K.Surya Prakash: *EE18BTECH11026*
Tatipelly Vamshi: *ES18BTECH11021*

Question : 04

4.1 Explain ML estimation for poisson regression

Mathematical expressions for poisson regression :

Question - 4

Poisson regression

Q1) : **ML estimation**

$$L = \prod_{i=}^{n} P(y_i/\theta_i) \rightarrow \text{Likelihood.}$$

$\theta_i \rightarrow$ parameters for i^{th} data point

Here, we try to fit data to a poisson distribution, with different parameters.

$$\therefore P(y_i/\theta_i) = P(y_i/\lambda_i) = \frac{e^{-\lambda_i}(\lambda_i)^{y_i}}{(y_i)!}$$

Here 'λ' $\rightarrow$ parameter for poisson

But $\lambda_i \rightarrow$ scalar for i^{th} data point.

$$X_i \xrightarrow{\beta \rightarrow LT} \lambda_i$$

X_i: $d \times 1$; λ_i: 1×1

} $d \rightarrow$ dimensions / features of a data point.

$\Rightarrow$ We perform a linear transformation & estimate λ_i for a given X_i

$$\lambda_i = \exp(X_i \beta)$$

$\beta \rightarrow D \times 1$ column vector

$X_i \rightarrow 1 \times D$ row vector

Inorder to fit data, we need to find parameter λ_i of distribution

→ MLE :-

→ We use MLE to find λ_i (indirectly we need to find β)

$$\beta \rightarrow ?$$

$$L = \prod_{i=1}^{N} \frac{e^{-\lambda_i} (\lambda_i)^{y_i}}{(y_i)!}, \text{ where } \lambda_i = \exp(X_i \beta)$$

→ Goal is to maximise 'L'

→ log-likelihood (it does not change the loss)

$$\log_e(L(\lambda)) = \sum_{i=1}^{N} -\lambda_i + y_i \log_e(\lambda_i) - \log(y_i)!$$

∴ Need to maximise $\log_e(L(\lambda_i))$

$$\Rightarrow \log(L(\beta)) = \sum_{i=1}^{N} -\lambda_i + y_i \log(e^{X_i \beta}) - \log(y_i))$$

$$= \sum e^{-(\bar{x}_i \beta)} + y_i \bar{x}_i \beta - \log(y_i))!$$

* Let $J(\beta)$ be loss function,

Aim is to minimise J, to find 'β'

$$J(\beta) = -\log(L(\beta))$$

⇒ Poisson regression loss function is

$$J(\beta) = \sum_{i=1}^{N} e^{(x_i\beta)} - y_i(\bar{x}_i\beta) - \log(y_i)!$$

→ Find β s.t. $J(\beta)$ is min.

$$\Rightarrow \frac{\partial J(\beta)}{\partial \beta_k} = \sum_{i=1}^{N} e^{(x_i\beta)} \cdot x_i^k - y_i(x_i^k)) \quad : k \to k\text{th element in } \bar{\beta}$$

$$= \sum_{i=1}^{N} x_i^k (e^{(x_i\beta)} - y_i)$$

$$\frac{\partial J(\beta)}{\partial \beta} = X^T \cdot (e^{(X\beta)} - y_i)$$

$$X = \begin{bmatrix} \bar{x}_1 \\ \vdots \\ \bar{x}_N \end{bmatrix}_{N \times D} \Rightarrow \bar{x}_1 = [x_1^1 \ldots x_1^D]_{D \times 1}$$

$$\beta = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_D \end{bmatrix}_{D \times 1}, \quad y = \begin{bmatrix} y_1 \\ \vdots \\ y_n \end{bmatrix}_{N \times 1}$$

For minimising $J(\beta)$.

$$\Rightarrow \frac{\partial J(\beta)}{\partial \beta} = 0 \Rightarrow X^T(e^{(X\beta)} - y) = 0$$

$$\Downarrow$$

Need to solve a non-linear eqn

→ Hence, we use gradient descent to solve it. Since no close form expression exists.

⇒ Gradient Descent:

$$\beta_{n+1} = \beta_n - \eta \left(\frac{\partial J}{\partial \beta} \right)$$

$$\beta_{n+1} = \beta_n - \eta \left(X^T (e^{X\beta_n} - Y) \right)$$

$\eta \rightarrow$ learning rate

→ When it converges

$$\lambda_i = \exp(x_i \beta)$$

$$\hat{y}_i = \hat{\lambda}_i$$

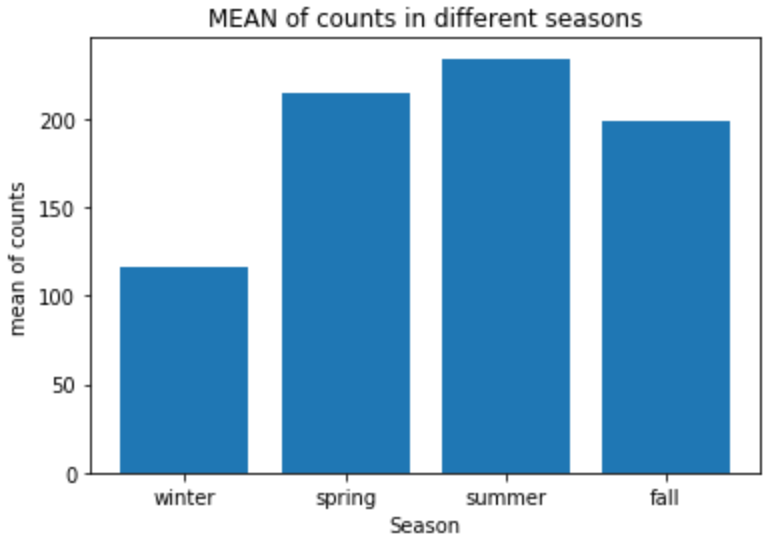
⇒ Prediction:

$$\boxed{\hat{y}_i = \exp(x_i \beta)}$$

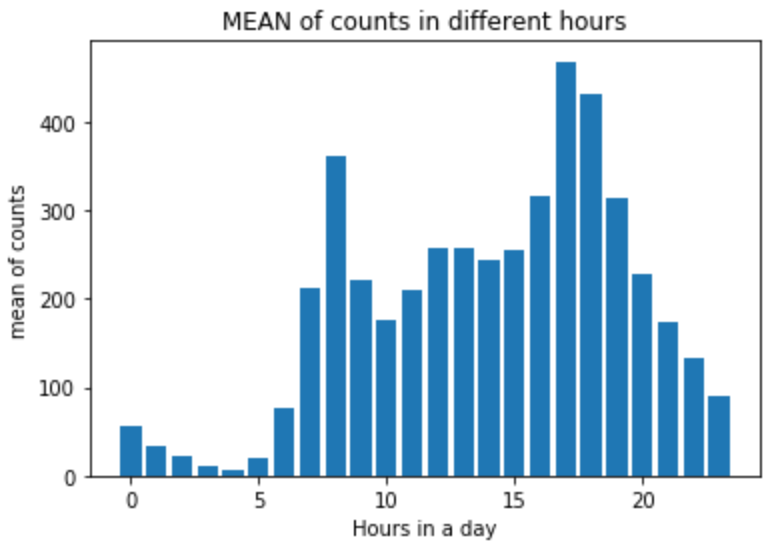
4.2 Showing statistics of data

Showed the mean of count over 4 features of data :

1. **Over different seasons** :
   Observations : We can see that mean is high in summer and spring , and least in winter.
   This might be a feature for learning, but since this features has 4 values and they are close wrt mean value.

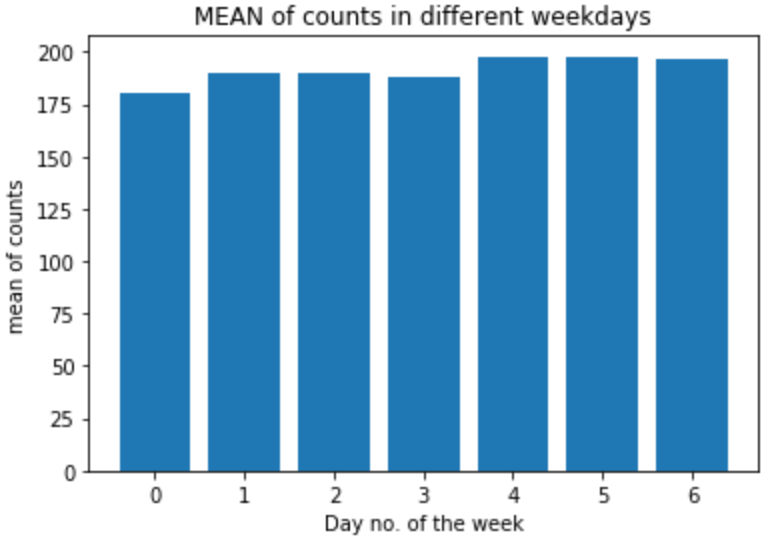


2. **Over hours in a day** :
   This result is intuitive ,since one can expect more traffic and drives during the evening hours (mostly working hours) , and very less during midnight hours. This follows a normal distribution , with mean at 18 hr ie; “6pm’.

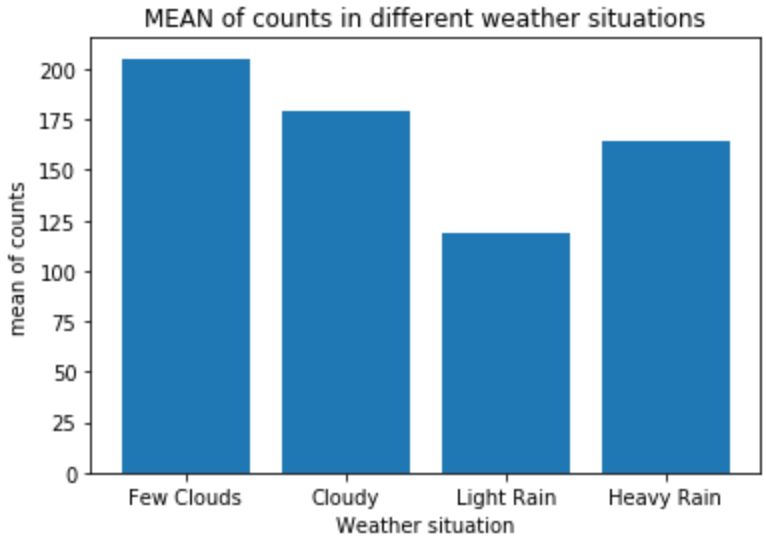


3. **<u>Over 7 days in a week</u>** :

   This feature might not be a good feature for regression, since the mean counts are spread across evenly. Thus count data , is less likely to be dependent on weekdays. Hence this feature is discarded and not used for training.



4. **<u>Over weather situation</u>** : The explanation for season data is applicable here as well. This is considered as a feature.



4.3: Plotting count vs feature

<u>NOTE</u> : We can see that, we cannot draw a simple plot to understand the feature characteristics. Because for a feature like 'hr', data having the same 'hr' feature, is observed to have different count values. Hence we opted for plotting a "BOX plot"

What is a box plot ??

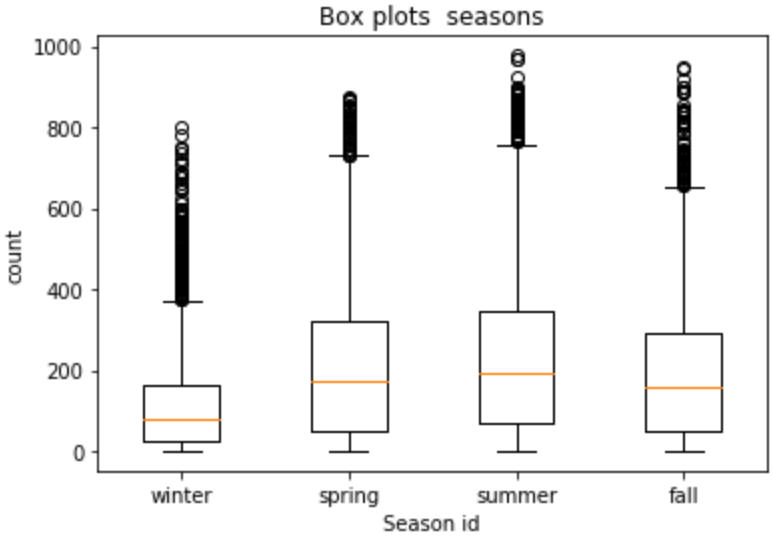
A Box plot describes the main statistics of that particular value of the feature. It tells about the median, the IQR and also the outliers. This will help us better understand how the features are spread across .

<u>Things to be noticed in a box plot</u> :

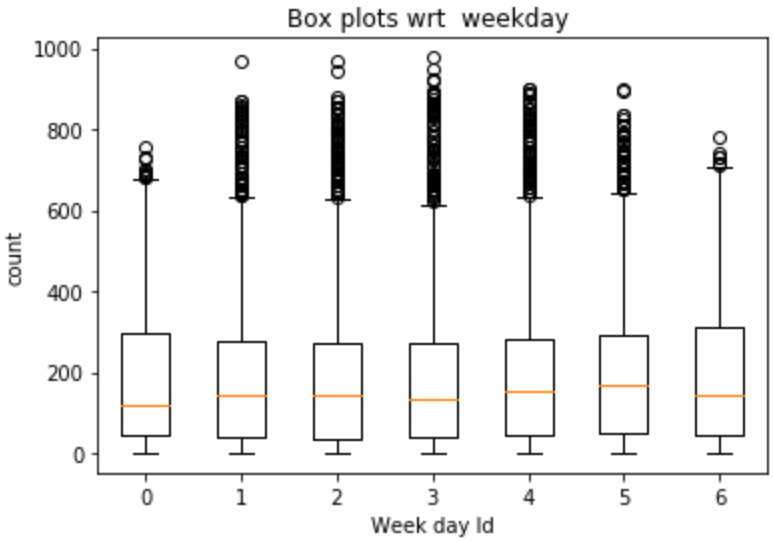
The box length --> IQR of the value of the feature : Tells about how is the count value spread for that value of feature

- Yellow line → Median
- The ends of the line → maximum : Q3 + 1.5*IQR
  → minimum : Q1 + 1.5*IQR
- Dots are the expected outliers , which lie very far from the **median**

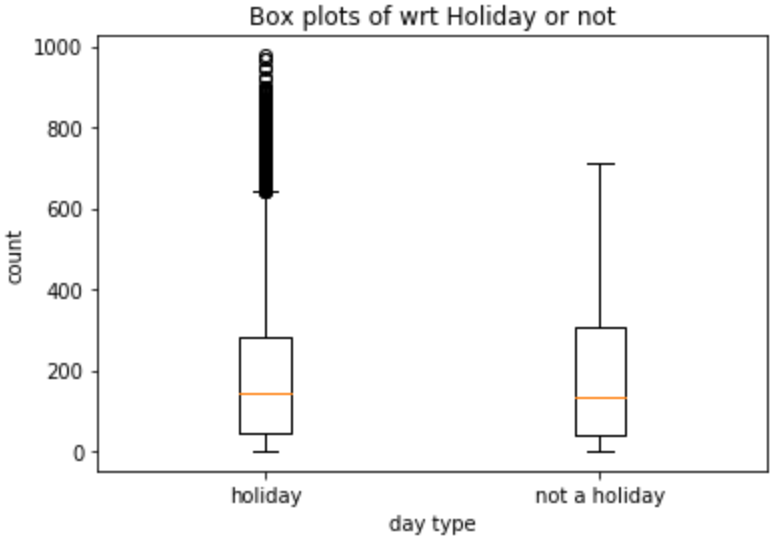
Feature 1: Seasons:



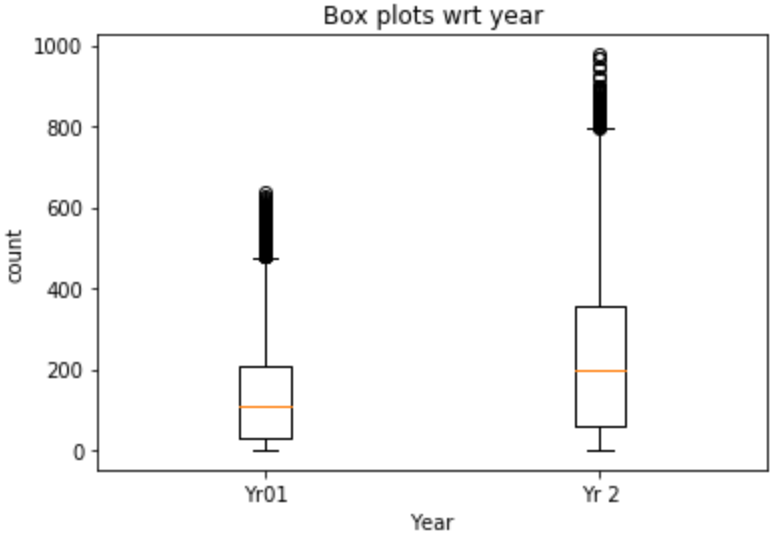
Feature 2: weekdays



Feature 3: Working day or a holiday

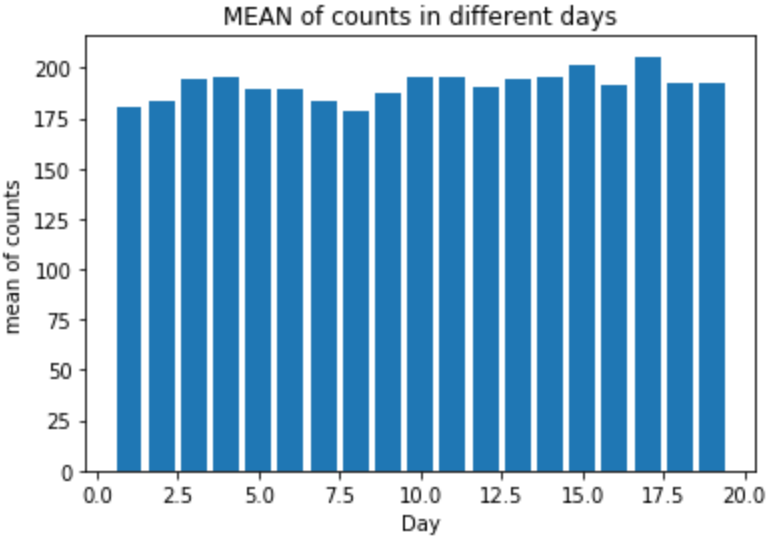


Feature 4: Year



Feature 5: Day

We extracted the day value from the date.
The training set consists of the first 19 days of the month.



Observations :

1. Through the box plots of different features, one can observe that there are a lot of outliers(tiny circles at the top of the plot) **.**
2. This means that the data has a high variance , which may not be a good sign for a prediction model. That is the reason we are not able to obtain firm predictions, resulting in high RMSE .

4.4 Building a Poisson regularisation model using gradient descent:

The algorithm is derived and has been explained in Q4.1 (for all 3 types of regularisation).

The aim of gradient descent is to find the best “beta” transformation matrix, such that

$$Y_{pred} = exp(X.beta)$$

Data description :

Total data points : 17379
Training data : 10886 (Later divided into training and validation data of 0.8 : 0.2 ratio)
Test data : 6493

Features considered : [bias , season , month , hr , working day , wind speed , temp]

Bias : Created feature consisting of 1.

Data Normalisation : We normalized the data before training, the procedure was that each column(feature) has 0 zero mean , and unit variance .

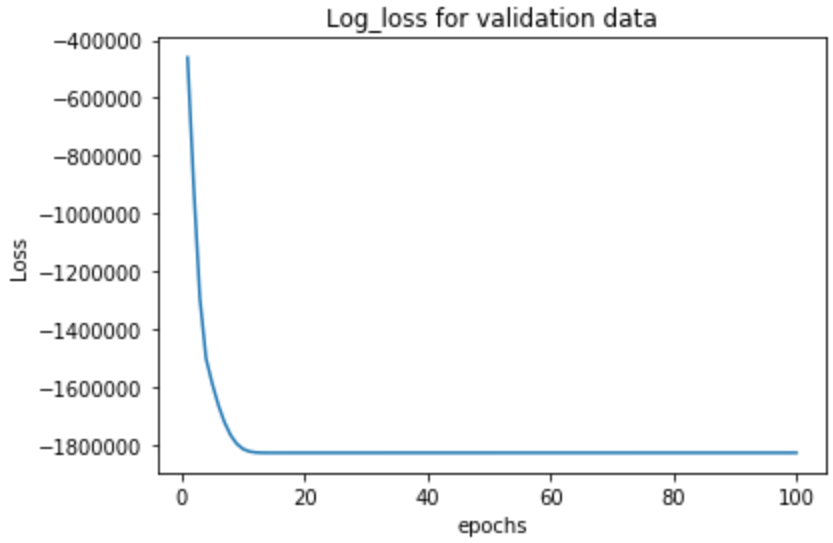
Learning rate : After experimenting with different values , we chose the learning rate to be $5*10^{-7}$.
Note : The mentioned learning rate seems to be too small. This is because we did not scale it up with the size of the training batch while computing gradients of the batch.

Results for no-regularisation :
Converged beta vector for considered features :

$$\begin{bmatrix} bias \\ season \\ mnth \\ hr \\ workingday \\ weathersit \\ windspeed \\ temp \end{bmatrix} := \begin{bmatrix} 5.114 \\ -0.073 \\ 0.207 \\ 0.341 \\ 0.003 \\ -0.107 \\ 0.081 \\ 0.314 \end{bmatrix}$$

Log-loss for validation data :

Log-loss plot over validation data : (NOTE : It is observed that the range of loss is in the order of 10^5, this is because We displayed the log loss , without adding the y! term in it. Also We did not scale it. The important aspect is that the decay is happening , which implies that the weights are converging to a point and after 20 epochs the weights converge.



<u>Error calculations</u> :

➔ *RMSE for validation data* **: 151**
➔ *RMS for testing data* **: 158**

Regularisation

- ❑ **Mathematical formulation**
- ❑ **For L1 norm regularisation**
- ❑ **And L2 norm regularisation**

4.4 Regularisation:

⇒ As expla derived in 4.1:

cost function (No regularisation).

$$J(\beta) = \sum_{i=1}^{N} e^{(x_i\beta)} - y_i(\bar{x}_i\bar{\beta}) - \log(y_i!)$$

$$\Rightarrow \beta_{n+1} = \beta_n - \eta\left(\frac{\partial J}{\partial \beta}\right)$$

For L_1 regularisation

* hyperparameter: λ

M → no. of features

$$J(\beta) = \sum_{i=1}^{N} e^{x_i\beta} - y_i(\bar{x}_i\bar{\beta}) - \log(y_i)! + \lambda \sum_{j=1}^{M} |\beta_j|$$

$$\Rightarrow$$

$$\therefore \left.\frac{\partial J}{\partial \beta}\right| = \left.\frac{\partial J}{\partial \beta}\right|_{\text{non regularised}} + \lambda \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix}_{M\times 1}$$

L_1-regularised

$$\Rightarrow$$ Hence, for L_1-regularisation

$$\beta_{n+1} = \beta_n - \eta\left(\left.\frac{\partial J}{\partial \beta}\right|_{\text{non-reg}}\right) - \lambda \begin{bmatrix} \vdots \\ \vdots \\ \vdots \end{bmatrix}_{M\times 1}$$

Use of L1-regularisation

↳ This tries to shrink the ~~features~~ weights of unimportant features.

⇒ Helping us to know "useful" features.

For L_2 regularisation

hyperparameter : λ

$$\Rightarrow J(\beta) = \sum_{i=1}^{N} e^{x_i\beta} - y_i(\bar{x}_i\bar{\beta}) - \log(y_i)! + \frac{\lambda}{2}\sum_{j=1}^{M} i\beta j|^2$$

$$\Rightarrow \left.\frac{\partial J}{\partial \beta}\right|_{L_2\text{-reg.}} = \left.\frac{\partial J}{\partial \beta}\right|_{\text{non-reg}} + \lambda(\beta)$$

$$\Rightarrow \beta_{n+1} = \beta_n - \eta\left(\left.\frac{\partial J}{\partial \beta}\right|_{\text{non-reg.}}\right) - \lambda'(\beta_n)$$

$$\boxed{\beta_{n+1}' = \beta_n''(1-\lambda) - \eta\left(\frac{\partial J}{\partial \beta}\right)}$$

⇒ Used for avoiding over-fitting.

Results for L1 norm regularisation :

"absolute value of magnitude" of beta (weights) added to loss as a penalty.

We need to find λ (hyper parameter) : This is done by varying the parameter and checking the RMSE for the validation set.

By observation of validation error , λ_{L1} = 0.025 is considered to be a good pick , yielding less rmse .

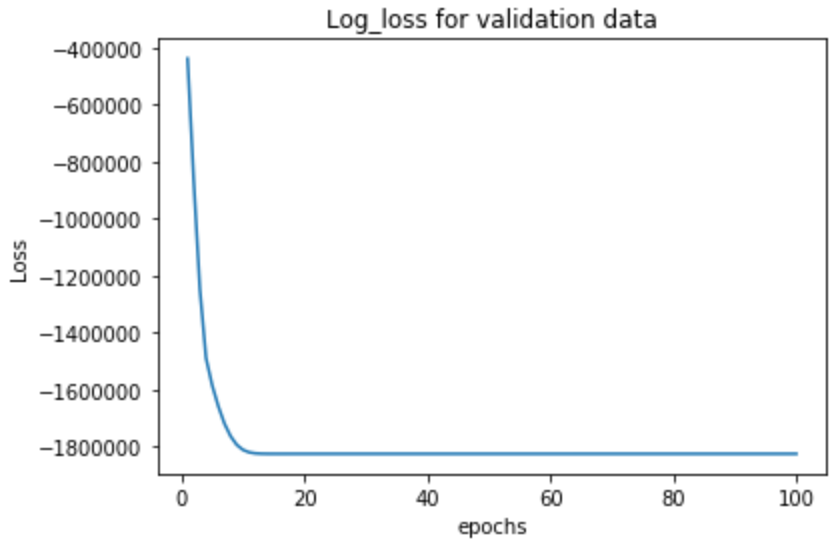
➔ *Learning rate* = $5*10^{-7}$

➔ *Hyper parameter* : λ_{L1} = 0.025

Converged beta vector for considered features :

$$\begin{bmatrix} bias \\ season \\ mnth \\ hr \\ workingday \\ weathersit \\ windspeed \\ temp \end{bmatrix} := \begin{bmatrix} 5.098 \\ -0.076 \\ 0.173 \\ 0.322 \\ -0.026 \\ -0.15 \\ 0.049 \\ 0.296 \end{bmatrix}$$

Log-loss for validation data :



Error calculations :

➔ *RMSE for validation data* **: 151**
➔ *RMS for testing data* **: 157**

Use of L1 norm regularisation :

The L1 regularisation, shrinks the weights of the less important features, thus aids us in choosing important features for prediction.

Results for L2 norm regularisation :

Adds “squared magnitude” of beta (weights) added to loss as a penalty.

Mathematical derivation is done in Q4.1 .

We need to find λ (hyper parameter) : This is done by varying the parameter and checking the RMSE for the validation set.

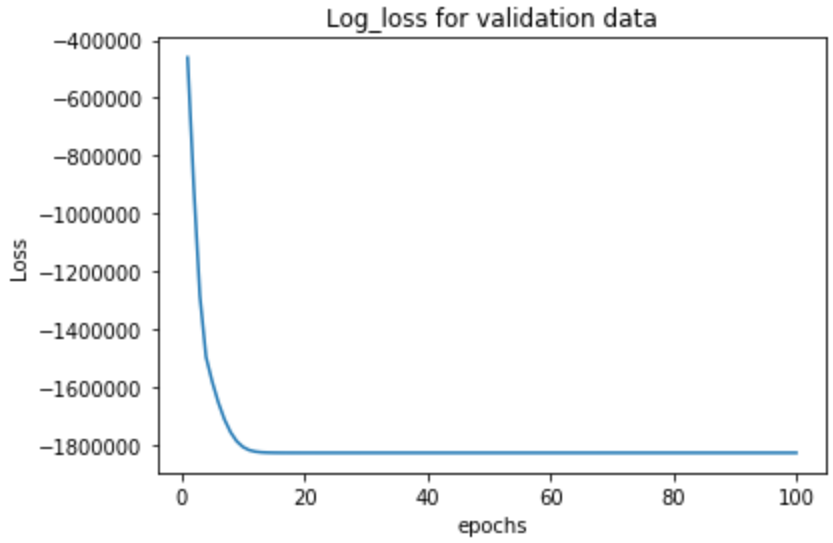
By observation of validation error , λ_{L2} = 0.008 is considered to be a good pick , yielding less rmse .

- ➔ *Learning rate* = $5*10^{-7}$
- ➔ *Hyper parameter* : λ_{L2} = 0.008

Converged beta vector for considered features :

$$\begin{bmatrix} bias \\ season \\ mnth \\ hr \\ workingday \\ weathersit \\ windspeed \\ temp \end{bmatrix} := \begin{bmatrix} 5.051 \\ -0.04 \\ 0.182 \\ 0.358 \\ 0.004 \\ -0.112 \\ 0.085 \\ 0.327 \end{bmatrix}$$

Log-loss for validation data :



Error calculations :

➔ *RMSE for validation data* **: 154**
➔ *RMS for testing data* **: 157**

Use of L2 regularisation :
It helps in avoiding overfitting issues **.**

Observation of regularisation :
We can observe that with L1 and L2 regularisation, the weights are decreased . This reflects the idea of adding the additional penalty(regularisation term) to the loss.

This also helps us overcome the problem of overfitting at the expense of some extra validation error.

4.5 Most Important Features

As explained earlier, the features whose weight values are close to zero are considered unimportant.

So on the basis of values of **converged beta weights**, and the plots of count over different features , these are considered as the most - important features.

1. **Hour** :
   We saw that more bikes are rented during the evening hours
2. **Weather sit** :
   The feature vs count plot also tells us that more bikes are rented during times when it is less cloudy

3. **Temperature** : From the weight vector value
4. **Month** : From the weight vector value.

Conclusion

In this question we designed a poisson regression model, which works better than typical regression models since the target feature is discrete.(Built on the idea of poisson distribution which models discrete Random variables) .

We also observed that there is a high variance of data over the features, which makes the predictions not 100% reliable, leading to high errors.

THE END