Foundations of Data Science - Assignment 2 report

 # Rohith Kumar Gattu - 2019A7PS0049H
 # Srikar Sashank M - 2019A7PS0160H
 # Ollala Nikhil Kumar - 2019A7PS0064H

In this assignment, we are required to develop the Greedy Forward and Backward feature selection algorithm from scratch and use it on a given house prices dataset. Feature selection is an impactful process in building the model as it tells us which features are actually the governing features in the given set of input features as some features may not be related to the target feature.
Feature selection is the process of reducing the number of input variables when developing a predictive model

Greedy forward feature selection:

- Assume that the number of input features is ‘n’. We consider n models with 1 feature each in greedy forward feature selection. In the following steps, we use the feature model that has the lowest RMS Error.
- The remaining features are then combined with the prior best one, and whichever one is the best is chosen.
- We employ them in the next steps since they yield the least RMS Error. In this instance, there would be n-1 models with two features We proceed in this manner until all of the n features have been completed.
- Our final model will take this into account. All n features would be represented by a single model.
- We keep note of which features were chosen for each feature as we go.
- In each situation, there are a variety of features as well as training and testing faults.

Greedy backward feature selection:

- Assume that the number of input features is n. We consider 1 model of given n features first in greedy backward feature selection. Then we take away one.
- Find the RMS Error with the least number of features from the n features.
- This stage would necessitate the creation of n models, each with n-1 features.
- The following step is to form the prior best collection of n-1 features, we eliminate one feature at a time from all the features, then we eliminate the model with the lowest RMS Error.
- In this step, there would be n-1 models with n-2 features apiece. We continue in this manner until all of the n features in our final model have been deleted.

- There would be only one model with no features. We keep track of which ones are which while doing so features were chosen for a specific quantity of features as well as the training and testing error in each case.

Implementations of forward and backward feature selection methods :

Greedy Forward feature selection:

```
def forward_selection(coeff,x,y):
    W_coeff={}
    Tr_Err={}
    features_selected=[]
    X=x[:,0].reshape((y.shape[0],1))
    for j in range(13):
        Tr_err={}
        w_coeff={}
        for i in range(13):
            if not(i in features_selected):
                r=np.c_[(X,x[:,i].reshape((y.shape[0],1)))]
                Tr_err[i],w_coeff[i],b= GD(coeff[0:j+2,:],r,y,0.0001,10000)
        idx=min(Tr_err, key=Tr_err.__getitem__)
        W_coeff[j]=w_coeff[idx]
        Tr_Err[j]=Tr_err[idx]
        if not(j==0 or Tr_Err[j-1]>Tr_Err[j]):
            W_coeff[j]=W_coeff[j-1]
            break
        X=np.hstack((X,x[:,idx].reshape((y.shape[0],1))))
        features_selected.append(idx)
    return features_selected,Tr_Err,W_coeff[j]
```

Backward feature selection :

```
def backward_selection(coeff,x,y):
    Tr_Err={}
    features_selected=[0,1,2,3,4,5,6,7,8,9,10,11,12]
    length=len(features_selected)
    W_coeff={}
    for j in range(13):
        Tr_err={}
        a={}
        for i in range(length):
            r=np.delete(x,i,1)
            Tr_err[i],a[i],b=GD(coeff[0:len(feature_list)-1-j,:],r,y,0.0001,10000)
        idx=min(Tr_err, key=Tr_err.__getitem__)
        W_coeff[j]=a[idx]
        Tr_Err[j]=Tr_err[idx]
        if not(j==0 or Tr_Err[j-1]>Tr_Err[j]):
            W_coeff[j]=W_coeff[j-1]
            break
        features_selected.pop(idx)
        x=np.delete(x,idx,1)
        length=length-1
    return features_selected,Tr_Err,W_coeff[j]
```

Subset of features that provide the optimal model :

1.**Greedy forward feature selection** : Greedy forward feature selection method that we implemented has resulted in some of the features as "**valuable**" or resulted in saying that the features that aren't so impactful for the outcome of the given attribute ('price' in our data set)

Below are the best features for this selection method.

```
            if not(j==0 or Tr_Err[j-1]>Tr_Err[j]):
                W_coeff[j]=W_coeff[j-1]
                break
            X=np.hstack((X,x[:,idx].reshape((y.shape[0],1))))
            features_selected.append(idx)
        return features_selected,Tr_Err,W_coeff[j]
```

```
[17] features_selected,Err,Wnew=forward_selection(W,X,y_train)
```

```
features_selected
```

```
[3, 9, 7, 12, 8, 11, 10, 5, 0, 6]
```

The linear regression model that contains the optimal features are :

3 - **sqft_lot**
9 - **sqft_above**
7 - **condition**
12 - **sqft_lot15**
8 - **grade**
11 - **sqft_living15**
10 - **sqft_basement**
5 - **waterfront**
0 - **bedrooms**
6 - **view**

2. **Greedy backward feature selection** :

Greedy backward feature selection method that we implemented have resulted in the following features as the optimal model (the model that contains the linear combination of these features)

Below are the best features for this selection method.

```
            length=length-1
        return features_selected,Tr_Err,W_coeff[j]

[21] bestb,Errorsb,wnewb=backward_selection(W,X,y_train)

[22] bestb

     [0, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12]
```

The linear regression model that contains the features
0 - **bedrooms**
3 - **sqft_lot**
4 - **floors**
5 - **waterfront**
6 - **view**
7 - **condition**
8 - **grade**
9 - **sqft_above**
10 - **sqft_basement**
11 - **sqft_living15**
12 - **sqft_lot15**

The table containing the minimum training error and minimum testing error obtained from the following methods are given below :

Method used	Min training error	Min testing error
Linear regression model without any pre-processing and feature selection	28407982437.049297	54587384589.615555
Greedy forward feature selection	0.09103832035036058	0.1540680155377123
Greedy backward feature selection	0.09080127308257367	0.08318371199775261

The linear regression model that we have implemented without pre processing and any feature selection , we have taken the

Learning rate = 0.00000000025(eta)
Iterations = 100000