Problem 1

PROBLEM 1A

Given: n data points : $\{(x_i, y_i)\}_{i=1}^n \sim y_i = x_i \beta_1 + \beta_0 + e$

$$\in \sim dist(0, \sigma^2)$$

RSS estimates $\hat{\beta}_0, \hat{\beta}_1 : \hat{\mu}_0 = \hat{\beta}_0 + \hat{\beta}_1 x_0$

$$\mu_0 = \beta_0 + \beta_1 x_0$$

To find: variance of $\hat{\mu}_0$

We know that β_0 & β_1 have the following form: $\hat{\beta}_0 = \bar{y} - \beta_1 \bar{x}$

$$\hat{\beta}_1 = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$$

$$\Rightarrow \hat{\mu}_0 = \hat{\beta}_0 + \hat{\beta}_1 x_0$$

$$= \bar{y} - \hat{\beta}_1 \bar{x} + \hat{\beta}_1 x_0$$

$$= \bar{y} + \hat{\beta}_1 (x_0 - \bar{x})$$

$$\Rightarrow Var(\hat{\mu}_0) = Var(\bar{y} + \hat{\beta}_1 (x_0 - \bar{x}))$$

$$= Var(\bar{y}) + (x_0 - \bar{x})^2 Var(\hat{\beta}_1) - ① \quad (\because \text{Applying linearity of variance}$$

because $\bar{y}, x_0, \bar{x}$ ~~are all constants~~

~~so independent~~ from $\hat{\beta}_1$)

is uncorrelated

& x_0 & $\bar{x}$ are-provided

#s so they are constant)

According Theorem 14.B in Rice's text book:

$$Var(\hat{\beta}_1) = \frac{\sigma^2}{\sum_{i=1}^n (x_i - \bar{x})^2} \quad - ②$$

We also know $Var(\bar{y}) = \frac{Var(\in)}{n} = \frac{\sigma^2}{n} - ③$

Substituting ② & ③ into ① :

$$Var(\hat{\mu}_0) = \frac{\sigma^2}{n} + (x_0 - \bar{x})^2 \frac{\sigma^2}{\sum_{i=1}^n (x_i - \bar{x})}$$

$$= \sigma^2 \left(\frac{1}{n} + \frac{(x_0 - \bar{x})^2}{\sum_{i=1}^n (x_i - \bar{x})} \right)$$

PROBLEM 1B

We know: $\text{Var}(\hat{\mu}_0) = \dfrac{\sigma^2}{n} + \dfrac{\sigma^2(x_0 - \bar{x})^2}{\sum\limits_{i=1}^{n}(x_i - \bar{x})}$

We can assign variables:

$$\frac{\sigma^2}{n} = \alpha_0 \quad , \quad \frac{\sigma^2}{\sum_{i=1}^{n}(x_i - \bar{x})} = \alpha_1 \quad , \quad (x_0 - \bar{x}) = z$$
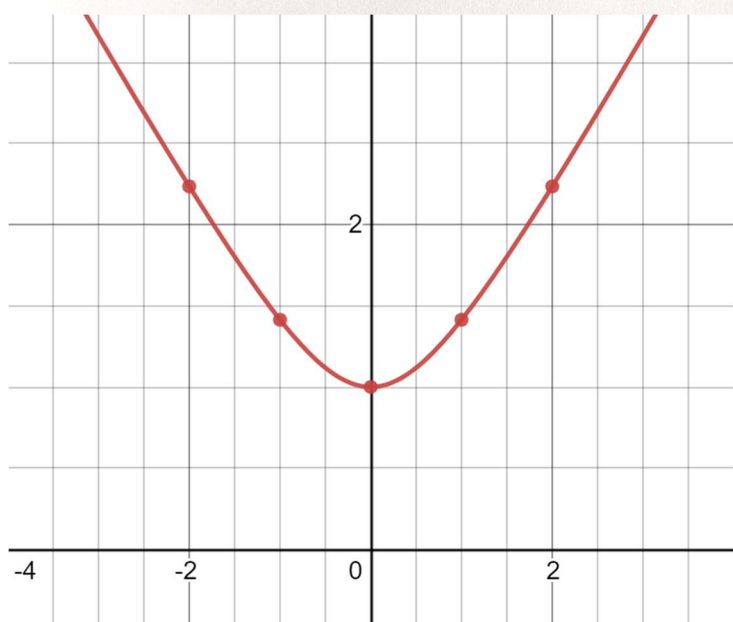
$\Rightarrow \text{Var}(\hat{\mu}_0) = \alpha_0 + \alpha_1 z^2$

$\Rightarrow \text{std.dev}(\hat{\mu}_0) = \sqrt{\alpha_0 + \alpha_1 z^2}$

Graph of $f(z) = \sqrt{\alpha_0 + \alpha_1 z^2}$ where $\alpha_0, \alpha_1 \geqslant 0$

For $\alpha_0 = \alpha_1 = 1$, we get the following graph:

(Digital)

PROBLEM 1C

We know standard deviation of $\hat{\mu}_0 = \sqrt{\dfrac{\sigma^2}{n} + \dfrac{\sigma^2 (x_0 - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})}}$

We also know that the CI can be written as:

$$\hat{\mu}_0 \pm Z_{\alpha/2} \dfrac{\sigma}{\sqrt{n}} \qquad ①$$

Alternatively: $\hat{\mu}_0 \pm s_{\hat{\mu}_0} t_{n-2}\left(\dfrac{\alpha}{2}\right) \qquad ②$

Substituting $\sigma = s_{\hat{\mu}_0} = \sqrt{\dfrac{\sigma^2}{n} + \dfrac{\sigma^2(x_0 - \bar{x})^2}{\sum_{i=1}^{n}(x_i - \bar{x})}}$,

we have the 95% confidence interval $(\alpha = 0.05)$

Problem 2

PROBLEM 2

Given: $X \sim N(0,1)$; $E \sim N(0,1)$; $Y = X + \beta E$

Prove:

$$r_{XY} = \frac{1}{\sqrt{\beta^2 + 1}} \qquad \text{where} \quad r_{XY} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

$$r_{X,Y} = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

$$= \frac{cov(X, X+\beta E)}{\sigma_X \sigma_{X+\beta E}} = \frac{cov(X,X) + cov(X, \beta E)}{\sigma_X \sqrt{Var(X+\beta E)}}$$

$$= \frac{cov(X,X) + \beta cov(X,E)}{1 \sqrt{Var(X) + \beta^2 Var(E)}} = \frac{Var(X) + 0}{\sqrt{1 + \beta^2}}$$

$$\left(\because cov(X,X) = Var(X) \right.$$
$$Var(X) = Var(E) = 1$$
$$\left. cov(X,E) = 0 \right)$$

$$= \frac{1}{\sqrt{1 + \beta^2}}$$

Problem 3

PROBLEM 3

Given: Given $\{x_i \in \mathbb{R}, y_i \in \mathbb{R}\}_{i=1}^n$

$$y = a + bx \quad ; \quad x = c + dy \qquad \text{w/ least squares}$$

Prove: $bd \le 1$

Hint: $|cov(X,Y)|^2 \le Var(X) Var(Y)$

$$b = \frac{cov(X,Y)}{Var(X)} \quad ; \quad d = \frac{cov(Y,X)}{Var(Y)}$$

$$\Rightarrow bd = \frac{(cov(X,Y))^2}{Var(X) Var(Y)} \le \frac{Var(X) Var(Y)}{Var(X) Var(Y)} = 1$$

$$\Rightarrow bd \le 1$$

explanation: $bd = 1$ when $|cov(X,Y)|^2 = Var(X) Var(Y)$ $\Rightarrow$ correlation of $+1$
i.e. X & Y are moving in the same direction together.

Problem 4

PROBLEM 4

Given: $Y \in \mathbb{R}^n$ & $X_1, X_2 \in \mathbb{R}^n$

$X_3 = X_1 + X_2$; use multiple regression to predict Y from X_1, X_2 & X_3

Hints: $A_{n,n}$ is invertible $\iff$ Rank$(A) = n$

Rank$(AB) \leq \min($Rank$(A),$ Rank$(B))$

Prove: Why this will not work.

$Y = \beta X + E$: $Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix}$, $\beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \beta_3 \end{bmatrix}$, $X = \begin{bmatrix} 1 & X_{1,1} & X_{1,2} & X_{1,3} \\ \vdots & \vdots & \vdots & \vdots \\ 1 & X_{n,1} & X_{n,2} & X_{n,3} \end{bmatrix}$

Looking at X, its column rank is 3 $\therefore$ the last column is a linear combination of other columns in the matrix. Thus, it does not have full column rank & cannot be invertible.

Least squares estimator for β:

$Y_i | X_i = x_i \sim N\left(\beta_0 + \sum_{j=1}^{p} \beta_j x_{ij}, \sigma^2\right)$ $(\because Y = X\beta + \epsilon)$

Least squares error: $(y - X\beta)^T (y - X\beta)$

$\Rightarrow$ optimal error is where $X^T X\beta = X^T y$ $\Rightarrow$ $\beta = (X^T X)^{-1} X^T Y$
minimum

However, given rank of $X =$ rank of $X^T = 3$, we get:

rank$(X^T X) = \min\{3, 3\} = 3$

$\Rightarrow$ $X^T X$ is not invertible because it does not have full rank

$\Rightarrow$ we cannot find β because $(X^T X)^{-1}$ is required to find this

Problem 5: Simple Linear Regression

ESE402 HW4

Sheil Sarda <sheils@seas.upenn.edu>

In this question, you will implement simple linear regression from scratch. The dataset you will work with is called the Boston data set. You can find more information about the data set here: https://www.cs.toronto.edu/~delve/data/boston/bostonDetail.html

You will use the pandas library to load the csv file into a dataframe:

```
1 import numpy as np
2 import pandas as pd
3 import matplotlib.pyplot as plt
4 %matplotlib inline
```

```
1 # read the csv file and load into a pandas dataframe
2 # make sure Boston.csv is in the same file path as this notebook
3 boston = pd.read_csv('Boston.csv')
```

```
1 # read the above link to learn more about what each of the columns indicate
2 boston.head()
```

	crim	zn	indus	chas	nox	rm	age	dis	rad	tax	ptratio	lstat
0	0.00632	18.0	2.31	0	0.538	6.575	65.2	4.0900	1	296	15.3	4.98
1	0.02731	0.0	7.07	0	0.469	6.421	78.9	4.9671	2	242	17.8	9.14
2	0.02729	0.0	7.07	0	0.469	7.185	61.1	4.9671	2	242	17.8	4.03
3	0.03237	0.0	2.18	0	0.458	6.998	45.8	6.0622	3	222	18.7	2.94
4	0.06905	0.0	2.18	0	0.458	7.147	54.2	6.0622	3	222	18.7	5.33

Simple linear regression builds a linear relationship between an input variable X and an output variable Y. We can define this linear relationship as follows:

$$Y = \beta_0 + \beta_1 X$$

Objective: find the linear relationship between the proportion of non-retail business acres per town (indus) and the full-value property-tax rate per 10,000 dollars (tax)

So our equation will look like:

$$TAX = \beta_0 + \beta_1 INDUS$$

Here, the coefficient β_0 is the intercept, and β_1 is the scale factor or slope. How do we determine the values of these coefficients?

There are several different methods to do so, but we will focus on the Ordinary Least Squares (OLS) method. This method minimizes the sum of the squares of the differences between the observed dependent variable and those predicted by the linear function.

Recall that a residual is the difference between any data point and the line of regression. When we develop a regression model, we want the sum of the residuals squared to be minimized, indicating that the model is a close fit to the data.

$$RSS = \sum_{i=1}^{n}(y_i - f(x_i))^2$$
$$= \sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_i))^2$$

This is the objective function we minimize to find β_0 and β_1.

```
1 # set X to 'indus' and y to 'tax'
2 X = boston['indus']
3 y = boston['tax']
```

First, visualize the data by plotting X and y using matplotlib. Be sure to include a title and axis labels.

```
1 # TODO: display plot
2 plt.rcParams["figure.figsize"] = (10,5)
3 plt.scatter(X, y, label="Boston", marker='x', alpha=0.8, c='Green')
4 # TODO: labels and title
5 plt.title("Property-tax rate vs non-retail business acres")
6 plt.xlabel("Proportion of non-retail business acres per town (%)")
7 plt.ylabel("Property-tax rate per 10,000 dollars (USD)")
8 plt.grid()
9 plt.legend()
10 plt.savefig("boston_indus_tax.png")
11 plt.show()
```

Property-tax rate vs non-retail business acres
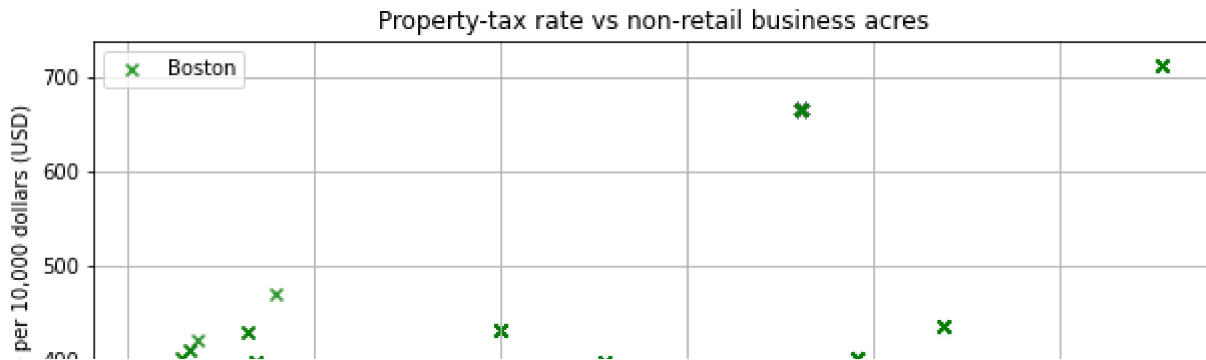
- ▾ TODO: What do you notice about the relationship between the variables?

A: In general, there is a positive correlation between the proportion of non-retail business acres per town and business taxes. A potential driver for this relationship could be that as the number of businesses in a town increase, the town's need to provide lower taxes as an incentive for companies to operate there and provide employment to the region declines.

Next, find the coefficients. The values for β_0 and β_1 are given by the following equations, where n is the total number of values. This derivation was done in class.

$$\beta_1 = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1\bar{x}$$

```
1 # TODO: implement function
2 def get_coeffs(X, y):
3     """
4     Params:
5         X: the X vector
6         y: the y vector
7     Returns:
8         a tuple (b1, b0)
9     """
10    X_mean = sum(X) / float(len(X))
11    y_mean = sum(y) / float(len(y))
12    X_var = sum((X - X_mean)**2)/float(len(X) - 1)
13
14    covariance = 0.0
15    for i in range(len(X)):
16        covariance += (X[i] - X_mean) * (y[i] - y_mean)
17    covariance = covariance / float(len(X) - 1)
18
19    b1 = covariance / X_var
20    b0 = y_mean - b1 * X_mean
21    return (b1, b0)
```

```
1 # run cell to call function and display the regression line
2 # the values are rounded for display convenience
3 b1, b0 = get_coeffs(X, y)
4 print("Regression line: TAX = " + str(round(b0)) + " + " + " + str(round(b1)) +"*INDUS")
```

      Regression line: TAX = 211 + 18*INDUS

Plot the regression line overlayed on the real y-values.

```
1 # TODO: plot y-values
2 plt.rcParams["figure.figsize"] = (10,5)
3 plt.scatter(X, y, label="Boston", marker='x', alpha=0.8, c='Green')
4
5 # TODO: plot regression line
6 y_predicted = X*b1 + b0
7 plt.plot(X, y_predicted, label="Regression Line", c='Purple')
8
9 # TODO: labels and title
10 plt.title("Property-tax rate vs non-retail business acres")
11 plt.xlabel("Proportion of non-retail business acres per town (%)")
12 plt.ylabel("Property-tax rate per 10,000 dollars (USD)")
13 plt.grid()
14 plt.legend()
15 plt.show()
```
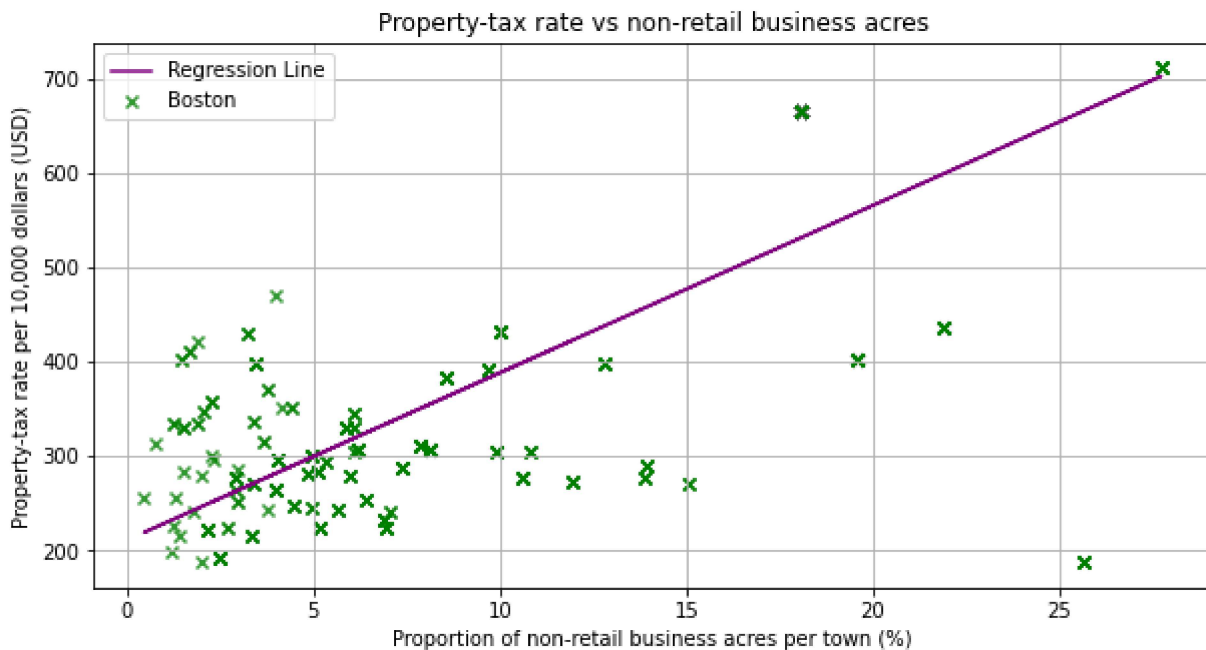

The line appears to fit the data, but first, let us find the RSS to evaluate this model. The RSS is used to measure the amount of variance in the data set that is not explained by the regression model. Recall that

$$RSS = \sum_{i=1}^{n}(y_i - (\beta_0 + \beta_1 x_i))^2$$

```
1 # TODO: implement function
2 def get_RSS(b0, b1, X, y):
3     """
4     Params:
5         b0: beta 0
6         b1: beta 1
7         X: X vector
8         y: y vector
9     Returns:
10        residual sum of squares (RSS)
11     """
12     rss = 0.0
13     for i in range(len(y)):
14         rss += (y[i] - (b0 + b1*X[i]))**2
15     return rss
```

```
1 # run this cell to print RSS
2 print("RSS:", get_RSS(b0, b1, X, y))
```

```
RSS: 6892554.224031512
```

We can also evaluate the model through the Root Mean Squared Error (RMSE) and the Coefficient of Determination (R^2 score).

- The RMSE is similar to the RSS, but provides a value with more interpretable units -- in our case, tax rate per 10,000 dollars.
- The R^2 value represents the proportion of the variance for the dependent variable that is explained by the independent variable.

Use the following equations to find the RMSE and R^2 score:

$$RMSE = \sqrt{(\sum_{i=1}^{n}\frac{1}{n}(\hat{y}_i - y_i)^2)}$$

$$R^2 = 1 - \frac{SS_r}{SS_t}$$

$$SS_t = \sum_{i=1}^{n}(y_i - \bar{y})^2$$

$$SS_r = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

```python
1 # TODO: implement function
2 def get_RMSE(b0, b1, X, y):
3     """
4     Params:
5         b0: beta 0
6         b1: beta 1
7         X: X vector
8         y: y vector
9     Returns:
10        rmse
11    """
12    rss = get_RSS(b0, b1, X, y)
13    rmse = (rss/len(y))**0.5
14    return rmse
```

```python
1 # run cell to print RMSE
2 print("RMSE: ", get_RMSE(b0, b1, X, y))
```

```
RMSE:  116.7118188706435
```

```python
1 # TODO: implement function
2 def get_R2(b0, b1, X, y):
3     """
4     Params:
5         b0: beta 0
6         b1: beta 1
7         X: X vector
8         y: y vector
9     Returns:
10        r2 score
11    """
12    SS_t = 0.0
13    y_mean = sum(y) / float(len(y))
14    for i in range(len(y)):
15        SS_t += (y[i] - y_mean)**2
16
17    SS_r = get_RSS(b0, b1, X, y)
18    r2 = 1 - SS_r / SS_t
19    return r2
```

```python
1 # run cell to print RMSE
2 print("R2: ", get_R2(b0, b1, X, y))
```

```
R2:  0.5194952370037837
```

▼ Q: Analyze what the above R^2 score indicates about the model.

A: Usually, the larger the R2, the better the regression model fits your observations. In this case, an R^2 of 52% implies the model explains 52% of the variation in the response variable around its mean.

A caveat about this metric is that it cannot determine whether the coefficient estimates and predictions are biased, which is why we must also assess residual plots.

Now, we will compare the above results with the results from using scikit-learn, a machine learning library in Python. Read the documentation (https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html) to learn how to use this library. Return the R^2 score and RMSE.

```python
1 # TODO: scikit learn function
2 from sklearn.linear_model import LinearRegression
3 from sklearn.metrics import mean_squared_error
4 from sklearn.metrics import r2_score
5
6 def linear_regression_SKL(X, y):
7     """
8     Params:
9         X: X vector
10        y: y vector
11    Returns:
12        rmse and r2 as a tuple
13    """
14    linear_regression = LinearRegression()
15    X_reshaped = np.array(X).reshape(len(X),1)
16    linear_regression.fit(X_reshaped, y)
17    y_predicted = linear_regression.predict(X_reshaped)
18
19    rmse = mean_squared_error(y, y_predicted, squared=False)
20    r2 = r2_score(y, y_predicted)
21
22    return (rmse, r2)
```

```python
1 # run this cell to print results from SKL LR
2 rmse, r2 = linear_regression_SKL(X, y)
3 print("RMSE: ", rmse)
4 print("R2: ", r2)
```

```
RMSE:  116.71181887064391
R2:  0.5194952370037791
```

Q: Analyze the results and compare the RMSE and R^2 to the previous method.

A: Comparing R^2 and $RMSE$ of both implementations:

Implementation	R^2	$RMSE$
Sci-kit Learn	0.5194952370037791	116.71181887064391
Sheil's Implementation	0.5194952370037837	116.7118188706435

This result helps validate our implementation of linear regression, since the two sets of predictions (from sci-kit learn and our own implementation) yield the same scores.