Soumya Saha
HW01

1. $$P(X=x) = \left(\frac{1}{1+e^{-\lambda}}\right)^{[x=1]} \left(\frac{1}{1+e^{\lambda}}\right)^{[x=0]}$$

(a) To check if the distribution is properly normalized, we need to verify the sum of all probabilities to be 1.

$$\sum_{x \in [0,1]} P(X=x) = \frac{1}{1+e^{-\lambda}} + \frac{1}{1+e^{\lambda}} = \frac{e^{\lambda}}{1+e^{\lambda}} + \frac{1}{1+e^{\lambda}} = \frac{1+e^{\lambda}}{1+e^{\lambda}} = 1$$

Thus, the distribution is properly normalized.

(b) Dataset contains a one's, b zero's.

The log likelihood function, $l(D,\lambda) = a \log\left(\frac{1}{1+e^{-\lambda}}\right) + b \log\left(\frac{1}{1+e^{\lambda}}\right)$

$$= a \log\left(\frac{e^{\lambda}}{1+e^{\lambda}}\right) + b \log\left(\frac{1}{1+e^{\lambda}}\right)$$

$$= a\lambda + (a+b) \log\left(\frac{1}{1+e^{\lambda}}\right)$$

$$l(D,\lambda) = a\lambda - (a+b) \log(1+e^{\lambda})$$

For MLE, we need to first solve,

$$\frac{\partial l(D,\lambda)}{\partial \lambda} = 0$$

$$a - \frac{(a+b)}{1+e^{\lambda}} \cdot e^{\lambda} = 0 \Rightarrow a + ae^{\lambda} = ae^{\lambda} + be^{\lambda}$$

$$\Rightarrow be^{\lambda} = a \Rightarrow \lambda = \log\left(\frac{a}{b}\right)$$

So, $\log\left(\frac{a}{b}\right)$ might be the MLE. We just have to confirm if $\frac{\partial^2 l}{\partial \lambda^2} < 0$ at MLE.

$$\frac{\partial^2 l(D,\lambda)}{\partial \lambda^2} = 0 - \frac{\partial}{\partial \lambda}\left[(a+b)\left(1 - \frac{1}{1+e^{\lambda}}\right)\right] = (a+b)\left[-\frac{e^{\lambda}}{(1+e^{\lambda})^2}\right] < 0$$

As, $e^{\lambda} > 0$ and $(1+e^{\lambda})^2 > 0$, the double derivative is 0 always.

MLE is $\lambda = \log\left(\frac{a}{b}\right)$

(c) Standard Bernoulli distribution, $P(X=x|\theta) = \theta^x (1-\theta)^{1-x}$ for $X=\{0,1\}$, $\theta \in [0,1]$

This parametrization, $P(X=x|\lambda) = \left(\frac{1}{1+e^{-\lambda}}\right)^x \left(\frac{1}{1+e^{\lambda}}\right)^{(1-x)}$ for $X=\{0,1\}$.

Comparing both expressions,

$$\boxed{\theta = \frac{1}{1+e^{-\lambda}}} \Rightarrow \theta + \theta e^{\lambda} = e^{\lambda} \Rightarrow e^{\lambda} = \frac{\theta}{1-\theta} \Rightarrow \boxed{\lambda = \log\left(\frac{\theta}{1-\theta}\right)}$$

2. $$L(D,\theta) = \sum_{n=1}^{N} \log\left(1+e^{-y_n(wx_n^T+b)}\right) + \lambda\|w-w_0\|_2^2 + \lambda(b-b_0)^2$$

(a) $$\frac{\partial L(D,\theta)}{\partial w} = \sum_{n=1}^{N} \frac{1}{1+e^{-y_n(wx_n^T+b)}} \cdot e^{-y_n(wx_n^T+b)} \{-y_n\}.x_n$$

$$= \sum_{n=1}^{N} \frac{-y_n x_n}{1+e^{y_n(wx_n^T+b)}} + \frac{\lambda(ww^T - ww_0^T - w_0w^T)}{\partial w} + 0 + 2\lambda(w-w_0)$$

[As, $\frac{\partial}{\partial w}(ww^T) = 2w$, $\frac{\partial}{\partial w}(ww_0^T) = w_0$]

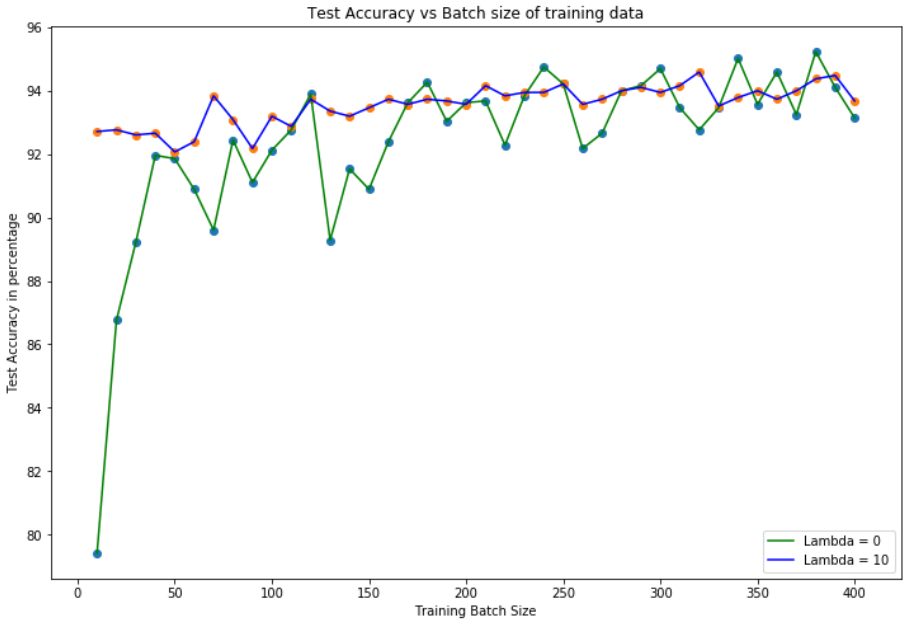
So, $$\boxed{\frac{\partial L(D,\theta)}{\partial w} = \sum_{n=1}^{N} \frac{-y_n \cdot x_n}{1+e^{y_n(wx_n^T+b)}} + 2\lambda(w-w_0)}$$

$$\frac{\partial L(D,\theta)}{\partial b} = \sum_{n=1}^{N} \frac{1}{1+e^{-y_n(wx_n^T+b)}} \cdot e^{-y_n(wx_n^T+b)} \cdot (-y_n) + 2\lambda(b-b_0)$$

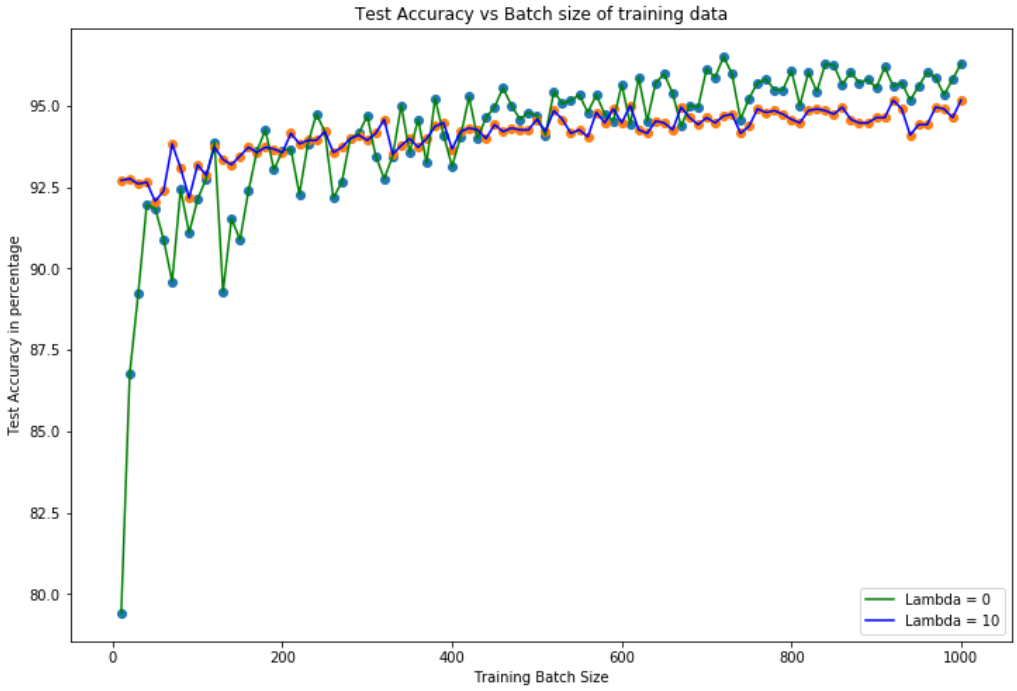
$$\boxed{\frac{\partial L(D,\theta)}{\partial b} = \sum_{n=1}^{N} \frac{-y_n}{1+e^{y_n(wx_n^T+b)}} + 2\lambda(b-b_0)}$$

(b) I first calculated the objective and objective_grad functions. I set the values of initial w and b to be zeros. Now our goal is to minimize the value of the objective function with the help of objective_grad function. In each update value of w and b will change. I used BFGS (Broyden-Fletcher-Goldfarb-Shanno) method to minimize the objective function. The scipy function fmin_l_bfgs_b implements this method. The function returns optimum value of w, b. This w and b are then finally used for predicting outputs to new test data.

2. c. **Plot showing the test accuracy as a function of the number of training data cases for both values of λ**



2. d. From the graph it is evident that the performance of the model for λ = 0 is poor when the training data size is less. For λ = 10 the regularization prior term is present and thus it helps in getting good accuracy even when the data size is less. But with increasing training data size, $(\sum_{i=1}^{N} \log(1 + \exp(-y_n(wx_{|n} + b))))$ term overwhelms the prior term. Also, regularization significantly reduces the variance of the model, without substantial increase in its bias. From the graph it can be seen λ = 0 plot has more variance in its prediction than the plot of λ = 10. But after certain increase of data, the model starts losing important information, giving rise to bias in the model and thus underfitting. Therefore, the value of λ should be carefully selected.



3.

$$L_{k,\delta}(y,y') = \begin{cases} \frac{1}{2k}(y-y')^{2k} & \text{if } |y-y'| \le \delta \\ \delta^{2k-1}\left(|y-y'| - \frac{(2k-1)}{2k}\delta\right) & \text{otherwise} \end{cases}$$

$$L_{k,\delta}(D,\theta) = \sum_{n=1}^{N} L_{k,\delta}(y_n, f(x_n,\theta)) \quad \text{where } f(x_n,\theta) = wx_n^T + b$$

(a)

$$L_{k,\delta}(D,\theta) = \sum_{n=1}^{N} L_{k,\delta}(y_n, wx_n^T + b)$$

$$\frac{\partial L_{k,\delta}(D,\theta)}{\partial w} = \sum_{n=1}^{N} \frac{\delta L_{k,\delta}(y_n, wx_n^T+b)}{\delta w} = \sum_{n=1}^{N} \begin{cases} \frac{\partial}{\partial w}\left(\frac{1}{2k}(y-(wx_n^T+b))^{2k}\right) \\ \frac{\partial}{\partial w}\left(\delta^{2k-1}\left(|y-(wx_n^T+b)| - \frac{2k-1}{2k}\delta\right)\right) \end{cases}$$

$$= \sum_{n=1}^{N} \begin{cases} (y-(wx_n^T+b))^{2k-1}(-x_n) & \text{if } |y-(wx_n^T+b)| \le \delta \\ \delta^{2k-1} \cdot \frac{\partial |y-(wx_n^T+b)|}{\partial w} & \text{otherwise} \end{cases}$$

$$= \sum_{n=1}^{N} \begin{cases} -(y-(wx_n^T+b))^{2k-1}x_n & \text{if } |y-(wx_n^T+b)| \le \delta \\ \delta^{2k-1}(-x_n) & \text{if } |y-(wx_n^T+b)| > \delta \ \&\ > 0 \\ \delta^{2k-1}(x_n) & \text{if } |y-wx_n^T+b)| > \delta \ \&\ < 0 \end{cases}$$

$$\frac{\partial L_{k,\delta}(D,\theta)}{\partial b} = \sum_{n=1}^{N} \begin{cases} \frac{\partial}{\partial b}\left(\frac{1}{2k}(y-(wx_n^T+b))^{2k}\right) & \text{if } |y-(wx_n^T+b)| \le \delta \\ \frac{\partial}{\partial b}\left(\delta^{2k-1}\left(|y-(wx_n^T+b)| - \frac{2k-1}{2k}\delta\right)\right) & \text{otherwise} \end{cases}$$

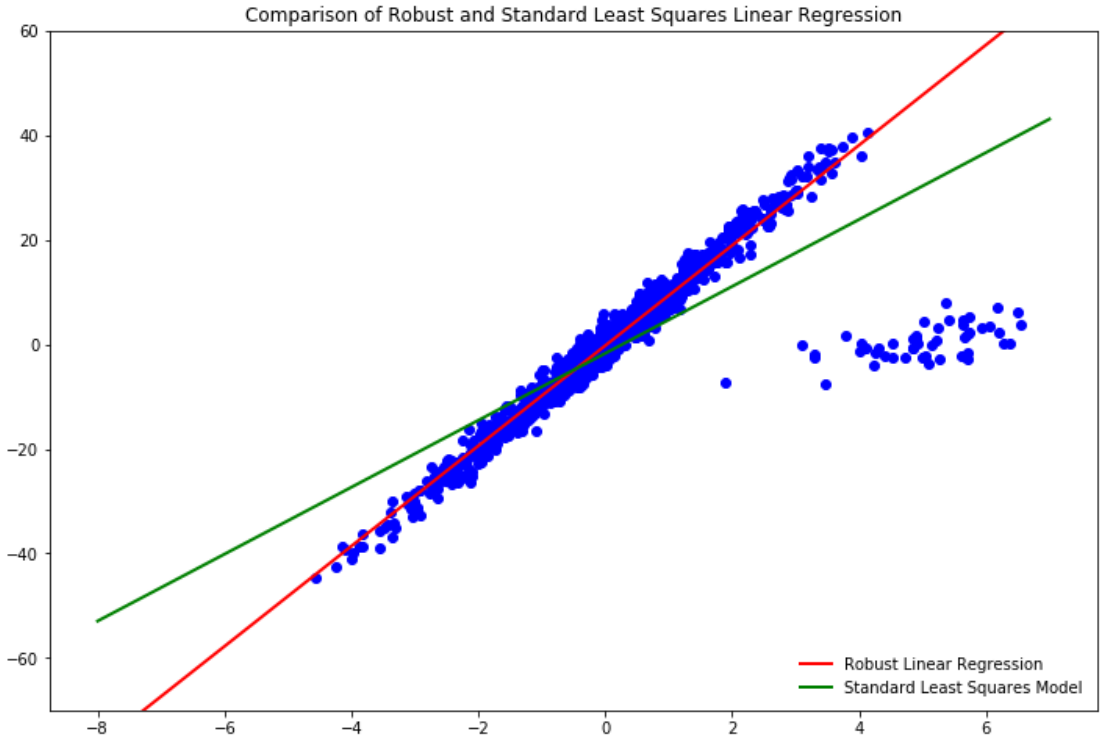
$$= \sum_{n=1}^{N} \begin{cases} (y-(wx_n^T+b))^{2k-1}\cdot(-1) & \text{if } |y-(wx_n^T+b)| \le \delta \\ \delta^{2k-1}\frac{\partial |y-(wx_n^T+b)|}{\partial b} & \text{otherwise} \end{cases}$$

$$= \sum_{n=1}^{N} \begin{cases} -(y-(wx_n^T+b))^{2k-1} & \text{if } |y-(wx_n^T+b)| \le \delta \\ -\delta^{2k-1} & \text{if } |y-(wx_n^T+b)| > \delta \ \&\ > 0 \\ \delta^{2k-1} & \text{if } |y-(wx_n^T+b)| > \delta \ \&\ < 0 \end{cases}$$

3.b. First I calculated the objective function and the objective_grad functions. Our goal is to minimize the objective function with the help of objective_grad function. First we set the parameters w and b with zeroes. Then I used scipy function fmin_l_bfgs_b functiom which implements the BFGS method to optimize the objective function. This function returns w and b of the learned model. This w and b are then finally used for prediting outputs to new test data.

3.c. For k=1 and δ = 1, the MSE achieved by the Generalized Robust Regression is 115.835. The MSE achieved by Standard Least-squares Regression is 76.043.

3.d. **Figure that includes a scatter plot of the training data, the regression line found using the robust model, and the regression line found using the standard least-squares model**



3.e. The above plot clearly portrays how the least-squares regression model is affected by the presence of outliers. Whereas the robust linear regression model has optimized objective function which minimizes the weightage of the outliers. Thus the ribust linear regression model will have better generalized performance on future test data.