Lab 07: Linear Regression: Building a Linear Model of a Data Set

In this lab we will explore Linear Regression as a first look at what might broadly be called Machine Learning, the process of summarizing, finding patterns in, and generally extracting useful information from data.

The approach we will take is to build a model for a joint random variable (X,Y), attempting to discover a linear relationship

\[ Y = \theta_1 X + \theta_0 \]

using the techniques of linear regression from lecture.
In [72]:
# Jupyter notebook specific
from IPython.display import Image
from IPython.core.display import HTML
from IPython.display import display_html
from IPython.display import display
from IPython.display import Math
from IPython.display import Latex
from IPython.display import HTML

# General useful imports
import numpy as np
from numpy import arange, linspace, mean, var, std
import matplotlib.pyplot as plt
import matplotlib.mlab as mlab
from numpy.random import seed, random, randint, uniform, choice, binomial, geometric, poisson, math
import collections
import pandas as pd

%matplotlib inline

# Numpy basic stats functions
# https://docs.scipy.org/doc/numpy-1.13.0/reference/routines.statistics.html
X = [1, 2, 3]
# mean of a list
mean(X)
# population variance
var(X)
# sample variance
var(X, ddof=1)
# population standard deviation
std(X)
# sample standard deviation
std(X, ddof=1)

# Scipy statistical functions
from scipy.stats import norm, expon
# https://docs.scipy.org/doc/scipy/reference/stats.html
# given random variable X (e.g., housing prices) normally distributed with mu = 60, sigma = 4

#a. Find P(X<50)
norm.cdf(x=50, loc=60, scale=40) # 0.4012936743170763

#b. Find P(X>50)
norm.sf(x=50, loc=60, scale=40) # 0.5987063256829237

c. Find P(60<X<80)
norm.cdf(x=80, loc=60, scale=40) - norm.cdf(x=60, loc=60, scale=40)

def rangeNormal(lo, hi, loc, scale):
    return norm.cdf(x=hi, loc=loc, scale=scale) - norm.cdf(x=lo, loc=loc, scale=scale)

#d. how much top most 5% expensive house cost at least? or find x where P(X>x) = 0.05
norm.isf(q=0.05, loc=60, scale=40)

#e. how much top most 5% cheapest house cost at least? or find x where P(X<x) = 0.05
norm.ppf(q=0.05, loc=60, scale=40)
Problem 1: Inferring a Model from the Data

Now suppose we don't know this model and are trying to determine the model from the data!

Of course, even with a perfect model, the problem is that the data is "noisy" and contains errors (due to the thermometers and our ability to read them). But we can attempt to construct a model using linear regression, which will fit the data with the least sum of squared errors.

We can generate this model and investigate the errors as we did in the "perfect" model in Problem 0.

Therefore we will assume we have a linear model of the relationship with unknown parameters; when we have quantities we are estimating, we will indicate the actual (usually unknown) parameter by ($\theta$) (possibly subscripted) and the estimate by ($\hat{\theta}$).

So we are trying to find a model

$$\hat{Y} = \hat{\theta}_0 + \hat{\theta}_1 X$$

which best fits the data, by determining $\hat{\theta}_0$ and $\hat{\theta}_1$. Every time you see a hat over a variable, you know that somewhere in calculating it, an estimate or prediction was made from data using the model.

Once we estimate these parameters, we can then calculate the errors, or residuals, from this model:

$$y_i = \hat{\theta}_0 + \hat{\theta}_1 x_i + e_i$$

Note carefully that we have $y_i$ on the left-hand side, not $\hat{y}_i$; the $e_i$ is what differentiates them, since $y_i = \hat{y}_i + e_i$.

The goal of Linear Regression, of course, is to determine the parameters $\hat{\theta}_1$ and $\hat{\theta}_0$ which minimize the sum of the squares of the errors, the Residual Sum of Squares (RSS):

$$RSS = \sum_{i=1}^{n} e_i^2.$$

Part A

In this part you will write functions to calculate various useful parameters for linear regression for the data in Problem 0.

For the mean values you may simply use the function mean from the numpy library; here is the notation we will use:

The mean of $X$:

- $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$

The mean of $Y$:

- $\bar{y} = \frac{1}{n} \sum_{i=1}^{n} y_i$

For these, you must write your own Python functions:

The sum of the squares of the deviations of $X$ from its mean value:

- $SSX = \sum_{i=1}^{n} (x_i - \bar{x})^2$

The sum of the squares of the deviations of $Y$ from its mean value:

```python
# Utility functions
# Round to 4 decimal places
def round4(x):
    return round(float(x)+0.00000000001,4)

def round4List(X):
    return [round(float(x)+0.00000000001,4) for x in X]
```
The sum of the products of the deviations of X and Y

\[ S_{xy} = \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}) \]

The correlation of X and Y:

\[ \rho = \frac{S_{xy}}{\sqrt{SS_x SS_y}} \]

The slope of the linear regression line:

\[ \hat{\theta}_1 = \frac{S_{xy}}{SS_x} \]

The y-intercept of the linear regression line:

\[ \hat{\theta}_0 = \bar{y} - \hat{\theta}_1 \bar{x} \]

---

Part B: Graphing the Predicted Linear Model and the Residuals

Now give a scatter plot of the data along with a line plot of the predicted linear model using linear regression, as well as a scatter plot of the residuals, as we did in Problem 0.

---

In [76]:
1 # Solution
2 3 X1 = X0
4 Y1 = Y0
5 6 print(X1)
7 print(Y1)
8 9 x_hat = mean(X1)
10 y_hat = mean(Y1)
11 12 # Code here
13 14 print("x_hat = " + str(x_hat))
15 print("y_hat = " + str(y_hat))
16 print("Sxy = " + str(Sxy))
17 print("SSx = " + str(SSx))
18 print("SSy = " + str(SSy))
19 print("rho = " + str(rho))
20 print("thetal_har = " + str(thetal_hat))
21 print("theta0_hat = " + str(theta0_hat))
22 23 print("Linear Regression Line: y = " + str(round4(theta1_hat)) + "$x + " + str(round4(theta0_hat))
24
[45.2, 47.1, 47.5, 49.6, 49.8, 52.0, 54.3, 58.6, 61.2, 63.2, 64.1]
x_hat = 53.8727272727
y_hat = 12.3236818182
Sxy = 262.567464545
SSx = 467.861818182
SSy = 148.860891516
rho = 0.994928337386
thetal_hat = 0.56120729314
theta0_hat = -17.9100856286
Linear Regression Line: y = 0.5612*x - 17.9101

---

In [77]:
1 print(X1)
2 print(Y1)

[45.2, 47.1, 47.5, 49.6, 49.8, 52.0, 54.3, 58.6, 61.2, 63.2, 64.1]
Part C: Displaying the Residuals

Graph the residuals, as in Problem 0.

```
In [78]:
# Solution
plt.figure(figsize=(12,8))
plt.title("Farenheit vs Celsius Measurements")
plt.xlabel("X = Farenheit")
plt.ylabel("Y = Celsius")

# draw the data
plt.scatter(X1,Y1)

# now draw the model
# C values predicted by linear model
# Code here

# plot them on the same graph
plt.plot(X1,Y1_hat,color='red')
plt.show()
print()
```
In [79]:

# Compute E1
# Code here
4 plt.figure(figsize=(12,8))
5 plt.title("Plot of Residuals")
6 plt.xlabel("X = Farenheit")
7 plt.ylabel("Yerror = Error in Celsius")

# draw the errors
10 plt.scatter(X1, E1)

# draw the baseline (errors of 0)
# Code here
14 plt.plot(X1, Yzero, color='red')
15 print()
16 plt.show()
17 print()

Analysis of Model

This model is obviously very similar to the model in Problem 0, except that it seems to correct for the positive bias observed in Problem 0. We note that the residuals do not seem to form any distribution or pattern.

Problem 2

For this problem you will use BU data about the relationship between high school GPA and BU GPA for a collection of 95 students.

Perform the same analysis as in the previous problems for this data, using the HS_GPA on the x axis and the BU_GPA data on the y axis.

Describe anything notable you see; is this a good model of the data?
In [80]:
    1 # Solution
    2
    3 studs = pd.read_csv("StudentData3.csv")
    4 #print(studs)
    5 X2 = list(studs['HS_GPA'])
    6 Y2 = list(studs['BU_GPA'])
    7
    8 x_hat = mean(X2)
    9 y_hat = mean(Y2)
    10
    11 # Code here
    12

<table>
<thead>
<tr>
<th>HS_GPA</th>
<th>GENDER</th>
<th>BU_GPA</th>
<th>SAT_TOTAL</th>
<th>Unnamed: 4</th>
<th>Unnamed: 5</th>
<th>Unnamed: 6</th>
</tr>
</thead>
<tbody>
<tr>
<td>4.0</td>
<td>0</td>
<td>3.93</td>
<td>1540</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000155</td>
</tr>
<tr>
<td>3.6</td>
<td>1</td>
<td>3.72</td>
<td>1600</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000299</td>
</tr>
<tr>
<td>3.6</td>
<td>0</td>
<td>3.35</td>
<td>1260</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000401</td>
</tr>
<tr>
<td>3.8</td>
<td>0</td>
<td>3.30</td>
<td>1220</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000452</td>
</tr>
<tr>
<td>3.4</td>
<td>0</td>
<td>2.08</td>
<td>1130</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000454</td>
</tr>
<tr>
<td>3.6</td>
<td>0</td>
<td>3.64</td>
<td>1270</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000483</td>
</tr>
<tr>
<td>3.9</td>
<td>0</td>
<td>2.78</td>
<td>1280</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000504</td>
</tr>
<tr>
<td>4.0</td>
<td>0</td>
<td>2.18</td>
<td>1300</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000565</td>
</tr>
<tr>
<td>3.5</td>
<td>1</td>
<td>3.56</td>
<td>1140</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000899</td>
</tr>
<tr>
<td>3.9</td>
<td>0</td>
<td>3.06</td>
<td>1190</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000894</td>
</tr>
<tr>
<td>3.4</td>
<td>1</td>
<td>3.34</td>
<td>1380</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000909</td>
</tr>
<tr>
<td>3.0</td>
<td>0</td>
<td>2.84</td>
<td>1310</td>
<td>NaN</td>
<td>NaN</td>
<td>0.000958</td>
</tr>
<tr>
<td>3.5</td>
<td>1</td>
<td>3.00</td>
<td>1240</td>
<td>NaN</td>
<td>NaN</td>
<td>0.001026</td>
</tr>
<tr>
<td>3.0</td>
<td>0</td>
<td>2.77</td>
<td>1170</td>
<td>NaN</td>
<td>NaN</td>
<td>0.001034</td>
</tr>
<tr>
<td>3.3</td>
<td>1</td>
<td>3.40</td>
<td>1370</td>
<td>NaN</td>
<td>NaN</td>
<td>0.001097</td>
</tr>
<tr>
<td>3.5</td>
<td>1</td>
<td>3.37</td>
<td>1240</td>
<td>NaN</td>
<td>NaN</td>
<td>0.001247</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
In [82]:
   plt.figure(figsize=(12,8))
   plt.title("Farenheit vs Celsius Measurements")
   plt.xlabel("X = Farenheit")
   plt.ylabel("Y = Celsius")
   
   # draw the data
   plt.scatter(X2,Y2)
   
   # now draw the model
   
   # C values predicted by linear model
   # Code here
Problem 3

For this problem, you must use the data below comparing airfare (X3) vs. distance (Y3) for 17 flights.

Perform the same analysis for this data.

Describe anything notable you see; is this a good model of the data?
In [85]:
X3 = [360, 360, 207, 111, 93, 141, 291, 183, 309, 300, 90, 162, 477, 84, 231, 54, 429]
Y3 = [1463, 1448, 681, 270, 190, 393, 1102, 578, 1204, 1138, 184, 502, 1828, 179, 818, 90, 1813]
x_hat = mean(X3)
y_hat = mean(Y3)

# Code here