PROJECT REPORT

ON

RECOMMENDATION SYSTEM FOR BETTER STUDENT PERFORMANCE IN CAMPUS

SUBMITTED BY

CHINMAY CHAUGHULE

SIDDHANT SHINDE

KUNAL TEMKAR

UNDER THE GUIDENCE OF

Prof. NEHA THAKUR

DEPARTMENT OF INFORMATION TECHNOLOGY

DATTA MEGHE COLLEGE OF ENGINEERING

SECTOR -3, AIROLI, NAVI MUMBAI- 400 708, (M.S.), INDIA

2019-2020

PROJECT REPORT

ON

RECOMMENDATION SYSTEM FOR BETTER STUDENT PERFORMANCE IN CAMPUS

SUBMITTED TO THE

UNIVERSITY OF MUMBAI, MUMBAI

Proposed to be submitted in the partial fulfillment of requirement for the

Degree of Bachelor of Engineering in Information Technology

Submitted by

CHINMAY CHAUGHULE
SIDDHANT SHINDE
KUNAL TEMKAR

UNDER THE GUIDENCE OF

Prof. NEHA THAKUR

DEPARTMENT OF INFORMATION TECHNOLOGY

DATTA MEGHE COLLEGE OF ENGINEERING

SECTOR -3, AIROLI, NAVI MUMBAI- 400 708, (M.S.), INDIA

2019-2020

Datta Meghe College of Engineering

(AICTE & Govt. of Maharashtra Recognized, Affiliated to University of Mumbai)

Department of Information Technology

C E R T I F I C A T E

Date:

This is to certify that, the project work embodied in this report entitled, "**'RECOMMENDATION SYSTEM FOR BETTER STUDENT'S PERFORMANCE IN CAMPUS**" submitted by CHINMAY CHAUGHULE, SIDDHANT SHINDE, KUNAL TEMKAR for the award of **Bachelor of Engineering (B.E.)** degree in the subject of **Information Technology**, is a work carried out by them under my guidance and supervision within the institute. The work described in this project report is carried out by the concerned student/s and has not been submitted for the award of any other degree of the University of Mumbai.

Further, it is certify that the student/s was regular during the academic year and has worked under the guidance of concerned faculty until the submission of this project work at the Datta Meghe College of Engineering.

Signature of the Guide **Signature of Head of Department** **Signature of Principal**

DATTA MEGHE COLLEGE OF ENGINEERING

Plot no 98, Cidco, Sector-3 Post Box No.15 Airoli, Navi Mumbai-400708

CERTIFICATE OF APPROVAL

Project Entitled: RECOMMENDATION SYSTEM FOR BETTER STUDENT'S PERFORMANCE IN CAMPUS

SUBMITTED BY

CHINMAY CHAUGHULE

SIDDHANT SHINDE

KUNAL TEMKAR

In the partial fulfillment of the degree of B.E in "INFORMATION TECHNOLOGY" is approved.

Signature of Internal Examiner **Signature of External Examiner**

Signature of Head of Department **Signature of Principal**

DECLARATION BY THE CANDIDATE

We wish to declare that the work embodied in this project report entitled " **RECOMMENDATION SYSTEM FOR BETTER STUDENT'S PERFORMANCE IN CAMPUS"** forms our own contribution to the project work carried out under the guidance of **NEHA THAKUR** Asst. Prof., Department Of Information Technology, Datta Meghe College Of Engineering, Airoli, Navi Mumbai, affiliated to University Of Mumbai.

This work is carried out, written, compiled and submitted by us for the award of degree of Bachelor of Engineering in Information Technology at the University of Mumbai. This work has not been submitted for any other degree of this University or any other University.

Mumbai _____

Date: Signature of Candidate/s

Acknowledgement

We would like to express our sincere humble, deep sense of a graduate to our project guide **Prof. Neha Thakur** for her counsel and consecutive guidance, active interest and constant encouragement. It would not have been possible for us to complete this project work done until now without her critical analysis and valuable guidance.

We are thankful to faculty and staff members, to **Prof. Shila Jawale** and especially our **Head of Department Prof. Satish Devane** for his cooperation and encouragement words during the project completion. His active interest and continuous directional path guides and motivate us to choose the right flow for the project.

Beside we are thankful to management and **Principal Prof. Sudhir Sawarkar** and we would like to thanks many other individuals at College who are contributed greatly in the completion of the project.

Chinmay Chaughule -05

Siddhant Shinde -42

Kunal Temkar -46

Abstract.

Placement opportunities are abundant in market due to many emerging technologies. Skill development of students should be done in placement point of view. Detail analysis of students should be done according to their previous year marks, their communication skills, and their curricular and extra-curricular achievements. Using this analysis better steps can be taken to improve student's performance and placement in particular institute. Data Mining algorithms can be used to tackle voluminous data of the student in educational institute and to predict student's placement in better way. Also, with this the students can be classified according to the company's criteria and those who are not in the list will be assessed on their weakness. This helps the TNP staffs and will get the recommendation of the students list for the particular company so that they can train the people more to that direction. We have used Collaborative filtering for the recommendation of the students list to TNP side. Also, Naive Bayes algorithm have been used to predict the placement chances of particular students based on historical data.

Keywords: Placement, Data Mining, Recommendation, Naïve bayes

TABLE OF CONTENTS

List of Figures

CHAPTER NO – 1

INTRODUCTION

1.1 OVERVIEW:

Students studying in final or third year of an Engineering college start feeling the pressure of the placement season with so much of placements activities happening around them. They feel the need to know where they stand and how they can improve their chances of getting job. Students are supposed to get guided through the placement point of view. Education institute contains each and every data about students in different views like personal information, academic information, personality as well as their achievements in different location. Using data mining, huge data of students can be analyzed in efficient manner to get some desirable results with respect to student's placement in institute.

Data Mining has powerful classifiers to predict how many students will be placed and which students will get placed in particular year and accordingly students will be trained with more attention.

This can help teachers to categorize students-accordingly some students may be given more attention in terms of aptitude, technical skills, presentation skills, communication skills, and so on by arranging extra technical lectures, aptitude tests and presentation skill development seminars.

1.2 BACKGROUND:

Every year there are a large number of companies visiting any institute for campus Placement opportunities are abundant in market due to many emerging technologies, Skill development of student should be done in placement point of view. Similarly, for improving the success factor of placement, the factors should be known so that the eligibility can be checked.

The scope of this project is to predict whether the student will get placed or not based on student's academics scores, their communication skills, their curricular and extracurricular achievements. Also, this system will able to classify the students according to the different company's criteria and their requirements based on the past record of the placed students accordingly.

This system can make us realize the importance of student's eligibility criteria in various companies based on their requirements. As, this will also help students in the beginning to starting preparing for the companies in advance. This system will also be very helpful for TPO to sort out eligible candidates.

Data Mining leverages the computing power of today's servers to transform mountains of data into useful information. Until very recently, only large corporations and universities with access to super-computers could perform useful data-mining tasks but as powerful servers become available and more affordable as compared with the price of a supercomputer –smaller companies are able to harness their server power to mine their stored data in order to gain a competitive edge in the marketplace.

1.3 PROBLEM STATEMENT:

The main objective of this model is to predict whether the student gets placed or not in campus recruitment. For this the data consider is the academic history of students like overall percentage, backlogs, and based on various other activities such as communication skills, certification courses, logical skills etc.

The eligibility criteria of students in various companies is more important and this can be realized by this model. This will help everyone as beginning from students they will prepare for companies in advance. This system can classify the student knowledge with ease and can be useful for organization in predicting his/her chances of placement. There are several classification algorithms and mathematics-based techniques which can be taken nearly as good assets for classifying the students' information set in the education field.

In Our system, Naive Bayes algorithm is applied to predict student performance which can facilitate to identify performance of students and also provides suggestion to improve performance for students such as we are going to classify the student's knowledge set for placement and non-placement classes based on that result, education organizations can give superior training to their students. Based on data received by system, student's performance is analyzed in numerous views to check the achievements of the students through their activities and suggests improvement for better placement.

1.4 OBJECTIVE AND PURPOSE:

The objective of this project is to have a system which helps the student to screen their knowledge w.r.t placement studies and to increase their performance based on it. Also, this system recommends the TPO of our college with the name of students who are eligible for the particular company based on previous data. For changing eligibility criteria of the company, the model evolution has been done so that new data will be used to train the machine.

The system primary aim of this work is to build a classification model by using various data mining algorithm which classifies the placement prediction of the students in percentage. Collaborative filtering is used to get the similarities of the previous placed students and accordingly new students will get recommended as a eligible student for the respective company.

1.5 PERIOD OF THE PROJECT:

The time period of the project is one and a half years. Duration is from December 2018 to February 2020 (Semester 6,7,8)

Semester 6 :

In semester 6, Review of literature was done. Here, we read different published papers on placement prediction by various data mining algorithm.

Semester 7:

In Semester 7, we started implementation part and added various features relevant to the topic under the guidance of project incharge.

Semester 8:

In semester 8, after completing the model building part, we started designing Graphical user interface for our project i.e. website for TPO as well as students. Some testing has been done by checking each components of our project and have done prediction on our team members data and observed the results for testing purpose.

1.6 SCOPE OF THE PROJECT:

Every year there are a large number of companies visiting any institute for campus Placement opportunities are abundant in market due to many emerging technologies, Skill development of student should be done in placement point of view. Similarly, for improving the success factor of placement, the factors should be known so that the eligibility can be checked.

The scope of this project is to predict whether the student will get placed or not based on student's academics scores, their communication skills, their curricular and extracurricular achievements. Also, this system will able to classify the students according to the different company's criteria and their requirements based on the past record of the placed students accordingly.

This system can make us realize the importance of student's eligibility criteria in various companies based on their requirements. As, this will also help students in the beginning to starting preparing for the companies in advance. This system will also be very helpful for TPO to sort out eligible candidates.

1.7 PROJECT SPECIFICATIONS:

There are various applications of this system, few of them being. Students will have an idea of where they stand and what to do next to bridge the gap and become better. Students will have a clear option which will help reduce the ambiguity in their mind. The college will have the statistics of all the students and what are the different domains they fall into. The college, will be able to take decisions to improve students and have better, insights of the students. The student's will get their resume based on the data they feed. The corporate end, will be able to filter the students and download the resumes of the students, according to their needs. The corporate end and the college end will be able to post there, requirements and send messages directly to the student, or maybe even globally. The corporate end there will be interview questions that will be prompted, different for different students based on the student resume.

The questions this work can provide the solutions to, can be given as follows:

1. What are the types of students the college has according to their academic scoring?

2. Predict in advance, the placement status of the pre - final year students.

3. What should the students learn next, to have a better chance of placement?

4. What are the clusters of domains, students in a college fall into?

5. Provide info to the hiring companies and prompt questions and proof of the student details and help companies' graphs and visuals to filter students.

So, we aim to fix the above vulnerabilities and answer those questions in our system, using artificial neural networks.

CHAPTER NO- 2

LITERATURE REVIEW

In Literature review, we discuss about the various aspects of the project by taking reference of the existing projects that are similar to the makers of this current project.

The Naive Bayes classifier is one of the simplest approaches to the classification task that is still capable of providing reasonable accuracy. Using Naive bayes, decision tree approach. The data is classified and evaluated the KSA score. It depends on quality of factor that can predict the student placement [1]. For placement of students' knowledge discovery and data mining (KDD) method is used. It includes various steps and the paper propose 6 different data mining algorithms for checking which is fittest among them [2]. This can be further used for assist teacher prediction for student class.

In [3]. A large amount of datasets C4.5 Algorithm, ID3 Algorithm, KNearest Neighbour Algorithm, Support vector-machine algorithm, etc. data mining methods are applied. Using tools like WEKA, Orange, and student data is analyzed. This have helped in predicting student's performance with the help of Data Mining Techniques, Machine Learning Algorithms and Statistical techniques and approaches. In[4] Student data is used to compute the distribution of placement class and it can be used to predict the students' placement in various company. In[5] basically two algorithms are used for predicting the placement of student based on some attribute values.

1.*C 4.5 Algorithm:* The decision trees generated by C4.5 can be used for classification, and for this reason, C4.5 is often referred to as a statistical classifier. C4.5 builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy.

2. *Naive Bayesian Classification:* Naive Bayes Classifiers are basic classifiers that work on probabilistic values based on the Bayes' theorem. This algorithm requires a number of parameters linear in number of features/variables. Bayesian classification gives a practical knowledge and prior information on learning algorithms. It calculates probabilities for hypothetical values and is robust to noise in input.

In [6] the dataset is not only about curricular but also about technical skills, communication skills & co-curricular. Analyzing the data for the fuzzy logic and k nearest neighbor (KNN) for finding fittest solution. The accuracy obtained after analysis for KNN is 97.33% and for the

Fuzzy logic is 92.67%. Hence, from the above said analysis and prediction it would be better if the KNN is used to predict the placement results. In[7]. Logistic Regression method had been used here. This model had categorized the students on the basis of likelihood to get placed or not. Placement and academic data provided (by their respective campus) is used in this model. Machine learning technique is also used to design and implement a logistic classifier which predicts the probability of the students for getting placed in company. The scope of this model is, it can predict the lagging skills of each student.

Machine Learning is one of the technique they had used here [8]. Machine Learning deals with the development as well as analysis and had studied the algorithms that had automatically detected the patterns from student's data and used it to predict future data. In this model, they have made use of classification trees, a typical decision tree in which the predictor variable i.e target variable which takes on finite set of categorical values only. This model helps the placement cell within an organization to identify the prospective students and pay attention in it so as to improve their technical as well as interpersonal skills. In[9] Bayes theorem is used in decision making and it uses the knowledge of prior events to predict future status .In this model, they had used some attributes of students for classification. Attributes such as Attendance, GPA, Reasoning, Quantitative, Communication and Technical skills are used for predicting Placements. And further they have calculated the probabilities from the datasets. Also, they have calculated the datasets with the help of WEKA and Rapid Miner tool for visualization.

2.1.1 ID3 classification algorithms:

Hitarthi Bhatt identified relevant attributes based on quantitative and qualitative aspects of a student's profile such as CGPA, academic performance, technical and communication skills and designed a model which can predict the placement of a student using ID3[1].

2.1.2 Classification algorithms:

S. Taruna and Mrinal Pandey implemented an empirical analysis on predicting academic performance by using classification techniques or mapping of data items into predefined groups and classes using supervised learning. They compared five classification algorithms namely Decision Tree, Naïve Bayes, Naïve Bayes Tree, K-Nearest Neighbour and Bayesian Network algorithms for predicting students' grade particularly for engineering students using a four-class prediction problem [6]. State of the art regression algorithms Kotsiantis and Pintelas,

2005 predicted the student marks (pass and fail classes) using the regression methods and available previous data. The scope of this work compares some of the state of the art regression algorithms in the application domain of predicting students' marks. A number of experiments have been conducted with six algorithms, which were trained using datasets provided by the Hellenic Open University [7].

2.1.3 Logistic Regression:

Saha and Goutam applied logistic regression method on the examination result data and analyzed the data under the University Grant Commission sponsored project entitled - Prospects and Problems of Educational development (Higher Secondary Stage) in Tripura - An In-depth Study.

2.1.4 Classification Algorithms:

Syed Tanveer Jishan, Raisul Islam Rashu, Naheena Haque and Rashedur M Rahman predicted student's performance using attributes such as Cumulative Grade Point Average, Quiz, Laboratory, Midterm and Attendance marks. Also, in order to improve the prediction model, they introduced some preprocessing techniques so that the prediction model provides with more precise results by applying three classification algorithms, e.g., Naïve Bayes, Decision Tree and Neural Network [9].

2.1.5 Decision tree algorithm C4.5:

Zhiwu Liu and Xiuzhi Zhang used decision tree algorithm C4.5 to establish a classification rule and an analysis-forecasting model for students' marks. They described how the analysisforecasting result can be used to find out the factors which can affect students' marks, so some negative learning habits or behaviors of students can be revealed and corrected in time and the teaching effect of the teacher can be checked, the teaching management can also be assisted [10].

2.1.6 Data mining techniques:

The initial results from Kabakchieva and Dorina's implemented research project aimed to show the high potential of data mining applications for university management. This paper is focused on the implementation of data mining techniques and methods for acquiring new knowledge from data collected by universities. The main goal of the research is to reveal the high potential of data mining applications for university management [12].

2.1.7 ID3 and C4.5 classification algorithms:

Kalpesh Adhatrao, Aditya Gaykar, Amiraj Dhawan, Rohit Jha and Vipul Honrao analyzed the data set containing information about students, such as gender, marks scored in the board examinations of classes X and XII, marks and rank in entrance examinations and results in first year of the previous batch of students. By applying the ID3 and C4.5 classification algorithms on this data, they have predicted the general and individual performance of freshly admitted students in future examinations [13].

CHAPTER 03

METHODOLOGY

3.1 TOOLS AND TECHNOLOGIES

A. Pycharm

PyCharm is an Integrated Development Environment (IDE) that provides tools for writing efficient code. It supports various frameworks such as Django, Flask, and Pyramid. PyCharm is majorly used in our project for application development with Python language. Developing such application using Python language felt easier than developing it in any other language. This is due to Python's literal syntax and the IDE used for development. An IDE provides an environment consisting of various useful tools, packages, and libraries. It helps in creating efficient programs for our project.

Some of the benefits of using it are**:**

- Auto-completion of code
- Excellent debugging
- Project navigation
- Database tools
- Support for web development

B. Sublime text 3

Sublime Text editor is used as an Integrated Development Editor (IDE) like Visual Studio and NetBeans. It natively supports many programming languages and markup languages, and functions can be added by users with plugins, typically community-built and maintained under free-software licenses. The present form of Sublime Text editor is 3.0 and is perfect with different operating systems like Windows, Linux and MacOS. In this project, we have used sublime text 3 for front end development purpose using Html and CSS.

Sublime Text offers us the following advantages –

- Ability to solve linker errors.
- Keeping track of all files and folders to work with.
- Problem solving capabilities.
- Keeping color combination for syntax combination.

C. Python

Python is a simple, general purpose, high level, and object-oriented programming language.Python is an interpreted scripting language also. Virtual environments help create separate Python setups while sharing a system-wide base install, for ease of maintenance. Virtual environments have their own set of private site packages (i.e. locally-installed libraries), and are optionally segregated from the system-wide site packages.
In order to create a program that predicts student's placement possibility , we need to use some of the very useful python packages. We have installed the following packages:

1. NumPy

NumPy is the fundamental package for scientific computing with Python. It contains among other things:

- a powerful N-dimensional array object
- sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, NumPy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases that deals with variety of datasets which includes categorical as well as numerical values.

2. Pandas

Pandas is the utility belt for data analysts using python. The package centers around the pandas DataFrame, a two-dimensional data structure with indexable rows and columns. It has effectively taken the best parts of Base R, R packages like plyr and reshape2 and consolidated them into a single library.
Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Key Features of Pandas

- Fast and efficient Data Frame object with default and customized indexing.
- Tools for loading data into in-memory data objects from different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and providing of data sets.

3. Matplotlib

Matplotlib is an amazing visualization library in Python for 2D plots of arrays. Matplotlib is a multi-platform data visualization library built on NumPy arrays and designed to work with the broader SciPy stack. Matplotlib has a procedural interface named the Pylab, which is designed to resemble MATLAB, a proprietary programming language developed by MathWorks. Matplotlib along with NumPy can be considered as the open source equivalent of MATLAB. Matplotlib and its dependency packages are available in the form of wheel packages on the standard Python package repositories and can be installed on Windows, Linux as well as MacOS systems using the pip package manager.

4. Sklearn

Scikit learn is a library used to perform machine learning in Python. Scikit learn is an open source library which is licensed under BSD and is reusable in various contexts, encouraging academic and commercial use. It provides a range of supervised and unsupervised learning algorithms in Python. Scikit learn consists popular algorithms and libraries. Scikit-learn is largely written in Python, and uses numpy extensively for high-performance linear algebra and array operations. Furthermore, some core algorithms are written in Cython to improve performance. Support vector machines are implemented by a Cython wrapper around LIBSVM; logistic regression and linear support vector machines by a similar wrapper around LIBLINEAR. In such cases, extending these methods with Python may not be possible.

3.2 DFD DIAGRAM

- **DFD 0 (Level 0)**

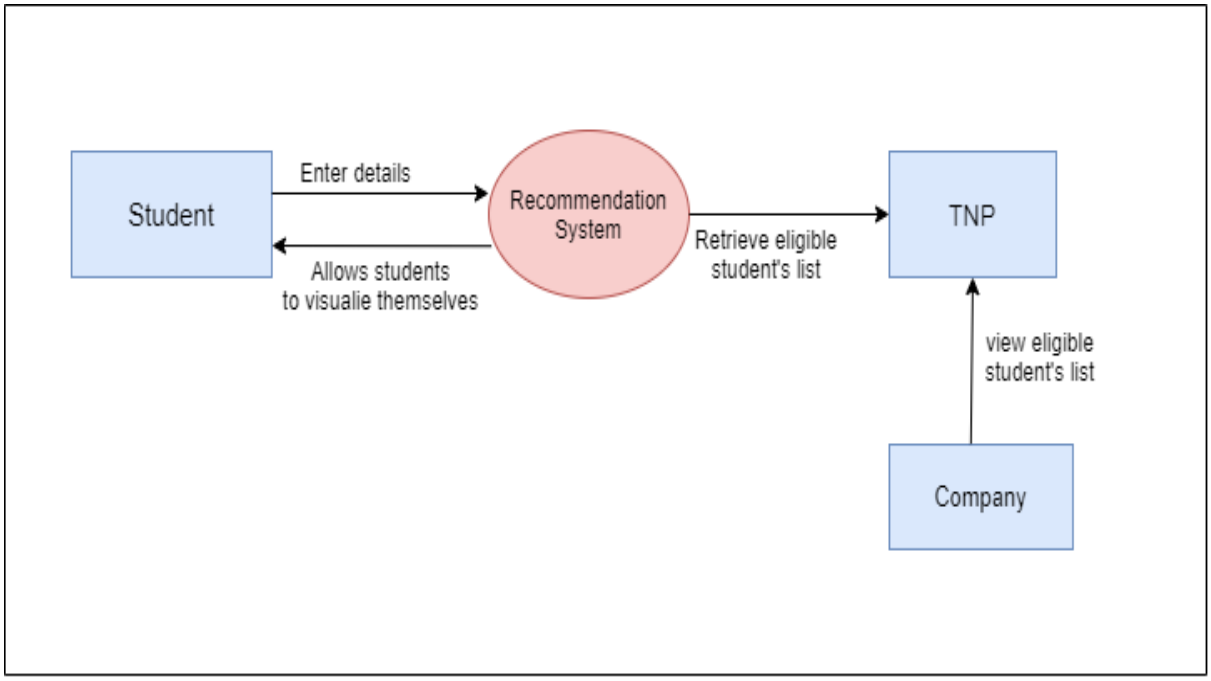


Fig 3.2(a) Data Flow Diagram(DFD 0)

❖ **DFD 1 (Level 1)**

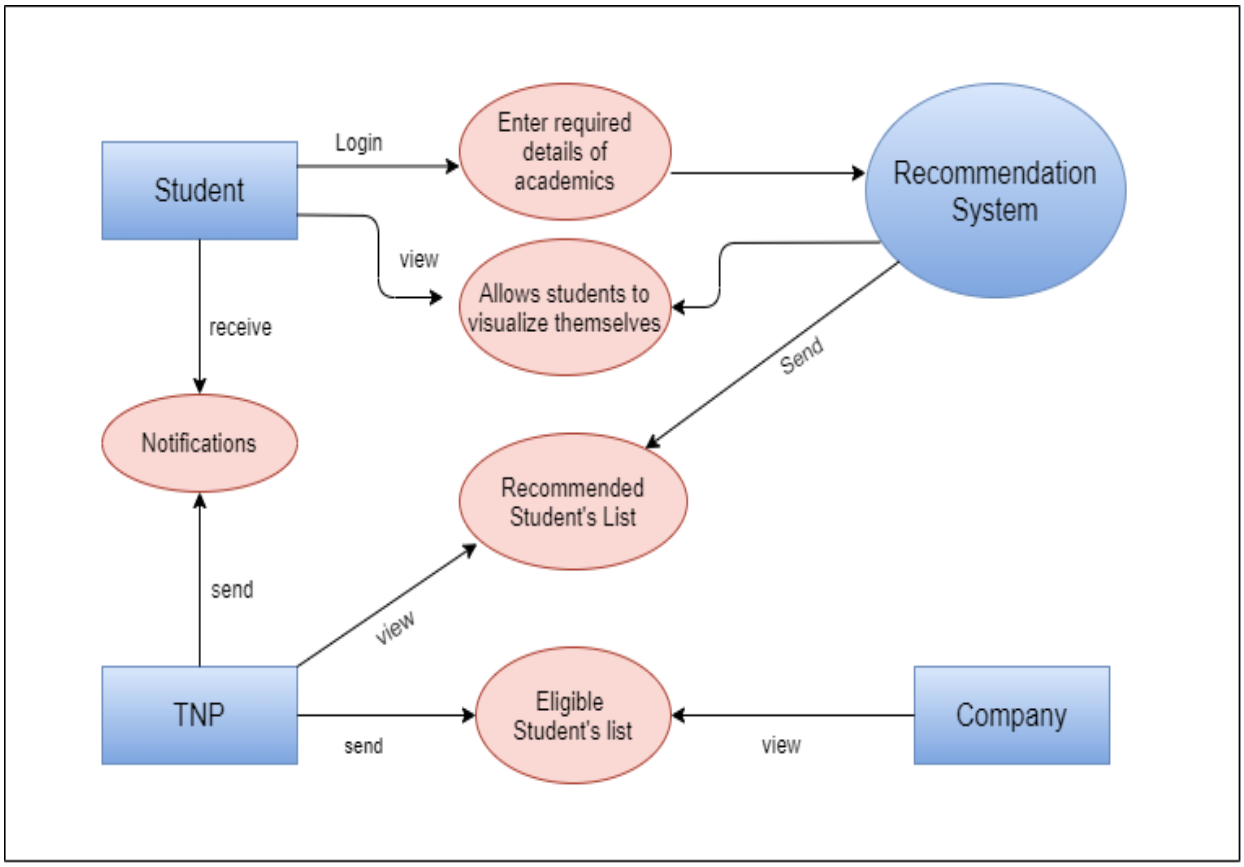


Fig 3.2(b) Data Flow Diagram(DFD 1)

3.3 SEQUENCE DIAGRAM

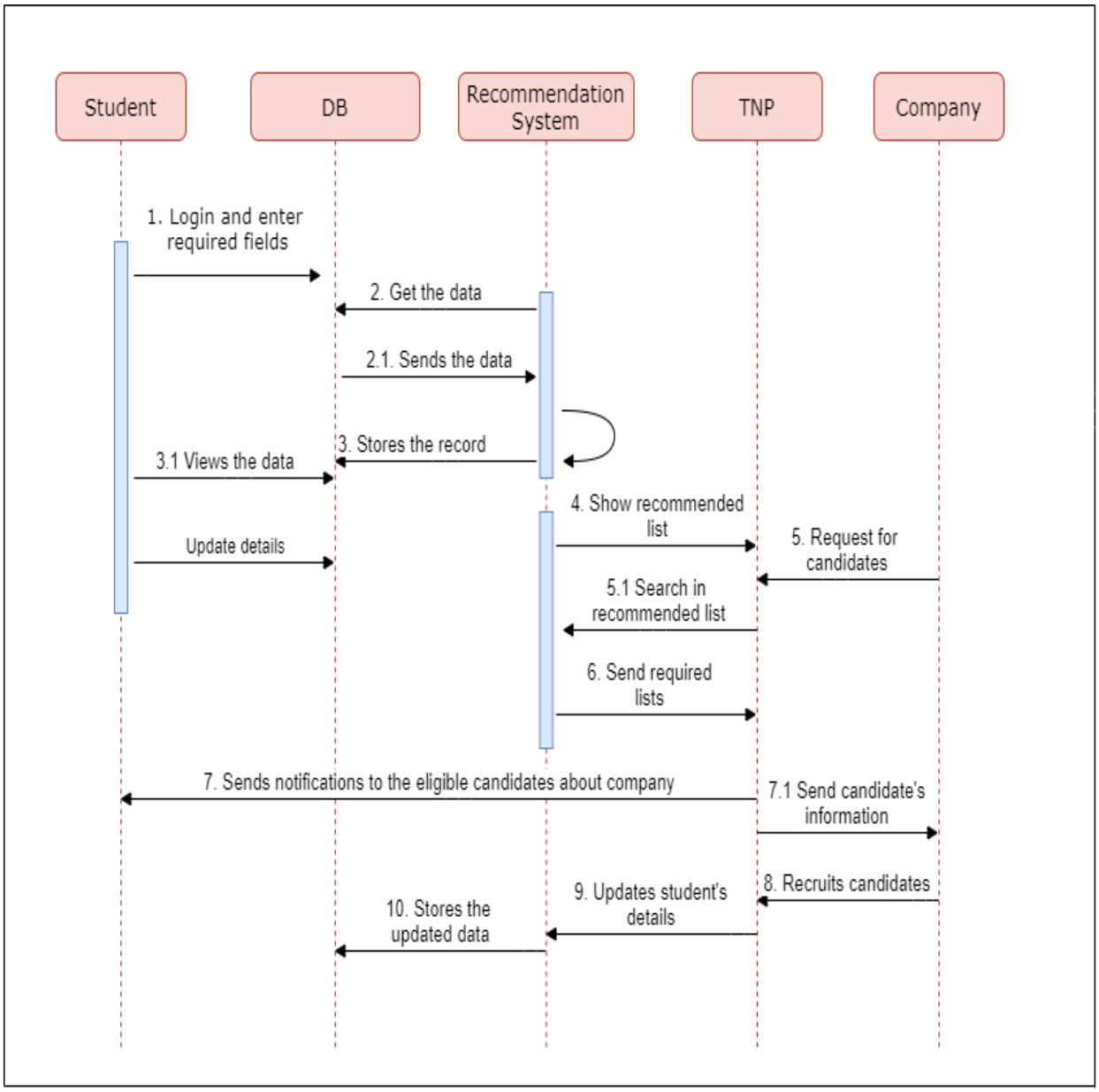


Fig 3.3 Sequence Diagram

3.4 USE CASE DIAGRAM

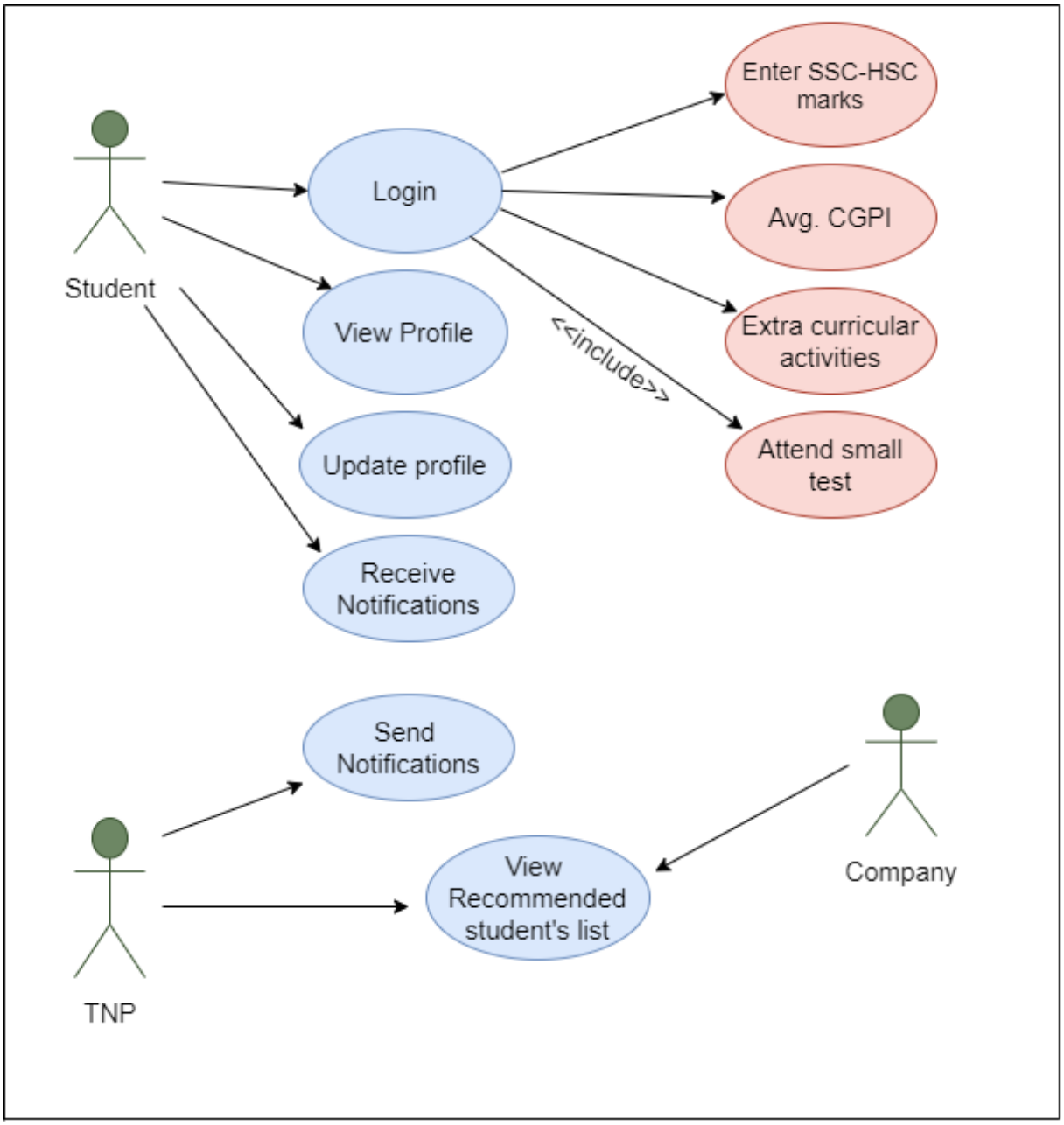


Fig 3.4 Use Case Diagram

CHAPTER 04.

IMPLEMENTATION

Python Code

```
import csv
import gzip
import pickle

from flask import Flask, render_template, request, url_for, session
from flask_mail import Mail,Message
import random,copy
import numpy as np
import pandas as pd
from matplotlib.ticker import PercentFormatter
from pandas import DataFrame
import sqlite3
from math import sqrt
import io
import base64
import matplotlib
matplotlib.use('Agg')
import matplotlib.pyplot as plt
from matplotlib.backends.backend_agg import FigureCanvasAgg as FigureCanvas
from matplotlib.figure import Figure
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.preprocessing import LabelEncoder
from sklearn import datasets, linear_model,metrics,tree
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC


from sklearn.externals import joblib
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics import confusion_matrix
from sklearn.preprocessing import LabelEncoder
from sklearn import datasets, linear_model,metrics,tree
from sklearn.svm.libsvm import predict_proba
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import GaussianNB
from sqlalchemy import null
msg1="null"
```

```
app=Flask(__name__)
mail=Mail(app)
app.config['MAIL_SERVER']='smtp.gmail.com'
app.config['MAIL_PORT']=465
app.config['MAIL_USERNAME']='chaughule20.cc@gmail.com'
app.config['MAIL_PASSWORD']='saibaba1918'
app.config['MAIL_USE_TLS']=False
app.config['MAIL_USE_SSL']=True
app.config['SEND_FILE_MAX_AGE_DEFAULT'] = 0
mail=Mail(app)
app.secret_key='chinmay'
@app.route('/notify',methods=['POST'])
def notify():
    global text
    msg=Message('Placement
Recommendation',sender='chaughule20.cc@gmail.com',recipients=['chaughule20.cc@g
mail.com','sidshinde2309@gmail.com','kunaltemkar27@gmail.com'])
    msg.body='You are recommended for %s the company.Please do well prepare and
follow thw ERP section to guide yourself.'%text
    mail.send(msg)
    return "sent"
@app.route('/back',methods=['POST'])
def back():
    return render_template('home.html')
@app.route('/logout',methods=['POST'])
def logout():
    return render_template('erplogin.html')
@app.route('/')
def my_form():
    return render_template('cal.html')
@app.route('/search')
def search():
    return render_template('search.html')
@app.route('/formupd',methods=['POST'])
def formupd():
    userid = session.get('userid', None)
    return render_template('genform.html',user=userid)
@app.route('/marks',methods=['POST'])
def marks():
    userid=session.get('userid', None)
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    a=cur.execute("select ssc,hsc,sem1,sem2,sem3,sem4,sem5,sem6,sem7,sem8,Name from
'details' where email=?",(userid,))
    l=[]
    l1=[]
    for i in a:
        l.append(i)
    for j in l[0]:
```

```
        l1.append(j)
    ssc=l1[0]
    hsc=l1[1]
    sem1=l1[2]
    sem2=l1[3]
    sem3=l1[4]
    sem4=l1[5]
    sem5=l1[6]
    sem6=l1[7]
    sem7=l1[8]
    sem8=l1[9]
    name=l1[10]
    return
render_template('marks.html',name=name,email=userid,ssc=ssc,hsc=hsc,sem1=sem1,sem2
=sem2,sem3=sem3,sem4=sem4,sem5=sem5,sem6=sem6,sem7=sem7,sem8=sem8)
@app.route('/extras',methods=['POST'])
def extras():
    return render_template('extras.html')
@app.route('/log',methods=['POST'])
def logical():
    return render_template('log.html')
@app.route('/verbal',methods=['POST'])
def verbal():
    return render_template('ver.html')
@app.route('/quantitative',methods=['POST'])
def quantitative():
    return render_template('quant.html')
@app.route('/communication',methods=['POST'])
def comm():
    return render_template('com.html')
@app.route('/com1',methods=['POST'])
def com1():
    return render_template('genform.html')
#@app.route('/back',methods=['POST'])
#def back():
#    return render_template('genform.html')

def correlationCoefficient(X, Y, n):
    sum_X = 0
    sum_Y = 0
    sum_XY = 0
    squareSum_X = 0
    squareSum_Y = 0

    i = 0
    while i < n:
        # sum of elements of array X.
        sum_X = sum_X + X[i]

        # sum of elements of array Y.
```

```
        sum_Y = sum_Y + Y[i]

        # sum of X[i] * Y[i].
        sum_XY = sum_XY + X[i] * Y[i]

        # sum of square of array elements.
        squareSum_X = squareSum_X + X[i] * X[i]
        squareSum_Y = squareSum_Y + Y[i] * Y[i]
        y1=n * squareSum_Y - sum_Y * sum_Y
        y2=n * squareSum_X - sum_X * sum_X
        i = i + 1
    print("abs",abs((n * squareSum_X - sum_X * sum_X) * (n * squareSum_Y - sum_Y *
sum_Y)))
    if abs((n * squareSum_X - sum_X * sum_X) * (n * squareSum_Y - sum_Y *
sum_Y))==0:
        corr=(float)(n * sum_XY - sum_X * sum_Y) / (float)( sqrt(0.7))
    else:
        corr = (float)(n * sum_XY - sum_X * sum_Y) / (float)( sqrt(abs((n * squareSum_X -
sum_X * sum_X) * (n * squareSum_Y - sum_Y * sum_Y))))
    return corr

@app.route('/extraupd',methods=['POST','GET'])
def extraupd():
    userid=session.get('userid',None)
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    if request.method == 'POST':
        value=request.form.get('mycheckbox')
        print(value)
    b=cur.execute("Update details set extras=? where email=?",(value,userid,))
    con.commit()
    return render_template('genform.html')
@app.route('/marksupd',methods=['POST','GET'])
def marksupd():
    userid = session.get('userid', None)
    ssc=request.form['ssc']
    hsc = request.form['hsc']
    sem1 = request.form['sem1']
    sem2 = request.form['sem2']
    sem3 = request.form['sem3']
    sem4 = request.form['sem4']
    sem5 = request.form['sem5']
    sem6 = request.form['sem6']
    sem7 = request.form['sem7']
    sem8 = request.form['sem8']
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    b=cur.execute("Update details set
```

```
ssc=?,hsc=?,sem1=?,sem2=?,sem3=?,sem4=?,sem5=?,sem6=?,sem7=?,sem8=? where
email=?",(ssc,hsc,sem1,sem2,sem3,sem4,sem5,sem6,sem7,sem8,userid,))
    con.commit()
    if b.rowcount>0:
        print("updated")
    return render_template('genform.html')
con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
cur = con.cursor()
le = LabelEncoder()
b=cur.execute("select * from 'hist'")
ratingse=b.fetchall()
ratings = pd.DataFrame(np.array(ratingse))

ssc = ratings.iloc[:, 4]
ssc1 = le.fit_transform(ssc.astype(str))
hsc = ratings.iloc[:, 5]
hsc1 = le.fit_transform(hsc.astype(str))
noc = ratings.iloc[:, 8]
noc1 = le.fit_transform(noc.astype(str))
val2 = ratings.iloc[:, 9] #extra curr
val3 = ratings.iloc[:, 10]#communication
val4 = ratings.iloc[:, 11]#logical
val5 = ratings.iloc[:, 12]#quants
val6 = ratings.iloc[:, 13]#verbal
val8 = ratings.iloc[:, 15]#certis
bl = ratings.iloc[:, 16]#backlogs
bl1 = le.fit_transform(bl.astype(str))
cgpa = ratings.iloc[:, 14]
cgpa1 = le.fit_transform(cgpa.astype(str))
val21 = le.fit_transform(val2.astype(str))
val31 = le.fit_transform(val3.astype(str))
val41 = le.fit_transform(val4.astype(str))
val51 = le.fit_transform(val5.astype(str))
val61 = le.fit_transform(val6.astype(str))
val81 = le.fit_transform(val8.astype(str))
features = list(zip(ssc1, hsc1, cgpa1, noc1, val21, val31, val41, val51, val61, val81, bl1))
con.commit()
@app.route('/predict4',methods=['POST'])
def predict4():
    userid = session.get('userid', None)
    l=[]
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    le = LabelEncoder()
    b1=cur.execute("select companies from 'hist'").fetchall()
    a = cur.execute("select * from 'details'")
    realse = a.fetchall()
    for i in realse:
```

```
        if i[1]==userid:
            keys=realse.index(i)
        else:
            pass
    reals = pd.DataFrame(np.array(realse))
    id = reals.iloc[:, 17]
    id1 = le.fit_transform(id.astype(str))
    ssc2 = reals.iloc[:, 9]
    ssc3 = le.fit_transform(ssc2.astype(str))
    hsc2 = reals.iloc[:, 8]
    hsc3 = le.fit_transform(hsc2.astype(str))
    cgpa2 = reals.iloc[:, 4]
    cgpa3 = le.fit_transform(cgpa2.astype(str))
    noc2 = reals.iloc[:, 10]
    noc3 = le.fit_transform(noc2.astype(str))
    verbal = reals.iloc[:, 11]
    logical = reals.iloc[:, 14]
    quant = reals.iloc[:, 13]
    comm = reals.iloc[:, 12]
    extra = reals.iloc[:, 16]
    dr = reals.iloc[:, 7]
    cert = reals.iloc[:, 15]
    fat = pd.DataFrame(np.array([ssc2, hsc2, cgpa2, noc2, extra, comm, quant, verbal, cert,
dr]))
    dr1 = le.fit_transform(dr.astype(str))
    val121 = le.fit_transform(verbal.astype(str))
    val221 = le.fit_transform(logical.astype(str))
    val321 = le.fit_transform(quant.astype(str))
    val421 = le.fit_transform(comm.astype(str))
    val521 = le.fit_transform(extra.astype(str))
    val621 = le.fit_transform(cert.astype(str))
    features1 = list(zip(ssc3, hsc3, cgpa3, noc3, val521, val421, val221, val321, val121,
val621, dr1))
    con.commit()
    global msg1
    if(msg1=='Y'):
        for i, k in enumerate(features):
            x = correlationCoefficient(k, features1[keys], len(k))
            print(x)
            if (x > 0.7 and k not in l):
                l.append(i)
        l1 = []
        for key in l:
            if b1[key] not in l1:
                l1.append(b1[key])
            print(l1)
            if ('FALSE') in l1:
                l1.remove(('FALSE'))
            df1 = DataFrame(l1, columns=['companies'])
            total = len(l1)
```

```
        return render_template("index1.html",total=total,
column_names=df1.columns.values,link_column="student_id",
row_data=list(df1.values.tolist()), zip=zip)
    else:
        return render_template("index1.html",msg='no comp')
text=""
@app.route('/company', methods=['POST'])
def my_form_post():
    global text
    text = request.form['text']
    le = LabelEncoder()
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    b = cur.execute("SELECT * from hist where companies=?", (text,))
    ratingse = b.fetchall()
    #print(ratingse)
    if len(ratingse)>0:
        ratings = pd.DataFrame(np.array(ratingse))
        #print(ratings)
        print("This is exec")
    else:
        b = cur.execute("SELECT * from hist where companies = 'LTI'")
        ratingse1 =b.fetchall()
        ratings=pd.DataFrame(np.array(ratingse1))
        #print(ratings)
        print("else is exec")
    b1=cur.execute("select id,Name from details").fetchall()
    a = cur.execute("select * from 'details'")
    realse = a.fetchall()
    reals = pd.DataFrame(np.array(realse))
    id = reals.iloc[:, 17]
    id1 = le.fit_transform(id.astype(str))
    ssc2 = reals.iloc[:, 9]
    ssc3 = le.fit_transform(ssc2.astype(str))
    hsc2 = reals.iloc[:, 8]
    hsc3 = le.fit_transform(hsc2.astype(str))
    cgpa2 = reals.iloc[:, 4]
    cgpa3 = le.fit_transform(cgpa2.astype(str))
    noc2 = reals.iloc[:, 10]
    noc3 = le.fit_transform(noc2.astype(str))
    verbal = reals.iloc[:, 11]
    logical= reals.iloc[:, 14]
    quant = reals.iloc[:, 13]
    comm = reals.iloc[:, 12]
    extra = reals.iloc[:, 16]
    dr = reals.iloc[:, 7]
    cert = reals.iloc[:, 15]
    fat = pd.DataFrame(np.array([ssc2, hsc2, cgpa2, noc2, extra, comm, quant, verbal, cert,
dr]))
```

```
    dr1 = le.fit_transform(dr.astype(str))
    val121 = le.fit_transform(verbal.astype(str))
    val221 = le.fit_transform(logical.astype(str))
    val321 = le.fit_transform(quant.astype(str))
    val421 = le.fit_transform(comm.astype(str))
    val521 = le.fit_transform(extra.astype(str))
    val621 = le.fit_transform(cert.astype(str))
    features1 = list(zip(ssc3, hsc3, cgpa3, noc3, val521, val421, val221, val321, val121,
val621, dr1))
    #print("*****Features1*********")
    #print("multiple tuple",features1)
    ssc = ratings.iloc[:, 4]
    #print(ssc)
    ssc1 = le.fit_transform(ssc.astype(float))
    hsc = ratings.iloc[:, 5]
    hsc1 = le.fit_transform(hsc.astype(float))
    noc = ratings.iloc[:, 8]
    noc1 = le.fit_transform(noc.astype(int))
    val2 = ratings.iloc[:, 9]
    val3 = ratings.iloc[:, 10]
    val4 = ratings.iloc[:, 11]
    val5 = ratings.iloc[:, 12]
    val6 = ratings.iloc[:, 13]
    val8 = ratings.iloc[:, 15]
    bl = ratings.iloc[:, 16]
    bl1 = le.fit_transform(bl.astype(str))
    cgpa = ratings.iloc[:, 14]
    cgpa1 = le.fit_transform(cgpa.astype(float))
    val21 = le.fit_transform(val2.astype(str))
    val31 = le.fit_transform(val3.astype(str))
    val41 = le.fit_transform(val4.astype(str))
    val51 = le.fit_transform(val5.astype(str))
    val61 = le.fit_transform(val6.astype(str))
    val81 = le.fit_transform(val8.astype(str))
    features = list(zip(ssc1, hsc1, cgpa1, noc1, val21, val31, val41, val51, val61, val81, bl1))
    #print("********Features 2**********************")
    #print("single tuple",features)
    l = []
    print("features[1]",features[1])
    for i, k in enumerate(features1):
        x = correlationCoefficient(k, features[1], len(k))
        if (x > 0.7 and k not in l):
            l.append(i)
    l1 = []
    for key in l:
        l1.append(b1[key])
        df1 = DataFrame(l1,columns=['student_id', 'Name'])
        total=len(l1)
    con.commit()
    return
```

```
render_template("index.html",name1=text,total=total,column_names=df1.columns.values,li
nk_column="student_id", row_data=list(df1.values.tolist()), zip=zip)
   #subset = df1[['student_id', 'ssc', 'hsc', 'cgpa', 'noc', 'verbal', 'logical', 'quant', 'comm',
'extra','drop', 'certis', 'branch']]
   #tuples = (tuple(x) for x in subset.values)
   #return df1.content
@app.route('/login', methods=['GET','POST'])
def erplogin():
   l=[]
   l1=[]
   if request.method=="POST":
      userid=request.form["txtemail"]
      session['userid'] = userid
      pswd=request.form["txtupass"]
      with sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db") as con:
         cur = con.cursor()
      c = cur.execute("SELECT email from details where email= (?)", [userid])
      userexists = c.fetchone()
      if userexists:
         c = cur.execute("SELECT id from details where id = (?)", [pswd])
         passwcorrect = c.fetchone()
         if passwcorrect:
            c = cur.execute("SELECT Name,id,log_per,ver_per,quant_per from details
where email = (?)", [userid])
            for rows in c:
               l.append(rows)
            for x in l[0]:
               l1.append(x)
            name=l1[0]
            uid=l1[1]
            lper=l1[2]
            per=l1[3]
            qper=l1[4]
            return
render_template('home.html',vper=per,lper=lper,qper=qper,name=name,id=uid,email=useri
d)

   return render_template('erplogin.html')
@app.route('/predict1',methods=['POST'])
def predict1():
   userid = session.get('userid', None)
   algo=request.form.get('choose')
   print("algotihm name",algo)
   con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
   cur = con.cursor()
   a = cur.execute("select * from 'hist'")
   realse = a.fetchall()
   data = pd.DataFrame(np.array(realse))
```

```
    dict1 = {'Yes': 1, 'No': 0, 'others': 0, 'cultural': 1, 'sports': 2, 'Excellent': 1, 'Good': 2,
'Average': 0,
             'Poor': 3}
    target = data.iloc[:, 18]
    va = data.iloc[:, 13]
    qu = data.iloc[:, 12]
    lr = data.iloc[:, 11]
    co = data.iloc[:, 10]
    ec = data.iloc[:, 9]
    dr = data.iloc[:, 19]
    le = LabelEncoder()
    target1 = le.fit_transform(target.astype(str))
    va1 = le.fit_transform(va.astype(str))
    qu1 = le.fit_transform(qu.astype(str))
    lr1 = le.fit_transform(lr.astype(str))
    co1 = le.fit_transform(co.astype(str))
    ec1 = le.fit_transform(ec.astype(str))
    dr1 = le.fit_transform(dr.astype(str))
    features = list(zip(va1, qu1, lr1, co1, ec1, dr1))
    features_train, features_test, target_train, target_test = train_test_split(features, target1,
test_size=0.33,
                                                                random_state=10)
    gb = GaussianNB()
    dt = DecisionTreeClassifier()
    dt.fit(features_train, target_train)
    gb.fit(features_train, target_train)
    loaded_model = joblib.load('model.pkl')
    dt_model=joblib.load('dtmodel.pkl')
    svm_model=joblib.load('svmmodel.pkl')
    knn_model=joblib.load('knnmodel.pkl')
    lr_model=joblib.load('lrmodel.pkl')
    rf_model=joblib.load('rfmodel.pkl')
    target_pred1 = dt.predict(features_test)
    target_pred = gb.predict(features_test)
    acc_naive = metrics.accuracy_score(target_test, target_pred1, normalize="True")
    acc_decision = metrics.accuracy_score(target_test, target_pred, normalize="True")
    print(target_pred)
    print(target_test)
    print(confusion_matrix(target_test, target_pred))
    print(acc_naive)
    print(acc_decision)
    n = cur.execute("SELECT * FROM details WHERE email=? ", (userid,))
    l=[]
    l1=[]
    for row in n:
        l.append(row)
        # print(l)
    for x in l[0]:
        l1.append(x)
    name=l1[0]
```

```
emailid=l1[1]
course=l1[2]
dept=l1[3]
cgpa=l1[4]
percent=l1[5]
passyr=l1[6]
backlog=l1[7]
hsc=l1[8]
ssc=l1[9]
noc=l1[10]
verbal=l1[11]
verbal1=dict1[verbal]
comm=l1[12]
comm1=dict1[comm]
quant=l1[13]
quant1=dict1[quant]
logical=l1[14]
print("logical ky",logical)
logical1=dict1[logical]
cert=l1[15]
extras=l1[16]
sem1=l1[20]
sem2=l1[21]
sem3=l1[22]
sem4=l1[23]
sem5=l1[24]
sem6=l1[25]
sem7=l1[26]
sem8=l1[27]
print("extra key",extras)
extras1=dict1[extras]
id=l1[17]
dob=l1[18]
drop=l1[19]
lquiz=l1[34]
vquiz=l1[35]
qquiz=l1[36]
drop1=dict1[drop]
if drop1==0:
    drop1=1
else:
    drop1=0
predicted = loaded_model.predict([[verbal1, quant1, logical1, comm1, extras1, drop1]])
a = loaded_model.predict_proba([[verbal1, quant1, logical1, comm1, extras1, drop1]])
if algo=='nb':
    algo="Naive Bayes"
    predicted = loaded_model.predict([[verbal1, quant1, logical1, comm1, extras1, drop1]])
    a = loaded_model.predict_proba([[verbal1, quant1, logical1, comm1, extras1, drop1]])
elif algo=='dt':
    algo="Decision Tree"
```

```
    predicted = dt_model.predict([[verbal1, quant1, logical1, comm1, extras1, drop1]])
    a = dt_model.predict_proba([[verbal1, quant1, logical1, comm1, extras1, drop1]])
elif algo=='svm':
    algo="Support Vector Machine"
    predicted=svm_model.predict([[verbal1, quant1, logical1, comm1, extras1, drop1]])
    a = svm_model.predict_proba([[verbal1, quant1, logical1, comm1, extras1, drop1]])
elif algo=='rf':
    algo="Random Forest Tree"
    predicted = rf_model.predict([[verbal1, quant1, logical1, comm1, extras1, drop1]])
    a = rf_model.predict_proba([[verbal1, quant1, logical1, comm1, extras1, drop1]])
elif algo=='knn':
    algo="K-Nearest Neighbour"
    predicted= knn_model.predict([[verbal1, quant1, logical1, comm1, extras1, drop1]])
    a = knn_model.predict_proba([[verbal1, quant1, logical1, comm1, extras1, drop1]])
elif algo=='lr':
    algo="Logistic Regression"
    predicted = lr_model.predict([[verbal1, quant1, logical1, comm1, extras1, drop1]])
    a = lr_model.predict_proba([[verbal1, quant1, logical1, comm1, extras1, drop1]])
#predicted = loaded_model.predict([[verbal1, quant1, logical1, comm1, extras1, drop1]])
print(predicted)
print("probability")
#a=loaded_model.predict_proba([[verbal1, quant1, logical1, comm1, extras1, drop1]])
print(a)
no=a[:,0]
yes=a[:,1]
print(no,yes)
for i in no:
    c=int(i*100)
for j in yes:
    d=int(j*100)
print(predicted)
global msg1
if lquiz and qquiz and vquiz:
    if int(predicted)==1:
        msg=' you can be placed'
        msg1='Y'
        d1=c
        d2=100-d1
    else:
        msg= 'sorry you cant  place! you need to work hard!!'
        d1=c
        d2=100-d1
        msg1='N'
elif sem1==None or sem2==None or sem3==None or sem4==None:
    msg="Need to complete courses upto sem4"
    d1=0
    d2=0
else:
    msg='Quiz for Logical/Quantitaive/Verbal not done'
    d1=0
```

```
        d2=0
    return render_template('result.html',msg=msg,no=d1,yes=d2,algo=algo)

@app.route('/predict3',methods=['GET','POST'])
def predict3():
    userid = session.get('userid', None)
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    a = cur.execute("select * from 'details' where email=?",(userid,))
    realse = a.fetchall()
    data = pd.DataFrame(np.array(realse))
    percentd = data.iloc[:, 5]
    backlogd = data.iloc[:, 7]
    verbald = data.iloc[:, 11]
    commd = data.iloc[:, 12]
    quantdd = data.iloc[:, 13]
    logical = data.iloc[:, 14]
    verbald=verbald.values[0]
    commd=commd.values[0]
    quantdd=quantdd.values[0]
    logical=logical.values[0]
    backlogd=backlogd.values[0]
    percentd=float(percentd.values[0])
    if verbald=='Average' or verbald=='Poor':
        ver="Verbal needs to improve"
    else :
        ver="verbal ok"
    if commd=='Average' or commd=='Poor':
        com="Communication needs to improve"
    else:
        com="communication ok"
    if quantdd=='Average' or quantdd=='Poor':
        qua="Quants needs to be improve"
    else :
        qua="quantitative ok"
    if logical=='Average' or logical=='Poor':
        log="Logical skills need to improve"
    else:
        log="Logical ok"
    if backlogd=='nil' or backlogd=='Nil' or backlogd=='Nill':
        bac="Dont allow the backlog in future"
    else:
        bac="Remove the backlog"
    if percentd>60:
        per="Cgpa is ok,maintain the cgpa as it is"
    else:
        per="Focus on cgpa"
    return
render_template('requirement.html',ver=ver,com=com,qua=qua,log=log,bac=bac,per=per)
```

```
@app.route('/insert', methods=['GET','POST'])
def insert():
    msg="msg"
    if request.method == "POST":
        try:
            userid = request.form["userid"]
            pswd = request.form["pass"]
            with sqlite3.connect("C:/Users/Chinmay Chaughule/data.db") as con:
                cur = con.cursor()
                cur.execute("INSERT into details (userid,pswd) values (?,?)", (userid, pswd))
                con.commit()
                msg = "Employee successfully Added"
                con.closed()
        except:
            con.rollback()
            msg = "We can not add the employee to the list"
    return render_template('valid.html', msg=msg)

@app.route('/profile', methods=['GET','POST'])
def profile():
    userid = session.get('userid', None)
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    a = cur.execute("select * from 'details' where email=?", (userid,))
    realse = a.fetchall()
    data = pd.DataFrame(np.array(realse))
    return render_template('profile.html',data=realse)
original_questions = {
'Synonyms for CORPULENT':['Obese','Lean','Gaunt','Emaciated'],
'Synonyms for BRIEF':['Short','Limited','Small','Little'],
'Synonym for EMBEZZLE':['Misappropriate','Balance','Remunerate','Clear'],
'Synonym for VENT':['Opening','Stodge','End','Past tense of go'],
'Synonym for AUGUST':['Dignified','Common','Ridiculous','Petty'],
'Synonym for CANNY':['Clever','Obstinate','Handsome','Stout'],
'Synonym for ALERT':['Watchful','Energetic','Observant','Intelligent'],
'Fate smiles....those who untiringly grapple with stark realities f
life':['on','with','over','round'],
 'The miser gazed ..... at the pile of gold coins in front of
him.':['avidly','admiringly','thoughtfully','earnestly'],
'Catching the earlier train will give us the ..... to do some
shopping.':['chance','luck','possibility','occasion'],
'Success in this examination depends ..... hard work alone.':['on','at','over','for'],
'If you smuggle goods into the country, they may be ..... by the customs
authority.':['confiscated','possessed','punished','fined'],
 'Piyush behaves strangely at times and, therefore, nobody gets ..... with
him':['along','about','through','up'],
'The ruling party will have to put its own house ..... order.':['in','on','to','into'],
'The man to who I sold my house was a cheat.':['to whom I sold','to whom I sell','to who
```

```
I sell','who was I sold'],
'They were all shocked at his failure in the competition.':['No correction required','were shocked at all','had all shocked at','had all shocked by'],
'He is too important for tolerating any delay.':['to tolerate','to tolerating','at tolerating','with tolerating'],
"To keeps one's temper":['To be in good mood','To become hungry','To preserve ones energy','None of these'],
'To have an axe to grind':['A private end to serve','To fail to arouse interest','To have no result','To work for both sides'],
'To drive home':['To emphasise',"To find one's roots",'To return to place of rest','Back to original position']
}
questions = copy.deepcopy(original_questions)
#print(questions)
def shuffle(q):
    selected_keys = []
    i = 0
    while i < len(q):
        current_selection = random.choice(list(q.keys()))
        #print(current_selection)
        if current_selection not in selected_keys:
            selected_keys.append(current_selection)
            i = i+1
    return selected_keys
@app.route('/verbal1',methods=['POST'])
def quiz():
    questions_shuffled = shuffle(questions)
    print(questions_shuffled)
    for i in questions.keys():
        random.shuffle(questions[i])
    return render_template('vscreen.html', q=questions_shuffled, o=questions)
per=0
@app.route('/quiz', methods=['POST'])
def quiz_answers():
    correct = 0
    for i in questions.keys():
        answered = request.form.get(i)
        print(answered)
        if original_questions[i][0] == answered:
            correct = correct+1
    userid = session.get('userid', None)
    print(correct)
    con = sqlite3.connect("C:/Users/Chinmay Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    print("username is ", userid)
    cur.execute("Update details set vquiz=? where email=?", (str(correct), userid,))
    print("updated")
    con.commit()
    con = sqlite3.connect("C:/Users/Chinmay
```

```
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    b = cur.execute("select vquiz,file1,file2,file3 from details where email=?", (userid,))
    realse = b.fetchall()
    data = pd.DataFrame(np.array(realse))
    mrk = int(data.iloc[:, 0])
    global per
    per= (mrk / 15) * 100
    cur.execute("Update details set ver_per=? where email=?",(per,userid,))
    print(per)
    if per > 75:
        cur.execute("Update details set verbal=? where email=?", ("Excellent", userid,))
    elif per > 60:
        cur.execute("Update details set verbal=? where email=?", ("Good", userid,))
    elif per > 45:
        cur.execute("Update details set verbal=? where email=?", ("Average", userid,))
    else:
        cur.execute("Update details set verbal=? where email=?", ("Poor", userid,))
    con.commit()

    return render_template('genform.html',user=userid)#'<h1>Correct Answers:
<u>'+str(correct)+'</u></h1>'
original_questions1 = {
'What was the day on 15th august 1947 ?' :
['Friday', 'Saturday' ,'Sunday' ,'Thursday'],


'Today is Monday. After 61 days, it will be' :
['Saturday',  'Tuesday', 'Monday', 'Sunday'],


'In an election between two candidates, one got 55% of the total valid votes, 20% of the
votes were invalid. If the total number of votes was 7500, the number of valid votes that
the other candidate got, was' :
['2700', '2900', '2500', '3100'],


'A bag contains 50 P, 25 P and 10 P coins in the ratio 5: 9: 4, amounting to Rs. 206. Find
the number of coins of each type respectively.' :
['200,360,160', '360,160,200', '160,360,200', '200,160,300'],


'If each side of a square is increased by 25%, find the percentage change in its area?' :
['56.25', '65', '65.34', '56', '58'],


'A problem is given to three students whose chances of solving it are 1/2, 1/3 and 1/4
respectively. What is the probability that the problem will be solved?' :
['3/4', '1/2', '1/4', '7/12'],
```

"A and B invest in a business in the ratio 3 : 2. If 5% of the total profit goes to charity and A's share is Rs. 855, the total profit is" :
['1500', '1000', '2000', '500'],

'A bag contains 6 white and 4 black balls .2 balls are drawn at random. Find the probability that they are of same colour' :
['7/15', '8/15', '1/9', '1/2'],

'If 20% of a = b, then b% of 20 is the same as' :
[' 4% of a', '6% of a', '8% of a', '10% of a'],

'A student multiplied a number by 3/5 instead of 5/3, What is the percentage error in the calculation ?' :
['64%', '54%', '74%', '84%'],

'A trader mixes 26 kg of rice at Rs. 20 per kg with 30 kg of rice of other variety at Rs. 36 per kg and sells the mixture at Rs. 30 per kg. His profit' :
['5%', '8%', '7%', '6%'],

'Fresh fruit contains 68% water and dry fruit contains 20% water. How much dry fruit can be obtained from 100 kg of fresh fruits ?':
['40', '50', '10', '67'],

'Fresh fruit contains 68% water and dry fruit contains 20% water. How much dry fruit can be obtained from 100 kg of fresh fruits ?' :
['125%', '150%', '110%', '160%'],

'A clock is set right at 8 a.m. The clock gains 10 minutes in 24 hours will be the true time when the clock indicates 1 p.m. on the following day?' :
[' 48 min. past 12.', '45 min past 12', '46 min past 12', '41 min past 12'],

'The last day of a century cannot be':
['Tuesday', 'monday', 'wednesday', 'friday'],

'The value of a machine depreciates at the rate of 10% every year. It was purchased 3 years ago. If its present value is Rs. 8748, its purchase price was ':
['12000', '14000', '17000', '34000'],

'Two numbers are respectively 20% and 50% more than a third number. The ratio of the two numbers is':
['4:5', '3:5', '2:5', '5:4'],

'Insert the missing number :7, 26, 63, 124, 215, 342, (....)' :
['511', '344', '232', '543'],

'Out of 7 consonants and 4 vowels, how many words of 3 consonants and 2 vowels can be formed?' :
['25200', '52000', '120', '24400'],

'It was Sunday on Jan 1, 2006. What was the day of the week Jan 1, 2010?' :
['friday', 'sunday', 'monday', 'wednesday'],

'If selling price is doubled, the profit triples. Find the profit percent ?' :
['100', '120', '200', '650'],

'Two cards are drawn at random from a pack of 52 cards.what is the probability that either both are black or both are queen?' :
['55/221', '55/190', '52/221', '19/221'],

'What was the day of the week on, 16th July, 1776?' :
['tuesday', 'monday', 'wednesday', 'friday'],

'In an examination, a student scores 4 marks for every correct answer and loses 1 mark for every wrong answer. If he attempts all 60 questions and secures 130 marks, the no of questions he attempts correctly is' :
['38', '40', '31', '32'],

'The average of runs of a cricket player of 10 innings was 32. How many runs must he make in his next innings so as to increase his average of runs by 4 ?' :
['76', '79', '74', '87'],

'A grocer has a sale of Rs 6435, Rs. 6927, Rs. 6855, Rs. 7230 and Rs. 6562 for 5 consecutive months. How much sale must he have in the sixth month so that he gets an average sale of Rs, 6500 ?' :
['4991', '5467', '6453', '5987'],

'Three number are in the ratio of 3 : 4 : 5 and their L.C.M. is 2400. Their H.C.F. is':

```
['40', '80', '32', '232'],


'A student has to obtain 33% of the total marks to pass. He got 125 marks and failed by
40 marks. The maximum marks are ':
['500', '600', '769', '200'],


"A man spends 35% of his income on food, 25% on children's education and 80% of the
remaining on house rent. What percent of his income he is left with ?":
['6', '8', '5', '2'],


'if the price of a book is first decreased by 25% and then increased by 20%, then the net
change in the price will be ' :
['10', '30', '40', '20']
}


questions1 = copy.deepcopy(original_questions1)
qper=0
@app.route('/quantitative1',methods=['POST'])
def quiz1():
    questions_shuffled1 = shuffle(questions1)
    print(questions_shuffled1)
    for i in questions1.keys():
        random.shuffle(questions1[i])
    return render_template('qscreen.html', q=questions_shuffled1, o=questions1)
@app.route('/quiz1', methods=['POST'])
def quiz_answers1():
    correct = 0
    for i in questions1.keys():
        answered = request.form.get(i)
        print(answered)
        if original_questions1[i][0] == answered:
            correct = correct+1
    userid = session.get('userid', None)
    print(correct)
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    print("username is ", userid)
    cur.execute("Update details set qquiz=? where email=?", (str(correct), userid,))
    print("updated")
    con.commit()
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    b = cur.execute("select qquiz,file1,file2,file3 from details where email=?", (userid,))
    realse = b.fetchall()
```

```
    data = pd.DataFrame(np.array(realse))
    mrk = int(data.iloc[:, 0])
    global qper
    qper = (mrk / 15) * 100
    print(qper)
    cur.execute("Update details set quant_per=? where email=?",(qper,userid,))
    if qper > 75:
        cur.execute("Update details set quant=? where email=?", ("Excellent", userid,))
    elif qper > 60:
        cur.execute("Update details set quant=? where email=?", ("Good", userid,))
    elif qper > 45:
        cur.execute("Update details set quant=? where email=?", ("Average", userid,))
    else:
        cur.execute("Update details set quant=? where email=?", ("Poor", userid,))
    con.commit()

    return render_template('genform.html',user=userid)
original_questions2 = {
'Synonyms for CORPULENT':['Obese','Lean','Gaunt','Emaciated'],
'Synonyms for BRIEF':['Short','Limited','Small','Little'],
'Synonym for EMBEZZLE':['Misappropriate','Balance','Remunerate','Clear'],
'Synonym for VENT':['Opening','Stodge','End','Past tense of go'],
'Synonym for AUGUST':['Dignified','Common','Ridiculous','Petty'],
'Synonym for CANNY':['Clever','Obstinate','Handsome','Stout'],
'Synonym for ALERT':['Watchful','Energetic','Observant','Intelligent'],
'Fate smiles....those who untiringly grapple with stark realities f
life':['on','with','over','round'],
 'The miser gazed ..... at the pile of gold coins in front of
him.':['avidly','admiringly','thoughtfully','earnestly'],
'Catching the earlier train will give us the ..... to do some
shopping.':['chance','luck','possibility','occasion'],
'Success in this examination depends ..... hard work alone.':['on','at','over','for'],
'Statement: Anger is energy, in a more proactive way and how to channelize it is in itself
a skill. Assumptions: I. Anger need to be channelized. II. Only skillful people can
channelize anger to energy.':[' If only assumption II is implicit.','If only assumption I is
implicit.','if either I or II is implicit.','if neither I or II is implicit.']

}


questions2 = copy.deepcopy(original_questions2)
lper=0
@app.route('/logical',methods=['POST','GET'])
def quiz3():
    userid = session.get('userid', None)
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    if request.method == 'POST':
        first=request.form.get("f",None)
```

```
        second=request.form.get("s",None)
        third=request.form.get("t",None)
        file1=request.form.get('myFile1')
        file2= request.form.get('myFile2')
        file3 = request.form.get('myFile3')
        print(file1,file2,file3)
        print(first,second,third)
    b = cur.execute( "Update details set
logicalfirst=?,logicalsecond=?,logicalthird=?,file1=?,file2=?,file3=? where
email=?",(first,second,third,file1,file2,file3, userid,))
    con.commit()
    #if(first!="none" && file1!=" " && second!="none")
    return render_template('genform.html')
@app.route('/logical1',methods=['POST'])
def quiz2():
    questions_shuffled2 = shuffle(questions2)
    print(questions_shuffled2)
    for i in questions2.keys():
        random.shuffle(questions2[i])
    return render_template('lscreen.html', q=questions_shuffled2, o=questions2)
@app.route('/quiz2', methods=['POST'])
def quiz_answers2():
    correct = 0
    for i in questions2.keys():
        answered = request.form.get(i)
        #print(answered)
        if original_questions2[i][0] == answered:
            correct = correct+1
    userid = session.get('userid', None)
    print(correct)
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    print("username is ",userid)
    cur.execute("Update details set lquiz=? where email=?",(str(correct),userid,))
    print("updated")
    con.commit()
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    b=cur.execute("select lquiz,file1,file2,file3 from details where email=?",(userid,))
    realse = b.fetchall()
    data = pd.DataFrame(np.array(realse))
    mrk=int(data.iloc[:,0])
    global lper
    lper=(mrk/15)*100
    cur.execute("Update details set log_per=? where email=?",(lper,userid,))
    print(lper)
    if lper>75:
        cur.execute("Update details set logical=? where email=?",("Excellent",userid,))
```

```
    elif lper>60:
        cur.execute("Update details set logical=? where email=?", ("Good", userid,))
    elif lper>45:
        cur.execute("Update details set logical=? where email=?", ("Average", userid,))
    else:
        cur.execute("Update details set logical=? where email=?", ("Poor", userid,))
    con.commit()
    return render_template('genform.html',user=userid)#'<h1>Correct Answers:
<u>'+str(correct)+'</u></h1>'
@app.route('/data_visualize', methods=['POST'])
def dv():
    userid = session.get('userid', None)
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    a = cur.execute("select * from 'hist'")
    realse = a.fetchall()
    placement = pd.read_csv('hist.csv')
    b=cur.execute("select * from 'details'").fetchall()
    mydata = pd.DataFrame(np.array(b))
    branch=mydata.iloc[:,3]
    cgpa=mydata.iloc[:,4]
    yearofpass=mydata.iloc[:,6]
    hsc=mydata.iloc[:,8]
    ssc=mydata.iloc[:,9]
    noc=mydata.iloc[:,10]
    verbal=mydata.iloc[:,11]
    comm=mydata.iloc[:,12]
    quantitative=mydata.iloc[:,13]
    logical=mydata.iloc[:,14]
    certis=mydata.iloc[:,15]
    extracurr=mydata.iloc[:,16]
    drops=mydata.iloc[:,19]
    #cur.execute("Insert into
hist(branch,cgpa,yearofpass,noc,ssc,hsc,verbal,comm,quantitative,logical,certis,extracurr,dr
ops)
values(?,?,?,?,?,?,?,?,?,?,?,?,?)",(branch,cgpa,yearofpass,noc,hsc,ssc,verbal,comm,quantitat
ive,logical,certis,extracurr,drops,))
    con.commit()
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    a = cur.execute("select * from 'hist'")
    realse = a.fetchall()
    placement = pd.read_csv('hist.csv')
    print(placement.head())
    data = pd.DataFrame(np.array(realse))
    print(data)
    name = ['hsc', 'ssc', 'verbal', 'comm', 'quantitative', 'logical', 'cert', 'extracurr',
'companies', 'recruited']
```

```
    # data1=pd.read_csv("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/hist.csv",name=name)
    dict1 = {'Yes': 1, 'No': 0, 'Others': 0, 'Cultural': 1, 'Sports': 2, 'Excellent': 1, 'Good': 2,
'Average': 0,
             'Poor': 3}
    target = data.iloc[:, 18]
    va = data.iloc[:, 13]
    qu = data.iloc[:, 12]
    lr = data.iloc[:, 11]
    co = data.iloc[:, 10]
    ec = data.iloc[:, 9]
    dr = data.iloc[:, 19]
    le = LabelEncoder()
    target1 = le.fit_transform(target.astype(str))
    va1 = le.fit_transform(va.astype(str))
    qu1 = le.fit_transform(qu.astype(str))
    lr1 = le.fit_transform(lr.astype(str))
    co1 = le.fit_transform(co.astype(str))
    ec1 = le.fit_transform(ec.astype(str))
    dr1 = le.fit_transform(dr.astype(str))
    features = list(zip(va1, qu1, lr1, co1, ec1, dr1))
    features_train, features_test, target_train, target_test = train_test_split(features, target1,
test_size=0.33,
                                                                random_state=10)
    print("features train", features_train)
    print("features test", features_test)
    print("target train", target_train)
    gb = GaussianNB()
    dt = DecisionTreeClassifier()
    lr = LogisticRegression()
    knn = KNeighborsClassifier()
    svm = SVC()
    svm.fit(features_train, target_train)
    knn.fit(features_train, target_train)
    lr.fit(features_train, target_train)
    dt.fit(features_train, target_train)
    gb.fit(features_train, target_train)
    # joblib.dump(gb, 'model.pkl')
    target_pred4 = svm.predict(features_test)
    target_pred3 = knn.predict(features_test)
    target_pred2 = lr.predict(features_test)
    target_pred1 = dt.predict(features_test)
    target_pred = gb.predict(features_test)
    # predict_proba(features_test)
    # plt.scatter(target_train,features_train)
    clf = RandomForestClassifier(n_estimators=100)

    # Train the model using the training sets y_pred=clf.predict(X_test)
    clf.fit(features_train, target_train)
```

```
    y_pred = clf.predict(features_test)
    acc_rf = metrics.accuracy_score(target_test, y_pred, normalize=True)
    acc_svm = metrics.accuracy_score(target_test, target_pred4, normalize="True")
    acc_knn = metrics.accuracy_score(target_test, target_pred3, normalize="True")
    acc_naive = metrics.accuracy_score(target_test, target_pred1, normalize="True")
    acc_decision = metrics.accuracy_score(target_test, target_pred, normalize="True")
    acc_logistic = metrics.accuracy_score(target_test, target_pred2, normalize="True")
    rs_rft=metrics.recall_score(target_test,y_pred)
    ps_rft=metrics.precision_score(target_test,y_pred)
    f1_rft=metrics.f1_score(target_test,y_pred)
    rs_naive=metrics.recall_score(target_test,target_pred1)
    ps_naive=metrics.precision_score(target_test,target_pred1)
    f1_naive=metrics.f1_score(target_test,target_pred1)
    rs_dt=metrics.recall_score(target_test, target_pred)
    ps_dt=metrics.precision_score(target_test, target_pred)
    f1_dt=metrics.f1_score(target_test, target_pred)
    rs_svm=metrics.recall_score(target_test, target_pred4)
    ps_svm=metrics.precision_score(target_test, target_pred4)
    f1_svm=metrics.f1_score(target_test, target_pred4)
    rs_knn=metrics.recall_score(target_test, target_pred3)
    ps_knn=metrics.precision_score(target_test, target_pred3)
    f1_knn=metrics.f1_score(target_test, target_pred3)
    rs_l=metrics.recall_score(target_test, target_pred2)
    ps_l=metrics.precision_score(target_test, target_pred2)
    f1_l=metrics.f1_score(target_test, target_pred2)
    con_dt=confusion_matrix(target_test, target_pred)
    class_dt=classification_report(target_test,target_pred)
    print("confus",con_dt,"type",type(con_dt))
    print(class_dt,type(class_dt))
    print("Second member",class_dt[2])
    X1 = ("SVM", "DT", "NB", "KNN", "RFT", "LR")
    Y1 = (acc_svm, acc_decision, acc_naive, acc_knn, acc_rf, acc_logistic)

    # plt.show()
    plt.bar(X1, Y1, align='center', width=0.3, color=['green', 'blue', 'red', 'yellow', 'cyan',
'black'])
    plt.xlabel('Algorithm')
    plt.ylabel('Accuracy')
    plt.gca().yaxis.set_major_formatter(PercentFormatter(1))
    plt.ylim((0, 1))
    plt.title('Algorithm comparison')
    plt.savefig("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/static/images/algo.png")
    plt.close()
    res = placement.groupby("recruited").backlogs.value_counts(normalize=True)
    res.unstack().plot(kind='bar')
    plt.savefig("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/static/images/backlogs.png")
    plt.close()
    res = placement.groupby("recruited").comm.value_counts(normalize=True)
```

```
    res.unstack().plot(kind='bar')
    plt.savefig("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/static/images/com.png")
    plt.close()
    res = placement.groupby("recruited").quantitative.value_counts(normalize=True)
    res.unstack().plot(kind='bar')
    plt.savefig("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/static/images/quants.png")
    plt.close()
    res = placement.groupby("recruited").logical.value_counts(normalize=True)
    res.unstack().plot(kind='bar')
    plt.savefig("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/static/images/log.png")
    plt.close()
    res = placement.groupby("recruited").verbal.value_counts(normalize=True)
    res.unstack().plot(kind='bar')
    plt.savefig("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/static/images/ver.png")
    plt.close()
    res = placement.groupby("recruited").extracurr.value_counts(normalize=True)
    res.unstack().plot(kind='bar')
    plt.savefig("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/static/images/ext.png")
    plt.close()
    res = placement.groupby("recruited").internship.value_counts(normalize=True)
    res.unstack().plot(kind='bar')
    plt.savefig("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/static/images/int.png")
    plt.close()
    return
render_template('dv.html',data=placement,cmdt=con_dt,class_dt=class_dt,rf=acc_rf,n=acc_
naive,d=acc_decision,l=acc_logistic,k=acc_knn,s=acc_svm,rfr=rs_rft,rfp=ps_rft,rff=f1_rft,nr
=rs_naive,np=ps_naive,nf=f1_naive,dr=rs_dt,dp=ps_dt,df=f1_dt,sr=rs_svm,sp=ps_svm,sf=f1
_svm,kr=rs_knn,kp=ps_knn,kf=f1_knn,lr=rs_l,lp=ps_l,lf=f1_l)
@app.route('/model_incremental', methods=['POST'])
def modelincremental():
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    a = cur.execute("select * from 'hist'")
    realse = a.fetchall()
    placement = pd.read_csv('hist.csv')
    return render_template('model1.html',data=placement)
@app.route('/modelinc', methods=['POST'])
def modelinc():
    return render_template('modelform.html')
@app.route('/refresh', methods=['POST'])
def refresh():
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
```

```
    cur = con.cursor()
    a = cur.execute("select * from 'hist'")
    realse = a.fetchall()
    placement = pd.read_csv('hist.csv')
    return render_template('model1.html', data=placement)
@app.route('/sendto', methods=['POST'])
def sendto():
    userid = session.get('userid', None)
    recruit=request.form['yes']
    comp=request.form['comp']
    print(recruit,comp)
    print(type(recruit))
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    a = cur.execute("select * from 'hist'")
    realse = a.fetchall()
    placement = pd.read_csv('hist.csv')
    b = cur.execute("select * from 'details' where email=?",(userid,))
    b1=b.fetchall()
    mydata = pd.DataFrame(np.array(b1))
    l=[]
    l1=[]
    for rows in b1:
        l.append(rows)
    print(l)
    for x in l[0]:
        l1.append(x)
    name=l1[0]
    branch = l1[ 3]
    cgpa = l1[4]
    yearofpass = l1[6]
    hsc = l1[8]
    ssc = l1[ 9]
    noc = l1[10]
    verbal =l1[11]
    comm = l1[12]
    quantitative = l1[13]
    logical =l1[14]
    certis = l1[15]
    extracurr =l1[16]
    drops = l1[19]
    print(drops)
    print(type(drops))
    check=cur.execute("select * from hist where name=?",(name,))
    num=check.fetchone()
    if num is None:
        cur.execute("Insert into
hist(name,branch,cgpa,yearofpass,noc,ssc,hsc,verbal,comm,quantitative,logical,certis,ex
tracurr,companies,recruited,drops)
```

```
values(?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)",(name,branch,cgpa,yearofpass,noc,hsc,ssc,verbal,com
m,quantitative,logical,certis,extracurr,comp,recruit,drops,))
        con.commit()
    else:
        pass
    con = sqlite3.connect("C:/Users/Chinmay
Chaughule/PycharmProjects/proj1/predictor1.db")
    cur = con.cursor()
    a = cur.execute("select * from 'hist'")
    realse = a.fetchall()
    placement = pd.read_csv('hist.csv')
    con.commit()
    return render_template('model1.html',data=placement)
app.run(debug=True)
```

CHAPTER 05

RESULTS AND DISCUSSIONS

5.1 DATA COLLECTION (DATABASE)

For the historical data it is not possible to get the whole data as it is in the normal format from TNP(Training and placement Cell). Thus data of students getting placed in the company has been collected from TNP and other columns are created by us using excel programming. Randomize values have been generated and that data is joined with the data collected from the TNP. Real time data i.e. present year data has been collected from the students and from staffs as well.

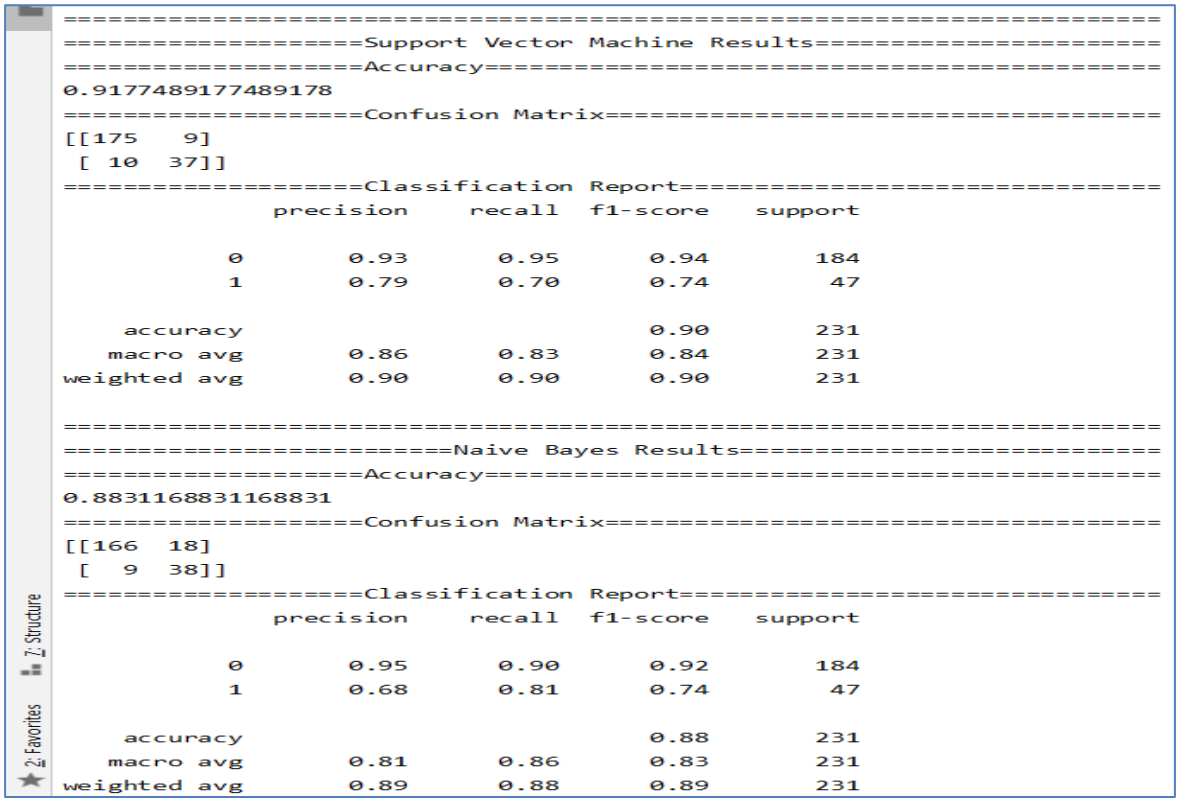
Database used for our project is SQLite which is the light weight.

5.2 SCREENSHOTS

Since Data Mining algorithm varies based on the dataset type and behavior of it. We have used various data mining algorithms to do the work. Some statistics or evaluation of classifier result for the following algorithms have been done-

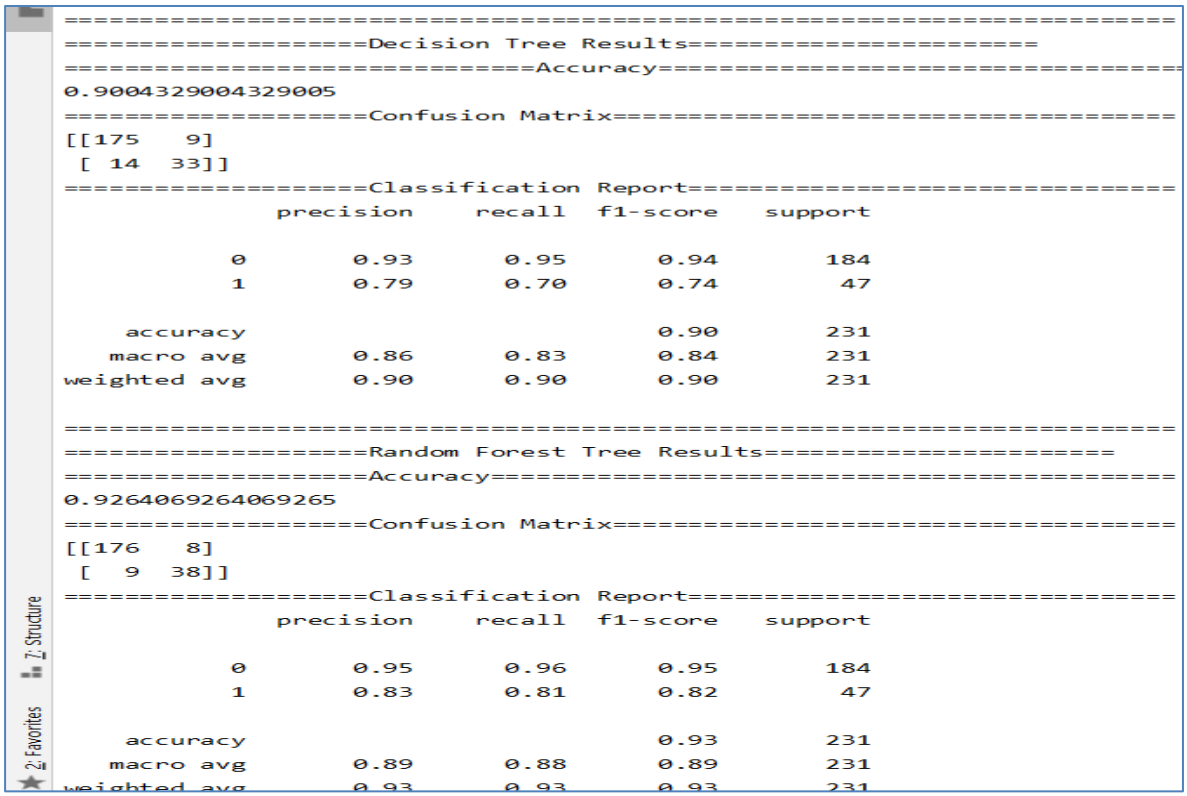
1. Naïve Bayes
2. Decision Tree
3. Support vector Machine
4. Random Forest Tree
5. KNN algorithm
6. Logistic Regression

Data Evaluation Results for different data mining algorithm



```
================================================================================
====================Support Vector Machine Results=============================
====================Accuracy====================================================
0.9177489177489178
====================Confusion Matrix============================================
[[175   9]
 [ 10  37]]
====================Classification Report=======================================
              precision    recall  f1-score   support

           0       0.93      0.95      0.94       184
           1       0.79      0.70      0.74        47

    accuracy                           0.90       231
   macro avg       0.86      0.83      0.84       231
weighted avg       0.90      0.90      0.90       231

================================================================================
============================Naive Bayes Results=================================
====================Accuracy====================================================
0.8831168831168831
====================Confusion Matrix============================================
[[166  18]
 [  9  38]]
====================Classification Report=======================================
              precision    recall  f1-score   support

           0       0.95      0.90      0.92       184
           1       0.68      0.81      0.74        47

    accuracy                           0.88       231
   macro avg       0.81      0.86      0.83       231
weighted avg       0.89      0.88      0.89       231
```

Fig 5.2.1. SVM & NB Evaluation Result



```
================================================================================
====================Decision Tree Results=======================
================================Accuracy====================================
0.9004329004329005
====================Confusion Matrix============================================
[[175   9]
 [ 14  33]]
====================Classification Report=======================================
              precision    recall  f1-score   support

           0       0.93      0.95      0.94       184
           1       0.79      0.70      0.74        47

    accuracy                           0.90       231
   macro avg       0.86      0.83      0.84       231
weighted avg       0.90      0.90      0.90       231

================================================================================
====================Random Forest Tree Results=======================
====================Accuracy====================================================
0.9264069264069265
====================Confusion Matrix============================================
[[176   8]
 [  9  38]]
====================Classification Report=======================================
              precision    recall  f1-score   support

           0       0.95      0.96      0.95       184
           1       0.83      0.81      0.82        47

    accuracy                           0.93       231
   macro avg       0.89      0.88      0.89       231
weighted avg       0.93      0.93      0.93       231
```

Fig 5.2.2. DT & RT Evaluation Result

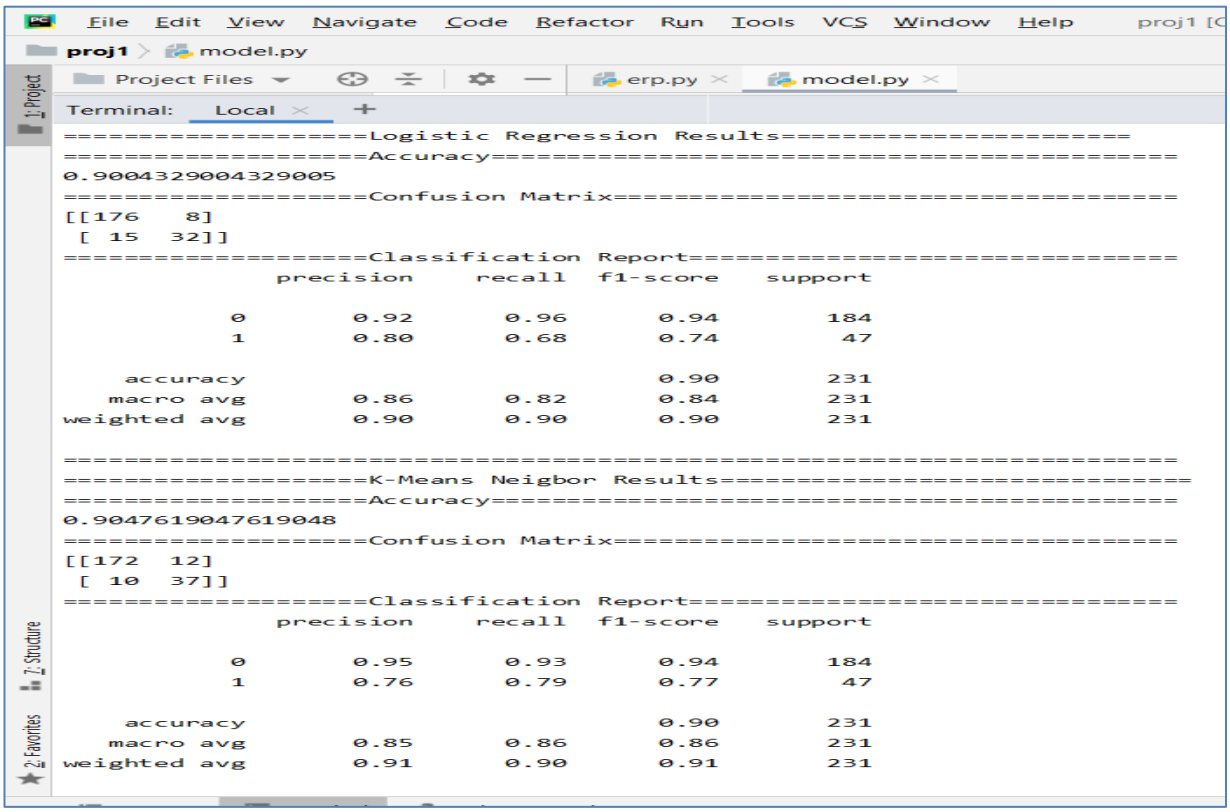


Fig 5.2.3. LR & KNN Result.

We have seen accuracy and classification reports for all the algorithms used. Now let us visualize the data used by us for the prediction purposes. The test data is tested with actual and predicted values depending on which data evaluation and visualization have been done. We have focuses on the attributes like communication skills, aptitude, drops, extra activities than academics etc. These all data have been compared with the class so that to check their dependency/relation with the prediction. Some results are as shown below.

For Decision Tree-

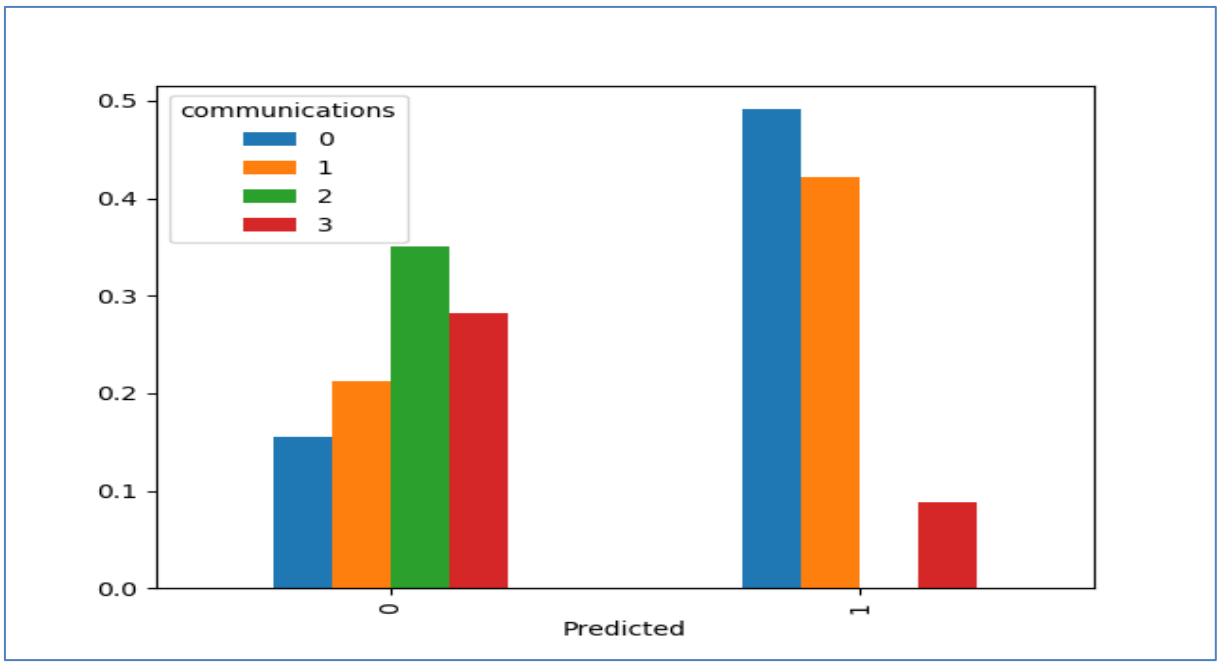


Fig No 5.2.4 Communications against Class

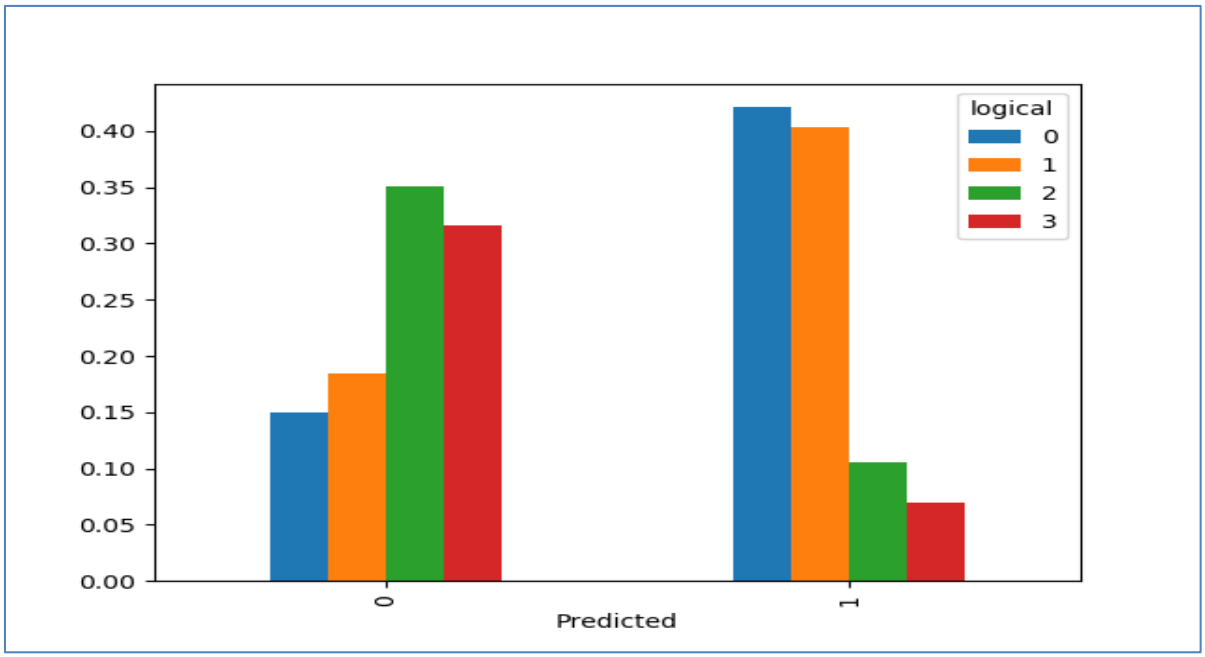


Fig no 5.2.5 Logical Reasoning against class.

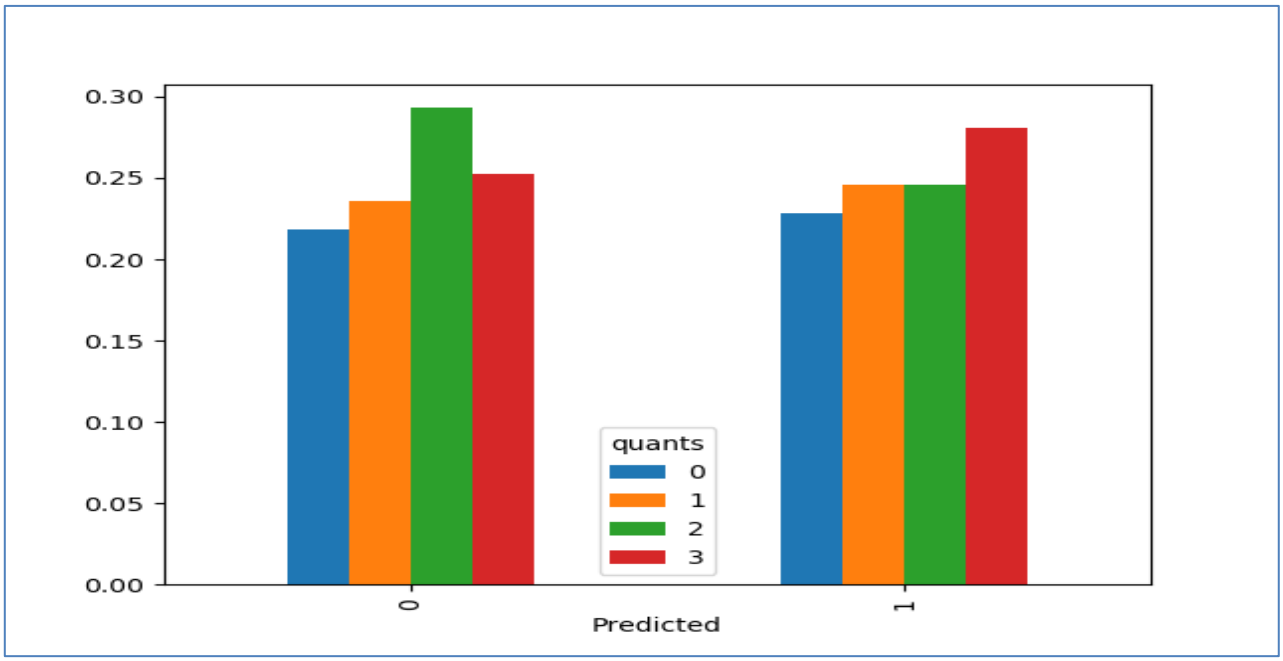


Fig no 5.2.6 Quantitative against class

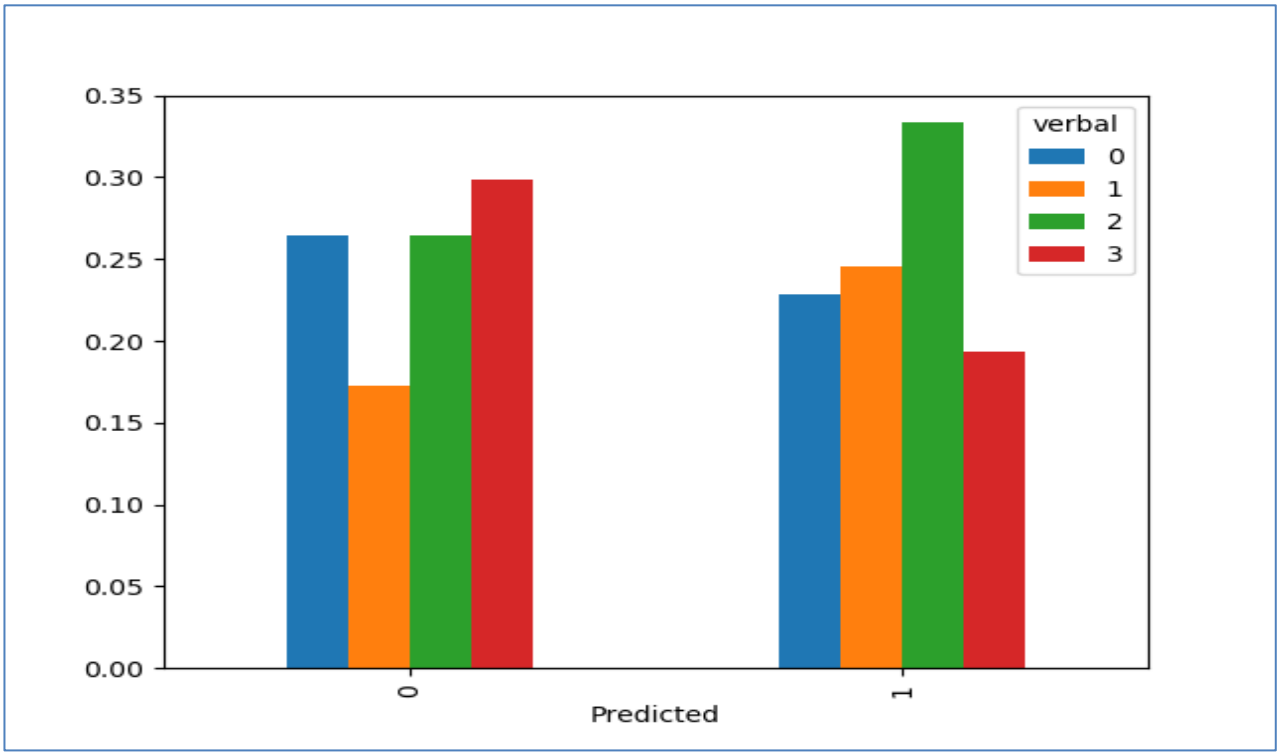


Fig no 5.2.7 Verbal against class.

Similarly done for other five algorithms. In the previous figure (fig no 5.2.7), Verbal skills are considered and their relation with class I.e. yes or no is judged. In the fig, on x axis 0 stands for not placed and 1 for placed. The color with 0 label is for Average,1 for excellent, 2 for good and 3 for poor. Thus we can see that for non-placed class category Poor, Good and Average are having great count whereas it flips for placed students with Good and Excellent verbal are been placed with huge count. This changes according to the algorithms used.

CHAPTER 06
CONCLUSION

Data Mining has powerful classifiers to predict how many students will be placed and which students will be placed in particular year and accordingly students will be trained with more attention. We have used different classification techniques in our project and compared every algorithm i.e. Naïve Bayes, Decision tree, Logistic-regression, KNN, SVM, RFT. We got highest accuracy for SVM & RFT to predict placement chances of individual students. On the other side TNP will look towards which all students are recommended by our system for the particular company and accordingly notification will be send to those students regarding to their company and also to the company about the recommended students list. For company recommended students Collaborative Filtering has been used.

CHAPTER 07

FUTURE SCOPE

Employing such a system is a proactive way to use data to manage, operate, and evaluate educational institute in a better way. It helps an educational institute improving the quality and placements of students being graduated from their institute. This system is also universal for all types of technological institutes which can help students as well as the institute to enhance in their own ways. By providing the recommended student's eligibility list to the TNP, this will improve the time and efforts of the staffs who does it manually. This system can be used at IT Companies as well for employees rating for giving raise. The future enhancements of the project is to focus on to add some more parameters to predict more efficient placement status. We can also enhance the project by predicting some solutions or suggestions for the output generated by system.

REFERENCES

[1] Ashok MV, Apoorva A," Data Mining Approach for Predicting Student and Institution's Placement Percentage "(2016) International Conference on Computational Systems and Information Systems for Sustainable Solutions

[2] Oktariani Nurul Pratiwi Teknik Informatika, (2013). Predicting student placement class using data mining. Proceedings of 2013 IEEE International Conference on Teaching, Assessment and Learning for Engineering (TALE).

[3] Sagardeep Roy, Anchal Garg," Analyzing Performance of Students by Using Data Mining Techniques" (2017)4th IEEE Uttar Pradesh Section International Conference on Electrical, Computer and Electronics

[4] K. Manikandan1, S. Sivakumar2, M. Ashokvel." A Classification Model for Predicting Campus Placement Performance Class using Data Mining Technique" (2018)

[5] Priyanka P. Wadekar, Yedhukrishnan P. Pillai, Manodeep U. Roy, Prof. Neelam Phadnis" Placement Predictor and Course Recommender System" (2018) International Research Journal of Engineering and Technology.

[6] Mangasuli Sheetal B1 , Prof. Savita Bakare2" Prediction of Campus Placement Using Data Mining Algorithm-Fuzzy logic and K nearest neighbour ("2016) International Journal of Advanced Research in Computer and Communication Engineering.

[7] Ajay Shiv Sharma, Swaraj Prince, Shubham Kapoor "PPS - Placement Prediction System using Logistic Regression"(2017)International Conference on Advanced Computing and Communication Systems.

[8] Senthil Kumar Thangavel,Divya Bharathi P, Abijith Sankar "Student Placement Analyzer: A Recommendation System Using Machine Learning" (2017)International Conference on Advanced Computing and Communication Systems (ICACCS 2017).