknowledgment

*We would like to express special thanks & gratitude to our guide, Ashok Urlana who gave us this golden opportunity to work on this scalable project on the topic of “Sentiment Analysis”, which led us into doing a lot of Research which diversified our knowledge to a huge extent for which we are thankful.*

*We are grateful to Dr.Manish Shrivastava Sir, for providing an excellent internship opportunity and a congenial atmosphere for progressing with my project.*

*At the outset, I would like to thank LTRC Lab, IIIT Hyderabad for providing all the necessary resources for the successful completion of my course work. Last, but not the least I thank my project mates and other guides for their physical and moral support throughout the work.*

With Sincere Regards, **TEAM Sentiment Analysis for English dataset.**

|   |           |
|---|-----------|
| <b>SENTIMENT ANALYSIS</b>                     | <b>0</b>  |
| <b>Introduction</b>                           | <b>5</b>  |
| Sentiment Analysis                            | 6         |
| Classification and Its types                  | 7         |
| Applications of Sentiment Analysis            | 7         |
| <b>Related Work</b>                           | <b>9</b>  |
| Overview                                      | 9         |
| Challenges                                    | 10        |
| <b>Methods for Sentiment Analysis Model</b>   | <b>12</b> |
| Logistic Regression                           | 12        |
| Decision Tree                                 | 13        |
| Random Forest                                 | 14        |
| Support Vector Machine                        | 15        |
| Multilayer Perceptron                         | 16        |
| <b>Implementation of Sentiment Analysis</b>   | <b>17</b> |
| Python Implementation of logistic regression  | 18        |
| Implementation using Support Vector Machine   | 23        |
| Implementation using multi layer perceptron   | 24        |
| <b>Sentiment Analysis model terminologies</b> | <b>25</b> |
| Accuracy                                      | 25        |
| Precision                                     | 26        |
| Recall / Sensitivity                          | 27        |
| F1 Score                                      | 28        |

## Introduction

Sentiment is an attitude, thought, or judgment prompted by feeling. Sentiment analysis, which is also known as opinion mining, studies people's sentiments towards certain entities. From a user's perspective, people are able to post their own content through various social media, such as forums, micro-blogs, or online social networking sites. From a researcher's perspective, many social media sites release their application programming interfaces (APIs), prompting data collection and analysis by researchers and developers. However, those types of online data have several flaws that potentially hinder the process of sentiment analysis. The first flaw is that since people can freely post their own content, the quality of their opinions cannot be guaranteed. The second flaw is that the ground truth of such online data is not always available. A ground truth is more like a tag of a certain opinion, indicating whether the opinion is positive, negative, or neutral.

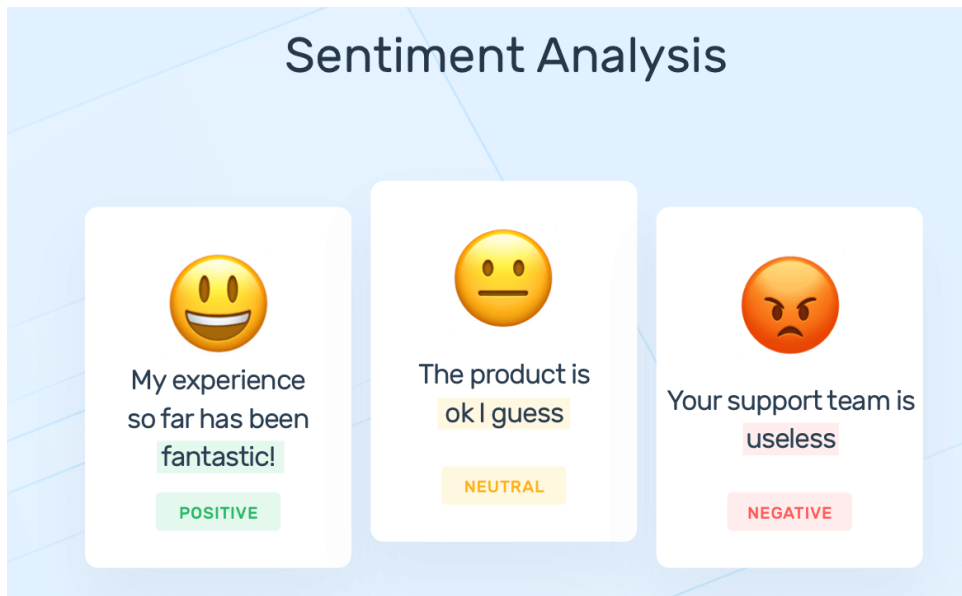
**“It is quite a boring movie..... but the scenes were good enough. ”**

The given line is a movie review that states that “it” (the movie) is quite boring but the scenes were good. Understanding such sentiments requires multiple tasks. Hence, SENTIMENTAL ANALYSIS is a kind of **text classification** based on Sentimental Orientation (SO) of opinion they contain. Sentiment analysis of product reviews has recently become very popular in text mining and computational linguistics research.

- Firstly, evaluative terms expressing opinions must be extracted from the review.
- Secondly, the SO, or the polarity, of the opinions must be determined.
- Thirdly, the opinion strength, or the intensity, of an opinion should also be determined.
- Finally, the review is classified with respect to sentiment classes, such as Positive and Negative, based on the SO of the opinions it contains.

## Sentiment Analysis

Sentiment analysis, also referred to as opinion mining, is a sub machine learning task where we want to determine which is the general sentiment of a given document. Using machine learning techniques and natural language processing we can extract the subjective information of a document and try to classify it according to its polarity such as positive, neutral or negative. It is a really useful analysis since we could possibly determine the overall opinion about a selling object, or predict stock markets for a given company like, if most people think positive about it, possibly its stock markets will increase, and so on. Sentiment analysis is actually far from being solved since the language is very complex (objectivity/subjectivity, negation, vocabulary, grammar,...) but it is also why it is very interesting to work on.



## Classification and Its types

A classification is a predictive modeling problem that predicts a class label for a given sample of input data. The goal of classification is to establish which category an observation belongs to, which is accomplished by understanding the relationship between the dependent and independent variables. There are several classification methods available for modeling classification predictive modeling challenges.

There are four major types of classification tasks you may encounter:

1. **Binary Classification** - The classification only will have 2 labels.
2. **Multi Class Classification** - The classification will have more than 2 class labels.
3. **Multi Label Classification** - The classification will have 2 or more class labels where every time one or more labels are predicted.
4. **Imbalanced Classification** - The no of examples in each class is unequally distributed.  
(A binary classifier).



## Applications of Sentiment Analysis

Sentiment Analysis has many applications in various fields. Sentiment regarding a product or service helps users in identifying their product of choice. Similarly, vendors build tools that analyze customer feedback which help improve user experience. The future might see applications wherein a system gauges the human emotion through sensory means and then creates an environment that helps improve human life in general. This section describes a few of these applications that have been built or are possibilities in the near future.

### **1.Social Media Marketing:**

Social media posts often contain some of the most honest opinions about your products, services, and businesses because they're unsolicited. With the help of sentiment analysis, you can wade through all that data in minutes, to analyze individual emotions and overall public sentiment on every social platform.

with the help of sentiment analysis software, you can wade through all that data in minutes, to analyze individual emotions and overall public sentiment on every social platform.

Sentiment analysis can read beyond simple definition to detect sarcasm, read common chat acronyms (lol, rofl, etc.), and correct for common mistakes like misused and misspelled words.

### **2. Customer Support:**

Customer Support Management represents many challenges due to the sheer number of requests, varied topics, and diverse branches within a company – not to mention the urgency of any given request.


Sentiment analysis with natural language understanding reads regular human language for meaning, emotion, tone, and more, to understand customer requests, just as a person would. You can automatically process customer support tickets, online chats, phone calls, and emails by sentiment to prioritize any urgent issues.

sentiment analysis classifier used to sort thousands of customer support messages instantly by understanding words and phrases that contain negative opinions.

### **3. Brand Monitoring and Reputation management:**

Brand monitoring is one of the most popular applications of sentiment analysis in business. Bad





reviews can snowball online, and the longer you leave them the worse the situation will be. With sentiment analysis tools, you will be notified about negative brand mentions immediately.

Not only that, you can keep track of your brand's image and reputation over time or at any given moment, so you can monitor your progress. Whether monitoring news stories, blogs, forums, and social media for information about your brand, you can transform this data into usable information and statistics.

#### **4. Listen to Voice Customer:**

Combine and evaluate all of your customer feedback from the web, customer surveys, chats, call centers, and emails. Sentiment analysis allows you to categorize and structure this data to identify patterns and discover recurring topics and concerns.

Listening to the voice of your customers, and learning how to communicate with your customers – what works and what doesn't – will help you create a personalized customer experience.

#### **5.Product Analysis:**

Find out what the public is saying about a new product right after launch, or analyze years of feedback you may have never seen. You can search keywords for a particular product feature (interface, UX, functionality) and use aspect based sentiment analysis to find only the information you need.

Discover how a product is perceived by your target audience, which elements of your product need to be improved, and know what will make your most valuable customers happy. All with sentiment analysis.

#### **6.Market and competitor research**

Use sentiment analysis for market and competitor research. Find out who's receiving positive mentions among your competitors, and how your marketing efforts compare. Analyze the positive language your competitors are using to speak to their customers and weave some of this language into your own brand messaging and tone of voice guide.

## Related Work

### Overview

From the detailed review of 13 research papers on sentiment analysis, we gather Information about datasets and model implementation. They have used various Machine Learning and Deep Learning models to accomplish the task of sentiment analysis. Mostly they have used Logistic Regression, Random Forest, Decision Tree, Support Vector Machine, Multilayer Perceptron models.

While reading the research papers we grasp the following elements :

1. Paper Reference
2. Paper Title ( Authors, Year )
3. Overview of the paper (Model and Approach)
4. Comparisions with Existing Softwares/models
5. Dataset Used with statistics
6. GitHub Repositories ( If exists )

- Sentiment analysis for numerous Foreign Languages, such as English, Arabic, and other European languages, have been built using diverse methodologies and by various researchers.

- Indian Languages have less Annotated data for sentiment analysis compared to foregin languages.

- **Models used :** Logistic Regression, Decision Trees, Random Forests, Support Vector Machine, Multilayer Perceptron model.

We collected the data needed for sentiment analysis. And we converted that data into a specified format. And we are implementing models with LR,RF,DT,SVM,MLP etc and Training those

models on the standard data.

### Sentiment analysis research references

1. <https://ieeexplore.ieee.org/document/6691379>
2. <https://ieeexplore.ieee.org/document/9083703>
3. <https://ieeexplore.ieee.org/document/9121057>
4. <https://ieeexplore.ieee.org/document/9498965>

### Challenges

Sentiment Analysis is a very challenging task. It requires a deep understanding of the problem. We discuss some of the challenges faced in Sentiment Analysis.

- **Identifying subjective portions of text:** The same word can be treated as subjective in one context, while it might be objective in some other. This makes it difficult to identify the subjective (sentiment-bearing) portions of text. For example: –
  - *The language of the author was very crude.*
  - *Crude oil is extracted from the sea beds.*

The same word “crude” is used as an opinion in the first sentence, while it is completely objective in the second sentence.

- **Associating sentiment with specific keywords:** Many sentences indicate an extremely strong opinion, but it is difficult to pinpoint the source of these sentiments. Hence an association to a keyword or phrase is extremely difficult. For example: –
  - *Every time I read ‘Pride and Prejudice’ I want to dig her up and beat her over the skull with her own shin-bone.*

In this example, “her” refers to the character in the book “Pride and Prejudice”, which is not explicitly mentioned. In such cases the negative sentiment must be associated with the character in the book.

- **Domain dependence:** The same sentence or phrase can have different meanings in different domains. The word *unpredictable* is positive in the domain of movies, but if the same word is used in the context of a vehicle’s steering, then it has a negative connotation.

- **Sarcasm Detection:** Sarcastic sentences express negative opinion about a target using positive words. For example: –

*Nice perfume. You must marinate in it.*

The sentence contains only positive words but still the sentence expresses a negative sentiment.

- **Thwarted expressions:** There are some sentences wherein a minority of the text determines the overall polarity of the document. Consider the following example: –

*This film should be brilliant. It sounds like a great plot, the actors are first grade, and the supporting cast is good as well, and Stallone is attempting to deliver a good performance. However, it can't hold up.*

Simple bag-of-words approaches will fail drastically in such cases, as most of the words used in here are positive, but the ultimate sentiment is negative.

- **Indirect negation of sentiment:** Sentiment can be negated in subtle ways as opposed to a simple no, not, etc. It is non-trivial to identify such negations easily. Consider the following example: –

*It avoids all cliches and predictability found in Hollywood movies.* While the words cliché and predictable bear a negative sentiment, the usage of “avoids” negates their respective sentiments.

- **Order dependence:** While in traditional text classification, the discourse structure does not play any role in classification, since the words are considered independent of each other, discourse analysis is essential for Sentiment Analysis/Opinion Mining. For example: –

*A is better than B, conveys the exact opposite opinion from, B is better than A.*

- **Entity Recognition:** Not everything in a text talks about the same entity. We need to separate out the text about a particular entity and then analyze its sentiment. Consider the following: –

*I hate Nokia, but I like Samsung.*

A simple bag-of-words approach will mark this as neutral, however, it carries a specific sentiment for both the entities present in the statement.

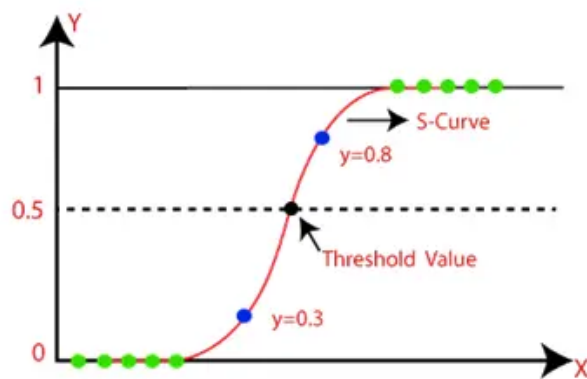
## Methods for Sentiment Analysis Model

### Logistic Regression

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

In logistic regression, we use the concept of the threshold value, which defines the probability of either 0 or 1. Such as values above the threshold value tends to 1, and a value below the threshold values tends to 0.

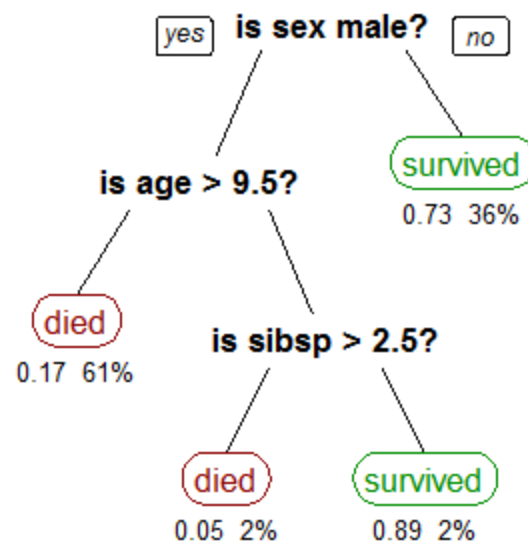


In Logistic regression, instead of fitting a regression line, we fit an "S" shaped logistic function, which predicts two maximum values (0 or 1). The curve from the logistic function indicates the likelihood of something such as whether the cells are cancerous or not, a mouse is obese or not based on its weight, etc.

## Decision Tree

Decision Tree is a **Supervised learning technique** that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It 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**.

In a Decision tree, there are two nodes, which are the **Decision Node** and **Leaf Node**. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.



In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of the root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and

moves further. It continues the process until it reaches the leaf node of the tree.

**Information Gain:** It calculates how much information a feature provides us about a class.

$$\text{Information Gain} = \text{Entropy}(S) - [(\text{Weighted Avg}) * \text{Entropy}(\text{each feature})]$$

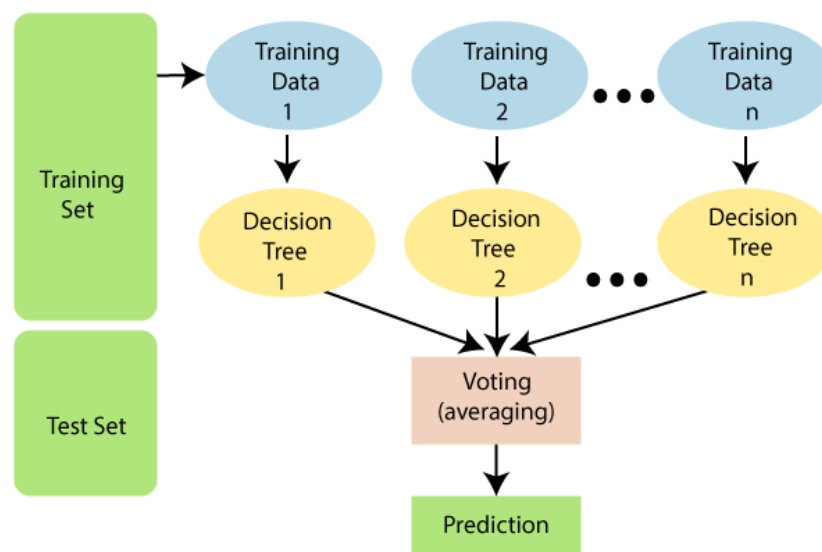
**Entropy:** Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data. Entropy can be calculated as:

$$\text{Entropy}(s) = -P(\text{yes}) \log_2 P(\text{yes}) - P(\text{no}) \log_2 P(\text{no})$$

## Random Forest

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of **ensemble learning**, which is a process of *combining multiple classifiers to solve a complex problem and to improve the performance of the model*.

Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.



Random Forest works in two-phase first is to create the random forest by combining N decision trees, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

**Step-1:** Select random K data points from the training set.

**Step-2:** Build the decision trees associated with the selected data points (Subsets).

**Step-3:** Choose the number N for decision trees that you want to build.

**Step-4:** Repeat Step 1 & 2.

**Step-5:** For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

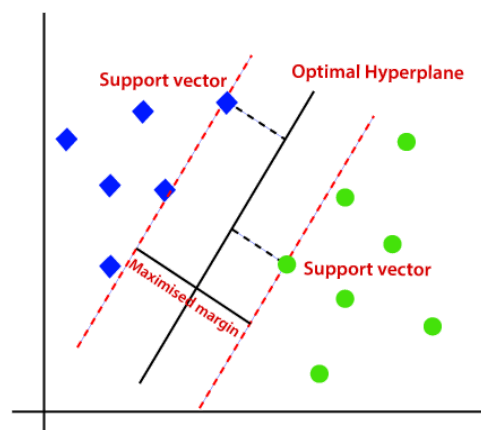
## Support Vector Machine

Support vector machines, so called as SVM is a supervised learning algorithm which can be used for classification and regression problems as support vector classification (SVC) and support vector regression (SVR). It is used for smaller datasets as it takes too long to process.

SVM is based on the idea of finding a hyperplane that best separates the features into different domains.

**Support Vectors:** These are the points that are closest to the hyperplane. A separating line will be defined with the help of these data points.

**Margin:** it is the distance between the hyperplane and the observations closest to the hyperplane (support vectors). In SVM large margin is considered a good margin. There are two types of margins: hard margin and soft margin. I will talk more about these two in the later section.





Suppose we have a dataset that has two classes (green and blue). We want to classify the new data point as either blue or green.

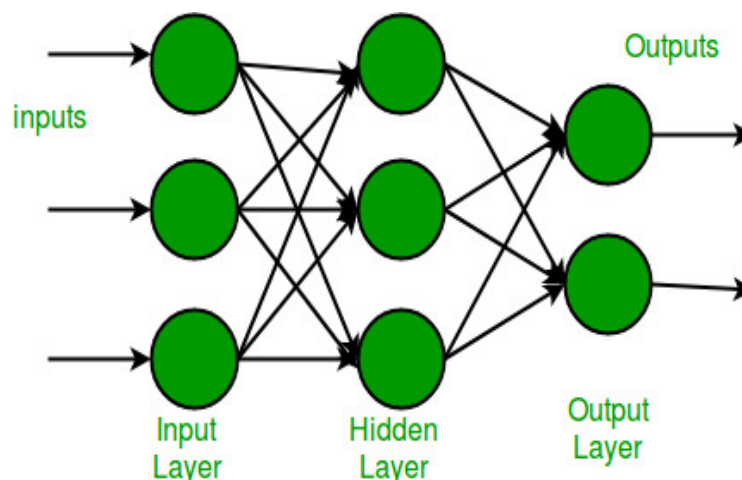
To classify these points, we can have many decision boundaries, but the question is which is the best and how do we find it?

The best hyperplane is that plane that has the maximum distance from both the classes, and this is the main aim of SVM. This is done by finding different hyperplanes which classify the labels in the best way then it will choose the one which is farthest from the data points or the one which has a maximum margin.

## Multilayer Perceptron

A Multilayer Perceptron has input and output layers, and one or more hidden layers with many neurons stacked together. And while in the Perceptron the neuron must have an activation function that imposes a threshold, like ReLU or sigmoid, neurons in a Multilayer Perceptron can use any arbitrary activation function.

A multi-layer perceptron has one input layer and for each input, there is one neuron(or node), it has one output layer with a single node for each output and it can have any number of hidden layers and each hidden layer can have any number of nodes.



The nodes in the input layer take input and forward it for further process, in the diagram above the nodes in the input layer forwards their output to each of the three nodes in the hidden layer, and in the same way, the hidden layer processes the information and passes it to the output layer.

Every node in the multilayer perceptron uses a sigmoid activation function. The sigmoid

activation function takes real values as input and converts them to numbers between 0 and 1 using the sigmoid formula.

$$\sigma(x) = 1 / (1 + \exp(-x))$$

Each layer is represented as  $y = f(W \cdot X + b)$ . Where  $f$  is the activation function (covered below),  $W$  is the set of parameters, or weights, in the layer,  $X$  is the input vector, which can also be the output of the previous layer, and  $b$  is the bias vector.

## Implementation of Sentiment Analysis

### Steps Involved In Implementation:

- Data Preparation
- Data Pre-Processing
- Fitting the Model to the training data
- Predicting the test results
- Calculating the test accuracy and performance(Confusion matrix)
- Visualize the the data

### Python Implementation of logistic regression

Data preparation and reading:

```
from sklearn import model_selection, preprocessing, linear_model, naive_bayes, metrics, svm
from sklearn.feature_extraction.text import TfidfVectorizer, CountVectorizer
from sklearn import decomposition, ensemble
import pandas as pd, numpy
```

#### ▼ Data Importing

```
df=pd.read_csv("data.csv")
```

```
[ ] df.head()
```

|   | Sentence   | Sentiment |
|---|--|-----------|
| 0 | The GeoSolutions technology will leverage Bene...    | positive  |
| 1 | \$ESI on lows, down \$1.50 to \$2.50 BK a real po... | negative  |
| 2 | For the last quarter of 2010 , Componenta 's n...    | positive  |
| 3 | According to the Finnish-Russian Chamber of Co...    | neutral   |
| 4 | The Swedish buyout firm has sold its remaining       | neutral   |

Counts of the reviews:

```
df["Sentiment"].value_counts()

neutral    3130
positive   1852
negative    860
Name: Sentiment, dtype: int64
```

Data Preprocessing:

# clean the data by doing removing stop words, punctuations and apply lemmatization

import string

punct = string.punctuation

#method for cleaning the data by removing the punctuations and stopwords

def text\_data\_cleaning(sentence):

doc = nlp(sentence)

tokens = []

for token in doc:

if token.lemma\_ != "-PRON-": #to check wheather the token is a pronoun or not?

temp = token.lemma\_.lower().strip()

else:

temp=token.lower\_

tokens.append(temp)

cleaned\_tokens = []



for token in tokens:

if token not in stopwords and token not in punct:

cleaned\_tokens.append(token)

return cleaned\_tokens

Splitting the data:

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test =
```

```
train_test_split(df['Sentence'],df['Sentiment'],test_size=0.3,random_state = 0,shuffle = True)
```

Label Encoding:

```
encoder = preprocessing.LabelEncoder()#this will help us to do it.
```

```
y_train = encoder.fit_transform(y_train)
```

```
y_test = encoder.fit_transform(y_test)
```

Feature Engineering:

```
# ngram level tf-idf
```

```
tfidf_vect_ngram = TfidfVectorizer(analyzer='word', token_pattern=r'\w{1,}',  
ngram_range=(2,3), max_features=5000)
```

```
tfidf_vect_ngram.fit(df["Sentence"])
```

```
xtrain_tfidf_ngram = tfidf_vect_ngram.transform(x_train)
```

```
xtest_tfidf_ngram = tfidf_vect_ngram.transform(x_test)
```

Training Model:

```
from sklearn.pipeline import Pipeline

from sklearn.linear_model import LogisticRegression

tfidf = TfidfVectorizer(tokenizer=
text_data_cleaning.strip_accents=None,lowercase=False,preprocessor=None)

lr_tfidf = Pipeline([('vect', tfidf),

                      ('classifier', LogisticRegression(random_state=0))])

lr_tfidf.fit(x_train,y_train)
```

Testing the Data:

```
#to predict we have a predict method in sklearn
y_pred = lr_tfidf.predict(x_test)

print(y_pred[0])
for i in range(5):
    print("--->test data--sentiment-->",y_pred[i])
```

```
2
--->test data--sentiment--> 2
--->test data--sentiment--> 1
--->test data--sentiment--> 1
--->test data--sentiment--> 2
--->test data--sentiment--> 1
```

Accuracy and classification Report:

```
[ ] #to compare or check the predicted data we have to import these
    from sklearn.metrics import classification_report, accuracy_score, confusion_matrix
```

```
▶ accuracy_score(y_test, y_pred)
```

```
0.677124928693668
```

```
[ ] print(classification_report(y_test, y_pred))
```

|              | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0            | 0.35      | 0.12   | 0.18     | 283     |
| 1            | 0.68      | 0.88   | 0.77     | 940     |
| 2            | 0.73      | 0.62   | 0.67     | 530     |
| accuracy     |           |        | 0.68     | 1753    |
| macro avg    | 0.59      | 0.54   | 0.54     | 1753    |
| weighted avg | 0.64      | 0.68   | 0.64     | 1753    |

We obtained an overall accuracy of model as 67%.

Implementation by Decision Tree:

Training the data:

## Importing the Decision Tree model to train

```
▶ from sklearn.tree import DecisionTreeClassifier # Import Decision Tree Classifier
   from sklearn.pipeline import Pipeline
   from sklearn.feature_extraction.text import TfidfVectorizer
```

```
] #creating pipeline object which will be follow the same sequences of execution
   pipeliner= Pipeline([("tfidf",TfidfVectorizer()),("classifier",DecisionTreeClassifier())])
   pipeliner=pipeliner.fit(x_train,y_train)
```

Testing the data:

Now take input from the user and test the data weather it is positive/negative/neutral

```
[ ] test1 = ["The GeoSolutions technology will leverage Benefon 's GPS solutions by providing Location Based Search Technology , a Communities Platform , location re
test2 = ["$ESI on lows, down $1.50 to $2.50 BK a real possibility"]
test3 = ["According to the Finnish-Russian Chamber of Commerce , all the major construction companies of Finland are operating in Russia ."]
test = ["Netflix has won the best selection of films","Hulu has a great UI","I dislike like the new crime series","I hate waiting for the next series to come ou

[ ] print(pipeline.predict(test1))
print(pipeline.predict(test2))
print(pipeline.predict(test3))
print(pipeline.predict(test))

['positive']
['negative']
['neutral']
['positive' 'neutral' 'negative' 'neutral']
```

We obtained an overall accuracy of model as 56%.

Implementation using Random Forest:

Training the Model:

### Building Model(Random Forest)

```
[ ] from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.ensemble import RandomForestClassifier
from sklearn.pipeline import Pipeline

[ ] #creating pipeline object which will be follow the same sequences of execution
pipeline= Pipeline([("tfidf",TfidfVectorizer()),("classifier",RandomForestClassifier(n_estimators=100))])

[ ] pipeline.fit(x_train,y_train)

Pipeline(steps=[('tfidf', TfidfVectorizer()),
                ('classifier', RandomForestClassifier())])
```

Testing the Model:

```
test1 = ["The GeoSolutions technology will leverage Benefon 's GPS solutions by providing Location Based Search Technology , a Communities Platform , location re
test2 = ["$ESI on lows, down $1.50 to $2.50 BK a real possibility"]
test3 = ["According to the Finnish-Russian Chamber of Commerce , all the major construction companies of Finland are operating in Russia ."]
test = ["Netflix has won the best selection of films","Hulu has a great UI","I dislike like the new crime series","I hate waiting for the next series to come ou

[ ] print(pipeline.predict(test1))
print(pipeline.predict(test2))
print(pipeline.predict(test3))
print(pipeline.predict(test))

['positive']
['negative']
['neutral']
['positive' 'neutral' 'neutral' 'neutral']
```

## Implementation using Support Vector Machine

Training the data:

▼ Importing the Decision Tree model to train

```
from sklearn.svm import SVC # Import SVM Classifier
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfVectorizer

[ ] #creating pipeline object which will be follow the same sequences of execution
    pipeliner= Pipeline([("tfidf",TfidfVectorizer()),("classifier",SVC())])
    pipeliner=pipeliner.fit(x_train,y_train)
```

Testing the data:

• Now take input from the user and test the data weather it is positive/negative/neutral

```
test1 = ["The GeoSolutions technology will leverage Benefon 's GPS solutions by providing Location Based Search Technology , a Communities Platform ,
test2 = ["$ESI on lows, down $1.50 to $2.50 BK a real possibility"]
test3 = ["According to the Finnish-Russian Chamber of Commerce , all the major construction companies of Finland are operating in Russia ."]
test = ["Netflix has won the best selection of films","Hulu has a great UI","I dislike like the new crime series","I hate waiting for the next series

[ ] print(pipeliner.predict(test1))
    print(pipeliner.predict(test2))
    print(pipeliner.predict(test3))
    print(pipeliner.predict(test))

['positive']
['negative']
['neutral']
['positive' 'positive' 'neutral' 'neutral']
```

We obtained the accuracy as 68%.

## Implementation using multi layer perceptron

Training the data:

```
#creating pipeline object which will be follow the same sequences of execution
#Initializing the MLPClassifier
classifier = MLPClassifier(hidden_layer_sizes=(150,100,50), max_iter=300,activation = 'relu',solver='adam',random_state=1)
pipeliner= Pipeline([("tfidf",TfidfVectorizer()),("classifier",classifier)])
pipeliner=pipeliner.fit(x_train,y_train)
```



Testing the data:

- Now take input from the user and test the data whether it is positive/negative/neutral

```
test1 = ["The GeoSolutions technology will leverage Benefon 's GPS solutions by providing Location Based Search Technology , a Communities Platform , locati  
test2 = ["$ESI on lows, down $1.50 to $2.50 BK a real possibility"]  
test3 = ["According to the Finnish-Russian Chamber of Commerce , all the major construction companies of Finland are operating in Russia ."]  
test = ["Netflix has won the best selection of films","Hulu has a great UI","I dislike like the new crime series","I hate waiting for the next series to co  
[ ] print(pipeline.predict(test1))  
print(pipeline.predict(test2))  
print(pipeline.predict(test3))  
print(pipeline.predict(test))  
['positive']  
['negative']  
['neutral']  
['positive' 'positive' 'neutral' 'neutral']
```

Accuracy Score:

```
[ ] y_pred = pipeline.predict(x_test)  
[ ] from sklearn.metrics import classification_report,accuracy_score,confusion_matrix  
accuracy_score(y_test,y_pred)  
0.6550308008213552  
[ ] print(classification_report(y_test,y_pred))  
precision    recall  f1-score   support  
negative     0.30     0.32     0.31         213  
neutral      0.72     0.70     0.71         780  
positive     0.72     0.73     0.72         468  
accuracy                    0.66        1461  
macro avg     0.58     0.58     0.58        1461  
weighted avg     0.66     0.66     0.66        1461
```

We obtained the accuracy score as 65%

## Sentiment Analysis model terminologies

1. Accuracy
2. Precision
3. Recall
4. F-1 score

## Accuracy

The base metric used for model evaluation is often *Accuracy*, describing the number of correct predictions over all predictions:

$$\frac{\text{True Positives} + \text{True Negatives}}{\text{True Positives} + \text{True Negatives} + \text{False Positives} + \text{False Negatives}} = \frac{\text{N. of Correct Predictions}}{\text{N. of all Predictions}} = \frac{\text{N. of Correct Predictions}}{\text{Size of Dataset}}$$

These three show the same formula for calculating accuracy, but in different wording. From more formalized to more intuitive (my opinion).

For example we have TP=55, TN=15, Dataset=100 then

$$(\text{TP} + \text{TN}) / \text{DatasetSize} = (55 + 15) / 100 = 0.7 = 70\%.$$

This is perhaps the most intuitive of the model evaluation metrics, and thus commonly used. But often it is useful to also look a bit deeper.

## Precision

*Precision* is a measure of how many of the positive predictions made are correct (true positives).

Formula:

$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} = \frac{\text{N. of Correctly Predicted Positive Instances}}{\text{N. of Total Positive Predictions you Made}} = \frac{\text{N. of Correctly Predicted People with Cancer}}{\text{N. of People you Predicted to have Cancer}}$$

For example, we have


TP=35, TN=25, Dataset=100, FP=30 then

Precision =  $35 / (35 + 30) = 35 / 65 = 0.53 = 53\%$ .

## Recall / Sensitivity

*Recall* is a measure of how many of the positive cases the classifier correctly predicted, over all the positive cases in the data. [It is sometimes also referred to as \*Sensitivity\*.](#)

Formula:



$$\frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} = \frac{\text{N. of Correctly Predicted Positive Instances}}{\text{N. of Total Positive Predictions you Made}} = \frac{\text{N. of Correctly Predicted People with Cancer}}{\text{N. of People you Predicted to have Cancer}}$$

For example we have TP=55,TN=20,Dataset=100 ,FN=18 then


$$\text{Recall} = 55 / (55 + 18) = 55 / 73 = 0.75 = 75\%.$$

## F1 Score

F1 Score is the Harmonic Mean between precision and recall. The range for the F1 Score is [0, 1]. It tells you how precise your classifier is (how many instances it classifies correctly), as well as how robust it is (it does not miss a significant number of instances). High precision but lower recall, gives you an extremely accurate, but it then misses a large number of instances that are difficult to classify. The greater the F1 Score, the better is the performance of our model. Mathematically, it can be expressed as :

F1 Score tries to find the balance between precision and recall.

Formula:


$$2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}$$

For example, a Precision of 0.15 and Recall of 1.30 would give :

an arithmetic mean of  $(0.15+1.30)/2=0.72$ ,

F1-score score (formula above) of  $2*(0.15*1.30)/(0.15+1.30)=\sim 0.26$ .