# In-Lab

## Task 1

As per requirements of the task following is the code for importing libraries

```
from keras.models import Sequential
from keras.layers import Dense, Dropout
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error
import numpy as np
from sklearn import linear_model
from sklearn import preprocessing
from sklearn import tree
from sklearn.ensemble import RandomForestRegressor,
GradientBoostingRegressor
import pandas as pd
import csv

import matplotlib.pyplot as plt
```

## Task 2

```
from google.colab import files
u = files.upload()
np.random.seed(7)
df = pd.read_csv("Alumni Giving Regression (Edited).csv",
delimiter=',')
dd_df_1 = df.head()
```

## Task 3

```
print(dd_df_1)
```

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| 0 | 24 | 0.42 | 0.16 | 0.59 | 0.81 | 0.08 |
| 1 | 19 | 0.49 | 0.04 | 0.37 | 0.69 | 0.11 |
| 2 | 18 | 0.24 | 0.17 | 0.66 | 0.87 | 0.31 |
| 3 | 8 | 0.74 | 0.00 | 0.81 | 0.88 | 0.11 |
| 4 | 8 | 0.95 | 0.00 | 0.86 | 0.92 | 0.28 |

Figure 1

```
summary_statistics = dd_df_1.describe()
print(summary_statistics)
```

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| count | 5.000000 | 5.000000 | 5.000000 | 5.000000 | 5.00000 | 5.000000 |
| mean | 15.400000 | 0.568000 | 0.074000 | 0.658000 | 0.83400 | 0.178000 |
| std | 7.127412 | 0.278873 | 0.084735 | 0.194602 | 0.08961 | 0.108028 |
| min | 8.000000 | 0.240000 | 0.000000 | 0.370000 | 0.69000 | 0.080000 |
| 25% | 8.000000 | 0.420000 | 0.000000 | 0.590000 | 0.81000 | 0.110000 |
| 50% | 18.000000 | 0.490000 | 0.040000 | 0.660000 | 0.87000 | 0.110000 |
| 75% | 19.000000 | 0.740000 | 0.160000 | 0.810000 | 0.88000 | 0.280000 |
| max | 24.000000 | 0.950000 | 0.170000 | 0.860000 | 0.92000 | 0.310000 |

Figure 2

## Task 4

```
corr = df.corr(method='pearson')
corr
print(corr)
```

| | A | B | C | D | E | F |
|---|---|---|---|---|---|---|
| A | 1.000000 | -0.691900 | 0.414978 | -0.604574 | -0.521985 | -0.549244 |
| B | -0.691900 | 1.000000 | -0.581516 | 0.487248 | 0.376735 | 0.540427 |
| C | 0.414978 | -0.581516 | 1.000000 | 0.017023 | 0.055766 | -0.175102 |
| D | -0.604574 | 0.487248 | 0.017023 | 1.000000 | 0.934396 | 0.681660 |
| E | -0.521985 | 0.376735 | 0.055766 | 0.934396 | 1.000000 | 0.647625 |
| F | -0.549244 | 0.540427 | -0.175102 | 0.681660 | 0.647625 | 1.000000 |

Figure 3

## Task 5

```
Y_POSITION = 5
model_1_features = [i for i in range(0,Y_POSITION)]
X = df.iloc[:,model_1_features]
Y = df.iloc[:,Y_POSITION]
X_train, X_test, y_train, y_test =
train_test_split(X,Y,test_size=0.20,random_state=2020)
```

## Task 6

```
model1 = linear_model.LinearRegression()
model1.fit(X_train,y_train)
y_pred_train1 = model1.predict(X_train)
print("Regression")
print("============================")
RMSE_train1 = mean_squared_error(y_train,y_pred_train1)
print("Regression Train set: RMSE {}".format(RMSE_train1))
print("=========================")
y_pred1 = model1.predict(X_test)
RMSE_test1 = mean_squared_error(y_test,y_pred1)
print("Regression Test set: RMSE {}".format(RMSE_test1))
print("==================")
coef_dict = {}
for coef, feat in zip(model1.coef_,model_1_features):
  coef_dict[df.columns[feat]] = coef
print(coef_dict)

x_values = np.arange(len(y_test))
plt.scatter(x_values,y_test,color='red',label='Predicted')
plt.xlabel('Index or Sequence of Values')
plt.ylabel('Values')
plt.title('Actual vs Predicted Values')
plt.legend()
plt.show()
```

```
Regression
============================
Regression Train set: RMSE 0.002761693322289229
=========================
Regression Test set: RMSE 0.004209824026356377
==================
{'A': -0.0009337757382416938, 'B': 0.16012156890162943, 'C': -0.044160015425349614, 'D': 0.15217907817100407, 'E': 0.17539950794101047}
```

Figure 4

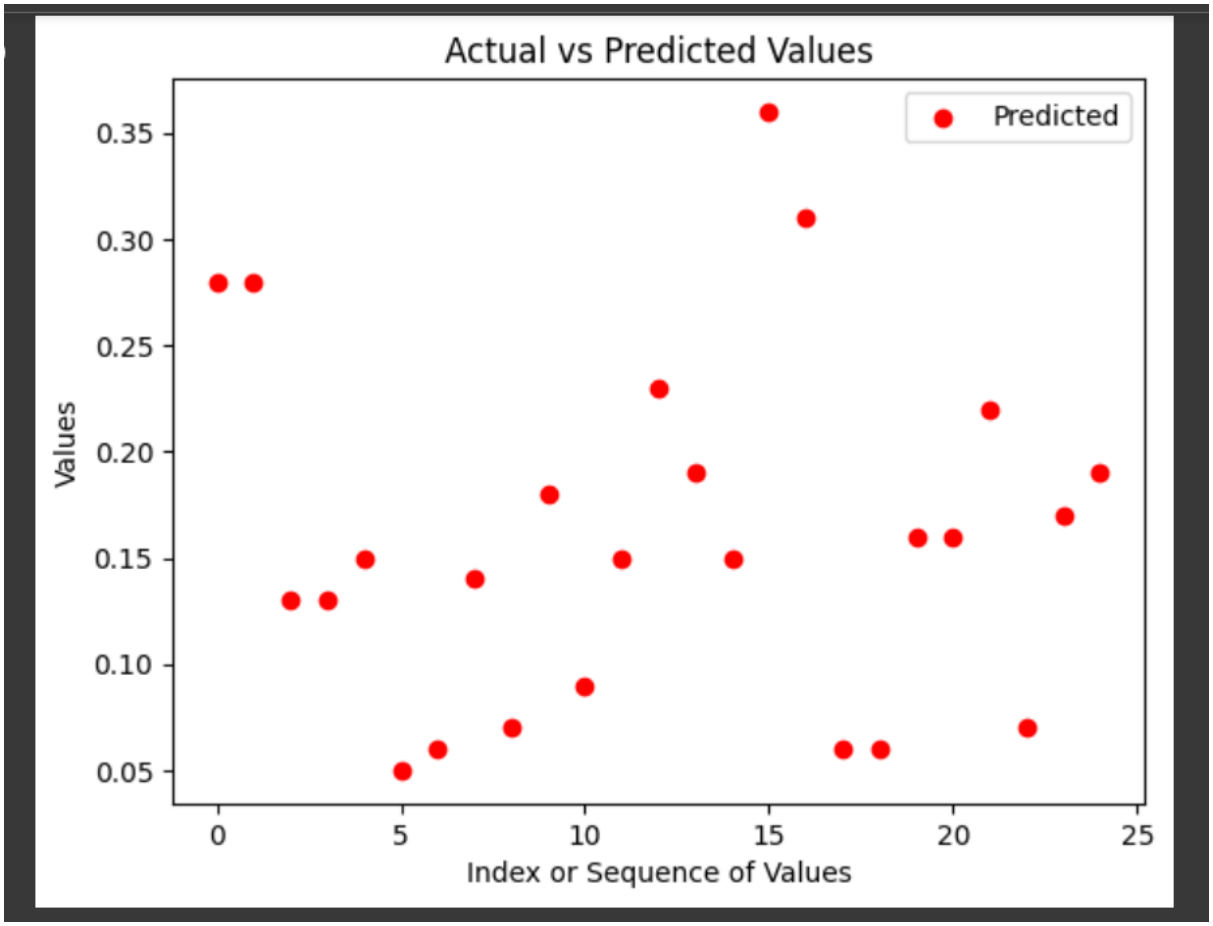
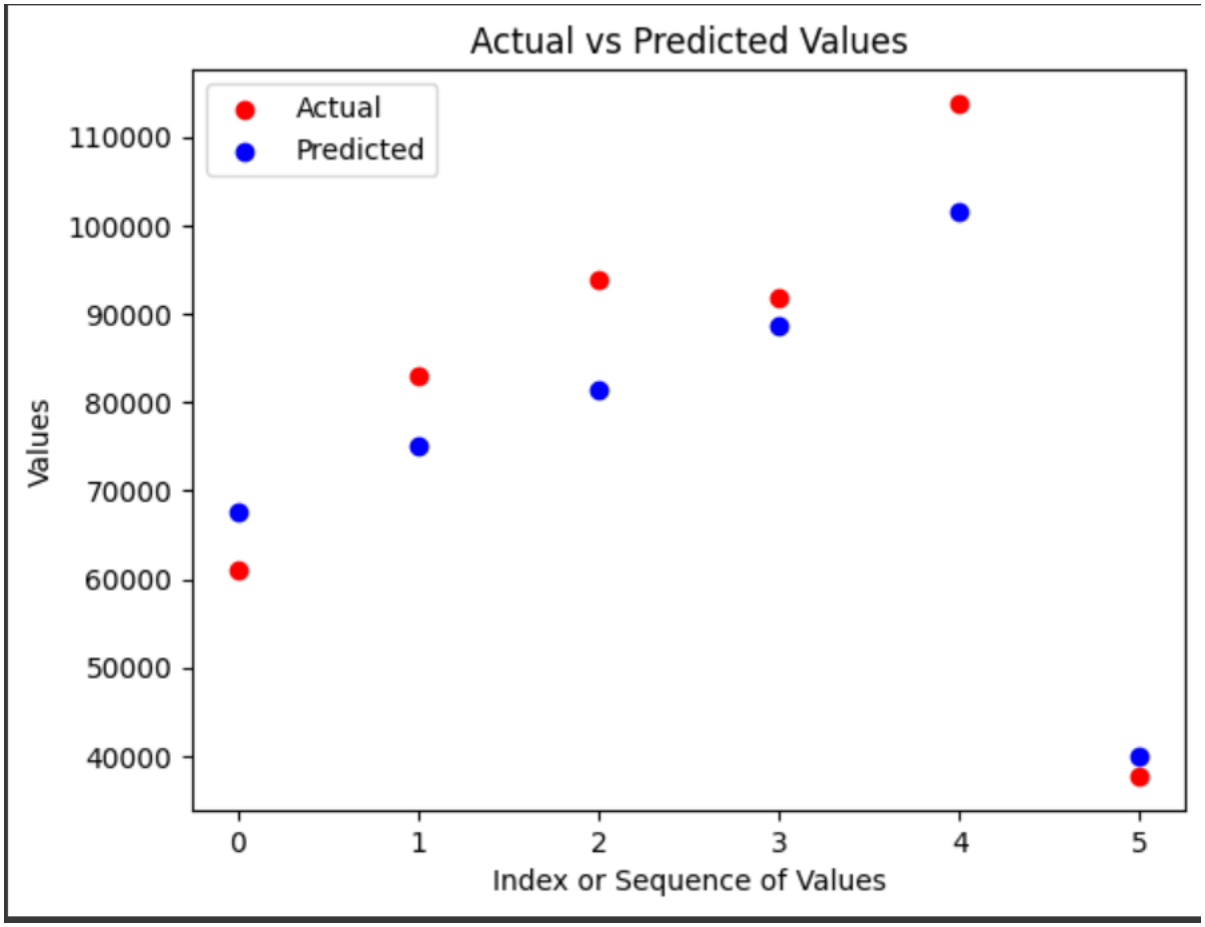

Actual vs Predicted Values
Predicted
Values
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0
5
10
15
20
25
Index or Sequence of Values

Figure 5

## Post-Lab


Actual vs Predicted Values
Actual
Predicted
110000
100000
90000
80000
70000
60000
50000
40000
Values
0
1
2
3
4
5
Index or Sequence of Values