LinearRegression_Multiple

April 24, 2018

0.1 LinearRegression Multiple

In [3]:
```
%matplotlib inline
from sklearn import linear_model
import pandas as pd
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="whitegrid", color_codes=True)
```

In [4]:
```
X_weight = [[700],[750],[760],[800],[820],[890],[900],[950],[970],[1000],[1090],[1100],[
# print(len(Y_weight))
X_height = [[10],[12],[14],[11],[15],[17],[13],[15],[17],[19],[13],[15],[14],[12],[10]]
# print(len(X_height))
Y_speed = [[100],[120],[140],[160],[170],[190],[220],[250],[300],[320],[350],[380],[400]
# print(len(X_speed))
```

In [5]:
```
for i in zip(X_weight,X_height,Y_speed):
    print(i[0], i[1], '>', i[2])
```

```
[700] [10] > [100]
[750] [12] > [120]
[760] [14] > [140]
[800] [11] > [160]
[820] [15] > [170]
[890] [17] > [190]
[900] [13] > [220]
[950] [15] > [250]
[970] [17] > [300]
[1000] [19] > [320]
[1090] [13] > [350]
[1100] [15] > [380]
[1200] [14] > [400]
[1240] [12] > [420]
[1290] [10] > [450]
```

In [6]:
```
x = []
x.append(X_weight)
```

```
        x.append(X_height)
        print(x)
```

```
[[[700], [750], [760], [800], [820], [890], [900], [950], [970], [1000], [1090], [1100], [1200],
```

In [7]:
```
        # add list to single dataframe
        df = pd.DataFrame(X_weight,columns=['X_weight'])
        df['X_height'] = pd.DataFrame(X_height)
        rs = pd.DataFrame(Y_speed,columns=['Y_speed'])
        print(df)
        print(rs)
```

```
    X_weight  X_height
0        700        10
1        750        12
2        760        14
3        800        11
4        820        15
5        890        17
6        900        13
7        950        15
8        970        17
9       1000        19
10      1090        13
11      1100        15
12      1200        14
13      1240        12
14      1290        10
    Y_speed
0       100
1       120
2       140
3       160
4       170
5       190
6       220
7       250
8       300
9       320
10      350
11      380
12      400
13      420
14      450
```

In [8]:
```
x_train, x_test, y_train, y_test = train_test_split(df, rs)
```

```
In [9]: lr = linear_model.LinearRegression()
        lr.fit(x_train,y_train)

Out[9]: LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)
```

In regression with multiple independent variables, the coefficient tells you how much the dependent variable is expected to increase when that independent variable increases by one, holding all the other independent variables constant. Remember to keep in mind the units which your variables are measured in.

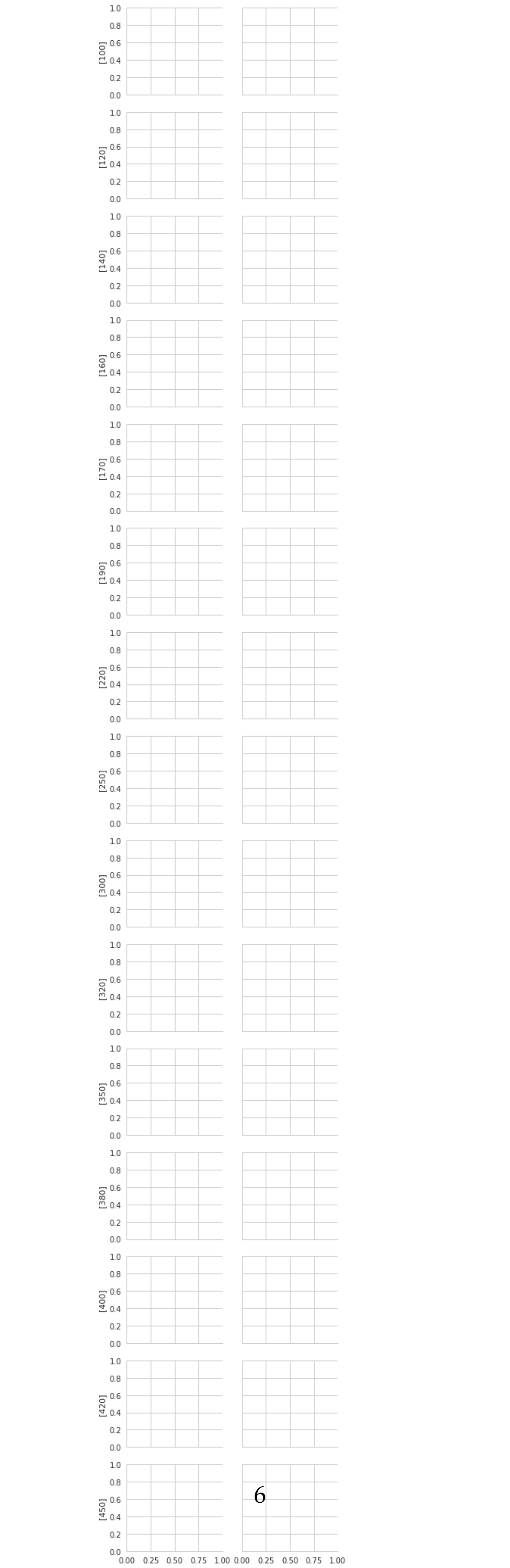
```
In [10]: lr.coef_

Out[10]: array([[0.61858104, 3.63642574]])
```

The intercept (often labeled the constant) is the expected mean value of Y when all X=0. Start with a regression equation with one predictor, X. If X sometimes = 0, the intercept is simply the expected mean value of Y at that value.

```
In [11]: lr.intercept_

Out[11]: array([-383.93583538])

In [12]: predicted = lr.predict(x_test)
         print('input: ')
         print(x_test)
         for i,j in zip(y_test.values, predicted):
             print('Expected: ',i,'Predicted: ',j)
         # predicted = lr.predict(x_test.tail(1))
         # print('input: ', x_test.tail(1).values)
         # print('Exptected: ',y_test.tail(1).values, 'Predicted:' ,predicted)

input:
    X_weight  X_height
11      1100        15
7        950        15
1        750        12
0        700        10
Expected:  [380] Predicted:  [351.04969483]
Expected:  [250] Predicted:  [258.26253881]
Expected:  [120] Predicted:  [123.63705357]
Expected:  [100] Predicted:  [85.43515009]


In [13]: lr.predict(x_test)

Out[13]: array([[351.04969483],
                [258.26253881],
                [123.63705357],
                [ 85.43515009]])
```

```
In [34]: x1 = [int(str(i).strip('[]')) for i in X_weight]
         x2 = [int(str(i).strip('[]')) for i in X_height]
         y = [int(str(i).strip('[]')) for i in Y_speed]
         ndf = pd.DataFrame({'weight': x1,'height':x2})
         ydf = pd.DataFrame({'speed': y})
         sns.pairplot(ndf,x_vars=[X_height,X_weight],y_vars=Y_speed,kind='reg')


        ---------------------------------------------------------------------------

        ValueError                                Traceback (most recent call last)

        <ipython-input-34-7a2d03b87788> in <module>()
          4 ndf = pd.DataFrame({'weight': x1,'height':x2})
          5 ydf = pd.DataFrame({'speed': y})
    ----> 6 sns.pairplot(ndf,x_vars=[X_height,X_weight],y_vars=Y_speed,kind='reg')


        ~/anaconda3/envs/py3/lib/python3.6/site-packages/seaborn/axisgrid.py in pairplot(data, h
       2039                     hue_order=hue_order, palette=palette,
       2040                     diag_sharey=diag_sharey,
    -> 2041                     size=size, aspect=aspect, dropna=dropna, **grid_kws)
       2042
       2043     # Add the markers here as PairGrid has figured out how many levels of the


        ~/anaconda3/envs/py3/lib/python3.6/site-packages/seaborn/axisgrid.py in __init__(self, d
       1268         if despine:
       1269             utils.despine(fig=fig)
    -> 1270         fig.tight_layout()
       1271
       1272     def map(self, func, **kwargs):


        ~/anaconda3/envs/py3/lib/python3.6/site-packages/matplotlib/figure.py in tight_layout(se
       2274             self, self.axes, subplotspec_list, renderer,
       2275             pad=pad, h_pad=h_pad, w_pad=w_pad, rect=rect)
    -> 2276         self.subplots_adjust(**kwargs)
       2277
       2278     def align_xlabels(self, axs=None):


        ~/anaconda3/envs/py3/lib/python3.6/site-packages/matplotlib/figure.py in subplots_adjust
       2086
       2087         """
    -> 2088         self.subplotpars.update(*args, **kwargs)
       2089         for ax in self.axes:
       2090             if not isinstance(ax, SubplotBase):
```

```
    ~/anaconda3/envs/py3/lib/python3.6/site-packages/matplotlib/figure.py in update(self, le
    239             if self.left >= self.right:
    240                 reset()
--> 241                 raise ValueError('left cannot be >= right')
    242
    243             if self.bottom >= self.top:


    ValueError: left cannot be >= right
```


[100]
[120]
[140]
[160]
[170]
[190]
[220]
[250]
[300]
[320]
[350]
[380]
[400]
[420]
[450]
0.0
0.2
0.4
0.6
0.8
1.0
0.00
0.25
0.50
0.75
1.00
[1090], [1100], [1200], [1240], [1290]]