בס"ד

Intro to machine learning
Major home work #3
presenting:
Yitzchak Grinboim – 318837317
Sasson Shmuel Lamey - 325172351

Section 1

Q1:

$$\frac{\partial\, \ell_{\,\delta}(\boldsymbol{w}, b; x_i, y_i)}{\partial b} = \begin{cases} \frac{\partial}{\partial \mathrm{b}}\left(\frac{1}{2}(w^T \cdot x_i + b - y_i)^2\right) \\ \frac{\partial}{\partial \mathrm{b}}\left(\delta\left(|w^T \cdot x_i + b - y_i| - \frac{1}{2}\delta\right)\right) \end{cases} \overset{=}{_{chain\ rule}} \begin{cases} (w^T \cdot x_i + b - y_i) \cdot \frac{\partial}{\partial \mathrm{b}}\left((w^T \cdot x_i + b - y_i)\right) \\ \delta \cdot \frac{\partial}{\partial \mathrm{b}}\left(\sqrt{(w^T \cdot x_i + b - y_i)^2}\right) \end{cases} =$$

$$\begin{cases} (w^T \cdot x_i + b - y_i) \\ \delta \cdot \left(\frac{1}{2\sqrt{(w^T \cdot x_i + b - y_i)^2}}\right) \cdot \frac{\partial}{\partial \mathrm{b}}((w^T \cdot x_i + b - y_i)^2) \end{cases} = \begin{cases} (w^T \cdot x_i + b - y_i) \\ \delta \cdot \left(\frac{1}{2\sqrt{(w^T \cdot x_i + b - y_i)^2}}\right) \cdot 2(w^T \cdot x_i + b - y_i) \cdot \frac{\partial}{\partial \mathrm{b}}(w^T \cdot x_i + b - y_i) \end{cases} =$$

$$\begin{cases} (w^T \cdot x_i + b - y_i) \\ \delta \cdot \left(\frac{2(w^T \cdot x_i + b - y_i)}{2\sqrt{(w^T \cdot x_i + b - y_i)^2}}\right) \end{cases} = \begin{cases} (w^T \cdot x_i + b - y_i) \\ \delta \cdot \left(\frac{(w^T \cdot x_i + b - y_i)}{|w^T \cdot x_i + b - y_i|}\right) \end{cases} = \begin{cases} w^T \cdot x_i + b - y_i \\ \delta \cdot sign(w^T \cdot x_i + b - y_i) \end{cases}$$

Where the upper row was for the case when $|w^T \cdot x_i - y_i + b| \le \delta$ and the lower otherwise.

Q2:

Some auxiliary calculations:

$$= \begin{bmatrix} x_{1_1} & \cdots & x_{m_1} \\ \vdots & \ddots & \vdots \\ x_{1_d} & \cdots & x_{m_d} \end{bmatrix} \cdot \begin{bmatrix} \left(\left(w^T \cdot x_1 + b - y_1\right) \cdot 1_{|w^T \cdot x_1 + b - y_1| \le \delta}\right) + \left(\left(w^T \cdot x_1 + b - y_1\right) \cdot 1_{|w^T \cdot x_1 + b - y_1| > \delta}\right) \\ \vdots \\ \left(\left(w^T \cdot x_m + b - y_m\right) \cdot 1_{|w^T \cdot x_m + b - y_m| \le \delta}\right) + \left(\left(w^T \cdot x_m + b - y_m\right) \cdot 1_{|w^T \cdot x_m + b - y_m| > \delta}\right) \end{bmatrix}$$

$$= \frac{1}{m}\sum_{i=1}^{m}(w^T \cdot x_i + b - y_i) \cdot x_i \underset{l^{\text{"}}c\ of\ x_i's:}{=} \begin{bmatrix} x_{1_1} & \cdots & x_{m_1} \\ \vdots & \ddots & \vdots \\ x_{1_d} & \cdots & x_{m_d} \end{bmatrix} \cdot \begin{bmatrix} w^T \cdot x_1 + b - y_1 \\ \vdots \\ w^T \cdot x_m + b - y_m \end{bmatrix} = \frac{1}{m}\begin{bmatrix} x_{1_1} & \cdots & x_{m_1} \\ \vdots & \ddots & \vdots \\ x_{1_d} & \cdots & x_{m_d} \end{bmatrix}\left(\begin{bmatrix} w^T \cdot x_1 \\ \vdots \\ w^T \cdot x_m \end{bmatrix} + \begin{bmatrix} b \\ \vdots \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}\right)$$

$$= \frac{1}{m}\begin{bmatrix} x_{1_1} & \cdots & x_{m_1} \\ \vdots & \ddots & \vdots \\ x_{1_d} & \cdots & x_{m_d} \end{bmatrix}\left(\begin{bmatrix} x_{1_1} & \cdots & x_{1_d} \\ \vdots & \ddots & \vdots \\ x_{m_1} & \cdots & x_{m_d} \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix} + \begin{bmatrix} b \\ \vdots \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}\right) = \frac{1}{m} \cdot X^T(X \cdot w + b \cdot 1_m - y)$$

$$\frac{1}{m}\sum_{i=1}^{m}\delta \cdot sign(w^T \cdot x_i + b - y_i) \cdot x_i \underset{a\ linear\ combination\ of\ x_i's:}{=} \frac{\delta}{m}\begin{bmatrix} x_{1_1} & \cdots & x_{m_1} \\ \vdots & \ddots & \vdots \\ x_{1_d} & \cdots & x_{m_d} \end{bmatrix} \cdot \begin{bmatrix} sign(w^T \cdot x_1 + b - y_1) \\ \vdots \\ sign(w^T \cdot x_m + b - y_m) \end{bmatrix}$$

$$= \frac{\delta}{m}\begin{bmatrix} x_{1_1} & \cdots & x_{m_1} \\ \vdots & \ddots & \vdots \\ x_{1_d} & \cdots & x_{m_d} \end{bmatrix} \cdot \begin{bmatrix} 1_{w^T \cdot x_1 + b - y_1 \ge 0} \\ \vdots \\ 1_{w^T \cdot x_1 + b - y_1 \ge 0} \end{bmatrix} = \frac{\delta}{m}\begin{bmatrix} x_{1_1} & \cdots & x_{m_1} \\ \vdots & \ddots & \vdots \\ x_{1_d} & \cdots & x_{m_d} \end{bmatrix} \cdot sign\left(\begin{bmatrix} w^T \cdot x_1 + b - y_1 \\ \vdots \\ w^T \cdot x_m + b - y_m \end{bmatrix}\right)$$

$$= \frac{\delta}{m}\begin{bmatrix} x_{1_1} & \cdots & x_{m_1} \\ \vdots & \ddots & \vdots \\ x_{1_d} & \cdots & x_{m_d} \end{bmatrix} \cdot sign\left(\begin{bmatrix} x_{1_1} & \cdots & x_{1_d} \\ \vdots & \ddots & \vdots \\ x_{m_1} & \cdots & x_{m_d} \end{bmatrix} \cdot \begin{bmatrix} w_1 \\ \vdots \\ w_d \end{bmatrix} + \begin{bmatrix} b \\ \vdots \\ b \end{bmatrix} - \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}\right) = \frac{\delta}{m} \cdot X^T \cdot sign(X \cdot w + b \cdot 1_m - y)$$

The 3 upper rows are when naively assuming that for any i: $|w^T \cdot x_i - y_i + b| \le \delta$, and the lower 2 are when assuming naively that for any i $|w^T \cdot x_i - y_i + b| > \delta$. But for some I it holds and for some it does not.
really it is like this:

$$\nabla_w \mathcal{L}_{\mathcal{H}}(\boldsymbol{w}, b) = \frac{1}{m}\sum_{i=1}^{m}\left(\left(\left(w^T \cdot x_i + b - y_i\right) \cdot 1_{|w^T \cdot x_i + b - y_i| \le \delta}\right) + \left(\left(w^T \cdot x_i + b - y_i\right) \cdot 1_{|w^T \cdot x_i + b - y_i| > \delta}\right)\right) \cdot x_i$$

this is a linear combination of $x_i's$ and the coefficients depend on $|w^T \cdot x_i + b - y_i|$.

$$= \begin{bmatrix} x_{1_1} & \cdots & x_{m_1} \\ \vdots & \ddots & \vdots \\ x_{1_d} & \cdots & x_{m_d} \end{bmatrix} \cdot \begin{bmatrix} \left(\left(w^T \cdot x_1 + b - y_1\right) \cdot 1_{|w^T \cdot x_1 + b - y_1| \le \delta}\right) + \left(\left(w^T \cdot x_1 + b - y_1\right) \cdot 1_{|w^T \cdot x_1 + b - y_1| > \delta}\right) \\ \vdots \\ \left(\left(w^T \cdot x_m + b - y_m\right) \cdot 1_{|w^T \cdot x_m + b - y_m| \le \delta}\right) + \left(\left(w^T \cdot x_m + b - y_m\right) \cdot 1_{|w^T \cdot x_m + b - y_m| > \delta}\right) \end{bmatrix}$$

So we can conclude for any entry in the result vector:

$$\nabla_w \mathcal{L}_{\mathcal{H}}(\boldsymbol{w}, b) = \frac{1}{m} X^T \cdot \begin{bmatrix} z_1 \\ \vdots \\ z_m \end{bmatrix}$$

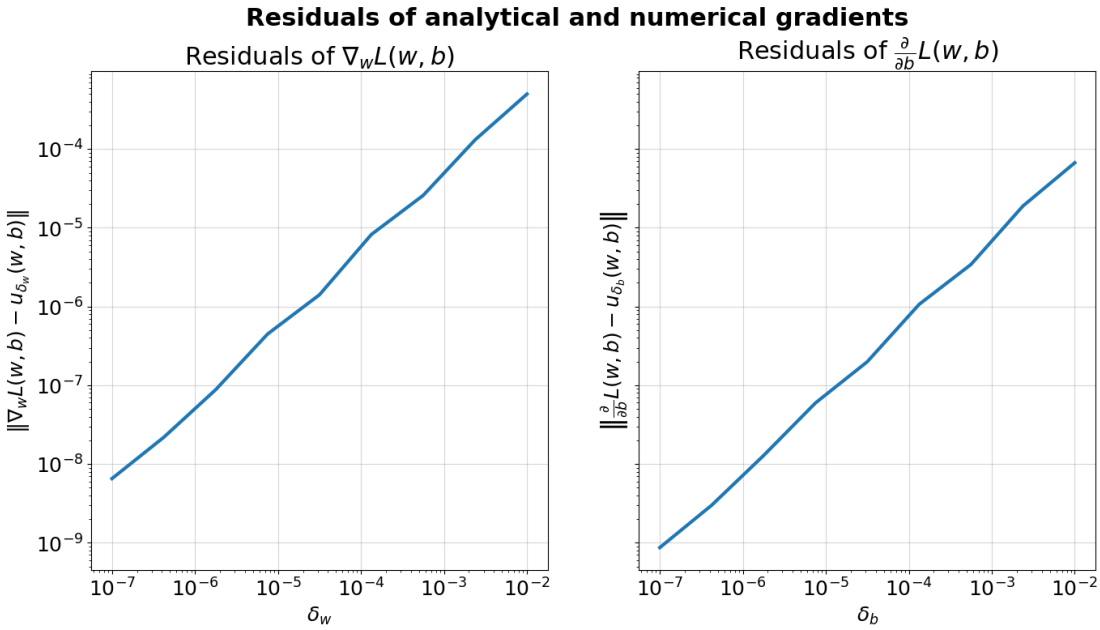
$$when\ Z_i = \begin{cases} \left[X \cdot w + b \cdot 1_m - y\right]_i, & \left|w^T \cdot x_i + b - y_i\right| \le \delta \\ \delta \cdot \left[sign\left(X \cdot w + b \cdot 1_m - y\right)\right]_i, & \left|w^T \cdot x_i + b - y_i\right| > \delta \end{cases}$$

Q3:

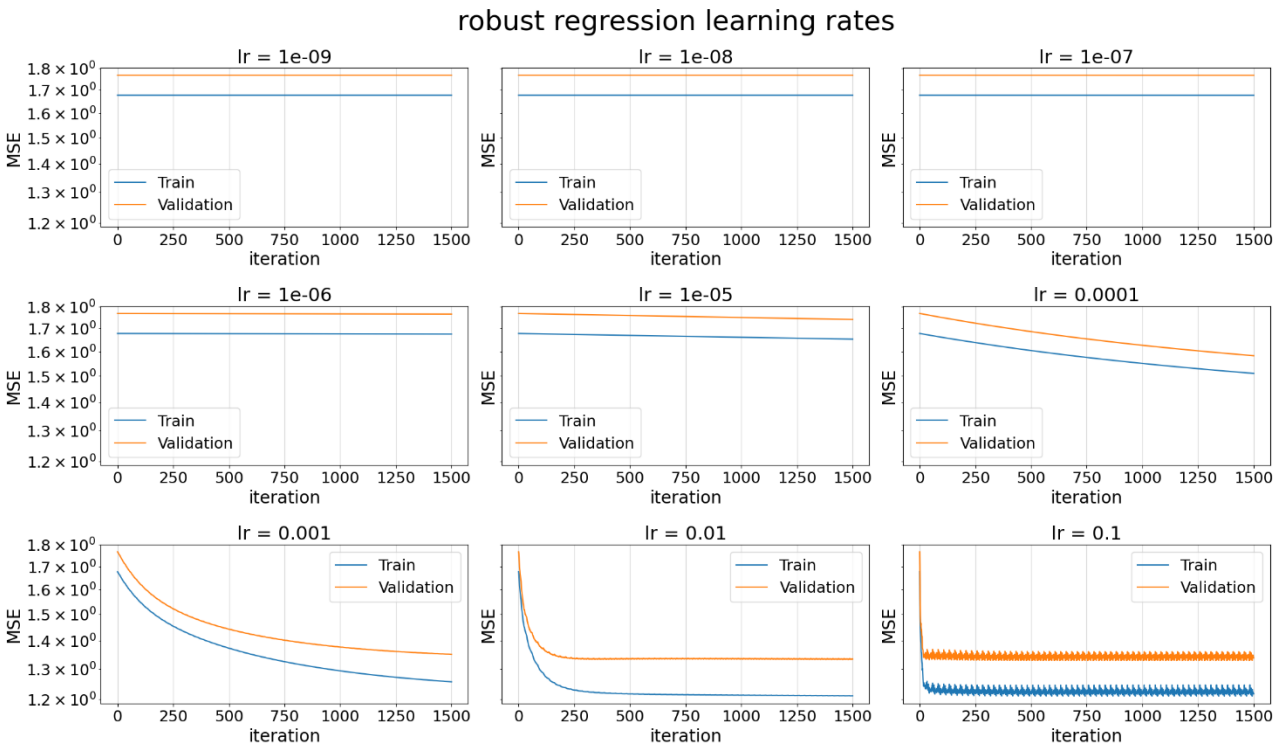
We want to find a good delta which means we want it to be a threshold which will cause only for outliers (which are estimated to be a small percentage of the data) to be calculated using the $\delta \cdot sign\left(w^T \cdot x_i + b - y_i\right)$. so we will have a beliefe p, which is belived to be the percentage of outliers. And we will calculate a regular regression, then save the residuals and look at the 100p% of the biggest residuals. Then we can get the smallest of those big residuals, and select it as our threshold, meaning we will take the square root of that:

```
def good_delta(X,Y, believed_p = 0.05):
    n = len(Y)  # Number of data points
    residuals = []

    w, b = ordinary_least_squares(X, Y)

    for i in range(n):
        residual = (w.T*X[i] + b - Y[i])**2
        residuals.append(residual)

    residuals.sort()

    percentile_index = floor((1-believed_p) * n)
    delta = residuals[percentile_index]

    return delta
```

Q4:



Q5:



As can be seen in the graphs the best learning rate that gives the best convergence is 0.01. This can be simply explained by saying that a for small learning rate 1500 steps might not be enough steps to reach convergence (as can be seen again, in the graphs) and for big learning rate, it might be to big in a way which will make it 'hop' around the minimum point (or even worse, but in the graphs we see the 'hop' (hop tralala)) and not converge. Enlarging the number of iterations doesn't seem to make a lot of sense because it looks like It already converged, and has no where to go anymore. Furthermore, it might start to 'hop' if too many iterations had been made.

Q6:

The Huber loss can be best exploited when there are outliers in the data, because it offers a middle ground between the sensitivity of MSE to outliers and the alternative, not to regard outliers, which can sometimes be a mistake because the model can't really know which are outliers. Unlike ordinary least squares, which can be heavily effected by outliers, Huber loss minimizes their impact by giving a different loss value for large residuals values. This makes it particularly useful in scenarios with noisy data or just data with little outliers, resulting in giving us more reliable regression models.

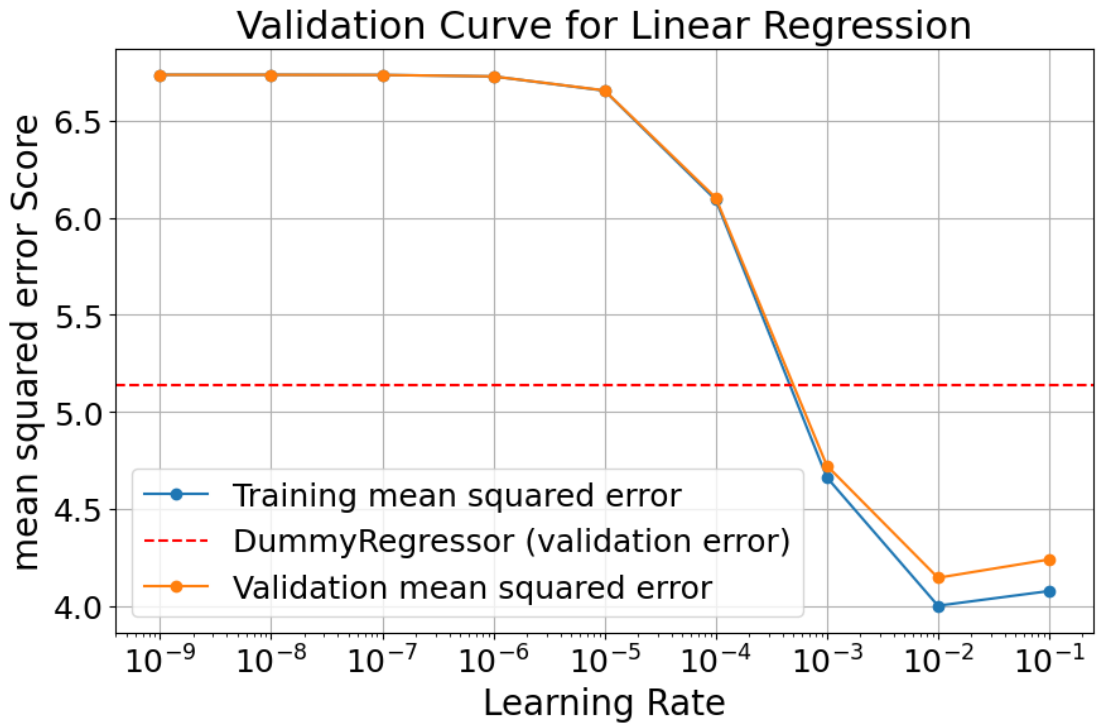
<u>Section 2:</u>

Q7:

The errors are:

Model	Section	Train huber loss	Valid MSE
		Cross validated	
Dummy	2	1.62162	5.13973

Q8:

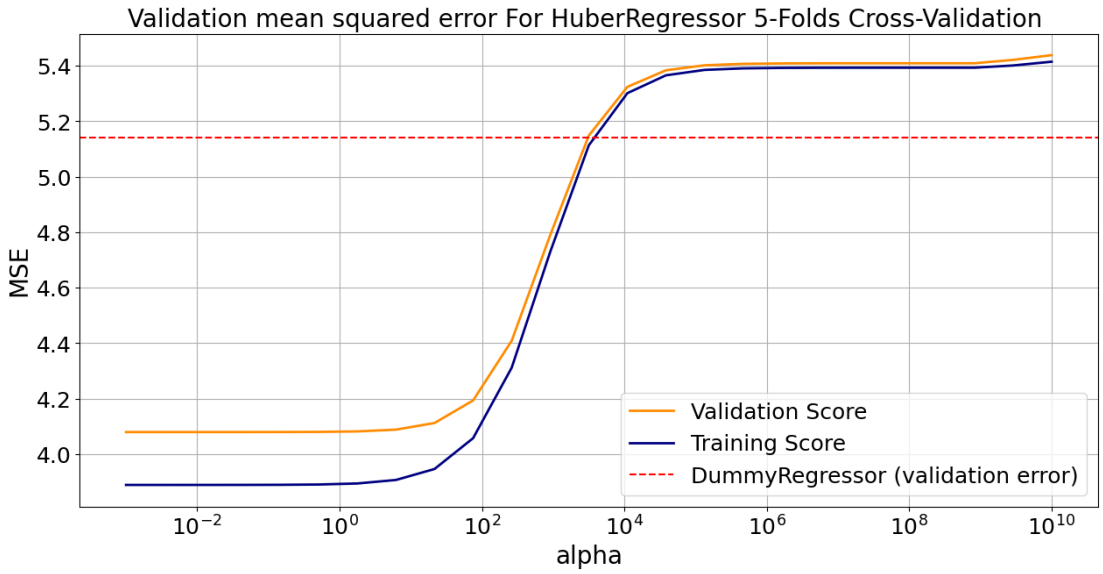


Model	section	Train huber loss	Valid mse
		Cross validated	
dummy	2	1.62162	5.13973
Linear regressor	2	1.23043	4.14708

Q9:
for the linear regression model it is likely that not normalizing beforehand would not matter very much. That is because for linear regression, every change that you make in the scaling of some feature, will be apparent accordingly in the way of predicting, when you come to predict a new x, it will have that feature in the same scale. But for models that use gradient descent for optimization it does matter when deciding on the direction from the gradient vector, because different scaling when not normalized will make it biased more towards the bigger values for no real good reason.

Section 3:

Q10:

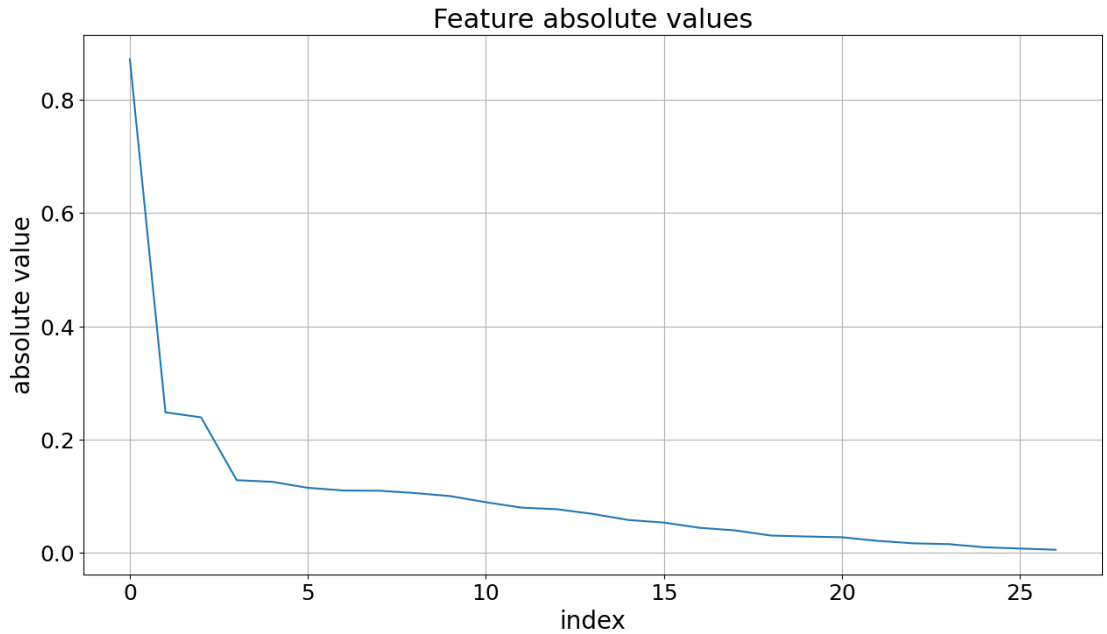


Optimal parameter we got is: 0.001
Validation accuracy is 4.079

Q11:

Model	section	Train huber loss	Valid MSE
		Cross validated	
dummy	2	1.62162	5.13973
Linear regressor	2	1.23043	4.14708
Ridge Linear	3	1.22534	4.14708

Q12:

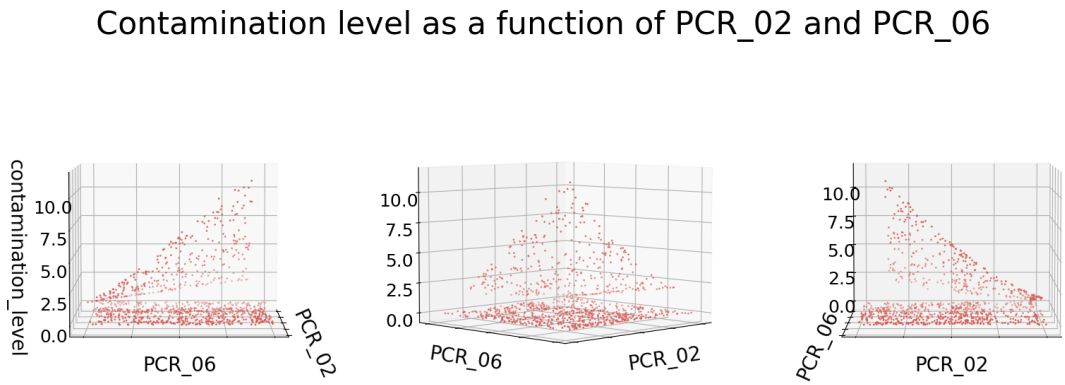


Q13:

Ridge regularization is not a good choice because it does not have the power to leave out feature by reducing their coefficients to zero. It can only lower them close to zero, but then if a feature has very large scale it can still effect the interpretability and there is no way using that to understand the roles of different features in determining behavior. Lasso regression can reduce coefficients to zero thus helps to find a subset of features that effect the model the most, and effectively improving the interp retability.

<u>Section 4 Feature Mappings (visualization):</u>

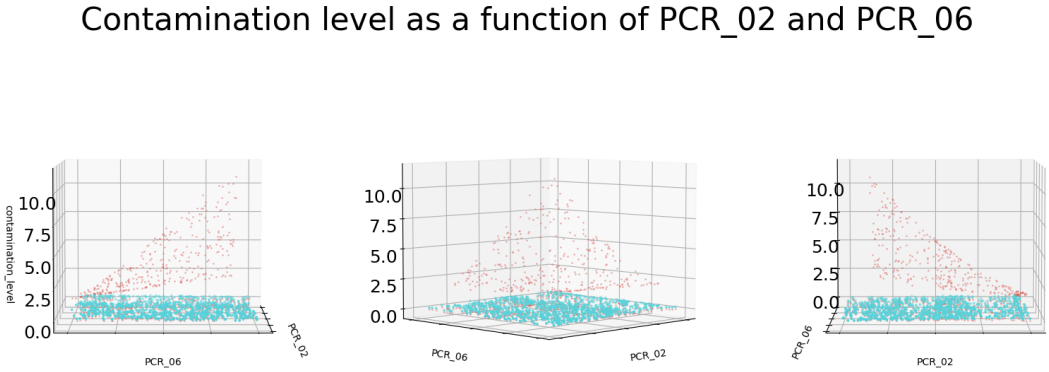
Q14:



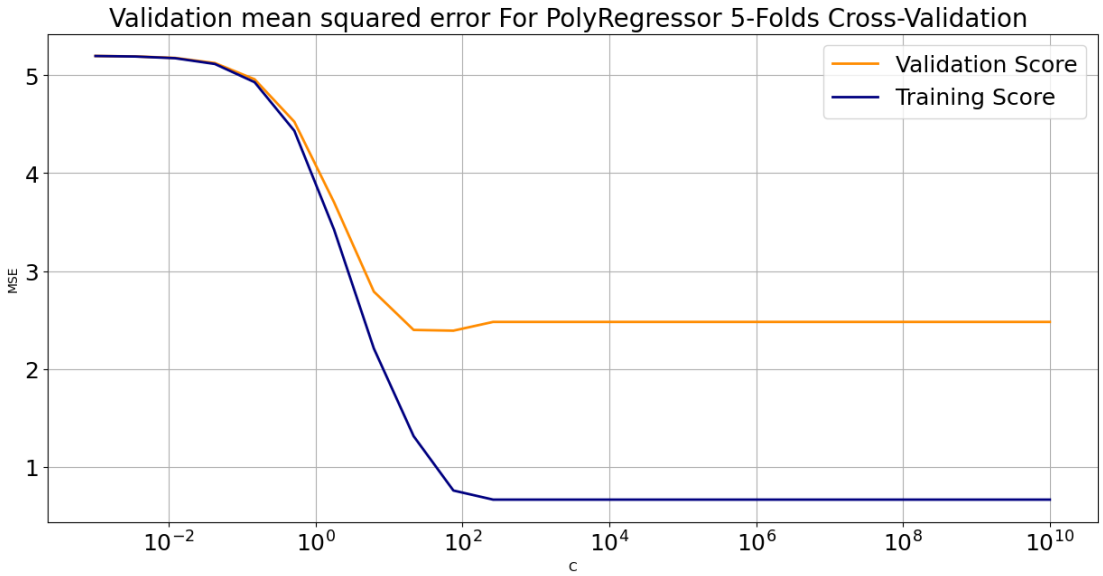
Explanation: we can see the there is some partition of the data such that in one part there

is no affect of combination of PCR2 and PCR6. But in the other part we see a pattern of relationship between them. So we might want to create some feature mapping to a have a better regression.

Q15-16:



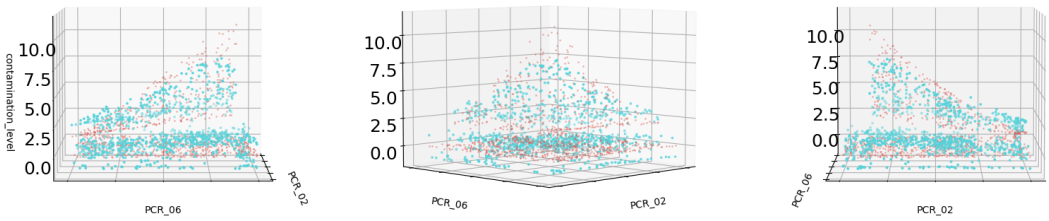
Q17:



$$best\ C\ value\ is\ 74.98942093324558\ , its\ train\ accuracy\ is\ 0.7595527767361085$$
$$and\ its\ validation\ accuracy\ is\ 2.391672498620812$$

Q18:

Contamination level as a function of PCR_02 and PCR_06



Q19:

It seems the last model was better in fitting the data when using the poly_reg. We can see from the graph that there is better correlation between this model predictaions and the actul values (than the formar graph/model).

We could also see that validation accuracy was better (by almost a factor of 2).

Q20:

Model	section	Train huber loss	Valid MSE	TEST MSE
		Cross validated		RETRAINED
dummy	2	1.62162	5.13973	5.320209
Linear regressor	2	1.23043	4.14708	4.338360
Ridge Linear	3	1.22534	4.14708	4.20895
SVR + Laplace	4	0.75	2.39	0.818121