TWITTER SENTIMENT ANALYSIS

A PROJECT REPORT

In partial fulfilment of the requirements for the award of the degree

BACHELOR OF TECHNOLOGY
IN
COMPUTER SCIENCE AND ENGINEERING

Under the guidance of
UPASAK PAL

BY

AMAR SHAW

FUTURE INSTITUTE OF ENGINEERING AND MANAGEMENT

In association with

(ISO9001:2015)

SDF Building, Module #132, Ground Floor, Salt Lake City, GP Block, Sector V, Kolkata, West Bengal 700091

(Note: All entries of the proforma of approval should be filled up with appropriate and complete information. Incomplete proforma of approval in any respect will be summarily rejected.)

1. Title of the Project: **TWITTER SENTIMENT ANALYSIS**

2. Project Members: **AMAR SHAW**

3. Name of the guide: **Mr. UPASAK PAL**

4. Address: Ardent Computech Pvt. Ltd
(An ISO 9001:2015 Certified)
SDF Building, Module #132, Ground Floor, Salt Lake City, GP Block, Sector V, Kolkata, West Bengal, 700091

Project Version Control History

Version	Primary Author	Description of Version	Date Completed
Final	**AMAR SHAW**	Project Report	25TH JAN,2019

Signature of Student

Date:

Signature of Approver

Date:

For Office Use Only

Approved **Not Approved**

MR.UPASAK PAL
Project Proposal Evaluator

DECLARATION

We hereby declare that the project work being presented in the project proposal entitled **"TWITTER SENTIMENT ANALYSIS"** in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** at **ARDENT COMPUTECH PVT. LTD, SALTLAKE, KOLKATA, WEST BENGAL,** is an authentic work carried out under the guidance of **MR. UPASAK PAL**. The matter embodied in this project work has not been submitted elsewhere for the award of any degree of our knowledge and belief.

Date:

Name of the Student: **AMAR SHAW**

Signature of the student:

Ardent Computech Pvt. Ltd (An ISO 9001:2015 Certified)

SDF Building, Module #132, Ground Floor, Salt Lake City, GP Block, Sector V, Kolkata, West Bengal 700091

CERTIFICATE

This is to certify that this proposal of minor project entitled **"TWITTER SENTIMENT ANALYSIS"** is a record of bonafide work, carried out by **AMAR SHAW** under my guidance at **ARDENT COMPUTECH PVT LTD**. In my opinion, the report in its present form is in partial fulfilment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY** and as per regulations of the **ARDENT®.** To the best of my knowledge, the results embodied in this report, are original in nature and worthy of incorporation in the present version of the report.

Guide / Supervisor

MR. UPASAK PAL

Project Engineer

Ardent Computech Pvt. Ltd (An ISO 9001:2015 Certified)

SDF Building, Module #132, Ground Floor, Salt Lake City, GP Block, Sector V, Kolkata, West Bengal 700091

ACKNOWLEDGEMENT

Success of any project depends largely on the encouragement and guidelines of many others. I take this sincere opportunity to express my gratitude to the people who have been instrumental in the successful completion of this project work.

I would like to show our greatest appreciation to ***Mr. Upasak Pal***, Project Engineer at Ardent, Kolkata. I always feel motivated and encouraged every time by his valuable advice and constant inspiration; without his encouragement and guidance this project would not have materialized.

Words are inadequate in offering our thanks to the other trainees, project assistants and other members at Ardent Computech Pvt. Ltd. for their encouragement and cooperation in carrying out this project work. The guidance and support received from all the members and who are contributing to this project, was vital for the success of this project.

Table of Contents

Ardent Computech Pvt. Ltd.

Ardent Computech Private Limited is an ISO 9001-2008 certified Software Development Company in India. It has been operating independently since 2003. It was recently merged with ARDENT TECHNOLOGIES.

Ardent Technologies

ARDENT TECHNOLOGIES is a Company successfully providing its services currently in UK, USA, Canada and India. The core line of activity at ARDENT TECHNOLOGIES is to develop customized application software covering the entire responsibility of performing the initial system study, design, development, implementation and training. It also deals with consultancy services and Electronic Security systems. Its primary clientele includes educational institutes, entertainment industries, resorts, theme parks, service industry, telecom operators, media and other business houses working in various capacities.

Ardent Collaborations

ARDENT COLLABORATIONS, the Research Training and Development Department of ARDENT COMPUTECH PVT LTD is a professional training Company offering IT enabled services & industrial trainings for B-Tech, MCA, BCA, MSc and MBA fresher's and experienced developers/programmers in various platforms. Summer Training / Winter Training / Industrial training will be provided for the students of B. Tech, M.Tech., MBA, MCA and BCA only. Deserving candidates may be awarded stipends, scholarships and other benefits, depending on their performance and recommendations of the mentors.

It is affiliated to National Council of Vocational Training (NCVT), Directorate General of Employment & Training (DGET), Ministry of Labor & Employment, and Government of India.

Abstract

This project addresses the problem of sentiment analysis in twitter; that is classifying tweets according to the sentiment expressed in them: positive and negative. Twitter is an on-line micro-blogging and social-networking platform which allows users to write short status updates of maximum length 140 characters. It is a rapidly expanding service with over 200 million registered users - out of which 100 million are active users and half of them log on twitter on a daily basis - generating nearly 250 million tweets per day. Due to this large amount of usage we hope to achieve a reflection of public sentiment by analyzing the sentiments expressed in the tweets. Analysing the public sentiment is important for many applications such as firms trying to find out the response of their products in the market, predicting political elections and predicting socioeconomic phenomena like stock exchange. The aim of this project is to develop a functional classifier for accurate and automatic sentiment classification of an unknown tweet stream.

Introduction

What is Twitter?

Twitter is an online news and social networking service on which users post and interact with messages known as "tweets". Tweets were originally restricted to 140 characters, but on November 7, 2017, this limit was doubled for all languages except Japanese, Korean, and Chinese. Registered users can post tweets, but those who are unregistered can only read them. Users access Twitter through its website interface, through Short Message

Service (SMS) or mobile-device application software ("app"). Twitter, Inc. is based in San Francisco, California, and has more than 25 offices around the world.

Twitter was created in March 2006 by Jack Dorsey, Noah Glass, Biz Stone, and Evan Williams and launched in July of that year. The service rapidly gained worldwide popularity. In 2012, more than 100 million users posted 340 million tweets a day, and the service handled an average of 1.6 billion search queries per day. In
2013, it was one of the ten most-visited websites and has been described as "the SMS of the Internet". As of 2016, Twitter had more than 319 million monthly active users. On the day of the 2016 U.S. presidential election, Twitter proved to be the largest source of breaking news, with 40 million election-related tweets sent by 10 p.m. (Eastern Time) that day.

Motivation

Working with these informal text genres presents challenges for natural language processing beyond those typically encountered when working with more traditional text genres, such as newswire data. Tweets and texts are short: a sentence or a headline rather than a document. The language used is very informal, with creative spelling and punctuation, misspellings, slang, new words, URLs, and genre-specific terminology and abbreviations, such as, RT for "re-tweet" and # hashtags, which are a type of tagging for Twitter messages. How to handle such challenges so as to automatically mine and understand the opinions and sentiments that people are communicating has only very recently been the subject of research (Jansen et al., 2009 ; Barbosa and Feng, 2010; Bifet and Frank, 2010; Davidov et al., 2010; OâĂŹConnor et al., 2010; Pak and Paroubek, 2010; Tumasjen et al., 2010; Kouloumpis et al., 2011). Another aspect of social media data such as Twitter messages is that it includes rich structured information about the individuals involved in the communication. For example, Twitter maintains information of who follows whom and re-tweets and tags inside of tweets provide discourse information. Modelling such structured information is important because: (i) it can lead to more accurate tools for extracting semantic information, and (ii) because it provides means for empirically studying properties of social interactions (e.g., we can study properties of persuasive language or what properties are associated with influential users).

Sentiment Analysis

What is sentiment analysis?

Opinion mining (sometimes known as **sentiment analysis** or **emotion AI**) refers to 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.

Generally speaking, sentiment analysis aims to determine the attitude of a speaker, writer, or other subject with respect to some topic or the overall contextual polarity or emotional reaction to a document, interaction, or event. The attitude may be a judgment or evaluation, affective state (that is to say, the emotional state of the author or speaker), or the intended emotional communication (that is to say, the emotional effect intended by the author or interlocutor).

Examples

The objective and challenges of sentiment analysis can be shown through some simple examples.

Simple cases

Coronet has the best lines of all day cruisers.

Bertram has a deep V hull and runs easily through seas.

Pastel-colored 1980s day cruisers from Florida are ugly.

I dislike old cabin cruisers.

Why Sentiment Analysis with Twitter?

According to *Statista* there are 313 million people using Twitter actively, that is more or less 23% of the total adults in the world. So it is not really a leap of faith to think that a large portion of your customers will be using it.

Most of the companies have accepted this social platform as a channel to communicate with clients. In fact, some of them have created independent accounts focused on customer service. All to be more effective at listening to their customers and react in an agile way to suggestions or complains.

However, listening is not enough for a business. Communication with clients is not enough if the feedback received goes nowhere or if you are not able to objectively measure how things are going. The challenging part is what comes next: analyze the messages you have received to enhance your business performance.

Machine Learning

What is machine learning?

Machine learning is a field of computer science that often uses statistical techniques to give computers the ability to "learn" (i.e., progressively improve performance on a specific task) with data, without being explicitly programmed

The name machine learning was coined in 1959 by Arthur Samuel. Evolved from the study of pattern recognition and computational learning theory in artificial intelligence, machine learning explores the study and construction of algorithms that can learn from and make predictions on data – such algorithms overcome following strictly static program instructions by making data-driven predictions or decisions, through building a model from sample inputs. Machine learning is employed in a range of computing tasks where designing and programming explicit algorithms with good performance is difficult or infeasible; example applications include email filtering, detection of network intruders or malicious insiders working towards a data breach, optical character recognition (OCR), learning to rank, and computer vision.

Machine learning is closely related to (and often overlaps with) computational statistics, which also focuses on prediction-making through the use of computers. It has strong ties to mathematical optimization, which delivers methods, theory and application domains to the field. Machine learning is sometimes conflated with data mining, where the latter subfield focuses more on exploratory data analysis and is known as unsupervised learning. Machine learning can also be unsupervised and be used to learn and establish baseline behavioral profiles for various entities and then used to find meaningful anomalies.

Within the field of data analytics, machine learning is a method used to devise complex models and algorithms that lend themselves to prediction; in commercial use, this is known as predictive analytics. These analytical models allow researchers, data scientists, engineers, and analysts to "produce reliable, repeatable decisions and results" and uncover "hidden insights" through learning from historical relationships and trends in the data.

Broadly, there are 3 types of Machine Learning Algorithms.

1. Supervised Learning

How it works: This algorithm consists of a target / outcome variable (or dependent variable) which is to be predicted from a given set of predictors (independent variables). Using these set of variables, we generate a function that map inputs to desired outputs. The training process continues until the model achieves a desired level of accuracy on the training data. Examples of Supervised Learning: Regression, Decision Tree, Random Forest, KNN, Logistic Regression etc.

2. Unsupervised Learning

How it works: In this algorithm, we do not have any target or outcome variable to predict / estimate. It is used for clustering population in different groups, which is widely used for segmenting customers in different groups for specific intervention. Examples of Unsupervised Learning: Apriori algorithm, K-means.

3. Reinforcement Learning:

How it works: Using this algorithm, the machine is trained to make specific decisions. It works this way: the machine is exposed to an environment where it trains itself continually using trial and error. This machine learns from past experience and tries to capture the best possible knowledge to make accurate business decisions. Example of Reinforcement Learning: Markov Decision Process

List of Common Machine Learning Algorithms

Here is the list of commonly used machine learning algorithms. These algorithms can be applied to almost any data problem:

1. Linear Regression
2. Logistic Regression
3. Decision Tree
4. SVM
5. Naive Bayes
6. kNN
7. K-Means
8. Random Forest

1. Linear Regression

It is used to estimate real values (cost of houses, number of calls, total sales etc.) based on continuous variable(s). Here, we establish relationship between independent and dependent variables by fitting a best line. This best fit line is known as regression line and represented by a linear equation **Y= a *X + b.**

The best way to understand linear regression is to relive this experience of childhood. Let us say, you ask a child in fifth grade to arrange people in his class by increasing order of weight, without asking them their weights! What do you think the child will do? He / she would likely look (visually analyze) at the height and build of people and arrange them using a combination of these visible parameters. This is linear regression in real life!

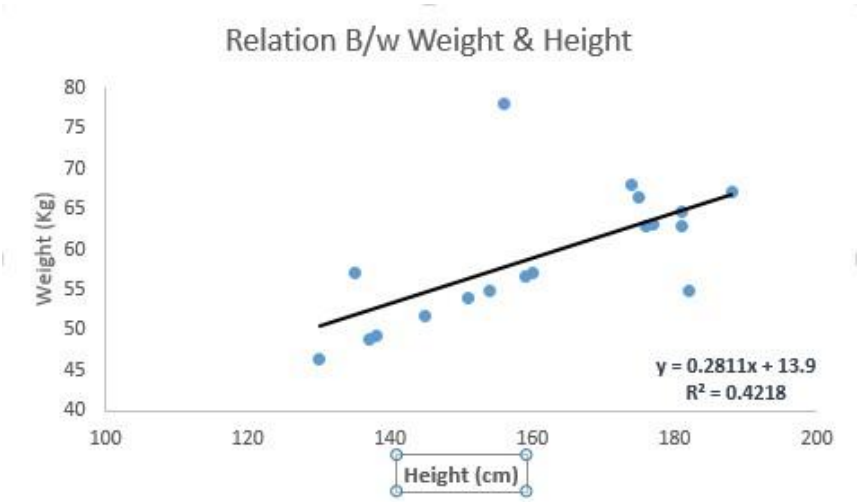
The child has actually figured out that height and build would be correlated to the weight by a relationship, which looks like the equation above.

In this equation:

Y – Dependent Variable
a – Slope
X – Independent variable
b – Intercept

These coefficients a and b are derived based on minimizing the sum of squared difference of distance between data points and regression line.

Look at the below example. Here we have identified the best fit line having linear equation **y=0.2811x+13.9**. Now using this equation, we can find the weight, knowing the height of a person.



Linear Regression is of mainly two types: Simple Linear Regression and Multiple Linear Regression. Simple Linear Regression is characterized by one independent variable. And, Multiple Linear Regression (as the name suggests) is characterized by multiple (more than 1) independent variables. While finding best fit line, you can fit a polynomial or curvilinear regression. And these are known as polynomial or curvilinear regression.

2. Logistic Regression

Don’t get confused by its name! It is a classification not a regression algorithm. It is used to estimate discrete values (Binary values like 0/1, yes/no, true/false) based on given set of independent variable(s). In simple

words, it predicts the probability of occurrence of an event by fitting data to a logit function. Hence, it is also known as **logit regression**. Since, it predicts the probability, its output values lies between 0 and 1 (as expected).

Again, let us try and understand this through a simple example.

Let's say your friend gives you a puzzle to solve. There are only 2 outcome scenarios – either you solve it or you don't. Now imagine, that you are being given wide range of puzzles / quizzes in an attempt to understand which subjects you are good at. The outcome to this study would be something like this – if you are given a trigonometry based tenth grade problem, you are 70% likely to solve it. On the other hand, if it is grade fifth history question, the probability of getting an answer is only 30%. This is what Logistic Regression provides you.

Coming to the math, the log odds of the outcome is modeled as a linear combination of the predictor variables.

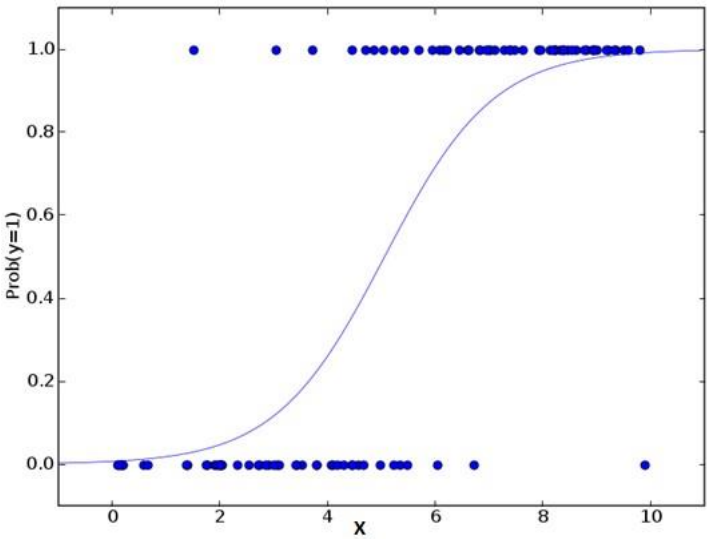
odds= p/ (1-p) = probability of event occurrence / probability of not event occurrence

ln(odds) = ln(p/(1-p))

logit(p) = ln(p/(1-p)) = b0+b1X1+b2X2+b3X3....+bkXk

Above, p is the probability of presence of the characteristic of interest. It chooses parameters that maximize the likelihood of observing the sample values rather than that minimize the sum of squared errors (like in ordinary regression).

Now, you may ask, why take a log? For the sake of simplicity, let's just say that this is one of the best mathematical way to replicate a step function. I can go in more details, but that will beat the purpose of this article.



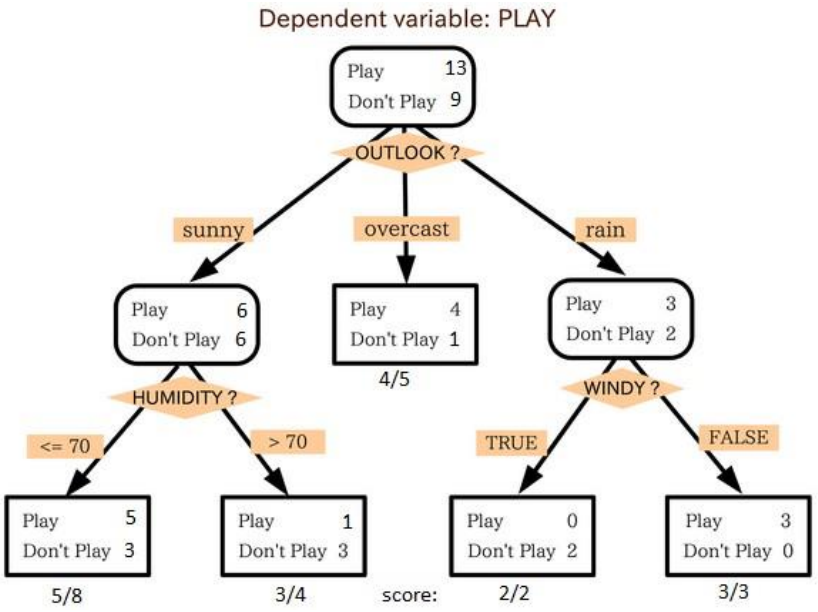
Furthermore..

There are many different steps that could be tried in order to improve the model:

- including interaction terms
- removing features
- regularization techniques
- using a non-linear model

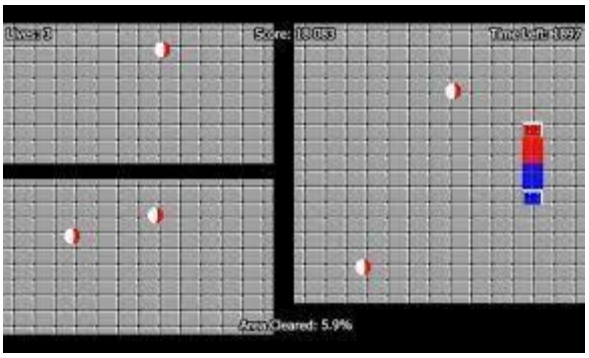
3. Decision Tree

This is one of my favorite algorithm and I use it quite frequently. It is a type of supervised learning algorithm that is mostly used for classification problems. Surprisingly, it works for both categorical and continuous dependent variables. In this algorithm, we split the population into two or more homogeneous sets. This is done based on most significant attributes/ independent variables to make as distinct groups as possible.
For more details, you can read: Decision Tree Simplified.



In the image above, you can see that population is classified into four different groups based on multiple attributes to identify 'if they will play or not'. To split the population into different heterogeneous groups, it uses various techniques like Gini, Information Gain, Chi-square, entropy.

The best way to understand how decision tree works, is to play Jezzball – a classic game from Microsoft (image below). Essentially, you have a room with moving walls and you need to create walls such that maximum area gets cleared off without the balls.

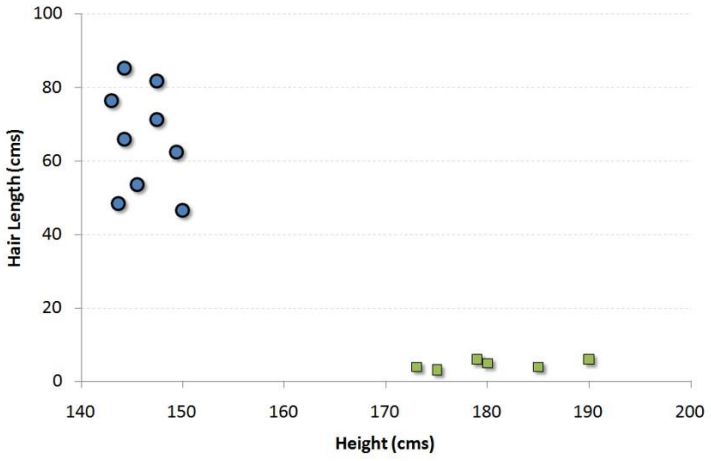


So, every time you split the room with a wall, you are trying to create 2 different populations with in the same room. Decision trees work in very similar fashion by dividing a population in as different groups as possible.

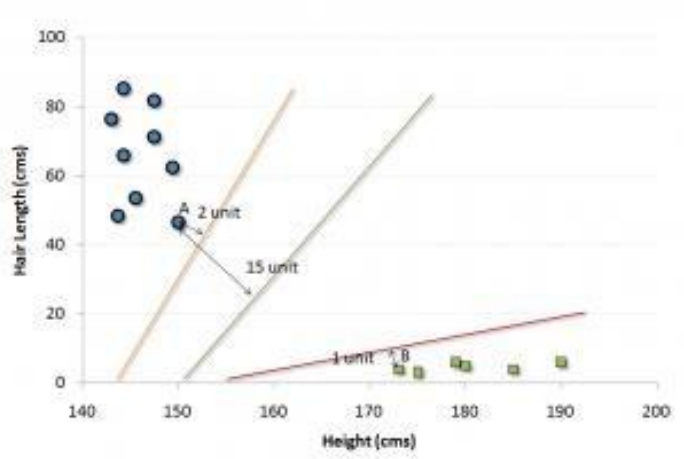
4. SVM (Support Vector Machine)

It is a classification method. In this algorithm, we plot each data item as a point in n-dimensional space (where n is number of features you have) with the value of each feature being the value of a particular coordinate.

For example, if we only had two features like Height and Hair length of an individual, we'd first plot these two variables in two dimensional space where each point has two co-ordinates (these co-ordinates are known as **Support Vectors**)



Now, we will find some *line* that splits the data between the two differently classified groups of data. This will be the line such that the distances from the closest point in each of the two groups will be farthest away.



In the example shown above, the line which splits the data into two differently classified groups is the *black* line, since the two closest points are the farthest apart from the line. This line is our classifier. Then, depending on where the testing data lands on either side of the line, that's what class we can classify the new data as.

Think of this algorithm as playing JezzBall in n-dimensional space. The tweaks in the game are:

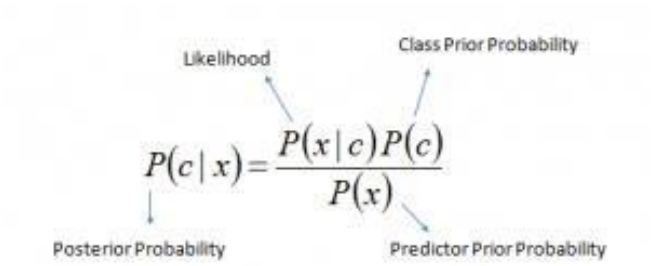
You can draw lines / planes at any angles (rather than just horizontal or vertical as in classic game)
The objective of the game is to segregate balls of different colors in different rooms.
And the balls are not moving.

5. Naive Bayes

It is a classification technique based on Bayes' theorem with an assumption of independence between predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. For example, a fruit may be considered to be an apple if it is red, round, and about 3 inches in diameter. Even if these features depend on each other or upon the existence of the other features, a naive Bayes classifier would consider all of these properties to independently contribute to the probability that this fruit is an apple.

Naive Bayesian model is easy to build and particularly useful for very large data sets. Along with simplicity, Naive Bayes is known to outperform even highly sophisticated classification methods.

Bayes theorem provides a way of calculating posterior probability P(c|x) from P(c), P(x) and P(x|c). Look at the equation below:



$$P(c \mid X) = P(x_1 \mid c) \times P(x_2 \mid c) \times \cdots \times P(x_n \mid c) \times P(c)$$

Here,

$P(c|x)$ is the posterior probability of *class* (*target*) given *predictor* (*attribute*).
$P(c)$ is the prior probability of *class*.
$P(x|c)$ is the likelihood which is the probability of *predictor* given *class*.
$P(x)$ is the prior probability of *predictor*.

Example: Let's understand it using an example. Below I have a training data set of weather and corresponding target variable 'Play'. Now, we need to classify whether players will play or not based on weather condition. Let's follow the below steps to perform it.

Step 1: Convert the data set to frequency table

Step 2: Create Likelihood table by finding the probabilities like Overcast probability = 0.29 and probability of playing is 0.64.

Weather	Play
Sunny	No
Overcast	Yes
Rainy	Yes
Sunny	Yes
Sunny	Yes
Overcast	Yes
Rainy	No
Rainy	No
Sunny	Yes
Rainy	Yes
Sunny	No
Overcast	Yes
Overcast	Yes
Rainy	No

Frequency Table		
Weather	No	Yes
Overcast		4
Rainy	3	2
Sunny	2	3
Grand Total	5	9

Likelihood table				
Weather	No	Yes		
Overcast		4	=4/14	0.29
Rainy	3	2	=5/14	0.36
Sunny	2	3	=5/14	0.36
All	5	9		
	=5/14	=9/14		
	0.36	0.64		

Step 3: Now, use Naive Bayesian equation to calculate the posterior probability for each class. The class with the highest posterior probability is the outcome of prediction.

Problem: Players will pay if weather is sunny, is this statement is correct?

We can solve it using above discussed method, so P (Yes | Sunny) = P (Sunny | Yes) * P(Yes) / P (Sunny)

Here we have P (Sunny |Yes) = 3/9 = 0.33, P(Sunny) = 5/14 = 0.36, P(Yes)= 9/14 = 0.64

Now, P (Yes | Sunny) = 0.33 * 0.64 / 0.36 = 0.60, which has higher probability.

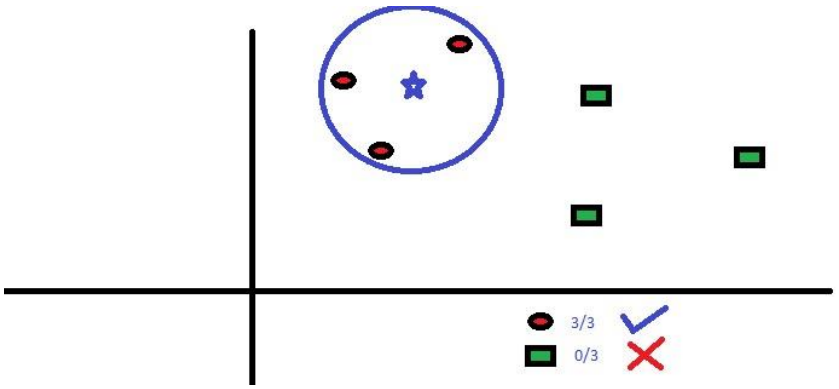
Naive Bayes uses a similar method to predict the probability of different class based on various attributes. This algorithm is mostly used in text classification and with problems having multiple classes.

6. kNN (k- Nearest Neighbors)

It can be used for both classification and regression problems. However, it is more widely used in classification problems in the industry. K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases by a majority vote of its k neighbors. The case being assigned to the class is most common amongst its K nearest neighbors measured by a distance function.

These distance functions can be Euclidean, Manhattan, Minkowski and Hamming distance. First three functions are used for continuous function and fourth one (Hamming) for categorical variables. If K = 1, then the case is simply assigned to the class of its nearest neighbor. At times, choosing K turns out to be a challenge while performing kNN modeling.

More: Introduction to k-nearest neighbors : Simplified.



KNN can easily be mapped to our real lives. If you want to learn about a person, of whom you have no information, you might like to find out about his close friends and the circles he moves in and gain access to his/her information!

Things to consider before selecting kNN:

KNN is computationally expensive

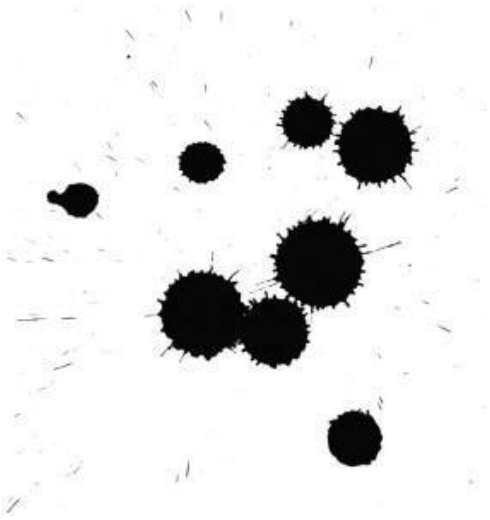
Variables should be normalized else higher range variables can bias it

Works on pre-processing stage more before going for kNN like outlier, noise removal

7. K-Means

It is a type of unsupervised algorithm which solves the clustering problem. Its procedure follows a simple and easy way to classify a given data set through a certain number of clusters (assume k clusters). Data points inside a cluster are homogeneous and heterogeneous to peer groups.

Remember figuring out shapes from ink blots? k means is somewhat similar this activity. You look at the shape and spread to decipher how many different clusters / population are present!



How K-means forms cluster:

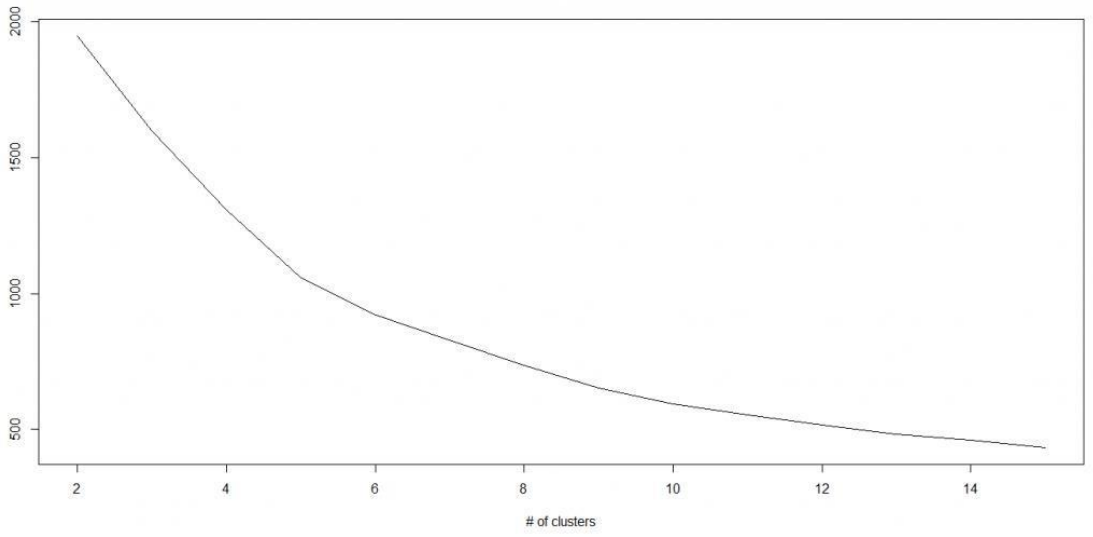
1. K-means picks k number of points for each cluster known as centroids.
2. Each data point forms a cluster with the closest centroids i.e. k clusters.
3. Finds the centroid of each cluster based on existing cluster members. Here we have new centroids.
4. As we have new centroids, repeat step 2 and 3. Find the closest distance for each data point from new centroids and get associated with new k-clusters. Repeat this process until convergence occurs i.e. centroids does not change.

How to determine value of K :

In K-means, we have clusters and each cluster has its own centroid. Sum of square of difference between centroid and the data points within a cluster constitutes within sum of square value for that cluster. Also, when

the sum of square values for all the clusters are added, it becomes total within sum of square value for the cluster solution.

We know that as the number of cluster increases, this value keeps on decreasing but if you plot the result you may see that the sum of squared distance decreases sharply up to some value of k, and then much more slowly after that. Here, we can find the optimum number of cluster.



8. Random Forest

Random Forest is a trademark term for an ensemble of decision trees. In Random Forest, we've collection of decision trees (so known as "Forest"). To classify a new object based on attributes, each tree gives a classification and we say the tree "votes" for that class. The forest chooses the classification having the most votes (over all the trees in the forest).

Each tree is planted & grown as follows:

1. If the number of cases in the training set is N, then sample of N cases is taken at random but *with replacement*. This sample will be the training set for growing the tree.
2. If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M and the best split on these m is used to split the node. The value of m is held constant during the forest growing.
3. Each tree is grown to the largest extent possible. There is no pruning.

9. Dimensionality Reduction Algorithms

In the last 4-5 years, there has been an exponential increase in data capturing at every possible stages. Corporates/ Government Agencies/ Research organizations are not only coming with new sources but also they are capturing data in great detail.

For example: E-commerce companies are capturing more details about customer like their demographics, web crawling history, what they like or dislike, purchase history, feedback and many others to give them personalized attention more than your nearest grocery shopkeeper.

As a data scientist, the data we are offered also consist of many features, this sounds good for building good robust model but there is a challenge. How'd you identify highly significant variable(s) out 1000 or 2000? In such cases, dimensionality reduction algorithm helps us along with various other algorithms like Decision Tree, Random Forest, PCA, Factor Analysis, identify based on correlation matrix, missing value ratio and others.

Tools and Packages

Python 3

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). Python is named after a TV Show called 'Monty Python's Flying Circus' and not after Python-the snake.

Python 3.0 was released in 2008. Although this version is supposed to be backward incompatibles, later on many of its important features have been backported to be compatible with version 2.7. This tutorial gives enough understanding on Python 3 version programming language.

Why Python?

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently whereas the other languages use punctuations. It has fewer syntactical constructions than other languages.

Python is Interpreted − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP

Python is Interactive − You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python is Object-Oriented − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.

Python is a Beginner's Language − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

Python was developed by Guido van Rossum in the late eighties and early nineties at the National Research Institute for Mathematics and Computer Science in the Netherlands.

Python is derived from many other languages, including ABC, Modula-3, C, C++, Algol-68, SmallTalk, and Unix shell and other scripting languages.

Python is copyrighted. Like Perl, Python source code is now available under the GNU General Public License (GPL).

Python is now maintained by a core development team at the institute, although Guido van Rossum still holds a vital role in directing its progress.

Python 1.0 was released in November 1994. In 2000, Python 2.0 was released. Python 2.7.11 is the latest edition of Python 2.

Meanwhile, Python 3.0 was released in 2008. Python 3 is not backward compatible with Python 2. The emphasis in Python 3 had been on the removal of duplicate programming constructs and modules so that "There should be one -- and preferably only one -- obvious way to do it." Python 3.5.1 is the latest version of Python 3.

Python Features

Python's features include −

Easy-to-learn − Python has few keywords, simple structure, and a clearly defined syntax. This allows a student to pick up the language quickly.

Easy-to-read − Python code is more clearly defined and visible to the eyes.

Easy-to-maintain − Python's source code is fairly easy-for-maintaining.

A broad standard library − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.

Interactive Mode − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.

Portable − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.

Extendable − You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.

Databases − Python provides interfaces to all major commercial databases.

GUI Programming − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

Scalable − Python provides a better structure and support for large programs than shell scripting.

Apart from the above-mentioned features, Python has a big list of good features. A, few are listed below −

It supports functional and structured programming methods as well as OOP.

It can be used as a scripting language or can be compiled to byte-code for building large applications.

It provides very high-level dynamic data types and supports dynamic type checking.

It supports automatic garbage collection.

It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Scikit-learn(sklearn)

Scikit-learn (formerly **scikits.learn**) is a free open source software for 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.

The scikit-learn project started as scikits.learn, a Google Summer of Code project by David Cournapeau. Its name stems from the notion that it is a "SciKit" (SciPy Toolkit), a separately-developed and distributed third-party extension to SciPy. The original codebase was later rewritten by other developers. In 2010 Fabian

Pedregosa, Gael Varoquaux, Alexandre Gramfort and Vincent Michel, all from INRIA took leadership of the project and made the first public release on February the 1st 2010. Of the various scikits, scikit-learn as well as scikit-image were described as "well-maintained and popular" in November 2012. As of 2018, scikit-learn is under active development.

Natural Language Tool Kit(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, sentiment analysis, opinion mining, 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.

```
C:\Users\HP>conda install -c anaconda nltk
Solving environment: done

## Package Plan ##

  environment location: C:\Users\HP\Anaconda3

  added / updated specs:
    - nltk


The following packages will be downloaded:

    package                    |            build
    ---------------------------|-----------------
    ca-certificates-2018.03.07 |                0         155 KB  anaconda
    openssl-1.1.1a             |       he774522_0         5.7 MB  anaconda
    ------------------------------------------------------------
                                           Total:         5.9 MB

The following packages will be UPDATED:

    ca-certificates: 2018.03.07-0      --> 2018.03.07-0      anaconda
    openssl:         1.1.1a-he774522_0 --> 1.1.1a-he774522_0 anaconda

Proceed ([y]/n)?


Downloading and Extracting Packages
ca-certificates-2018 | 155 KB    | ############################################################################ | 100%
openssl-1.1.1a       | 5.7 MB    | ############################################################################ | 100%
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
```

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and I Python shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits.

Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatterplots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

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.

The core functionality of NumPy is its "ndarray", for *n*-dimensional array, data structure. These arrays are strided views on memory. In contrast to Python's built-in list data structure (which, despite the name, is a dynamic array), these arrays are homogeneously typed: all elements of a single array must be of the same type.

Such arrays can also be views into memory buffers allocated by C/C++, Cython, and Fortran extensions to the CPython interpreter without the need to copy data around, giving a degree of compatibility with existing numerical libraries. This functionality is exploited by the SciPy package, which wraps a number of such libraries (notably BLAS and LAPACK). NumPy has built-in support for memory-mappedndarrays.

Problem Statement

To addresses the problem of sentiment analysis in twitter; that is classifying tweets according to the sentiment expressed in them: positive, negative for respective number of countries.

Hardware and Software Requirements:

- **Hardware Requirement:**
    1. **Processor- dual core**
    2. **RAM- 2gb**
    3. **Disk space -128 GB**
- **Software Requirement:**
    1. **Anaconda**
    2. **Jupyter**
    3. **Python 3**
    4. **NLTK**

Feasibility Study

A feasibility study is a preliminary study which investigates the information of prospective users and determines the resources requirements, costs, benefits and feasibility of proposed system. A feasibility study takes into account various constraints within which the system should be implemented and operated. In this stage, the resource needed for the implementation such as computing equipment, manpower and costs are estimated.

Hardware Feasibility:

In terms of Hardware, the project is very latent as any processor with 2 cores. 4 GB of RAM and quite minimal Secondary memory like HDD is required to complete the computation Although any other configuration with similar features can run the computation quite effectively, Higher configurations can run it even faster_ Hence, this project is highly feasible in terms of hardware.

Software Feasibility:

Any computer system having the required software can run the program and compute the values quite eminently_ A limiting factor is the number of tweets the Twitter API lets the user access per unit time. Hence, this project is highly feasible in terms of software.

Economic Feasibility:

This project was done with the use of Open-Source Licenses and hence the cost of making this project was totally free in terms of financial outlets. Although the request to use the Twitter API was the main time-consuming factor, it turned out to be free of cost making the project is highly feasible economically.

Methodology

Naive Bayes Classifier

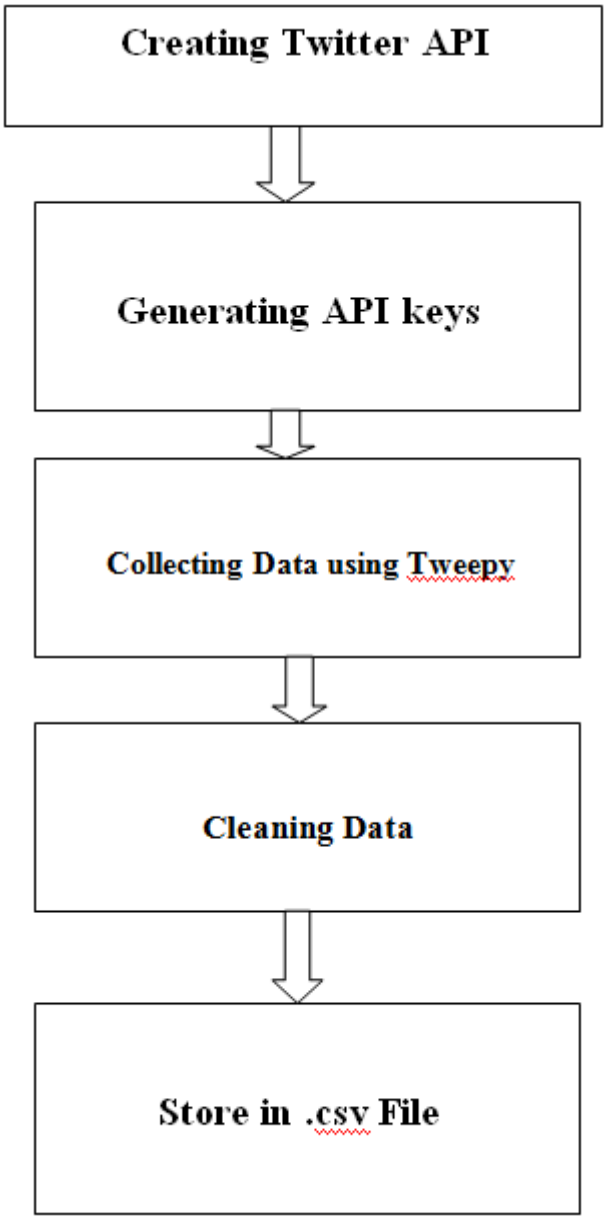
Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e.- every pair of features being classified is independent of each other.
Feature matrix contains all the vectors(rows) of dataset in which each vector consists of the value of dependent features_ In our dataset- features are •positive', and 'negative'

Decision tree

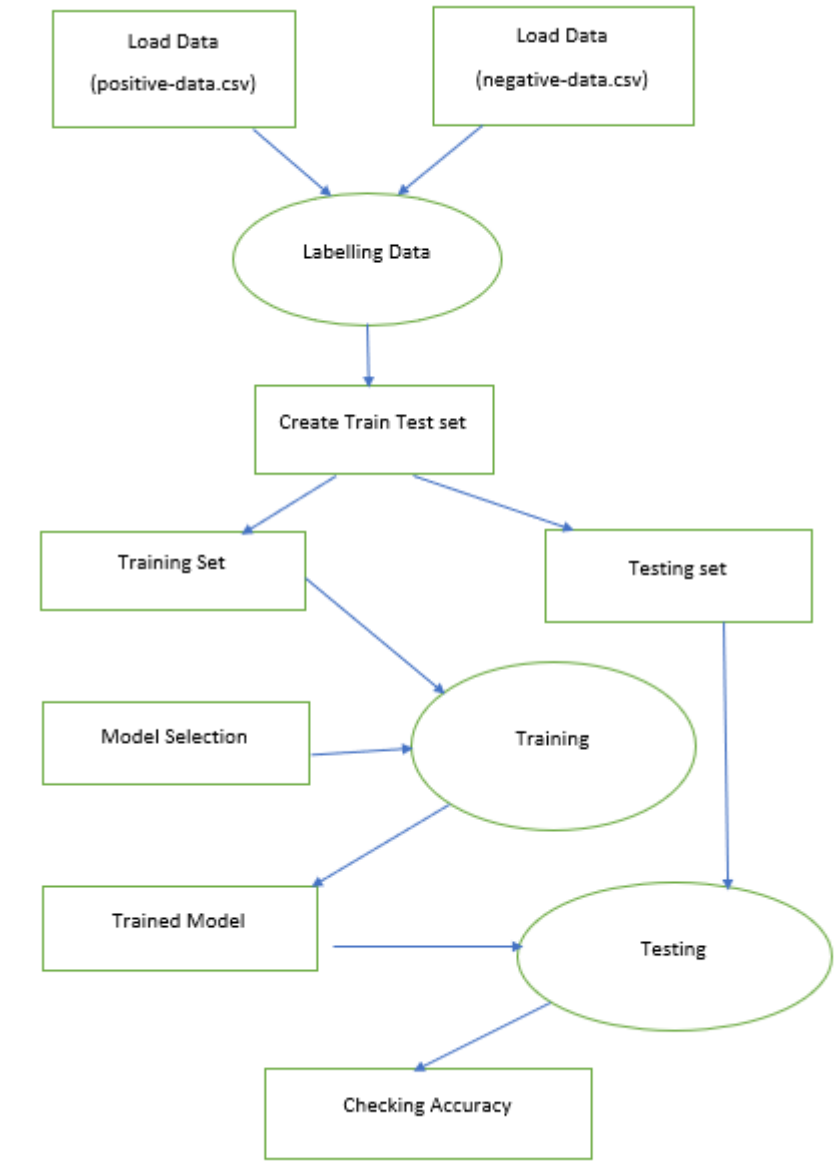
A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility_ It is one way to display an algorithm that only contains conditional control statements. A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules.

Flow-Chart

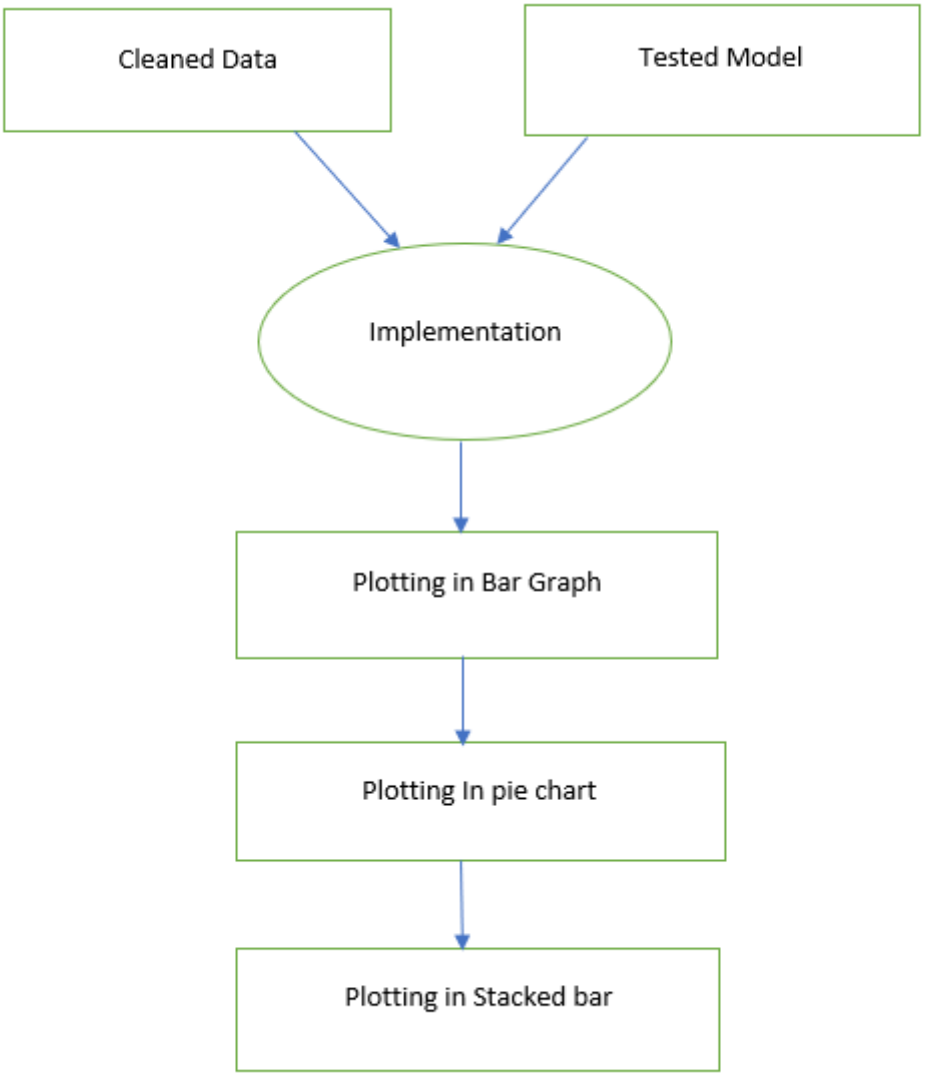
Data Gathering:



Training & Testing:



Implementation:

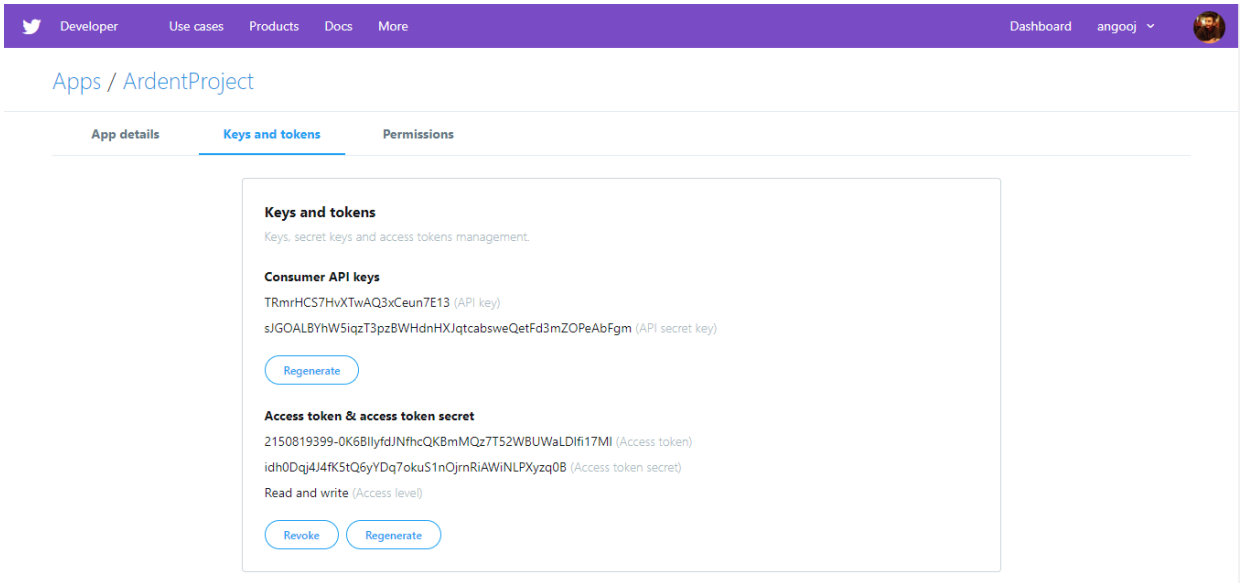


Implementation

Tweets are generally saved in json format. Opening a json file looks like a python 'dictionary'. 'Tweepy' is a package that helps us to collect data directly in a '.csv' file and download it or use it. So, first we're going to install tweepy.

```
Collecting tweepy
  Downloading https://files.pythonhosted.org/packages/d5/5f/daac4b4e9b30d7d2a6fdd16a880ff79f27918fe388e4dfc1983dec3a9876/tweepy-3.7.0-py2.py3-none-any.whl
Collecting requests-oauthlib>=0.7.0 (from tweepy)
  Downloading https://files.pythonhosted.org/packages/c2/e2/9fd03d55ffb70fe51f587f20bcf407a6927eb121de86928b34d162f0b1ac/requests_oauthlib-1.2.0-py2.py3-none-any.whl
Requirement already satisfied: six>=1.10.0 in c:\users\hp\anaconda3\lib\site-packages (from tweepy) (1.12.0)
Requirement already satisfied: requests>=2.11.1 in c:\users\hp\anaconda3\lib\site-packages (from tweepy) (2.21.0)
Requirement already satisfied: PySocks>=1.5.7 in c:\users\hp\anaconda3\lib\site-packages (from tweepy) (1.6.8)
Collecting oauthlib>=3.0.0 (from requests-oauthlib>=0.7.0->tweepy)
  Downloading https://files.pythonhosted.org/packages/b8/03/ec2be6c125f330361afe33bff48ef7549c47da8fefc658a2df2242175b4b/oauthlib-3.0.0-py2.py3-none-any.whl (142kB)
    100% |████████████████████████████████| 143kB 211kB/s
Requirement already satisfied: certifi>=2017.4.17 in c:\users\hp\anaconda3\lib\site-packages (from requests>=2.11.1->tweepy) (2018.11.29)
Requirement already satisfied: urllib3<1.25,>=1.21.1 in c:\users\hp\anaconda3\lib\site-packages (from requests>=2.11.1->tweepy) (1.24.1)
Requirement already satisfied: idna<2.9,>=2.5 in c:\users\hp\anaconda3\lib\site-packages (from requests>=2.11.1->tweepy) (2.8)
Requirement already satisfied: chardet<3.1.0,>=3.0.2 in c:\users\hp\anaconda3\lib\site-packages (from requests>=2.11.1->tweepy) (3.0.4)
Installing collected packages: oauthlib, requests-oauthlib, tweepy
Successfully installed oauthlib-3.0.0 requests-oauthlib-1.2.0 tweepy-3.7.0
```

To perform sentiment analysis in twitter basically we need to have a twitter account. But for authentication we need to create a twitter app first. You can create your twitter app by clicking on https://apps.twitter.com/.



After creating your own app there, you'll be provided you **unique** *Consumer Key (API Key)*, *Consumer Secret (API Secret)*, *Access Token* and *Access Token Secret* which we will use to collect the data from twitter for sentiment analysis.

```
consumer_key = "TRmrHCS7HvXTwAQ3xCeun7E13"
consumer_secret = "sJGOALBYhW5iqzT3pzBWHdnHXJqtcabsweQetFd3mZOPeAbFgm"
access_key = "2150819399-0K6BIlyfdJNfhcQKBmMQz7T52WBUWaLDlfi17Ml"
access_secret = "idh0Dqj4J4fK5tQ6yYDq7okuS1nOjrnRiAWiNLPXyzq0B"
```

Here, we will import all necessary liberaries required and in next step we have downloaded “stopwords” keyword from nltk.

```
In [2]:
import nltk
from nltk.metrics import recall as recall
from nltk.metrics import precision as precision
from nltk.metrics import f_measure as f_measure
from nltk.corpus import stopwords
import collections
from nltk.classify import SklearnClassifier
from sklearn.svm import LinearSVC
```

```
In [3]:
nltk.download('stopwords')
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\HP\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.
```

```
Out[3]: True
```

Here, we import punctuation from string library:

```
from string import punctuation
```

Here, the function get_all_tweets collect all the tweet for a given screen_name and save it as a .csv file:

```
def get_all_tweets(screen_name):
    auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
    auth.set_access_token(access_key, access_secret)
    api = tweepy.API(auth)

    alltweets = []

    new_tweets = api.user_timeline(screen_name=screen_name, count=200)
    alltweets.extend(new_tweets)
    oldest = alltweets[-1].id - 1
    print("...%s tweets downloaded so far" % (len(alltweets)))
    c = 5

    while c > 0:
        # print("getting tweets before %s" % (oldest))
        new_tweets = api.user_timeline(screen_name = screen_name, count = 200, max_id = oldest)

        alltweets.extend(new_tweets)
        oldest = alltweets[-1].id - 1

        print("...%s tweets downloaded so far" % (len(alltweets)))
        c -= 1

    outtweets = [[tweet.text.encode("utf-8")] for tweet in alltweets]
  # write the csv inside particular folder

    with open('%s_tweets.csv' % screen_name, 'w') as f:
       writer = csv.writer(f)
       writer.writerows(outtweets)
```

The function clean_data clears all the special character,links from the collected tweet for a given filename:

```
def clean_data(filename):
    file = open(filename, 'rt')
    text = file.read()
    file.close()

    # split into words by white space
    words = text.split("\n")
    words = [word.lower() for word in words]
    print(words[:1])

    # remove punctuation from each word
    import string
    table = str.maketrans('', '', string.punctuation)
    stripped = [w.translate(table) for w in words]
    print(stripped[:1])

    # create cleaned file
    file = open("cleaned_"+filename, "w")
    for i in stripped:
        file.write(i+"\n")
    file.close()
```

Here we are removing all the stopwords like is ,and from the given tweets.

```
def word_split(data):
    data_new = []
    for word in data:
        word_filter = [i.lower() for i in word.split()]
        data_new.append(word_filter)
    return data_new

def word_feats(words):
    return dict([(word, True) for word in words])

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

def stopword_filtered_word_feats(words):
    return dict([(word, True) for word in words if word not in stopset])
```

Here we are creating two empty list posdata and negdata which will collect all positive and negative tweets and store it into positive-data.csv and negative-data.csv.

```
posdata = []
negdata = []
with open('positive-data.csv', 'r') as myfile:
    reader = csv.reader(myfile, delimiter=',')
    for col in reader:
        posdata.append(col[0])
with open('negative-data.csv', 'r') as myfile:
    reader = csv.reader(myfile, delimiter=',')
    for col in reader:
        negdata.append(col[0])
```

```
negfeats = [(stopword_filtered_word_feats(f), 'neg') for f in word_split(negdata)]
posfeats = [(stopword_filtered_word_feats(f), 'pos') for f in word_split(posdata)]
```


```
negfeats = [(stopword_filtered_word_feats(f), 'neg') for f in word_split(negdata)]
posfeats = [(stopword_filtered_word_feats(f), 'pos') for f in word_split(posdata)]
```


```
trainfeats = negfeats[:negcutoff] + posfeats[:poscutoff]
testfeats = negfeats[negcutoff:] + posfeats[poscutoff:]
```

Now training and testing part of the positive and negative data is implemented.

```
refsets = collections.defaultdict(set)
testsets = collections.defaultdict(set)
```

We are using Naïve Bayes algorithm. It is a classification technique based on Bayes' theorem with an assumption of independence between predictors. And downloading tweets for various screen_list.

```
classifier = nltk.NaiveBayesClassifier.train(trainfeats)
for i, (feats, label) in enumerate(testfeats):
    refsets[label].add(i)
    observed = classifier.classify(feats)
    testsets[observed].add(i)

accuracy = nltk.classify.util.accuracy(classifier, testfeats)*100
pos_precision = nltk.precision(refsets['pos'], testsets['pos'])
```

```
screen_list = ["india","china","usa","canada","russia","france"]
```

```
for i in range(len(screen_list)):
    get_all_tweets(screen_list[i])
```

```
...199 tweets downloaded so far
...260 tweets downloaded so far
...260 tweets downloaded so far
...260 tweets downloaded so far
...260 tweets downloaded so far
...260 tweets downloaded so far
...200 tweets downloaded so far
...400 tweets downloaded so far
...600 tweets downloaded so far
...800 tweets downloaded so far
...1000 tweets downloaded so far
...1200 tweets downloaded so far
...200 tweets downloaded so far
...400 tweets downloaded so far
...600 tweets downloaded so far
...800 tweets downloaded so far
...1000 tweets downloaded so far
...1200 tweets downloaded so far
...200 tweets downloaded so far
...400 tweets downloaded so far
...600 tweets downloaded so far
...800 tweets downloaded so far
...1000 tweets downloaded so far
...1200 tweets downloaded so far
...200 tweets downloaded so far
...400 tweets downloaded so far
...600 tweets downloaded so far
...800 tweets downloaded so far
...1000 tweets downloaded so far
...1200 tweets downloaded so far
```

Calling clean_data function.

```
for i in screen_list:
    clean_data(i+'_tweets.csv')
```

For each elements in the screen_list here we are counting number of positive and negative tweets.

```
p = []
n = []
for i in screen_list:
    my_data = []
    error = 0
    with open('cleaned_'+i+'_tweets.csv', 'r') as myfile:
        reader = csv.reader(myfile, delimiter=',')
        for col in reader:
            try:
                my_data.append(col[0])
            except:
                error+=1
    print("Total: ", len(my_data))
    print("Errors:", error)
    print("Sucess", len(my_data)- error)

    result = collections.defaultdict(set)
    my_data_feats = [(stopword_filtered_word_feats(f), 'pos') for f in word_split(my_data)]
    for i, (feats, label) in enumerate(my_data_feats):
        observed = classifier.classify(feats)
        result[observed].add(i)
    p.append(len(result['pos']))
    n.append(len(result['neg']))
```

```
for i in p:
    print(i)
```

```
124
335
669
460
560
657
```

```
Total:  260
Errors: 261
Sucess -1
Total:  1200
Errors: 1201
Sucess -1
Total:  1200
Errors: 1201
Sucess -1
Total:  1200
Errors: 1201
Sucess -1
Total:  1200
Errors: 1201
Sucess -1
Total:  1200
Errors: 1201
Sucess -1
```

```
for i in n:
    print(i)
```

```
136
865
531
740
640
543
```

Importing matplotlib library for ploting various chart based on tweets for visualization part.

```
import matplotlib.pyplot as plt
plt.figure(figsize=(12,8))
for i in range(len(screen_list)):
    plt.subplot(2,3,i+1)
    plt.bar(['pos','neg'],[p[i],n[i]],color = ['g','r'])
    plt.title('tweets for screen name : '+screen_list[i])
    plt.ylabel('number of tweets')
plt.show()
```

```
<Figure size 1200x800 with 6 Axes>
```

```
G_pos = []
for i in p:
    G_pos.append(i)
G_pos
```

```
[124, 335, 668, 460, 560, 661]
```

```
G_neg = []
for i in n:
    G_neg.append(i)
G_neg
```

```
[136, 865, 532, 740, 640, 539]
```

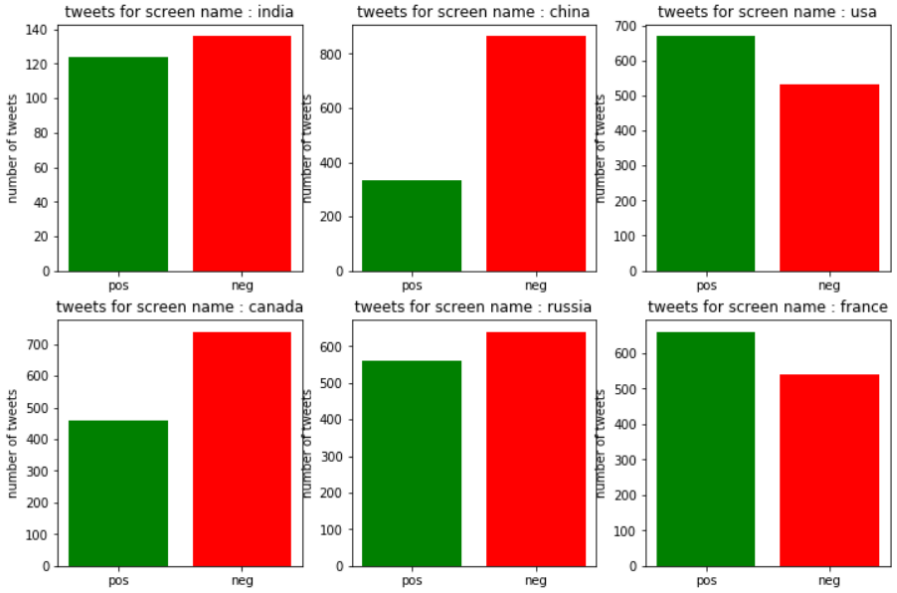
```
GT_pos = sum(G_pos)
GP_pos = [i/GT_pos for i in G_pos]
GP_pos
```

```
[0.04415954415954416,
 0.1193019943019943,
 0.2378917378917379,
 0.16381766381766383,
 0.19943019943019943,
 0.2353988603988604]
```

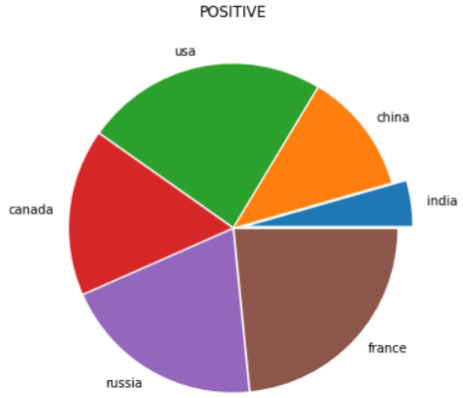
```
GT_neg = sum(G_neg)
GP_neg = [i/GT_neg for i in G_neg]
GP_neg
```

```
[0.039397450753186555,
 0.25057937427578214,
 0.1541135573580533,
 0.21436848203939746,
 0.1853997682502897,
 0.15614136732329084]
```

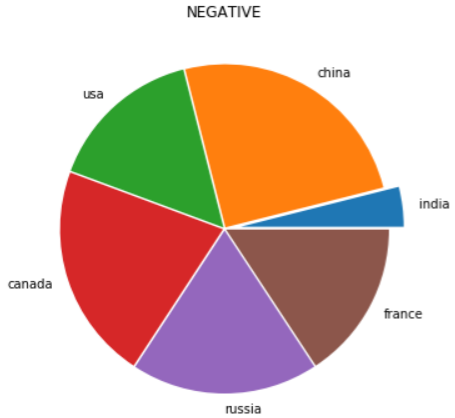
```
import matplotlib.pyplot as plt
plt.figure(figsize=(12,8))
for i in range(len(screen_list)):
    plt.subplot(2,3,i+1)
    plt.bar(['pos','neg'],[p[i],n[i]],color = ['g','r'])
    plt.title('tweets for screen name : '+screen_list[i])
    plt.ylabel('number of tweets')
plt.show()
```




```
plt.figure(1,figsize=(6,6))
plt.pie(GP_pos,labels=screen_list, explode=[0.1,0.01,0.01,0.01,0.01,0.01])
plt.title('POSITIVE')
plt.show()
```




```
plt.figure(1,figsize=(6,6))
plt.pie(GP_neg,labels=screen_list, explode=[0.1,0.01,0.01,0.01,0.01,0.01])
plt.title('NEGATIVE')
plt.show()
```



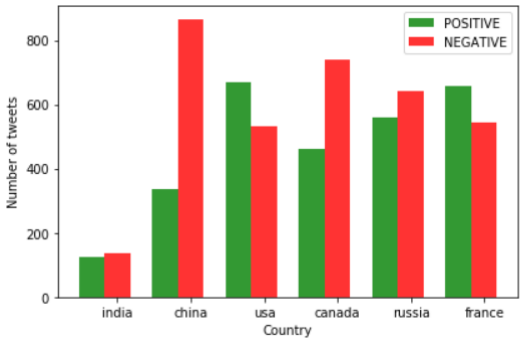
```
import numpy as np
n_groups = 6


# create plot
fig, ax = plt.subplots()
index = np.arange(n_groups)
bar_width = 0.35
opacity = 0.8

rects1 = plt.bar(index, p, bar_width,
                 alpha=opacity,
                 color='g',
                 label='POSITIVE')

rects2 = plt.bar(index + bar_width, n, bar_width,
                 alpha=opacity,
                 color='r',
                 label='NEGATIVE')
plt.xlabel('Country')
plt.ylabel('Number of tweets')
plt.xticks(index + bar_width, screen_list)
plt.legend()

plt.tight_layout()
plt.show()
```

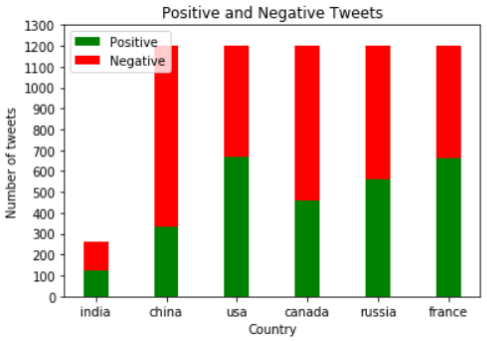



```
n_groups = 6
index = np.arange(n_groups)
width = 0.35

p1 = plt.bar(index, p, width, yerr=None, color='g')
p2 = plt.bar(index, n, width,
             bottom=p, yerr=None, color='r')

plt.xlabel('Country')
plt.ylabel('Number of tweets')
plt.title('Positive and Negative Tweets')
plt.xticks(index, screen_list)
plt.yticks(np.arange(0, 1301, 100))
plt.legend((p1[0], p2[0]), ('Positive', 'Negative'))

plt.show()
```



Conclusion

Twitter is a source of vast unstructured and noisy data sets that can be processed to locate interesting patterns and trends. We predicted the postive and negative views of towards six countries using twitter data with accuracy of 82% by using Naïve Bayes algorithm. Finally, we conclude that using different NLTK classifiers, it is easier to classify tweets and more we improve the training datasets more we increase its accuracy.

References

Following sources are used to implement this project. Some are official documentations; some are books written by researchers.

https://matplotlib.org/
http://www.numpy.org/
http://scikit-learn.org/stable/
https://www.python.org/
https://www.nltk.org/
Sentiment Analysis and Opinion Mining by Bing Liu