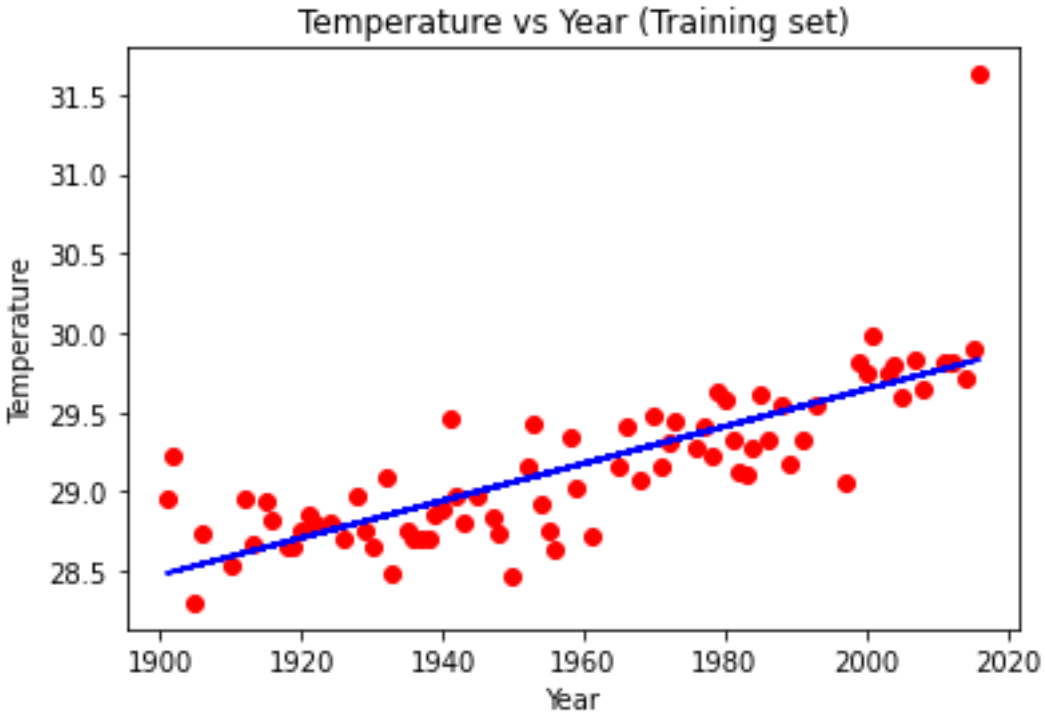
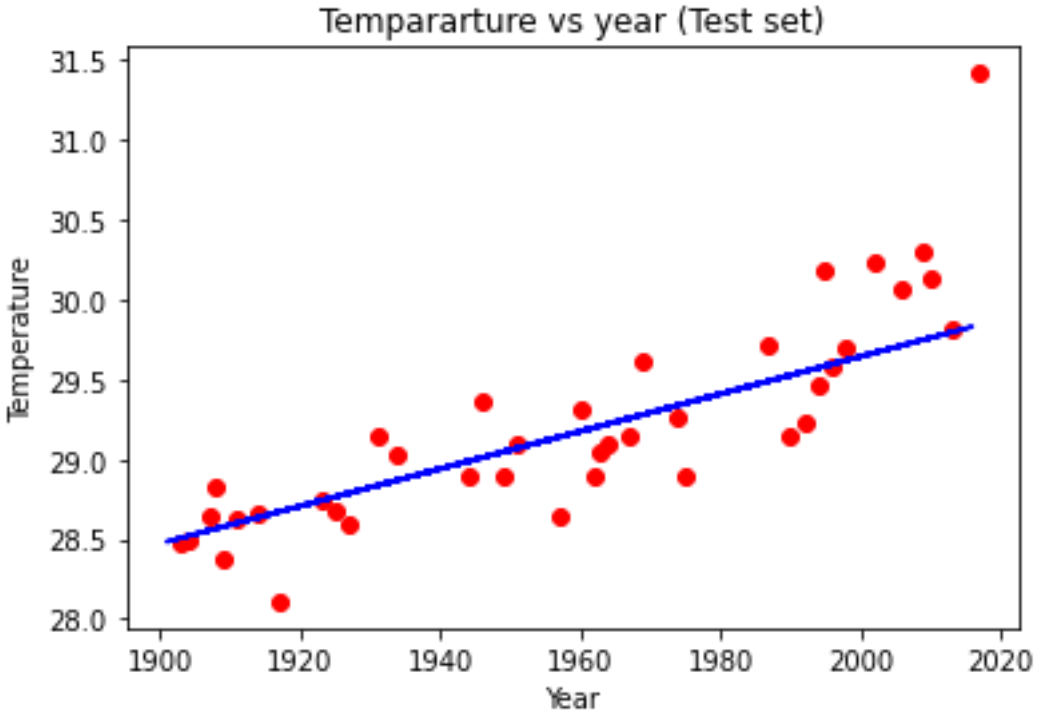
Harshal Gaikwad
T2217

Experiment No.01
CODE & OUTPUT

```
# Importing the libraries
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('/home/lenovo/Desktop/temperatures.csv')
X=dataset[["YEAR"]]
y=dataset[["ANNUAL"]]

# Splitting the dataset into the Training set and Test set
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 1/3, random_state = 0)

# Training the Simple Linear Regression model on the Training set
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)

# Predicting the Test set results
y_pred = regressor.predict(X_test)

# Visualising the Training set results
plt.scatter(X_train, y_train, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Temperature vs Year (Training set)')
plt.xlabel('Year')
plt.ylabel('Temperature')
plt.show()

# Visualising the Test set results
plt.scatter(X_test, y_test, color = 'red')
plt.plot(X_train, regressor.predict(X_train), color = 'blue')
plt.title('Tempararture vs year (Test set)')
plt.xlabel('Year')
plt.ylabel('Temperature')
plt.show()
```

Harshal Gaikwad
T2217





Harshal Gaikwad
T2217

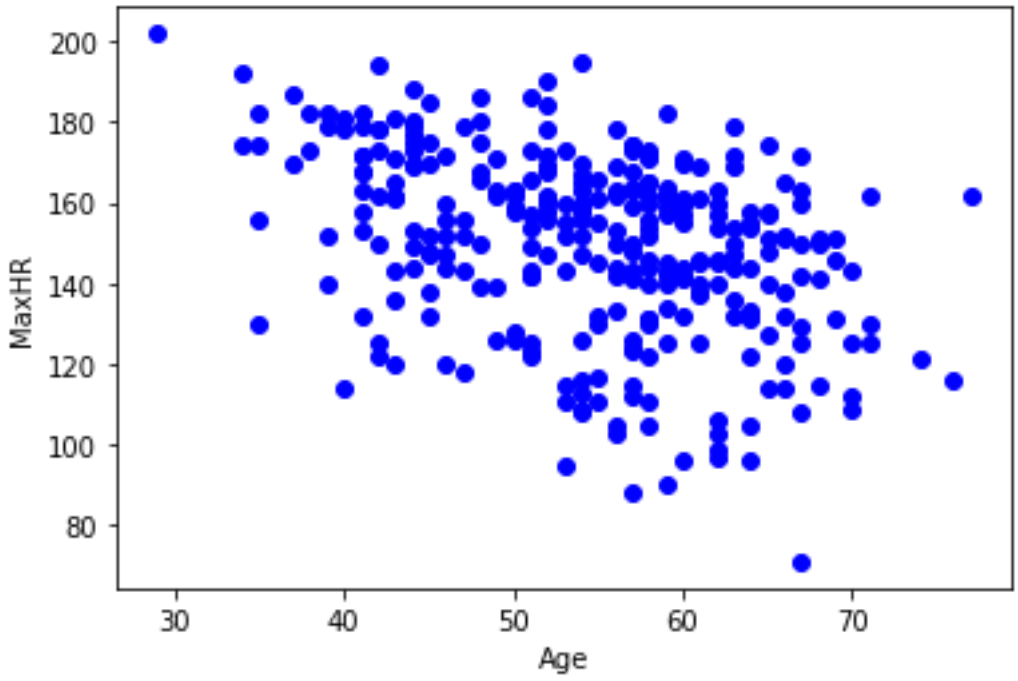
Experiment No.02
CODE & OUTPUT

import pandas as pd

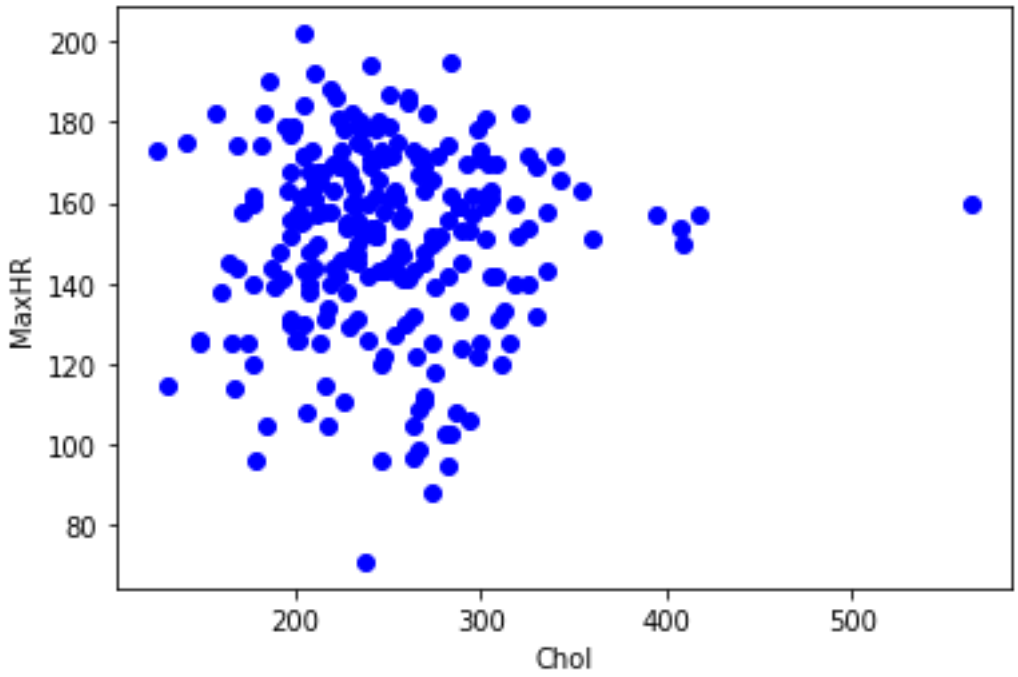
```
df=pd.read_csv('/home/lenovo/Desktop/Heart.csv')
df.shape

#Find Missing Values
import pandas as pd
df=pd.read_csv('/home/lenovo/Desktop/Heart.csv')
df.notnull()

# Finding Datatypes Of Each Column
import pandas as pd
df = pd.read_csv("/home/lenovo/Desktop/Heart.csv")
result = df.dtypes
print(result)

#finding zeros
import pandas as pd
df=pd.read_csv('/home/lenovo/Desktop/Heart.csv')
df.isin([0]).any().any()

#finding zeros
import pandas as pd
df=pd.read_csv('/home/lenovo/Desktop/Heart.csv')
(df==0).sum()

#finding mean
import pandas as pd
df=pd.read_csv('/home/lenovo/Desktop/Heart.csv')
df["Age"].mean()

# Random Splitting Training And Testing Data
import matplotlib.pyplot as plt
import pandas as pd
import pylab as pl
import numpy as np
import matplotlib.pyplot as plt
df = pd.read_csv("/home/lenovo/Desktop/Heart.csv")
df.head()
cdf = df[['Age','MaxHR','Sex','RestECG','Chol','RestBP']]
cdf.head(9)
plt.scatter(cdf.Age, cdf.MaxHR, color='blue')
```

```
plt.xlabel("Age")
plt.ylabel("MaxHR")
plt.show()
msk = np.random.rand(len(df)) < 0.8
train = cdf[msk]
test = cdf[~msk]
plt.scatter(train.Chol, train.MaxHR, color='blue')
plt.xlabel("Chol")
plt.ylabel("MaxHR")
plt.show()
```

```
Unnamed: 0      int64
Age             int64
Sex             int64
ChestPain      object
RestBP          int64
Chol            int64
Fbs             int64
RestECG         int64
MaxHR           int64
ExAng           int64
Oldpeak       float64
Slope           int64
Ca            float64
Thal           object
AHD            object
dtype: object
```



Harshal Gaikwad
T2217



Harshal Gaikwad
T2217

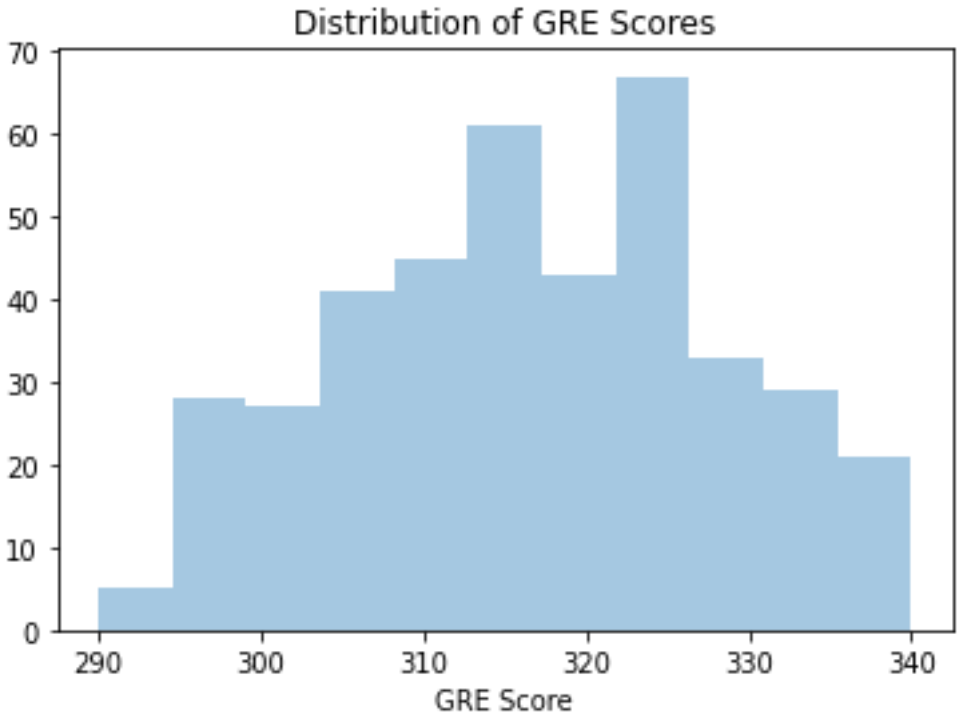
Experiment no. 03
CODE & OUTPUT

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings("ignore")
df = pd.read_csv('Admission_Predict.csv')
print(df.head(5))
df = df.drop(['Serial No.'], axis=1)
print(df.isnull().sum())
fig = sns.distplot(df['GRE Score'], kde=False)
plt.title("Distribution of GRE Scores")
plt.show()
fig = sns.distplot(df['TOEFL Score'], kde=False)
plt.title("Distribution of TOEFL Scores")
plt.show()
fig = sns.distplot(df['University Rating'], kde=False)
plt.title("Distribution of University Rating")
plt.show()
fig = sns.distplot(df['SOP'], kde=False)
plt.title("Distribution of SOP Ratings")
plt.show()
fig = sns.distplot(df['CGPA'], kde=False)
plt.title("Distribution of CGPA")
plt.show()
fig = sns.regplot(x="GRE Score", y="TOEFL Score", data=df)
plt.title("GRE Score vs TOEFL Score")
plt.show()
fig = sns.regplot(x="GRE Score", y="CGPA", data=df)
plt.title("GRE Score vs CGPA")
plt.show()
fig = sns.lmplot(x="CGPA", y="LOR ", data=df, hue="Research")
plt.title("LOR vs CGPA")
plt.show()
fig = sns.lmplot(x="CGPA", y="LOR ", data=df, hue="Research")
plt.title("LOR vs CGPA")
plt.show()
fig = sns.lmplot(x="GRE Score", y="LOR ", data=df, hue="Research")
plt.title("GRE Score vs LOR")
plt.show()
fig = sns.regplot(x="CGPA", y="SOP", data=df)
plt.title("SOP vs CGPA")
plt.show()
fig = sns.regplot(x="GRE Score", y="SOP", data=df)
```

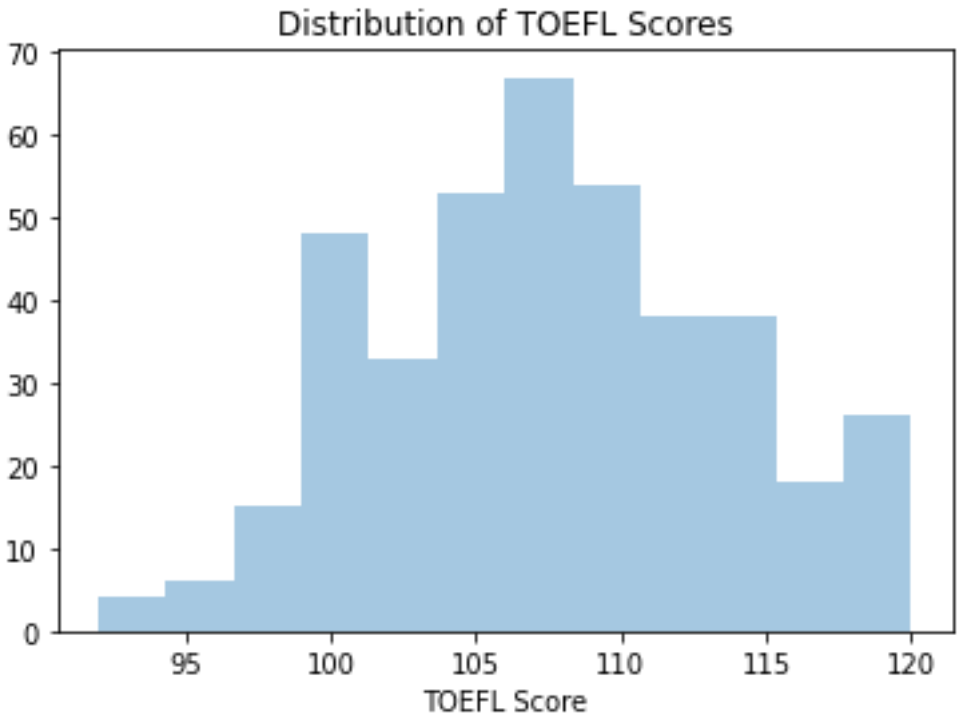
```
plt.title("GRE Score vs SOP")
plt.show()
fig = sns.regplot(x="TOEFL Score", y="SOP", data=df)
plt.title("SOP vs TOEFL")
plt.show()
import numpy as np
corr = df.corr()
print(corr)
fig, ax = plt.subplots(figsize=(8, 8))
colormap = sns.diverging_palette(220, 10, as_cmap=True)
dropSelf = np.zeros_like(corr)
dropSelf[np.triu_indices_from(dropSelf)] = True
colormap = sns.diverging_palette(220, 10, as_cmap=True)
sns.heatmap(corr, cmap=colormap, linewidths=.5, annot=True, fmt=".2f", mask=dropSelf)
plt.show()
from sklearn.model_selection import train_test_split
X = df.drop(['Chance of Admit '], axis=1)
y = df['Chance of Admit ']
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.20, shuffle=False)
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor
from sklearn.svm import SVR
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error
models = [['DecisionTree :',DecisionTreeRegressor()],
['Linear Regression :', LinearRegression()], ['SVM :', SVR()]]
print("Results...")
for name,model in models: model = model
model.fit(X_train, y_train)
predictions = model.predict(X_test)
print(name, (np.sqrt(mean_squared_error(y_test, predictions))))
classifier = RandomForestRegressor()
classifier.fit(X,y)
feature_names = X.columns
print(feature_names)
importance_frame = pd.DataFrame()
importance_frame['Features'] = X.columns
importance_frame['Importance'] = classifier.feature_importances_
importance_frame = importance_frame.sort_values(by=['Importance'], ascending=True)
plt.barh([1,2,3,4,5,6,7], importance_frame['Importance'], align='center', alpha=0.5)
plt.yticks([1,2,3,4,5,6,7], importance_frame['Features'])
plt.xlabel('Importance')
plt.title('Feature Importances')
plt.show()
```

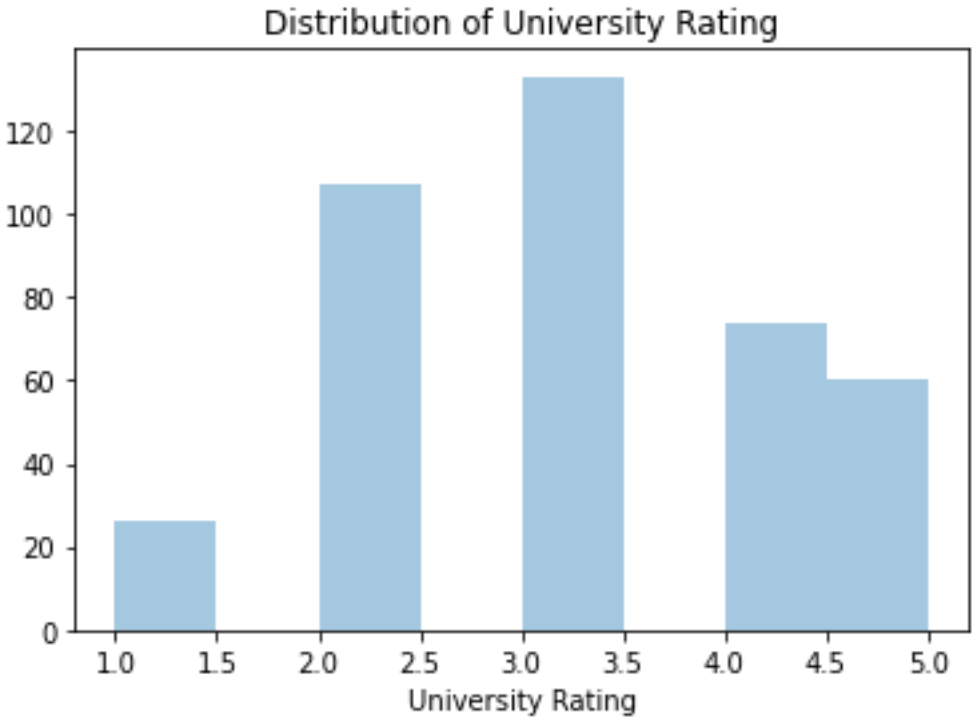
Harshal Gaikwad
T2217

```
   Serial No.  GRE Score  TOEFL Score  University Rating  SOP  LOR   CGPA  \
0           1        337          118                  4  4.5   4.5  9.65
1           2        324          107                  4  4.0   4.5  8.87
2           3        316          104                  3  3.0   3.5  8.00
3           4        322          110                  3  3.5   2.5  8.67
4           5        314          103                  2  2.0   3.0  8.21

   Research  Chance of Admit
0         1              0.92
1         1              0.76
2         1              0.72
3         1              0.80
4         0              0.65
GRE Score            0
TOEFL Score          0
University Rating    0
SOP                  0
LOR                  0
CGPA                 0
Research             0
Chance of Admit      0
dtype: int64
```

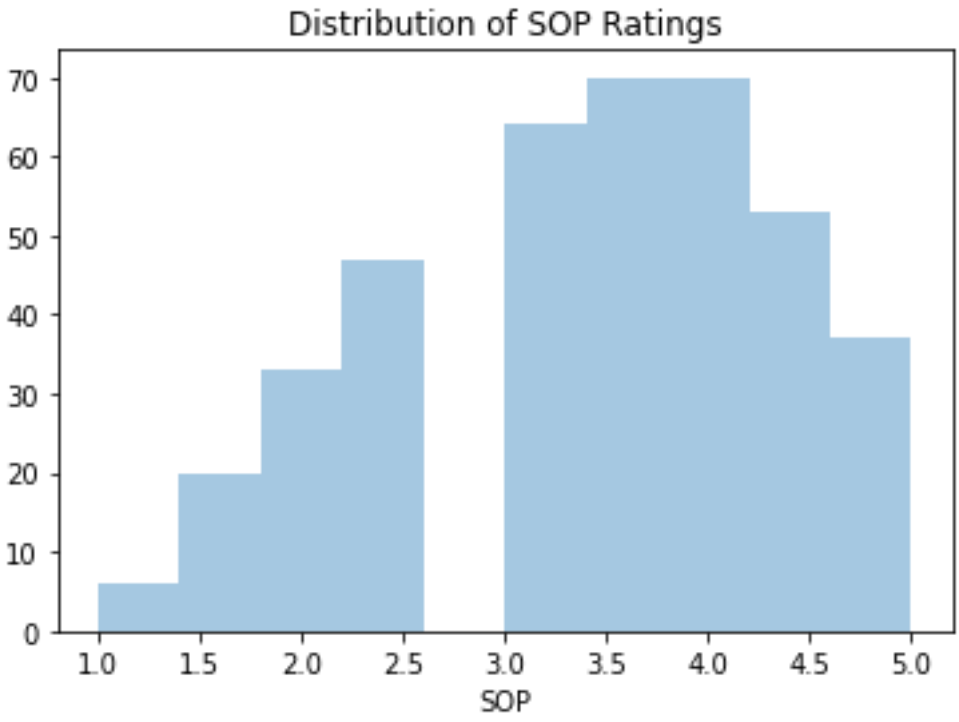


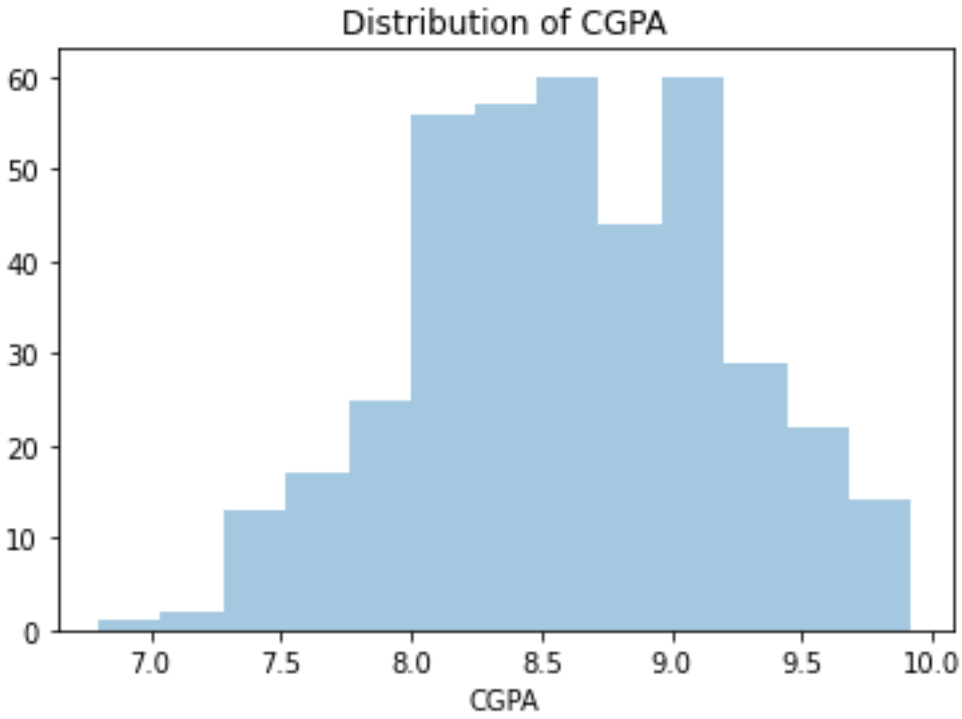
Harshal Gaikwad
T2217



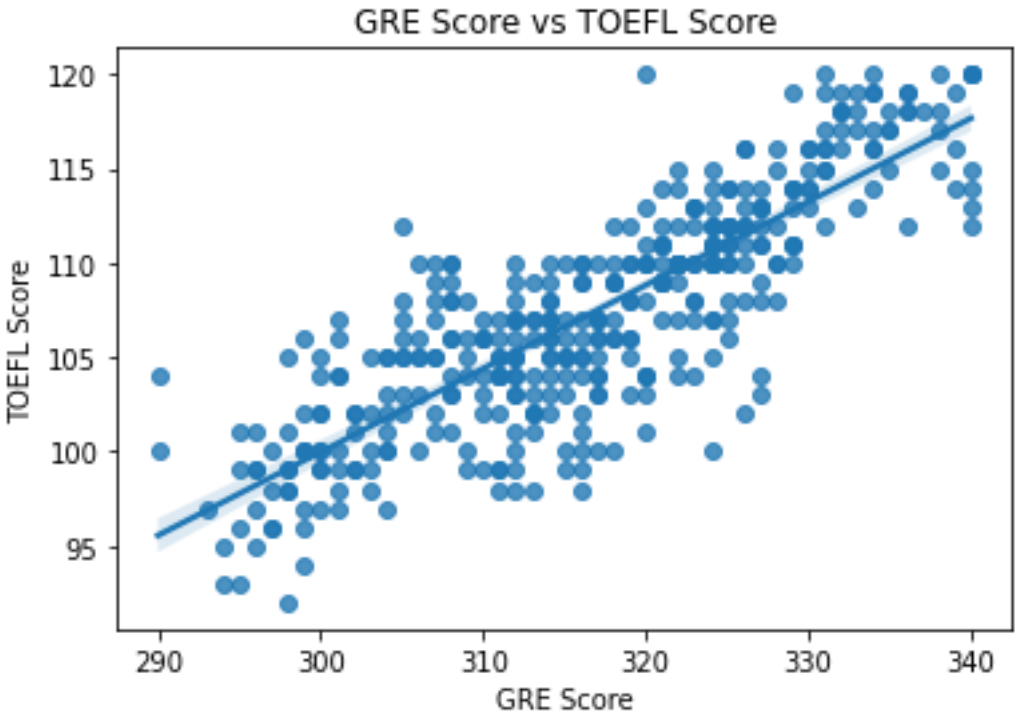


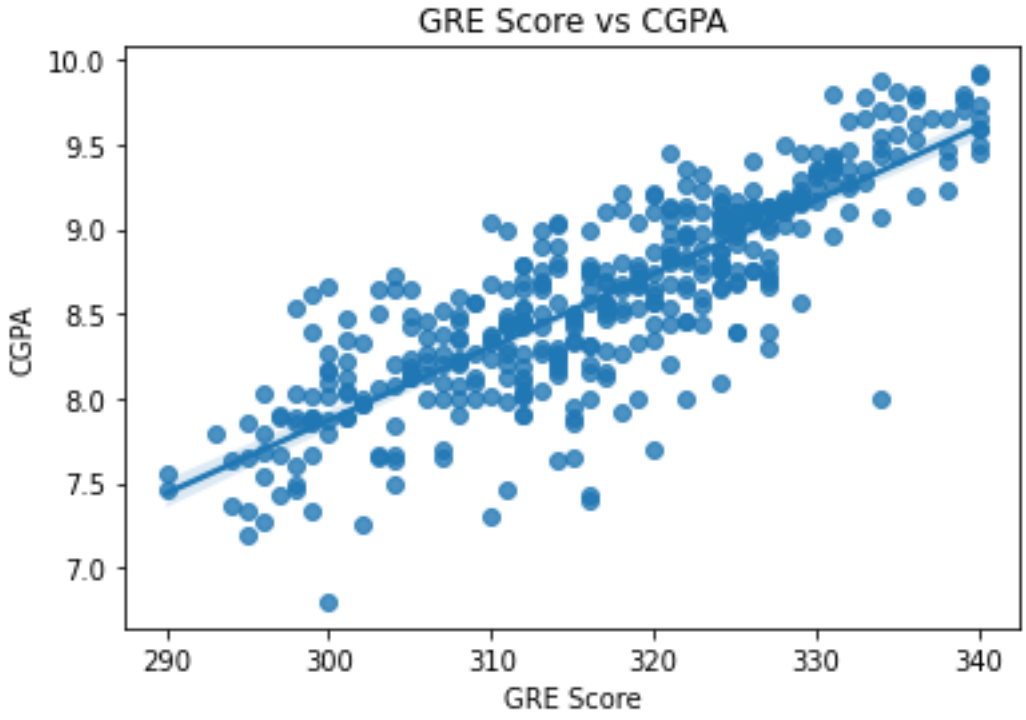
Harshal Gaikwad
T2217



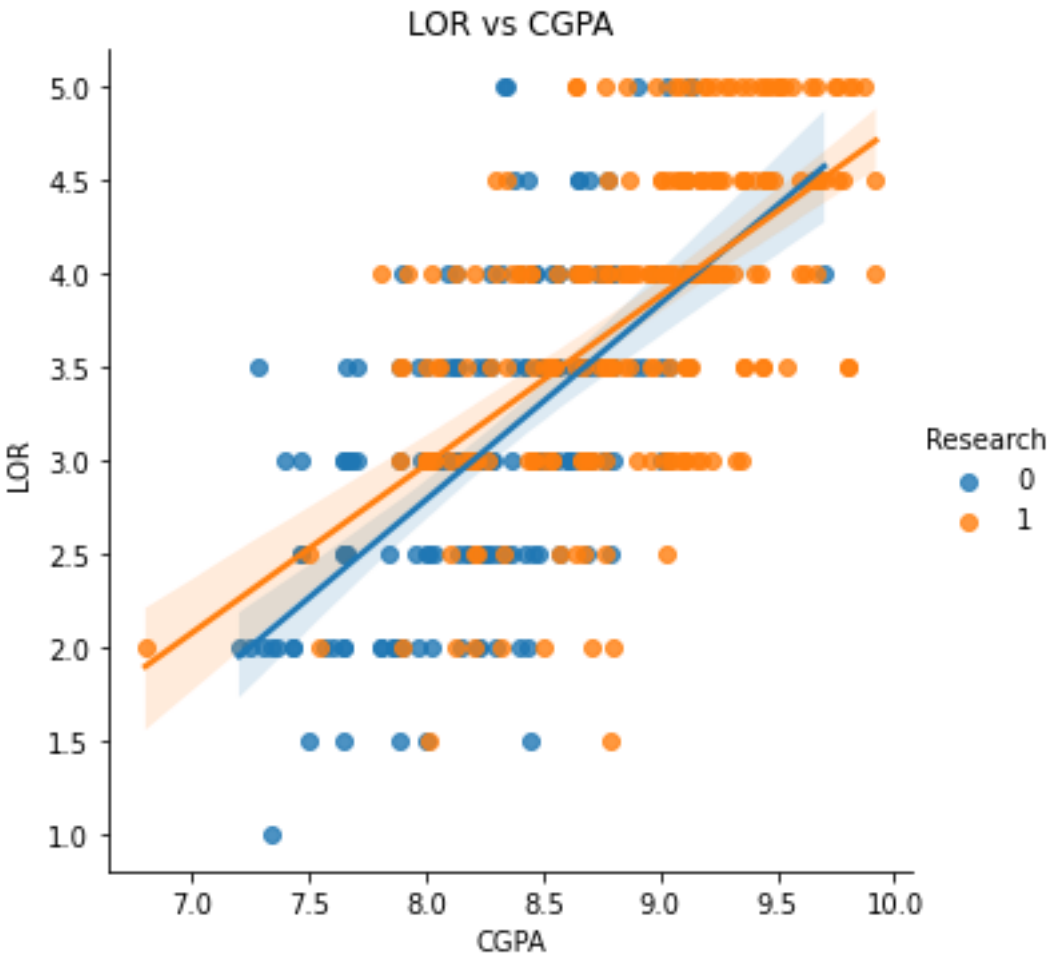


Harshal Gaikwad
T2217

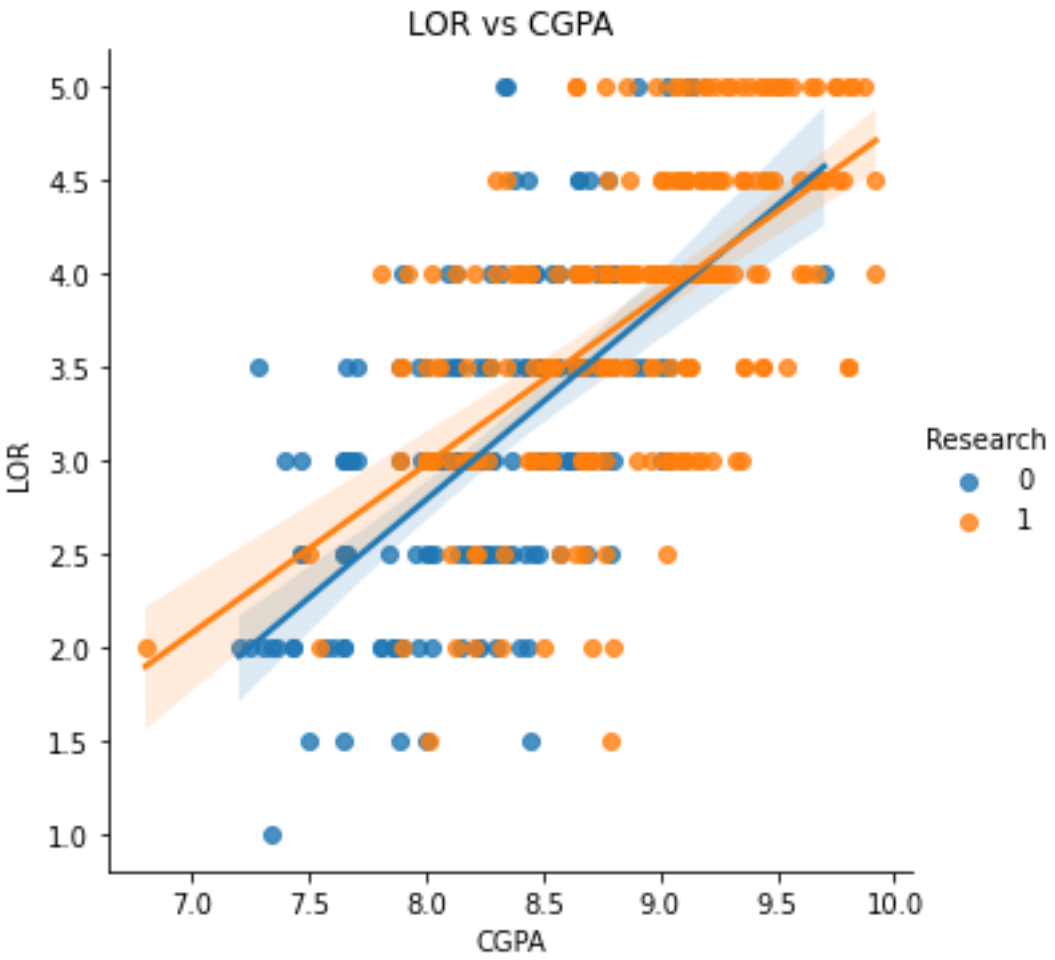




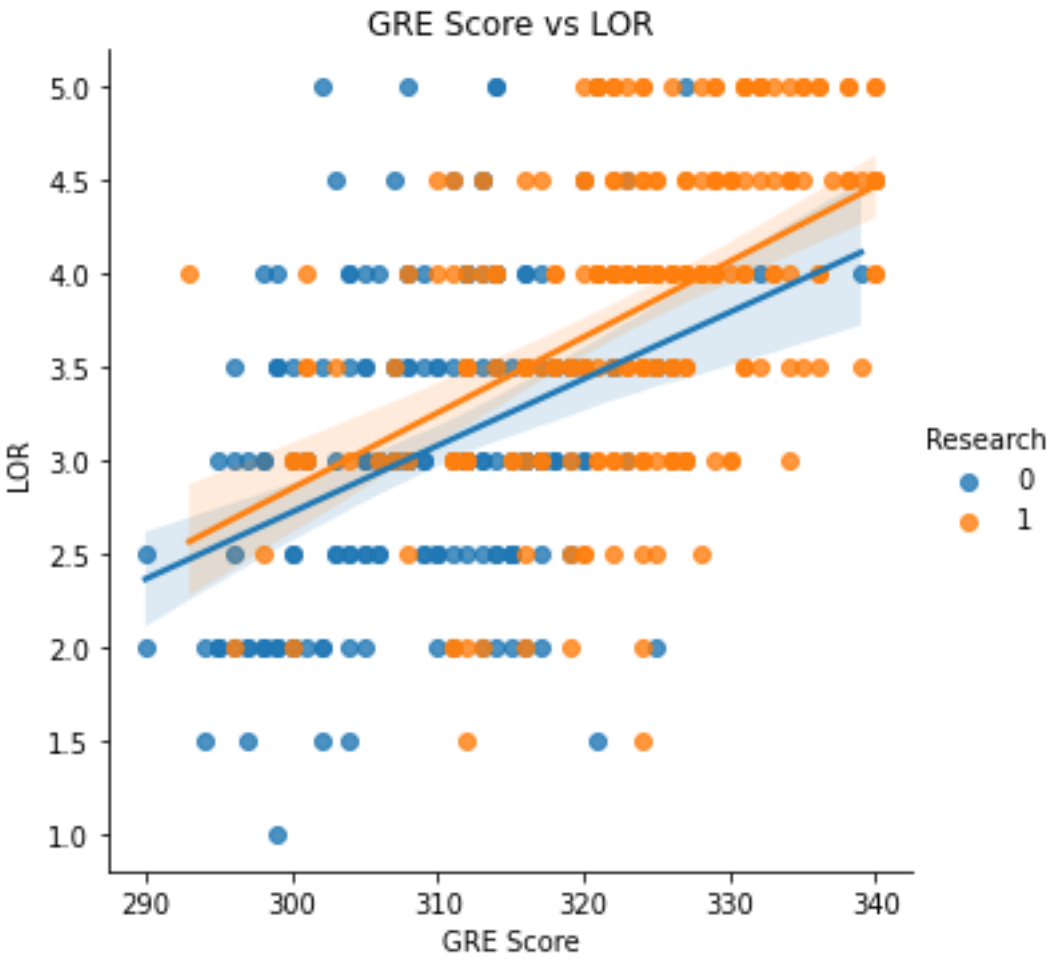
Harshal Gaikwad
T2217

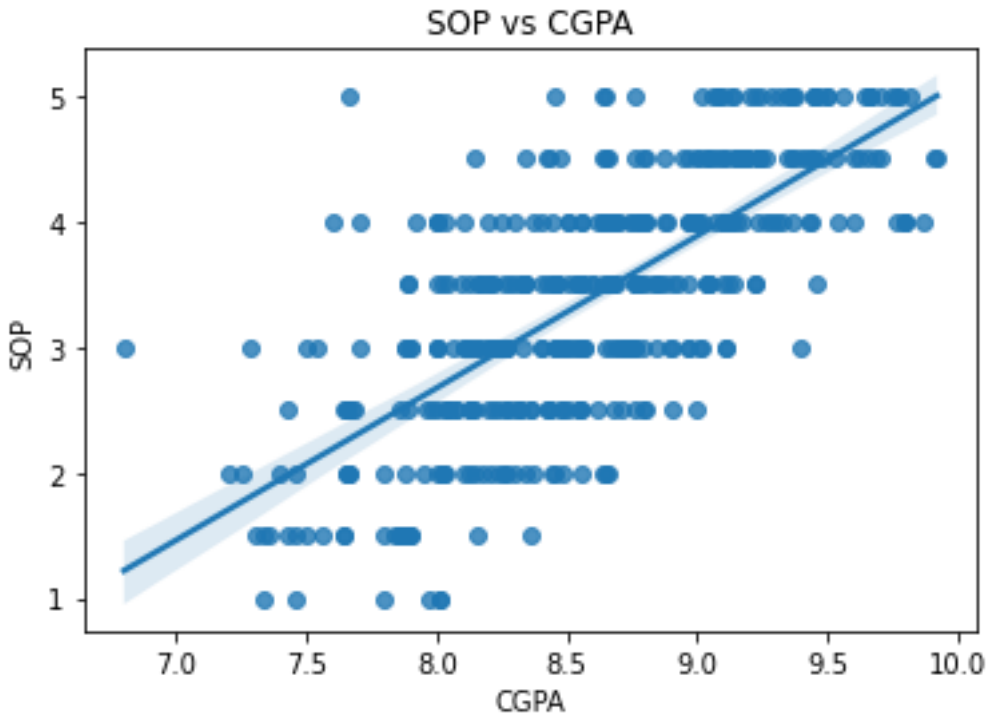


Harshal Gaikwad
T2217

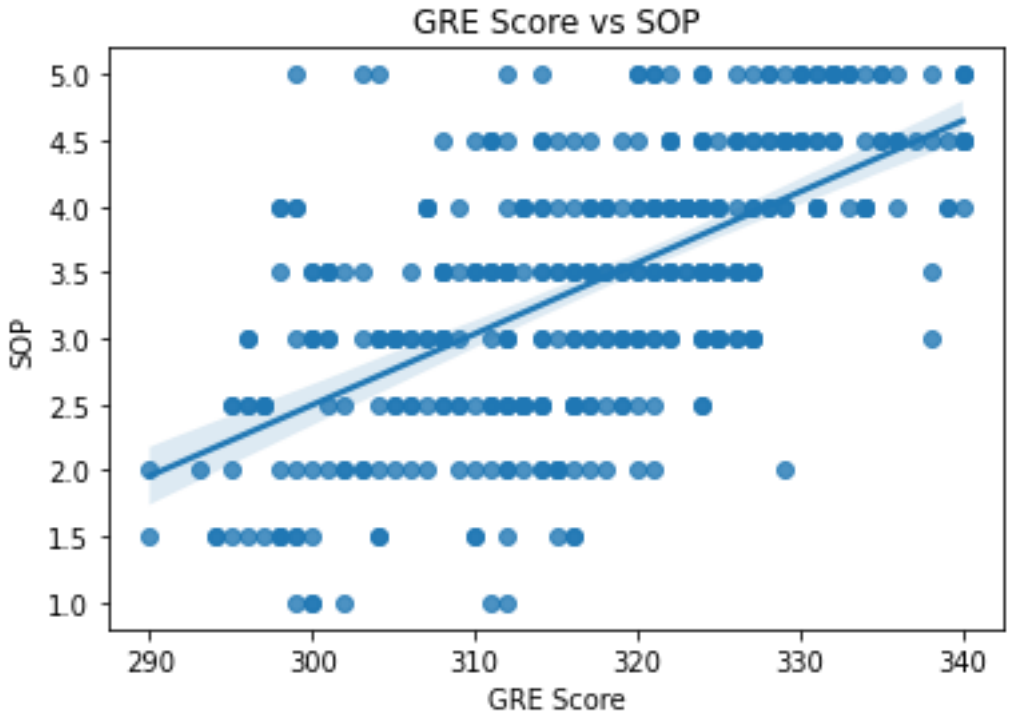


Harshal Gaikwad
T2217





Harshal Gaikwad
T2217



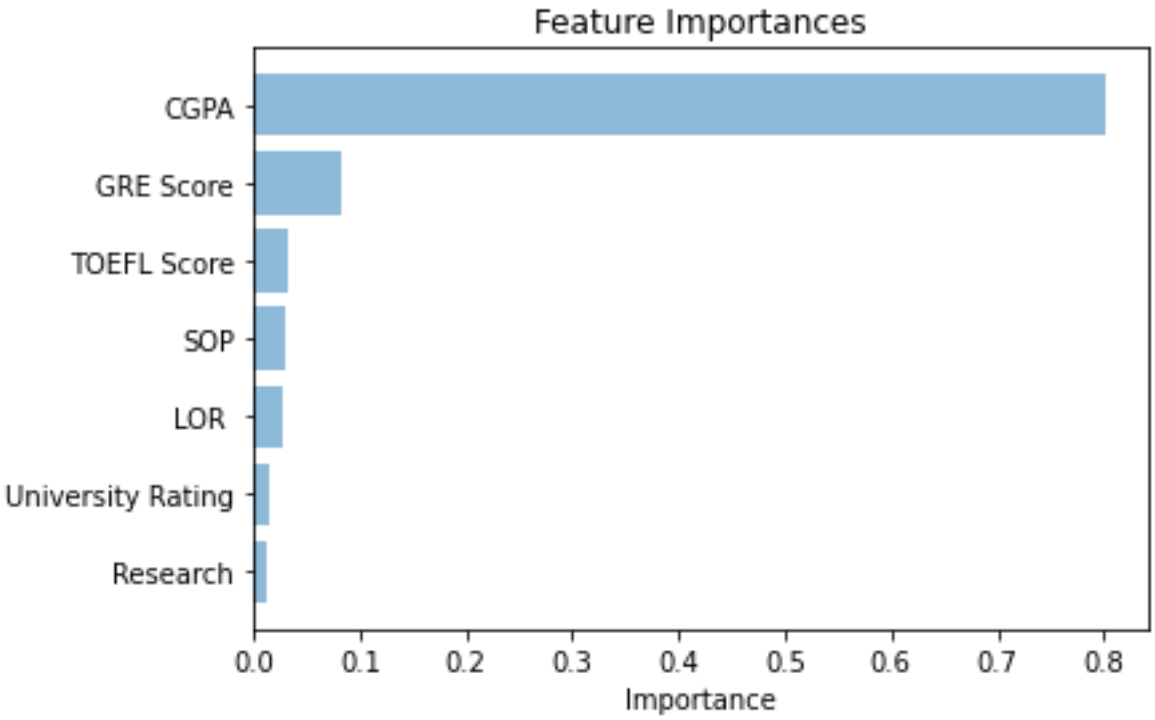
	GRE Score	TOEFL Score	University Rating	SOP	\
GRE Score	1.000000	0.835977	0.668976	0.612831	
TOEFL Score	0.835977	1.000000	0.695590	0.657981	
University Rating	0.668976	0.695590	1.000000	0.734523	
SOP	0.612831	0.657981	0.734523	1.000000	
LOR	0.557555	0.567721	0.660123	0.729593	
CGPA	0.833060	0.828417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

	LOR	CGPA	Research	Chance of Admit
GRE Score	0.557555	0.833060	0.580391	0.802610
TOEFL Score	0.567721	0.828417	0.489858	0.791594
University Rating	0.660123	0.746479	0.447783	0.711250

```
SOP                0.729593  0.718144  0.444029          0.675732
LOR                1.000000  0.670211  0.396859          0.669889
CGPA               0.670211  1.000000  0.521654          0.873289
Research           0.396859  0.521654  1.000000          0.553202
Chance of Admit    0.669889  0.873289  0.553202          1.000000
```


```
Results...
SVM : 0.08180727044650482
Index(['GRE Score', 'TOEFL Score', 'University Rating', 'SOP', 'LOR ', 'CGPA',
       'Research'],
      dtype='object')
```

Harshal Gaikwad
T2217



Harshal Gaikwad
T2217

Experiment No.04
CODE & OUTPUT

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
dataset = pd.read_csv('/home/lenovo/Desktop/Mall_Customers.csv')
X = dataset.iloc[:, [3, 4]].values
from sklearn.cluster import KMeans
wcss = []
for i in range(1, 11):
kmeans = KMeans (n_clusters = i, init = 'k-means++', random_state= 42)
kmeans.fit(X)
wcss.append(kmeans.inertia_)
plt.plot(range(1, 11), wcss)
plt.title('The Elbow Method')
plt.xlabel('Number of clusters')
plt.ylabel('WCSS')
plt.show()
kmeans = KMeans(n_clusters = 5, init = 'k-means++', random_state = 42 )
y_kmeans = kmeans.fit_predict(X)
plt.scatter(X[y_kmeans == 0,0], X[y_kmeans == 0,1], s = 100, c = 'red', label = 'Cluster 1')
plt.scatter(X[y_kmeans == 1,0], X[y_kmeans == 1,1], s = 100, c = 'blue', label = 'Cluster 2')
plt.scatter(X[y_kmeans == 2,0], X[y_kmeans == 2,1], s = 100, c = 'green', label = 'Cluster 3')
plt.scatter(X[y_kmeans == 3,0], X[y_kmeans == 3,1], s = 100, c = 'cyan', label = 'Cluster 4')
plt.scatter(X[y_kmeans == 4,0], X[y_kmeans == 4,1], s = 100, c = 'magenta', label = 'Cluster 5')
plt.scatter(kmeans.cluster_centers_[:,0], kmeans.cluster_centers_[:,1], s = 300, c = 'yellow', label = 'Centroids')
plt.title('Clusters of customers')
plt.xlabel('Annual Income (k$)')
plt.ylabel('Spending Score (1-100)')
plt.legend()
plt.show()
```

Harshal Gaikwad
T2217