Project for WEEK4

Dataset for predicting admissions for master’s programs.

The dataset contains several parameters which are considered important during the application for Masters Programs.
1.Describe the dataset
1.1. GRE Scores (out of 340)
1.2. TOEFL Scores (out of 120)
1.3. University Rating (out of 5)
1.4. Statement of Purpose and Letter of Recommendation Strength (out of 5)
1.5. Undergraduate GPA (out of 10)
1.6. Research Experience (either 0 or 1)
1.7. Chance of Admit (ranging from 0 to 1)
2.Length of dataset : 400
3.You can find the dataset here :
https://www.kaggle.com/adityadeshpande23/admissionpredictioncsv#Admission_Predict.csv
(https://www.kaggle.com/adityadeshpande23/admissionpredictioncsv#Admission_Predict.csv)

2.Data processing and modeling

2.1.Date Cleaning

In [1]:

```
import csv
from sklearn import linear_model
import random
```

In [2]:

```
path = "Admission_Predict.csv"
csv_file = open(path, mode='r')
reader = csv.reader(csv_file,delimiter=",")
headers = next(reader)
```

In [3]:

```
headers
```

Out[3]:

```
['Serial No.',
 'GRE Score',
 'TOEFL Score',
 'University Rating',
 'SOP',
 'LOR ',
 'CGPA',
 'Research',
 'Chance of Admit ']
```

In [4]:

```
X = []
Y = []
data = []
for row in reader:
    x_values = row[1:]
    values = [float(value) for value in x_values]
    if values[6] == float(1):
        values[6] = True
    else: values[6] = False
    data.append(values)

random.shuffle(data)
N = len(data)
X = [x[:7] for x in data]
Y = [x[7] for x in data]

X_train = X[:3*N//5]
X_valid = X[3*N//5:4*N//5]
X_test= X[4*N//5:]
y_train = Y[:3*N//5]
y_valid = Y[3*N//5:4*N//5]
y_test = Y[4*N//5:]

def MSE(model,X,y):
    predictions = model.predict(X)
    differences=[(a-b)**2 for (a,b) in zip(predictions,y)]
    return sum(differences) / len(differences)
bestModel = None
bestMSE = None

for lamb in [0.01,0.1,1,10,100]:
    model = linear_model.Ridge(lamb,fit_intercept=False) #Fit a model for each lambda v
alue
    model.fit(X_train,y_train)

    mseTrain = MSE(model,X_train,y_train)
    mseValid = MSE(model,X_valid,y_valid)

    print("lambda = "+str(lamb) + ', training/validation error ='   + str(mseTrain) +
'/' +str(mseValid))
    if not bestModel or mseValid < bestMSE:
        bestModel = model
        bestMSE = mseValid
mseTest = MSE(bestModel,X_test,y_test)
print("test error = " + str(mseTest))
```

```
lambda = 0.01, training/validation error =0.004727937578034999/0.006576544
589373302
lambda = 0.1, training/validation error =0.0047279919171912/0.006569382714
917744
lambda = 1, training/validation error =0.004732859760073529/0.006506867575
445166
lambda = 10, training/validation error =0.0049434146385454/0.0063160558727
45927
lambda = 100, training/validation error =0.006200756545284055/0.0070018027
178110715
test error = 0.004774415892471616
```

In [6]:

```
print("So the model we choose is " + str(bestModel))
```

```
So the model we choose is Ridge(alpha=10, copy_X=True, fit_intercept=Fals
e, max_iter=None,
      normalize=False, random_state=None, solver='auto', tol=0.001)
```

In [ ]:

```

```

In [ ]:

```

```