CSE574

Programming Assignment #2

Classification and Regression

Date: April 6th 2016

Group #32

Gian Pietro Farina (50176742)

Meghana Ananth Gad (50182335)

Ashwin Mittal (50168798)

1 Problem 1

1.1 Mathematical Analysis

Let's consider this generic formula, to classify new vector $\mathbf{x}$:
$\arg\max_{c\in\mathcal{C}} \frac{p(y=c)p(\mathbf{x}|y=c)}{\sum_{c\in\mathcal{C}} p(y=c)p(\mathbf{x}|y=c)}$. In our case the class conditional probabilities are modelled as normal distributions so this means, using the notation from [] that the above formula translates in:
$\arg\max_{c\in\mathcal{C}} \frac{\pi_c|2\pi\mathbf{\Sigma}_c|^{-\frac{1}{2}}\mathbf{exp}[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_c)^T\mathbf{\Sigma}_c^{-1}(\mathbf{x}-\boldsymbol{\mu}_c)]}{\sum_{c\in\mathcal{C}} \pi_c|2\pi\mathbf{\Sigma}_c|^{-\frac{1}{2}}\mathbf{exp}[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_c)^T\mathbf{\Sigma}_c^{-1}(\mathbf{x}-\boldsymbol{\mu}_c)]}$. Where π_c are the prior probabilities, $\boldsymbol{\mu}_c$ the means and $\mathbf{\Sigma}_c$ the covariance matrices for every class. Notice that performing the previous maximization is the same (by the property of the exponential) as minimizing the following formula: $(\mathbf{x}-\boldsymbol{\mu}_c)^T\mathbf{\Sigma}_c^{-1}(\mathbf{x}-\boldsymbol{\mu}_c)$ which is called Mahalanobis distance and which is indeed what we implemented in our code because we wanted to be able to call a single function from both QDAlearn and LDAlearn functions, and hence we needed to be general. But to see why we get linear and quadratic boundaries let's keep analyzing the exponenential formula. In LDA we make the assumption that: $\forall\ c \in \mathcal{C}\ \mathbf{\Sigma}_c = \mathbf{\Sigma}$ so we can rewrite the above formula as:
$\arg\max_{c\in\mathcal{C}} \frac{\pi_c\mathbf{exp}[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_c)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_c)]}{\sum_{c\in\mathcal{C}} \pi_c\mathbf{exp}[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_c)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu}_c)]}$. With some calculations we can go further and see the equality with:

$$\arg\max_{c\in\mathcal{C}} \frac{\mathbf{exp}[log(\pi_c)+\boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\mathbf{x}-\frac{1}{2}\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x}-\frac{1}{2}\boldsymbol{\mu}_c\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_c]}{\sum_{c\in\mathcal{C}}\mathbf{exp}[log(\pi_c)+\boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\mathbf{x}-\frac{1}{2}\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x}-\frac{1}{2}\boldsymbol{\mu}_c\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_c]} = \quad (1)$$

$$\arg\max_{c\in\mathcal{C}} \frac{\cancel{\mathbf{exp}[-\frac{1}{2}\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x}]}\cdot\mathbf{exp}[log(\pi_c)+\boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\mathbf{x}-\frac{1}{2}\boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_c]}{\cancel{\mathbf{exp}[-\frac{1}{2}\mathbf{x}^T\mathbf{\Sigma}^{-1}\mathbf{x}]}\cdot\sum_{c\in\mathcal{C}}\mathbf{exp}[log(\pi_c)+\boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\mathbf{x}-\frac{1}{2}\boldsymbol{\mu}_c\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_c]}. \quad (2)$$

So we can see that the only quadratic term in $\mathbf{x}$ got cancelled out. Since we only care about the maximum class probability we can disregard about the denominator (which doesn't vary with $\mathbf{c}$) and consider only:
$\arg\max_{c\in\mathcal{C}} \mathbf{exp}[log(\pi_c)+\boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\mathbf{x}-\frac{1}{2}\boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_c]$. Since log is a monotonic function we can disregard the exponentiation and only care about what is inside it, getting: $\arg\max_{c\in\mathcal{C}}\ log(\pi_c)+\boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\mathbf{x}-\frac{1}{2}\boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_c$.
In our case the prior probabilities π_c for every class where all very similar and about 20% so we can disregard them, and consider only:
$\arg\max_{c\in\mathcal{C}} \boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\mathbf{x}-\frac{1}{2}\boldsymbol{\mu}_c^T\mathbf{\Sigma}^{-1}\boldsymbol{\mu}_c$. The previous formula is **linear** in $\mathbf{x}$ and indeed

if we consider the boundary with any two classes $c, c' \in \mathcal{C}$ given by the equation: $p(y = c^{'}|\mathbf{x}) = p(y = c|\mathbf{x})$ iff $\boldsymbol{\mu}_{c'}^T \boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_{c'}^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_{c'} = \boldsymbol{\mu}_c^T \boldsymbol{\Sigma}^{-1}\mathbf{x} - \frac{1}{2}\boldsymbol{\mu}_c^T \boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_c$ we see that this is the equation of a line. In the case of QDA we cannot perform the canceling from (1) to (2) and so we end up with a quadratic term in $\mathbf{x}$ in the final equation and this is why we end up with quadratic boundaries.

1.2 Experimental Results

The accuracy achieved with LDA is 97% while with QDA is 94%: evidently the data is best separated by lines and not by curves. In the following we can see the graphic results of LDA and QDA:

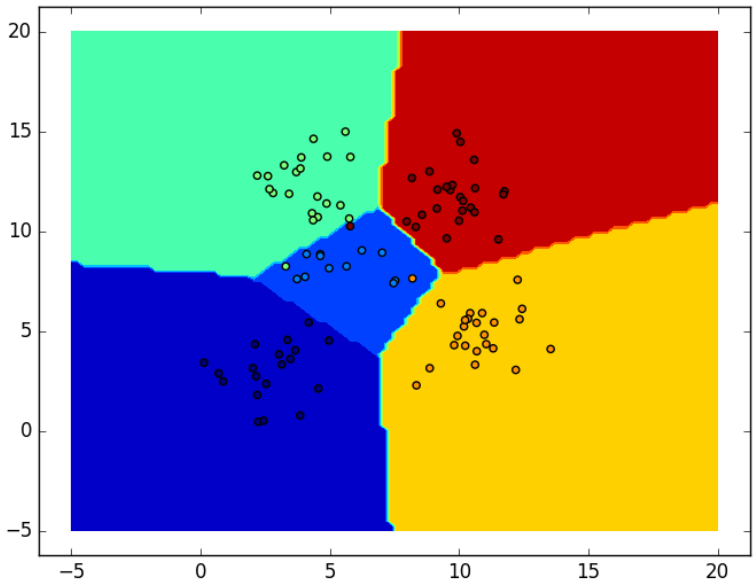


Figure 1: LDA

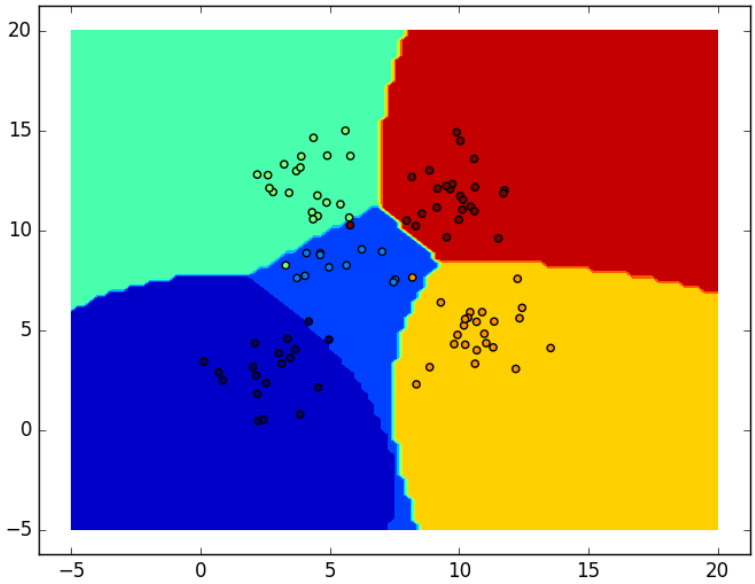


Figure 2: QDA

2 Problem 2

RMSE without intercept on training data – 138.20074835
RMSE with intercept on training data – 46.76708559

RMSE without intercept on test data – 326.76499438
RMSE with intercept on test data – 60.89203709

In this case, with Linear Regression we fit a straight line through the data (X/Y coordinates with X as predictor/independent variable and Y as response/dependent variable). The line is defined by its slope (m) and its intercept (b):

Y = mX + b + error
(where "error" are the residuals, typically assumed to be normally distributed with mean 0 and some variance σ^2). The RMSE is less with the intercept, i.e the method using intercept performs better. If we drop the intercept we make the slope steeper and force the line to pass through the origin and try to fit the full model, which does not give accurate results on our data set.

3 Problem 3

The magnitude of weights learnt using linear regression is larger when compared to the weights learnt using ridge regression. The range of weights (difference between maximum and minimum value) learnt using linear regression is 162553.925 and in ridge regression is 314.992. On average the weights learnt by linear regression vary by 865.485, when compared to the weights learnt by ridge regression, to compute this value we just computed the average component wise difference between the weights learnt in the two scenarios.

The difference in magnitude of weights learnt using the two approaches is because of the presence of the regularization term λ in the equation used by ridge regression for minimizing the residual errors.

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \mathbf{w}^\top \mathbf{x}_i)^2 + \frac{1}{2}\lambda||\mathbf{w}||_2^2$$

Equation 3.1

In linear regression the optimal weights are found by minimizing the squared loss error function J(w) as below

$$J(\mathbf{w}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - \mathbf{w}^\top \mathbf{x}_i)^2$$

Equation 3.2

$$\hat{\mathbf{w}} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

Equation 3.3

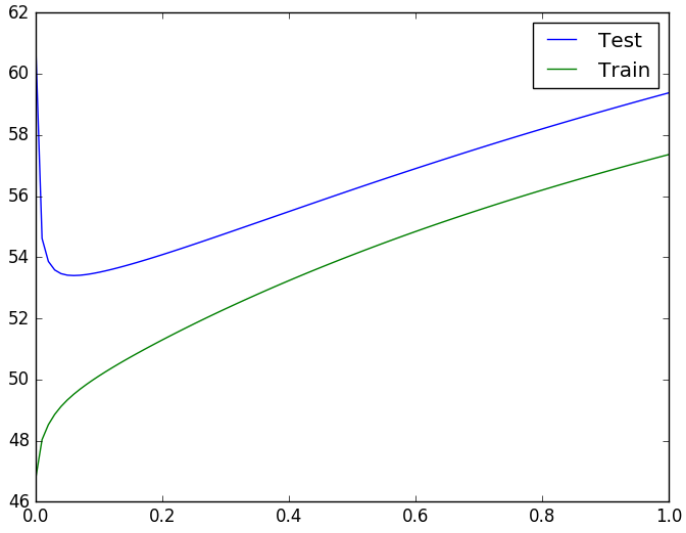
Minimization of Equation 3.2 with respect to w, results in Equation 3.3, which is then used to find the weights and learn the relationship between X and y. The weights learnt in linear regression, tend to over fit the training data and might perform poorly on testing data. This problem of overfitting is minimized in ridge regression.

In ridge regression, instead of just minimizing the residual errors, we add a penalty term λ on the weights: Equation 3.1. The optimal weights used in ridge regression are learnt by the equation below

$$\hat{\mathbf{w}}_{ridge} = (\lambda \mathbf{I}_D + \mathbf{X}^T \mathbf{X})^{-1} \mathbf{X}^T \mathbf{y}$$

This is obtained by minimizing Equation 3.1 with respect to w. The regularization factor λ is used to ensure that the regression line learnt during the training phase does not over fit the training data. But if the value of λ is increased then the regression line learnt would under fit the data.

In our assignment, we performed ridge regression by varying λ between 0 and 1 in steps of 0.01. Below is a plot of λ (along x-axis) against the root means square error calculated on testing data (along y-axis).

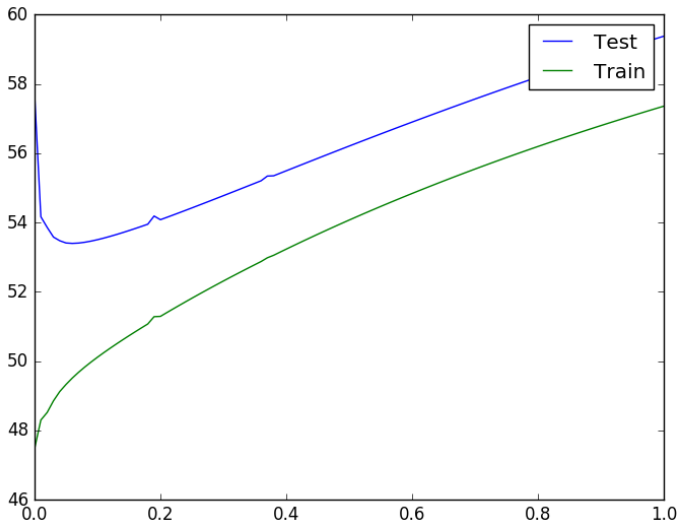


λ vs RMSE

From the above plot we infer optimal value of λ to be ~0.05, as the lowest value of root mean square error is observed at the point where λ = ~0.05. We notice that as the value of λ increases the root mean square error also increases and this is because weights learnt with larger values of λ under fit the training data and perform poorly on the testing data as well.

4 Problem 4

While training and testing the data using gradient descent based Ridge Regression Learning and varying the regularization parameter λ, we observe that the optimal λ remains almost same, i.e. 0.05, when compared to Problem 3. However, as the λ increases this method performs worse than the previous method. A close comparison between the plots shows that the curve in Problem 3 is a lower bound for the curve obtained in Problem 4.



λ vs RMSE

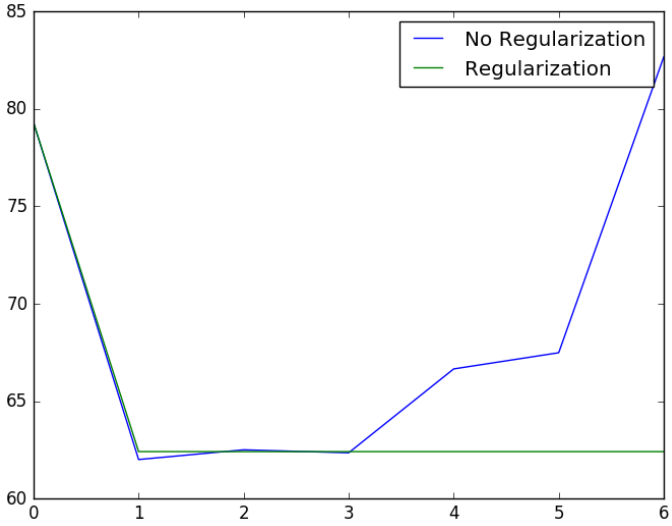
5 Problem 5

The comparison of RMSE for test data using No Regularization and using optimal Regularization (as found in previous problem) is as follows:

p	**RMSE (λ = 0)**	**RMSE (λ = λopt)**
0	79.2868513165	79.2898604296
1	62.0083440367	62.416796333
2	62.5070243981	62.4146141215
3	62.3536329193	62.4146033867
4	66.6582919959	62.4146030051
5	67.489483458	62.4146030085
6	82.6647394523	62.4146030086

The best value of p = 1, in case of No Regularization.
The best value of p = 4, in case of Regularization.

Overall, the method with Regularization performs better as it does not overfit for the complex curves for higher values of p. We want to stress that for p>=1, the performance of Regularization doesn't vary much and RMSE remains almost the same.



p vs RMSE

6 Problem 6

The RMSE form the above 4 problems for the test data are as follows:

Linear Regression – 60.89203709
Ridge Regression – 53.4
Ridge Regression (with gradient descent) – 53.4
Non-linear Regression – 62.008

In our experiments we use RMSE as the metric to choose the best setting.
We observe that RMSE is minimum in case of Ridge Regression and as explained in Problem 4, the overall model is better in case of Ridge Regression when we use the closed formula for learning weights (Problem 3) rather than the gradient descent. Therefore, we would recommend using the Ridge Regression for anyone using regression for predicting diabetes level using the input features.