**Assignment No.**

# Title: Naïve Bayes.

# Problem Statement: Identifying an Email is Spam or Ham classified using algorithm.

# Problem Definition:

Spamming is one of the major attacks that accumulate the large number of compromised machines by sending unwanted messages, viruses and phishing through emails.

Once you open this link then they will track you and try to hack your information.

* Unwanted email irritating Internet consumers and Identity Theft.
* Critical email messages are missed and/or delayed.
* Loss of Internet performance and bandwidth.
* Billions of dollars lost worldwide.
* Spam can crash mail servers and fill up hard drives.

# Prerequisite:

Basics of Python, Mining Algorithm, Concept of Naïve Bayes Algorithm.

# Software Requirements:

Anaconda with Python 3.7, Jupyter Notebook

# Hardware Requirement:

PIV, 2GB RAM, 500 GB HDD, Lenovo A13-4089Model.

# Objectives:

* To give knowledge to the user about the fake emails and relevent emails
* To classify that mail is spam or not.

# Outcomes:

After completion of this assignment any external email can be detected and classified as spam email so that user will be aware of such emails.

# Theory:

**Motivation:**

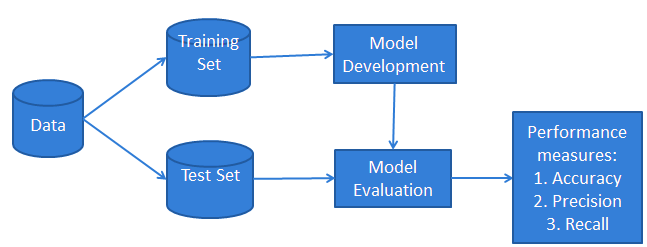
Text analysis of contents of mails is a widely used approach towards the spams. Naive Bayes is one of the most popular algorithms used in these approaches. But rejecting mails based on text analysis can be serious problem in case of false positives. The strategy is to accept all the mails except the ones from the domain/e-mail ids and classify spam and non spam emails. It will reduce Network Resource Costs, IT Administration Costs, Legal Liability Risk and increases Security and Control.

**What is Classification?**

It is a **Data analysis task**, i.e. the process of finding a model that describes and distinguishes data classes and concepts. Classification is the problem of identifying to which of a set of categories , a new observation belongs to, on the basis of a training set of data containing observations and whose categories membership is known.

# Classification Algorithms:

1. **Learning Step (Training Phase)**: Construction of Classification Model  
   Different Algorithms are used to build a classifier by making the model learn using the training set available. The model has to be trained for the prediction of accurate results.
2. **Classification Step**: Model used to predict class labels and testing the constructed model on test data and hence estimate the accuracy of the classification rules.

# What is Naïve Bayes?

It is a [**classification techniqu**e](https://courses.analyticsvidhya.com/courses/introduction-to-data-science-2/?utm_source=blog&utm_medium=6stepsnaivebayesarticle) based on Bayes’ Theorem with an assumption of independence among predictors.  It is one of the simplest supervised learning algorithms. Naive Bayes classifier is the fast, have high accuracy and speed on large datasets. Naïve Bayes has been studied extensively since the 1960s. Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

# Naïve Bayes – Text Classification Example :

Suppose we are building a classifier that says whether a text is about sports or not.

|  |  |
| --- | --- |
| TEXT | TAG |
| “A great game” | Sports |
| “The election was over” | Not sports |
| “Very clean match” | Sports |
| “A clean but forgettable game” | Sports |

# Now, which tag does the sentence *A very close game* belong to?

# Since Naive Bayes is a probabilistic classifier, we want to calculate the probability that the sentence “A very close game” is Sports and the probability that it’s *Not Sports*. Then, we take the largest probability as output. Another eg. That based on weather the children should play or not.

# Image result for Naïve Bayes Classification Steps

# Mathematical Formulation for Naïve Bayes Algorithm :

# The expression for Posterior Probability is as follows:

# Bayes Rule - Classification Algorithms - Edureka

* **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**

# Naïve Bayes Classification Steps :

Algorithm can be executed in the following steps:

**Step 1**: Calculate the prior probability for given class labels

**Step 2**: Find Likelihood probability with each attribute for each class

**Step 3**: Put these value in Bayes Formula and calculate posterior probability.

**Step 4**: See which class has a higher probability, given the input belongs to the higher probability class.

# Working of Algorithm:

# Eg : I have a training data set of weather namely, sunny, overcast and rainy, and corresponding binary variable ‘Play’. Now, we need to classify whether players will play or not based on weather condition.

**Step 1:** Convert the data set to the frequency table.

**Step 2:** Create a Likelihood table by finding the probabilities like

 Overcast probability = 0.29 and probability of playing is 0.64.

**Step 3:** Now, use the Naive Bayesian equation to calculate the posterior probability for each class.

P(Yes | Sunny) = P( Sunny | Yes) \* P(Yes) / P (Sunny)

P (Yes | Sunny) = 0.33 \* 0.64 / 0.36 = 0.60

P (No | Sunny) = 0.40 \* 0.36 / 0.36 = 0.40, which has higher probability is the outcome of prediction i.e. 0.60 > 0.40 so the children can play.

### The Bag of Words Model

# [Removing stopwords](https://en.wikipedia.org/wiki/Stop_words) : These are common words that don’t really add anything to the classification, such as a, able, either, else, ever and so on. So for our purposes, *The election was over* would be *election over* and *a very close game* would be *very close game.*

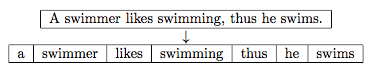
# https://sebastianraschka.com/images/blog/2014/naive_bayes_1/stop-1.png

# [Lemmatizing words](https://en.wikipedia.org/wiki/Lemmatisation) : This is grouping together different inflections of the same word. So election, elections, elected, and so on would be grouped together and counted as more appearances of the same word.

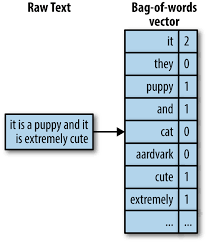
# https://sebastianraschka.com/images/blog/2014/naive_bayes_1/lemma-1.png

#### Tokenization

Splits a sentence into individual words, removes punctuation, and converts all letters to lowercase.



# [Using TF-IDF](https://en.wikipedia.org/wiki/Tf%E2%80%93idf) : Instead of just counting frequency we could do something more advanced like also penalizing words that appear frequently in most of the texts.

Let D1D1 and D2D2 be two documents in a training dataset:

* **D1D1: “Each state has its own laws.”**
* **D2D2: “Every country has its own culture.”**

Based on these two documents, the vocabulary could be written as \

V={each:1,state:1,has:2,its:2,own:2,laws:1,every:1,country:1,culture:1}

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | each | state | has | its | own | laws | every | country | culture |
| xD1xD1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |
| xD2xD2 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| ΣΣ | 1 | 1 | 2 | 2 | 2 | 1 | 1 | 1 | 1 |

# 

# There are three types of Naive Bayes model :

# [Gaussian:](http://scikit-learn.org/stable/modules/naive_bayes.html) It is used in classification and it assumes that features follow a normal distribution.

# [Multinomial](http://scikit-learn.org/stable/modules/naive_bayes.html): It is used for discrete counts.

# [Bernoulli](http://scikit-learn.org/stable/modules/naive_bayes.html): The binomial model is useful if your feature vectors are binary (i.e. zeros and ones).

# Algorithm:

# Handle Data : Load the dataset and split it into training and test datasets.

# Summarize Data : summarize the properties in the training dataset so that we can calculate probabilities and make predictions.

# Make Predictions : Generate predictions given a test dataset and a summarized training dataset.

# Evaluate Accuracy : Evaluate the accuracy of predictions made for a test dataset as the percentage correct out of all predictions made.

# Tie it together : Use all of the code elements to present a complete and standalone implementation of the Naive Bayes algorithm.

# Applications of Naïve Bayes:

* **Real time Prediction:**Naive Bayes is an eager learning classifier and it is sure fast. Thus, it could be used for making predictions in real time.
* **Multi class Prediction:**This algorithm is also well known for multi class prediction feature. Here we can predict the probability of multiple classes of target variable.
* **Text classification/ Spam Filtering/ Sentiment Analysis:** Naive Bayes classifiers mostly used in text. As a result, it is widely used in Spam filtering and Sentiment Analysis.
* **Recommendation System:**Naive Bayes Classifier and [Collaborative Filtering](https://en.wikipedia.org/wiki/Collaborative_filtering) together builds a Recommendation System that uses machine learning and data mining techniques to filter unseen information and predict whether a user would like a given resource or not.

**Why Naïve Bayes Is Used(Comparative Analysis)**

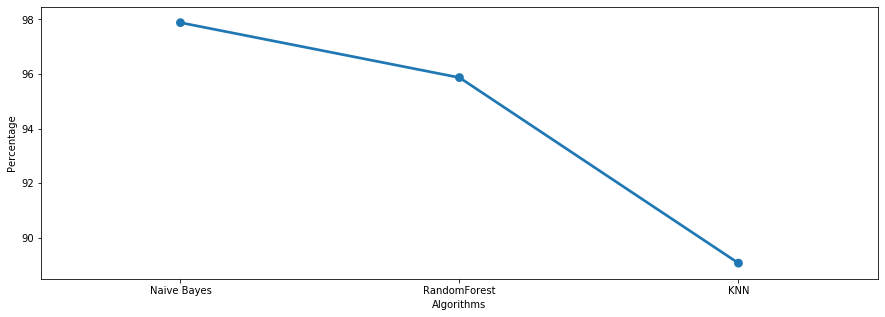
* **WITH RANDOM FOREST**

Naive Bayes performs well when we have multiple classes and working with text classification. Advantage is if the conditional independence assumption actually holds, a Naive Bayes classifier will converge quicker than discriminative models like logistic regression, so you need less training data and requires less model training time.

The main difference between Naive Bayes(NB) and Random Forest (RF) are their model size. Naive Bayes model size is low and quite constant with respect to the data. The NB models cannot represent complex behavior so it won’t get into over fitting. On the other hand, Random Forest model size is very large and if not carefully built, it results to over fitting. NB can adapt quickly to the changes and new data while using a RF you would have to rebuild the forest every time something changes.

* **WITH K-NEAREST NEIGHBOUR**

Naive Bayes is an eager learning classifier and it is much faster than K-NN. It takes a probabilistic estimation route and generates probabilities for each class. The best part with this classifier is that, it learns over time. In a spam filtering task, the type of spam words in email evolves over time. In the same way, the classifier also calculates probability estimates for the newly occuring spam words in a "bag of words" model and makes sure it performs well. This feature of the classifier is due to the inherent nature of the algorithm being generative but not discriminative.



# Conclusion:

In this way, we have studied and implemented the Naïve Bayes as Text classification is equally good and comparable with other method of classifications. Thus we are able to classify the emails as spam or non-spam.