



Government of Tamil Nadu

Naan Muthalvan - Project-Based Experiential Learning

Intelligent Admissions: The future of University decision making with Machine Learning

Submitted by

Team ID: NM2023TMID22881

D.Ramya - (20326ER060)

J.Ramya - (20326ER061)

S.Sandhiya - (20326ER062)

S.Sangeetha - (20326ER063)

Under the guidance of
Mrs. P.SANGEETHA M.SC., M.Phil.,
Guest Lecturer

PG and Research Department of Computer Science



M.V.MUTHIAH GOVERNMENT ART COLLEGE FOR WOMEN

(Affiliated To Mother Teresa Women's University, Kodaikanal)

Reaccredited with "A" Grade by NAAC

DINDIGUL-624001. APRIL-2023

M.V.MUTHIAH GOVERNMENT ARTS COLLEGE FOR WOMEN

(Affiliated to Mother Teresa Women's University, Kodaikanal)

Reaccredited with "A" Grade by NAAC

Dindigul-624001



PG & RESEARCH DEPARTMENT OF COMPUTER SCIENCE

BONAFIDE CERTIFICATE

This is to certify that this is a bonafide record of the project entitled, done by **Ms.D.RAMYA- (20326ER060), Ms. J.RAMYA (20326ER061), Ms.S.SANDHIYA (20326ER062), AND Ms.S.SANGEETHA (20326ER063)**. This is submitted in partial fulfillment for the award of the degree of **Bachelor of Science in Computer Science in M.V.MUTHIAH GOVERNMENT ARTS COLLEGE FOR WOMEN,DINDIGUL** during the period of December 2022 to April 2023.

Project Mentor(s)

Head of the Department

Submitted for viva-voce Examination held on _____

INTERNAL EXAMINER

EXTERNAL EXAMINER

TABLE OF CONTENTS

S.No	CONTENTS	PAGE NO.
1	INTRODUCTION	1
	1.1 Over view	1
	1.2 Purpose	2
2	PROBLEM DEFINITION	3
	2.1 Empathy Map	3
	2.2 Ideation & Brainstorming Map	4
3	RESULT	8
4	ADVANTAGES & DISADVANTAGE	10
5	APPLICATION	12
6	CONCLUSION	13
7	FUTURE SCOPE	14
8	APPENDIX	15
	8.1 Source Code	15

1. INTRODUCTION

1.1 Overview

Admission predict is a common task in the field of education, where the goal is to predict whether a student will be admitted to a particular college or university based on a set of input features such as their grades, test scores, extracurricular activities, and personal characteristics. This task can be framed as a binary classification problem, where the model must learn to distinguish between students who are likely to be admitted and those who are not.

To solve this task, various machine learning algorithms such as logistic regression, decision trees, and neural networks have been applied. These models are trained on a labeled dataset that contains information about past applicants and their admission status. The trained models are then used to predict the admission status of new applicants based on their input features.

Admission prediction models have many potential applications, such as assisting universities with their admissions process, helping students identify which schools they are most likely to be accepted to, and providing insights into the factors that contribute to admission decisions. However, it is important to note that admission prediction models must be used ethically and responsibly, and should not be used to unfairly discriminate against certain groups of applicants based on factors such as race, gender, or socioeconomic status.

1.2 Purpose

The purpose of admission prediction is to develop models that can accurately predict whether a student will be admitted to a particular college or university based on a set of input features. This task is important because it can help both students and universities make better-informed decisions about the college admissions process.

For students, admission prediction models can help them identify which schools they are most likely to be accepted to based on their academic and personal characteristics. This can help students make more informed decisions about where to apply, and can also help them focus their efforts on schools where they are more likely to be admitted.

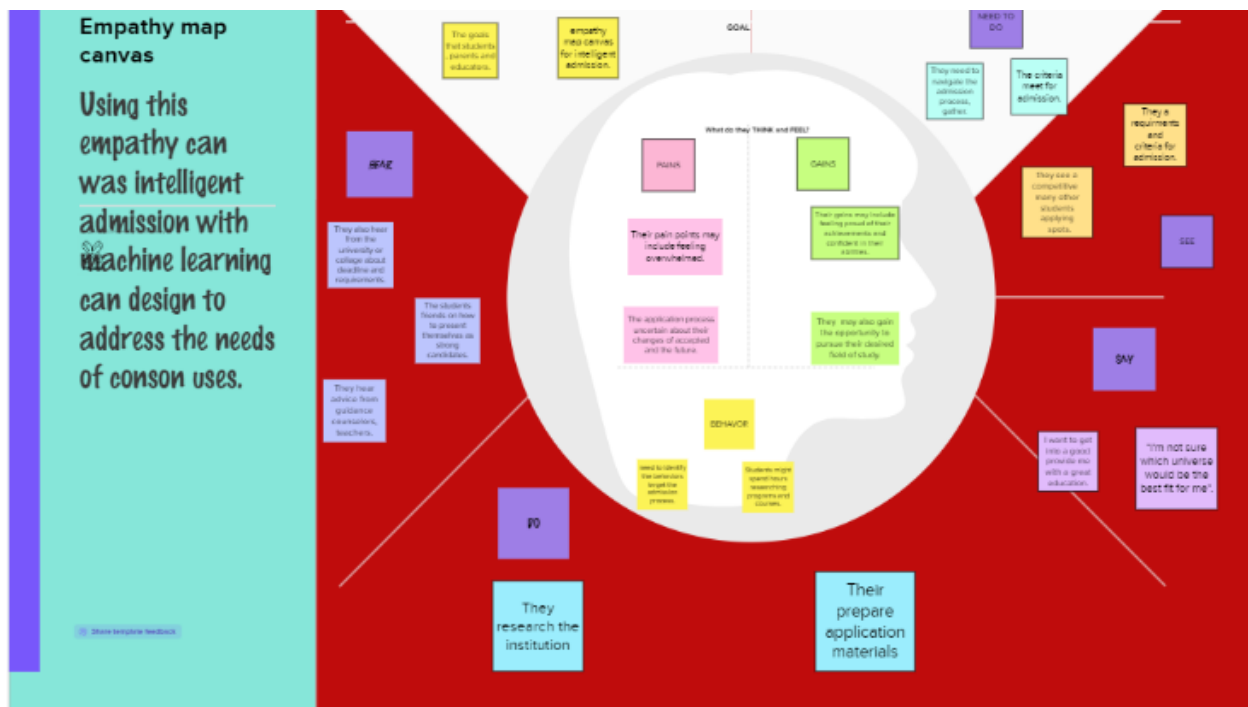
For universities, admission prediction models can assist with the admissions process by providing a more efficient and objective way of evaluating applicants. By using data-driven models to predict admission decisions, universities can potentially reduce bias and ensure that all applicants are evaluated fairly and consistently.

Overall, the purpose of admission prediction is to improve the efficiency and fairness of the college admissions process, and to help students and universities make better-informed decisions about which schools to apply to and admit.

2. Problem Definition

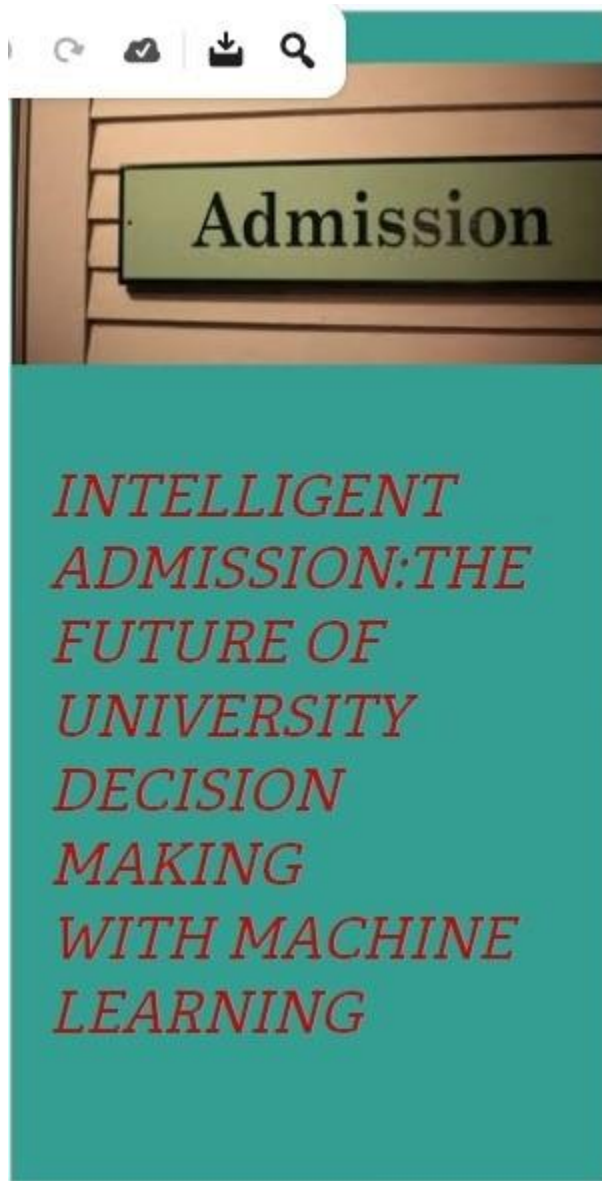
2.1 Empathy map

An empathy map & Design Thinking map is a collaborative tool teams can use to gain a deeper insight into their customers.



2.2 Ideation & Brainstrom Map

A mind map is a highly effective tool used by creatives, marketers, and project manager to inspire their teams.





The image is a screenshot of a presentation slide. At the top, there is a white toolbar with icons for undo, redo, checkmark, download, and search. The slide itself has a light blue background. The title 'Define your problem statement' is in black. Below it, there are five numbered points in a dark red font. The text is slightly blurry, suggesting it was taken from a screen.

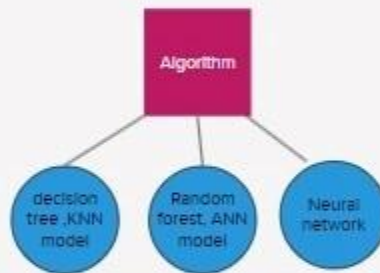
Define your problem statement

1. **student admission are playing very important role in major activity of any university**
2. **The aim of project is to help student in short listing university with their profiles**
3. **This project is design to development intelligent admission**
4. **The goal intelligent admission is provide convience,savetime,bring more object , transperancy and speed transaction over the manual opertion.**
5. **This predicted output gives them fair idea about their admission.**

Group ideas

1. Intelligent admission the goal of identifying who are best suited to the university.
2. The algorithm used by
20 minutes
decision tree,
random forest,
neural network,
KNN model,
ANN model using.
3. The Google colab cloud based platform for running python code and ml models.
4. The 8GB Hard disk required and 16GB RAM is required.
5. ML libraries such as scikit-learn, TensorFlow, Keras and PyTorch.

The goal of admission process is to identify who are best suited to university.



The Google colab cloud based platform for running python code and ml models.

The 8GB Hard disk required and 16GB RAM is required.

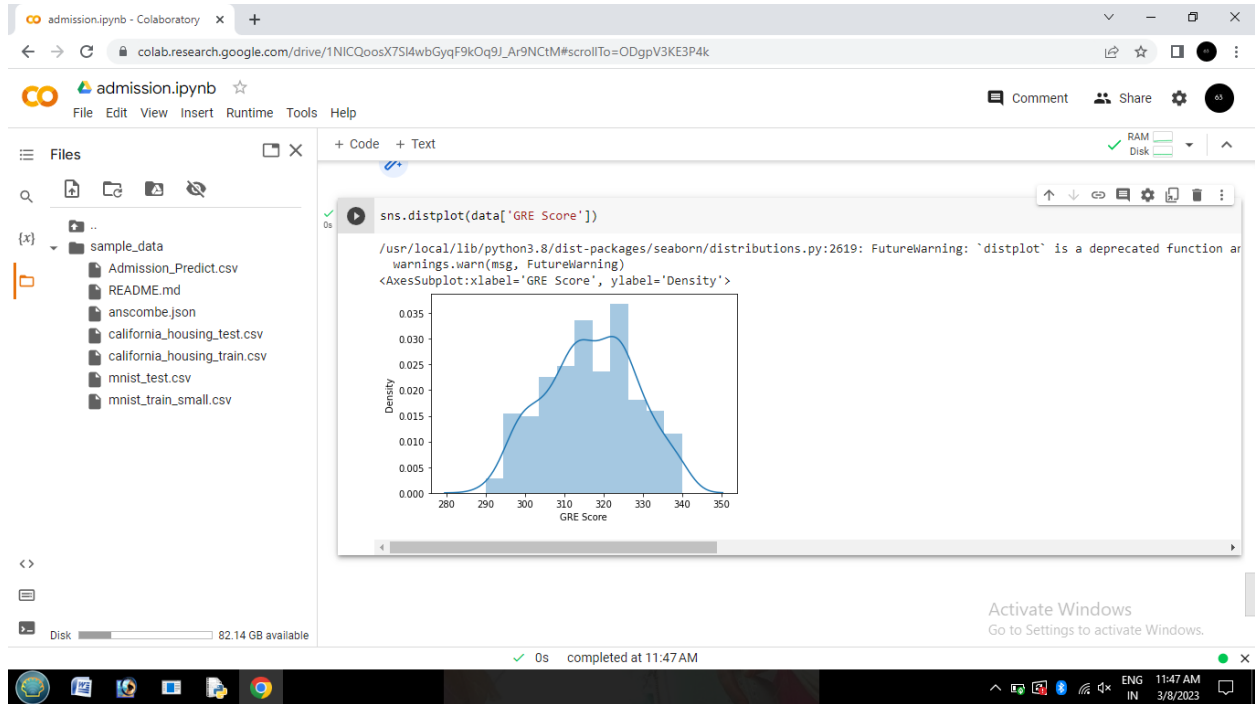
ML libraries such as



w.Keras
h.



3. RESULT



← → ↻ 🌐 localhost:5000 🔍 ☆ 📄 ⚙️ 👤 ⋮

📱 Apps 🖨️ HP Connected 🔄 Reverso 🌟 Sci-Hub: removing... 🌐 Google 📺 (1) GradeTrust: A Se... 📺 (1) Leach Protocol... 🚦 ACT Fibernet Portal... 📞 WhatsAppweb 📁 05_59_06project th... 🖼️ आयातित लोग (संय...

UNIVERSITY ADMISSION PREDICTION SYSTEM

Enter your details and get probability of your admission

Enter GRE Score

Enter TOEFL Score

Select University no

☐ 1

☒ 2

☐ 3

☐ 4

☐ 5

Enter SOP


Enter LOR

Enter CGPA

Research

☐ Research

☒ NO Research



SCREEN LAYOUT

File Edit View Insert Runtime Tools Help Cannot save changes

Files

- sample_data
 - Admission_Predict.csv
 - README.md
 - anscombe.json
 - california_housing_test.csv
 - california_housing_train.csv
 - mnist_test.csv
 - mnist_train_small.csv

Code

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
data=pd.read_csv('/content/sample_data/Admission_Predict.csv')
data
```

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	1	337	118	4	4.5	4.5	9.65	1	0.92
1	2	324	107	4	4.0	4.5	8.87	1	0.76
2	3	316	104	3	3.0	3.5	8.00	1	0.72
3	4	322	110	3	3.5	2.5	8.67	1	0.80
4	5	314	103	2	2.0	3.0	8.21	0	0.65
...
395	396	324	110	3	3.5	3.5	9.04	1	0.82
396	397	325	107	3	3.0	3.5	9.11	1	0.84
397	398	330	116	4	5.0	4.5	9.45	1	0.91
398	399	312	103	3	3.5	4.0	8.78	0	0.67

Activate Windows
Go to Settings to activate Windows.

completed at 11:04 AM

File Edit View Insert Runtime Tools Help Cannot save changes

Files

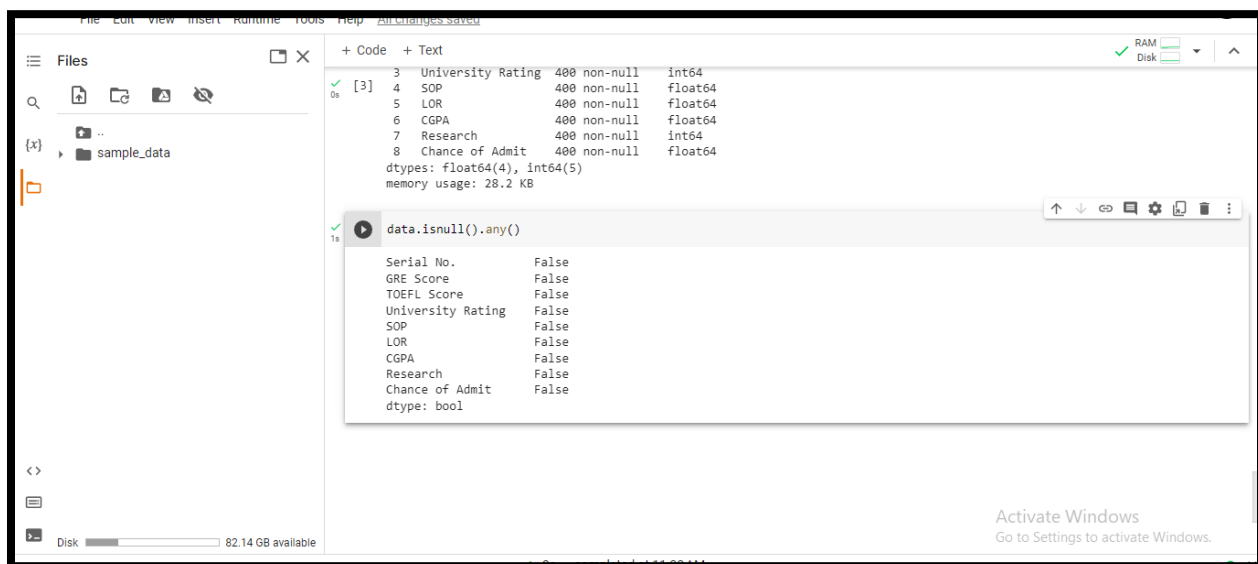
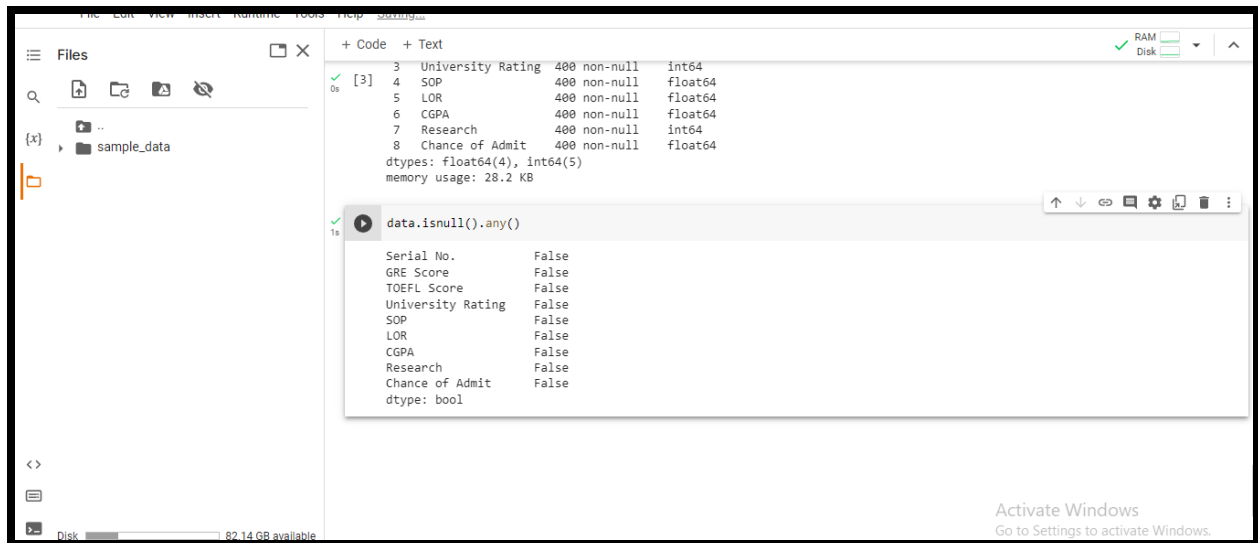
- sample_data

Code

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 9 columns):
#   Column              Non-Null Count  Dtype
---  ---
0   Serial No.          400 non-null   int64
1   GRE Score            400 non-null   int64
2   TOEFL Score          400 non-null   int64
3   University Rating    400 non-null   int64
4   SOP                  400 non-null   float64
5   LOR                  400 non-null   float64
6   CGPA                 400 non-null   float64
7   Research             400 non-null   int64
8   Chance of Admit      400 non-null   float64
dtypes: float64(4), int64(5)
memory usage: 28.2 KB
```

Activate Windows
Go to Settings to activate Windows.



File Edit View Insert Runtime Tools Help All changes saved

Files

- sample_data
 - Admission_Predict.csv
 - README.md
 - anscombe.json
 - california_housing_test.csv
 - california_housing_train.csv
 - mnist_test.csv
 - mnist_train_small.csv

Code + Text

```
Chance of Admit    False
dtype: bool
```

```
[5] data=data.rename(columns={'Chance of Admit':'Chance of Admit'})
```

```
data.describe()
```

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
count	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000
mean	200.500000	316.807500	107.410000	3.087500	3.400000	3.452500	8.598925	0.547500	0.724350
std	115.614301	11.473646	6.069514	1.143728	1.006869	0.898478	0.596317	0.498362	0.142609
min	1.000000	290.000000	92.000000	1.000000	1.000000	1.000000	6.800000	0.000000	0.340000
25%	100.750000	308.000000	103.000000	2.000000	2.500000	3.000000	8.170000	0.000000	0.640000
50%	200.500000	317.000000	107.000000	3.000000	3.500000	3.500000	8.610000	1.000000	0.730000
75%	300.250000	325.000000	112.000000	4.000000	4.000000	4.000000	9.062500	1.000000	0.830000
max	400.000000	340.000000	120.000000	5.000000	5.000000	5.000000	9.920000	1.000000	0.970000

Activate Windows
Go to Settings to activate Windows.

File Edit View Insert Runtime Tools Help All changes saved

Files

- sample_data
 - Admission_Predict.csv
 - README.md
 - anscombe.json
 - california_housing_test.csv
 - california_housing_train.csv
 - mnist_test.csv
 - mnist_train_small.csv

Code + Text

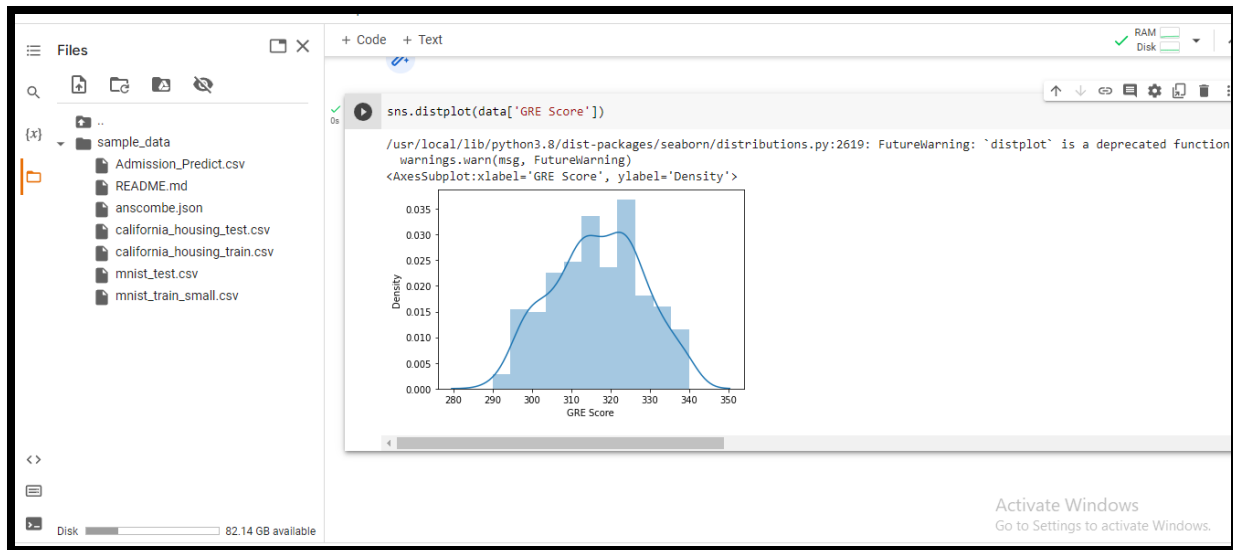
```
Chance of Admit    False
dtype: bool
```

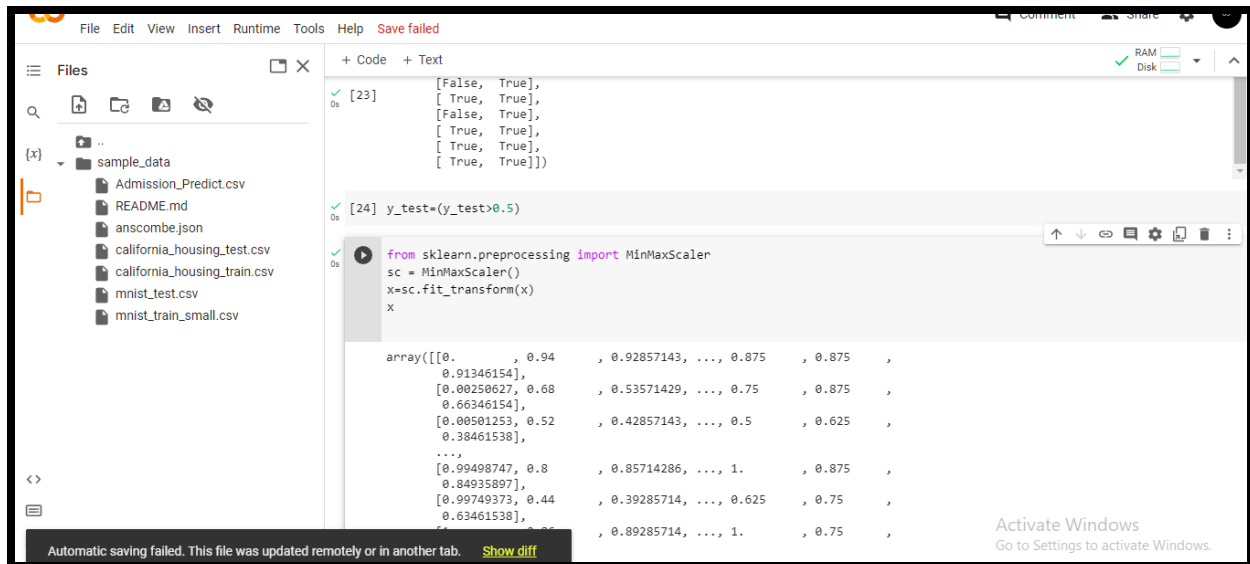
```
[5] data=data.rename(columns={'Chance of Admit':'Chance of Admit'})
```

```
data.describe()
```

	Serial No.	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
count	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000	400.000000
mean	200.500000	316.807500	107.410000	3.087500	3.400000	3.452500	8.598925	0.547500	0.724350
std	115.614301	11.473646	6.069514	1.143728	1.006869	0.898478	0.596317	0.498362	0.142609
min	1.000000	290.000000	92.000000	1.000000	1.000000	1.000000	6.800000	0.000000	0.340000
25%	100.750000	308.000000	103.000000	2.000000	2.500000	3.000000	8.170000	0.000000	0.640000
50%	200.500000	317.000000	107.000000	3.000000	3.500000	3.500000	8.610000	1.000000	0.730000
75%	300.250000	325.000000	112.000000	4.000000	4.000000	4.000000	9.062500	1.000000	0.830000
max	400.000000	340.000000	120.000000	5.000000	5.000000	5.000000	9.920000	1.000000	0.970000

Activate Windows
Go to Settings to activate Windows.





The screenshot shows the PyCharm IDE interface. The top toolbar includes buttons for 'File', 'Edit', 'View', 'Insert', 'Running', 'Tools', 'Help', and a 'Connect' dropdown. The main editor area displays a Python script. The first part of the script is a function `load_model` that attempts to load a model from 'model.h5'. It includes error handling for `IOError` with a message: `raise IOError(f'No file or directory found at {filepath_str}')`. Below this, an `OSError` is raised with the message: `OSError: No file or directory found at model.h5`. A 'SEARCH STACK OVERFLOW' button is visible. The second part of the script is a web application template with a `home` function that returns `render_template('Demo2.html')` and a `y_predict` function that processes input data and returns a prediction. The bottom right corner of the IDE has a watermark that says 'Activate Windows Go to Settings to activate Windows.'

PYCHARM

for html

The screenshot shows a web browser window with the title 'UNIVERSITY ADMISSION PR...'. The form contains several input fields: 'Enter QRE Score' with the value '120', 'Enter TOEFL Score' with the value '120', 'select university on:' with a dropdown menu showing '1', 'Enter SOP' with the value '90', 'Enter LOR' with the value '80', and 'Enter CGPA' with the value '90'. There are two radio buttons for 'Research': 'Research' (selected) and 'No Research'. At the bottom, there are 'Submit' and 'Reset' buttons. The bottom right corner of the browser window has a watermark that says 'Activate Windows Go to Settings to activate Windows.'

4. ADVANTAGES & DISADVANTAGES

Advantages:

- **Increased Efficiency:** Intelligent systems can quickly process large amounts of data, reducing the time and resources required for the admission process.
- **Objective and Unbiased:** An intelligent system can eliminate human bias from the admission process, making it more objective and fair.
- **Improved Accuracy:** Intelligent systems can analyze data and provide more accurate predictions and recommendations for admission.
- **Consistency:** With an intelligent system, all applicants are evaluated using the same criteria, ensuring consistency in the admission process.
- **Enhanced Security:** Intelligent systems can help prevent fraud and ensure the authenticity of application materials, improving the security of the admission process.

Disadvantages:

- Lack of Human Touch: An intelligent system may not be able to assess a candidate's non-academic qualities such as leadership potential, interpersonal skills, and emotional intelligence.
- Cost: Developing and implementing an intelligent admission system can be expensive, and may require ongoing maintenance and updates.
- Potential for Technical Issues: Any system can have technical glitches or errors, and an intelligent admission system is no exception. This could result in inaccurate assessments or even exclusion of certain applicants.
- Limited Scope: An intelligent system can only evaluate data that is available, and may not be able to take into account factors such as personal circumstances, potential, or extenuating circumstances that could affect an applicant's academic performance.
- Ethical Concerns: The use of an intelligent admission system raises ethical concerns about privacy, data protection, and discrimination, particularly if the system is not designed or implemented appropriately.
- It's important to note that any decision about whether or not to use an intelligent admission system should be carefully considered and weighed against these potential advantages and disadvantages.

5. APPLICATION

Once you've weighed up all the factors and carefully made your decision, it's time for the really fun part: applying.

Though this might seem obvious, ensure you take care over this. You don't want to miss out simply because you forgot to submit the required evidence or applied too late.

"Students should check entry requirements and deadlines before applying to make sure that they have the best possible chance of gaining a place on their chosen course," Berry confirms.

"If they are unsure whether their qualifications are acceptable, they might like to contact the admissions office or international office in their chosen institution to check before submitting a full application."

She emphasizes the importance of applying in good time: "Students should try to make an application as early as possible as this will give them plenty of time to make all the necessary arrangements for a move abroad, including organizing their finances, applying for scholarships and obtaining a student visa."

6. CONCLUSION

- In conclusion, admission prediction is an important task in the field of education that can help both students and universities make better-informed decisions about the college admissions process. By developing models that can accurately predict whether a student will be admitted to a particular school based on their input features, admission prediction can help students identify which schools they are most likely to be accepted to, and can assist universities in evaluating applicants more efficiently and objectively.
- However, it is important to use admission prediction models ethically and responsibly, and to ensure that they are not used to unfairly discriminate against certain groups of applicants. Additionally, admission prediction models should be constantly monitored and updated to ensure that they remain accurate and unbiased over time.
- Overall, admission prediction has the potential to improve the fairness and efficiency of the college admissions process, and can provide valuable insights into the factors that contribute to admission decisions. By continuing to develop and refine admission prediction models, we can help ensure that the college admissions process is as transparent and equitable as possible.

7. FUTURE ENHANCEMENT

- There are several future enhancements that can be made to admission prediction models to improve their accuracy and usefulness:
- Incorporating more data: Admission prediction models can be improved by incorporating additional data sources, such as social media activity, personal essays, and letters of recommendation. This can provide a more comprehensive view of the applicant and improve the accuracy of the model.
- Using more advanced machine learning algorithms: While logistic regression and decision trees are commonly used for admission prediction, more advanced machine learning algorithms such as random forests, gradient boosting, and deep learning can be applied to improve model accuracy.

8 . APPENDIX

step 1

```
import numpy as np
```

```
import pandas as pd
```

```
import matplotlib.pyplot as plt
```

```
import seaborn as sns
```

```
%matplotlib inline
```

```
data=pd.read_csv('/content/sample_data/Admission_Predict.csv')
```

```
data
```

step 2

```
data.info()
```

step 3

data.isnull().any()

step 4

data=data.rename(columns={'Chance of Admit ':'Chance of Admit'})

step 5

data.describe()

step 6

sns.distplot(data['GRE Score'])

step 7

sns.pairplot(data=data,hue='Research',markers=["^", "v"],palette='inferno')

step 8

```
sns.scatterplot(x='University Rating',y='CGPA',data=data,color='Red',s=100)
```

step 9

```
category = ['GRE Score','TOEFL Score','University Rating','SOP','LOR',  
'CGPA','Research','Chance of Admit']
```

```
color = ['yellowgreen','gold','lightskyblue','pink','red','purple','orange','gray']
```

```
start = True
```

```
for i in np.arange(4):
```

```
    fig = plt.figure(figsize=(14,8))
```

```
    plt.subplot2grid((4,2),(i,0))
```

```
    data[category[2*i]].hist(color=color[2*i],bins=10)
```

```
    plt.title(category[2*i])
```

```
    plt.subplot2grid((4,2),(i,1))
```

```
    data[category[2*i+1]].hist(color=color[2*i+1],bins=10)
```

```
    plt.title(category[2*i+1])
```

```
plt.subplots_adjust(hspace = 0.7, wspace = 0.2)
```

```
plt.show()
```

step 10

```
x=data.iloc[:,0:7].values
```

```
y=data.iloc[:,0:7].values
```

step 11

x

step12

y

step13

```
from sklearn.model_selection import train_test_split
```

```
x_train,x_test,y_train,y_test=train_test_split(x,y,  
text_size=0.30,random_state=101)
```

step 14

```
y_train=(y_train>0.5)
```

```
y_train
```

step 15

```
y_text=(y_text>0.5)
```

step 16

```
from sklearn.preprocessing import MinMaxScaler
```

```
sc=MinMaxScaler()
```

```
x=sc.fit_transform(x)
```

```
x
```

step 17

```
from sklearn.linear_model.logistic import LogisticRegression
```

```
cls =LogisticRegression(random_state =0)
```

```
lr=cls.fit(x_train, y_train)
```

```
y_pred =lr.predict(x_test)
```

```
y_pred
```

step 18

```
import tensorflow as tf
```

```
from tensorflow import keras
```

```
from keras import Sequential
```

```
from keras.layers import Dense
```

step 19

```
model=keras.Sequential()
```

```
model.add(Dense(7,activation = 'relu',input_dim=7))
```

```
model.add(Dense(7,activation='relu'))
```

```
model.add(Dense(1,activation='linear'))
```

```
model.summary()
```

step 20

```
model.summary()
```

```
model.compile(loss = 'binary_crossentropy', optimizer = 'adam', metrics =  
['accuracy'])
```

```
model.fit(x_train, y_train, batch_size = 20, epochs = 100)
```

```
from sklearn.metrics import accuracy_score
```

```
train_predictions = model.predict(x_train)
```

```
print(train_predictions)
```

```
train_acc = model.evaluate(x_train, y_train, verbose=0)[1]
```

```
print(train_acc)
```

```
test_acc = model.evaluate(x_test, y_test, verbose=0)[1]
```

```
print(test_acc)
```

```
print(classification_report(y_text, pred))
```



```
pred=model.predict(x_test)
```

```
pred = (pred>0.5)
```

```
pred
```

```
from sklearn.metrics import  
accuracy_score,recall_score,roc_auc_score,confusion_matrix
```

```
print("\nAccuracy score: %f" %(accuracy_score(y_test,y_pred) * 100))
```

```
print("Recall score : %f" %(recall_score(y_test,y_pred) * 100))
```

```
print("ROC score : %f\n" %(roc_auc_score(y_test,y_pred) * 100))
```

```
print(confusion_matrix(y_test,y_pred))
```

```
from sklearn.metrics import  
accuracy_score,recall_score,roc_auc_score,confusion_matrix
```

```
print(classification_report(y_train,pred))
```

```
from sklearn.metrics import  
accuracy_score,recall_score,roc_auc_score,confusion_matrix
```

```
print(classification_report(y_test,pred))
```

```
model.save('model.h5')
```

```
import numpy as np
```

```
from flask import Flask, request, jsonify, render_template
```

```
import pickle
```

```
app = Flask(__name__)
```

```
from tensorflow.keras.models import load_model
```

```
#model = pickle.load(open('University.pkl', 'rb'))
```

```
model = load_model('model.h5')
```

```
@app.route('/')
```

```
def home():
```

```
    return render_template('Demo2.html')
```

```
@app.route('/y_predict',methods=['POST'])
```

```
def y_predict():
```

```
min1=[290.0, 92.0, 1.0, 1.0, 6.8,0.0]
```

```
max1=[340.0, 120.0, 5.0, 5.0, 5.0, 9,92, 1.0]
```

```
k= [float(x) for x in request.form.values())
p[]
for i in range(7):
    l=(k[i]-min1[i])/(max1[i]-min1[i])
    p.append(1)
prediction = model.predict([p])
print(prediction)
output=prediction[0]
if(output==False):
    return render_template('nochance.html', prediction_text='You Dont have
a chance of getting')
else:
    return render_template('chance.html', prediction_text='You have a
chance of getting admission')
if __name__ == "__main__":
app.run(debug=False)
```

