**Drug Demand prediction using multiple linear regressors**

**A JAVA MINI PROJECT REPORT**

**Submitted by**

**N. Akshith [9919004200]**

**M. Vikas [9919004169]**

**G. Bhanu Prakash [9919004094]**

Under the guidance of

**Mrs. G. Elizabeth Rani M.Tech**

(Assistant Professor, Department of Computer Science and Engineering)

***In partial fulfilment for the award of the degree***

***Of***

**BACHELOR OF TECHNOLOGY**

In

**COMPUTER SCIENCE AND ENGINEERING**

***Of***



**Kalasalingam Academy of Research and Education**

**(Deemed to be an university)**

**Anand Nagar, Krishnankovil – 626 126**

**Academic Year ODD (2020-2021)**

**Kalasalingam Academy of Research and Education**

**(DEEMED TO BE AN UNIVERSITY)**

Under Section 3 of UGC Act - 1956

*Anand Nagar, Krishnankoil – 626 126*



**BONAFIDE CERTIFICATE**

This is to certify that the Mini project titled **“** **Drug Demand prediction using multiple linear regressors ”** is a bonafide record of the work done by **N. Akshith [9919004200], M. Vikas [9919004169] and G. Bhanu Prakash [9919004094]** in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Specialization of the Computer Science and Engineering, during the academic year 2020-2021.

**Mrs.G. Elizabeth Rani Dr.A.FRANCIS SAVIOUR DEVARAJ**

**Supervisor Head of the Department**

**Assistant Professor/ CSE Department of CSE**

**Submitted for the University Viva voce Examination held on 25.11.2020**

**Internal Examiner External Examiner**

**ACKNOWLEDGEMENT**

First and foremost, I wish to thank the **Almighty God** for his grace and benediction to complete this Project work successfully. I would like to convey my special thanks from the bottom of my heart to my dear **Parents** and affectionate **Family members** for their honest support for the completion of this Project work.

I express deep sense of gratitude to “Kalvivallal” Thiru. **T. Kalasalingam** B.com., Founder Chairman, “Ilayavallal” **Dr.K.Sridharan** Ph.D., Chancellor, **Dr.S.ShasiAnanth**, Ph.D., Vice President (Academic) , **Mr.S.Arjun Kalasalingam** M.S., Vice President (Administration), **Dr.R.Nagaraj** Ph.D., Vice-Chancellor, **Dr.V.Vasudevan** Ph.D., Registrar , **Dr.P.Deepalakshmi** M.E., Ph.D., Dean (School of Computing) . And also a special thanks to **Dr.A. Francis Saviour Devaraj** Professor & Head, Department of CSE, Kalasalingam Academy of Research and Education for granting the permission and providing necessary facilities to carry out Project work.

I would like to express my special appreciation and profound thanks to my enthusiastic Project Supervisor **Mrs.G.Elizabeth Rani M.Tech.,** Assistant Professor / CSEof Kalasalingam Academy of Research and Education [KARE] for his inspiring guidance, constant encouragement with my work during all stages. I am extremely glad that I had a chance to do my Project under my Guide, who truly practices and appreciates deep thinking. I will be forever indebted to my Guide for all the time he has spent with me in discussions. And during the most difficult times when writing this report, he gave me the moral support and the freedom I needed to move on.

Besides my Project guide, I would like to thank the rest of Class committee members and all faculty members and Non-Teaching staff for their insightful comments and encouragement Finally, but by no means least, thanks go to all my school and college teachers, well-wishers, friends for almost unbelievable support.

**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **Chapter Number** | **CONTENT** | **Page Number** |
| 1. | 1 | INTRODUCTION | 5 |
| 2. | 2 | FEATURES | 6 |
| 3. | 3 | SYSTEM ANALYSIS | 7 |
| 4. | 4 | SYSTEM DIAGRAM | 8 |
| 5. | 5 | SYSTEM DESCRIPTION | 9 |
| 6. | 6 | MACHINE LEARNING ALGORITHM | 10 |
| 7. | 7 | MODULE | 11 |
| 8. | 8 | SOURCE CODE | 12 |
| 9. | 9 | OUTPUT SCREENSHOT | 21 |
| 10. | 10 | CONCLUSION | 23 |
| 11. | 11 | REFERENCES | 24 |

**CHAPTER 1**

**INTRODUCTION:**

Drug Demand prediction is an application of machine learning on the time series data.

Time series data is uniqueness and charter are to have a repeated pattern in data after a significant or identifiable varying interval.

On a basic observation of drug demand, it is a time series data. If their prediction is wrong then, the scenario will be very harsh such as leading a company to face a heavy loss or company shut down. The Machine learning technology of advanced computer sciences is built to handle these types of scenarios by having a glimpse on the past data. Machine learning is not just limited to the prediction task, it is a very powerful tool in analysing the patterns in data, assistant services and largescale automation tasks with a great performance.

In a parametrical industry environment, for a company it is important to predict the drug demand

General Outline:

Drug demand prediction java project is determined to apply the linear regression algorithm of supervised machine learning to predict the future demand of the drug. Linear regression is simplest of the machine learning algorithms and thus, it also fasts to train and predict results.

This project includes three parts:

1. Front end (GUI)
2. Backend (data parser, calculations, formatting)
3. linear regression class (ml part)

Usage part of this project is aimed to be simple and less error prone. Three main inputs needed to be feed to the GUI framework by user are csv file path, number of forecasts and length of pattern observed data (length of period).

The ml part is modified to understand the pattern in data, with the help of multiple regressors.

**CHAPTER 2**

**FEATURES**

Features of Drug Demand predictor java project: There are many machine learning algorithms that are available for time series data with varying complexity. A basic approach of linear regressors is built to work for data that are corelated data on the basis of the working of linear regressors.

The use of multiple linear regressors can serve in predicting time series data.

The usage of drug demand predictor needs csv file, and two basic details (forecasts count and pattern period length in data).

A csv file is abbreviated as Comma Separated Value document. In this file type, values are stored with a comma (,) as separator or a delimiter.

The components of first part of project (GUI) part are:

1. labels for user to identify the text field input category.
2. Text fields serve to take input from user
3. Button to return the data in text input and call backend class to perform.

The components of second part of project (backend) part are:

A backend class that gets input from the frontend class of java and process it and initiates linear regressor objects and trains.

The components of third part of project (ML) part are:

A linear regression class with aa constructor that takes 1d array of x and y.

It has methods such as predict, get alpha, get beta. Etc.

**CHAPTER 3**

**SYSTEM ANALYSIS**

User has to enter a csv file path which only consists of numeric separated by commas, number of forecasts in data and the periodic pattern length of data.

As this model, applies linear regressors the model has cannot be used for data with heavy noise, that may lead the model bias to the noise. This model works well when the periodic length in forecast is less, the model complexity increases with increase in periodic length of time series data.

The Data processing task in backend part of code includes the task such as reading a csv file with a basic Scanner method available in standard java package. As our document is a csv file, the delimiter parameter of scanner tool is set to “,” character which is used as separator of each numeric value in csv file.

The data feeding task that is passing the data in the format of single dimensional double int array for data input (feature variable) and target array i.e. single dimensional double int array for training (or fitting in terms of linear regression task) liner regression model.

Then, as per the length of periodic interval of a forecast. The model objects that are trained are called to predict using “double predict (double x)” method in the linear regression class of ml part of java project.

The generated predictions are passed to the frontend, which resets the result or output label of frame to create visual representation of the predictions.

Thus, the analysis of the working the java drug demand predictor is produced and written.

**CHAPTER 4**

**SYSTEM DIAGRAM**

This part of the documentation deals with the workflow of the drug demand predictor using java as programming language. As discussed in the previous chapters, the workflow and the project designing is simple and consists of three parts

1. GUI part
2. ML part
3. Backend part



Above figure represents the pictorial view of GUI part

To describe workflow in a diagrammatic representation,

1. The GUI of the project is initiated. Here the user needs to enter the data correctly.
2. On the click of submission button, the backend framework is created and initiated to build multiple regressors and train them with the pre-processed data.
3. The ML part is handled by the backend (here as a middleware program). The ML part has the mathematical implementation and methods of Linear regressors.

**CHAPTER 5**

**SYSTEM DESCRIPTION**

The first part is the GUI / front end part. The second part is backend part and finally, the core part the linear regression class. Overall, the project is aimed to down to earth i.e. simple and forward.

Linear Regressors is aimed to implementation of graphical straight line. The performance of the mode depends on the noise in data.

.

The GUI part is built using the java’s standard library namely swing library. The use of JLabels, JButtons, JTextFields...etc.

The Backend part deals with IO readers and writers’ work. In this project to be simple the use of scanner. Delimiter parameter of scanner object is set to “,” character.

Now scanner object can be used as csv reader which reads one value at a call. The data is read, type casted and store in two single dimensional double data type arrays as input and predicate variable.

To analyse the pattern in forecast, Multiple regressors of count equal to number of inputs in each forecast are trained with data corresponding to the inputs of that count in each forecast are trained to fit to the data.

These ways the project designed analyses the patten and using the fitted linear graph, the predictions are made. The predictions are then represented on the result label section.

**CHAPTER 6**

**MACHINE LEARNING ALGORITHM**

The ML algorithm used in this project is linear regression. Linear Regressors is aimed to implementation of graphical straight line,

Y = m\*x + c

Where m is alpha or slope value and c is intercept to y axis, similarly, is target variable and x is data variable or input variable. The performance of the model is calculate using R square score which ranges from 1 to 0.

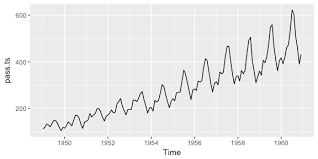
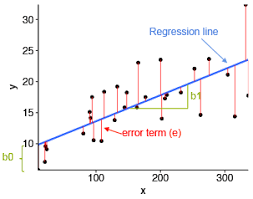


Figure describing the linear regression. (on left side)

Time series data pictorial example (link of [Image source](https://www.google.com/imgres?imgurl=https%3A%2F%2Fuc-r.github.io%2Fpublic%2Fimages%2Fanalytics%2Ftime_series%2Funnamed-chunk-6-1.png&imgrefurl=https%3A%2F%2Fuc-r.github.io%2Fts_exploration&tbnid=V__MHIhY9-flXM&vet=12ahUKEwjOhOiI5KLtAhWZSSsKHR25B20QMygyegQIARBU..i&docid=9xGVBjjCYzY44M&w=576&h=288&q=time%20series%20data&ved=2ahUKEwjOhOiI5KLtAhWZSSsKHR25B20QMygyegQIARBU) (right side))

Single linear regression model can not be used to fit to time series data, so the use of Multiple linear regressors is considered for this project.

To fit the data, the usage of multiple linear regressors is considered for simplicity. The data in each forecast is separated and a multiple data sources are made from the single data source passed. Each data is passed to train with one new linear regression. These ways the pattern in data is analysed.

To optimize, the model various optimizers and regularizer can be used. One of the well-known optimizers is stochastic gradient descent. Regularizer are used to regulate the weights of the model to control overfitting and underfitting nature of model.

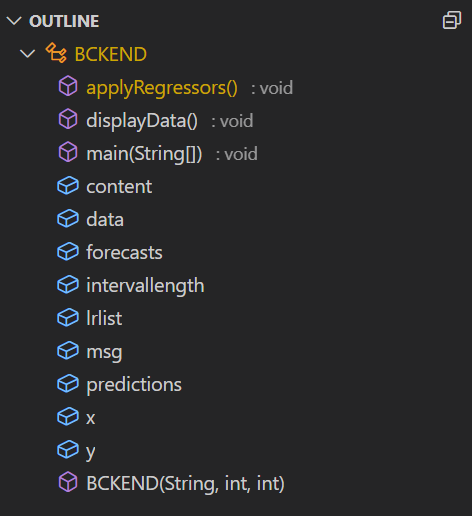
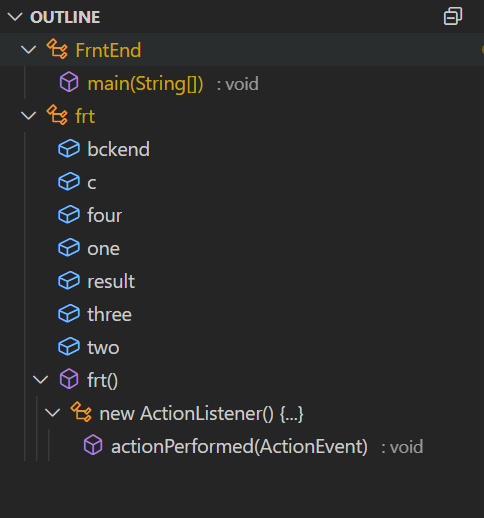
**CHAPTER 7**

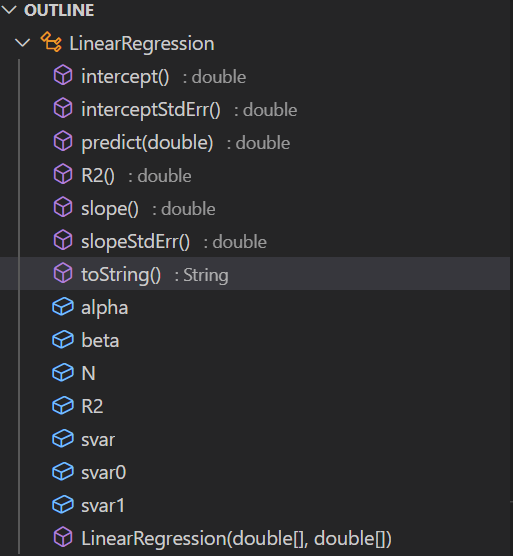
**MODULE**

The project Module consists of three java files.

Each java file’s outline in pictorial representation s shown below:

Frontend java file, Backend java file, Linear Regression java file.





Blue box label represent variables and purple box are methods of class. local variables are ignored.

1. Frontend java class with labels, buttons, and other GUI tools for GUI framework.
2. Linear regression java class with methods and variables as shown above.
3. Backend java class with mentioned methods and variables.

**CHAPTER 8**

**Source code**

The following code describes the

Create a Java file that contains the frontend program that builds the frame for applet or GUI interface.

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

import java.io.FileNotFoundException;

class frt extends JFrame{

Container c;

JTextField one, two, three, four;

BCKEND bckend;JLabel result;

frt() {

this.setLayout(null);

// title

this.setTitle("Drug Demand Prediction");

this.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

this.setBounds(50, 50, 850, 476);

this.setResizable(false);

c = getContentPane();

// head

JPanel heading = new JPanel();

heading.setBackground(new Color(0, 0, 0, 50));

heading.setBounds(0, 0, 970, 80);

JLabel head = new JLabel(" Drug Demand Prediction");

head.setBounds(20, 5, 850, 80);

head.setForeground(Color.white);

head.setFont(new Font("times new roman", Font.ITALIC, 60));

// labels

JLabel csv = new JLabel("CSV File Path :");

csv.setFont(new Font("times new roman", Font.ITALIC, 22));

csv.setForeground(Color.white);

csv.setBounds(10, 90, 260, 28);

JLabel fore = new JLabel("NO.of Forecasts :");

fore.setFont(new Font("times new roman", Font.ITALIC, 22));

fore.setForeground(Color.white);

fore.setBounds(10, 150, 260, 28);

JLabel inter = new JLabel("Period of interval :");

inter.setFont(new Font("times new roman", Font.ITALIC, 22));

inter.setForeground(Color.white);

inter.setBounds(10, 210, 260, 28);

JLabel out = new JLabel("Output for next Periodic :");

out.setFont(new Font("times new roman", Font.ITALIC, 22));

out.setForeground(Color.white);

out.setBounds(10, 270, 260, 28);

result = new JLabel("== output to be calculated ==");

result.setFont(new Font("times new roman", Font.BOLD, 22));

result.setForeground(Color.red);

result.setBackground(Color.white);

result.setBounds(360, 270, 10080, 25);

JLabel algo = new JLabel("Algorithm : ");

algo.setFont(new Font("times new roman", Font.BOLD, 22));

algo.setForeground(Color.white);

algo.setBounds(10, 380, 150, 28);

JLabel algoname = new JLabel("Multiple linear regressions ");

algoname.setFont(new Font("times new roman", Font.BOLD, 22));

algoname.setForeground(Color.white);

algoname.setBounds(165, 383, 200, 25);

JLabel mini = new JLabel("Java Mini Project");

mini.setFont(new Font("times new roman", Font.BOLD, 30));

mini.setForeground(Color.black);

mini.setBounds(580, 406, 250, 28);

// text fileds

one = new JTextField();

one.setFont(new Font("times new roman", Font.BOLD, 16));

one.setForeground(Color.black);

// one.setBackground(Color.blue);

one.setCaretColor(Color.blue);

one.setBorder(null);

one.setBounds(360, 90, 380, 25);

two = new JTextField();

two.setFont(new Font("times new roman", Font.BOLD, 16));

two.setForeground(Color.black);

// two.setBackground(Color.blue);

two.setCaretColor(Color.blue);

two.setBorder(null);

two.setBounds(360, 150, 380, 25);

three = new JTextField();

three.setFont(new Font("times new roman", Font.BOLD, 16));

three.setForeground(Color.black);

// three.setBackground(Color.blue);

three.setCaretColor(Color.blue);

three.setBorder(null);

three.setBounds(360, 210, 380, 25);

// JTextField four = new JTextField();

// four.setFont(new Font("times new roman",Font.BOLD,16));

// four.setForeground(Color.white);

// // four.setBackground(Color.blue);

// four.setCaretColor(Color.white);

// four.setBorder(null);

// four.setBounds(360,270,380,25);

// JTextField five = new JTextField();

// five.setFont(new Font("times new roman",Font.BOLD,16));

// five.setForeground(Color.white);

// // five.setBackground(Color.blue);

// five.setCaretColor(Color.white);

// five.setBorder(null);

// five.setBounds(165,383,200,25);

JButton b = new JButton("Click Here to submit");

b.setBounds(50, 100, 95, 30);

b.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

String msg="";

String fname = one.getText();

int intl = Integer.parseInt(three.getText());

int fore = Integer.parseInt(two.getText());

try {

bckend = new BCKEND(fname, fore, intl);

for(double d :bckend.predictions){

msg+=String.format("%.1f ", d);}

result.setText(msg);

} catch (FileNotFoundException e1) {

result.setText("file not found error");

}

} });

b.setBounds(185,350,200,25);

ImageIcon background\_img = new ImageIcon("imagepath.jpg");

JLabel background\_lbl404 = new JLabel("",background\_img,JLabel.CENTER);

background\_lbl404.setBounds(10,0,860,760);

background\_lbl404.add(heading);

background\_lbl404.add(head);

background\_lbl404.add(csv);

background\_lbl404.add(fore);

background\_lbl404.add(out);

background\_lbl404.add(mini);

background\_lbl404.add(algo);

background\_lbl404.add(inter);

background\_lbl404.add(one);

background\_lbl404.add(two);

background\_lbl404.add(three);

background\_lbl404.add(result);// background\_lbl404.add(four);

background\_lbl404.add(algoname);// background\_lbl404.add(five);

background\_lbl404.add(b);

c.add(background\_lbl404);

this.setVisible(true);

}

}

public class FrntEnd{

public static void main(String[] args) {

frt obj1 = new frt();}}

Create a Java file that contains backend program that makes application of ml to predict the output based on input data.

import java.io.File;

import java.io.FileInputStream;

import java.io.FileNotFoundException;

import java.nio.file.Files;

import java.nio.file.Path;

import java.util.Arrays;

import java.util.Scanner;

public class BCKEND {

double x[][], y[];

String msg, content;

double[][] data;

double predictions[];

int intervallength, forecasts;

LinearRegression lrlist[];

BCKEND(String filename, int forecasts, int intervallength) throws FileNotFoundException {

predictions = new double[intervallength];

data = new double[intervallength][forecasts];

this.intervallength = intervallength;

this.forecasts = forecasts;

Scanner sc = new Scanner(

new File(filename));

sc.useDelimiter(","); // String [] values ;

double[] dValues = new double[forecasts \* intervallength];

int i = 0;

int k = 0;

while (sc.hasNext()) {

double value = Double.valueOf(sc.next());

dValues[i++] = value;

}

sc.close();

for (double d : dValues) {

System.out.printf(" %3.2f", d);

}

i = 0;

while (i < dValues.length) {

for (int j = 0; j < intervallength; j++) {

data[j][k] = dValues[i++];

}

k++;

}

displayData();

this.applyRegressors();

for (int j = 0; j < intervallength; j++) {

predictions[j] = lrlist[j].predict(data[j][forecasts - 1]);

}

for (double d : predictions) {

System.out.printf(" %3.2f", d);

}

}

void displayData() {

for (double[] d : data) {

for (double v : d) {

System.out.println(v + " ");

}

System.out.println();

}

}

void applyRegressors() {

lrlist = new LinearRegression[intervallength];

int i = 0;

double[] x = new double[forecasts - 1];

double[] y = new double[forecasts - 1];

for (int a = 0; a < intervallength; a++) {

for (int j = 0, k = 0; j < forecasts - 1; j++, k++) {

x[j] = data[i][j];

}

for (int j = 1, k = 0; j < forecasts; j++, k++) {

y[k] = data[i][j];

}

lrlist[a] = new LinearRegression(x, y);

i++;

System.out.println(lrlist[a].toString());

}

}

}

Create a Java file that contains the Linear Regression class with proper fields and methods.

public class LinearRegression {

private final int N;

private final double alpha, beta;

private final double R2;

private final double svar, svar0, svar1;

public LinearRegression(double[] x, double[] y) {

if (x.length != y.length) {

throw new IllegalArgumentException("array lengths are not equal");

}

N = x.length;

// first pass

double sumx = 0.0, sumy = 0.0, sumx2 = 0.0;

for (int i = 0; i < N; i++)

sumx += x[i];

for (int i = 0; i < N; i++)

sumx2 += x[i] \* x[i];

for (int i = 0; i < N; i++)

sumy += y[i];

double xbar = sumx / N;

double ybar = sumy / N;

// second pass: compute summary statistics

double xxbar = 0.0, yybar = 0.0, xybar = 0.0;

for (int i = 0; i < N; i++) {

xxbar += (x[i] - xbar) \* (x[i] - xbar);

yybar += (y[i] - ybar) \* (y[i] - ybar);

xybar += (x[i] - xbar) \* (y[i] - ybar);

}

beta = xybar / xxbar;

alpha = ybar - beta \* xbar;

// more statistical analysis

double rss = 0.0; // residual sum of squares

double ssr = 0.0; // regression sum of squares

for (int i = 0; i < N; i++) {

double fit = beta \* x[i] + alpha;

rss += (fit - y[i]) \* (fit - y[i]);

ssr += (fit - ybar) \* (fit - ybar);

}

int degreesOfFreedom = N - 2;

R2 = ssr / yybar;

svar = rss / degreesOfFreedom;

svar1 = svar / xxbar;

svar0 = svar / N + xbar \* xbar \* svar1;

}

public double intercept() {

return alpha;

}

public double slope() {

return beta;

}

public double R2() {

return R2;

}

public double interceptStdErr() {

return Math.sqrt(svar0);

}

public double slopeStdErr() {

return Math.sqrt(svar1);

}

public double predict(double x) {

return beta \* x + alpha;

}

public String toString() {

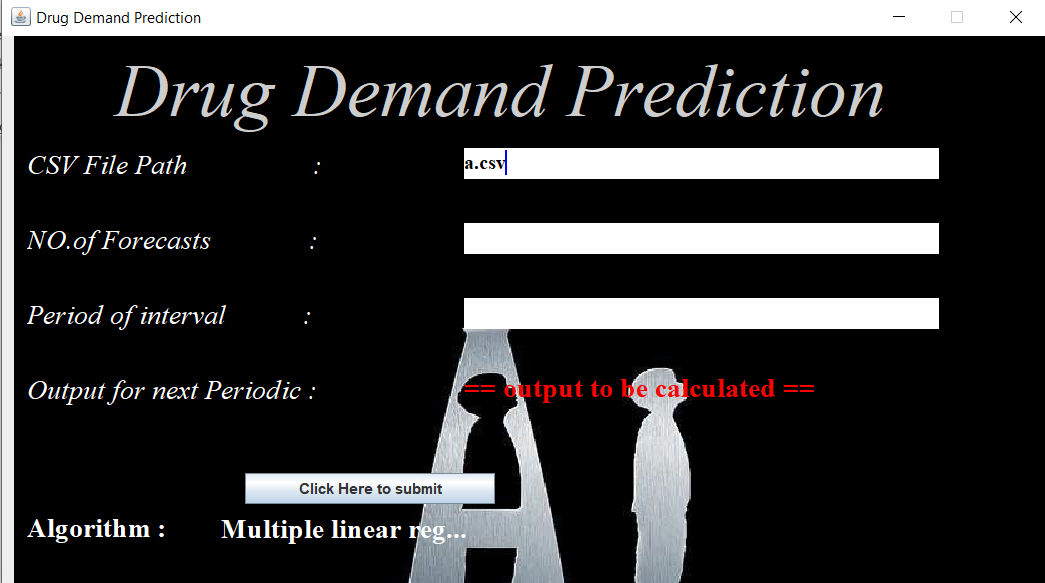
String s = "";

s += String.format("%.2f N + %.2f", slope(), intercept());

return s + " (R^2 = " + String.format("%.3f", R2()) + ")";}}

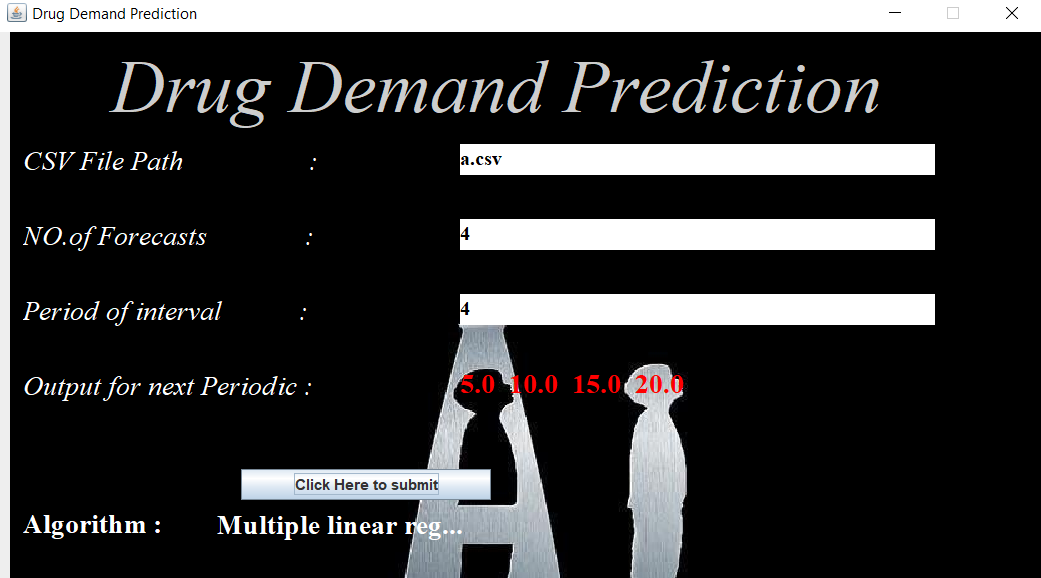
**CHAPTER 9**

**PROJECT OUTPUT SCREENSHOT:**









Last figure has the predictions in label with red colour as font colour.

**CHAPTER 10**

**CONCLUSION:**

So as a part of conclusion of the project, this section provides a general overview in two paraphs.

This project is built using the java project language which is implemented with OOP concept. This project has used swing library tools for GUI programming and custom designed backend framework and Linear regression java class.

Multiple Linear regressors are used to analyse the time series data of drug demand.

**Difficulties:**

Usage of multiple linear regressors make the complexity increase with the increase in the length of pattern. Due to simplicity of linear regression, it cannot handle the heavy noise data to provide results accurately. This model is used with data of less noise and no error. The csv file has to have no heading just numeric data in one row format and numerical separated buy commas.

**CHAPTER 11**

**REFERENCES**

The following books were very helpful during the completion of project:

1. Core Java Volume-1

Author : Car S. Horstmann

1. Effective Java

Author : Joshua Bloch

1. Head First Java

Author : Kathy Sierrab & Bert Bates

1. Java complete reference 11 th edition

Author : Herbert Schildt