**SPAM MESSAGE DETECTION USING MACHINE LEARNING**

A PROJECT REPORT

Submitted by

**Ruthvik.N**

**15MIS1096**

In partial fulfillment for the award of the degree of

Master of Technology

In

Software Engineering (5 Year Integrated Programme)









**“School of Computing Science and Engineering”**

Vellore Institute of Technology

Vandalur - Kelambakkam Road, Chennai - 600 127

November - 2020



School of Computing Science and Engineering

DECLARATION

I hereby declare that the project entitled Spam Message Detection Using Machine Learning submitted by me to the School of Computing Science and Engineering, VIT Chennai, 600 127, in partial fulfillment of the requirements of the award of the degree of Master of Technology in Software Engineering (5 year Integrated Programme) and as part of SWE4099 – Capstone Project is a bona-fide record of the work carried out by me under the supervision of Graceline Jasmine. I further declare that the work reported in this project, has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma of this institute or of any other institute or University

Signature of Candidate

Place: Chennai

Date:



School of Computer Science and Engineering

CERTIFICATE

This is to certify that the report entitled Spam Message Detection Using Machine Learning is prepared and submitted by RUTHVIK NAGABANDI(Reg. No. 15MIS1096) to VIT Chennai, in partial fulfillment of the requirement for the award of the degree of Master of Technology in Software Engineering (5 year Integrated Programme) and as part of SWE4099 – Capstone Project is a bona-fide record carried out under my guidance. The project fulfills the requirements as per the regulations of this University and in my opinion meets the necessary standards for submission.

Guide/Supervisor HoD

Name: Graceline Jasmine Name: Dr. Asnath Victy Phamila Y

Date: Date:

Examiner Examiner

Name: Name:

Date: Date:

(Seal of SCSE)

Spam Message Detection Using Machine Learning iii

Acknowledgement

“I obliged to give my appreciation to a number of people without whom I could not have completed this thesis successfully.”

“I would like to place on record my deep sense of gratitude and thanks to my internal guide Prof. Graceline Jasmine School of Computer Science and Engineering (SCSE), VIT University whose esteemed support and immense guidance encouraged me to complete the project successfully.”

“I would like to thank our HoD Dr. Asnath Victy Phamila Y, School of Computer Science and Engineering (SCSE) and Project Coordinator Dr. C. Sweetlin Hemalatha, VIT University, for their valuable support and encouragement to take up and complete this thesis.”

“Special mention to our dean Dr. Jagadeesh Kannan R , Associate Dean Dr. S. Geetha, School of Computing Science and Engineering (SCSE), VIT University, for motivating us in every aspect of software engineering...”

“I thank our management of VIT University, Chennai campus for permitting me to use the library and laboratory resources. I also thank all the faculty members for giving me the courage and the strength that I needed to complete my goal. This acknowledgment would be incomplete without expressing the whole hearted thanks to my family and friends who motivated me during the course of my work.”

RUTHVIK NAGABANDI

15MIS1096

School of Computing Science and Engineering, Vellore Institute of Technology, Chennai

Spam Message Detection Using Machine Learning **IV**

**Abstract**

“In general, spam emails are randomly sent by all kinds of groups to multiple addressees, but mostly lazy advertisers and criminals who want to steer you to phishing sites.”

“Machine learning is also a branch of AI involved in developing and researching systems that can learn from data. A machine learning program could be trained to distinguish between emails from spam and non-spam (ham). Thus, the goal is to evaluate machine learning in order to determine the best techniques to use in spam filtering based on content. Current spam techniques could be combined to improve productivity and investigate techniques for spam mail forecasting based on machine learning by predicting outcomes with the best accuracy.”

“Dataset analysis by supervised machine learning technique (SMLT) to capture multiple information, variable recognition, uni-variate analysis, bi-variate and multi-variate analysis, missing value treatments and data validation analysis, data cleaning / preparation and data visualization will be performed over the entire dataset. Our analysis provides a comprehensive guide for model parameter sensitivity analysis regarding performance in predicting spam mails by calculating accuracy.”

“In addition, to match and discuss the performance of different machine learning algorithms from the given dataset with evaluation classification report, sensitivity, specificity, identify the confusion matrix and categorize data from priority, the result shows that the efficiency of the proposed machine learning algorithm technique is often compared with the best accuracy With precision, Recall and F1 Score.”

Spam Message Detection Using Machine Learning V

Contents

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Declaration | |  | I |
|  | Certificate | |  | ii |
|  | Acknowledgement | | | iii |
|  | Abstract | |  | iv |
| 1 | Introduction | |  | 1 |
|  | 1.1 | Background | | 2 |
|  | 1.2 | objectives | | 3 |
|  | 1.3 | Statement | | 3 |
|  | 1.4 | Motivation | | 3 |
|  | 1.5 | Challenges | | 4 |
| 2 Planning & Requirements Specification | | | |  |
|  | 2.1 | System Planning | | 8 |
|  | 2.2 | Requirements | |  |
|  |  | 2.2.1 | User requirements | 9 |
|  |  | 2.2.2 | Non-Functional requirements | 9 |
|  | 2.3 | System Requirements | |  |
|  |  | 2.3.1 | Hardware Requirements | 10 |
|  |  | 2.3.2 | Software Requirements | 10 |
| 3 | System Design | | | 15 |
| 4 | Implementation of System | | | 25 |
| 5 | Results & Discussion | | | 37 |
| 6 | Conclusion and Future Work | | | 39 |
|  | References | | | 40 |
|  | Appendix | | | 41 |

Spam Message Detection Using Machine Learning viii

List of Tables

|  |  |  |
| --- | --- | --- |
| S.NO | Name | Page No |
| 1 | Data set | 8 |

School of Computing Science and Engineering, Vellore Institute of Technology, Chennai

ix

List of Figures

|  |  |
| --- | --- |
| Fig 1.1 Process of Machine learning | 1 |
| Fig 1.2 data flow diagram for Machine learning model | 8 |
| Fig 2.1 Process of dataflow diagram | 14 |
| Fig 3.1 Architectural Design | 15 |
| Fig 3.2 Workflow Diagram | 16 |
| Fig 3.3 Use Case Diagram | 17 |
| Fig 3.4 Class Diagram | 18 |
| Fig 3.5 Activity Diagram | 19 |
| Fig 3.6 Sequence Diagram | 20 |
| Fig 3.7Entity Relationship Diagram (ERD) | 21 |
| Fig 3.8 Collaboration Diagram | 22 |
| Fig 4.1 checking the duplicate values | 26 |
| Fig 4.2 Top 30 words of ham | 28 |
| Fig 4.3 Top 30 words of spam | 29 |
| Fig 4.4 Working model of proposed work | 30 |
| Fig 5.1 Open the anaconda navigator | 55 |
| Fig 5.2 Launch the jupyter notebook platform | 56 |

**1. INTRODUCTION**

“Machine learning from past data is to predict the longer-term. Machine learning (ML) is also a kind of AI (AI) that gives computers the facility to strive for without specific programming. Machine learning focuses on the event of computer programs that will alter when exposed to new data and hence the fundamentals of machine learning, implementing a simple python-based machine learning algorithm. The coaching and analysis process requires the use of professional algorithms. It feeds the training data to an algorithm, and the algorithm uses this training data to make predictions on a replacement test. Machine learning is also divided in approximately 3 groups. There is supervised learning, unsupervised learning and reinforcement learning   supervised learning software is given both the input file and then a person must first mark the corresponding filing to find out data. Learning without supervision is not marks. It provided the algorithm for the preparation. This algorithm holds the ability for input data clustering. Finally, reinforcement learning communicates dynamically with its environment, and receives positive or feedback to boost its performance.”

“Data scientists use several different kinds of machine learning algorithms to get patterns that trigger actionable insights in python. At a high level, these different algorithms are often divided into two categories that are assisted by the way they "learn" to shape predictions about data: supervised and unsupervised analysis. Classification is the method of predicting a given data point group. Often classes are called targets / labels, or divisions. Predictive modeling of the classification is that the job is to approximate a mapping function from input variables(X) to discrete output variables(y). Classification may also be a supervised learning technique in machine learning and statistics, through which the pc program learns from the data input provided to it and then uses this learning to identify new observation. This collection of data will simply be bi-class (such as defining whether the individual is male or female, or whether the mail is spam or non-spam) or it can also be multi-class. Some examples of classification problems include: recognition of speech, recognition of handwriting, identification of bio metrics, classification of documents etc.”

Analyses Predicts

Machine Learning

Past Dataset

Trains

Fig 1.1 Process of Machine learning

1. **1 Background**

“Supervised Machine Learning is the bulk of practical application of supervised learning in machine learning. Supervised learning is where input variables (X) and output variable ( y) are present, and using an algorithm to find the mapping function from input to output is y = f(X). The goal is to approximate the mapping function so well that you can simply predict the output variables (y) for that data until you have new input file (X). Supervised machine learning algorithm techniques include logistic regression, multi-class classification, decision trees and vector machine support etc. Supervised learning requires that the data is already labelled with correct answers to train the algorithm. Problems of supervised learning are also further divided into problems of classification. The objective of this problem is to construct a succinct model which can predict the value of the dependent attribute from the variables of the attribute. The distinction between the two tasks is that for categorical classification the dependent variable is numerical. A model of classification seeks to draw certain conclusions from the observed values. Given one or more inputs a model of classification may seek to predict the value of 1 or more outcomes. A classification problem is when the output variable, like "red" or "blue," may also be a category”

“Spam emails are messages sent randomly by all types of groups to multiple addresses”, but “mostly lazy advertisers and criminals who want to steer you to phishing sites. Spam detection is one of today's main Machine Learning applications within the interwebs. Machine learning can be a branch of AI involved in developing and researching systems that can learn from data. A machine learning program could be trained to distinguish between emails from spam and non-spam (ham). Using various methods to evaluate the relationship between the email, and thus the SPAM or HAM category including, message size endorsed, word count, special keywords. Then use different techniques to construct classification models to distinguish spam sms. Compare accuracy of each technique and plot precision graphs in a single bar diagram. Generate a word-cloud for spam SMS as well. Before implementing any supervised learning methods, we used a bunch of data cleaning operations to get rid of messy and dirty data as it has damaged and messy history. We will use the Naive Bayes algorithm in this mission to construct a model that will identify SMS data set messages as spam or not spam, supporting the training we provide to the model. At the top of each processing by different classifier, we have plotted confusion matrix to match which one of the simplest classifiers for filtering SPAM SMS. We can classify emails with a high number of emails as spam or non-spam, and it will be difficult to handle all possible mails. Training during training, data from a training data set are given to the machine learning system labeled. The classified training data in our project are a broad collection of emails which are identified as spam or ham.”

#### “The classifier (the part of the machine learning system that actually predicts labels of future emails) learns from the training data through determining the connections between an email's features and its label during the training process. The machine learning program is given unlabeled data during the testing (Classification). In our case, emails without the spam / ham label are those data. The classifier determines whether the e-mail is spam or ham, based on the features of an email. This distinction is related to live results with the spam / ham truth factor.”

#### 1.2 Objectives:

“The aim is to build a machine learning model for real-time spam message prediction, to potentially replace the updatable supervised machine learning classification models by predicting results in the form of best accuracy by comparing supervised algorithms.”

“The goal of recognizing spam e-mails is to:”

* To provide the user with awareness of fraudulent emails and valid emails
* To mark mail as or not spam

**1.3 Problem Description/ Problem Statements:**

“Monitoring and preserving spam message has become one of the most essential activities in many ISP‘s today.”

* Unwanted emails of Internet contact annoying
* Missing and losing important email messages
* Spam can crash main servers and fill up hard derives

**1.4 Motivation**

“Smishing represents a serious risk to security, particularly for financial institutions. JPMorgan Chase had a data breach in 2014, where hackers revealed the phone numbers and other information of more than 83 million accounts. Sensitive information, such as account holders' social security numbers, was not accessed, but the disclosure of these users' phone numbers was assumed to be the motive for a series of subsequent smiling attacks.”

“This project presents the specifics of the roving proxy system for the detection of SMS spam and SMS phishing. The framework aims at protecting organizations and businesses from the risk of SMishing attacks. System feasibility and design studies are provided along with an upgrade method analysis to identify the system's minimum specifications for adapting to the current trends in spam and SMishing. Index terms — SMS spam, SMishin, protection at BYOD. We have discussed this scenario in particular in this research report, and to the best of our knowledge, no prior work has done so except to analyze and block spam SMSs in general. In order to avoid these attacks, we proposed to deploy institutions the roving proxy server system, in order to shield their members from spam SMS in general and from smiling in particular.”

**1.5 Challenges**

“The most challenging part in spam detection is training our results. First change is data preparation here in the phase of data preparation, we will explore the data to check for possible problems. To make it suitable for our machine learning algorithm, we'll pre-process the data. We have email text data in this problem and a corresponding label that categorizes whether the email is spam, we need to turn it into vectors of numbers before we can feed it into our training model. Your machine learning model is just as strong as the data from which you train it. Next daunting activity is Data Exploration We must examine the data in the data exploration stage to discover and address any possible data problems. The actual implementation here would be different depending on the problem you are trying to solve with machine learning, and the type of training data you have (e.g. images, audio, text). Then we will divide our samples into 2 subsets, a training subset of 90 percent and a test subset of 10 percent. We are going to use the subset of 90 per cent to train our model. We will use the subset of 10 per cent to test the output of the trained model on unseen results. We are going to pre-process our data in a format that can be fed into our computer training. We'll use the test data to assess the accuracy of the trained model. And then to forecast, we use our trained model.”

**Overview of the system**

“The Classifiers' Prediction Accuracy was tested using data validation, and the results were compared for accuracy. This has to find Learning Dataset Accuracy, Testing Dataset Accuracy, Specification, False Positive Frequency, Precision and Recall by comparing Python Code algorithms. The next steps to include are”

* Define a problem
* Preparing data
* Evaluating algorithms
* Improving results
* Predicting results

Below are the measures involved in Creating the Data Model

**Data collection** (Splitting Training set & Test) set)

**Pre Processing** (Label Encoder ())

**Building classification Model**

**Prediction (Message** Prediction**)**

Fig 1.2 data flow diagram for Machine learning model

**Advantages:**

* It improves accuracy score by comparing popular machine learning algorithms.
* These reports are to the investigation of applicability of machine learning techniques for diabetes stages forecasting in operational conditions by attribute prediction.
* Finally, it highlights some observations on future research issues, challenges, and needs.

**2. Planning & Requirements Specification**

**2.1 System Planning**

“Planning is the most important step for creating a successful system and during this phase we come to know what exactly we want to do.”

**Objectives**

The main objective of proposed model is:

* To provide the user with awareness of fraudulent emails and valid emails
* To classify that there is no mail spam
* And also to reduce the user time, space, money

**Resources**

* Anaconda platform
* Required packages install (NLTK, Matplotlib, Word cloud, Math, Pandas, Numpy)
* Implementation of proposed algorithm model
* Training the model to get the expected results

**Table 1: Dataset**

|  |  |
| --- | --- |
| Data | 5500 spam and ham email messages |
| Attributes | Label, messages |

**Issues**

* Issues in model processing, but it can be solved by understanding and studying the process completely.
* Maintaining the datasets.
* Installation of software
* Installation of the required packages

**2.2 Requirements**

“Requirements are the basic constrains that are required to develop a system. Requirements are collected while designing the system. The following are the requirements that are to be discussed.”

1. Functional requirements

2. Non-Functional requirements

3. System requirements

A. Hardware requirements

B. software requirements

**2.2.1 Functional requirements:**

“The specification of the software specifications is a technical specification of the software product specifications. It is the first step in the process of requisite analysis. It lists specifications for a particular software program. The following details to follow the special libraries like SK-learn, pandas, numpy, matplotlib and seaborn.”

Process of functional steps,

1. Problem define
2. Preparing data
3. Evaluating algorithms
4. Improving results
5. Prediction the result

**2.2.2 Non-Functional Requirements:**

* Makes email data easily accessible in datasets.
* User will arrive at the test as soon as possible.
* It should be easy to use, i.e. the user just needs to type the words and then click on the result or the user only needs to enter a pair of appropriate sentences.

**2.3 System Requirements**

**2.3.1 Software Requirements**

Operating System : Windows 10

Tool : Anaconda with Jupyter Notebook

**2.3.2 Hardware requirements**

Processor : Pentium IV/III

Hard disk : minimum 80 GB

RAM : minimum 2 GB

**2.3.3 Technologies Used**

**Frontend:** GUI using python (Tkinter)

**Backend:** Data analysis using python (Pandas, Numpy etc)

**Building a Predictive Model**

“Machine learning requires collecting data have a lot of data from past. There are ample historical records and records to collect data. Data cannot be used directly before pre-processing of the data. So it's preprocess customary, what very algorithm with pattern. Training and checking this model works and with minimal errors predicting correctly. Tuned model includes increasing the accuracy by tuning time to time.”

Data Gathering

Data Pre-Processing

Choose model

Train model

Test model

Tune model

Prediction

Fig 2.1 Process of dataflow diagram

**3. System Design**

“Design is significant representation of engineering of something that is to be built. Software design may be a process design is that thanks to correct conversion of specifications into a finished product, the great. Design produces a representation or model, offers information about the structure of the program, the architecture, the interfaces and the components required to implement a system.”

**Architecture**

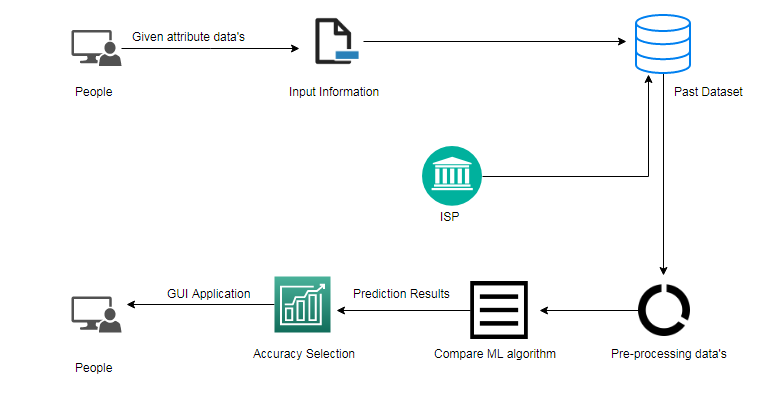


Fig 3.1 Architectural Design

**3.2 Work flow diagram**

Source Data

Data Processing and Cleaning

Testing Dataset

Training Dataset

Find and compare ML model

Supervised ML algorithm

Prediction of message is spam or not

Fig 3.2 Workflow Diagram

**3.3 Use Case Diagram**

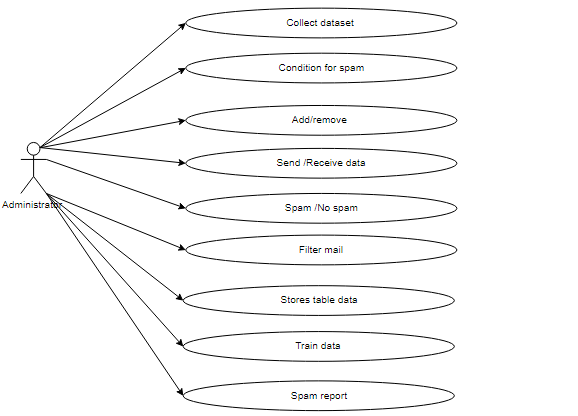


Fig 3.3 Use Case Diagram

“Use case diagrams are considered for an overview of a system's top level specifications. So when evaluating a system's requirements the functionalities are identified in use cases. So, case uses can be said to be nothing but machine functionalities written in an ordered way. Then the actors (Patient / Doctor) are the second items that are important for job cases.”

**3.4 Class Diagram**

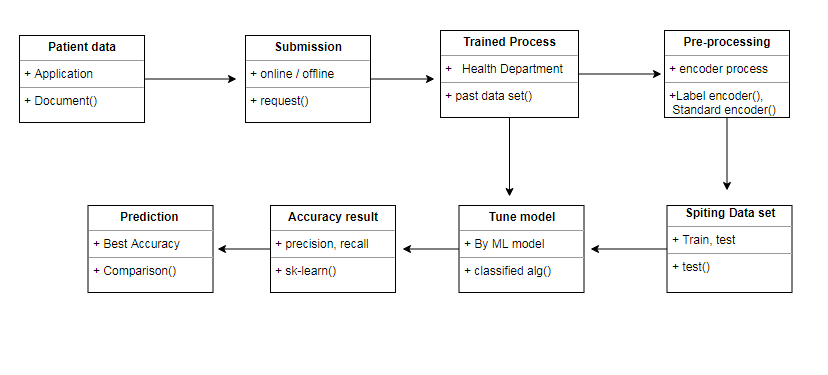


Fig 3.4 Class Diagram

“Class diagram is essentially a graphical representation of the system's static view, and represents various aspects of the appliance. So the entire structure is represented by a series of complexity diagrams. For illustrate the feature of the method, the name of the category diagram should be important. That entity and its relationships should be defined earlier. Responsibility (attributes and methods) of each class should be clearly established for each class minimum number of properties, and because unnecessary properties complicate the diagram. Using notes whenever appropriate to illustrate any part of the diagram and it should be comprehensible to the developer / coder at the drawing level. Finally, the diagram should be drawn on plain paper before creating the ultimate edition, and reworked as much as possible to shape it correctly.”

**3.5 Activity Diagram**

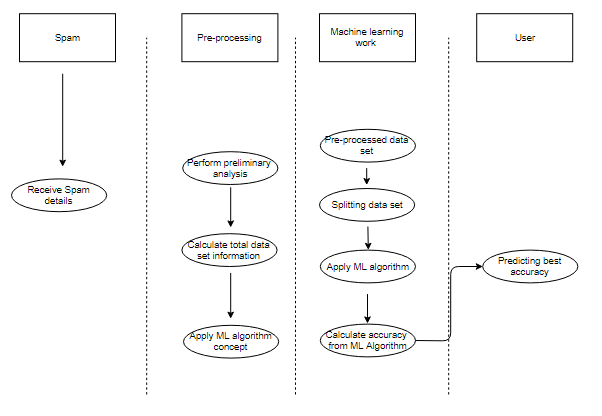


Fig 3.5 Activity Diagram

“Activity is a particular operation of the device. Task diagrams are not only used to illustrate a system's complex design, but they also use forward and reverse engineering techniques to build the executable framework. The only thing missing in the operation diagram is that it part of the post. It shows no message to be due to one activity to another. Action diagram is found a few times because of flow diagram. Although if the diagrams look like a flow map, it is not. It displays numerous flows, such as parallel, branched, concurrent and single.”

**3.6 Sequence Diagram**

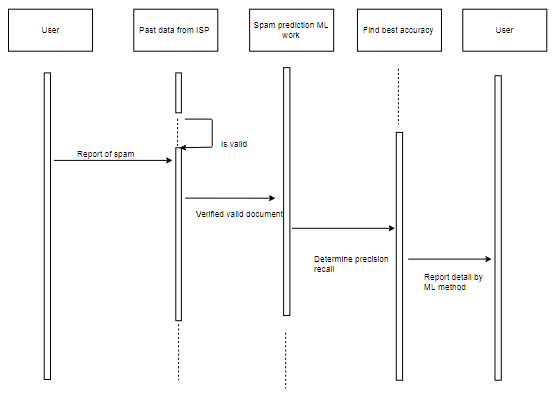
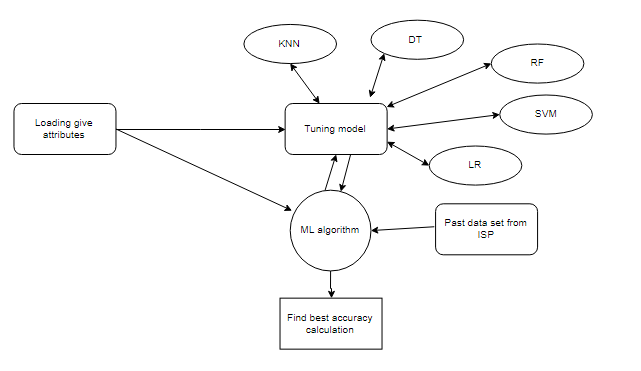


Fig 3.6 Sequence Diagram

“Even action may be a particular function of the machine. Event diagrams do not appear to be only used to represent a system's fluid design, but they do use forward and reverse engineering techniques to construct the executable system. The only thing lacking in the operation diagram is that it is a part of the post. This reveals no message that comes with one operation to a different one. Activity diagram may well be considered for some time because of the flow chart. Although if the diagrams look like a flow map, it is not. It displays numerous flows, such as parallel, branched, concurrent and single.”

**3.7 Entity Relationship Diagram (ERD)**



**Fig 3.7** Entity Relationship Diagram (ERD)

“An entity relationship diagram (ERD), also referred to as an entity relationship model, and may be a graphical representation of a system representing the relationships within that system between individuals, objects, locations, concepts, or events. An ERD may be a technique in data processing that will help describe business processes and be used as the basis for an electronic database. Person relationship diagrams provide a clear starting point for the design of a database that can also help define system specifications within an organization. An ERD can still act a reference point after an electronic database is unrolled, should any debugging or business process re-engineering be needed later.”

**3.8 Collaboration Diagram**

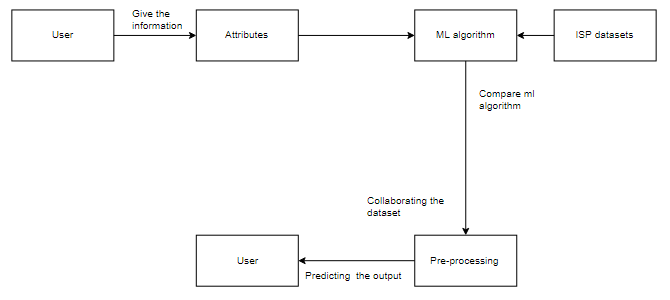


Fig 3.8 Collaboration Diagram

“A collaborative diagram may be a variety of visual presentation showing how different software objects interact with each other within an overall IT architecture, and how users (such as doctors or patients) can enjoy this collaboration. A collaborative diagram often falls within the type of a visible diagram resembling a flow chart. At a glance it can demonstrate how one piece of software complements other pieces of a larger program.”

**Model Selection:**

“This is the most exciting introduce to every Dataset by applying Machine Learning. This is also called selection of Algorithms to predict the most efficient results. Data scientists usually use various types of Machine Learning algorithms for the big data sets. But all those different algorithms are often classified into two groups at the high level: supervised learning and unsupervised learning. Supervised learning: Supervised learning would be a device selection in which input and desired output data would be generated. Classification of input and output data is labelled to provide a learning framework for future processing.”

“Problems of supervised learning are also grouped further into problems of regression and classification. A regression question is when the output variable, like "salary" or "weight," will be a real or continuous value A classification issue is whether the output variable will be a "spam" or "not spam" unsupervised learning category such as filtering emails: unsupervised learning is when the algorithm uses information that is neither classified nor labelled and enables the algorithm to operate on information without guidance. In our dataset we have the vector or quantity of tip results i.e. Y only have two sets of values, either M (Malign) or B (Benign). And we visit to use supervised learning classification algorithm.”

**Spam classifier methods:**

“For spam and non-spam messages, we previously revealed the foremost occurring / common words, bigrams, and trigrams from the messages separately. Now we would also like to see some essential terms that will determine whether a message can or may not be a spam. Take a note here that almost all of the occurring / common word in a very set of messages may not be a keyword determining what the whole sentence is all about. Terms like business, investment, acquisition, for example are essential terms that will connect a sentence to a business article. Other terms like money, nice, building etc. are also the common words in the messages, but they don't have a lot of relevant information to generate.”

**TF-IDF: (Term Frequency-Inverse Document Frequency)**

“We will use the Term Frequency-Inverse Document Frequency (TF-IDF) strategy to find the important terms, the tf-idf weight may be a weight often used in information retrieval and text mining. TF means Frequency of Words. It tests how much a word appears in a document itself. Because each document is different in length, it is likely that in long documents a word will appear far more frequently than in shorter ones. Thus, the word frequency is generally divided by the duration of the document as standardization process.”

**“TF = (The number of times w in a document)/ (the total number of terms in a document)”**

“Second part idf stands for frequency of Inverse Documents. It tests whether a word is relevant. All words are equally relevant when computing TF. It is understood, however, that certain words, such as "is," "of," and "that," that occur frequently but have little meaning. Therefore we want to overwhelm the common terms and scale up the uncommon ones.”

**“IDF = log e (Total number of documents / Record number with term w inside)”**

**“We measure a final tf-idf score by multiplying TF score with IDF score for each word then finally, by selecting words with the next Tf-Idf score, we'll separate important terms.”**

### Finding important words using Tf-IDF:

“Now we'll have to figure out which are the most important words in both spam and non-spam messages and then we'll look at those words in word cloud form. We’re going to interpret those terms and that's going to help us explain why a specific message has been classified as spam, and another as non-spam.”

### Bag of Words:

“For the machine learning algorithm we need to represent text data and hence the bag-of - words model helps us realize the mission. The layout of the bag-of - words is simple to learn and to enforce. It is how to extract features from the text that should be used in algorithms for machine learning. A bag-of-words can be a text representation representing the occurrence of words within a document. It has two things to it:”

* A vocabulary of words known.
* An indicator of known verbs' presence.
* It is possible to achieve a vocabulary by tokenizing messages into different unique tokens. We’d like to attain the token after having every token. They will do so in the following ways
* Count the number of times each word appears in the course of a text.
* There are frequencies. Calculate the frequency of each word appearing from all the words inside the document during a document.
* IDF score \* TF-IDF score

4. Implementation of System

**4.1 Modules**

* Data validation and pre-processing technique (Module-01)
* Exploration data analysis of visualization and training a model by given attributes (Module-02)
* Performance measurements of logistic regression and decision tree algorithms (Module-03)
* Performance measurements of Support vector classifier and Random forest (Module-04)
* Performance measurements of KNN and Naive Bayes (Module-05)
* GUI based prediction of spam or not using spam classifier (Module-06)

**4.2 MODULE EXPLANATION:**

**Variable Identification Process / data validation process (Module-01):**

“Validation techniques in machine learning are accustomed to get the Machine Learning (ML) model error rate, which can be known as the dataset’s near-truth error rate. If the amount of information is high enough to represent the population, then the validation techniques may not be required. In real-world scenarios, however, work with data samples that cannot be a true representative of the given dataset population. To find the missing value, duplicate the value and the data type description whether it is a float variable or an integer. The accustomed data sample provides an unbiased evaluation of a model fit on the training dataset while tuning hyper parameters for the model.”

“The research is a lot of biased when the talent is inserted into the model setup on the validation dataset. The validation set is used to measure a given model, however this can often be used for frequent evaluation. It uses that expertise as machine learning engineers to fine-tune the hyper parameters of the model. A time-consuming to-do list will add up to data collection, data processing, and therefore the method of addressing data content, quality, and structure. It helps to understand your data and its properties during the data identification tactics; this information will encourage your choice of which algorithm to use to construct your model. Statistical data, for example, are often analyzed by regression algorithms; classification algorithms are also used to analyzing discrete data.”

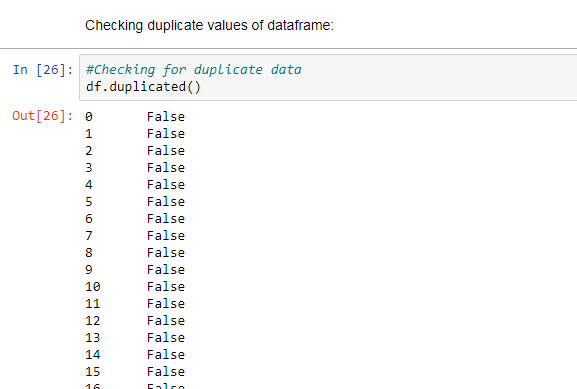


Fig 4.1 checking the duplicate values

**Data Validation/ Cleaning/Preparing Process:**

“Importing packages in the library with loading of the provided dataset. Duplicate values to evaluate the variable recognition by data form, data type and determine the missing values. A validation dataset could be a sample of data held back from training your model that is used to giving an estimate of model skill while tuning models and procedures that you can only use to create the most effective use of validation and test datasets when evaluating your models. Cleaning / preparing data by renaming the given dataset and dropping the column etc. to investigate the uni-variate, bi-variate and multi-variate method. The info improvement steps and techniques may vary from dataset to dataset.”

“The primary purpose of data cleaning is to detect and remove errors and anomalies in order to increase data prices in analytics and better processes.”

**Data Pre-processing:**

“Pre-processing refers to the transformations applied to our data before the algorithm loads it. Data Preprocessing can be a method of converting the data into a clean collection of data. In other words, once the information is obtained from different sources, it is obtained in a raw format that is not feasible for analysis. In order to achieve better results from the model applied in the Machine Learning method the information has to be in a proper manner. Some specified Machine Learning model needs information during a specified format; for example, null values are not supported by the Random Forest Algorithm. Thus null values should be managed from the initial data set to execute random forest algorithms. And another thing is that data set should be structured in such a way as to execute in a given dataset both one Machine Learning and Deep Learning algorithm.”

**Exploration data analysis of visualization (Module-02):**

“Data visualization is an essential competency in applied statistics and machine learning. Statistics is primarily based on quantitative explanations and information estimates. Visualization of data provides a vital collection of resources for a comprehensive understanding. This will help when exploring and knowing about the dataset, and may help with pattern identification, corrupt data, outliers, and much more. Data visualizations are also specifically designed with a touch domain awareness and show key relationships in plots and charts that are more vivid and stakeholder than affiliation or significance steps. Visualization of data and analysis of exploratory data are entire fields themselves and it would suggest a deeper dive into some of the books listed at the top. Data Visualization the anticipated results are visualized in graphical or tabular format after the classification and regression process for a better understanding of the users. This process is called Visualization of Data. We can get the description of the ends in numeric format as well.”

“Sometimes data doesn't add up until, like with charts and plots, it can look into a very visual form. Having the ability to quickly visualize samples of knowledge et al is a crucial skill in applied statistics as well as machine-learning. It will discover the many varieties of plots you actually need to learn when visualizing Python data and how to use them to understand your own data better. How to diagram time series data with line plots and categorical bar charts quantities.”

* How to use histograms and box plots to summarize data distributions.
* How to summarize the relationship with scatter plots between variables.

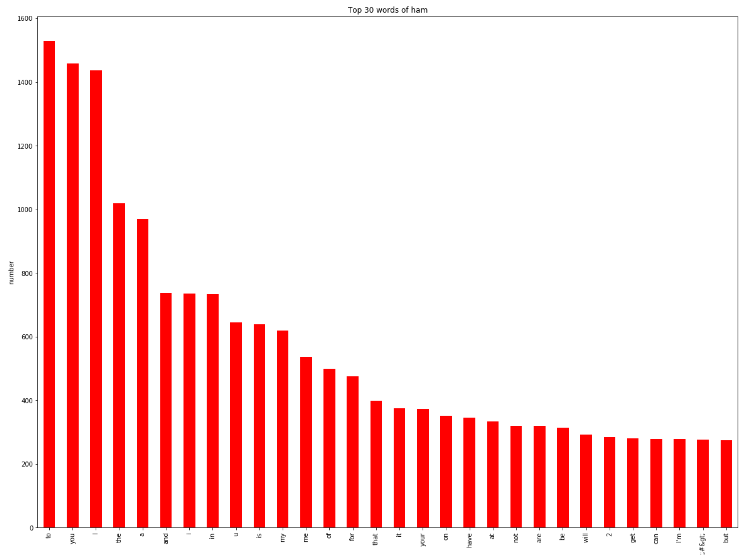


Fig 4.2 Top 30 words of ham

“Many algorithms of machine learning are sensitive to the range and distribution of attribute values within the input file. Input file outliers can skew and mislead the machine learning algorithms training process leading to longer training times, less accurate models and, ultimately, poorer outcomes.”

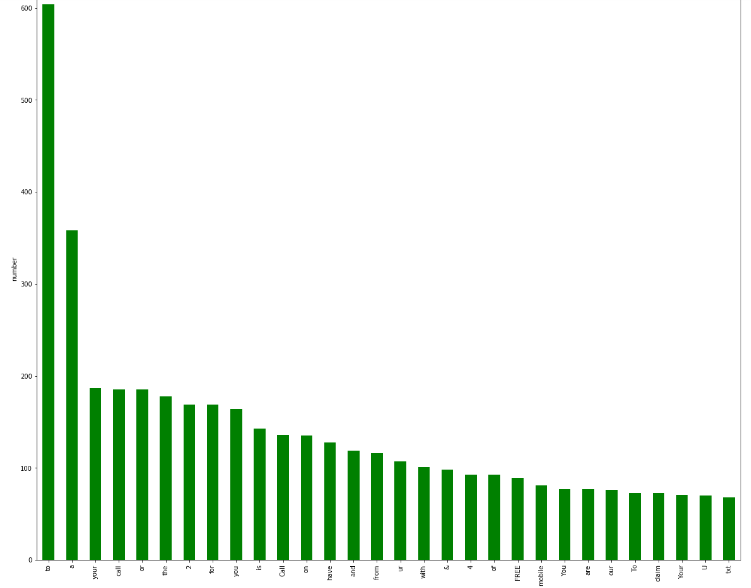
“Even before predictive models on training data are prepared, outliers can end up in misleading representations and suddenly misleading interpretations of collected data. In descriptive statistics such as mean and variance, and in plots such as histograms and scatterplots, outliers can skew the summary distribution of attribute values, compressing the body of information. Finally, in the case of fraud detection and computer security, outliers can represent samples of data instances relevant to the matter, such as anomalies.”

Fig 4.3 Top 30 words of spam

“It couldn't fit the model on the training data, and can't say the model would work on the important data accurately. To do this, we need to make sure our model has the right patterns from the details, and it doesn't get too much noise. Cross-validation may be a technique whereby we can train our model using the dataset sub-set and then evaluate it using the dataset’s complementary subset.”

The three intervalidation steps are as follows:

1. Book a portion of the sample dataset.
2. Train the model using the residual datasets.
3. Using the reserve portion of the data set to check the pattern.

**Advantages of train/test split:**

1. This runs K times faster than cross-validation with Leaving One Out since the K-fold cross-validation repeats the K-times split train / test.
2. Simpler to look at the detailed test results.

**Advantages of cross-validation:**

1. More accurate estimate of out-of-sample accuracy.
2. More “efficient” use of information as every observation is employed for both training and testing.

**Training the Dataset:**

1. The first line imports iris data set which is already predefined in sklearn module and raw dataset is basically a table which contains information about various varieties.

1. For example, to import any algorithm and train\_test\_split class from sklearn and numpy module for use in this program.
2. To encapsulate load\_data () method in data\_dataset variable. Further divide the dataset into training data and testing data using train\_test\_split method. The X prefix in variable signify the feature values and y prefix signify target values.
3. This method divides dataset into training and test data randomly in ratio of 67:33 / 70:30. Then we encapsulate any algorithm.
4. In the next line, we fit our training data into this algorithm so that computer can get trained using this data. Now the training part is complete.

**Testing the Dataset:**

* “Now, the dimensions of new features in a numpy array called ‘n’ and it want to predict the species of this features and to do using the predict methodology that takes this array as input and spits out predicted target value as output.”
* “So, the expected target worth comes intent on be zero. Finally to search out the take a look at score that is that the magnitude relation of no. of predictions found correct and total predictions created and finding accuracy score technique that primarily compares the particular values of the take a look at set with the expected values.”

Source data

Test dataset

Data Processing

Model

Supervised ML Algorithm

Training dataset

**Fig 4.4 Working model of proposed work**

[**Logistic Regression**](https://en.wikipedia.org/wiki/Logistic_regression) **(Module-03):**

“It is a statistical procedure for analyzing a knowledge set during which there are one or more independent variables that determine an outcome. The outcome is measured with a divided variable (in that there are solely 2 potential outcomes).The goal of logistic regression is to seek out the simplest fitting model to explain the connection between the dichotomous characteristic of interest (dependent variable = response or outcome variable) and a group of independent (predictor or explanatory) variables. Logistic regression may be a Machine Learning classification algorithm that's wont to predict the probability of a categorical variable. In logistic regression, the variable may be a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.).”

“In other words, the logistic regression model predicts P(Y=1) as a function of X. Logistic regression Assumptions:”

* Binary logistic regression requires the dependent variable to be binary.
* For a binary regression, the factor level 1 of the dependent variable should represent the desired outcome.
* Only the meaningful variables should be included.
* The independent variables should be independent of each other. That is, the model should have little.
* The independent variables are linearly related to the log odds.
* Logistic regression requires quite large sample sizes.

[**Decision Tree**](https://www.geeksforgeeks.org/decision-tree/)**:**

“It is one amongst the foremost powerful and popular algorithm. Decision-tree rule falls beneath the class of supervised learning algorithms. It works for both continuous additionally as categorical output variables. Assumptions of Decision tree: At the beginning, we consider the whole training set as the root.”

* Attributes are assumed to be categorical for information gain, attributes are assumed to be continuous.
* On the idea of attribute values records are distributed recursively.
* We use statistical strategies for ordering attributes as root or internal node.

“Decision tree builds classification or regression models within the style of a tree structure. It breaks down a dataset into smaller and smaller subsets while at the identical time an associated decision tree is incrementally developed. A choice node has two or more branches and a leaf node represents a classification or decision. The topmost decision node in an exceedingly tree which corresponds to the simplest predictor called root node. Decision trees will handle categorical and numerical information. Decision tree builds classification or regression models within the kind of a tree structure. It utilizes an if-then rule set which is mutually exclusive and exhaustive for classification. The foundations are learned sequentially using the training data one at a time. Anytime a rule is learned, the tuples covered by the foundations are removed.”

“This process is sustained on the training set until meeting a termination condition. It’s constructed in an exceedingly top-down recursive divide-and-conquer manner. All the attributes should be categorical. Otherwise, they ought to be discretized ahead. Attributes within the top of the tree have more impact towards within the classification and that they are identified using the data gain concept. A choice tree are often easily fitted by generating too many branches and should reflect anomalies thanks to noise or outliers.”

**Support Vector Machines (SVM) (Module-04):**

“A classifier that categorizes knowledge the info the information} set by setting Associate in Nursing best hyper plane between data. I selected this classifier because it is implausibly versatile within the range of various kernelling functions will which will that may} be applied and this model can yield a high sure thing rate. Support Vector Machines area unit maybe one among the foremost widespread and talked regarding machine learning algorithms. They were very widespread round the time they were developed within the Nineteen Nineties and still be the go-to methodology for a high-performing algorithmic program with very little calibration. How to disentangle the many names used to refer to support vector machines.”

* The representation employed by SVM when the model is truly stored on disk.
* How a learned SVM model representation are often used to make predictions for new data.
* How to find out an SVM model from training data.

**Random Forest:**

“Random forests or random decision forests are an entity learning method for classification, regression and other tasks, that operate by constructing an outsized number of decision trees at training time and outputting the category that is the mode of the classes (classification) or mean prediction (regression) of the individual trees. Random call forests correct for call trees’ habit of over fitting to their coaching set. Random forest may well be a sort of supervised machine learning algorithm supported ensemble learning. Ensemble learning may well be a sort of learning where you join different types of algorithms or same algorithm multiple times to form a more powerful prediction model. The random forest rule combines multiple rule of the same sort i.e. multiple decision trees, resulting in a forest of trees, hence the name "Random Forest". The random forest algorithm are going to be used for both regression and classification tasks.”

“The following are the essential steps involved in performing the random forest algorithm: Pick N random records from the dataset.”

* Build a decision tree based on these N records.
* Choose the number of trees you want in your algorithm and repeat steps 1 and 2.
* “In case of a regression problem, for a replacement record, each tree within the forest predicts a worth for Y (output). The ultimate value are often calculated by taking the typical of all the values predicted by all the trees in forest. Or, just in case of a classification problem, each tree within the forest predicts the category to which the new record belongs. Finally, the new record is assigned to the category that wins the bulk vote.”

**K-Nearest Neighbor (KNN) (Module-05):**

“K-Nearest Neighbor is a supervised machine learning algorithm which stores all instances corresponded to training data points in n-dimensional space. When an unknown discrete data is received, it analyzes the closest k number of instances saved (nearest neighbors) and returns the foremost common class because the prediction and for real-value data it returns the mean of k nearest neighbors. Within the distance-weighted nearest neighbor algorithm, it weights the contribution of every of the k neighbors in step with their distance using the subsequent query giving greater weight to the closest neighbors.”

“Usually KNN is powerful too noisy data since it's averaging the k-nearest neighbors. The k-nearest-neighbors algorithm may be a classification algorithm, and it's supervised: it takes a bunch of labeled points and uses them to be told the way to label other points. To label a brand new point, it's at the labeled points closest to it new point (those are its nearest neighbors), and has those neighbors vote, so whichever label the foremost of the neighbors have is that the label for the new point (the “k” is that the number of neighbors it checks). Makes predictions about the validation set using the complete training set. KNN makes a prediction a few new instance by looking through the complete set to seek out the k “closest” instances. Closeness is decided by employing a proximity measurement (Euclidean) across all features.”

**Naive Bayes algorithm:**

“The Naive Bayes algorithm is an intuitive method that uses the chances of every attribute belonging to every class to create a prediction. It’s the supervised learning approach you'd come up with if you wanted to model a predictive modeling problem probabilistically. Naive Bayes simplifies the calculation of all the probabilities by assumption that the probability of each attribute belonging to a given class value is independent of all other attributes. This is often a powerful assumption but leads to a quick and effective method. The probability of a category value given a price of an attribute is termed the probability. By multiplying the conditional probabilities together for every attribute for a given class value, we've got a probability of a knowledge instance belonging to it class. To create a prediction we will calculate probabilities of the instance belonging to every class and choose the category value with the best probability.”

“Naive Bayes could be a statistical classification technique supported Bayes Theorem. It’s one among the best supervised learning algorithms. Naive Bayes classifier is that the fast, accurate and reliable algorithm. Naive Bayes classifiers are high accuracy and speed on large datasets. Naive Bayes classifier assumes that the effect of a specific feature in a very class is independent of other features. As an example, a loan applicant is desirable or not betting on his/her income, previous loan and transaction history, age, and placement. Whether or not these features are interdependent, these features are still considered independently. This assumption simplifies computation, and that is why it's considered as naive. This assumption is termed class conditional independence.”

**Module-06:**

“Tkinter is a python binding library for developing GUI (Graphical User Interfaces). We use the tkinter library for creating an application of UI (User Interface), to create windows and all other graphical user interface and Tkinter will come with Python as a standard package, it can be used for security purpose of each users or accountants. There will be two kinds of pages like registration user purpose and login entry purpose of users.”

**Accuracy calculation:**

**“False Positives (FP):** an individual who can pay predicted as defaulter. When actual class is not any and predicted class is yes. E.g. if actual class says this passenger didn't survive but predicted class tells you that this passenger will survive.”

“**False Negatives (FN):** an individual who default predicted as payer. When actual class is yes but predicted class in no. E.g. if actual class value indicates that this passenger survived and predicted class tells you that passenger will die.”

**“True Positives (TP):** an individual who won't pay predicted as defaulter. These are the correctly predicted positive values which suggests that the worth of actual class is yes and therefore the value of predicted class is additionally yes. E.g. if actual class value indicates that this passenger survived and predicted class tells you an equivalent thing.”

**“True Negatives (TN):** an individual who default predicted as payer. These are the correctly predicted negative values which suggests that the worth of actual class is not any and value of predicted class is additionally no. E.g. if actual class says this passenger didn't survive and predicted class tells you an equivalent thing.”

“It achieved precision, recall, true positive rate (TPR), and false positive rate (FPR) for every classification techniques because it is shown within the above tables and also achieved different interesting confusion matrix for every classification techniques and that we can see the classification performance of every classifiers by the assistance of confusion matrix. We use a confusion matrix to compute the accuracy rate of every severity class. For every class, it demonstrates how instances from that class receive the varied classifications. Here within the next table we've shown instances that are correctly classified and incorrectly classified in accordance with overall accuracy of every classification techniques. All classifiers perform similarly well with reference to the amount of correctly classified instances.”

**Comparing Algorithm with prediction in the form of best accuracy result:**

“It is important to check the performance of multiple different machine learning algorithms consistently and it'll discover to make a test harness to check multiple different machine learning algorithms in Python with scikit-learn. It can use this test harness as a template on your own machine learning problems and add more and different algorithms to check. Each model will have different performance characteristics. Using resampling methods like cross validation, you'll get an estimate for the way accurate each model is also on unseen data. It has to be ready to use these estimates to decide on one or two best models from the suite of models that you simply have created. When have a brand new dataset, it's a decent idea to visualize the information using different techniques so as to appear at the information from different perspectives. The identical idea applies to model selection. You ought to use variety of various ways of staring at the estimated accuracy of your machine learning algorithms so as to decide on the one or two to finalize. How to try to this can be to use different visualization methods to indicate the typical accuracy, variance and other properties of the distribution of model accuracies.”

“In the next section you may discover exactly how you'll do this in Python with scikit-learn. The key to a good comparison of machine learning algorithms is ensuring that every algorithm is evaluated within the same way on the identical data and it can do this by forcing each algorithm to be evaluated on the same test harness.”

In the example below 5 different algorithms are compared:

* Logistic Regression
* Random Forest
* K-Nearest Neighbors
* Decision tree
* Support Vector Machines
* Naive Bayes
* “Now, the dimensions of new features in a numpy array called ‘n’ and it want to predict the species of this features and to do using the predict method which takes this array as input and spits out predicted target value as output.”
* “So, the predicted target value comes out to be 0.Finally to find the test score which is the ratio of no. of predictions found correct and total predictions made and finding accuracy score method which basically compares the actual values of the test set with the predicted values.”

**5. RESULT AND DISCUSSION**

**Performance of Machine Learning parameters:**

Performance measurements confusion matrix using machine learning algorithm:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Parameters** | **LR** | **DT** | **RF** | **SVC** | **NB** | **KNN** |
| **TP** | 0 | 149 | 144 | 42 | 0 | 116 |
| **TN** | 1448 | 1370 | 1376 | 1447 | 1448 | 1406 |
| **FP** | 0 | 78 | 72 | 1 | 0 | 42 |
| **FN** | 224 | 75 | 80 | 182 | 224 | 108 |
| **TPR** | 0 | 0.66 | 0.64 | 0.18 | 0 | 0.51 |
| **TNR** | 1 | 0.94 | 0.95 | 0.99 | 1 | 0.97 |
| **FPR** | 0 | 0.05 | 0.04 | 0 | 0 | 0.02 |
| **FNR** | 1 | 0.33 | 0.35 | 0.81 | 1 | 0.48 |
| **PPV** | - | 0.65 | 0.66 | 0.97 | - | 0.73 |
| **NPV** | 0.86 | 0.94 | 0.94 | 0.88 | 0.86 | 0.92 |

Comparison of accuracy calculations using machine learning algorithm:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Parameters** | **LR** | **DT** | **RF** | **SVC** | **NB** | **KNN** |
| **Precision** | 0.87 | 0.95 | 0.95 | 0.89 | 0.87 | 0.93 |
| **Recall** | 1 | 0.95 | 0.95 | 1 | 1 | 0.97 |
| **F1-Score** | 0.93 | 0.95 | 0.95 | 0.94 | 0.93 | 0.95 |
| **Sensitivity** | 1 | 0.94 | 0.95 | 0.99 | 1 | 0.97 |
| **Specificity** | 0 | 0.66 | 0.64 | 0.18 | 0 | 0.51 |
| **Accuracy (%)** | 86.60 | 90.84 | 90.90 | 89.05 | 86.60 | 91.02 |

**Performance of Spam classifier parameters:**

Comparison of accuracy calculations using spam classifier methods:

|  |  |  |
| --- | --- | --- |
| **Parameters** | **SC (TF-IDF)** | **SC (BOW)** |
| **Precision** | 0.89 | 0.87 |
| **Recall** | 0.72 | 0.57 |
| **F1-Score** | 0.80 | 0.69 |
| **Accuracy (%)** | 94.91 | 92.73 |

**Accuracy Comparison of proposed work:**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Parameters** | **LR** | **DT** | **RF** | **SVM** | **NB** | **KNN** | **SC (TF-IDF)** | **SC (BOW)** |
| **Accuracy (%)** | 86.60 | 90.84 | 90.90 | 89.05 | 86.60 | 91.02 | 94.91 | 92.73 |

**6. CONCLUSION AND FUTURE WORK**

**Conclusion**

“The analytical process started from data cleaning and processing, missing value, exploratory analysis and finally model building and evaluation. Find and compare the accuracy of each algorithm which is helpful for future perdition and the highest accuracy result is Spam classifier of TF-IDF method classifier (94.91%). Additionally, calculate classification report and confusion matrix on public test set of given attributes by supervised machine learning algorithm method.”

**Future Work**

* ISP wants to automate the detecting the message as spam or not by given attributes from eligibility process (real time).
* To automate this process by show the prediction result in web application or desktop application in future.
* To optimize the work to implement in Artificial Intelligence environment.

**References**

[1] Dataset- https://www.kaggle.com/uciml/sms-spam-collection-dataset

[2] Tokenization- https://nlp.stanford.edu/IR-book/html/htmledition/tokenization-1.html

[3] Stemming-

https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html

[4] Dropping common terms (stop words)-

https://nlp.stanford.edu/IR-book/html/htmledition/dropping-common-terms-stop-words-1.html

[5]A review of machine learning approaches to Spam filtering

Authors: - S.GuzellaWalmir M.Caminhas

[6] Survey of review spam detection using machine learning techniques.

Authors:-Taghi M. Khoshgoftaar, Joseph D.Prusa, Aaron N. Richter and Hamzah Al

Najada

[7] Uncovering social spammers: machine learning

Authors: -Kyumin Lee, James Caverlee Steve Webb

[8]Detecting Spam Bots in Online Social Networking Sites: A Machine Learning Ap

proach by Alex Hai Wang

**APPENDIX**

**Sample Code:**

#import library packages

import pandas as p

import numpy as n

#Load given dataset

data = p.read\_csv('spam.csv', encoding='latin-1')

del data["Unnamed: 2"]

del data["Unnamed: 3"]

del data["Unnamed: 4"]

#shape

data.shape

data = data.rename(columns={"v1":"label", "v2":"text"})

#To describe the dataframe

df.describe()

#Checking datatype and information about dataset

df.info()

#Checking for duplicate data

df.duplicated()

#find sum of duplicate data

sum(df.duplicated())

#Checking sum of missing values

df.isnull().sum()

from sklearn.preprocessing import LabelEncoder

var\_mod = ['label', 'text']

le = LabelEncoder()

for i in var\_mod:

df[i] = le.fit\_transform(df[i]).astype(str)

#preprocessing, split test and dataset, split response variable

X = df.drop(labels='label', axis=1)

#Response variable

y = df.loc[:,'label']

#We'll use a test size of 30%. We also stratify the split on the response variable, which is very important to do because there are so few fraudulent transactions.

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=1, stratify=y)

print("Number of training dataset: ", len(X\_train))

print("Number of test dataset: ", len(X\_test))

print("Total number of dataset: ", len(X\_train)+len(X\_test))

df.groupby('label').describe()

#plotting graph for distribution

import matplotlib.pyplot as plt

import seaborn as sns

sns.countplot(x = "label", data = df)

df.loc[:, 'label'].value\_counts()

plt.title('Distribution of Spam and Ham')

# plotting graph by length.

ham =df[df['label'] == 'ham']['text'].str.len()

sns.distplot(ham, label='Ham')

spam = df[df['label'] == 'spam']['text'].str.len()

sns.distplot(spam, label='Spam')

plt.title('Distribution by Length')

plt.legend()

#plotting graph by digits.

ham1 = df[df['label'] == 'ham']['text'].str.replace(r'\D+', '').str.len()

sns.distplot(ham1, label='Ham')

spam1 = df[df['label'] == 'spam']['text'].str.replace(r'\D+', '').str.len()

sns.distplot(spam1, label='Spam')

plt.title('Distribution by Digits')

plt.legend()

#for counting frequently occurence of spam and ham.

from collections import Counter

count1 = Counter(" ".join(df[df['label']=='ham']["text"]).split()).most\_common(30)

data1 = p.DataFrame.from\_dict(count1)

data1 = data1.rename(columns={0: "words of ham", 1 : "count"})

count2 = Counter(" ".join(df[df['label']=='spam']["text"]).split()).most\_common(30)

data2 = p.DataFrame.from\_dict(count2)

data2 = data2.rename(columns={0: "words of spam", 1 : "count\_"})

#Graph for top 30 words of ham

data1.plot.bar(legend = False, color = 'red',figsize = (20,15))

y\_pos = n.arange(len(data1["words of ham"]))

plt.xticks(y\_pos, data1["words of ham"])

plt.title('Top 30 words of ham')

plt.xlabel('words')

plt.ylabel('number')

plt.show()

import nltk

from nltk.corpus import stopwords

from nltk.stem.porter import PorterStemmer

import re

import string

# remove whitespaces

df['text']=df['text'].str.strip()

# lowercase the text

df['text'] = df['text'].str.lower()

#remove punctuation

punc = string.punctuation

table = str.maketrans('','',punc)

df['text']=df['text'].apply(lambda x: x.translate(table))

# tokenizing each message

df['word\_tokens']=df.apply(lambda x: x['text'].split(' '),axis=1)

# removing stopwords

df['cleaned\_text'] = df.apply(lambda x: [word for word in x['word\_tokens'] if word not in stopwords.words('english')],axis=1)

# stemming

ps = PorterStemmer()

df['stemmed']= df.apply(lambda x: [ps.stem(word) for word in x['cleaned\_text']],axis=1)

# remove single letter words

df['final\_text'] = df.apply(lambda x: ' '.join([word for word in x['stemmed'] if len(word)>1]),axis=1)

# label encoding ham=0 and spam=1

df.loc[df['label']=='ham','label']=0

df.loc[df['label']=='spam','label']=1

# divide the set in training and test

from sklearn.model\_selection import train\_test\_split

X,X\_test,y,y\_test = train\_test\_split(df.loc[:,'text':],df['label'],test\_size=0.2)

# Now we'll create a vocabulary for the training set with word count

from collections import defaultdict

vocab=defaultdict(int)

for text in X['final\_text'].values:

for elem in text.split(' '):

vocab[elem]+=1

from wordcloud import WordCloud

# Now we look at the types of words in ham and spam. We plot wordclouds for both

ham\_text=' '.join(X.loc[y==0,'final\_text'].values)

ham\_wordcloud = WordCloud(background\_color='white',max\_words=2000).generate(ham\_text)

spam\_text=' '.join(X.loc[y==1,'final\_text'].values)

spam\_wordcloud = WordCloud(background\_color='white',max\_words=2000).generate(spam\_text)

plt.figure(figsize=[20,30])

plt.subplot(1,2,1)

plt.imshow(spam\_wordcloud,interpolation='bilinear')

plt.title('SPAM')

plt.axis('off')

plt.subplot(1,2,2)

plt.imshow(ham\_wordcloud, interpolation='bilinear')

plt.axis('off')

plt.title('HAM')

X = df.drop(labels='label', axis=1)

#Response variable

y = df.loc[:,'label']

del df

#We'll use a test size of 30%. We also stratify the split on the response variable, which is very important to do because there are so few fraudulent transactions.

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=1, stratify=y)

#for our convienient we delete X,y variable for differentiate confusion

del X, y

# Prevent view warnings

X\_train.is\_copy = False

X\_test.is\_copy = False

from sklearn.linear\_model import LogisticRegression

logR= LogisticRegression()

logR.fit(X\_train,y\_train)

predictR = logR.predict(X\_test)

print("")

print('Classification report of Logistic Regression Results:')

print("")

print(classification\_report(y\_test,predictR))

x = (accuracy\_score(y\_test,predictR)\*100)

print('Accuracy result of Logistic Regression is:', x)

print("")

cm1=confusion\_matrix(y\_test,predictR)

print('Confusion Matrix result of Logistic Regression is:\n',cm1)

print("")

sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])

print('Sensitivity : ', sensitivity1 )

print("")

specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])

print('Specificity : ', specificity1)

print("")

TN = cm1[0][0]

FN = cm1[1][0]

TP = cm1[1][1]

FP = cm1[0][1]

print("True Positive :",TP)

print("True Negative :",TN)

print("False Positive :",FP)

print("False Negative :",FN)

print("")

TPR = TP/(TP+FN)

TNR = TN/(TN+FP)

FPR = FP/(FP+TN)

FNR = FN/(TP+FN)

print("True Positive Rate :",TPR)

print("True Negative Rate :",TNR)

print("False Positive Rate :",FPR)

print("False Negative Rate :",FNR)

print("")

PPV = TP/(TP+FP)

NPV = TN/(TN+FN)

print("Positive Predictive Value :",PPV)

print("Negative predictive value :",NPV)

from sklearn.tree import DecisionTreeClassifier

dtree = DecisionTreeClassifier()

dtree.fit(X\_train, y\_train)

predictDT = dtree.predict(X\_test)

print("")

print('Classification report of Decision Tree Classifier Results:')

print("")

print(classification\_report(y\_test,predictDT))

x = (accuracy\_score(y\_test,predictDT)\*100)

print('Accuracy result of Decision Tree Classifier is', x)

print("")

cm2=confusion\_matrix(y\_test,predictDT)

print('Confusion Matrix result of Decision Tree Classifier is:\n', confusion\_matrix(y\_test,predictDT))

print("")

sensitivity1 = cm2[0,0]/(cm2[0,0]+cm2[0,1])

print('Sensitivity : ', sensitivity1 )

print("")

specificity1 = cm2[1,1]/(cm2[1,0]+cm2[1,1])

print('Specificity : ', specificity1)

from sklearn.ensemble import RandomForestClassifier

rfc = RandomForestClassifier()

rfc.fit(X\_train,y\_train)

predictR = rfc.predict(X\_test)

print("")

print('Classification report of Random Forest Results:')

print("")

print(classification\_report(y\_test,predictR))

x = (accuracy\_score(y\_test,predictR)\*100)

print('Accuracy result of Random Forest is:', x)

print("")

cm1=confusion\_matrix(y\_test,predictR)

print('Confusion Matrix result of Random Forest is:\n',cm1)

print("")

sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])

print('Sensitivity : ', sensitivity1 )

print("")

specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])

print('Specificity : ', specificity1)

print("")

from sklearn.svm import SVC

s = SVC()

s.fit(X\_train,y\_train)

predicts = s.predict(X\_test)

print("")

print('Classification report of Support Vector Machines Results:')

print("")

print(classification\_report(y\_test,predicts))

x = (accuracy\_score(y\_test,predicts)\*100)

print('Accuracy result of Support Vector Machines is:', x)

print("")

cm2=confusion\_matrix(y\_test,predicts)

print('Confusion Matrix result of Support Vector Machines is:\n',cm2)

print("")

sensitivity1 = cm2[0,0]/(cm2[0,0]+cm2[0,1])

print('Sensitivity : ', sensitivity1 )

print("")

specificity1 = cm2[1,1]/(cm2[1,0]+cm2[1,1])

print('Specificity : ', specificity1)

print("")

from sklearn.naive\_bayes import GaussianNB

gnb = GaussianNB()

gnb.fit(X\_train,y\_train)

predictR = gnb.predict(X\_test)

print("")

print('Classification report of Naive Bayes Results:')

print("")

print(classification\_report(y\_test,predictR))

x = (accuracy\_score(y\_test,predictR)\*100)

print('Accuracy result of Naive Bayes is:', x)

print("")

cm1=confusion\_matrix(y\_test,predictR)

print('Confusion Matrix result of Naive Bayes is:\n',cm1)

print("")

sensitivity1 = cm1[0,0]/(cm1[0,0]+cm1[0,1])

print('Sensitivity : ', sensitivity1 )

print("")

specificity1 = cm1[1,1]/(cm1[1,0]+cm1[1,1])

print('Specificity : ', specificity1)

print("")

from sklearn.neighbors import KNeighborsClassifier

knnc = KNeighborsClassifier()

knnc.fit(X\_train,y\_train)

predictR = knnc.predict(X\_test)

print("")

print('Classification report of K-Nearest Neighbor Results:')

print("")

print(classification\_report(y\_test,predictR))

x = (accuracy\_score(y\_test,predictR)\*100)

print('Accuracy result of K-Nearest Neighbor is:', x)

print("")

cm2=confusion\_matrix(y\_test,predictR)

print('Confusion Matrix result of K-Nearest Neighbor is:\n',cm2)

print("")

sensitivity1 = cm2[0,0]/(cm2[0,0]+cm2[0,1])

print('Sensitivity : ', sensitivity1 )

print("")

specificity1 = cm2[1,1]/(cm2[1,0]+cm2[1,1])

print('Specificity : ', specificity1)

print("")

# # Spam Filter In Python3 and Python4

from nltk.tokenize import word\_tokenize

from nltk.corpus import stopwords

from nltk.stem import PorterStemmer

import matplotlib.pyplot as plt

from wordcloud import WordCloud

from math import log,sqrt

import pandas as pd

import numpy as np

import tkinter

from tkinter import messagebox

# # loading the dataset in the mails

mails=pd.read\_csv('spam.csv',encoding='latin-1')

mails.drop(['Unnamed: 2','Unnamed: 3','Unnamed: 4'],axis=1,inplace=True)

mails.rename(columns={'v1':'labels','v2':'text'},inplace=True)

mails['label']=mails['labels'].map({'ham':0,'spam':1})

mails.drop(['labels'],axis=1,inplace=True)

totalMails=4825+747

trainIndex,testIndex=list(),list()

for i in range(mails.shape[0]):

if np.random.uniform(0,1)<0.75:

trainIndex+=[i]

else:

testIndex+=[i]

trainData=mails.loc[trainIndex]

testData=mails.loc[testIndex]

trainData.reset\_index(inplace=True)

trainData.drop(['index'],axis=1,inplace=True)

testData.reset\_index(inplace=True)

testData.drop('index',axis=1,inplace=True)

def process\_message(text, lower\_case = True, stem = True, stop\_words = True, gram = 2):

if lower\_case:

text = text.lower()

words = word\_tokenize(text)

words = [w for w in words if len(w) > 2]

if gram > 1:

w = []

for i in range(len(words) - gram + 1):

w += [' '.join(words[i:i + gram])]

return w

if stop\_words:

sw = stopwords.words('english')

words = [word for word in words if word not in sw]

if stem:

stemmer = PorterStemmer()

words = [stemmer.stem(word) for word in words]

return words

class SpamClassifier(object):

def \_\_init\_\_(self, trainData, method = 'tf-idf'):

self.mails, self.labels = trainData['text'], trainData['label']

self.method = method

def train(self):

self.calc\_TF\_and\_IDF()

if self.method == 'tf-idf':

self.calc\_TF\_IDF()

else:

self.calc\_prob()

def calc\_prob(self):

self.prob\_spam =dict()

self.prob\_ham = dict()

for word in self.tf\_spam:

self.prob\_spam[word] = (self.tf\_spam[word] + 1) / (self.spam\_words + \

len(list(self.tf\_spam.keys())))

for word in self.tf\_ham:

self.prob\_ham[word] = (self.tf\_ham[word] + 1) / (self.ham\_words + \

len(list(self.tf\_ham.keys())))

self.prob\_spam\_mail, self.prob\_ham\_mail = self.spam\_mails / self.total\_mails, self.ham\_mails / self.total\_mails

def calc\_TF\_and\_IDF(self):

noOfMessages = self.mails.shape[0]

self.spam\_mails, self.ham\_mails = self.labels.value\_counts()[1], self.labels.value\_counts()[0]

self.total\_mails = self.spam\_mails + self.ham\_mails

self.spam\_words = 0

self.ham\_words = 0

self.tf\_spam = dict()

self.tf\_ham = dict()

self.idf\_spam = dict()

self.idf\_ham = dict()

for i in range(noOfMessages):

message\_processed = process\_message(self.mails[i])

count = list() #To keep track of whether the word has ocured in the message or not.

#For IDF

for word in message\_processed:

if self.labels[i]:

self.tf\_spam[word] = self.tf\_spam.get(word, 0) + 1

self.spam\_words += 1

else:

self.tf\_ham[word] = self.tf\_ham.get(word, 0) + 1

self.ham\_words += 1

if word not in count:

count += [word]

for word in count:

if self.labels[i]:

self.idf\_spam[word] = self.idf\_spam.get(word, 0) + 1

else:

self.idf\_ham[word] = self.idf\_ham.get(word, 0) + 1

def calc\_TF\_IDF(self):

self.prob\_spam = dict()

self.prob\_ham = dict()

self.sum\_tf\_idf\_spam = 0

self.sum\_tf\_idf\_ham = 0

for word in self.tf\_spam:

self.prob\_spam[word] = (self.tf\_spam[word]) \* log((self.spam\_mails + self.ham\_mails) \

/ (self.idf\_spam[word] + self.idf\_ham.get(word, 0)))

self.sum\_tf\_idf\_spam += self.prob\_spam[word]

for word in self.tf\_spam:

self.prob\_spam[word] = (self.prob\_spam[word] + 1) / (self.sum\_tf\_idf\_spam + len(list(self.prob\_spam.keys())))

for word in self.tf\_ham:

self.prob\_ham[word] = (self.tf\_ham[word]) \* log((self.spam\_mails + self.ham\_mails) \

/ (self.idf\_spam.get(word, 0) + self.idf\_ham[word]))

self.sum\_tf\_idf\_ham += self.prob\_ham[word]

for word in self.tf\_ham:

self.prob\_ham[word] = (self.prob\_ham[word] + 1) / (self.sum\_tf\_idf\_ham + len(list(self.prob\_ham.keys())))

self.prob\_spam\_mail, self.prob\_ham\_mail = self.spam\_mails / self.total\_mails, self.ham\_mails / self.total\_mails

def classify(self, processed\_message):

pSpam, pHam = 0, 0

for word in processed\_message:

if word in self.prob\_spam:

pSpam += log(self.prob\_spam[word])

else:

if self.method == 'tf-idf':

pSpam -= log(self.sum\_tf\_idf\_spam + len(list(self.prob\_spam.keys())))

else:

pSpam -= log(self.spam\_words + len(list(self.prob\_spam.keys())))

if word in self.prob\_ham:

pHam += log(self.prob\_ham[word])

else:

if self.method == 'tf-idf':

pHam -= log(self.sum\_tf\_idf\_ham + len(list(self.prob\_ham.keys())))

else:

pHam -= log(self.ham\_words + len(list(self.prob\_ham.keys())))

pSpam += log(self.prob\_spam\_mail)

pHam += log(self.prob\_ham\_mail)

return pSpam >= pHam

def predict(self, testData):

result = dict()

for (i, message) in enumerate(testData):

processed\_message = process\_message(message)

result[i] = int(self.classify(processed\_message))

return result

def metrics(labels, predictions):

true\_pos, true\_neg, false\_pos, false\_neg = 0, 0, 0, 0

for i in range(len(labels)):

true\_pos += int(labels[i] == 1 and predictions[i] == 1)

true\_neg += int(labels[i] == 0 and predictions[i] == 0)

false\_pos += int(labels[i] == 0 and predictions[i] == 1)

false\_neg += int(labels[i] == 1 and predictions[i] == 0)

precision = true\_pos / (true\_pos + false\_pos)

recall = true\_pos / (true\_pos + false\_neg)

Fscore = 2 \* precision \* recall / (precision + recall)

accuracy = (true\_pos + true\_neg) / (true\_pos + true\_neg + false\_pos + false\_neg)

print("Precision: ", precision)

print("Recall: ", recall)

print("F-score: ", Fscore)

print("Accuracy: ", accuracy)

root = tkinter.Tk()

root.withdraw()

def alldonewithflyingcolors(pm,new):

if sc\_tf\_idf.classify(pm)==True:

#ctypes.windll.user32.MessageBoxW(0,'Alert')

# print('AlertIt is a spam email')

#tkMessageBox.showinfo('alert')

messagebox.showwarning("Message Passed to Spam Filter Model",message=new)

messagebox.showerror("Alert",message="Answer==Spam Email")

#messagebox.ABORT()

else:

messagebox.showwarning("Message Passed to Spam Filter Model",message=new)

messagebox.showinfo("Good News",message="Answer==Good Email")

def metrics(labels, predictions):

true\_pos, true\_neg, false\_pos, false\_neg = 0, 0, 0, 0

for i in range(len(labels)):

true\_pos += int(labels[i] == 1 and predictions[i] == 1)

true\_neg += int(labels[i] == 0 and predictions[i] == 0)

false\_pos += int(labels[i] == 0 and predictions[i] == 1)

false\_neg += int(labels[i] == 1 and predictions[i] == 0)

precision = true\_pos / (true\_pos + false\_pos)

recall = true\_pos / (true\_pos + false\_neg)

Fscore = 2 \* precision \* recall / (precision + recall)

accuracy = (true\_pos + true\_neg) / (true\_pos + true\_neg + false\_pos + false\_neg)

print("Precision: ", precision)

print("Recall: ", recall)

print("F-score: ", Fscore)

print("Accuracy: ", accuracy)

root = tkinter.Tk()

root.withdraw()

def alldonewithflyingcolors(pm,new):

if sc\_tf\_idf.classify(pm)==True:

#ctypes.windll.user32.MessageBoxW(0,'Alert')

# print('AlertIt is a spam email')

#tkMessageBox.showinfo('alert')

messagebox.showwarning("Message Passed to Spam Filter Model",message=new)

messagebox.showerror("Alert",message="Answer==Spam Email")

#messagebox.ABORT()

else:

messagebox.showwarning("Message Passed to Spam Filter Model",message=new)

messagebox.showinfo("Good News",message="Answer==Good Email")

sc\_tf\_idf=SpamClassifier(trainData,'tf-idf')

sc\_tf\_idf.train()

preds\_tf\_idf=sc\_tf\_idf.predict(testData['text'])

metrics(testData['label'],preds\_tf\_idf)

#%% [markdown]

sc\_bow=SpamClassifier(trainData,'bow')

sc\_bow.train()

preds\_bow=sc\_bow.predict(testData['text'])

metrics(testData['label'],preds\_bow)

**Snapshots**

**OUTPUT SCREENSHOTS:**

**Software involved steps:**

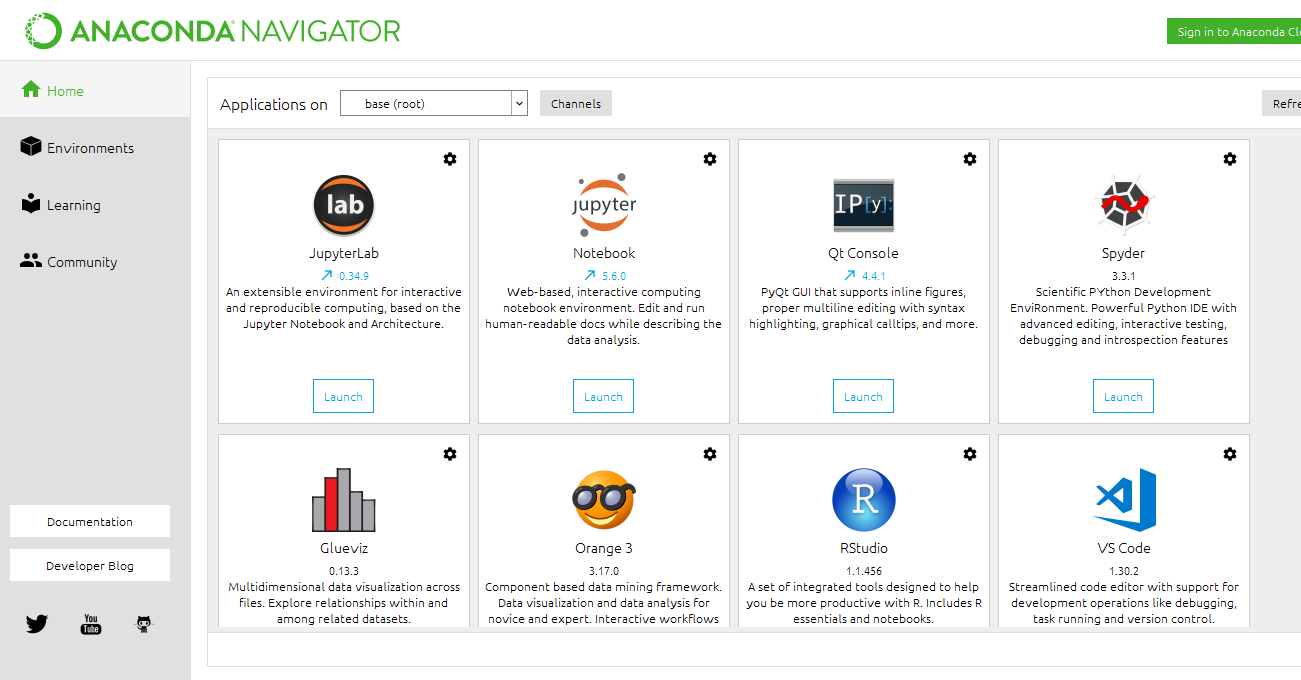
****

Fig 5.1 Open the anaconda navigator

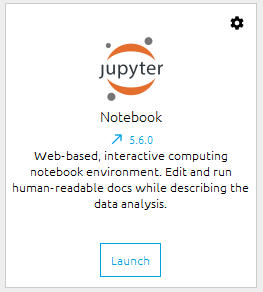
****

Fig 5.2 Launch the jupyter notebook platform