CSL603 - Machine Learning
Lab 2

Aditya Gupta
2015CSB1003

November 10, 2017

1 Linear Ridge Regression

Given X and Y we will find W that minimizes $J(W)$, the error function and are defined as:

$$f(X) = \underbrace{\begin{pmatrix} 1 & x_{11} & \cdots & x_{1D} \\ 1 & x_{21} & \cdots & x_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{N1} & \cdots & x_{ND} \end{pmatrix}}_{X} \underbrace{\begin{pmatrix} w_0 \\ w_1 \\ \vdots \\ w_D \end{pmatrix}}_{W} = \underbrace{\begin{pmatrix} y_0 \\ y_1 \\ \vdots \\ y_N \end{pmatrix}}_{Y}$$

$$\min_W J(W) \equiv \min_W \frac{1}{2}(XW - Y)^T(XW - Y) + \frac{1}{2}\lambda||W||^2$$

Which when solved gives us:

$$W = (X^T X + \lambda I)^{-1} X^T Y$$

Observations

The following observations were obtained:

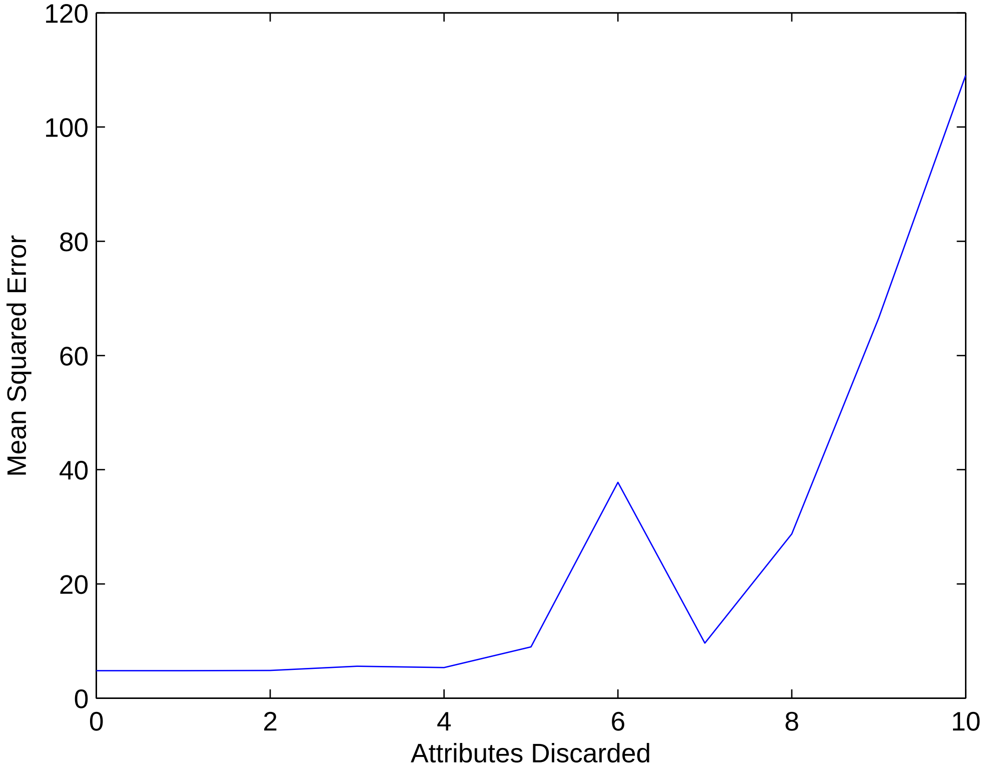

Figure 1: Mean Squared Error for the data after discarding increasing number of least significant weights in W.

- A particular value of λ(say 0) were chosen and then the magnitude of entries in the weights W were compared and one by one the least significant ones were discarded and the mean squared error changes can be seen in Figure 1. We can see that discarding 2-4 least significant attributes does not make any major change to the mean squared error, hence we can conclude that the input data contained some attributes that were irrelevant in estimating the output values.

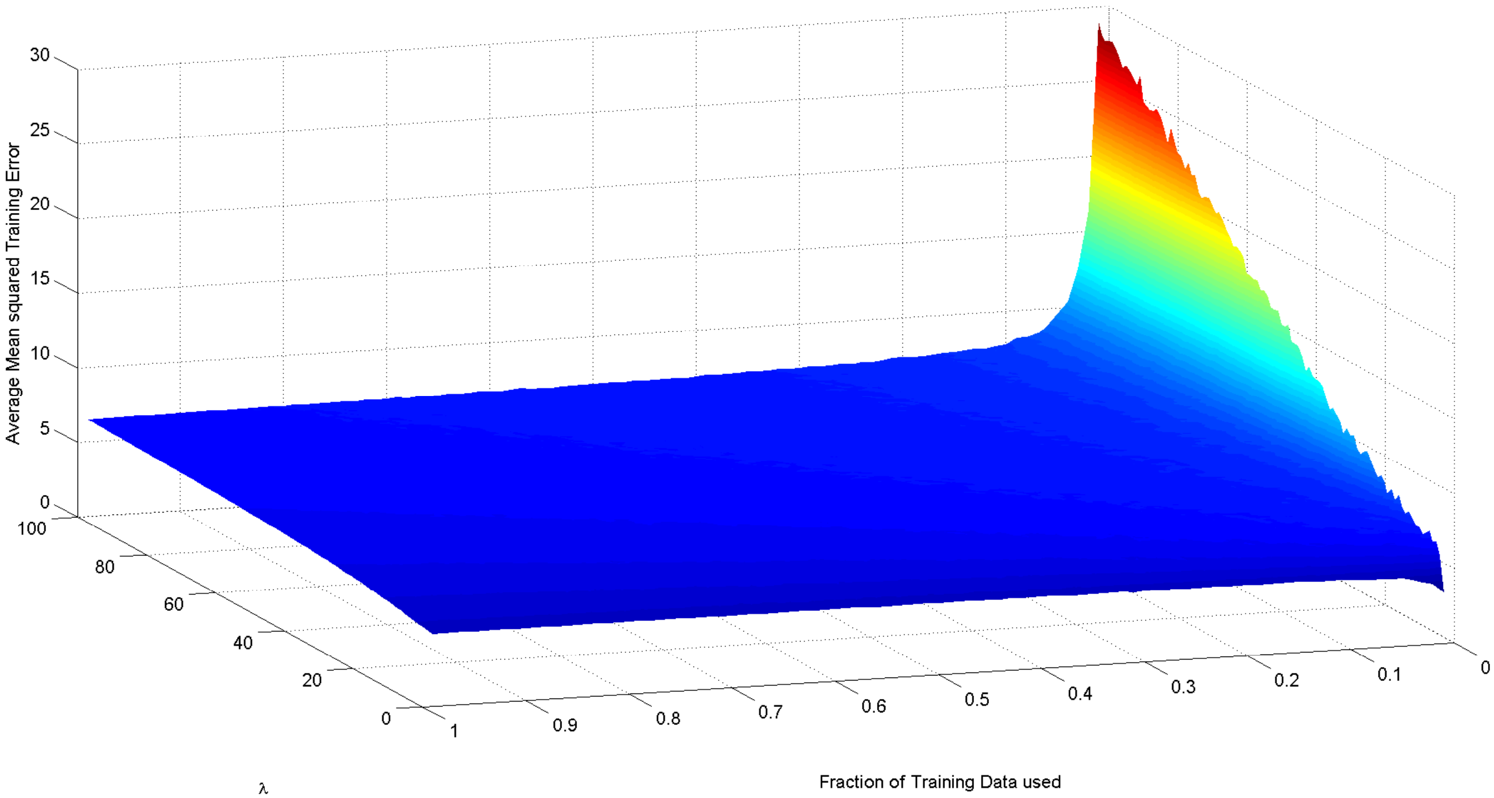

Figure 2: Average Mean Sqaured Error for various values of training set fraction and λ values used in Ridge Regression for Training Data.

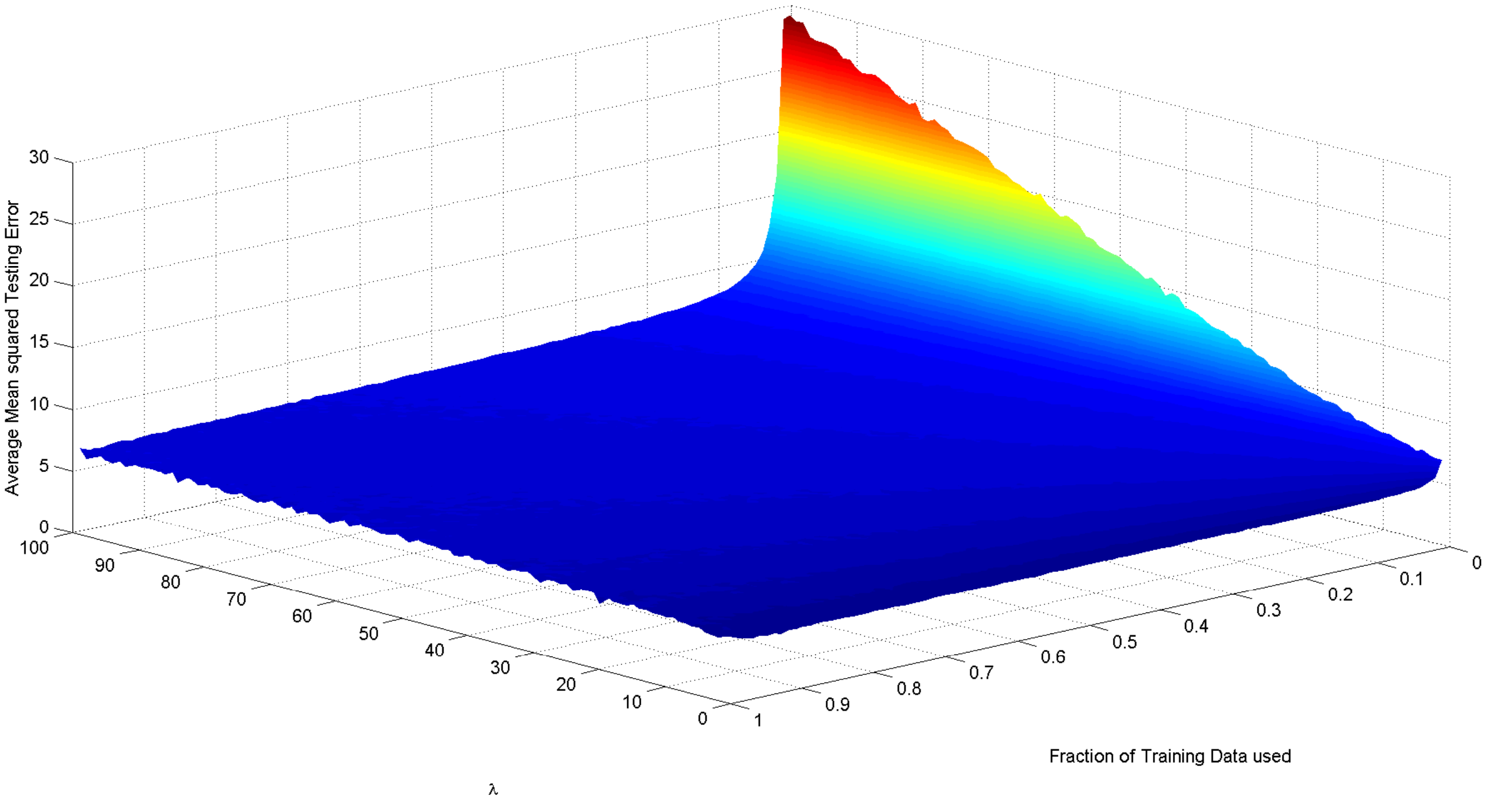

Figure 3: Average Mean Sqaured Error for various values of training set fraction and λ values used in Ridge Regression for Testing Data.

- The effect of λ on error was observed for different partitions of the data into training and testing sets. The average mean squared error for 100 repetitions for splitting-fractions varying from 1% to 99% and lambda values from 0 to 100 were observed. The surface corresponding to the average mean absolute error can be seen in Figure 2 and 3. We can see that for low values of training set fraction or high λ values the average mean squared error increased quite a bit.

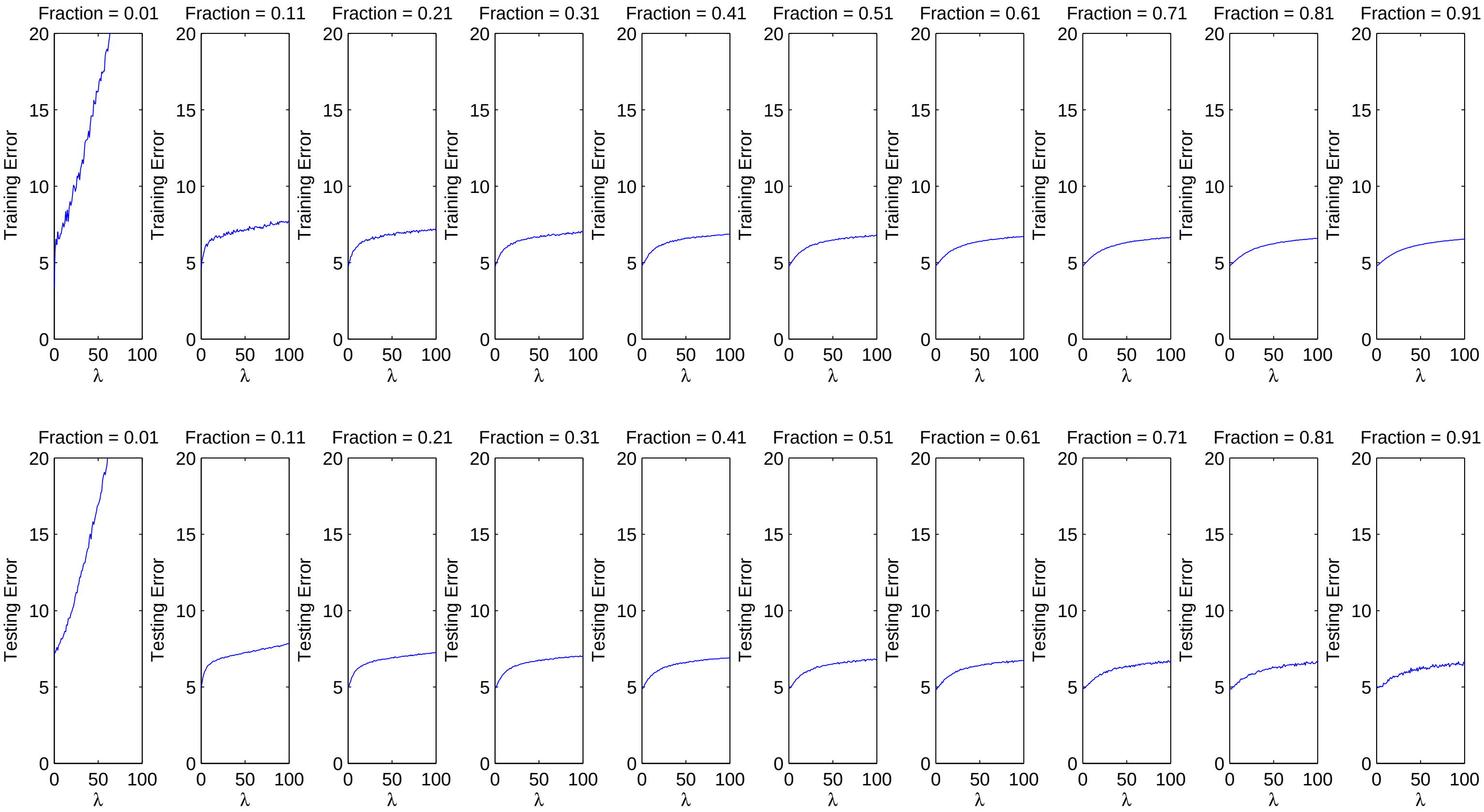

Figure 4: Average Mean Squared Error for various training set fractions varying against the λ values

- Figure 2 and 3's surfaces can be plotted into different graphs for few particular values of splitting-fractions and varying lambda and observing the change in average mean absolute error. Figure 4 shows that for high λ values the average mean squared error increases and is shaped like a convex function and the increase is more apparent in low training set fraction values.

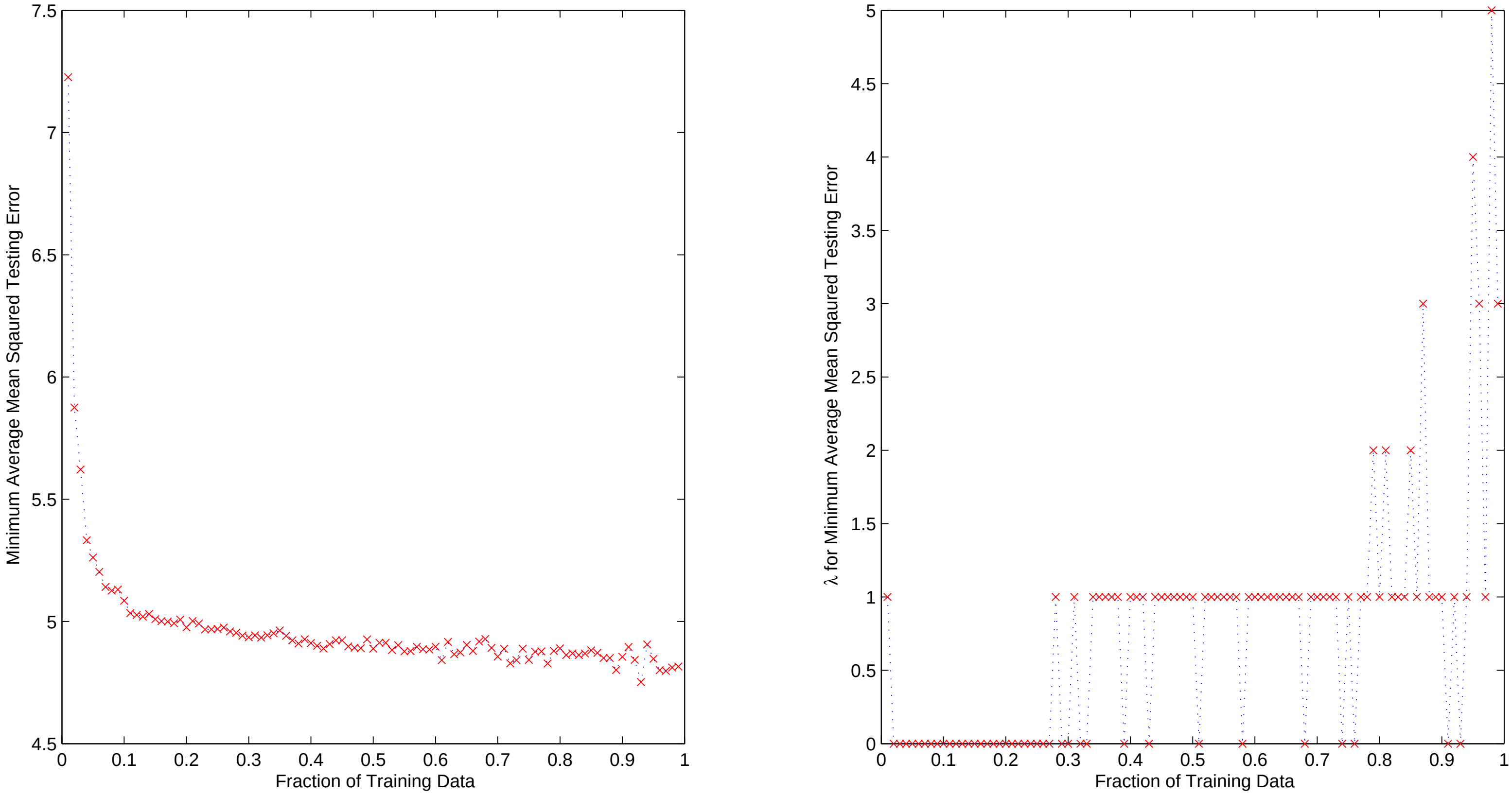

Figure 5: Minimum Average Mean Squared Error for various values of training set fraction and the corresponding λ values.

- Now we noted the minimum average mean squared testing error for each training set fraction values. Also the corresponding λ value was observed. We can see from Figure 5 that with high training set fraction the minimum average mean squared error decreases and though the λ values at which these values are obtained in general have higher magnitude (λ) as we increase the training set fraction, we can say that probably the model is overfitting and to penalize it we need higher magnitude of λ.

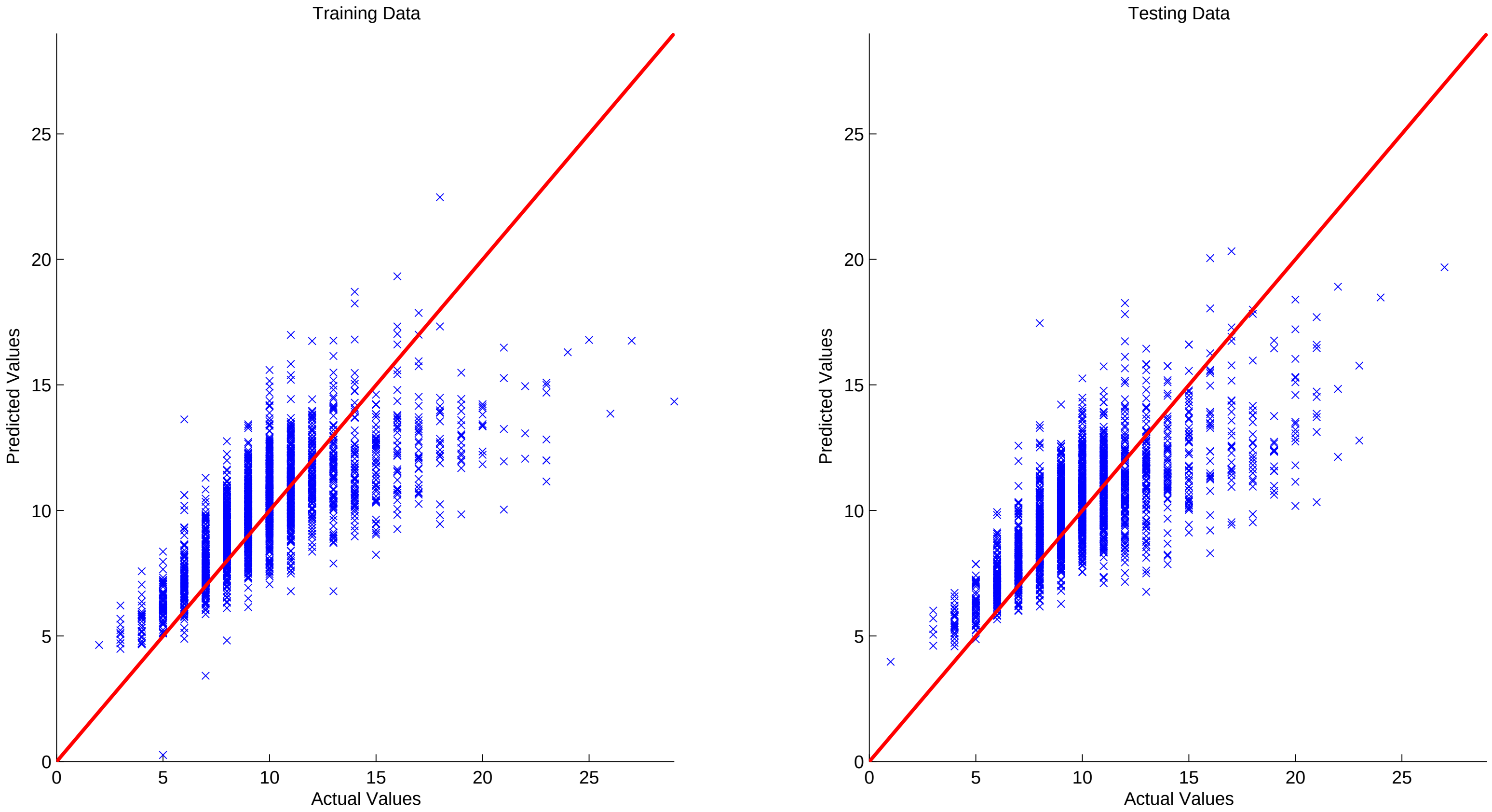

Figure 6: Relation between the actual data set values and predicted values for training and testing data. The reference line $y = x$ is shown in red.

- The correspondence between the actual and predicted values were also observed. For perfect prediction, this should correspond to a straight line through the origin at 45° degrees. Figure 6 shows that the actual values and the predicted values lie close to the line $y = x$ thus ensuring that there are significant correlation and accuracy to the predicted values.

Summary: Conclusions

- **Underfitting**: With high λ values the average mean squared error increases.
- **Decrease in Variance**: With high training set fraction the average mean squared error decreases.

- **Correctness of the Model**: The actual and predicted values lie quite close to the line $y = x$.
- **Redundant Attributes**: Discarding few attributes doesn't make quite a difference hence their irrelevancy to the use in prediction of output values.

2 Regularized Logistic Regression

Given x_1 and x_2 attribute values for different individual whether the credit card application should be accepted or rejected.

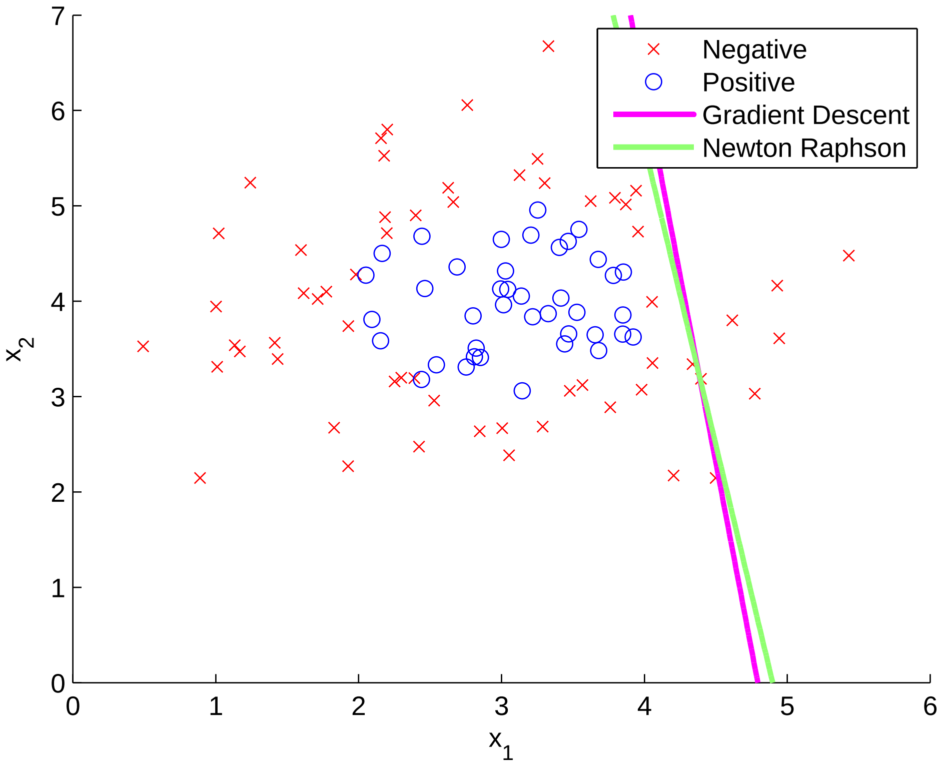

Figure 7: Distribution of the dataset, the positive examples are in blue circles and negative examples in red crossed. The magenta line represents the class boundary obtained using gradient descent (linear classification), similarly the green one for Newton Raphson method.

- As we know:

$$\sigma(\theta) = \frac{1}{1 + e^{-\theta}}$$

$$f(x_i) = \sigma\left(\sum_{d=1}^{D} w_i x_{id}\right)$$

$$J(w) = -\sum_{i=1}^{N}[y_i \log(f(x_i)) + (1 - y_i)\log(1 - f(x_i))]$$

We obtain:

$$\nabla_w J(w) = X^T(\mathbf{F}(X) - \mathbf{Y})$$

Hence, for Gradient Descent:

$$w' = w - \alpha X^T(\mathbf{F}(X) - \mathbf{Y})$$

and for Newton Raphson:

$$w' = w - H^{-1}\nabla J(w)$$

where

$$H = X^T R X \text{ where } R = \text{diag}(f(x_1)(1 - f(x_1)), f(x_2)(1 - f(x_2)), ...)$$

so:

$$w' = w - (X^T R X)^{-1}(\mathbf{F}(X) - \mathbf{Y})$$

- The dataset plotted can be seen in Figure 7. It can be seen that the data is not linearly separable and hence both Gradient Descent and Newton Raphson do not classify it properly.

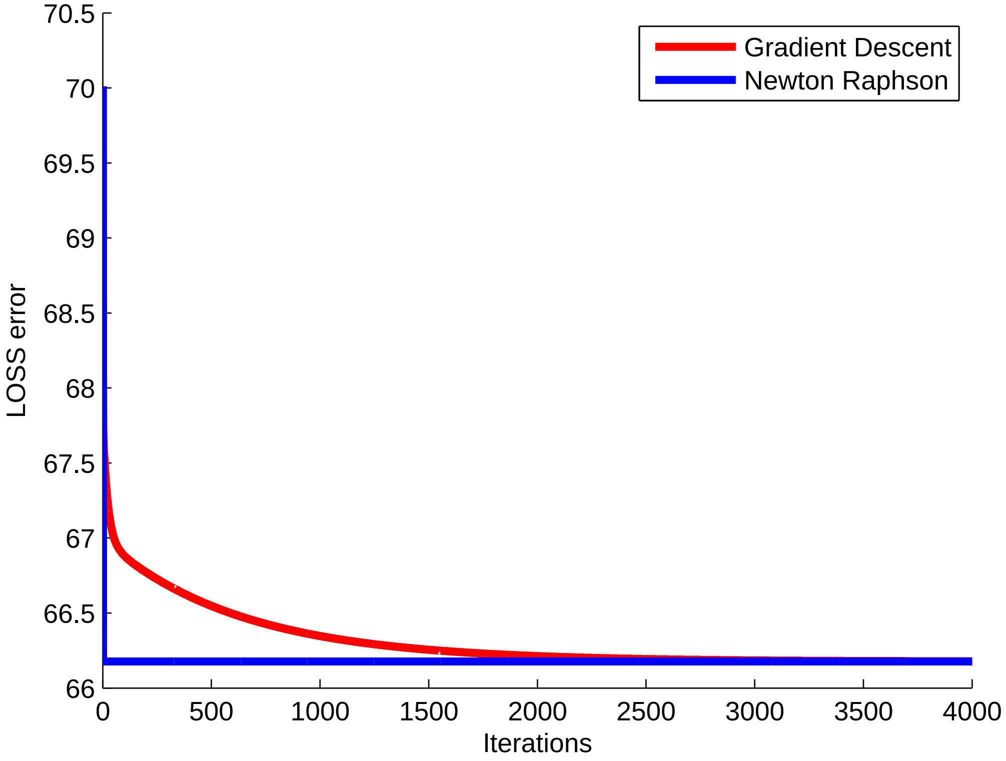

Figure 8: The loss error in prediction of classes by Gradient Descent and Newton Raphson methods using linear classification as a function of number of iterations.

- Both the methods reach almost the same solution. The number of iterations required by Gradient Descent is however large (~ 2000) as compared to Newton Raphson (~ 10) as seen in Figure 8.
- We can perform feature transformation to map x_1, x_2 to higher degree polynomials, then we obtain the Figures 9, 10, 11 and 12.

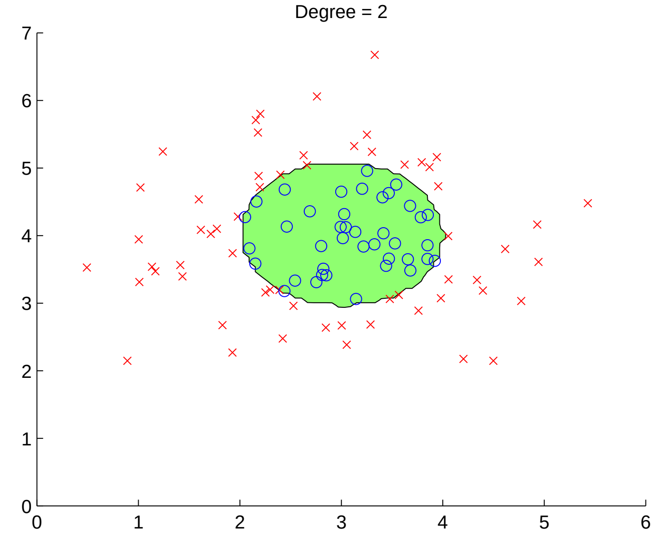

Figure 9: Plot of classification boundary for degree 2.

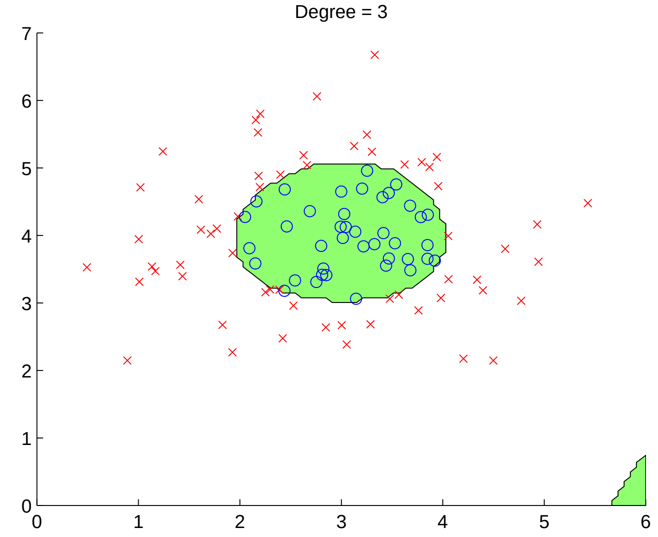

Figure 10: Plot of classification boundary for degree 3.

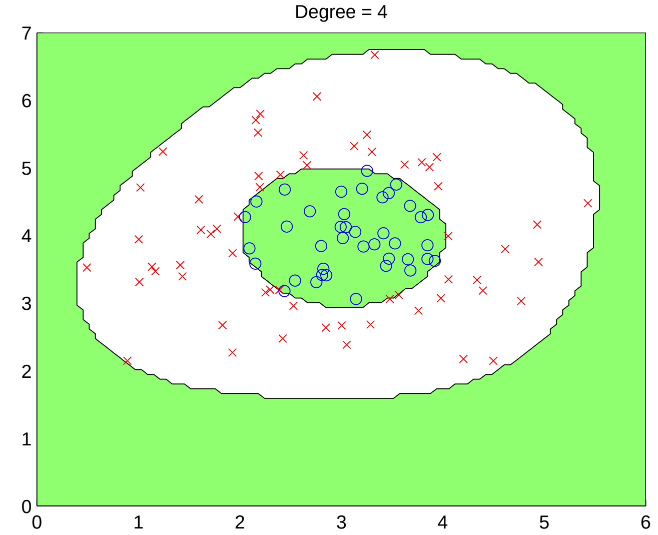

Figure 11: Plot of classification boundary for degree 4.

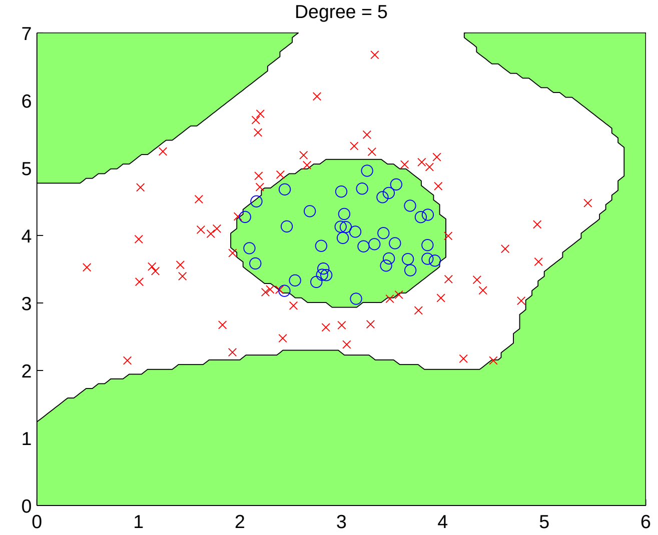

Figure 12: Plot of classification boundary for degree 5.

- We can also introduce a regularization parameter λ that will modify our equations in the following way:

$$J(w) = -\sum_{i=1}^{N}[y_i \log(f(x_i)) + (1 - y_i)\log(1 - f(x_i))] + \frac{\lambda}{2}||w||^2$$

$$\nabla_w J(w) = X^T(\mathbf{F}(X) - \mathbf{Y}) + \lambda w$$

For Newton Raphson:

$$w' = w - H^{-1}\nabla J(w)$$

where

$$H = X^T RX + \lambda I$$

so:

$$w' = w - (X^T RX + \lambda I)^{-1}(\mathbf{F}(X) - \mathbf{Y} + \lambda w)$$

- Some examples of figures then obtained are Figure 15 and 16. These boundaries underfit the given data but have simpler weights.

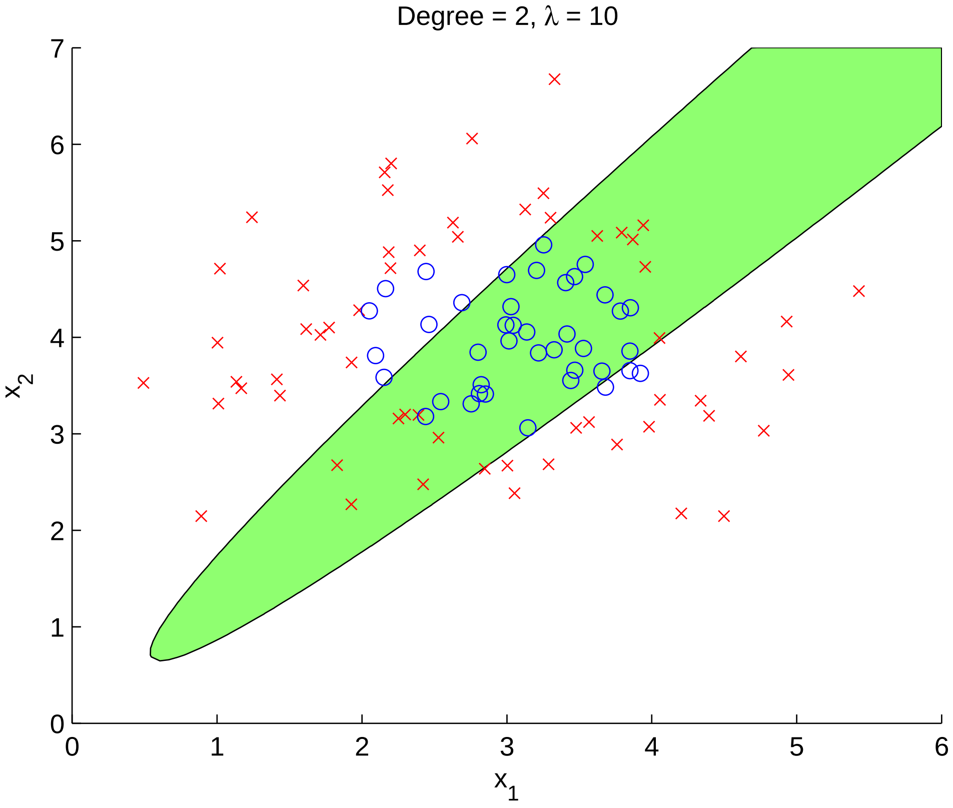

Figure 13: Plot of classification boundary for degree 2 ($\lambda = 10$).

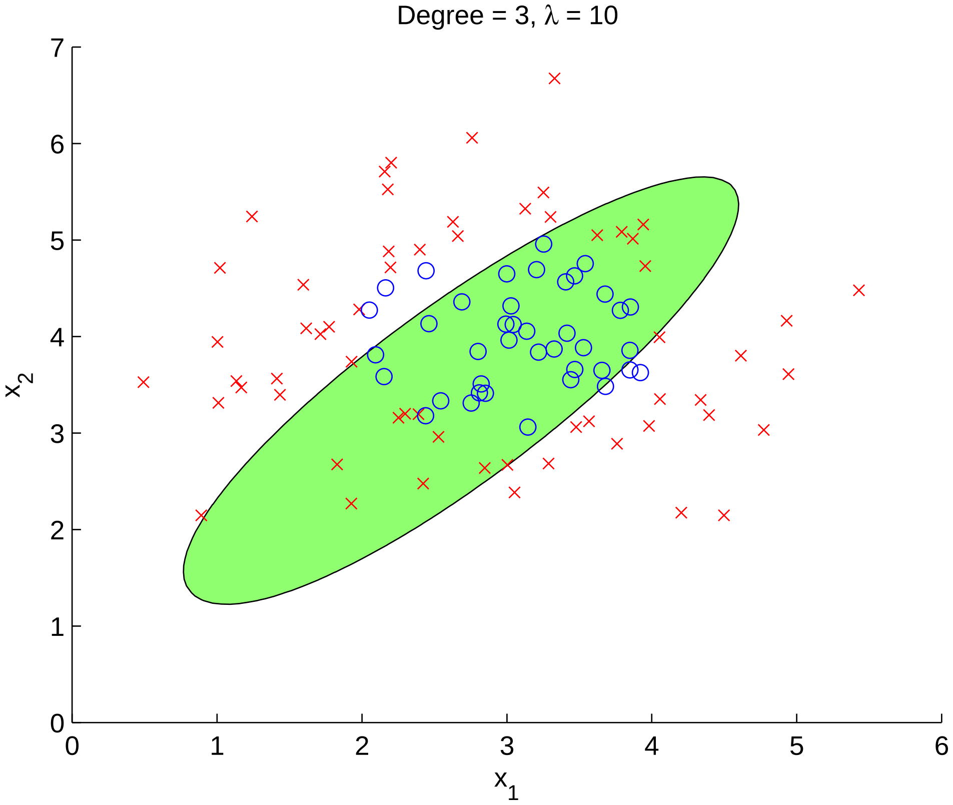

Figure 14: Plot of classification boundary for degree 3 ($\lambda = 10$).

- Now we divide the given data (we generate 1000 points using `l2reglog.m` with introducing 10% noise) into training and testing sets (50% - 50%) and check for overfitting and underfitting. The following appropriate figures were obtained.

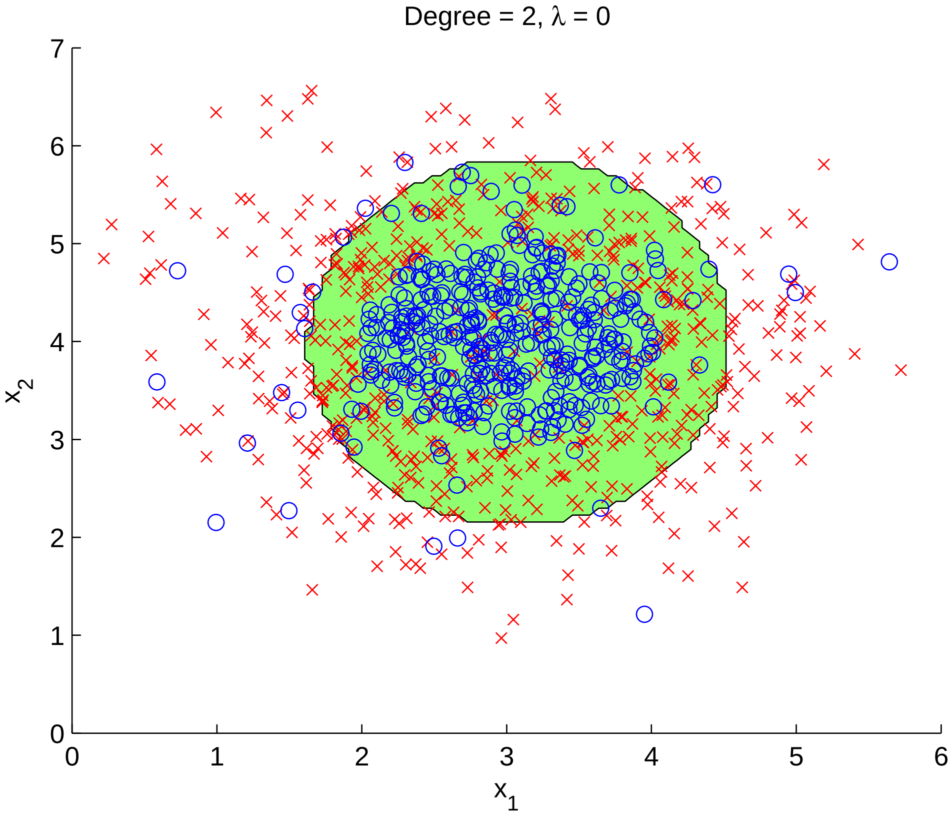

Figure 15: Plot of classification boundary for degree 2 ($\lambda = 0$) overfitting the data.

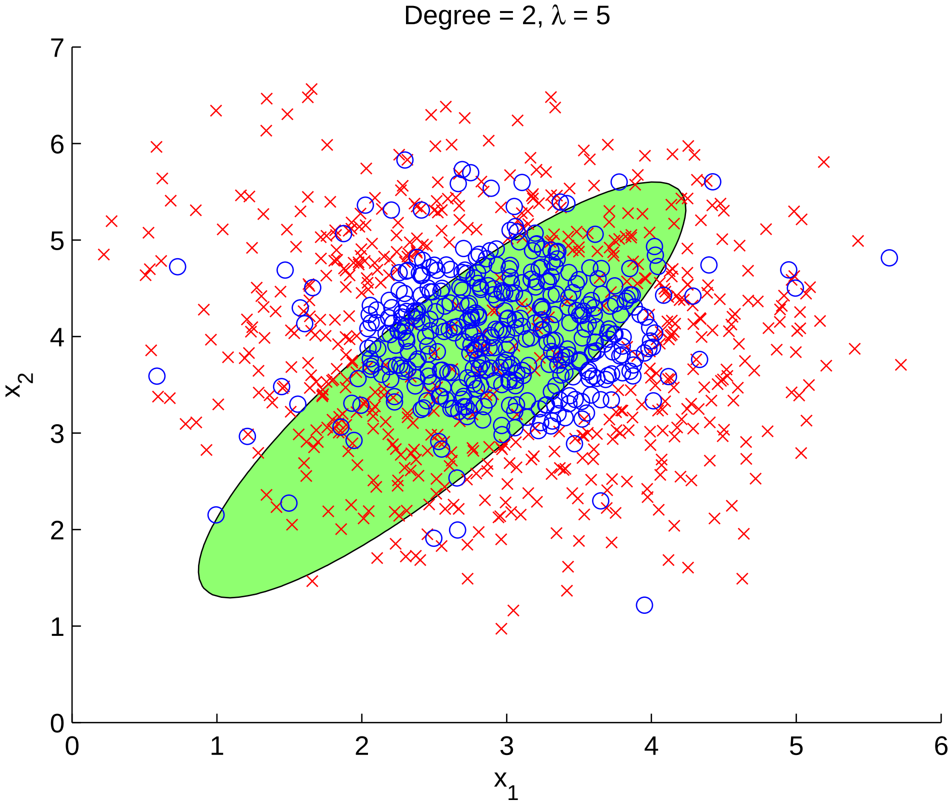

Figure 16: Plot of classification boundary for degree 2 ($\lambda = 10$) underfitting the data.