

# CS5542 Big Data Apps and Analytics

In Class Programming –9  
22<sup>nd</sup> October 2020

Submit ICP Feedback in Class. : [Lnk to Feed back Form](#)

## Variational Autoencoders:

Create a linear regression model in python using any dataset of your choice. For this model you can also create your own data. Find the best fit line in the data and calculate SSE (sum of square error) or MSE (Mean square error) , Y intercept, and Slope for the relationship in data. Explain your findings and understanding of these terms in detail in the report.

ICP Requirements:

- 1) Successfully executing the code with linear regression model and calculating following:
  - a. SSE or MSE
  - b. Y intercept
  - c. Slope

(75 points)

- 2) Detail explanation of each in report (5 points)
- 3) overall code quality (10 points)
- 4) Pdf Report quality, video explanation (10 points)

Submission Guidelines:

Same as previous ICPs.

## ICP Report:

### What I learned in the ICP:

I learned another method of consuming / linking data in CoLab. I learned how to compute the Mean Square Error, Slope, and Y intercept for linear regression by hand and by using the Regression Model object.

Looking at the data and the results of the linear regression, the data does suggest that the SAT score can predict how a student will do in college. However, it is not a strong, compact correlation. There are students who get a total score of over 1400 (SAT only had Math and Verbal at this time), and they had a GPA of around 2. There are students who had a total score of under 1000, and their GPA was around 4. So there is a good amount of error, variance, in the prediction, but in general the SAT does give a general indication on how a student will do in College, based upon this given data set.

### Description of what task I was performing:

I chose to use a data set of SAT (Standardize College entrance tests) and GPA scores as a model to predict how well a student will do in college, based up their GPA in college.

The first part of the lab, computed the values by hand, and the second part used the LinearRegression object to produce the answers.

The first item that need to be computed was the slope.

x is the SAT total score (sat\_sum)

y is the GPA (fy\_gpa)

x\_mean is the mean of x

y\_mean is the mean of y

n is the number of elements

$S_{xy} = \text{sum}(x * y) - n * x\_mean * y\_mean$

$S_{yy} = \text{sum}(x * x) - n * x\_mean * x\_mean$

**Slope =  $S_{xy} / S_{xx}$**

Now that we have the slope, we can compute the y intercept.

**y\_intercept =  $y\_mean - \text{slope} * x\_mean$**

Now that we have the y\_intercept we can compute the Mean Squared Error.

```
y_pred = slope * x + y_intercept
```

```
error = y - y_pred
```

```
se = sum(error^2)
```

**Mean Squared Error = se / n (mse = se/n)**

Using the LinearRegression object, I just had to plug in the values and the object did all of the math for me.

Started with calling the fit method and passing in the x and y values:

```
x = x.reshape(-1,1)
```

Reshaping x, so it can be used in the object.

```
regressionModel = LinearRegression()
```

```
regressionModel.fit(x,y)
```

Next we compute the expected value of y:

```
y_predicted = regressionModel.predict(x)
```

Next we can use the object to give us the Mean Squared Error:

```
mse = mean_squared_error(y,y_predicted)
```

Finally we can use the object to give the Slope and Y intercept:

```
slope is regressionModel.coef_[0]
```

and

```
Y intercept is regressionModel.intercept_
```

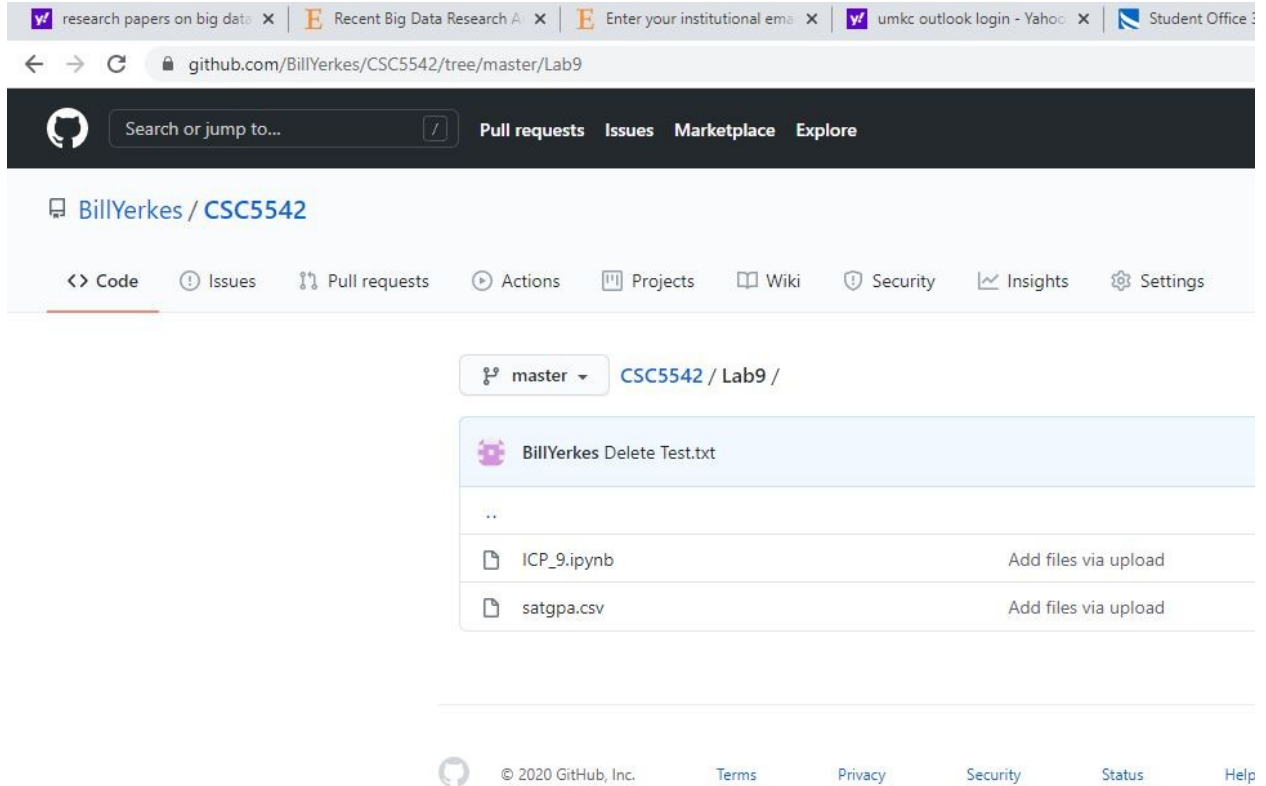
### **Challenges I faced:**

The biggest challenge I faced was finding a data set to work with.

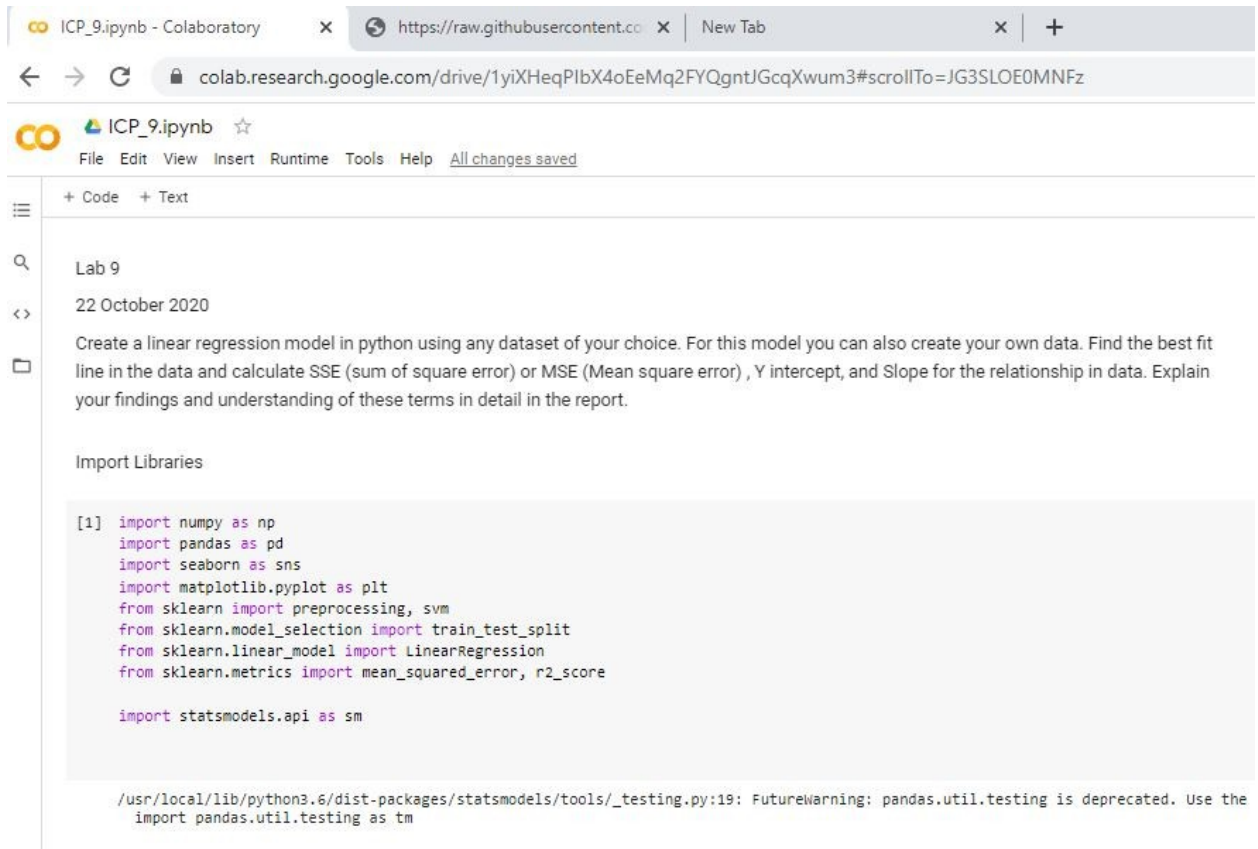
## Screen Shots

[GitHub:](#)

### Image of GitHub repository



## Lab 9, Libraries declaration:



ICP\_9.ipynb - Colaboratory

https://raw.githubusercontent.com/colab.research.google.com/drive/1yiXHeqPIbX4oEeMq2FYQgntJGcqXwum3#scrollTo=JG3SLOE0MNFz

ICP\_9.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

Lab 9

22 October 2020

Create a linear regression model in python using any dataset of your choice. For this model you can also create your own data. Find the best fit line in the data and calculate SSE (sum of square error) or MSE (Mean square error) , Y intercept, and Slope for the relationship in data. Explain your findings and understanding of these terms in detail in the report.

Import Libraries

```
[1] import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn import preprocessing, svm
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score

import statsmodels.api as sm
```

/usr/local/lib/python3.6/dist-packages/statsmodels/tools/\_testing.py:19: FutureWarning: pandas.util.testing is deprecated. Use the
import pandas.util.testing as tm

## Data for the Lab. Using SAT / GPA data. Determining if GPA can be predicted based upon SAT Score.

ICP\_9.ipynb - Colaboratory

https://raw.githubusercontent.com/BillyVerkes/CSC5542/master/Lab9/satgpa.csv

colab.research.google.com/drive/1yiXHeqPlbX4oEeMq2FYQgntJGcqXwum3

ICP\_9.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

Data for the Lab.

We are using a data set of SAT scores, High School GPA, and Future GPA.

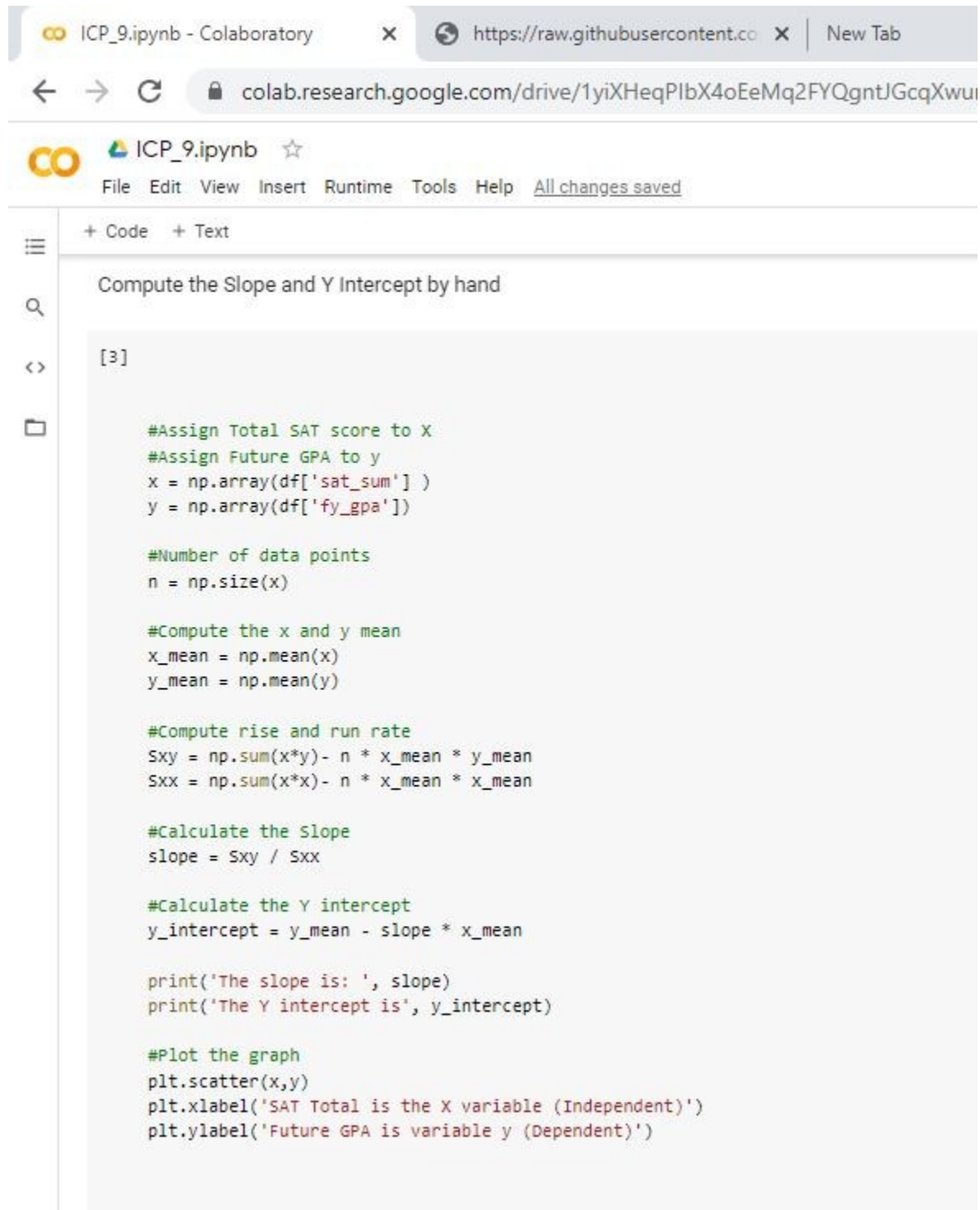
Using the SAT scores to predict the Future GPA.

```
[2] #Data is stored in GITHUB, url to the data
url = 'https://raw.githubusercontent.com/BillyVerkes/CSC5542/master/Lab9/satgpa.csv'

#Read the data and show the top 20 rows
df = pd.read_csv(url, error_bad_lines=False)
df.head(20)
```

	sex	sat_v	sat_m	sat_sum	hs_gpa	fy_gpa
0	1	65	62	127	3.40	3.18
1	2	58	64	122	4.00	3.33
2	2	56	60	116	3.75	3.25
3	1	42	53	95	3.75	2.42
4	1	55	52	107	4.00	2.63
5	2	55	56	111	4.00	2.91
6	1	57	65	122	2.80	2.83
7	1	53	62	115	3.80	2.51
8	2	67	77	144	4.00	3.82
9	1	41	44	85	2.60	2.54
10	1	58	70	128	3.75	3.38
11	2	45	57	102	3.75	3.02
12	2	43	45	88	2.80	2.60
13	2	50	58	108	3.75	3.81
14	1	54	66	120	3.50	3.02
15	2	55	57	112	3.75	3.48
16	2	44	53	97	3.20	2.15
17	1	53	68	121	4.00	3.80
18	2	54	49	103	3.75	3.13
19	2	47	55	102	4.00	3.40

## Computing the Slope and Y Intercept by hand



```
[3]

#Assign Total SAT score to X
#Assign Future GPA to y
x = np.array(df['sat_sum'] )
y = np.array(df['fy_gpa'])

#Number of data points
n = np.size(x)

#Compute the x and y mean
x_mean = np.mean(x)
y_mean = np.mean(y)

#Compute rise and run rate
Sxy = np.sum(x*y)- n * x_mean * y_mean
Sxx = np.sum(x*x)- n * x_mean * x_mean

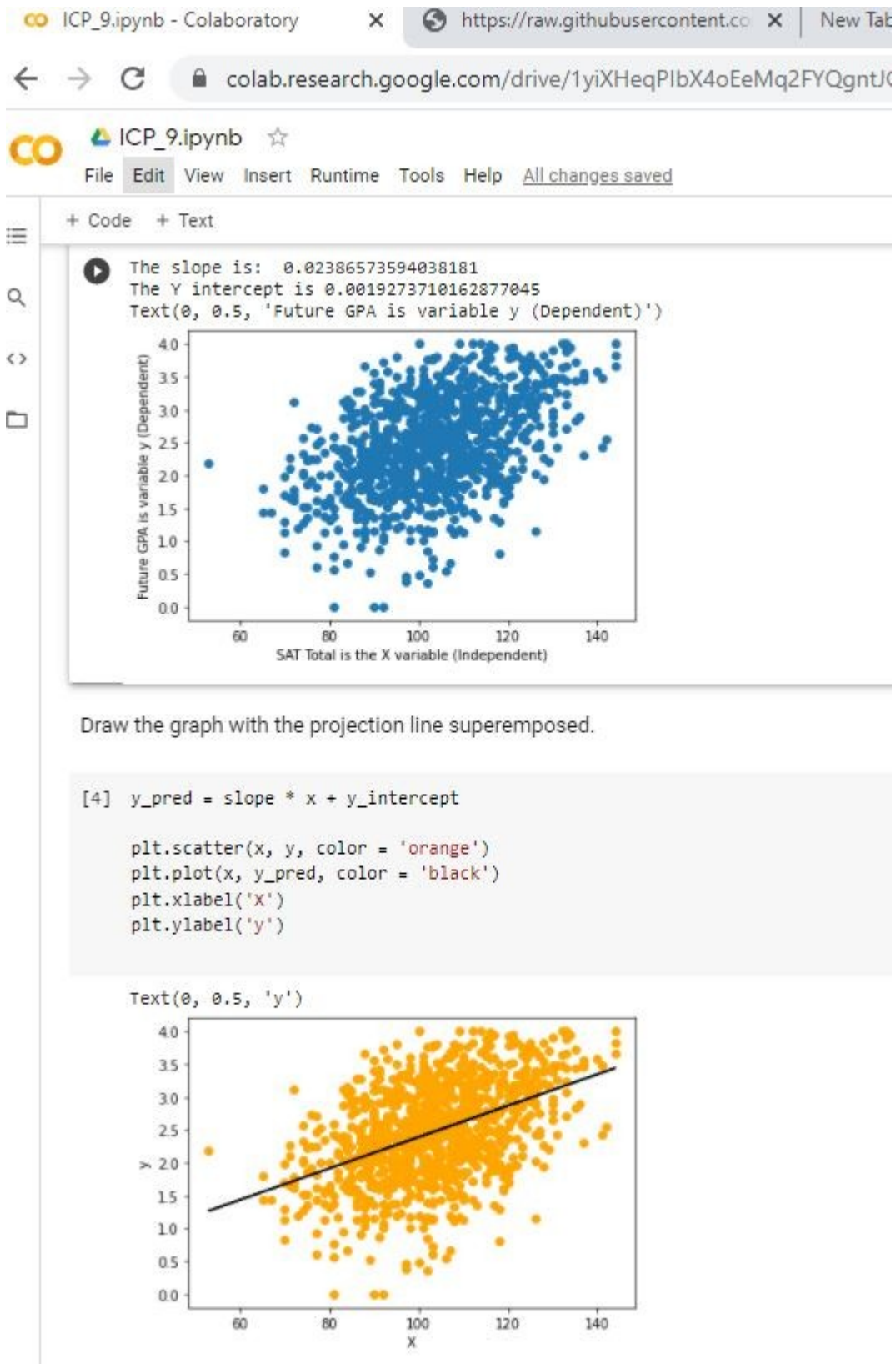
#Calculate the slope
slope = Sxy / Sxx

#Calculate the Y intercept
y_intercept = y_mean - slope * x_mean

print('The slope is: ', slope)
print('The Y intercept is', y_intercept)

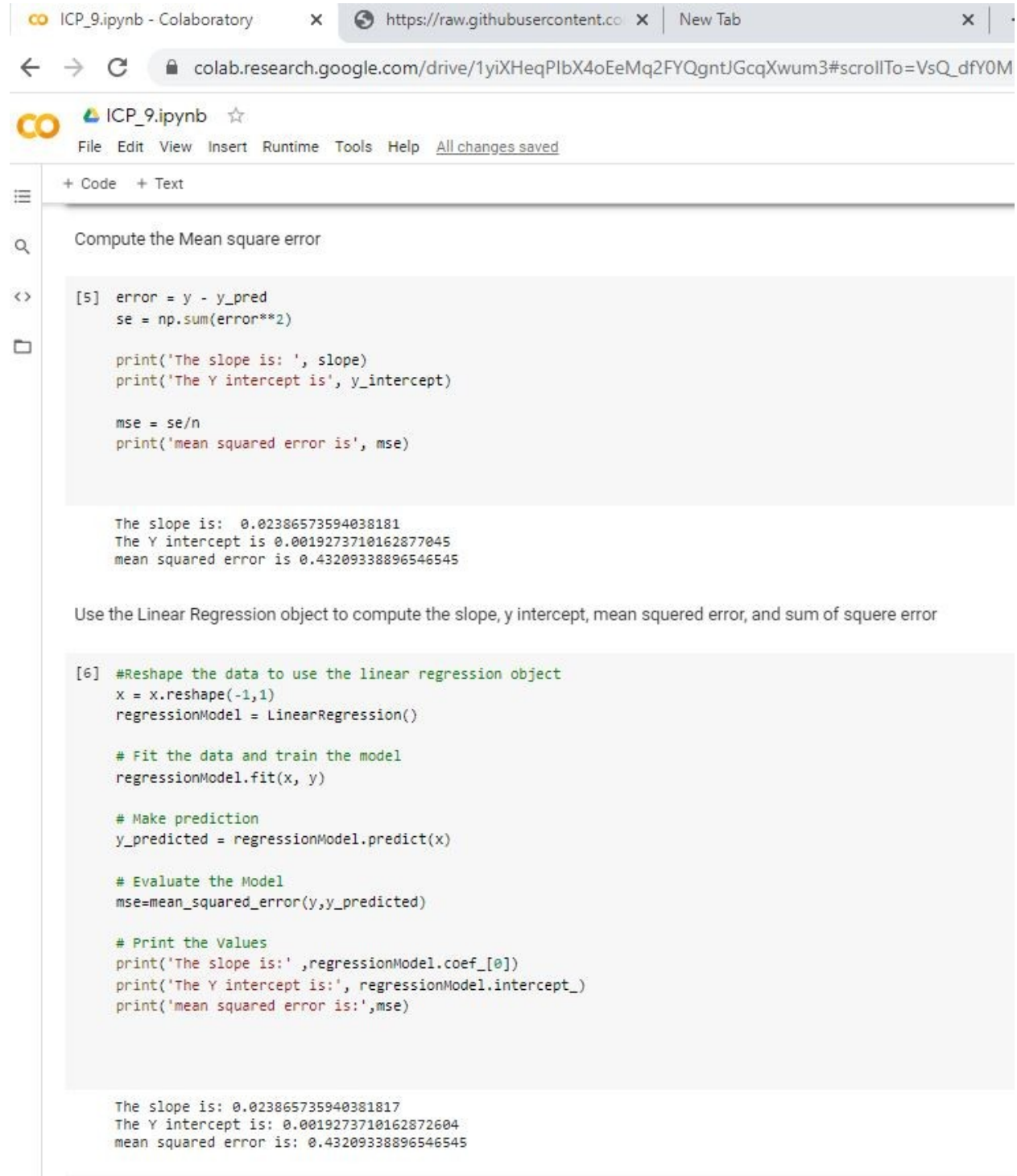
#Plot the graph
plt.scatter(x,y)
plt.xlabel('SAT Total is the X variable (Independent)')
plt.ylabel('Future GPA is variable y (Dependent)')
```

## Plotting the Data:





## Compute Mean Square Error and use Linear Regression Object to produce the same results:



ICP\_9.ipynb - Colaboratory

https://raw.githubusercontent.com/... New Tab

colab.research.google.com/drive/1yiXHeqPIbX4oEeMq2FYQgntJGcqXwum3#scrollTo=VsQ\_dfY0M

ICP\_9.ipynb

File Edit View Insert Runtime Tools Help All changes saved

+ Code + Text

Compute the Mean square error

```
[5] error = y - y_pred
     se = np.sum(error**2)

     print('The slope is: ', slope)
     print('The Y intercept is', y_intercept)

     mse = se/n
     print('mean squared error is', mse)
```

The slope is: 0.02386573594038181  
The Y intercept is 0.0019273710162877045  
mean squared error is 0.43209338896546545

Use the Linear Regression object to compute the slope, y intercept, mean squared error, and sum of square error

```
[6] #Reshape the data to use the linear regression object
     x = x.reshape(-1,1)
     regressionModel = LinearRegression()

     # Fit the data and train the model
     regressionModel.fit(x, y)

     # Make prediction
     y_predicted = regressionModel.predict(x)

     # Evaluate the Model
     mse=mean_squared_error(y,y_predicted)

     # Print the Values
     print('The slope is:',regressionModel.coef_[0])
     print('The Y intercept is:', regressionModel.intercept_)
     print('mean squared error is:',mse)
```

The slope is: 0.02386573594038181  
The Y intercept is: 0.0019273710162877045  
mean squared error is: 0.43209338896546545

[Video Link](#)

**Any in site about the data or the ICP in general**

I enjoyed the lab, going to try and figure out on my own how to add more dimensions to the model and see if that improve the prediction or not.