1.) Import the data from CCLE into a new Google Colab file

```
import pandas as pd
from google.colab import drive
import matplotlib.pyplot as plt
```

```
drive.mount('/content/gdrive/', force_remount = True)
```

```
Mounted at /content/gdrive/
```

```
import sklearn as sk
```

```
from sklearn.linear_model import LinearRegression
```

```
import statsmodels.api as sm
```

```
df = pd.read_csv("/content/gdrive/MyDrive/Econ441B/insurance.csv")
```

```
df.loc[df['sex']=='female', 'sex'] = 1
```

```
df.loc[df['sex']=='male', 'sex'] = 0
```

```
df.loc[df['smoker']=='yes', 'smoker'] = 1
```

```
df.loc[df['smoker']=='no', 'smoker'] = 0
```

```
df.head()
```

	age	sex	bmi	children	smoker	region	charges
0	19	1	27.900	0	1	southwest	16884.92400
1	18	0	33.770	1	0	southeast	1725.55230
2	28	0	33.000	3	0	southeast	4449.46200
3	33	0	22.705	0	0	northwest	21984.47061
4	32	0	28.880	0	0	northwest	3866.85520

```
df.loc[df['region']== 'southwest', 'region'] = 0
df.loc[df['region']== 'northwest', 'region'] = 1
df.loc[df['region']== 'southeast', 'region'] = 2
df.loc[df['region']== 'northeast', 'region'] = 3
```

2.) Split the data into 80/20, in/out sample

```
import numpy as np
```

```
# every row and every column except for the last one, needs to be in an array for sklearn
data = np.array(df.iloc[:, :-1])
```

```
target= np.array(df.iloc[:, -1]) # changed to -3
```

```
cut = int((len(data) * .8)//1)
```

```
in_data = data[:cut]
out_data = data[cut:]
```

```
in_target = target[:cut]
```

```
out_target = data[cut:]
```

3.) Normalize the Data

```
from sklearn import preprocessing


scaler = preprocessing.StandardScaler().fit(in_data)
# making a scaler object that is fit to our in sample data, allows us to scale any data with respect to the mean/sd of sample data



in_data_scale = scaler.transform(in_data)


# scaling our out of sample data based on the in sample data
out_data_scale = scaler.transform(out_data)
```

4.) Get lambda from Lasso cross validation

```
# use function LassoCV


#.alpha


# have to figure this one out
from sklearn.linear_model import LassoCV


modCV = LassoCV().fit(in_data_scale, in_target)


a = modCV.alpha_


a
```

```
     176.27926602952408
```

```
#region shud be 4 binary variables, so duplicate columns
```

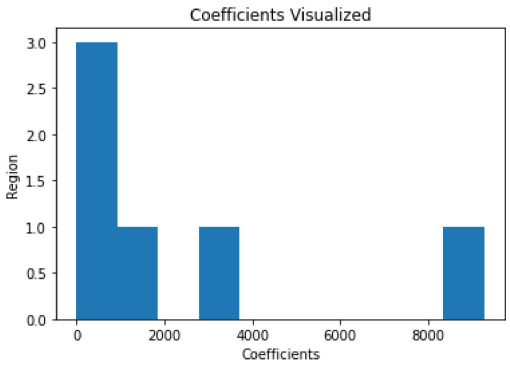
5.) Run a lambda regression with that Lambda

```
# use Lasso function/ the alpha above
from sklearn.linear_model import Lasso



mod1 = sk.linear_model.Lasso(alpha = a).fit(in_data_scale, in_target)


coef = mod1.coef_
```

6.) Visualize the coefficients

```
plt.hist(coef)
plt.title('Coefficients Visualized')
plt.xlabel('Coefficients')
plt.ylabel('Region')


# 0 is southwest
# 1 is northwest
# 2 is southeast
#3 is northeast
```

```
Text(0, 0.5, 'Region')
```



7.) Interpret the coefficients

In the plot above, region was broken up as such: 0 is southwest, 1 is northwest, 2 is southeast, 3 is northeast. Coefficients with a score less than 1000 were associated with the northeast. Coefficients with scores between 1000 and 2000, between 3000 and 4000, and between 8000 and 9000 were associated with the northwest.

8.) Compare in and out of sample MSE's

```
mod1.predict(np.array(in_data_scale))
```

```
array([25165.72081561,  4111.67312725,  6985.78958892, ...,
       11841.48932999, 11618.73145994, 12554.34362287])
```

```
mod1.predict(np.array(out_data_scale))
```

```
array([32986.21227301, 14520.02948007,  3662.30461561, 12114.62425727,
        9888.77478496,  6372.648214  , 10528.91591359,  2421.05429294,
       28839.7208477 , 15927.63497427,   863.27520379,  5700.16014472,
        4864.98739871,  6632.40560948, 14286.22429435, 28829.13566582,
       11789.53587507, 12743.01953297, 16744.39589255,  9545.28697148,
       34923.08968067, 12042.68110969,  5341.72764788, 26808.48015779,
       12686.61071212,  4452.71331449, 36314.49634992,  4809.70881489,
       11602.38640984,  5926.09691852, 27119.07929919, 11624.75913734,
        8391.26716732, 14489.97823528,  7077.00973455, 12057.37637924,
       10265.67116259, 10021.72039712,  4688.90230071,  7488.6928644 ,
       12825.64830864, 34230.04490394, 33016.22956642,  4853.64537152,
        2470.81460765, 13032.22051585,  9995.63703699, 29181.66608218,
       31595.7989852 ,  3468.91602343, 27201.58043261, 12713.11374767,
       37372.64615687,  6106.52567756, 31322.22579959, 12329.75551131,
       11604.78395058,  9280.29791928,  7517.10108397,  -737.21138698,
        7634.60634472,  9947.97046638, 15636.47470898,  7583.84130221,
        6171.02732521, 10885.82225   ,  7706.68739799,  2239.70674643,
        6516.39982714, 26677.58518913, 13085.46550774,  9810.28660434,
        9668.61331196,  9237.70192577, 11668.99012105, 12763.85370359,
       36862.3185657 ,  3609.29720996,  9419.2999362 ,  9599.62853347,
        2937.08603496, 14455.53458775, 33753.41080621,  8812.54343215,
       10631.32835315,  5920.97992949, 30498.84468544,  2374.39857245,
        3521.26406719,  9173.18118121, 10446.07435382, 10886.79816277,
        8608.59691824,  2318.28657257,  8089.8796793 ,  5947.75555275,
       15510.24740914,  4141.34161075,  8029.4447984 ,  8806.04925029,
       25516.12085816, 31411.09848076, 15716.30676406,  7923.01461824,
        6517.74877383,  2529.13157531, 32750.58296812,  7417.67793797,
        5536.6846713 , 29035.75260186, 12498.95917205,  4018.21757058,
        4226.27015928,  9863.64253546, 26947.30362642,  8290.41505988,
       28760.35318764, 14330.03723086, 31144.77498778,  3045.498387  ,
        7214.43579922,  6454.84788056, 13853.36813165, 11984.68039924,
        3348.56122338,  3082.72529933, 25946.27673358,  9492.6367593 ,
        8149.38126681,  4996.42945546,  6210.49956346, 13213.08781765,
        4165.3479939 , 12075.9538776 , 25942.67436185,  3496.49350528,
       14658.26415865, 31645.52267021, 29729.76787057, 15037.68275649,
        7599.82466473,  9758.79620989,   321.25194396, 12464.8627062 ,
        5463.25036477,  5552.92012596,  6734.97237331,  8204.29816305,
       34198.23877728,  8505.84398823,  3762.22279666,  7305.51775472,
        9299.09059563, 24691.02177627,  7101.77459124,  9651.4068841 ,
        4544.09342551, 11420.40563813, 13501.65605656, 12933.01178218,
       36419.56680593, 23609.55354276, 10844.70663798, 10400.37481619,
        9401.78668481,  4855.54083743, 11622.92290616, 12008.46398649,
```

```
 6339.64156291,  8336.15798256, 38436.68220386, 40002.4506467 ,
 1863.51235005,  6113.68732043,  3679.84266914,  4707.29123898,
 8766.95272429,  7165.21418905,  5601.37784181, 30871.09325685,
27445.26128871,  -381.91982962, 25237.97318062,  8884.89356503,
 6017.2249374 , 10724.71846508, 13582.79099494, 11141.66480911,
14976.16504573,  9324.07929832,  3538.55347258,  7290.8403024 ,
 8165.77400852,  8907.5631135 , 12187.11420851, 35491.12974167,
11782.42972926, 27810.91551648,  3865.00833426,  9299.89006032,
 6000.91517322,  5618.2236322 ,  9122.7248108 ,  6536.11157541,
26972.05206453, 10268.90719162,  3964.80585797,  5961.14822345,
31523.41757929,  3943.03029485, 11176.78697955, 32843.35732354,
23526.92907365,  3004.3477735 , 38450.04063874,  8412.1155594 ,
 1582.63445599,  5973.97091656, 29423.61015808, 10771.86682672,
 5472.29261909, 27238.25490016,  1433.24496651,  9183.04846802,
11414.1403745 ,   820.90305628,  1699.48711019,  4762.69752916,
 6139.10618232,  1832.68481449, 32697.16857718, 37849.7128077 ,
```