CS251 Proj5

CS251 Project 5 - Single and Multiple Linear Regression

The main goal of this project was to implement the capability of running a linear regression on a data set and producing the result visually. The first step was to implement the single linear regression from Lab 5 work into our program and make it interact flawlessly with the user movements. Next, we were asked to create another linear regression function inside our analysis.py file to handle multiple linear regressions. We then also incorporated all of these capabilities into our GUI.

How to Run a Linear Regression:

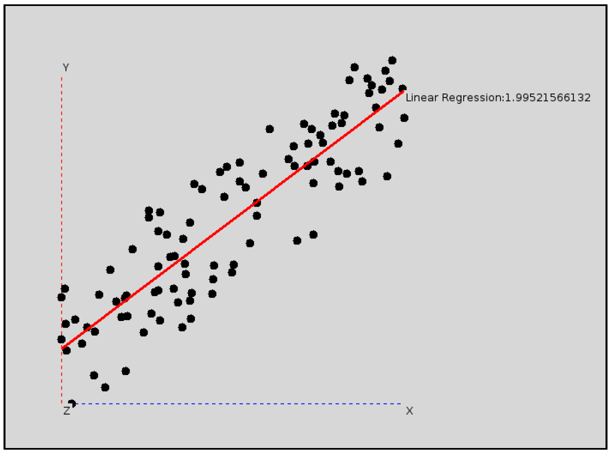
1. Open a data file by clicking on "File" tab and then "Open"

2. Choose to then run a "Linear Regression" under the "Command" tab

3. Select the X-axis and Y-axis values in the DialogBox

4. Now, the user will see a graphical display of a linear regression on the data. The user can interact with this new plot and data and move, zoom, and rotate the axes.

LINEAR REGRESSION ON "simple-data.csv":



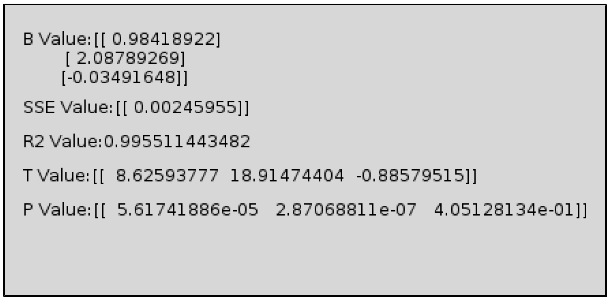
How to Run a Multiple Linear Regression:(Extension 2 = GUI functional)

1. Open a data file by clicking on "File" tab and then "Open"
2. Choose to then run a "Multiple LR" under the "Command" tab

3. Select the correct independent and dependent variables in the DialogBox(I allow the user to select multiple values in a single dialogbox!)

4. Now, the user will see a data legend pop up on the canvas of the display containing analysis numbers on the data. This can now be saved in a text file using my Extension 1.

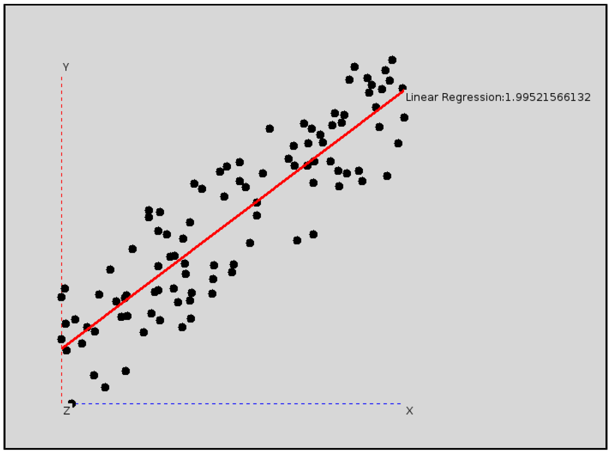
MULTI_LINEAR REGRESSION ON "data-clean.csv":



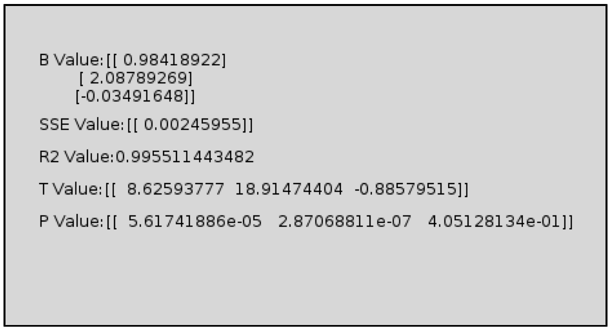
```
B Value:[[ 0.98418922]
       [ 2.08789269]
       [-0.03491648]]
SSE Value:[[ 0.00245955]]
R2 Value:0.995511443482
T Value:[[  8.62593777  18.91474404  -0.88579515]]
P Value:[[  5.61741886e-05   2.87068811e-07   4.05128134e-01]]
```

Required Screenshots:

simple-data.csv

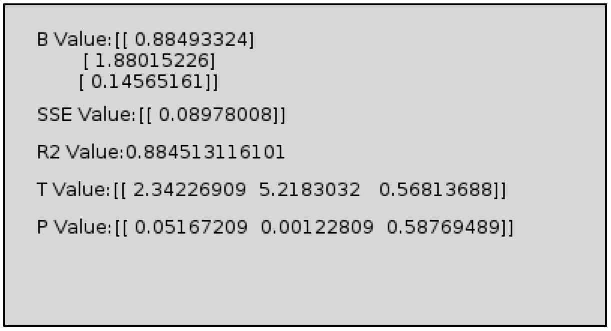


data-clean.csv:



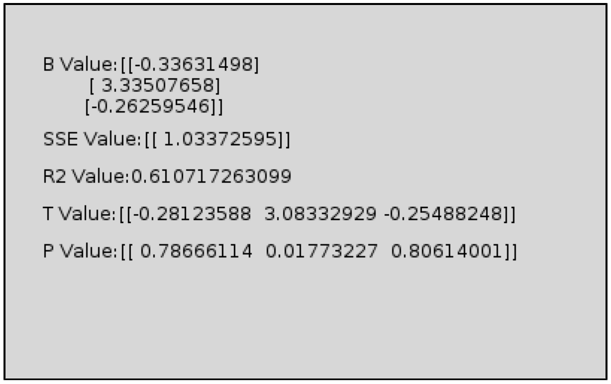
```
B Value:[[ 0.98418922]
      [ 2.08789269]
      [-0.03491648]]
SSE Value:[[ 0.00245955]]
R2 Value:0.995511443482
T Value:[[  8.62593777  18.91474404  -0.88579515]]
P Value:[[  5.61741886e-05   2.87068811e-07   4.05128134e-01]]
```

data-good.csv



```
B Value:[[ 0.88493324]
      [ 1.88015226]
      [ 0.14565161]]
SSE Value:[[ 0.08978008]]
R2 Value:0.884513116101
T Value:[[ 2.34226909  5.2183032   0.56813688]]
P Value:[[ 0.05167209  0.00122809  0.58769489]]
```

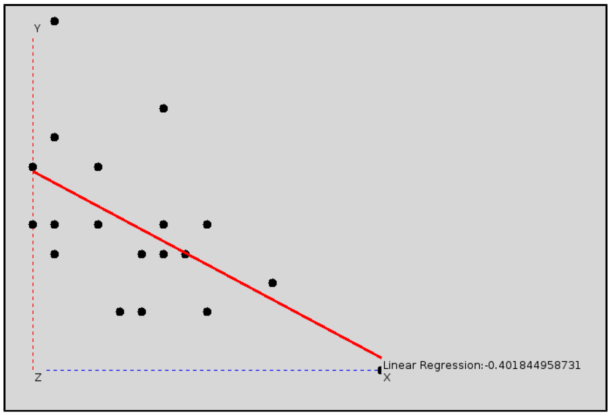
noisy-data.csv



```
B Value:[[-0.33631498]
      [ 3.33507658]
      [-0.26259546]]
SSE Value:[[ 1.03372595]]
R2 Value:0.610717263099
T Value:[[-0.28123588  3.08332929 -0.25488248]]
P Value:[[ 0.78666114  0.01773227  0.80614001]]
```

TASK 5. I ran some tests on my .csv data file on Baseball Pitching. This is the file I used: MLBPitching.csv

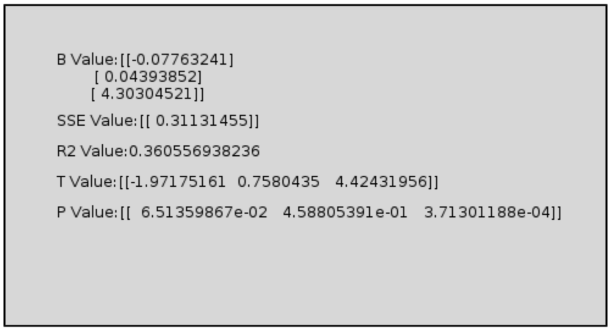
I ran a Linear Regression with Wins on the X and Loses on the Y. Here is my result:



Also, here are the numerical results to the LinReg: Slope, Intercept, R^2--
-0.401844958731, 15.0538922156, 0.129082693225

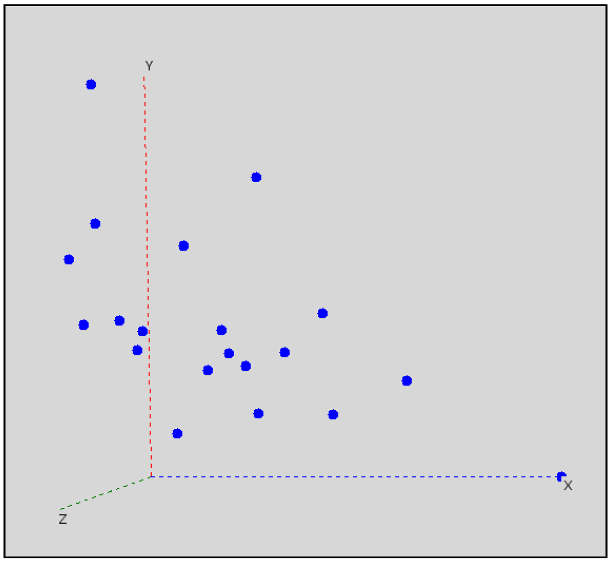
These results make total sense, because the more loses you have, the less wins you can possibly have and visa-versa. This is why the relationship has a negative slopes of about 1/2.

I ran a Multiple Linear Regression with Wins and Loses as the Independent Variables and ERA(earned runs against) as the Dependent Variable. Here is my result:



```
B Value:[[-0.07763241]
       [ 0.04393852]
      [ 4.30304521]]

SSE Value:[[ 0.31131455]]

R2 Value:0.360556938236

T Value:[[-1.97175161  0.7580435   4.42431956]]

P Value:[[  6.51359867e-02   4.58805391e-01   3.71301188e-04]]
```

Finally, here is the data plotted with W,L, and ERA in 3-D:

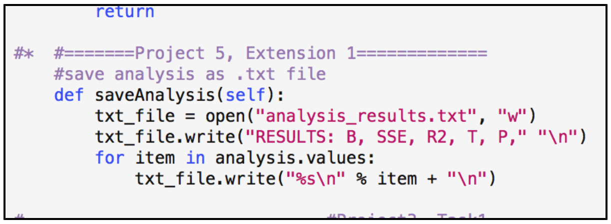


Extensions:

Extension 1. Saving the analysis results as a .txt file. I implemented a function called "saveAnalysis" that saves the results of my analysis.py class. I also, created a drop-down Command tab action to save the analysis results.

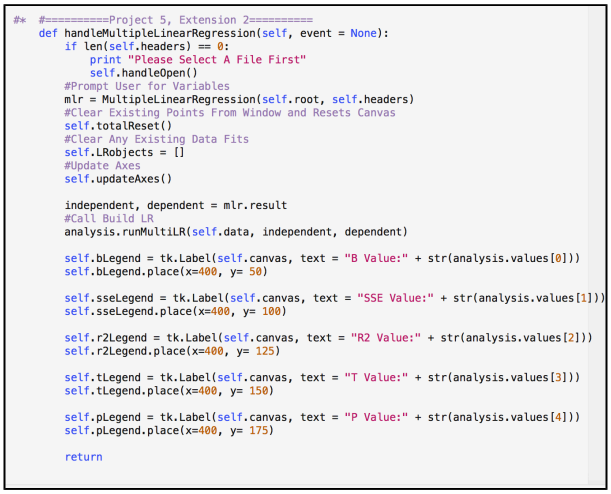
analysis_results.txt

analysis_results.txt



```
        return

#*  #=======Project 5, Extension 1=============
    #save analysis as .txt file
    def saveAnalysis(self):
        txt_file = open("analysis_results.txt", "w")
        txt_file.write("RESULTS: B, SSE, R2, T, P," "\n")
        for item in analysis.values:
            txt_file.write("%s\n" % item + "\n")
```

Extension 2. GUI Functionality! I added a method in my analysis.py class that runs a multiple linreg (runMultiLR())and then I created a method in my display.py class that handle a multiple linear regression (handleMultipleLR()). I also, created another DialogBox for the multiple linear regression and a drop-down Command tab action to run the multipleLR.



```
#*  #=========Project 5, Extension 2=========
    def handleMultipleLinearRegression(self, event = None):
        if len(self.headers) == 0:
            print "Please Select A File First"
            self.handleOpen()
        #Prompt User for Variables
        mlr = MultipleLinearRegression(self.root, self.headers)
        #Clear Existing Points From Window and Resets Canvas
        self.totalReset()
        #Clear Any Existing Data Fits
        self.LRobjects = []
        #Update Axes
        self.updateAxes()

        independent, dependent = mlr.result
        #Call Build LR
        analysis.runMultiLR(self.data, independent, dependent)

        self.bLegend = tk.Label(self.canvas, text = "B Value:" + str(analysis.values[0]))
        self.bLegend.place(x=400, y= 50)

        self.sseLegend = tk.Label(self.canvas, text = "SSE Value:" + str(analysis.values[1]))
        self.sseLegend.place(x=400, y= 100)

        self.r2Legend = tk.Label(self.canvas, text = "R2 Value:" + str(analysis.values[2]))
        self.r2Legend.place(x=400, y= 125)

        self.tLegend = tk.Label(self.canvas, text = "T Value:" + str(analysis.values[3]))
        self.tLegend.place(x=400, y= 150)

        self.pLegend = tk.Label(self.canvas, text = "P Value:" + str(analysis.values[4]))
        self.pLegend.place(x=400, y= 175)

        return
```

Wrap-Up:

What I learned. This week I was learned how to produce a linear regression on the data ran in my program. I gained knowledge on how to produce mathematically relevant reports on a set of data including linear regression, standard deviation, standard error, and variance/covariance. Also, I became more proficient with updating and improving my GUI's functionality and making it better looking. One thing to note, I definitely have a couple of small functional errors that I did not have time to take care of. I had to leave early for Spring Break and didn't clean up everything in my code... Sorry.

Who helped me. I received help this week from Stephanie and Bruce as well CS251 classmates Julia Saul and Steven Parrott. Also, I worked alongside Steven Parrott.