CMPE462 - Machine Learning - Assignment 1

Berk Atıl - 2016400102

1 Introduction

In the first part of this assignment, we are asked to implement Perceptron Learning Algorithm(PLA) and apply it on 2D data which is also crated by us. To create the data points $y = -3x + 1$ function should be used. We will apply this algorithm on 50,100, and 5000 data points. In the second part, we are asked to implement Multiple Linear Regression algorithm and apply it on 2 different data sets. Lastly, regularization must be added.

2 Part 1

2.1 Step 1

I generated 25 positive and 25 negative data points randomly. After that, I applied PLA on this data and the result is shown on Figure 1.

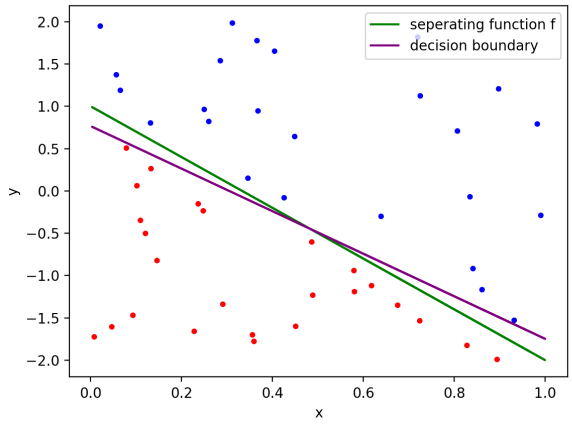

Figure 1: PLA on 50 data points

As it can be seen from the figure 1 that the separating function and decision boundary are a little bit different. This is expected because depending on the

initialization of the weight vector and selection of misclassified examples during the update, the resulting decision boundary may differ. It seems that the decision boundary separates the data points from different classes perfectly. Lastly, this decision boundary is generated with **59** iterations.

2.2 Step 2

I generated 50 positive and 50 negative data points randomly. After that, I applied PLA on this data and the result is shown on Figure 2.

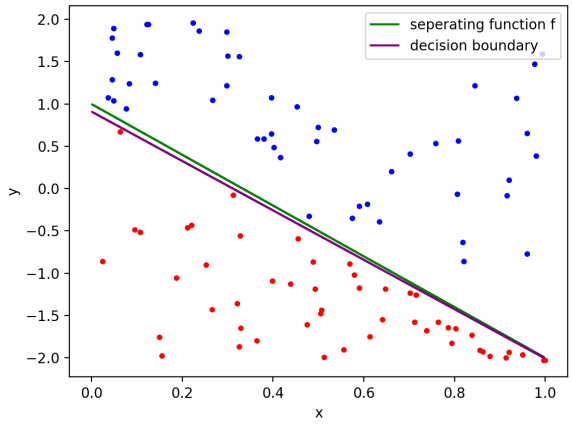

Figure 2: PLA on 100 data points

Again we have the same reason for the difference between separating function and decision boundary. Also, the decision boundary differentiates the data points from different classes perfectly as expected. Lastly, this decision boundary is generated with **124** iterations.

2.3 Step 3

I generated 2500 positive and 2500 negative data points randomly. After that, I applied PLA on this data and the result is shown on Figure 3.

Again we have the same reason for the difference between separating function and decision boundary. Also, the decision boundary differentiates the data points from different classes perfectly as expected. Lastly, this decision boundary is generated with **133** iterations.

2.4 Analysis of the Results

We have close decision boundary and target separating function for each step as expected. They are of course slightly different due to initialization of the

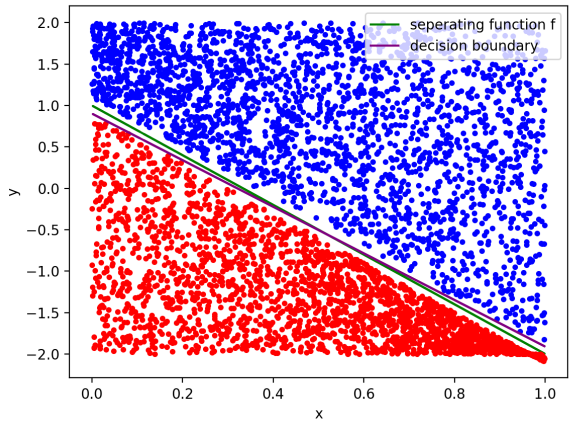

Figure 3: PLA on 5000 data points

weights and selection of misclassifed example at each iteration. As long as the data points from different classes are correctly separated all decision boundaries are valid. The number of iterations needed to generate the decision boundary depends on L_2 norm of the weight vector, max L_2 norm of the data points(the one with the highest value) and the distance between closest data points from different classes. For each step we should have similar L_2 norm of the weight vector and max L_2 norm of the data points. This is because, I used the same target separating function and same method to create data points. Also, because the method to create data points are the same for all steps, the difference between the closest points from different classes should not differ much. However, because of the randomness, some variations of these measurements are expected. When I look at the number of iterations, the second and third step are very similar, the first one is half of the others. The reason for the difference is probably that the minimum difference between data points from different classes is higher for the step 1(If we analyze the figures, it seems like this)

3 Part 2

I implemented the Multiple Linear Regression with closed form solution given below.

$$w^* = (X^T X + \lambda I)^{-1} X^T t$$

where X is the features, λ is the regularization parameter and t is the ground truth vector.

3.1 Step 1

Time to complete this step is 25ms including reading the data.

3.2 Step 2

Time to complete this step is 111ms including reading the data.

3.3 Step 3

Time to complete this step is 126ms including reading the data. For this step, I needed to choose λ. In order to choose this, I implemented 5-fold cross validation and looked at the plot to find the best λ. Initially, I tired values between 0 and 1. The plot for this is shown on figure 4.

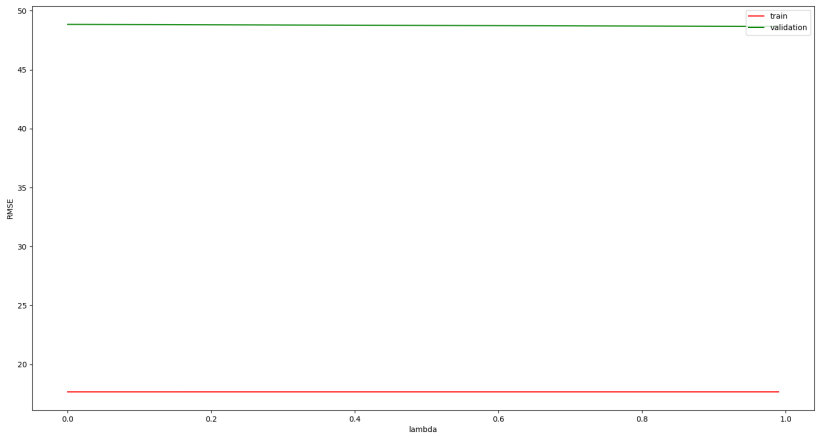

Figure 4: Regularization with λ between 0 and 1

It is clear from the figure 4 that with these λ values, we cannot avoid overfitting. The root mean square error values do not differ much and there is overfitting. After that, I changed the scale of λ and its plot is shown on figure 5.

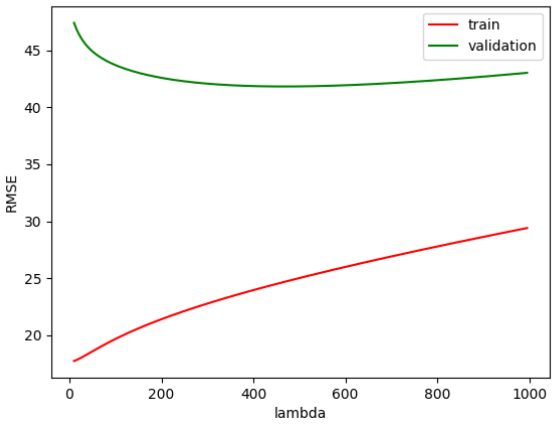

Figure 5: Regularization with λ between 10 and 1000

From figure 5, I chose 400 as a λ. However, this is also not a very good choice because there is still significant gap between the train and validation performances.

3.4 Analysis of the Results

Since I implemented the closed form solution, I can analyze the running time of each step. In the first data set (data set for step 1), we have **1000** data points and **100** features. In the second one (the one for step 2 and step 3), we have **1000** data points and **500** features. The reason for higher running time of step 2 and 3 is the higher number of features in the data set2. Because of larger matrices, the operations on these matrices require more time. The reason for the difference between step 2 and 3 may be because of the regularization because it requires additional operation. Also, the situation of the computer may also cause this.