PROJECT REPORT

Project name: University Admit Eligibility Predictor

Table Of Contents:

Author: S Ashwin Kumar

INTRODUCTION

a) Overview:

The project is implemented using a Machine-Learning model that predicts whether the user is eligible for an admission in the selected rated universities with provided details such as marks and others. The algorithm works in such a way that when the user provides the details such as (GRE Score, TOEFL Score, University Rating, SOP, LOR, CGPA, Research) the percentage of chance of admit is displayed.

The user is provided with a UI (Web based application) in which the user can enter the details mentioned above for prediction. The main advantage of this is that the user can avoid long process of having to check the eligibility of a university admission by himself and make use of this application to predict the eligibility / chance of admit.

b) Purpose:

The purpose of this project is to make the prediction of eligibility of an admission to a rated university with ease using a UI with the provided user details (GRE Score, TOEFL Score, University Rating, SOP, LOR, CGPA, Research). This also eliminates the possibility of human errors.

LITERATURE SURVEY

a) Existing problem:

Previous research done in this area used Naive Bayes algorithm which will evaluate the success probability of student application into a respective university but the main drawback is they didn't consider all the factors which will contribute in the student admission process like TOEFL/IELTS, SOP, LOR and under graduate score.

Bayesian Networks Algorithm have been used to create a decision support network for evaluating the application submitted by foreign students of the university. This model was developed to forecast the progress of prospective students by comparing the score of students currently studying at university. The model thus predicted whether the aspiring student should be admitted to university on the basis of various scores of students. Since the comparisons are made only with students who got admission into the universities but not with students who got their admission rejected so this method will not be that much accurate.

b) Proposed solution:

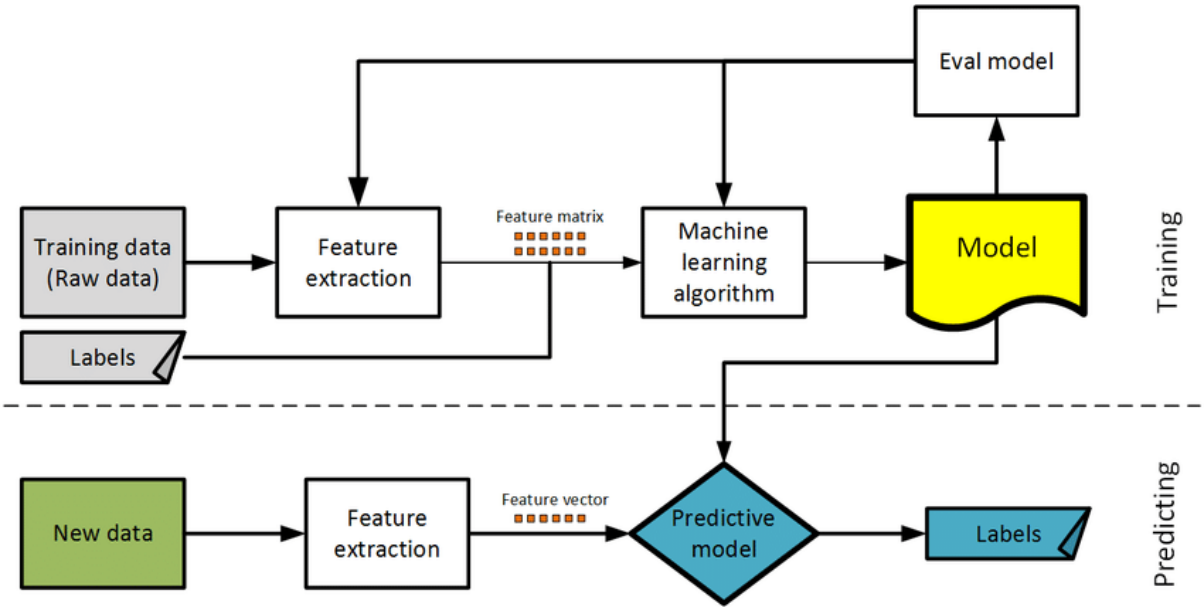
These problems can be resolved by using regression algorithms / classification algorithms as they can consider most of the features for prediction. Linear regression / KNN classification / Random forest Regressor can be used as the machine learning model for the model. XG boost model can also be used which performs better on small to medium scale datasets but the model giving accurate and desired results only will be selected.

The aim of the proposed system is to address the limitations of the current system. The requirements for the system have been gathered from the defects recorded in the past and also based on the feedback from users of previous metrics tools. Following are the objectives of the proposed system:

- Reach to geographically scattered student.
- Reducing time in activities
- Paperless admission with reduced man power
- Operational efficiency

THEORETICAL ANALYSIS

a)Block diagram:

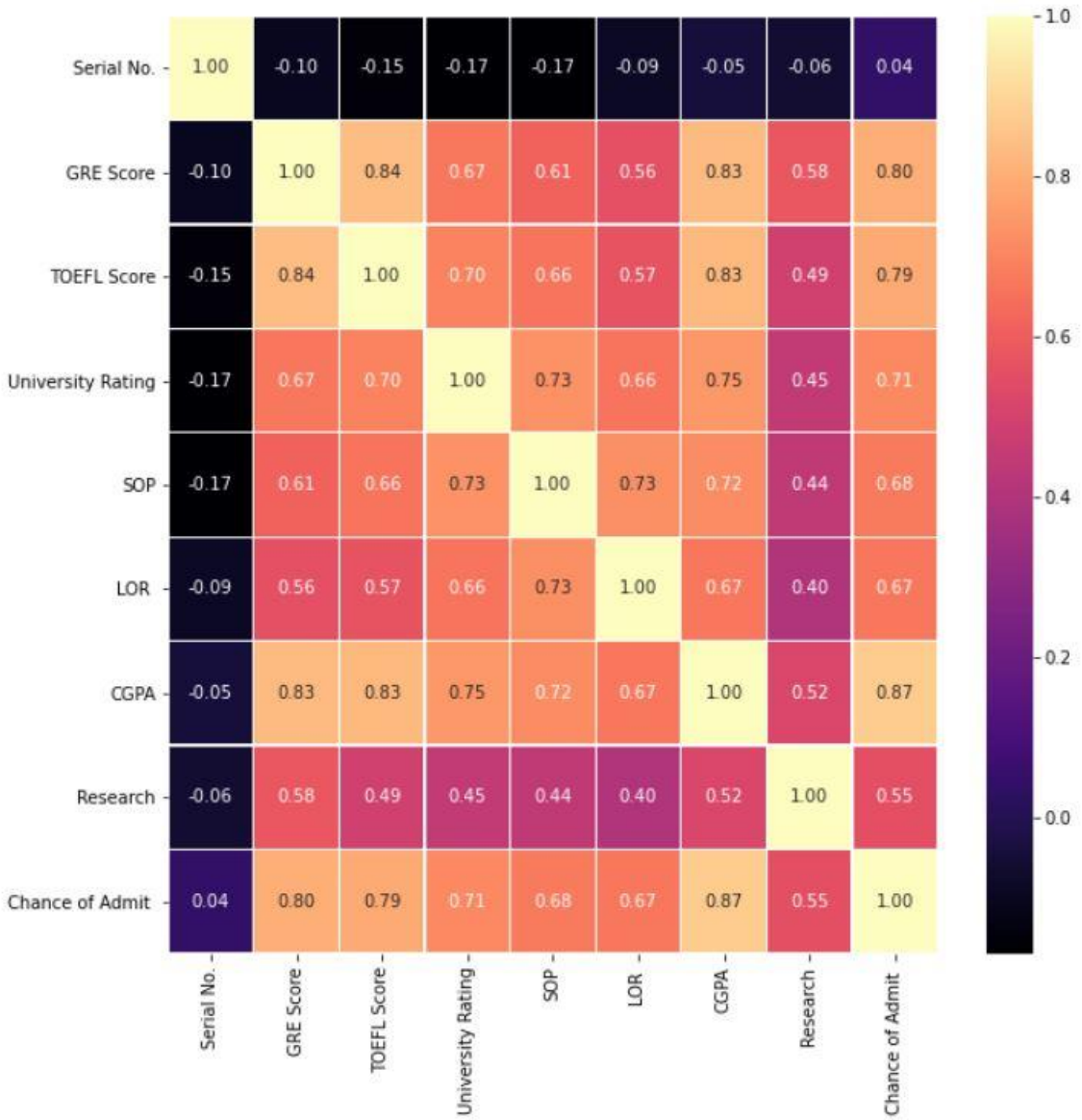


b)Hardware/Software designing:

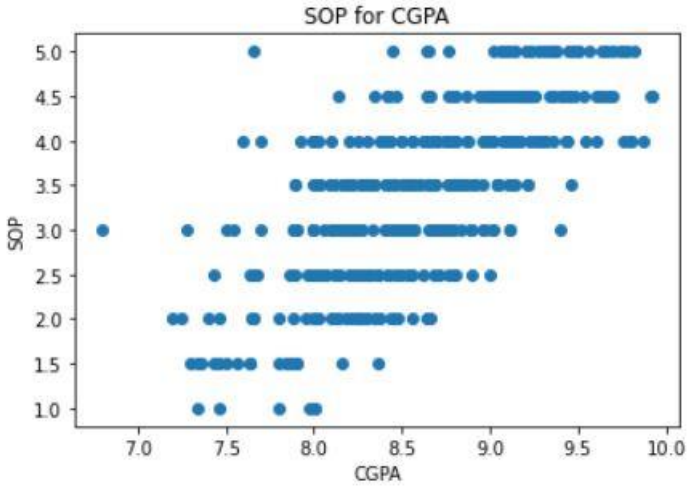
ITEM	CONTENT
CPU	Intel i3 7th gen or above/ AMD Ryzen 3500 or above
GPU	Integrated / Dedicated (atleast 2GB)
RAM	8 GB
Operating system	Windows 10/ Linux / MacOS
Programming language	Python 3.9
ML library	Scikit learn
Other libraries	Matplot,seaborn,numpy,pandas,pickle

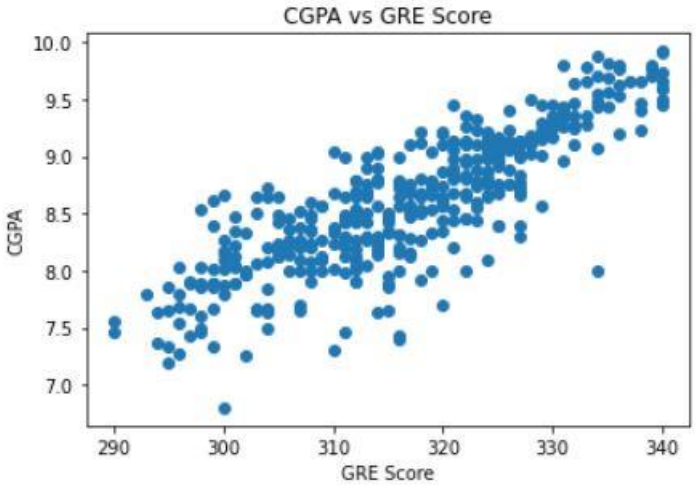
EXPERIMENTAL ANALYSIS

Heat map of the data.

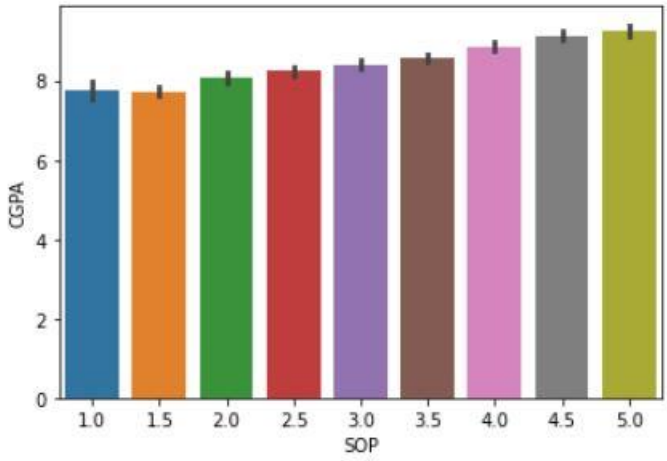


Scatter Plots , Bar graphs

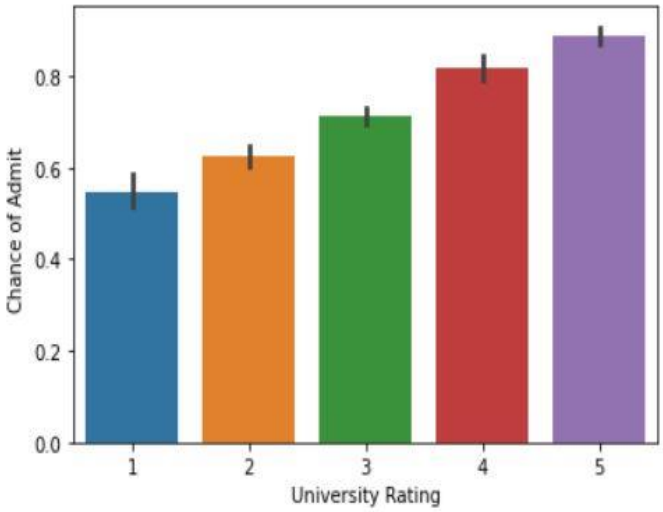




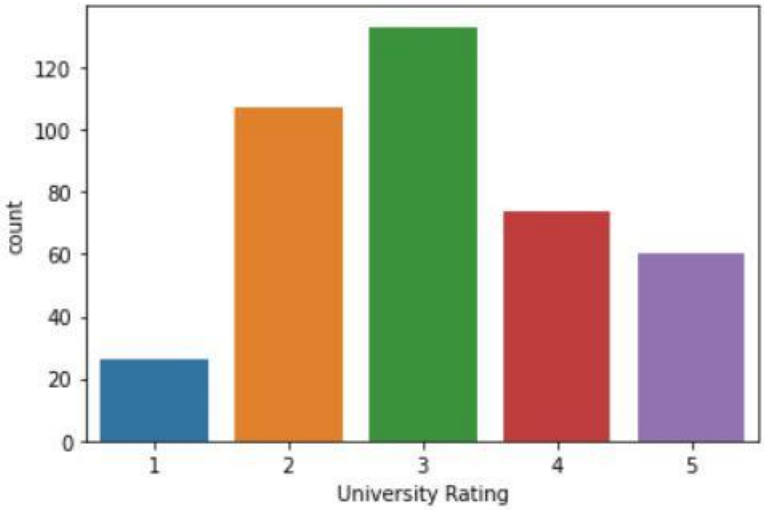

```
<AxesSubplot:xlabel='SOP', ylabel='CGPA'>
```



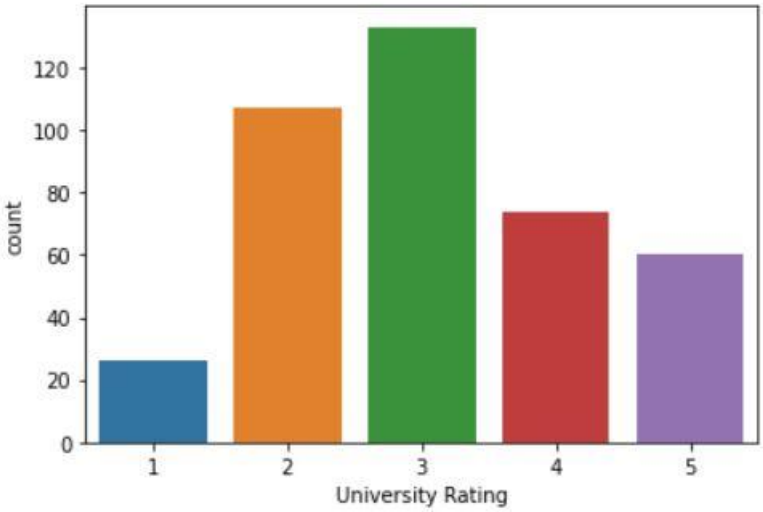
```
<AxesSubplot:xlabel='University Rating', ylabel='Chance of Admit '>
```



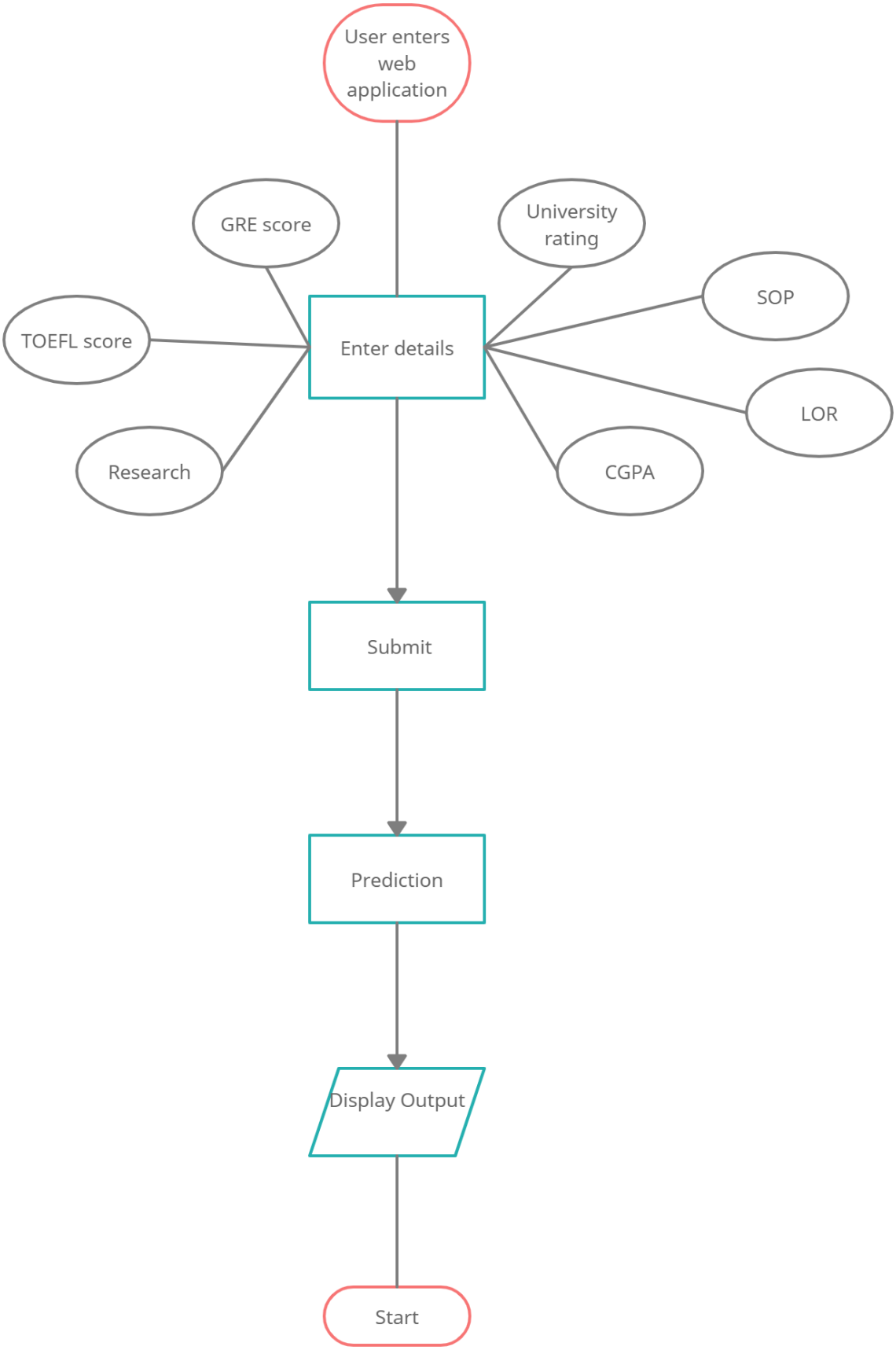

```
<AxesSubplot:xlabel='University Rating', ylabel='count'>
```



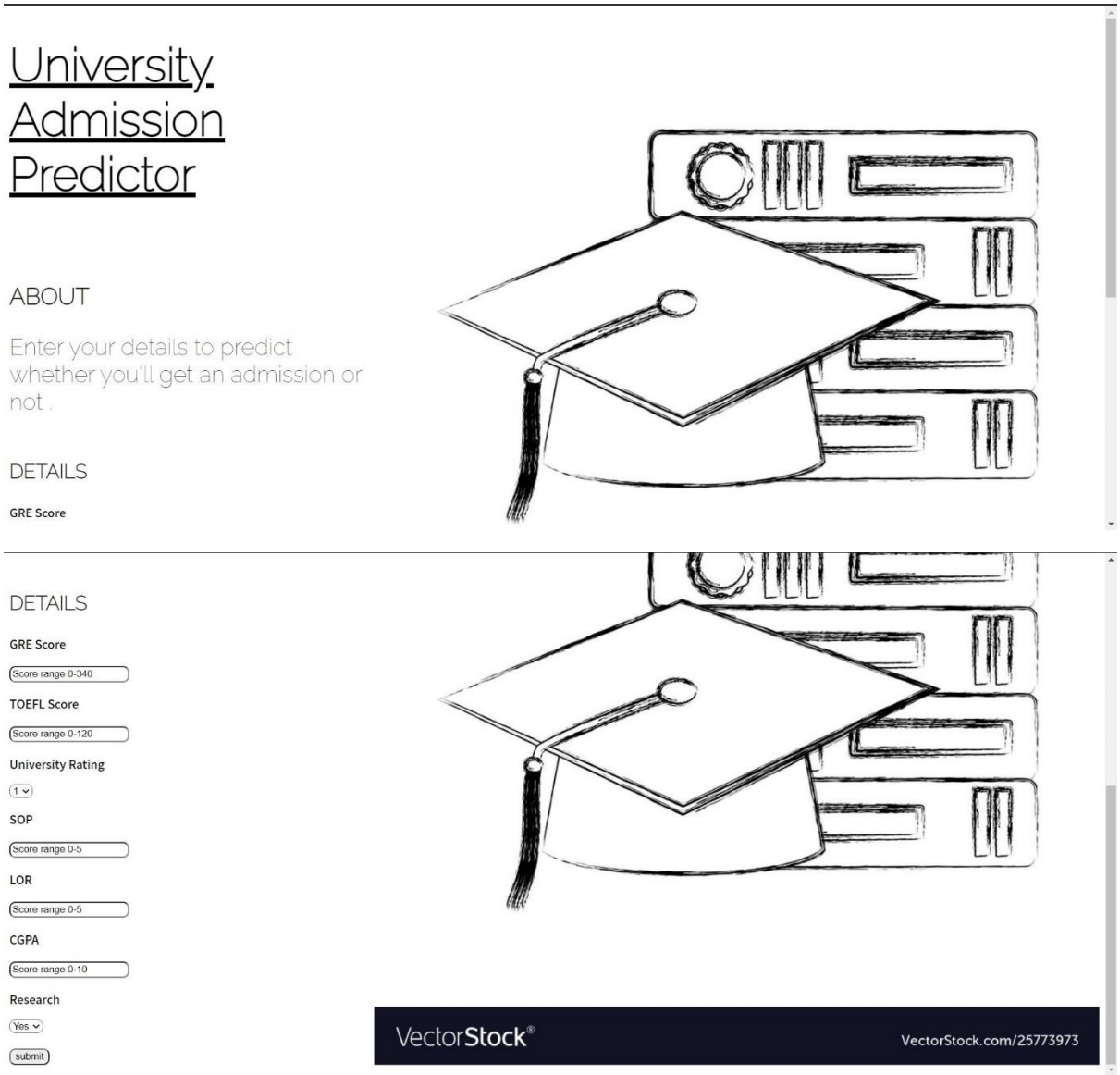

```
<AxesSubplot:xlabel='University Rating', ylabel='count'>
```



FLOWCHART


User enters web application
GRE score
University rating
TOEFL score
Enter details
SOP
LOR
Research
CGPA
Submit
Prediction
Display Output
Start

RESULT


University Admission Predictor
ABOUT
Enter your details to predict whether you'll get an admission or not .
DETAILS
GRE Score
DETAILS
GRE Score
Score range 0-340
TOEFL Score
Score range 0-120
University Rating
1
SOP
Score range 0-5
LOR
Score range 0-5
CGPA
Score range 0-10
Research
Yes
submit
VectorStock®
VectorStock.com/25773973

Predicting chance of admission

A Machine Learning Web App Using Flask

Prediction : **You've a 62% chance to get the admission !**

Predicting chance of admission

A Machine Learning Web App Using Flask

Prediction : **You don't have a chance!**

ADVANTAGES

- It helps student for making decision for choosing a right college.
- Here the chance of occurrence of error is less when compared with the existing system.
- Avoids data redundancy and inconsistency.
- It is fast, efficient and reliable.

DISADVANTAGES

- Machine errors are unavoidable when occurred. (Hardware failure, network failure, others).

- The predictions made are not 100% accurate but accurate to an acceptable value.

APPLICATIONS

- Reach to geographically scattered student.
- Reducing time in activities
- Paperless admission with reduced man power
- Operational efficiency

CONCLUSION

The project uses a Random forest regressor to predict the output and a web application is built to make the UI more accessible and easy using various technologies such as python, HTML5, CSS, Flask, Scikit, Matplot, Numpy, Pandas, Seaborn and other libraries. After the deployment of the web application, it can be accessed from anywhere with internet connection. This project reduces the long hours of analysis to predict the eligibility of the admission to a rated university.

FUTURE SCOPE

The future scope of this project is very broad. Few of them are:

- This can be implemented in less time for proper admission process.
- This can be accessed anytime anywhere, since it is a web application provided only an internet connection.
- The user had not need to travel a long distance for the admission and his/her time is also saved as a result of this automated system.

BIBLIOGRAPHY

- L. Chang , Applying Data Mining to Predict College Admissions Yield, Chapter 4 in J. Luan and C. Zhao (Eds.), Data mining in action: Case studies, Spring 2008 - College of Education.
- Rensong Dong,The module of prediction of College Entrance Examination aspiration, Fuzzy Systems and Knowledge Discovery (FSKD), 31 May 2012 ,1559-1562.

- Data Visualizaton, Machine Learning https://www.analyticsvidhya.com/blog/2017/09/common-machin e-learning-algorithms/
- International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume-8, Issue-6 March 2020
- Journal of Network Communications and Emerging Technologies (JNCET) Volume 8, Issue 4, April (2018)

APPENDIX

a) Source code:

Python code(ML model):

```
# IMPORTING LIBRARIES , DATASET
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
data=pd.read_csv('Admission_Predict.csv')
print(data.columns,data.shape)

# DATA PRE-PROCESSING
print(data.isnull().sum())
print(data.corr())
print(data.info())
print(data.describe())
print("University ratings:",set(data["University
Rating"]),"Research:",set(data["Research"]))

#DATA VISUALISATION
sns.barplot(data["GRE Score"],data["CGPA"])
sns.barplot(data["SOP"],data["CGPA"])
sns.barplot(data["TOEFL Score"],data["CGPA"])
```

```
plt.scatter(data['GRE Score'],data['CGPA'])
plt.title('CGPA vs GRE Score')
plt.xlabel('GRE Score')
plt.ylabel('CGPA')
plt.show()

plt.scatter(data['CGPA'],data['SOP'])
plt.title('SOP for CGPA')
plt.xlabel('CGPA')
plt.ylabel('SOP')
plt.show()

plt.figure(figsize=(10, 10))
sns.heatmap(data.corr(), annot=True, linewidths=0.05, fmt= '.2f',cmap="magma")
plt.show()

data.Research.value_counts()
sns.countplot(x="University Rating",data=data)

data.Research.value_counts()
sns.countplot(x="University Rating",data=data

sns.barplot(x="University Rating", y="Chance of Admit ", data=data)

# DATA TRANSFORMATION
x=data.drop(['Serial No.','Chance of Admit '],axis=1)
y=data['Chance of Admit ']
print(x.shape,y.shape)

from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2)
print(x_train.shape)
print(x_test.shape)
print(y_train.shape)
```

```
print(y_test.shape)

from sklearn.preprocessing import MinMaxScaler
mms=MinMaxScaler()
x_train[x_train.columns]=mms.fit_transform(x_train[x_train.columns].values)
x_test[x_test.columns]=mms.transform(x_test[x_test.columns].values)

# MODEL BUILDING(RANDOM FOREST REGRESSOR)
from sklearn.ensemble import RandomForestRegressor
model=RandomForestRegressor()
model.fit(x_train,y_train)
y_pred=model.predict(x_test)
print(y_pred)
print(y_test)

#evaluation
from sklearn.metrics import mean_squared_error, r2_score,mean_absolute_error
print('model score:',model.score(x_test,y_test))
print('Mean Absolute Error:', mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:', mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test, y_pred)))
print('roc score:',roc_auc_score(y_test>0.5, y_pred>0.5))
print('recall score:',recall_score(y_test>0.5, y_pred>0.5))

# MODEL BUILDING(LINEAR REGRESSION)
x1=data.drop(['Serial No.','Chance of Admit '],axis=1)
y1=data['Chance of Admit ']
x1_train,x1_test,y1_train,y1_test=train_test_split(x1,y1,test_size=0.2)
from sklearn.preprocessing import StandardScaler
sc=StandardScaler()
x1_train=sc.fit_transform(x1_train)
x1_test=sc.fit_transform(x1_test)

from sklearn.linear_model import LinearRegression
```

```
model1=LinearRegression()
model1.fit(x1_train,y1_train)
y1_pred=model1.predict(x1_test)
print('model score:',model1.score(x1_test,y1_test))
print('Mean Absolute Error:', mean_absolute_error(y1_test, y1_pred))
print('Mean Squared Error:', mean_squared_error(y1_test, y1_pred))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y1_test, y1_pred)))
print('roc score:',roc_auc_score(y1_test>0.5, y1_pred>0.5))
print('recall score:',recall_score(y1_test>0.5, y1_pred>0.5))

# MODEL BUILDING (LOGISTIC REGRESSION)
x2=data.iloc[:,1:8].values
y2=data.iloc[:,-1:].values
x2_train,x2_test,y2_train,y2_test=train_test_split(x1,y1,test_size=0.2)
y2_train=y2_train>0.5
y2_test=y2_test>0.5

from sklearn.linear_model import LogisticRegression
model2=LogisticRegression()
model2.fit(x2_train,y2_train)

#evaluation
from sklearn.metrics import accuracy_score,roc_auc_score,recall_score
y2_pred=model2.predict(x2_test)
print('model score:',model2.score(x2_test,y2_test))
print('roc score:',roc_auc_score(y2_test, y2_pred))
print('recall score:',recall_score(y2_test, y2_pred))
print(type(y2_test),type(y2_pred))

# SAVING THE MODEL
#Though the accuracy of Logistic regression model is more we prefer Random forest regressor if we also want the percentage of chance or else we can use Logistic regression model.
import pickle
```

```
pickle.dump(model,open('model.pkl','wb'))
```

HTML:

Inde.html:

```
<!DOCTYPE html>
<html>
<head>
	<title>University Admission Predictor</title>
</head>
<link rel="preconnect" href="https://fonts.gstatic.com">
<link
href="https://fonts.googleapis.com/css2?family=Raleway:wght@100&display=swap"
rel="stylesheet">
<link
href="https://fonts.googleapis.com/css2?family=Noto+Sans+HK:wght@500&display
=swap" rel="stylesheet">
<style type="text/css">

	h1,h2{
		font-family: 'Raleway', sans-serif;
		color: black;
	}
	h2,h1,form,p,b{
		text-align: left;
		color: black;
	}
	label,p,b{
		font-family: 'Noto Sans HK', sans-serif;
		color: black;
	}
	.elements{
		padding-top: 2px;
	}
```

```
</style>
<body>

    <img src="https://cdn4.vectorstock.com/i/1000x1000/39/73/graduation-hat-stacked-books-school-vector-25773973.jpg" style="position: -webkit-sticky; position: sticky; top: 0;" align="right">
    <h1 style="font-size: 4rem; text-decoration-line: underline; text-decoration-thickness: auto;">University Admission Predictor</h1>
    <br>
    <br>
    <p style="font-size: 2rem; font-family: 'Raleway', sans-serif;"> <strong> ABOUT </strong></p>
    <p style="font-size: 2rem; font-family: 'Raleway', sans-serif;">Enter your details to predict whether you'll get an admission or not .</p>

    <form action="/predict" method="post" class="elements" style="font-size: 1rem;">
        <p style="font-size: 1.7rem; font-family: 'Raleway', sans-serif;"> <strong> DETAILS </strong></p>
        <p class="elements">GRE Score</p>
        <p><input type="text" name="gre" value="Score range 0-340" style=" border-radius: 8px;"></p>
        <p class="elements">TOEFL Score</p>
        <p><input type="text" name="tofl" value="Score range 0-120" style=" border-radius: 8px;"></p>
        <p class="elements"><label>University Rating</label> </p>
        <select name="rating" style=" border-radius: 8px;">
            <option value="1">1</option>
            <option value="2">2</option>
            <option value="3">3</option>
            <option value="4">4</option>
            <option value="5">5</option>
    </select>
        <br>
```

```
                <p class="elements">SOP</p>
                <p><input type="text" name="sop" value="Score range 0-5" style="
border-radius: 8px;"></p>
                <p class="elements">LOR</p>
                <p><input type="text" name="lor" value="Score range 0-5" style="
border-radius: 8px;"></p>
                <p class="elements">CGPA</p>
                <p><input type="text" name="cgpa" value="Score range 0-10" style="
border-radius: 8px;"></p>
                <p class="elements"><label>Research</label></p>
                <select name="research" style=" border-radius: 8px;">
                        <option value="Yes">Yes</option>
                        <option value="No">No</option>
      </select>




                <p class="elements"> <input type = "submit" value = "submit" style="
border-radius: 8px;"/> </p>
        </form>

</body>
</html>
```

Chance.html:

```
<!DOCTYPE html>
<html>
<head>
        <title>eligibility</title>
</head>
<body>
```

```
        <img src="https://thumbs.dreamstime.com/b/group-smiling-girls-boys-students-standing-together-happy-friends-isolated-white-background-flat-cartoon-style-vector-150091547.jpg" style="float: right;">
        <div style="padding-top: 15%">
        <h2>Predicting chance of admission</h2>
        <h3>A Machine Learning Web App Using Flask</h3>
        <p>Prediction : <b>You've a <b>{{p}}</b> chance to get the admission !</b><img src="thumbsup.png" height="2%" width="2%" ></p>
    </div>
</body>
</html>
```

Nochance.html:

```
<!DOCTYPE html>
<html>
<head>
        <title>eligibility</title>
</head>
<body>
        <img src="https://ih1.redbubble.net/image.638538259.9192/st,small,845x845-pad,1000x1000,f8f8f8.u1.jpg" style="float: right; " height="50%" width="50%">
        <div style="padding-top: 15%">
        <h2>Predicting chance of admission</h2>
        <h3>A Machine Learning Web App Using Flask</h3>
        <p>Prediction : <b>You don't have a chance!</b><img src="thumbsdown.png" height="2%" width="2%"></p>
        </div>
</body>
</html>
```

Python code for backend:

```
import pickle
from flask import Flask , request, render_template
from math import ceil
app = Flask(__name__)
model = pickle.load(open("model.pkl","rb"))
```

```
@app.route('/')
def index():
        return render_template('index.html')

@app.route('/predict',methods = ['GET','POST'])
def admin():
    gre=(eval(request.form["gre"])-290)/(340-290)
    tofl=(eval(request.form["tofl"])-92)/(120-92)
    rating=(eval(request.form["rating"])-1.0)/4.0
    sop=(eval(request.form["sop"])-1.0)/4.0
    lor=(eval(request.form["lor"])-1.0)/4.0
    cgpa=(eval(request.form["cgpa"])-290.0)/(340.0-290.0)
    research=request.form["research"]
    if (research=="Yes"):
        research=1
    else:
        research=0
    preds=[[gre,tofl,rating,sop,lor,cgpa,research]]
    xx=model.predict(preds)
    if (xx>0.5):
        return render_template("chance.html",p=str(ceil(xx[0]*100))+"%")
    return render_template("nochance.html")
if __name__ == '__main__':
    app.run(debug = False, port=4000)
```

b) UI output screenshot:

DETAILS

GRE Score

300

TOEFL Score

97

University Rating

2

SOP

3

LOR

3

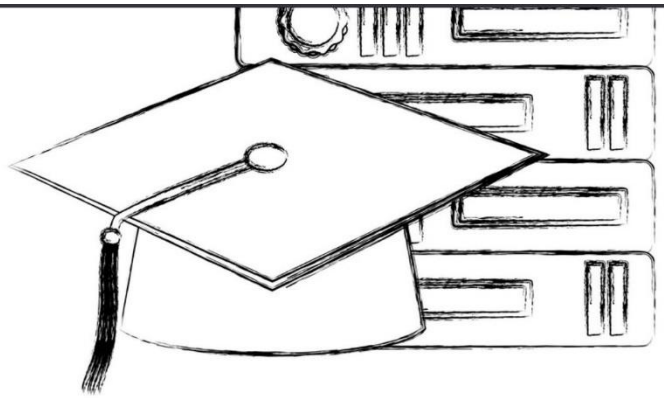
CGPA

8

Research

No

submit



Predicting chance of admission

A Machine Learning Web App Using Flask

Prediction : **You don't have a chance!**