

HEALTH DOCUMENT CLASSIFICATION USING MACHINE LEARNING

A PROJECT REPORT

submitted

in the partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

by

G Sai Koushik - 17B81A05H7

M Prashanth Reddy-17B81A05E7

P Sai Krishna - 17B81A05H8

Under the guidance of

Mrs. Gundeti Sandhya



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CVR COLLEGE OF ENGINEERING

(An Autonomous institution, NBA, NAAC Accredited and Affiliated to JNTUH, Hyderabad)

Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),

Rangareddy (D), Telangana- 501 510

May 2021

CVR COLLEGE OF ENGINEERING

(An UGC Autonomous Institute, Accredited by NAAC with 'A' Grade)

Department of Computer Science and Engineering



CERTIFICATE

This is to certify that the project entitled “Health Document Classification using Machine Learning” being submitted by G Sai Koushik(17B81A05H7),M Prashanth Reddy(17B81A05E7),P Sai Krishna (17B81A05H8) in partial fulfillment for the award of Bachelor of Technology in Computer Science and Engineering to the CVR College of Engineering , is a record of bonafide work carried out by them under my guidance and supervision during the year 2020-2021.

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

Signature of the project guide,

Mrs. Gundeti Sandhya

Signature of the HOD

Dr. A. Vani Vathsala

External Examiner

ACKNOWLEDGEMENT

It is my proud privilege and duty to acknowledge the kind of help and guidance received from several people in presentation of the seminar and preparation of this report. It would not have been possible to present the seminar and report in this form without their valuable help, cooperation and guidance.

I am extremely grateful to, **Dr A VANI VATHSALA**, Professor and Head of Department of Computer Science and Engineering and to our guide **Mrs. GUNDETI SANDHYA** for the guidance and encouragement and for providing me with best facilities and atmosphere for the creative work. I would like to thank my seminar guides, Department of Computer Science and Engineering, for extending his support and providing the necessary facilities.

I express my thanks to the Principal of **CVR COLLEGE OF ENGINEERING**, for the valuable guidance, care and timely support throughout the seminar work.

I would like to thank all my faculty members without whom this seminar report would have been a distinct reality. I also extend my heartfelt my family and friends for their encouragement, which helped me to keep my spirit alive and to complete this work successfully.

Thanking you,

G Sai Koushik (17B81A05H8)

M Prashanth Reddy (17B81A05E7)

P Sai Krishna (17B81A05H8)

LIST OF FIGURES

Figure	Title	Page
2.2	Document Classification Process	17
3.3	Use case Diagram	23
3.4	Sequence Diagram	24
3.5	Class Diagram	25
3.7	System Flow Chart	26
3.8	System Architecture	27
3.9	Activity Diagram	27

ABBREVIATIONS

HTML – Hyper Text Markup Language

ML – Machine Learning

CSS – Cascading Style Sheets

JS – JavaScript

NLP – Natural Language Processing

Table of Contents			
			Page No.
		List of Figures	
		Abbreviations	
		Abstract	7
1		Introduction	
	1.0	Introduction	8
	1.1	Motivation & Background of Study	8
	1.2	Problem Statement	9
	1.3	Aim and Project Objectives	9
	1.4	Scope of Study	9
	1.5	Significance of Study	10
	1.6	Definition of Terms	10
	1.7	Organization of work	12
2		Literature Survey	
	2.0	Document Classification	13
	2.1	Text Categorization	13
	2.2	Taxonomy of Text Classification Process	14
	2.2.1	Tokenization	14
	2.2.2	Stemming	15
	2.2.3	Stop Word Removal	15
	2.2.4	Vector Representation of the Documents	15
	2.2.5	Feature Selection and Process	16
	2.3	Assortment of Machine Learning Algorithms	17
	2.4	Review and Related Work	19
3		System Analysis and Design	
	3.1	Analysis of Existing System	21
	3.2	Analysis of Proposed System	21
	3.2.1	Requirements of System	21
	3.3	Training a Model	22
	3.4	Classifying the Document	22
	3.5	Use Case Diagram	23
	3.6	Sequence Diagram	24
	3.7	Class Diagram	25
	3.8	System Flow Chart Diagram	26

	3.9	System Architecture	27
4		Implementation & Testing	
	4.1	System Requirements	28
	4.1.2	Software Specifications	28
	4.1.3	Hardware Specifications	29
	4.2	System Sample output	29
	4.2.1	Screenshots	30
	4.3	Algorithm Description	35
5		Summary & Conclusion	39
6		References	41

ABSTRACT

Due to the massive increase in medical documents every day (including books, journals, blogs, articles, doctors' instructions and prescriptions, emails from patients, etc.), it is becoming very challenging to handle and to categorize them manually. One of the most challenging projects in information systems is extracting information from unstructured texts, including medical document classification. The discovery of knowledge from medical datasets is important in order to make effective medical diagnosis. Developing a classification algorithm that classifies a medical document by analyzing its content and categorizing it under predefined topics is the primary aim of this research. In this project work we were able to succeed in applying Natural Language Processing which is a branch of Machine Learning to Classifying Health related documents. We made use of the Open NLP Application Programming Interface which is an API for training a model and classifying the documents. We make use of Materialize which is a HTML5, CSS and JavaScript framework for building the user interface. The software is also built using the Model-View-Controller (MVC) architecture. The algorithm classified the articles correctly under the actual subject headings and got the total subject headings correct. This holds promising solutions for the global health arena to index and classify medical documents expeditiously.

CHAPTER ONE

1.0 INTRODUCTION

This chapter introduces the topic of the project work A System for Health Document Classification Using Machine Learning. In this chapter, we will consider the background of the study, statement of the problem, aims and objectives, methodology used to design the system, scope of the study, its significance, definition of terms, and we conclude with the project layout or organization of the project work.

1.1 MOTIVATION & BACKGROUND OF THE STUDY

Contemporarily, most hospitals, medical laboratories and other health facilities make use of some kind of information system. These could be either a hospital management system or a pharmacy management system. Among other functions that these systems provide, they are mainly used in collecting patient records. These information systems stores patient records in digital format. Numerous patient data are being recorded on a daily basis which forms a large data set popularly referred to as “Big Data”.

Every day physicians and other health workers are required to work with this “Big Data” in other to provide solution. Some of the everyday tasks include information retrieval and data mining. Retrieving information from big data can be very laborious and time consuming. This has given rise to the study of text or document classification in other to aid the process of retrieving information from big data. Today, text classification is a necessity due to the very large amount of text documents that we have to deal with daily.

Document classification is the task of grouping documents into categories based upon their content. Document classification is a significant learning problem that is at the core of many information management and retrieval tasks. Document classification performs an essential role in various applications that deals with organizing, classifying, searching and concisely representing a significant amount of information. Document classification is a longstanding problem in information retrieval which has been well studied (Russell, 2018).

Usually, machine learning, statistical pattern recognition, or neural network approaches are used to construct classifiers automatically. Machine learning approaches to classification suggest the automatic construction of classifiers using induction over pre-classified sample documents. In this project work we will employ machine learning in classifying health documents.

1.2 STATEMENT OF THE PROBLEM

With the explosion of information filled by the growth of the World Wide Web it is no longer feasible for a human observer to understand all the data coming in or even classify it into categories. Also in the health sector, numerous patient records are being collected everyday and are used for analysis. How do we efficiently classify or categorize these health documents to complement easy retrieval.

1.3 AIM AND OBJECTIVES OF THE STUDY

The aim of this project is to develop A System for Health Document Classification Using Machine Learning.

Other objectives include:

1. Study the various machine learning classification algorithm.
2. Implement Machine Learning classification algorithm

1.4 SCOPE OF THE STUDY

As stated earlier, statistical pattern recognition, or neural network are used in classifying documents, this project work will concentrate on using machine learning algorithm to classify document.

1.5 SIGNIFICANCE OF THE STUDY

The software delivered from this project work will greatly reduce the time used by doctors, physicians and other health workers in searching and retrieving documents.

Other importance of this project work includes:

1. Helps students and other interested individuals that want to develop a similar application.
2. It will serve as source of materials for those interested in investigating the processes involved in developing a document classification system using machine learning.
3. It will serve as source of materials for students who are interested in studying machine learning.

1.6 DEFINITION OF TERMS

Document Classification: is the task of grouping documents into categories based upon their content.

Health Document: A health certificate is written by a doctor and displays the official results of a physical examination.

Machine Learning: the study and construction of algorithms that can learn from and make predictions on data.

Flask: Flask is an API of Python that allows us to build up web-applications. Flask's framework is more explicit than Django's framework and is also easier to learn because it has less base code to implement a simple web-Application. A Web-Application Framework or Web Framework is the collection of modules and libraries that helps the developer to write applications without writing the low-level codes such as protocols, thread management, etc. Flask is based on WSGI(Web Server Gateway Interface) toolkit and Jinja2 template engine.

Python Pickle: Python pickle module is used for serializing and de-serializing python object structures. The process to converts any kind of python objects (list, dict, etc.) into byte streams (0s and 1s) is called pickling or serialization or flattening or marshallng. We can converts the byte stream (generated through pickling) back into python objects by a process called as unpickling.

Jinja: [Jinja2](#) is a template engine written in pure Python. It provides a [Django](#)-inspired non-XML syntax but supports inline expressions and an optional [sandboxed](#) environment. It is small but fast, apart from being an easy-to-use standalone template engine.

HTML: (Hypertext Markup Language) is the code that is **used** to structure a web page and its content. For example, content could be structured within a set of paragraphs, a list of bulleted points, or using images and data tables.

CSS: It is the acronym of “Cascading Style Sheets”. **CSS** is a computer language for laying out and structuring web pages (HTML or XML). This language contains coding elements and is composed of these “cascading style sheets” which are equally called **CSS** files.

Java script: JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. Where HTML and CSS are languages that give structure and style to web pages, JavaScript gives web pages interactive elements that engage a user

Heroku: **Heroku** is a container-based cloud Platform as a Service (PaaS). Developers **use Heroku** to deploy, manage, and scale modern apps. Our platform is elegant, flexible, and easy to **use**, offering developers the simplest path to getting their apps to market.

Bootstrap is a free and opensource front end development framework for the creation of websites and web apps. In computers, the word **bootstrap** means to boot: to load a program into a computer using a much smaller initial program to load in the desired program (which is usually an operating system).

1.7 ORGANIZATION OF WORK

Chapter one introduces the background of the project with the statement of the problems, objectives of the project, its significance, scope, and constraints are pointed out.

Chapter two reviews literatures on machine learning, document classification and the review of related literature.

Chapter three discusses system Investigation and Analysis. It deals with detailed investigation and analysis of the existing system and problem identification. It also proposed for the new system.

Chapter four covers the system design and implementation.

Chapter five was the summary and conclusion of the project.

CHAPTER TWO

LITERATURE REVIEW

2.0 DOCUMENT CLASSIFICATION

Classification can be divided in two principal phases. The first phase is document representation, and the second phase is classification. The standard document representation used in text classification is the vector space model. The difference of classification systems is in document representation models. The more relevant the representation is, the more relevant the classification will be. The second phase includes learning from training corpus, making a model for classes and classifying the new documents according to the model.

2.1 TEXT CATEGORIZATION

Text categorization, the activity of labeling natural language texts with thematic categories from a set arranged in advance has accumulated an important status in the information systems field, due to because of augmentation of availability of documents in digital form and the confirms need to access them in easy ways.. Currently text categorization is applied in many contexts, ranging from document indexing depending on a managing vocabulary, to document filtering, automated metadata creation, vagueness of word sense, population of and in general any application needs document organization or chosen and adaptive document execution. These days text categorization is a discipline at the crossroads of ML and IR, and it claims a number of characteristics with other tasks like information/ knowledge pulling from texts and text mining (PAZIENZA, 1997). “Text mining” is mostly used to represent all the tasks that, by analyzing large quantities of text and identifying usage patterns, try to extract probably helpful (although only probably correct) information. Concentrating on the above opinion, text categorization is an illustration of text mining which includes:

1. the automatic assignment of documents to a predetermined set of categories,
2. the automatic reorganization of such a set of categories, or
3. the automatic identification

The text classification is a crucial part of information management process. As net resources constantly grow, increasing the effectiveness of text classifiers is necessary. Document retrieval, its categorization, routing and aforementioned information filtering is often based on the text categorization (Hull, 1996).

2.2 TAXONOMY OF TEXT CLASSIFICATION PROCESS

The task of building a classifier for documents does not vary from other tasks of Machine Learning. The main point is the representation of a document (Leopold, 2002).

One special certainty of the text categorization problem is that the number of features (unique words or phrases) reaches orders of tens of thousands flexibly. This develops big hindrances in applying many sophisticated learning algorithms to the text categorization, so dimension reduction methods are used which can be used either in choosing a subset of the original features (Brank, 2002), or transforming the features into new ones, that is, adding new features.

2.2.1 TOKENIZATION

The process of breaking a stream of text up into tokens that is words, phrases, symbols, or other meaningful elements is called Tokenization where the list of tokens is input to the next processing of text classification. Generally, tokenization occurs at the word level.

Nevertheless, it is not easy to define the meaning of the “word”. Where a tokenize process responds on simple heuristics, for instance:

All contiguous strings of alphabetic characters are part of one token; similarly with numbers. Tokens are divided by whitespace characters, like a space or line break, or by punctuation characters. Punctuation and whitespace may or may not be added in the resulting list of tokens. In languages like English (and most programming languages) words are separated by whitespace, this approach is straightforward. Still, tokenization is tough for languages with no word boundaries like Chinese. Simple white spaced limited tokenization also shows toughness in word collocations like New York which must be considered as single token. Some ways to mention this problem

are by improving more complex heuristics, querying a table of common collocations, or fitting the tokens to a language model that identifies collocations in a next processing.

2.2.2 STEMMING

In linguistic morphology and information collection, stemming is the process for decreasing deviated (or sometimes derived) words to their stem, original form. The stem need not be identical to the morphological root of the word; it is usually enough if it is concern words map of similar stem, even if this stem is not a valid root. In computer science algorithms for stemming have been studied since 1968. Many search engines consider words with the similar stem as synonyms as a kind of query broadening, a process called conflation.

2.2.3 STOP WORD REMOVAL

Typically in computing, stop words are filtered out prior to the processing of natural language data (text) which is managed by man but not a machine. A prepared list of stop words do not exist which can be used by every tool. Though any stop word list is used by any tool in order to support the phrase search the list is ignored. Any group of words can be selected as the stop words for a particular cause. For a few search machines, these is a list of common words, short function words, like the, is, at, which and on that create problems in performing text mining phrases that consist them. Therefore it is needed to eliminate stop words contains lexical words, like “want” from phrases to raise performance.

2.2.4 VECTOR REPRESENTATION OF THE DOCUMENTS

Vector denotation of the documents is an algebraic model for symbolizing text documents (and any objects, in general) as vectors of identifiers, like, for example, index terms which will be utilized in information filtering, information retrieval, indexing and relevancy rankings where its primary use is in the SMART Information

Retrieval System.

A sequence of words is called a document (Leopold, 2002). Thus every document is generally denoted by an array of words. The group of all the words of a training group is called vocabulary, or feature set. Thus a document can be produced by a binary vector, assigning the value 1 if the document includes the feature-word or 0 if there is no word in the document.

2.2.5 FEATURE SELECTION AND TRANSFORMATION

The main objective of feature-selection methods is to decrease the dimensionality of the dataset by eliminating features that are not related for the classification (Forman, 2003). The transformation procedure is explained for presenting a number of benefits, involving tiny dataset size, tiny computational needs for the text categorization algorithms (especially those that do not scale well with the feature set size) and comfortable shrinking of the search space. The goal is to reduce the curse of dimensionality to yield developed classification perfection. The other advantage of feature selection is its quality to decrease over fitting, i. e. the phenomenon by which a classifier is tuned also to the contingent characteristics of the training data rather than the constitutive characteristics of the categories, and therefore, to augment generalization.

Feature Transformation differs considerably from Feature Selection approaches, but like them its aim is to decrease the feature set volume. The approach does not weight terms in order to neglect the lower weighted but compacts the vocabulary based on feature concurrencies.

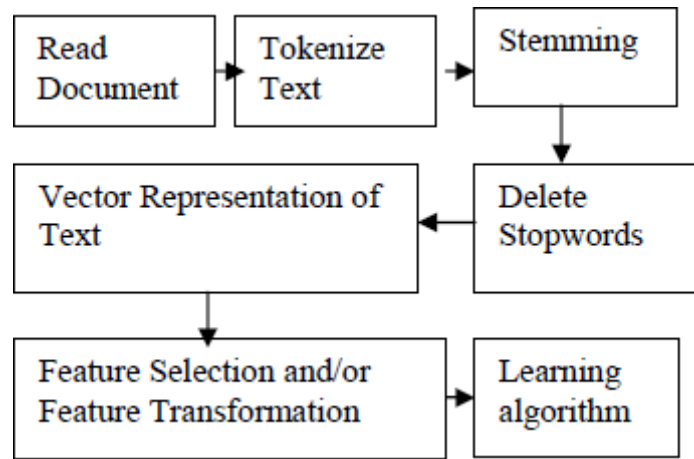


Figure 2 Document classification process

2.3 ASSORTMENT OF MACHINE LEARNING ALGORITHMS FOR TEXT CLASSIFICATION

After feature opting and transformation the documents can be flexibly denoted in a form that can be utilized by a ML algorithm. Most of the text classifiers adduced in the literature utilizing machine learning techniques, probabilistic models, etc. They regularly vary in the approach taken are decision trees, naïve Bayes, rule induction, neural networks, nearest neighbors, and lately, support vector machines. Though most of the approaches adduced, automated text classification is however a major area of research first due to the effectiveness of present automated text classifiers is not errorless and nevertheless require development.

Naive Bayes is regularly utilized in text classification applications and experiments due to its easy and effectiveness (Kim, 2002). Nevertheless, its performance is reduced due to it does not model text.

Schneider addressed the problems and display that they can be resolved by a few plain corrections. Klopotek and Woch presented results of empirical evaluation of a Bayesian multinet classifier depending on a novel method of learning very large tree-like Bayesian networks (Klopotek, 2003). The study advices that tree-like Bayesian networks are able to deal a text classification task in one hundred thousand variables with sufficient speed and

accuracy.

When Support vector machines (SVM), are applied to text classification supplying excellent precision, but less recollection. Customizing SVMs means to develop recollect which helps in adjusting the origin associated with an SVM. Shanahan and Roma explained an automatic process for adjusting the thresholds of generic SVM (Shanahan, 2003),for improved results. Johnson et al. explained a fast decision tree construction algorithm that receives benefits of the sparse text data, and a rule simplification method that translates the decision tree into a logically equivalent rule set.

Lim introduced a method which raises performance of KNN based text classification by utilizing calculated parameters. Some variants of the KNN method with various decision functions, k values, and feature sets are also introduced and evaluated to discover enough parameters.

For immediate document classification, Corner classification (CC) network, feed forward neural network is used. A training algorithm, Text CC is introduced in. The complexity of of text classification tasks generally varies. As the number of different classes augments as of complexity and hence the training set size is required. In multi-class text classification task, unavoidable some classes are a bit harder than others to classify. Reasons for this are: very few positive training examples for the class, and lack of good forecasting features for that class.

When training a binary classifier per category in text categorization, we use all the documents in the training corpus that has the category as related training data and all the documents in the training corpus that are of the other categories are non related training data. It is a regular case that there is an overwhelming number of non related training documents specially when there is high number of categories with every allotted to a tiny documents, which is an “imbalanced data problem”. This problem gives a certain risk to classification algorithms, which can accomplish perfection by simply classifying every example as negative. To resolve this problem, cost sensitive learning is required.

2.4 REVIEW OF RELATED WORK

LI et al, investigate four different methods for document classification: the naive Bayes classifier, the nearest neighbor classifier, decision trees and a subspace method. These were applied to seven-class Yahoo news groups (business, entertainment, health, international, politics, sports and technology) individually and in combination. They studied three classifier combination approaches: simple voting, dynamic classifier selection and adaptive classifier combination. Their experimental results indicate that the naive Bayes classifier and the subspace method outperform the other two classifiers on our data sets. Combinations of multiple classifiers did not always improve the classification accuracy compared to the best individual classifier. Among the three different combination approaches, the adaptive classifier combination method introduced here performed the best. The best classification accuracy that they were able to achieve on this seven-class problem is approximately 83%, which is comparable to the performance of other similar studies. However, the classification problem considered here is more difficult because the pattern classes used in our experiments have a large overlap of words in their corresponding documents (LI, 1998).

Goller et al, thoroughly evaluate a wide variety of methods on a document classification task for German text. They evaluate different feature construction and selection methods and various classifiers. Their main results are: feature selection is necessary not only to reduce learning and classification time, but also to avoid overfitting (even for Support Vector Machines); surprisingly, their morphological analysis does not improve classification quality compared to a letter 5-gram approach. Support Vector Machines are significantly better than all other classification methods (Goller, 2002).

Ankit et al, discusses the different types of feature vectors through which document can be represented and later classified. They compares the Binary, Count and TfIdf feature vectors and their impact on document classification. To test how well each of the three mentioned feature vectors perform, they used the 20-newsgroup dataset and converted the documents to all the three feature vectors. For each feature vector representation, they trained the Naïve Bayes classifier and then tested the generated classifier on test documents. In their results, they found that

TfIdf performed 4% better than Count vectorizer and 6% better than Binary vectorizer if stop words are removed. If stop words are not removed, then TfIdf performed 6% better than Binary vectorizer and 11% better than Count vectorizer. Also, Count vectorizer performs better than Binary vectorizer, if stop words are removed by 2% but lags behind by 5% if stop words are not removed. Thus, they can conclude that TfIdf should be the preferred vectorizer for document representation and classification (Ankit, 2017).

CHAPTER THREE

SYSTEM ANALYSIS AND DESIGN

3.0 INTRODUCTION

This chapter shows all the modules and components used to design the system, and how they work together. It also shows us how the users of the system interact with the system.

3.1 ANALYSIS OF THE EXSISTING SYSTEM

Currently the existing system is manual, health workers presently classify health documents through stacking of physical files in file cabinets. This makes it difficult to retrieve files when a file of a particular category is required.

3.2 ANALYSIS OF THE PROPOSED SYSTEM

System analysis and design deal with planning the development of information systems through understanding and specifying in detail what a system should do and how the components of the system should be implemented and work together. System analysts solve business problems through analyzing the requirements of information systems and designing such systems by applying analysis and design techniques.

3.2.1 REQUIREMENTS OF THE SYSTEM

For the system to serve its intended purpose properly, the system will have to meet the following requirements.

1. It should be able to accept as input text documents with the following extension doc.
2. It should be able to search for defined text in documents.
3. It should be able to summarize documents.
4. It should be able to categorize and summarize text.
5. It should be able to tokenize text, carry out stemming and lemmatization.

6. It should be able to identify sentences.
7. It should be able to perform Conference resolution, Word Sense Disambiguation and Sentence Boundary Disambiguation.

3.3 TRAINING A MODEL

In machine learning, models are used to train algorithms. The algorithm learns from the model to the point that when it will produce similar result when similar data (similar to the model) is presented to the algorithm. In this project work we make use of the ML algorithm for document classification.

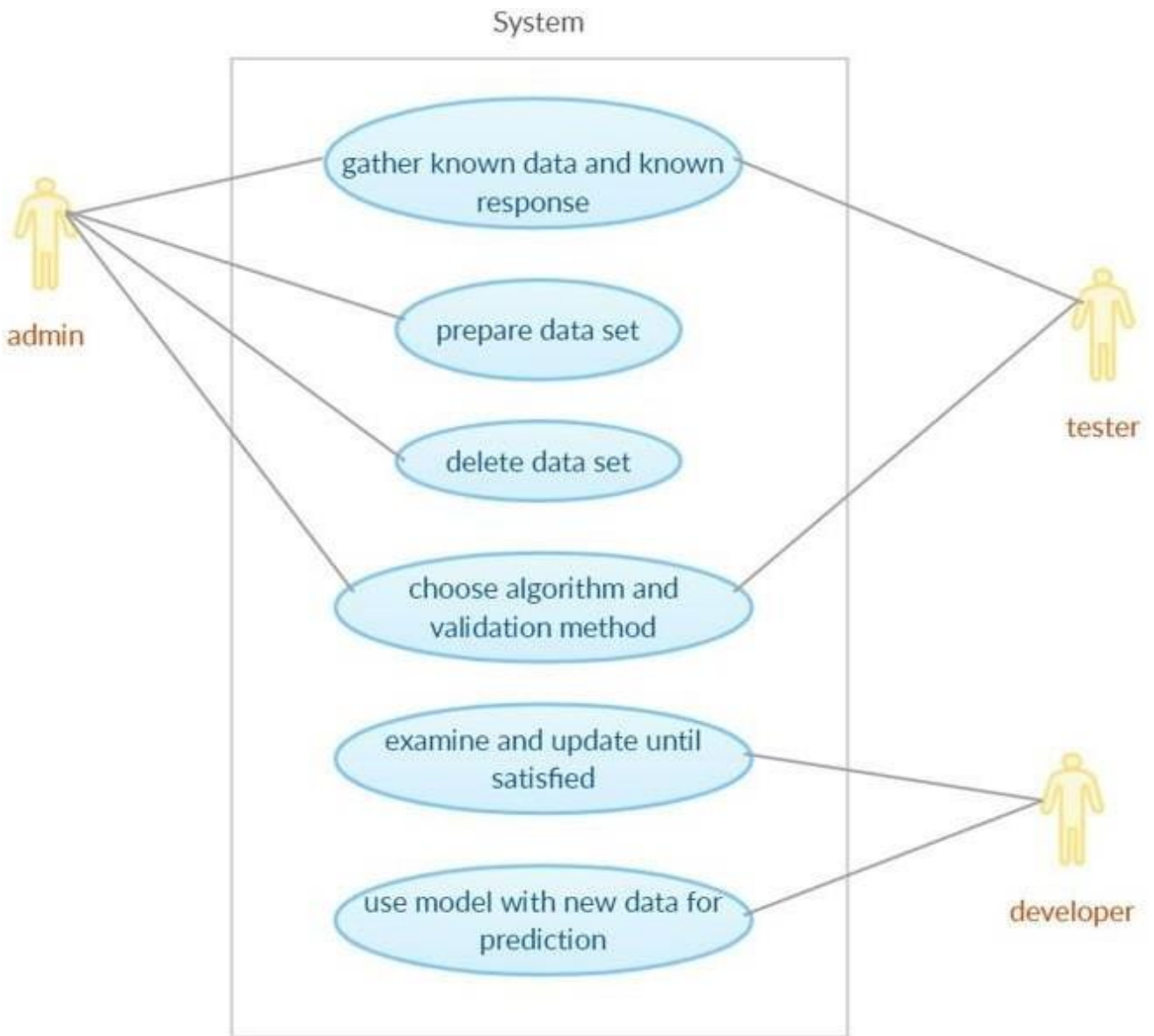
In other to carry out the classification, we first train a model. Our model is built to identify disease such as malaria, hypertension and diarrhea. We opted to start with these diseases as a little Google search shows them to be the most common diseases and their medical specialist. In order to construct a model in ML, you need to create a file of training data. The training file format consists of a series of lines, it includes columns like description, medical specialists, keywords. We use numerous lines of text containing the words like pulmonary ,skin diseases , tooth decay, heart attack, asthma , bone fracture, hypertension to create a training file called “mtsamples.csv”. The en-diseases train file is passed to the train method of the model_predict. The train method trains the file and outputs a model file with a .pkl file name extension.

3.4 CLASSIFYING THE DOCUMENT

After training, the model file produced will be used to classify the health documents. The trained method of the Health Document pickle is used to classify the documents into respective categories.

3.5 USE CASE DIAGRAM

The use case diagram is used to show the interaction between the system use cases and its clients without much detail. A use case diagram displays an actor and its use cases, the actors are also the users of the system.



3.6 SEQUENCE DIAGRAM

Sequence diagrams are simple subsets of interaction diagrams. They map out sequential events in an engineering or business process in order to streamline activities. Sequence diagrams are used to show how objects interact in a given situation. An important characteristic of a sequence diagram is that time passes from top to bottom: the interaction starts near the top of the diagram and ends at the bottom (i.e. **L**ower equals **L**ater).

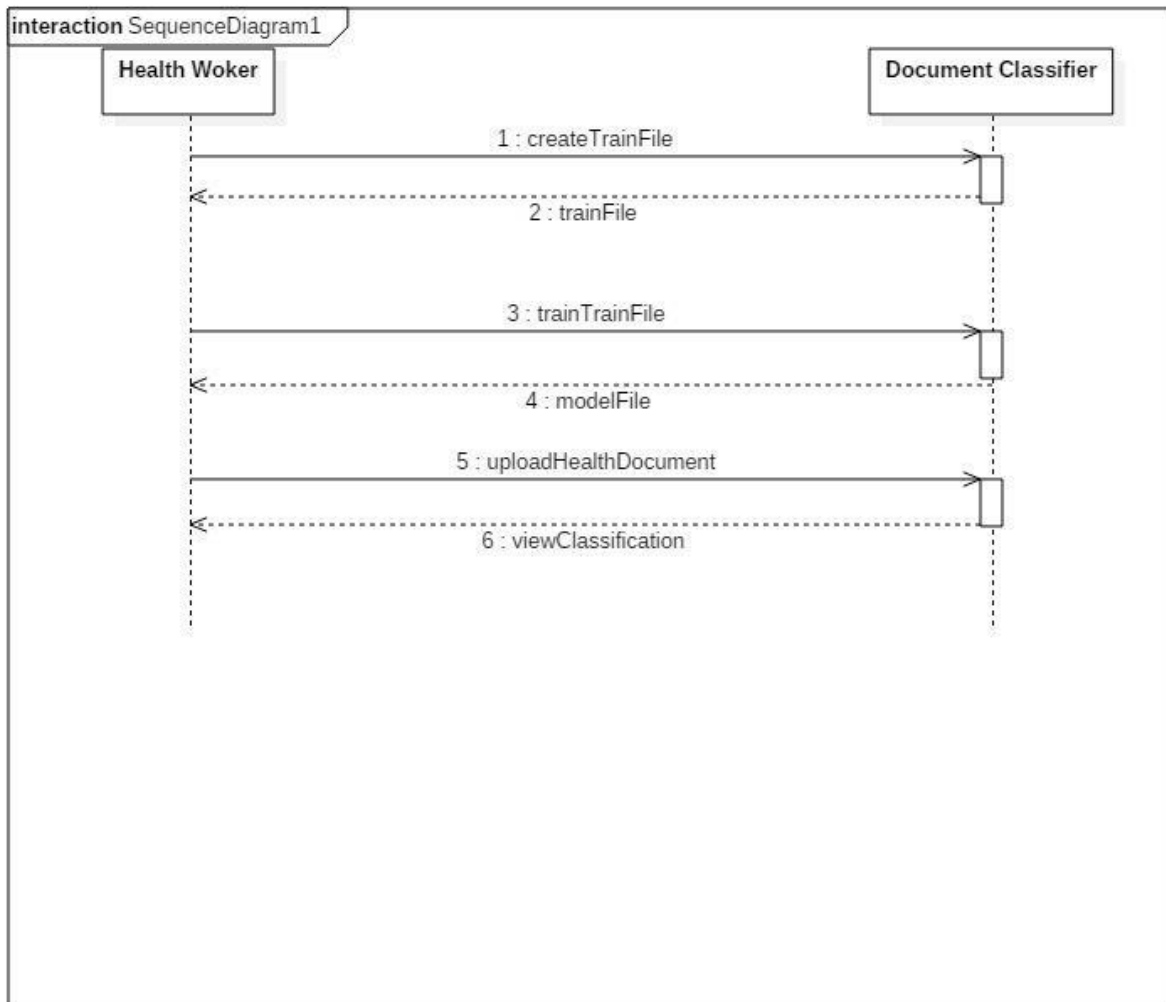


Fig: 3.6 Sequence Diagram

3.7 CLASS DIAGRAMS

We begin our OOD process by identifying the classes required to build the system. We describe these classes using class diagrams and implement them in Java. The class diagram enables us to model via class diagrams, each class is modeled as a rectangle with three compartments. The top one contains the name of the class centered horizontally in bold face. The middle compartment contains the class attributes, while the bottom compartment contains the class behavior or operation. Below is the class diagram for the system.

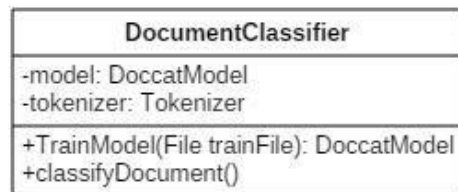
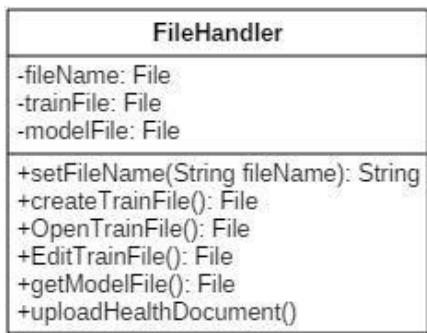
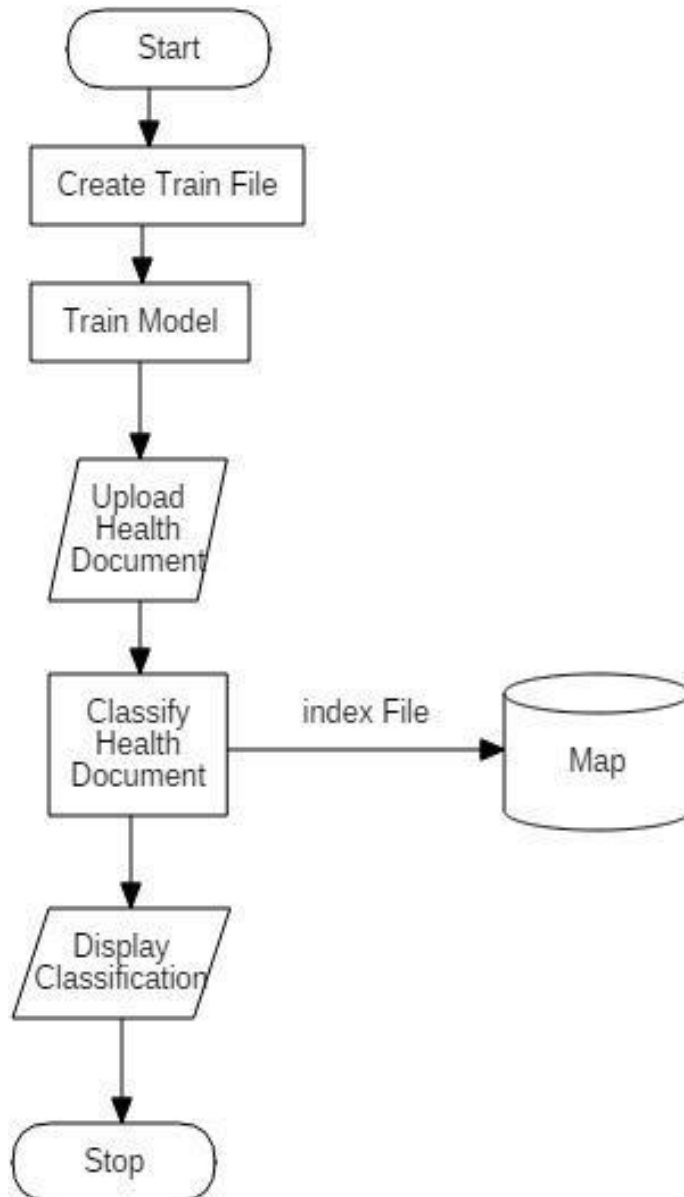


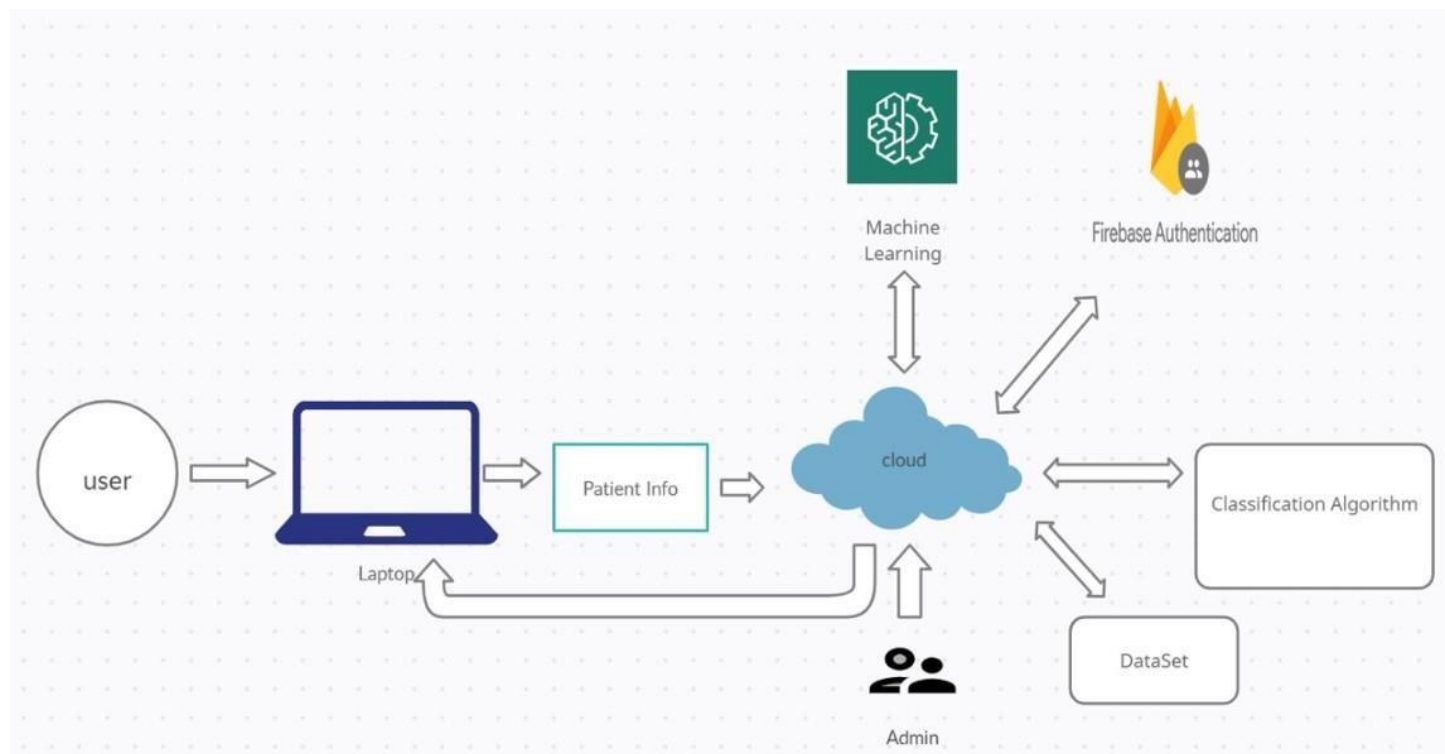
Figure 3.7 Class Diagram

3.8 SYSTEM FLOW CHART

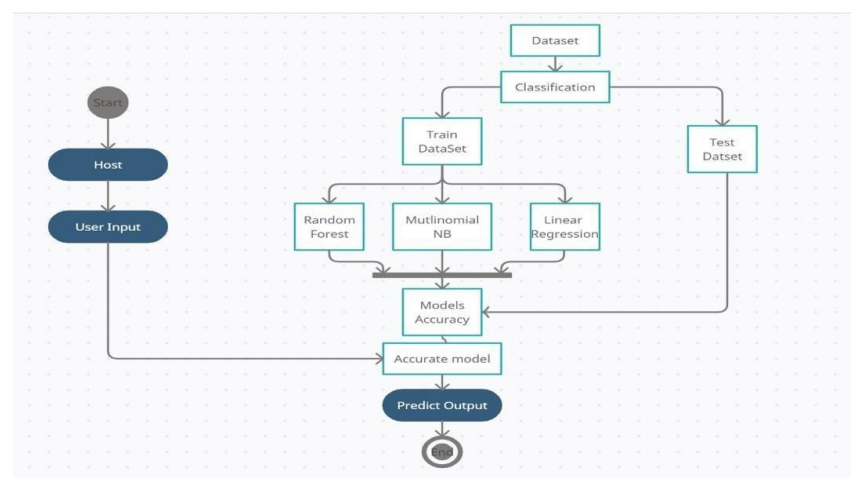
This is a graphical representation of the sequence of operations in an information system or program. Information system flowcharts show how data flows from source documents through the computer to final distribution to users. The following figures are the system flow chart for our system.



3.9 SYSTEM ARCHITECTURE



3.10 ACTIVITY



CHAPTER FOUR

SYSTEM IMPLEMENTATION

4.0 INTRODUCTION

After careful requirement gathering, analysis and design, the system is implemented. Implementation involves testing the system with required data and observing the results to see if the system has been properly designed or if it contains bugs. This is usually done with data which has known results. In this chapter we will implement the system designed.

4.1 SYSTEM REQUIREMENTS

To implement the application, the computer on which it will run has to meet some hardware and software requirements. Also since it has been designed as a web enabled application, the server on which the system will be deployed also has to meet certain hardware and software requirements. The following section will outline these requirements.

4.2 SOFTWARE AND HARDWARE SPECIFICATIONS

4.2.1 Software Requirements

- Operating system: Windows, Linux
- Technology: Flask, Python Pickle, Jinja
- Web technologies: Html, JS, CSS
- Python Version: Python 3.7.9
- Cloud Platform: Heroku (PaaS)
- Google Firebase Authentication

4.2.2 Hardware Requirements

- Hardware: Pentium based systems
- RAM :1GB
- Storage:10GB

4.2 OUTPUT

C:\Windows\System32\cmd.exe - python app.py

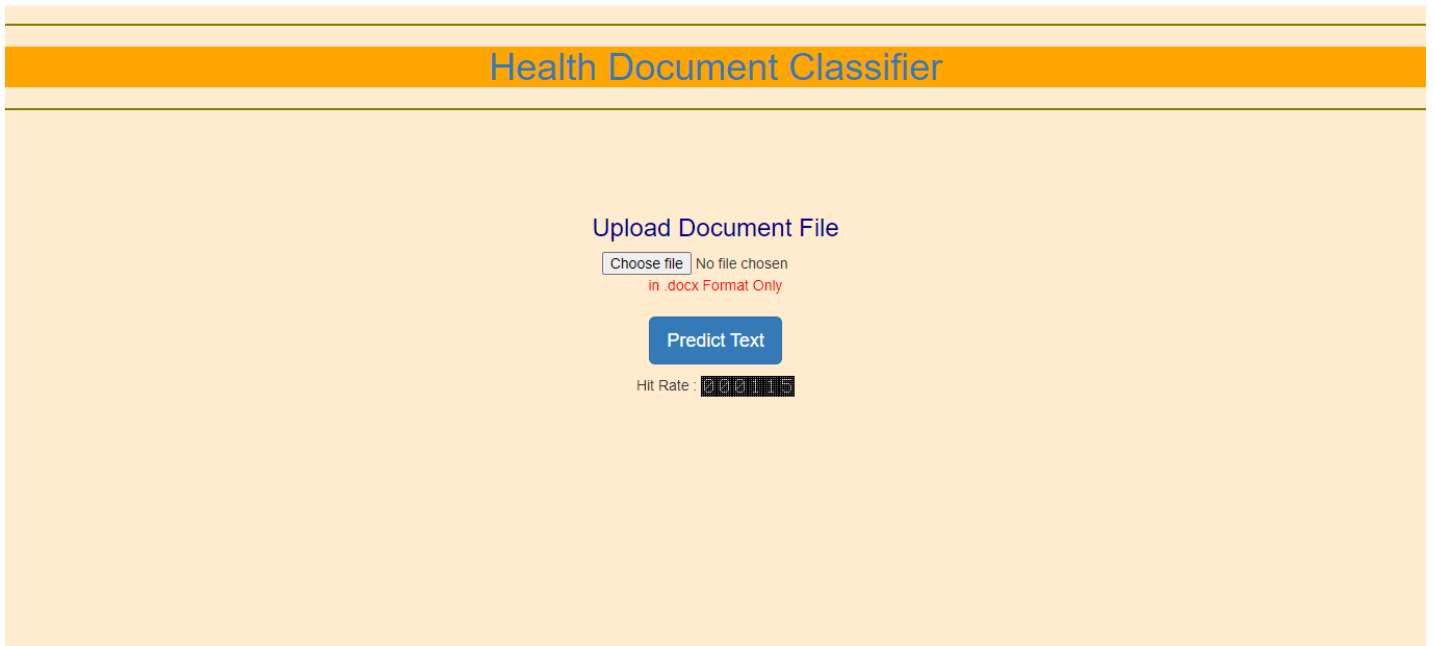
```
C:\Users\pmogili\Desktop\Medical doc classification\MAJOR PROJECT\MajorProject>python model.py
[' Surgery']
[' Radiology']
end_end_end
```

```
C:\Users\pmogili\Desktop\Medical doc classification\MAJOR PROJECT\MajorProject>python app.py
* 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: off
* Running on http://127.0.0.1:6442/ (Press CTRL+C to quit)
```

4.2.1 Home Page

In this page we will be having ‘upload document’ field for uploading the user document in docx format

After uploading the file, user have to click ‘Predict text.’



Health Document Classifier

Upload Document File

Choose file No file chosen

in .docx Format Only

Predict Text

Hit Rate: 0.0015

4.2.2 After Uploading Required Document

Now user uploaded the file and must click on 'Predict Text'

Health Document Classifier

Upload Document File

Choose file

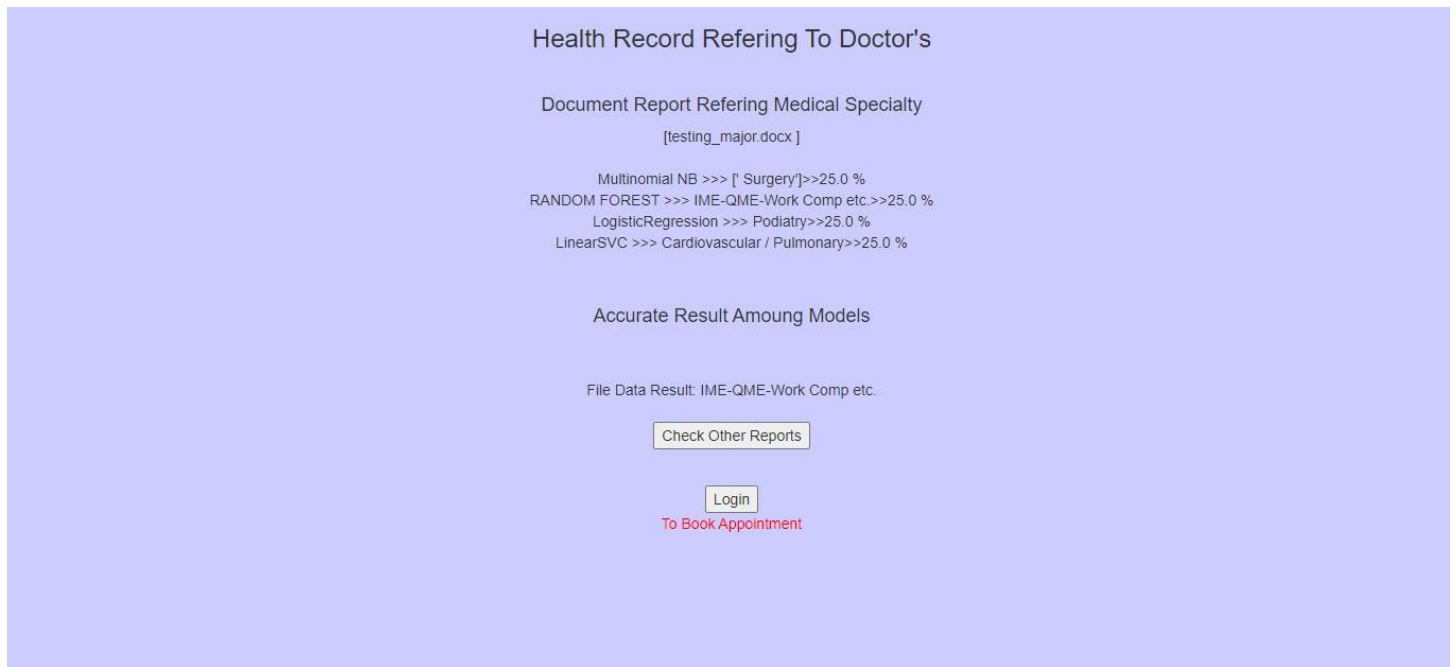
testing_major.docx

in .docx Format Only

Predict Text

Hit Rate : 000115

4.2.3 Health Record



This page is the result of uploaded document, this refers to the doctor based on the content given in the file.

This page consists of 4 Machine learning models which are:

Multinomial Naive Bayes, Random Forest, Logistic Regression, Linear SVM models

If two or more models showing same result, then that will be the final accurate result.

If Four models showing Four different results, then we will be considering the

Random Forest model's result.

If the user is satisfied with the result, he will be logging in and can book appointment.

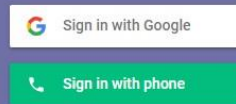
Even in this page we will be having 'Check other Documents' button which is used to upload another document.

Which is nothing but going back to home page.

4.2.4 Login page

Health Dashboard Login

App:



This is Health Dashboard Login page

If the user wants to book an appointment then he can sign in.

In this user can sign in with either with google or with his mobile using phone number.


User can sign in for booking respective doctor's appointment.

4.2.5 User Authentication

Health Dashboard Login

App:

Enter your phone number


 +91

Phone number
9177880625

CANCEL

VERIFY

By tapping Verify, an SMS may be sent. Message & data rates may apply.

 Privacy • Terms

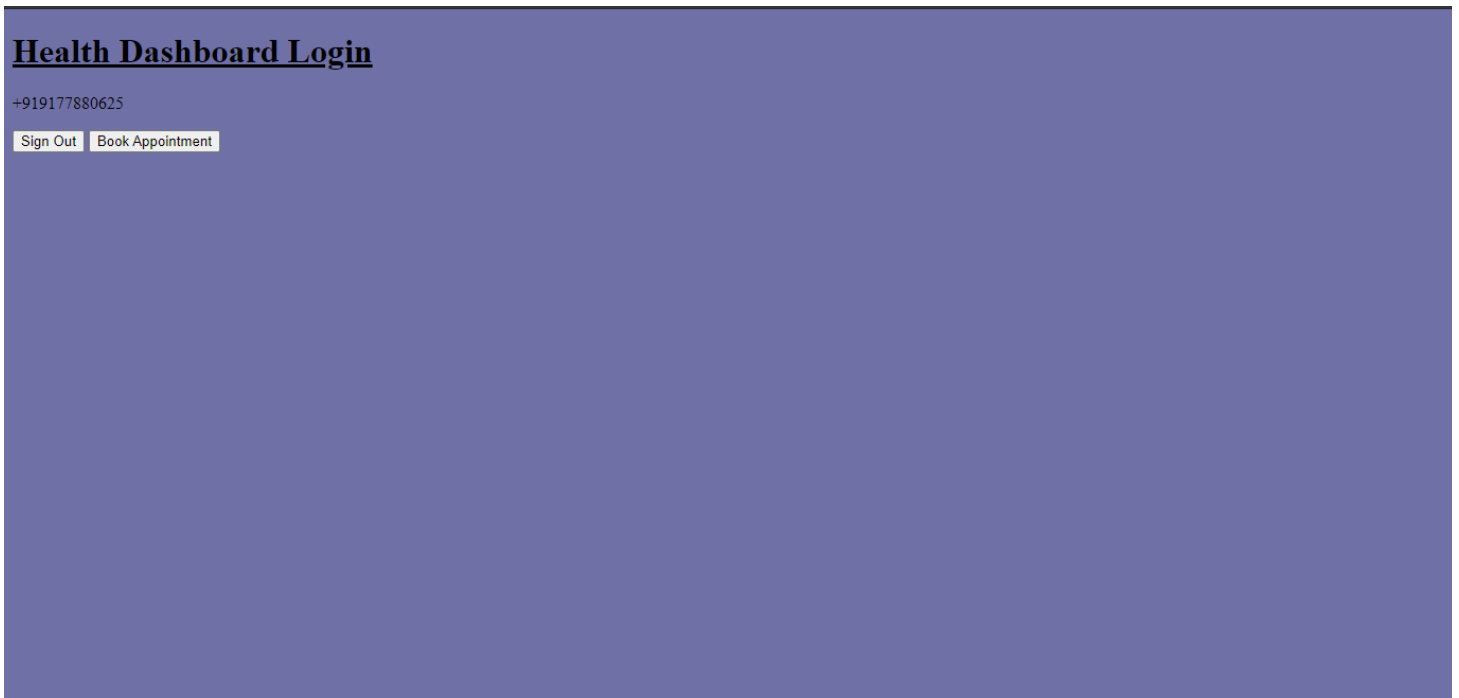
After choosing mobile or email, the user will be authenticated.

After entering mobile number, the user will receive an OTP (One Time Password) for the given mobile number.

After entering the OTP, the user will be successfully signed in for booking an appointment.

Same with the 'Email' as well, the OTP will be sent to the given email address, the user needs to enter the OTP to Sign in.

4.2.6 Book an appointment



After successful sign in, the user can now book an appointment for the respective specialist shown in the Health record page.

ALGORITHM DESCRIPTION

Multinomial Naive Bayes algorithm

Multinomial Naive Bayes algorithm is a probabilistic learning method that is mostly used in Natural Language Processing (NLP). The algorithm is based on the Bayes theorem and predicts the tag of a text such as a piece of email or newspaper article. It calculates the probability of each tag for a given sample and then gives the tag with the highest probability as output.

Naive Bayes classifier is a collection of many algorithms where all the algorithms share one common principle, and that is each feature being classified is not related to any other feature. The presence or absence of a feature does not affect the presence or absence of the other feature.

Bayes theorem, formulated by Thomas Bayes, calculates the probability of an event occurring based on the prior knowledge of conditions related to an event. It is based on the following formula:

$$P(A|B) = P(A) * P(B|A)/P(B)$$

Where we are calculating the probability of class A when predictor B is already provided.

$P(B)$ = prior probability of B

$P(A)$ = prior probability of class A

$P(B|A)$ = occurrence of predictor B given class A probability

This formula helps in calculating the probability of the tags in the text.

Random Forest Algorithm

Random Forest can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.

As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting.

Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

The Working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Logistic Regression

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables.

Logistic regression predicts the output of a categorical dependent variable. Therefore, the outcome must be a categorical or discrete value. It can be either yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1.

Logistic Regression Equation:

- We know the equation of the straight line can be written as:

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

- In Logistic Regression y can be between 0 and 1 only, so for this let's divide the above equation by (1-y):

$$\frac{y}{1-y}; 0 \text{ for } y=0, \text{ and infinity for } y=1$$

- But we need range between -[infinity] to +[infinity], then take logarithm of the equation it will become:

$$\log \left[\frac{y}{1-y} \right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \dots + b_nx_n$$

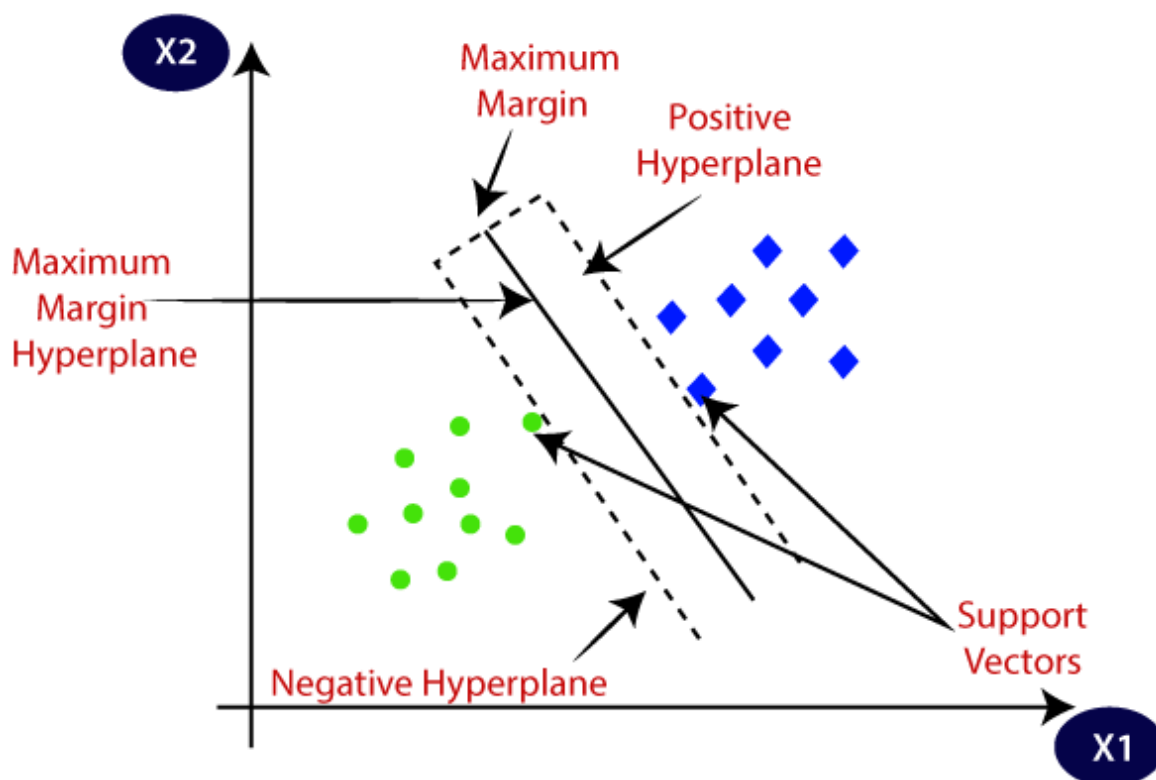
The above equation is the final equation for Logistic Regression.

Support Vector Machine Algorithm

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



CHAPTER FIVE

SUMMARY AND CONCLUSION

5.0 INTRODUCTION

This chapter summarizes and concludes the project work; it also gives recommendations and insight to future work.

5.1 SUMMARY

In this project work we were able to succeed in applying Natural Language Processing which is a branch of Machine Learning to Classifying Health related documents. We made use of the Multinomial NB, Linear SVM, Random-forest, Logistic regression by considering their accuracies for training a model and classifying the documents. We make use of Materialize which is a HTML5, CSS and JavaScript framework for building the user interface. The software is also built using the Model-View-Controller (MVC) architecture.

5.2 RECOMMENDATION

To properly use the system, we recommend the following:

1. The system is hosted in Heroku platform, so that all users can access it from their respective locations (details of this can be found in chapter four).
2. Medical Personnel should be trained on how to use the system.
3. The model should be properly trained and use proper data set to ensure accurate classification by the system, a poorly trained model will lead to erroneous classification.

5.3 FUTURE WORK

Due to the limited time involved in developing this project work, some key features could not be integrated, it is my recommendation that in future work, the following features be added.

1. A crawler should be implemented such that the model is constantly being updated from the internet.
2. When there is new data added to the model from the internet, a listener (should be implemented) that triggers the retraining of the algorithm should be notified.
3. The choice of booking an appointment with specialist as a patient with his document report.

5.4 CONCLUSION

In conclusion we can see that applying Natural Language Processing to classification of text and text based documents is the most effective instead of using other machine learning techniques such as clustering which can be regarded as over kill. Natural language processing has a lot of potential outside document classification; its relevance has been seen in the area of sentiment analysis. It is my recommendation that further research be carried out in the field of Natural Language processing.

CHAPTER SIX

REFERENCES

- Russell Power, Jay Chen, Trishank Karthik and Lakshminarayanan Subramanian (2018), “Document Classification for Focused Topics” <https://cs.nyu.edu/~jchen/publications/aaai4d-power.pdf>.
- Hull D., J. Pedersen, and H. Schutze (1996), “Document routing as statistical classification,” in AAAI Spring Symp. On Machine Learning in Information Access Technical Papers, Palo Alto.
- Fox C. (1992), “Lexical analysis and stoplist,” in Information Retrieval Data Structures and Algorithms, W. Frakes and R. Baeza-Yates, Eds. Prentice Hall, , pp. 102–130.
- Geisser S. (1992), Predictive Inference. NY: Chapman and Hall.
- Liu H. and Motoda (1998), Feature Extraction, construction and selection: A Data Mining Perspective. Boston, Massachusetts: Springer.
- Wang Y. and X. Wang (2005), “A new approach to feature selection in text classification,” in Proceedings of 4th International Conference on Machine Learning and Cybernetics, vol. 6, pp. 3814–3819.
- Montanes E. (2003), J. Ferandez, I. Diaz, E. F. Combarro, and J. Ranilla, “Measures of rule quality for feature selection in text categorization,” in 5th international Symposium on Intelligent data analysis. Germany: Springer-Verlag, , pp. 589–598.
- Aurangzeb K., B. Baharum, L. H. Lee, and K. Khairullah (2010), “A review of machine learning algorithms for text-documents classification,” Journal of Advances in Information Technology, vol. 1, no. 1.
- Wang Z.-Q., X. Sun, D.-X. Zhang, and X. Li (2006), “An optimal svm based text classification algorithm,” in Fifth International Conference on Machine Learning and Cybernetics, pp. 13–16.
- PAZIENZA, M. T., ed. 1997. Information Extraction. Lecture Notes in Computer Science, Vol. 1299. Springer, Heidelberg, Germany.
- RILOFF, E. 1995. Little words can make a big difference for text classification. In Proceedings of SIGIR-95, 18th ACM International Conference on Research and Development in Information

Retrieval (Seattle, WA, 1995), 130–136.

Leopold, Edda & Kindermann, Jörg (2002), "Text Categorization with Support Vector Machines. How to Represent Texts in Input Space?", Machine Learning 46, pp. 423 - 444.

Brank J., Grobelnik M., Milic-Frayling N., Mladenic D. (2002), "Interaction of Feature Selection Methods and Linear Classification Models", Proc. of the 19th International Conference on Machine Learning, Australia.

Forman, G., An Experimental Study of Feature Selection Metrics for Text Categorization. Journal of Machine Learning Research, 3 2003, pp. 1289-1305.

LI Y. H. AND A. K. JAIN (1998), Classification of Text Documents, THE COMPUTER JOURNAL, Vol. 41, No. 8.

Goller C., J. Löning, T. Will, W. Wolff (2000), Automatic Document Classification: A thorough Evaluation of various Methods, International Symposiums for Information swissenschaft (ISI 2000), Darmstadt, 8. – 10. November 2000.

Ankit Basarkar (2017), DOCUMENT CLASSIFICATION USING

MACHINE LEARNING, San Jose State University SJSU ScholarWorks

http://scholarworks.sjsu.edu/?utm_source=scholarworks.sjsu.edu%2Fet_projects%2F531&utm_medium=PDF&utm_campaign=PDFCoverPages

Retrieved 5 March 2018.

Kim S. B. , Rim H. C. , Yook D. S. and Lim H. S. (2002), "Effective Methods for Improving Naive Bayes Text Classifiers", LNAI 2417, 2002, pp. 414-423.

Klopotek M. and Woch M. (2003), "Very Large Bayesian Networks in Text Classification", ICCS 2003, LNCS 2657, 2003, pp. 397-406

Shanahan J. and Roma N. (2003), Improving SVM Text Classification Performance through Threshold Adjustment, LNAI 2837, 2003, 361- 372.