Homework 3 - Math 4803

Dennis Goldenberg

March 2023

1 Theoretical Questions

1. Deriving Equation 5.2 on page 202

$$\mathrm{CV}_{(n)} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - h_i}\right)^2$$

Proof. Before we begin the proof, we are going to define 4 different equations as variables to make computation easier:

$$\begin{aligned}
a &= \sum_{i=1}^{n}(x_i - \bar{x})^2 \\
b &= \sum_{i=1}^{n}(x_i - \bar{x}) = \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} \bar{x} = n\bar{x} - n\bar{x} = 0 \\
c &= \sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y}) \\
d &= \sum_{i=1}^{n}(y_i - \bar{y}) = \sum_{i=1}^{n} y_i - \sum_{i=1}^{n} \bar{y} = n\bar{y} - n\bar{y} = 0
\end{aligned}$$

For typical least-squares linear regression, here are the coefficients arrived at given the data (equation 3.4) given the textbook:

$$\begin{aligned}
\hat{\beta}_1 &= \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2}, \\
\hat{\beta}_0 &= \bar{y} - \hat{\beta}_1\bar{x},
\end{aligned}$$

Note here that $\hat{\beta}_1 = \frac{c}{a}$, and $\hat{\beta}_0 = \bar{y} - \frac{c}{a}\bar{x}$, so:

$$\begin{aligned}
\hat{y_j} &= \beta_0 + \beta_1 x_j \\
&= \bar{y} - \frac{c}{a}\bar{x} + \frac{c}{a}x_j \\
&= \bar{y} + \frac{c}{a}\left(x_j - \bar{x}\right)
\end{aligned}$$

Then, as a result, we can calculate the residual:

$$\begin{aligned} y_j - \hat{y_j} &= y_j - \bar{y} - \frac{c}{a}(x_j - \bar{x}) \\ &= (y_j - \bar{y}) - \frac{c}{a}(x_j - \bar{x}) \end{aligned}$$

For Leave-One-Out-Cross-Validation (LOOCV), we train the dataset on all but one element. The formula we are given is an average of mean-squared errors for each possible case (averaging over LOOCV where every individual element is left out once). Therefore, if we pick some arbitrary point (x_j, y_j), and call the predicted outcome using LOOCV y_j^*, the mean squared error would simply be $(y_j - y_j^*)^2$, as the test set is only one element. Therefore, to derive the formula for average cross-validation Mean Squared error, it would be sufficient to prove that:

$$\forall j \in \{1, 2, .., n\}, y_j - y_j^* = \frac{y_j - \hat{y_j}}{1 - h_j} \quad (1)$$

Here h_j is the leverage represented by the formula:

$$h_j = \frac{1}{n} + \frac{(x_j - \bar{x})^2}{\sum_{i=1}^n (x_i - \bar{x})^2} = \frac{1}{n} + \frac{(x_j - \bar{x})^2}{a}$$

Note that, as a result:

$$\begin{aligned} 1 - h_j &= 1 - \frac{1}{n} - \frac{(x_j - \bar{x})^2}{a} \\ &= \frac{n-1}{n} - \frac{(x_j - \bar{x})^2}{a} \\ &= \frac{(n-1)a - n(x_j - \bar{x})^2}{na} \end{aligned}$$

Therefore:

$$\frac{1}{1 - h_j} = \frac{na}{(n-1)a - n(x_j - \bar{x})^2} \quad (2)$$

Now, we can begin to solve. we first note that, if (x_j, y_j) is our selected data point to leave out, we have different averages on the dataset:

$$\bar{x_j} = \frac{1}{n-1} \sum_{i=1, \neq j}^{n} x_i = \frac{n\bar{x} - x_j}{n-1}$$

and

$$\bar{y_j} = \frac{1}{n-1} \sum_{i=1, \neq j}^{n} y_i = \frac{n\bar{y} - y_j}{n-1}$$

Note that, with these new averages, the coefficients that would be calculated would also be different; call them $\hat{\beta}_{0j}$ and $\hat{\beta}_{1j}$. Therefore, if we calculate the new residual:

$$\begin{aligned}
y_j - y_j^* &= y_j - (\hat{\beta}_{0j} + \hat{\beta}_{1j} x_j) \\
&= y_j - (\bar{y}_j - \hat{\beta}_{1j}\bar{x}_j + \hat{\beta}_{1j} x_j) \\
&= (y_j - \bar{y}_j) - (x_j - \bar{x}_j)\hat{\beta}_{1j} \\
&= \left(y_j - \frac{n\bar{y} - y_j}{n-1}\right) - \left(x_j - \frac{n\bar{x} - x_j}{n-1}\right)\hat{\beta}_{1j} \\
&= \frac{n}{n-1}(y_j - \bar{y}) - \frac{n}{n-1}(x_j - \bar{x})\hat{\beta}_{1j} \\
&= \left(\frac{n}{n-1}\right)\left(y_j - \bar{y} - (x_j - \bar{x})\hat{\beta}_{1j}\right) \\
&= \left(\frac{n}{n-1}\right)\left(y_j - \bar{y} - (x_j - \bar{x})\left(\frac{\sum_{i=1,\neq j}^{n}(x_i - \bar{x_j})(y_j - \bar{y_j})}{\sum_{i=1,\neq j}^{n}(x_i - \bar{x_j})^2}\right)\right) \quad \text{(i)}
\end{aligned}$$

We plug in the alternative versions of the averages continue from (i):

$$\begin{aligned}
&y_j - y_j^* \\
&= \left(\frac{n}{n-1}\right)\left(y_j - \bar{y} - (x_j - \bar{x})\left(\frac{\sum_{i=1,\neq j}^{n}(x_i - \bar{x_j})(y_j - \bar{y_j})}{\sum_{i=1,\neq j}^{n}(x_i - \bar{x_j})^2}\right)\right) \\
&= \left(\frac{n}{n-1}\right)\left(y_j - \bar{y} - (x_j - \bar{x})\left(\frac{\sum_{i=1,\neq j}^{n}\left(x_i - \bar{x} + \frac{x_j - \bar{x}}{n-1}\right)\left(y_j - \bar{y} + \frac{y_j - \bar{y}}{n-1}\right)}{\sum_{i=1,\neq j}^{n}\left(x_i - \bar{x} + \frac{x_j - \bar{x}}{n-1}\right)^2}\right)\right) \\
&= \left(\frac{n}{n-1}\right)\left(\frac{(y_j - \bar{y}) * f - (x_j - \bar{x}) * g}{f}\right) \quad \text{(ii)}
\end{aligned}$$

Here:

$$f = \sum_{i=1,\neq j}^{n}\left(x_i - \bar{x} + \frac{x_j - \bar{x}}{n-1}\right)^2$$

and

$$g = \sum_{i=1,\neq j}^{n}\left(x_i - \bar{x} + \frac{x_j - \bar{x}}{n-1}\right)\left(y_i - \bar{y} + \frac{y_j - \bar{y}}{n-1}\right)$$

Let us break this down, piece by piece; first, we simplify f:

$$\begin{aligned}
f &= \sum_{i=1,\neq j}^{n}\left(x_i - \bar{x} + \frac{x_j - \bar{x}}{n-1}\right)^2 \\
&= \sum_{i=1,\neq j}^{n}(x_i - \bar{x})^2 + \frac{2 * (x_j - \bar{x})}{n-1}\sum_{i=1,\neq j}^{n}(x_i - \bar{x}) + \sum_{i=1,\neq j}^{n}\left(\frac{x_j - \bar{x}}{n-1}\right)^2 \\
&= a - (x_j - \bar{x})^2 + \frac{2(x_j - \bar{x})}{n-1} * (0 - (x_j - \bar{x})) + \frac{(x_j - \bar{x})^2}{n-1}
\end{aligned}$$

We reduce to get:

$$\begin{aligned} f &= a + \left(-1 - \frac{2}{n-1} + \frac{1}{n-1}\right)(x_j - \bar{x})^2 \\ &= a + \left(-1 - \frac{1}{n-1}\right)(x_j - \bar{x})^2 \\ &= a - \left(\frac{n}{n-1}\right)(x_j - \bar{x})^2 \end{aligned}$$

Now, we simplify g:

$$\begin{aligned} g &= \sum_{i=1,\neq j}^{n} \left(x_i - \bar{x} + \frac{x_j - \bar{x}}{n-1}\right)\left(y_i - \bar{y} + \frac{y_j - \bar{y}}{n-1}\right) \\ &= \sum_{i=1,\neq j}^{n} (x_i - \bar{x})(y_i - \bar{y}) + \sum_{i=1,\neq j}^{n} \frac{(x_i - \bar{x})(y_j - \bar{y})}{n-1} + \sum_{i=1,\neq j}^{n} \frac{(y_i - \bar{y})(x_j - \bar{x})}{n-1} \\ &+ \frac{(y_j - \bar{y})(x_j - \bar{x})}{n-1} \\ &= (c - (x_j - \bar{x})(y_j - \bar{y})) + \frac{y_j - \bar{y}}{n-1}(b - (x_j - \bar{x})) + \frac{x_j - \bar{x}}{n-1}(d - (y_j - \bar{y})) \\ &+ \frac{(y_j - \bar{y})(x_j - \bar{x})}{n-1} \\ &= c + (x_j - \bar{x})(y_j - \bar{y})\left(-1 - \frac{2}{n-1} + \frac{1}{n-1}\right) \\ &= c - \left(\frac{n}{n-1}\right)(x_j - \bar{x})(y_j - \bar{y}) \end{aligned}$$

We plug our simplified formulas for f and g back into (ii):

$$\begin{aligned} y_j - y_j^* &= \left(\frac{n}{n-1}\right)\left(\frac{(y_j - \bar{y}) * f - (x_j - \bar{x}) * g}{f}\right) \\ &= \left(\frac{n}{n-1}\right)\left(\frac{(y_j - \bar{y}) * \left(a - \left(\frac{n}{n-1}\right)(x_j - \bar{x})^2\right) - (x_j - \bar{x}) * g}{\left(a - \left(\frac{n}{n-1}\right)(x_j - \bar{x})^2\right)}\right) \\ &= n * \left(\frac{(y_j - \bar{y}) * \left(a - \left(\frac{n}{n-1}\right)(x_j - \bar{x})^2\right) - (x_j - \bar{x}) * g}{(n-1)a - n(x_j - \bar{x})^2}\right) \quad \text{(iii)} \end{aligned}$$

Now, we simplify the numerator:

$$\begin{aligned}
&(y_j - \bar{y}) * \left(a - \left(\frac{n}{n-1}\right)(x_j - \bar{x})^2\right) - (x_j - \bar{x}) * g \\
&= a(y_j - \bar{y}) - \frac{n}{n-1}(x_j - \bar{x})^2(y_j - \bar{y}) - (x_j - \bar{x})(c - \frac{n}{n-1}(x_j - \bar{x})(y_j - \bar{y})) \\
&= a(y_j - \bar{y}) - c(x_j - \bar{x}) - \frac{n}{n-1}(x_j - \bar{x})^2(y_j - \bar{y}) + \frac{n}{n-1}(x_j - \bar{x})^2(y_j - \bar{y}) \\
&= a(y_j - \bar{y}) - c(x_j - \bar{x})
\end{aligned}$$

So, we continue from (iii):

$$\begin{aligned}
y_j - y_j^* &= n * \left(\frac{(y_j - \bar{y}) * \left(a - \left(\frac{n}{n-1}\right)(x_j - \bar{x})^2\right) - (x_j - \bar{x}) * g}{(n-1)a - n(x_j - \bar{x})^2}\right) \\
&= \frac{n(a(y_j - \bar{y}) - c(x_j - \bar{x}))}{(n-1)a - n(x_j - \bar{x})^2} \\
&= \frac{n(a(y_j - \bar{y}) - c(x_j - \bar{x}))}{na} * \frac{na}{(n-1)a - n(x_j - \bar{x})^2} \\
&= \frac{a(y_j - \bar{y}) - c(x_j - \bar{x})}{a} * \frac{1}{1 - h_j} \text{ (from 1)} \\
&= \left(y_j - \bar{y} - \frac{c}{a}(x_j - \bar{x})\right) * \frac{1}{1 - h_j} \\
&= (y_j - \hat{y}_j) * \frac{1}{1 - h_j} \\
&= \frac{y_j - \hat{y}_j}{1 - h_j}
\end{aligned}$$

Thus:

$$\begin{aligned}
&\forall j, y_j - y_j^* = \frac{y_j - \hat{y}_j}{1 - h_j} \\
&\Rightarrow (y_j - y_j^*)^2 = \left(\frac{y_j - \hat{y}_j}{1 - h_j}\right)^2 \\
&\Rightarrow CV_{(n)} = \frac{1}{n}\sum_{j=1}^{n}(y_j - y_j^*)^2 = \frac{1}{n}\sum_{j=1}^{n}\left(\frac{y_j - \hat{y}_j}{1 - h_j}\right)^2
\end{aligned}$$

$\square$

2. Section 7.9 Exercise 1

Solution

a) - Note that, for $x \leq \xi$, $(x-\xi)_+^3 = 0$; therefore:

$$\forall x \leq \xi, f(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4(0) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$

Thus, to fit said polynomial, set $a_1 = \beta_0$, $b_1 = \beta_1$, $c_1 = \beta_2$ and $d_1 = \beta_3$, and you get:

$$f_1(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 = f(x)$$

b) - When $x \geq \xi$, we have that, grouping like terms together:

$$\begin{aligned} f(x) &= \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x-\xi)^3 \\ &= \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3 + \beta_4 (x^3 - 3x^2\xi + 3x\xi^2 - \xi^3) \\ &= (\beta_0 - \beta_4\xi^3) + (\beta_1 + 3\beta_4\xi^2)x + (\beta_2 - 3\beta_4\xi)x^2 + (\beta_3 + \beta_4)x^3 \end{aligned}$$

We set $a_2 = \beta_0 - \beta_4\xi^3$, $b_2 = \beta_1 + 3\beta_4\xi^2$, $c_2 = \beta_2 - 3\beta_4\xi$, $d_2 = \beta_3 + \beta_4$; thus:

$$\begin{aligned} f(x) &= (\beta_0 - \beta_4\xi^3) + (\beta_1 + 3\beta_4\xi^2)x + (\beta_2 - 3\beta_4\xi)x^2 + (\beta_3 + \beta_4)x^3 \\ &= a_2 + b_2 x + c_2 x^2 + d_2 x^3 \\ &= f_2(x) \end{aligned}$$

c) - We plug in ξ for the second equation to get:

$$\begin{aligned} f_2(\xi) &= (\beta_0 - \beta_4\xi^3) + (\beta_1 + 3\beta_4\xi^2)\xi + (\beta_2 - 3\beta_4\xi)\xi^2 + (\beta_3 + \beta_4)\xi^3 \\ &= \beta_0 + \beta_1\xi + \beta_2\xi^2 + \beta_3\xi^3 - \beta_4\xi^3 + 3\beta_4\xi^3 - 3\beta_4\xi^3 + \beta_4\xi^3 \\ &= \beta_0 + \beta_1\xi + \beta_2\xi^2 + \beta_3\xi^3 \\ &= f_1(\xi) \end{aligned}$$

d) - We take the derivative of both equations:

$$f_1'(x) = \beta_1 + 2\beta_2 x + 3\beta_3 x^2$$

and

$$f_2'(x) = (\beta_1 + 3\beta_4\xi^2) + 2(\beta_2 - 3\beta_4\xi)x + 3(\beta_3 + \beta_4)x^2$$

Plugging in ξ to the second equation, we get:

$$\begin{aligned} f_2'(\xi) &= (\beta_1 + 3\beta_4\xi^2) + 2\beta_2\xi - 6\beta_4\xi^2 + 3\beta_3\xi^2 + 3\beta_4\xi^2 \\ &= \beta_1 + 2\beta_2\xi + 3\beta_3\xi^2 + 3\beta_4\xi^2 + 3\beta_4\xi^2 - 6\beta_4\xi^2 \\ &= \beta_1 + 2\beta_2\xi + 3\beta_3\xi^2 \\ &= f_1'(\xi) \end{aligned}$$

e) - We take the second derivative of both equations:

$$f_1''(x) = 2\beta_2 + 6\beta_3 x$$

and

$$f_2''(x) = 2(\beta_2 - 3\beta_4\xi) + 6(\beta_3 + \beta_4)x$$

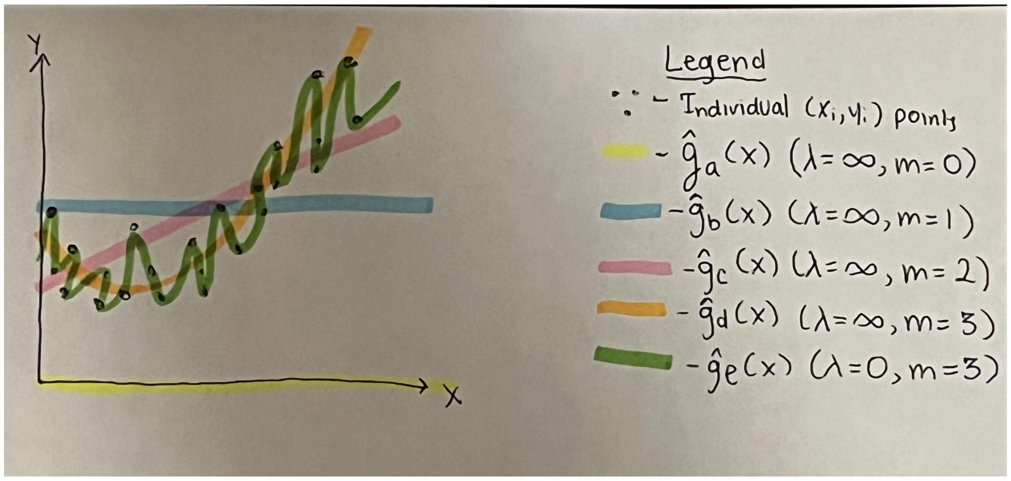
We plug in ξ to get:

$$\begin{aligned} f_2''(\xi) &= 2(\beta_2 - 3\beta_4\xi) + 6(\beta_3 + \beta_4)\xi \\ &= 2\beta_2 - 6\beta_4\xi + 6\beta_3\xi + 6\beta_4\xi \\ &= 2\beta_2 + 6\beta_3\xi \\ &= f_1''(\xi) \end{aligned}$$

3. Section 7.9 Exercise 2

$$\hat{g} = \arg\min_{g} \left(\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int \left[g^{(m)}(x) \right]^2 dx \right)$$

Solution



All $\hat{g}$ curves are pictured above; below are explanations.
a) - Here we have that $\lambda = \infty$. Therefore the sum of squared residuals (SSR) term becomes far outweighed in size by the integral term. Note that we are integrating over the squared values of $g(x)$ over the domain of x. Since the term inside the integral is $g(x)^2$, we know it is non-negative, so minimizing it means trying to reduce the integral to as close to 0 as possible. Note that, for $g(x)$, since we are integrating over the values themselves, the best way to do this is to just set $\hat{g}(x)$ to 0 for all x, which is illustrated in **yellow**.
b) - This case is the same as a), except $m = 1$; therefore, we are integrating over $g'(x)^2$, a non-negative function via the square, and $g'(x)$ is just the rate of change of $g(x)$ instantaneously at any given x, to minimize this, we must ensure that the rate of change is always 0; this means a constant function, but we are now not constrained only to $\hat{g}(x) = 0$; to minimize the SSR term with the constraint of a constant function, we can use $\hat{g}(x) = \bar{y}$, as the average would, when summed over all y's, be the least wrong without any additional information; this is illustrated in **blue**.
c) - This is also the same as a), except now $m = 2$; we are minimizing the integral term over second derivatives $g''(x)$, or the rate of change of the rate of change. Setting this to 0 everywhere results in a curve with a constant rate of

change; as the textbook indicates, when this is the case, the entire problem simplifies to a Least-Squares linear problem, as a line is the curve with maximum flexibility that still satisfies the $g''(x) = 0 \; \forall x$ constraint, but minimizes the RSS term. This is illustrated in **pink**.
d) - Once again, we have the same problem as a), except $m = 3$, so are minimizing the integral term over third derivatives $g'''(x)$; as with cases a-c, in this case $g'''(x) = 0 \; \forall x$ deals with the $\lambda = \infty$ problem. The most flexible equation that satisfies this constraint is a polynomial degree 2, or $\hat{g}(x) = ax^2 + bx + c$; since the data looks roughly quadratic, we fit it best with said curve to minimize RSS while still satisfying the constraint. This is shown in **orange**.
e) - In this case, as $\lambda = 0$, the integral term, via multiplication, gets wiped out; the new problem is:

$$\hat{g} = \arg\min_{g} \left\{ \sum_{i=1}^{n} (y_i - g(x_i))^2 \right\}$$

Note that m is completely absent in this equation, and so we have absolutely no constraints on the flexibility of our function. Therefore, we can literally create a function that makes the RSS approach 0 by having $g(x)$ go through every single point (provided there are no two data points with the same x value but different i's; in this case the curve would be a rough approximation) of the dataset. This is shown in **green**.

4. Section 7.9 Exercise 5

$$\hat{g}_1 = \arg\min_g \left(\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int \left[g^{(3)}(x) \right]^2 dx \right)$$

$$\hat{g}_2 = \arg\min_g \left(\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int \left[g^{(4)}(x) \right]^2 dx \right)$$

Solution

a) - Note that, as $\lambda \to \infty$, the integral term tends to dominate the sum. To minimize both functions, then, would be to set $g_1^{(3)}(x) = 0 \ \forall x$ and $g_2^{(4)}(x) = 0$ $\forall x$, as, due to those terms being squared before they are integrated, having a value of 0 is the smallest possible value. Note that having the third derivative equal to 0 everywhere is a stronger constraint than having the fourth derivative equal to 0 everywhere, as:

$$\forall x, g_1^{(3)}(x) = 0 \Rightarrow g_1^{(4)}(x) = 0$$

. Therefore, $\hat{g}_2$ would be allowed more flexibility (including potentially a non-zero cubic term in the function) than $\hat{g}_1$ meaning that:

- $\hat{g}_2$'s RSS value on training data is likely smaller than $\hat{g}_1$'s

b) - Conversely, with added flexibility comes the increased possibility that $\hat{g}_2$ will in fact overfit to the training data and be less generalizible. In this context, the increased restriction of having $g_1^{(3)}(x) = 0 \ \forall x$ may work to prevent overfitting; therefore:

- $\hat{g}_1$'s RSS value on training data is likely smaller than $\hat{g}_2$'s

c) - At $\lambda = 0$, the integral term completely disappears; therefore:

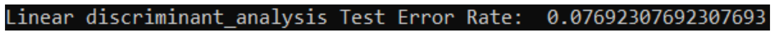
$$\hat{g}_1 = \arg\min_g \left\{ \sum_{i=1}^{n} (y_i - g(x_i))^2 \right\} = \hat{g}_2$$

Thus, the training and test RSS will be **the same**.

2 Programming Questions

5. Section 4.8 Exercise 14 (parts d, e, and g)

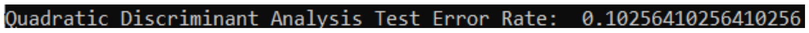
d) - This was done in `hw3_coding.py`. When Linear Discriminant Analysis was executed on the dataset (reusing the data pre-processing code as well as the 80% train, 20% test split from Homework 2), the following was produced:



```
Linear discriminant_analysis Test Error Rate:  0.07692307692307693
```

So the LDA test error rate was approximately **0.077**, or 7.7% incorrectly classified.

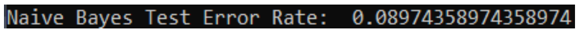
e) - This was also done in `hw3_coding.py`. When Quadratic Discriminant Analysis was executed on the dataset, the following was produced:



```
Quadratic Discriminant Analysis Test Error Rate:  0.10256410256410256
```

So the QDA test error rate was approximately **0.1**, or 10% incorrectly classified.

g) - This was done in `hw3_coding.py` as well. When Naive Bayes was executed on the dataset, the following was produced:



```
Naive Bayes Test Error Rate:  0.08974358974358974
```

So the Gaussian Naive Bayes test error rate was roughly **0.09**, or 9.0% incorrectly classified.

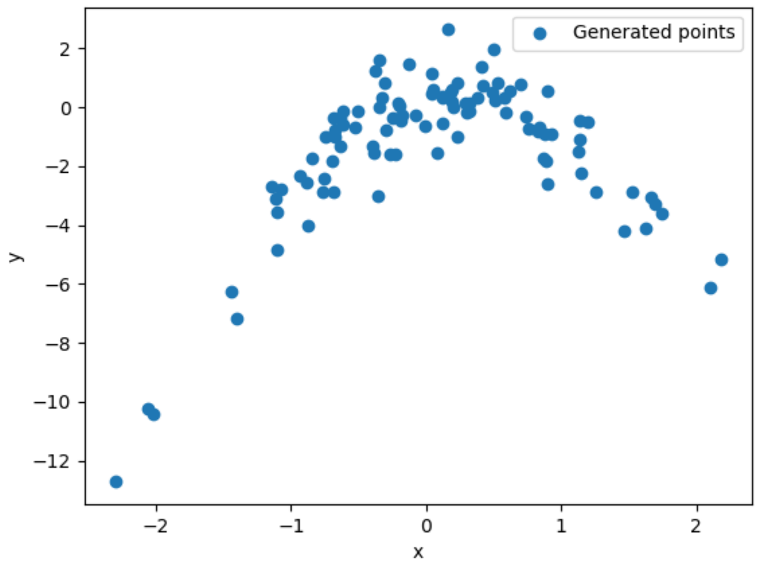
6. Section 5.4 Exercise 8

Solution

a) - This was done in `hw3_coding.py`. Note, in this generation of data, **n = 100**, or the number of datapoints, and **p = 2** (as there are two classes of variables, x and x^2, but no constant term). The equation we are generating data from is in the following form:

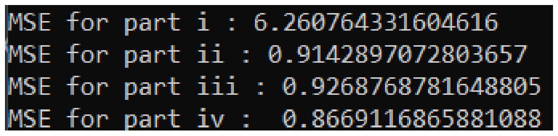
$$y_i = x_i - 2x_i^2 + \epsilon_i$$

b) - The following scatterplot was produced:



The scatter looks roughly quadratic, with many points concentrated around the point $(0, 0)$.

c) - Leave-One-Out Cross validation was implemented in `hw3_coding.py`. The following average MSE's were generated:

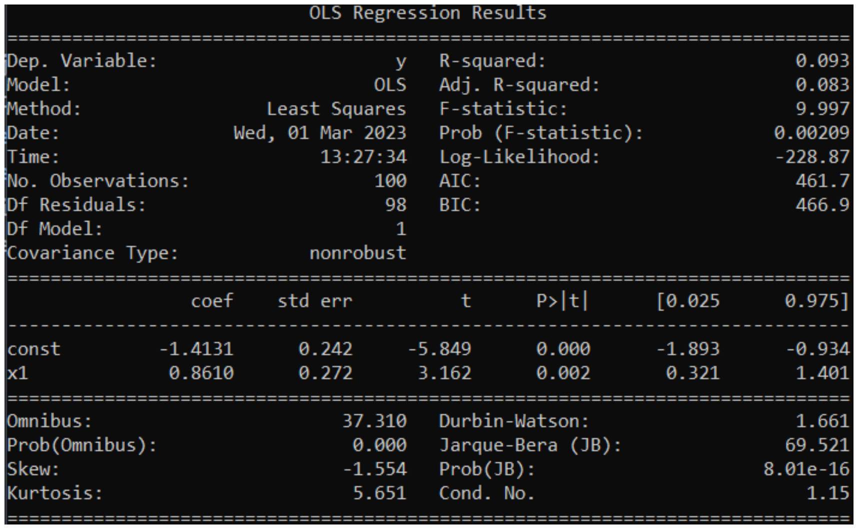


```
MSE for part i : 6.260764331604616
MSE for part ii : 0.9142897072803657
MSE for part iii : 0.9268768781648805
MSE for part iv :  0.8669116865881088
```

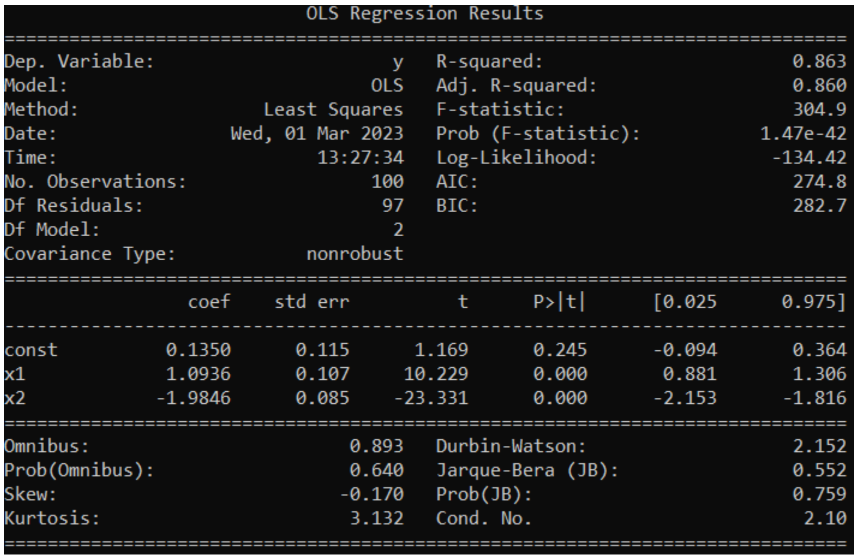
d) - When a different random seed was tried, the exact same results were arrived at. This is because Leave-one-out cross validation essentially fits n models, one for each datapoint a model must be fit with that datapoint as the test datapoint. All the new random seed did was change the order in which the testing was done, but the same n models were run in each regression case.

e) - Note that the simple linear regression had the highest average error rate, which makes sense as it did not include an x^2 term, whereas the original data generation did. There is a significant difference in average MSE when the squared term was included, as is evident in (ii), with the x^3 and x^4 not having a significant impact on average accuracy of the model. This makes sense, as the data generation only had an x and an x^2 term. However, the lowest average error rate was achieved with (iv), which contained an x^2 and an x^3. This seems counter-intuitive, given the y values were not generated with an x^3 or x^4 term, but this is likely explainable due to the random error given to the y's during generation and the extra terms increasing potential predictive power by virtue of existence (R^2 cannot decrease with more terms) .

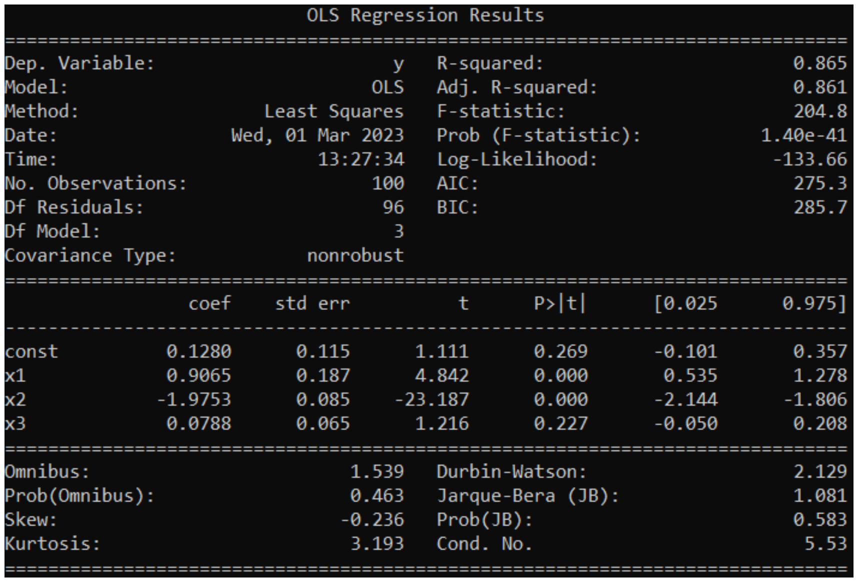
f) - the following results were generated when the 4 models were run:



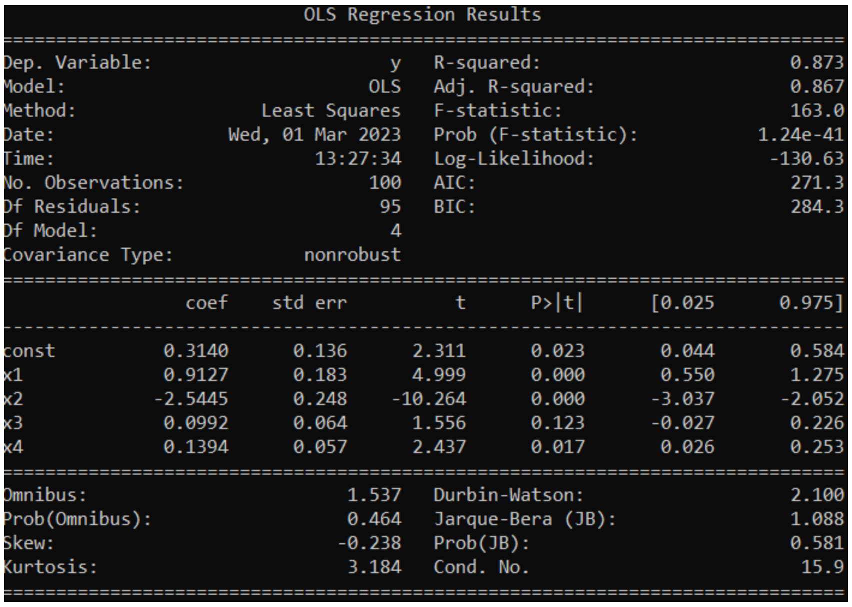
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.093
Model:                            OLS   Adj. R-squared:                  0.083
Method:                 Least Squares   F-statistic:                     9.997
Date:                Wed, 01 Mar 2023   Prob (F-statistic):            0.00209
Time:                        13:27:34   Log-Likelihood:                -228.87
No. Observations:                 100   AIC:                             461.7
Df Residuals:                      98   BIC:                             466.9
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -1.4131      0.242     -5.849      0.000      -1.893      -0.934
x1             0.8610      0.272      3.162      0.002       0.321       1.401
==============================================================================
Omnibus:                       37.310   Durbin-Watson:                   1.661
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               69.521
Skew:                          -1.554   Prob(JB):                     8.01e-16
Kurtosis:                       5.651   Cond. No.                         1.15
==============================================================================
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.863
Model:                            OLS   Adj. R-squared:                  0.860
Method:                 Least Squares   F-statistic:                     304.9
Date:                Wed, 01 Mar 2023   Prob (F-statistic):           1.47e-42
Time:                        13:27:34   Log-Likelihood:                -134.42
No. Observations:                 100   AIC:                             274.8
Df Residuals:                      97   BIC:                             282.7
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.1350      0.115      1.169      0.245      -0.094       0.364
x1             1.0936      0.107     10.229      0.000       0.881       1.306
x2            -1.9846      0.085    -23.331      0.000      -2.153      -1.816
==============================================================================
Omnibus:                        0.893   Durbin-Watson:                   2.152
Prob(Omnibus):                  0.640   Jarque-Bera (JB):                0.552
Skew:                          -0.170   Prob(JB):                        0.759
Kurtosis:                       3.132   Cond. No.                         2.10
==============================================================================
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.865
Model:                            OLS   Adj. R-squared:                  0.861
Method:                 Least Squares   F-statistic:                     204.8
Date:                Wed, 01 Mar 2023   Prob (F-statistic):           1.40e-41
Time:                        13:27:34   Log-Likelihood:                -133.66
No. Observations:                 100   AIC:                             275.3
Df Residuals:                      96   BIC:                             285.7
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.1280      0.115      1.111      0.269      -0.101       0.357
x1             0.9065      0.187      4.842      0.000       0.535       1.278
x2            -1.9753      0.085    -23.187      0.000      -2.144      -1.806
x3             0.0788      0.065      1.216      0.227      -0.050       0.208
==============================================================================
Omnibus:                        1.539   Durbin-Watson:                   2.129
Prob(Omnibus):                  0.463   Jarque-Bera (JB):                1.081
Skew:                          -0.236   Prob(JB):                        0.583
Kurtosis:                       3.193   Cond. No.                         5.53
==============================================================================
```



```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.873
Model:                            OLS   Adj. R-squared:                  0.867
Method:                 Least Squares   F-statistic:                     163.0
Date:                Wed, 01 Mar 2023   Prob (F-statistic):           1.24e-41
Time:                        13:27:34   Log-Likelihood:                -130.63
No. Observations:                 100   AIC:                             271.3
Df Residuals:                      95   BIC:                             284.3
Df Model:                           4
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.3140      0.136      2.311      0.023       0.044       0.584
x1             0.9127      0.183      4.999      0.000       0.550       1.275
x2            -2.5445      0.248    -10.264      0.000      -3.037      -2.052
x3             0.0992      0.064      1.556      0.123      -0.027       0.226
x4             0.1394      0.057      2.437      0.017       0.026       0.253
==============================================================================
Omnibus:                        1.537   Durbin-Watson:                   2.100
Prob(Omnibus):                  0.464   Jarque-Bera (JB):                1.088
Skew:                          -0.238   Prob(JB):                        0.581
Kurtosis:                       3.184   Cond. No.                         15.9
==============================================================================
```

To summarize, the linear regression produced the following models:

$\hat{y}_i = -1.4131 + .8610x \ , \ F_{\text{stat}} = 9.997 \ , \ R^2 = 0.093$

$\hat{y}_{ii} = .135 + 1.0936x - 1.9846x^2 \ , \ F_{\text{stat}} = 304.9 \ , \ R^2 = 0.863$

$\hat{y}_{iii} = .1280 + 0.9065x - 1.9753x^2 + .0788x^3 \ , \ F_{\text{stat}} = 204.8 \ , \ R^2 = 0.865$

$\hat{y}_{iv} = .314 + .9127x - 2.5445x^2 + .0992x^3 + .1394x^4 \ , \ F_{\text{stat}} = 163 \ , \ R^2 = 0.873$

These results **agree with the cross-validation results**; the biggest jump in R^2 is from the first model to the second model (via the inclusion of x^2), and there is very little predictive power in the x^3 and x^4 terms, as evidenced by the only slight increase in R^2. Note that the coefficients for the second model on x and x^2, being 1.0936 and -1.9846 respectively, are the closest to the true coefficients from generation, which works in tandem with the joint significance of the second model being the highest to show that it is the model most representative of the sample.

7. Section 5.4 Exercise 9

Solution

a) - To provide an estimate of "medv", we will use the sample's mean:

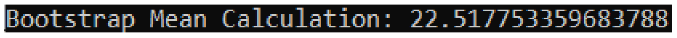
$$\overline{medv} = \frac{1}{n}\sum_{i=1}^{n} medv_i = 23.5328 = \$23,532.80$$

b) - The sample standard error was calculated as the following:

$$SE(\overline{medv}) = \sqrt{\frac{\sum_{i=1}^{n}(medv_i - \overline{medv})^2}{n-1}} = 0.40886$$

This means that this sample mean calculation is likely to deviate from the true mean by about 0.40886 = \$408.86, which is fairly small given that the values are in the thousands.

c) - The bootstrap was created using 1000 samples, and with samples (with replacement) the same size as the original dataset (that being 506). The average bootstrap sample mean (let us call it $\overline{medv_B}$) was calculated as:



```
Bootstrap Mean Calculation: 22.517753359683788
```

This is close to the actual sample mean. Following formula 5.8 in the textbook, the standard error was estimated at:

$$\widehat{SE}(\overline{medv}) = \sqrt{\frac{1}{1000-1}\sum_{i=1}^{1000}\left(\widehat{medv_i} - \overline{medv_B}\right)^2} = 0.4039$$

This answer matches fairly closely with the answer in part b).

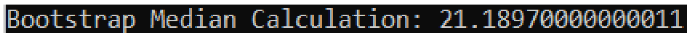
d) - Utilizing the hint, we can construct the following 95% confidence interval for $\hat{\mu}_{medv}$:

$$CI_{\hat{u}_{medv}} = (\overline{medv_B} - 2\widehat{SE}(\overline{medv}), \overline{medv_B} + 2\widehat{SE}(\overline{medv})) = (21.7100, 23.3255)$$

e) - We will estimate the population median (η_{medv}) with the sample median:

$$\overline{\eta}_{medv} = 21.2$$

f) - The bootstrap was created using 1000 samples, and with samples (with replacement) the same size as the original dataset (that being 506). The average boostrap Median (let us call it $\overline{median_B}$), was calculated as follows:



```
Bootstrap Median Calculation: 21.18970000000011
```

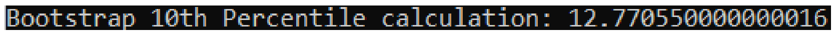
Using this, we follow formula 5.8 in the textbook to get, with $\widehat{median_i}$ representing the median of the sample:

$$\widehat{SE}(\overline{\eta}_{medv}) = \sqrt{\frac{1}{1000-1}\sum_{i=1}^{1000}\left(\widehat{median_i} - \overline{median_B}\right)^2} = 0.3789$$

g) - We will estimate the population 10th percentile $\mu_{0.1}$ with the sample 10th percentile:

$$\hat{\mu}_{0.1} = 12.75$$

h) - Using the same methodology as in parts c) and f), we first calculate the average bootstrap 10th percentile (call it $\bar{\mu}_B$):



```
Bootstrap 10th Percentile calculation: 12.770550000000016
```

Then, we use this to calculate the standard error using the same boostrapping formula:

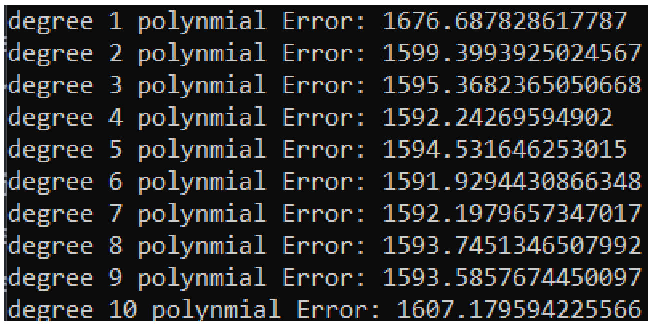
$$\widehat{SE}(\bar{\mu}_{0.1}) = \sqrt{\frac{1}{1000-1}\sum_{i=1}^{1000}(\hat{\mu}_i - \bar{\mu}_B)^2} = 0.50262$$

8. Section 7.9 Exercise 6 (Skip part about ANOVA)

a) - To check to see which degree polynomial was best, 5-fold cross validation was run on polynomials starting from degree 1 going all the way up to degree 10. Mean squared error was calculated using equation 5.3 in the textbook:

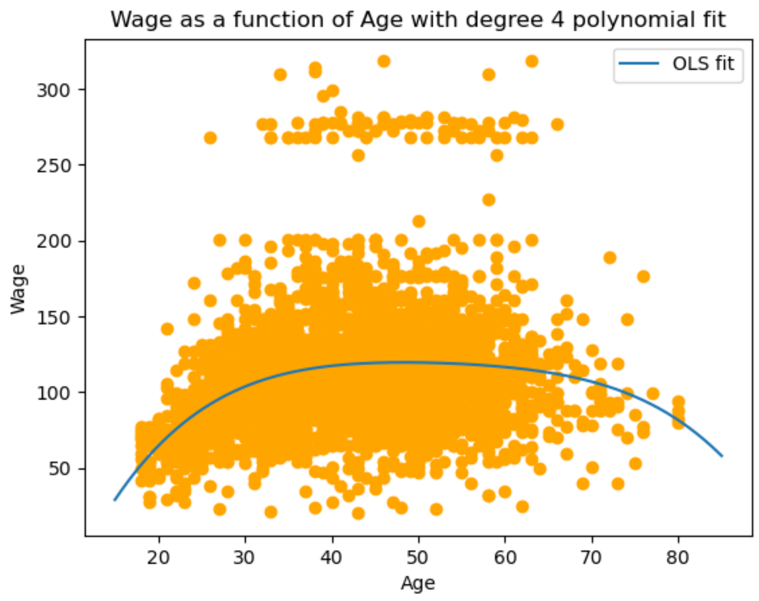
$$CV_5 = \frac{1}{5}\sum_{i=1}^{5} MSE_i$$

This resulting average mean squared errors are summarized below:



```
degree 1 polynmial Error: 1676.687828617787
degree 2 polynmial Error: 1599.3993925024567
degree 3 polynmial Error: 1595.3682365050668
degree 4 polynmial Error: 1592.24269594902
degree 5 polynmial Error: 1594.531646253015
degree 6 polynmial Error: 1591.9294430866348
degree 7 polynmial Error: 1592.1979657347017
degree 8 polynmial Error: 1593.7451346507992
degree 9 polynmial Error: 1593.5857674450097
degree 10 polynmial Error: 1607.179594225566
```

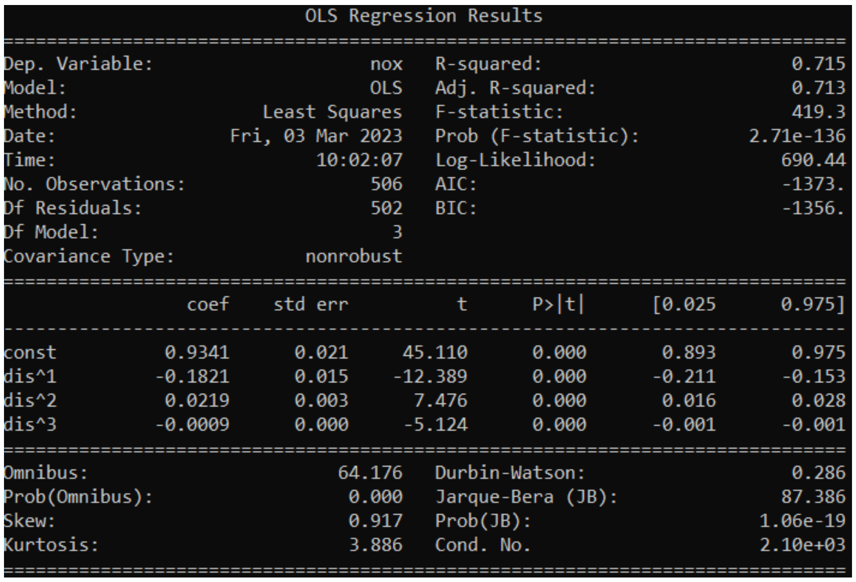
Note that the average Mean squared error decreased from the 1st degree polynomial fitting to the 2nd, and then the 3rd, and then the 4th. However, higher degree polynomials either produced errors within a range of the 4th degree error or, in the case of degree 10, error significantly higher. Therefore, we will choose the 4th degree polynomial model (quadratic). The curve looks like this when overlayed above the data:



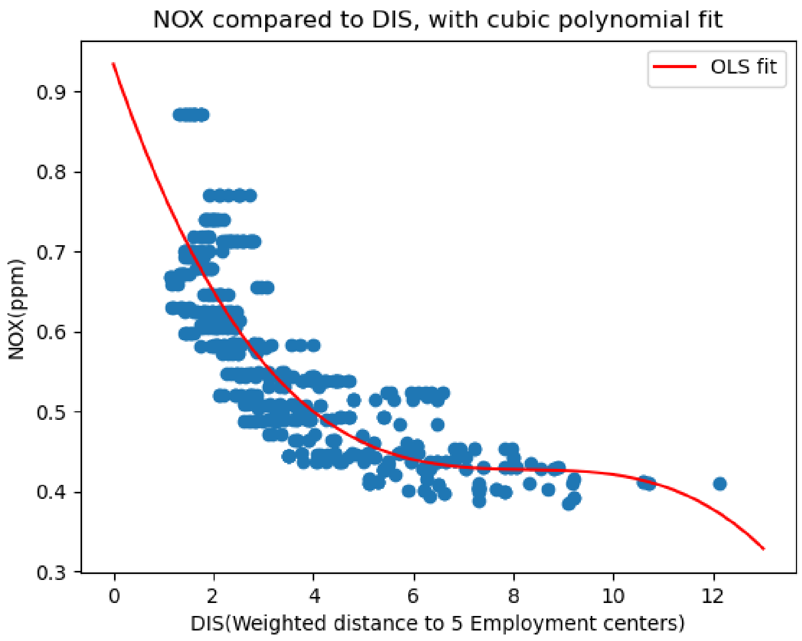
b) - This problem was attempted in `hw3_coding.py`. However, due to errors in the predict function, it was not completed. The code is under the function *Q8_b(max_cuts, num_splits, random_seeding).*

9. Section 7.9 Exercise 9

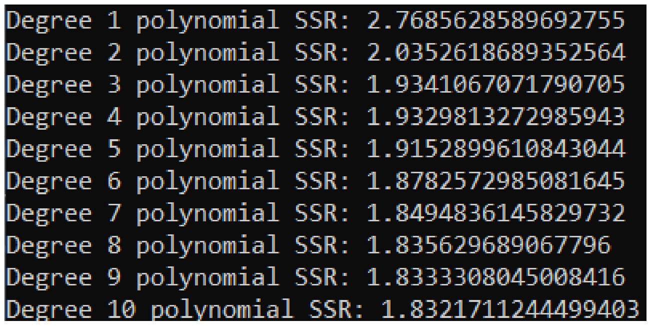
a) - Given the data for weighted distance and nitrous oxide concentration from the Boston housing dataset, after fitting a cubic polynomial to the data, the following regression output and plot were produced:



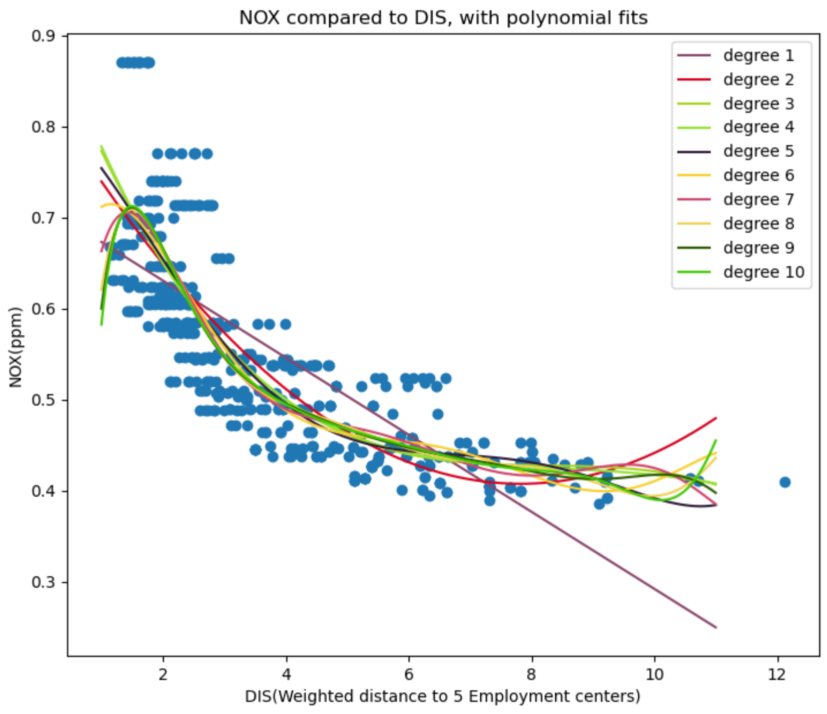
```
                            OLS Regression Results
==============================================================================
Dep. Variable:                    nox   R-squared:                       0.715
Model:                            OLS   Adj. R-squared:                  0.713
Method:                 Least Squares   F-statistic:                     419.3
Date:                Fri, 03 Mar 2023   Prob (F-statistic):          2.71e-136
Time:                        10:02:07   Log-Likelihood:                 690.44
No. Observations:                 506   AIC:                            -1373.
Df Residuals:                     502   BIC:                            -1356.
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          0.9341      0.021     45.110      0.000       0.893       0.975
dis^1         -0.1821      0.015    -12.389      0.000      -0.211      -0.153
dis^2          0.0219      0.003      7.476      0.000       0.016       0.028
dis^3         -0.0009      0.000     -5.124      0.000      -0.001      -0.001
==============================================================================
Omnibus:                       64.176   Durbin-Watson:                   0.286
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               87.386
Skew:                           0.917   Prob(JB):                     1.06e-19
Kurtosis:                       3.886   Cond. No.                     2.10e+03
==============================================================================
```



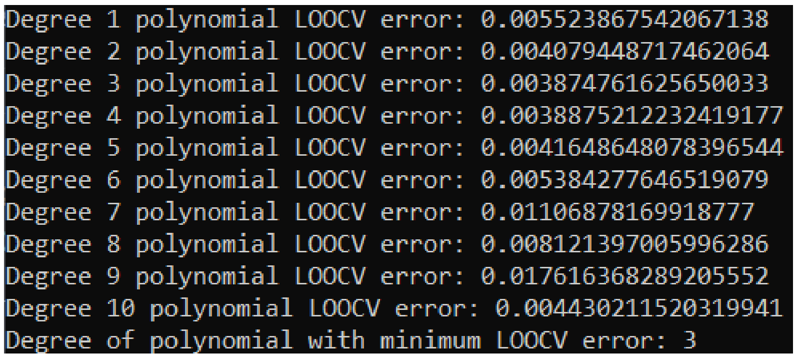
b) - Now, we fit polynomials for 10 different degrees, and plotted. The following sum of squared residuals information and plot were produced:



```
Degree 1 polynomial SSR: 2.7685628589692755
Degree 2 polynomial SSR: 2.0352618689352564
Degree 3 polynomial SSR: 1.9341067071790705
Degree 4 polynomial SSR: 1.9329813272985943
Degree 5 polynomial SSR: 1.9152899610843044
Degree 6 polynomial SSR: 1.8782572985081645
Degree 7 polynomial SSR: 1.8494836145829732
Degree 8 polynomial SSR: 1.835629689067796
Degree 9 polynomial SSR: 1.8333308045008416
Degree 10 polynomial SSR: 1.8321711244499403
```



c) - To validate the best degree polynomial, we used Leave-One-Out cross-validation and then calculated the Average mean-squared error in each case: Here was the resulting CV_n values for each degree of polynomial:

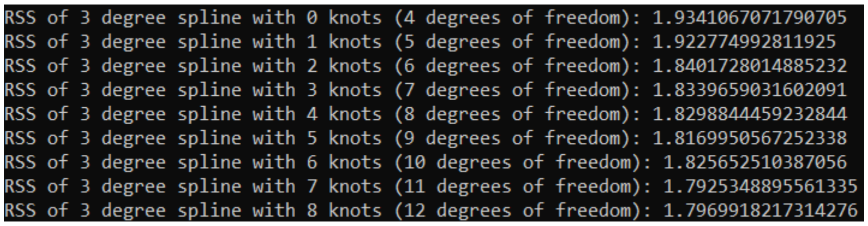


```
Degree 1 polynomial LOOCV error: 0.005523867542067138
Degree 2 polynomial LOOCV error: 0.004079448717462064
Degree 3 polynomial LOOCV error: 0.003874761625650033
Degree 4 polynomial LOOCV error: 0.0038875212232419177
Degree 5 polynomial LOOCV error: 0.0041648648078396544
Degree 6 polynomial LOOCV error: 0.005384277646519079
Degree 7 polynomial LOOCV error: 0.01106878169918777
Degree 8 polynomial LOOCV error: 0.008121397005996286
Degree 9 polynomial LOOCV error: 0.017616368289205552
Degree 10 polynomial LOOCV error: 0.004430211520319941
Degree of polynomial with minimum LOOCV error: 3
```

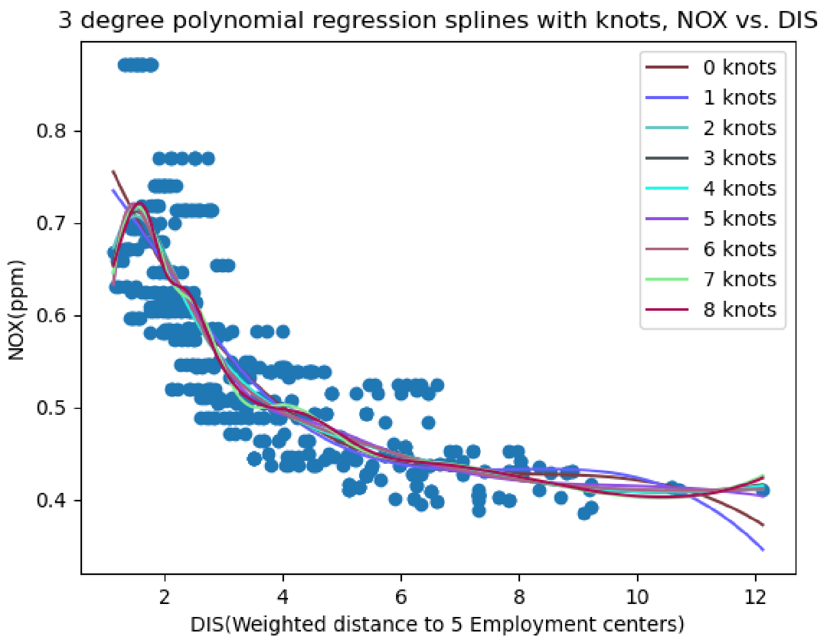
As evidenced at the bottom, the polynomial of the graphic, the degree polynomial with the minimum leave-one-out cross validation error was the degree 3 polynomial, so we choose the **degree 3 polynomial**. This intuitively follows from part a as well, as the 3rd degree polynomial seemed to fit the data relatively well.

d) - According to the textbook, a cubic spline with K knots has $K + 4$ degrees of freedom. Since we chose a degree-3 (cubic) polynomial as the optimal polynomial, a fit with 4 degrees of freedom has $K = 4 - 4 = 0$ knots, so it is just the original cubic regression fit shown in part a).

e) - We once again used a 3rd-degree polynomial to create the spline. The knots were placed as follows: given a fixed j number of knots, the i'th knot was placed at the $\left(\frac{i}{(j+1)} * 100\right)$'th quantile of data; for example, when $j = 3$, the knots are at the 25th, 50th, and 75th quantile of data to evenly split the data into sections between knots. The following sum of squared residuals calculations were the result for different numbers of knots, as well as the graph:

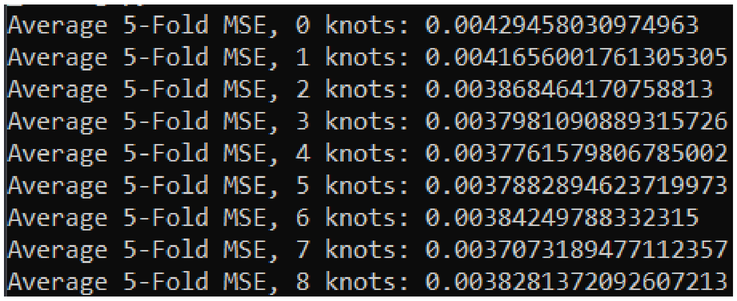


```
RSS of 3 degree spline with 0 knots (4 degrees of freedom): 1.9341067071790705
RSS of 3 degree spline with 1 knots (5 degrees of freedom): 1.922774992811925
RSS of 3 degree spline with 2 knots (6 degrees of freedom): 1.8401728014885232
RSS of 3 degree spline with 3 knots (7 degrees of freedom): 1.8339659031602091
RSS of 3 degree spline with 4 knots (8 degrees of freedom): 1.8298844459232844
RSS of 3 degree spline with 5 knots (9 degrees of freedom): 1.8169950567252338
RSS of 3 degree spline with 6 knots (10 degrees of freedom): 1.825652510387056
RSS of 3 degree spline with 7 knots (11 degrees of freedom): 1.7925348895561335
RSS of 3 degree spline with 8 knots (12 degrees of freedom): 1.7969918217314276
```



Note that, using this knot breakdown, the greatest drop in RSS occurs when adding the second knot, and the accuracy of the model practically levels off from there. The models with more knots might be a tad overfit.

f) - To check which number of knots is best, 5-fold cross validation was used, and the evaluation was based on average mean-squared error. The following was produced when all mean-squared errors were compared to one another, (calculations were based on Equation 5.3 of the textbook):



```
Average 5-Fold MSE, 0 knots: 0.00429458030974963
Average 5-Fold MSE, 1 knots: 0.0041656001761305305
Average 5-Fold MSE, 2 knots: 0.003868464170758813
Average 5-Fold MSE, 3 knots: 0.0037981090889315726
Average 5-Fold MSE, 4 knots: 0.0037761579806785002
Average 5-Fold MSE, 5 knots: 0.0037882894623719973
Average 5-Fold MSE, 6 knots: 0.00384249788332315
Average 5-Fold MSE, 7 knots: 0.0037073189477112357
Average 5-Fold MSE, 8 knots: 0.0038281372092607213
```

These results align with our answer in e), as the addition of the second knot decreases average Mean Squared Error noticeably, but it practically levels off as more knots are added. So the optimal model, it seems, is **a cubic regression spline with 2 knots**, with more knots likely causing over-fitting.