ML GROUP 19

SPAM/HAM Classification

Omkar Raghatwan- 191070059

Flavin Dabre- 201070905

Ishaan Shivhare - 191070070

Harshdeep Telang - 191070024

- Problem Statement
- Introduction
- Motivation
- Approach
- Brief Idea

Write your message

Problem Statement

Understanding SPAM or HAM classifier from the aspect of Machine Learning concepts, work with various classification algorithms, and select high accuracy producing algorithm and develop app for SMS: spam or ham detector.

Introduction

- Spam is the abuse of electronic messaging systems to send unsolicited messages in bulk indiscriminately.Spam emerges from greedy and malicious intent.
- SMS spam is used for commercial advertising and spreading phishing links. Commercial spammers use malware to send SMS spam because sending SMS spam is illegal in most countries.
- People classify SMS Spam as annoying, wasting time, and violating personal privacy.
- Due to all of its adverse consequences,there is a serious need to detect and deal with spam effectively.

Motivation

- Every time SMS spam arrives at a user's inbox, and the mobile phone alerts the user to the incoming message. When the user realizes that the message is unwanted, he or she will be disappointed, and also SMS spam takes up some of the mobile phone's storage.
- SMS spam detection is an important task where spam SMS messages are identified and filtered. As more significant numbers of SMS messages are communicated every day, it is challenging for a user to remember and correlate the newer SMS messages received in context to previously received SMS.
- Thus, using the knowledge of artificial intelligence with the amalgamation of machine learning, and data mining we will try to develop web-based SMS text spam or ham detector.

Approach

Our Approach is consists of two parts

PART 1:

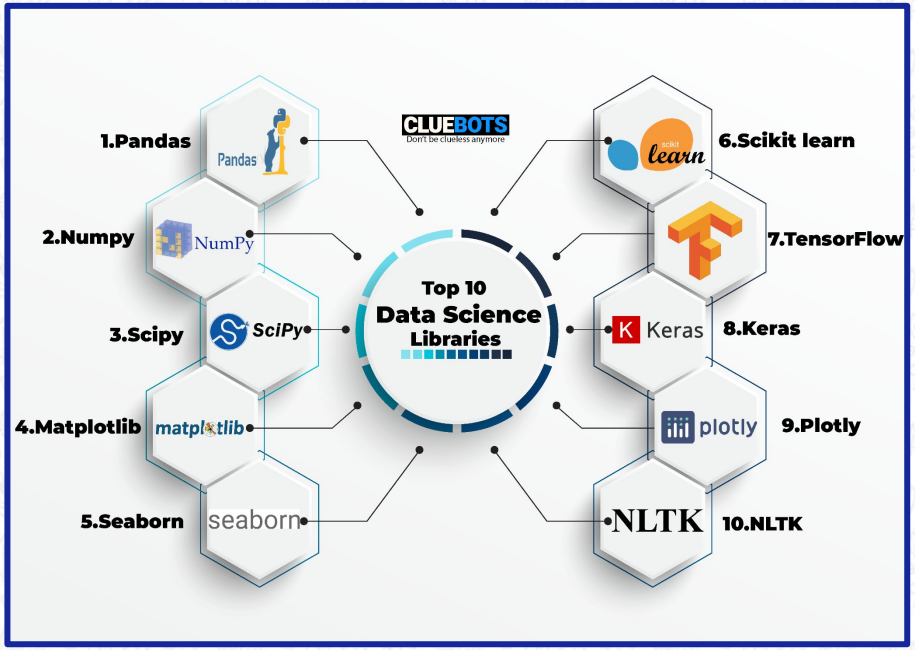
- Exploring Data Source
- Data Preparation
- Exploratory Data Analysis

PART 2:

- Naïve Bayes Behind Spam or Ham
- SVC On Spam or Ham
- KNN on Spam or Ham
- Performance Measurement Criterion

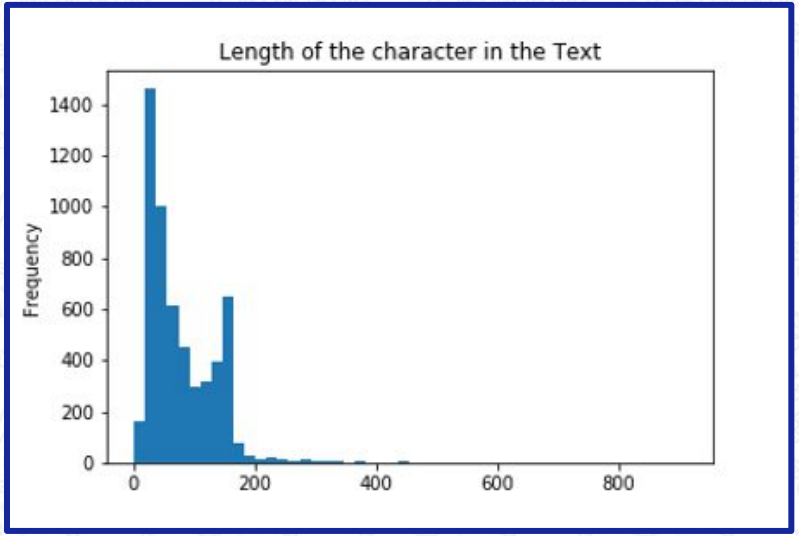
Python Libraries in Use:

- Pandas
- Matplotlib
- Sklearn
- Xgboost
- NLTK

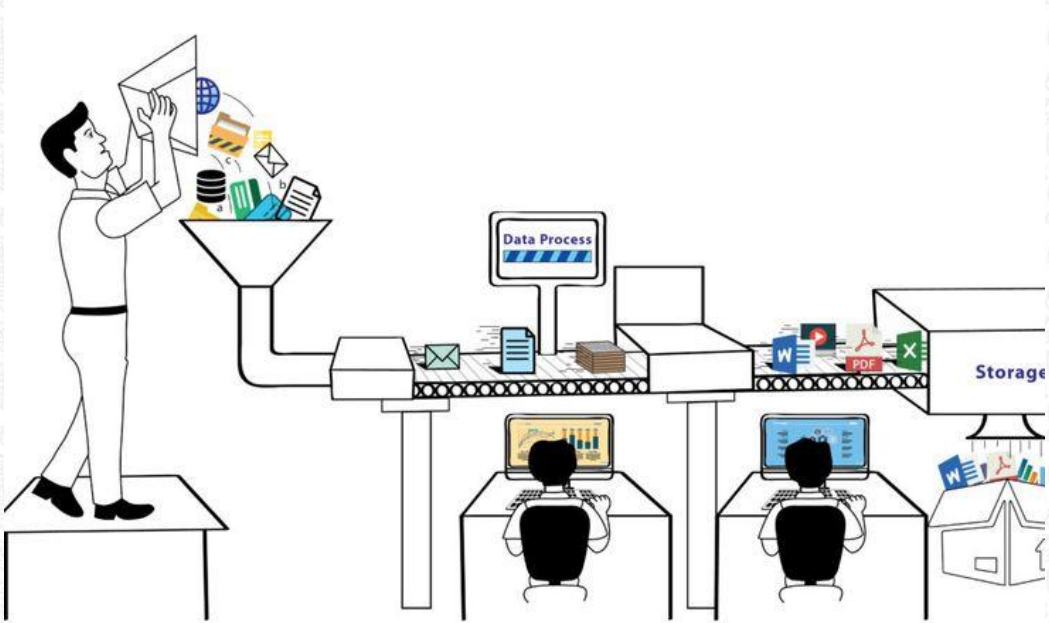


Dataset

- The dataset is taken from UCI Machine Learning.
- The data set has 5 columns and 5572 rows. Out of these 5572 data points, *type* of 747 is labeled as spam and 4825 as 'ham' and contain 3 extra column as *Unnamed:2/3/4* which is redundant.
- If the *type* has value ham, it means the text or message is not spam but if the value of *type* is spam then it means the *text* is spam and *text* are not in a chronological order.



Data Exploration & Pre-processing



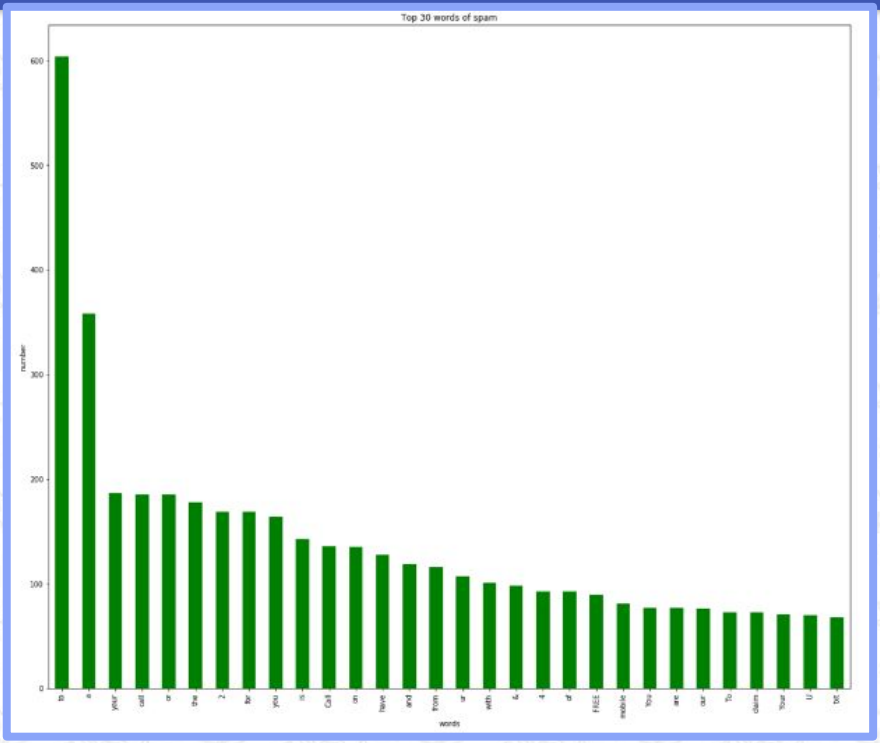
Data Exploration (EDA)

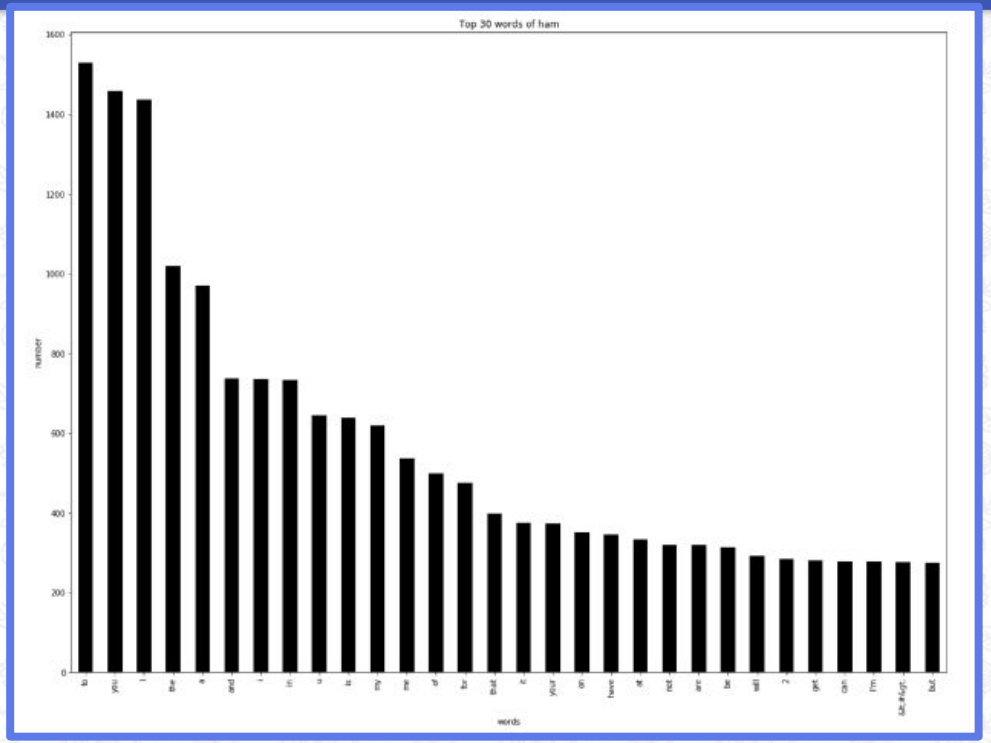
On the Exploratory Data Analysis (EDA) cycle we see:

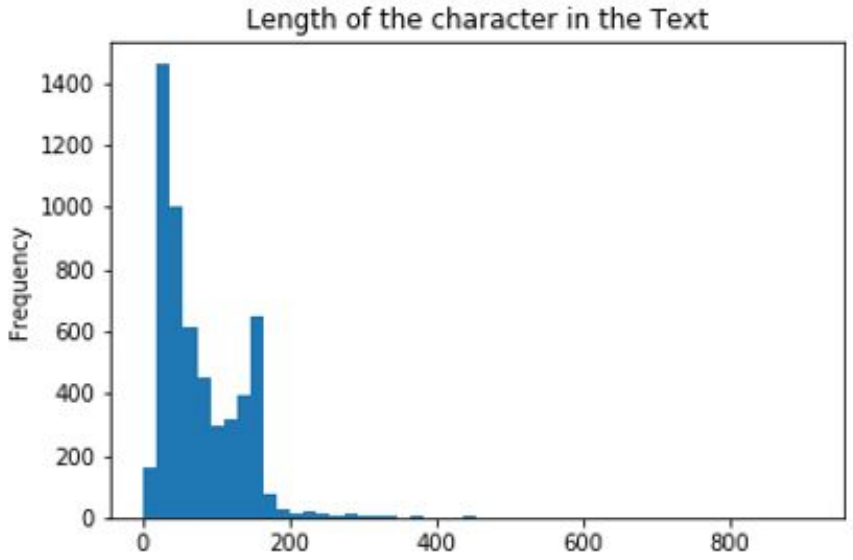
- the length character
- distribution of digits and non digits
- top 30 words of spam and ham words.

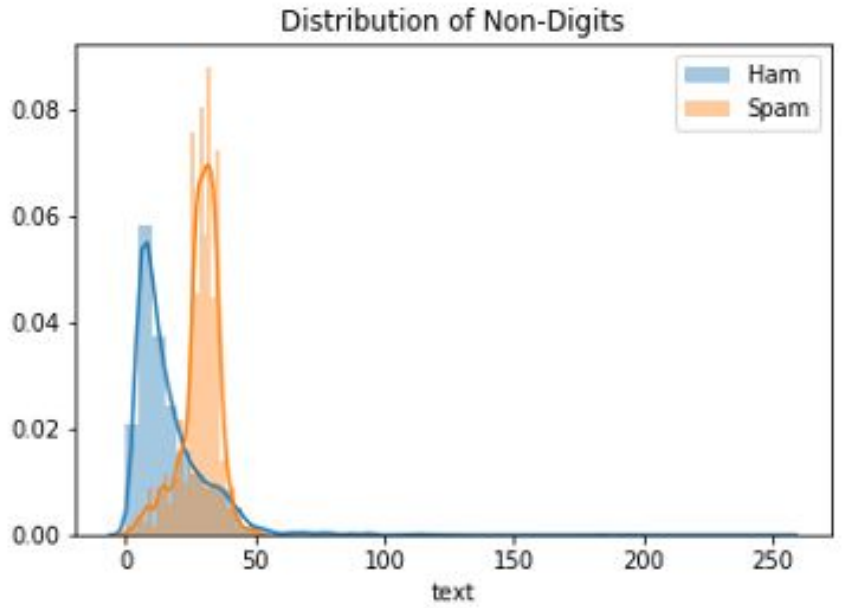
This allows us to gain an insight into the dataset and the approach we should take.

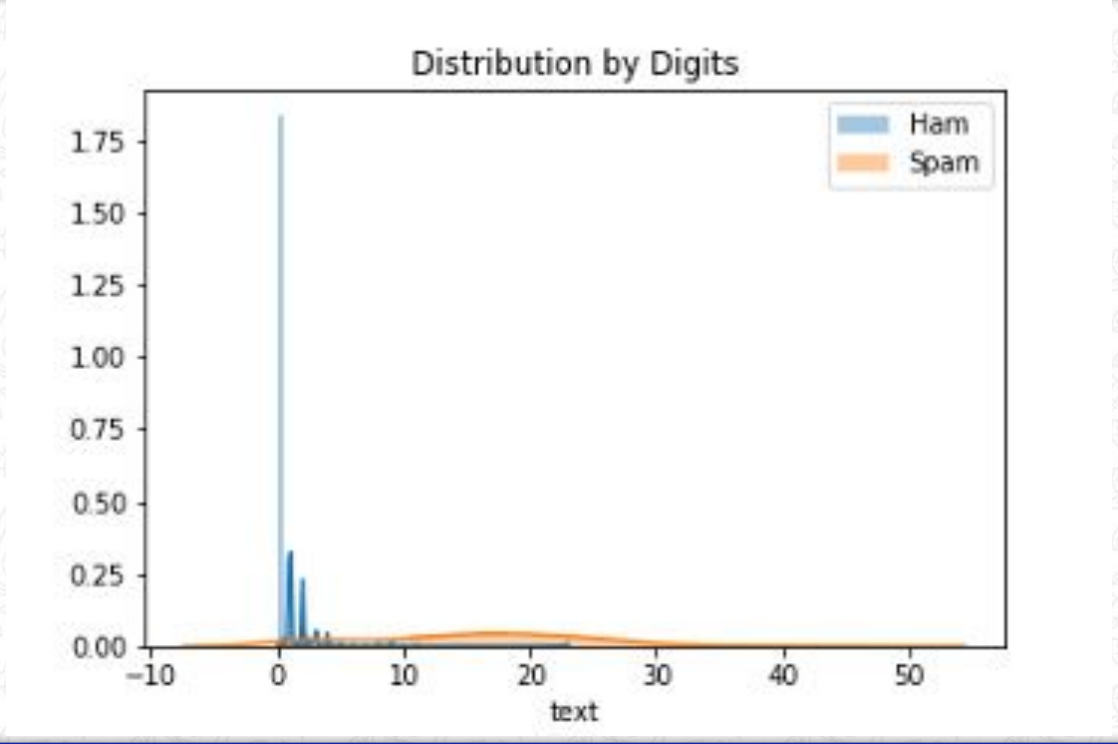
After preprocessing, vectorization and gaining some insights from EDA, the project goes for training a model and work with the various different classification algorithm.


Top 30 words of spam
number
words
0
100
200
300
400
500
600
to
a
your
call
or
the
2
for
you
is
Call
on
have
and
from
ur
with
4
i
of
FREE
mobile
You
are
our
To
claims
Your
U
txt


Top 30 words of ham
number
words
0
200
400
600
800
1000
1200
1400
1600
to
you
I
the
a
and
i
in
u
is
my
me
of
for
But
c
your
on
have
at
not
are
be
will
2
get
can
I'm
Ok..#Ugh..
but


Length of the character in the Text
Frequency
1400
1200
1000
800
600
400
200
0
0
200
400
600
800


Distribution of Non-Digits
Ham
Spam
0.08
0.06
0.04
0.02
0.00
0
50
100
150
200
250
text


Distribution by Digits
Ham
Spam
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
−10
0
10
20
30
40
50
text

Data Pre-processing

- **Tokenization** is the process of converting the normal text strings in to a list of tokens (words that we actually want).
- E.g:
  Raw message: “How are you doing?”
  Tokenized message: [How, are, you, doing,?]
- **Vectorization** is the process of converting each of the messages into a vector.

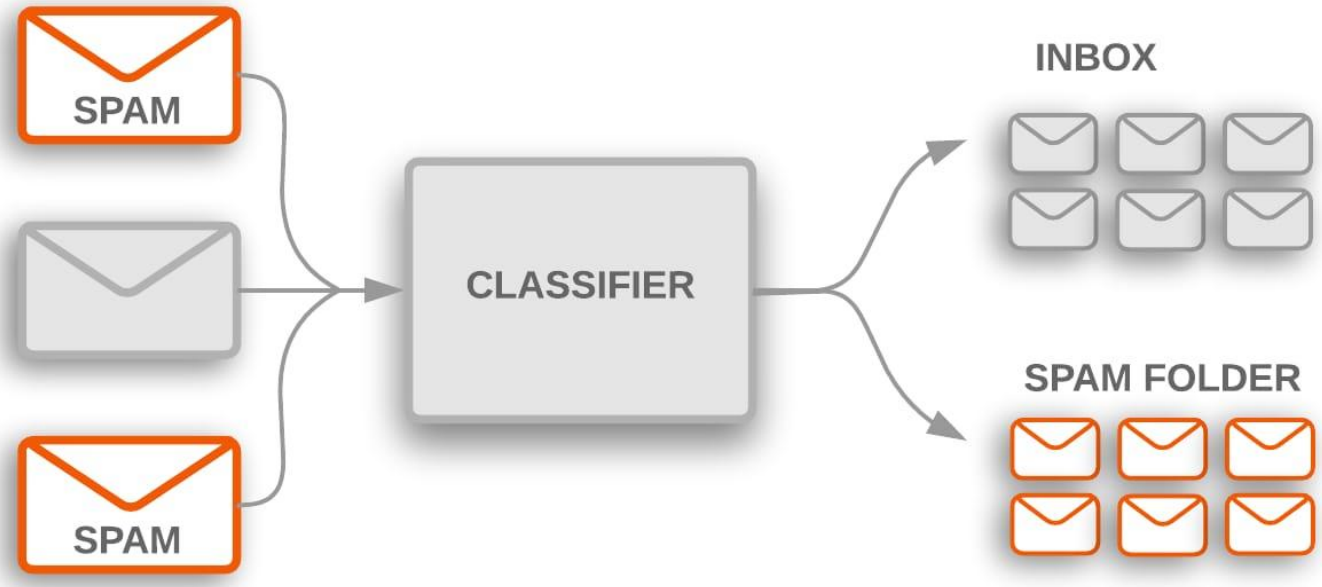
  Steps for Vectorization:

  1. Count how many times does a word occur in each message (Known as term frequency).
  2. Weigh the counts, so that frequent tokens get lower weight (inverse document frequency).
  3. Normalize the vectors to unit length, to abstract from the original text length (L2 norm) by TF-IDF.

TF-IDF

- **TF: Term Frequency, which measures how frequently a term occurs in a document.**
- TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document).
- **IDF: Inverse Document Frequency, which measures how important a term is.**
- While computing TF, all terms are considered equally important. However it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance.
- DF(t) = log_e(Total number of documents / Number of documents with term t in it).

CLASSIFICATION ALGORITHMS:



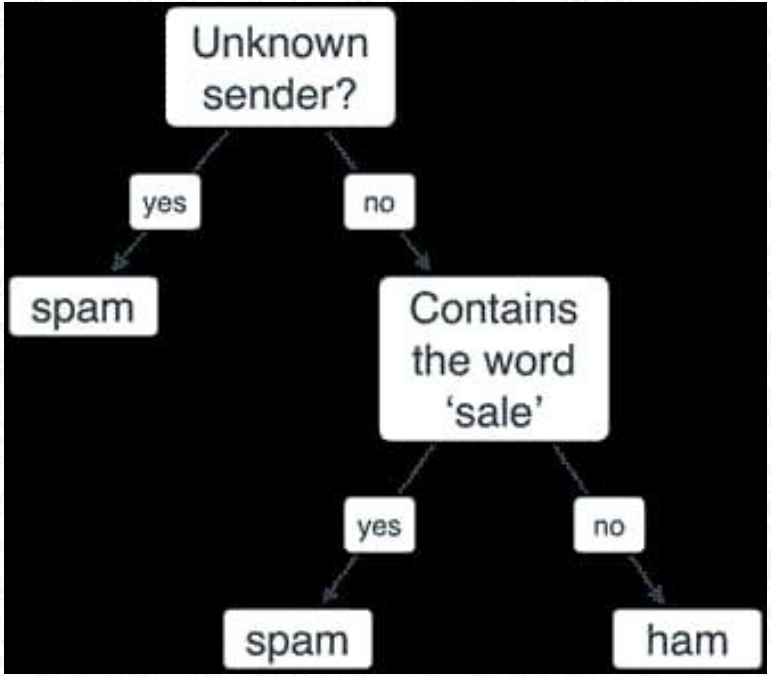
Decision Tree

A decision tree is a predictive machine-learning model that decides the target value (dependent variable) of a new sample based on various attribute values of the available data.

Accuracy: 97.3%

Decision Tree confusion_matrix:
[[1421 27]

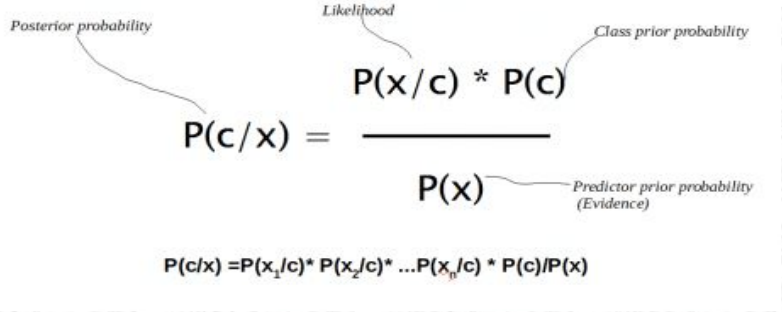
[18 206]]

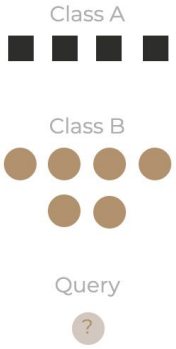


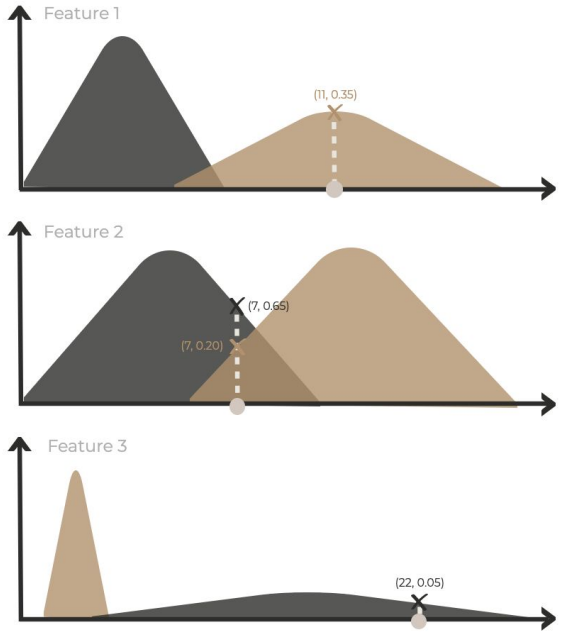
Naive Bayes

Naive Bayes uses the Bayes' Theorem and assumes that all predictors are independent. In other words, this classifier assumes that the presence of one particular feature in a class doesn't affect the presence of another one.

Accuracy: **98.**2655 %

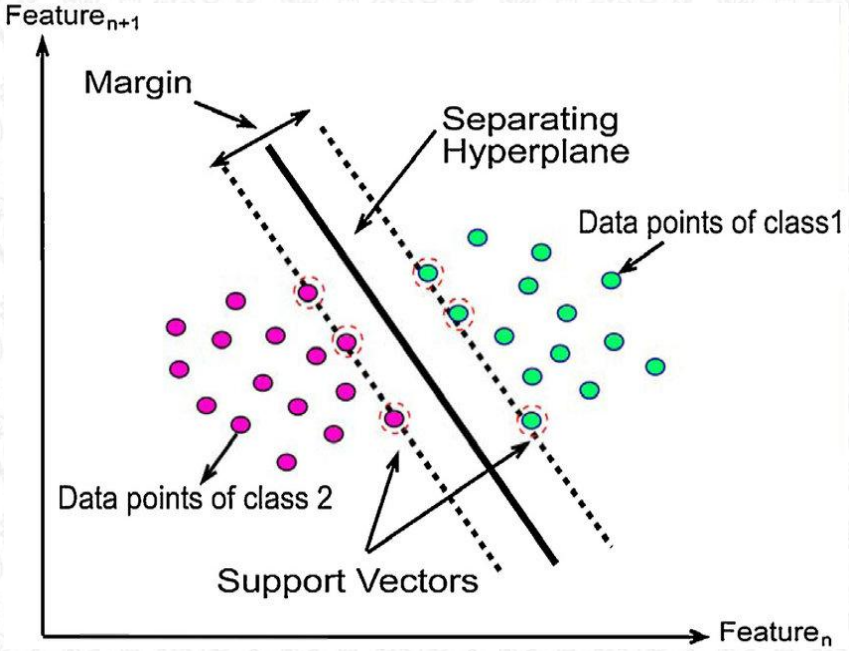






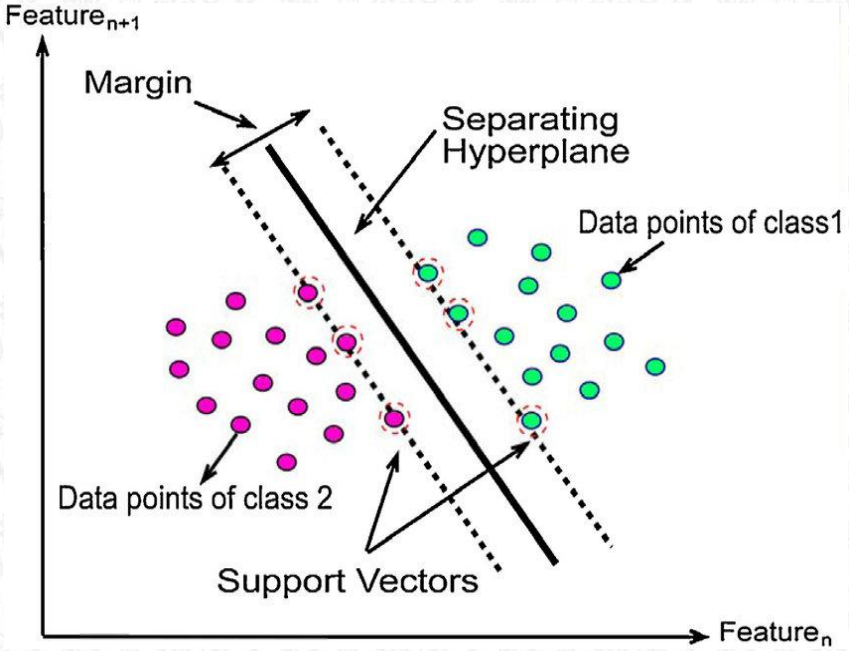
Support Vector Classification

- Comes under Supervised learning
- We use labeled dataset to train model
- We plot a line separating the two classes called decision boundary. The best boundary is called Hyperplane
- Then we draw 2 more lines parallel to hyperplane passing through the nearest points from the corresponding opponent classes



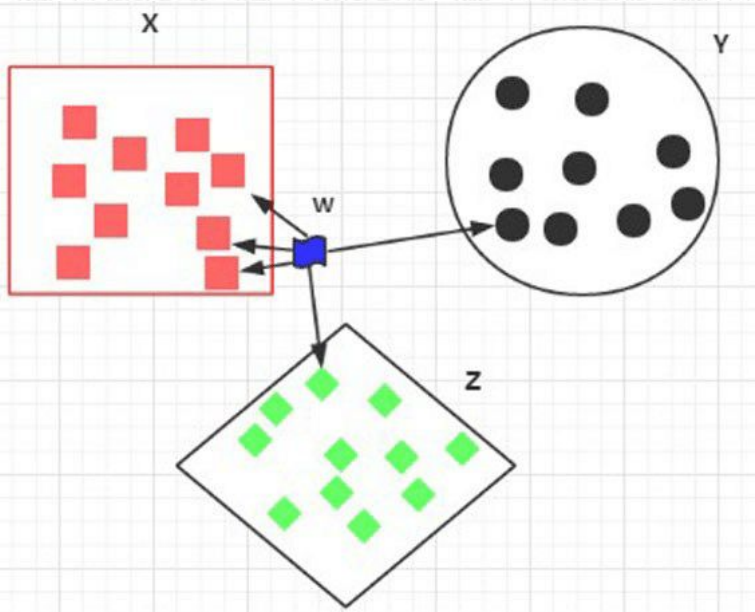
Support Vector Classification

- The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as **Support Vector**.
- Distance between the vectors and the hyperplane is called as **margin**.
- Goal of SVM is to maximize this margin. The hyperplane with maximum margin is called the **optimal hyperplane**.



K-Nearest Neighbour

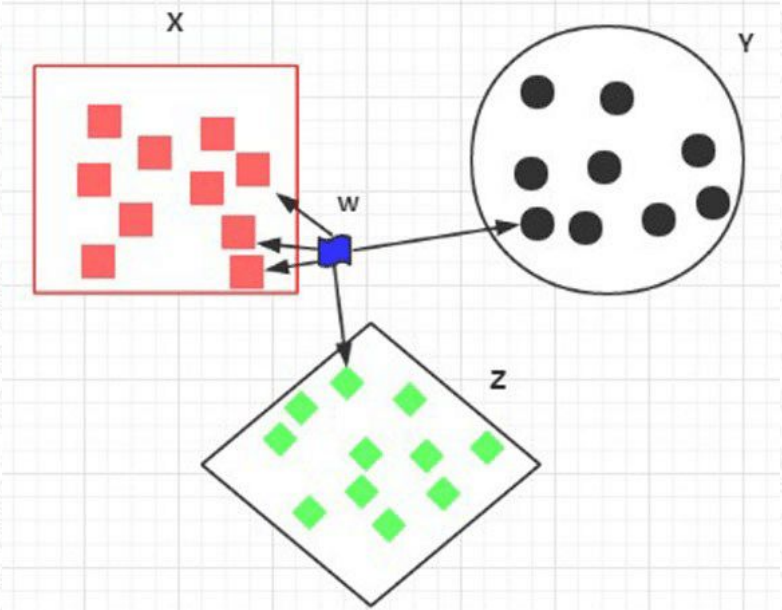
- K-Nearest Neighbors based classification is a type of lazy learning
- It does not attempt to construct a general internal model but only stores instances of the training data.
- Classification is computed from a simple majority vote of the k nearest neighbors of each point.
- This algorithm is simple to implement, robust to noisy training data, and useful if training data is extensive.



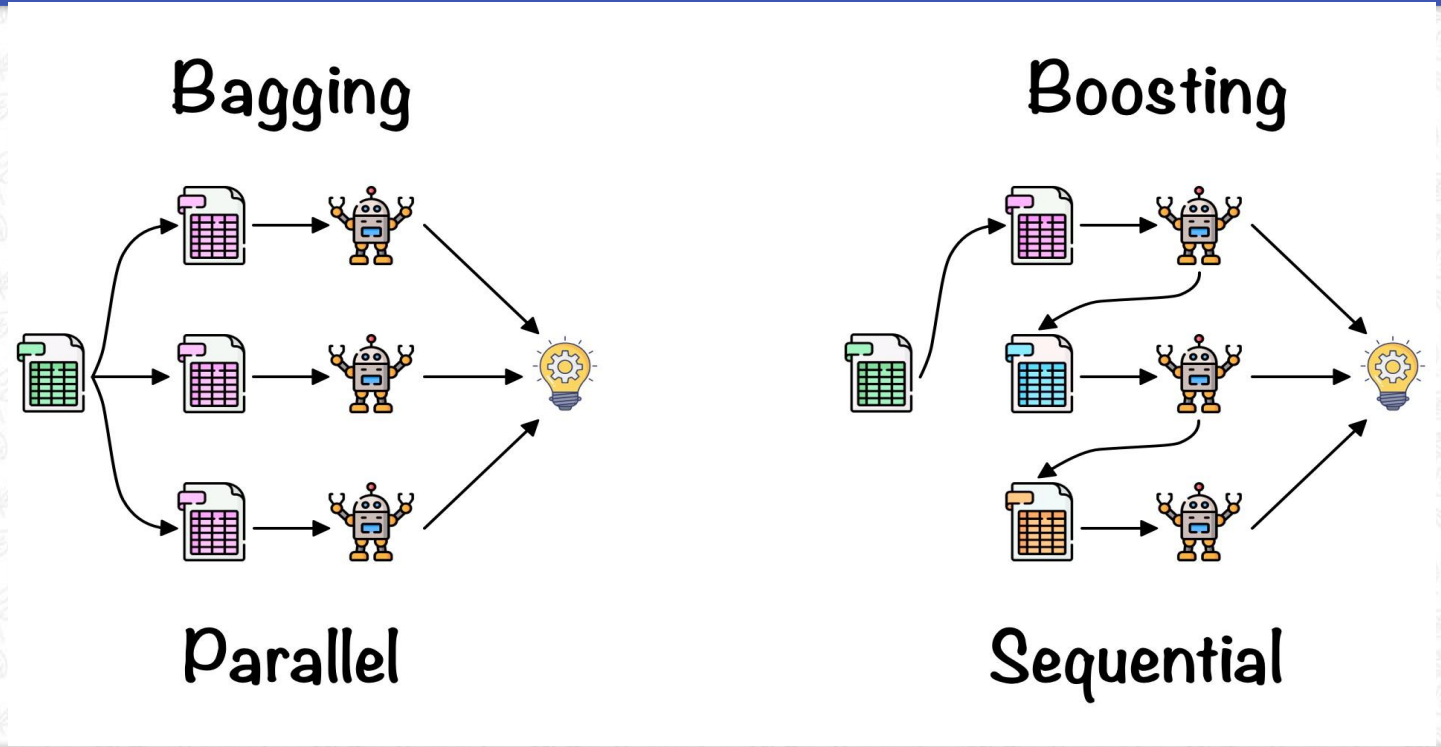
K-Nearest Neighbour

- The disadvantage is that it needs to determine the value of K each time
- The computation cost is high as it needs to compute the distance of each instance to all the training samples.

- Accuracy achieved: 86.06%

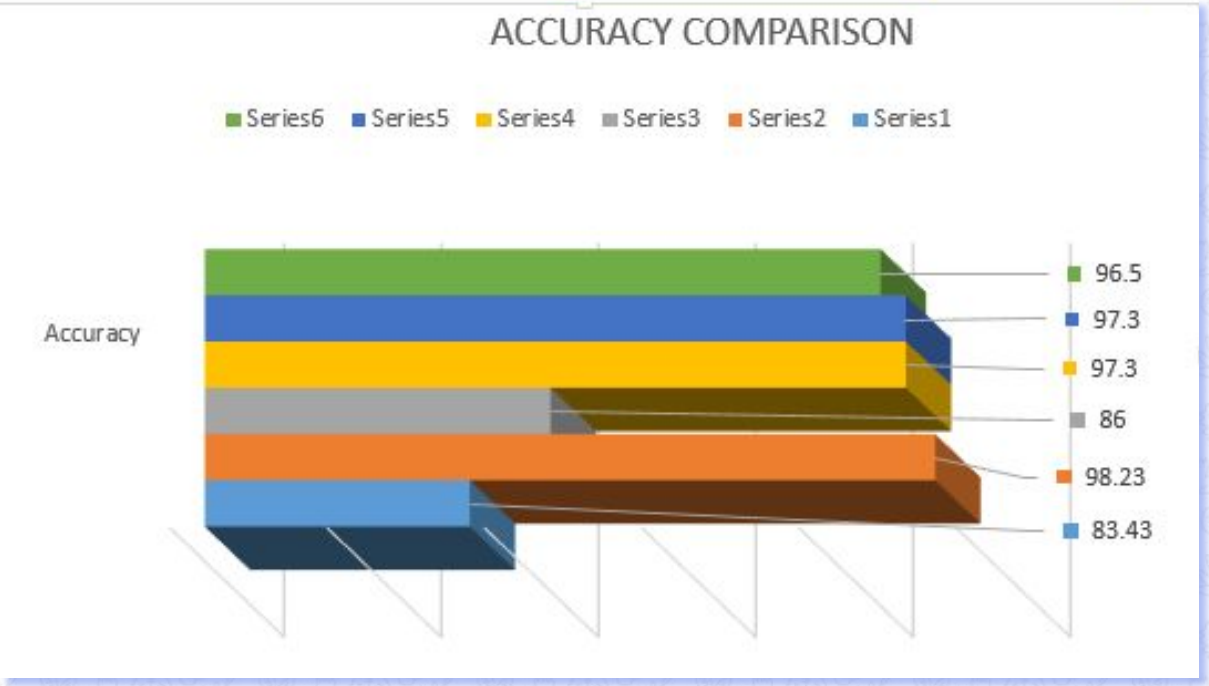


Bagging & Boosting



Conclusions: (based on accuracy of ML models)

Series No.	Model
1	SVC
2	NB
3	KNN
4	DT
5	Bagging
6	Boosting



THANK YOU!