Semester II, Academic Year 2022–2023

Machine Learning

[Course Code: COL341]

Submission: Assignment 1

Name : Mayand Dangi

Entry Number : 2019PH10637

System Specifications

Processor	Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz
RAM	16 GB
Operating System	Windows 10

3.1 Linear Regression

Observations: without normalizing gradient

Learning rate = 0.1

For the learning rate $\eta = 0.1$, maxit $= 60$ and reltol $= 10^{-5}$
MSE of training set: 5.25902e+285
MAE of training set: 7.13923e+142
MSE of validation set: 5.37766e+285
MAE of validation set: 7.26328e+142

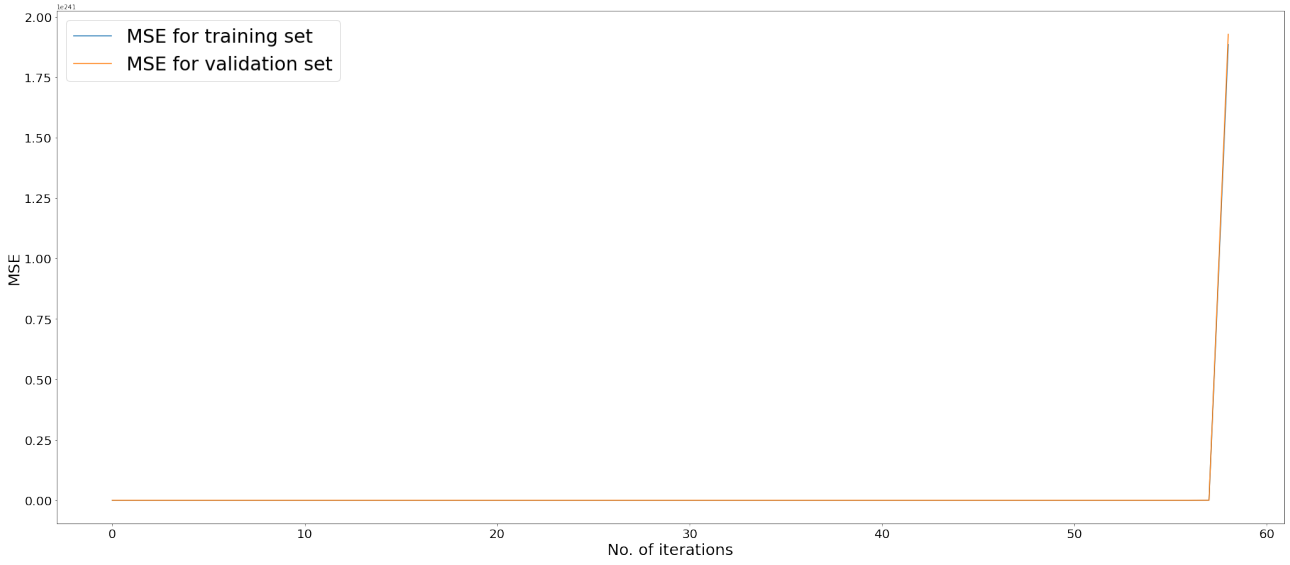

Figure 1: Plot of Mean squared error vs No. of iterations for $\eta = 0.01$ over total iterations

Learning rate = 0.01

For the learning rate $\eta = 0.01$, maxit = 150 and reltol = 10^{-5}
MSE of training set: 9.44301e+296
MAE of training set: 3.02520e+148
MSE of validation set: 9.65604e+296
MAE of validation set: 3.07776e+148

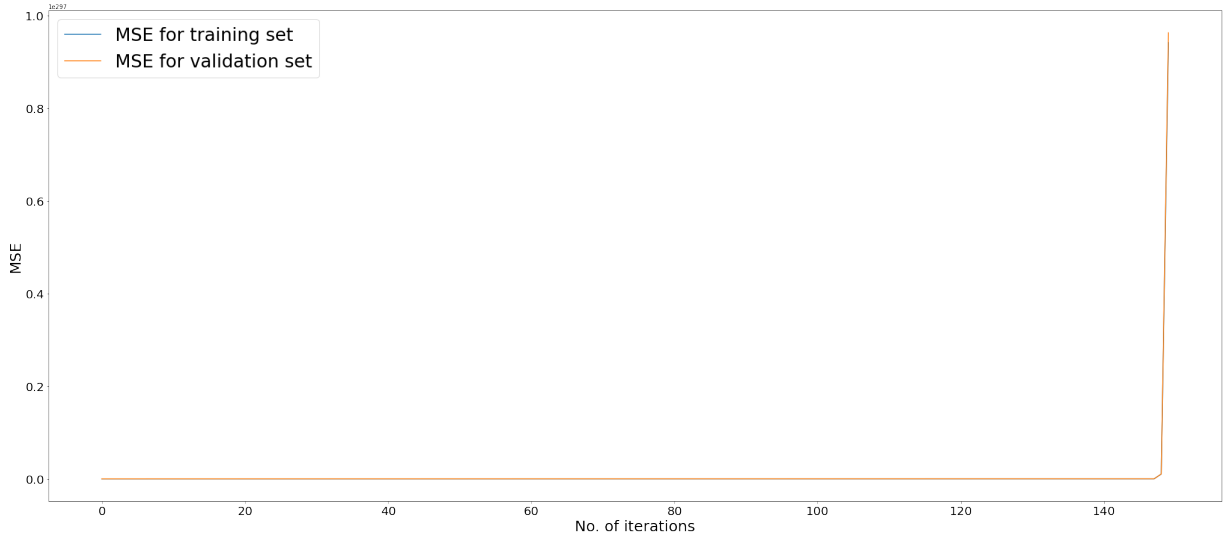

Figure 2: Plot of Mean squared error vs No. of iterations for $\eta = 0.01$ over total iterations

Learning rate = 0.001

For the learning rate $\eta = 0.01$, maxit = 800 and reltol = 10^{-5}
MSE of training set: 0.25180
MAE of training set: 0.38729
MSE of validation set: 0.71601
MAE of validation set: 0.62423

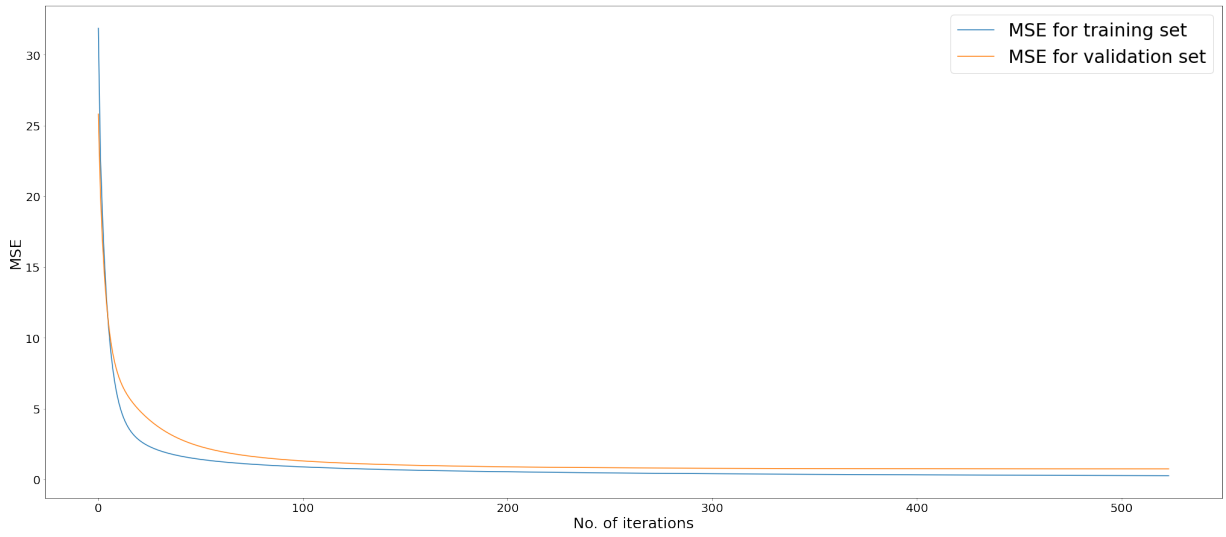

Figure 3: Plot of Mean squared error vs No. of iterations for $\eta = 0.01$ over total iterations

For $\eta = 0.1$ and $\eta = 0.01$, the mean square error diverges, however it converges for $eta = 0.001$, indicating that our learning rates were too high or we experienced the scenario of a vanishing or exploding gradient.

Observations: with normalized gradient

Learning rate = 0.1

For the learning rate $\eta = 0.1$, maxit is set to 800 and we are considering reltol = 10^{-6}.
MSE of training set: 1.49107
MAE of training set: 1.16110
MSE of validation set: 2.00211
MAE of validation set: 1.20754
Both the training data and the validation data have such high MSE and MAE values, which means that the minimum solution has not been found.

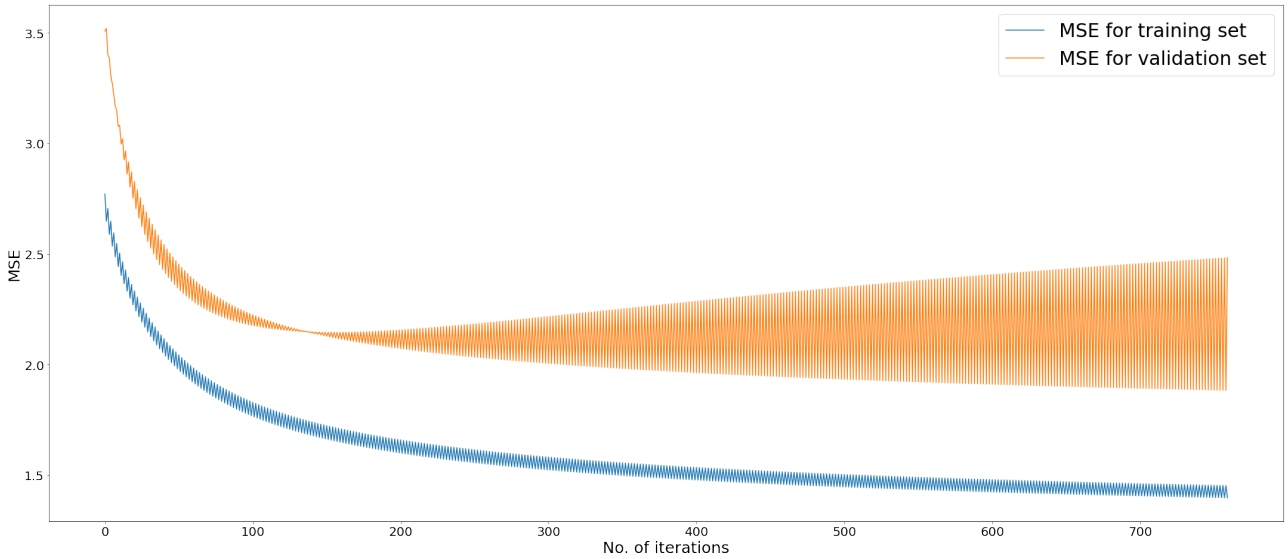

Figure 4: Plot of Mean squared error vs No. of iterations

From the above plot, we can conclude that our cost function is oscillating. We can draw the following conclusion:

- Learning rates are high: high learning rates may have caused the overshoot and made the MSE oscillate back and forth, causing high fluctuations.
- Poor convergence: The oscillating MSE plot indicates poor convergence, i.e., we haven't arrived at minima yet.

Learning rate = 0.01

For the learning rate $\eta = 0.01$, maxit is set to 800 and we are considering reltol = 10^{-5}.
MSE of training set: 0.20552
MAE of training set: 0.34879

MSE of validation set: 0.77198

MAE of validation set: 0.69197

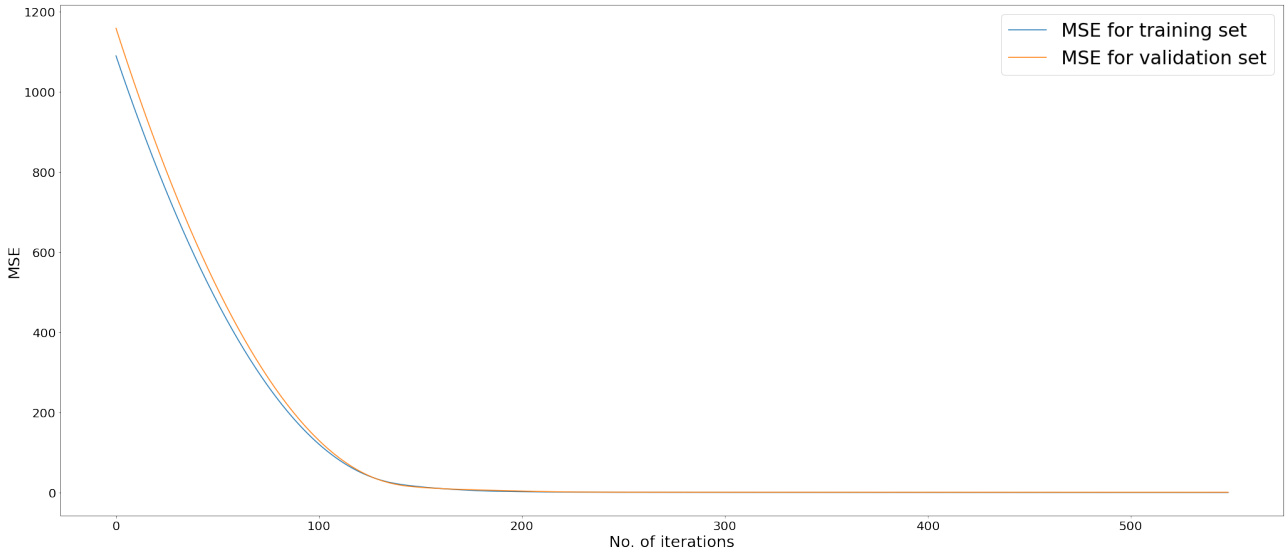

Figure 5: Plot of mean squared error vs no. of iterations for $\eta = 0.01$ over total iterations

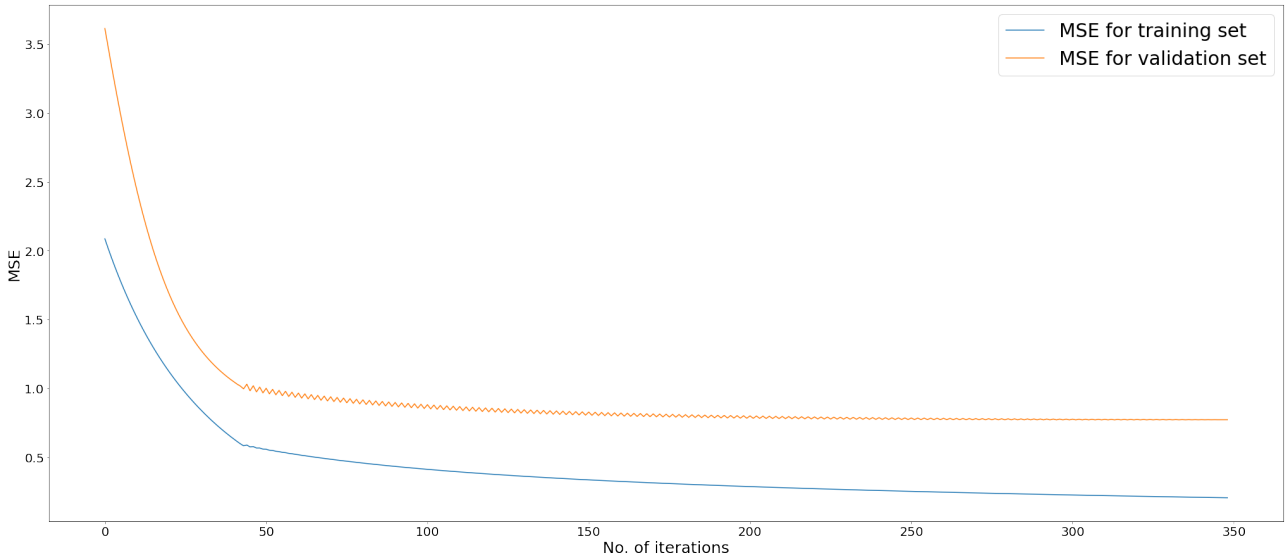

Figure 6: Zoomed version of Figure 5 after 200^{th} iterations

Also, if the reltol is set of 10^{-6}, so that we get MSE distribution for complete iteration

MSE of training set: 0.11609

MAE of training set: 0.26661

MSE of validation set: 0.95644

MAE of validation set: 0.71654

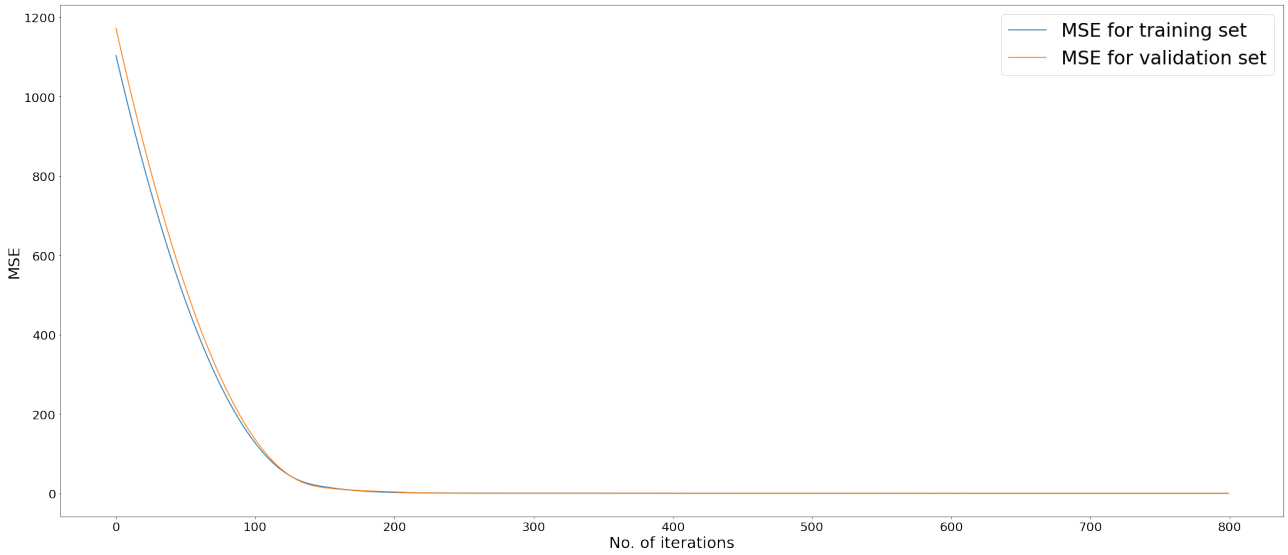

Figure 7: Plot of Mean squared error vs No. of iterations for $\eta = 0.01$ over total iterations

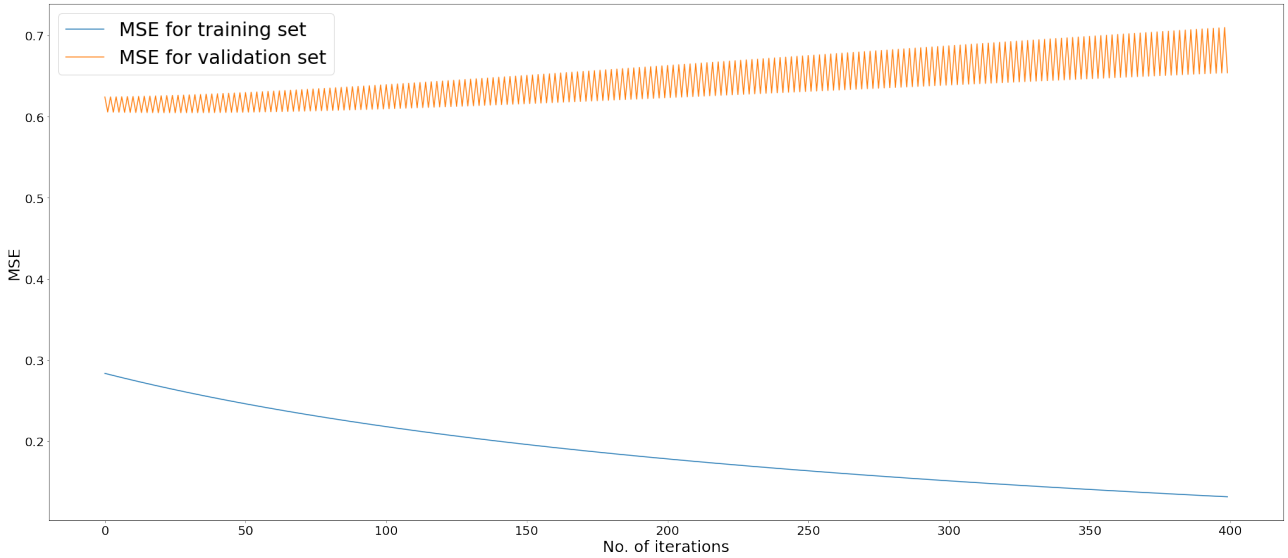

Figure 8: Zoomed version of Figure 2 after 200^{th} iterations

In Figure 4 and Figure 6, we can see that there is oscillation in the MSE value for the validation set, while the MSE is constantly decreasing for the training set without any oscillation. Nonetheless, after a significant number of iterations, the oscillation in the validation set($\eta = 0.1$) fades and reappears. However, in other cases, it is increasing constantly($\eta = 0.1$).

This observation may lead us to draw the conclusion that, after certain iterations, our model is getting over-fitted, as the MSE on the training data is decreasing quickly but the MSE on the validation data is oscillating. Here, the model is fitting the noise in the training data instead of the underlying relationship. Another possibility is that learning is slightly higher.

Learning rate = 0.001

For the learning rate $\eta = 0.001$, maxit is set to 5000 and we are considering reltol $= 10^{-5}$.
MSE of training set: 0.21986

MAE of training set: 0.36577
MSE of validation set: 0.78489
MAE of validation set: 0.65693

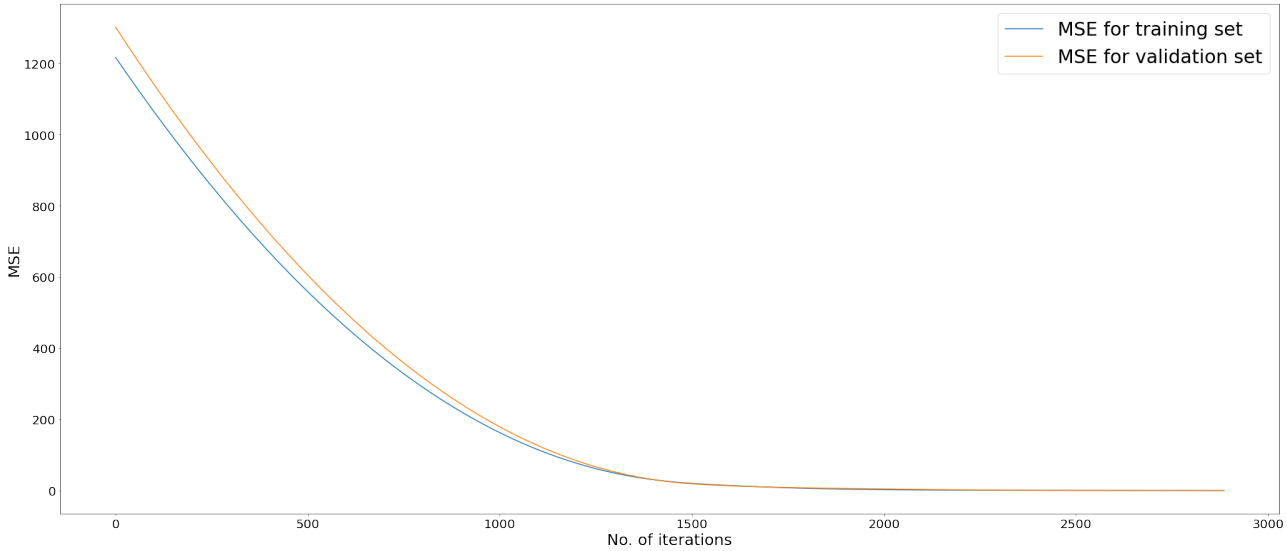

Figure 9: Plot of Mean squared error vs No. of iterations for $\eta = 0.001$ over total iterations

We didn't get any oscillations at $\eta = 0.001$, but we did get them in Figures 3 and 5. So, we can say that this is one of the optimal learning rates. Also, it can be well observed that as we have decreased the learning rates, the number of iterations required for them to converge has also increased. Also, our reltol stopping criteria is working properly, as it stopped training when the relative change in MSE on validation data was sufficiently small.

3.2 Ridge Regression

We have, $E_{in}(\theta) = \sum_{i=1}^{m}\big(y_i - \sum_{j=1}^{n}\theta_i x_{ij}\big)^2 + \lambda\sum_{j=1}^{n}\theta_j^2$, which will in matrix form will be $E_{in}(\theta) = ||X\theta - \mathbf{y}||^2 + \lambda\theta^T\theta$. Thus, the gradient of $E_{in}(\theta)$ will be

$$\begin{aligned}\nabla_\theta E_{in}(\theta) &= \nabla_\theta\big[||X\theta - \mathbf{y}||^2 + \lambda\theta^T\theta\big] \\ &= \nabla_\theta||X\theta - \mathbf{y}||^2 + \lambda\nabla_\theta\theta^T\theta \\ &= \big((X^TX + (X^TX)^T)\theta - 2X^T\mathbf{y}\big) + 2\lambda\theta \\ &= 2\big(X^TX\theta - X^T\mathbf{y}\big) + 2\lambda\theta\end{aligned}$$

Therefore the new update equation will be given by

$$\theta(t+1) = \theta(t) - \eta\Big(2\big(X^TX\theta - X^T\mathbf{y}\big) + 2\lambda\theta\Big)$$

where, η is learning rate.

Observations

$\lambda = 5$, **Learning rate**$= 0.1$

For the learning rate $\eta = 0.1$, maxit $= 30$ and reltol $= 10^{-5}$
MSE for training set: 8.50108e+253
MAE for training set: 9.076878e+126
MSE for validation set: 8.69286e+253
MAE for validation set: 9.23459e+126

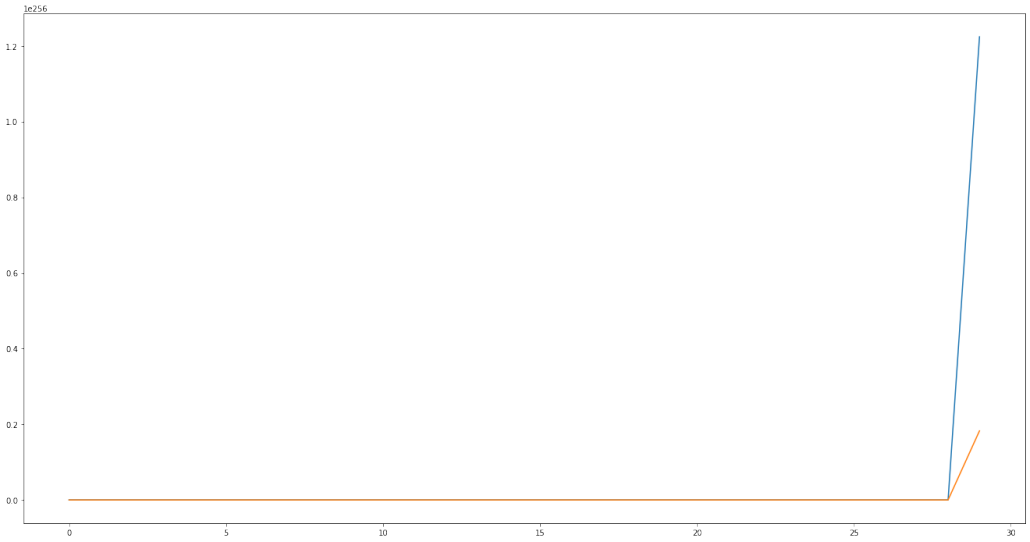

Figure 10: Plot of Mean squared error vs No. of iterations for $\eta = 0.1$ over total iterations

$\lambda = 5$, **Learning rate**$= 0.01$

For the learning rate $\eta = 0.01$, maxit $= 45$ and reltol $= 10^{-5}$
MSE for training set: 8.50108e+253
MAE for training set: 9.076878e+126
MSE for validation set: 8.69286e+253
MAE for validation set: 9.23459e+126

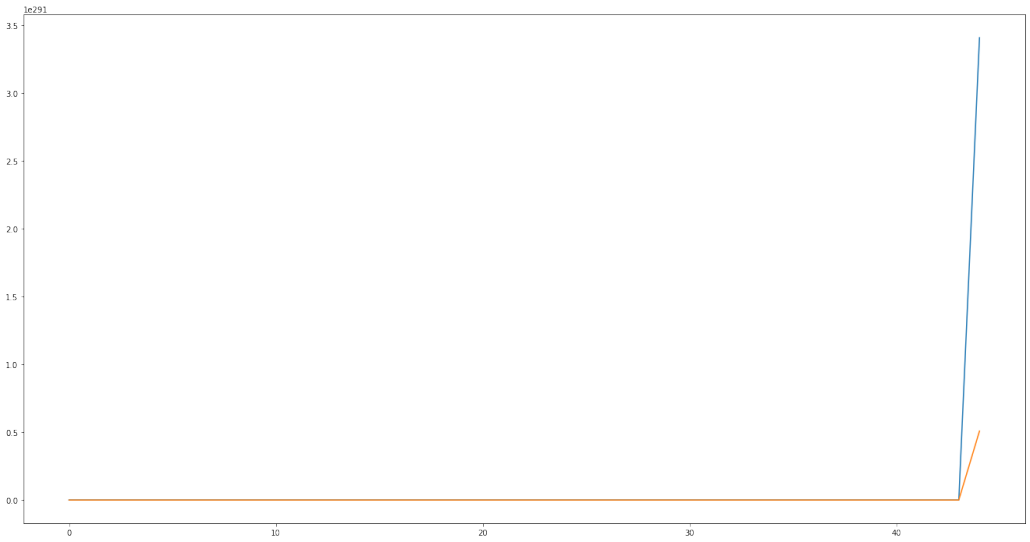

Figure 11: Plot of Mean squared error vs No. of iterations for $\eta = 0.01$ over total iterations

$\lambda = 5$, **Learning rate**= 0.001

For the learning rate $\eta = 0.001$, maxit = 65 and reltol = 10^{-5}
MSE for training set: 1.82356e+286
MAE for training set: 1.32941e+143
MSE for validation set: 1.86470e+286
MAE for validation set: 1.35251e+143

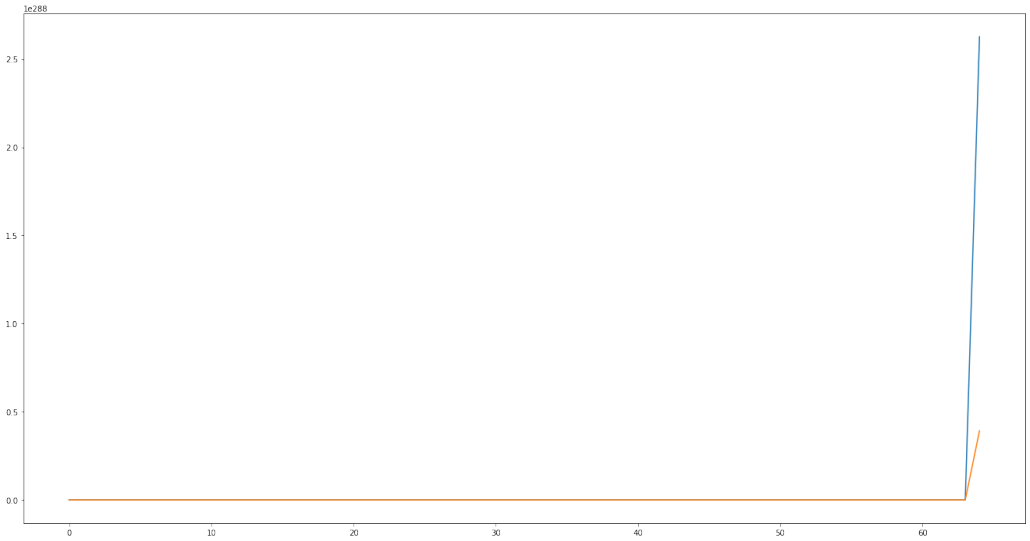

Figure 12: Plot of Mean squared error vs No. of iterations for $\eta = 0.001$ over total iterations

$\lambda = 5$, **Learning rate**= 0.00001

For the learning rate $\eta = 0.00001$, maxit = 1000 and reltol = 10^{-5}
MSE for training set: 0.20230
MAE for training set: 0.34215
MSE for validation set: 0.77124

MAE for validation set: 0.67459

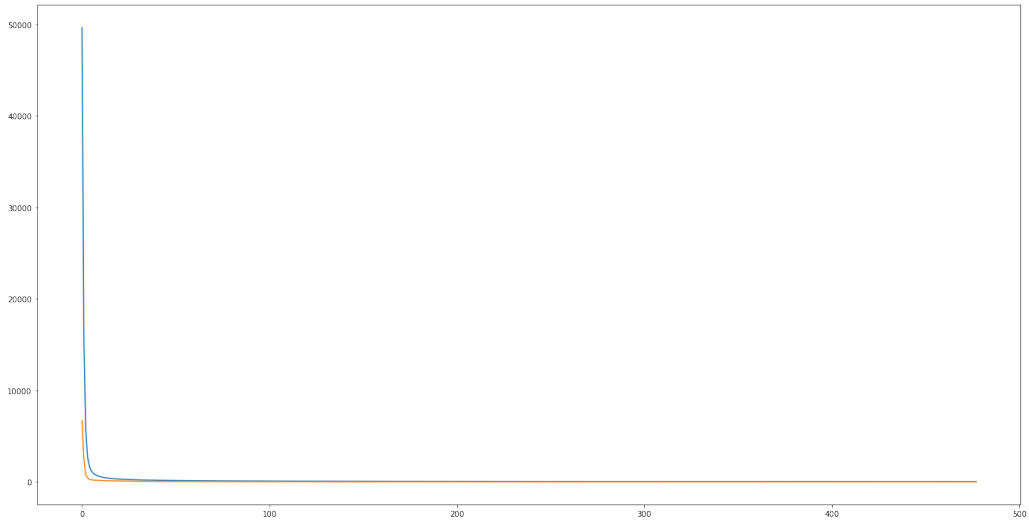

Figure 13: Plot of Mean squared error vs No. of iterations for $\eta = 0.00001$ over total iterations

$\lambda = 25$, **Learning rate**$= 0.1$

For the learning rate $\eta = 0.1$, maxit $= 35$ and reltol $= 10^{-5}$
MSE for training set: 5.69777e+295
MAE for training set: 7.43108e+147
MSE for validation set: 5.82631e+295
MAE for validation set: 7.56020e+147

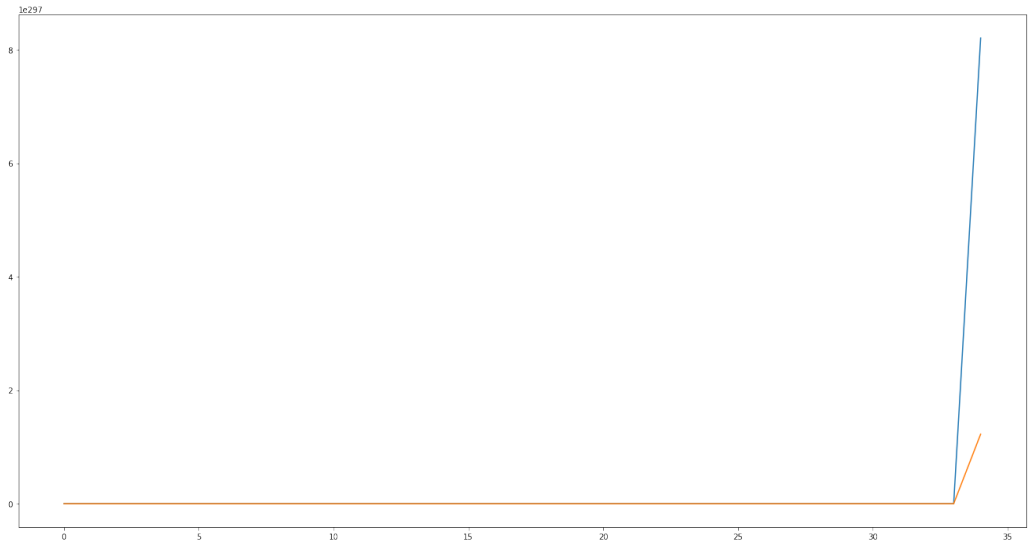

Figure 14: Plot of Mean squared error vs No. of iterations for $\eta = 0.1$ over total iterations

$\lambda = 25$, **Learning rate**$= 0.01$

For the learning rate $\eta = 0.01$, maxit $= 45$ and reltol $= 10^{-5}$
MSE for training set: 2.42787e+289

MAE for training set: 4.85078e+144
MSE for validation set: 2.48264e+289
MAE for validation set: 4.93507e+144

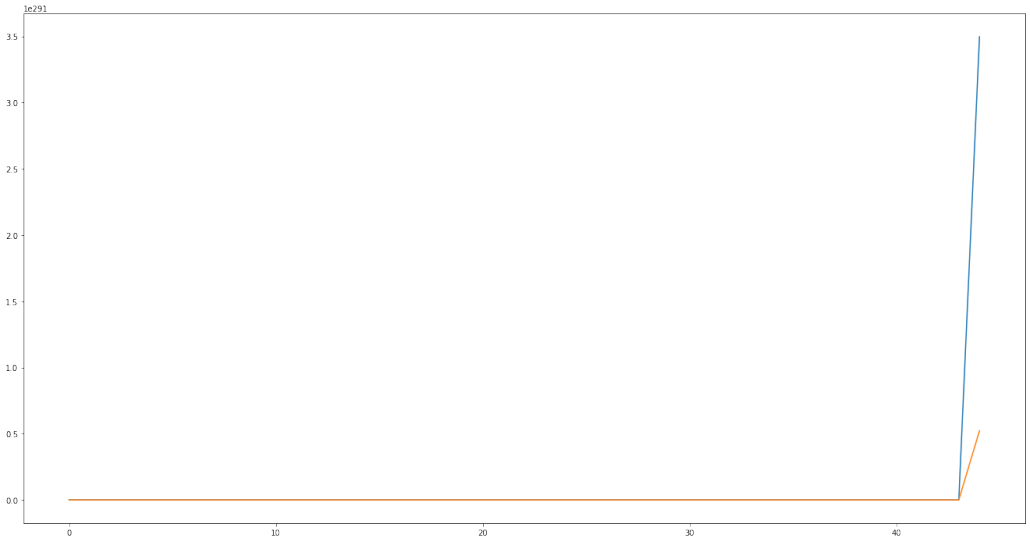

Figure 15: Plot of Mean squared error vs No. of iterations for $\eta = 0.01$ over total iterations

$\lambda = 25$**, Learning rate**$= 0.001$

For the learning rate $\eta = 0.001$, maxit $= 65$ and reltol $= 10^{-5}$
MSE for training set: 1.78563e+286
MAE for training set: 1.31551e+143
MSE for validation set: 1.82592e+286
MAE for validation set: 1.33837e+143

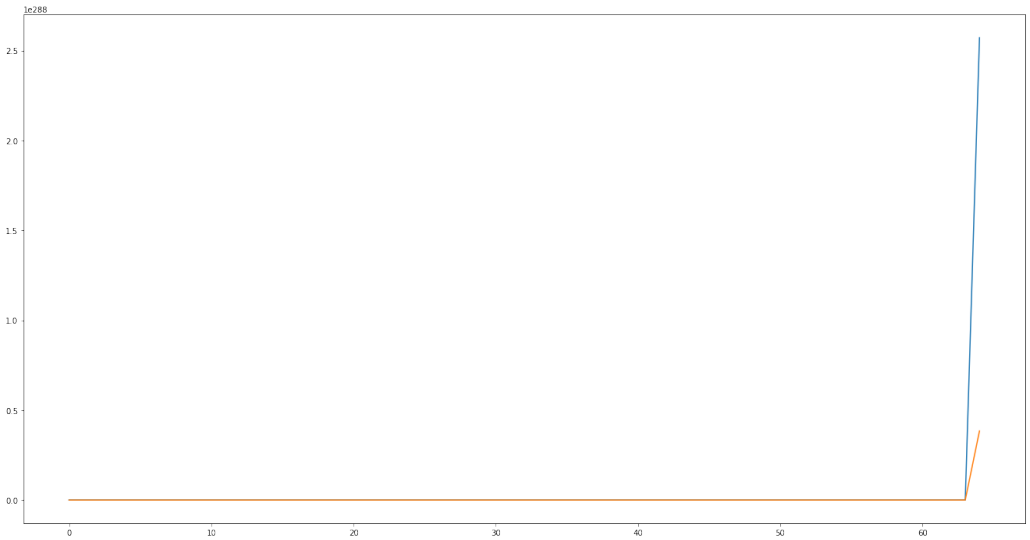

Figure 16: Plot of Mean squared error vs No. of iterations for $\eta = 0.001$ over total iterations

$\lambda = 25$**, Learning rate**$= 0.00001$

For the learning rate $\eta = 0.00001$, maxit $= 5000$ and reltol $= 10^{-5}$
MSE for training set: 0.08174
MAE for training set: 0.22249
MSE for validation set: 0.77448
MAE for validation set: 0.67287

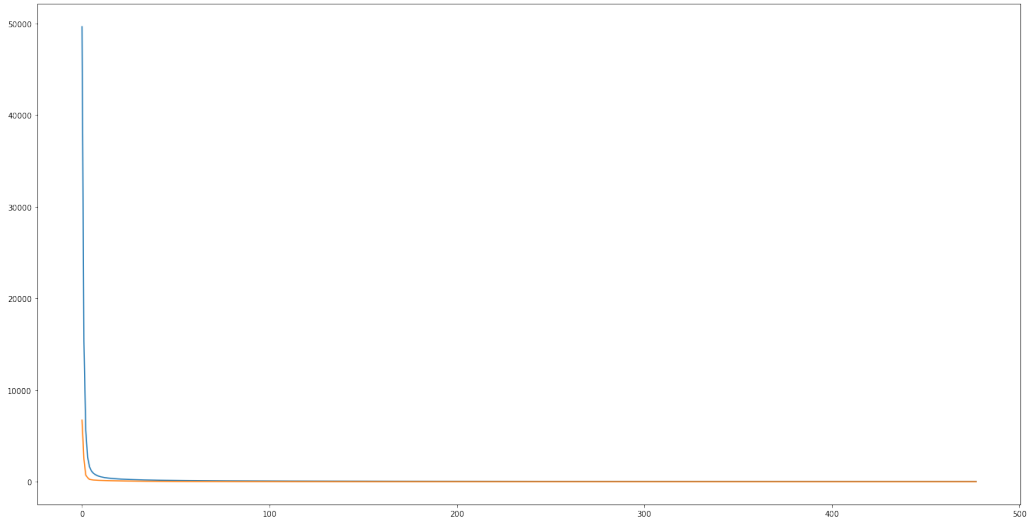

Figure 17: Plot of Mean squared error vs No. of iterations for $\eta = 0.1$ over total iterations

The above results for different cases shows that MSE loss diverges for the learning rate 0.1, 0.01 and 0.001. For $\eta = 0.00001$, our model converges and we get the good expected results. The MSE and MAE loss is less for $\lambda = 25$ as compares to $\lambda = 5$.

Comparision of Linear and Ridge regression results

The combination of MSE and MAE from ridge regression and linear regression provides us insights about the bias-variance tradeoff.

In general, MSE is a more sensitive metric to large errors since it squares the differences between the predicted and actual values before taking the mean. This means that large errors have a greater impact on the overall MSE than they do on the MAE. On the other hand, MAE is more robust to outliers and large errors since it only takes the absolute value of the differences.

Comparing the MSE and MAE between ridge regression and linear regression can provide insights into the bias-variance tradeoff. If the MSE for ridge regression is less than that for linear regression, then ridge regression has a better balance between bias and variance and is less likely to overfit the training data. On the other hand, if the MSE is larger for ridge regression, it could have added too much bias and doesn't fit the data well enough. Same with the MAE; if the MAE for ridge regression is less than that for linear regression, this indicates

that ridge regression produces more accurate predictions on average and may be less impacted by outliers. If the MAE is greater for ridge regression, it may imply that the model is overly penalised. In our case, ridge regression has lower MSE and MAE losses than linear regression, indicating that it has a better fit than the linear regression model. This may also indicate that our features are co-linear and the penalty term $||\theta||^2$ penalised the cost function which eventually help in regularising and prevent overfitting.

3.3 Using Scikit-Learn Library

Scikit-Learn library has inbuilt function for linear regression and ridge regression.

Linear Regression

```
regr = LinearRegression()
regr.fit(X_train, y_train)

yt_pred = regr.predict(X_train)
yv_pred = regr.predict(X_validation)
```

MSE of training data 1.95006e-29
MAE of training data 3.26744e-15
MSE of validation data 1.02503
MAE of validation data 0.83215
The MSE and MAE errors observed during training are $1.95e-29$ and $3.26e-15$, respectively. This is much lower than what was achieved using the algorithm implemented in Task 3.1. However, the MSE observed in the validation set is quite large. This indicates that the model might be overfitted, and the high MAE indicates that there are a large number of outliers. The model might have learned the noise and random fluctuations in the training data instead of the underlying patterns and relationships.

Ridge Regression

```
ridge = Ridge(alpha = 5.0)
ridge.fit(X_train, y_train)

sk_rmse_t = mean_squared_error(y_true=y_train,y_pred=ridge.predict(X_train))
sk_rmse_v =
    mean_squared_error(y_true=y_validation,y_pred=ridge.predict(X_validation))
```

For, $\lambda = 5$
MSE of training data 0.010807
MAE of training data 0.078513

MSE of validation data 0.93190
MAE of validation data 0.78030
The MSE and MAE loss is slightly higher from the achieved value by the model implemented in task 3.2.
For, $\lambda = 25$
MSE of training data 0.07889
MAE of training data 0.21908
MSE of validation data 0.77168
MAE of validation data 0.68159
The MSE and MAE loss in this case is almost same w.r.t the loss generated by the model implemented in task 3,2 for $\lambda = 25$.

If we compare the losses of ridge and linear regression implemented using Scikit-Learn, we can say that ridge regression has a better balance for the bias-variance tradeoff, while the linear regression model is overfitted.

3.4 Feature Selection

First of all we have selected best 10 feature, from the inbuilt method SelectKBest of Scikit-learn library.

```
# combine the training and validation set to select same best 10 feature
X_comb = np.vstack((X_train, X_validation))
y_comb = np.vstack((y_train, y_validation))
X_new = SelectKBest(r_regression, k=10).fit_transform(X_comb, np.ravel(y_comb))

# split the training and validation set
X_train_new = X_new[0:144,:]
X_valid_new = X_new[144:,:]

w4, MSE_t4, MSE_v4, MSE_relative4 = gradient_descent_withE(X_train_new,
    y_train, X_valid_new, y_validation, 0.1, 5000, 10e-5, 0)
```

The losses reported are:
MSE for training set: 1.80495
MAE for training set: 1.06236
MSE for validation set: 1.72795
MAE for validation set: 1.05551

The MSE and MAE losses are much higher for the model with 10 features compared to the original model having all features. It could be because the model is missing important information that is contained in the other features. The exclusion of these features may not be able to capture the full complexity of the relationship between the predictors and the target variable, leading to increased error in the predictions. It is also possible that the feature

selection process did not accurately identify the 10 best features for our dataset. Now, we will select best 10 features using SelectFromModel method.

```
    # combine the training and validation set to select same best 10 feature
    X_comb = np.vstack((X_train, X_validation))
    y_comb = np.vstack((y_train, y_validation))

    bestfeature_model = SelectFromModel(Ridge(), max_features=10)
    bestfeature_model.fit(X_comb, y_comb)
    X_new = bestfeature_model.transform(X_comb)

    # split the training and validation set
    X_train_new = X_new[0:144,:]
    X_valid_new = X_new[144:,:]

    w4, MSE_t4, MSE_v4, MSE_relative4 = gradient_descent_withE(X_train_newr,
       y_train, X_valid_newr, y_validation, 0.1, 5000, 1e-6, 0)
```

The losses reported are:
MSE for training set: 2.02133
MAE for training set: 1.13087
MSE for validation set: 2.23345
MAE for validation set: 1.22002

For this case also, we have much higher MSE than the model having all the features.

3.5 Classification

The cost function for the multiclassification using logistic regression can be given by:

$$J(\theta) = -\Big[\sum_{i=1}^{N}\sum_{k=1}^{r} \mathbb{I}(\mathbf{y}^{(i)} = k) \log h_{\theta_k}(\mathbf{x}^{(i)})\Big]$$

where,

$$h_{\theta_k}(\mathbf{x}^{(i)}) = \frac{exp(\theta^{(k)T}\mathbf{x^{(i)}})}{1 + \sum_{j=1}^{8} exp(\theta^{(j)T}\mathbf{x}^{(i)})}$$

Let's calculate the gradient of $J(\theta)$

$$\begin{aligned}\nabla_{\theta^{(k)}} J(\theta) &= -\nabla_{\theta^{(k)}}\Big[\sum_{i=1}^{N}\sum_{k=1}^{r} \mathbb{I}(\mathbf{y}^{(i)} = k) \log h_{\theta_k}(\mathbf{x}^{(i)})\Big] \\ &= -\nabla_{\theta^{(k)}}\Big[\sum_{i=1}^{N}\sum_{k=1}^{r} \mathbb{I}(\mathbf{y}^{(i)} = k) \log \frac{exp(\theta^{(k)T}\mathbf{x^{(i)}})}{1 + \sum_{j=1}^{8} exp(\theta^{(j)T}\mathbf{x}^{(i)})}\Big]\end{aligned}$$

$$= -\Big[\sum_{i=1}^{N}\sum_{k=1}^{r} \nabla_{\theta^{(k)}} \mathbb{I}(\mathbf{y}^{(i)} = k)(\log(exp(\theta^{(k)T}\mathbf{x}^{(i)})) - \log(1 + \sum_{j=1}^{8} exp(\theta^{(j)T}\mathbf{x}^{(i)})))\Big]$$

$$= -\Big[\sum_{i=1}^{N}\sum_{k=1}^{r} \mathbb{I}(\mathbf{y}^{(i)} = k)(\nabla_{\theta^{(k)}}\theta^{(k)T}\mathbf{x}^{(i)} - \nabla_{\theta^{(k)}} \log(1 + \sum_{j=1}^{8} exp(\theta^{(j)T}\mathbf{x}^{(\mathbf{i})})))\Big]$$

$$= -\Big[\sum_{i=1}^{N}\sum_{k=1}^{r} \mathbb{I}(\mathbf{y}^{(i)} = k)\mathbf{x}^{(i)} - \frac{\nabla_{\theta^{(k)}}(1 + \sum_{j=1}^{8} exp(\theta^{(j)T}\mathbf{x}^{(i)}))}{1 + \sum_{j=1}^{8} exp(\theta^{(j)T}\mathbf{x}^{(i)})}\Big]$$

$$= -\Big[\sum_{i=1}^{N}\sum_{k=1}^{r} \mathbf{x}^{(i)}\Big(\mathbb{I}(\mathbf{y}^{(i)} = k) - \frac{exp(\theta^{(k)T}\mathbf{x}^{(i)}))}{1 + \sum_{j=1}^{8} exp(\theta^{(j)T}\mathbf{x}^{(i)})}\Big)\Big]$$

Now, we will try to vectorize the above equation so that we can use that as update parameter for our weights. For which first of all we have to create a matrix say Y corresponding to term $\mathbb{I}(\mathbf{y}^{(i)} = k)$. This matrix will be like, that each column of matrix Y corresponds to particular class and particular cell of given column will be 1 if $\mathbf{y}^{(i)} = k$, where k is the class corresponding to the selected column and rest will be zero. And, this simplifies our equation to

$$\nabla_\theta J(\theta) = X^T(Y - h_\theta(X))$$

where, the dimension of X is $N \times (d+1)$, Y is $N \times r$ and $h_\theta(X)$ is $(d+1) \times k$. Here, the hyper parameter θ in matrix form will be like:

$$\begin{bmatrix} | & | & ..... & | \\ \theta^{(1)} & \theta^{(2)} & ..... & \theta^{(r)} \\ | & | & ..... & | \end{bmatrix}$$

So, update function will be like

$$\theta(t+1) = \theta(t) - \eta X^T(Y - h_\theta(X))$$

where, η is learning rate.

3.6 Visualization

The plot of the training and validation MSE loss (in the same graph) across the iterations for Tasks 3.1 and 3,2 can be found in sections 3.1 and 3,2.

For task 3.4, as it is already mentioned in section 3.4 that we are getting higher MSE as compared to original model MSE.

The reported MSE for the model where 10 best features are picked from SelectKBest techniques at various learning rates, along with their plots vs number of iterations, are as follows:

- Learning rate = 0.1

MSE for training set: 1.92494
MAE for training set: 1.10476
MSE for validation set: 1.62311
MAE for validation set: 1.02702

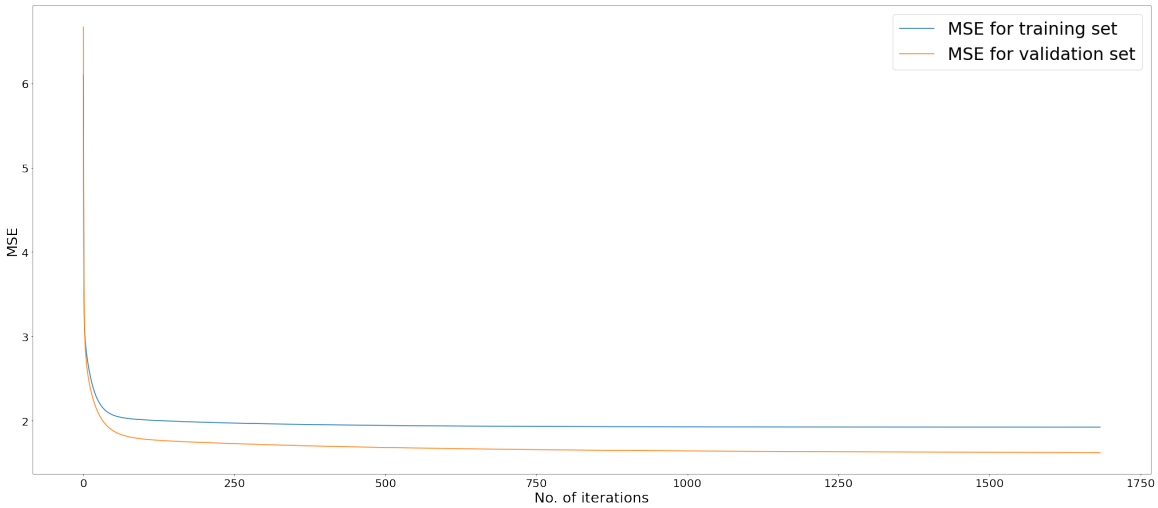

Figure 18: Plot of Mean squared error vs No. of iterations for $\eta = 0.1$ over total iterations

- Learning rate = 0.01
  MSE for training set: 1.95314
  MAE for training set: 1.11027
  MSE for validation set: 1.69881
  MAE for validation set: 1.03989

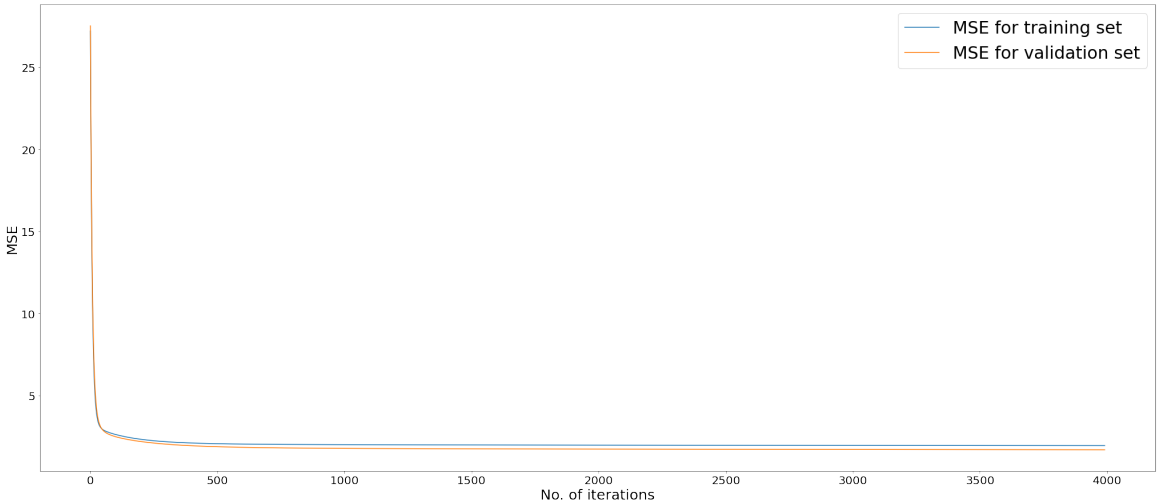

Figure 19: Plot of Mean squared error vs No. of iterations for $\eta = 0.01$ over total iterations

- Learning rate = 0.001
  MSE for training set: 1.95315
  MAE for training set: 1.11027

MSE for validation set: 1.69882
MAE for validation set: 1.03989

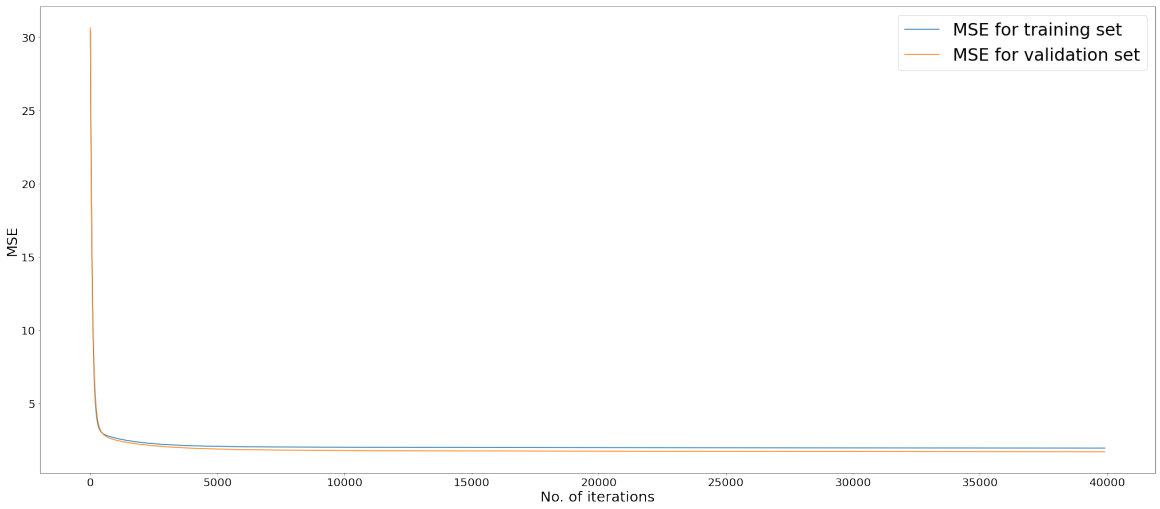

Figure 20: Plot of Mean squared error vs No. of iterations for $\eta = 0.001$ over total iterations

The MSE for training and validation set are almost same for every learning rates, which indicates that we are with perfect learning rates and not stopped so early, but still these MSE are higher than expected, that can be due to the model is missing important information that is contained in the other features.

The reported MSE for the model where 10 best features are picked from SelectFromModel techniques at various learning rates, along with their plots vs number of iterations, are as follows:

- Learning rate = 0.1
  MSE for training set: 2.02133
  MAE for training set: 1.13087
  MSE for validation set: 2.23345
  MAE for validation set: 1.22002

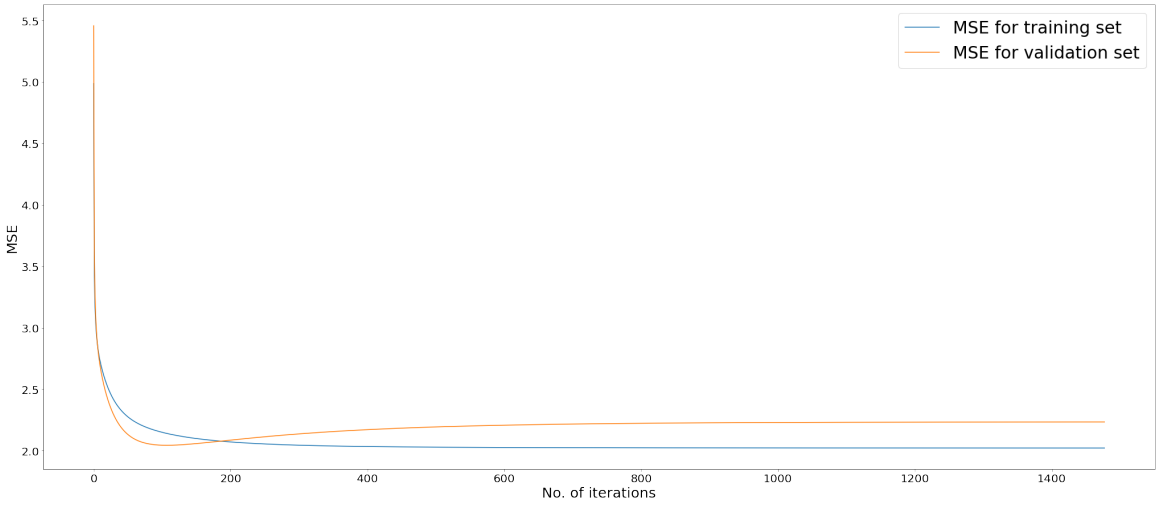

Figure 21: Plot of Mean squared error vs No. of iterations for $\eta = 0.1$ over total iterations

- Learning rate = 0.01
  MSE for training set: 2.13989
  MAE for training set: 1.16850
  MSE for validation set: 2.04582
  MAE for validation set: 1.18477

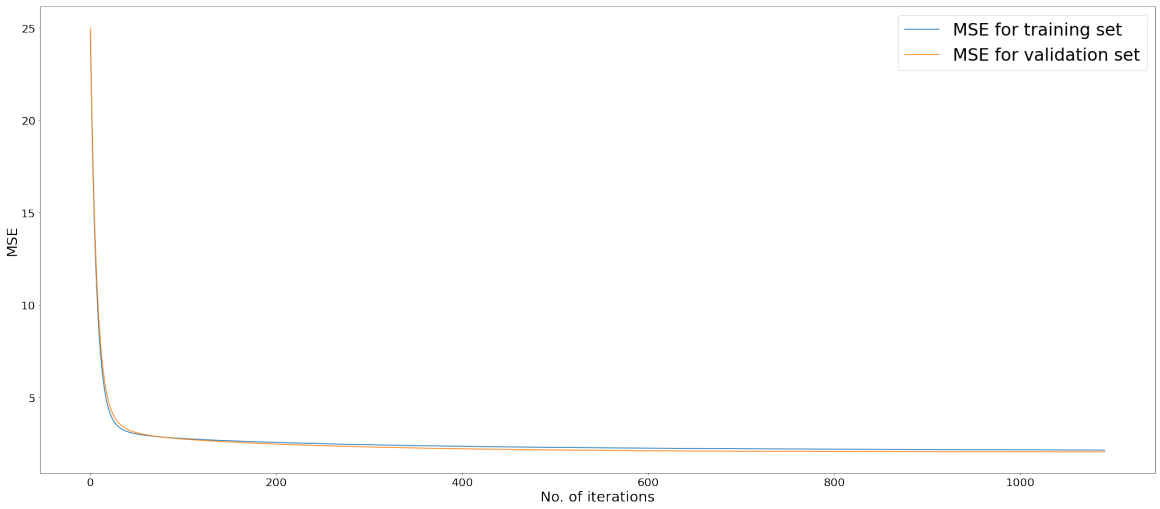

Figure 22: Plot of Mean squared error vs No. of iterations for $\eta = 0.01$ over total iterations

- Learning rate = 0.001
  MSE for training set: 2.15115
  MAE for training set: 1.17442
  MSE for validation set: 2.04105
  MAE for validation set: 1.18473

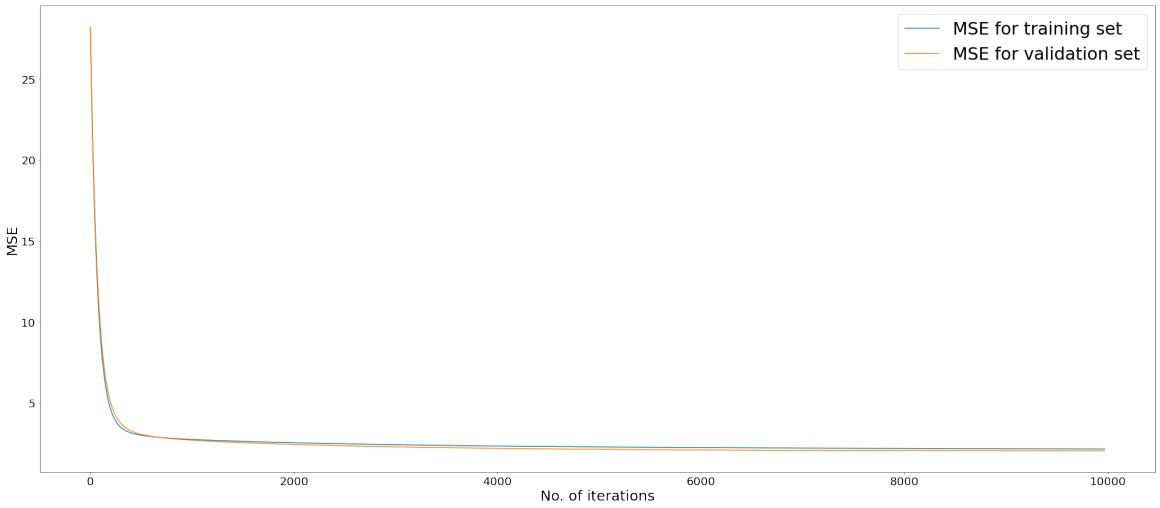

Figure 23: Plot of Mean squared error vs No. of iterations for $\eta = 0.001$ over total iterations

Visualization for task 3.5: Classification problem For the classification problem, the observation on different cases is reported below:

- Learning rate = 0.01 and reltol = 1e-4
  Accuracy for training set: 0.993
  Accuracy for validation set: 0.427

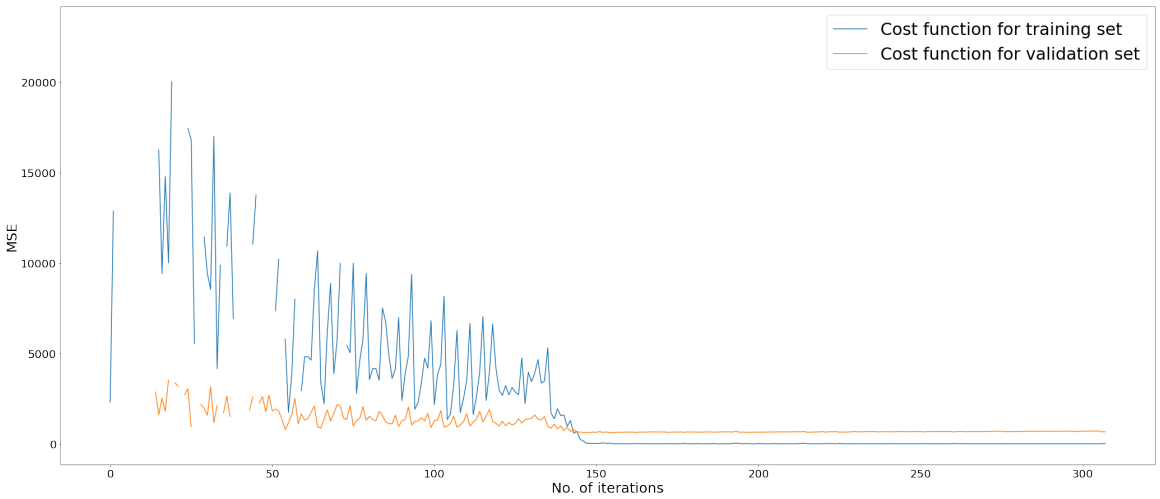

Figure 24: Plot of Cost Function vs No. of iterations for $\eta = 0.01$ over total iterations

- Learning rate = 0.005 and reltol = 1e-4
  Accuracy for training set: 0.99
  Accuracy for validation set: 0.428

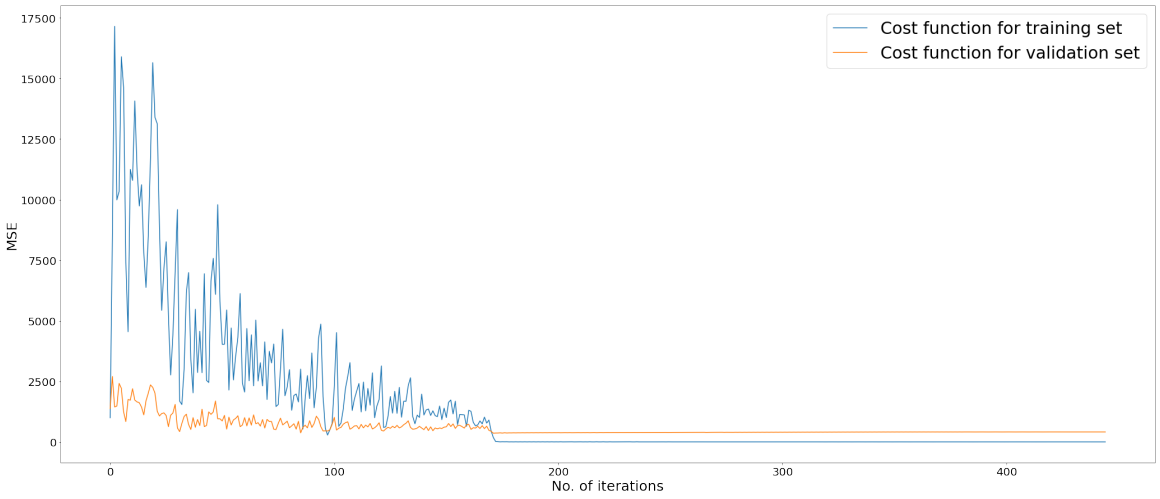

Figure 25: Plot of Cost Function vs No. of iterations for $\eta = 0.005$ over total iterations

- Learning rate = 0.005 and reltol = 1e-6
  Accuracy for training set: 0.993
  Accuracy for validation set: 0.476

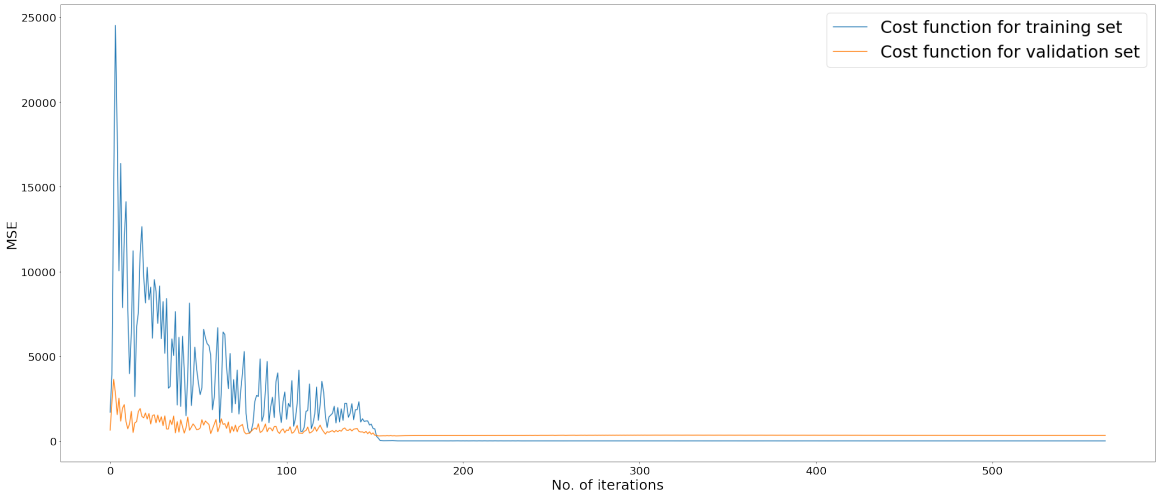

Figure 26: Plot of Cost Function vs No. of iterations for $\eta = 0.005$ over total iterations

- Learning rate = 0.001 and reltol = 1e-4
  Accuracy for training set: 0.993
  Accuracy for validation set: 0.428

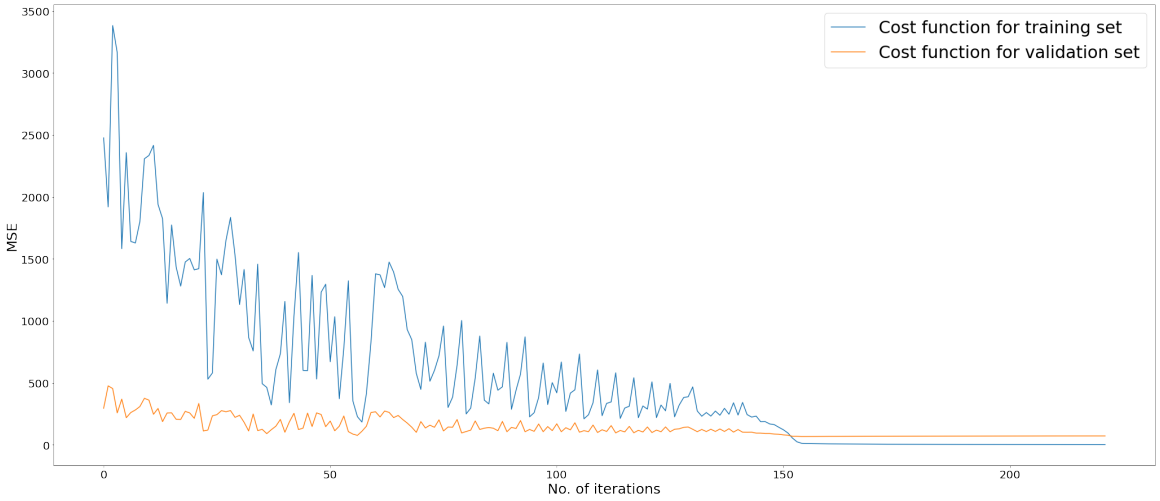

Figure 27: Plot of Cost Function vs No. of iterations for $\eta = 0.001$ over total iterations

- Learning rate = 0.001 and reltol = 1e-6
  Accuracy for training set: 0.993
  Accuracy for validation set: 0.476

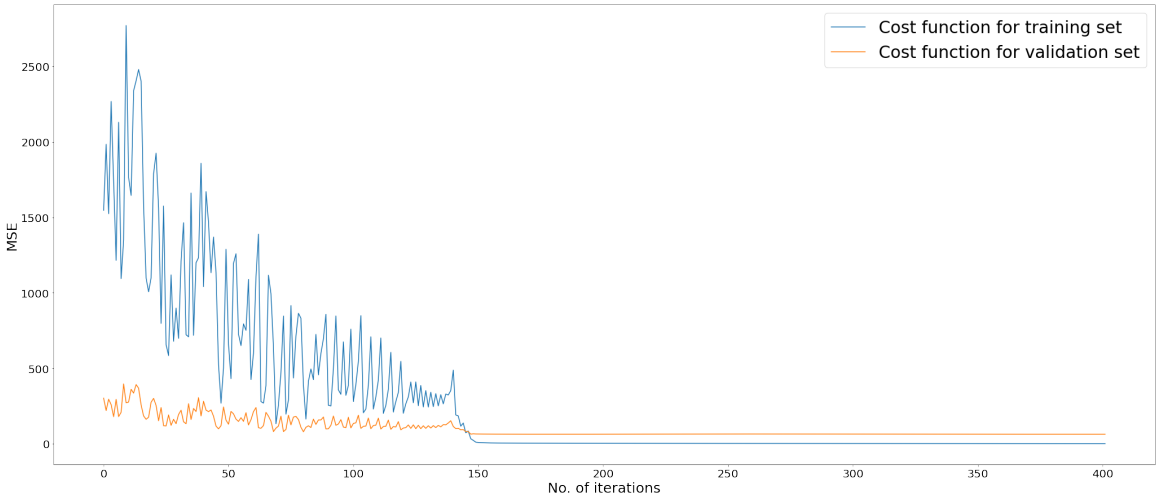

Figure 28: Plot of Cost Function vs No. of iterations for $\eta = 0.001$ over total iterations

- Learning rate = 0.0005 and reltol = 1e-5
  Accuracy for training set: 0.993
  Accuracy for validation set: 0.476

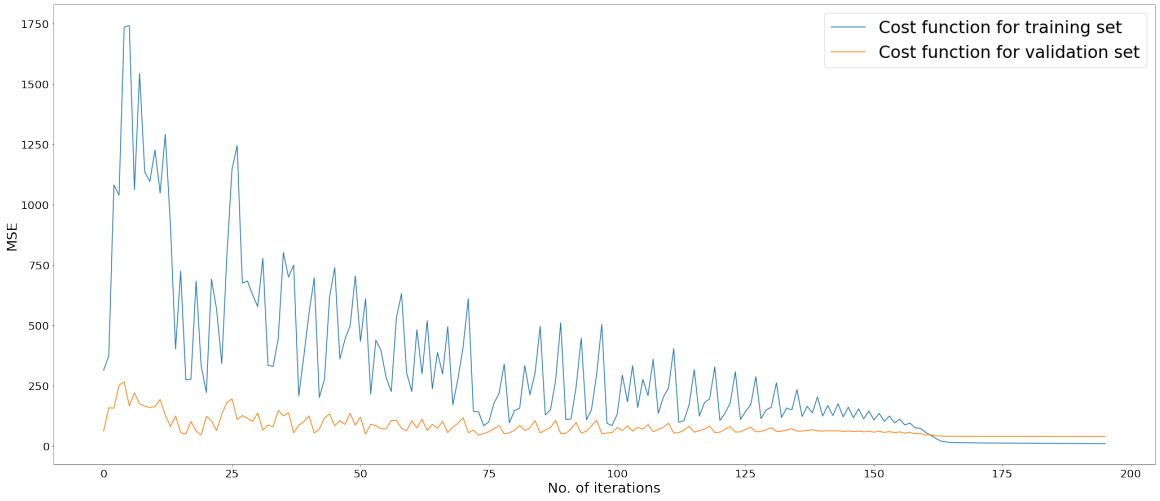

Figure 29: Plot of Cost Function vs No. of iterations for $\eta = 0.0005$ over total iterations

- Learning rate = 0.0001 and reltol = 1e-5
  Accuracy for training set: 0.861
  Accuracy for validation set: 0.524

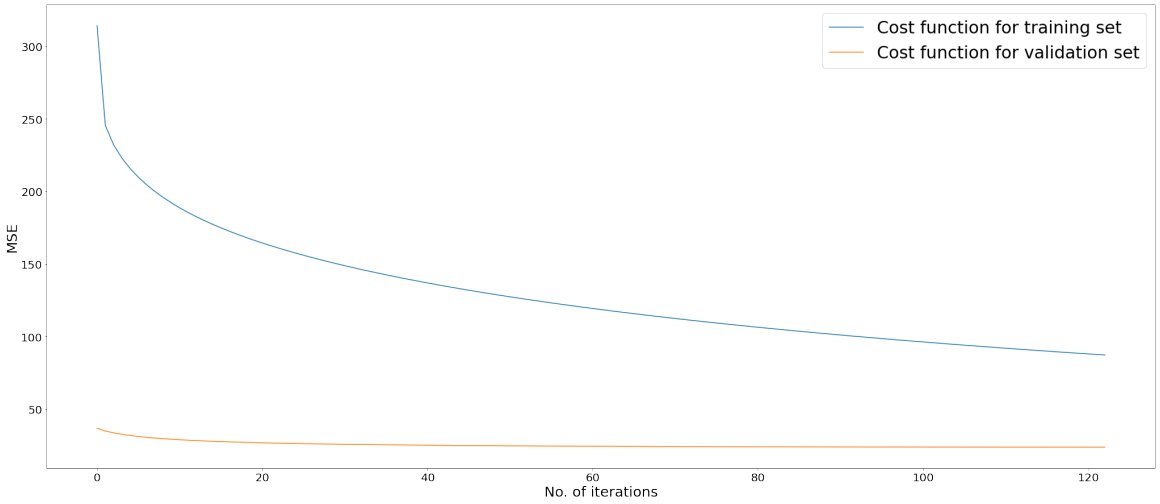

Figure 30: Plot of Cost Function vs No. of iterations for $\eta = 0.0001$ over total iterations

The initial learning rates = 0.01, 0.005, 0.001, 0.0005 shows fluctuations in the cost function w.r.t to no. of iteration which imply of higher learning rates but eventually all converges. The accuracy on training set for all these cases are around 0.99 and 0.048 for validation set. If we look for learning rate = 0.0001, then there is no fluctuation as such, however the training set's accuracy has reduced to 0.86 while the validation set's accuracy has improved. Although we can achieved the earlier accuracy by setting hyper parameters $\eta = 0.0001$ and $reltol = 0.2e-5$.

The large difference in accuracy for the classification model may be due to high model complexity, leading to overfitting and decreased accuracy on new, unseen data. The hyper-parameters of the model may not be set optimally, resulting to decreased accuracy, or the distribution of classes in the data may be unbalanced, making it difficult for the model to accurately predict the minority class.

Linear Regression after normalization of features

- Learning rate = 0.1 and reltol = 1e-5
  MSE for training set: 2.45466e+234
  MAE for training set: 1.27241e+117
  MSE for validation set: 2.42036e+234
  MAE for validation set: 1.19974e+117

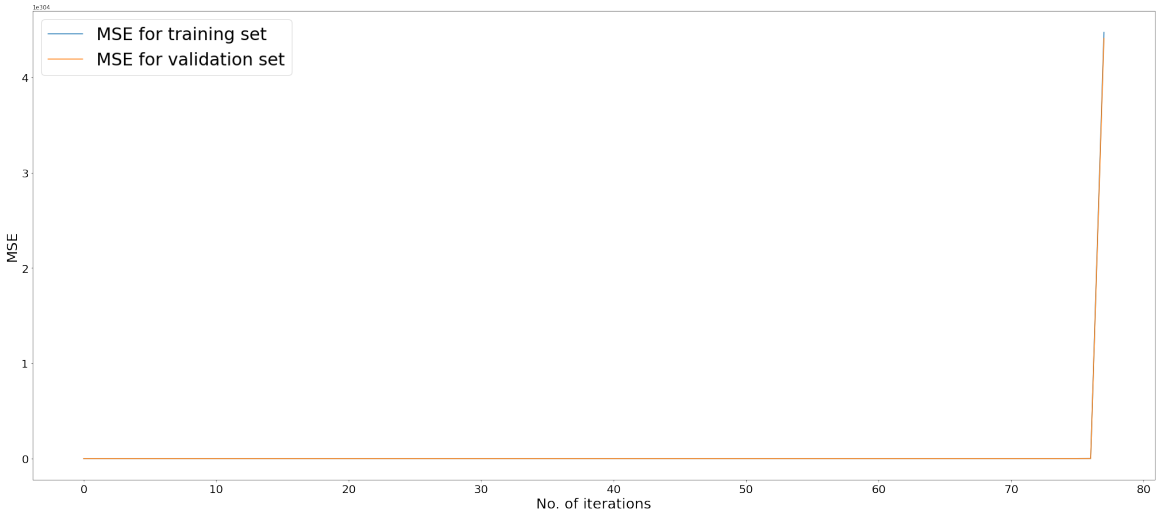

Figure 31: Plot of Cost Function vs No. of iterations for $\eta = 0.1$ over total iterations

- Learning rate = 0.01 and reltol = 1e-5
  MSE for training set: 2.07105e+181
  MAE for training set: 3.69597e+90
  MSE for validation set: 2.04217e+181
  MAE for validation set: 3.48488e+90

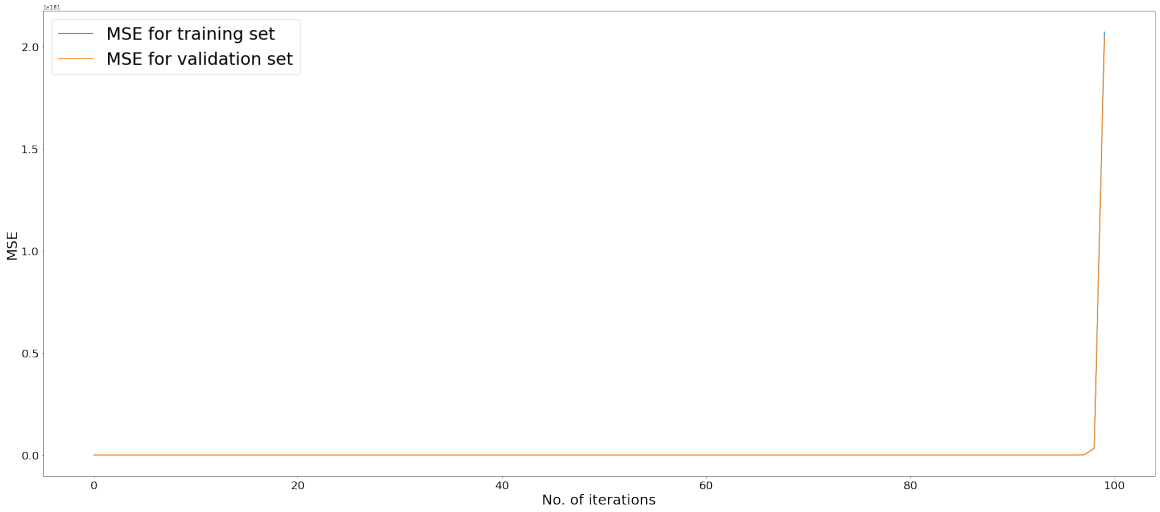

Figure 32: Plot of Cost Function vs No. of iterations for $\eta = 0.01$ over total iterations

- Learning rate = 0.001 and reltol = 1e-5
  MSE for training set: 2.53827e-05
  MAE for training set: 0.00497
  MSE for validation set: 0.90663
  MAE for validation set: 0.73172

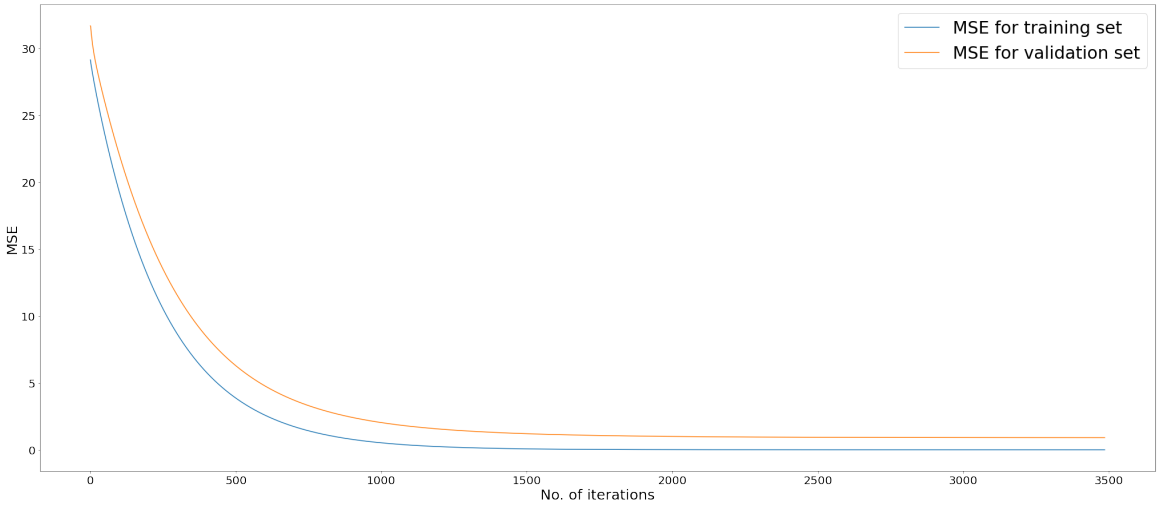

Figure 33: Plot of Cost Function vs No. of iterations for $\eta = 0.001$ over total iterations

- Learning rate = 0.001 and reltol = 1e-6
  MSE for training set: 2.53e-07
  MAE for training set: 0.00048
  MSE for validation set: 0.90252
  MAE for validation set: 0.72974

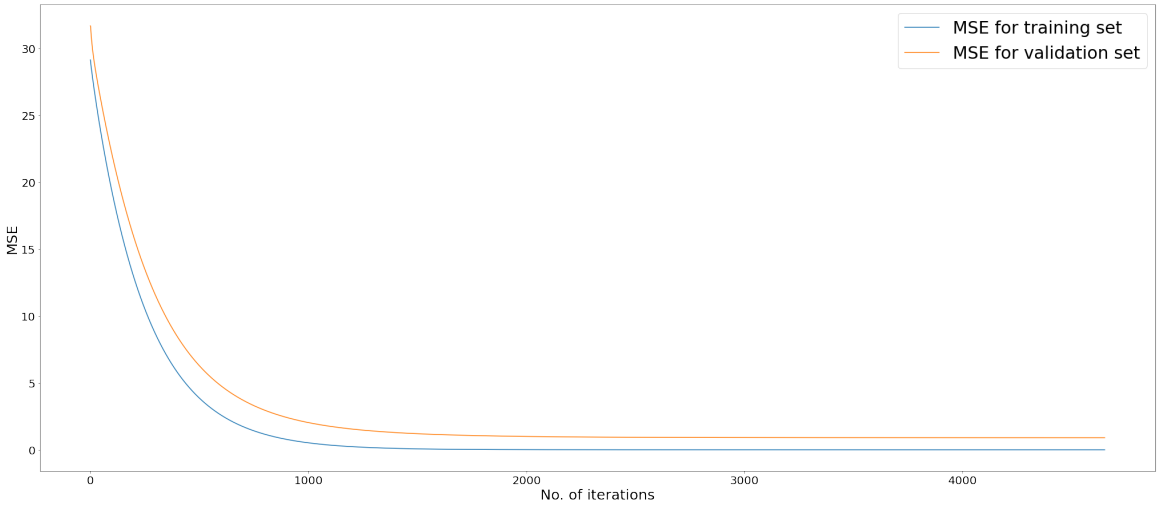

Figure 34: Plot of Cost Function vs No. of iterations for $\eta = 0.001$ over total iterations

Divergence is observed for learning rates of 0.1 and 0.01 as it was in Task 3.1 when the feature was not normalized. For learning rate = 0.001, we get a better convergence, and our model fits well, as the MSE for the training set is much lower than the MSE obtained for Task 3.1. However, MSE for the validation set is slightly higher, which may be due to overfitting of the data, where the model becomes too closely tied to the training data and performs poorly on new, unseen data. The other possibility is that normalising has amplified the impact of outliers.

MSE loss for four sizes of data

The learning rate and reltol is fixed for all cases, i.e. $\eta = 0.001$ and $reltol = 1e-5$

- One-fourth of training data used
  MSE for training set: 2.28564e-05
  MAE for training set: 0.00331
  MSE for validation set: 1.07359
  MAE for validation set: 0.86821

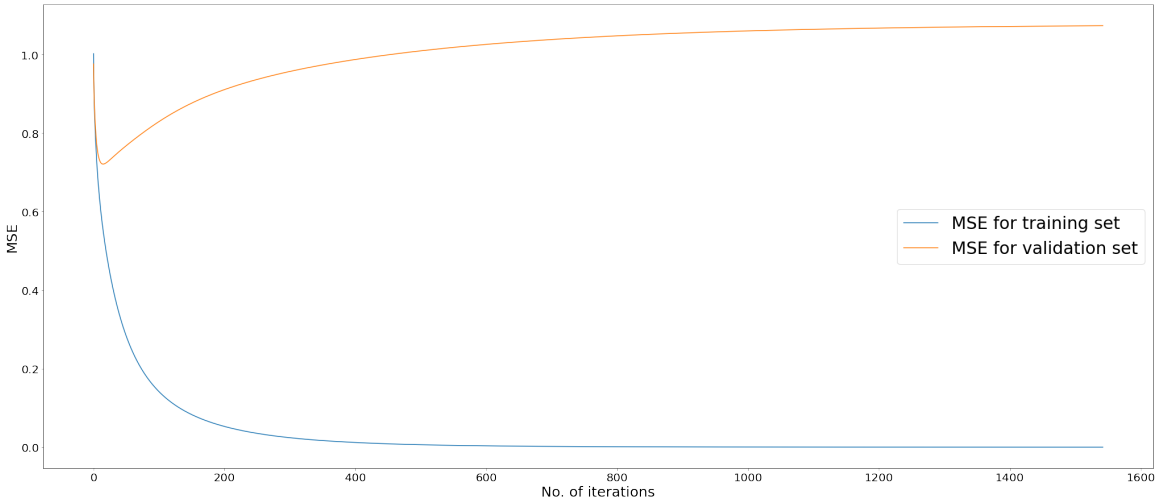

Figure 35: Plot of Cost Function vs No. of iterations

- One-half of training data used
  MSE for training set: 0.28412
  MAE for training set: 0.42274
  MSE for validation set: 0.84474
  MAE for validation set: 0.70560

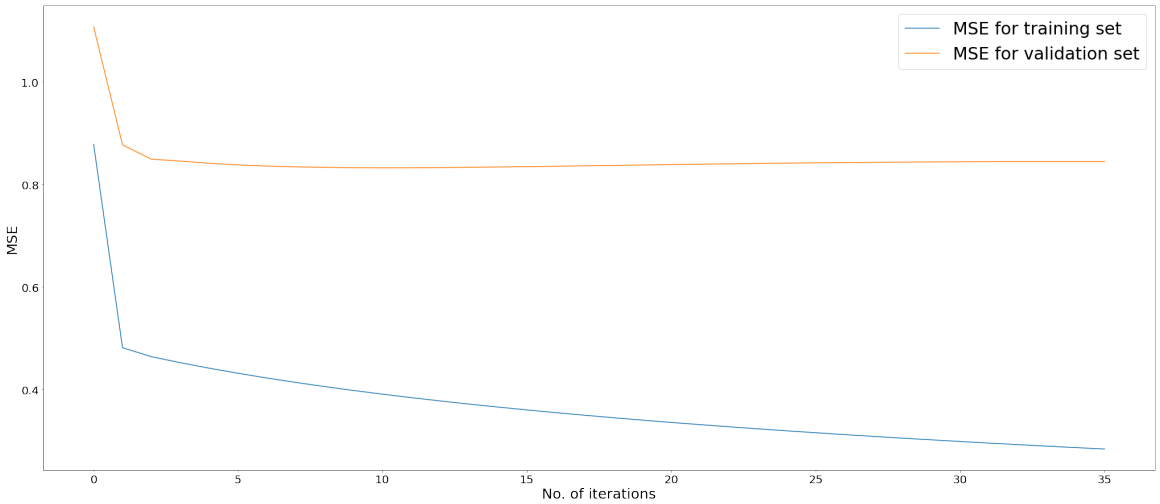

Figure 36: Plot of Cost Function vs No. of iterations

- Three-fourths of training data used

MSE for training set: 0.39684
MAE for training set: 0.49003
MSE for validation set: 0.65104
MAE for validation set: 0.63583

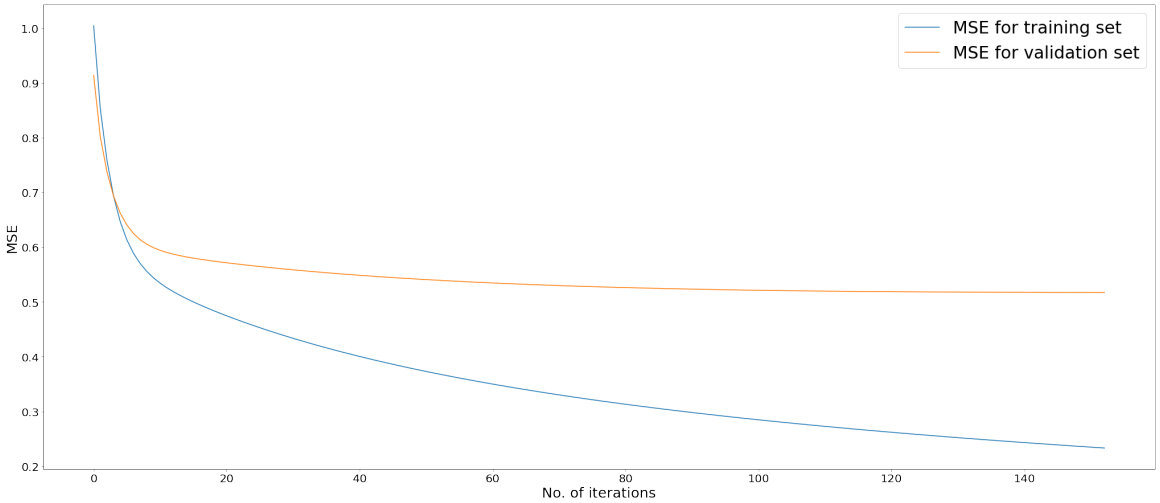

Figure 37: Plot of Cost Function vs No. of iterations

- Complete set of training data used
  MSE for training set: 0.27565
  MAE for training set: 0.40895
  MSE for validation set: 0.53426
  MAE for validation set: 0.57655

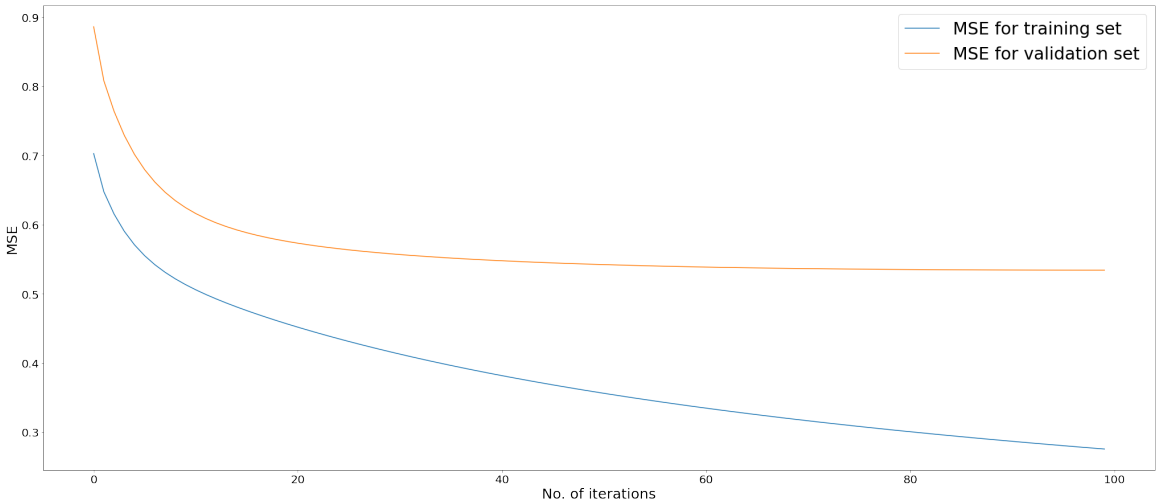

Figure 38: Plot of Cost Function vs No. of iterations

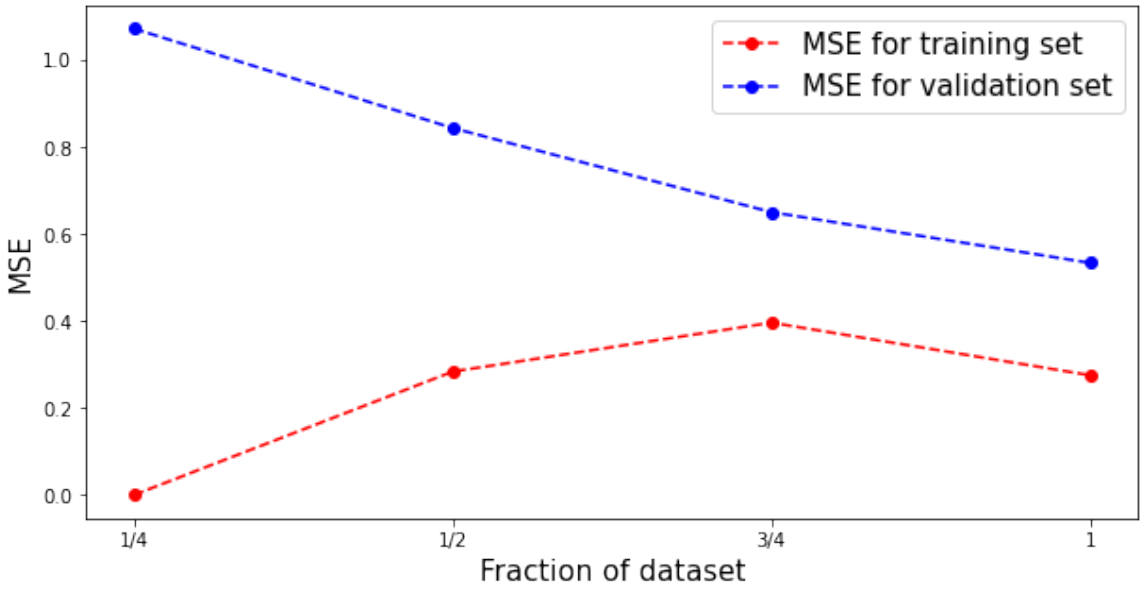

Figure 39: Plot of Cost Function vs No. of iterations

In the case of training with a quarter of the data, the low MSE for the training set and the high MSE for the validation set indicate that our model is overfit due to lower number of data points. As the size of the data set increases, MSE loss of validation set decreases as our model get introduced to more data points, thus our hypothesis function modeled well, and also the difference between the MSE of the training set and the validation set decreases, as predicted by the generalisation bound.
Note: Data is selected in randomly, thus it may cause different MSE and MAE value on every new execution.
Note: Important Observation: For complete set of training data, if we shuffle the data randomly then we get a good result and this can be observed from the change in MSE and MAE loss in two results.

Divided data in two half's, training model with one and predicting with other
For linear regression, $\eta = 0.001$ and $reltol = 1e-5$ For ridge regression, $\eta = 0.00001$, $\lambda = 100$ and $reltol = 1e-5$

- Training with first half of data and 2nd half for testing
  Reported MSE and MAE loss on training and validation data for linear regression -
  MSE for training set: 0.31235
  MAE for training set: 0.45374
  MSE for validation set: 0.55294
  MAE for validation set: 0.57529
  MAE loss obtained on the prediction from second half set is 0.678579.

  Reported MSE and MAE loss on training and validation data for ridge regression -
  MSE for training set: 0.12030
  MAE for training set: 0.28253

MSE for validation set: 0.63497
MAE for validation set: 0.60052
MAE loss obtained on the prediction from second half set is 0.67793.

The MAE loss in both regression obtained is nearly equal indicating that regularization term in model does not have significant impact on training and also linear regression model is not overtrained. We can't draw a solid conclusion from this as size and quality of training data also matters. MAE for ridge may also be higher if the variance in data is low.

- Training with 2nd half of data and first half for testing
  Reported MSE and MAE loss on training and validation data for linear regression -
  MSE for training set: 0.18373
  MAE for training set: 0.35044
  MSE for validation set: 0.55080
  MAE for validation set: 0.58837
  MAE loss obtained on the prediction from second half set is 0.66275.

  Reported MSE and MAE loss on training and validation data for ridge regression -
  MSE for training set: 0.21364
  MAE for training set: 0.37188
  MSE for validation set: 0.56829
  MAE for validation set: 0.58278
  MAE loss obtained on the prediction from second half set is 0.64941.

.

MSE loss against the correct score values

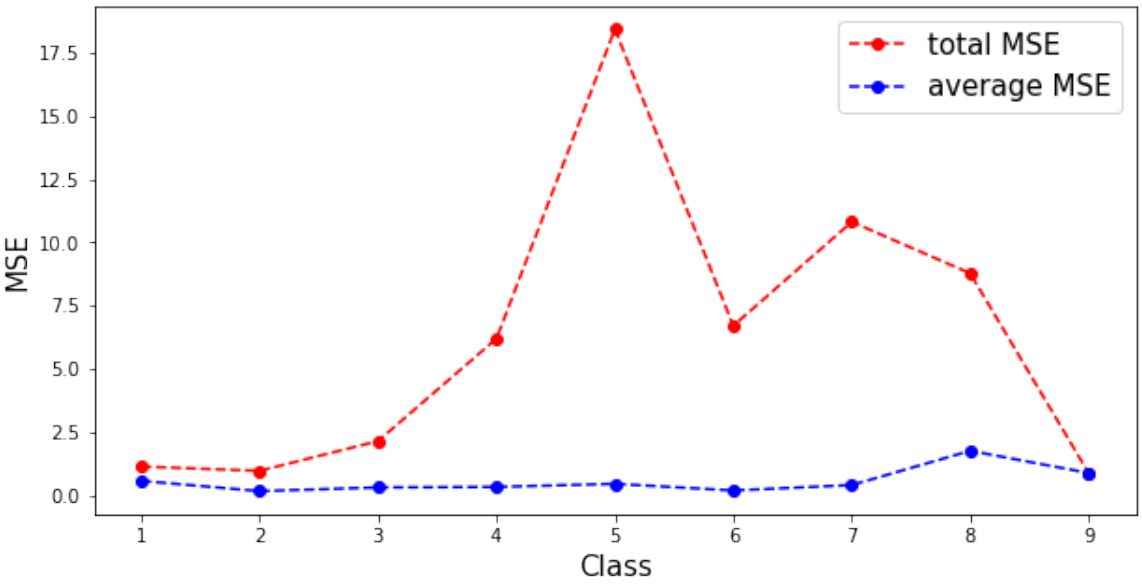

Figure 40: Plot for training data

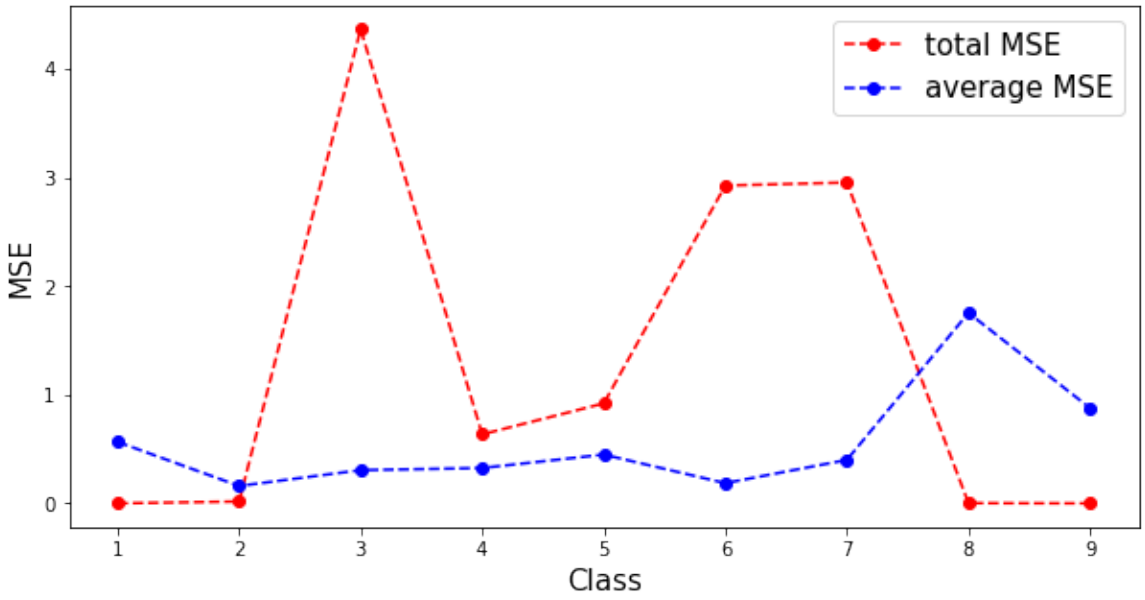

Figure 41: Plot for validation data

The graph of the total MSE loss vs the right score values might provide important insight into a regression model's performance. In the plot of MSE vs Score for training data, total MSE loss is high for class 5 while it is low for other class but the average is nearly same for all the class as the number of data points for class 5 is relatively higher. While in the case, of validation data's plot MSE is slightly higher for class 3. Also, if the average MSE loss is high, it indicates that the model is not producing accurate predictions. If the average MSE loss is minimal, it indicates that the model is producing generally accurate predictions.

SelectKBest method gives better performance, there this method is selected to obtain the MSE for different ranges of best features.

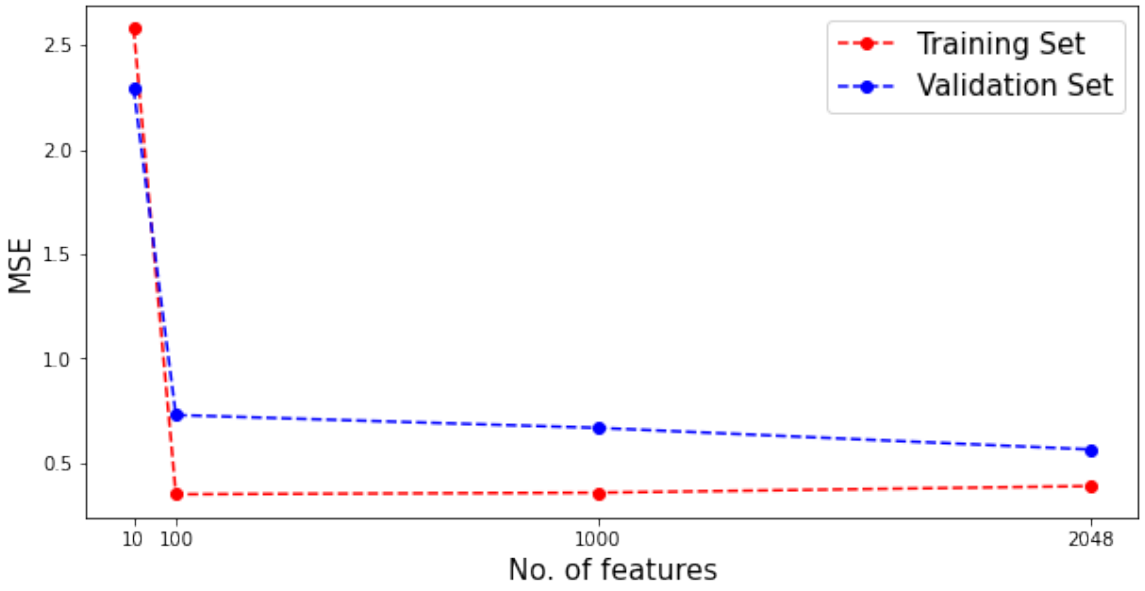

Figure 42: Plot for MSE corresponding to Best selected features

3.7 Generalisation Analysis

The generalization bound on Linear Regression is found as $E_{out} = E_{in} + O\left(\sqrt{\frac{d}{N}}\right)$, where, d is number of dimension of data and N is number of data points.

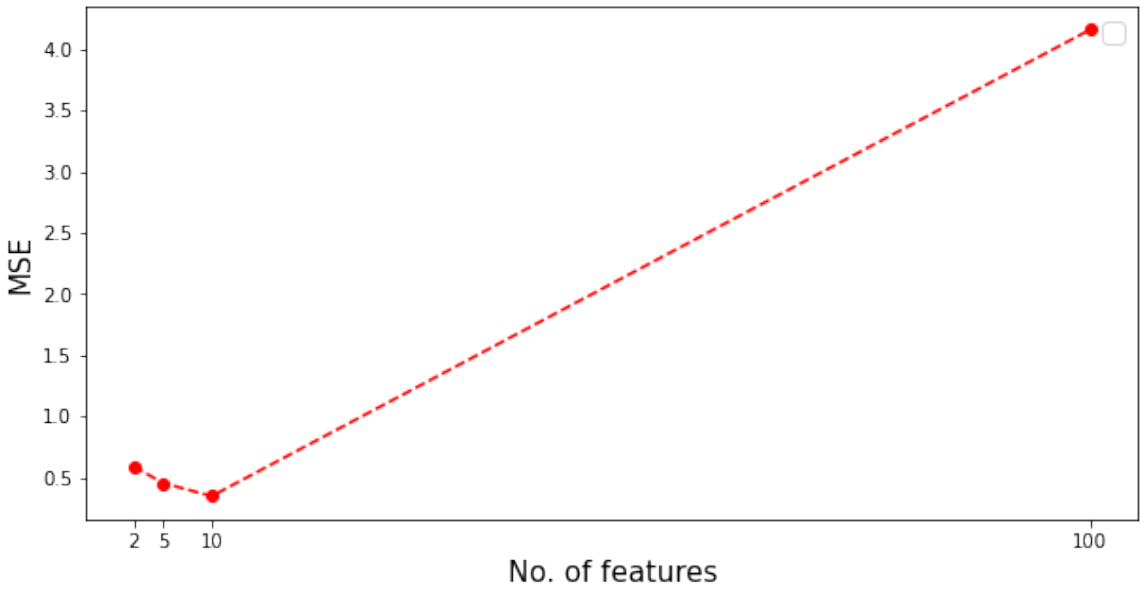

Figure 43: Plot for $E_{out} - E_{in}$ vs d

In the graph we observe, that MSE first decrease upto number of features (d) = 10 and then it increases, while theoretically $E_{out} - E_{in} \propto \sqrt{d}$ and since N is constant we expect it to increase. However, adding more features to a model improves the model's capacity to fit well over the training data and minimises bias. But, beyond a certain point, adding additional features leads to overfitting, when the model gets overly complicated and begins to match the noise in the training data, as shown in the plot.