1. INTRODUCTION

1.1 Overview

The world markets are developing rapidly and continuously looking for the best knowledge and experience among people. Young workers who want to stand out in their jobs are always looking for higher degrees that can help them in improving their skills and knowledge. As a result, the number of students applying for graduate studies has increased in the last decade. This fact has motivated us to study the grades of students and the possibility of admission for master’s programs that can help universities in predicting the possibility of accepting master’s students submitting each year and provide the needed resources.

The dataset presented in this paper is related to educational domain. Admission is a dataset with 400 rows that contains 7 different independent variables which are: Graduate Record Exam1 (GRE) score. Test of English as a Foreigner Language2 (TOEFL) score. University Rating (Uni Rating) that indicates the Bachelor University ranking among the other universities. Statement of purpose (SOP) which is a document written to show the candidate's life, ambitious and the motivations for the chosen degree/ university. Letter of Recommendation Strength (LOR) which verifies the candidate professional experience, builds credibility, boosts confidence and ensures your competency. Undergraduate GPA (CGPA) out of 10. Research Experience that can support the application, such as publishing research papers in conferences, working as research assistant with university professor. One dependent variable can be predicted which is chance of admission that is according to the input given will be ranging from 0 to 1. We are developing four Regression Models which are multiple Linear Regression, Random forest Regression, Multiple Linear Regression using Dimensionality reduction and Random forest Regression using Dimensionality reduction to finding the accuracy of those models. Out of those we use high accuracy models.

So, we believe that a predictive model generated using all the past data can be a useful resource to predict the outcome for the applicants.

In the Guided Project, our goal is to predict the outcome of applications that are filed by many students every year.

1.2 Purpose

Our project aims that this prediction algorithm could be a useful resource for the students to easily process the admission to their preferred university .

In order to predict the case status of the applicants, we will be feeding the model with the dataset which contains the required fields by which the machine can classify the case status as eligible or ineligible.

2.LITERATURE SURVEY

2.1 Existing Problem

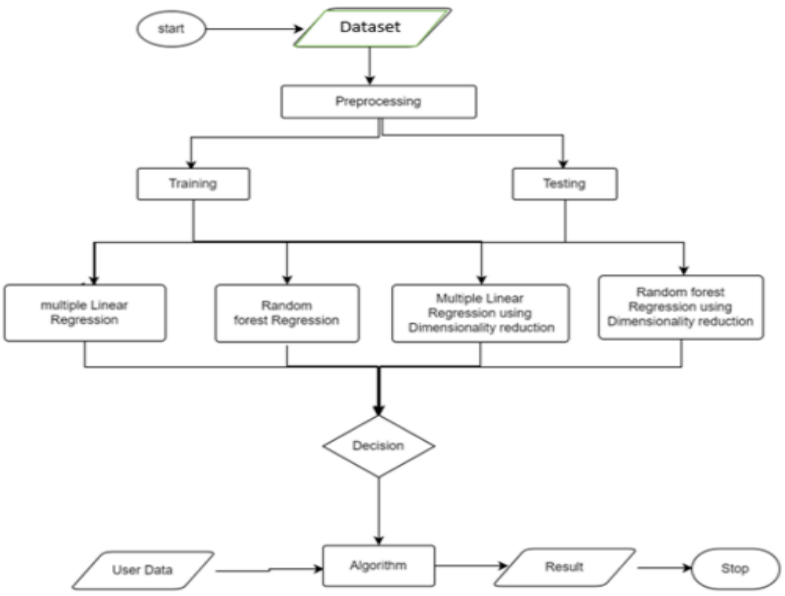
In India every year lacks of students getting the graduation degree and willing to join post-graduation in other countries. Newly graduate students usually are not knowledgeable of the requirements and the procedures of the postgraduate admission and might spent a considerable amount of money to get advice from consultancy organizations to help them identify their admission chances. Human consultant and calculations might be bias and inaccurate.

2.2 Proposed Solution

This paper helps on predicting the eligibility of Indian students getting admission in best university based on their Test attributes like GRE, TOEFL, LOR, CGPA etc. according to their scores the possibilities of chance of admit is calculated.

3.THEORITICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware / Software designing

• IBM Watson Studio

Watson Studio provides you with the environment and tools to solve your business problems by collaboratively working with data. It provides a suite of tools for data scientists, application developers and subject matter experts, allowing them to collaboratively connect to data, wrangle that data and use it to build, train and deploy models at scale. Successful AI projects require a combination of algorithms + data + team, and a very powerful compute infrastructure.

• IBM Watson Machine Learning

IBM Watson Machine Learning is a full-service IBM Cloud offering that makes it easy for developers and data scientists to work together to integrate predictive capabilities with their applications. The Machine Learning service is a set of REST APIs that you can call from any programming language to develop applications that make smarter decisions, solve tough problems, and improve user outcomes.

• IBM Cloud Object Storage

4 EXPERIMENTAL INVESTIGATIONS

The dataset [5] presented in this paper is related to educational domain. Admission is a dataset with 500 rows that contains 7 different independent variables which are:

• Graduate Record Exam1 (GRE) score. The score will be out of 340 points.

• Test of English as a Foreigner Language2 (TOEFL) score, which will be out of 120 points.

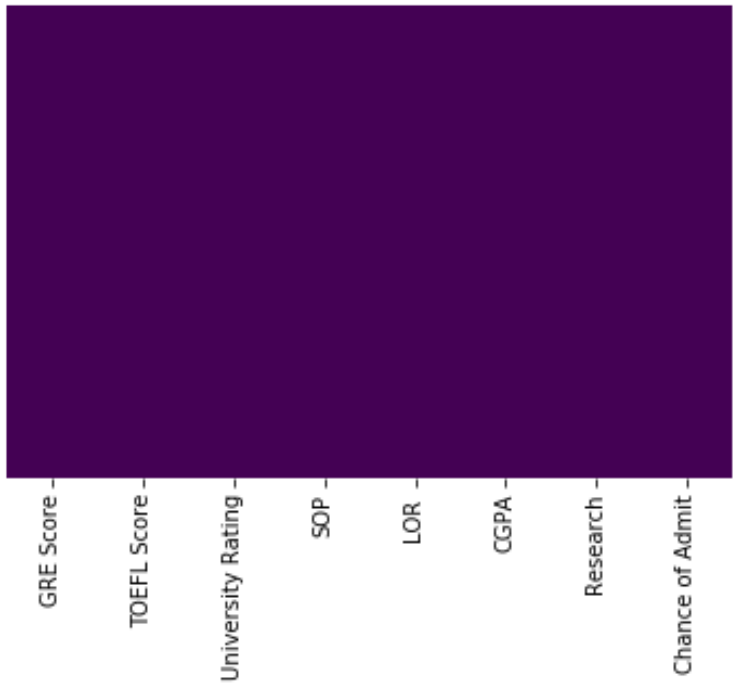
• University Rating (Uni.Rating) that indicates the Bachelor University ranking among the other universities. The score will be out of 5

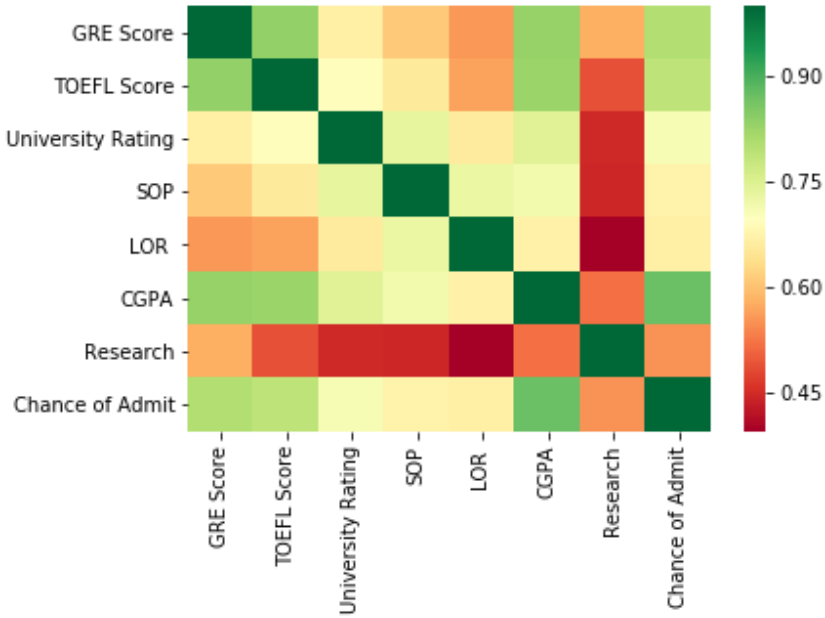
• Statement of purpose (SOP) which is a document written to show the candidate's life, ambitious and the motivations for the chosen degree/ university. The score will be out of 5 points.

• Letter of Recommendation Strength (LOR) which verifies the candidate professional experience, builds credibility, boosts confidence and ensures your competency. The score is out of 5 points

• Undergraduate GPA (CGPA) out of 10

• Research Experience that can support the application, such as publishing research papers in conferences, working as research assistant with university professor (either 0 or 1). One dependent variable can be predicted which is chance of admission that is according to the input given will be ranging from 0 to 1.





As can be seen from the Fig. the dataset is highly imbalanced, we apply some pre-processing techniques to create a dataset which has more relevant data to generate a model leaving all the noise in the data. We performed exploratory data analysis to get some facts which the data provides us and basing on them. we considered the relevance of relationship among the features and accordingly discarded few features and created some to remove the redundant information. We observed that features 'LON' and 'LAT' have missing values nearly 100000 points hence we remove both the features entirely. Also, we transformed few features into new features. The features of our final dataset after transformation is: CASE_STATUS, FULL_TIME_POSITION, PREVAILING_WAGE, YEAR, SOC_NAME.

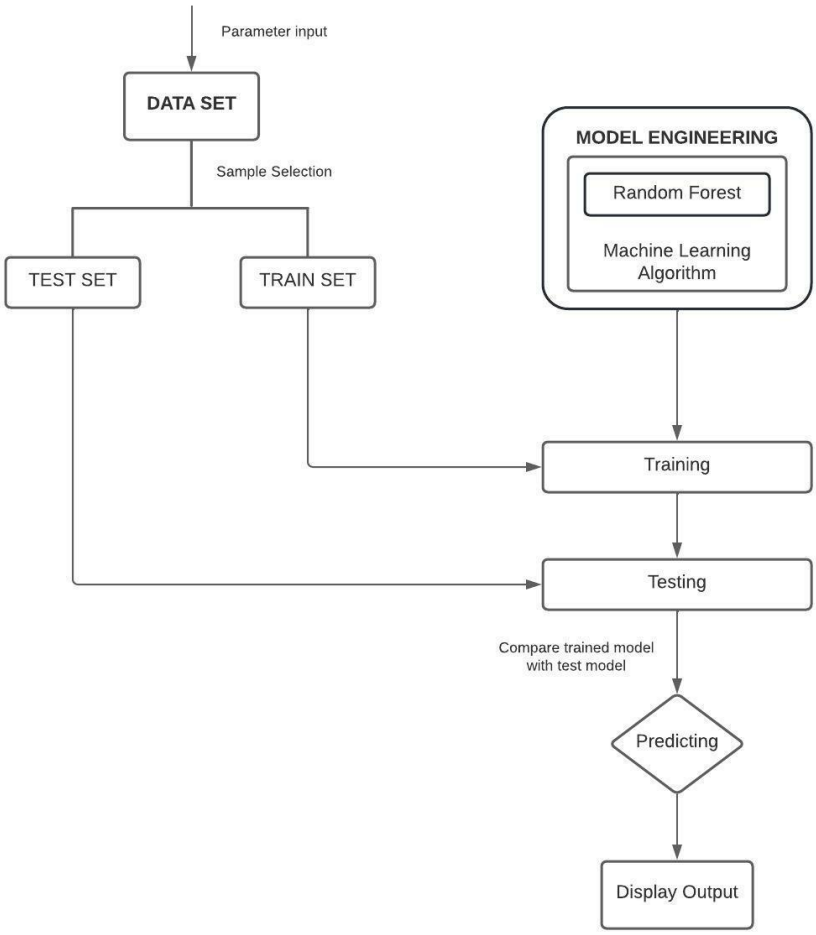
After all the pre-processing steps we performed on our dataset to get the final transformed dataset we split the data into train and test. For the prediction task we use these 3 classifiers. They are Gaussian Naïve Bayes Classifier, Random-Forest classifier and XG-Boost.

1. Naïve Bayes: Naive Bayes is a simple and interpretable model which assumes all features are conditionally independent given labels and are in gaussian distribution. The function fit was used to fit the learning model on the data and the function score was used to find out the F-score of this algorithm and to assess its performance.

2. Random-Forest: It is an ensemble technique which uses bagging technique. It uses number of meta-classifiers on various sub samples of the dataset and then averages the prediction to improve the final predictive outcome. This classifier can also control overfitting by proper parameter tuning.

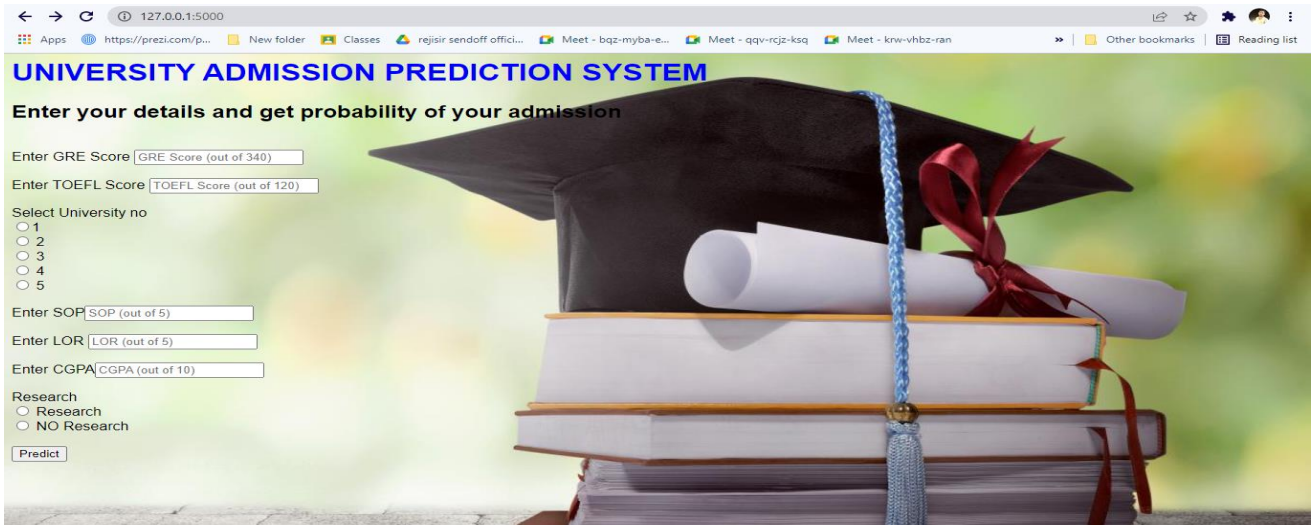
3. XG-Boost: XGBoost or Extreme Gradient Boost algorithm is an ensemble method. It uses 'Bagging and Boosting' techniques. In Baggi ng technique, trees are grown to their maximum extent and Boosting techniques uses trees with fewer splits. On aggregation of the two models, the final model gives us the outcome with less MSE (Mean Squared Error).

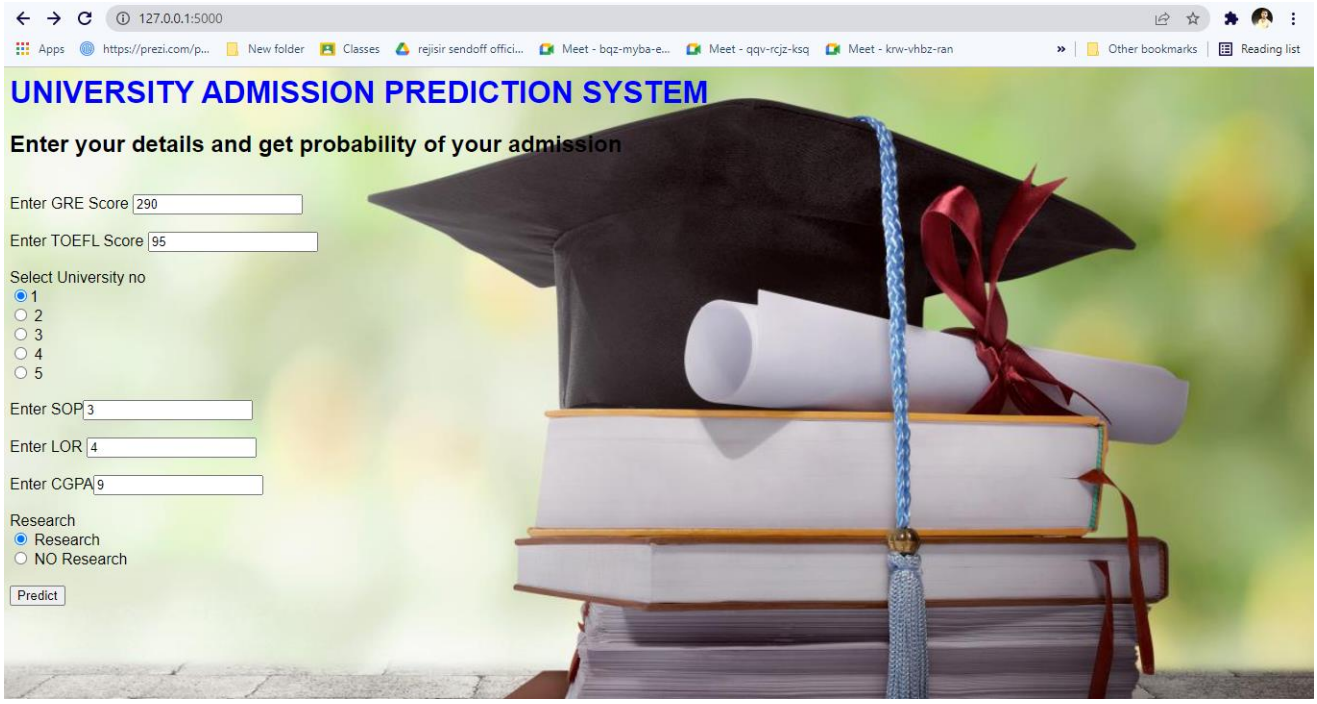
5 FLOWCHART

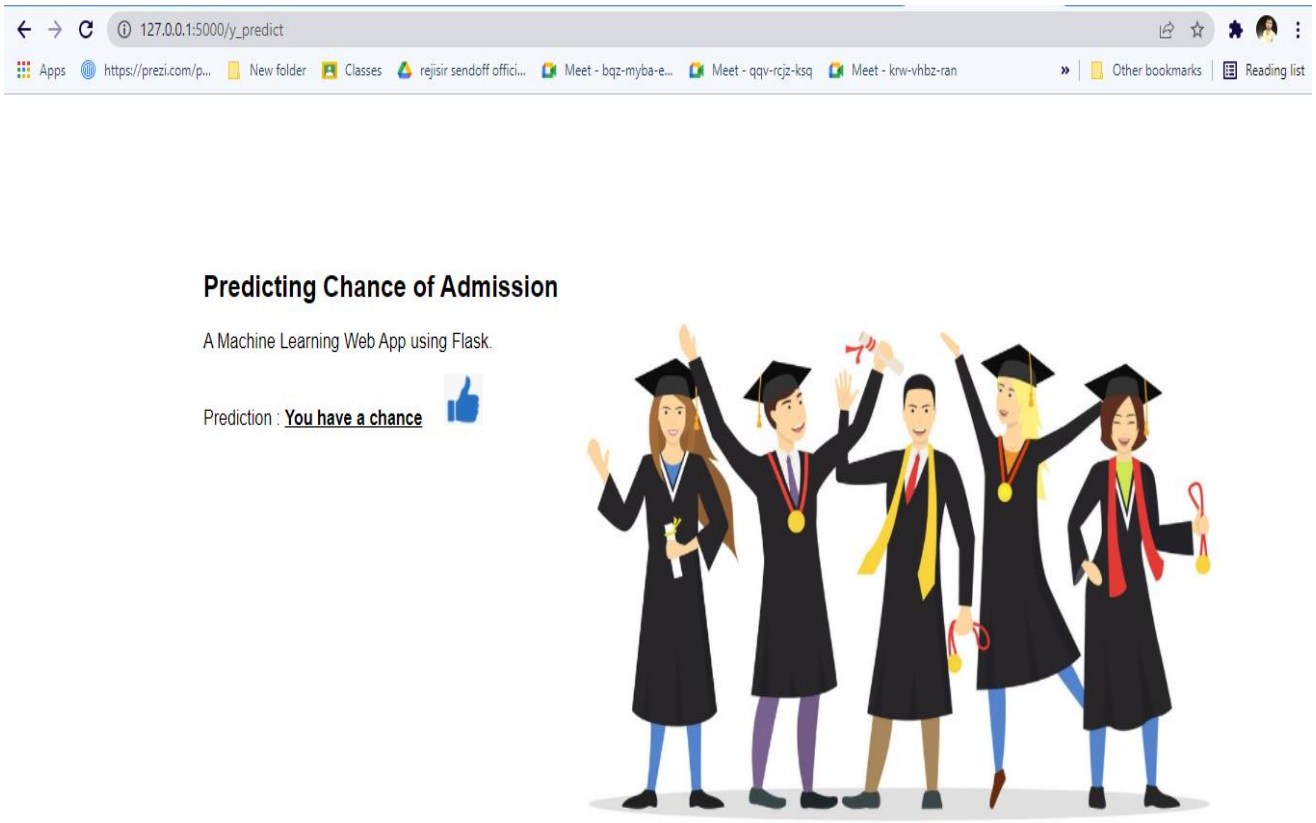


6. RESULT

The final result of the project is the predict the certification status of the University admission Prediction applications.




127.0.0.1:5000
Apps
https://prezi.com/p...
New folder
Classes
rejisir sendoff offici...
Meet - bqz-myba-e...
Meet - qqv-rcjz-ksq
Meet - krw-vhbz-ran
Other bookmarks
Reading list
UNIVERSITY ADMISSION PREDICTION SYSTEM
Enter your details and get probability of your admission
Enter GRE Score 290
Enter TOEFL Score 95
Select University no
1
2
3
4
5
Enter SOP 3
Enter LOR 4
Enter CGPA 9
Research
Research
NO Research
Predict


127.0.0.1:5000/y_predict
Apps
https://prezi.com/p...
New folder
Classes
rejisir sendoff offici...
Meet - bqz-myba-e...
Meet - qqv-rcjz-ksq
Meet - krw-vhbz-ran
Other bookmarks
Reading list
Predicting Chance of Admission
A Machine Learning Web App using Flask.
Prediction : You have a chance

7. ADVANTAGES & DISADVANTAGES

The main advantage of this proposed application is reduction of time. One can infer from these applications to know the case status of the application as eligible or ineligible. we don't need to check all data to know accepted or rejected.it take necessary data and make prediction. Hence it helps to reduce huge amount time for checking each and every application.

Disadvantages

- Need more datasets, to increase the accuracy of the algorithms.
- A large amount of data is used in the process of training and learning, so the use of data should be of good quality, unbiased.
- The proposed application can only be used by students or faculties for prediction.
- The proposed application is Web-based, hence cannot be used in Mobile devices.
- The result of the application depends upon the accuracy of the algorithms

8. APPLICATIONS

University admissions requires a lot of time and efforts. With our prediction application we round up the favorable universities by thoroughly analyzing the existing admission criteria. The application provides high accuracy and reliability. Therefor the applicant can take a lot out of theirplates.

9.CONCLUSION

In this work, Gaussian Naive Bayes, Random Forest Classifier and XGBoost Classifier were considered for determining the status of H1-B visa applications. Random Forest Classifier performed the best in terms of accuracy, precision and F1 score over others. We achieved a best of 86.808% classification accuracy. Naïve Bayes classifier has performance of 51.92% accuracy. This leads to conclusion that how much important is feature selection and feature transformation is. Our results showed that the most predictive features are EMPLOYER SUCCESS RATE and PREVAILING WAGE. One can infer from these results that the chance of being certified increases with the amount of wage and how successful your sponsor was in the previous H1B applications.

10. FUTURE SCOPE

Supplemental data concerning the Standard Occupational Classification (SOC) can be gathered and used in coordination with this data set to obtain a more comprehensive analysis of how the H-1B Visa selection process works.

By using the wage evaluations and ranges under SOC, the wage attribute in this data set can be correctly put in to a range of salaries which can then be used to classify the visa petitions based on occupation roles rather than location wise. In addition, other classification algorithms other than the discriminative models can be experimented with this testbed and their performances can also be analyzed.

11. BIBILOGRAPHY

Import Necessary Libraries

Let us import necessary libraries to get started!

In [2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
```

The Data

Let's start by reading in the Admission_Predict.csv file into a pandas dataframe.

In [77]:
```
#read_csv is a pandas function to read csv files
data = pd.read_csv('Admission_Predict.csv')
```

In [78]:
```
#head() method is used to return top n (5 by default) rows of a DataFrame or series.
data.head()
```

Out[78]:

	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

In [79]:
```
#let us drop Serial No. Column as it is not required for prediction
data.drop(["Serial No."],axis=1,inplace=True)
data.head()
```

Out[79]:

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	337	118	4	4.5	4.5	9.65	1	0.92
1	324	107	4	4.0	4.5	8.87	1	0.76
2	316	104	3	3.0	3.5	8.00	1	0.72
3	322	110	3	3.5	2.5	8.67	1	0.80
4	314	103	2	2.0	3.0	8.21	0	0.65

#describe() method computes a summary of statistics like count, mean, standard deviation, min, max and quartile values.

In [80]:
```
data.describe()
```

In [80]:
```
data.describe()
```

Out[80]:

	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
mean	316.807500	107.410000	3.087500	3.400000	3.452500	8.598925	0.547500	0.724350
std	11.473646	6.069514	1.143728	1.006869	0.898478	0.596317	0.498362	0.142609
min	290.000000	92.000000	1.000000	1.000000	1.000000	6.800000	0.000000	0.340000
25%	308.000000	103.000000	2.000000	2.500000	3.000000	8.170000	0.000000	0.640000
50%	317.000000	107.000000	3.000000	3.500000	3.500000	8.610000	1.000000	0.730000
75%	325.000000	112.000000	4.000000	4.000000	4.000000	9.062500	1.000000	0.830000
max	340.000000	120.000000	5.000000	5.000000	5.000000	9.920000	1.000000	0.970000

```
From the data we infer that there are only decimal values and no categorical values
```

In [81]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 400 entries, 0 to 399
Data columns (total 8 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   GRE Score          400 non-null    int64
 1   TOEFL Score        400 non-null    int64
 2   University Rating  400 non-null    int64
 3   SOP                400 non-null    float64
 4   LOR                400 non-null    float64
 5   CGPA               400 non-null    float64
 6   Research           400 non-null    int64
 7   Chance of Admit    400 non-null    float64
dtypes: float64(4), int64(4)
memory usage: 25.1 KB
```

In [82]:
```
#Let us rename the column Chance of Admit because it has trainling space
data=data.rename(columns = {'Chance of Admit ':'Chance of Admit'})
```

Exploratory Data Analysis

Missing Data

We can use seaborn to create a simple heatmap to see where we have missing data!

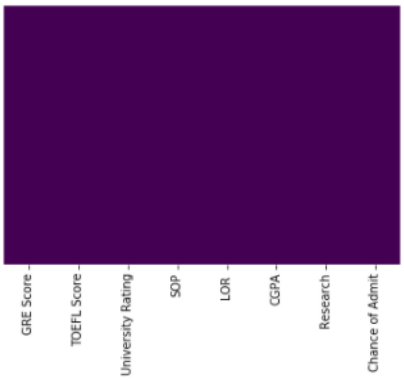
In [101]:
```
data.isnull().any()
```

Out[101]:
```
GRE Score            False
TOEFL Score          False
University Rating    False
SOP                  False
LOR                  False
CGPA                 False
Research             False
Chance of Admit      False
dtype: bool
```

Heatmap:It is way of representing the data in 2-D form.It gives coloured visual summary of the data

In [102]:
```
sns.heatmap(data.isnull(),yticklabels=False,cbar=False,cmap='viridis')
```

Out[102]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2b6e1bb0408>
```




```
From the heatmap,we see that there are no missing values in the dataset
```

In [103]:
```
data.corr()
```

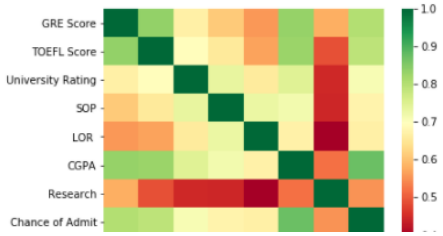
In [103]:
```
data.corr()
```

Out[103]:

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
GRE Score	1.000000	0.835977	0.668976	0.612831	0.557555	0.833060	0.580391	0.802610
TOEFL Score	0.835977	1.000000	0.695590	0.657981	0.567721	0.828417	0.489858	0.791594
University Rating	0.668976	0.695590	1.000000	0.734523	0.660123	0.746479	0.447783	0.711250
SOP	0.612831	0.657981	0.734523	1.000000	0.729593	0.718144	0.444029	0.675732
LOR	0.557555	0.567721	0.660123	0.729593	1.000000	0.670211	0.396859	0.669889
CGPA	0.833060	0.828417	0.746479	0.718144	0.670211	1.000000	0.521654	0.873289
Research	0.580391	0.489858	0.447783	0.444029	0.396859	0.521654	1.000000	0.553202
Chance of Admit	0.802610	0.791594	0.711250	0.675732	0.669889	0.873289	0.553202	1.000000

In [104]:
```
corr = data.corr()
sns.heatmap(corr,xticklabels=corr.columns.values,
            yticklabels=corr.columns.values,cmap="RdYlGn")
```

Out[104]: <matplotlib.axes._subplots.AxesSubplot at 0x2b6e1c0d588>




```
We see that the output variable "Chance of Admit" depends on CGPA,GRE,TOEFEL.The columns SOP,LOR and Reserach have less
lepact on university admission.
```

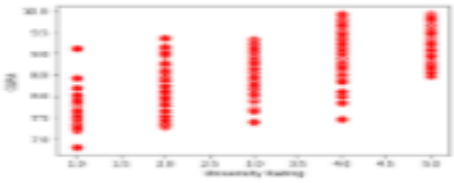
In [105]:
```
sns.pairplot(data=data,hue='Research',markers=["^", "v"],palette='inferno')
```

Out[105]: <seaborn.axisgrid.PairGrid at 0x2b6e1c181c8>

Pair plot usually gives pair wise relationships of the colums in the dataset
From the above pairplot we infer that
1.GRE score TOEFL score and CGPA all are linearly related to each other

In [305]:
```
sns.scatterplot(x='University Rating',y='CGPA',data=data,color='Red', s=300)
```

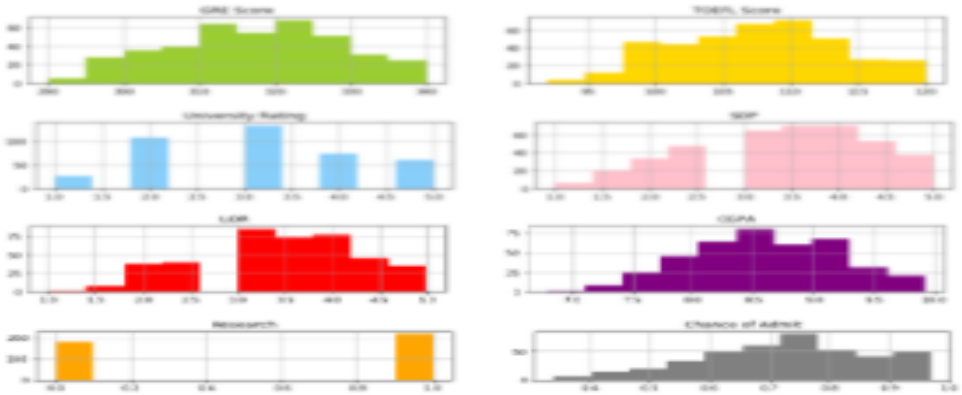
Out[305]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2bee99feec8>
```



From the above scatter plot we infer that as the CGPA increases the university ratings increases

In [307]:
```
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)
```



In [308]:
```
print('Mean CGPA Score is : ',int(data['CGPA'].mean()))
print('Mean GRE Score is : ',int(data['GRE Score'].mean()))
print('Mean TOEFL Score is : ',int(data['TOEFL Score'].mean()))
```

```
Mean CGPA Score is : 8
Mean GRE Score is : 316
Mean TOEFL Score is : 107
```

The chance of admission is high if the aspirant score more than the above mean values

```
category = ['GRE Score','TOEFL Score','University Rating','SOP','LOR ','CGPA','Research','Chance of Admit']
for i,col in enumerate(category):
    x=data[col].unique()
    g = data.groupby(by=col)[col].count().sort_values(ascending=False)
    r=range(len(x),0)
    print(g.head())
    plt.figure(figsize=(5,4))
    plt.bar(r,g.head())
    plt.xticks(r, g.index)
```

Machine Learning

1.Let's start by splitting the data into dependent and independent variable

In [310]:
```
data.head()
```

Out[310]:

	GRE Score	TOEFL Score	University Rating	SOP	LOR	CGPA	Research	Chance of Admit
0	337	118	4	4.5	4.5	9.65	1	0.92
1	324	107	4	4.0	4.5	8.87	1	0.76
2	316	104	3	3.0	3.5	8.00	1	0.72
3	322	110	3	3.5	2.5	8.67	1	0.80
4	314	103	2	2.0	3.0	8.21	0	0.65

In [311]:
```
x=data.iloc[:,0:7].values
```

Out[311]:
```
array([[337. , 118. ,  4. , ...,  4.5 ,  9.65,  1. ],
       [324. , 107. ,  4. , ...,  4.5 ,  8.87,  1. ],
       [316. , 104. ,  3. , ...,  4.5 ,  8. ,  1. ],
       ...,
       [330. , 116. ,  4. , ...,  5.5 ,  9.45,  1. ],
       [312. , 103. ,  3. , ...,  4. ,  8.78,  0. ],
       [333. , 117. ,  4. , ...,  4. ,  9.66,  1. ]])
```

In [312]:
```
y=data.iloc[:,7:].values
```

Out[312]:
```
array([[0.92],
       [0.76],
       [0.72],
       [0.8 ],
       [0.65],
       [0.9 ],
       [0.75],
       [0.68],
       [0.5 ],
       [0.45],
       [0.52],
       [0.84],
       [0.78],
       [0.62],
       [0.61],
       [0.54],
       [0.66],
       [0.65],
       [0.63],
```

2.Data Normalisation

There is huge disparity between the x values so we let us use feature scaling. Feature scaling is a method used to normalize the range of independent variables or features of data.

In [114]:
```
from sklearn.preprocessing import MinMaxScaler
sc = MinMaxScaler()
x=sc.fit_transform(x)
```

Out[114]:
```
array([[0.94        , 0.9285714, 0.7       , ..., 0.875     , 0.9149651,
        1.        ],
       [0.88        , 0.5357143, 0.7       , ..., 0.875     , 0.8437651,
        1.        ],
       [0.52        , 0.4285714, 0.5       , ..., 0.625     , 0.8858168,
        1.        ],
       ...,
       [0.8         , 0.8571429, 0.7       , ..., 0.875     , 0.9795897,
        1.        ],
       [0.44        , 0.8928571, 0.5       , ..., 0.75      , 0.6858168,
        0.        ],
       [0.88        , 0.8928571, 0.7       , ..., 0.75      , 0.9168667,
        1.        ]])
```

3.Splitting our data into a training set and test set .

In [116]:
```
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x,y, test_size=0.30,random_state=101)
```

Let us convert it into classification problem

chance of admit>0.5 as true chance of admit<0.5 as false

In [115]:
```
y_train=(y_train>0.5)
```

Out[115]:
```
array([[ True],
       [ True],
       [ True],
       [ True],
       [False],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [ True],
       [False],
       [False],
       [ True],
       [ True],
```

In [116]:
```
y_test=(y_test>0.5)
```

In [117]:
```
from sklearn.linear_model.logistic import LogisticRegression
clr = LogisticRegression(random_state =0)
```

```
C:\Users\hubski\anaconda3\lib\site-packages\sklearn\utils\validation.py:585: DataConversionWarning: A column-vector y was passe
d when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

In [118]:
```
y_pred =lr.predict(x_test)
```

Out[118]:
```
array([ True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
       False,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True,  True,  True,  True,  True, False,  True,
        True,  True,  True,  True,  True,  True,  True,  True,  True,
        True,  True,  True])
```

In [119]:
```
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))
```

```
Accuracy score: [illegible]
Recall score : [illegible]
MSE score : [illegible]

[[  1  11]
 [  1 [illegible]]]
```

Save the model to reuse it again

In [120]:
```
import pickle
pickle.dump(lr,open('university.pkl','wb'))
```

In [ ]:

In [ ]:

Logistic Regression has a good Accuracy score

In [ ]:

Linear Regression

In [54]:
```
from sklearn.linear_model import LinearRegression
mr=LinearRegression()
mr.fit(x_train,y_train)
y_pred_mr=mr.predict(x_test)
```

In [55]:
```
from sklearn.metrics import r2_score
```

Out[55]: [illegible]

In [24]:
```
#import library for random forest regressor
from sklearn.ensemble import RandomForestRegressor
rf=RandomForestRegressor(n_estimators=1000,random_state=0)
#n_estimators no of decision trees
```

```
C:\Users\hulsel\anaconda3\lib\site-packages\ipykernel_launcher.py:5: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
```

Out[24]:
```
RandomForestRegressor(bootstrap=True, ccp_alpha=0.0, criterion='mse',
                      max_depth=None, max_features='auto', max_leaf_nodes=None,
                      max_samples=None, min_impurity_decrease=0.0,
                      min_impurity_split=None, min_samples_leaf=1,
                      min_samples_split=2, min_weight_fraction_leaf=0.0,
                      n_estimators=1000, n_jobs=None, oob_score=False,
                      random_state=0, verbose=0, warm_start=False)
```

In [24]:
```
y_pred=rf.predict(x_test)
```

Out[24]: array([0.5[illegible], 0.6[illegible], 0.6[illegible], 0.6[illegible], 0.6[illegible]])

In [25]:
```
y_test[0:5]
```

Out[25]:
```
array([[0.52],
       [0.54],
       [0.72],
       [0.75],
       [0.76]])
```

In [26]:
```
from sklearn.metrics import r2_score
```

Out[26]: [illegible]

In [27]:
```
from sklearn.neighbors import KNeighborsRegressor
knn=KNeighborsRegressor(n_neighbors=25,p=1)
#n_estimators no of decision trees
```

Out[27]:
```
KNeighborsRegressor(algorithm='auto', leaf_size=30, metric='minkowski',
                    metric_params=None, n_jobs=None, n_neighbors=25, p=1,
                    weights='uniform')
```

In [28]:
```
y_pred=rf.predict(x_test)
```

Out[28]: array([0.5[illegible], 0.6[illegible], 0.6[illegible], 0.6[illegible], 0.6[illegible]])

In [29]:
```
y_test[0:5]
```

Out[29]:
```
array([[0.52],
       [0.54],
       [0.72],
       [0.75],
       [0.78]])
```

In [30]:
```
from sklearn.metrics import r2_score
```

Out[30]: 0.7001[illegible]

svm

In [31]:
```
from sklearn.svm import SVR
svr=SVR()
#n_estimators no of decision trees
```

```
C:\Users\hulsel\anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning: A column-vector y was passe
d when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

Out[31]:
```
SVR(C=1.0, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma='scale',
    kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

In [32]:
```
y_pred=rf.predict(x_test)
from sklearn.metrics import r2_score
```

Out[32]: 0.7001[illegible]

In [33]:
```
from sklearn.svm import SVR
clf = SVR()
clf.fit(x_train, y_train)
predictions=clf.predict(x_test)
predictions=clf.predict(x_test)
from sklearn.metrics import r2_score
R2=r2_score(y_test,predictions)
```

```
R-square- 67.55
```

```
C:\Users\hulsel\anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning: A column-vector y was passe
d when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

In [34]:
```
from sklearn.svm import SVR
clf = SVR()
clf.fit(x_train, y_train)
predictions=clf.predict(x_test)
predictions=clf.predict(x_test)
from sklearn.metrics import r2_score
R2=r2_score(y_test,predictions)
```

```
R-square- 67.55
```

```
C:\Users\hulsel\anaconda3\lib\site-packages\sklearn\utils\validation.py:761: DataConversionWarning: A column-vector y was passe
d when a 1d array was expected. Please change the shape of y to (n_samples, ), for example using ravel().
  y = column_or_1d(y, warn=True)
```

In [35]:
```
from sklearn.ensemble import RandomForestRegressor
reg_rf = RandomForestRegressor()
reg_rf.fit(x_train, y_train)
```

```
C:\Users\hulsel\anaconda3\lib\site-packages\ipykernel_launcher.py:4: DataConversionWarning: A column-vector y was passed when a
1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [36]:
```
from sklearn import metrics
print('MAE:', metrics.mean_absolute_error(y_test, y_pred))
print('MSE:', metrics.mean_squared_error(y_test, y_pred))
```

```
MAE: 0.05[illegible]
MSE: 0.00[illegible]
RMSE: 0.0[illegible]
```

In [38]:
```
metrics.r2_score(y_test, y_pred)
```

Out[38]: 0.9[illegible]

In [ ]:
```
```