A

Mini Project

On

DRUG RECOMMENDATION SYSTEM BASED ON SENTIMENTAL ANALYSIS OF DRUG REVIEWS USING MACHINE LEARNING

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

S. DIVYA (207R1A05B4)

B. LAHARI (207R1A0567)

M. SRI HARSHA (207R1A0592)

Under the Guidance of

TABEEN FATIMA

(Assistant Professor)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE, New Delhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya (V), Medchal Road, Hyderabad-501401.

2020-2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project entitled **"DRUG RECOMMENDATION SYSTEM BASED ON SENTIMENTAL ANALYSIS OF DRUG REVIEWS USING MACHINE LEARNING"** being submitted by **S. DIVYA (207R1A05B4), B. LAHARI (207R1A0567) & M.SRI HARSHA (207R1A0592)** in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by them under our guidance and supervision during the year 2023-24.

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

TABEEN FATIMA
(Assistant Professor)
INTERNAL GUIDE

Dr. A. RAJI REDDY
DIRECTOR

Dr. K. SRUJAN RAJU
HOD

EXTERNAL EXAMINER

Submitted for viva voice Examination held on ____________________

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project.

We take this opportunity to express our profound gratitude and deep regard to our guide **Ms. Tabeen Fatima,** Assistant Professor for her exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by her shall carry us a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Project Review Committee (PRC) **Dr. J. Narasimha Rao, Mr. G. Vinesh Shanker, Ms. Shilpa & Dr. K. Maheshwari** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju,** Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy,** Director for being cooperative throughout the course of this project. We also express our sincere gratitude to Sri. **Ch. Gopal Reddy,** Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project.

S. DIVYA (207R1A05B4)

B. LAHARI (207R1A0567)

M. SRI HARSHA (207R1A0592)

ABSTRACT

This project introduces a Drug Recommendation System (DRS) driven by the power of sentiment analysis and machine learning, aimed at empowering both patients and healthcare professionals in making informed decisions regarding medications choices.

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 specialist's 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. This abstract provides a glimpse into the innovation, potential, and the future impact of the Drug Recommendation System based on sentiment analysis of drug reviews using machine learning.

LIST OF FIGURES/TABLES

FIGURE NO	FIGURE NAME	PAGE NO
Figure 3.1	Architecture	7
Figure 3.2	Use Case Diagram	8
Figure 3.3	Class Diagram	9
Figure 3.4	Sequence diagram	10
Figure 3.5	Activity diagram	11

LIST OF SCREENSHOTS

SCREENSHOT NO.	SCREENSHOT NAME	PAGE NO.
Screenshot 5.1	Drug recommendation system application	26
Screenshot 5.2	Drug Prediction	26
Screenshot 5.3	Drug Recommendation Type Ratio	27
Screenshot 5.4	Drug Recommendation Trained and Tested Accuracy in bar chart	27
Screenshot 5.5	Drug Recommendation Trained and Tested Accuracy results	28

TABLE OF CONTENTS

ABSTRACT	i
LIST OF FIGURES	ii
LIST OF SCREENSHOTS	iii
1. INTRODUCTION	1
1.1 PROJECT SCOPE	1
1.2 PROJECT PURPOSE	1
1.3 PROJECT FEATURES	1
2. SYSTEM ANALYSIS	2
2.1 PROBLEM DEFINITION	2
2.2 EXISTING SYSTEM	2
2.2.1 DISADVANTAGES OF THE EXISTING SYSTEM	3
2.3 PROPOSED SYSTEM	3
2.3.1 ADVANTAGES OF PROPOSED SYSTEM	3
2.4 FEASIBILITY STUDY	4
2.4.1 ECONOMIC FEASIBILITY	4
2.4.2 TECHNICAL FEASIBILITY	4
2.4.3 OPERATIONAL FEASIBILITY	5
2.5 HARDWARE & SOFTWARE REQUIREMENTS	5
2.5.1 HARDWARE REQUIREMENTS	5
2.5.2 SOFTWARE REQUIREMENTS	6
3. ARCHITECTURE	7
3.1 ARCHITECTURE	7
3.2 DESCRIPTION	7
3.3 USE CASE DIAGRAM	8
3.4 CLASS DIAGRAM	9
3.5 SEQUENCE DIAGRAM	10
3.6 ACTIVITY DIAGRAM	11
4. IMPLEMENTATION	13
4.1 SAMPLE CODE	13
5. SCREENSHOTS	26
6. TESTING	29
6.1 INTRODUCTION TO TESTING	29

TABLE OF CONTENTS

6.2 TYPES OF TESTING 29

6.2.1 UNIT TESTING 29

6.2.2 INTEGRATION TESTING 30

6.2.3 FUNCTIONAL TESTING 30

6.3 TEST CASES 31

6.3.1 CLASSIFICATION 31

7. CONCLUSION & FUTURE SCOPE 33

7.1 PROJECT CONCLUSION 33

7.2 FUTURE SCOPE 33

8. BIBLIOGRAPHY 34

8.1 REFERENCES 34

8.2 GITHUB LINK 34

1. INTRODUCTION

1. INTRODUCTION

1.1 PROJECT SCOPE

This project is titled “Drug Recommendation System based on sentimental analysis of Drug Reviews using Machine Learning”. The scope of this project is to design, develop, and implement a Drug Recommendation System (DRS) that leverages the power of sentiment analysis and machine learning to provide personalized and evidence-based drug recommendations. The DRS aims to assist both patients and healthcare professionals in making informed decisions regarding medication choices by analyzing sentiments expressed within drug reviews.

1.2 PROJECT PURPOSE

The primary purpose of this project is to design, develop, and implement a Drug Recommendation System (DRS) that leverages the capabilities of sentiment analysis and machine learning to serve both patients and healthcare professionals in making informed and personalized decisions regarding medication choices. The project aims to contribute significantly to the advancement of healthcare practices and the well-being of patients and healthcare professionals alike.

1.3 PROJECT FEATURES

These features collectively create a comprehensive Drug Recommendation System that supports informed healthcare decisions, enhances patient outcomes, and promotes the responsible use of medications based on sentiment analysis of drug reviewsusing machine learning. A Drug Recommendation System based on sentiment analysis ofdrug reviews using machine learning is a complex application designed to provide personalized and effective medication recommendations.

2. SYSTEM ANAYSIS

2. SYSTEM ANALYSIS

System analysis for a Drug Recommendation System based on Sentiment Analysis of Drug Reviews using Machine Learning involves a detailed examination of the system's requirements, components, and functionalities. System analysis is a critical phase in the development of a Drug Recommendation System based on Sentiment Analysis of Drug Reviews using Machine Learning. It lays the foundation for designing and building a system that meets the needs of both users and healthcare professionals while adhering to stringent security and privacy requirements.

2.1 PROBLEM DEFINITION

This challenge stems from the vast amount of medical information available, the diverse needs of patients, and the subjective nature of individual experiences with medications. To address this problem, we aim to develop a Drug Recommendation System that leverages sentiment analysis of drug reviews using machine learning to offer tailored medication recommendations.

2.2 EXISTING SYSTEM

There are several existing systems that use sentiment analysis and machine learning to recommend drugs based on reviews. These systems analyze the sentiment of the reviews and use that information to recommend drugs that are most likely to be effective for the patient's specific needs. This system uses a combination of machine learning algorithms and natural language processing techniques to analyze drug reviews and extract important features such as drug effectiveness, side effects, and dosage. The system also takes into account the patient's medical history, symptoms, age and gender to provide personalized drug recommendations.

2.2.1 DISADVANTAGES OF EXISTING SYSTEM

Following are the disadvantages of existing system:

- Biased Reviews
- Limited Data Availability
- Security and Privacy Concerns
- Limited Scope
- It Provides Information about one disease

2.3 PROPOSED SYSTEM

This is a customary system that proposes an item to the user, dependent on their advantage and necessity. 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. This system predicts the reviews based upon sentiment analysis of the patient who have already used that particular drug previously.

2.3.1 ADVANTAGES OF THE PROPOSED SYSTEM

- Suggests the quality of medication
- Improves the quality of rapid patient care
- Increase the efficiency of medical treatment

2.4 FEASIBILITY STUDY

The feasibility study should provide a comprehensive understanding of whether the Drug Recommendation System based on Sentimental analysis of Drug Reviews using machine learning is viable, both from a technical and economic standpoint, and whether it aligns with legal, operational and ethical requirements. It serves as the foundation for making informed decisions about proceeding with the project. The aspects involved in the feasibility analysis are:

- Economic Feasibility
- Technical Feasibility
- Operational Feasibility

2.4.1 ECONOMIC FEASIBILITY

Cost Estimation: Estimate the development costs, including software development, data acquisition, hardware, and ongoing maintenance expenses.

Return on Investment (ROI): Evaluate the potential benefits of the system, including increased medication adherence, improved patient outcomes, and potential revenue streams (e.g., premium features or partnerships).

Funding Sources: Identify potential funding sources, such as grants, investors, or internal budget allocation, to cover project expenses.

2.4.2 TECHNICAL FEASIBILITY

Machine Learning and NLP Expertise: Evaluate the availability of machine learning and natural language processing expertise within the development team or the ability to acquire such expertise.

Data Sources: Assess the availability and accessibility of relevant data sources, including drug reviews, medical literature, and user profiles.

Computational Resources: Determine if the necessary computational resources such as powerful hardware and cloud services, are accessible and affordable.

2.4.3 OPERATONAL FEASIBILITY

User Adoption: Analyze the willingness of potential users (patients and healthcare providers) to adopt the system and incorporate it into their healthcare routines.

Integration with Existing Systems: Assess the compatibility and integration possibilities with existing healthcare systems, such as electronic health records (EHRs) and telemedicine platforms.

Workflow Integration: Determine how the system will fit into the workflows of healthcare providers and patients without causing disruption.

2.5 HARDWARE & SOFTWARE REQUIREMENTS

2.5.1 HARDWARE REQUIREMENTS:

Hardware interfaces specify the logical characteristics of each interface between the software product and the hardware components of the system. The following are some hardware requirements.

- System : Pentium IV
- Hard Disk : 20 GB.
- Monitor : SVGA
- Mouse : Two or Three button Mouse.
- Ram : 4 GB
- Keyboard : Standard Windows Keyboard

2.5.2 SOFTWARE REQUIREMENTS:

Software Requirements specifies the logical characteristics of each interface and software components of the system. The following are some software requirements.

- Operating System : Windows 7 Ultimate.
- Platform : Python
- Designing : HTML, CSS, JavaScript
- Front End : Python
- Back End : Django-ORM

3. ARCHITECTURE

3. ARCHITECTURE

3.1 ARCHITECTURE

This project architecture shows the procedure followed for classification, starting from input to final prediction.

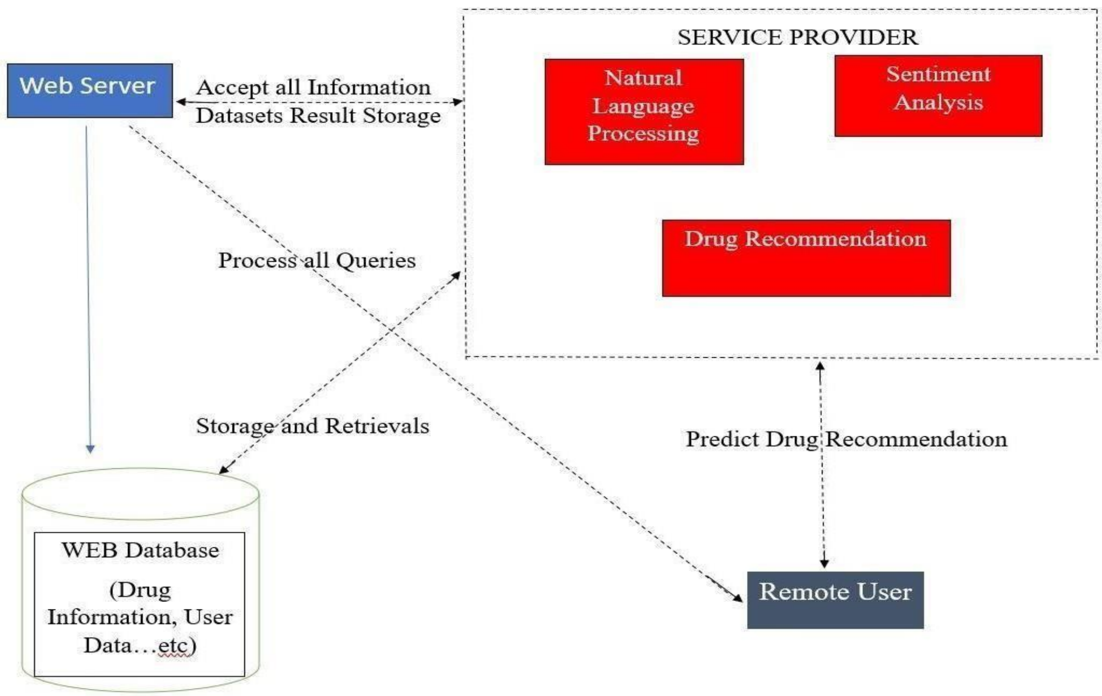


Figure 3.1: Architecture

3.2 DESCRIPTION

A Drug Recommendation System based on Sentiment Analysis of Drug Reviews using Machine Learning is a sophisticated healthcare application designed to assist patients and healthcare professionals in making informed decisions about medication. This system leverages advanced technologies, including machine learning and natural language processing (NLP), to analyze drug reviews and provide personalized drug recommendations.

3.3 USE CASE DIAGRAM

In the use case diagram, we have basically one actor who is the user in the trained model.

A use case diagram is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of usersthe system has. The use cases are represented by either circles or ellipses. The actors are often shown as stick figures.

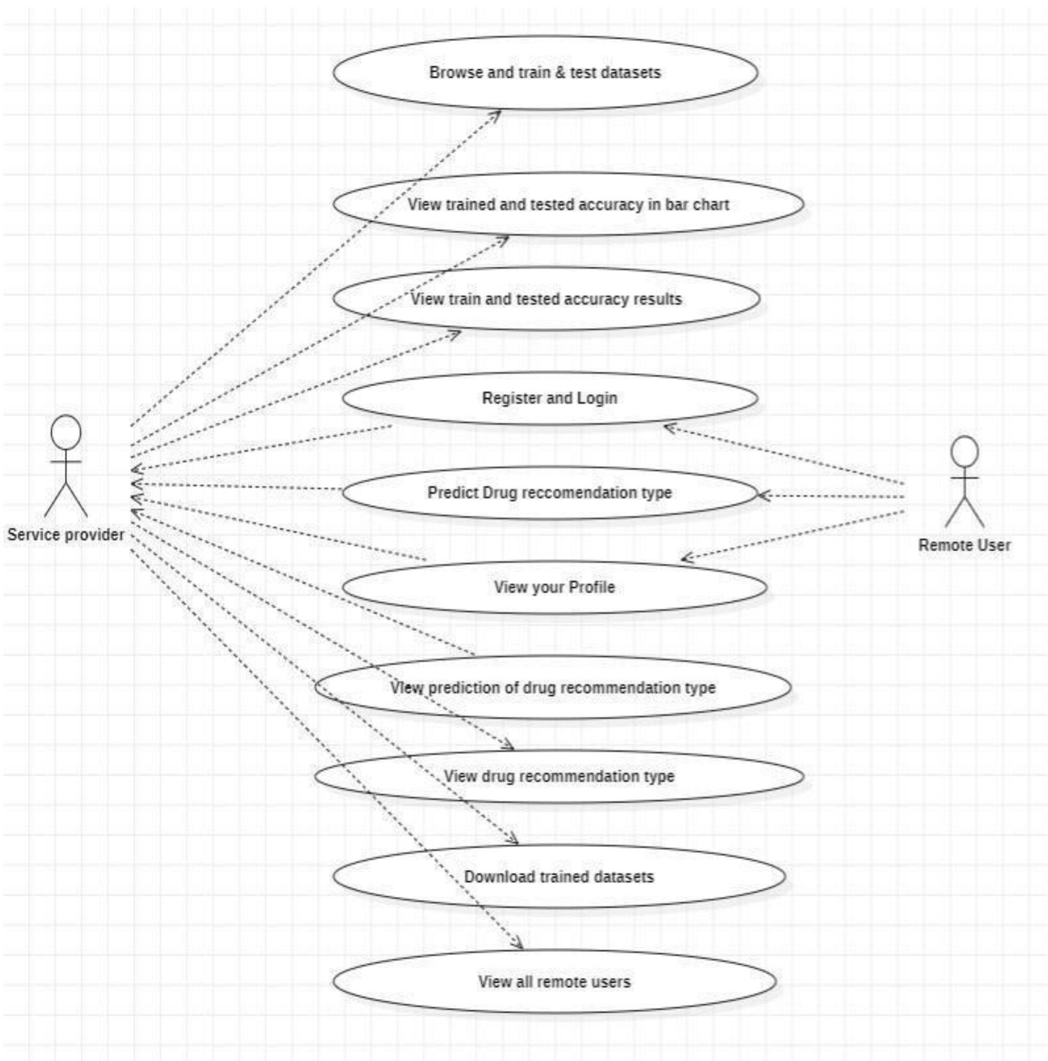


Figure 3.2: Use Case Diagram

3.4 CLASS DIAGRAM

Class diagram 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 objects.

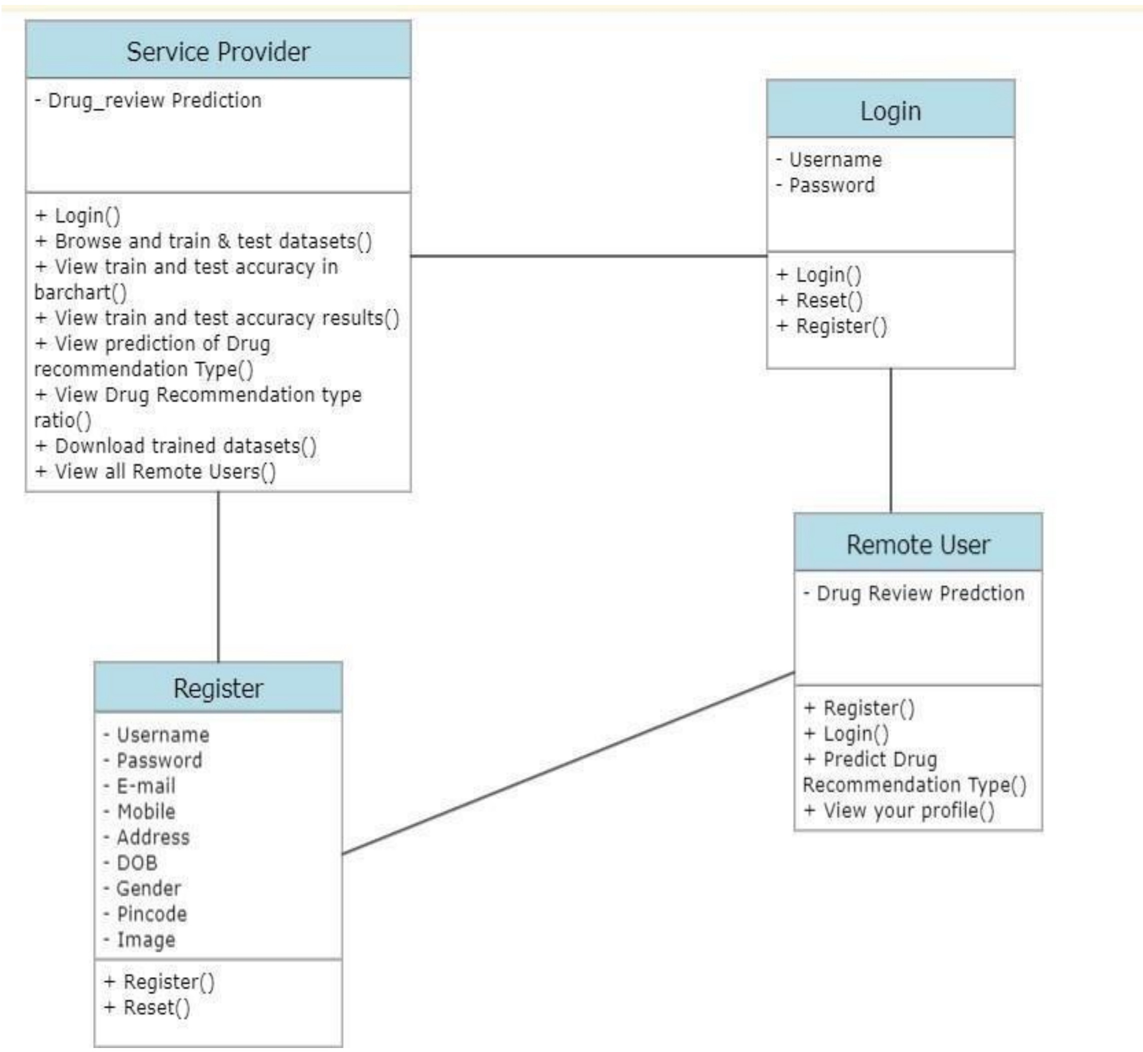


Figure 3.3: Class Diagram

3.5 SEQUENCE DIAGRAM

A sequence diagram shows object interactions arranged in time sequence. It depicts the objects involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development.

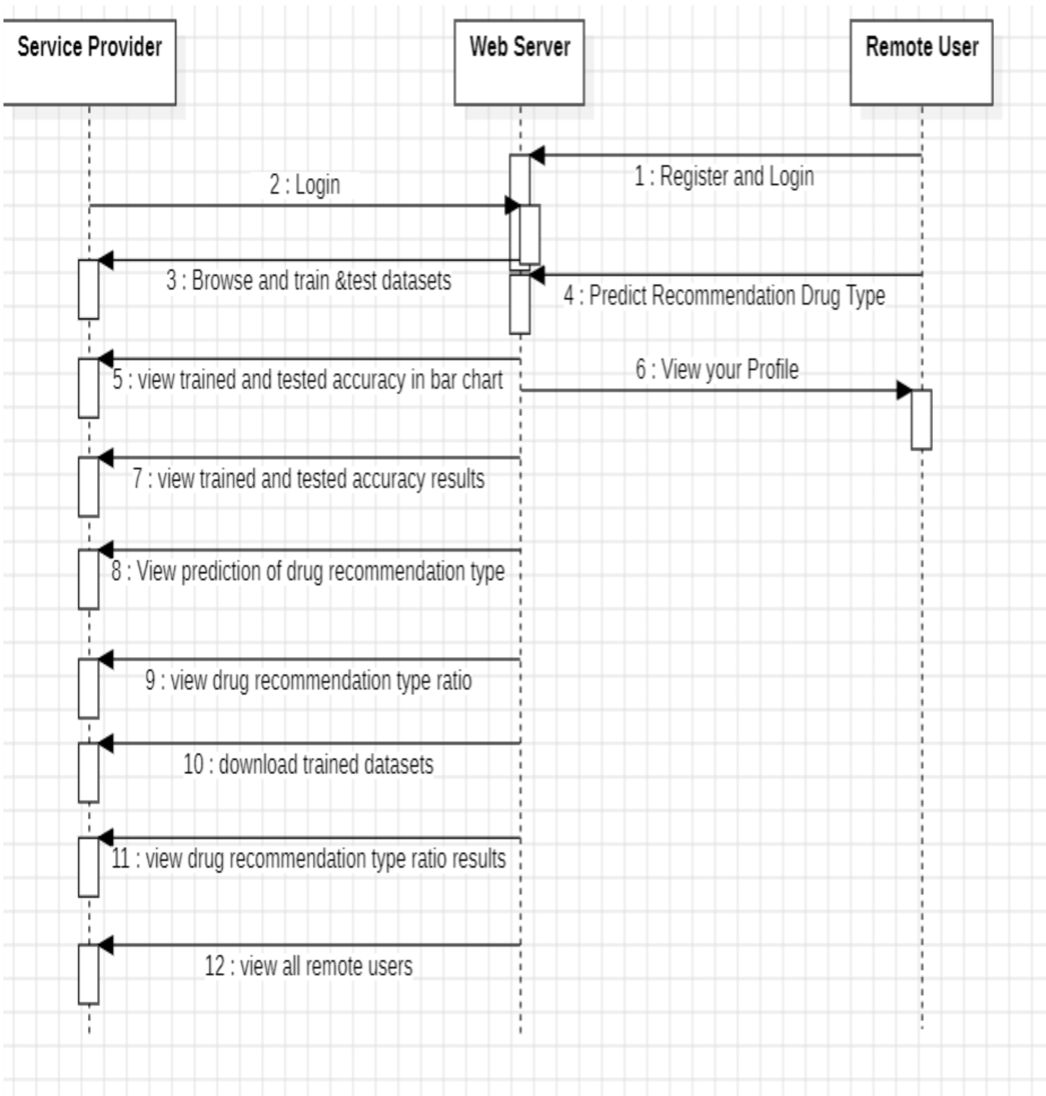
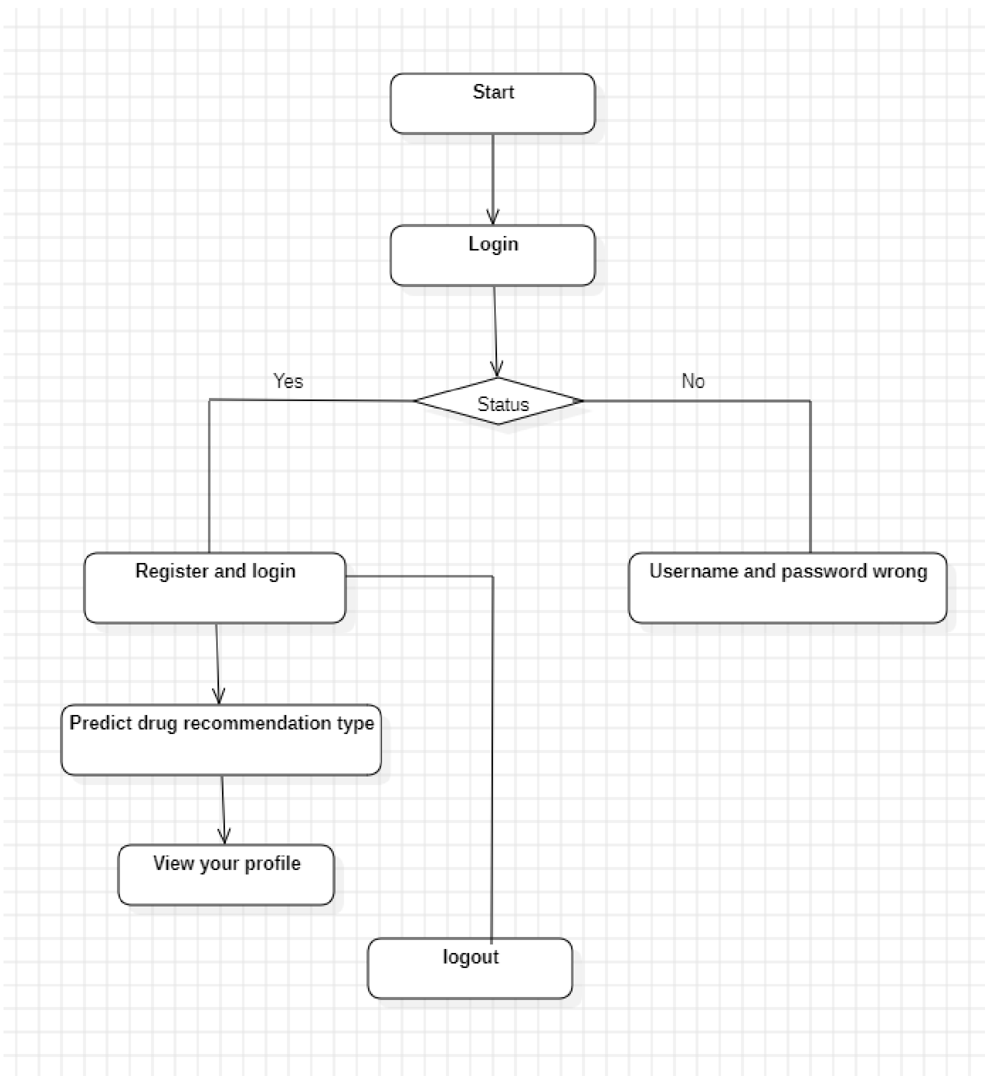


Figure 3.4: Sequence Diagram

3.6 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. They can also include elements showing the flow of data between activities through one or more data stores.

- Remote User



- Service Provider

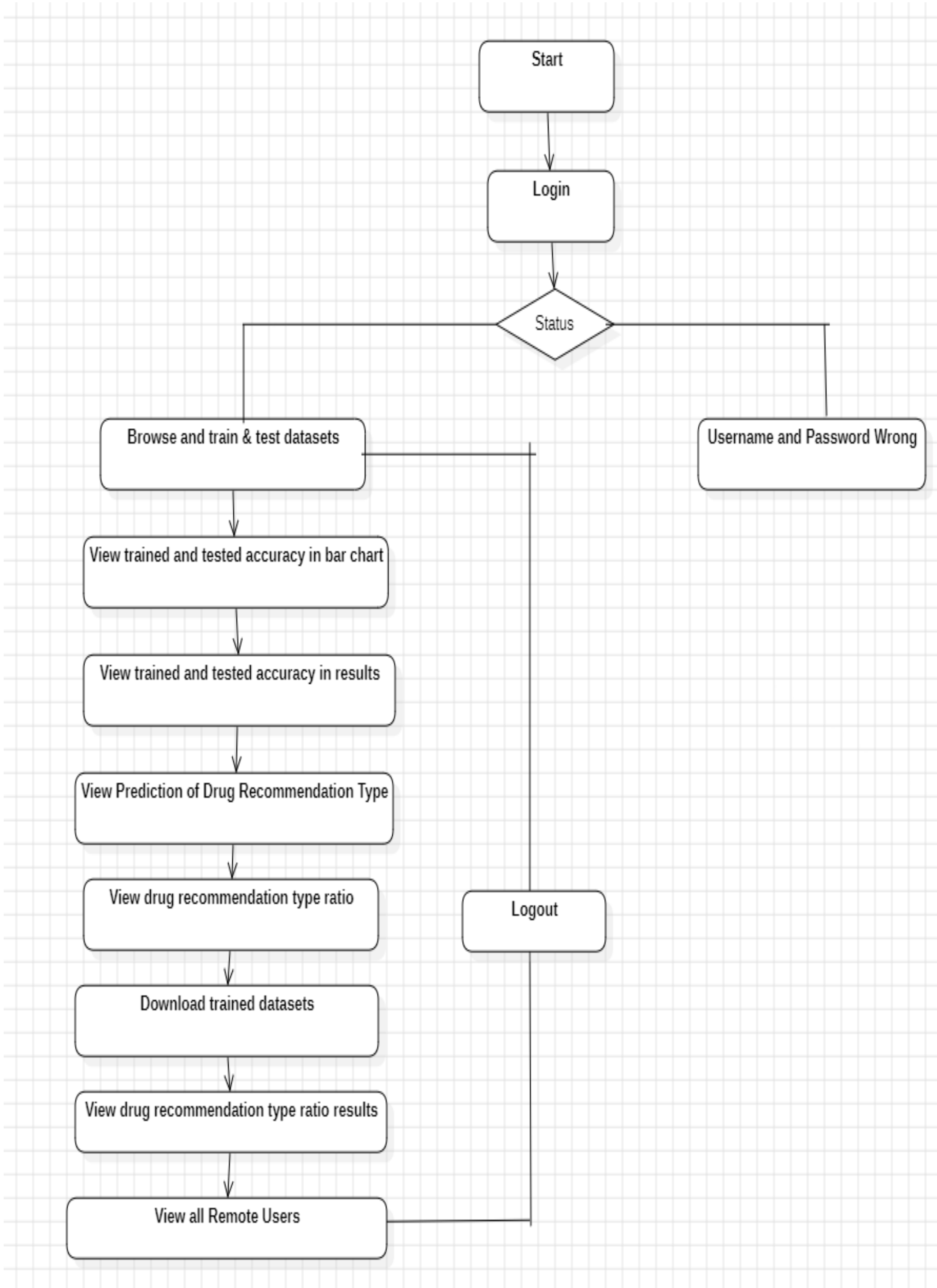


Figure 3.5: Activity Diagram

4. IMPLEMENTATION

4. IMPLEMENTATION

4.1 SAMPLE CODE

```
from django.db.models import Count
from django.db.models import Q
from django.shortcuts import render, redirect, get_object_or_404
import datetime
import openpyxl


import nltk
import re
import string
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import CountVectorizer
from nltk.stem.wordnet import WordNetLemmatizer
import pandas as pd
from wordcloud import WordCloud, STOPWORDS
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import VotingClassifier
# Create your views here.
from Remote_User.models import
ClientRegister_Model,drug_recommendation_Type,detection_ratio,detection_accuracy


def login(request):

    if request.method == "POST" and 'submit1' in request.POST:
        username = request.POST.get('username')
        password = request.POST.get('password')
        try:
            enter = ClientRegister_Model.objects.get(username=username,password=password)
            request.session["userid"] = enter.id
```

```
            return redirect('ViewYourProfile')
        except:
            pass

    return render(request,'RUser/login.html')
def Add_DataSet_Details(request):
    return render(request, 'RUser/Add_DataSet_Details.html', {"excel_data": ''})



def Register1(request):


    if request.method == "POST":
        username = request.POST.get('username')
        email = request.POST.get('email')
        password = request.POST.get('password')
        phoneno = request.POST.get('phoneno')
        country = request.POST.get('country')
        state = request.POST.get('state')
        city = request.POST.get('city')
        ClientRegister_Model.objects.create(username=username, email=email, password=password,
phoneno=phoneno,
                                            country=country, state=state, city=city)


        return render(request, 'RUser/Register1.html')
    else:
        return render(request,'RUser/Register1.html')


def ViewYourProfile(request):
    userid = request.session['userid']
    obj = ClientRegister_Model.objects.get(id= userid)
    return render(request,'RUser/ViewYourProfile.html',{'object':obj})



def Predict_Drug_Recommendation_Type(request):
    if request.method == "POST":
        review = request.POST.get('keyword')
```

```
if request.method == "POST":
    review = request.POST.get('keyword')
    dname = request.POST.get('dname')

df = pd.read_csv('Drugs_Review_Datasets.csv')
df
df.columns
df.rename(columns={'rating': 'Rating', 'review': 'Review'}, inplace=True)


def apply_recommend(Rating):
    if (Rating <= 7):
        return 0  # Neagtive
    else:
        return 1  # Positive


df['recommend'] = df['Rating'].apply(apply_recommend)
df.drop(['Rating'], axis=1, inplace=True)
recommend = df['recommend'].value_counts()


#
df.drop(['Id','ProductId','UserId','ProfileName','HelpfulnessNumerator','HelpfulnessDenominator','Time','Summary'], axis=1, inplace=True)


def preprocess_text(text):
    '''Make text lowercase, remove text in square brackets,remove links,remove punctuation
    and remove words containing numbers.'''
    text = text.lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    text = re.sub('"@', '', text)
    text = re.sub('@', '', text)
    text = re.sub('https: //', '', text)
    text = re.sub('\n\n', '', text)
```

```
    return text
df['processed_content'] = df['Review'].apply(lambda x: preprocess_text(x))
cv = CountVectorizer()
X = df['processed_content']
y = df['recommend']
X = cv.fit_transform(X)
models = []
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
X_train.shape, X_test.shape, y_train.shape
print("Naive Bayes")
from sklearn.naive_bayes import MultinomialNB
NB = MultinomialNB()
NB.fit(X_train, y_train)
models.append(('naive_bayes', NB))

# SVM Model
print("SVM")
from sklearn import svm
lin_clf = svm.LinearSVC()
lin_clf.fit(X_train, y_train)
models.append(('svm', lin_clf))


#classifier = VotingClassifier(models)
#classifier.fit(X_train, y_train)
#y_pred = classifier.predict(X_test)


review_data = [review]
vector1 = cv.transform(review_data).toarray()

predict_text = lin_clf.predict(vector1)

pred = str(predict_text).replace("[", "")
pred1 = pred.replace("]", "")
prediction = int(pred1)
```

```
        if prediction == 0:
            val = 'Negative'
        else:
            val = 'Positive'


        print(val)
        print(pred1)

        drug_recommendation_Type.objects.create(Drug_Name=dname,Drug_Review=review,
    Prediction=val)


        return render(request, 'RUser/Predict_Drug_Recommendation_Type.html',{'objs': val})
    return render(request, 'RUser/Predict_Drug_Recommendation_Type.html')

import nltk
import re
import
string
from nltk.corpus import stopwords
from sklearn.feature_extraction.text import
CountVectorizerfrom nltk.stem.wordnet import
WordNetLemmatizer
import pandas as pd
from wordcloud import WordCloud, STOPWORDS
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import accuracy_score, confusion_matrix,
classification_reportfrom sklearn.metrics import accuracy_score
from sklearn.metrics import f1_score
from sklearn.tree import
DecisionTreeClassifierfrom
sklearn.ensemble import VotingClassifier #
Create your views here.
from Remote_User.models import
ClientRegister_Model,drug_recommendation_Type,detection_ratio,detection_accuracy

def
  serviceproviderlogin(request
  ):if request.method  ==
  "POST":
    admin = request.POST.get('username')
```

```
  password = request.POST.get('password')
     if admin == "Admin" and password
       =="Admin":
       detection_accuracy.objects.all().delete()
       return redirect('View_Remote_Users')
  return
render(request,'SProvider/serviceproviderlogin.html')def
View_Drug_Recommendation_Type_Ratio(request):
  detection_ratio.objects.all().delete()
  ratio = ""
  kword =
  'Negative'
  print(kword)
  obj =
  drug_recommendation_Type.objects.all().filter(Q(Prediction=kword)
  )obj1 = drug_recommendation_Type.objects.all()
  count = obj.count();
  count1 =
  obj1.count();
  ratio = (count / count1) *
  100if ratio != 0:
     detection_ratio.objects.create(names=kword, ratio=ratio)

  ratio12 = ""
  kword12 =
  'Positive'
  print(kword12)
  obj12 =
  drug_recommendation_Type.objects.all().filter(Q(Prediction=kword12))
  obj112 = drug_recommendation_Type.objects.all()
  count12 = obj12.count();
  count112 =
  obj112.count();
  ratio12 = (count12 / count112) *
  100if ratio12 != 0:
     detection_ratio.objects.create(names=kword12, ratio=ratio12)
  obj = detection_ratio.objects.all()
  return render(request, 'SProvider/View_Drug_Recommendation_Type_Ratio.html', {'objs': obj})
def View_Remote_Users(request):
```

```
obj=ClientRegister_Model.objects.all()


    return render(request,'SProvider/View_Remote_Users.html',{'objects':obj})

def ViewTrendings(request):
    topic =
drug_recommendation_Type.objects.values('topics').annotate(dcount=Count('topics')).order_by('-
dcount')
    return  render(request,'SProvider/ViewTrendings.html',{'objects':topic})

def charts(request,chart_type):
    chart1 = detection_ratio.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/charts.html", {'form':chart1, 'chart_type':chart_type})

def charts1(request,chart_type):
    chart1 =
    detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio')) return
    render(request,"SProvider/charts1.html", {'form':chart1,
    'chart_type':chart_type})

def
    View_Prediction_Of_Drug_RecommendationType(requ
    est):obj =drug_recommendation_Type.objects.all()
    return render(request, 'SProvider/View_Prediction_Of_Drug_RecommendationType.html',
{'list_objects':obj})

def likeschart(request,like_chart):
    charts =detection_accuracy.objects.values('names').annotate(dcount=Avg('ratio'))
    return render(request,"SProvider/likeschart.html", {'form':charts, 'like_chart':like_chart})


def Download_Trained_DataSets(request):

    response = HttpResponse(content_type='application/ms-
    excel')# decide file name
    response['Content-Disposition'] = 'attachment;
    filename="TrainedData.xls"'# creating workbook
    wb =
    xlwt.Workbook(encoding='utf-8')#
    adding sheet
    ws=wb.add_sheet("sheet")
```

```
    # Sheet header, first row
    row_num = 0
    font_style =
    xlwt.XFStyle()# headers
    are bold
    font_style.font.bold
    =True
    # writer = csv.writer(response)
    obj =
    drug_recommendation_Type.objects.all()
    data = obj # dummy method to fetch data.
    for my_row in data:
        row_num = row_num
        + 1

        ws.write(row_num, 0, my_row.Drug_Review,
        font_style)ws.write(row_num, 1,
        my_row.Prediction, font_style)
    wb.save(respons
    e)return
    response

def train_model(request):
    detection_accuracy.objects.all().delete
    ()

    df =
    pd.read_csv('Drugs_Review_Datasets.csv')
    df
    df.columns
    df.rename(columns={'rating': 'Rating', 'review': 'Review'}, inplace=True)

    def
        apply_recommend(Rating
        ):if (Rating <= 7):
            return 0  #
        Neagtiveelse:
            return 1  # Positive
    df['recommend'] =
    df['Rating'].apply(apply_recommend)
```

```
df.drop(['Rating'], axis=1, inplace=True)
recommend = df['recommend'].value_counts()

#df.drop(['Id','ProductId','UserId','ProfileName','HelpfulnessNumerator','HelpfulnessDenominator',
'Time','Summar y'], axis=1, inplace=True)

def preprocess_text(text):
   '''Make text lowercase, remove text in square brackets,remove links,remove
   punctuationand remove words containing numbers.''

print(svm_acc)
print("CLASSIFICATION
REPORT")
print(classification_report(y_test, predict_svm))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_svm))
models.append(('svm', lin_clf))
detection_accuracy.objects.create(names="SVM",
ratio=svm_acc) print("Logistic Regression")
from sklearn.linear_model import LogisticRegression
reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train,
y_train)y_pred = reg.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, y_pred) *
100)print("CLASSIFICATION
REPORT")

print(classification_report(y_test, y_pred))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test,
y_pred))models.append(('logistic',
reg))
detection_accuracy.objects.create(names="Logistic Regression", ratio=accuracy_score (y_test,
y_pred) * 100)print("Decision Tree Classifier")

dtc =
DecisionTreeClassifier()
dtc.fit(X_train, y_train)
dtcpredict =
dtc.predict(X_test)
print("ACCURACY")
```

```
print(accuracy_score(y_test, dtcpredict) *
100)print("CLASSIFICATION
REPORT")
print(classification_report(y_test, dtcpredict))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test,
dtcpredict))
models.append(('DecisionTreeClassifier',
dtc))
detection_accuracy.objects.create(names="Decision Tree Classifier", ratio=accuracy_score
(y_test, dtcpredict) * 100)
print("SGD Classifier")
from sklearn.linear_model import SGDClassifier

sgd_clf = SGDClassifier(loss='hinge', penalty='l2', random_state=0)
sgd_clf.fit(X_train, y_train)
sgdpredict =
sgd_clf.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, sgdpredict) *
100)print("CLASSIFICATION
REPORT")
print(classification_report(y_test, sgdpredict))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test,
sgdpredict))
models.append(('SGDClassifier',
sgd_clf))
detection_accuracy.objects.create(names="SGD Classifier", ratio=accuracy_score(y_test,
sgdpredict) * 100)data = 'Lasbeled_Data.csv'
# df['predict_nb'] =
predict_textdf.to_csv(data,
index=False)
df.to_markdown

obj = detection_accuracy.objects.all()
return render(request,'SProvider/train_model.html', {'objs': obj})

  text = text.lower()
  text = re.sub('\[.*?\]', '', text)
```

```
    text = re.sub('https?://\S+|www\.\S+', '',
    text)text = re.sub('<.*?>+', '', text)
    text = re.sub('[%s]' % re.escape(string.punctuation), '',
    text)text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '',
    text)text = re.sub('"@', '',
    text)
    text = re.sub('@', '', text)
    text = re.sub('https: //', '', text)
    text = re.sub('\n\n', '', text)
    text = re.sub('"""', '',
    text)return text
df['processed_content'] = df['Review'].apply(lambda x: preprocess_text(x))

cv = CountVectorizer()
X =
df['processed_content']y
= df['recommend']
print("Review")
print(X)
print("Recmmend
")print(y)
X =
cv.fit_transform(X)
models = []
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20)
X_train.shape, X_test.shape, y_train.shape

print(X_test)
print("Naive
Bayes")
from sklearn.naive_bayes import
MultinomialNBNB = MultinomialNB()
NB.fit(X_train, y_train)
predict_nb =
NB.predict(X_test)
naivebayes = accuracy_score(y_test, predict_nb) *
100print(naivebayes)
print(confusion_matrix(y_test, predict_nb))
```

```
print(classification_report(y_test, predict_nb))
models.append(('naive_bayes', NB))
detection_accuracy.objects.create(names="Naive Bayes",
ratio=naivebayes)# SVM Model
print("SVM")
from sklearn import svm
lin_clf =
svm.LinearSVC()
lin_clf.fit(X_train,
y_train)
predict_svm = lin_clf.predict(X_test)
svm_acc = accuracy_score(y_test, predict_svm) * 100

print(svm_acc)
print("CLASSIFICATION
REPORT")
print(classification_report(y_test, predict_svm))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test, predict_svm))
models.append(('svm', lin_clf))
detection_accuracy.objects.create(names="SVM",
ratio=svm_acc) print("Logistic Regression")
from sklearn.linear_model import LogisticRegression
reg = LogisticRegression(random_state=0, solver='lbfgs').fit(X_train,
y_train)y_pred = reg.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, y_pred) *
100)print("CLASSIFICATION
REPORT")
print(classification_report(y_test, y_pred))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test,
y_pred))models.append(('logistic',
reg))
detection_accuracy.objects.create(names="Logistic Regression", ratio=accuracy_score (y_test,
y_pred) * 100)print("Decision Tree Classifier")
dtc =
DecisionTreeClassifier()
dtc.fit(X_train, y_train)
dtcpredict =
```

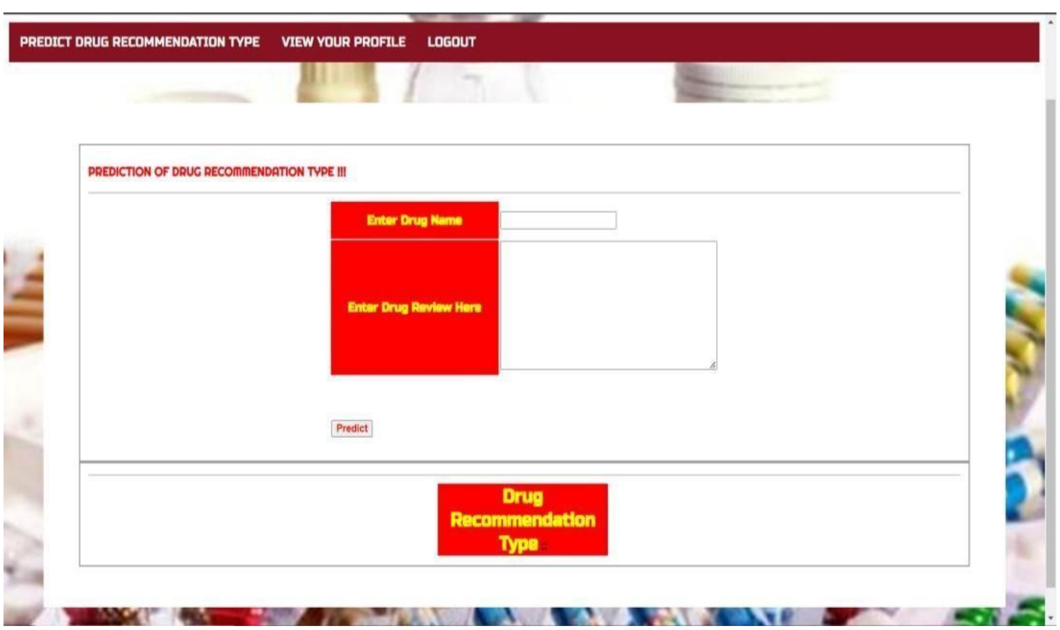
```
dtc.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, dtcpredict) *
100)print("CLASSIFICATION
REPORT")
print(classification_report(y_test, dtcpredict))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test,
dtcpredict))
models.append(('DecisionTreeClassifier',
dtc))
detection_accuracy.objects.create(names="Decision Tree Classifier", ratio=accuracy_score(y_test,
dtcpredict)
* 100)
print("SGD Classifier")
from sklearn.linear_model import SGDClassifier
sgd_clf = SGDClassifier(loss='hinge', penalty='l2', random_state=0)
sgd_clf.fit(X_train, y_train)
sgdpredict =
sgd_clf.predict(X_test)
print("ACCURACY")
print(accuracy_score(y_test, sgdpredict) *
100)print("CLASSIFICATION
REPORT")
print(classification_report(y_test, sgdpredict))
print("CONFUSION MATRIX")
print(confusion_matrix(y_test,
sgdpredict))
models.append(('SGDClassifier',
sgd_clf))
detection_accuracy.objects.create(names="SGD Classifier", ratio=accuracy_score(y_test,
sgdpredict) * 100)data = 'Lasbeled_Data.csv'
# df['predict_nb'] =
predict_textdf.to_csv(data,
index=False)
df.to_markdown

obj = detection_accuracy.objects.all()
return render(request,'SProvider/train_model.html', {'objs': obj})
```

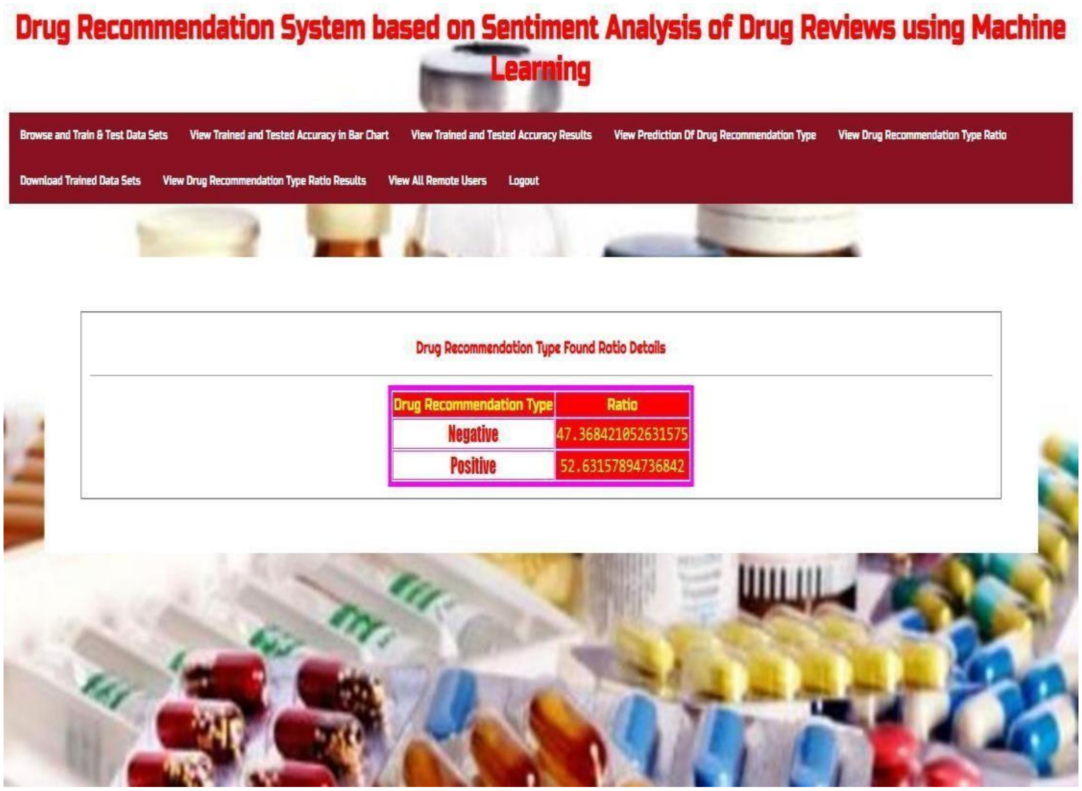
5. SCREENSHOTS

5. SCREENSHOTS

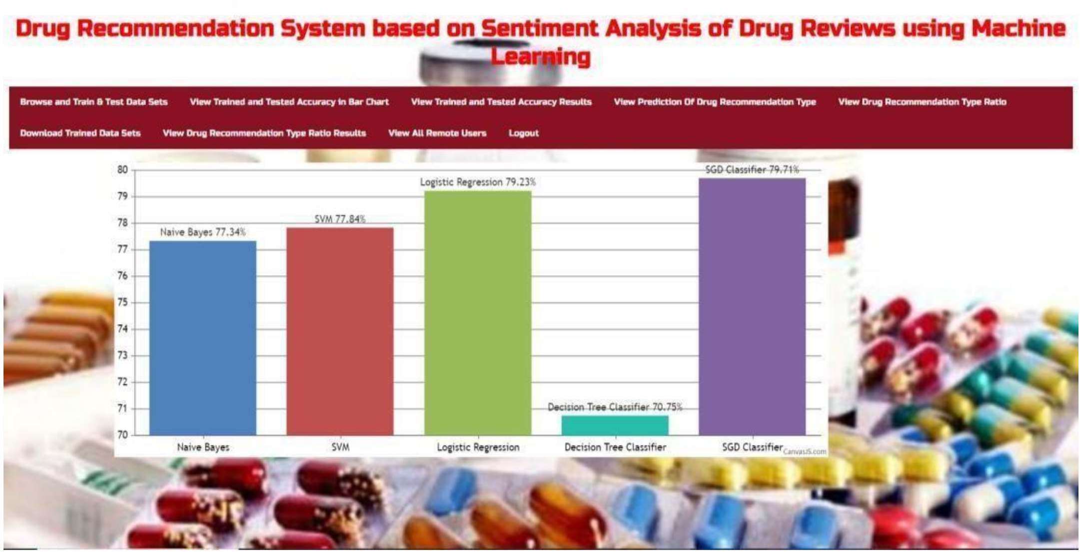
Screenshot 5.1: Drug Recommendation System Application



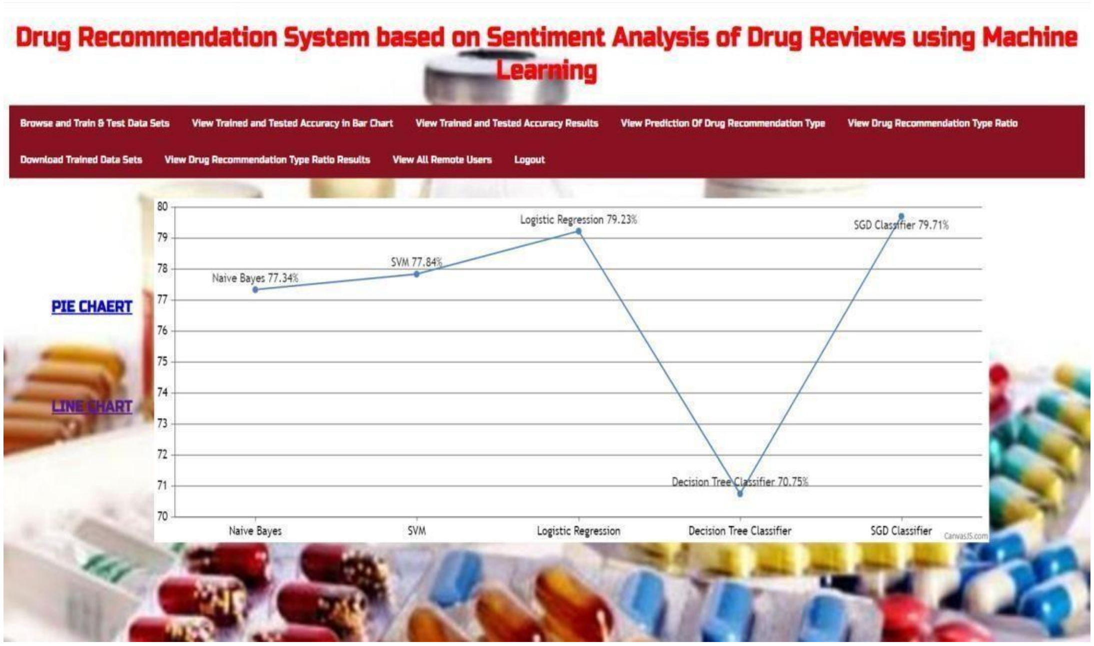
Screenshot 5.2: Drug Prediction



Screenshot 5.3: Drug Recommendation Type Ratio



Screenshot 5.4: Drug Recommendation Trained and Tested Accuracy in bar chart



Screenshot 5.5: Drug Recommendation Trained and Tested Accuracy Results

6. TESTING

6. TESTING

6.1 INTRODUCTION TO 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 of tests. Each test type addresses a specific testing requirement.

6.2 TYPES OF TESTING

6.2.1 UNIT TESTING

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

6.2.2 INTEGRATION TESTING

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

6.2.3 FUNCTIONAL TESTING

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.
Functional testing is centered on the following items:

Valid Input	: identified classes of valid input must be accepted.
Invalid Input	: identified classes of invalid input must be rejected.
Functions	: identified functions must be exercised.
Output	: identified classes of application outputs must be exercised.

Systems/Procedures: interfacing systems or procedures must be invoked. Organization and preparation of functional tests is focused on requirements, key functions, or special test cases.

6.3 TEST CASES

6.3.1 CLASSIFICATION

Test Case ID	Test Scenario	Test Steps	Expected Result	Status
TC001	User Login	1. Enter valid username and password. 2. Click the "Login" button.	Redirect to user profile page	Passed
TC002	User Login - Invalid Credentials	1. Enter invalid username and password. 2. Click the "Login" button.	Display an error message	Passed
TC003	User Registration	1. Enter valid registration details. 2. Click the "Register" button.	User profile is created	Passed
TC004	User Registration - Existing Username	1. Enter an existing username. 2. Click the "Register" button.	Display an error message	Passed
TC005	User Profile Display	1. Log in as a registered user.	User profile page is displayed	Passed
TC006	User Profile Update	1. Log in as a registered user. 2. Update profile information. 3. Click the "Update" button.	Profile information is updated	Passed
TC007	User Profile Update - Invalid Input	1. Log in as a registered user. 2. Enter invalid input in the update form. 3. Click "Update".	Display error messages	Passed
TC008	User Logout	1. Click the "Logout" button.	Redirect to the login page	Passed
TC009	Dataset Upload	1. Log in as an authenticated user. 2. Upload a valid dataset file.	Dataset is uploaded successfully	Passed
TC010	Dataset Upload - Invalid File Format	1. Log in as an authenticated user. 2. Upload an invalid file format.	Display an error message	Passed
TC011	Dataset Upload - Missing File	1. Log in as an authenticated user. 2. Attempt to upload without selecting a file.	Display an error message	Passed
TC012	Perform Sentiment Analysis	1. Log in as an authenticated user. 2. Enter a valid review for analysis. 3. Click "Analyze".	Sentiment analysis result displayed	Passed
TC013	Perform Sentiment Analysis - Empty Review	1. Log in as an authenticated user. 2. Submit an empty review for analysis. 3. Click "Analyze".	Display an error message	Passed
TC014	Perform Drug Recommendation	1. Log in as an authenticated user. 2. Enter valid preferences. 3. Click "Recommend".	List of recommended drugs displayed	Passed

TC015	Perform Drug Recommendation - No Preferences	1. Log in as an authenticated user. 2. Leave the preferences empty. 3. Click "Recommend".	List of recommended drugs displayed	Passed
TC016	User Feedback	1. Log in as an authenticated user. 2. Provide feedback for a recommended drug. 3. Submit.	Feedback is recorded	Passed
TC017	User Feedback - Invalid Rating	1. Log in as an authenticated user. 2. Provide invalid rating (e.g., greater than 5). 3. Submit.	Display an error message	Passed
TC018	User Feedback - Empty Comment	1. Log in as an authenticated user. 2. Submit feedback with an empty comment. 3. Submit.	Display an error message	Passed

7. CONCLUSION

7. CONCLUSION & FUTURE SCOPE

7.1 PROJECT 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 multiplythem bythe normalized useful Count to get the overall score of the drug by condition to build a recommender system.

7.2 FUTURE SCOPE

The future scope of Drug Recommendation System based on Sentimental analysis of Drug Reviews using machine learning is promising and continuous to evolve with advancements in technology and healthcare. Some key areas of future development and growth are:

- Personalized Recommendations
- Real time Updates
- Data Privacy and Security
- Clinical Trails and Research
- Global Expansion

8. BIBLIOGRAPHY

8. BIBLIOGRAPHY

8.1 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.

8.2 GITHUB LINK

https://github.com/bollepallylahari/DrugRecommendationSystem