HW3 — ISYE6416

Ashish Dhiman — ashish.dhiman9@gatech.edu

May 7, 2023

Contents

1 Question 1

1.1 part a: Algorithm Efficiency

Efficiency of an algorithm refers to the amount of resources it needs to complete it's designated task. Looking at the specific types of resources helps define the two times of efficiency measures:

1. **Time Complexity:** Concerned with the time an algorithm needs to complete a task

2. **Space Complexity:** Concerned with the amount of memory resources the algorithm expends

1.2 part b: Algorithm Robustness

Algorithm Robustness is the measure of algorithms to absorb and accept variability in input data and without failing. In other words, an algorithm is robust if it can accommodate large amounts of variability in the data, including invalid or unexpected inputs. An example of a robust algorithm is Linear Robust optimisation which allows for uncertainty in coefficients of LP constraints.

1.3 part c: Algorithm Stability

Algorithm Stability is the property where minor alterations in input do not cause a lot of variability in the output of the algorithm. The key difference b/w robustness and stability is that, while stability focuses on putting inertia on output, i.e. minimal delta in output, while robustness is focused more on mantaining feasibility of output.

1.4 part d: Algorithm Accuracy

Two types of algorithm accuracy are:

1. **Absolute acc.:** deviation between true and predicted value or

$$= |x_{true} - x_{pred}|$$

2. **Relative acc.:** ratio of absolute deviation to true value or

$$= \frac{|x_{true} - x_{pred}|}{x_{true}}$$

Sometimes approximate algorithms are preferable beacause they might be more efficient, or more stable and robust. for example: Stochastic Gradient vs gradient descent

1.5 part e: Worst case complexity of Quick Sort

Worst case scenario of Quick sort occurs for cases where pivot is at the extreme end. In such cases, the recursive algorithm does not create any gains.

$$\begin{aligned}
\text{Let } T(N) \text{ be time complexity to sort N elements} \\
\text{then for worst case i.e pivot on one end} \\
T(N) = N + T(N-1) \quad \text{(find pivot from N elements + sort rest N-1)} \\
\text{Similarly} \\
T(N-1) = (N-1) + T(N-2) \\
\implies T(N) = N + N - 1 + N - 2 + \ldots + 2 + T(1) \qquad (1) \\
\text{With one element no sorting required, or} T(1) = 0 \\
\implies T(N) = N + N - 1 + \ldots + 2 \\
= \frac{(N)(N+1)}{2} - 1 \\
= O(N^2)
\end{aligned}$$

Hence Proved

1.6 part f: Bisection

The problem boils down to finding roots of $f(x) = F(x) - 0.95$ where F is cdf of t distribution with 5 dof. The implementation of bisection algorithm is given below:

```
from scipy.stats import t

df = 5
start,end = 1.291,2.582

f= lambda x: t.cdf(x,df=df) - 0.95
f_start = f(start)
f_end = f(end)

print (f"Starting range : ({start:.4f},{end:.4f})")

c=0
while (end-start)>=10**(-4):
  c+=1
  mid = 0.5*(start+end)
  f_mid = f(mid)
  if f_mid >0:
    end = mid
  if f_mid<0:
    start = mid
  print (f"search range for iter={c} is ({start:.4f},{end:.4f})")

print (f"True Root of f:{t.ppf(0.95,df=df):.4f}")
```

The output for the code is given below:

```
Starting range : (1.2910,2.5820)
search range for iter=1 is (1.9365,2.5820)
search range for iter=2 is (1.9365,2.2592)
search range for iter=3 is (1.9365,2.0979)
search range for iter=4 is (1.9365,2.0172)
search range for iter=5 is (1.9768,2.0172)
search range for iter=6 is (1.9970,2.0172)
search range for iter=7 is (2.0071,2.0172)
search range for iter=8 is (2.0121,2.0172)
search range for iter=9 is (2.0147,2.0172)
search range for iter=10 is (2.0147,2.0159)
search range for iter=11 is (2.0147,2.0153)
search range for iter=12 is (2.0150,2.0153)
search range for iter=13 is (2.0150,2.0151)
search range for iter=14 is (2.0150,2.0151)
True Root of f:2.0150
```

1.7 part g: Matrix Multiplication Complexity

Let there be two matrices:

$$A = []_{n,m}, B = [.]_{m,k}$$

Now product AB can be seen as dot product between n rows of A and k columns of B. Each of these dot products in itself requires a multiplication of m elements. There fore the time complexity of matrix product AB is $O(nmk)$.

Now If we both A and B are square matrices of dimensions (n, n), the order becomes $O(n * n * n) = O(n^3)$

2 Question 2: Rank of Product of matrices

We are given two matrices:

$$A = []_{4,3}, B = [.]_{3,5}$$

Also it is given

$$Rank(A) = 2$$

Now we have:

$$r = Rank(AB) = min\{Rank(A), Rank(B)\}$$

But

$$Rank(B) <= min(3,5) <= 3$$

$$\implies r = \begin{cases} 2 & Rank(B) \in \{2,3\} \\ 1, & Rank(B) = 1 \end{cases}$$

- Example of B with Rank =1 so r = 1, $B = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$
- Example of B with Rank =2 so r = 2, $B = \begin{bmatrix} 1 & 0 & 2 & 0 & 2 \\ 0 & 1 & 0 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$
- Example of B with Rank =2 so r = 3, $B = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \end{bmatrix}$

3 Question 3: k-means Algorithm

Given cost function,

$$J = \sum_{i=1}^{m} \sum_{j=1}^{k} c_{ij} \|x_i - \mu_j\|_2^2,$$

3.1 Centroid calculation

Given cluster assignment c_{ij} we want find cluster centroids μ_j that minimise the above cost function.

$$\begin{aligned} \text{First order Optimality condition} : \nabla_{\mu_j} J = 0 \\ \nabla_{\mu_j} (\sum_{i=1}^{m} \left[c_{i1} |x_i - \mu_1|_2^2 + c_{i2} |x_i - \mu_2|_2^2 + \ldots + c_{ik} |x_i - \mu_k|_2^2 \right]) \\ \implies \nabla_{\mu_j} J = \sum_{i=1}^{m} -2 * c_{ij} (x_i - \mu_j) = 0 \\ \implies \sum_{i=1}^{m} (c_{ij} x_i) = \sum_{i=1}^{m} (c_{ij}) \mu_j \\ \implies \mu_j = \frac{\sum_{i=1}^{m} (c_{ij} x_i)}{\sum_{i=1}^{m} (c_{ij})} \end{aligned} \tag{2}$$

The first order condition is enough in this case because J is Convex in μ

3.2 Cluster Assignment

Now given cluster centroids μ_j we want cluster assignment c_{ij} that minimises the above cost function.

The above cost function has a minimum value of 0 and it is attained when the $c_{ij=0}$. However that would mean no cluster assignment, and we want one point to be assigned to one cluster. In other words:

$$\sum j = 1^k c_{ij} = 1$$

We also know that for a any given set of points $x_1, x_2, \dots x_n$

$$\sum_i |x_i - \bar{x}|_2^2 < \sum_i |x_i - x_j|_2^2$$

where $\bar{x}$ is centroid of $x_1, x_2, \dots x_n$.

Thus putting the two properties together, we land at the optimal value of $c_i j$ as :

$$c_{ij} = \begin{cases} 1 & if \quad j = \arg\min_{j'} |x_i - \mu_{j'}|_2^2 \\ 0, & otherwise \end{cases}$$

4 Question 4: Spline and Smoothing Spline

4.1 part a: Expression of spline function

We want $s(x)$, such that it passes through (x_i, y_i), $i = 0, 1, 2$ exactly.
Let s_{01} spline function between x_0 and x_1, and similarly for s_{12}.
Also let $h = x_1 - x_0 = x_2 - x_1$

With also impose:

- s to be continuous, $s_{01}(x_1) = s_{12}(x_1)$
- s to be continuously differentiable, $s_{01}^{'}(x_1) = s_{12}^{'}(x_1)$
- s second derivative differentiable, $s_{01}^{''}(x_1) = s_{12}^{''}(x_1)$
- Fixed curvature at endpoints $s_{01}^{''}(x_0) = s_{12}^{''}(x_2) = 0$

$$
\begin{aligned}
&\text{Now given we are fitting cubic spline, we can write} \\
s_{01}(x) &= a_0 + b_0(x - x_0) + c_0(x - x_0)^2 + d_0(x - x_0)^3 \\
s_{12}(x) &= a_1 + b_1(x - x_1) + c_1(x - x_1)^2 + d_1(x - x_1)^3 \\
&\text{First derivative is} \\
s_{01}^{'}(x) &= b_0 + 2c_0(x - x_0) + 3d_0(x - x_0)^2 \\
s_{12}^{'}(x) &= b_1 + 2c_1(x - x_1) + 3d_1(x - x_1)^2 \\
&\text{Second derivative is} \\
s_{01}^{''}(x) &= 2c_0 + 6d_0(x - x_0) \\
s_{12}^{''}(x) &= 2c_1 + 6d_1(x - x_1) \\
\text{Using } s_{01}^{''}(x_0) &= 0 \text{ and } s_{12}^{''}(x_2) = 0 \\
c_0 &= 0 \\
2c_1 &= -6d_1 h \implies c_1 = -3d_1 h \\
\text{Similarly we can use } s_{01}(x_1) &= s_{12}(x_1) \\
a_0 + b_0(x - 1 - x_0) + c_0(x_1 - x_0)^2 + d_0(x_1 - x_0)^3 &= a_1 \\
\implies a_0 + b_0 h + c_0 h^2 + d_0 h^3 &= a_1 \\
\text{Also we can use } s_{01}^{'}(x_1) &= s_{12}^{'}(x_1) \\
b_0 + 2c_0(x_1 - x_0) + 3d_0(x_1 - x_0)^2 &= b_1 \\
\implies b_0 + 2c_0 h + 3d_0 h^2 &= b_1 \\
\text{Also we can use } s_{01}^{''}(x_1) &= s_{12}^{''}(x_1) \\
2c_0 + 6d_0 h &= 2c_1 \\
c_0 + 3d_0 h &= c_1 \\
\implies c_1 &= 3d_0 h \\
&\text{Also we can use there is no noise, so} \\
\implies a_0 &= y_0 \\
\implies a_1 &= y_1 \\
\implies y_2 &= a_1 + b_1 h + c_1 h^2 + d_1 h^3
\end{aligned} \tag{3}
$$

Solving the above we get:

$$\begin{aligned}
a_0 &= y_0 \\
b_0 &= \frac{6y_1 - 5y_0 - y_2}{4h} \\
c_0 &= 0 \\
d_0 &= \frac{y_0 - 2y_1 + y_2}{4h^3} \\
a_1 &= y_1 \\
b_1 &= \frac{y_2 - y_0}{2h} \\
c_1 &= \frac{3(y_0 - 2y_1 + y_2)}{4h^2} \\
d_1 &= \frac{2y_1 - y_0 - y_2}{4h^3}
\end{aligned} \tag{4}$$

With the above coefficient values we have the spline functions parameterised in terms of input points.

4.2 part b: Spline with noise

We want

$$\min_s \quad \alpha \sum_{i=0}^{2} (y_i - f_i)^2 + (1-\alpha) \int_{x_0}^{x_2} [s''(x)]^2 dx$$

We know the solution of above equation is given by:

$$\sigma = M^{-1}Q[\alpha I + (1-\alpha)IQ^T M^{-1}Q]^{-1}\alpha Iy$$

For alpha = 1/2, we have:

$$\sigma = M^{-1}Q[\frac{1}{2}I + \frac{1}{2}IQ^T M^{-1}Q]^{-1}\frac{1}{2}Iy$$

$$\sigma = [\frac{1}{2}M^{-1}Q + \frac{1}{2}M^{-1}QQ^T M^{-1}Q]\frac{1}{2}Iy$$

5 Question 5: Histogram and KDE

5.1 part a: 1D Histogram and KDE

Histogram for the two variables is:

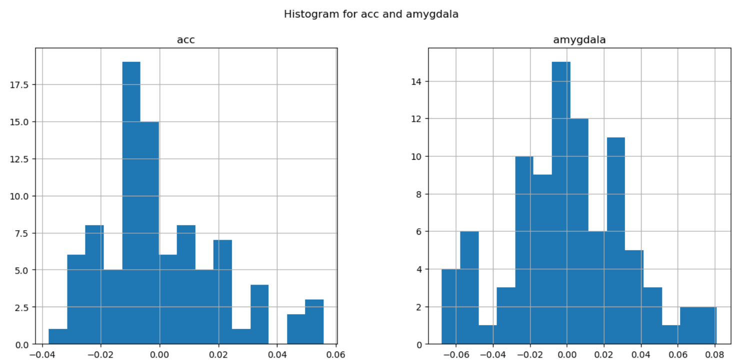

Figure 1: Histogram of acc(left) and amygdala (right)

Because there are 90 points in data, I have chosen number of bins as 15, with 6 points in each bin.

We can see that the two distributions are not normal, and have great thinning out between the center and the tails. Distribution of amygdala also looks more symmetric.

KDE for the two variables is:

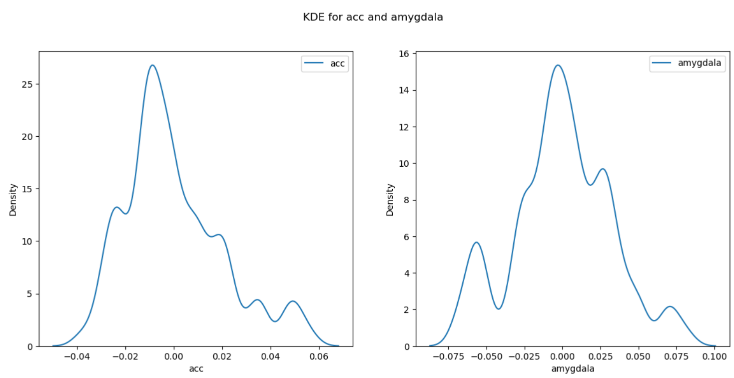

Figure 2: KDE with gaussian kernel of acc(left) and amygdala (right)

With proper tuning of bandwidth the distribution from KDE echoes the insights from Histogram, i.e thinning out before tails and relatively more symmetric density for amygdala.

5.2 part b: 2D Density estimate from Histogram

Since we have 90 points, I have picked number of bins for each variable as 6, giving a total of 36 bins. In my tuning I find this to be the optimum, because on average we get 2 to 3 points in each bin. Also a higher value becomes too granular, while lower value is too blurred.

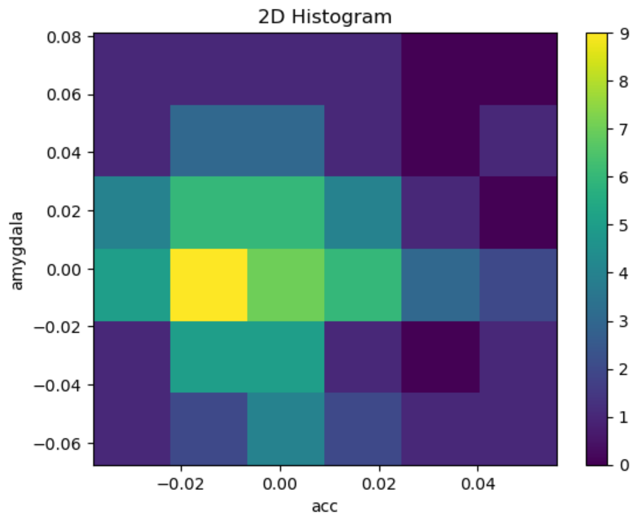

Figure 3: 2D Histogram

As we see the joint distribution is unimodal, with the location of mode being $acc \in (-0.02, 0.00)$ and $amygdala \in (-0.02, 0.00)$

5.3 part c: 2D Density estimate from KDE

The density plot below is given in the form of contour plots:

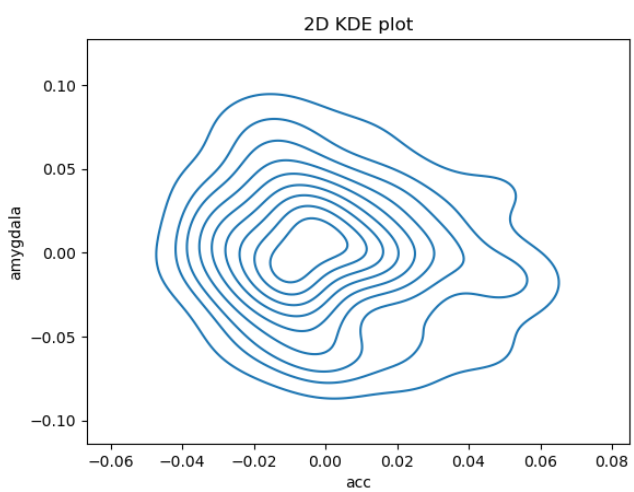

Figure 4: Joint density plot with KDE

Independence

Based on the joint density plot the two variables do seem independent. We can see that if I suppose I draw a horizontal line to get $\hat{P}(acc|amygala = c)$ the distribution is roughly the same as the marginal KDE density from part a above. Similarly, if instead I imagine a vertical line, similar intuition holds.

The above assertion however is more strong near the mode, while need futher qualifications around the tails, especially higher end of acc.

In contrast to above independent density plot, compare the plot of non-independent plot below:

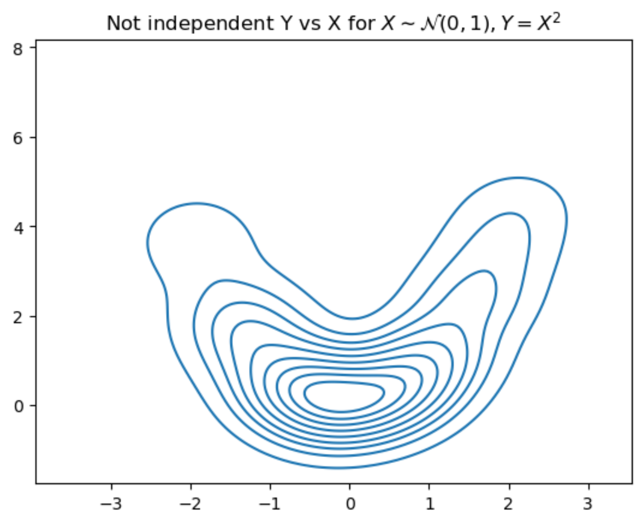

Figure 5: Joint density plot with KDE

The independence argument is further strengthened when looking at scatter plot, since there is little to none apparent corelation.

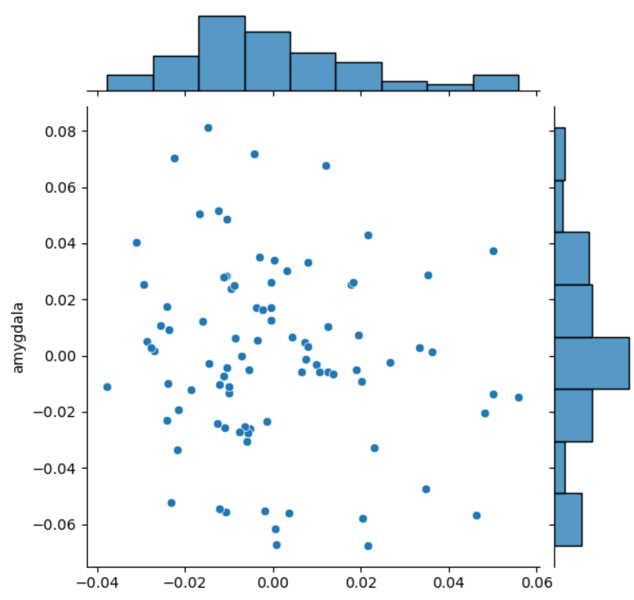

Figure 6: Scatter Plot of acc and amygdala

5.4 part d: Condition acc and amygdala on orientation

KDE Estimate of $\hat{p}(acc|orientation)$

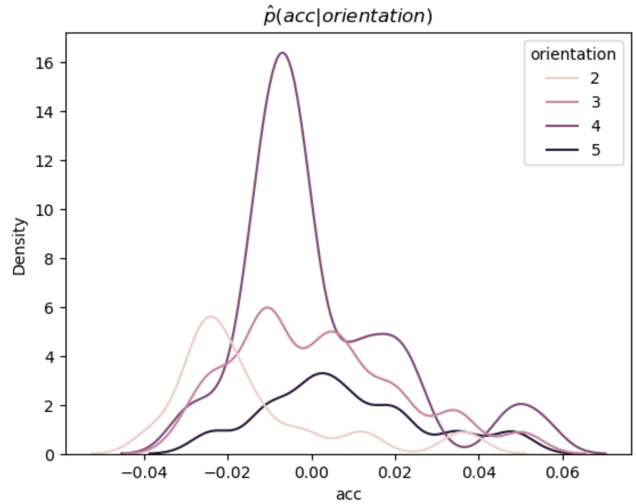

Figure 7: Conditional KDE for acc

From the looks of it seems for conservative political views or orientation =2, acc has a lower mode compared to other orientation values.

KDE Estimate of $\hat{p}(amygdala|orientation)$

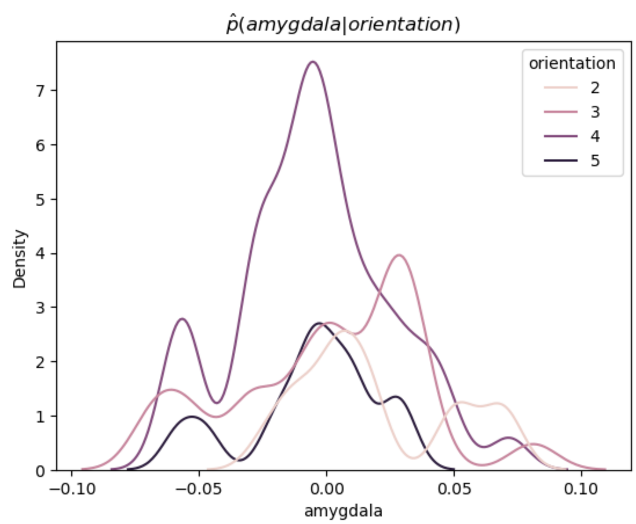

Figure 8: Conditional KDE for amygdala

From the looks of it seems for conservative political views or orientation =2, amygdala has a slightly higher mode compared to other orientation values.

Sample means conditioned on orientation

The conditional sample means of acc and amygdala on basis of Political Orientation is:

orientation	*Sample Mean*	
	amygdala	**acc**
2	0.019062	-0.014769
3	0.000588	0.001671
4	-0.004720	0.001310
5	-0.005692	0.008142

Table 1: Sample means of amygdala and acc conditioned on political orientation

Similar to KDE plot, we can see that the for conservative political orientation amygdala has higher values while acc has lower value.

6 Question 6: Nonlinear Regression using spline

6.1 part a: Linear Regression

I have fit an OLS regression after adding intercept term:

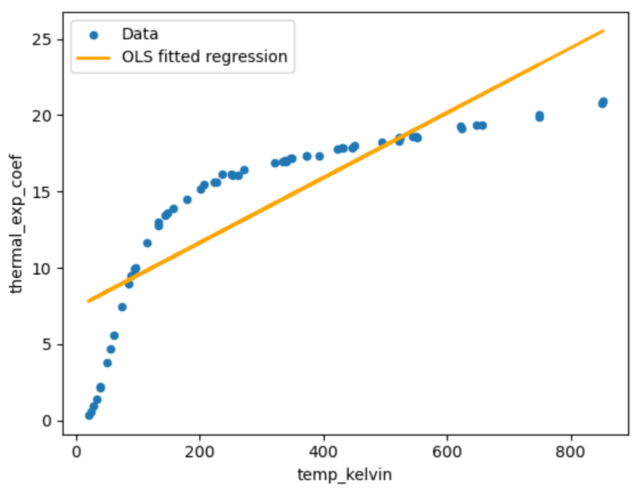

Figure 9: OLS Linear Regression

The model fit is:

$$\hat{y} = 7.3841 + 0.0213x$$

The fitting error is:

$$MSE = \frac{1}{n-2}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = 10.89 \quad (n = 59)$$

The summary of fitted model is:

```

                            OLS Regression Results
==============================================================================
Dep. Variable:        thermal_exp_coef   R-squared:
          0.686
Model:                             OLS   Adj. R-squared:
          0.681
Method:                  Least Squares   F-statistic:
          124.6
Date:                 Fri, 24 Feb 2023   Prob (F-statistic):
       5.70e-16
Time:                         11:54:28   Log-Likelihood:
        -153.14
No. Observations:                   59   AIC:
          310.3
Df Residuals:                       57   BIC:
          314.4
Df Model:                            1
Covariance Type:             nonrobust
===============================================================================
                  coef    std err          t      P>|t|      [0.025
          0.975]
-------------------------------------------------------------------------------
const           7.3841      0.743      9.943      0.000       5.897
           8.871
temp_kelvin     0.0213      0.002     11.164      0.000       0.017
           0.025
==============================================================================
Omnibus:                         7.338   Durbin-Watson:
          0.599
Prob(Omnibus):                   0.026   Jarque-Bera (JB):
          7.674
Skew:                           -0.863   Prob(JB):
         0.0216
Kurtosis:                        2.624   Cond. No.
           673.
==============================================================================
```

6.2 part b: Spline fitting

The leave one out CV curve for smoothing hyperparameter λ is given below:

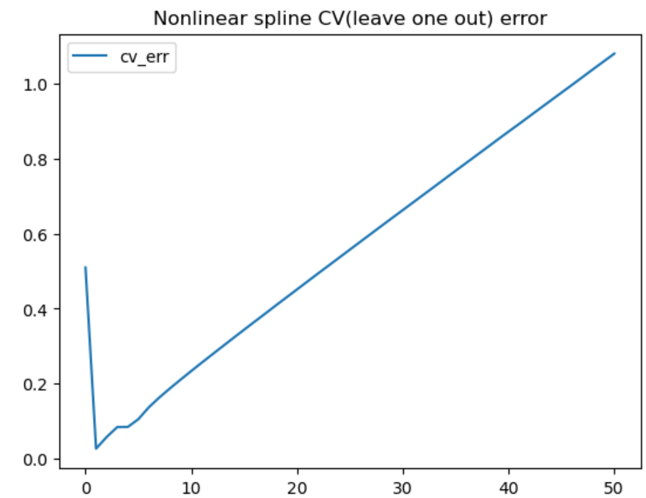

Figure 10: CV curve

From above $\lambda_{best} = 1$
The spline fit for this choice is:

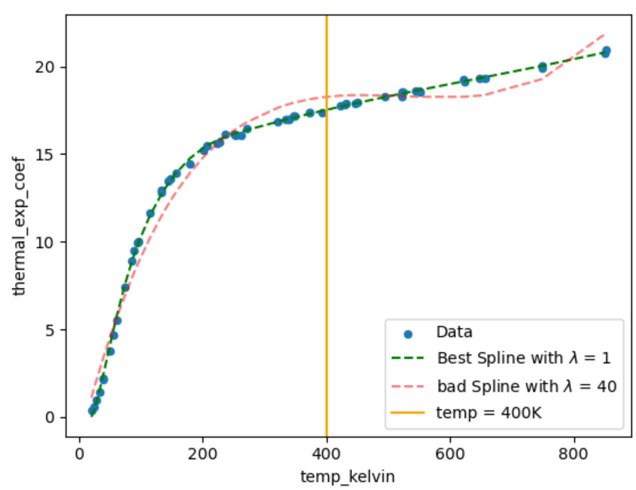

Figure 11: Spline fitting

For temp = 400, OLS fit = 15.90 For temp = 400, Spline fit = 17.52
From the graph hints that non linear spline might be a better fit.

7 References:

1. https://www.baeldung.com/cs/quicksort-time-complexity-worst-case
2. https://en.wikipedia.org/wiki/Robustness_(computer_science)
3. https://www.statlect.com/matrix-algebra/matrix-product-and-rank
4. https://towardsdatascience.com/k-means-clustering-algorithm-applications-evaluation-me

5. https://docs.scipy.org/doc/scipy/tutorial/interpolate/smoothing_splines.html#spline-smoothing-in-1-d