A PROJECT REPORT ON

"Drug Recommendation System based on Sentiment Analysis of Drug Reviews using Machine Learning"

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENT FOR THE DEGREE OF 5TH SEMESTER

BACHELOR OF COMPUTER APPLICATION

SUBMITTED TO BENGALURU CITY UNIVERSITY BY

Name of Students: **CHETHAN V**

Registration Number: **(R2014624)**

UNDER THE GUIDANCE OF

Mrs. Syeeda M.

Assistant Professor

Department of Computer Science

SESHADRIPURAM COLLEGE

Seshadripuram, Bengaluru-560020

NAAC ACCREDITED

ACKNOWLEDGEMENT

It's quite difficult for an individual to make things possible without the co-operation of others. It is my privilege to thank all those who have been with us from the start to end.

We wish to express our sincere gratitude to the principal **Dr. B. G. BHASKARA** for her constant support & co-operation.

We express my sincere thanks to H.O.D of BCA Department **Prof. VEENA R**, & to our project guide **Mrs. Syeeda M.** for their constructive information and useful suggestion which helped us to complete our project successfully on time.

We wish to thank the members of teaching & non-teaching staff of our department for their help during my project.

Lastly, we thank our parents and friends who supported me to complete the project on time.

DECLARATION

We hereby declare that the project entitled **"Drug Recommendation System based on Sentiment Analysis of Drug Reviews using Machine Learning"** is Bonafide work carried out by us under the guidance of **Mrs. Syeeda M.**

This project, as presented in this report, is our original work and has not been presented for any other University award.

This project has been submitted for the partial fulfilment of requirements for the

Degree of Bachelor of Computer Application of

BENGALURU CITY UNIVERSITY

Submitted by-

CHETHAN V (R2014624)

ABSTRACT

Since coronavirus has shown up, inaccessibility of legitimate clinical resources is at its peak, like the shortage of specialists and healthcare workers, lack of proper equipment and medicines etc. The entire medical fraternity is in distress, which results in numerous individual's demise. Due to unavailability, individuals started taking medication independently without appropriate consultation, making the health condition worse than usual. As of late, machine learning has been valuable in numerous applications, and there is an increase in innovative work for automation. This paper intends to present a drug recommender system that can drastically reduce specialists heap. In this research, we build a medicine recommendation system that uses patient reviews to predict the sentiment using various vectorization processes like Bow, TF-IDF,Word2Vec, and Manual Feature Analysis, which can help recommend the top drug for a given disease by different classification algorithms. The predicted sentiments were evaluated by precision, recall, f1score, accuracy, and AUC score. The results show that classifier Linear SVC using TF-IDF vectorization outperforms all other models with 93% accuracy

CONTENTS

1. Introduction **1**

- 1.1 Machine Learning
- 1.2 Inferring
- 1.3 Supervised Learning
- 1.4 Unsupervised Learning
- 1.5 Machine Learning algorithm
- 1.6 Artificial Intelligence
- 1.7 Approaches
- 1.8 Models

2. System Design **24**

- 2.1 System Architecture
- 2.2 Input & Output Design

3. System Analysis **32**

- 3.1 Existing System
- 3.2 Proposed System
- 3.3 System Study

4. System Requirements **36**

5. Software Environment **38**

- 5.1 Python
- 5.2 Interactive mode programming
- 5.3 Script mode programming
- 5.4 Flask framework

6. Implementation **50**

- 6.1 Module
- 6.2 Source Code

7. Screenshots **69**

8. Testing **83**

9. Conclusion **87**

10. Bibliography **89**

1. INTRODUCTION

Chapter 1:

Introduction

1.1 What is Machine Learning?

Machine Learning is a system of computer algorithms that can learn from example through self-improvement without being explicitly coded by a programmer. Machine learning is a part of artificial Intelligence which combines data with statistical tools to predict an output which can be used to make actionable insights.

The breakthrough comes with the idea that a machine can singularly learn from the data (i.e., example) to produce accurate results. Machine learning is closely related to data mining and Bayesian predictive modeling. The machine receives data as input and uses an algorithm to formulate answers.

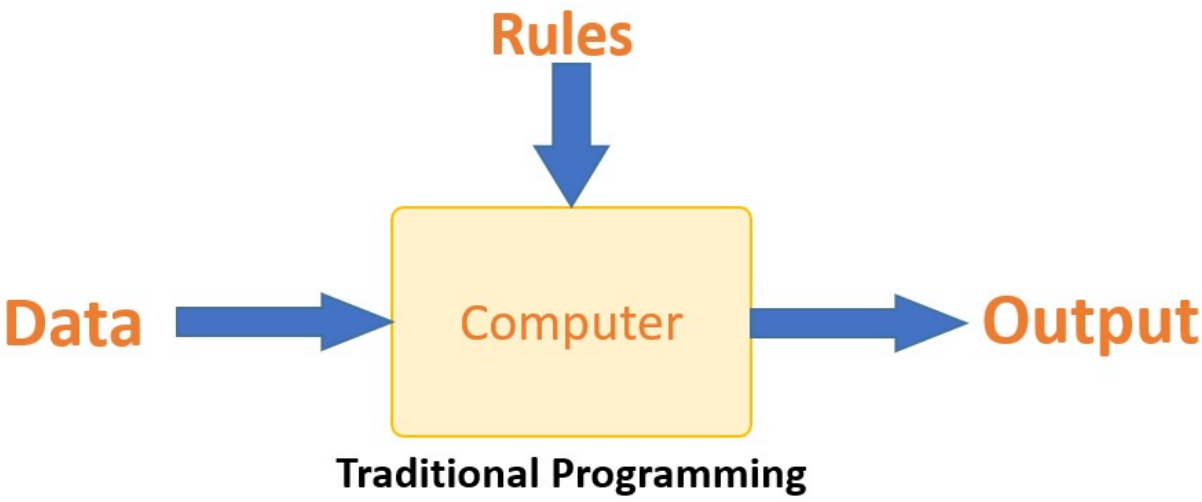
A typical machine learning tasks are to provide a recommendation. For those who have a Netflix account, all recommendations of movies or series are based on the user's historical data. Tech companies are using unsupervised learning to improve the user experience with personalizing recommendation.

Machine learning is also used for a variety of tasks like fraud detection, predictive maintenance, portfolio optimization, automatize task and so on.

Machine Learning vs Traditional Programming

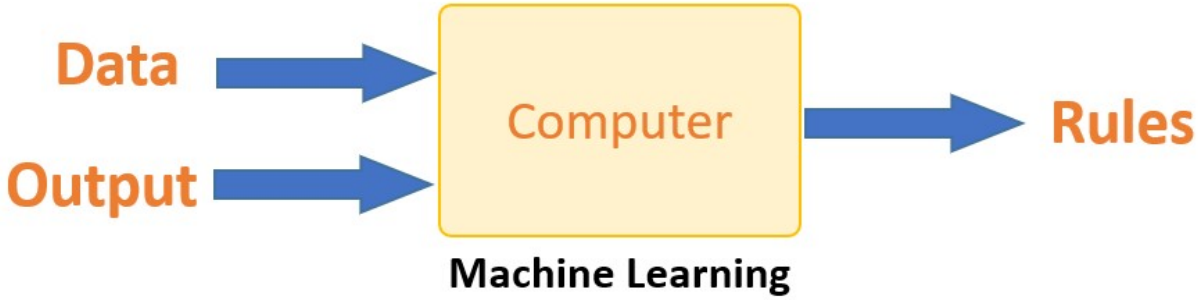
Traditional programming differs significantly from machine learning. In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.

- Traditional programming differs significantly from machine learning. In traditional programming, a programmer code all the rules in consultation with an expert in the industry for which software is being developed. Each rule is based on a logical foundation; the machine will execute an output following the logical statement. When the system grows complex, more rules need to be written. It can quickly become unsustainable to maintain.



Traditional Programming

- Machine learning is supposed to overcome this issue. The machine learns how the input and output data are correlated and it writes a rule. The programmers do not need to write new rules each time there is new data. The algorithms adapt in response to new data and experiences to improve efficacy over time.



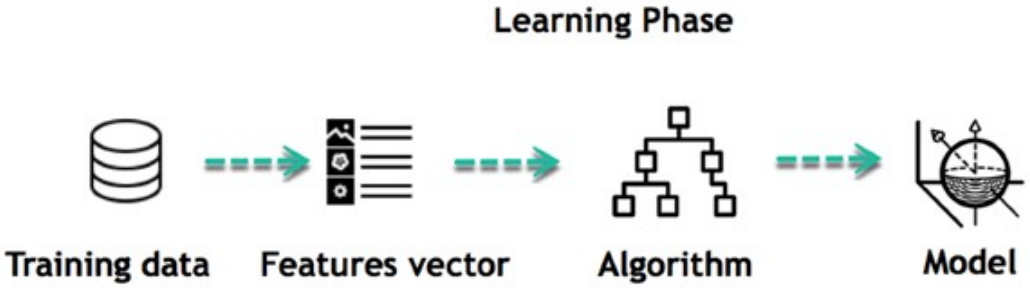
Machine Learning

How does Machine Learning Work?

Machine learning is the brain where all the learning takes place. The way the machine learns is similar to the human being. Humans learn from experience. The more we know, the more easily we can predict. By analogy, when we face an unknown situation, the likelihood of success is lower than the known situation. Machines are trained the same. To make an accurate prediction, the machine sees an example. When we give the machine a similar example, it can figure out the outcome. However, like a human, if its feed a previously unseen example, the machine has difficulties to predict.

The core objective of machine learning is the **learning** and **inference**. First of all, the machine learns through the discovery of patterns. This discovery is made thanks to the **data**. One crucial part of the data scientist is to choose carefully which data to provide to the machine. The list of attributes used to solve a problem is called a **feature vector.** You can think of a feature vector as a subset of data that is used to tackle a problem.

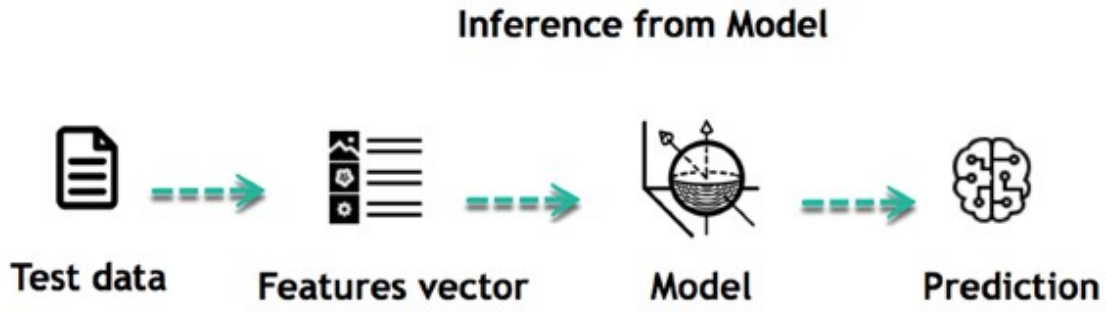
The machine uses some fancy algorithms to simplify the reality and transform this discovery into a **model**. Therefore, the learning stage is used to describe the data and summarize it into a model.



For instance, the machine is trying to understand the relationship between the wage of an individual and the likelihood to go to a fancy restaurant. It turns out the machine finds a positive relationship between wage and going to a high-end restaurant: This is the model

1.2 Inferring

When the model is built, it is possible to test how powerful it is on never-seen-before data. The new data are transformed into a features vector, go through the model and give a prediction. This is all the beautiful part of machine learning. There is no need to update the rules or train again the model. You can use the model previously trained to make inference on new data.

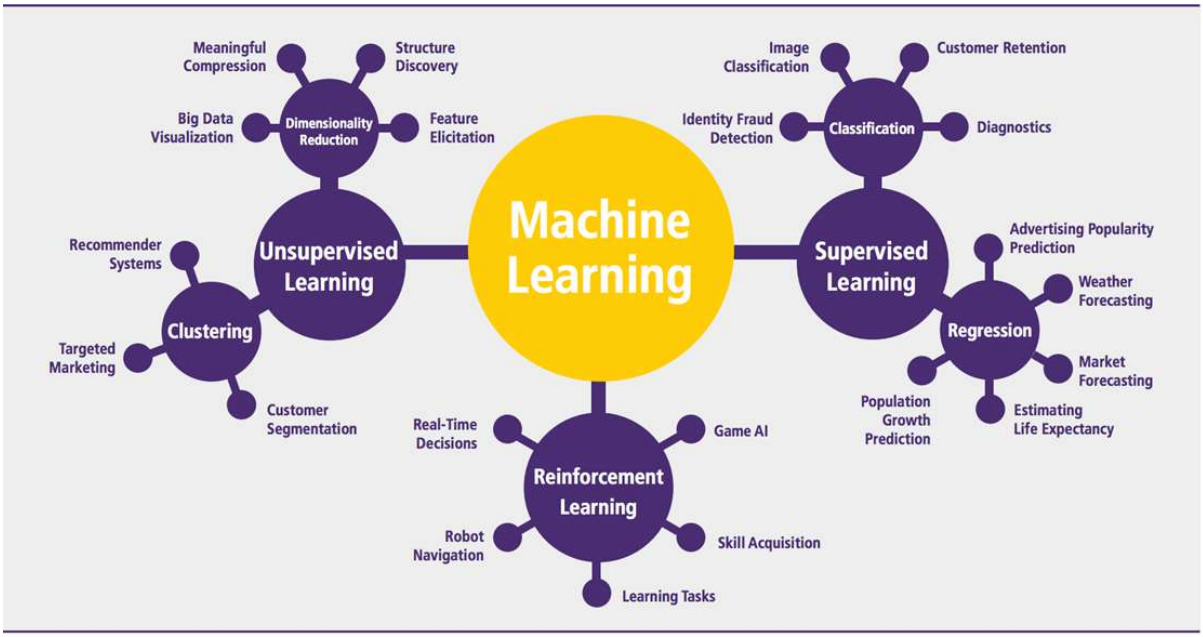


The life of Machine Learning programs is straightforward and can be summarized in the following points:

1. Define a question
2. Collect data
3. Visualize data
4. Train algorithm
5. Test the Algorithm
6. Collect feedback
7. Refine the algorithm
8. Loop 4-7 until the results are satisfying
9. Use the model to make a prediction

Once the algorithm gets good at drawing the right conclusions, it applies that knowledge to new sets of data

Machine Learning Algorithms and Where they are Used?



Machine learning Algorithms

Machine learning can be grouped into two broad learning tasks: Supervised and Unsupervised. There are many other algorithms

1.3 Supervised learning

An algorithm uses training data and feedback from humans to learn the relationship of given inputs to a given output. For instance, a practitioner can use marketing expense and weather forecast as input data to predict the sales of cans.
You can use supervised learning when the output data is known. The algorithm will predict new data.

There are two categories of supervised learning:

- Classification task
- Regression task

Classification

Imagine you want to predict the gender of a customer for a commercial. You will start gathering data on the height, weight, job, salary, purchasing basket, etc. from your customer database.

You know the gender of each of your customer, it can only be male or female. The objective of the classifier will be to assign a probability of being a male or a female (i.e., the label) based on the information (i.e., features you have collected). When the model learned how to recognize male or female, you can use new data to make a prediction. For instance, you just got new information from an unknown customer, and you want to know if it is a male or female. If the classifier predicts male = 70%, it means the algorithm is sure at 70% that this customer is a male, and 30% it is a female.
The label can be of two or more classes. The above Machine learning example has only two classes, but if a classifier needs to predict object, it has dozens of classes (e.g., glass, table, shoes, etc. each object represents a class)

Regression

When the output is a continuous value, the task is a regression. For instance, a financial analyst may need to forecast the value of a stock based on a range of feature like equity, previous stock performances, macroeconomics index. The system will be trained to estimate the price of the stocks with the lowest possible error.

Algorithm Name	Description	Type
Linear regression	Finds a way to correlate each feature to the output to help predict future values.	Regression
Logistic regression	Extension of linear regression that's used for classification tasks. The output variable 3is binary (e.g., only black or white) rather than continuous (e.g., an infinite list of potential colors)	Classification
Decision tree	Highly interpretable classification or regression model that splits data-feature values into branches at decision nodes (e.g., if a feature is a color, each possible color becomes a new branch) until a final decision output is made	Regression Classification
Naive Bayes	The Bayesian method is a classification method that makes use of the Bayesian theorem. The theorem updates the prior knowledge of an event with the independent probability of each feature that can affect the event.	Regression Classification
Support vector machine	Support Vector Machine, or SVM, is typically used for the classification task. SVM algorithm finds a hyperplane that optimally divided the classes. It is best used with a non-linear solver.	Regression Classification
Random forest	The algorithm is built upon a decision tree to improve the accuracy drastically. Random forest generates many times simple decision trees and uses the 'majority vote' method to decide on which label to return. For the classification task, the final prediction will be the one with the most vote; while for the regression task, the average prediction of all the trees is the final prediction.	Regression Classification

Algorithm Name	Description	Type
AdaBoost	Classification or regression technique that uses a multitude of models to come up with a decision but weighs them based on their accuracy in predicting the outcome	Regression Classification
Gradient-boosting trees	Gradient-boosting trees is a state-of-the-art classification/regression technique. It is focusing on the error committed by the previous trees and tries to correct it.	Regression Classification

1.4 Unsupervised learning

In unsupervised learning, an algorithm explores input data without being given an explicit output variable (e.g., explores customer demographic data to identify patterns)
You can use it when you do not know how to classify the data, and you want the algorithm to find patterns and classify the data for you

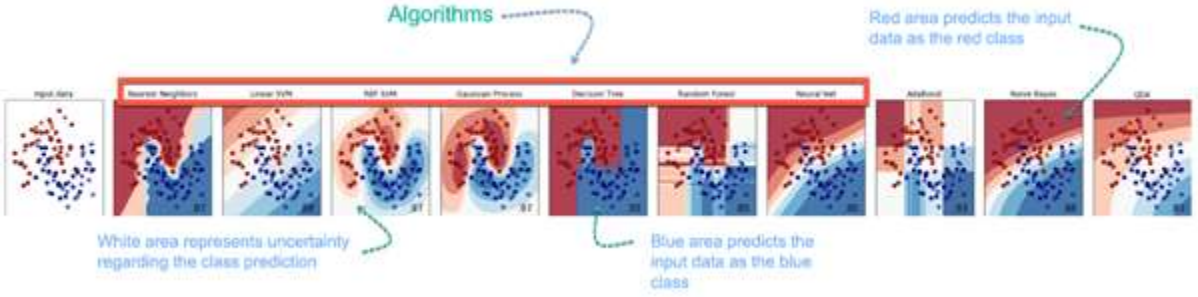
Algorithm	Description	Type
K-means clustering	Puts data into some groups (k) that each contains data with similar characteristics (as determined by the model, not in advance by humans)	Clustering
Gaussian mixture model	A generalization of k-means clustering that provides more flexibility in the size and shape of groups (clusters)	Clustering
Hierarchical clustering	Splits clusters along a hierarchical tree to form a classification system. Can be used for Cluster loyalty-card customer	Clustering
Recommender system	Help to define the relevant data for making a recommendation.	Clustering
PCA/T-SNE	Mostly used to decrease the dimensionality of the data. The algorithms reduce the number of features to 3 or 4 vectors with the highest variances.	Dimension Reduction

How to Choose Machine Learning Algorithm

1.5 Machine Learning (ML) algorithm:

There are plenty of machine learning algorithms. The choice of the algorithm is based on the objective.

In the Machine learning example below, the task is to predict the type of flower among the three varieties. The predictions are based on the length and the width of the petal. The picture depicts the results of ten different algorithms. The picture on the top left is the dataset. The data is classified into three categories: red, light blue and dark blue. There are some groupings. For instance, from the second image, everything in the upper left belongs to the red category, in the middle part, there is a mixture of uncertainty and light blue while the bottom corresponds to the dark category. The other images show different algorithms and how they try to classified the data.



1.5.1 Challenges and Limitations of Machine Learning

The primary challenge of machine learning is the lack of data or the diversity in the dataset. A machine cannot learn if there is no data available. Besides, a dataset with a lack of diversity gives the machine a hard time. A machine needs to have heterogeneity to learn meaningful insight. It is rare that an algorithm can extract information when there are no or few variations. It is recommended to have at least 20 observations per group to help the machine learn. This constraint leads to poor evaluation and prediction.

1.5.2 Application of Machine Learning

Augmentation:

- Machine learning, which assists humans with their day-to-day tasks, personally or commercially without having complete control of the output. Such machine learning is used in different ways such as Virtual Assistant, Data analysis, software solutions. The primary user is to reduce errors due to human bias.

Automation:

- Machine learning, which works entirely autonomously in any field without the need for any human intervention. For example, robots performing the essential process steps in manufacturing plants.

Finance Industry

- Machine learning is growing in popularity in the finance industry. Banks are mainly using ML to find patterns inside the data but also to prevent fraud.

Government organization

- The government makes use of ML to manage public safety and utilities. Take the example of China with the massive face recognition. The government uses Artificial intelligence to prevent jaywalker.

Healthcare industry

- Healthcare was one of the first industry to use machine learning with image detection.

Marketing

- Broad use of AI is done in marketing thanks to abundant access to data. Before the age of mass data, researchers develop advanced mathematical tools like Bayesian analysis to estimate the value of a customer. With the boom of data, marketing department relies on AI to optimize the customer relationship and marketing campaign.

1.5.3 Example of application of Machine Learning in Supply Chain

Machine learning gives terrific results for visual pattern recognition, opening up many potential applications in physical inspection and maintenance across the entire supply chain network.
Unsupervised learning can quickly search for comparable patterns in the diverse dataset. In turn, the machine can perform quality inspection throughout the logistics hub, shipment with damage and wear.
For instance, IBM's Watson platform can determine shipping container damage. Watson combines visual and systems-based data to track, report and make recommendations in real-time.
In past year stock manager relies extensively on the primary method to evaluate and forecast the inventory. When combining big data and machine learning, better forecasting techniques have been implemented (an improvement of 20 to 30 % over traditional forecasting tools). In term of sales, it means an increase of 2 to 3 % due to the potential reduction in inventory costs.

1.5.4 Example of Machine Learning Google Car

For example, everybody knows the Google car. The car is full of lasers on the roof which are telling it where it is regarding the surrounding area. It has radar in the front, which is informing the car of the speed and motion of all the cars around it. It uses all of that data to figure out not only how to drive the car but also to figure out and predict what potential drivers around the car are going to do. What's impressive is that the car is processing almost a gigabyte a second of data.

1.5.5 Why is Machine Learning Important?

Machine learning is the best tool so far to analyze, understand and identify a pattern in the data. One of the main ideas behind machine learning is that the computer can be trained to automate tasks that would be exhaustive or impossible for a human being. The clear breach from the traditional analysis is that machine learning can take decisions with minimal human intervention.

Take the following example for this ML tutorial; a retail agent can estimate the price of a house based on his own experience and his knowledge of the market.

A machine can be trained to translate the knowledge of an expert into features. The features are all the characteristics of a house, neighborhood, economic environment, etc. that make the price difference. For the expert, it took him probably some years to master the art of estimate the price of a house. His expertise is getting better and better after each sale.

For the machine, it takes millions of data, (i.e., example) to master this art. At the very beginning of its learning, the machine makes a mistake, somehow like the junior salesman. Once the machine sees all the example, it got enough knowledge to make its estimation. At the same time, with incredible accuracy. The machine is also able to adjust its mistake accordingly.

Most of the big company have understood the value of machine learning and holding data. McKinsey have estimated that the value of analytics ranges from $9.5 trillion to $15.4 trillion while $5 to 7 trillion can be attributed to the most advanced AI techniques.

Machine learning (ML) is the study of computer algorithms that improve automatically through experience. It is seen as a part of artificial intelligence. Machine learning algorithms build a model based on sample data, known as "training data", in order to make predictions or decisions without being explicitly programmed to do so. Machine learning algorithms are used in a wide variety of applications, such as email filtering and computer vision, where it is difficult or unfeasible to develop conventional algorithms to perform the needed tasks.

A subset of machine learning is closely related to computational statistics, which focuses on making predictions using computers; but not all machine learning is statistical learning. The study of mathematical optimization delivers methods, theory and application domains to the field of machine learning. Data mining is a related field of study, focusing on exploratory data analysis through unsupervised learning. In its application across business problems, machine learning is also referred to as predictive analytics.

1.5.6 Overview

Machine learning involves computers discovering how they can perform tasks without being explicitly programmed to do so. It involves computers learning from data provided so that they carry out certain tasks. For simple tasks assigned to computers, it is possible to program algorithms telling the machine how to execute all steps required to solve the problem at hand; on the computer's part, no learning is needed. For more advanced tasks, it can be challenging for a human to manually create the needed algorithms. In practice, it can turn out to be more

effective to help the machine develop its own algorithm, rather than having human programmers specify every needed step.

The discipline of machine learning employs various approaches to teach computers to accomplish tasks where no fully satisfactory algorithm is available. In cases where vast numbers of potential answers exist, one approach is to label some of the correct answers as valid. This can then be used as training data for the computer to improve the algorithm(s) it uses to determine correct answers. For example, to train a system for the task of digital character recognition, the MNIST dataset of handwritten digits has often been used.

1.5.7 Machine learning approaches

Machine learning approaches are traditionally divided into three broad categories, depending on the nature of the "signal" or "feedback" available to the learning system:

Supervised learning: The computer is presented with example inputs and their desired outputs, given by a "teacher", and the goal is to learn a general rule that maps inputs to outputs.

Unsupervised learning: No labels are given to the learning algorithm, leaving it on its own to find structure in its input. Unsupervised learning can be a goal in itself (discovering hidden patterns in data) or a means towards an end (feature learning).

Reinforcement learning: A computer program interacts with a dynamic environment in which it must perform a certain goal (such as driving a vehicle or playing a game against an opponent). As it navigates its problem space, the program is provided feedback that's analogous to rewards, which it tries to maximize.
Other approaches have been developed which don't fit neatly into this three-fold categorization, and sometimes more than one is used by the same machine learning system. For example: topic modeling, dimensionality reduction or meta learning.

As of 2020, deep learning has become the dominant approach for much ongoing work in the field of machine learning.

1.5.8 History and relationships to other fields

The term machine learning was coined in 1959 by Arthur Samuel, an American IBM and pioneer in the field of computer gaming and artificial intelligence. A representative book of the machine learning research during the 1960s was the Nilsson's book on Learning Machines, dealing mostly with machine learning for pattern classification. Interest related to pattern recognition continued into the 1970s, as described by Duda and Hart in 1973. In 1981 a report was given on using teaching strategies so that a neural network learns to recognize 40 characters (26 letters, 10 digits, and 4 special symbols) from a computer terminal.

Tom M. Mitchell provided a widely quoted, more formal definition of the algorithms studied in the machine learning field: "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P if its performance at tasks in T, as measured by P, improves with experience E "This definition of the tasks in which machine learning is concerned offers a fundamentally operational definition rather than defining the field in cognitive terms. This follows Alan Turing's proposal in his paper "Computing Machinery and Intelligence", in which the question "Can machines think?" is replaced with the question "Can machines do what we (as thinking entities) can do?".

Modern day machine learning has two objectives, one is to classify data based on models which have been developed, the other purpose is to make predictions for future outcomes based on these models. A hypothetical algorithm specific to classifying data may use computer vision of moles coupled with supervised learning in order to train it to classify the cancerous moles. Where as, a machine learning algorithm for stock trading may inform the trader of future potential predictions.

1.6 Artificial intelligence

Machine Learning as subfield of AI
Part of Machine Learning as subfield of AI or part of AI as subfield of Machine Learning
As a scientific endeavor, machine learning grew out of the quest for artificial intelligence. In the early days of AI as an academic discipline, some researchers were interested in having machines learn from data. They attempted to approach the problem with various symbolic methods, as well as what was then termed "neural networks"; these were mostly perceptron's and other models that were later found to be reinventions of the generalized linear models of statistics. Probabilistic reasoning was also employed, especially in automated medical diagnosis.
However, an increasing emphasis on the logical, knowledge-based approach caused a rift between AI and machine learning. Probabilistic systems were plagued by theoretical and practical problems of data acquisition and representation. By 1980, expert systems had come to dominate AI, and statistics was out of favor. Work on symbolic/knowledge-based learning did continue within AI, leading to inductive logic programming, but the more statistical line of research was now outside the field of AI proper, in pattern recognition and information retrieval. Neural networks research had been abandoned by AI and computer science around the same time. This line, too, was continued outside the AI/CS field, as "connectionism", by researchers from other disciplines including Hopfield, Rumelhart and Hinton. Their main success came in the mid-1980s with the reinvention of backpropagation.

Machine learning (ML), reorganized as a separate field, started to flourish in the 1990s. The field changed its goal from achieving artificial intelligence to tackling solvable problems of a

practical nature. It shifted focus away from the symbolic approaches it had inherited from AI, and toward methods and models borrowed from statistics and probability theory.

As of 2020, many sources continue to assert that machine learning remains a subfield of AI. The main disagreement is whether all of ML is part of AI, as this would mean that anyone using ML could claim they are using AI. Others have the view that not all of ML is part of AI where only an 'intelligent' subset of ML is part of AI.

The question to what is the difference between ML and AI is answered by Judea Pearl in The Book of Why. Accordingly, ML learns and predicts based on passive observations, whereas AI implies an agent interacting with the environment to learn and take actions that maximize its chance of successfully achieving its goals.

1.6.1 Data mining

Machine learning and data mining often employ the same methods and overlap significantly, but while machine learning focuses on prediction, based on known properties learned from the training data, data mining focuses on the discovery of (previously) unknown properties in the data (this is the analysis step of knowledge discovery in databases). Data mining uses many machine learning methods, but with different goals; on the other hand, machine learning also employs data mining methods as "unsupervised learning" or as a preprocessing step to improve learner accuracy. Much of the confusion between these two research communities (which do often have separate conferences and separate journals, ECML PKDD being a major exception) comes from the basic assumptions they work with: in machine learning, performance is usually evaluated with respect to the ability to reproduce known knowledge, while in knowledge discovery and data mining (KDD) the key task is the discovery of previously unknown knowledge. Evaluated with respect to known knowledge, an uninformed (unsupervised) method will easily be outperformed by other supervised methods, while in a typical KDD task, supervised methods cannot be used due to the unavailability of training data.

1.6.2 Optimization

Machine learning also has intimate ties to optimization: many learning problems are formulated as minimization of some loss function on a training set of examples. Loss functions express the discrepancy between the predictions of the model being trained and the actual problem instances (for example, in classification, one wants to assign a label to instances, and models are trained to correctly predict the pre-assigned labels of a set of examples).

1.6.3 Generalization

The difference between optimization and machine learning arises from the goal of generalization: while optimization algorithms can minimize the loss on a training set, machine

learning is concerned with minimizing the loss on unseen samples. Characterizing the generalization of various learning algorithms is an active topic of current research, especially for deep learning algorithms.

1.6.7 Statistics

Machine learning and statistics are closely related fields in terms of methods, but distinct in their principal goal: statistics draws population inferences from a sample, while machine learning finds generalizable predictive patterns. According to Michael I. Jordan, the ideas of machine learning, from methodological principles to theoretical tools, have had a long pre-history in statistics. He also suggested the term data science as a placeholder to call the overall field.

Leo Bierman distinguished two statistical modeling paradigms: data model and algorithmic model, wherein "algorithmic model" means more or less the machine learning algorithms like Random forest.

Some statisticians have adopted methods from machine learning, leading to a combined field that they call statistical learning.

1.6.8 Theory

A core objective of a learner is to generalize from its experience. Generalization in this context is the ability of a learning machine to perform accurately on new, unseen examples/tasks after having experienced a learning data set. The training examples come from some generally unknown probability distribution (considered representative of the space of occurrences) and the learner has to build a general model about this space that enables it to produce sufficiently accurate predictions in new cases.

The computational analysis of machine learning algorithms and their performance is a branch of theoretical computer science known as computational learning theory. Because training sets are finite and the future is uncertain, learning theory usually does not yield guarantees of the performance of algorithms. Instead, probabilistic bounds on the performance are quite common. The bias–variance decomposition is one way to quantify generalization error.

For the best performance in the context of generalization, the complexity of the hypothesis should match the complexity of the function underlying the data. If the hypothesis is less complex than the function, then the model has under fitted the data. If the complexity of the model is increased in response, then the training error decreases. But if the hypothesis is too complex, then the model is subject to overfitting and generalization will be poorer.

In addition to performance bounds, learning theorists study the time complexity and feasibility of learning. In computational learning theory, a computation is considered feasible if it can be done in polynomial time. There are two kinds of time complexity results. Positive results show that a certain class of functions can be learned in polynomial time. Negative results show that certain classes cannot be learned in polynomial time.

1.7 Approaches

Types of learning algorithms

The types of machine learning algorithms differ in their approach, the type of data they input and output, and the type of task or problem that they are intended to solve.

1.7.1 Supervised learning

A support vector machine is a supervised learning model that divides the data into regions separated by a linear boundary. Here, the linear boundary divides the black circles from the white.
Supervised learning algorithms build a mathematical model of a set of data that contains both the inputs and the desired outputs. The data is known as training data, and consists of a set of training examples. Each training example has one or more inputs and the desired output, also known as a supervisory signal. In the mathematical model, each training example is represented by an array or vector, sometimes called a feature vector, and the training data is represented by a matrix. Through iterative optimization of an objective function, supervised learning algorithms learn a function that can be used to predict the output associated with new inputs. An optimal function will allow the algorithm to correctly determine the output for inputs that were not a part of the training data. An algorithm that improves the accuracy of its outputs or predictions over time is said to have learned to perform that task.

Types of supervised learning algorithms include active learning, classification and regression. Classification algorithms are used when the outputs are restricted to a limited set of values, and regression algorithms are used when the outputs may have any numerical value within a range. As an example, for a classification algorithm that filters emails, the input would be an incoming email, and the output would be the name of the folder in which to file the email.

Similarity learning is an area of supervised machine learning closely related to regression and classification, but the goal is to learn from examples using a similarity function that measures how similar or related two objects are. It has applications in ranking, recommendation systems, visual identity tracking, face verification, and speaker verification.

1.7.2 Unsupervised learning

Unsupervised learning algorithms take a set of data that contains only inputs, and find structure in the data, like grouping or clustering of data points. The algorithms, therefore, learn from test data that has not been labeled, classified or categorized. Instead of responding to feedback, unsupervised learning algorithms identify commonalities in the data and react based on the presence or absence of such commonalities in each new piece of data. A central application of unsupervised learning is in the field of density estimation in statistics, such as finding the probability density function. Though unsupervised learning encompasses other domains involving summarizing and explaining data features.

Cluster analysis is the assignment of a set of observations into subsets (called clusters) so that observations within the same cluster are similar according to one or more predesignated criteria, while observations drawn from different clusters are dissimilar. Different clustering techniques make different assumptions on the structure of the data, often defined by some similarity metric and evaluated, for example, by internal compactness, or the similarity between members of the same cluster, and separation, the difference between clusters. Other methods are based on estimated density and graph connectivity.

1.7.3 Semi-supervised learning

Semi-supervised learning falls between unsupervised learning (without any labeled training data) and supervised learning (with completely labeled training data). Some of the training examples are missing training labels, yet many machine-learning researchers have found that unlabeled data, when used in conjunction with a small amount of labeled data, can produce a considerable improvement in learning accuracy.

In weakly supervised learning, the training labels are noisy, limited, or imprecise; however, these labels are often cheaper to obtain, resulting in larger effective training sets.

1.7.4 Reinforcement learning

Reinforcement learning is an area of machine learning concerned with how software agents ought to take actions in an environment so as to maximize some notion of cumulative reward. Due to its generality, the field is studied in many other disciplines, such as game theory, control theory, operations research, information theory, simulation-based optimization, multi-agent systems, swarm intelligence, statistics and genetic algorithms. In machine learning, the environment is typically represented as a Markov decision process (MDP). Many reinforcement learning algorithms use dynamic programming techniques. Reinforcement learning algorithms do not assume knowledge of an exact mathematical model of the MDP, and are used when exact models are infeasible. Reinforcement learning algorithms are used in autonomous vehicles or in learning to play a game against a human opponent.

1.7.5 Self learning

Self-learning as a machine learning paradigm was introduced in 1982 along with a neural network capable of self-learning named crossbar adaptive array (CAA). It is a learning with no external rewards and no external teacher advice. The CAA self-learning algorithm computes, in a crossbar fashion, both decisions about actions and emotions (feelings) about consequence situations. The system is driven by the interaction between cognition and emotion. The self-learning algorithm updates a memory matrix W =||w(a,s)|| such that in each iteration executes the following machine learning routine:

In situation s perform an action a;
Receive consequence situation s';
Compute emotion of being in consequence situation v(s');
Update crossbar memory w'(a,s) = w(a,s) + v(s').
It is a system with only one input, situation s, and only one output, action (or behavior) a. There is neither a separate reinforcement input nor an advice input from the environment. The backpropagated value (secondary reinforcement) is the emotion toward the consequence situation. The CAA exists in two environments, one is the behavioral environment where it behaves, and the other is the genetic environment, wherefrom it initially and only once receives initial emotions about situations to be encountered in the behavioral environment. After receiving the genome (species) vector from the genetic environment, the CAA learns a goal-seeking behavior, in an environment that contains both desirable and undesirable situations.

1.7.8 Feature learning

Several learning algorithms aim at discovering better representations of the inputs provided during training. Classic examples include principal components analysis and cluster analysis. Feature learning algorithms, also called representation learning algorithms, often attempt to preserve the information in their input but also transform it in a way that makes it useful, often as a pre-processing step before performing classification or predictions. This technique allows reconstruction of the inputs coming from the unknown data-generating distribution, while not being necessarily faithful to configurations that are implausible under that distribution. This replaces manual feature engineering, and allows a machine to both learn the features and use them to perform a specific task.

Feature learning can be either supervised or unsupervised. In supervised feature learning, features are learned using labeled input data. Examples include artificial neural networks, multilayer perceptron's, and supervised dictionary learning. In unsupervised feature learning, features are learned with unlabeled input data. Examples include dictionary learning, independent component analysis, autoencoders, matrix factorization and various forms of clustering.

Manifold learning algorithms attempt to do so under the constraint that the learned representation is low-dimensional. Sparse coding algorithms attempt to do so under the constraint that the learned representation is sparse, meaning that the mathematical model has many zeros. Multilinear subspace learning algorithms aim to learn low-dimensional representations directly from tensor representations for multidimensional data, without reshaping them into higher-dimensional vectors. Deep learning algorithms discover multiple levels of representation, or a hierarchy of features, with higher-level, more abstract features defined in terms of (or generating) lower-level features. It has been argued that an intelligent machine is one that learns a representation that disentangles the underlying factors of variation that explain the observed data.

Feature learning is motivated by the fact that machine learning tasks such as classification often require input that is mathematically and computationally convenient to process. However, real-world data such as images, video, and sensory data has not yielded to attempts to algorithmically define specific features. An alternative is to discover such features or representations thorough examination, without relying on explicit algorithms.

1.7.9 Sparse dictionary learning

Sparse dictionary learning is a feature learning method where a training example is represented as a linear combination of basis functions, and is assumed to be a sparse matrix. The method is strongly NP-hard and difficult to solve approximately. A popular heuristic method for sparse dictionary learning is the K-SVD algorithm. Sparse dictionary learning has been applied in several contexts. In classification, the problem is to determine the class to which a previously unseen training example belongs. For a dictionary where each class has already been built, a new training example is associated with the class that is best sparsely represented by the corresponding dictionary. Sparse dictionary learning has also been applied in image de-noising. The key idea is that a clean image patch can be sparsely represented by an image dictionary, but the noise cannot.

Anomaly detection

In data mining, anomaly detection, also known as outlier detection, is the identification of rare items, events or observations which raise suspicions by differing significantly from the majority of the data. Typically, the anomalous items represent an issue such as bank fraud, a structural defect, medical problems or errors in a text. Anomalies are referred to as outliers, novelties, noise, deviations and exceptions.

In particular, in the context of abuse and network intrusion detection, the interesting objects are often not rare objects, but unexpected bursts of inactivity. This pattern does not adhere to the common statistical definition of an outlier as a rare object, and many outlier detection methods (in particular, unsupervised algorithms) will fail on such data unless it has been

aggregated appropriately. Instead, a cluster analysis algorithm may be able to detect the micro-clusters formed by these patterns.

Three broad categories of anomaly detection techniques exist. Unsupervised anomaly detection techniques detect anomalies in an unlabeled test data set under the assumption that the majority of the instances in the data set are normal, by looking for instances that seem to fit least to the remainder of the data set. Supervised anomaly detection techniques require a data set that has been labeled as "normal" and "abnormal" and involves training a classifier (the key difference to many other statistical classification problems is the inherently unbalanced nature of outlier detection). Semi-supervised anomaly detection techniques construct a model representing normal behavior from a given normal training data set and then test the likelihood of a test instance to be generated by the model.

Robot learning

In developmental robotics, robot learning algorithms generate their own sequences of learning experiences, also known as a curriculum, to cumulatively acquire new skills through self-guided exploration and social interaction with humans. These robots use guidance mechanisms such as active learning, maturation, motor synergies and imitation.

Association rules

Association rule learning is a rule-based machine learning method for discovering relationships between variables in large databases. It is intended to identify strong rules discovered in databases using some measure of "interestingness".

Rule-based machine learning is a general term for any machine learning method that identifies, learns, or evolves "rules" to store, manipulate or apply knowledge. The defining characteristic of a rule-based machine learning algorithm is the identification and utilization of a set of relational rules that collectively represent the knowledge captured by the system. This is in contrast to other machine learning algorithms that commonly identify a singular model that can be universally applied to any instance in order to make a prediction. Rule-based machine learning approaches include learning classifier systems, association rule learning, and artificial immune systems.

Based on the concept of strong rules, Rakesh Agrawal, Tomasz Malinski and Arun Swami introduced association rules for discovering regularities between products in large-scale transaction data recorded by point-of-sale (POS) systems in supermarkets. For example, the rule {\display style\{\mahram {onions, potatoes\{Rightarrow\{mahram{burger}\}\}\{{\mahram{onions,potatoes}}\}\Right arrow\{{\mahram{burger}}\} found in the sales data of a supermarket would indicate that if a customer buys onions and potatoes together, they are likely to also buy hamburger meat. Such information can be used as the basis for decisions about marketing activities such as

promotional pricing or product placements. In addition to market basket analysis, association rules are employed today in application areas including Web usage mining, intrusion detection, continuous production, and bioinformatics. In contrast with sequence mining, association rule learning typically does not consider the order of items either within a transaction or across transactions.

Learning classifier systems (LCS) are a family of rule-based machine learning algorithms that combine a discovery component, typically a genetic algorithm, with a learning component, performing either supervised learning, reinforcement learning, or unsupervised learning. They seek to identify a set of context-dependent rules that collectively store and apply knowledge in a piecewise manner in order to make predictions.

Inductive logic programming (ILP) is an approach to rule-learning using logic programming as a uniform representation for input examples, background knowledge, and hypotheses. Given an encoding of the known background knowledge and a set of examples represented as a logical database of facts, an ILP system will derive a hypothesized logic program that entails all positive and no negative examples. Inductive programming is a related field that considers any kind of programming language for representing hypotheses (and not only logic programming), such as functional programs.

Inductive logic programming is particularly useful in bioinformatics and natural language processing. Gordon Plotkin and Ehud Shapiro laid the initial theoretical foundation for inductive machine learning in a logical setting. Shapiro built their first implementation (Model Inference System) in 1981: a Prolog program that inductively inferred logic programs from positive and negative examples. The term inductive here refers to philosophical induction, suggesting a theory to explain observed facts, rather than mathematical induction, proving a property for all members of a well-ordered set.

1.8 Models

Performing machine learning involves creating a model, which is trained on some training data and then can process additional data to make predictions. Various types of models have been used and researched for machine learning systems.

Artificial neural networks

An artificial neural network is an interconnected group of nodes, akin to the vast network of neurons in a brain. Here, each circular node represents an artificial neuron and an arrow represents a connection from the output of one artificial neuron to the input of another.
Artificial neural networks (ANNs), or connectionist systems, are computing systems vaguely inspired by the biological neural networks that constitute animal brains. Such systems "learn"

to perform tasks by considering examples, generally without being programmed with any task-specific rules.

An ANN is a model based on a collection of connected units or nodes called "artificial neurons", which loosely model the neurons in a biological brain. Each connection, like the synapses in a biological brain, can transmit information, a "signal", from one artificial neuron to another. An artificial neuron that receives a signal can process it and then signal additional artificial neurons connected to it. In common ANN implementations, the signal at a connection between artificial neurons is a real number, and the output of each artificial neuron is computed by some non-linear function of the sum of its inputs. The connections between artificial neurons are called "edges". Artificial neurons and edges typically have a weight that adjusts as learning proceeds. The weight increases or decreases the strength of the signal at a connection. Artificial neurons may have a threshold such that the signal is only sent if the aggregate signal crosses that threshold. Typically, artificial neurons are aggregated into layers. Different layers may perform different kinds of transformations on their inputs. Signals travel from the first layer (the input layer) to the last layer (the output layer), possibly after traversing the layers multiple times.

The original goal of the ANN approach was to solve problems in the same way that a human brain would. However, over time, attention moved to performing specific tasks, leading to deviations from biology. Artificial neural networks have been used on a variety of tasks, including computer vision, speech recognition, machine translation, social network filtering, playing board and video games and medical diagnosis.

Deep learning consists of multiple hidden layers in an artificial neural network. This approach tries to model the way the human brain processes light and sound into vision and hearing. Some successful applications of deep learning are computer vision and speech recognition.

Decision trees

Decision tree learning uses a decision tree as a predictive model to go from observations about an item (represented in the branches) to conclusions about the item's target value (represented in the leaves). It is one of the predictive modeling approaches used in statistics, data mining, and machine learning. Tree models where the target variable can take a discrete set of values are called classification trees; in these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels. Decision trees where the target variable can take continuous values (typically real numbers) are called regression trees. In decision analysis, a decision tree can be used to visually and explicitly represent decisions and decision making. In data mining, a decision tree describes data, but the resulting classification tree can be an input for decision making.

Support vector machines

Support vector machines (SVMs), also known as support vector networks, are a set of related supervised learning methods used for classification and regression. Given a set of training examples, each marked as belonging to one of two categories, an SVM training algorithm builds a model that predicts whether a new example falls into one category or the other. An SVM training algorithm is a non-probabilistic, binary, linear classifier, although methods such as Platt scaling exist to use SVM in a probabilistic classification setting. In addition to performing linear classification, SVMs can efficiently perform a non-linear classification using what is called the kernel trick, implicitly mapping their inputs into high-dimensional feature spaces.

Regression analysis

Regression analysis encompasses a large variety of statistical methods to estimate the relationship between input variables and their associated features. Its most common form is linear regression, where a single line is drawn to best fit the given data according to a mathematical criterion such as ordinary least squares. The latter is often extended by regularization (mathematics) methods to mitigate overfitting and bias, as in ridge regression. When dealing with non-linear problems, go-to models include polynomial regression (for example, used for trendline fitting in Microsoft Excel, logistic regression (often used in statistical classification) or even kernel regression, which introduces non-linearity by taking advantage of the kernel trick to implicitly map input variables to higher-dimensional space.

Bayesian networks

A simple Bayesian network. Rain influences whether the sprinkler is activated, and both rain and the sprinkler influence whether the grass is wet.
A Bayesian network, belief network, or directed acyclic graphical model is a probabilistic graphical model that represents a set of random variables and their conditional independence with a directed acyclic graph (DAG). For example, a Bayesian network could represent the probabilistic relationships between diseases and symptoms. Given symptoms, the network can be used to compute the probabilities of the presence of various diseases. Efficient algorithms exist that perform inference and learning. Bayesian networks that model sequences of variables, like speech signals or protein sequences, are called dynamic Bayesian networks. Generalizations of Bayesian networks that can represent and solve decision problems under uncertainty are called influence diagrams.

Genetic algorithms

A genetic algorithm (GA) is a search algorithm and heuristic technique that mimics the process of natural selection, using methods such as mutation and crossover to generate new genotypes in the hope of finding good solutions to a given problem. In machine learning, genetic

algorithms were used in the 1980s and 1990s. Conversely, machine learning techniques have been used to improve the performance of genetic and evolutionary algorithms.

Training models

Usually, machine learning models require a lot of data in order for them to perform well. Usually, when training a machine learning model, one needs to collect a large, representative sample of data from a training set. Data from the training set can be as varied as a corpus of text, a collection of images, and data collected from individual users of a service. Overfitting is something to watch out for when training a machine learning model. Trained models derived from biased data can result in skewed or undesired predictions. Algorithmic bias is a potential result from data not fully prepared for training.

Federated learning

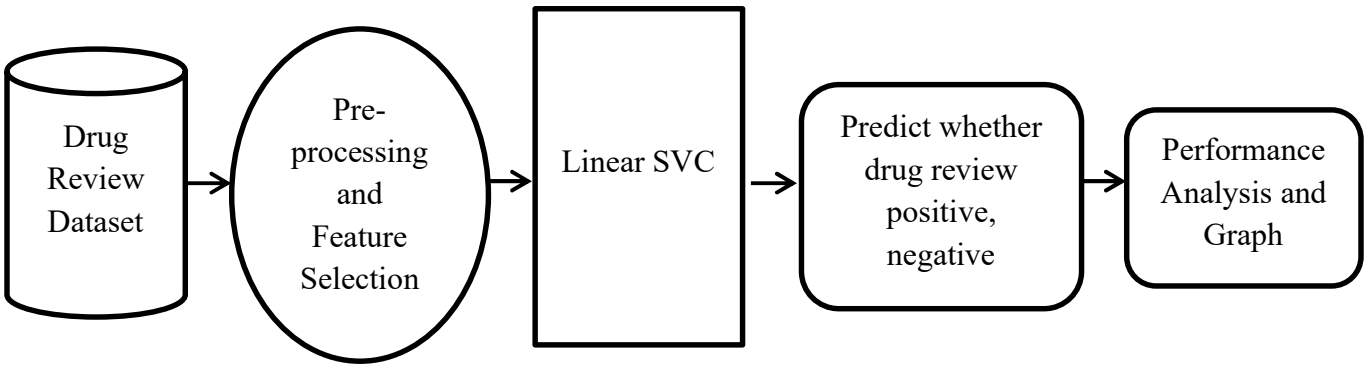
Federated learning is an adapted form of distributed artificial intelligence to training machine learning models that decentralizes the training process, allowing for users' privacy to be maintained by not needing to send their data to a centralized server. This also increases efficiency by decentralizing the training process to many devices. For example, Aboard uses federated machine learning to train search query prediction models on users' mobile phones without having to send individual searches back to Google

2. SYSTEM DESIGN

Chapter 2:

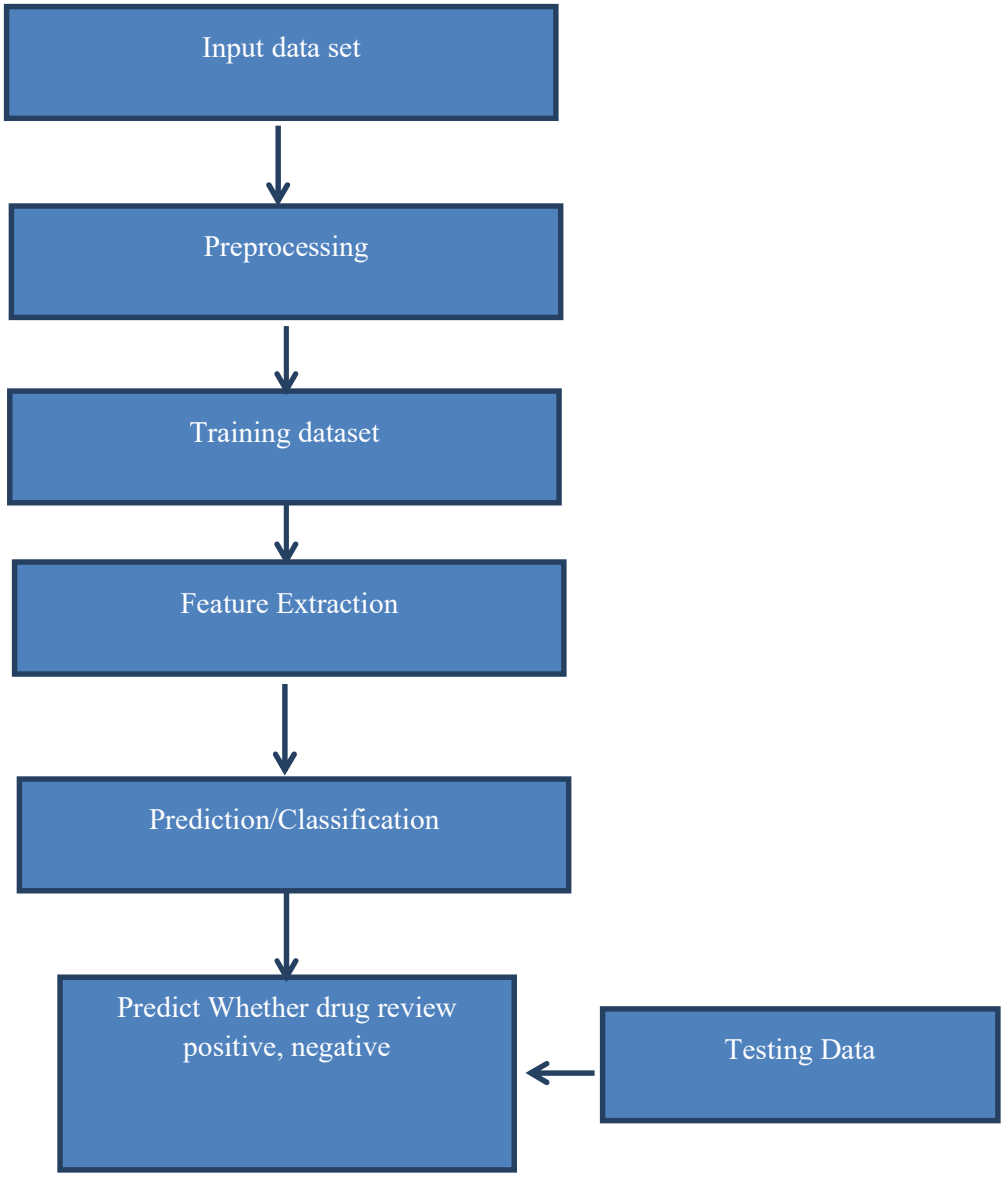
SYSTEM DESIGN

2.1 SYSTEM ARCHITECTURE:



2.1.2 DATA FLOW DIAGRAM:

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



2.1.3 UML DIAGRAMS:

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling 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 Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.
The UML is a very important part of developing object-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

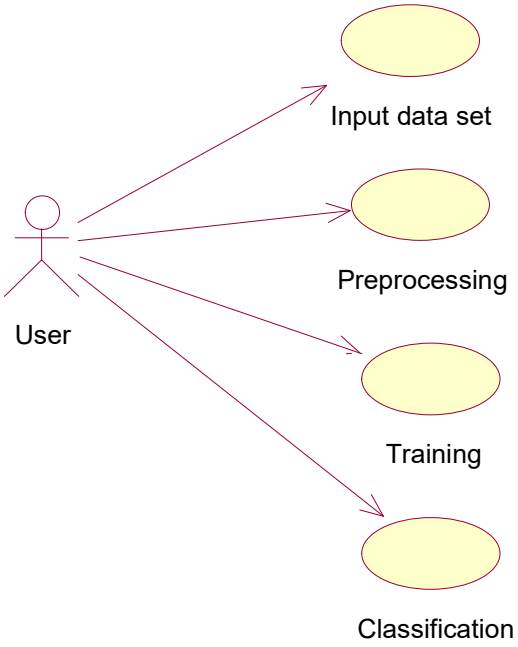
GOALS:

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

1. Provide users a ready-to-use, expressive visual modeling 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 modeling 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.

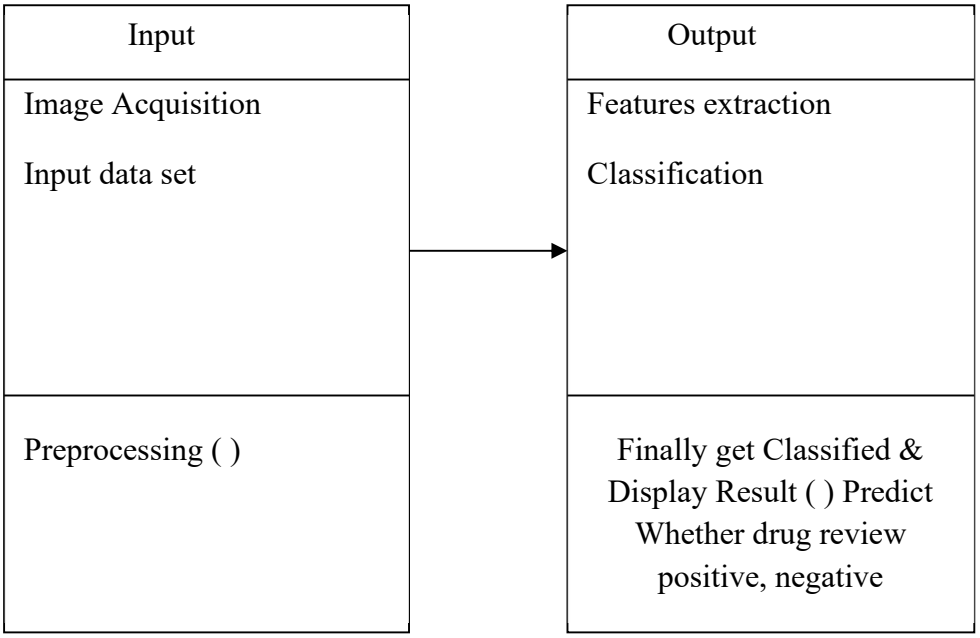
2.1.4 USE CASE DIAGRAM:

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.



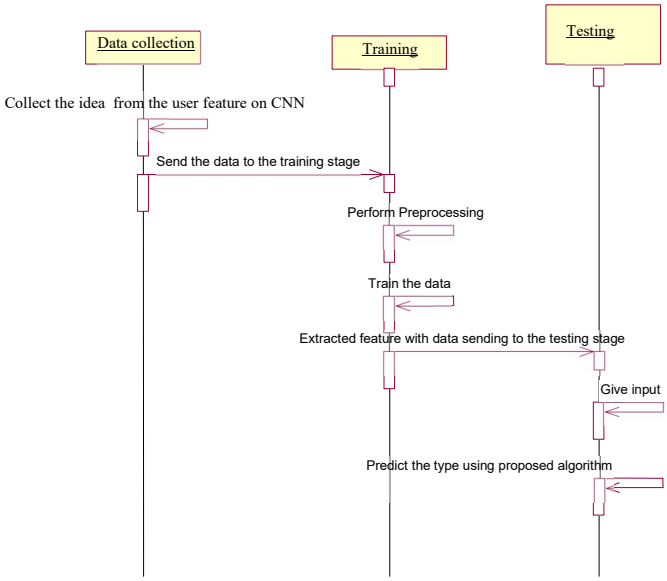
2.1.5 CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling 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.



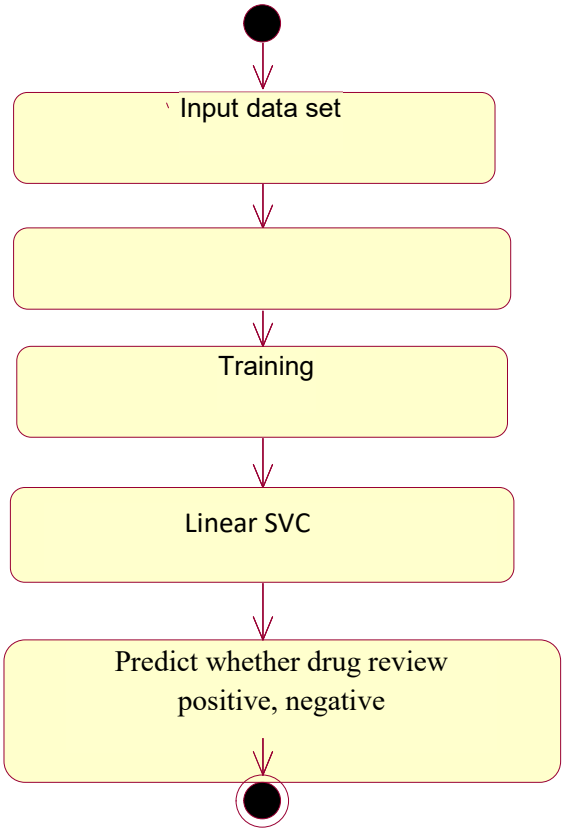
2.1.6 SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling 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.



2.1.7 ACTIVITY DIAGRAM:

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling 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.



2.2 INPUT DESIGN AND OUTPUT DESIGN

2.2.1 INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

2.2.2 OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- Convey information about past activities, current status or projections of the
- Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

3. SYSTEM ANALYSIS

Chapter 3:

SYSTEM ANALYSIS

3.1 EXISTING SYSTEM:

- Xiaohong Jiang et al. examined three distinct algorithms, decision tree algorithm, support vector machine (SVM), and backpropagation neural network on treatment data. SVM was picked for the medication proposal module as it performed truly well in each of the three unique boundaries - model exactness, model proficiency, model versatility. Additionally, proposed the mistake check system to ensure analysis, precision and administration quality.
- Mohammad Mehedi Hassan et al. developed a cloud assisted drug proposal (CADRE). As per patients' side effects, CADRE can suggest drugs with top-N related prescriptions. This proposed framework was initially founded on collaborative filtering techniques in which the medications are initially bunched into clusters as indicated by the functional description data. However, after considering its weaknesses like computationally costly, cold start, and information sparsity, the model is shifted to a cloud-helped approach using tensor decomposition for advancing the quality of experience of medication suggestion.

3.1.1 DISADVANTAGES OF EXISTING SYSTEM:

- Most of the recommender technologies are applied to the e-government area, e-business area, e-commerce/e-shopping area, e-learning area, e-tourism area and so on. However, medicine area includes rare recommender technologies, and this will focus on the design of the medicine recommender system and mining knowledge from medical case data. Commonly used recommendation techniques include collaborative filtering (CF), content-based (CB), knowledge-based (KB) techniques and hybrid recommendation technologies.
- CB mainly generates recommendations by using traditional retrieval methods and machine learning methods, but CB has overspecialized recommendations.
- Collaborative filtering (CF) based recommendation techniques help people to make choices based on the opinions of other people who share similar interests, while CF has sparseness, scalability and cold-start problems.
- Knowledge-based (KB) recommendation offers items to users based on knowledge about the users, items and/or their relationships. Usually, KB recommendations retain a functional knowledge base that describes how a particular item meets a specific user's need.

3.2 PROPOSED SYSTEM:

- A recommender framework is a customary system that proposes an item to the user, dependent on their advantage and necessity. These frameworks employ the customers' surveys to break down their sentiment and suggest a recommendation for their exact need. In the drug recommender system, medicine is offered on a specific condition dependent on patient reviews using sentiment analysis and feature engineering. Sentiment analysis is a progression of strategies, methods, and tools for distinguishing and extracting emotional data, such as opinion and attitudes, from language.
- The dataset used in this research is Drug Review Dataset (Drugs.com) taken from the UCI ML repository. This dataset contains the following attributes, name of drug used (text), review (text) of a patient, condition (text) of a patient, useful count (numerical) which suggest the number of individuals who found the review helpful, date (date) of review entry, and a 10-star patient rating (numerical) determining overall patient contentment.
- In this work, each review was classified as positive or negative, depending on the user's star rating. Ratings above five are classified as positive, while negative ratings are from one to five-star ratings. Linear SVC was picked as the best algorithm since the accuracy that we achieved is Train Accuracy: 0.903 & Test Accuracy: 0.8369 which is greater than all other existing systems.

3.2.1 ADVANTAGES OF PROPOSED SYSTEM:

- Recommender framework has gotten to be a profitable investigation field as the advancement of counterfeit brilliantly advances. Not at all like most current recommender frameworks that specialize in e-business, book and pictorial suggestion, our framework points at giving a virtual fully fledged specialist for unpracticed amateurs and patients in abuse right pharmaceutical. Since high accuracy and strength is vital for such an online pharmaceutical recommender framework, in this way we tend to evaluate a few information preparing approaches to induce an genuine trade-off among the precision, productivity and quantifiability.
- In the recommendation framework, we simply just added the best-predicted result of each method. For better results and understanding, require a proper assembling of different predicted results. This paper intends to show only the methodology that one can use to extract sentiment from the data and perform classification to build a recommender system.
- The proposed system gives faster predictions.

3.3 SYSTEM STUDY

3.3.1 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

3.3.2 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.

3.3.3 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.

3.3.4 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.

4. SYSTEM REQUIREMENTS

Chapter 4:

SYSTEM REQUIREMENTS

HARDWARE REQUIREMENTS:

- Device name : LAPTOP-LVDIMT80
- Processor : Intel(R) Core(TM) i3-1005G1 CPU @ 1.20GHz 1.19 GHz
- Installed RAM : 8.00 GB (7.70 GB usable)
- System type : 64-bit operating system, x64-based processor
- Monitor : 14'' LED
- Input Devices : Keyboard, Mouse

SOFTWARE REQUIREMENTS:

- Operating system : Windows 11.
- Coding Language : Python
- Web Framework : Flask

5. SOFTWARE ENVIRONMENT

Chapter 5:

Software Environment

5.1 Python: 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 do not 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.

5.1.1 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, Small talk, 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.

5.1.2 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.

Python is available on a wide variety of platforms including Linux and Mac OS X. Let's understand how to set up our Python environment.

5.1.3 Getting Python

The most up-to-date and current source code, binaries, documentation, news, etc., is available on the official website of Python https://www.python.org.

Windows Installation

Here are the steps to install Python on Windows machine.

- Open a Web browser and go to https://www.python.org/downloads/.
- Follow the link for the Windows installer python-XYZ.msifile where XYZ is the version you need to install.
- To use this installer python-XYZ.msi, the Windows system must support Microsoft Installer 2.0. Save the installer file to your local machine and then run it to find out if your machine supports MSI.
- Run the downloaded file. This brings up the Python install wizard, which is really easy to use. Just accept the default settings, wait until the install is finished, and you are done.

The Python language has many similarities to Perl, C, and Java. However, there are some definite differences between the languages.

First Python Program

Let us execute programs in different modes of programming.

5.2 Interactive Mode Programming:

Invoking the interpreter without passing a script file as a parameter brings up the following prompt –

$ python

```
Python2.4.3(#1,Nov112010,13:34:43)
[GCC 4.1.220080704(RedHat4.1.2-48)] on linux2
Type"help","copyright","credits"or"license"for more information.
>>>
```

Type the following text at the Python prompt and press the Enter −

```
>>>print"Hello, Python!"
```

If you are running new version of Python, then you would need to use print statement with parenthesis as in **print ("Hello, Python!");**. However in Python version 2.4.3, this produces the following result −

```
Hello, Python!
```

5.3 Script Mode Programming:

Invoking the interpreter with a script parameter begins execution of the script and continues until the script is finished. When the script is finished, the interpreter is no longer active.

Let us write a simple Python program in a script. Python files have extension **.py**. Type the following source code in a test.py file −

```
print"Hello, Python!"
```

We assume that you have Python interpreter set in PATH variable. Now, try to run this program as follows −

```
$ python test.py
```

This produces the following result −

```
Hello, Python!
```

5.4 Flask Framework:

Flask is a web application framework written in Python. Armin Ronacher, who leads an international group of Python enthusiasts named Pocco, develops it. Flask is based on Werkzeug WSGI toolkit and Jinja2 template engine. Both are Pocco projects.

Http protocol is the foundation of data communication in world wide web. Different methods of data retrieval from specified URL are defined in this protocol.

The following table summarizes different http methods −

Sr.No	Methods & Description
1	**GET** Sends data in unencrypted form to the server. Most common method.
2	**HEAD** Same as GET, but without response body

3	**POST** Used to send HTML form data to server. Data received by POST method is not cached by server.
4	**PUT** Replaces all current representations of the target resource with the uploaded content.
5	**DELETE** Removes all current representations of the target resource given by a URL

By default, the Flask route responds to the **GET** requests. However, this preference can be altered by providing methods argument to **route()** decorator.

In order to demonstrate the use of **POST** method in URL routing, first let us create an HTML form and use the **POST** method to send form data to a URL.

Save the following script as login.html

```
<html>
<body>
<formaction="http://localhost:5000/login"method="post">
<p>Enter Name:</p>
<p><inputtype="text"name="nm"/></p>
<p><inputtype="submit"value="submit"/></p>
</form>
</body>
</html>
```

Now enter the following script in Python shell.

```
from flask importFlask, redirect,url_for, request
```

```
app=Flask(__name__)
@app.route('/success/<name>')
def success(name):
return'welcome %s'% name
@app.route('/login',methods=['POST','GET'])
def login():
ifrequest.method=='POST':
user=request.form['nm']
return redirect(url_for('success',name= user))
else:
user=request.args.get('nm')
return redirect(url_for('success',name= user))
if __name__ =='__main__':
app.run(debug =True)
```

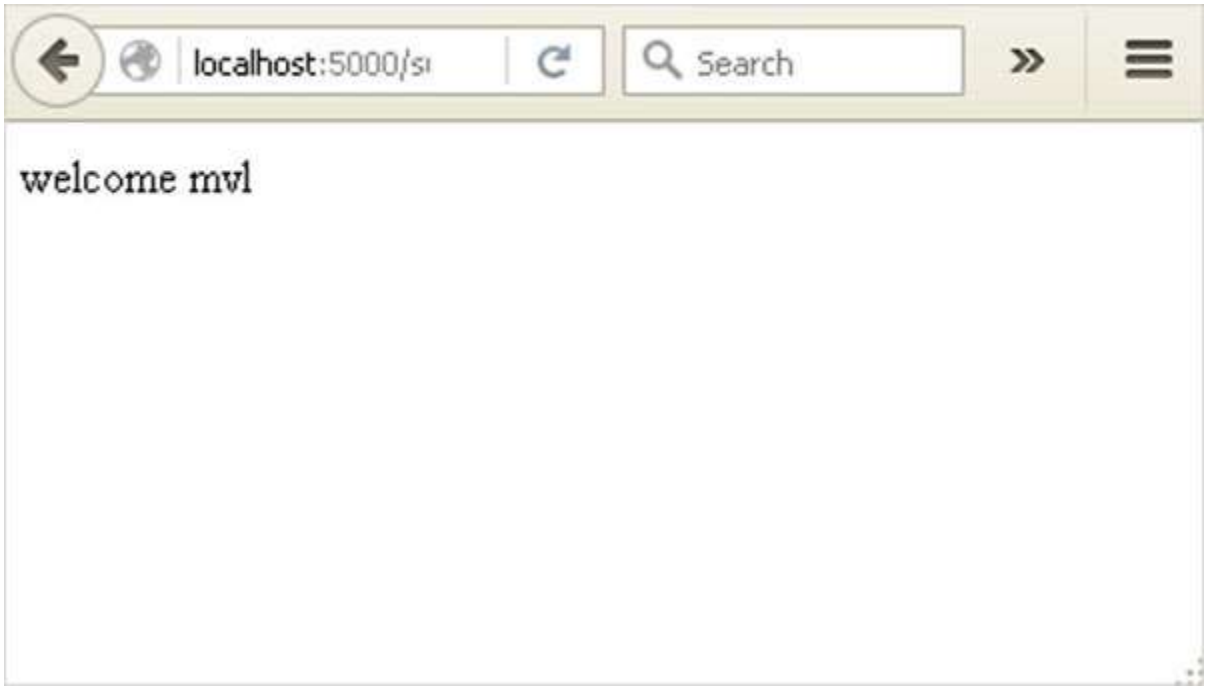
After the development server starts running, open **login.html** in the browser, enter name in the text field and click **Submit**.

Form data is POSTed to the URL in action clause of form tag.

http://localhost/login is mapped to the **login()** function. Since the server has received data by **POST** method, value of 'nm' parameter obtained from the form data is obtained by −

```
user = request.form['nm']
```

It is passed to **'/success'** URL as variable part. The browser displays a **welcome** message in the window.



Change the method parameter to **'GET'** in **login.html** and open it again in the browser. The data received on server is by the **GET** method. The value of 'nm' parameter is now obtained by −

```
User = request.args.get('nm')
```

Here, **args** is dictionary object containing a list of pairs of form parameter and its corresponding value. The value corresponding to 'nm' parameter is passed on to '/success' URL as before.

What is Python?

Python is a popular programming language. It was created in 1991 by Guido van Rossum.
It is used for:

- web development (server-side),
- software development,
- mathematics,
- system scripting.

What can Python do?

- Python can be used on a server to create web applications.
- Python can be used alongside software to create workflows.
- Python can connect to database systems. It can also read and modify files.
- Python can be used to handle big data and perform complex mathematics.
- Python can be used for rapid prototyping, or for production-ready software development.

Why Python?

- Python works on different platforms (Windows, Mac, Linux, Raspberry Pi, etc).
- Python has a simple syntax similar to the English language.
- Python has syntax that allows developers to write programs with fewer lines than some other programming languages.
- Python runs on an interpreter system, meaning that code can be executed as soon as it is written. This means that prototyping can be very quick.
- Python can be treated in a procedural way, an object-orientated way or a functional way.

Good to know

- The most recent major version of Python is Python 3, which we shall be using in this tutorial. However, Python 2, although not being updated with anything other than security updates, is still quite popular.
- In this tutorial Python will be written in a text editor. It is possible to write Python in an Integrated Development Environment, such as Thonny, PyCharm, NetBeans or Eclipse which are particularly useful when managing larger collections of Python files.

Python Syntax compared to other programming languages

- Python was designed to for readability, and has some similarities to the English language with influence from mathematics.
- Python uses new lines to complete a command, as opposed to other programming languages which often use semicolons or parentheses.
- Python relies on indentation, using whitespace, to define scope; such as the scope of loops, functions and classes. Other programming languages often use curly-brackets for this purpose.

Python Install

Many PCs and Macs will have python already installed.

To check if you have python installed on a Windows PC, search in the start bar for Python or run the following on the Command Line (cmd.exe):

```
C:\Users\Your Name>python --version
```

To check if you have python installed on a Linux or Mac, then on linux open the command line or on Mac open the Terminal and type:

```
python --version
```

If you find that you do not have python installed on your computer, then you can download it for free from the following website: https://www.python.org/

Python QuickStart

Python is an interpreted programming language, this means that as a developer you write Python (.py) files in a text editor and then put those files into the python interpreter to be executed.

The way to run a python file is like this on the command line:

```
C:\Users\Your Name>python helloworld.py
```

Where "helloworld.py" is the name of your python file.

Let's write our first Python file, called helloworld.py, which can be done in any text editor.

helloworld.py

```
print("Hello, World!")
```

Simple as that. Save your file. Open your command line, navigate to the directory where you saved your file, and run:

```
C:\Users\Your Name>python helloworld.py
```

The output should read:

```
Hello, World!
```

Congratulations, you have written and executed your first Python program.

The Python Command Line

To test a short amount of code in python sometimes it is quickest and easiest not to write the code in a file. This is made possible because Python can be run as a command line itself.

Type the following on the Windows, Mac or Linux command line:

```
C:\Users\Your Name>python
```

From there you can write any python, including our hello world example from earlier in the tutorial:

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, World!")
```

Which will write "Hello, World!" in the command line:

```
C:\Users\Your Name>python
Python 3.6.4 (v3.6.4:d48eceb, Dec 19 2017, 06:04:45) [MSC v.1900 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello, World!")
Hello, World!
```

Whenever you are done in the python command line, you can simply type the following to quit the python command line interface:

```
exit()
```

Execute Python Syntax

As we learned in the previous page, Python syntax can be executed by writing directly in the Command Line:

```
>>> print("Hello, World!")
Hello, World!
```

Or by creating a python file on the server, using the .py file extension, and running it in the Command Line:

```
C:\Users\Your Name>python myfile.py
```

Python Indentations

Where in other programming languages the indentation in code is for readability only, in Python the indentation is very important.

Python uses indentation to indicate a block of code.

Example

```
if 5 > 2:
  print("Five is greater than two!")
```

Python will give you an error if you skip the indentation:

Example
if 5 > 2:
print("Five is greater than two!")

Comments

Python has commenting capability for the purpose of in-code documentation.
Comments start with a #, and Python will render the rest of the line as a comment:

Example
Comments in Python:
#This is a comment.
print("Hello, World!")
Docstrings
Python also has extended documentation capability, called docstrings.
Docstrings can be one line, or multiline.
Python uses triple quotes at the beginning and end of the docstring:

Example
Docstrings are also comments:
"""This is a multiline docstring."""
print("Hello, World!")

6. IMPLEMENTATION

Chapter 6:

IMPLEMENTATION

6.1 MODULES:

- Data Collection
- Dataset
- Data Preparation
- Model Selection
- Analyze and Prediction
- Accuracy on test set
- Saving the Trained Model
- Database connecting using My SQL

MODULES DESCSRIPTION:

Data Collection:

This is the first real step towards the real development of a machine learning model, collecting data. This is a critical step that will cascade in how good the model will be, the more and better data that we get, the better our model will perform.
There are several techniques to collect the data, like web scraping, manual interventions and etc.
Drug Recommendation System based on Sentiment Analysis of Drug Reviews using Machine Learning
Data set Link: https://www.kaggle.com/jessicali9530/kuc-hackathon-winter-2018

Dataset:

The train dataset consists of 161297 and test dataset consists of 53766 There are 3 columns in the dataset, which are described below
Index: unique id
Drug Name: Name of drug used
Condition: Condition of a patient
Review : Review of a patient
Rating : 1 to 10
Date : The day, month, or year
Useful Count: review count

Data Preparation:

We will transform the data. by getting rid of missing data and removing some columns. First we will create a list of column names that we want to keep or retain.
Next we drop or remove all columns except for the columns that we want to retain.
Finally we drop or remove the rows that have missing values from the data set.

TF-IDF Vectorizer transformer:

TF-IDF is an abbreviation for Term Frequency Inverse Document Frequency. This is very common algorithm to transform text into a meaningful representation of numbers which is used to fit machine algorithm for prediction.
Term Frequency (TF)
It is a measure of the frequency of a word (w) in a document (d). TF is defined as the ratio of a word's occurrence in a document to the total number of words in a document. The denominator term in the formula is to normalize since all the corpus documents are of different lengths.

$$TF(w,d) = \frac{occurences\ of\ w\ in\ document\ d}{total\ number\ of\ words\ in\ document\ d}$$

Image by Author
EXAMPLE:

Documents	Text	Total number of words in a document
A	Jupiter is the largest planet	5
B	Mars is the fourth planet from the sun	8

Image by Author
The initial step is to make a vocabulary of unique words and calculate TF for each document. TF will be more for words that frequently appear in a document and less for rare words in a document.

Words	TF (for A)	TF (for B)
Jupiter	1/5	0
Is	1/5	1/8
The	1/5	2/8
largest	1/5	0
Planet	1/5	1/8
Mars	0	1/8
Fourth	0	1/8
From	0	1/8
Sun	0	1/8

Image by Author

Inverse Document Frequency (IDF)

It is the measure of the importance of a word. Term frequency (TF) does not consider the importance of words. Some words such as' of', 'and', etc. can be most frequently present but are of little significance. IDF provides weightage to each word based on its frequency in the corpus D.

IDF of a word (w) is defined as

$$IDF(w, D) = \ln\left(\frac{Total\ number\ of\ documents\ (N)\ in\ corpus\ D}{number\ of\ documents\ containing\ w}\right)$$

Image by Author

In our example, since we have two documents in the corpus, N=2.

Words	TF (for A)	TF (for B)	IDF
Jupiter	1/5	0	ln(2/1) = 0.69
Is	1/5	1/8	ln(2/2) = 0
The	1/5	2/8	ln(2/2) = 0
largest	1/5	0	ln(2/1) = 0.69
Planet	1/5	1/8	ln(2/2) = 0
Mars	0	1/8	ln(2/1) = 0.69
Fourth	0	1/8	ln(2/1) = 0.69
From	0	1/8	ln(2/1) = 0.69
Sun	0	1/8	ln(2/1) = 0.69

Image by Author

Term Frequency — Inverse Document Frequency (TFIDF)

It is the product of TF and IDF.

TFIDF gives more weightage to the word that is rare in the corpus (all the documents).

TFIDF provides more importance to the word that is more frequent in the document.

$$TFIDF\ (w, d, D) = TF(w, d) * IDF(w, D)$$

Image by Author

Words	TF (for A)	TF (for B)	IDF	TFIDF (A)	TFIDF (B)
Jupiter	1/5	0	ln(2/1) = 0.69	0.138	0
Is	1/5	1/8	ln(2/2) = 0	0	0
The	1/5	2/8	ln(2/2) = 0	0	0
largest	1/5	0	ln(2/1) = 0.69	0.138	0
Planet	1/5	1/8	ln(2/2) = 0	0.138	0
Mars	0	1/8	ln(2/1) = 0.69	0	0.086
Fourth	0	1/8	ln(2/1) = 0.69	0	0.086
From	0	1/8	ln(2/1) = 0.69	0	0.086
Sun	0	1/8	ln(2/1) = 0.69	0	0.086

After applying TFIDF, text in A and B documents can be represented as a TFIDF vector of dimension equal to the vocabulary words

Model Selection:

We used Linear SVC. The Linear Support Vector Classifier (SVC) method applies a linear kernel function to perform classification and it performs well with a large number of samples. If we compare it with the SVC model, the Linear SVC has additional parameters such as penalty normalization which applies 'L1' or 'L2' and loss function. The kernel method cannot be changed in linear SVC, because it is based on the kernel linear method.

Analyze and Prediction:

In the actual dataset, we chose only 2 features

1. **Review** : Review of a patient

2. **Labels** : Labels
   Positive
   Negative

Accuracy on test set:

We got a accuracy of 83.02% on test set.

Saving the Trained Model:

Once you're confident enough to take your trained and tested model into the production-ready environment, the first step is to save it into a .h5 or pkl file using a library like `pickle`. Make sure you have `pickle` installed in your environment.
Next, let's import the module and dump the model into. pkl file

Database connecting using My SQL:

So long as that works, do a quick control+d to exit the python instance.
Next, we want to make a Python file that can connect to the database. Generally, you will have a separate "connect" file, outside of any main files you may have. This is usually true across languages, and here's why. Initially, you may have just a simple __init__.py, or app.py, or whatever, and that file does all of your operations. What can happen in time, however, is that your website does other things. For example, with one of my websites, Sentdex.com, I perform a lot of analysis, store that analysis to a database, and I also operate a website for users to use. Generally, for tasks, you will use what is called a "cron." A cron is a scheduled task that runs when you program it to run. Generally this runs another file, almost certain to not be your

website's file. So then, to connect to a database, you'd have to write the database connecting code again in the file being run by your cron.
As time goes on, these sorts of needs stack up where you have some files modifying the database, but you still want the website to be able to access it, and maybe modify it too. Then, consider what might happen if you change your database password. You'd then need to go to every single file that connects to the database and change that too. So, usually, you will find the smartest thing to do is to just create one file, which houses the connection code.

Import the module:

Create a connection function to run our code. Here we specify where we're connecting to, the user, the user's password, and then the database that we want to connect to.
As a note, we use "localhost" as our host. This just means we'll use the same server that this code is running on. You can connect to databases remotely as well, which can be pretty neat. To do that, you would connect to a host by their IP, or their domain. To connect to a database remotely, you will need to first allow it from the remote database that will be accessed/modified.
Next, let's go ahead and edit our __init__.py file, adding a register function. For now we'll keep it simple, mostly just to test our connection functionality.
We allow for GET and POST, but aren't handling it just yet.
We're going to just try to run the imported connection function, which returns c and conn (cursor and connection objects).
If the connection is successful, we just have the page say okay, otherwise it will output the error.

6.2 Source code:

Admin:

```
<!DOCTYPE html>
<html lang="en">
<head>
   <title>Drug Recommendation </title>
   <meta charset="utf-8">
   <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no"><linkhref="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700&display=swap" rel="stylesheet">
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-awesome/4.7.0/css/font-awesome.min.css"> <link rel="stylesheet" href="../static/css/animate.css">
   <link rel="stylesheet" href="../static/css/owl.carousel.min.css">
   <link rel="stylesheet" href="../static/css/owl.theme.default.min.css">
   <link rel="stylesheet" href="../static/css/magnific-popup.css">
  <link rel="stylesheet" href="../static/css/flaticon.css">
   <link rel="stylesheet" href="../static/css/style.css">
         <style> #customers {font-family: "Trebuchet MS", Arial, Helvetica, sans-serif;
            font-size: 20px;
            border-collapse: collapse;
            width: 100%;}
         #customers td, #customers th {
            border: 1px solid #ddd;
            padding: 15px;}
         #customers tr:nth-child(even){background-color: #f2f2f2;}
         #customers tr:hover {background-color: #ddd;}
         #customers th {
            padding-top: 12px;
            padding-bottom: 12px;
            text-align: left;
```

```
            background-color: #1DA1F2;
            color: white;
        }
    </style>
</head>
<body><nav class="navbar navbar-expand-lg navbar-dark ftco_navbar bg-dark ftco-navbar-light" id="ftco-navbar">
        <div class="container">
            <a class="navbar-brand" href="index.html">Drug<span> Recommendation</span></a>
            <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#ftco-nav" aria-controls="ftco-nav" aria-expanded="false" aria-label="Toggle navigation">
                <span class="fa fa-bars"></span> Menu
            </button>
            <div class="collapse navbar-collapse" id="ftco-nav">
                <ul class="navbar-nav ml-auto">
                <li class="nav-item "><a href="{{ url_for('first')}}" class="nav-link">Home</a></li>
                            <li class="nav-item "><a href="{{ url_for('userdetail')}}" class="nav-link">Register details</a></li>
                <li class="nav-item "><a href="{{ url_for('admin')}}" class="nav-link">Review details</a></li>
                                <li class="nav-item "><a href="{{ url_for('postive')}}" class="nav-link">Positive_reviews</a></li>
                </ul>
            </div>
        </div>
    </nav>
    <!-- END nav -->
    <section class="hero-wrap hero-wrap-2" style="background-image: url('../static/images/bg_10.jpg');" data-stellar-background-ratio="0.5">
```

```
    <div class="overlay"></div>
    <div class="container">
      <div class="row no-gutters slider-text align-items-end">
        <div class="col-md-9 ftco-animate pb-5">
          <h1 class="mb-0 bread">Review details</h1>
        </div>
      </div>
    </div>
  </section><section class="ftco-section bg-light">
                        <div class="container">
                                <div class="row justify-content-center"
                                    <div class="section-title">
       <center><h2>Review details</h2></center>
                    <center><table id="customers" style="margin-right: 30>
<tr>
         <th>user_id</th><th>user_name</th><th>Email</th><th>age</th>
         <th>drug Name</th><th>Review</th><th> pred</th>
     </tr>
   {% for admin in userDetails %}
   <tr>
   <td> {{admin[0]}} </td> <td> {{admin[1]}} </td><td> {{admin[2]}} </td><td> {{admin[4]}} </td><td> {{admin[9]}} </td><td> {{admin[6]}} </td><td> {{admin[7]}} </td>
</tr>
{% endfor %}
 </table></center> </div></div></div></section>
<footer class="footer ftco-section">
<div class="container">
</div></footer>
```

```
<!-- loader -->

 <div id="ftco-loader" class="show fullscreen"><svg class="circular" width="48px" height="48px"><circle class="path-bg" cx="24" cy="24" r="22" fill="none" stroke-width="4" stroke="#eeeeee"/><circle class="path" cx="24" cy="24" r="22" fill="none" stroke-width="4" stroke-miterlimit="10" stroke="#F96D00"/></svg></div>

<script src="../static/js/jquery.min.js"></script>

  <script src="../static/js/jquery-migrate-3.0.1.min.js"></script>

  <script src="../static/js/popper.min.js"></script>

  <script src="../static/js/bootstrap.min.js"></script>

<script src="../static/js/jquery.easing.1.3.js"></script>

  <script src="../static/js/jquery.waypoints.min.js"></script>

  <script src="../static/js/jquery.stellar.min.js"></script>

  <script src="../static/js/jquery.animateNumber.min.js"></script>

  <script src="../static/js/owl.carousel.min.js"></script>

  <script src="../static/js/jquery.magnific-popup.min.js"></script>

  <script src="../static/js/scrollax.min.js"></script>

<script
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBVWaKrjvy3MaE7SQ74_uJiULgl1JY0H2s&sensor=false"></script>

<script src="../static/js/google-map.js"></script>

  <script src="../static/js/main.js"></script>

</body></html>
```

Login:

```
<!DOCTYPE html>

<html lang="en">

  <head>

    <title>Drug Recommendation </title>

    <meta charset="utf-8">

    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
```

```
  <link
href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700&display=swap"
rel="stylesheet">

  <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-
awesome/4.7.0/css/font-awesome.min.css">

  <link rel="stylesheet" href="../static/css/animate.css">

  <link rel="stylesheet" href="../static/css/owl.carousel.min.css">

  <link rel="stylesheet" href="../static/css/owl.theme.default.min.css">

  <link rel="stylesheet" href="../static/css/magnific-popup.css">

  <link rel="stylesheet" href="../static/css/flaticon.css">

  <link rel="stylesheet" href="../static/css/style.css">

 </head>

 <body>

       <nav class="navbar navbar-expand-lg navbar-dark ftco_navbar bg-dark ftco-navbar-
light" id="ftco-navbar">

          <div class="container">

              <a class="navbar-brand" href="index.html">Drug<span>
Recommendation</span></a>

            <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#ftco-nav" aria-controls="ftco-nav" aria-expanded="false" aria-label="Toggle
navigation">

              <span class="fa fa-bars"></span> Menu

            </button>

            <div class="collapse navbar-collapse" id="ftco-nav">

              <ul class="navbar-nav ml-auto">

          <li class="nav-item "><a href="{{ url_for('first')}}" class="nav-
link">Home</a></li>

              <li class="nav-item "><a href="{{ url_for('upload')}}" class="nav-
link">Upload</a></li>

                        <li class="nav-item active "><a href="{{ url_for('login')}}"
class="nav-link">Login</a></li>

              <li class="nav-item "><a href="{{ url_for('register')}}" class="nav-
link">Register</a></li>
```

```
            </ul></div></div></nav>
  <!-- END nav -->
  <section class="hero-wrap hero-wrap-2" style="background-image: url('../static/images/bg_4.jpg');" data-stellar-background-ratio="0.5">
    <div class="overlay"></div>
    <div class="container">
      <div class="row no-gutters slider-text align-items-end">
        <div class="col-md-9 ftco-animate pb-5">
          <h1 class="mb-0 bread">Login</h1>
        </div></div></div></section>
                <section class="ftco-section bg-light">
                    <div class="container">
                        <div class="row justify-content-center">
                          <body id="page-top">
 <main id="main">
  <!-- ======= What We Do Section ======= -->
    <div class="container"><div class="section-title"><div class="col-md-6 col-lg-4" style="margin-left:250px">
        <h2>Login</h2></div>
     <form action="{{ url_for('login') }}" method="post">
    <div class="row">
      <!-- Portfolio Item 1 -->
      <div class="col-md-6 col-lg-4" style="margin-left:200px">
                <div class="control-group">
                        <!-- Username -->
                        <div class="controls">
                        <b> Username </b>:<input type="text" name="username" placeholder="Username">
                        </div></div></br>
                        <div class="control-group">
```

```
                        <!-- Password-->
                          <div class="controls">
                          <b> Password </b>:  <input type="password" name="password" placeholder="Password"  required>
                        </div> </div>
                        <div class="control-group">
                        <!-- Button --> </br>
                        <div class="controls"><div class="msg">{{ msg }}</div>
                        <input type="submit" class="btn btn-primary" value="Login" style="margin-left: 60px">
                        </div></div>
 </div>
 </div>
 </form>
   {% with messages = get_flashed_messages() %}
 {% if messages %}
   <script>
     var messages = {{ messages | safe }};
     for (var i=0; i<messages.length; i++) {
       alert(messages[i]);
     }
   </script>
 {% endif %}
{% endwith %}
</body></div></div></section>
<footer class="footer ftco-section">
<div class="container">
</div></footer>
<!-- loader -->
```

```
<div id="ftco-loader" class="show fullscreen"><svg class="circular" width="48px" height="48px"><circle class="path-bg" cx="24" cy="24" r="22" fill="none" stroke-width="4" stroke="#eeeeee"/><circle class="path" cx="24" cy="24" r="22" fill="none" stroke-width="4" stroke-miterlimit="10" stroke="#F96D00"/></svg></div>

<script src="../static/js/jquery.min.js"></script>

<script src="../static/js/jquery-migrate-3.0.1.min.js"></script>

<script src="../static/js/popper.min.js"></script>

<script src="../static/js/bootstrap.min.js"></script>

<script src="../static/js/jquery.easing.1.3.js"></script>

<script src="../static/js/jquery.waypoints.min.js"></script>

<script src="../static/js/jquery.stellar.min.js"></script>

<script src="../static/js/jquery.animateNumber.min.js"></script>

<script src="../static/js/owl.carousel.min.js"></script>

<script src="../static/js/jquery.magnific-popup.min.js"></script>

<script src="../static/js/scrollax.min.js"></script>

<script src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBVWaKrjvy3MaE7SQ74_uJiULgl1JY0H2s&sensor=false"></script>

<script src="../static/js/google-map.js"></script>

<script src="../static/js/main.js"></script>

</body>

</html>
```

Register:

```
<!DOCTYPE html>

<html lang="en">

<head>

<title>Drug Recommendation </title>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
```

```
  <link
href="https://fonts.googleapis.com/css?family=Roboto:300,400,500,700&display=swap"
rel="stylesheet">

   <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/font-
awesome/4.7.0/css/font-awesome.min.css">

   <link rel="stylesheet" href="../static/css/animate.css">

   <link rel="stylesheet" href="../static/css/owl.carousel.min.css">

   <link rel="stylesheet" href="../static/css/owl.theme.default.min.css">

   <link rel="stylesheet" href="../static/css/magnific-popup.css">

   <link rel="stylesheet" href="../static/css/flaticon.css">

   <link rel="stylesheet" href="../static/css/style.css">

 </head>

 <body><nav class="navbar navbar-expand-lg navbar-dark ftco_navbar bg-dark ftco-navbar-
light" id="ftco-navbar">

          <div class="container">

                <a class="navbar-brand" href="index.html">Drug<span>
Recommendation</span></a>

           <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#ftco-nav" aria-controls="ftco-nav" aria-expanded="false" aria-label="Toggle
navigation">

             <span class="fa fa-bars"></span> Menu

           </button>

           <div class="collapse navbar-collapse" id="ftco-nav">

            <ul class="navbar-nav ml-auto">

           <li class="nav-item "><a href="{{ url_for('first')}}" class="nav-
link">Home</a></li>

                <li class="nav-item "><a href="{{ url_for('upload')}}" class="nav-
link">Upload</a></li>

                              <li class="nav-item "><a href="{{ url_for('login')}}"
class="nav-link">Login</a></li>

                <li class="nav-item active"><a href="{{ url_for('register')}}" class="nav-
link">Register</a></li>

                </ul></div></div></nav>
```

```
<!-- END nav -->

<section class="hero-wrap hero-wrap-2" style="background-image: url('../static/images/bg_2.jpg');" data-stellar-background-ratio="0.5">

<div class="overlay"></div><div class="container"><div class="row no-gutters slider-text align-items-end">

<div class="col-md-9 ftco-animate pb-5">

<h1 class="mb-0 bread">Register</h1>

</div></div></div></section>

<section class="ftco-section bg-light"><div class="container"><div class="row justify-content-center><div class="row">

<form action="{{ url_for('register') }}" method="post" >

<!-- Portfolio Item 1 -->

<div class="col-md-30 col-lg-21" style="margin-left:-50px"><div class="control-group">

<!-- Username -->

<h2>Register</h2>

</br>

<div class="controls">

<input type="text" name="username" class= "form-control" placeholder="Username" required>

</div> </div></br>

<div class="control-group">

<!-- Password-->

<div class="controls">

<input type="password" name="password" class= "form-control" placeholder="Password" required>

</div></div> </br>

<div class="controls"></br>

<!-- Password-->

<div class="controls">
```

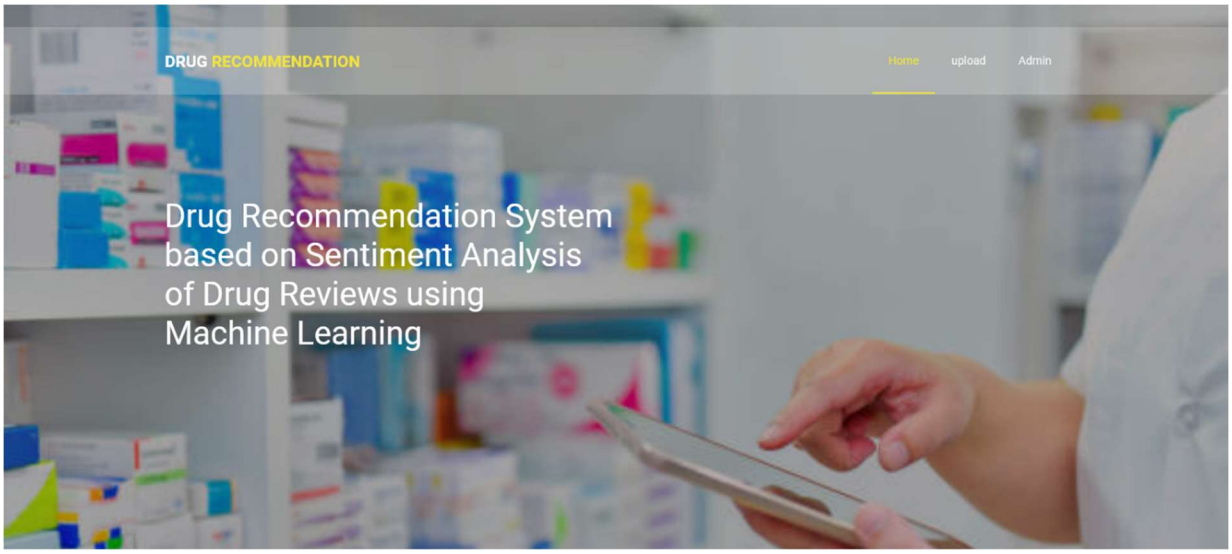
```
                        <input type="email" name="email" class= "form-control" placeholder="Email" required>
                          </div></div> </br>
                         <div class="control-group">
                            <!-- Password-->
                            <div class="controls">
                              <input type="age" name="age" class= "form-control" placeholder="age" required>
                        </div></div></br>
                          <div class="control-group">
                            <!-- Button --><div class="controls">
                                 <div class="msg">{{ msg }}</div>
                             <input type="submit" class="btn btn-primary" value="submit" style="margin-left: 50px" >
                            </div></div>
  </div>
  </form>
  </div></div></div></section>
 <footer class="footer ftco-section"><div class="container"></div></footer>
 <!-- loader -->
  <div id="ftco-loader" class="show fullscreen"><svg class="circular" width="48px" height="48px"><circle class="path-bg" cx="24" cy="24" r="22" fill="none" stroke-width="4" stroke="#eeeeee"/><circle class="path" cx="24" cy="24" r="22" fill="none" stroke-width="4" stroke-miterlimit="10" stroke="#F96D00"/></svg></div>
  <script src="../static/js/jquery.min.js"></script>
  <script src="../static/js/jquery-migrate-3.0.1.min.js"></script>
  <script src="../static/js/popper.min.js"></script>
  <script src="../static/js/bootstrap.min.js"></script>
  <script src="../static/js/jquery.easing.1.3.js"></script>
  <script src="../static/js/jquery.waypoints.min.js"></script>
  <script src="../static/js/jquery.stellar.min.js"></script>
```

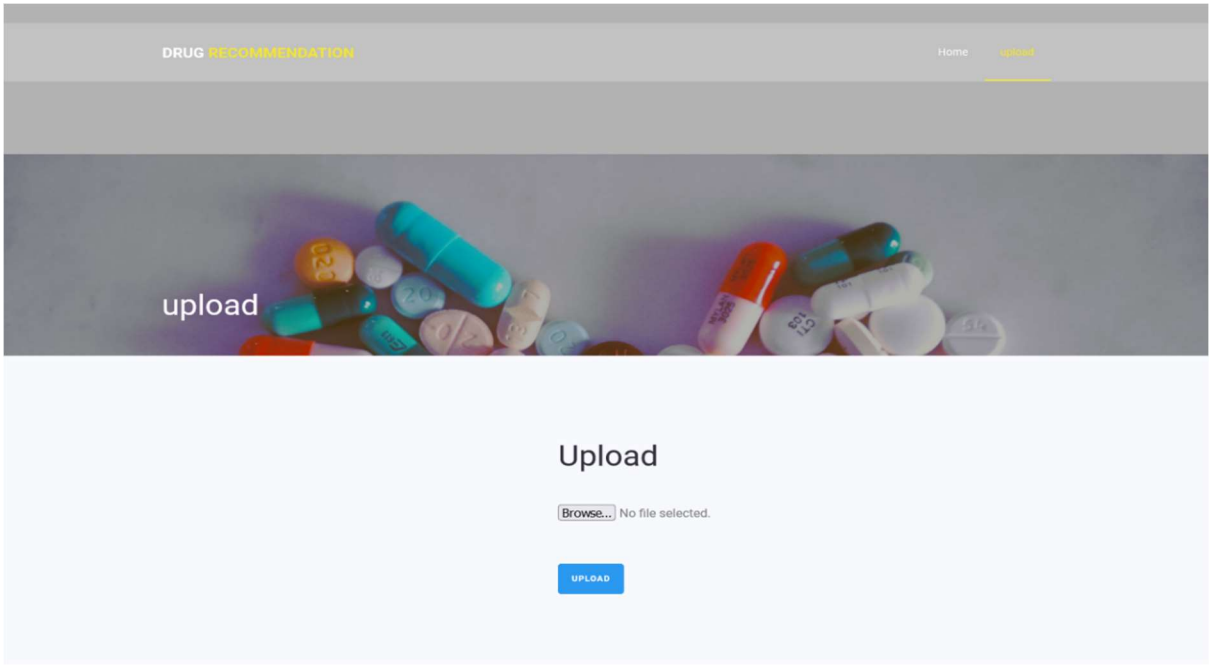
```
  <script src="../static/js/jquery.animateNumber.min.js"></script>

  <script src="../static/js/owl.carousel.min.js"></script>

  <script src="../static/js/jquery.magnific-popup.min.js"></script>

  <script src="../static/js/scrollax.min.js"></script>

  <script
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyBVWaKrjvy3MaE7SQ74_uJiULgl1JY0H2s&sensor=false"></script>

  <script src="../static/js/google-map.js"></script>

  <script src="../static/js/main.js"></script>  </body>

</html>
```

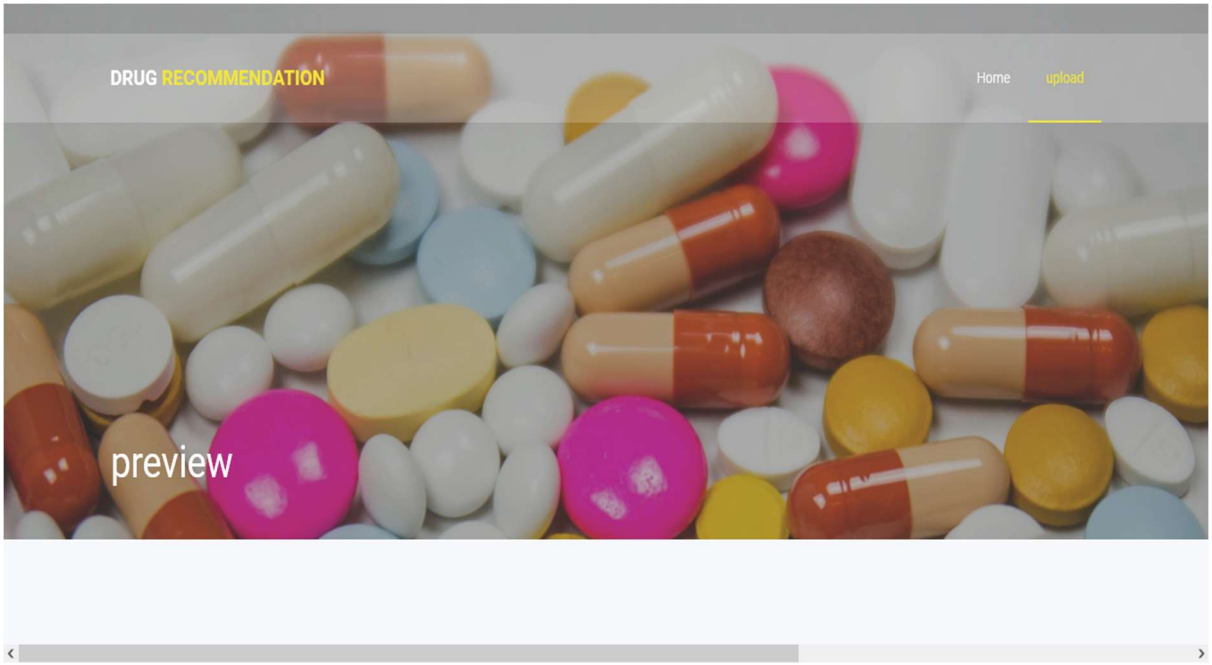
7. SCREENSHOTS

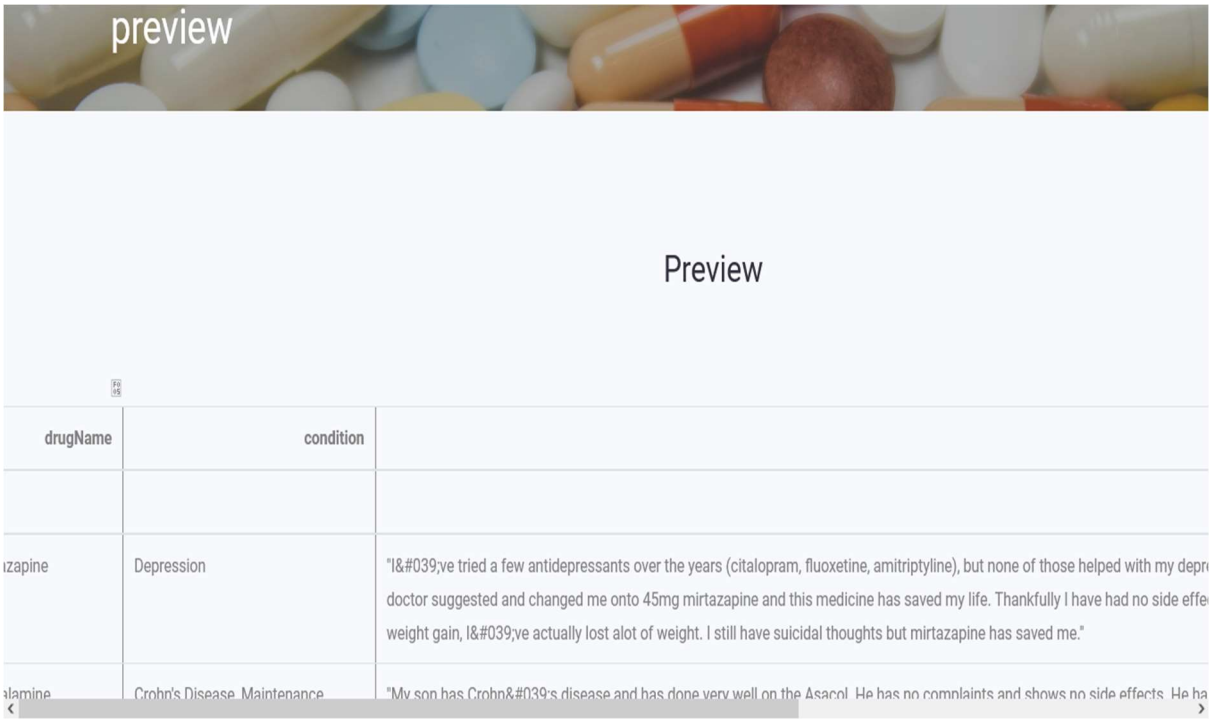
Chapter 7:

SCREENSHOTS





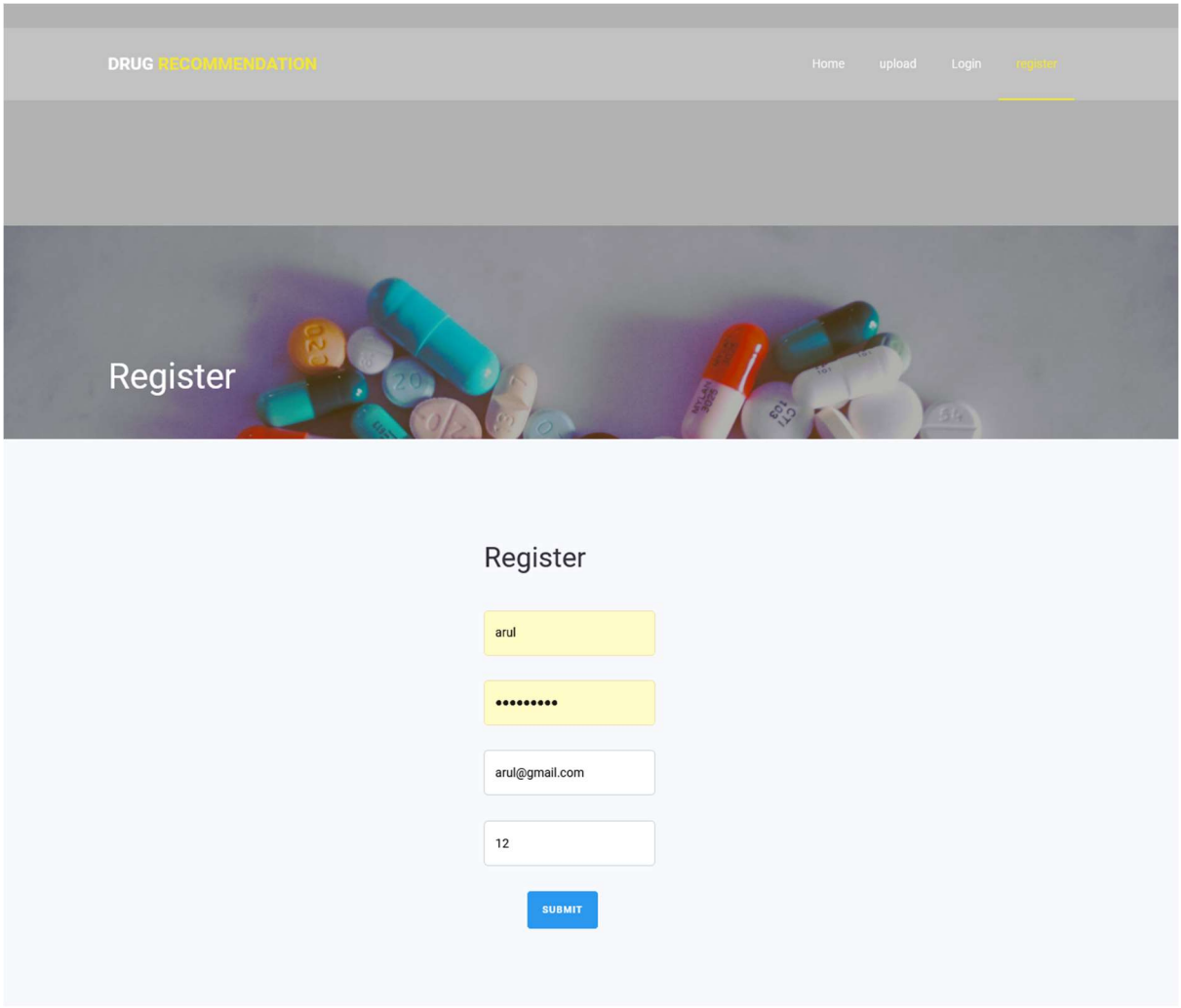

DRUG RECOMMENDATION
Home
upload
preview

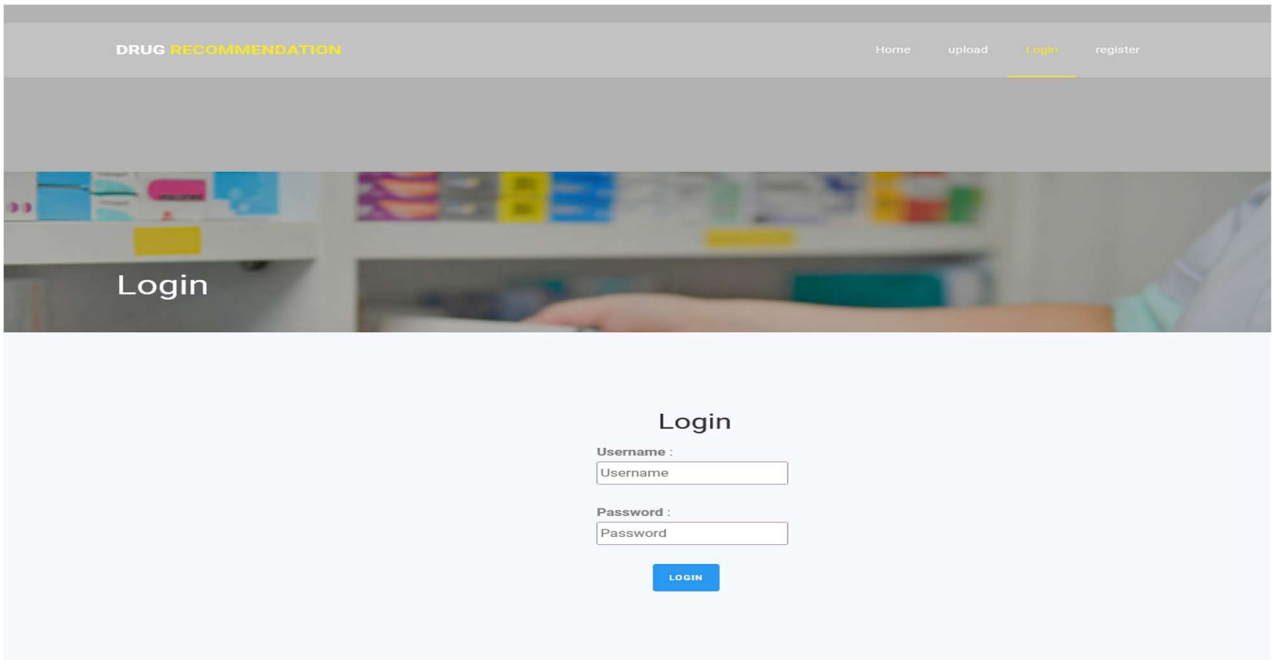

preview
Preview
drugName
condition
Depression
"I've tried a few antidepressants over the years (citalopram, fluoxetine, amitriptyline), but none of those helped with my depr
doctor suggested and changed me onto 45mg mirtazapine and this medicine has saved my life. Thankfully I have had no side effe
weight gain, I've actually lost alot of weight. I still have suicidal thoughts but mirtazapine has saved me."
Crohn's Disease, Maintenance
"My son has Crohn's disease and has done very well on the Asacol. He has no complaints and shows no side effects. He ha

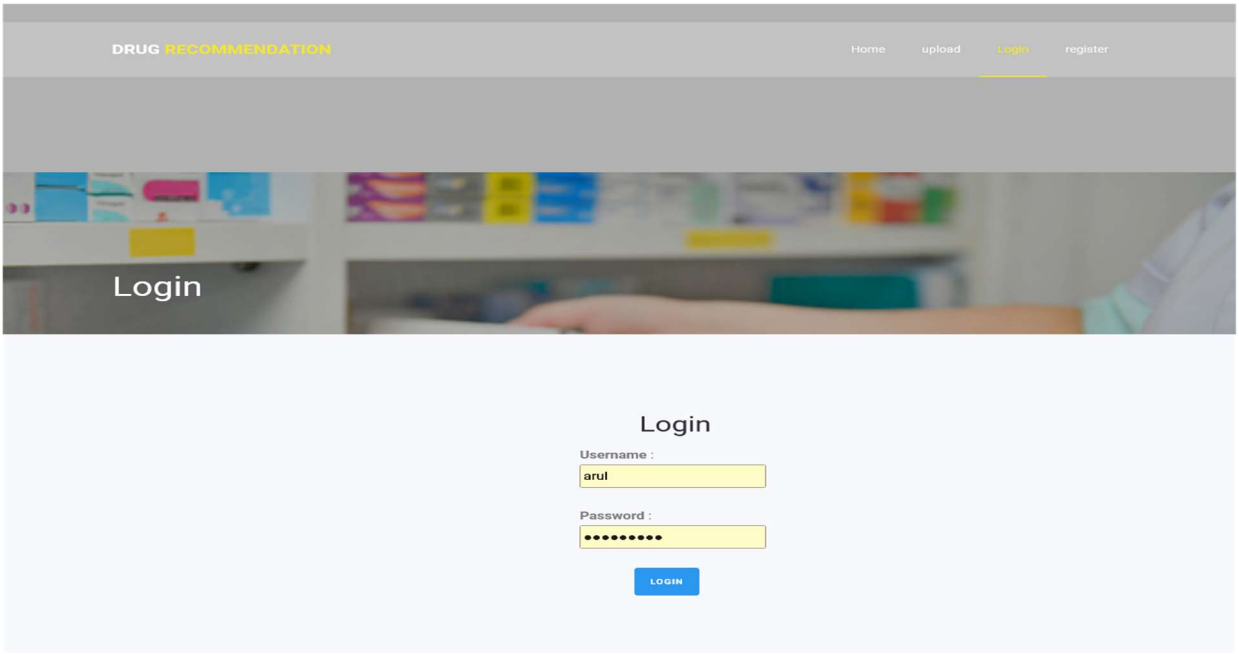
:asone / terol	Asthma	"Ive had asthma since I was 5 and now I'm 42. I've always only used rescue inhalers for most of it and since I'v my inhaler and I'm never short of breath. It's apparent that this is not for everyone as I've read some reviews. I have never had any side effects. I'm a fan.....actually was given a free 12 month prescription for this and that alone saved m guessing!"
alopram	Anxiety	"If you just started Lexapro and are considering quitting because of the side effects - don't! Trust me. Wait it out. It gets so n Lexapro, I was extremely nauseous and anxious, and couldn't sleep at night. I've been on it now for six months and fe The only side effect I experience is occasional drowsiness. I can sleep at night, I have no sexual side effects, and I haven't g off this medicine without consulting your Doctor first and having him/her help you through it. If you take yourself off it (especially c Be smart and stay strong. The good definitely outweighs the bad."
itermine	Weight Loss	"Started Phentermine 37.5mg about 10 days ago. Weight 199LB and 5’8. MD suggested due to health problems aNd impair only took the pill 2 days, skip a day, one pill, and skip a day again.\r\nI could barely eat bites of food... that’s why I staggered and felt “off” so I’d force myself to eat fruit, or something small. \r\nI really want the weight off but I know diet pill....just short term booster.\r\nI do exercise 3 x week light low impact. I plan to work hard with healthy eating, exercise, and this p for the assistance, but realistically learning to have healthy habits is keyâ€ï,"
inum chloride hydrate	Hyperhidrosis	"I have always had excessive sweating and body odour because of that and always been looking for medication for this. Tried so n they did not seem to work at all. I put this on for 2 nights continously and I was surprised at the results! It really works! Although it it's worth the price."

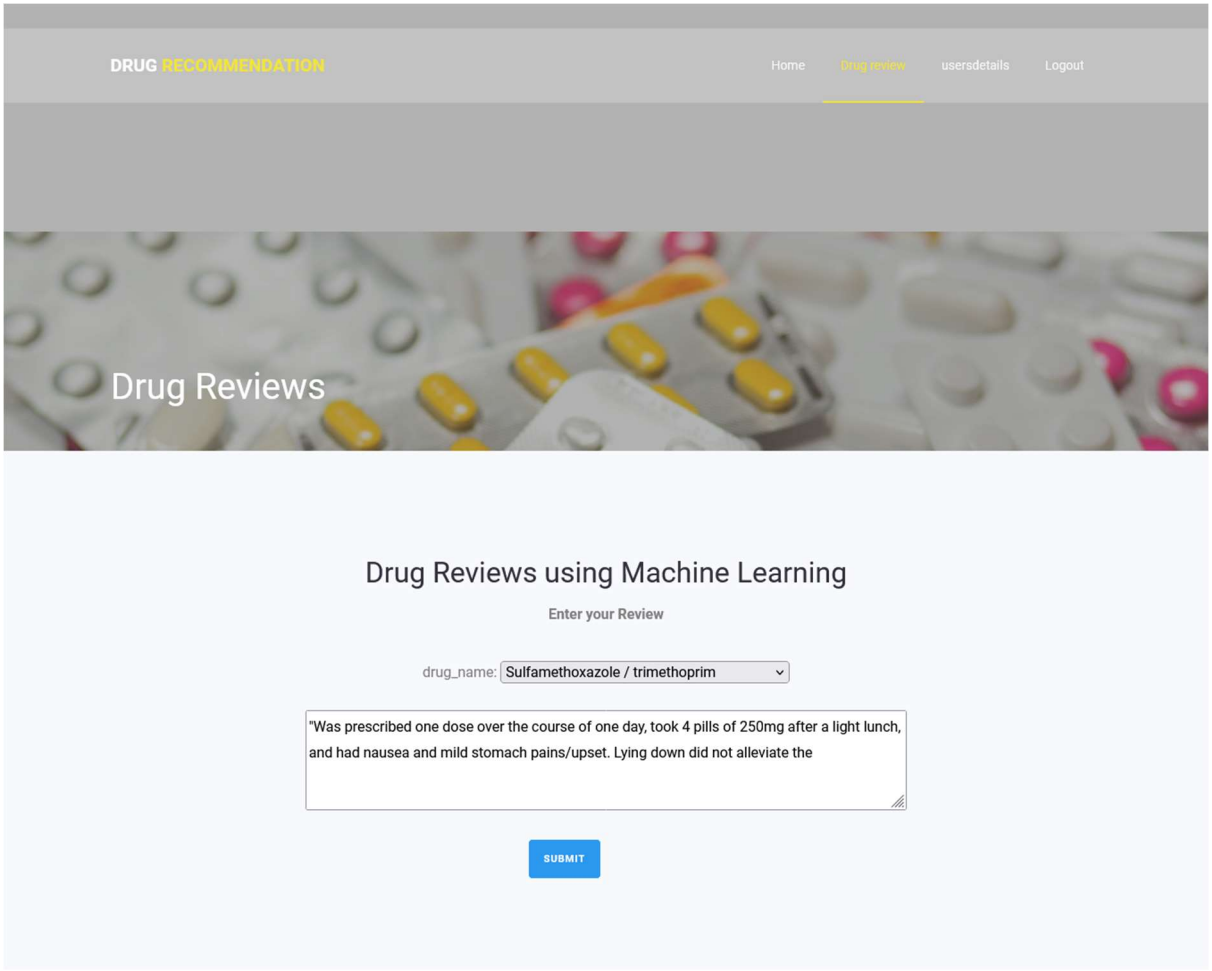
-Provera	Abnormal Uterine Bleeding	"I am a user of the birth control shot and I honestly love the fact that I do not have to worry about taking the pill. I just go to the doc But the only thing I hate is the bleeding for long periods. I have been on theshot for almost a year and a half. I remember the first th pain one day in my stomach, it felt so bad. I was in class and I had to run to the bathroom. When I was sitting on the toilet, this HUG the size of a tennis ball. Afterwards, my cramping stopped."
printec	Birth Control	"Tri sprintec makes me sick every single day, like throwing up constantly unless I eat a lot. That is the downside. Upside: It cleared lose weight, overall a good medicine for acne, maybe not for a birth control?"

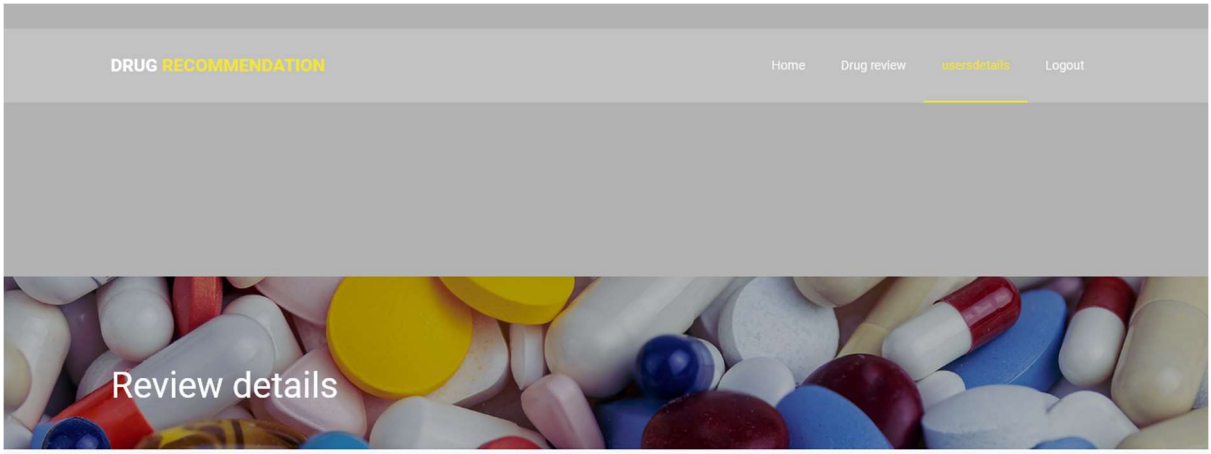
CLICK TO TRAIN | TEST


DRUG RECOMMENDATION
Home
upload
Login
register
Register
Register
arul
arul@gmail.com
12
SUBMIT


DRUG RECOMMENDATION
Home
upload
Login
register
Login
Login
Username :
Username
Password :
Password
LOGIN

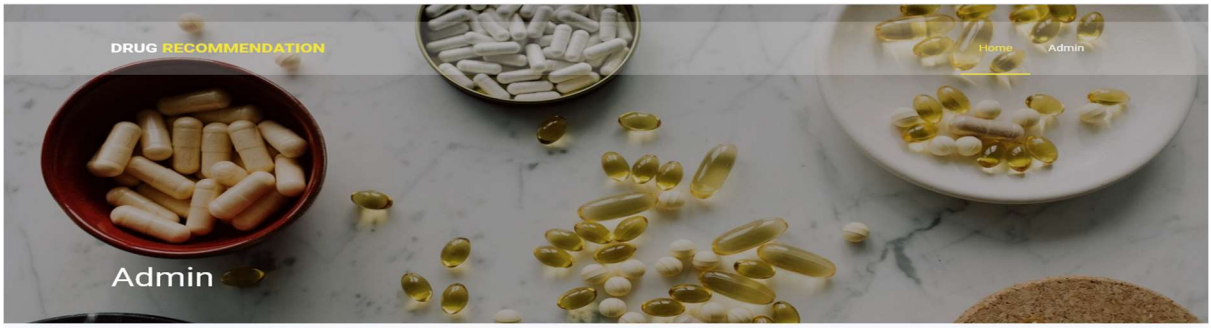

DRUG RECOMMENDATION
Home
upload
Login
register
Login
Login
Username :
arul
Password :
LOGIN


DRUG RECOMMENDATION
Home
Drug review
usersdetails
Logout
Drug Reviews
Drug Reviews using Machine Learning
Enter your Review
drug_name:
Sulfamethoxazole / trimethoprim
"Was prescribed one dose over the course of one day, took 4 pills of 250mg after a light lunch, and had nausea and mild stomach pains/upset. Lying down did not alleviate the
SUBMIT



Drug Review details

User ID	Username	Email_Id	age	Review	post
1	santhosh	Sandy@123	18	"I had a wonderful experience with Ziana. This is the only acne medication that didn't dry out my skin after using it. I thought I would never have a good experience with acne medications. Ziana is a smooth, moisturizing acne medication that leaves your skin feeling great. With a couple days the redness and swelling of my acne went down significantly. I don't see myself ever using another acne medication."	Your review Has Been Posted Successfully
1	santhosh	Sandy@123	18	"Taking drug for about 5 years and blood pressure stays around 140/93. All of a sudden started having ringing in ears and decided to go off Diovan after reading this was a side effect. Also noticed shortness of breath, weight gain and tiredness. Now blood pressure is about the same without Diovan or any other medication except still have the ringing in the ears."	Your review Has Been Posted Successfully
2	arul	Sandy@123	12	"Was prescribed one dose over the course of one day, took 4 pills of 250mg after a light lunch, and had nausea and mild stomach pains/upset. Lying down did not alleviate the	Your review Has Been Posted Successfully

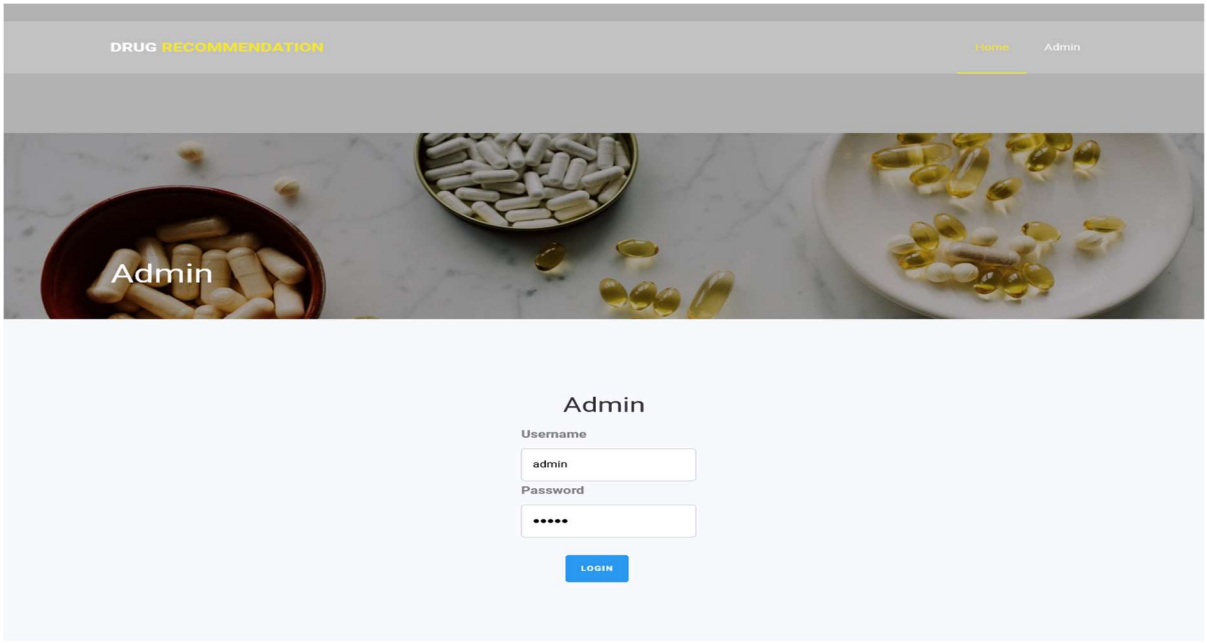


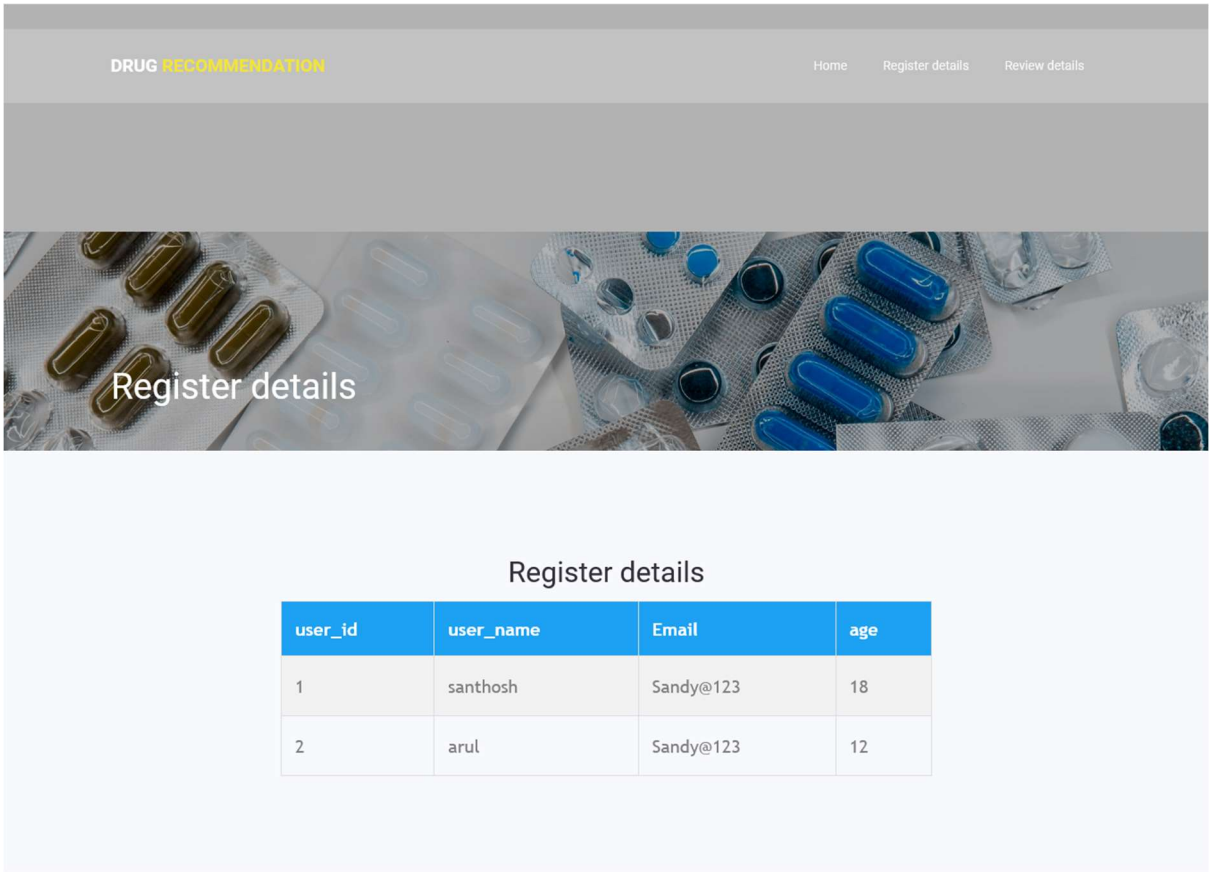
Admin

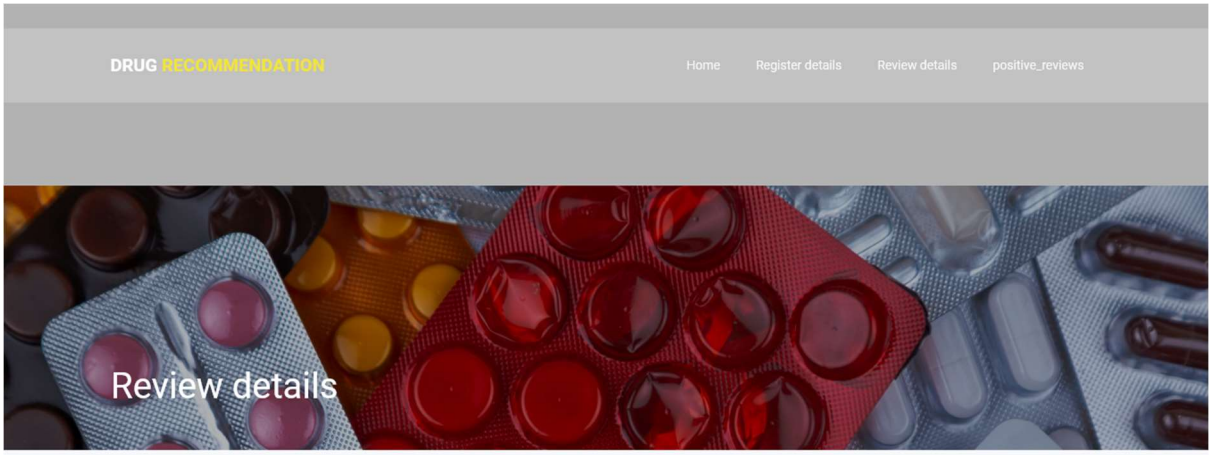
Username

Password

LOGIN

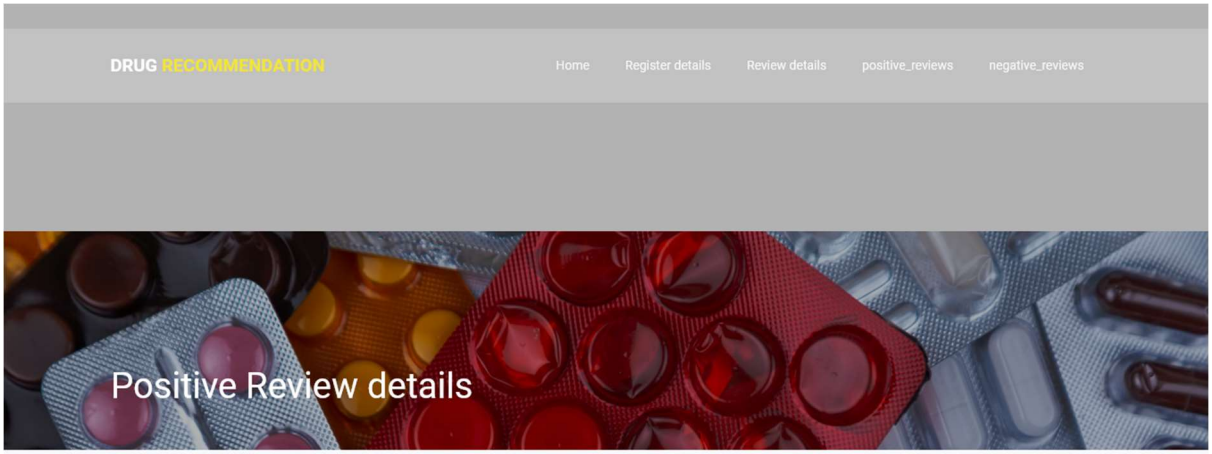

DRUG RECOMMENDATION
Home
Admin
Admin
Admin
Username
admin
Password
LOGIN


DRUG RECOMMENDATION
Home
Register details
Review details
Register details
Register details
user_id
user_name
Email
age
1
santhosh
Sandy@123
18
2
arul
Sandy@123
12



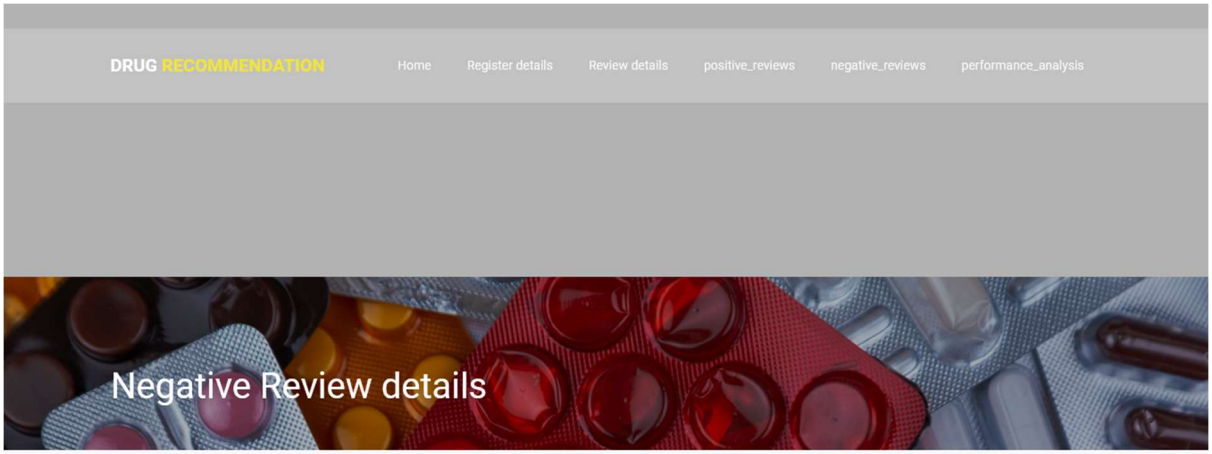
Review details

user_id	user_name	Email	age	drug Name	Review	pred
1	santhosh	Sandy@123	18	Ziana	"I had a wonderful experience with Ziana. This is the only acne medication that didn't dry out my skin after using it. I thought I would never have a good experience with acne medications. Ziana is a smooth, moisturizing acne medication that leaves your skin feeling great. With a couple days the redness and swelling of my acne went down significantly. I don't see myself ever using another acne medication."	Positive
1	santhosh	Sandy@123	18	Diovan	"Taking drug for about 5 years and blood pressure stays around 140/93. All of a sudden started having ringing in ears and decided to go off Diovan after reading this was a side effect. Also noticed shortness of breath, weight gain and tiredness. Now blood pressure is about the same without Diovan or any other medication except still have the ringing in the ears."	Negative
2	arul	Sandy@123	12	Sulfamethoxazole / trimethoprim	"Was prescribed one dose over the course of one day, took 4 pills of 250mg after a light lunch, and had nausea and mild stomach pains/upset. Lying down did not alleviate the	Positive



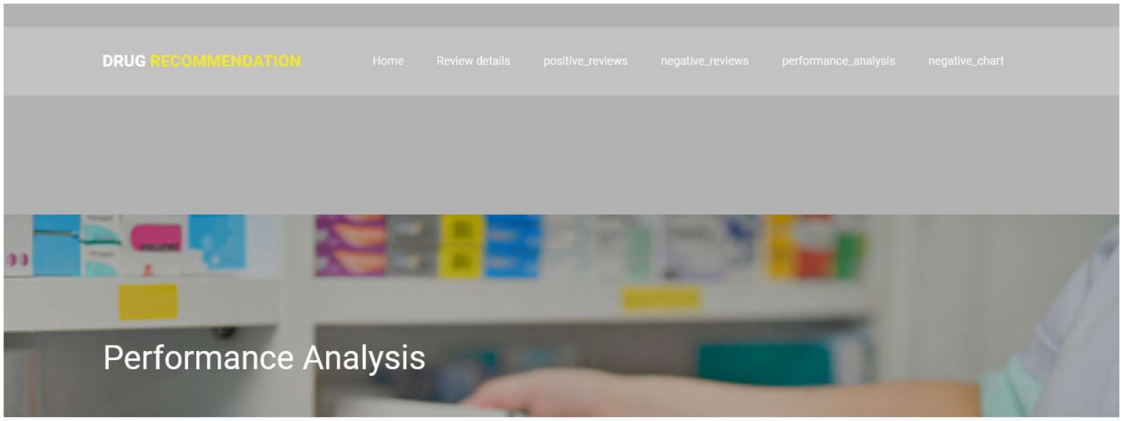
Positive Review details

user_id	user_name	Email	age	drug Name	Review	pred
1	santhosh	Sandy@123	18	Ziana	"I had a wonderful experience with Ziana. This is the only acne medication that didn't dry out my skin after using it. I thought I would never have a good experience with acne medications. Ziana is a smooth, moisturizing acne medication that leaves your skin feeling great. With a couple days the redness and swelling of my acne went down significantly. I don't see myself ever using another acne medication."	Positive
2	arul	Sandy@123	12	Sulfamethoxazole / trimethoprim	"Was prescribed one dose over the course of one day, took 4 pills of 250mg after a light lunch, and had nausea and mild stomach pains/upset. Lying down did not alleviate the	Positive



Negative Review details

user_id	user_name	Email	age	drug Name	Review	pred
1	santhosh	Sandy@123	18	Diovan	"Taking drug for about 5 years and blood pressure stays around 140/93. All of a sudden started having ringing in ears and decided to go off Diovan after reading this was a side effect. Also noticed shortness of breath, weight gain and tiredness. Now blood pressure is about the same without Diovan or any other medication except still have the ringing in the ears."	Negative

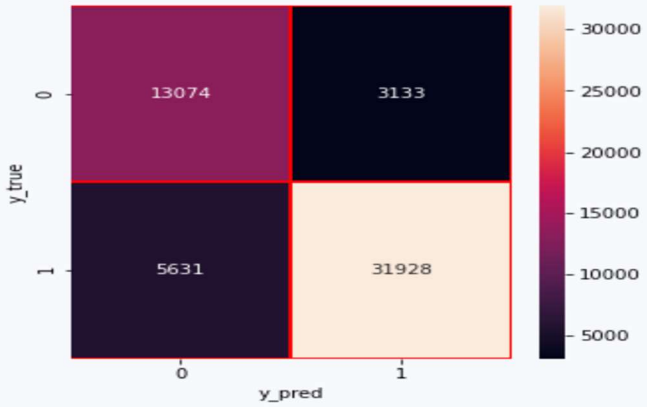


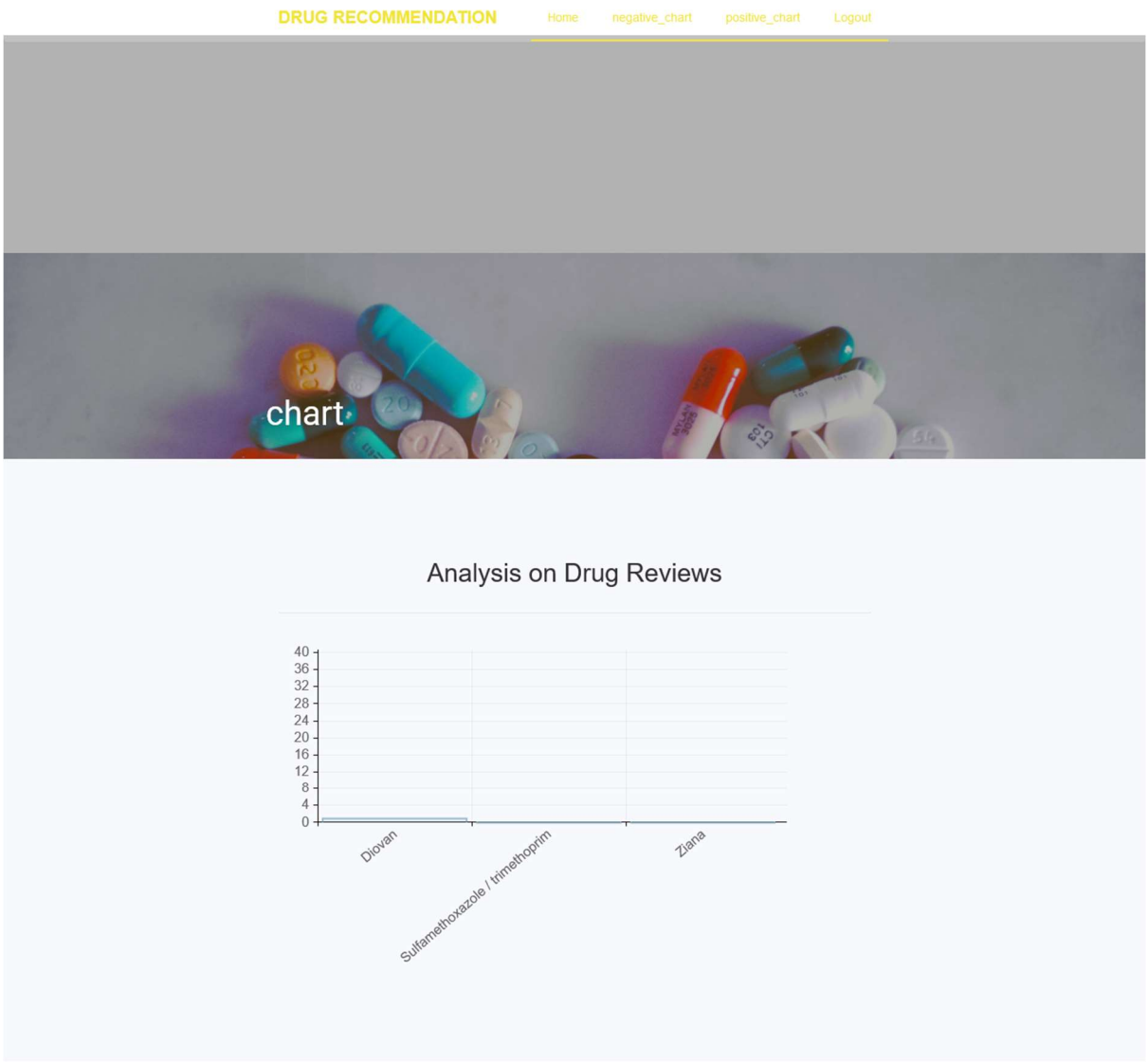
PERFORMANCE ANALYSIS

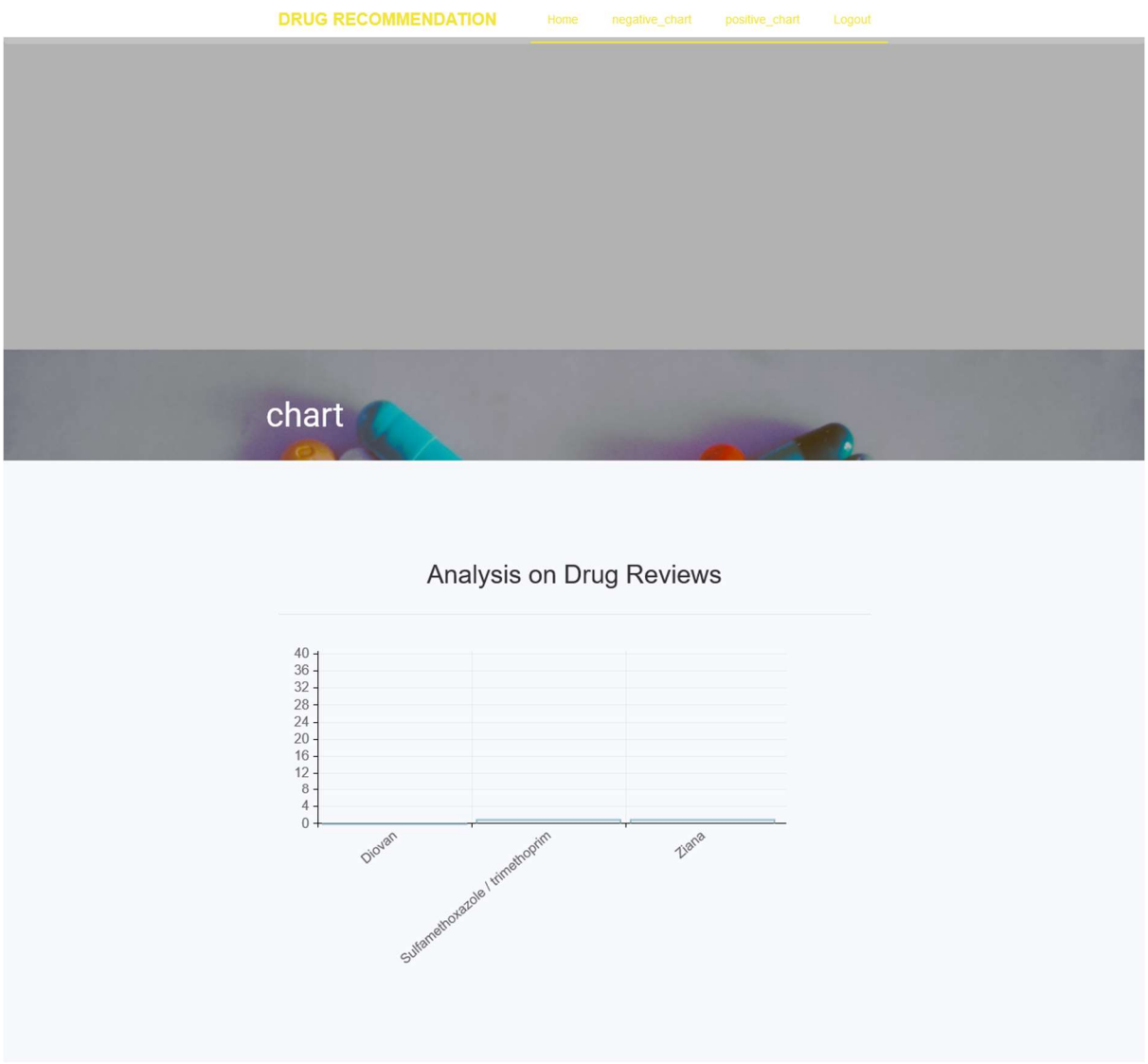
Precision and recall

	Recall	Precision
Negative	0.70	0.81
Positive	0.91	0.85

Confusion Matrix




DRUG RECOMMENDATION
Home
negative_chart
positive_chart
Logout
chart
Analysis on Drug Reviews
40
36
32
28
24
20
16
12
8
4
0
Diovan
Sulfamethoxazole / trimethoprim
Ziana


DRUG RECOMMENDATION
Home
negative_chart
positive_chart
Logout
chart
Analysis on Drug Reviews
40
36
32
28
24
20
16
12
8
4
0
Diovan
Sulfamethoxazole / trimethoprim
Ziana


DRUG RECOMMENDATION
Home
upload
Admin
Drug Recommendation System based on Sentiment Analysis of Drug Reviews using Machine Learning

8. TESTING

Chapter 8:

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, subassemblies, 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 o0f test. Each test type addresses a specific testing requirement.

TYPES OF TESTS:

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.

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 satisfaction, 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.

Functional test

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.

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.

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.

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.

6.1 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.

6.2 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.

6.3 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.

9. CONCLUSION

Chapter 9:

CONCLUSION

Reviews are becoming an integral part of our daily lives; whether go for shopping, purchase something online or go to some restaurant, we first check the reviews to make the right decisions. Motivated by this, in this research sentiment analysis of drug reviews was studied to build a recommender system using different types of machine learning classifiers, such as Logistic Regression, Perceptron, Multinomial Naive Bayes, Ridge classifier, Stochastic gradient descent, Linear SVC, applied on Bow, TF-IDF, and classifiers such as Decision Tree, Random Forest, Lgbm, and Cat boost were applied on Word2Vec and Manual features method. We evaluated them using five different metrics, precision, recall, f1score, accuracy, and AUC score, which reveal that the Linear SVC on TF-IDF outperforms all other models with 93% accuracy. On the other hand, the Decision tree classifier on Word2Vec showed the worst performance by achieving only 78% accuracy. We added best-predicted emotion values from each method, Perceptron on Bow (91%), Linear SVC on TF-IDF (93%), LGBM on Word2Vec (91%), Random Forest on manual features (88%), and multiply them by the normalized Useful Count to get the overall score of the drug by condition to build a recommender system.

Future Work:

Future work involves comparison of different oversampling techniques, using different values of n-grams, and optimization of algorithms to improve the performance of the recommender system.

10. BIBLIOGRAPHY

Chapter 9:

REFERENCES

[1] Telemedicine, https://www.mohfw.gov.in/pdf/Telemedicine.pdf

[2] Wittich CM, Burkle CM, Lanier WL. Medication errors: an overview for clinicians. Mayo Clin Proc. 2014 Aug;89(8):1116-25.

[3] CHEN, M. R., & WANG, H. F. (2013). The reason and prevention of hospital medication errors. Practical Journal of Clinical Medicine, 4.

[4] Drug Review Dataset, https://archive.ics.uci.edu/ml/datasets/Drug%2BReview%2BDataset%2B%2528Drugs.com%2529#

[5] Fox, Susannah, and Maeve Duggan. "Health online 2013. 2013." URL: http://pewinternet.org/Reports/2013/Health-online.aspx

[6] Bartlett JG, Dowell SF, Mandell LA, File TM Jr, Musher DM, Fine MJ. Practice guidelines for the management of community-acquired pneumonia in adults. Infectious Diseases Society of America. Clin Infect Dis. 2000 Aug;31(2):347-82. Doi: 10.1086/313954. Epub 2000 Sep 7. PMID: 10987697; PMCID: PMC7109923.

[7] Fox, Susannah & Duggan, Maeve. (2012). Health Online 2013. Pew Research Internet Project Report.

[8] T. N. Tekade and M. Emmanuel," Probabilistic aspect mining approach for interpretation and evaluation of drug reviews," 2016 International Conference on Signal Processing, Communication, Power and Embedded System (SCOPES), Paralakhemundi, 2016, pp. 1471-1476, Doi: 10.1109/SCOPES.2016.7955684.

[9] Doulaverakis, C., Nikolaidis, G., Kleontas, A. et al. Galen OWL: Ontology-based drug recommendations discovery. J Biomed Semant 3, 14 (2012). https://doi.org/10.1186/2041-1480-3-14

[10] Leilei Sun, Chuanren Liu, Chonghui Guo, Hui Xiong, and Yanming Xie. 2016. Data-driven Automatic Treatment Regimen Development and Recommendation. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining

(KDD ’16). Association for Computing Machinery, New York, NY, USA, 1865–1874. DOI: https://doi.org/10.1145/2939672.2939866

[11] V. Goel, A. K. Gupta and N. Kumar,” Sentiment Analysis of Multilingual Twitter Data using Natural Language Processing,” 2018 8th International Conference on Communication Systems and Network Technologies (CSNT), Bhopal, India, 2018, pp. 208-212, Doi: 10.1109/CSNT.2018.8820254.

[12] Shimada K, Takada H, Mitsuyama S, et al. Drug-recommendation system for patients with infectious diseases. AMIA Annu Symp Proc. 2005; 2005:1112.