Predicting House Sale Prices in Kings County from the Areas of the Living Spaces

Jinanwa Chika

Introduction

This report aims to investigate how well the prices of houses in King County can be predicted from the area of the living spaces. The dataset used for this analysis contains sample house sale prices for King County, areas of living spaces and basements, number of bedrooms and bathrooms and fifteen other house related features. A simple linear regression model would be used to predict the prices of houses in King County from the areas of living spaces. The results of this analysis can provide real estate dealers with a reliable estimate for the value of properties in King County.

Before the simple linear regression model is used, we would determine, using a scatterplot, if there is a linear association (correlation) between the variables of interest. This linear association will be quantified using the Pearson's correlation coefficient *r*. Afterwards, a least square regression line would be used to produce a mathematical equation that describes the relationship between area of living spaces and the prices of houses which will be used to predict the prices of houses in King County. Again, the coefficient of determination R^2 (r^2 for simple linear regression) is computed to measure how well the model predicts the prices of houses. Furthermore, a confidence interval for the slope of the regression equation is computed and interpreted. Finally, a *p*-value for the regression slope is computed and interpreted to evaluate the statistical significance of this slope.

Dataset

This dataset contains house sale prices and eighteen other house-related features in King County from May 2014 to May 2015 retrieved from Kaggle (Kaggle, 2017). It is a multivariate dataset of 19 attributes which include our variables of interest-house sale prices and areas of living spaces. The research question is, "How well can the prices of houses in King County be predicted from the areas of the living spaces?"

The variables of interest are house sale prices and the areas of living spaces. House sale prices is the dependent variable or the response variable (y-variable) to be predicted. Its unit is USD ($) because houses sold are sold in USD in King County. It is a discrete quantitative variable. It is quantitative because it has a numerical value and discrete because it is countable, that is the number of possible digits that house prices can take on is finite. The areas of living spaces are the independent or predictor variable (x-variable). It is also a discrete quantitative variable because it has numerical values and its digits are countable. The scatter plot in *Fig.* 1 (Visualized in Appendix A) below shows a positive linear relationship between the two variables, that is, as the areas of living spaces increase, the prices of houses tends to increase. This is a reasonable observation because having larger living space typically makes a house more valuable, thus increasing its price. Identifying the types and classes of these variables is important because the type of regression model and methods used for this analysis are exclusively for discrete quantitative variables.

Methods

The data is downloaded into Python using a shareable Google Drive link and loaded into a Pandas package for analysis. The columns-House sale prices and areas of living spaces are extracted, and a scatterplot is produced as shown in *Fig 1.* below.

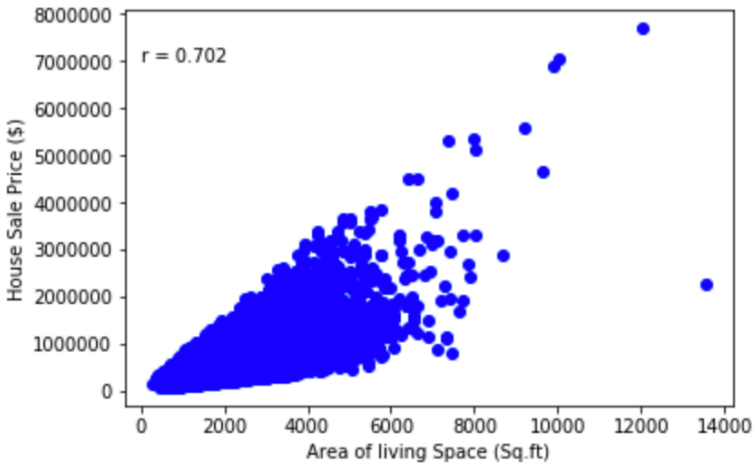


Figure 1. Scatterplot showing the relationship between the areas of living spaces of the house sale prices in King County between May 2014 and May 2015. There is a strong positive correlation (r = 0.702, calculated in Appendix B) between the two variables.

Calculating the Correlation Co-efficient *r*

There appears to be a linear relationship between the areas of living spaces and the prices of houses. A Pearson correlation coefficient r = 0.702 (calculated with the formula-derived in Appendix B, $\frac{cov(x,y)}{s_x s_y}$, where *cov(x,y)* is the co-variance of *x and* y and s_x *and* s_y are the standard deviations of *x and* y respectively) suggests a strong positive correlation between the two variables. This means that as the areas of living spaces increases, the prices of the houses tend to increase. While we cannot infer causation from this correlation, it is possible that the size of the house and the amount of materials spent to build this house are possible extraneous variables that produce the observed correlation. Bigger houses are more likely to have larger areas of living spaces and this means that more building materials will be needed for these houses. Hence, more expenses are incurred in building these houses as more materials are purchased. This increases the value and prices of the houses.

Fitting the least squared regression line

The least squared line of regression (derived in Appendix C) is fitted to the scatterplot as shown in *Fig 4.* Before the line is fitted, the following assumptions were analyzed and verified.

1. Linearity: The data in the scatterplot in *Fig 1.* showed a linear trend (positive) hence this assumption is satisfied.
2. Nearly normal residuals: The histogram of the residuals plotted in *Fig 2.* below shows a nearly normal distribution with mean of 0. It is not strongly skewed; hence this condition is satisfied (Kindly refer to Appendix D for more rigorous treatment of this distribution and the least square line).

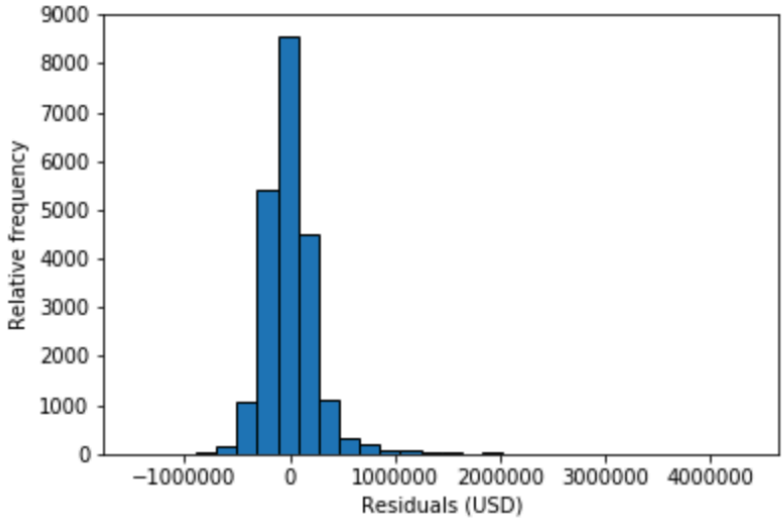


Figure 2. Histogram of the frequency distribution of the residuals. The histogram resembles a normal distribution with mean of 0. The significance of a mean of 0 is analyzed rigorously in Appendix D.

3. Constant Variability: The variability of the points around the least squares line is constant, that is the scatterplot is homoscedastic. The variability of all points in *Fig 3.* (plotted in Appendix D) are not constant, a case of heteroscedasticity. This implies that for smaller values of *x,* the regression line is suitable but is less accurate for predictions for larger *x,* (typically $x > 4000$). This is a limitation of this model.

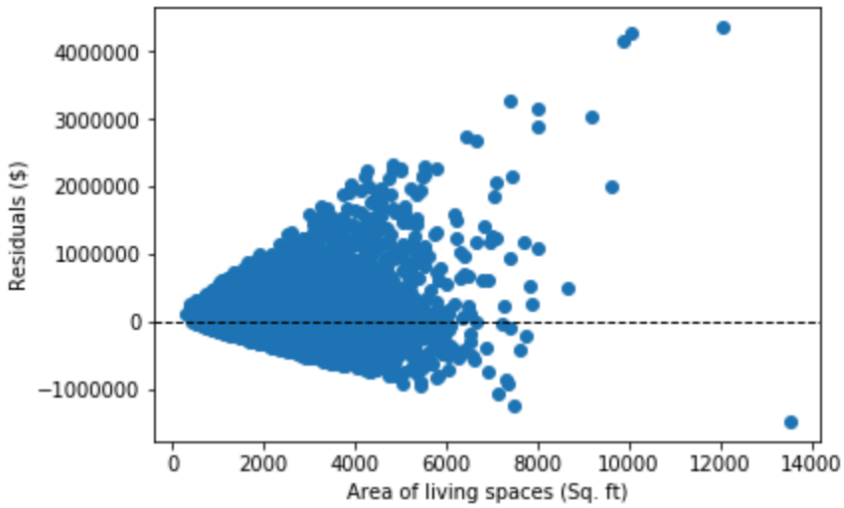


Figure 3. Scatterplot of area of living spaces vs residuals showing the variability of errors at each value of *x* around the least square regression line. The scatterplot is heteroscedastic suggesting that prediction is less accurate for larger *x*.

4. Independent Observations: I am assuming that there is no underlying structure in data collection (consecutive/sequential observations).

With the above assumptions, the least squared line is fitted with the data points as shown in *Fig 4.* below.

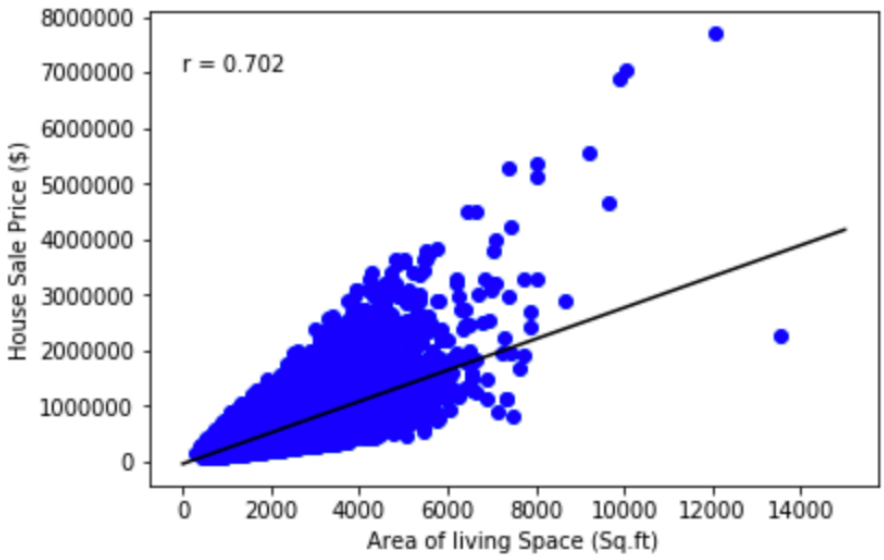


Figure 4. Scatterplot of the areas of living spaces vs house sale prices in King County fitted with a least squared regression line which presents a mathematical relationship between the two variables. The equation of the least squared regression line (calculated in Appendix C) is given as $y = 280.81x - 43867.6$

Interpreting the regression line parameters

The equation of regression line is:

House sale price = 280.81×(area of living space) – 43867.6. The slope b_1 = 280.81$/sq.ft means that if the area of living space increases by 1 sq.ft, the price of the house increases (because the slope is positive) by $280.81. An intercept b_0 = -$43867.6 means that when the

area of the living space is 0, the average cost of the house is $-43867.6. This does not have a practical value because prices of houses cannot be negative.

Computing and interpreting R^2

The co-efficient of determination (computed in Appendix E by squaring the correlation coefficient *r*) r^2 ($r^2 = R^2$ for simple linear regression) is 0.493. This means that 49.3% of the variation in the housing sale prices is reduced by taking into account the predictor variable- areas of living spaces. This medium R^2 value suggests a moderate line fit with the data points. This R^2 value is not very high because of the heteroscedasticity described above which makes less accurate prediction for large *x*. Hence, this model is an average model for predicting house sale prices.

Inferences from the Regression Line

- **Confidence Intervals for the Slope of the Regression Line**

  The confidence interval for the slope of the regression line is a range of plausible values of the slope of the true population. It is calculated in Appendix F using the formula $b_1 \pm t_{df=n-2} \times SE(b_1)$ where $SE(b_1) = \sqrt{\frac{1-R^2}{n-2}} \frac{s_y}{s_x}$ is the standard error of the slope and the parameters have their usual meanings (*n* is the size of the sample population). The standard error of the slope is 1.94 and the t-score (used because we are dealing with samples) for a 95% confidence interval with df = 21611 is 1.65. Hence, the confidence interval is [277.62, 284]. This means that we are 95% confident that the slope of the true population's regression line is between 277.62 and 284. In terms of a frequentist probability theory, this means that if we repeat this process many times (constructing confidence intervals for the slopes for the slope of the true population), then about 95% of the intervals will capture the true slope of the

population's regression line. This inference is an example of a type of induction called statistical generalization. It is statistical generalization because we made use of confidence intervals, t scores etc. (statistical tools) on a sample of houses in King County to make a generalization of the population of houses in the county. The induction is a strong induction because we have used a 95% confidence interval which means a 95% confidence of capturing the slope of the true population. While we are not 100% of this generalization/induction, this argument can be made stronger by constructing a 99% confidence interval which provides a wider range for capturing the parameter.

- **Assessing the statistical significance of the slope of the regression equation**

The null hypothesis H_0 is that the slope of the true population is 0. This implies that there is no linear relationship between the area of living spaces and the house sale prices. The alternative hypothesis is that the slope of the true population is not equal to 0. This implies that there is some linear relationship between our variables of interest. We are trying to determine if the estimated slope b_0 is sufficiently far from 0 so we can conclude that the true slope is non-zero. A type 1 error is rejecting the null hypothesis when it is true. This means saying that there is some linear relationship between our variables of interest when there is no linear relationship between the variables in real life. A type 2 error is failing to reject the null hypothesis when the alternative is true. This means saying that there is no linear relationship between our variables of interest when indeed there is a linear relationship between them in real life. We are interested in identifying if there is a linear relationship, so a Type 2 error is costlier, hence the choice of $\alpha = 0.05$. A t-score is computed as 144.92 in Appendix G using the formula $\frac{b1 - 0}{SE(b1)}$. The *p*-value for a two-tailed test (also computed in Appendix G) is 0.00 which is less than α . This means that the probability of getting this result or a more extreme result given that the null hypothesis is true is 0. Hence, we reject the null hypothesis because

the *p*-value provides evidence for a statistically significant linear relationship. A common misinterpretation of *p*-value is that it is the probability that the null hypothesis is true given the result. This is an example of the confusion of the inverse fallacy. This fallacy is analyzed in detail in Appendix H and corrected.

Conclusion

The areas of living spaces and the prices of houses show a strong positive correlation of 0.702. This means that generally as the areas of living spaces increases, the prices of houses tend to increase as well. The simple linear regression model makes use of the areas of living spaces as the predictor variable to predict the house sale prices (the response variable) using the equation: house sale price = 280.81×(area of living space) – 43867.6. The R^2 value of 0.493 suggests that this model performs moderately in predicting house sale prices. The heteroscedasticity in the residual plot of *Fig.3* shows that the predictions of this model for large *x* are less accurate. Thus, this model only performs well for smaller values of *x.* This model can be improved by taking other relevant variables such as number of bedrooms to build a multiple linear regression model with higher R^2.

References

Diez, D., Barr, C., & Rundel, M. (2015). Introduction to Linear Regression. In *Open Intro Statisitics*. Retrieved from https://drive.google.com/file/d/0B-DHaDEbiOGkc1RycUtIcUtIelE/view

Kaggle. (2017, January 16). *House Sales in King County, USA*. Retrieved January 30, 2020, from Kaggle website: https://www.kaggle.com/harlfoxem/housesalesprediction

Appendix

Appendix A- Import, Extract data and Scatterplot

In [109]:
```
import pandas as pd
from scipy import stats
import numpy as np
import matplotlib.pyplot as plt
url_data = 'https://docs.google.com/spreadsheets/d/1SlFAdx0R5fhQVV1rAoU6xcRIwbuV4-Qe87Mauluqgp0/export?format=csv'
#download csv file to python
housing_data = pd.read_csv(url_data) #feed into a Pandas module for analysis
```

In [110]:
```
housing_data
```

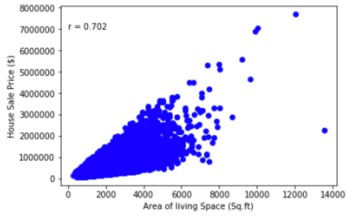
Out[110]:

sqft_living	sqft_lot	floors	waterfront	view	...	grade	sqft_above	sqft_basement	yr_built	yr_renovated	zipcode	lat	long	sqft_living15	sqft_lot15
1180	5650	1.0	0	0	...	7	1180	0	1955	0	98178	47.5112	-122.257	1340	5650
2570	7242	2.0	0	0	...	7	2170	400	1951	1991	98125	47.7210	-122.319	1690	7639
770	10000	1.0	0	0	...	6	770	0	1933	0	98028	47.7379	-122.233	2720	8062
1960	5000	1.0	0	0	...	7	1050	910	1965	0	98136	47.5208	-122.393	1360	5000
1680	8080	1.0	0	0	...	8	1680	0	1987	0	98074	47.6168	-122.045	1800	7503
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1530	1131	3.0	0	0	...	8	1530	0	2009	0	98103	47.6993	-122.346	1530	1509
2310	5813	2.0	0	0	...	8	2310	0	2014	0	98146	47.5107	-122.362	1830	7200
1020	1350	2.0	0	0	...	7	1020	0	2009	0	98144	47.5944	-122.299	1020	2007
1600	2388	2.0	0	0	...	8	1600	0	2004	0	98027	47.5345	-122.069	1410	1287
1020	1076	2.0	0	0	...	7	1020	0	2008	0	98144	47.5941	-122.299	1020	1357

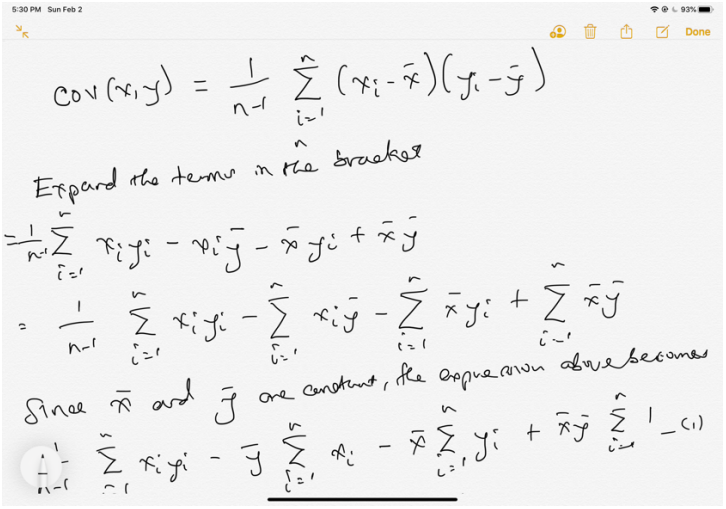
```
price = list(housing_data['price'].values) #extract all the prices in the price column
print(price)
```

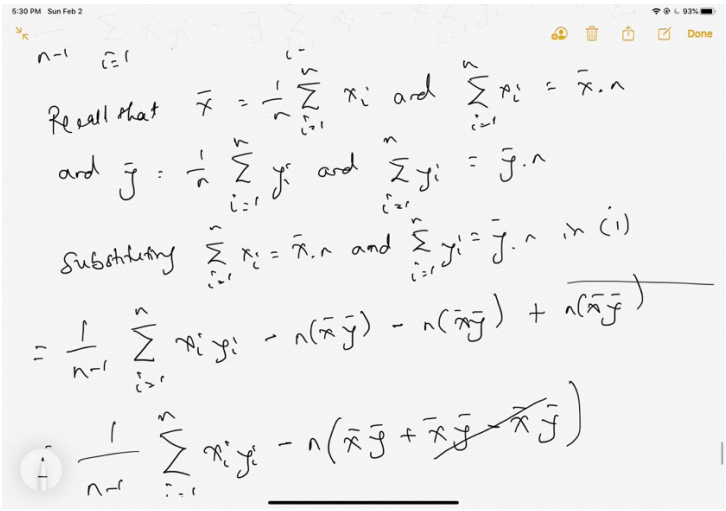
```
living_space = list (housing_data['sqft_living'].values) #extract areas of living spaces from sqft_living
print(living_space)
```

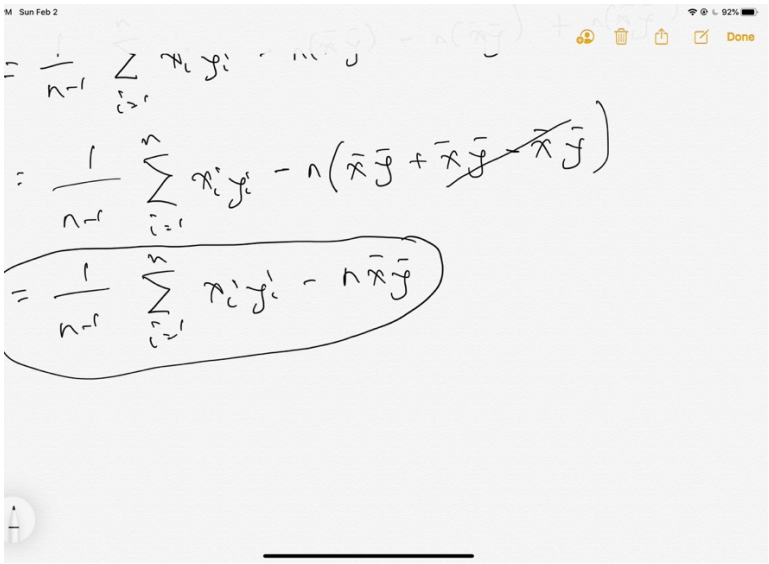
```
In [118]: plt.scatter(living_space,price, c='blue') #scatterplot of the variables of interest
          plt.xlabel('Area of living Space (Sq.ft)')
          plt.ylabel('House Sale Price ($)')
          plt.text(0,7000000,'r = 0.702')
          plt.show()
```

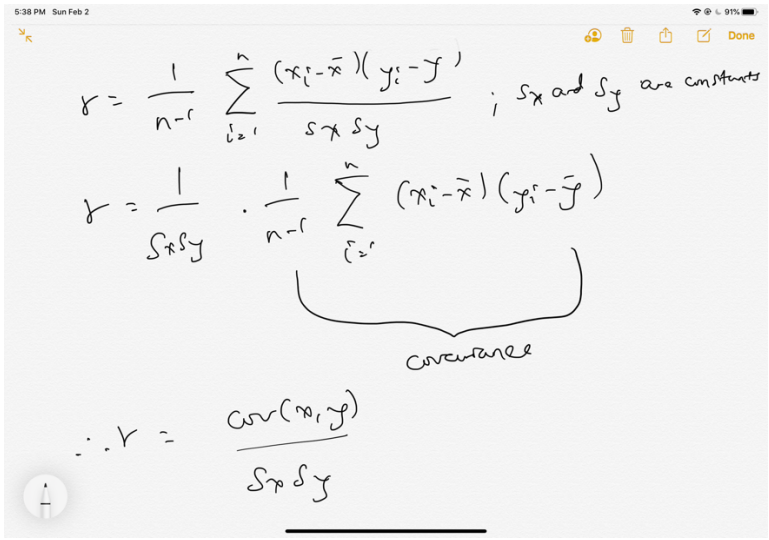


Appendix B- Formula Derivation and Code Implementation of r


5:30 PM Sun Feb 2
Done
$\text{Cov}(x,y) = \frac{1}{n-1}\sum_{i=1}^{n}(x_i-\bar{x})(y_i-\bar{y})$
Expand the terms in the brackets
$= \frac{1}{n-1}\sum_{i=1}^{n} x_i y_i - x_i\bar{y} - \bar{x}y_i + \bar{x}\bar{y}$
$= \frac{1}{n-1}\sum_{i=1}^{n} x_i y_i - \sum_{i=1}^{n} x_i\bar{y} - \sum_{i=1}^{n}\bar{x}y_i + \sum_{i=1}^{n}\bar{x}\bar{y}$
Since $\bar{x}$ and $\bar{y}$ are constant, the expression above becomes
$= \frac{1}{n-1}\sum_{i=1}^{n} x_i y_i - \bar{y}\sum_{i=1}^{n} x_i - \bar{x}\sum_{i=1}^{n} y_i + \bar{x}\bar{y}\sum_{i=1}^{n} 1$ ...(1)


5:30 PM Sun Feb 2
Done
Recall that $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$ and $\sum_{i=1}^{n} x_i = \bar{x}.n$
and $\bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i$ and $\sum_{i=1}^{n} y_i = \bar{y}.n$
Substituting $\sum_{i=1}^{n} x_i = \bar{x}.n$ and $\sum_{i=1}^{n} y_i = \bar{y}.n$ in (1)
$= \frac{1}{n-1}\sum_{i=1}^{n} x_i y_i - n(\bar{x}\bar{y}) - n(\bar{x}\bar{y}) + n(\bar{x}\bar{y})$
$\frac{1}{n-1}\sum_{i=1}^{n} x_i y_i - n(\bar{x}\bar{y} + \bar{x}\bar{y} - \bar{x}\bar{y})$



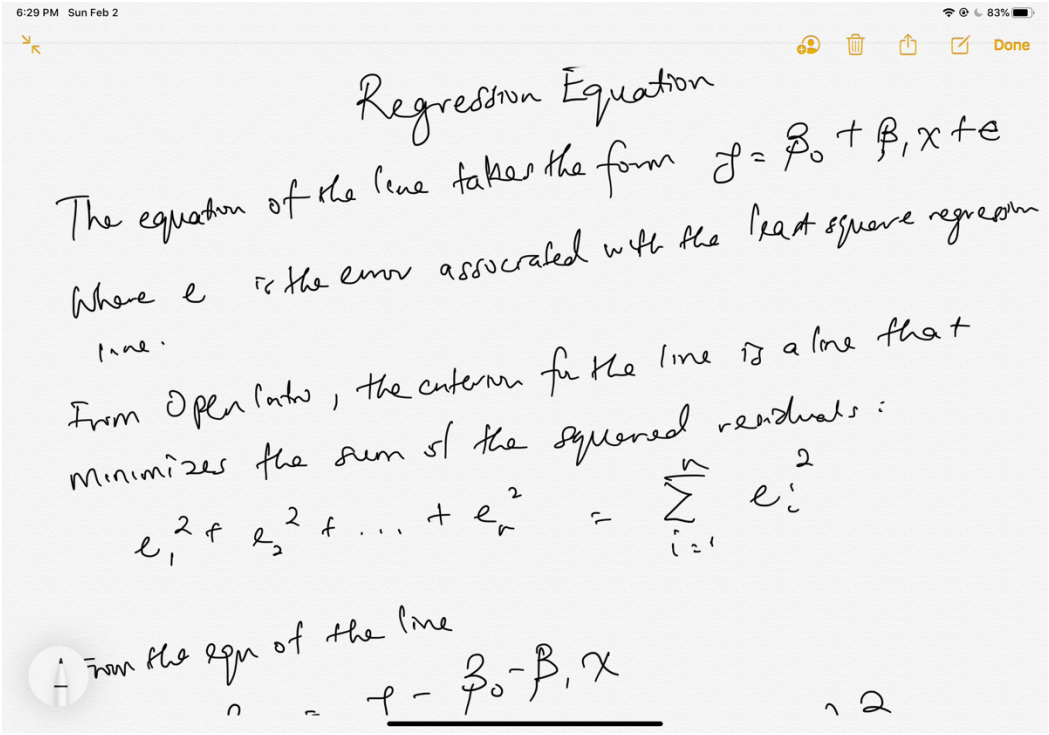


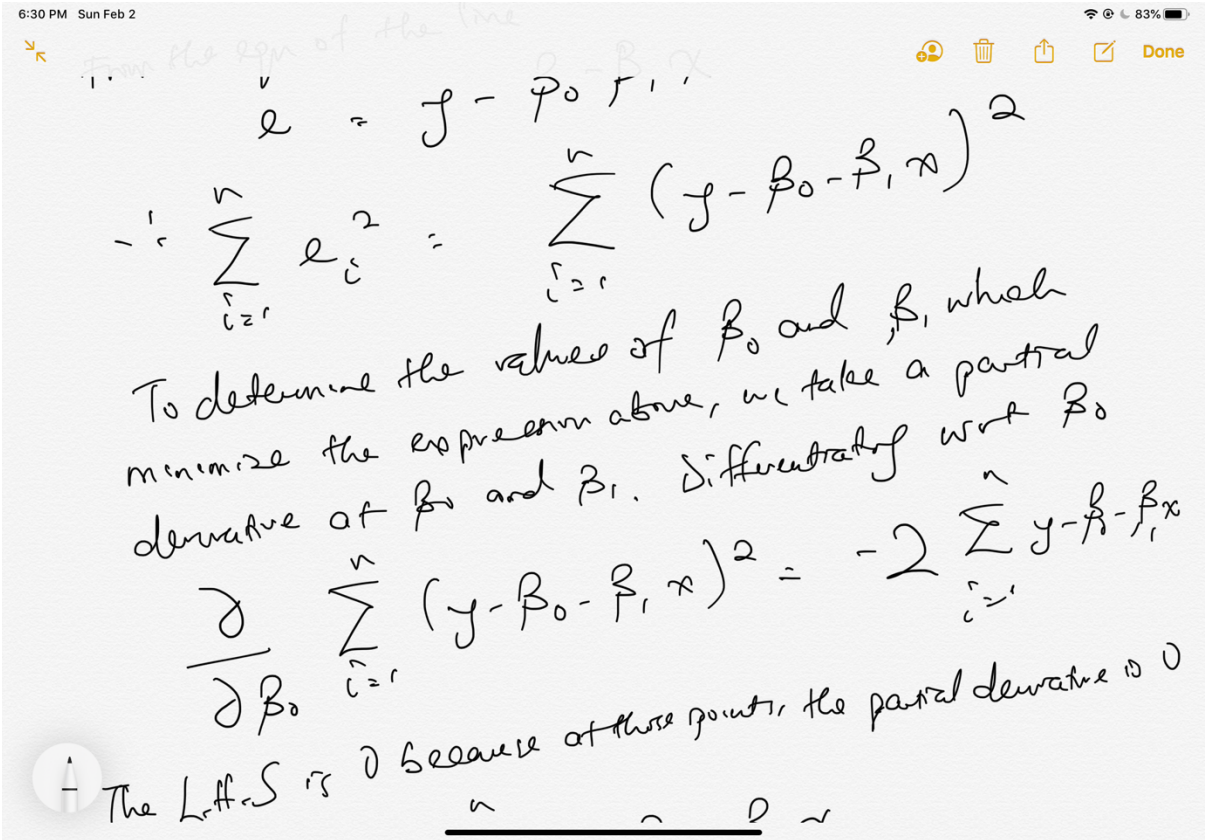

```
def mean(array):
    """Function that returns the mean of list/array object"""
    length = len(array) #total number of elements in list
    total=0 #initializes the sum of elements in list
    for elem in array: #iterates through each element
        total += elem #add element to total
    mean=total/length #compute the average
    if mean == np.mean(array):  #compare with mean generated by numpy module
        return mean
def standard_deviation(array):
    """Function that returns the standard deviation of a list/array object"""
    total = 0 #initializes sum of elements in the list
    total_square_deviation = 0 #initializes the sum of the squares of the deviations
    deviation_list = [] #initializes list which will contain squares of the deviations
    n = len(array) #number of elements in list
    for elem in array: #loops through each elem in array
        total += elem #adds the elem to total
    average = total/len(array) #computes mean
    for num in array: #loops through each element in array
        deviation_square = (num-average)**2 #square difference between number and mean
        deviation_list += [deviation_square] #add the square of deviations of each element to the list
    for num in deviation_list: #loops through each element in list
        total_square_deviation += num #sums the square of the deviations
    variance = total_square_deviation/(n-1) #computes variance with Bessel's correction for sample data
    std_dev = variance **0.5 #square root of variance

    return std_dev
```

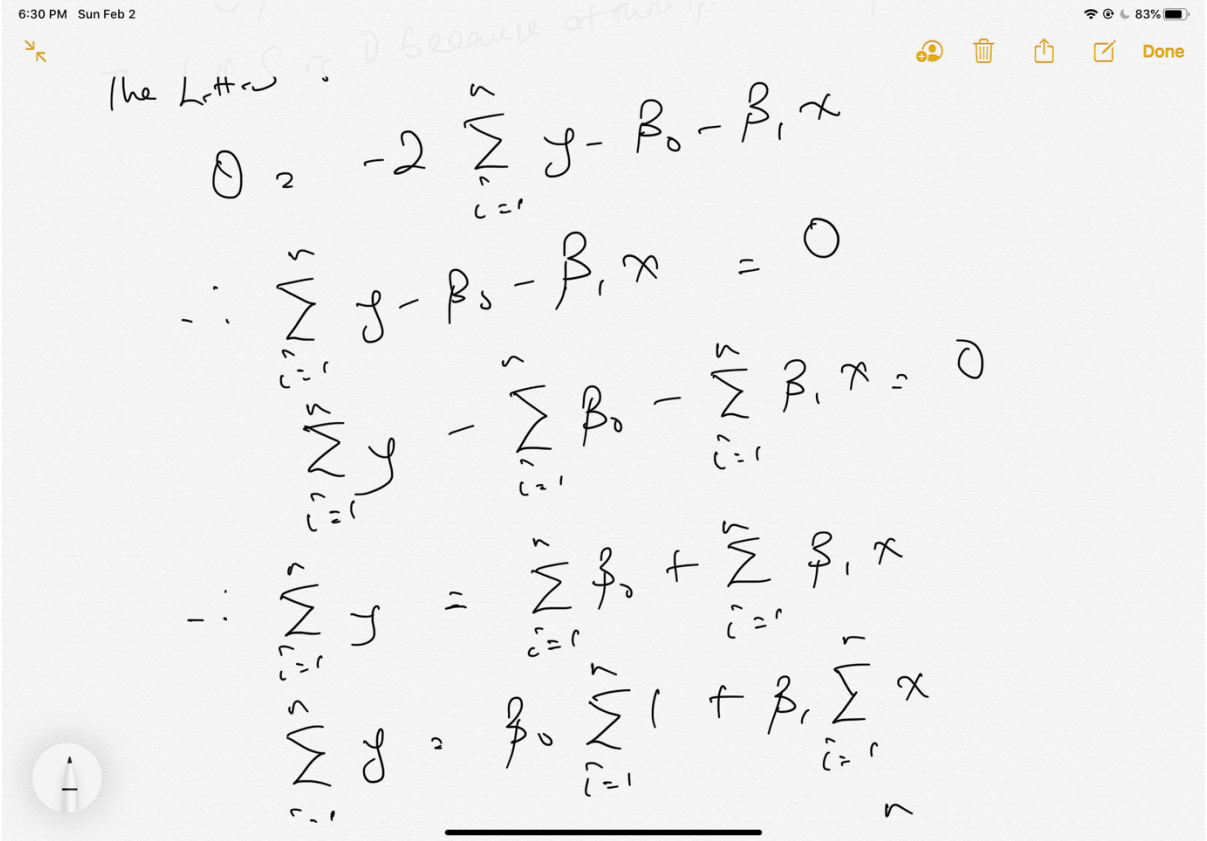
```
def covariance(x,y):
    """Function that returns the cov(x,y). The formula of cov this function uses is derived in the hand
    calculation attached in the appendix B
    """
    summation = 0 #initialize sum
    n = len(x) #get sample size n
    b = mean(x)*mean(y)
    for i, j in zip (x,y): #iterate through paired elements x and y
        inner_terms = (i*j) - b
        #evaluates the inner terms of the formula
        summation+= inner_terms #increments the sum
    covariance = (1/(n-1))*summation
    return covariance

def correlation_r(x,y):
    """Function returns the correlation between two variables. The formula this function uses is also
    derived in the hand calculation attached in appendix B"""
    r=covariance(x,y)/(standard_deviation(x)*standard_deviation(y))
    return r


print( 'The correlation coefficient r is', correlation_r(living_space,price))
```

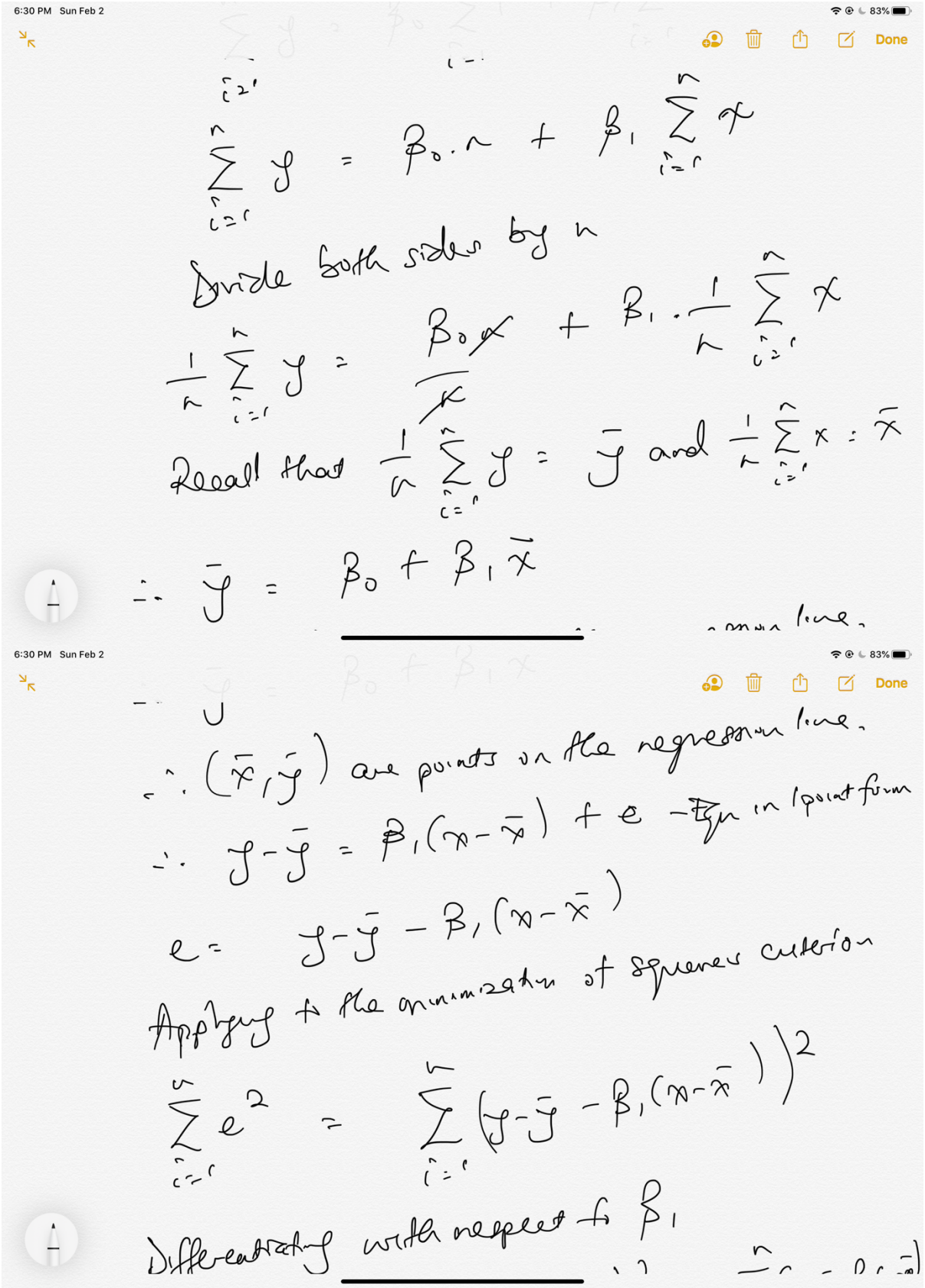
```
The correlation coefficient r is 0.7020437212326098
```

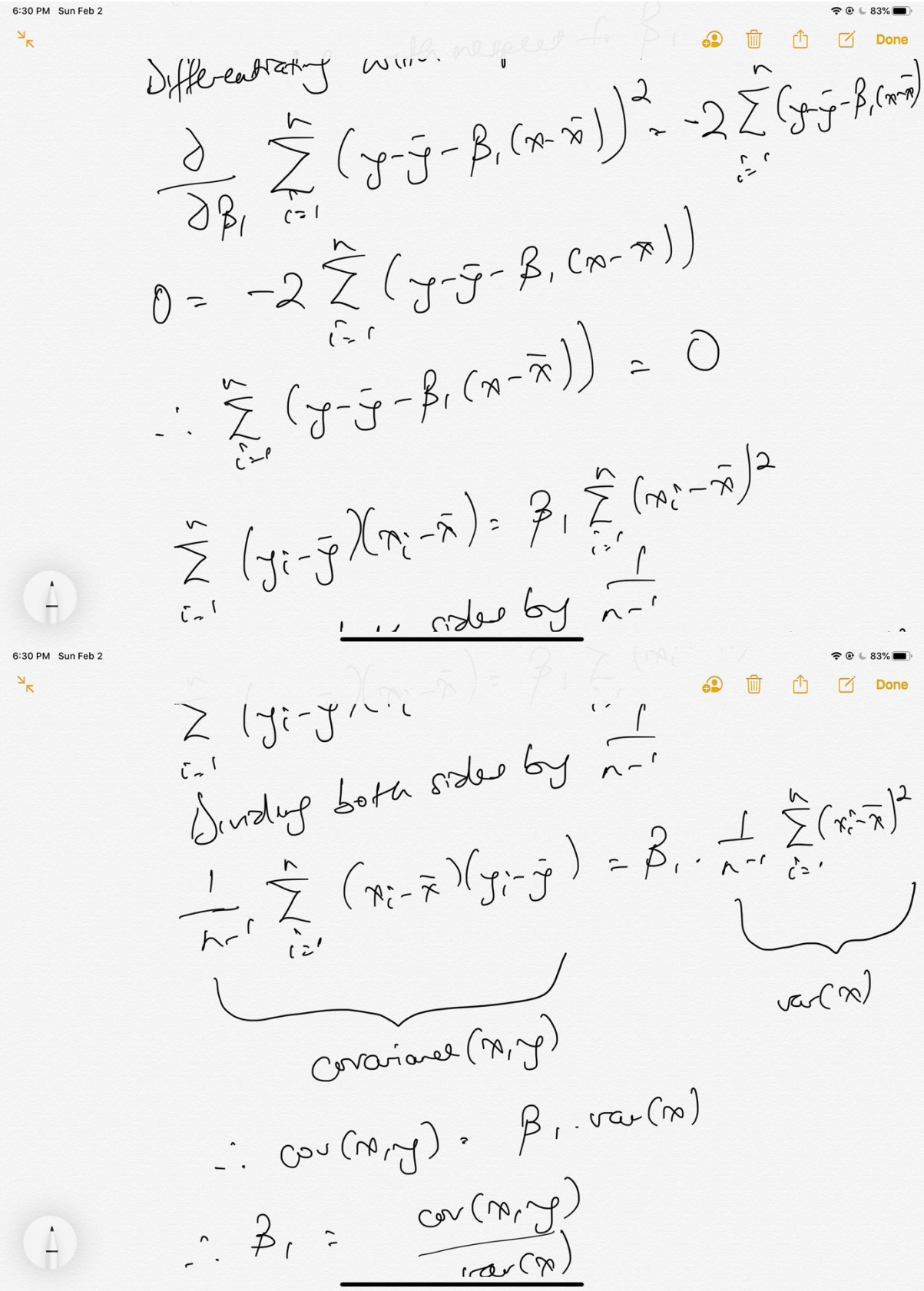
Appendix C- Calculating the Regression Equation and Code Implementation

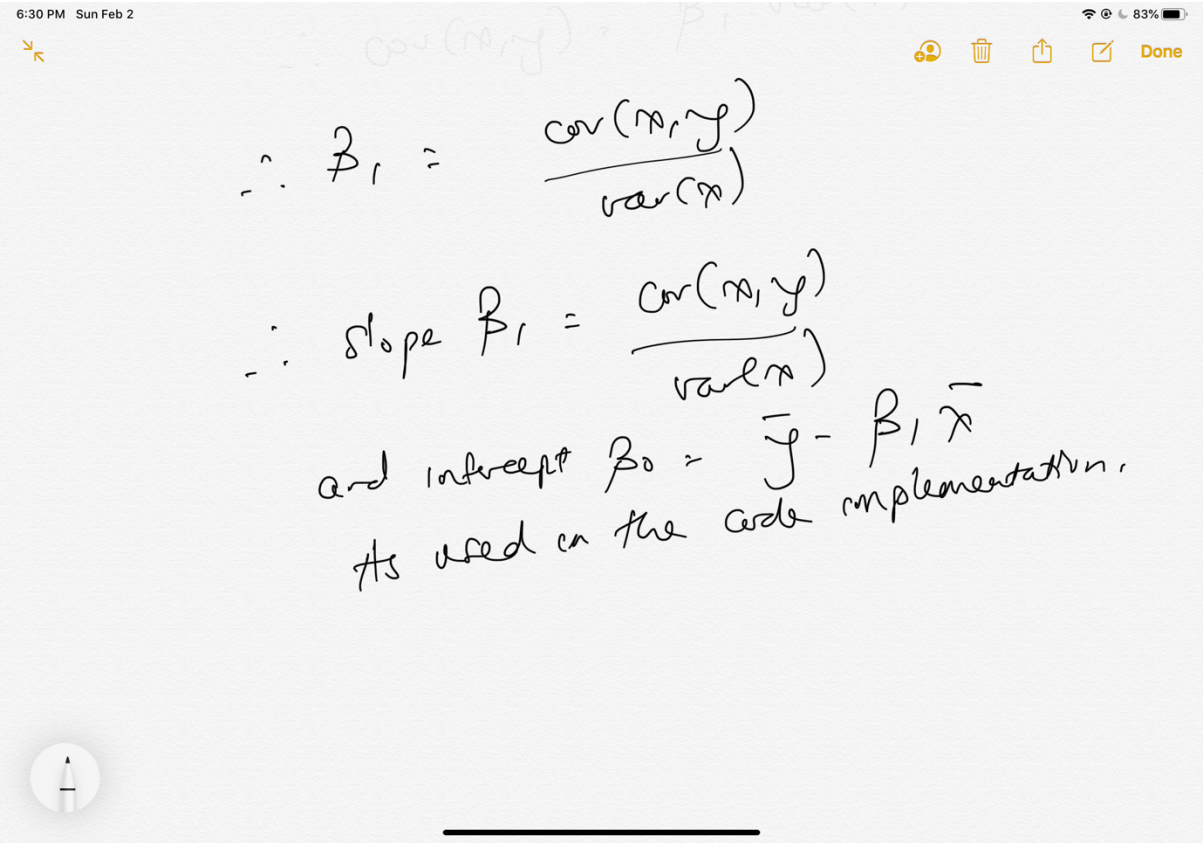



6:30 PM Sun Feb 2
83%
Done
$e = y - \beta_0 - \beta_1 x$
$\therefore \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y - \beta_0 - \beta_1 x)^2$
To determine the values of β_0 and β_1 which minimize the expression above, we take a partial derivative of β_0 and β_1. Differentiating wrt β_0
$\frac{\partial}{\partial \beta_0} \sum_{i=1}^{n} (y - \beta_0 - \beta_1 x)^2 = -2 \sum_{i=1}^{n} y - \beta_0 - \beta_1 x$
The L.H.S is 0 because at those points, the partial derivative is 0


6:30 PM Sun Feb 2
83%
Done
The L.H.S:
$0 = -2 \sum_{i=1}^{n} y - \beta_0 - \beta_1 x$
$\therefore \sum_{i=1}^{n} y - \beta_0 - \beta_1 x = 0$
$\sum_{i=1}^{n} y - \sum_{i=1}^{n} \beta_0 - \sum_{i=1}^{n} \beta_1 x = 0$
$\therefore \sum_{i=1}^{n} y = \sum_{i=1}^{n} \beta_0 + \sum_{i=1}^{n} \beta_1 x$
$\sum_{i=1}^{n} y = \beta_0 \sum_{i=1}^{n} 1 + \beta_1 \sum_{i=1}^{n} x$


6:30 PM Sun Feb 2
83%
Done
$\sum_{i=1}^{n} y = \beta_0 \cdot n + \beta_1 \sum_{i=1}^{n} x$
Divide both sides by n
$\frac{1}{n}\sum_{i=1}^{n} y = \beta_0 + \beta_1 \cdot \frac{1}{n}\sum_{i=1}^{n} x$
Recall that $\frac{1}{n}\sum_{i=1}^{n} y = \bar{y}$ and $\frac{1}{n}\sum_{i=1}^{n} x = \bar{x}$
$\therefore \bar{y} = \beta_0 + \beta_1 \bar{x}$
6:30 PM Sun Feb 2
83%
Done
$\therefore (\bar{x}, \bar{y})$ are points on the regression line.
$\therefore y - \bar{y} = \beta_1 (x - \bar{x}) + e$ – Eqn in point form
$e = y - \bar{y} - \beta_1 (x - \bar{x})$
Applying the minimization of squares criterion
$\sum_{i=1}^{n} e^2 = \sum_{i=1}^{n} (y - \bar{y} - \beta_1 (x - \bar{x}))^2$
Differentiating with respect to β_1

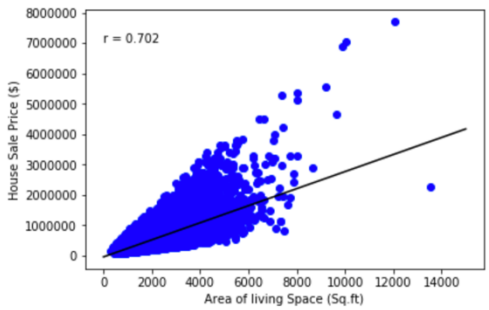

6:30 PM Sun Feb 2
83%
Done
Differentiating with respect to β_1
$\frac{\partial}{\partial \beta_1} \sum_{i=1}^{n} (y-\bar{y}-\beta_1(x-\bar{x}))^2 = -2 \sum_{i=1}^{n} (y-\bar{y}-\beta_1(x-\bar{x}))$
$0 = -2 \sum_{i=1}^{n} (y-\bar{y}-\beta_1(x-\bar{x}))$
$\therefore \sum_{i=1}^{n} (y-\bar{y}-\beta_1(x-\bar{x})) = 0$
$\sum_{i=1}^{n} (y_i-\bar{y})(x_i-\bar{x}) = \beta_1 \sum_{i=1}^{n} (x_i-\bar{x})^2$
dividing both sides by $\frac{1}{n-1}$
6:30 PM Sun Feb 2
83%
Done
$\sum_{i=1}^{n} (y_i-\bar{y})(x_i-\bar{x})$ dividing both sides by $\frac{1}{n-1}$
$\frac{1}{n-1} \sum_{i=1}^{n} (x_i-\bar{x})(y_i-\bar{y}) = \beta_1 \cdot \frac{1}{n-1} \sum_{i=1}^{n} (x_i-\bar{x})^2$
covariance(x,y)
var(x)
$\therefore cov(x,y) = \beta_1 \cdot var(x)$
$\therefore \beta_1 = \frac{cov(x,y)}{var(x)}$




```
[115]: def regression_eqn(x,y):
           """Function returns the eqn of the least square regression line. The formula this function uses is
           derived in the hand calculation attached in Appendix C"""
           variance_x = (standard_deviation(x))**2 #compute variance
           global slope
           slope = covariance(x,y)/variance_x #compute slope
           global intercept
           intercept = mean(y) - slope*mean(x) #compute intercept
           a= 'Regression line equation is', 'y'+'='+ str(slope)+'x' + str(intercept)
           return a
       regression_eqn(living_space,price)
```

```
[115]: ('Regression line equation is', 'y=280.8066899295349x-43867.60153392691')
```

```
[131]: plt.scatter(living_space,price, c='blue')#scatterplot of the variables of interest
       plt.plot(list_x,list_y, c='black') #plot regression line
       plt.xlabel('Area of living Space (Sq.ft)')
       plt.ylabel('House Sale Price ($)')
       plt.text(0,7000000,'r = 0.702')
       plt.show()
```



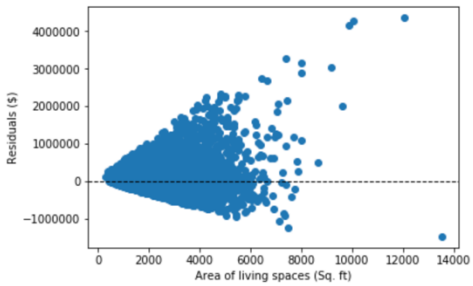
Appendix D- Analysis of the Histogram of Residuals using the least square regression line criterion, Scatterplot of Residuals, and Histogram Plotting,

In deriving the equation of the least squared line in Appendix C, we noted that the sum of errors is 0 (sigma e) after taking partial derivatives at the slope and intercepts. Where errors represent the residuals $y_1 - \hat{y}$. If the sum of residuals is 0, it means that the mean of the residuals is also 0 (0/n = 0 where n is the sample size). This is evident in the histogram in *Fig 2.* where the observed mean (midpoint of the histogram is 0). Therefore, this is more evidence that the regression line satisfies the least square criterion of sum of errors = 0.

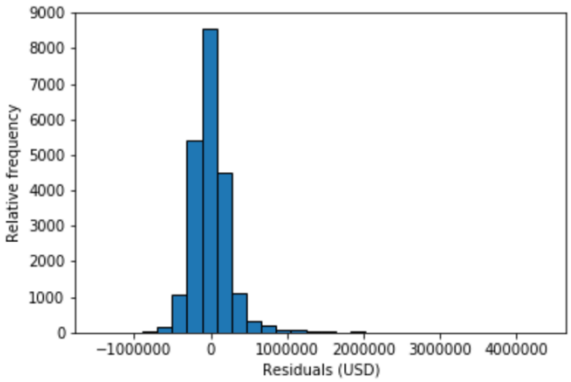
```
residual_list = [] #list of residuals used for heteroscedastic plot below
for r,s in zip(living_space, price): #iterate through the paired values
    residual = s-predict_value(r)  #computes the residuals-y-y^
    residual_list += [residual] #append to residual list
print(residual_list)
```

```
636.710906505934, -287795.7783251429, -253584.56306941528, 105768.83074737567, 39526.823768515256, -280078.193678636,
233606.1157917083, 79072.0279558315, -41836.18669977551, 196485.11230227805, -151916.855692729, -371502.39437652205,
32749.425862173375, 289468.38747844077, -182450.94352860574, -184503.77134628245, -126278.36297164985, -217745.778325
14292, -422430.08779548726, 51529.57770803606, -68716.56018010568, 253628.34849458735, -378966.9295708848, -221765.25
708850112, -416400.86965044995, -267260.3873976616, 63202.77082694083, -357471.79926172434, 37302.91858325247, -14381
5.18321034522, 52847.01748754084, -270210.31351950567, 66981.6195811989, 60880.09485512687, 68083.144307271, -104918.
2326624894, 130536.5631501943, -93695.70444698707, -58514.88769772195, -121169.97902302898, 105618.60911290825, 2800
1.098344557104, 32386.341515726876, 11981.619581198902, -61562.207636356936, 218377.97910380817, 264749.3519840175, -
20795.704446987074, 479031.69345935475, 246460.76384808036, -238328.56306941528, 139863.6655439129, 16465.6335389199
1, -128634.51421739173, -499066.7044736105, 34849.49974032916, -120008.641037122, 73475.37292059895, 291709.017487540
84, -84523.25010964065, -191927.9720441685, -149937.7114258476, 114425.29904244316, -115019.7573885615, 33344.6300494
8967, -328421.72538356856, -118302.09886389872, -21584.292582924303, -168200.57413782662, 338663.8133002245, 72364.10
881284781, -110840.21009008755, -200828.56306941528, -42532.98949131963, 317384.9645459664, 55749.35198401753, 15036.
563150194299, -221325.070348336, 38719.01748754084, 148415.55966076406, -172233.28500394302, -65029.7573885615, -9176
6.63405826152, -11860.53515397315, -141955.8132194453, -286683.1372476313, -232795.77832514292, 177320.28159529192, -
105068.45429695689, -172624.77483571268, -267421.01740676153, -113787.48979138013, -18787.489791380125, 31768.8307473
75672, -5411.986001889454, 84091.50671918964, -33766.63405826152, -173563.58460611734, 496578.27461643144, 904375.225
1642873, -77342.43336037546, -137156.10873206845, 17188.16175442218, -51781.243130780174, 52939.908114961756, 47396.0
8089740586, 120122.10183398734, -18882.76785685221, 273912.0669396849, 88960.76384808036, 298012.2146959966, 70904.44
330932456, -68776.37343994057, 70950.49974032916, 122586.63702835015, -246834.809730015, 236850.87671008962, 183971.8
```

```
In [150]: plt.scatter(living_space,residual_list) #plots scatterplot of sg.ft vs residuals to test for homoscedasticity
          plt.axhline(linewidth=1,color='k',linestyle='dashed') #plot a horizontal line (regression line) at 0
          plt.xlabel('Area of living spaces (Sq. ft)')
          plt.ylabel('Residuals ($)')
          plt.show()
```




```
:  plt.hist(residual_list, bins= 30, edgecolor='black') #plot histogram of residuals
   plt.xlabel('Residuals (USD)')
   plt.ylabel('Relative frequency')
   plt.show()
```



Appendix E- Computing R^2

```
In [143]: Rsquared = (correlation_r(living_space,price))**2 #compute R squared
          print(Rsquared)
```

```
0.49286538652213036
```

Appendix F – Computing Confidence Intervals

```
In [144]: def standard_error(x,y):
              """Function returns the standard error using the formula stated in the main paper"""
              global std_err
              std_err = ((((1-Rsquared)/(len(x)-2))**0.5)* (standard_deviation(y)/standard_deviation(x))
              return 'The standard error is', std_err
          standard_error(living_space,price)
```

```
Out[144]: ('The standard error is', 1.937614990233579)
```

```
In [145]: stats.t.ppf(0.95,21611)
```

```
Out[145]: 1.6449241389530618
```

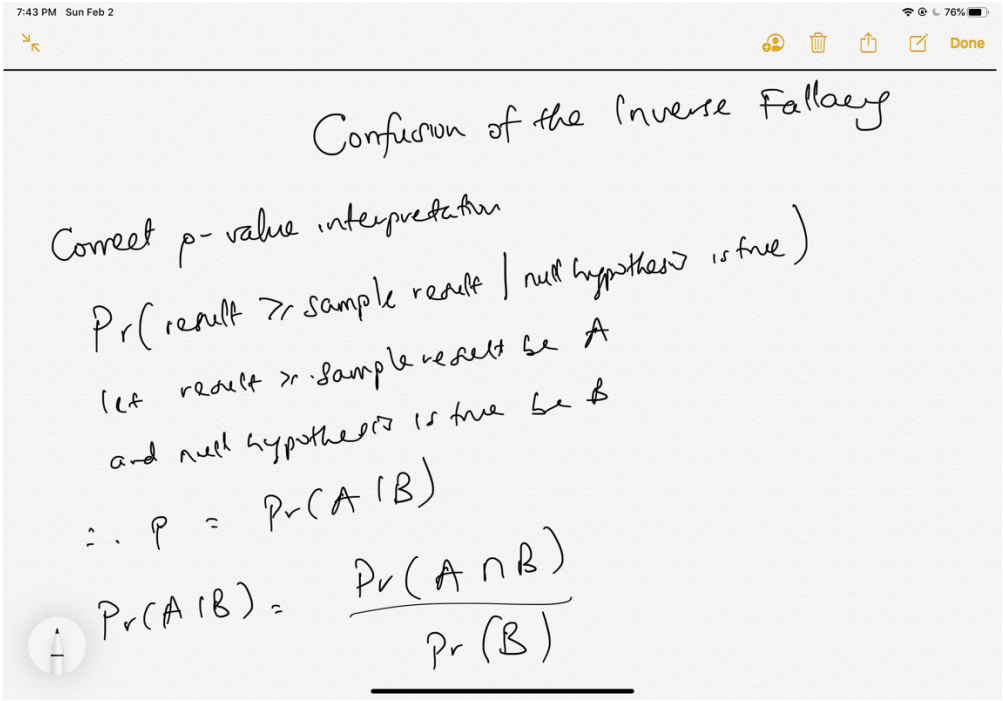
```
In [146]: def confidence_interval_95(x,y):
              """Function returns the confidence interval for population parameter"""
              global df
              df= len(x)-2 #degress of freedom
              t_score = stats.t.ppf(0.95,df)
              lower_bound = slope - (t_score*1.94) #compute lowerbound
              upper_bound = slope + (t_score*1.94) #compute upperbound
              return [lower_bound,upper_bound]
          confidence_interval_95(living_space,price)
```

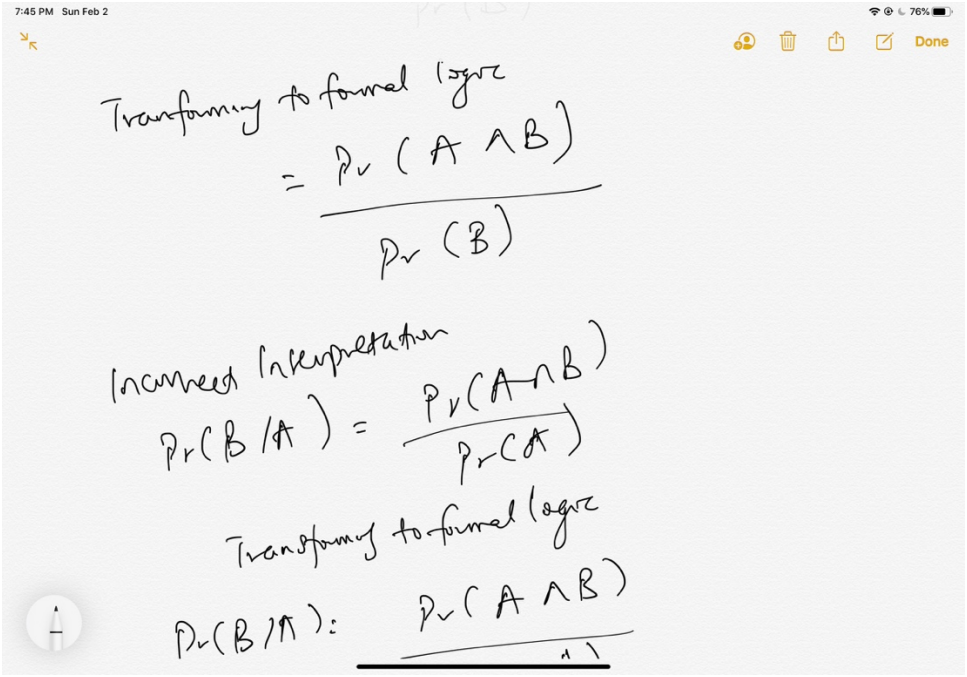
```
Out[146]: [277.615537099966, 283.99784275910383]
```

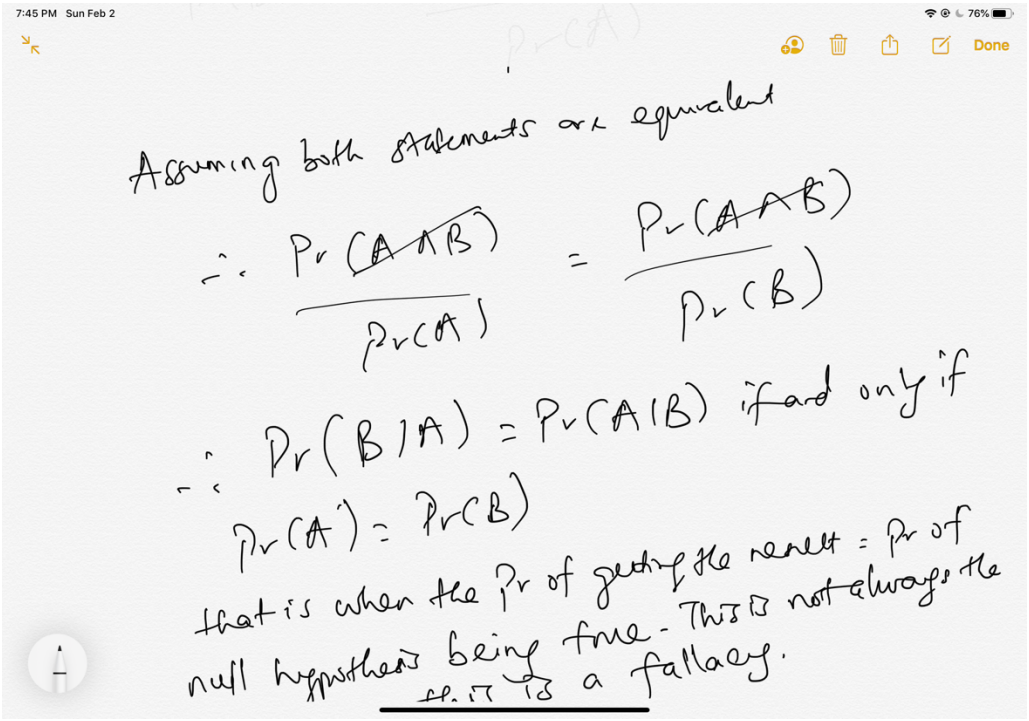
Appendix G – Significance Testing for Regression Slope

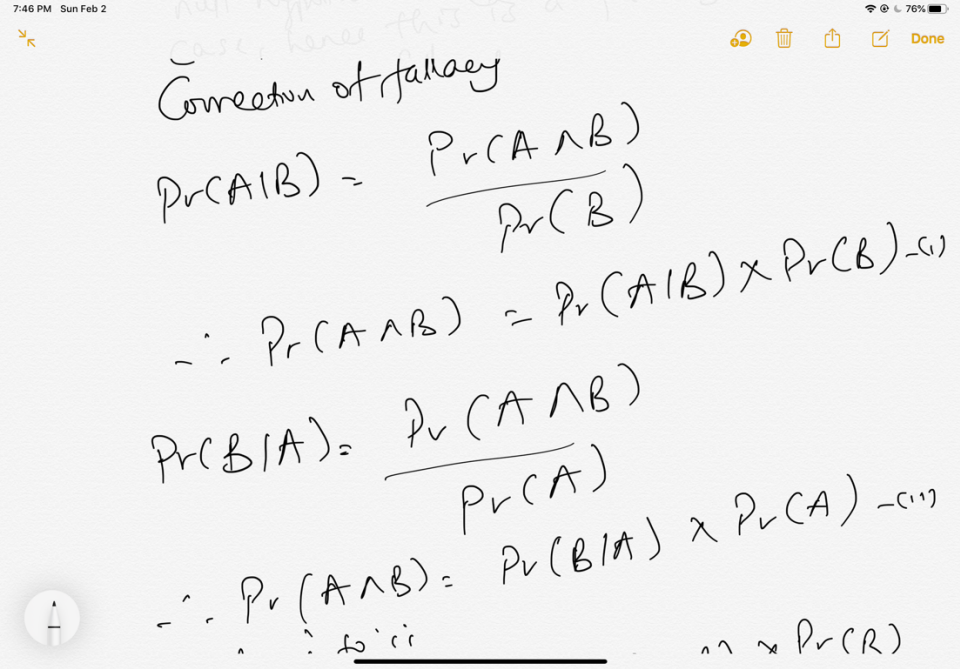
```
In [147]: def t_score_significance(x,y):
              """Function returns the t-score for the hypothesis testing"""
              global t_score
              t_score = slope/std_err #compute t_score
              return t_score

          t_score_significance(living_space,price)

Out[147]: 144.92388392168854

In [148]: def p_value():
              """Function returns the p-value, if output =='nan', the value is too small to be expressed as a float
              that is, approximately == 0"""
              percentile = stats.norm.ppf(t_score,df)
              return 1-percentile
          p_value()

Out[148]: nan
```

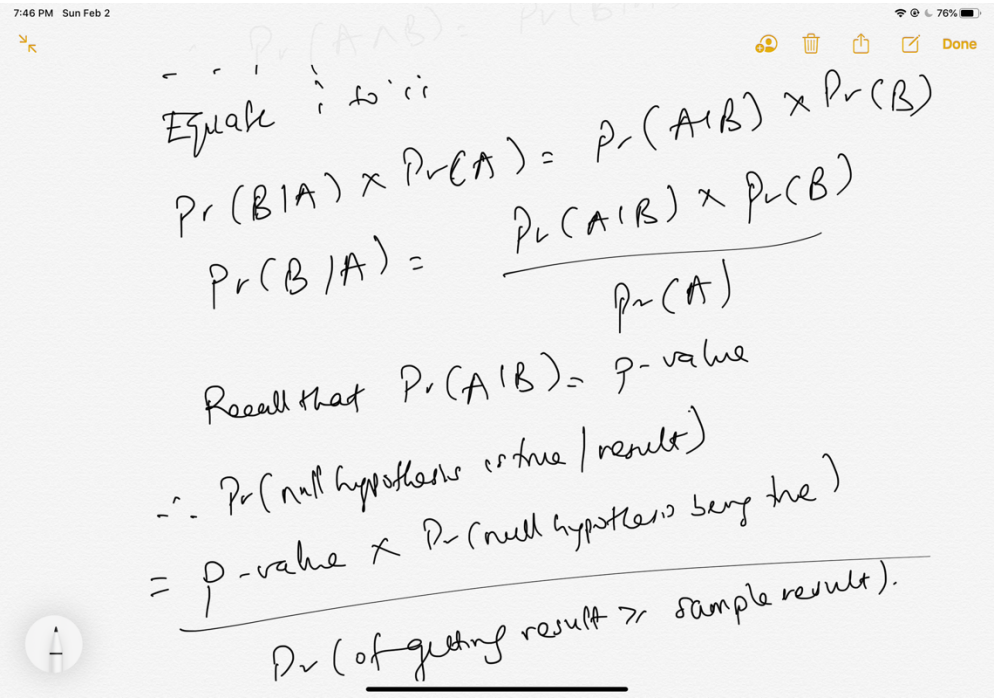
Appendix H- Correcting Common Misinterpretation of p-value




7:45 PM Sun Feb 2
76%
Done
Transforming to formal logic
$= \frac{Pr(A \wedge B)}{Pr(B)}$
Incorrect Interpretation
$Pr(B/A) = \frac{Pr(A \wedge B)}{Pr(A)}$
Transforming to formal logic
$Pr(B/A): \frac{Pr(A \wedge B)}{1}$


7:45 PM Sun Feb 2
76%
Done
Assuming both statements are equivalent
$\therefore \frac{Pr(A \wedge B)}{Pr(A)} = \frac{Pr(A \wedge B)}{Pr(B)}$
$\therefore Pr(B/A) = Pr(A/B)$ if and only if
$Pr(A) = Pr(B)$
that is when the Pr of getting the result = Pr of null hypothesis being true - This is not always the case. It is a fallacy.


7:46 PM Sun Feb 2
76%
Done
Correction of fallacy
$Pr(A|B) = \frac{Pr(A \wedge B)}{Pr(B)}$
$\therefore Pr(A \wedge B) = Pr(A|B) \times Pr(B)$ -(i)
$Pr(B|A) = \frac{Pr(A \wedge B)}{Pr(A)}$
$\therefore Pr(A \wedge B) = Pr(B|A) \times Pr(A)$ -(ii)


7:46 PM Sun Feb 2
76%
Done
Equate (i) to (ii)
$Pr(B|A) \times Pr(A) = Pr(A|B) \times Pr(B)$
$Pr(B|A) = \frac{Pr(A|B) \times Pr(B)}{Pr(A)}$
Recall that $Pr(A|B) = p\text{-value}$
$\therefore Pr(\text{null hypothesis is true} | \text{result})$
$= \frac{p\text{-value} \times Pr(\text{null hypothesis being true})}{Pr(\text{of getting result} \geq \text{sample result})}$.