In [ ]:
```
#Define Machine Learning
#1. The Broader Notion of Building Statistical Artifacts That Become M
ore Accurate Over Time Based on Experience
#2. Linear Algebra + Statistical Analysis, Written In Code
#3. Cost = Sumation(answer - guess)**2
```

In [1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

%matplotlib inline
plt.style.use('seaborn-poster')
```

In [2]:
```
df = pd.read_csv('/Users/swllms/DAT-10-14-SW/class material/Unit3/Data
/housing.csv')
```

In [3]:
```
df.head() #housing prices from the boston area that includes 13 charac
teristic of the house to predict price
#Target variable = price , guess = prediction
```

Out[3]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LST
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5

In [4]:
```
#See additonal notebook on machine learning terms:
#Regression, classification, supervised, unsupervised, structured, uns
tructered
```

In [ ]:
```
#Machine Learning: Linear Models - Most common algorithm used for Mach
ine Learning
#Based on the old equation y = mx + b
#Can extend this intuition to multiple variables
```

In [ ]:
```
#LSTAT = Lower economic status
#Linear regression is the second best model for everything, similar to python, taking the linear combination and creating a prediidiction
#Linear Models: Very well understood, Performant, Results are interpretable: easy to understand what causes what
#not great with subtile relations or outliers, requires more data preperation than other techniques, not as accurate
```

In [ ]:
```
#Scikit Learn: The main library used to implement ML methods; runs on CPU and not GPU (grpahic cards); used most
```

In [6]:
```
from sklearn.linear_model import LinearRegression
```

In [ ]:
```
Big Three Methods
#fit() - apply the algorithm to your data
#score() - evaluate your algorithm
#predict() - estimate answer based on new info

#get_params() - access parameters of your algorithm
#set_params() - change parameters of your algorithm
```

In [7]:
```
lreg = LinearRegression() #Must initialize the algorithm before you can use.
#(creating an instance of algo to be used later on)
```

In [8]:
```
X = df[['LSTAT']] #X is upercase and generally multi colms (sklearn requres the dimensions of your data to do predictions)
y = df['PRICE']
```

In [10]:
```
X.shape #The 1 indicates the diminsion of the table needed for predictions
```

Out[10]:
```
(506, 1)
```

In [11]:
```
type(X) #second [] make a dataframe
```

Out[11]:
```
pandas.core.frame.DataFrame
```

In [12]:
```
type(y) #this is a series
```

Out[12]:
```
pandas.core.series.Series
```

In [13]:
```
lreg.fit(X,y) #this allows Sklearn to learn your dataset and open up more information to be discovered () contains the different parameters/arguments
```

Out[13]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [14]:
```
lreg.coef_ #Basically the slope of the line or M in the equation y=mx=b
```

Out[14]:
```
array([-0.95004935])
```

In [15]:
```
lreg.intercept_ #eqauals b in the equation #note the trailing _ is only available after you call fit.
```

Out[15]:
```
34.55384087938311
```

In [17]:
```
X[:1]*lreg.coef_[0] + lreg.intercept_ #manually calculating the predicition
```

Out[17]:

	LSTAT
0	29.822595

In [19]:
```
lreg.predict(X[:1])
```

Out[19]:
```
array([29.8225951])
```

In [26]:
```
Z = df[['LSTAT', 'RM']]
y = df['PRICE']
```

In [27]:
```
lreg.fit(Z,y)
```

Out[27]:
```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None, normalize=False)
```

In [37]:
```
Z[:1]
```

Out[37]:

	LSTAT	RM
0	4.98	6.575

In [ ]:
```
#W= (-0.64*LSTAT) + (5.094*RM) + -1.35 # Linear combination of your va
riables
#W= (-.64*4.98) + (5.094*6.575) + -1.35
#W= -3.198 + 33.498 + -1.35
#w=28.94
```

In [35]:
```
lreg.coef_
```

Out[35]:
```
array([-0.64235833,  5.09478798])
```

In [38]:
```
Z[:1]*lreg.coef_
```

Out[38]:

	LSTAT	RM
0	-3.198945	33.498231

In [29]:
```
lreg.intercept_
```

Out[29]:
```
-1.3582728118744605
```

In [32]:
```
(Z[:1]*lreg.coef_).sum(axis=1) + lreg.intercept_
```

Out[32]:
```
0    28.941014
dtype: float64
```

In [33]:
```
lreg.predict(Z[:1])
```

Out[33]:
```
array([28.94101368])
```

In [39]:
```
lreg.predict(Z)
```

Out[39]:
```
array([28.94101368, 25.48420566, 32.65907477, 32.40652   , 31.630406
99,
       28.05452701, 21.28707846, 17.78559653,  8.10469338, 18.246506
73,
       17.99496223, 20.73221309, 18.5534842 , 23.64474107, 23.108958
23,
       22.9239452 , 24.65257604, 19.73611045, 18.9297215 , 20.573775
96,
       13.51732408, 20.14832175, 17.90896697, 15.48764606, 18.352810
36,
       16.56210901, 18.74440281, 18.34995811, 23.51018847, 24.948889
35,
       13.23095259, 21.20092715, 11.15596625, 15.89983805, 16.633986
22,
       22.65107562, 21.07107521, 22.81275431, 22.53014238, 29.466865
94,
```

```
       33.15564849, 30.0244275 , 26.33937234, 25.50630935, 23.427473
37,
       21.03183392, 19.03080004, 17.28696205,  6.35742724, 16.776524
46,
       20.38222834, 23.73891662, 28.42223975, 23.78518476, 19.132935
49,
       32.4841017 , 27.4553513 , 30.83048667, 25.54262118, 22.915991
73,
       19.44389291, 19.76157796, 27.21060683, 26.99027936, 29.664116
44,
       27.68813019, 21.54751591, 23.38578845, 18.73350058, 22.978224
72,
       27.01833368, 22.66525802, 25.99579831, 25.61529631, 26.246142
71,
       24.92488095, 22.94287168, 23.32670532, 22.46574406, 22.723050
97,
       29.51629037, 27.72630168, 26.43243306, 25.2371736 , 25.012840
44,
       28.2255716 , 21.02614874, 24.4054201 , 30.80793576, 31.046288
82,
       25.67580476, 26.00650589, 26.22070738, 26.2964101 , 23.676482
54,
       28.12301466, 22.75656203, 37.04724285, 36.18974997, 32.448476
79,
       26.86335045, 28.26259609, 24.44557513, 21.27514504, 22.141006
43,
       17.87168992, 16.38850335, 20.80666424, 23.74364784, 20.388489
45,
       21.85328041, 26.32686783, 18.35457994, 18.70127166, 23.791886
65,
       18.72006301, 22.37314344, 22.7011548 , 18.68527463, 19.097460
2 ,
       19.31744339, 20.06438082, 17.49427343, 12.15448035, 17.301326
23,
       19.62580113,  9.72808395, 16.60421542, 21.52041395, 15.580195
21,
       23.45015298, 22.9960428 , 23.96269155, 18.64893707, 16.853198
85,
       20.02281172, 18.05910154, 22.15148074, 14.78682073, 18.121833
23,
       14.57757085,  2.1089178 ,  8.94081614,  9.5289207 ,  4.806797
07,
       12.01521584, 16.6199277 ,  4.76981119,  6.87170965, 13.378669
27,
       20.77476672, 17.64344278, 16.39142155, 17.58503369, 20.155224
73,
       20.33664069, 15.13378593, 31.06641541, 25.41634701, 27.061768
88,
       26.95118125, 35.68531448, 37.15793504, 39.17794689, 20.989565
04,
```

```
       23.42349342, 36.66157528, 20.78556599, 23.70551495, 23.987063
52,
       19.30437583, 20.87150978, 17.59364191, 25.52296755, 22.299755
65,
       28.56843941, 22.81810809, 26.77487939, 29.14685309, 30.965861
31,
       33.34652688, 23.87381831, 31.99876804, 28.43022539, 18.212749
54,
       21.54294556, 35.68051731, 28.90362563, 29.11400321, 31.785467
43,
       30.77957096, 29.96284283, 33.36854692, 30.05522306, 29.474177
53,
       36.85537831, 33.14662523, 29.31968013, 31.44880281, 31.248719
37,
       32.13454487, 25.26308832, 35.41532933, 36.20371198, 37.723261
85,
       21.67268811, 23.82468505, 16.50389459, 20.11954835, 11.036220
24,
       17.91313514, 10.7702086 , 17.93015692, 25.09567941,  7.233030
98,
       24.05457308, 19.96157774, 26.25685672, 17.44974913, 24.366048
5 ,
       27.81829904, 16.26099047, 27.31015547, 27.47711073, 38.095881
16,
       40.11963326, 37.593241  , 31.05029451, 35.28222297, 29.607510
72,
       21.63017953, 33.03191447, 39.53034953, 38.12112827, 27.738286
58,
       22.65974818, 26.2971678 , 33.09082226, 27.57564911, 27.563715
69,
       26.47044207, 21.72921661, 23.82712868, 27.87886702, 19.107397
21,
       15.34007899, 23.87668985, 23.84194009, 25.30124695, 28.654642
19,
       27.90170267, 29.13590581, 31.81864228, 38.44563265, 25.540397
94,
       22.63688679, 34.6205424 , 39.69788713, 30.99783605, 29.067994
07,
       29.17926861, 32.29101132, 37.63141893, 28.74470749, 30.151666
9 ,
       20.25610379, 24.87609035, 36.13403709, 34.66994109, 20.034680
79,
       20.12614728, 26.20006279, 26.98602111, 33.59902374, 30.804779
47,
       31.6471762 , 31.77928355, 30.7465393 , 27.04758648, 30.231985
02,
       36.06770189, 31.19378511, 35.65788274, 36.97788005, 29.711071
5 ,
       26.23178496, 22.07656307, 25.68882728, 25.93338997, 25.980182
55,
```

```
        31.4580143 , 32.77247603, 29.40108735, 24.34605866, 22.575781
51,
        28.63713459, 27.26046561, 17.9655936 , 27.77563603, 31.469350
88,
        29.74890034, 26.11397583, 26.16312839, 31.09167539, 31.056069
79,
        26.61258457, 32.28899561, 28.69897184, 29.52933863, 22.663488
44,
        15.85869849, 25.99071639, 21.79919554, 25.49103786, 26.138114
54,
        20.32037179, 16.99787506, 17.86059947, 24.50183176, 21.608944
54,
        26.75585477, 26.71309362, 24.47318223, 20.18149016, 27.393559
1 ,
        28.14312838, 26.84952519, 21.4175594 , 22.13363049, 26.192109
32,
        24.10506731, 19.72807333, 24.33872776, 27.17181276, 26.453920
61,
        24.25367199, 22.24792619, 21.89219835, 24.04457816, 22.877463
1 ,
        23.07991295, 32.00653973, 26.40524101, 28.14429469, 30.702027
87,
        22.51774887, 20.55210662, 27.75396669, 28.59791904, 30.210970
42,
        27.86559837, 28.63379008, 23.61548826, 30.02884824, 22.322526
95,
        25.30645274, 18.98504346, 22.69880125, 22.48321949, 21.642391
25,
        26.23420929, 21.37418212, 19.41434893, 18.80265585, 39.975890
1 ,
        12.21067586, 14.93414327,  9.76025658, 21.8687353 , 30.294198
7 ,
        32.48537902, 24.18925437, 22.86946459,  1.30195776, -4.666386
08,
        27.26661571, 17.58856481, 19.61202573, 15.92900559, 16.356028
29,
        23.08722293, 18.44620086, 11.68681678, 10.98863617,  1.220325
33,
         5.7358631 ,  4.1767872 ,  3.566624  ,  3.83528036, 12.709463
15,
        16.75749984, 17.41964693,  7.80331745, 20.44917324, 18.132185
29,
        20.61292555, 18.83136329, 15.12569572,  6.77386462,  9.204947
2 ,
        11.94829024, 17.90524602, 18.22245167, 13.19432385,  9.233228
34,
        12.83401278,  4.73131635, 19.42149161, 10.30089127, 20.845366
61,
        21.47817885, 18.92688695,  0.14255003, 12.0068039 , -2.089337
11,
```

```
        12.76108347, 16.62815786,  8.55205663, 15.74595036, 18.801331
87,
        21.65619078, 19.15599797, 18.35983721, 14.77469265, 15.971353
31,
        13.01347737, 18.30140239, 20.91255925, 16.37019782, 15.678485
84,
        19.6522403 , 20.80637788, 23.64816923, 21.01273047, 20.525595
06,
        17.4673983 , 19.96458442, 12.99448493,  7.02626334, 12.612940
49,
        14.08066091, 18.74010433, 19.6694889 , 19.57297267, 13.184911
28,
        16.14520992, 19.52022303, 19.92887586, 18.50703427, 18.931042
25,
        21.82372195, 21.15852796, 19.63007869, 25.55441624, 20.900936
31,
        20.23926219, 16.87694825, 18.00283848, 20.31851051, 20.180465
41,
        22.23437762, 21.71554578, 21.83890979, 25.21421407, 21.783288
6 ,
        18.90608793, 17.94945984, 15.53065977, 17.18748418, 18.267041
93,
        19.59726742, 22.11000978, 22.21261176, 26.71344914, 14.638761
4 ,
        14.55497489, 19.67707879,  9.66333655, 18.5712701 , 21.955843
78,
        23.54446528, 28.05969258, 30.11309322, 21.30452171, 19.984167
27,
        24.00387777, 20.16873308, 21.37144731, 14.82770934, 10.827580
06,
         5.52428703, 17.5164286 , 20.54835994, 20.00295862, 20.103791
02,
        16.22366838, 12.52317924, 19.10367626, 21.00798639, 17.314990
63,
        20.14301944, 26.02005928, 23.98921598, 30.56006716, 29.093234
75,
        24.30151506])
```

In [40]:

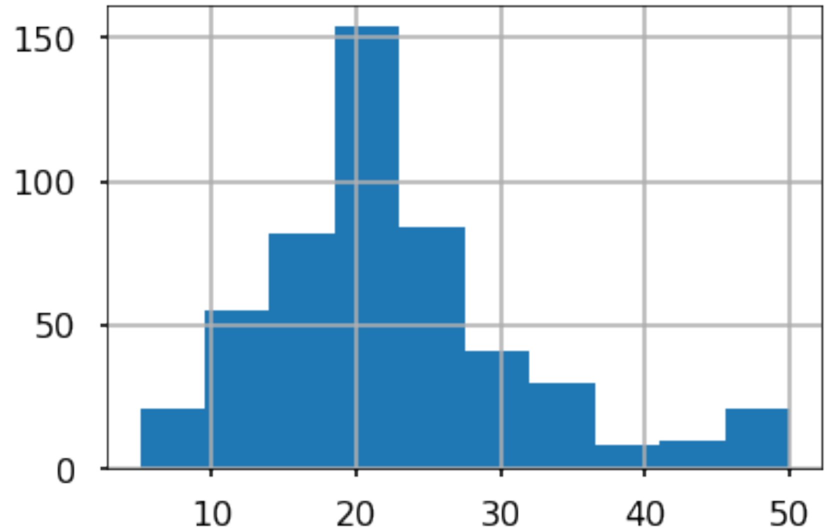
```
lreg.score(Z, y) #this the R Squared Value = how much of that number i
s acutally explained in our model.
#63% of observations are included in this model. How much of the chang
e in y can be explained in this model
```

Out[40]:

```
0.6385616062603403
```

In [41]:
```
y.hist() df
```

Out[41]: <matplotlib.axes._subplots.AxesSubplot at 0x1a1bcb58d0>



In [44]:
```
df['PREDICTIONS'] = lreg.predict(X)
df.head()
```

Out[44]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LST
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	396.90	4
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	396.90	9
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	392.83	4
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	394.63	2
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	396.90	5

In [43]:
```
lreg.predict(X)
```

Out[43]:
```
array([28.94101368, 25.48420566, 32.65907477, 32.40652   , 31.630406
99,
       28.05452701, 21.28707846, 17.78559653,  8.10469338, 18.246506
73,
       17.99496223, 20.73221309, 18.5534842 , 23.64474107, 23.108958
23,
       22.9239452 , 24.65257604, 19.73611045, 18.9297215 , 20.573775
96,
       13.51732408, 20.14832175, 17.90896697, 15.48764606, 18.352810
36,
       16.56210901, 18.74440281, 18.34995811, 23.51018847, 24.948889
35,
       13.23095259, 21.20092715, 11.15596625, 15.89983805, 16.633986
```

```
22,
         22.65107562, 21.07107521, 22.81275431, 22.53014238, 29.466865
94,
         33.15564849, 30.0244275 , 26.33937234, 25.50630935, 23.427473
37,
         21.03183392, 19.03080004, 17.28696205,  6.35742724, 16.776524
46,
         20.38222834, 23.73891662, 28.42223975, 23.78518476, 19.132935
49,
         32.4841017 , 27.4553513 , 30.83048667, 25.54262118, 22.915991
73,
         19.44389291, 19.76157796, 27.21060683, 26.99027936, 29.664116
44,
         27.68813019, 21.54751591, 23.38578845, 18.73350058, 22.978224
72,
         27.01833368, 22.66525802, 25.99579831, 25.61529631, 26.246142
71,
         24.92488095, 22.94287168, 23.32670532, 22.46574406, 22.723050
97,
         29.51629037, 27.72630168, 26.43243306, 25.2371736 , 25.012840
44,
         28.2255716 , 21.02614874, 24.4054201 , 30.80793576, 31.046288
82,
         25.67580476, 26.00650589, 26.22070738, 26.2964101 , 23.676482
54,
         28.12301466, 22.75656203, 37.04724285, 36.18974997, 32.448476
79,
         26.86335045, 28.26259609, 24.44557513, 21.27514504, 22.141006
43,
         17.87168992, 16.38850335, 20.80666424, 23.74364784, 20.388489
45,
         21.85328041, 26.32686783, 18.35457994, 18.70127166, 23.791886
65,
         18.72006301, 22.37314344, 22.7011548 , 18.68527463, 19.097460
2 ,
         19.31744339, 20.06438082, 17.49427343, 12.15448035, 17.301326
23,
         19.62580113,  9.72808395, 16.60421542, 21.52041395, 15.580195
21,
         23.45015298, 22.9960428 , 23.96269155, 18.64893707, 16.853198
85,
         20.02281172, 18.05910154, 22.15148074, 14.78682073, 18.121833
23,
         14.57757085,  2.1089178 ,  8.94081614,  9.5289207 ,  4.806797
07,
         12.01521584, 16.6199277 ,  4.76981119,  6.87170965, 13.378669
27,
         20.77476672, 17.64344278, 16.39142155, 17.58503369, 20.155224
73,
         20.33664069, 15.13378593, 31.06641541, 25.41634701, 27.061768
```

```
88,
        26.95118125, 35.68531448, 37.15793504, 39.17794689, 20.989565
04,
        23.42349342, 36.66157528, 20.78556599, 23.70551495, 23.987063
52,
        19.30437583, 20.87150978, 17.59364191, 25.52296755, 22.299755
65,
        28.56843941, 22.81810809, 26.77487939, 29.14685309, 30.965861
31,
        33.34652688, 23.87381831, 31.99876804, 28.43022539, 18.212749
54,
        21.54294556, 35.68051731, 28.90362563, 29.11400321, 31.785467
43,
        30.77957096, 29.96284283, 33.36854692, 30.05522306, 29.474177
53,
        36.85537831, 33.14662523, 29.31968013, 31.44880281, 31.248719
37,
        32.13454487, 25.26308832, 35.41532933, 36.20371198, 37.723261
85,
        21.67268811, 23.82468505, 16.50389459, 20.11954835, 11.036220
24,
        17.91313514, 10.7702086 , 17.93015692, 25.09567941,  7.233030
98,
        24.05457308, 19.96157774, 26.25685672, 17.44974913, 24.366048
5 ,
        27.81829904, 16.26099047, 27.31015547, 27.47711073, 38.095881
16,
        40.11963326, 37.593241  , 31.05029451, 35.28222297, 29.607510
72,
        21.63017953, 33.03191447, 39.53034953, 38.12112827, 27.738286
58,
        22.65974818, 26.2971678 , 33.09082226, 27.57564911, 27.563715
69,
        26.47044207, 21.72921661, 23.82712868, 27.87886702, 19.107397
21,
        15.34007899, 23.87668985, 23.84194009, 25.30124695, 28.654642
19,
        27.90170267, 29.13590581, 31.81864228, 38.44563265, 25.540397
94,
        22.63688679, 34.6205424 , 39.69788713, 30.99783605, 29.067994
07,
        29.17926861, 32.29101132, 37.63141893, 28.74470749, 30.151666
9 ,
        20.25610379, 24.87609035, 36.13403709, 34.66994109, 20.034680
79,
        20.12614728, 26.20006279, 26.98602111, 33.59902374, 30.804779
47,
        31.6471762 , 31.77928355, 30.7465393 , 27.04758648, 30.231985
02,
        36.06770189, 31.19378511, 35.65788274, 36.97788005, 29.711071
```

```
5 ,
        26.23178496, 22.07656307, 25.68882728, 25.93338997, 25.980182
55,
        31.4580143 , 32.77247603, 29.40108735, 24.34605866, 22.575781
51,
        28.63713459, 27.26046561, 17.9655936 , 27.77563603, 31.469350
88,
        29.74890034, 26.11397583, 26.16312839, 31.09167539, 31.056069
79,
        26.61258457, 32.28899561, 28.69897184, 29.52933863, 22.663488
44,
        15.85869849, 25.99071639, 21.79919554, 25.49103786, 26.138114
54,
        20.32037179, 16.99787506, 17.86059947, 24.50183176, 21.608944
54,
        26.75585477, 26.71309362, 24.47318223, 20.18149016, 27.393559
1 ,
        28.14312838, 26.84952519, 21.4175594 , 22.13363049, 26.192109
32,
        24.10506731, 19.72807333, 24.33872776, 27.17181276, 26.453920
61,
        24.25367199, 22.24792619, 21.89219835, 24.04457816, 22.877463
1 ,
        23.07991295, 32.00653973, 26.40524101, 28.14429469, 30.702027
87,
        22.51774887, 20.55210662, 27.75396669, 28.59791904, 30.210970
42,
        27.86559837, 28.63379008, 23.61548826, 30.02884824, 22.322526
95,
        25.30645274, 18.98504346, 22.69880125, 22.48321949, 21.642391
25,
        26.23420929, 21.37418212, 19.41434893, 18.80265585, 39.975890
1 ,
        12.21067586, 14.93414327,  9.76025658, 21.8687353 , 30.294198
7 ,
        32.48537902, 24.18925437, 22.86946459,  1.30195776, -4.666386
08,
        27.26661571, 17.58856481, 19.61202573, 15.92900559, 16.356028
29,
        23.08722293, 18.44620086, 11.68681678, 10.98863617,  1.220325
33,
         5.7358631 ,  4.1767872 ,  3.566624  ,  3.83528036, 12.709463
15,
        16.75749984, 17.41964693,  7.80331745, 20.44917324, 18.132185
29,
        20.61292555, 18.83136329, 15.12569572,  6.77386462,  9.204947
2 ,
        11.94829024, 17.90524602, 18.22245167, 13.19432385,  9.233228
34,
        12.83401278,  4.73131635, 19.42149161, 10.30089127, 20.845366
```

```
61,
        21.47817885, 18.92688695,  0.14255003, 12.0068039 , -2.089337
11,
        12.76108347, 16.62815786,  8.55205663, 15.74595036, 18.801331
87,
        21.65619078, 19.15599797, 18.35983721, 14.77469265, 15.971353
31,
        13.01347737, 18.30140239, 20.91255925, 16.37019782, 15.678485
84,
        19.6522403 , 20.80637788, 23.64816923, 21.01273047, 20.525595
06,
        17.4673983 , 19.96458442, 12.99448493,  7.02626334, 12.612940
49,
        14.08066091, 18.74010433, 19.6694889 , 19.57297267, 13.184911
28,
        16.14520992, 19.52022303, 19.92887586, 18.50703427, 18.931042
25,
        21.82372195, 21.15852796, 19.63007869, 25.55441624, 20.900936
31,
        20.23926219, 16.87694825, 18.00283848, 20.31851051, 20.180465
41,
        22.23437762, 21.71554578, 21.83890979, 25.21421407, 21.783288
6 ,
        18.90608793, 17.94945984, 15.53065977, 17.18748418, 18.267041
93,
        19.59726742, 22.11000978, 22.21261176, 26.71344914, 14.638761
4 ,
        14.55497489, 19.67707879,  9.66333655, 18.5712701 , 21.955843
78,
        23.54446528, 28.05969258, 30.11309322, 21.30452171, 19.984167
27,
        24.00387777, 20.16873308, 21.37144731, 14.82770934, 10.827580
06,
         5.52428703, 17.5164286 , 20.54835994, 20.00295862, 20.103791
02,
        16.22366838, 12.52317924, 19.10367626, 21.00798639, 17.314990
63,
        20.14301944, 26.02005928, 23.98921598, 30.56006716, 29.093234
75,
        24.30151506])
```

In [50]:
```
naive_model_cost = np.sum((y - y.mean())**2)
```

In [54]:
```
model_cost = np.sum((y - df['PREDICTIONS'])**2)
```

In [55]:
```
model_cost, naive_model_cost
```

Out[55]:
```
(15439.309201313532, 42716.29541501976)
```

In [57]:

```
1 - (model_cost/naive_model_cost)
#Man. calculate the R squared value; highest R value is 1 and technica
lly no minimum; 0 means you are predicting as well as the average.
# The informational gain it has above and beyond the target value in t
his case it is y.
```

Out[57]:

```
0.6385616062603403
```

In [ ]: