1. Import Packages and Observe Dataset

In [ ]:
```
import pandas as pd
import numpy as np

import seaborn as sns
import matplotlib.pyplot as plt
%matplotlib inline
```

In [9]:
```
from sklearn import preprocessing
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression, Ridge, Lasso
from sklearn.metrics import r2_score
```

In [10]:
```
data = pd.read_csv(r"C:\Users\admin\Downloads\car-mpg.csv")
data.head()
```

Out[10]:

	mpg	cyl	disp	hp	wt	acc	yr	origin	car_type	car_name
0	18.0	8	307.0	130	3504	12.0	70	1	0	chevrolet chevelle malibu
1	15.0	8	350.0	165	3693	11.5	70	1	0	buick skylark 320
2	18.0	8	318.0	150	3436	11.0	70	1	0	plymouth satellite
3	16.0	8	304.0	150	3433	12.0	70	1	0	amc rebel sst
4	17.0	8	302.0	140	3449	10.5	70	1	0	ford torino

In [24]:
```
# 1. Drop 'car_name' if it exists
if 'car_name' in data.columns:
    data = data.drop(['car_name'], axis=1)

# 2. Replace numbers in 'origin' with names and make dummy columns
if 'origin' in data.columns:
    data['origin'] = data['origin'].replace({1: 'america', 2: 'europe', 3: 'asia'})
    data = pd.get_dummies(data, columns=['origin'])

# 3. Replace '?' with NaN
data = data.replace('?', np.nan)

# 4. Make sure all numbers are numeric (fix warning)
data = data.apply(pd.to_numeric, errors='coerce')

# 5. Fill missing numbers with median
data = data.fillna(data.median())
```

In [25]:
```
data.head()
```

Out[25]:

	mpg	cyl	disp	hp	wt	acc	yr	car_type	origin_america	origin_asia	origin_eur
0	18.0	8	307.0	130.0	3504	12.0	70	0	True	False	F
1	15.0	8	350.0	165.0	3693	11.5	70	0	True	False	F
2	18.0	8	318.0	150.0	3436	11.0	70	0	True	False	F
3	16.0	8	304.0	150.0	3433	12.0	70	0	True	False	F
4	17.0	8	302.0	140.0	3449	10.5	70	0	True	False	F

In [26]:
```
X=data.drop(['mpg'],axis=1)
Y=data[['mpg']]
```

In [30]:
```
from sklearn import preprocessing

# Scale X
X_s = preprocessing.scale(X)
X_s = pd.DataFrame(X_s, columns=X.columns)

# Scale Y
Y_s = preprocessing.scale(Y)
Y_s = pd.DataFrame(Y_s, columns=Y.columns)
```

In [31]:
```
X_s
```

Out[31]:

	cyl	disp	hp	wt	acc	yr	car_type	origin_ame
0	1.498191	1.090604	0.673118	0.630870	-1.295498	-1.627426	-1.062235	0.773
1	1.498191	1.503514	1.589958	0.854333	-1.477038	-1.627426	-1.062235	0.773
2	1.498191	1.196232	1.197027	0.550470	-1.658577	-1.627426	-1.062235	0.773
3	1.498191	1.061796	1.197027	0.546923	-1.295498	-1.627426	-1.062235	0.773
4	1.498191	1.042591	0.935072	0.565841	-1.840117	-1.627426	-1.062235	0.773
...	...	...	...	...	...	...	...	
393	-0.856321	-0.513026	-0.479482	-0.213324	0.011586	1.621983	0.941412	0.773
394	-0.856321	-0.925936	-1.370127	-0.993671	3.279296	1.621983	0.941412	-1.292
395	-0.856321	-0.561039	-0.531873	-0.798585	-1.440730	1.621983	0.941412	0.773
396	-0.856321	-0.705077	-0.662850	-0.408411	1.100822	1.621983	0.941412	0.773
397	-0.856321	-0.714680	-0.584264	-0.296088	1.391285	1.621983	0.941412	0.773

398 rows × 10 columns

In [36]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, Y_train, Y_test = train_test_split(X_s, Y_s, test_size=0.2, random
X_train.shape, X_test.shape, Y_train.shape, Y_test.shape
```

Out[36]: ((318, 10), (80, 10), (318, 1), (80, 1))

2.a Simple Linear Model

In [39]:
```
regression_model=LinearRegression()
regression_model.fit(X_train,Y_train)

for idX,col_name in enumerate(X_train.columns):
    print('The coefficient for {} is {}'.format(col_name,regression_model.coef_[0][

intercept=regression_model.intercept_[0]
print('The intercept is {}'.format(intercept))
```

```
The coefficient for cyl is 0.3079552263085646
The coefficient for disp is 0.2749461027215073
The coefficient for hp is -0.2003169751506641
The coefficient for wt is -0.6755882897308452
The coefficient for acc is 0.03311218992661376
The coefficient for yr is 0.3720678277926433
The coefficient for car_type is 0.37184733079847426
The coefficient for origin_america is -0.07869861984419448
The coefficient for origin_asia is 0.05552985672521807
The coefficient for origin_europe is 0.04186331320884182
The intercept is 0.004215857290129611
```

2.b Regularized Ridge Regression

In [41]:
```
from sklearn.linear_model import Ridge

ridge_model = Ridge(alpha=0.4)
ridge_model.fit(X_train, Y_train)

print('Ridge model coefficients:', ridge_model.coef_)
```

```
Ridge model coefficients: [ 0.30194173  0.26525512 -0.19939913 -0.66652074  0.031584
92  0.37102275
  0.36713844 -0.07824287  0.05547931  0.04133693]
```

2.c Regularized Lasso Regression

In [43]:
```
from sklearn.linear_model import Lasso

# Create and train Lasso model
lasso_model = Lasso(alpha=0.1)
lasso_model.fit(X_train, Y_train)  # use .fit(), not .fir()
```

```
# Print coefficients
print('Lasso model coefficients:', lasso_model.coef_)  # lowercase variable name
```

```
Lasso model coefficients: [-0.         -0.         -0.02693123 -0.49686961  0.
0.28991927
  0.12882705 -0.03280583  0.          0.        ]
```