

A Project report

on

SPAM MESSAGE CLASSIFICATION USING MACHINE LEARNING

Submitted in partial fulfillment of the requirements

for the award of the degree of

BACHELOR OF TECHNOLOGY

in

Computer Science & Engineering

By

R.S. SANTHOSHI (184G1A0584)

M. SIREESHA (184G1A0589)

K. TEJDEEP REDDY (194G5A0509)

Under the Guidance of

Mr. Lingam Suman, M.Tech.,(Ph.D)

Assistant Professor



Department of Computer Science & Engineering
SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUA & Approved by AICTE)

(Accredited by NAAC with 'A' Grade & Accredited by NBA (EEE, ECE & CSE))

Rotarypuram Village, B K Samudram Mandal, Ananthapuramu-515701.

2021-2022

SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY

(Affiliated to JNTUA & Approved by AICTE)
(Accredited by NAAC with 'A' Grade & Accredited by NBA (EEE, ECE & CSE)
Rotarypuram Village, B K Samudram Mandal, Ananthapuramu- 515701.

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



Certificate

This is to certify that the Project report entitled **SPAM MESSAGE CLASSIFICATION USING MACHINE LEARNING** is the bonafide work carried out by **R.S. Santhoshi** bearing Roll Number **184G1A0584**, **M. Sireesha** bearing Roll Number **184G1A0589**, **K. Tejdeep Reddy** bearing Roll Number **194G5A0509** in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science & Engineering** during the academic year 2021 - 2022.

Signature of the Guide

Mr. Lingam Suman M.Tech, (Ph.D.)
Assistant Professor

Head of the Department

Mr. P. Veera Prakash M.Tech, (Ph.D.)
Assistant Professor & HOD

Date:

Place: Rotarypuram

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of people who made it possible, whose constant guidance and encouragement crowned our efforts with success. It is a pleasant aspect that we have now the opportunity to express our gratitude for all of them.

It is with immense pleasure that we would like to express my indebted gratitude to our Guide **Mr. Lingam Suman**, M.Tech., (Ph.D.) **Assistant Professor, Computer Science & Engineering**, who has guided us a lot and encouraged us in every step of the project work. We thank him for the stimulating guidance, constant encouragement and constructive criticism which have made possible to bring out this project in time.

We are very much thankful to **Mr. P.Veera Prakash**, M.Tech., (Ph.D.), **Assistant Professor & Head of the Department, Computer Science & Engineering**, for his kind support and for providing necessary facilities to carry out the work.

We wish to convey our special thanks to **Dr. G. Bala Krishna**, Ph.D., **Principal Of Srinivasa Ramanujan Institute of Technology** for giving the required information in doing our project work. Not to forget, we thank all other faculty and non- teaching staff, and my friends who had directly or indirectly helped and supported us in completing our project in time.

We also express our sincere thanks to the Management for providing excellent facilities.

Finally, We wish to convey our gratitude to our family who fostered all the requirements and facilities that we need.

Project Associates

184G1A0584

184G1A0589

194G5A0509

Declaration

We, Ms R.S. Santhoshi with reg no: 184G1A0584, Ms M. Sireesha with reg no: 184G1A0589, Mr K. Tejdeep Reddy with reg no: 194G5A0509 students of SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY, Rotarypuram, hereby declare that the dissertation entitled “SPAM MESSAGE CLASSIFICATION USING MACHINE LEARNING” embodies the report of our project work carried out by us during IV year Bachelor of Technology under the guidance of Mr. Lingam Suman M.Tech, (Ph.D.) Department of CSE, SRINIVASA RAMANUJAN INSTITUTE OF TECHNOLOGY, and this work has been submitted for the partial fulfillment of the requirements for the award of the Bachelor of Technology degree.

The results embodied in this project have not been submitted to any other University or Institute for the award of any Degree or Diploma.

R.S. SANTHOSHI

Reg no: 184G1A0584

M. SIREESHA

Reg no: 184G1A0589

K. TEJDEEP REDDY

Reg no: 194G5A0509

CONTENTS	Page No.
List of figures	VII
List of abbreviations	IX
Abstract	X
Chapter 1 Introduction	1
Chapter 2 Literature Survey	3
Chapter 3 Analysis	5
3.1 Feasibility Study	5
3.2 System Requirements Specifications	6
3.2.1 Functional and Non-Functional Requirements	6
3.2.2 Hardware and Software Specifications	7
3.3 Software Installation	8
Chapter 4 Block Diagram and Architecture	11
Chapter 5 Design & Implementation	13
5.1 System Design	13
5.1.1 UML Diagrams	13
5.2 Algorithms	20
5.2.1 Support Vector Machine	20
5.2.2 Naive Bayes	24
5.2.3 Applications of Naive bayes Algorithm	26
Chapter 6 Introduction to Python	27
Chapter 7 System Testing	44
7.1 Types of Tests	44
7.1.1 Unit Testing	44
7.1.2 Integration Testing	44
7.1.3 Functional Testing	45
7.2 System Test	45
7.2.1 White Box Testing	45
7.2.2 Black Box Testing	45
7.2.3 Unit Testing	46
7.2.4 Integration Testing	46

7.2.5 Acceptance Testing	47
Chapter 8 Execution and Results	48
8.1 Execution	48
8.2 Results	48
Conclusion	54
References	55

LIST OF FIGURES

Fig.No.	Title	Page No.
Fig.3.1	Python version page	8
Fig.3.2	Download Page	9
Fig.3.3	PyCharm open page	9
Fig.3.4	CMD Terminal Screen	10
Fig.4.1	Block diagram of proposed method	11
Fig.4.2	Architecture diagram	12
Fig.5.1.1	Use Case Diagram	14
Fig.5.1.2	Class Diagram	15
Fig.5.1.3	Sequence Diagram	15
Fig.5.1.4	Collaboration Diagram	16
Fig.5.1.5	Deployment Diagram	16
Fig.5.1.6	Activity Diagram	17
Fig.5.1.7	Component Diagram	17
Fig.5.1.8	E-R Diagram	18
Fig.5.1.9	DFD Diagram	19
Fig.5.1.10	Level-1 Diagram	19
Fig.5.1.11	Level-2 Diagram	20
Fig.5.2.1	Support Vector Graph	21
Fig.5.2.2	Hyper Plane graph	21
Fig.5.2.3	Hyper Plane in 2D and 3D	22
Fig.5.2.4	Frequency table	25
Fig.8.1	Runtime Environment	48
Fig.8.2.1	Home Page	48
Fig.8.2.2	About Page	49
Fig.8.2.3	Upload Page	49
Fig.8.2.4	View Page	50
Fig.8.2.5	Pre-processing Page	50

Fig.8.2.6	Naïve Bayes Accuracy	51
Fig.8.2.7	Support Vector Classifier Accuracy	51
Fig.8.2.8	Comparative Chart	52
Fig.8.2.9	Spam Prediction	52
Fig.8.2.10	Ham Prediction	53

LIST OF ABBREVIATIONS

UML	Unified Modeling Language
ER	Entity-relationship
DBMS	Data Base Management System
DFD	Data Flow Diagram
HTML	Hyper Text Markup Language
SVM	Support Vector machine
SMS	Short Message Service

ABSTRACT

We use some communication means to convey messages digitally. Digital tools allow two or more persons to coordinate with each other. This communication can be textual, visual, audio, and written. Smart devices including cell phones are the major sources of communication these days. Intensive communication through SMSs is causing spamming as well. Unwanted text messages define as junk information that we received in gadgets. Most of the companies promote their products or services by sending spam texts which are unwelcome. In general, most of the time spam emails more in numbers than Actual messages. In this paper, we have used text classification techniques to define SMS and spam filtering in a short view, which segregates the messages accordingly. In this paper, we apply some classification methods along with “machine learning algorithms” to identify how many SMS are spam or not. For that reason, we compared different classified methods on dataset collection on which work done. We got 98% results from Naïve Bayes and Support Vector Machine. But there are some variations in their decimal percentages.

Keywords: Spam Messages, Classification, Spam Filtering, Comparison.

CHAPTER-1

INTRODUCTION

Data science is an associative field that utilize experimental techniques, algorithms, mechanisms, and systems for essential knowledge and observation from many organized and unorganized data, which is linked to data mining, deep learning, and big data. Data mining is a field of computer science; this technique function is to analyse rough data into useful information. Furthermore, system principally took knowledge from data. There are many methods such as classification, clustering, and much more for that purpose. Short message services called SMS. Through SMS, you must send messages of 160 characters to each other and large messages split into small multiple messages. It used the standard communication protocols to facilitate cell phones to interchanged short text messages. In past years, text messaging rate increased and the government aims to go on with the change of fast technology. SMS spam is sophisticated in a certain aspect. The lowest SMS rates has allowed people and service providers to leave the problem, and finite opportunity of cell phone spam filtering software. SMS spam is less than email spam. Even though it is an explanation for approximately 1% of transcript directed in the United States and 30% of typescript letters sent trendy Asia. In 2004 SMS spam in the United States, illegal under telephone customer Protection. Who receives unwanted SMS know how to take along the guidance counsellor in the direction of an unimportant situation court from countries. Now China, three highest moveable cell phone hands sign up a shared strategy to competition mobile unsolicited messages by set parameters on the digit of typescript messages directed collectively time since 2009. In this study, we present some algorithms application of classification that classifies objects. We use Classification algorithms for prediction of text messages are spam or not. At this establishment, there must be a training set that holds the items. SMS contain text messages that's why in this study we use text classification, and the using of classification algorithms is to summarize the SMS spam class or SMS into a human being; those texts sent from people means that these messages from human class or mobile phone, and spam messages mostly come

from organizations and companies to promote their products by ads. Because of mobile phones or smartphones generally a communication device and used by people in a wide range for a call or send messages, and voice mail messages are frequently in use.

When we compare SMS spam and email filter spam datasets, email filter spam has a large number of datasets than SMS spam datasets, because they usually are tiny in size. The size of spam SMS is small, that's why the filtering mechanisms scheme of email filtering spam system couldn't be applied to the SMS. In some countries in which Korea, the email spamming less than SMS spamming. But in west areas, the opposite method applied there was email spam more due to low cost than SMS because SMS spamming more expensive and less in the amount. About 50% of the SMS messages are received as a text message on mobile phones which shows as spammed. That's why an SMS filtering system should work in reserve resources as in cell phone hardware. In this study, we used the actual data: ham and spam. we apply several algorithms of classification in which some use in previous work and some new procedure, which can be used for further comparisons and analysis.

CHAPTER-2

LITERATURE SURVEY

1. Tiago, Almeida, José María GómezAkebo Yamakami. Contributions to the Study of SMS Spam Filtering. University of Campinas, Sao Paulo, Brazil.

The growth of mobile phone users has lead to a dramatic increasing of SMS spam messages. In practice, fighting mobile phone spam is difficult by several factors, including the lower rate of SMS that has allowed many users and service providers to ignore the issue, and the limited availability of mobile phone spam-filtering software. On the other hand, in academic settings, a major handicap is the scarcity of public SMS spam datasets, that are sorely needed for validation and comparison of different classifiers. Moreover, as SMS messages are fairly short, content-based spam filters may have their performance degraded. In this paper, we offer a new real, public and non-encoded SMS spam collection that is the largest one as far as we know. Moreover, we compare the performance achieved by several established machine learning methods. The results indicate that Support Vector Machine outperforms other evaluated classifiers and, hence, it can be used as a good baseline for further comparison.

2. Duan, L., Li, N., & Huang, L. (2009). "A new spam short message classification" 2009 First International Workshop on Education Technology and Computer Science, 168- 171.

This paper proposes an approach of dual-filtering messages. First the combination of KNN classification algorithm and rough set separates spam messages from messages. To avoid lowering precision for reduction, it needs to use KNN classification algorithm to re-filter some messages. This method not only improves the speed of classification but also retains high accuracy based on rough set of KNN classification algorithm.

3. Inwhae Joe and Hyetaek Shim, "An SMS Spam Filtering System Using Support Vector Machine," Division of Computer Science and Engineering, Hanyang University, Seoul, 133-791 South Korea.

This paper describes a powerful and adaptive spam filtering system for SMS (Short Messaging Service) that uses SVM (Support Vector Machine) and a thesaurus.

The system isolates words from sample data using a pre-processing device and integrates meanings of isolated words using a thesaurus, generates features of integrated words through chi-square statistics, and studies these features. The system is realized in a Windows environment and its performance is experimentally confirmed.

4. B. G. Becker. Visualizing Decision Table Classifiers. Pages 102- 105, IEEE (1998).

Decision tables [1], like decision trees [2] or neural nets [3], are classification models used for prediction. They are induced by machine learning algorithms. A decision table consists of a hierarchical table in which each entry in a higher level table gets broken down by the values of a pair of additional attributes to form another table. The structure is similar to dimensional stacking [4]. Presented here is a visualization method that allows a model based on many attributes to be understood even by those unfamiliar with machine learning. Various forms of interaction are used to make this visualization more useful than other static designs.

CHAPTER 3

ANALYSIS

3.1 Feasibility study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ◆ Economical Feasibility
- ◆ Technical Feasibility
- ◆ Social Feasibility

Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user.

This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3.2 SYSTEM REQUIREMENTS SPECIFICATIONS

3.2.1 Functional and non-functional requirements

Requirement's analysis is very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and non-functional requirements.

Functional Requirements These are the requirements that the end user specifically demands as basic facilities that the system should offer. All these functionalities need to be necessarily incorporated into the system as a part of the contract. These are represented or stated in the form of input to be given to the system, the operation performed and the output expected. They are basically the requirements stated by the user which one can see directly in the final product, unlike the non-functional requirements.

Examples of functional requirements:

- 1) Authentication of user whenever he/she logs into the system
- 2) System shutdown in case of a cyber-attack
- 3) A verification email is sent to user whenever he/she register for the first time on some software system.

Non-functional requirements These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to other. They are also called non-behavioral requirements.

They basically deal with issues like:

- Portability
- Security
- Maintainability

- Reliability
- Scalability
- Performance
- Reusability
- Flexibility

Examples of non-functional requirements:

- 1) Emails should be sent with a latency of no greater than 12 hours from such an activity.
- 2) The processing of each request should be done within 10 seconds
- 3) The site should load in 3 seconds whenever of simultaneous users are > 10000.

3.2.2 Hardware and Software Specifications H/W

Specifications

- Processor : I3/Intel Processor
- RAM : 8GB (min)
- Hard Disk : 128 GB

S/W Specifications

- Operating System : Windows 10
- Server-side Script : Python 3.6
- IDE : PyCharm
- Libraries Used : Numpy, IO, OS, Flask, Keras, Tensor Flow.

Modules

System

- **Pre-processing-** Resizing and reshaping the images into appropriate format to train our model.
- **Model training-** Use the pre-processed training dataset is used to train our model using SVM, Naive bayes algorithm along with some of the transfer learning methods.
- **Prediction-** The results of our model is display of message images are

either with spam or normal.

- **Generate results-** System can generate results after prediction.

User

- **Upload data-** User can upload messages data into the system.
- **View data-** User can view the uploaded data.
- **View results-** User can view the predicted results.

3.3 SOFTWARE INSTALLATION FOR MACHINE LEARNING PROJECTS

Installing Python

1. To download and install python visit the official website of Python_ <https://www.python.org/downloads/> and choose your version.



Fig.3.1: Python version page

2. Once the download is complete, run the exe for install Python. Now click on Install Now.
3. You can see Python installing at this point.
4. When it finishes, you can see a screen that says the Setup was successful. Now click on "Close".

Installing PyCharm

1. To download PyCharm visit the <https://www.jetbrains.com/pycharm/download/> and click the "DOWNLOAD" link under the Community Section.

Download PyCharm

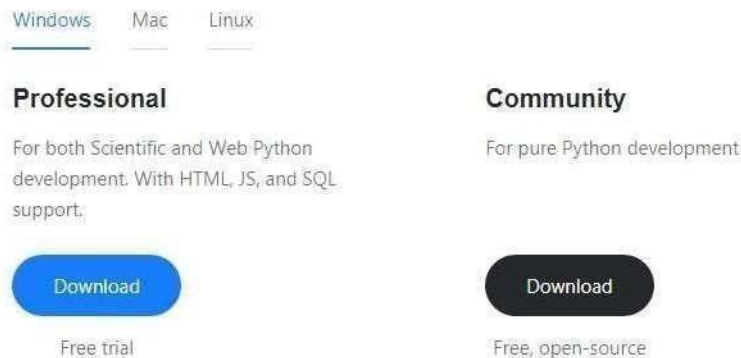


Fig.3.2: Download Page

2. Once the download is complete, run the exe for install PyCharm.
The setup wizard should have started. Click “Next”.
3. On the next screen, Change the installation path if required. Click “Next”.
4. On the next screen, you can create a desktop shortcut if you want and click on “Next”.
5. Choose the start menu folder. Keep selected Jet Brains and click on “Install”.
6. Wait for the installation to finish.
7. Once installation finished, you should receive a message screen that PyCharm is installed. If you want to go ahead and run it, click the “Run PyCharm Community Edition” box first and click “Finish”.
8. After you click on "Finish," the Following screen will appear.
9. You need to install some packages to execute your project in a proper way.

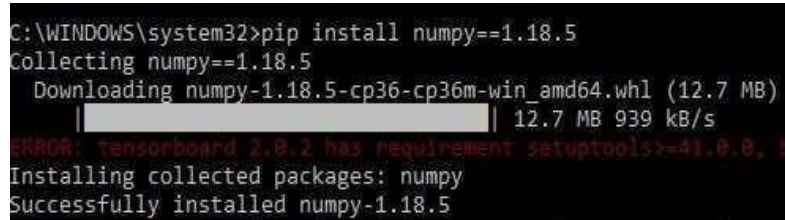


Fig.3.3: PyCharm open page

Open the command prompt/ anaconda prompt or terminal as administrator.

10. The prompt will get open, with specified path, type “pip install package name” which you want to install (like NumPy, pandas, scikit-learn).

Ex: Pip install NumPy



```
C:\WINDOWS\system32>pip install numpy==1.18.5
Collecting numpy==1.18.5
  Downloading numpy-1.18.5-cp36-cp36m-win_amd64.whl (12.7 MB)
    | 12.7 MB 939 kB/s
ERROR: tensorboard 2.0.2 has requirement setuptools>=41.0.0, b
Installing collected packages: numpy
Successfully installed numpy-1.18.5
```

Fig.3.4: CMD Terminal Screen

CHAPTER-4

BLOCK DIAGRAM AND ARCHITECTURE

Block Diagram

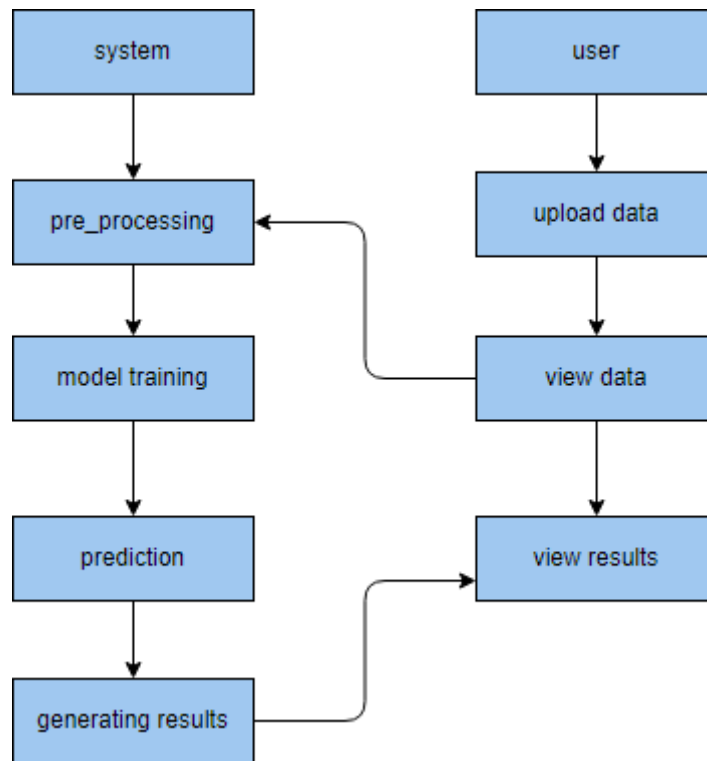


Fig.4.1: Block diagram of proposed method

At first the user has upload the data and it will be viewed what type of data is uploaded. Next the system undergoes pre-processing of the data and then it is trained by using the model. After model training the system will predict whether it is spam or not. At last we can view the results.

Architecture

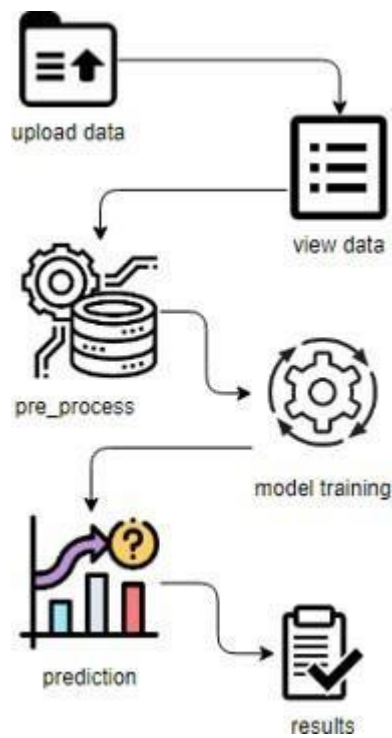


Fig.4.2: Architecture diagram

The above architecture shows the whole process of the spam message classification. The process starts by uploading the data, after that data is viewed and it can be pre-processed. In model training the model trains the data and tests the data. Then prediction is done and gives the results

CHAPTER-5

DESIGN AND IMPLEMENTATION

5.1 SYSTEM DESIGN

5.1.1 UML DIAGRAMS

UML stands for Unified Modelling Language. UML is a standardized general-purpose modelling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: A Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modelling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modelling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modelling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modelling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modelling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

USE CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

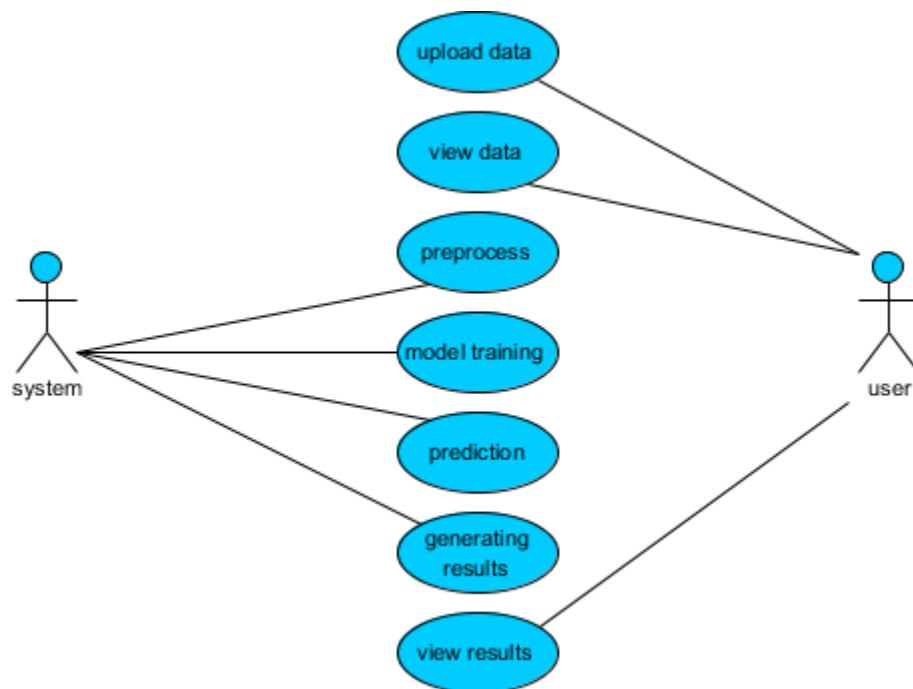


Fig.5.1.1: Use Case Diagram

CLASS DIAGRAM

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

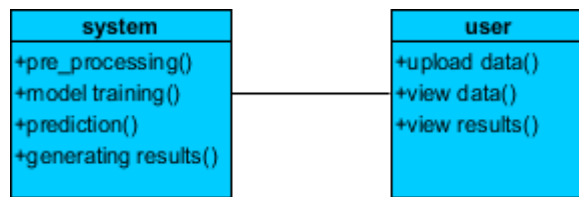


Fig.5.1.2: Class Diagram

SEQUENCE DIAGRAM

A sequence diagram in Unified Modelling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

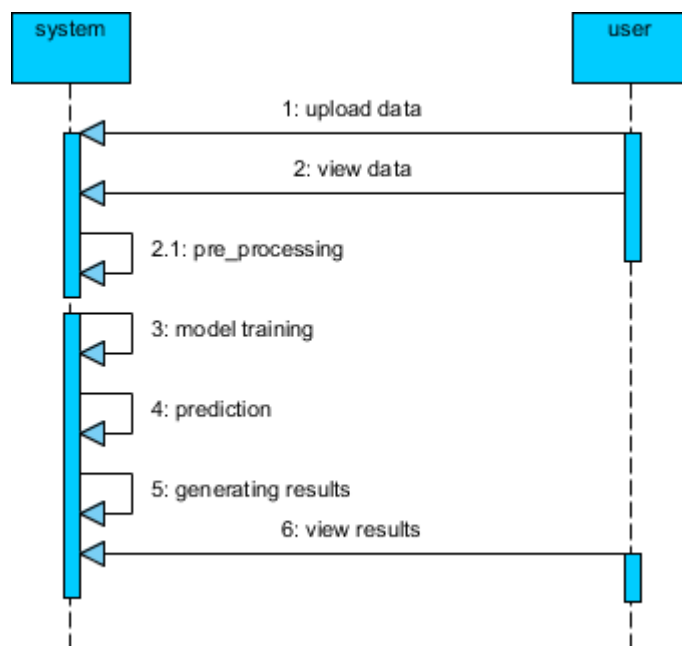


Fig.5.1.3: Sequence Diagram

COLLABORATION DIAGRAM

In collaboration diagram the method call sequence is indicated by some numbering technique as shown below. The number indicates how the methods are called one after another. We have taken the same order management system to describe the collaboration diagram. The method calls are similar to that of a sequence diagram. But the difference is that the sequence diagram does not describe the object organization whereas the collaboration diagram shows the object organization.

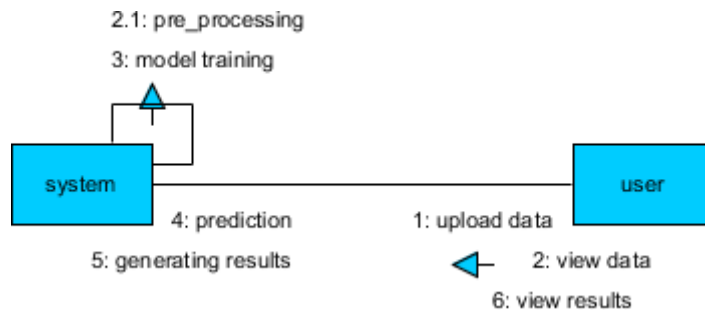


Fig.5.1.4: Collaboration Diagram

DEPLOYMENT DIAGRAM

Deployment diagram represents the deployment view of a system. It is related to the component diagram. Because the components are deployed using the deployment diagrams. A deployment diagram consists of nodes. Nodes are nothing but physical hardware used to deploy the application.

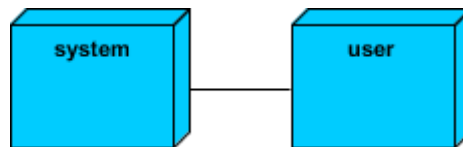


Fig.5.1.5: Deployment Diagram

ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modelling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.

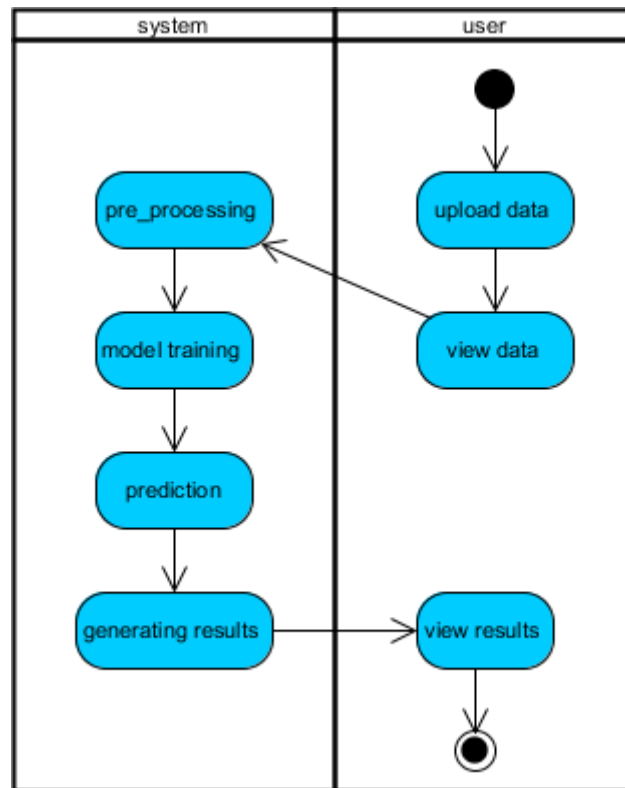


Fig.5.1.6: Activity Diagram

COMPONENT DIAGRAM

A component diagram, also known as a UML component diagram, describes the organization and wiring of the physical components in a system. Component diagrams are often drawn to help model implementation details and double-check that every aspect of the system's required function is covered by planned development.

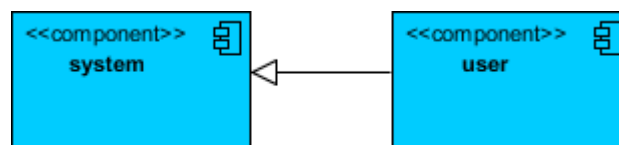


Fig.5.1.7: Component Diagram

ER DIAGRAM

An Entity–relationship model (ER model) describes the structure of a database with the help of a diagram, which is known as Entity Relationship Diagram (ER Diagram).

An ER model is a design or blueprint of a database that can later be implemented as a database. The main components of E-R model are: entity set and relationship set.

An ER diagram shows the relationship among entity sets. An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database. Let's have a look at a simple ER diagram to understand this concept.

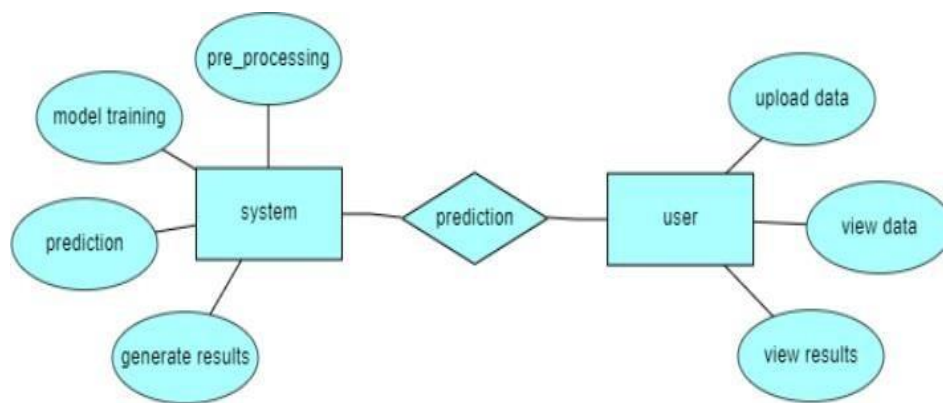


Fig.5.1.8: E-R Diagram

DFD DIAGRAM

A Data Flow Diagram (DFD) is a traditional way to visualize the information flows within a system. A neat and clear DFD can depict a good amount of the system requirements graphically. It can be manual, automated, or a combination of both. It shows how information enters and leaves the system, what changes the information and where information is stored. The purpose of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communications tool between a systems analyst and any person who plays a part in the system that acts as the starting point for redesigning a system.

Context Level Diagram

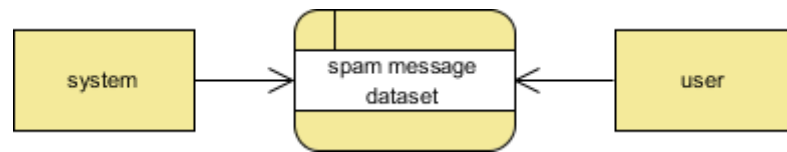


Fig.5.1.9: DFD Diagram

Level 1 Diagram

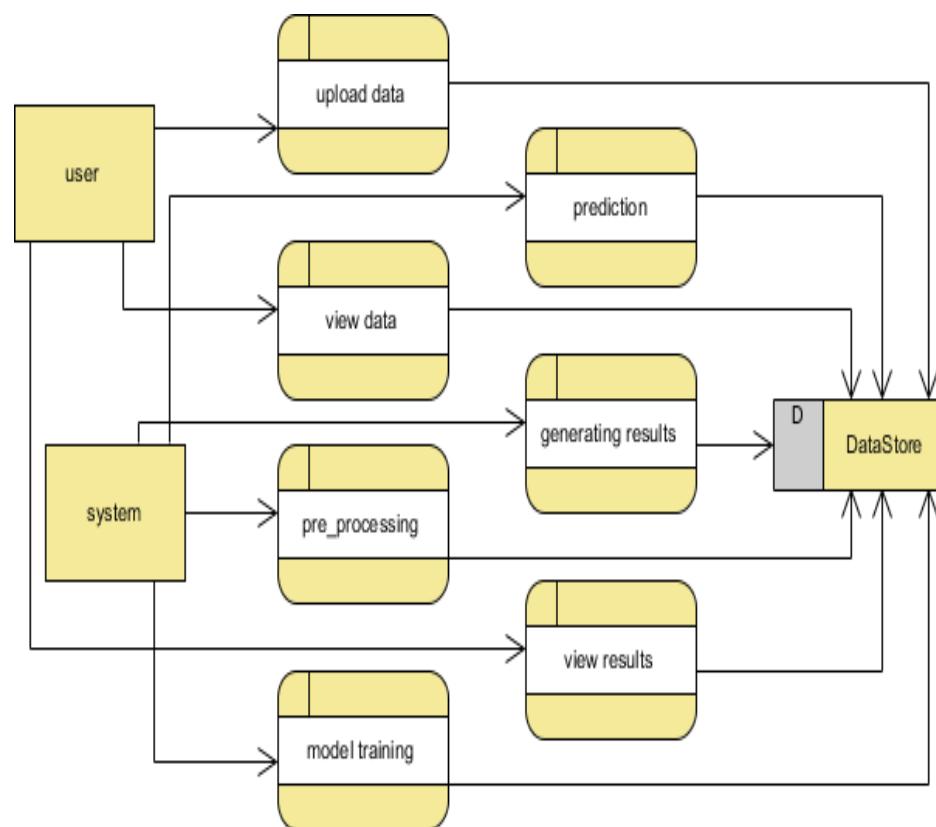


Fig.5.1.10: Level-1 Diagram

Level 2 Diagram

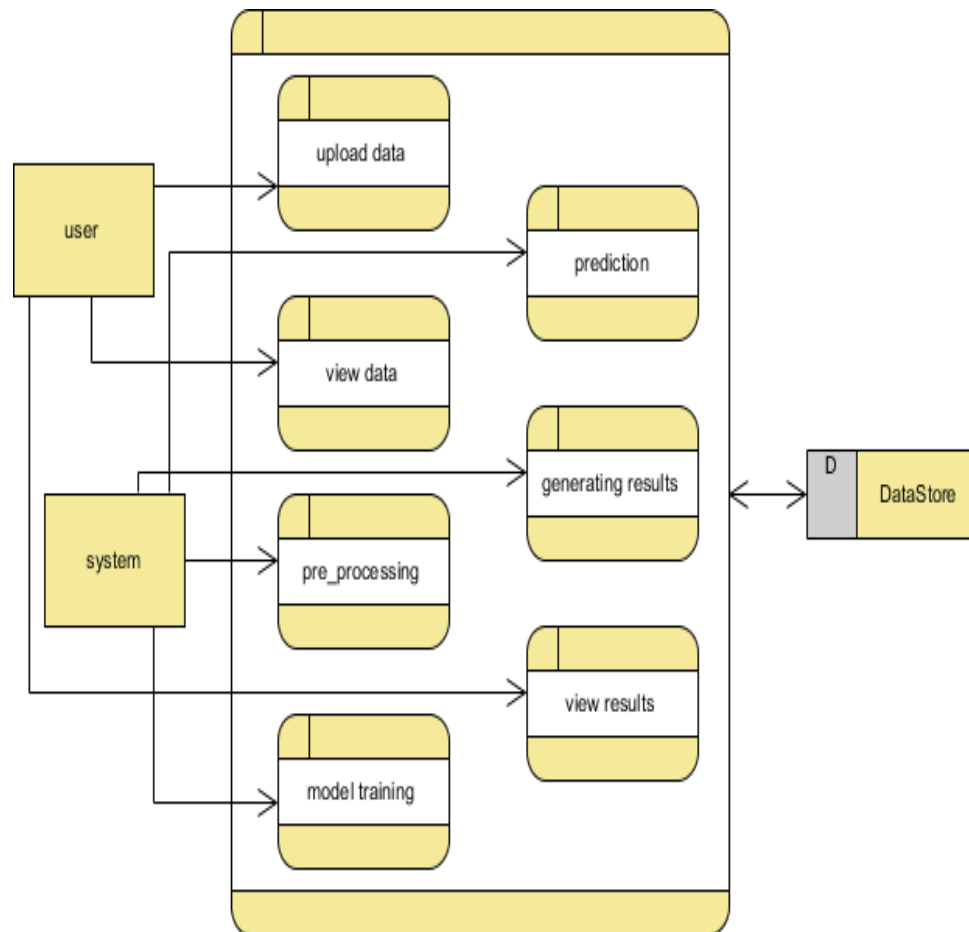


Fig.5.1.11: Level-2 Diagram

5.2 ALGORITHMS

5.2.1 SUPPORT VECTOR MACHINES

The objective of the support vector machine algorithm is to find a hyper plane in an N- dimensional space (N — the number of features) that distinctly classifies the data points.

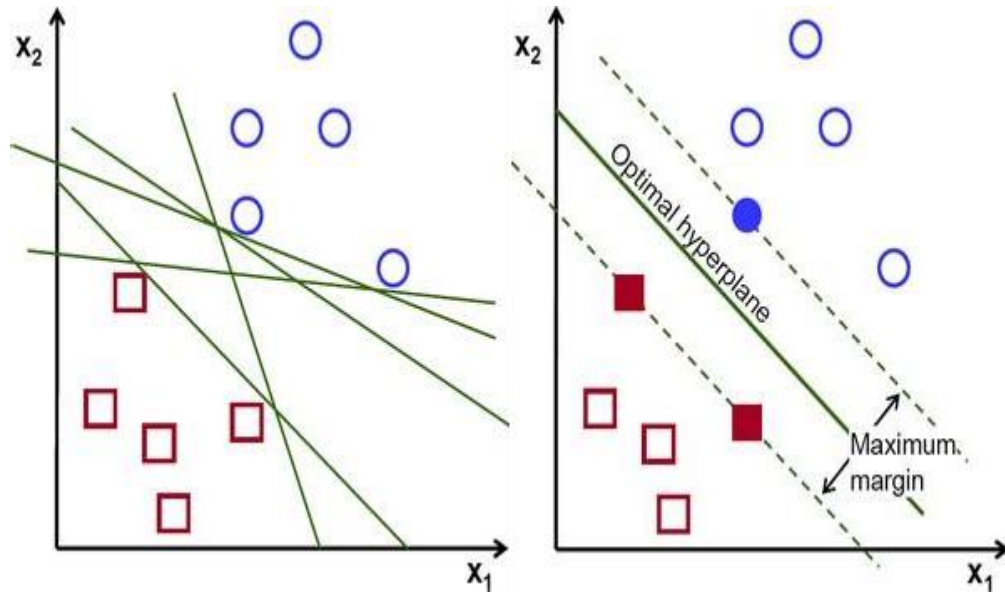


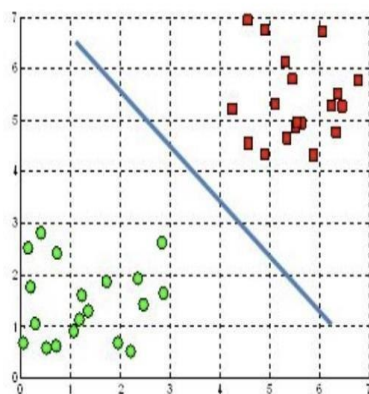
Fig.5.2.1: Support Vector Graph

Possible hyper planes

To separate the two classes of data points, there are many possible hyper planes that could be chosen. Our objective is to find a plane that has the maximum margin, i.e. the maximum distance between data points of both classes. Maximizing the margin distance provides some reinforcement so that future data points can be classified with more confidence.

Hyper planes and Support Vectors

A hyperplane in \mathbb{R}^2 is a line



A hyperplane in \mathbb{R}^3 is a plane

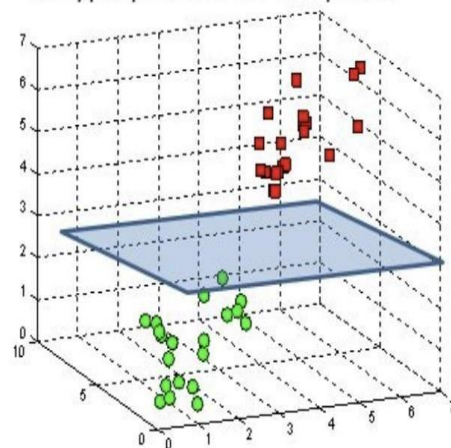


Fig.5.2.2: Hyper Plane graph

Hyper planes in 2D and 3D feature space

Hyper planes are decision boundaries that help classify the data points. Data points falling on either side of the hyper plane can be attributed to different classes. Also, the dimension of the hyper plane depends upon the number of features. If the number of input features is 2, then the hyper plane is just a line. If the number of input features is 3, then the hyper plane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds.

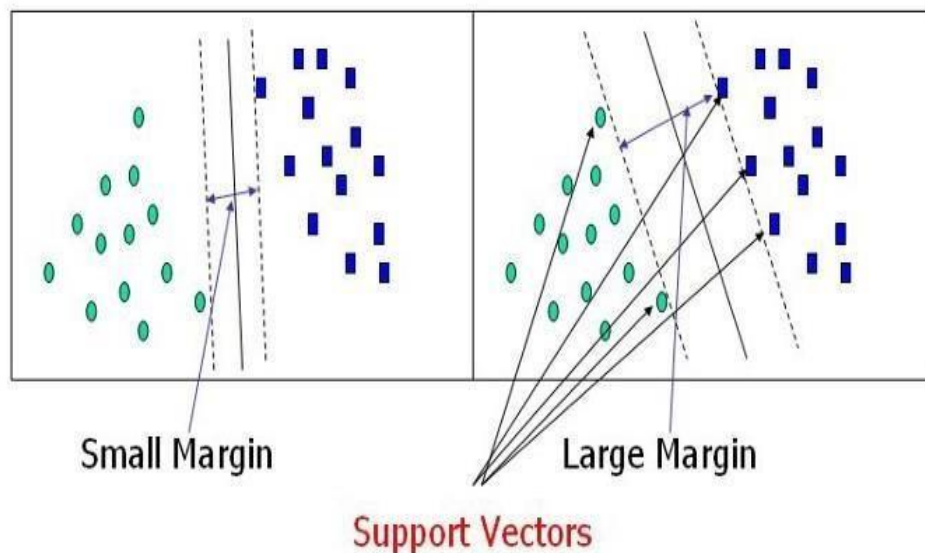


Fig.5.2.3: Hyper Plane in 2D and 3D

Support Vectors

Support vectors are data points that are closer to the hyper plane and influence the position and orientation of the hyper plane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyper plane. These are the points that help us build our SVM.

Large Margin Intuition

In logistic regression, we take the output of the linear function and squash the value within the range of $[0,1]$ using the sigmoid function. If the squashed value is greater than a threshold value (0.5) we assign it a label 1, else we assign it a label 0. In SVM, we take the output of the linear function and if that output is greater than 1, we identify it with one class and if the output is - 1, we identify

is with another class. Since the threshold values are changed to 1 and -1 in SVM, we obtain this reinforcement range of values $([-1, 1])$ which acts as margin.

Cost Function and Gradient Updates

In the SVM algorithm, we are looking to maximize the margin between the data points and the hyper plane. The loss function that helps maximize the margin is hinge loss.

$$c(x, y, f(x)) = \begin{cases} 0, & \text{if } y * f(x) \geq 1 \\ 1 - y * f(x), & \text{else} \end{cases}$$

$$c(x, y, f(x)) = (1 - y * f(x))_+$$

Hinge loss function (function on left can be represented as a function on the right)

The cost is 0 if the predicted value and the actual value are of the same sign. If they are not, we then calculate the loss value. We also add a regularization parameter the cost function. The objective of the regularization parameter is to balance the margin maximization and loss. After adding the regularization parameter, the cost functions look as below.

$$\min_w \lambda \|w\|^2 + \sum_{i=1}^n (1 - y_i \langle x_i, w \rangle)_+$$

Loss function for SVM

Now that we have the loss function, we take partial derivatives with respect to the weights to find the gradients. Using the gradients, we can update our weights.

$$\frac{\partial}{\partial w_k} \lambda \|w\|^2 = 2\lambda w_k$$

$$\frac{\partial}{\partial w_k} (1 - y_i \langle x_i, w \rangle)_+ = \begin{cases} 0, & \text{if } y_i \langle x_i, w \rangle \geq 1 \\ -y_i x_{ik}, & \text{else} \end{cases}$$

When there is no misclassification, i.e., our model correctly predicts the class of our data point, we only have to update the gradient from the regularization parameter.

$$w = w - \alpha \cdot (2\lambda w)$$

Gradient Update — No misclassification

When there is a misclassification, i.e. our model make a mistake on the prediction of the class of our data point, we include the loss along with the regularization parameter to perform gradient update.

$$w = w + \alpha \cdot (y_i \cdot x_i - 2\lambda w)$$

5.2.2 NAIVE BAYES:

A Naive Bayes classifier is a probabilistic machine learning model that's used for classification task. The crux of the classifier is based on the Bayes theorem.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Using Bayes theorem, we can find the probability of **A** happening, given that **B** has occurred. Here, **B** is the evidence and **A** is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of one particular feature does not affect the other. Hence it is called naive.

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

Step 1: Convert the data set into a frequency table

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

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

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

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

=4/14	0.29
=5/14	0.36
=5/14	0.36

Fig.5.2.4: Frequency table

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

Problem: Players will play if weather is sunny. Is this statement being correct? We can solve it using above discussed method of posterior probability.

$$P(\text{Yes} | \text{Sunny}) = P(\text{Sunny} | \text{Yes}) * P(\text{Yes}) / P(\text{Sunny})$$

Here we have $P(\text{Sunny} | \text{Yes}) = 3/9 = 0.33$, $P(\text{Sunny}) = 5/14 = 0.36$, $P(\text{Yes}) = 9/14 = 0.64$ Now, $P(\text{Yes} | \text{Sunny}) = 0.33 * 0.64 / 0.36 = 0.60$, which has higher probability.

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

- It is easy and fast to predict class of test data set. It also performs well in multi class prediction
- When assumption of independence holds, a Naive Bayes classifier performs better compare to other models like logistic regression and you need less training data.
- It performs well in case of categorical input variables compared to numerical variable(s). For numerical variable, normal distribution is assumed (bell curve, which is a strong assumption).

5.2.3 Applications of Naive Bayes Algorithms

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 classification (due to better result in multi class problems and independence rule) have higher success rate as compared to other algorithms. As a result, it is widely used in Spam filtering (identify spam e-mail) and Sentiment Analysis (in social media analysis, to identify positive and negative customer sentiments)

Recommendation System- Naive Bayes Classifier and 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

CHAPTER 6

INTRODUCTION TO PYTHON

What is a Script?

Up to this point, I have concentrated on the interactive programming capability of Python. This is a very useful capability that allows you to type in a program and to have it executed immediately in an interactive mode

Scripts are reusable

Basically, a script is a text file containing the statements that comprise a Python program. Once you have created the script, you can execute it over and over without having to retype it each time.

Scripts are editable

Perhaps, more importantly, you can make different versions of the script by modifying the statements from one file to the next using a text editor. Then you can execute each of the individual versions. In this way, it is easy to create different programs with a minimum amount of typing.

You will need a text editor

Just about any text editor will suffice for creating Python script files. You can use Microsoft Notepad, Microsoft WordPad, Microsoft Word, or just about any word processor if you want to.

Difference between a script and a program Script

Scripts are distinct from the core code of the application, which is usually written in a different language, and are often created or at least modified by the end-user. Scripts are often interpreted from source code or byte code, whereas the applications they control are traditionally compiled to native machine code.

Program

The program has an executable form that the computer can use directly to execute the instructions.

The same program in its human-readable source code form, from which executable programs are derived (e.g., compiled)

Python

What is Python? Chances you are asking yourself this. You may have found this book because you want to learn to program but don't know anything about programming languages. Or you may have heard of programming languages like C, C++, C#, or Java and want to know what Python is and how it compares to "big name" languages. Hopefully I can explain it for you.

Python concepts

If you're not interested in the how and whys of Python, feel free to skip to the next chapter. In this chapter I will try to explain to the reader why I think Python is one of the best languages available and why it's a great one to start programming with.

- Open-source general-purpose language.
- Object Oriented, Procedural, Functional
- Easy to interface with C/ObjC/Java/Fortran
- Easy-ish to interface with C++ (via SWIG)
- Great interactive environment

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

- **Python is Interpreted** – Python is processed at runtime by the interpreter. You donot need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** – You can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** – Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** – Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

History of Python

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

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

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

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

Python Features

Python's features include –

- **Easy-to-learn** – Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** – Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** – Python's source code is fairly easy-to-maintain.
- **A broad standard library** – Python's bulk of the library is very portable and cross- platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** – Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** – Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** – You can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** – Python provides interfaces to all major commercial databases.
- **GUI Programming** – Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.
- **Scalable** – Python provides a better structure and support for large programs than shell scripting.

- Apart from the above-mentioned features, Python has a big list of good features, few are listed below –
- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- IT supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

Dynamic vs Static

Types Python is a dynamic-typed language. Many other languages are static typed, such as C/C++ and Java. A static typed language requires the programmer to explicitly tell the computer what type of “thing” each data value is.

For example, in C if you had a variable that was to contain the price of something, you would have to declare the variable as a “float” type.

This tells the compiler that the only data that can be used for that variable must be a floating-point number, i.e., a number with a decimal point.

If any other data value was assigned to that variable, the compiler would give an error when trying to compile the program.

Python, however, doesn’t require this. You simply give your variables names and assign values to them. The interpreter takes care of keeping track of what kinds of objects your program is using. This also means that you can change the size of the values as you develop the program. Say you have another decimal number (a.k.a. a floating-point number) you need in your program.

With a static typed language, you have to decide the memory size the variable can take when you first initialize that variable. A double is a floating-point value that can handle a much larger number than a normal float (the actual memory sizes depend on the operating environment).

If you declare a variable to be a float but later on assign a value that is too big to it, your program will fail; you will have to go back and change that variable to be a double.

With Python, it doesn't matter. You simply give it whatever number you want and Python will take care of manipulating it as needed. It even works for derived values.

For example, say you are dividing two numbers. One is a floating-point number and one is an integer. Python realizes that it's more accurate to keep track of decimals so it automatically calculates the result as a floating-point number.

Variables

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory.

Based on the data type of a variable, the interpreter allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals or characters in these variables.

Standard Data Types

The data stored in memory can be of many types. For example, a person's age is stored as a numeric value and his or her address is stored as alphanumeric characters. Python has various standard data types that are used to define the operations possible on them and the storage method for each of them.

Python has five standard data types –

- Numbers
- String
- List
- Tuple
- Dictionary

Python Numbers

Number data types store numeric values. Number objects are created when you assign a value to them

Python Strings

Strings in Python are identified as a contiguous set of characters represented in the quotation marks. Python allows for either pairs of single or double quotes. Subsets of strings can be taken using the slice operator ([] and [:]) with indexes starting at 0 in the beginning of the string and working their way from -1 at the end.

Python Lists

Lists are the most versatile of Python's compound data types. A list contains items separated by commas and enclosed within square brackets ([]). To some extent, lists are similar to arrays in C. One difference between them is that all the items belonging to a list can be of different data type.

The values stored in a list can be accessed using the slice operator ([] and [:]) with indexes starting at 0 in the beginning of the list and working their way to end -1. The plus (+) sign is the list concatenation operator, and the asterisk (*) is the repetition operator.

Python Tuples

A tuple is another sequence data type that is similar to the list. A tuple consists of a number of values separated by commas. Unlike lists, however, tuples are enclosed within parentheses.

The main differences between lists and tuples are: Lists are enclosed in brackets ([]) and their elements and size can be changed, while tuples are enclosed in parentheses (()) and cannot be updated. Tuples can be thought of as **read-only** lists.

Python Dictionary

Python's dictionaries are kind of hash table type. They work like associative arrays or hashes found in Perl and consist of key-value pairs. A dictionary key can be almost any Python type, but are usually numbers or strings. Values, on the other hand, can be any arbitrary Python object.

Dictionaries are enclosed by curly braces ({ }) and values can be assigned and accessed using square braces ([]).

Different modes in python

Python has two basic modes: normal and interactive.

The normal mode is the mode where the scripted and finished .py files are run in the Python interpreter.

Interactive mode is a command line shell which gives immediate feedback for each statement, while running previously fed statements in active memory. As new lines are fed into the interpreter, the fed program is evaluated both in part and in whole

Some Python Libraries

1. Pandas
2. Numpy
3. Pymysql
4. Scikit-learn

Pandas

- Pandas provide us with many Series and Data Frames. It allows you to easily organize, explore, represent, and manipulate data.
- Smart alignment and indexing featured in Pandas offer you a perfect organization and data labelling.
- Pandas has some special features that allow you to handle missing data or value with a proper measure.
- This package offers you such a clean code that even people with no or basic knowledge of programming can easily work with it.
- It provides a collection of built-in tools that allows you to both read and write data in different web services, data-structure, and databases as well.
- Pandas can support JSON, Excel, CSV, HDF5, and many other formats. In fact, you can merge different databases at a time with Pandas.

Numpy

- Arrays of Numpy offer modern mathematical implementations on huge amount of data. Numpy makes the execution of these projects much easier and hassle-free.
- Numpy provides masked arrays along with general array objects. It also comes with functionalities such as manipulation of logical shapes, discrete

Fourier transform, general linear algebra, and many more.

- While you change the shape of any N-dimensional arrays, Numpy will create new arrays for that and delete the old ones.
- This python package provides useful tools for integration. You can easily integrate Numpy with programming languages such as C, C++, and Fortran code.
- Numpy provides such functionalities that are comparable to MATLAB. They both allow users to get faster with operations.

Pymysql

- PyMySQL is a database connector for Python, libraries to enable Python programs to talk to a MySQL server.
- Access to the port settings through Python properties.
- PyMySQL is a pure Python MySQL driver, first written as a rough port of the MySQL-Python driver.
- PyMySQL meets all of criterion for a driver.
- It is fully open source, hosted on Github, released on Pypi, is actively maintained.
- It is written in pure Python so is eventlet-monkeypatch compatible, and is fully Python 3 compatible.

Scikit-Learn

- The random module is a simple and efficient tool for predictive data analysis.
- Accessible to everybody, and reusable in various contexts.
- The library is focused on modelling data. It is not focused on loading, manipulating and summarizing data. For these features, refer to NumPy and Pandas.
- The functionality that scikit-learn provides include:
- Regression, including Linear and Logistic Regression
- Classification, including K-Nearest Neighbours.
- Clustering, including K-Means and K-Means++
- Model selection

Python class and objects

These are the building blocks of OOP. Class creates a new object. This object can be anything, whether an abstract data concept or a model of a physical object, e.g., a chair. Each class has individual characteristics unique to that class, including variables and methods. Classes are very powerful and currently “the big thing” in most programming languages. Hence, there are several chapters dedicated to OOP later in the book.

The class is the most basic component of object-oriented programming. Previously, you learned how to use functions to make your program do something.

Now will move into the big, scary world of Object-Oriented Programming (OOP). To be honest, it took me several months to get a handle on objects.

When I first learned C and C++, I did great; functions just made sense for me.

Having messed around with BASIC in the early '90s, I realized functions were just like subroutines so there wasn't much new to learn.

However, when my C++ course started talking about objects, classes, and all the new features of OOP, my grades definitely suffered.

Once you learn OOP, you'll realize that it's actually a pretty powerful tool. Plus many Python libraries and APIs use classes, so you should at least be able to understand what the code is doing.

One thing to note about Python and OOP: it's not mandatory to use objects in your code in a way that works best; maybe you don't need to have a full-blown class with initialization code and methods to just return a calculation. With Python, you can get as technical as you want.

As you've already seen, Python can do just fine with functions. Unlike languages such as Java, you aren't tied down to a single way of doing things; you can mix functions and classes as necessary in the same program. This lets you build the code.

Objects are an encapsulation of variables and functions into a single entity. Objects get their variables and functions from classes. Classes are essentially a template to create your objects.

Here's a brief list of Python OOP ideas

- The class statement creates a class object and gives it a name. This creates a new namespace.
- Assignments within the class create class attributes. These attributes are accessed by qualifying the name using dot syntax: `ClassName.Attribute`.
- Class attributes export the state of an object and its associated behaviour. These attributes are shared by all instances of a class.
- Calling a class (just like a function) creates a new instance of the class.
- This is where the multiple copies part comes in.
- Each instance gets ("inherits") the default class attributes and gets its own namespace. This prevents instance objects from overlapping and confusing the program.
- Using the term `self` identifies a particular instance, allowing for per-instance attributes. This allows items such as variables to be associated with a particular instance.

Inheritance

First off, classes allow you to modify a program without really making changes to it.

To elaborate, by subclassing a class, you can change the behaviour of the program by simply adding new components to it rather than rewriting the existing components.

As we've seen, an instance of a class inherits the attributes of that class.

However, classes can also inherit attributes from other classes. Hence, a subclass inherits from a superclass allowing you to make a generic superclass that is specialized via subclasses.

The subclasses can override the logic in a superclass, allowing you to change the behaviour of your classes without changing the superclass at all.

Operator Overloads

Operator overloading simply means that objects that you create from classes can respond to actions (operations) that are already defined within Python, such as addition, slicing, printing, etc.

Even though these actions can be implemented via class methods, using overloading ties the behaviour closer to Python's object model and the object

interfaces are more consistent to Python's built-in objects, hence overloading is easier to learn and use.

User-made classes can override nearly all of Python's built-in operation methods.

Exceptions

I've talked about exceptions before but now I will talk about them in depth. Essentially, exceptions are events that modify program's flow, either intentionally or due to errors.

They are special events that can occur due to an error, e.g. trying to open a file that doesn't exist, or when the program reaches a marker, such as the completion of a loop.

Exceptions, by definition, don't occur very often; hence, they are the "exception to the rule" and a special class has been created for them.

Exceptions are everywhere in Python.

Virtually every module in the standard Python library uses them, and Python itself will raise them in a lot of different circumstances.

Here are just a few examples

- Accessing a non-existent dictionary key will raise a Key Error exception.
- Searching a list for a non-existent value will raise a Value Error exception
- Calling a non-existent method will raise an Attribute Error exception.
- Referencing a non-existent variable will raise a Name Error exception.
- Mixing data types without coercion will raise a Type Error exception.

One use of exceptions is to catch a fault and allow the program to continue working; we have seen this before when we talked about files.

This is the most common way to use exceptions. When programming with the Python command line interpreter, you don't need to worry about catching exceptions.

Your program is usually short enough to not be hurt too much if an exception occurs.

Plus, having the exception occur at the command line is a quick and easy way to tell if your code logic has a problem.

However, if the same error occurred in your real program, it will fail and stop working.

Exceptions can be created manually in the code by raising an exception.

It operates exactly as a system-caused exceptions, except that the programmer is doing it on purpose. This can be for a number of reasons. One of the benefits of using exceptions is that, by their nature, they don't put any overhead on the code processing.

Because exceptions aren't supposed to happen very often, they aren't processed until they occur.

Exceptions can be thought of as a special form of the if/elif statements. You can realistically do the same thing with if blocks as you can with exceptions.

However, as already mentioned, exceptions aren't processed until they occur; if blocks are processed all the time.

Proper use of exceptions can help the performance of your program.

The more infrequent the error might occur, the better off you are to use exceptions; using if blocks require Python to always test extra conditions before continuing.

Exceptions also make code management easier: if your programming logic is mixed in with error-handling if statements, it can be difficult to read, modify, and debug your program.

User-Defined Exceptions

I won't spend too much time talking about this, but Python does allow for a programmer to create his own exceptions.

You probably won't have to do this very often but it's nice to have the option when necessary.

However, before making your own exceptions, make sure there isn't one of the built-in exceptions that will work for you.

They have been "tested by fire" over the years and not only work effectively, they have been optimized for performance and are bug-free.

Making your own exceptions involves object-oriented programming, which will be covered in the next chapter

To make a custom exception, the programmer determines which base exception to use as the class to inherit from, e.g., making an exception for negative numbers or one for imaginary numbers would probably fall under the Arithmetic Error exception class.

To make a custom exception, simply inherit the base exception and define what it will do.

Python modules

Python allows us to store our code in files (also called modules). This is very useful for more serious programming, where we do not want to retype a long function definition from the very beginning just to change one mistake. In doing this, we are essentially defining our own modules, just like the modules defined already in the Python library.

To support this, Python has a way to put definitions in a file and use them in a script or in an interactive instance of the interpreter. Such a file is called a module; definitions from a module can be imported into other modules or into the main module.

Testing code

As indicated above, code is usually developed in a file using an editor. To test the code, import it into a Python session and try to run it.

Usually there is an error, so you go back to the file, make a correction, and test again. This process is repeated until you are satisfied that the code works.

The entire process is known as the development cycle.

There are two types of errors that you will encounter. Syntax errors occur when the form of some command is invalid.

This happens when you make typing errors such as misspellings, or call something by the wrong name, and for many other reasons. Python will always give an error message for a syntax error.

Functions in Python

It is possible, and very useful, to define our own functions in Python. Generally speaking, if you need to do a calculation only once, then use the interpreter. But when you or others have need to perform a certain type of calculation many times, then define a function.

You use functions in programming to bundle a set of instructions that you want to use repeatedly or that, because of their complexity, are better self-contained in a sub- program and called when needed. That means that a function is a piece of code written to carry out a specified task.

To carry out that specific task, the function might or might not need multiple inputs. When the task is carried out, the function can or cannot return one or more values.

There are three types of functions in python:

`help()`, `min()`, `print()`.

Python Namespace

Generally speaking, a namespace (sometimes also called a context) is a naming system for making names unique to avoid ambiguity. Everybody knows a name spacing system from daily life, i.e., the naming of people in first name and family name (surname).

An example is a network: each network device (workstation, server, printer,) needs a unique name and address. Yet another example is the directory structure of file systems.

The same file name can be used in different directories, the files can be uniquely accessed via the pathnames. Many programming languages use namespaces or contexts for identifiers. An identifier defined in a namespace is associated with that namespace.

This way, the same identifier can be independently defined in multiple namespaces. (Like the same file names in different directories)

Programming languages, which support namespaces, may have different rules that determine to which namespace an identifier belongs.

Namespaces in Python are implemented as Python dictionaries, this means it is a mapping from names (keys) to objects (values). The user doesn't have to know this to write a Python program and when using namespaces.

Some namespaces in Python

- **global names** of a module
- **local names** in a function or method invocation
- **built-in names**: this namespace contains built-in functions (e.g. `abs()`, `cmp()`, ...) and built-in exception names.

Garbage Collection

Garbage Collector exposes the underlying memory management mechanism of Python, the automatic garbage collector.

The module includes functions for controlling how the collector operates and to examine the objects known to the system, either pending collection or stuck in reference cycles and unable to be freed.

Python XML Parser

XML is a portable, open-source language that allows programmers to develop applications that can be read by other applications, regardless of operating system and/or developmental language.

What is XML? The Extensible Markup Language XML is a markup language much like HTML or SGML.

This is recommended by the World Wide Web Consortium and available as an open standard.

XML is extremely useful for keeping track of small to medium amounts of data without requiring a SQL-based backbone.

XML Parser Architectures and APIs The Python standard library provides a minimal but useful set of interfaces to work with XML.

The two most basic and broadly used APIs to XML data are the SAX and DOM interfaces.

Simple API for XML SAX: Here, you register callbacks for events of interest and then let the parser proceed through the document. This is useful when your documents are large or you have memory limitations, it parses the file as it reads it from disk and the entire file is never stored in memory.

Document Object Model DOM API: This is a World Wide Web Consortium recommendation wherein the entire file is read into memory and stored in a hierarchical tree-based form to represent all the features of an XML document.

SAX obviously cannot process information as fast as DOM can when working with large files. On the other hand, using DOM exclusively can really kill your resources, especially if used on a lot of small files.

SAX is read-only, while DOM allows changes to the XML file. Since these two different APIs literally complement each other, there is no reason why you cannot use them both for large projects.

Python Web Frameworks

A web framework is a code library that makes a developer's life easier when building reliable, scalable and maintainable web applications.

Why are web frameworks useful?

Web frameworks encapsulate what developers have learned over the past twenty years while programming sites and applications for the web. Frameworks make it easier to reuse code for common HTTP operations and to structure projects so other developers with knowledge of the framework can quickly build and maintain the application.

Common web framework functionality

Frameworks provide functionality in their code or through extensions to perform common operations required to run web applications. These common operations include:

1. URL routing
2. HTML, XML, JSON, and other output format templating
3. Database manipulation
4. Security against Cross-site request forgery (CSRF) and other attacks
5. Session storage and retrieval

Not all web frameworks include code for all of the above functionality. Frameworks fall on the spectrum from executing a single use case to providing every known web framework feature to every developer. Some frameworks take the "batteries-included" approach where everything possible comes bundled with the framework while others have a minimal core package.

Comparing web frameworks

There is also a repository called [compare-python-web-frameworks](#) where the same web application is being coded with varying Python web frameworks, templating engines and object.

Web framework resources

- When you are learning how to use one or more web frameworks it's helpful to have an idea of what the code under the covers is doing.
- Frameworks is a really well-done short video that explains how to choose between web frameworks. The author has some particular opinions about what should be in a framework. For the most part I agree although I've found

sessions and database ORMs to be a helpful part of a framework when done well.

- what is a web framework? is an in-depth explanation of what web frameworks are and their relation to web servers.
- Django vs Flask vs Pyramid: Choosing a Python web framework contains background information and code comparisons for similar web applications built in these three big Python frameworks.
- This fascinating blog post takes a look at the code complexity of several Python web frameworks by providing visualizations based on their code bases.
- Python's web frameworks benchmarks is a test of the responsiveness of a framework with encoding an object to JSON and returning it as a response as well as retrieving data from the database and rendering it in a template. There were no conclusive results but the output is fun to read about nonetheless.
- What web frameworks do you use and why are they awesome? is a language agnostic Reddit discussion on web frameworks. It's interesting to see what programmers in other languages like and dislike about their suite of web frameworks compared to the main Python frameworks.
- This user-voted question & answer site asked "What are the best general purpose Python web frameworks usable in production?". The votes aren't as important as the list of the many frameworks that are available to Python developers.

Web frameworks learning checklist

1. Choose a major Python web framework (Django or Flask are recommended) and stick with it. When you're just starting it's best to learn one framework first instead of bouncing around trying to understand every framework.
2. Work through a detailed tutorial found within the resources links on the framework's page.
3. Study open-source examples built with your framework of choice so you can take parts of those projects and reuse the code in your application.
4. Build the first simple iteration of your web application then go to the deployment section to make it accessible on the web.

CHAPTER 7

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the

Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.1 TYPES OF TESTS

7.1.1 Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.1.2 Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.1.3 Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

7.2 SYSTEM TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.2.1 White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

7.2.2 Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested.

Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

7.2.3 Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

7.2.4 Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level– interact without error.

Test Results- All the test cases mentioned above passed successfully. No defects encountered.

7.2.5 Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results- All the test cases mentioned above passed successfully. No defects encountered.

CHAPTER 8

EXECUTION AND RESULTS

8.1 Execution

To execute the code in PyCharm:

```
>python app.py
```

Run Time Environment

```
D:\project\code>python app.py
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\tejde\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data] C:\Users\tejde\AppData\Roaming\nltk_data...
[nltk_data] Package punkt is already up-to-date!
* Serving Flask app 'app' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Restarting with stat
[nltk_data] Downloading package stopwords to
[nltk_data] C:\Users\tejde\AppData\Roaming\nltk_data...
[nltk_data] Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to
[nltk_data] C:\Users\tejde\AppData\Roaming\nltk_data...
[nltk_data] Package punkt is already up-to-date!
* Debugger is active!
* Debugger PIN: 271-962-697
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

Fig.8.1: Run Time Environment

8.2 Results

1. Home Page



Fig.8.2.1: Home Page

The above home page shows the brief information of algorithms namely, Naïve Bayes and SVM.

2. About Page



Fig.8.2.2: About Page

The about page gives us the information of classification of the spam message, it's importance of spam classification and it's necessity.

3. Upload Page

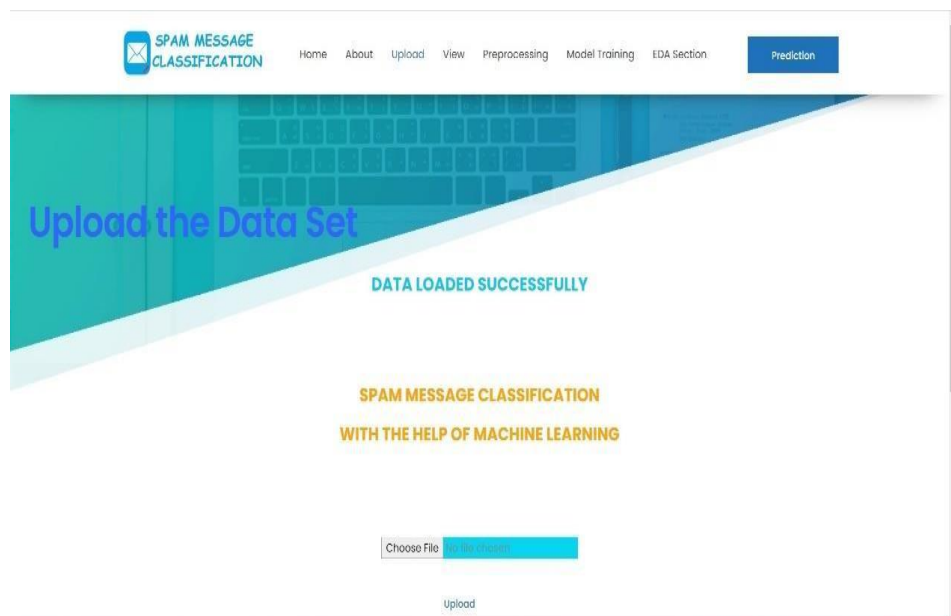
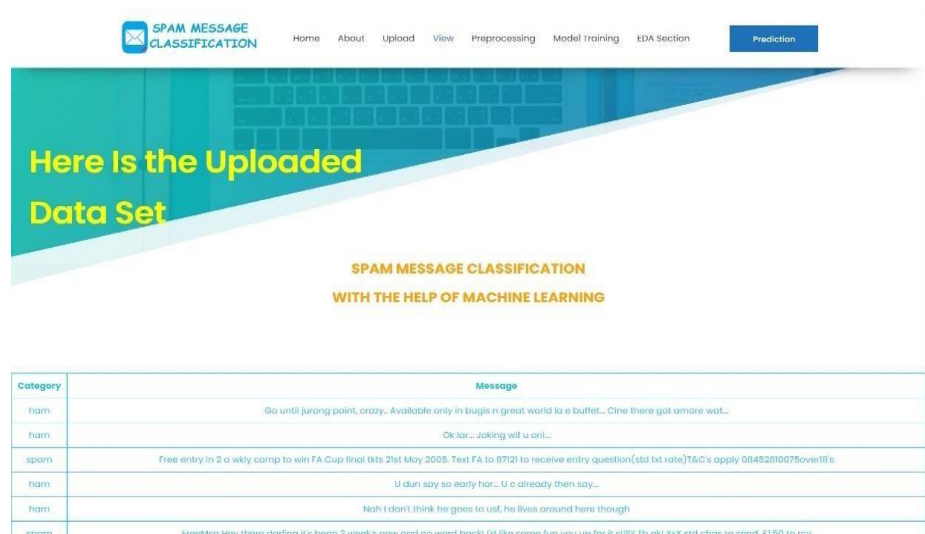


Fig.8.2.3: Upload Page

In this page we are going to upload the dataset which is related to our project. After uploading the data set it will give us the message as the data loaded successfully.

4. View Page

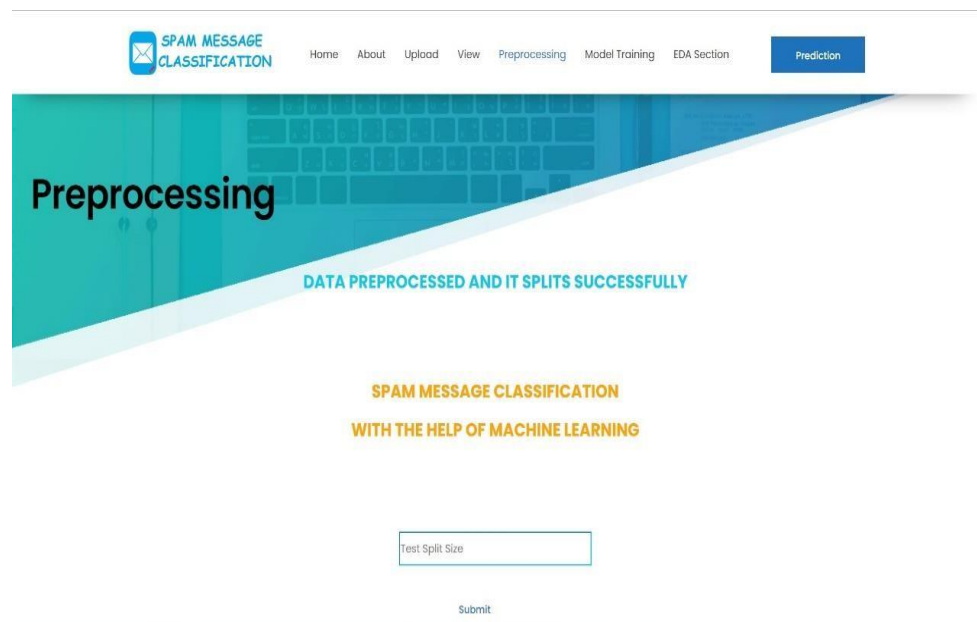


Category	Message
ham	Go until jurong point, crazy.. Available only in bugis n great world to e buffet... Cine there got amore wat...
ham	Ok lar... Joking wif u oni...
spam	Free entry in 2 a wkly comp to win FA Cup final tkts 21st May 2005. Text FA to 87121 to receive entry question(std txt rate)T&C's apply GB45210075over18's
ham	U dun say so early hor... U c already then say...
ham	Nah i don't think he goes to usf, he lives around here though
spam	Freebies New theatre clublines it's been 5 week's now and no word back! I'd like some fun you up for it still? It's ok! xxx std chgs to send. £1.50 to rec

Fig.8.2.4: View Page

This view page gives us the information about which dataset we have uploaded and it also displays about the data in the dataset.

5. Pre-processing Page



Preprocessing

DATA PREPROCESSED AND IT SPLITS SUCCESSFULLY

SPAM MESSAGE CLASSIFICATION
WITH THE HELP OF MACHINE LEARNING

Test Split Size

Submit

Fig.8.2.5: Pre-processing Page

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When we give the testing value and submit it will display the message as data preprocessed and it splits successfully.

6. Naive Bayes Accuracy

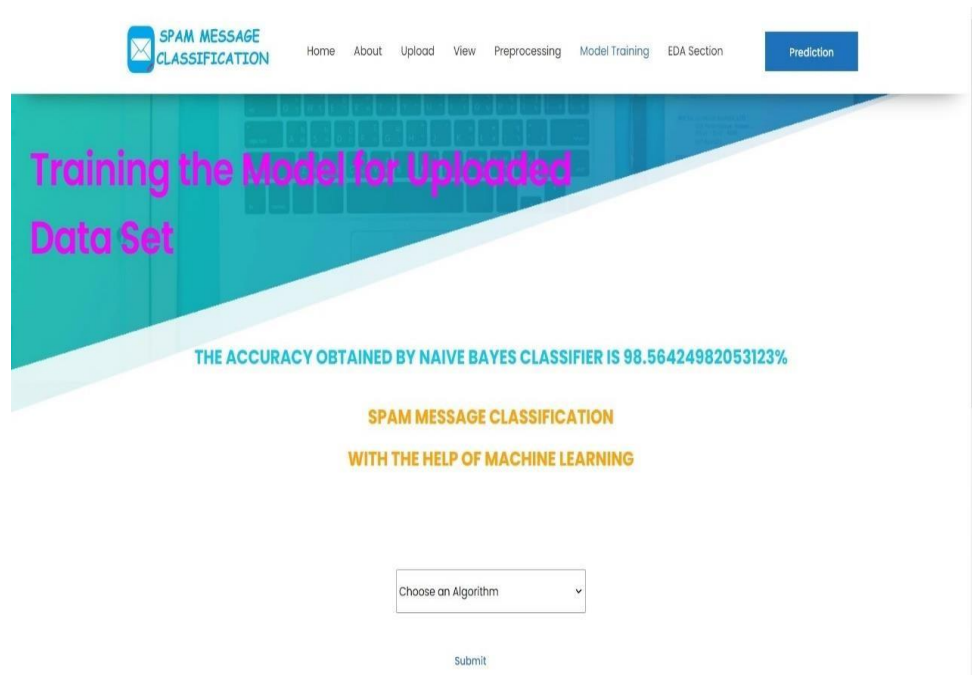


Fig.8.2.6: Naive Bayes Accuracy

This page shows the accuracy rate of Naïve Bayes classifier for 25% testing value is: 98.5642%

7. Support Vector Classifier Accuracy

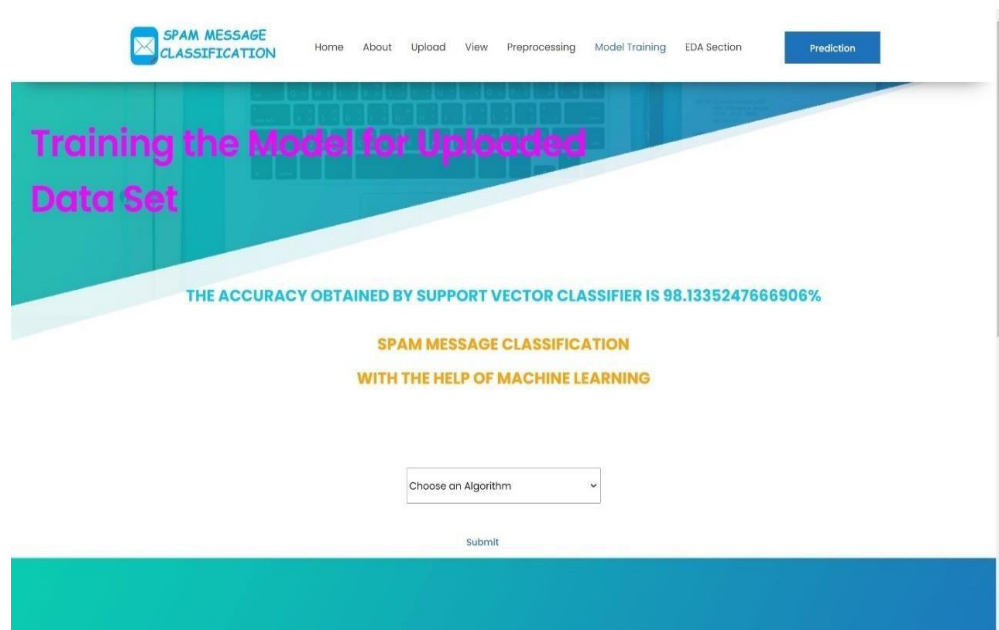


Fig.8.2.7: Support Vector Classifier Accuracy

This page shows the accuracy rate of SVM classifier for 25% testing value is: 98.1335%

8. Comparative Chart

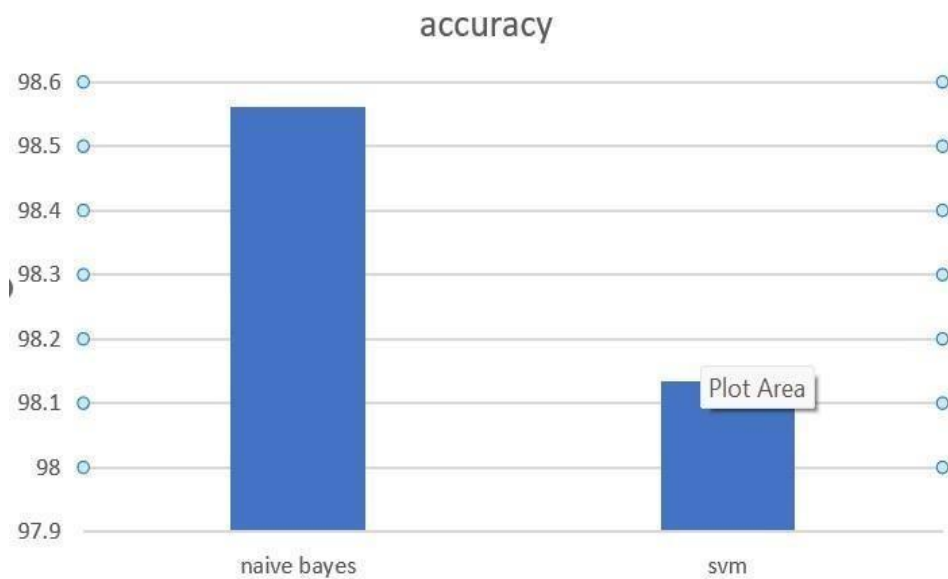


Fig.8.2.8: Comparative Chart

This chart shows the comparative analysis of Naive Bayes and SVM.

9. Prediction Page

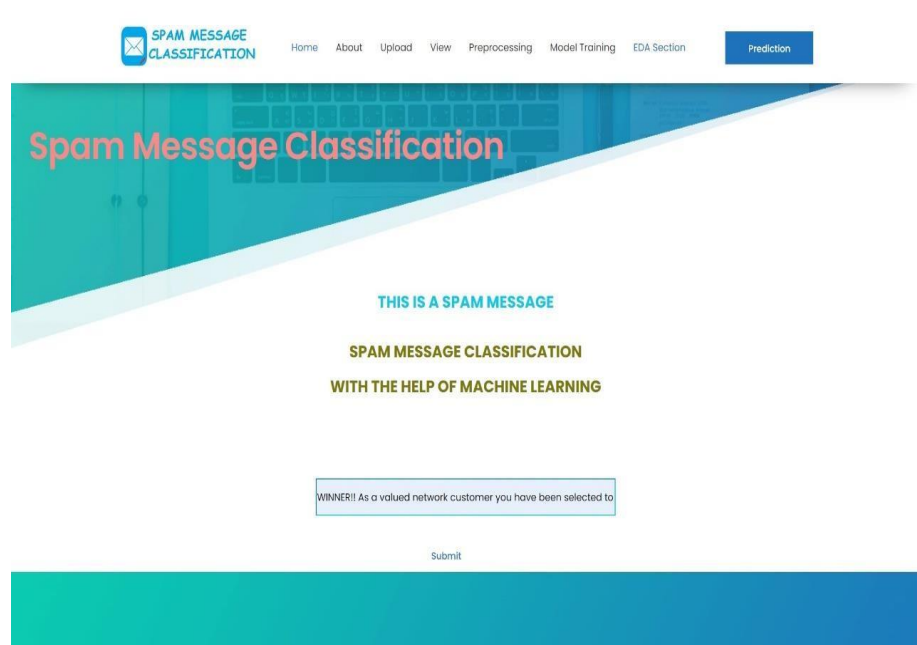


Fig.8.2.9: Spam Prediction

By entering the message and clicking submit button its show the whether the message is spam or not. Here it shows the spam message.

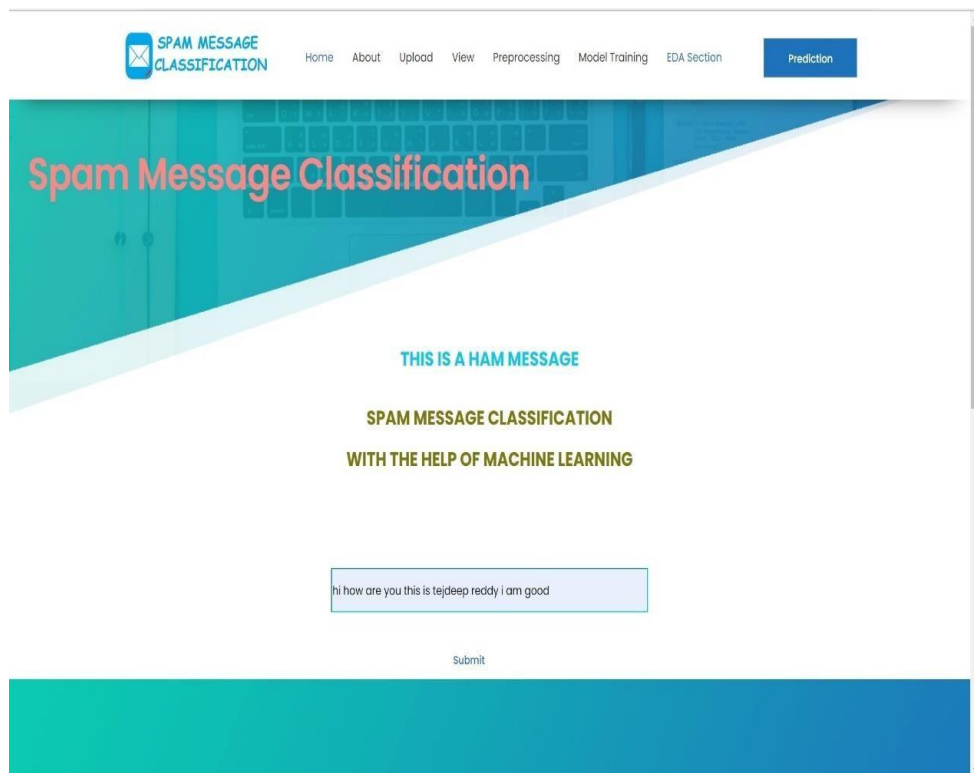


Fig.8.2.10: Ham Prediction

By entering the message and clicking submit button its show the whether the message is spam or not. Here it shows the Ham message

CONCLUSION

In this paper, we proposed Spam Message Classification using two algorithms namely, Naïve Bayes and Support Vector Machine. By using this we can separate the spam and ham messages in the users data. These both algorithms given the accuracy of 98% with some variations in their decimal points. So, here both the algorithms shows that we can use any of these algorithms to get the result.

REFERENCES

- [1]. J. Han, M. Kamber. Data Mining Concepts and Techniques. by Elsevier inc., Ed: 2nd, 2006
- [2]. A. Tiago, Almeida , José María GómezAkebo Yamakami. Contributions to the Study of SMS Spam Filtering. University of Campinas, Sao Paulo, Brazil.
- [3]. M. Bilal Junaid, Muddassar Farooq. Using Evolutionary Learning Classifiers To Do Mobile Spam (SMS) Filtering. National University of Computer & Emerging Sciences (NUCES) Islamabad, Pakistan.
- [4]. Inwhae Joe and Hyetaek Shim, "An SMS Spam Filtering System Using Support Vector Machine," Division of Computer Science and Engineering, Hanyang University, Seoul, 133-791 South Korea.
- [5]. Xu, Qian, Evan Wei Xiang, Qiang Yang, Jiachun Du, and Jieping Zhong. "Sms spam detection using noncontent features." IEEE Intelligent Systems 27, no. 6 (2012): 44-51. Yadav, K., Kumaraguru, P., Goyal, A., Gupta, A., and Naik, V. "SMSAssassin: Crowdsourcing driven mobile-based system for SMS spam filtering," Proceedings of the 12th Workshop on Mobile Computing Systems and Applications, ACM, 2011, pp. 1-6.
- [6]. Duan, L., Li, N., & Huang, L. (2009). "A new spam short message classification" 2009 First International Workshop on Education Technology and Computer Science, 168-171.
- [7]. Weka The University of Waikato, Weka 3: Data Mining Software in Java, viewed on 2011 September 14.
- [8]. Mccallum, A., & Nigam, K. (1998). "A comparison of event models for naive Bayes text classification". AAAI-98 Workshop on 'Learning for Text Categorization'
- [9]. Bayesian Network Classifiers in Weka, viewed on 2011 September 14.
- [10]. Llorca, Xavier, and Josep M. Garrell (2001) Evolution of decision trees, edn., Forth Catalan Conference on Artificial Intelligence (CCIA2001).
- [11]. B. G. Becker. Visualizing Decision Table Classifiers. Pages 102- 105, IEEE (1998).