CAP4612 - Homework 1

- All the code you need to do this homework I have provided. You might have to do a little modification that is all. Do not overthink it.
- ***Important your results might be different than mine due to the samples that you are using. I provide you with numbers and figures are just to guide you.***
- If I ask you to make a comment on some of your calculations, there is no right or wrong answer that I'm looking for. I want to see how you are interpreting and understanding the material.
- Please note that this data is fictional.

0. On the top of your python script file put the following information:

```
# Name:    YOUR NAME
# ID:      YOUR PANTHER ID

# CERTIFICATION: I understand FIU's academic policies, and I certify that this work is my
#                own and that none of it is the work of any other person.
```

1. Load the COP4612_HW1.csv data into a dataframe and show the first 10 entries of the data frame. What you name the dataframe is up to you. Here's an example of what I'm looking for:

	salary	sex	pay_grade	position	age	year_at_company
0	51945.0300	M	GS10	FinancialAnalyst	49	19
1	120282.2700	M	GS15	Accountant	20	0
2	73131.8100	F	GS11	FinancialAnalyst	40	16
3	69548.8500	M	GS11	Developer	32	4
4	114453.9500	M	GS13	Accountant	57	21
5	100371.5900	F	GS10	Secretary	64	40
6	108991.5600	M	GS13	Developer	35	10
7	83882.5200	M	GS11	Developer	37	12
8	76271.7500	M	GS11	Developer	33	5
9	152449.0500	M	GS15	Accountant	56	14

2. Print out the features (columns) name of the dataframe.

```
Index(['salary', 'sex', 'pay_grade', 'position', 'age', 'year_at_company'], dtype='object')
```

3. Print out the features (columns) dtype information.

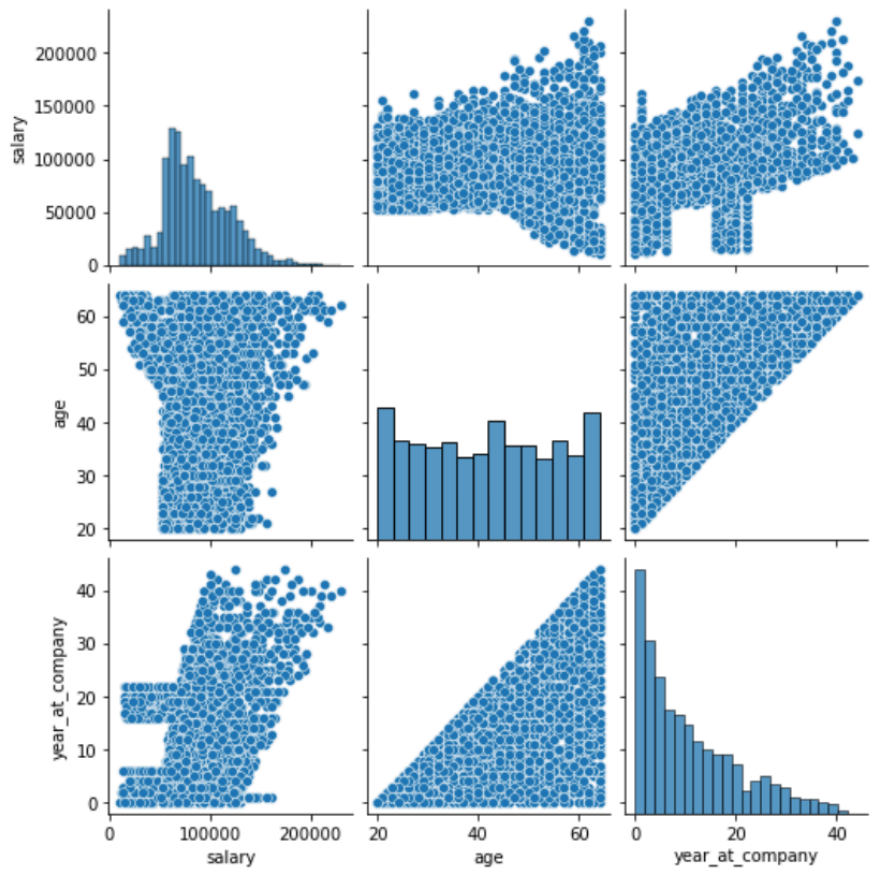
```
--------------------------------
Feature Info
--------------------------------
<class 'pandas.core.frame.DataFrame'>
Int64Index: 3000 entries, 0 to 2999
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   salary           3000 non-null   float64
 1   sex              3000 non-null   object
 2   pay_grade        3000 non-null   object
 3   position         3000 non-null   object
 4   age              3000 non-null   int64
 5   year_at_company  3000 non-null   int64
dtypes: float64(1), int64(2), object(3)
memory usage: 228.6+ KB
```

4. Print out the correlation of the features of the dataframe:

```
                 salary    age  year_at_company
salary           1.0000 0.0525           0.3904
age              0.0525 1.0000           0.6652
year_at_company  0.3904 0.6652           1.0000
```

5. Show a pairwise plot of the feature of the dataframe. Here's an example of what I'm looking for:

Hint use: seaborn



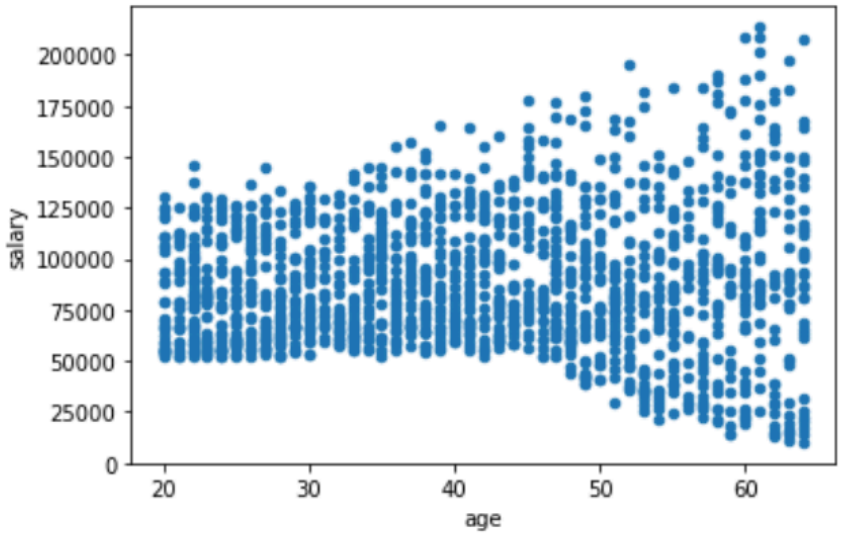
6. Create dummy variables for features: sex, position, paygrade, then print out the column names of the dataframe and show the first five entries of the dataframe.

```
Index(['salary', 'age', 'year_at_company', 'sex_F', 'sex_M',
       'position_Accountant', 'position_Developer', 'position_Engineer',
       'position_FinancialAnalyst', 'position_Secretary', 'pay_grade_GS10',
       'pay_grade_GS11', 'pay_grade_GS12', 'pay_grade_GS13', 'pay_grade_GS14',
       'pay_grade_GS15'],
      dtype='object')
```

	salary	age	year_at_company	sex_F	sex_M	position_Accountant	position_Developer	position_Engineer	position_FinancialAnalyst	position_Secretary
0	51945.0300	49	19	0	1	0	0	0	1	0
1	120282.2700	20	0	0	1	1	0	0	0	0
2	73131.8100	40	16	1	0	0	0	0	1	0
3	69548.8500	32	4	0	1	0	1	0	0	0
4	114453.9500	57	21	0	1	1	0	0	0	0

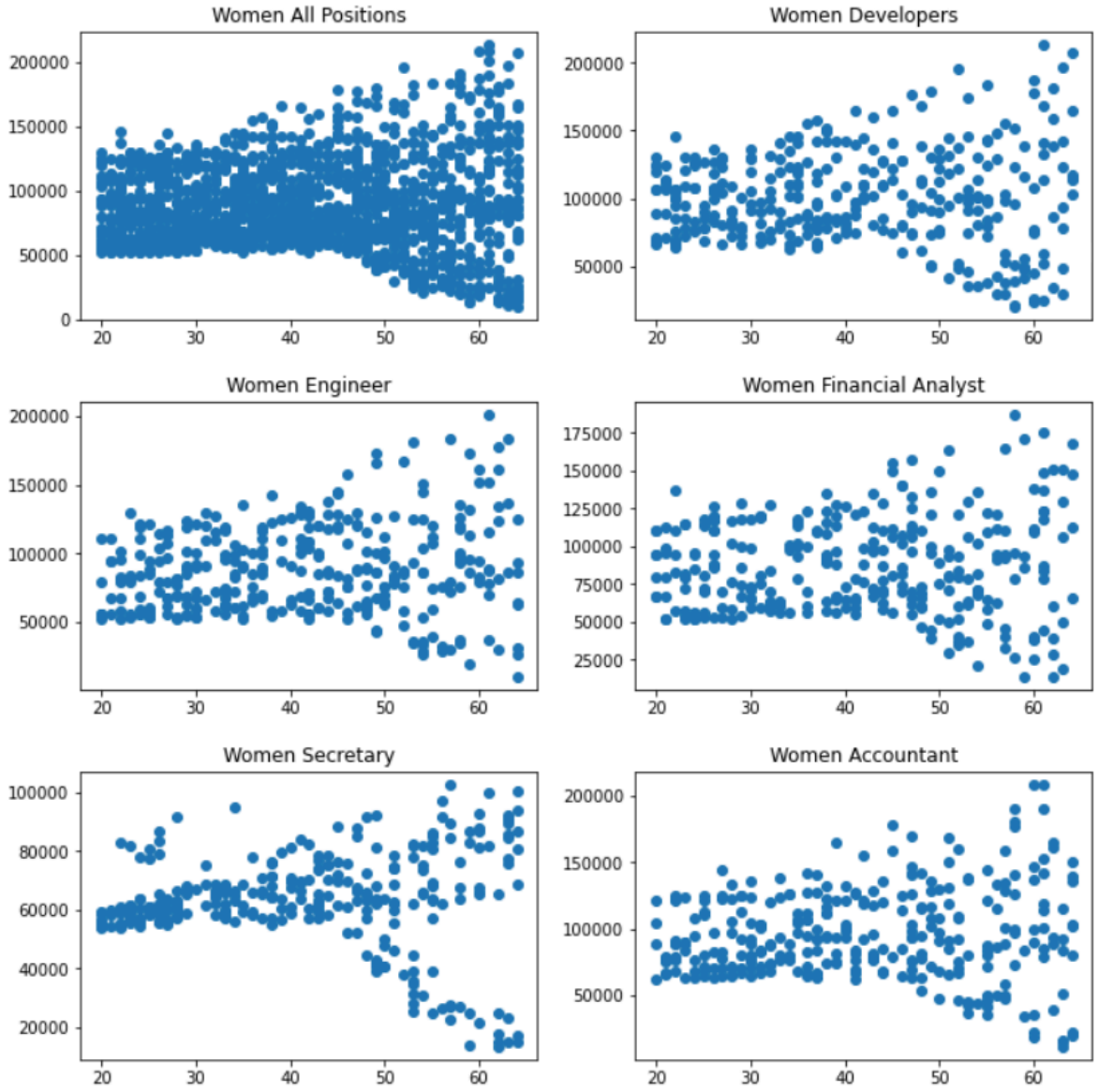
7. Plot the only women data with x-axis -> age and y-axis -> salary.

Hint: filter data on sex_F or sex_M (look at the data and think about this why you can use both) dummy variable



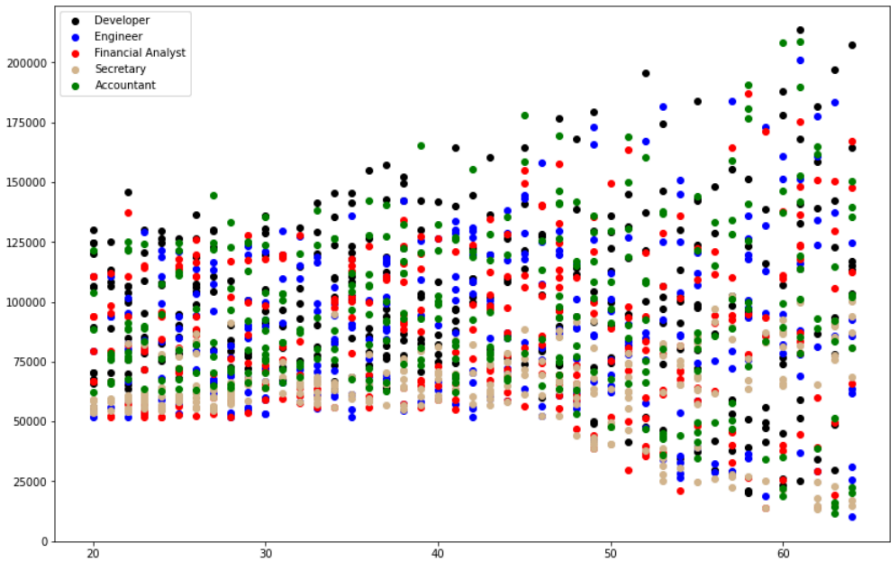
8. Subplot the only women data by position with x-axis -> age and y-axis -> salary. Write a 1-3 sentence comment about what you see in these graphs.

Hint: create a dataframe for each position filter on sex_F or sex_M (look at the data and think about this why you can use both) and positions …
data_women_developer, data_women_engineer ….



9. Plot the only women data by position with x-axis -> age and y-axis -> salary. Color code each position data. I will be checking your legend.

Give a comment on what you see in the diagram.



10. Create a dataframe that stores the following information shown below. This is only women data for each position. The "All" position is the aggregation of all positions together. Output the dataframe on the screen.

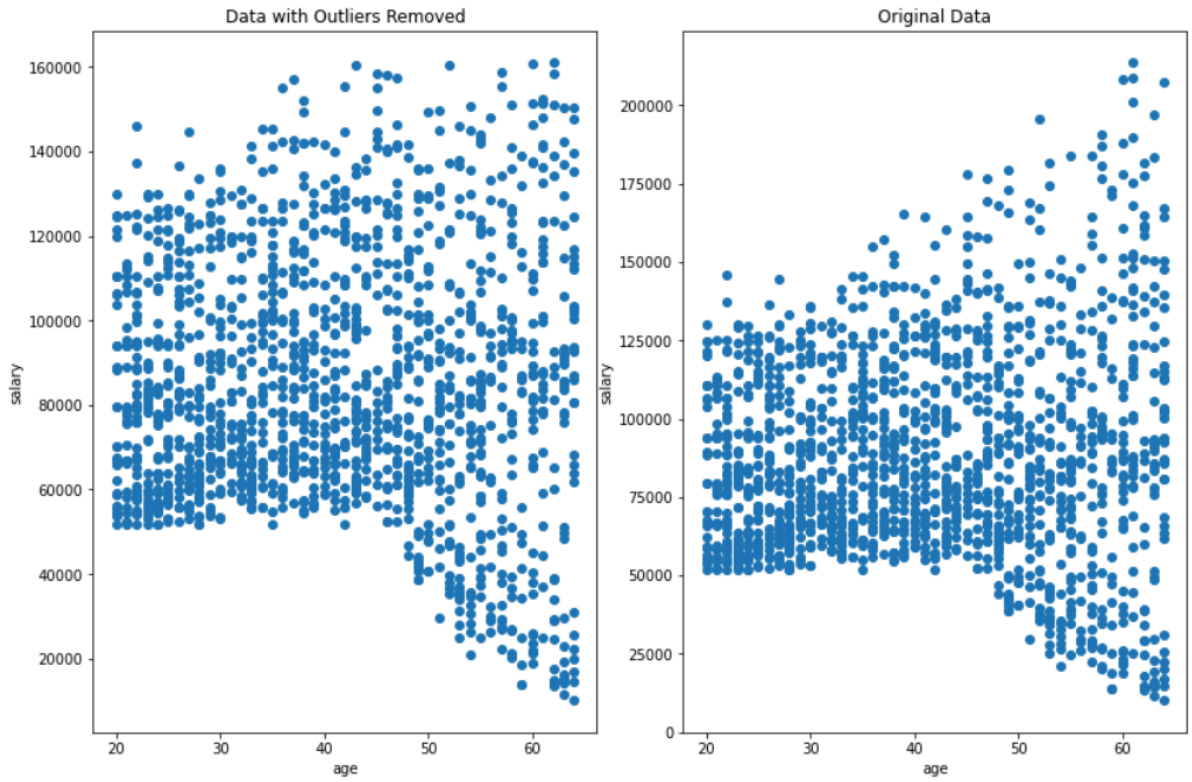
Hint:

```
women_stat_df = pd.DataFrame(columns=['Position','Count','Min Salary', 'Max Salary', 'Mean Salary','Std Dev Salary', 'Avg Age'])
position = 'All'
count = data_women.count()[0]
min_salary = round(data_women['salary'].min(),2)
max_salary = round(data_women['salary'].max(),2)
mean_salary = round(data_women['salary'].mean(),2)
std_dev_salary = data_women['salary'].std()
avg_age = data_women['age'].mean()
….
```

	Position	Count	Min Salary	Max Salary	Mean Salary	Std Dev Salary	Avg Age
0	All	1533	10086.0600	213617.0500	87549.9900	33495.5531	41
1	Developer	326	20152.5300	213617.0500	100963.2700	34717.4438	41
2	Engineer	329	10086.0600	201232.5800	88267.1800	32627.2545	41
3	Financial Analyst	287	13991.5400	186865.1900	87178.3300	32097.6775	42
4	Secretary	291	13411.3100	102608.6700	63296.3600	16671.1745	42
5	Accountant	300	11410.8800	208665.8800	96069.2700	34256.2358	42

11. Remove all outliers for all the category position shown in step 10 (women data only). For each category set the outlier remove range to 2.2. After you have removed all the outliers plot the all position category data with outliers and without outliers.

Hint: use the remove_outliers(df,columns,n_std) in notes.



12. Automation, for each category shown in step 10 (women data only) automate the model estimation with incremental train test size. Please note that the step size is 0.2 this is different than what was shown in the dataframe. I will be checking this. Output could each dataframe to the screen.

Important these calculations are done on the dataframes where all the outliers were removed in step 11.

Make a comment on the results of these data frame outputs. I am particularly interested in the Test_R_Score and Train_R_Score.

Why do you think we're getting these type of Test_R_Scores?
Why are the R_Train scores so low?

Remember your numbers are going to be different than mine.

```
Women Positions: All
```

	Train	Test	Test_R_Score	Test_RMSE	Train_R_Score	Train_RMSE	Model_Var	Model_Error	Avg_Salary
0	0.2000	0.8000	-0.0033	868743494.0613	0.0053	1024126905.1515	0.2121	759.3546	91979.5200
1	0.4000	0.6000	-0.0046	876600876.4949	0.0047	933042235.2036	0.6263	1201.5254	90803.1900
2	0.6000	0.4000	0.0012	981895722.9515	0.0001	842391046.0115	1.7484	436.6004	85781.8600
3	0.8000	0.2000	-0.0022	924021981.9197	0.0014	891151136.5970	4.1475	-1595.6908	88747.4100

```
Women Positions: Developer
```

	Train	Test	Test_R_Score	Test_RMSE	Train_R_Score	Train_RMSE	Model_Var	Model_Error	Avg_Salary
0	0.2000	0.8000	-0.0037	1065705114.4531	0.0077	696604495.5023	0.3795	3593.0701	102663.6800
1	0.4000	0.6000	-0.0342	991166967.5622	0.0363	1002786254.5315	0.6520	1860.5397	116118.0600
2	0.6000	0.4000	-0.0578	1105741297.4726	0.0109	918192039.4517	1.8016	7665.7574	104932.1100
3	0.8000	0.2000	-0.0145	1121690512.0695	0.0076	950299067.2857	4.6661	-5161.5888	107941.8800

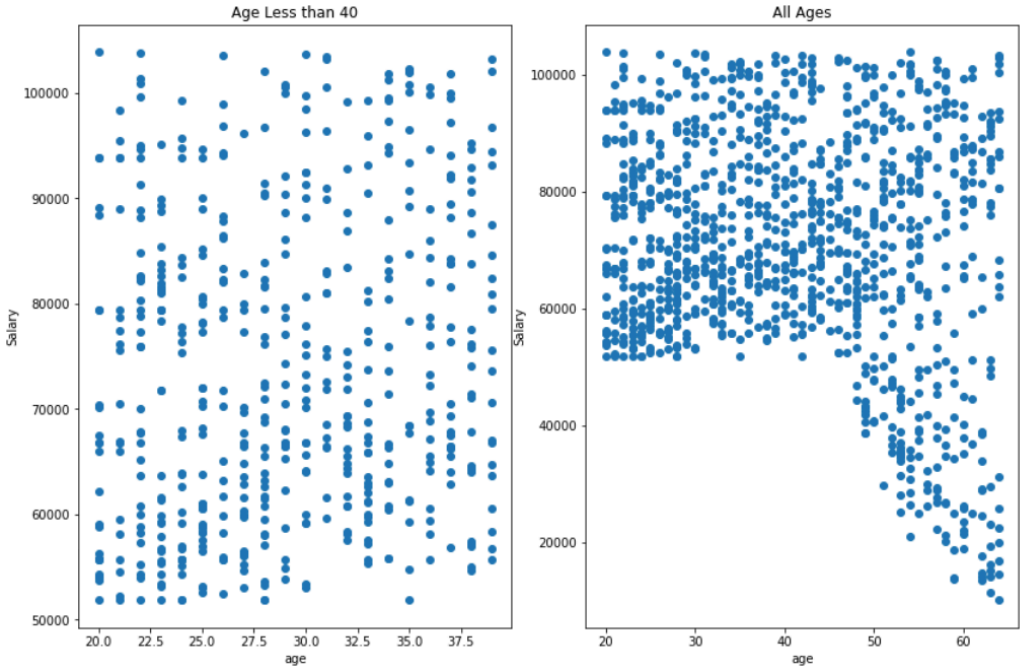
.
.
.

```
Women Positions: Engineer
```

	Train	Test	Test_R_Score	Test_RMSE	Train_R_Score	Train_RMSE	Model_Var	Model_Error	Avg_Salary
0	0.2000	0.8000	-0.0374	816359937.2272	0.0236	931476898.7051	0.2165	-3015.7942	72365.8200
1	0.4000	0.6000	-0.0059	729653028.7070	0.0062	961097143.1386	0.5048	493.3284	77388.3300
2	0.6000	0.4000	-0.0029	739162780.8929	0.0001	877485252.3846	1.2504	-1911.0733	84857.6200
3	0.8000	0.2000	-0.0890	1010286165.0952	0.0004	780156199.3167	5.1394	8520.2930	85378.0300

13. Filter the all position category data (women only) just that it only contains women that are of age less than 40 then plot the original data with the filtered data next to it, see below.

Give a one to two sentence comment regarding the images. How do you think this would affect the estimation of the linear regression model?



14. Use the filtered data that you created in step 13 (women only, position -> all, age < 40), and re-do an automation estimate similar to step 12. Print out the data frame showing automated dataframe.

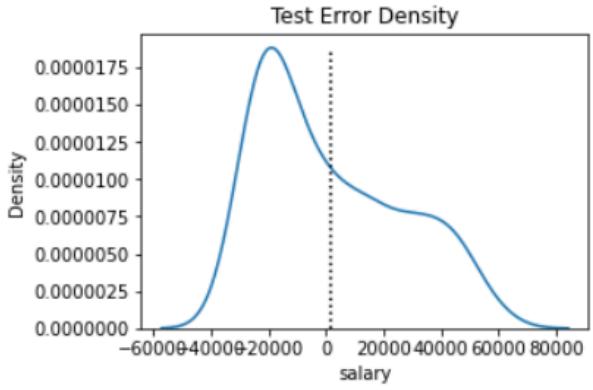
	Train	Test	Test_R_Score	Test_RMSE	Train_R_Score	Train_RMSE	Model_Var	Model_Error	Avg_Salary
0	0.2000	0.8000	0.0164	212806580.0779	0.0047	186666637.6181	0.2830	-193.9145	68876.5900
1	0.4000	0.6000	0.0232	213385830.1206	0.0127	196265647.5526	0.7237	-337.3061	65620.6600
2	0.6000	0.4000	0.0283	205197011.0980	0.0151	207097625.1347	1.4768	-103.7510	64382.7200
3	0.8000	0.2000	-0.0014	206577904.4296	0.0258	206259600.0238	3.9871	-464.5488	61568.6900

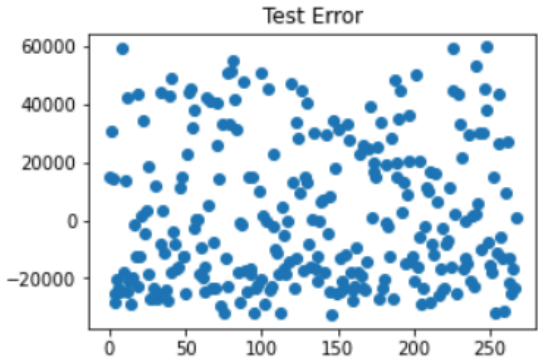
15. From step 14's results pick the best suited train test size and re-estimate the model that gives the following output:

Note you are picking what you think is best. Not what I think is best….
Give a comment on why you picked this combination of train test size.

Hint: Look at the notes there is code that does this. You need to modify it a little.

```
--------------------------------------
Model Info
--------------------------------------
salary = 69046.73411900894 + 498.72835797928394 age  + error

Test r_score is:          0.027376029370041466
Test rmse is:             620495376.5571196

Training-Test Split:      0.6 training 0.4 test

Training r_score is:      0.01444839164154843
Training rmse is:         570649786.7779077
Model Variance:           1.6229086977377503

Mean of Test Error:       1760.9724760128322
Mean of Train Error:      -1.0186340659856796e-11

Average Salary:  69046.73
```





16. Put the data file and your python script into a folder and zip the folder. Upload the zipped file to Canvas using the assignment link. You are done ☺