University School of Management Studies

Academic Year: 2021~22

Project on:

Amazon product review Sentimental Analysis

Post Graduate Diploma

In

Data Analytics

Under supervision of

Mr. Santanoo Pattnaik

Submitted By:-
Nishant

01416640621

<u>Certificate</u>

This is to certify that the dissertation entitled **“Amazon product review Sentimental Analysis”** is the bona fide research work carried out by Nishant students of PGDDA, at University School of Management Studies, Guru Gobind Singh Indraprastha University during the year 2021-2022, in partial fulfilment of the requirements for the award of the degree of Post Graduate Diploma in Data Analytics and that the dissertation has not formed the basis for the award previously of any degree or diploma to the best of my knowledge and belief.

Dr Santanoo Pattnaik

USMS

GGSIPU

Declaration

I hereby declare that this project work entitled "**Amazon product review Sentimental Analysis**" is a record of an original work done by us under the guidance of Dr. Santanoo Pattnaik and this project is submitted in partial fulfilment of the requirements for the award of the degree of Post Graduate Diploma in Data Analytics. The results embodied in this thesis have not been submitted to any other university or institute for the award of any degree or diploma.

Name: Nishant

Enrollment: 01416640621

Course: PGDDA2021~22

Acknowledgement

Materials in this report have been drawn from a wide variety of sources including weekly journals, books, magazine and internet. Sources of these works are cited where they are discussed in the text, and I hope that I have made no omissions.

I am deeply indebted to our project in-charge **Mr. SANTANOO PATNAIK** without whose support and encouragement, this project couldn’t have been accomplished.

I sincerely extend our deep felt regards for their able guidance throughout this training period, for providing us with eminent environment and essential resources to complete our project work.

Table of Contents

Abstract

The world we see nowadays is becoming more digitalized. In this digitalized world e-commerce is taking the ascendancy by making products available within the reach of customers where the customer doesn't have to go out of their house. As now a day's people are relying on online products so the importance of a review is going higher. For selecting a product, a customer needs to go through thousands of reviews to understand a product. But in this prospering day of machine learning, going through thousands of reviews would be much easier if a model is used to polarize those reviews and learn from it. We used supervised learning method on a large scale amazon dataset to polarize it and get satisfactory accuracy.

Introduction

As the commercial site of the world is almost fully undergone in online platform people is trading products through different e-commerce website. And for that reason reviewing products before buying is also a common scenario. Also now a day, customers are more inclined towards the reviews to buy a product. So analyzing the data from those customer reviews to make the data more dynamic is an essential field nowadays. In this age of increasing machine learning based algorithms reading thousands of reviews to understand a product is rather time consuming where we can polarize a review on particular category to understand its popularity among the buyers all over the world.

The objective of this paper is to categorize the positive and negative feedbacks of the customers over different products and build a supervised learning model to polarize large amount of reviews. A study on amazon last year revealed over 88% of online shoppers trust reviews as much as personal recommendations. Any online item with large amount of positive reviews provides a powerful comment of the legitimacy of the item. Conversely, books, or any other online item, without reviews puts potential prospects in a state of distrust. Quite simply, more reviews look more convincing. People value the consent and experience of others and the review on a material is the only way to understand others impression on the product. Opinions, collected from users‟ experiences regarding specific products or topics, straightforwardly influence future customer purchase decisions. Similarly, negative reviews often cause sales loss [2]. For those understanding the feedback of customers and polarizing accordingly over a large amount of data is the goal. There are some similar works done over amazon dataset. In [5] did opinion mining over small set of dataset of Amazon product reviews to understand the polarized attitudes towards the products.

Literature review

This section exhibits the study of the traditional schemes which has been analyzed the polarization-based on customer review posted on Amazon eCommerce website.

Kunpang Zhang et al. introduced compositional sentiment rules to calculate the degree of textual sentiment. The system resulted in a crisp form for avoiding the machine learning approach.

Yu Cheng et al. classified the dataset using the support vector machine for extracting the feature map. Meanwhile, the system faced a problem when the sarcastic sentence passed through their model for using linear regression.

In a research work of Bo Pang et al., the movie reviews categorized into the binary classifier by applying support vector machine and Naive Bayes classifier. The accuracy of the Naive Bayes models has been increased to 81% whereas the SVM models are extended from 72% to 82%.

In a study work of Kian Kenyon-Dean et al., they introduced a method to identify the ambiguous data. But, their model had a serious issue on recognizing the complicated human sentiment.

Nurulhuda et al. presented the process of selecting feature based on chi-square statistics value help reducing the dimensionality and categorized the dataset using a support vector machine. Conversely, no comparison has been presented on the work.

In a research work of Georgios Paltoglou et al., they explored whether more advanced information retrieval feature weighting systems could improve classification accuracy. But they considered only a thousand features for sentiment analysis.

In addition to exploring the use of a tree kernel to overcome the need for tedious feature engineering, Apoorv Agarwal et al. introduced POS-specific previous polarity characteristics. But their precision result is not up to the mark.

Rui Xia et al. made a comparative study of the effectiveness of the ensemble technique for sentiment classification in a research job. The ensemble framework is applied to assignments of sentiment classification, to incorporate distinct function sets and classification algorithms effectively to synthesize a more precise classification process. On the contrary, they did not create a table of comparison between the classification of SVM and Naive Bayes.

In a research work of Geetika Gautam et al., they analyzed the polarization of the sentiment on twitter dataset bt machine learning approaches. Eventually, their accuracy should be increased further by selecting other features and a well-preprocessing technique.

In a research work of Tirath Prasad Sahu et al., they examined the sentiment eloquence to analyze the polarity of the film review on a scale of 0(highly disliked) to 4(highly liked) and implemented feature extraction and ranking and applied these features to train their multilabel classifier to categorize the movie review into its correct label. The comparison between the two approaches leading to machine learning was not mentioned.

Statement of Problem

Sentiment analysis or opinion mining is the computational study of people's opinions, sentiments, emotions, appraisals, and attitudes towards entities such as products, services, organizations, individuals, issues, events, topics, and their attributes. Nowadays, if one wants to buy a consumer product, one is no longer limited to asking one's friends and family for opinions because there are many user reviews and discussions about the product in public forums on the Web. The most common way of selecting the product from the wide variety available on an ecommerce platform like website is by going through the reviews given by people.

But, it is still a hectic task to read thousands of review and interpreting the intentions of the reviewer. Sentiment classification aims to determine the overall intention of a written text which can be of admiration or criticism type. This can be achieved by using classification type machine learning algorithms such as Logistic Regression. This paper is an attempt to analyze the natural language based reviews available on ecommerce website to classify them in two categories namely positive and negative and generating a model that can predict whether a model is of positive intent or of negative intend.

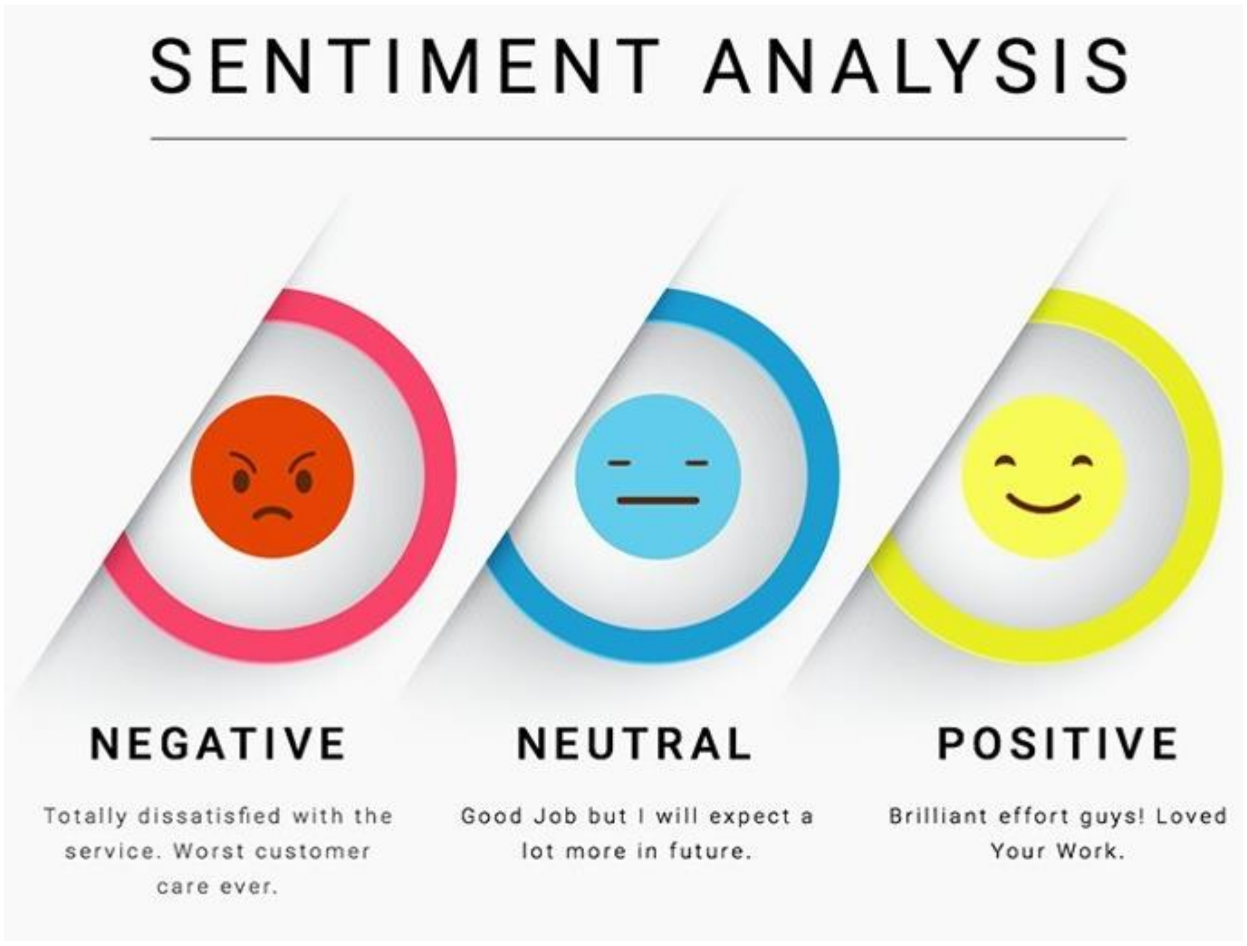
Objective of the study

Regardless of the name, the goal of sentiment analysis is the same: to know a user or audience opinion on a target object by analyzing a vast amount of text from various sources. In this paper we have the following objectives:

1. Text analysis using NLP Algorithms and Machine learning Classification algorithms to find the sentiments (Positive or Negative) of the reviews given for different products on Amazon website.
2. Compare different Machine Learning Algorithms and find the optimum classification model for given dataset.

Understanding the Terminology

Sentimental analysis: Sentiment analysis (also known as opinion mining or emotion AI) is the use of natural language processing, text analysis, computational linguistics, and biometrics to systematically identify, extract, quantify, and study affective states and subjective information. Sentiment analysis is widely applied to voice of the customer materials such as reviews and survey responses, online and social media, and healthcare materials for applications that range from marketing to customer service to clinical medicine. With the rise of deep language models, such as RoBERTa, also more difficult data domains can be analyzed, e.g., news texts where authors typically express their opinion/sentiment less explicitly.


SENTIMENT ANALYSIS
NEGATIVE
Totally dissatisfied with the service. Worst customer care ever.
NEUTRAL
Good Job but I will expect a lot more in future.
POSITIVE
Brilliant effort guys! Loved Your Work.

Natural Language Processing: Natural language processing (NLP) is a subfield of linguistics, computer science, and artificial intelligence concerned with the interactions between computers and human language, in particular how to program computers to process and analyze large amounts of natural language data. The goal is a computer capable of "understanding" the contents of documents, including the contextual nuances of the language within them. The technology can then accurately extract information and insights contained in the documents as well as categorize and organize the documents themselves.

<u>Text Mining</u>: Text mining, also referred to as text data mining, similar to text analytics, is the process of deriving high-quality information from text. It involves "the discovery by computer of new, previously unknown information, by automatically extracting information from different written resources. Text analysis involves information retrieval, lexical analysis to study word frequency distributions, pattern recognition, tagging/annotation, information extraction, data mining techniques including link and association analysis, visualization, and predictive analytics. The overarching goal is, essentially, to turn text into data for analysis, via application of natural language processing (NLP), different types of algorithms and analytical methods. An important phase of this process is the interpretation of the gathered information.

<u>Machine learning in Text Mining</u>: Machine learning aims to develop an algorithm in order to optimize the performance of the system by using example data. The solution that machine learning provides for sentiment analysis involves two main steps. The first step is to “learn” the model from the training data and the second step is to classify the unseen data with the help of the trained model .Machine learning algorithms can be classified in different categories:

i. supervised learning
ii. semi-supervised learning
iii. unsupervised learning

a. In Supervised learning the process where the algorithm is learning from the training data can be seen as a teacher supervising the learning process of its students. The supervisor is somehow teaching the algorithm what conclusions it should come up with as an output. So, both input and the desired output data are provided. It is also required that the training data is already labeled. If the classifier gets more labeled data, the output will be more precise. The goal of this approach is that the algorithm can correctly predict the output for new input data. If the output were widely different from the expected result, the supervisor can guide the algorithm back to the right path. There are however some challenges involved when working with supervised. The supervised learning works fine as long as the labelled data is provided. This means that if the machine faces unseen data, it will either give wrong class label after classification or remove it because it has not ”learnt” how to label it.

b. Unsupervised learning in difference with supervised learning is trained on unlabeled data with no corresponding output. The algorithm should find out the underlying structure of the data set on its own. This means that it has to discover similar patterns in the data to determine the output without having the right answers. One of the most important methods in unsupervised learning problems is clustering. Clustering is simply the method of identifying similar groups of data in the data set. For sentiment classification in an unsupervised manner it is usually the sentiment words and phrases that are used. This means that the classification of a review is predicted based on the average semantic orientation of the phrases in that review. This is obvious since the dominating factor for sentiment classification is often the sentiment words.

c. Finally, Semi-supervised learning which has the benefit of both supervised and unsupervised learning, refers to problems in which a smaller amount of data is labelled, and the rest of the training data set is unlabeled. This is useful for when collecting data can be cheap but labelling it can be time consuming and expensive. This approach is highly favorable both in theory and practice because of the fact that having lots of unlabeled data during the training process tends to improve the accuracy of the final model while building it requires much less time and cost. In a study by Dasgupta and Vincent Ng. (2009) a semi-supervised learning was experimented where they used 2000 documents as unlabeled data and 50 randomly labeled documents.

Applications of Sentimental analysis

Social media monitoring: 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 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.

Comment 1

"Love the user interface. Setup took five minutes and we were ready to go."

Comment 2

"Took me 2 hours to set up, then I find out I have to update my OS. Love it!"

Sentiment analysis would classify the second comment as negative, even though they both use words that, without context, would be considered positive. Keeping track of customer comments allows you to engage with customers in real time. You'll be able to quickly respond to negative or positive comments, and get regular, dependable insights about your customers, which you can use to monitor your progress from one quarter to the next.

Customer support: Customer support management presents 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 (NLU) 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. Try out our sentiment analysis classifier to see how sentiment analysis could be used to sort thousands of customer support messages instantly by understanding words and phrases that contain negative opinions.

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. You can also trust machine learning to follow trends and anticipate outcomes, to stay ahead and go from reactive to proactive.

Listen to voice of the customer (VoC): 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.

Listen to your employees: By analyzing the sentiment of employee feedback, you'll know how to better engage your employees, reduce turnover, and increase productivity. Use sentiment analysis to evaluate employee surveys or analyze Glassdoor reviews, emails, Slack messages, and more. Process unstructured data to go beyond who and what to uncover the why – discover the most common topics and concerns to keep your employees happy and productive.

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.

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.

Understanding the Data

Data Source: The data was downloaded from kaggle through the link:

https://www.kaggle.com/snap/amazon-fine-food-reviews

Tools used:

1. R-Studio

- When you see powerful analytics, statistics, and visualizations used by data scientists and business leaders, chances are that the R language is behind them.
- Open-source R is the statistical programming language that data experts the world over use for everything from mapping broad social and marketing trends online to developing financial and climate models that help drive our economies and communities.
- R was first implemented in the early 1990's by Robert Gentleman and Ross Ihaka, both faculty members at the University of Auckland. Robert and Ross established R as an open-source project in 1995.
- RStudio provides open source and enterprise-ready professional software for data science.
- R-Studio is used to make the word cloud during initial data exploration.

2. Python Programming Language

- Python is a high-level, interpreted, general-purpose programming language. Its design philosophy emphasizes code readability with the use of significant indentation.
- Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including structured (particularly procedural), object-oriented and functional programming. It is often described as a "batteries included" language due to its comprehensive standard library.
- Guido van Rossum began working on Python in the late 1980s as a successor to the ABC programming language and first released it in 1991 as Python 0.9.0.[34] Python 2.0 was released in 2000 and introduced new features such as list comprehensions, cycle-detecting garbage collection, reference counting, and Unicode support. Python 3.0, released in 2008, was a major revision that is not completely backward-compatible with earlier versions. Python 2 was discontinued with version 2.7.18 in 2020.

- All the data pre-processing and exploration is done using python.

Libraries used in R.

Tm package

Tm provides a set of predefined sources, e.g., DirSource, VectorSource, or DataframeSource, which handle a directory, a vector interpreting each component as document, or data frame like structures (like CSV files), respectively. The tm package utilizes the Corpus as its main structure. A corpus is simply a collection of documents, but like most things in R, the corpus has specific attributes that enable certain types of analysis. Corpora in R exist in two ways:

- Volitile Corpus (VCorpus) is a temporary object within R and is the default when assigning documents to a corpus.
- Permanent Corpus (PCorpus) is a permanent object that can be stored outside of R.

It is important to note that differences between a VCorpus and PCorpus will not affect operations covered in this tutorial–only how the corpus is stored.

SnowballC

An R interface to the C 'libstemmer' library that implements Porter's word stemming algorithm for collapsing words to a common root to aid comparison of vocabulary. Currently supported languages are Danish, Dutch, English, Finnish, French, German, Hungarian, Italian, Norwegian, Portuguese, Romanian, Russian, Spanish, Swedish and Turkish.

Wordcloud

The procedure of creating word clouds is very simple in R if you know the different steps to execute. The text mining package (tm) and the word cloud generator package (wordcloud) are available in R for helping us to analyze texts and to quickly visualize the keywords as a word cloud.

RColorBrewer

RColorBrewer is an R packages that uses the work from http://colorbrewer2.org/ to help you choose sensible colour schemes for figures in R. The colors are split into three group, sequential, diverging, and qualitative.

Sequential – Light colours for low data, dark for high data

Diverging – Light colours for mid-range data, low and high contrasting dark colours

Qualitative – Colours designed to give maximum visual difference between classes

Libraries used in Python.

Pandas

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. It is free software released under the three-clause BSD license. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals. Its name is a play on the phrase "Python data analysis" itself. Wes McKinney started building what would become pandas at AQR Capital while he was a researcher there from 2007 to 2010.

Features of pandas

- DataFrame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and subsetting of large data sets.
- Data structure column insertion and deletion.
- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.
- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Time series-functionality: Date range generation[7] and frequency conversions, moving window statistics, moving window linear regressions, date shifting and lagging.
- Provides data filtration.

Numpy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors. NumPy is a NumFOCUS fiscally sponsored project.

Features of Numpy

- NumPy targets the CPython reference implementation of Python, which is a non-optimizing bytecode interpreter. Mathematical algorithms written for this version of Python often run much slower than compiled equivalents due to the absence of compiler optimization. NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays; using these requires rewriting some code, mostly inner loops, using NumPy.
- Using NumPy in Python gives functionality comparable to MATLAB since they are both interpreted,[21] and they both allow the user to write fast programs as long as most operations work on arrays or matrices instead of scalars. In comparison, MATLAB boasts a large number of additional toolboxes, notably Simulink, whereas NumPy is intrinsically integrated with Python, a more modern and complete programming language. Moreover, complementary Python packages are available; SciPy is a library that adds more MATLAB-like functionality and Matplotlib is a plotting package that provides MATLAB-like plotting functionality. Internally, both MATLAB and NumPy rely on BLAS and LAPACK for efficient linear algebra computations.
- Python bindings of the widely used computer vision library OpenCV utilize NumPy arrays to store and operate on data. Since images with multiple channels are simply represented as three-dimensional arrays, indexing, slicing or masking with other arrays are very efficient ways to access specific pixels of an image. The NumPy array as universal data structure in OpenCV for images, extracted feature points, filter kernels and many more vastly simplifies the programming workflow and debugging.

Matplotlib

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

Matplotlib was originally written by John D. Hunter. Since then it has an active development community and is distributed under a BSD-style license. Michael Droettboom was nominated as matplotlib's lead developer shortly before John Hunter's death in August 2012 and was further joined by Thomas Caswell. Matplotlib is a NumFOCUS fiscally sponsored project.

Scikit Learn

Scikit-learn (formerly scikits.learn and also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy. Scikit-learn is a NumFOCUS fiscally sponsored project.

Scikit-learn integrates well with many other Python libraries, such as Matplotlib and plotly for plotting, NumPy for array vectorization, Pandas dataframes, SciPy, and many more.

NLTK

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical natural language processing (NLP) for English written in the Python programming language. It was developed by Steven Bird and Edward Loper in the Department of Computer and Information Science at the University of Pennsylvania. NLTK includes graphical demonstrations and sample data. It is accompanied by a book that explains the underlying concepts behind the language processing tasks supported by the toolkit, plus a cookbook.

NLTK is intended to support research and teaching in NLP or closely related areas, including empirical linguistics, cognitive science, artificial intelligence, information retrieval, and machine learning. NLTK has been used successfully as a teaching tool, as an individual study tool, and as a platform for prototyping and building research systems. There are 32 universities in the US and 25 countries using NLTK in their courses. NLTK supports classification, tokenization, stemming, tagging, parsing, and semantic reasoning functionalities.

Getting the glimpse of the data

```
df = pd.read_csv('C:/Users/welcome/Desktop/PGDDA/Sem-2/Project/Reviews.csv')
```

```
df.head()
```

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...

```
df2.shape
```

```
(74774, 10)
```

The dataset consist of 74774 entries for 10 columns as detailed

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 74774 entries, 0 to 74773
Data columns (total 10 columns):
 #   Column                  Non-Null Count  Dtype 
---  ------                  --------------  ----- 
 0   Id                      74774 non-null  int64 
 1   ProductId               74774 non-null  object
 2   UserId                  74774 non-null  object
 3   ProfileName             74770 non-null  object
 4   HelpfulnessNumerator    74774 non-null  int64 
 5   HelpfulnessDenominator  74774 non-null  int64 
 6   Score                   74774 non-null  int64 
 7   Time                    74774 non-null  int64 
 8   Summary                 74772 non-null  object
 9   Text                    74774 non-null  object
dtypes: int64(5), object(5)
memory usage: 5.7+ MB
```

In the data our target variable is score (ratings given by customers) having values on likert scale of 1 to 5. And feature variable in the form of text data in the column text (comments given by customers).

Checking for null values

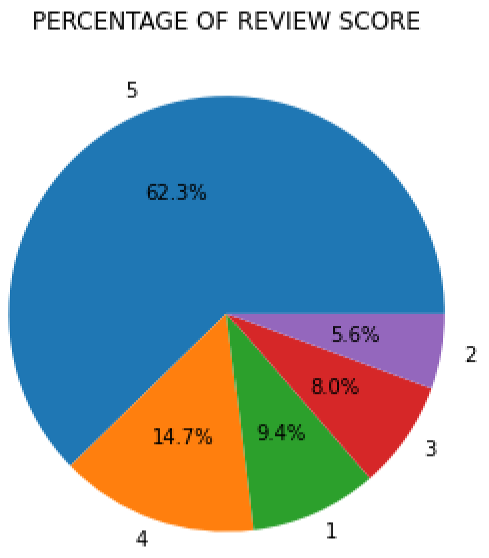
```
df1.isnull().sum()          # to check null values in the columns
```

```
HelpfulnessNumerator      0
HelpfulnessDenominator    0
Score                     0
Summary                   2
Text                      0
dtype: int64
```

Null values present in the dataset was dropped before further analysis.

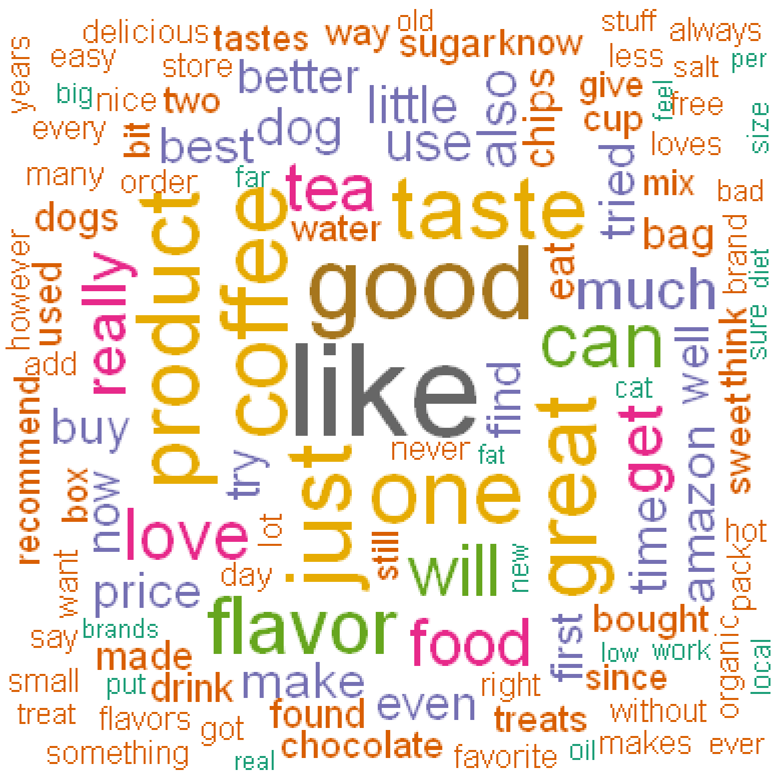
Distribution of the ratings is as given below.

```
plt.figure(figsize=(5,5))
plt.pie(rev,labels=rev.keys(),autopct='%0.1f%%')         # check the percentage of different feedbacks star given by customers
plt.title('PERCENTAGE OF REVIEW SCORE')
plt.show()
```



It can be interpreted from the above figure that 5 star ratings (score) is mostly given which means the products are liked by the customers.

Getting a glimpse of the most repeated words using 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.

The reasons one should use word clouds to present the text data are:

- Word clouds add simplicity and clarity. The most used keywords stand out better in a word cloud
- Word clouds are a potent communication tool. They are easy to understand, to be shared, and are impactful.
- Word clouds are visually engaging than a table data.

Confusion Matrix

A confusion matrix is a table that is used to define the performance of a classification algorithm. A confusion matrix visualizes and summarizes the performance of a classification algorithm.

The confusion matrix consists of four basic characteristics (numbers) that are used to define the measurement metrics of the classifier. These four numbers are:

i. TP (True Positive): TP represents the number of patients who have been properly classified to have malignant nodes, meaning they have the disease.
ii. TN (True Negative): TN represents the number of correctly classified patients who are healthy.
iii. FP (False Positive): FP represents the number of misclassified patients with the disease but actually they are healthy. FP is also known as a Type I error.
iv. FN (False Negative): FN represents the number of patients misclassified as healthy but actually they are suffering from the disease. FN is also known as a Type II error.

Accuracy of an algorithm is represented as the ratio of correctly classified patients (TP+TN) to the total number of patients (TP+TN+FP+FN).

Data Preparation

The raw text is unstructured in nature. Wikipedia defines unstructured text data as data is not organized in a predefined manner. Unstructured information has text, dates, numbers and facts. These make for irregularities and ambiguity which makes it difficult for the machine to understand the raw text.

We cannot go straight from raw text to fitting a machine learning or deep learning algorithm on it.

In order for machine to be able to deal with text data, the text data needs to be first cleaned and prepared so that it can be fed to the Machine Learning Algorithm for analysis.

Checking for missing values in the data

What is missing values?

The problem of missing value is quite common in many real-life datasets. Missing value can bias the results of the machine learning models and/or reduce the accuracy of the model. Missing data is defined as the values or data that is not stored (or not present) for some variable/s in the given dataset.

Why is data missing?

Reasons for the missing data from the dataset affect the approach of handling missing data. So it's necessary to understand why the data could be missing.

Some of the reasons are listed below:

- Past data might get corrupted due to improper maintenance.
- Observations are not recorded for certain fields due to some reasons. There might be a failure in recording the values due to human error.
- The user has not provided the values intentionally.

Why do we need to care about missing values?

- Many machine learning algorithms fail if the dataset contains missing values. However, algorithms like K-nearest and Naive Bayes support data with missing values.

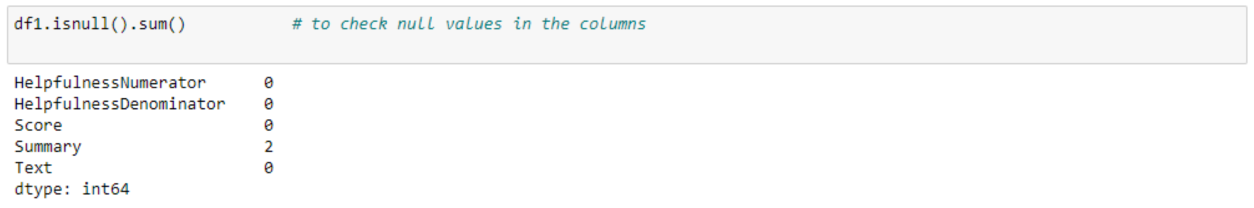
- You may end up building a biased machine learning model which will lead to incorrect results if the missing values are not handled properly.
- Missing data can lead to a lack of precision in the statistical analysis.

How to handle missing values?

There are 2 primary ways of handling missing values:

- Deleting the Missing values
- Imputing the Missing Values

Checking for missing values in the dataset

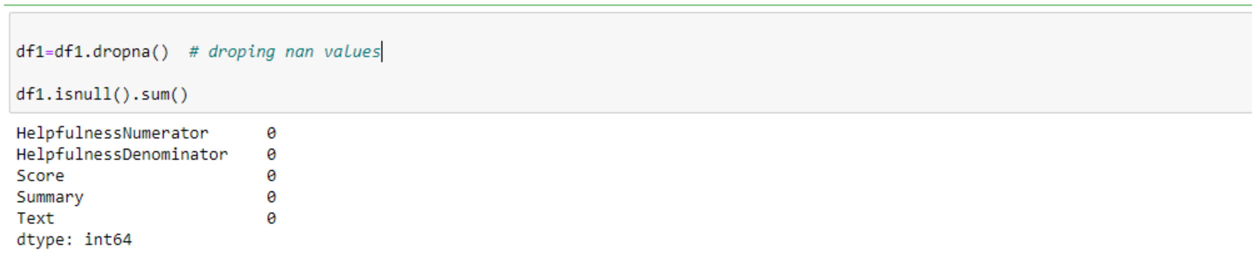


```
df1.isnull().sum()          # to check null values in the columns
```

```
HelpfulnessNumerator      0
HelpfulnessDenominator    0
Score                     0
Summary                   2
Text                      0
dtype: int64
```

As we can be seen that there are some missing values in summary column.

We will adopt the option of deleting the missing values as number of rows that will be deleted is very less than the data.



```
df1=df1.dropna()  # droping nan values

df1.isnull().sum()
```

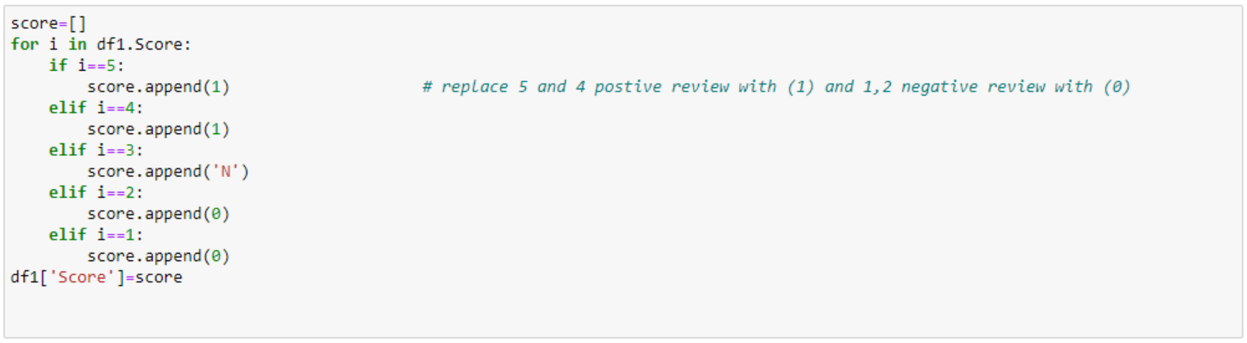
```
HelpfulnessNumerator      0
HelpfulnessDenominator    0
Score                     0
Summary                   0
Text                      0
dtype: int64
```

Label Encoding

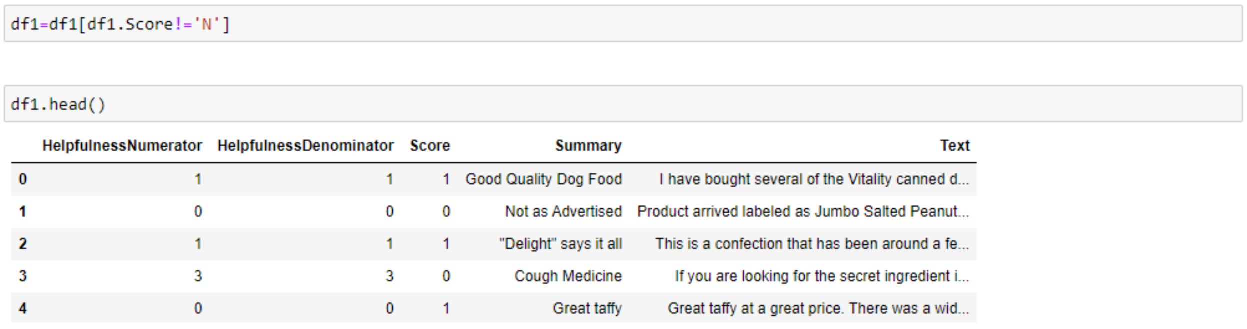
In machine learning, we usually deal with datasets that contain multiple labels in one or more than one columns. These labels can be in the form of words or numbers. To make the data understandable or in human-readable form, the training data is often labelled in words.

Label Encoding refers to converting the labels into a numeric form so as to convert them into the machine-readable form. Machine learning algorithms can then decide in a better way how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning.

In our data set, our target variable is on a likert scale of 1 to 5 varying from poor to good responses. To fit the model of binary classification, we need to convert this 1 to 5 scale classification to a binary class labels of 0 and 1.



```python
score=[]
for i in df1.Score:
    if i==5:
        score.append(1)                    # replace 5 and 4 postive review with (1) and 1,2 negative review with (0)
    elif i==4:
        score.append(1)
    elif i==3:
        score.append('N')
    elif i==2:
        score.append(0)
    elif i==1:
        score.append(0)
df1['Score']=score
```



```python
df1=df1[df1.Score!='N']
```

```python
df1.head()
```

	HelpfulnessNumerator	HelpfulnessDenominator	Score	Summary	Text
0	1	1	1	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	0	0	0	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	1	1	1	"Delight" says it all	This is a confection that has been around a fe...
3	3	3	0	Cough Medicine	If you are looking for the secret ingredient i...
4	0	0	1	Great taffy	Great taffy at a great price. There was a wid...

Transforming the text data to run Machine learning algorithms

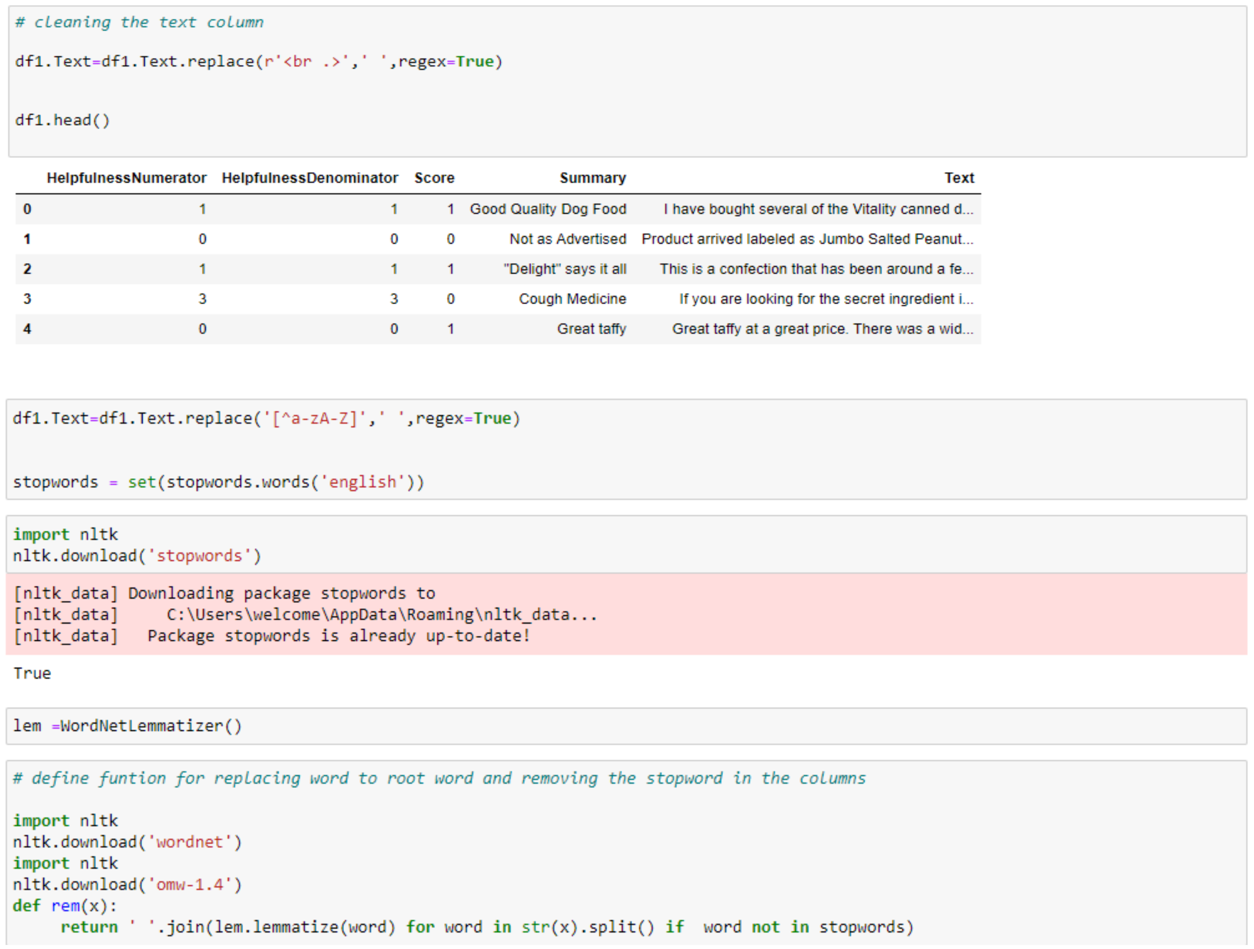
Lemmatization and removing unwanted symbols/text/stopwords

In many languages, words appear in several inflected forms. For example, in English, the verb 'to walk' may appear as 'walk', 'walked', 'walks' or 'walking'. The base form, 'walk', that one might look up in a dictionary, is called the lemma for the word. The association of the base form with a part of speech is often called a lexeme of the word.

Lemmatization is closely related to stemming. The difference is that a stemmer operates on a single word without knowledge of the context, and therefore cannot discriminate between words which have different meanings depending on part of speech. However, stemmers are typically easier to implement and run faster. The reduced "accuracy" may not matter for some applications. In fact, when used within information retrieval systems, stemming improves query recall accuracy, or true positive rate, when compared to lemmatization. Nonetheless, stemming reduces precision, or the proportion of positively-labeled instances that are actually positive, for such systems.[5]

For instance:

- The word "better" has "good" as its lemma. This link is missed by stemming, as it requires a dictionary look-up.
- The word "walk" is the base form for the word "walking", and hence this is matched in both stemming and lemmatization.
- The word "meeting" can be either the base form of a noun or a form of a verb ("to meet") depending on the context; e.g., "in our last meeting" or "We are meeting again tomorrow". Unlike stemming, lemmatization attempts to select the correct lemma depending on the context.



```
# cleaning the text column

df1.Text=df1.Text.replace(r'<br .>',' ',regex=True)


df1.head()
```

	HelpfulnessNumerator	HelpfulnessDenominator	Score	Summary	Text
0	1	1	1	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	0	0	0	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	1	1	1	"Delight" says it all	This is a confection that has been around a fe...
3	3	3	0	Cough Medicine	If you are looking for the secret ingredient i...
4	0	0	1	Great taffy	Great taffy at a great price. There was a wid...

```
df1.Text=df1.Text.replace('[^a-zA-Z]',' ',regex=True)


stopwords = set(stopwords.words('english'))
```

```
import nltk
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\welcome\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

```
True
```

```
lem =WordNetLemmatizer()
```

```
# define funtion for replacing word to root word and removing the stopword in the columns

import nltk
nltk.download('wordnet')
import nltk
nltk.download('omw-1.4')
def rem(x):
    return ' '.join(lem.lemmatize(word) for word in str(x).split() if  word not in stopwords)
```

```
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\welcome\AppData\Roaming\nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
[nltk_data] Downloading package omw-1.4 to
[nltk_data]     C:\Users\welcome\AppData\Roaming\nltk_data...
[nltk_data]   Package omw-1.4 is already up-to-date!
```

```
df1['Text']=df1['Text'].apply(lambda x:rem(x))
```

```
df1.Text=df1.Text.str.lower()

df1.Summary=df1.Summary.str.lower()            # for for converting all the text in the lower case

df1.head()
```

	HelpfulnessNumerator	HelpfulnessDenominator	Score	Summary	Text
0	1	1	1	good quality dog food	i bought several vitality canned dog food prod...
1	0	0	0	not as advertised	product arrived labeled jumbo salted peanuts p...
2	1	1	1	"delight" says it all	this confection around century it light pillow...
3	3	3	0	cough medicine	if looking secret ingredient robitussin i beli...
4	0	0	1	great taffy	great taffy great price there wide assortment ...

Converting text to numeric form using CountVectorizer

In order to use textual data for predictive modeling, the text must be parsed to remove certain words – this process is called tokenization. These words need to then be encoded as integers, or floating-point values, for use as inputs in machine learning algorithms. This process is called feature extraction (or vectorization).

Scikit-learn's CountVectorizer is used to convert a collection of text documents to a vector of term/token counts. It also enables the pre-processing of text data prior to generating the vector representation. This functionality makes it a highly flexible feature representation module for text.

```
cv=CountVectorizer(max_features=41157,ngram_range=(1,2),min_df=10)
X=cv.fit_transform(df1.Text)
y=df1.Score
```

Now we have two variables Namely X and Y.

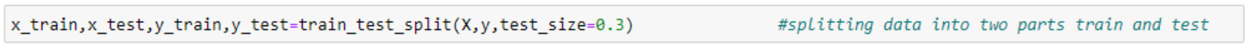
Y being the target variable having binary class labels 0 and 1

X is the variable storing the text data converted into numeric form to be trained using different machine learning classification algorithms to classify whether the review given by particular customers to a particular product is of a positive intend or is of a negative intend.

Split the data for training and testing

The train-test split is used to estimate the performance of machine learning algorithms that are applicable for prediction-based Algorithms/Applications. This method is a fast and easy procedure to perform such that we can compare our own machine learning model results to machine results.

Scikit-learn alias sklearn is the most useful and robust library for machine learning in Python. The scikit-learn library provides us with the model_selection module in which we have the splitter function train_test_split().



```
x_train,x_test,y_train,y_test=train_test_split(X,y,test_size=0.3)          #splitting data into two parts train and test
```

Model Building and implementation

Logistic Regression

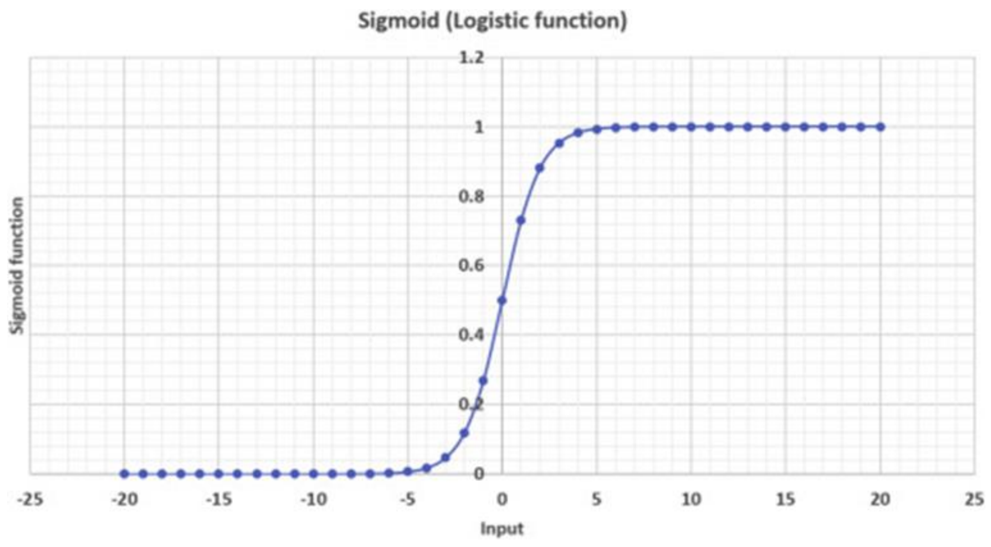
Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression predicts the output of a categorical dependent 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. Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems.

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. Logistic Regression is a significant machine learning algorithm because it has the ability to provide probabilities and classify new data using continuous and discrete datasets.

Logistic Regression can be used to classify the observations using different types of data and can easily determine the most effective variables used for the classification.

The sigmoid function is a mathematical function used to map the predicted values to probabilities. It maps any real value into another value within a range of 0 and 1. The value of the logistic regression must be between 0 and 1, which cannot go beyond this limit, so it forms a curve like the "S" form. The S-form curve is called the sigmoid function or the logistic function. 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.

The below image is showing the logistic function (Sigmoid function):



Assumptions for Logistic Regression:

- The dependent variable must be categorical in nature.
- The independent variable should not have multi-collinearity.

Type of Logistic Regression:

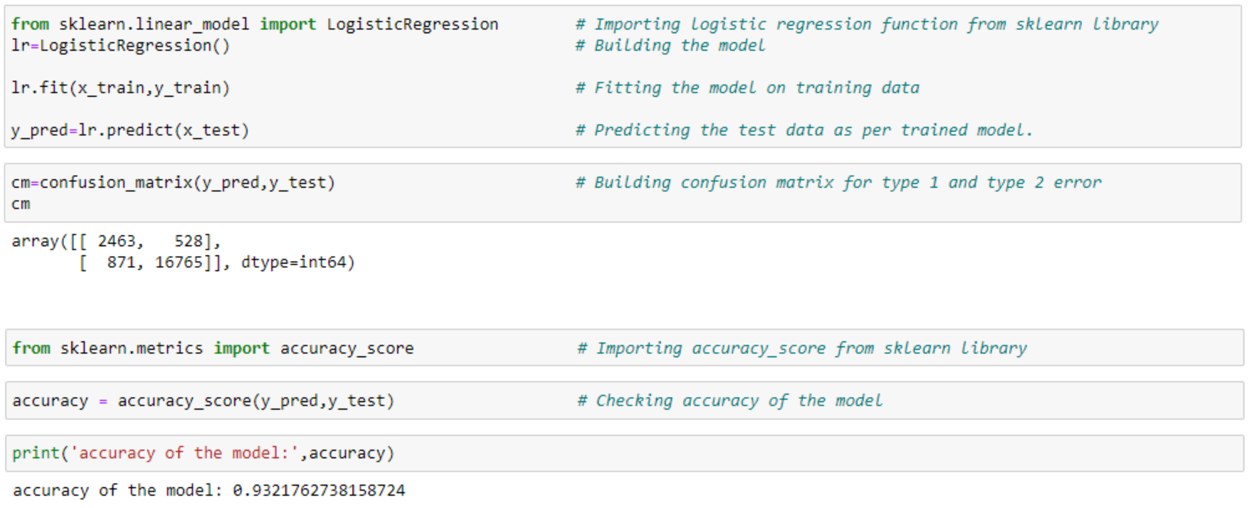
On the basis of the categories, Logistic Regression can be classified into three types:

Binomial: In binomial Logistic regression, there can be only two possible types of the dependent variables, such as 0 or 1, Pass or Fail, etc.

Multinomial: In multinomial Logistic regression, there can be 3 or more possible unordered types of the dependent variable, such as "cat", "dogs", or "sheep"

Ordinal: In ordinal Logistic regression, there can be 3 or more possible ordered types of dependent variables, such as "low", "Medium", or "High".

Python Programming



```
from sklearn.linear_model import LogisticRegression        # Importing logistic regression function from sklearn library
lr=LogisticRegression()                                    # Building the model

lr.fit(x_train,y_train)                                    # Fitting the model on training data

y_pred=lr.predict(x_test)                                  # Predicting the test data as per trained model.
```

```
cm=confusion_matrix(y_pred,y_test)                         # Building confusion matrix for type 1 and type 2 error
cm
```

```
array([[ 2463,   528],
       [  871, 16765]], dtype=int64)
```

```
from sklearn.metrics import accuracy_score                 # Importing accuracy_score from sklearn library
```

```
accuracy = accuracy_score(y_pred,y_test)                   # Checking accuracy of the model
```

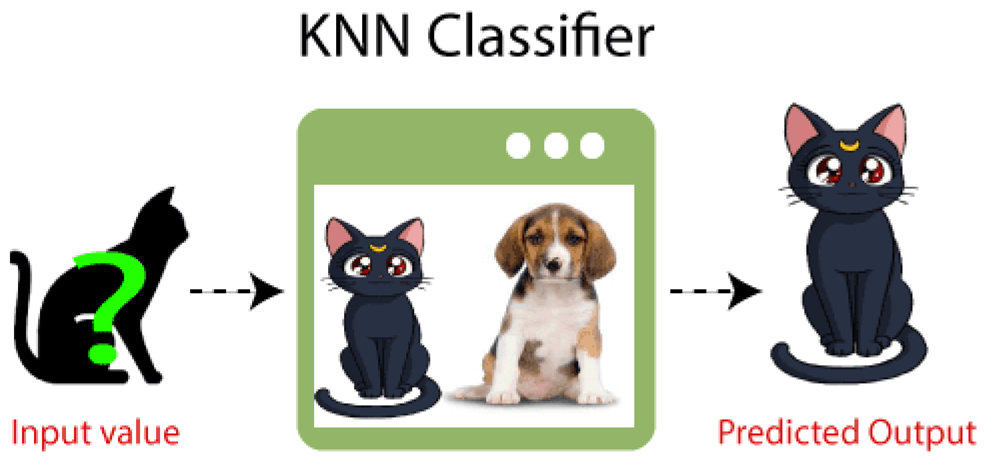
```
print('accuracy of the model:',accuracy)
```

```
accuracy of the model: 0.9321762738158724
```

K-Nearest Neighbor

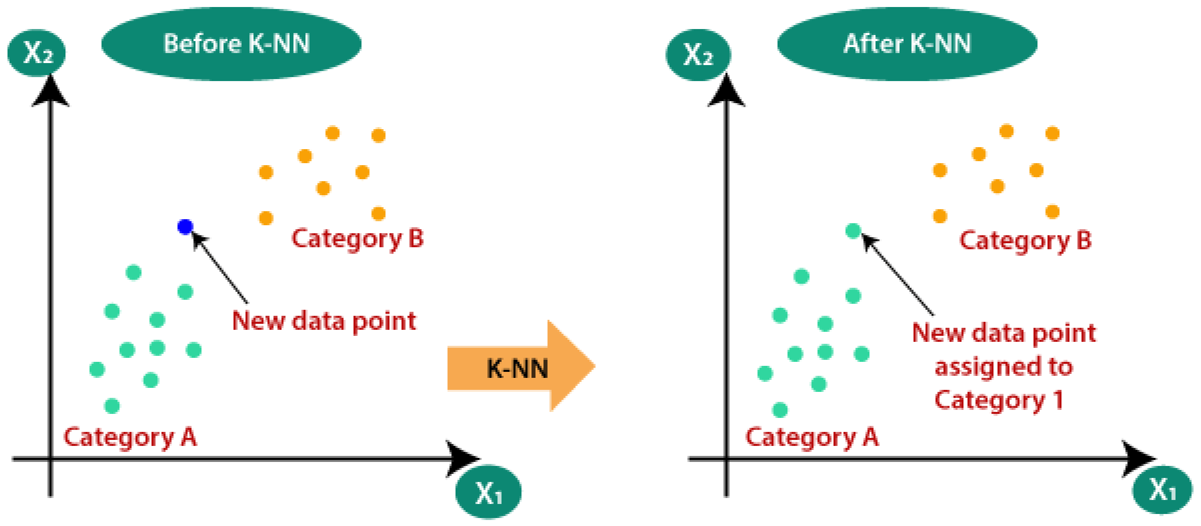
K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm. K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems. K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data.

It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset. KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data. Example: Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.



Why do we need KNN?

Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset.



Advantages of KNN Algorithm:

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

Disadvantages of KNN Algorithm:

- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

Python Programming

```
from sklearn.neighbors import KNeighborsClassifier          # Importing KNN from Sklearn Library
KNN = KNeighborsClassifier()                                # Building the model.
```

```
KNN.fit(x_train,y_train)                                    # Fitting the model on training data
```

```
KNeighborsClassifier()
```

```
y_pred_KNN = KNN.predict(x_test)                            #Predicting the test data as per trained model
```

```
cm_KNN = confusion_matrix(y_pred_KNN,y_test)                # Building confusion matrix for type 1 and type 2 error
cm_KNN
```

```
array([[  494,   437],
       [ 2840, 16856]], dtype=int64)
```

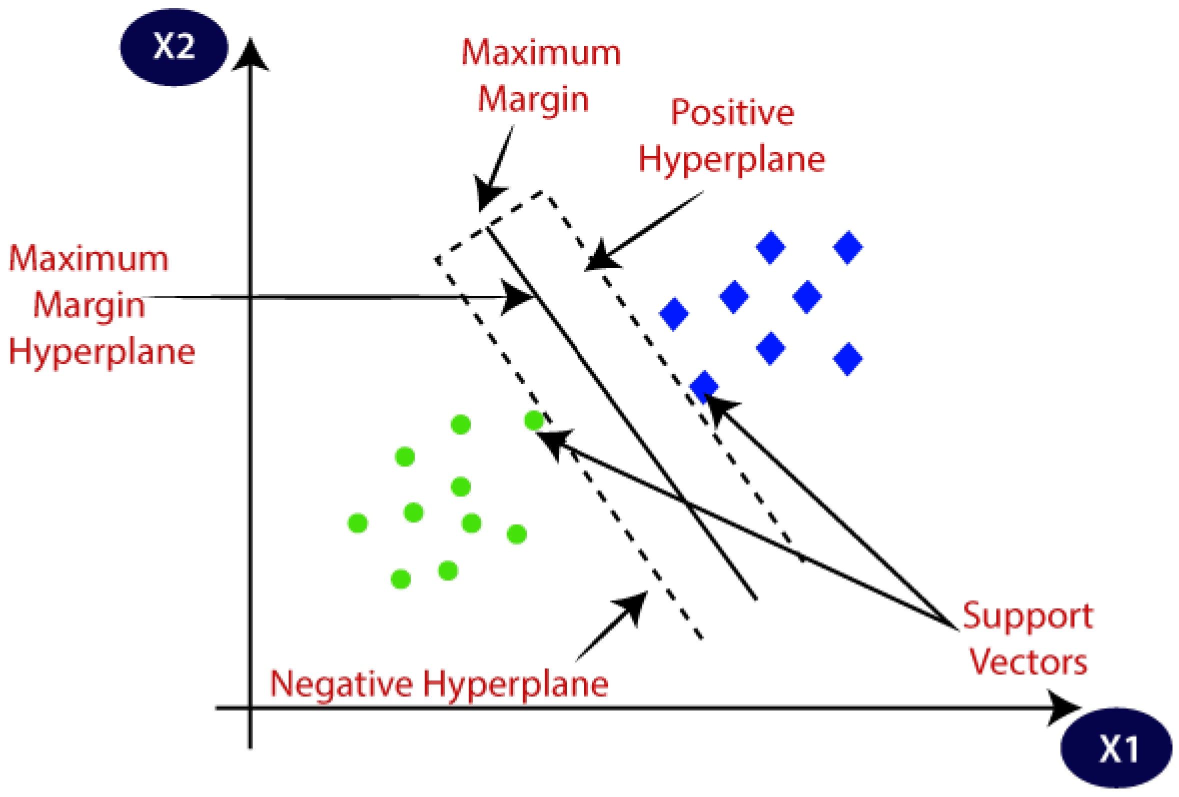
```
acc_KNN = accuracy_score(y_pred_KNN,y_test)                 # Checking for accuracy of the model
acc_KNN
```

```
0.8411305570368934
```

Support Vector Machine

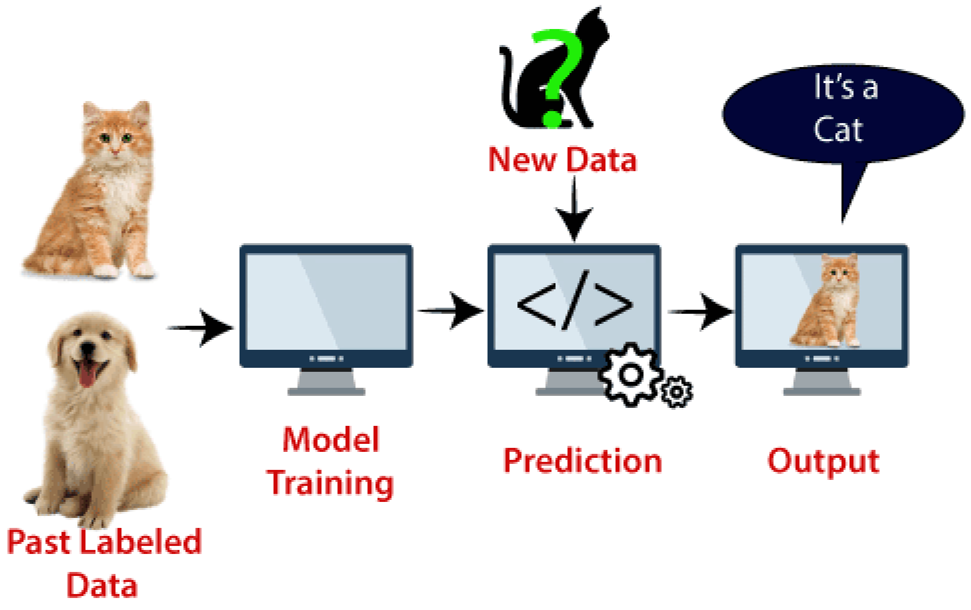
Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning. The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



Example:

SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:



SVM algorithm can be used for Face detection, image classification, text categorization, etc.

Types of SVM

SVM can be of two types:

- **Linear SVM**: Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM**: Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Python Programming

```
from sklearn.svm import SVC                                  # Importing SVC(support vector classifier) from Sklearn library
```

```
svm = SVC()                                                  # Building the model
```

```
svm.fit(x_train,y_train)                                     # Training the model on the training data
```

```
SVC()
```

```
y_pred_SVM = svm.predict(x_test)                             # Predicting the test data as per trained model
```

```
cm_SVM = confusion_matrix(y_pred_SVM,y_test)                 # Building confusion matrix for type 1 and type 2 error
```

```
cm_SVM
```

```
array([[ 1875,   199],
       [ 1459, 17094]], dtype=int64)
```

```
acc_SVM = accuracy_score(y_pred_SVM,y_test)                  # Checking for accuracy
```

```
acc_SVM
```

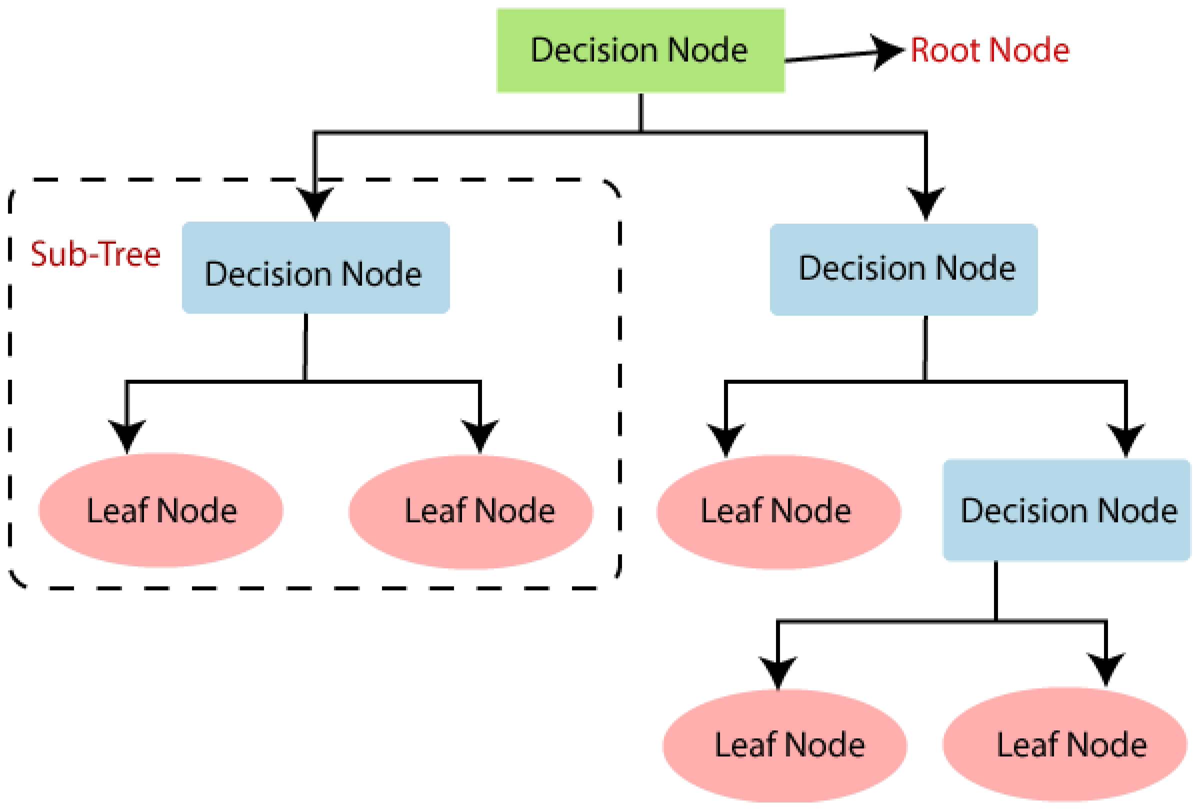
```
0.9196199156445436
```

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. The decisions or the test are performed on the basis of features of the given dataset. It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm. A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.

Below diagram explains the general structure of a decision tree:



Why use Decision Trees?

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

- Root Node: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- Leaf Node: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- Splitting: Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- Branch/Sub Tree: A tree formed by splitting the tree.
- Pruning: Pruning is the process of removing the unwanted branches from the tree.
- Parent/Child node: The root node of the tree is called the parent node, and other nodes are called the child nodes.

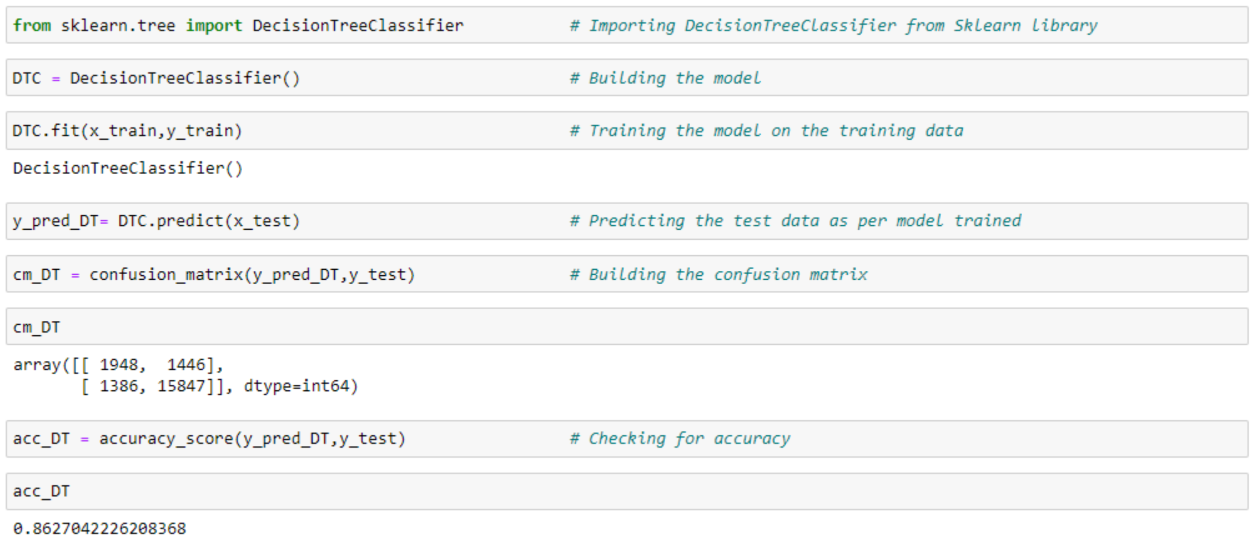
Advantages of the Decision Tree

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.
- There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
- For more class labels, the computational complexity of the decision tree may increase.

Python Programming



```
from sklearn.tree import DecisionTreeClassifier            # Importing DecisionTreeClassifier from Sklearn library

DTC = DecisionTreeClassifier()                             # Building the model

DTC.fit(x_train,y_train)                                   # Training the model on the training data

DecisionTreeClassifier()

y_pred_DT= DTC.predict(x_test)                             # Predicting the test data as per model trained

cm_DT = confusion_matrix(y_pred_DT,y_test)                 # Building the confusion matrix

cm_DT

array([[ 1948,  1446],
       [ 1386, 15847]], dtype=int64)

acc_DT = accuracy_score(y_pred_DT,y_test)                  # Checking for accuracy

acc_DT

0.8627042226208368
```

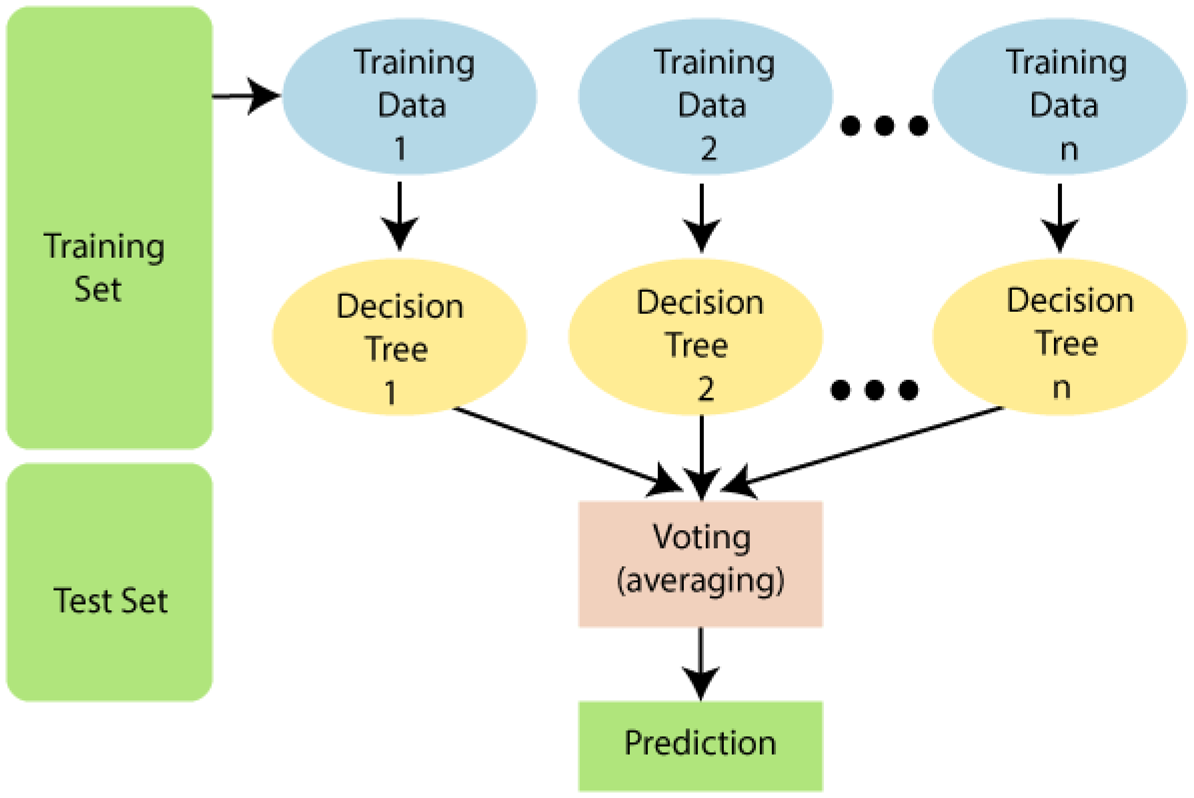
Random forest Classifier

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.

As the name suggests, "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." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

The below diagram explains the working of the Random Forest algorithm:



Assumptions for Random Forest

Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not. But together, all the trees predict the correct output. Therefore, below are two assumptions for a better Random forest classifier:

- There should be some actual values in the feature variable of the dataset so that the classifier can predict accurate results rather than a guessed result.
- The predictions from each tree must have very low correlations.

Why use Random Forest?

Below are some points that explain why we should use the Random Forest algorithm:

- It takes less training time as compared to other algorithms.
- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

Applications of Random Forest

There are mainly four sectors where Random forest mostly used:

- **Banking**: Banking sector mostly uses this algorithm for the identification of loan risk.
- **Medicine**: With the help of this algorithm, disease trends and risks of the disease can be identified.
- **Land Use**: We can identify the areas of similar land use by this algorithm.
- **Marketing**: Marketing trends can be identified using this algorithm.

Advantages of Random Forest

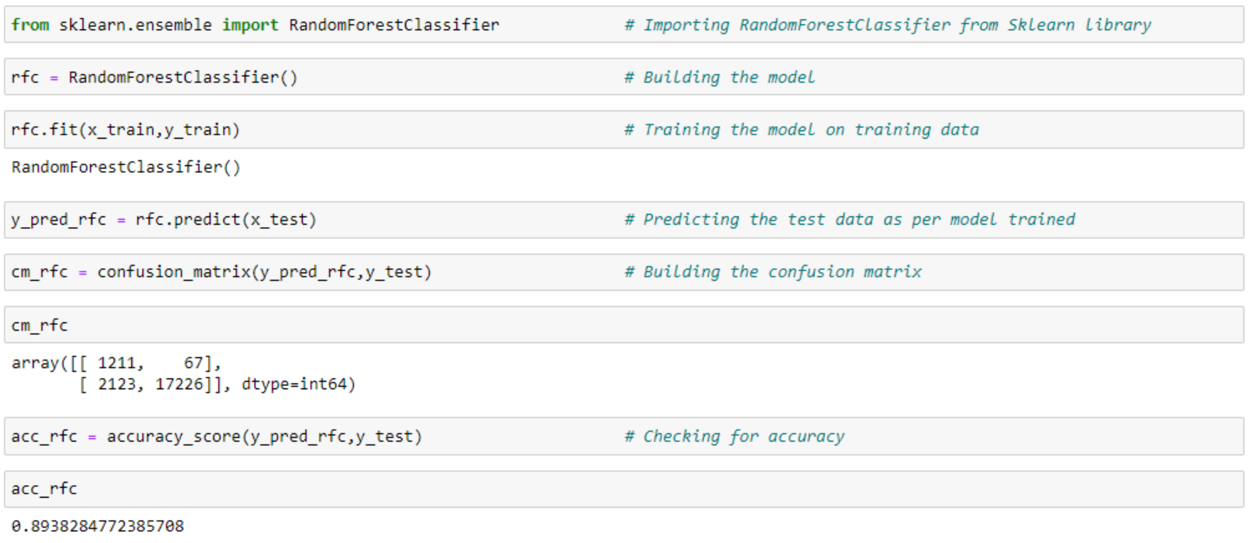
Random Forest is capable of performing both Classification and Regression tasks.

- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages of Random Forest

- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

Python Programming



```
from sklearn.ensemble import RandomForestClassifier            # Importing RandomForestClassifier from Sklearn library

rfc = RandomForestClassifier()                                 # Building the model

rfc.fit(x_train,y_train)                                       # Training the model on training data

RandomForestClassifier()

y_pred_rfc = rfc.predict(x_test)                               # Predicting the test data as per model trained

cm_rfc = confusion_matrix(y_pred_rfc,y_test)                   # Building the confusion matrix

cm_rfc

array([[ 1211,    67],
       [ 2123, 17226]], dtype=int64)

acc_rfc = accuracy_score(y_pred_rfc,y_test)                    # Checking for accuracy

acc_rfc

0.8938284772385708
```

Artificial Neural Network

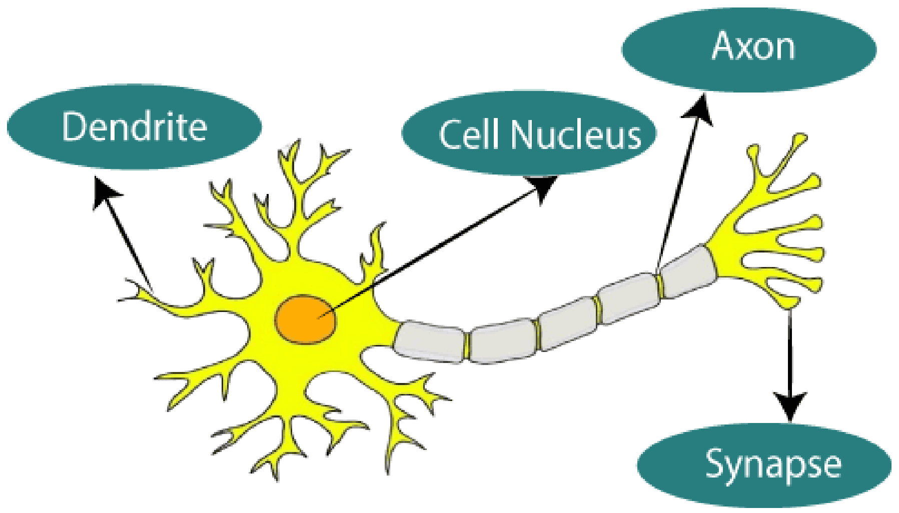
Artificial Neural Network Tutorial provides basic and advanced concepts of ANNs. Our Artificial Neural Network tutorial is developed for beginners as well as professions.

The term "Artificial neural network" refers to a biologically inspired sub-field of artificial intelligence modeled after the brain. An Artificial neural network is usually a computational network based on biological neural networks that construct the structure of the human brain. Similar to a human brain has neurons interconnected to each other, artificial neural networks also have neurons that are linked to each other in various layers of the networks. These neurons are known as nodes.

Artificial neural network tutorial covers all the aspects related to the artificial neural network. In this tutorial, we will discuss ANNs, Adaptive resonance theory, Kohonen self-organizing map, Building blocks, unsupervised learning, Genetic algorithm, etc.

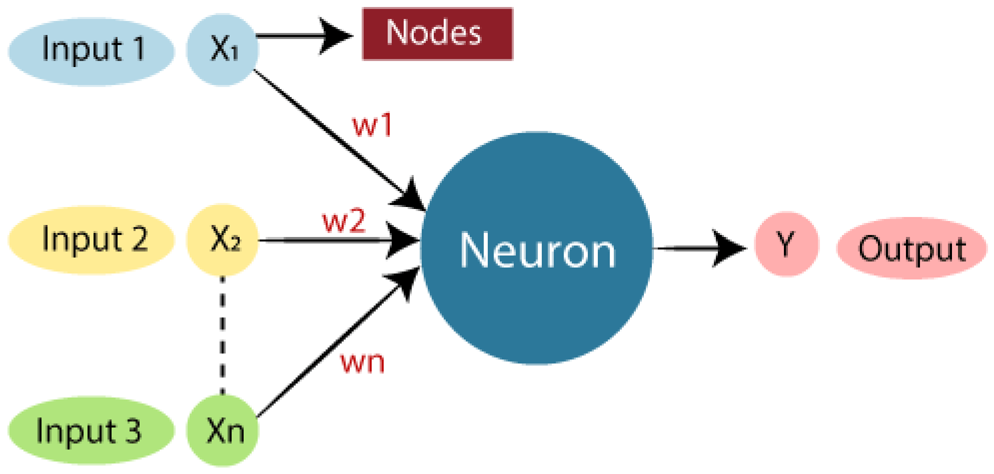
What is Artificial Neural Network?

The term "Artificial Neural Network" is derived from Biological neural networks that develop the structure of a human brain. Similar to the human brain that has neurons interconnected to one another, artificial neural networks also have neurons that are interconnected to one another in various layers of the networks. These neurons are known as nodes.



The given figure illustrates the typical diagram of Biological Neural Network.

The typical Artificial Neural Network looks something like the given figure.



Dendrites from Biological Neural Network represent inputs in Artificial Neural Networks, cell nucleus represents Nodes, synapse represents Weights, and Axon represents Output.

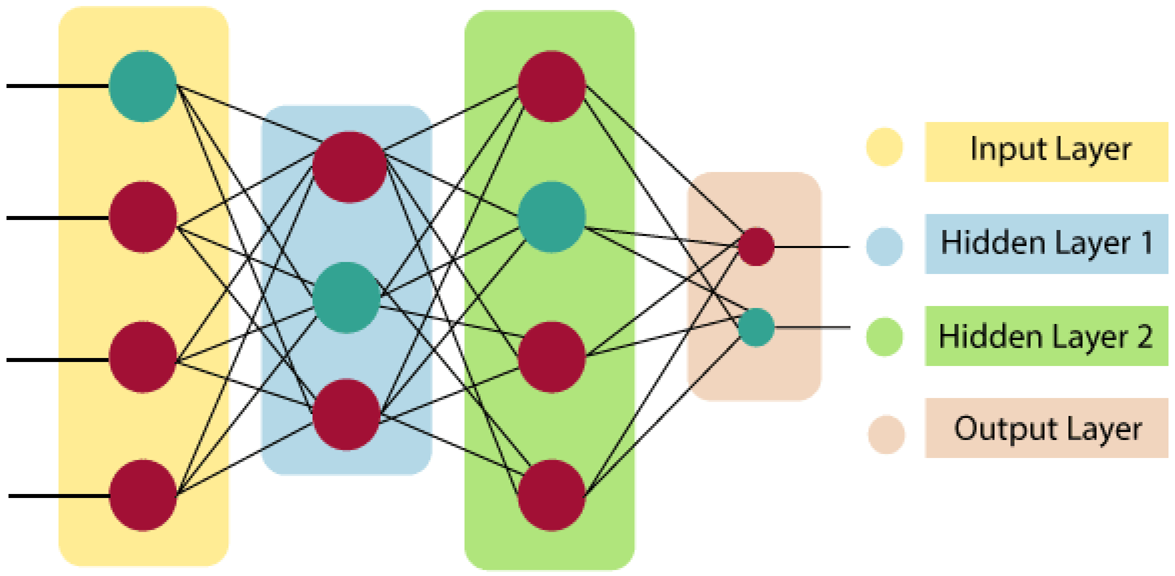
An Artificial Neural Network in the field of Artificial intelligence where it attempts to mimic the network of neurons makes up a human brain so that computers will have an option to understand things and make decisions in a human-like manner. The artificial neural network is designed by programming computers to behave simply like interconnected brain cells.

There are around 1000 billion neurons in the human brain. Each neuron has an association point somewhere in the range of 1,000 and 100,000. In the human brain, data is stored in such a manner as to be distributed, and we can extract more than one piece of this data when necessary from our memory parallelly. We can say that the human brain is made up of incredibly amazing parallel processors.

We can understand the artificial neural network with an example, consider an example of a digital logic gate that takes an input and gives an output. "OR" gate, which takes two inputs. If one or both the inputs are "On," then we get "On" in output. If both the inputs are "Off," then we get "Off" in output. Here the output depends upon input. Our brain does not perform the same task. The outputs to inputs relationship keep changing because of the neurons in our brain, which are "learning."

The architecture of an artificial neural network:

To understand the concept of the architecture of an artificial neural network, we have to understand what a neural network consists of. In order to define a neural network that consists of a large number of artificial neurons, which are termed units arranged in a sequence of layers. Lets us look at various types of layers available in an artificial neural network.



- **Input Layer:** As the name suggests, it accepts inputs in several different formats provided by the programmer.
- **Hidden Layer:** The hidden layer presents in-between input and output layers. It performs all the calculations to find hidden features and patterns
- **Output Layer:** The input goes through a series of transformations using the hidden layer, which finally results in output that is conveyed using this layer.

Advantages of Artificial Neural Network (ANN)

- **Parallel processing capability**: Artificial neural networks have a numerical value that can perform more than one task simultaneously.
- **Storing data on the entire network**: Data that is used in traditional programming is stored on the whole network, not on a database. The disappearance of a couple of pieces of data in one place doesn't prevent the network from working.
- **Capability to work with incomplete knowledge**: After ANN training, the information may produce output even with inadequate data. The loss of performance here relies upon the significance of missing data.
- **Having a memory distribution**: For ANN is to be able to adapt, it is important to determine the examples and to encourage the network according to the desired output by

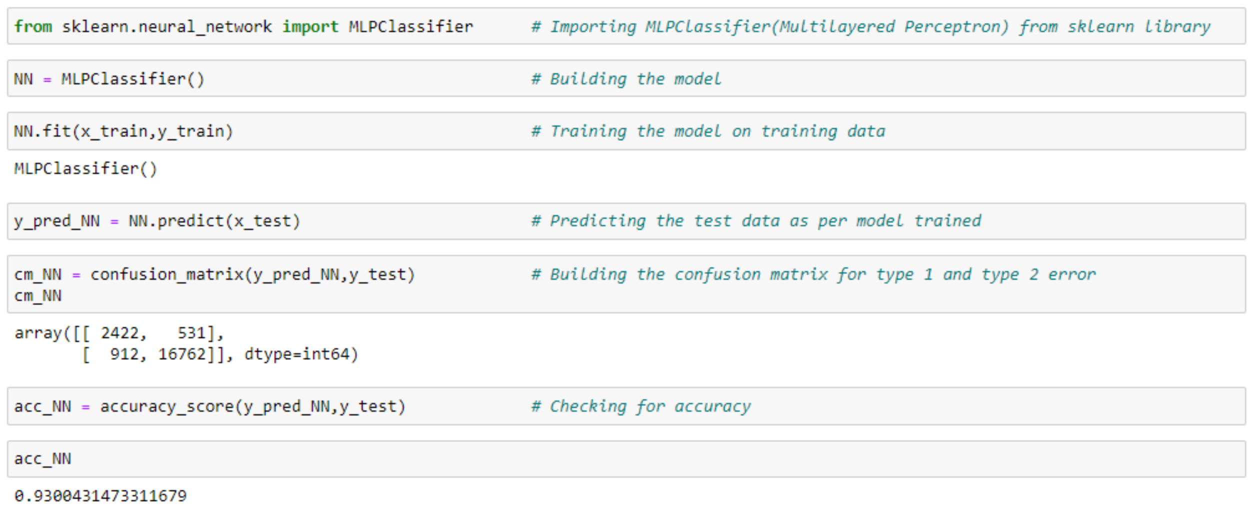
demonstrating these examples to the network. The succession of the network is directly proportional to the chosen instances, and if the event can't appear to the network in all its aspects, it can produce false output.

- **Having fault tolerance**: Extortion of one or more cells of ANN does not prohibit it from generating output, and this feature makes the network fault-tolerance.

Disadvantages of Artificial Neural Network:

- **Assurance of proper network structure:** There is no particular guideline for determining the structure of artificial neural networks. The appropriate network structure is accomplished through experience, trial, and error.
- **Unrecognized behavior of the network:** It is the most significant issue of ANN. When ANN produces a testing solution, it does not provide insight concerning why and how. It decreases trust in the network.
- **Hardware dependence:** Artificial neural networks need processors with parallel processing power, as per their structure. Therefore, the realization of the equipment is dependent.
- **Difficulty of showing the issue to the network:** ANNs can work with numerical data. Problems must be converted into numerical values before being introduced to ANN. The presentation mechanism to be resolved here will directly impact the performance of the network. It relies on the user's abilities.
- **The duration of the network is unknown:** The network is reduced to a specific value of the error, and this value does not give us optimum results.

Python Programming



```
from sklearn.neural_network import MLPClassifier       # Importing MLPClassifier(Multilayered Perceptron) from sklearn library
```

```
NN = MLPClassifier()                                   # Building the model
```

```
NN.fit(x_train,y_train)                                # Training the model on training data
```

```
MLPClassifier()
```

```
y_pred_NN = NN.predict(x_test)                         # Predicting the test data as per model trained
```

```
cm_NN = confusion_matrix(y_pred_NN,y_test)             # Building the confusion matrix for type 1 and type 2 error
cm_NN
```

```
array([[ 2422,   531],
       [  912, 16762]], dtype=int64)
```

```
acc_NN = accuracy_score(y_pred_NN,y_test)              # Checking for accuracy
```

```
acc_NN
```

```
0.9300431473311679
```

Model Comparison

Sr. No.	Model	Accuracy (%)
1.	Logistic Regression	93.22
2.	K-Nearest Neighbors	84.11
3.	Support Vector Machine	91.96
4.	Decision Tree	86.27
5.	Random Forest Classifier	89.38
6.	Artificial Neural Network	93.00

As per the above table, Logistic regression and Artificial Neural network are found to be the best models that classifies the sentiments of the reviews given by various customers for various products.

Conclusion

This research was able to present a comparison study between Logistic Regression, K-Nearest Neighbor, Support Vector Machine, Decision Tree, Random Forest Classifier and Neural Network to analyze the polarization of the sentiment of Amazon product reviews. In this work, the models were trained with almost 77000 datasets after the preprocessing procedure. In the meantime, almost 23000 test sets have been passed through the models for the statistical measurement. The system provides an accuracy of

1. 93.22% for the Logistic Regression.
2. 93.00% for Artificial Neural Network
3. 91.96% for Support Vector Machine
4. 89.38% for Random Forest Classifier
5. 86.27% for Decision Tree
6. 84.11% for K-Nearest Neighbor

Experimental results have confirmed that the Logistic Regression can polarize the feedback of Amazon products with a higher accuracy rate.

Scope for further research

In this research study, we have used machine learning classification algorithms to analyze the polarization (Negative or positive) of amazon product reviews. In that process, we have labelled the score as 0 and 1 and followed a supervised learning approach in our study.

We could have also used unsupervised learning approach like PCA (Principal component Analysis) to encode the dataset for further analysis.

Bibliography

In completion of this project, I have taken aid from various sources from internet namely:

1. www.geeksforgeeks.org
2. www.towardsdatascience.com
3. www.javatpoint.com
4. www.wikipedia.org
5. www.kaggle.com
6. www.researchgate.net
7. www.rpubs.com
8. www.scikit-learn.org
9. www.github.com