CS584 Assignment 4

Jingyu Zhu

1. Introduction

In this assignment, I'm going to apply the support vector machine algorithm in classifying the data.

In the first part, I will create two data sets, one with linearly separable features and the other with features that are not separable.

In the second part, I will implement the support vector machine algorithm with hard margins and apply it on the created datasets and observe the results.

In the third part, I will show the procedures that how to develop the expression of the dual L_D that need to be maximized.

In the forth part, I will repeat the process in the second part but with the soft margins support vector machine algorithm.

In the fifth part, I will implement the Gaussian kernel based support vector machine algorithm and the polynomial kernel based support vector machine algorithm and apply them on the created dataset to the effects.

In the sixth part, I will introduce a solution to tackle the problem that one dataset is substantially larger than the other.

2. Proposed Solution

Support Vector Machine supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression

analysis. More formally, a support vector machine constructs a hyperplane or set of hyperplanes in a high or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier.

3. Implementation Details

Given some training data D, a set of n points of the form:

$$D = \{(x_i, y_i | x_i \in R^p, y_i \in \{-1,1\})\}_{i=1}^{n}$$

where the y_i is either 1 or −1, indicating the class to which the point x_i belongs. Each x_i is a p-dimensional real vector. We want to find the maximum-margin hyperplane that divides the points having $y_i = 1$ from those having $y_i = -1$. Any hyperplane can be written as the set of points x satisfying

$$w \cdot x - b = 1$$

and

$$w \cdot x - b = -1 \ .$$

By using geometry, we find the distance between these two hyperplanes is $\frac{2}{||w||}$, so we want to minimize $||w||$. As we also have to prevent data points from falling into the margin, we add the following constraint: for each i either:

$$w \cdot x_i - b \geq 1 \quad \text{for } x_i \text{ of the first class}$$

or:

$$w \cdot x_i - b \leq -1 \quad \text{for } x_i \text{ of the second class}$$

We can put this together to get the optimization problem:

Minimize (in w,b)

$$||w|| ,$$

subject to (for any i = 1, 2, …, n)

$$y_i(w \cdot x_i - b) \geq 1 .$$

The optimization problem is difficult to solve because it depends on $||w||$, the norm of w, which involves a square root. Fortunately it is possible to alter the equation by substituting $||w||$ with $\frac{1}{2}||w||^2$ without changing the solution. This is a quadratic programming optimization problem. More clearly:

$$\text{argmin}_{(w,b)} \frac{1}{2}||w||^2$$

subject to (for any i = 1, 2, …, n)

$$y_i(w \cdot x_i - b) \geq 1 .$$

By introducing Lagrange multipliers α, the previous constrained problem can be expressed as:

$$\text{argmin}_{(w,b)} max_{\alpha \geq 0} \{ \frac{1}{2} \left|\left|w\right|\right|^2 - \sum_{i=1}^{n} \alpha_i [y_i(w \cdot x_i - b) - 1] \} .$$

That is we look for a saddle point. In doing so all the points which can be separated as $y_i(w \cdot x_i - b) - 1 > 0$ do not matter since we must set the corresponding α_i to zero.

This problem can now be solved by standard quadratic programming techniques and programs. The "stationary" Karush–Kuhn–Tucker condition implies that the solution can be expressed as a linear combination of the training vectors:

$$w = \sum_{i=1}^{n} \alpha_i y_i \, x_i$$

Only a few α_i will be greater than zero. The corresponding x_i are exactly the support vectors, which lie on the margin and satisfy $y_i(w \cdot x_i - b) = 1$. From this one can derive that the support vectors also satisfy:

$$b = w \cdot x_i - y_i,$$

which allows one to define the offset b. The b depends on y_i and x_i, so it will vary for each data point in the sample. In practice, it is more robust to average over all N_{SV} support vectors, since the average over the sample is an unbiased estimator of the population mean:

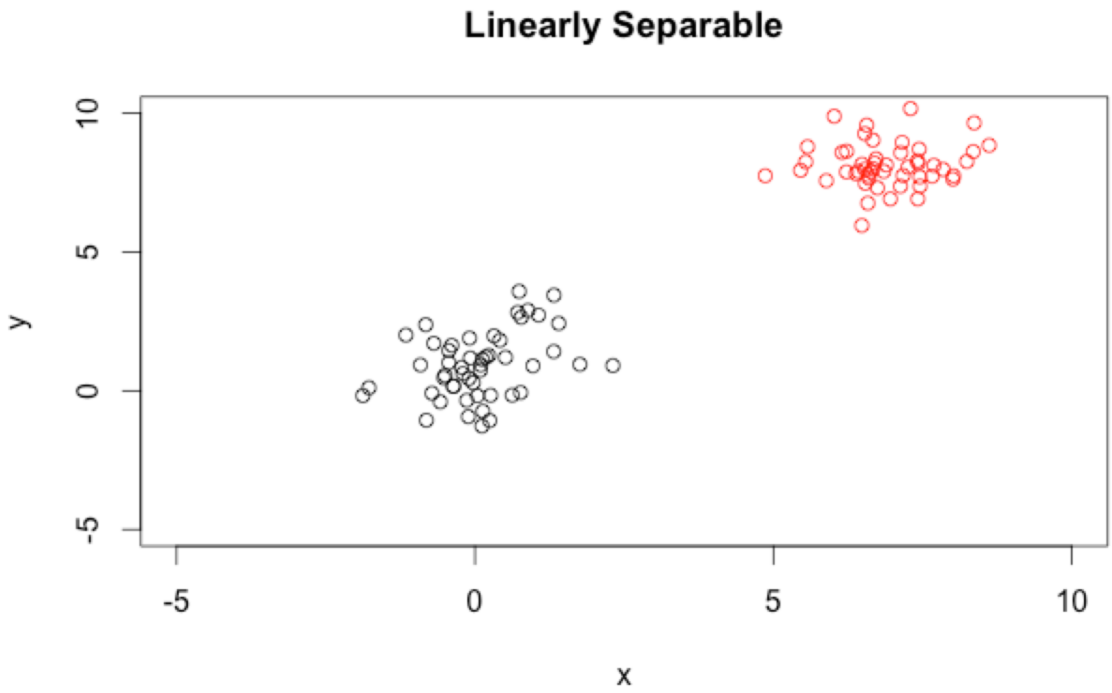
$$b = \frac{1}{N_{SV}} \sum_{i=1}^{N_{SV}} (w \cdot x_i - y_i)$$

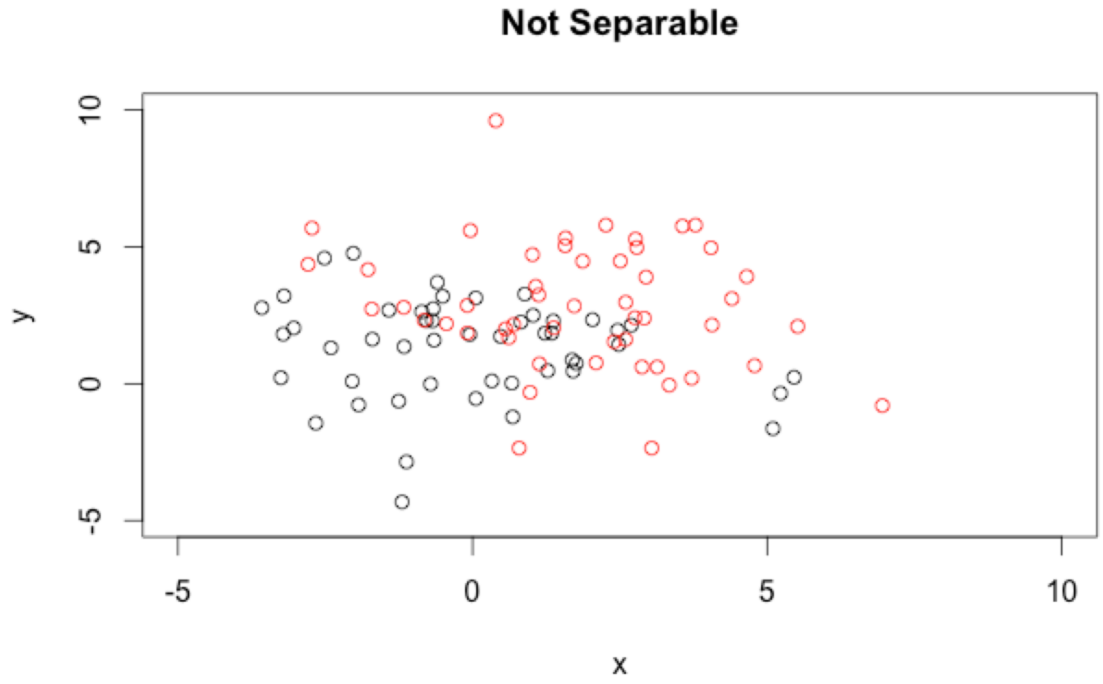
After we get all the necessary parameters, we can classify the testing examples by calculating $w^T x - w_0$. If $w^T x - w_0 > 1$, then we classify it to class1. Otherwise, we classify it to class 0.

4. Results

Part1:

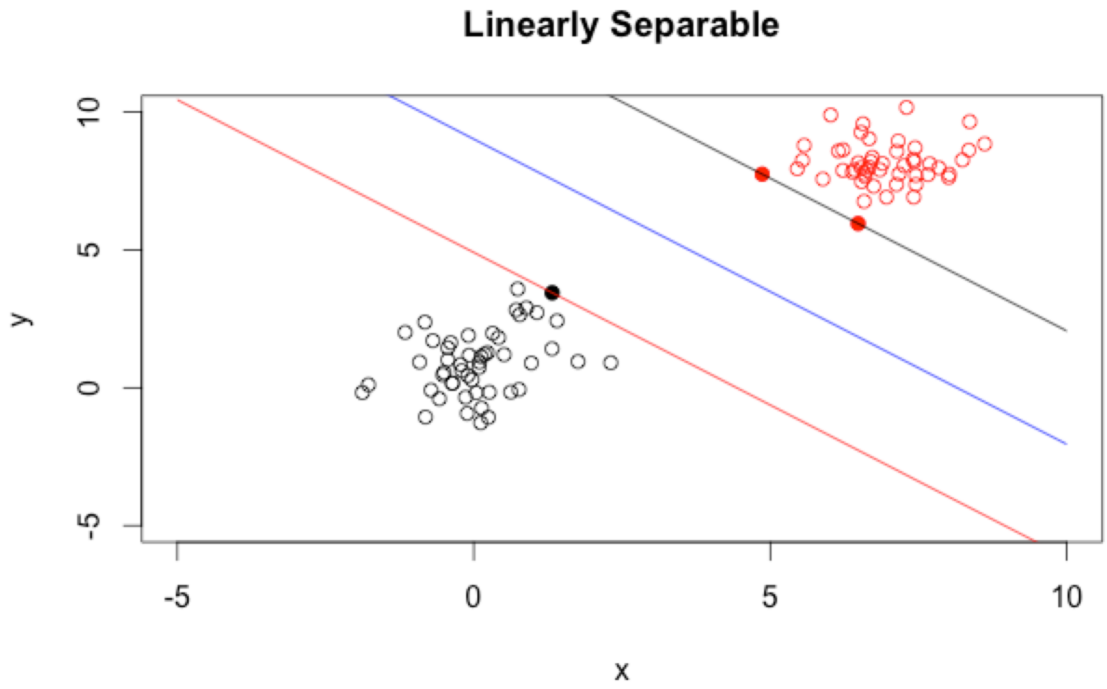
The datasets are generated as follow:

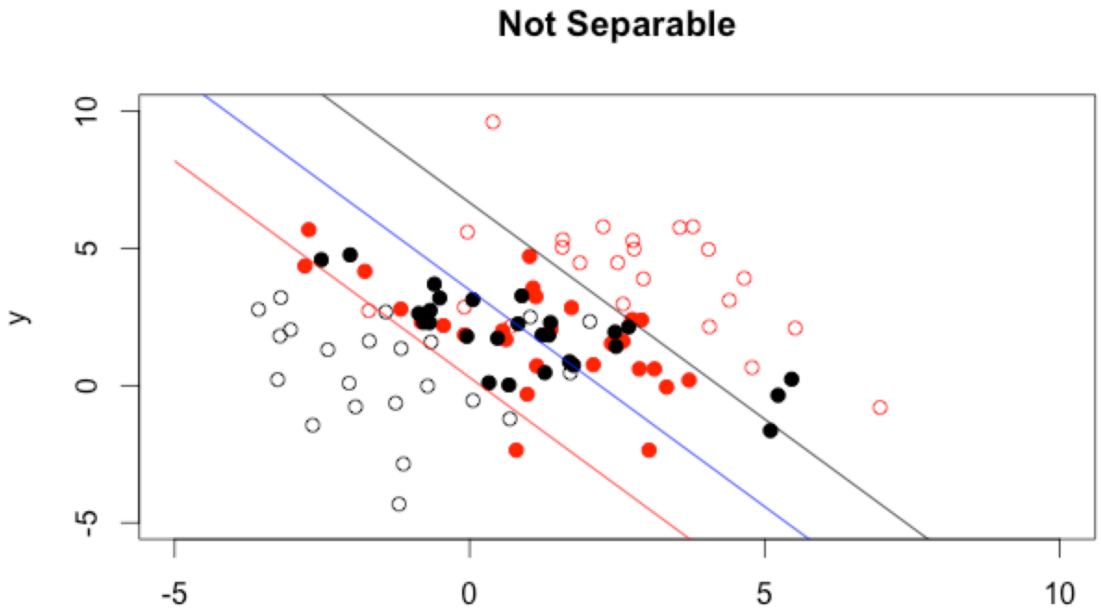




Part2:

I implement the support vector algorithm with hard margin and the data and the margins are plotted as follow:



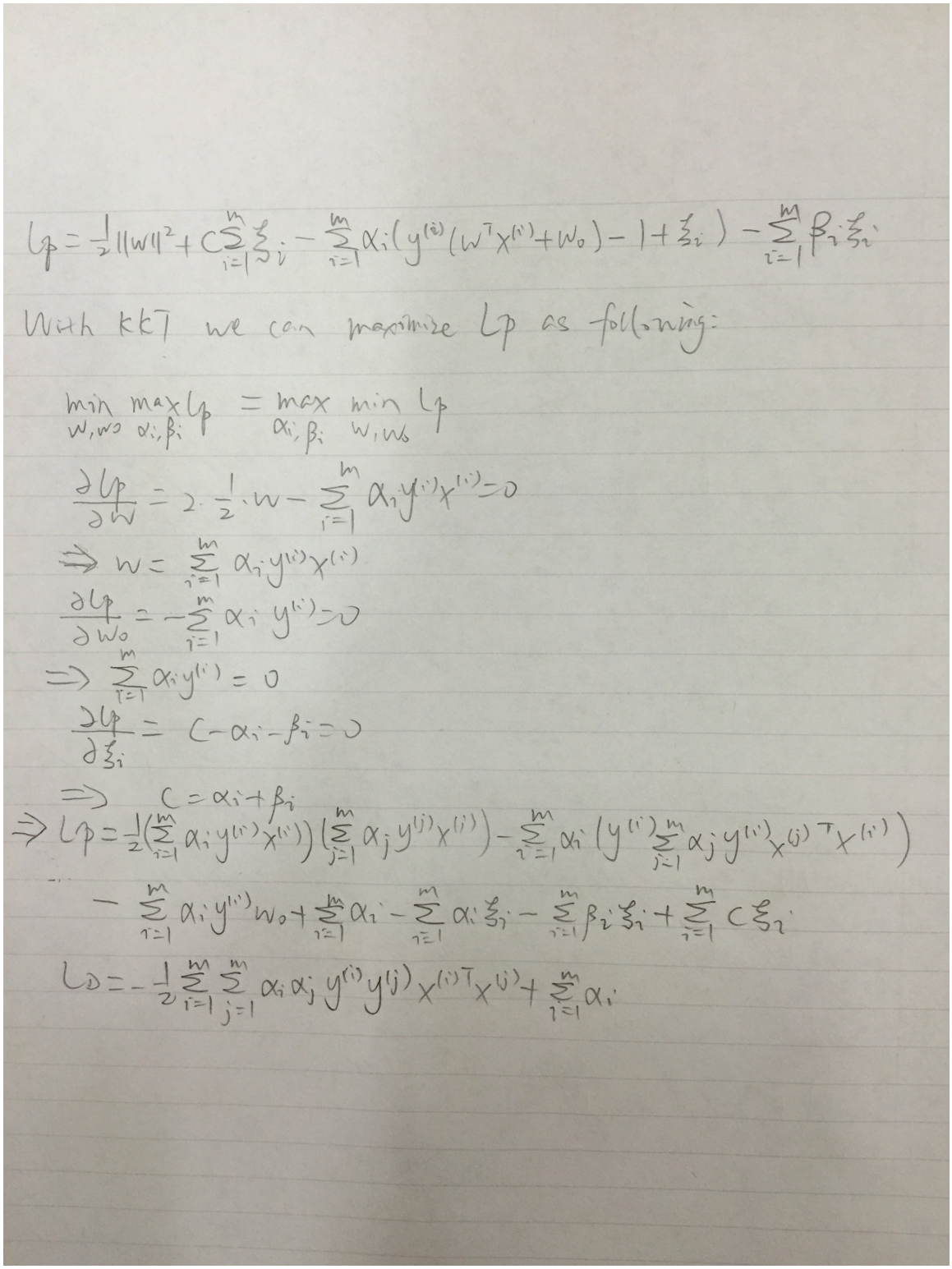


Then I applied the algorithm on the testing data with 10 cross validations, which gave the following confusion matrices:

```
> confusionMatrixLinear
     [,1] [,2]
[1,]   50    0
[2,]    0   50
> confusionMatrixLinearNotSeparable
     [,1] [,2]
[1,]   34   16
[2,]   16   34
```

Part3:

The procedures showing how to develop the expression of the dual L_D is as follow:



$$L_p = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\alpha_i\left(y^{(i)}(w^T x^{(i)} + w_0) - 1 + \xi_i\right) - \sum_{i=1}^{m}\beta_i\xi_i$$

With KKT we can maximize L_p as following:

$$\min_{w,w_0}\max_{\alpha_i,\beta_i} L_p = \max_{\alpha_i,\beta_i}\min_{w,w_0} L_p$$

$$\frac{\partial L_p}{\partial w} = 2\cdot\frac{1}{2}\cdot w - \sum_{i=1}^{m}\alpha_i y^{(i)} x^{(i)} = 0$$

$$\Rightarrow w = \sum_{i=1}^{m}\alpha_i y^{(i)} x^{(i)}$$

$$\frac{\partial L_p}{\partial w_0} = -\sum_{i=1}^{m}\alpha_i y^{(i)} = 0$$

$$\Rightarrow \sum_{i=1}^{m}\alpha_i y^{(i)} = 0$$

$$\frac{\partial L_p}{\partial \xi_i} = C - \alpha_i - \beta_i = 0$$

$$\Rightarrow C = \alpha_i + \beta_i$$

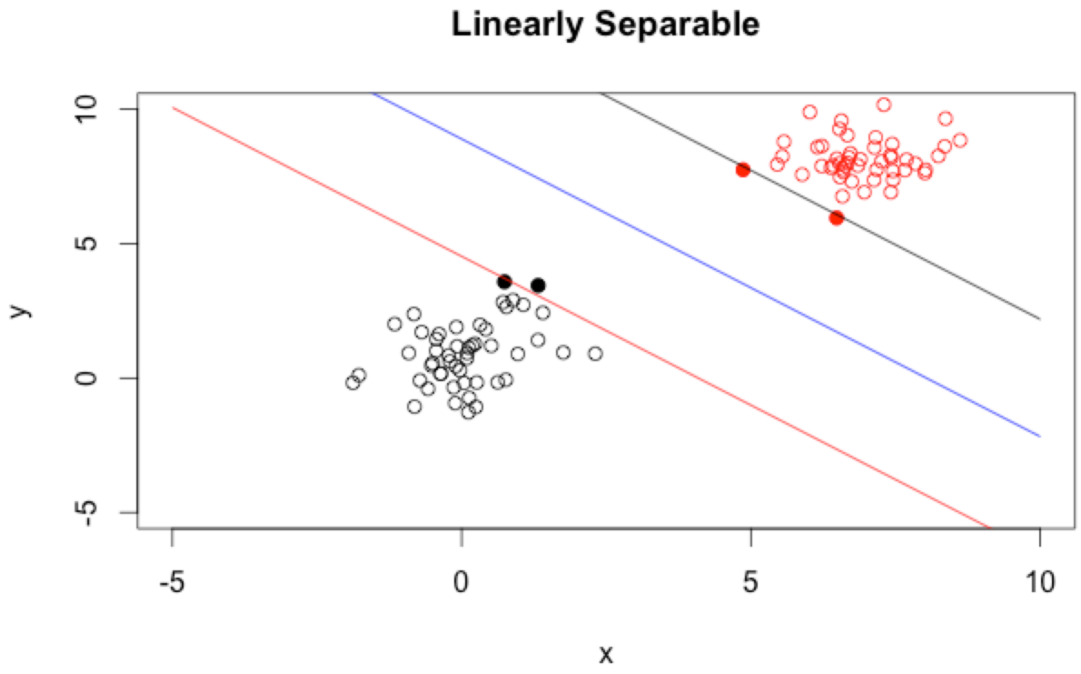
$$\Rightarrow L_p = \frac{1}{2}\left(\sum_{i=1}^{m}\alpha_i y^{(i)} x^{(i)}\right)\left(\sum_{j=1}^{m}\alpha_j y^{(j)} x^{(j)}\right) - \sum_{i=1}^{m}\alpha_i\left(y^{(i)}\sum_{j=1}^{m}\alpha_j y^{(j)} x^{(j)T} x^{(i)}\right)$$

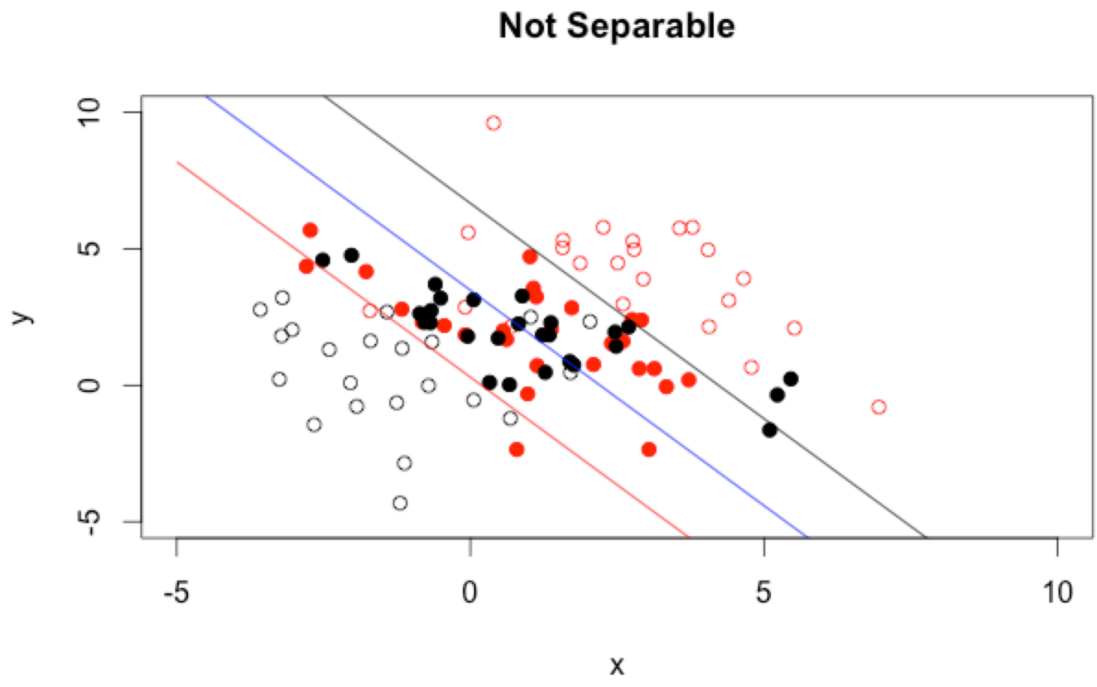
$$- \sum_{i=1}^{m}\alpha_i y^{(i)} w_0 + \sum_{i=1}^{m}\alpha_i - \sum_{i=1}^{m}\alpha_i\xi_i - \sum_{i=1}^{m}\beta_i\xi_i + \sum_{i=1}^{m} C\xi_i$$

$$L_D = -\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y^{(i)} y^{(j)} x^{(i)T} x^{(j)} + \sum_{i=1}^{m}\alpha_i$$

Part4:

I implement the support vector algorithm with soft margin. Basically, the algorithm is the same as that with hard margin and we just need to modify C to allow some errors (in order to achieve the soft margin). The data and the margins are plotted as follow:



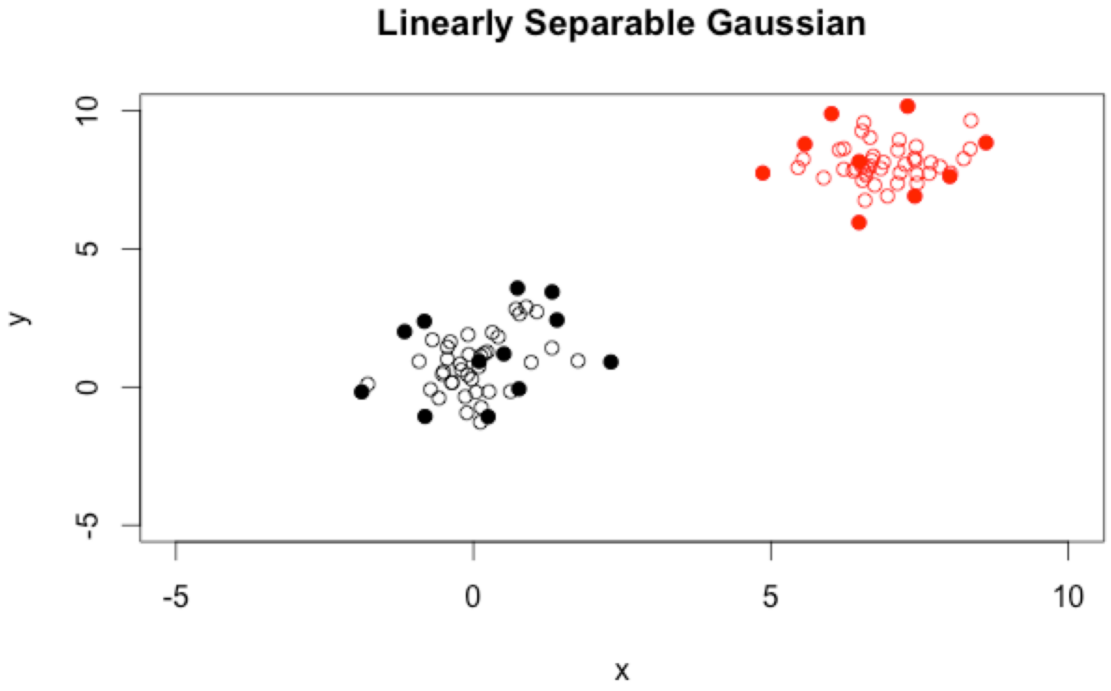


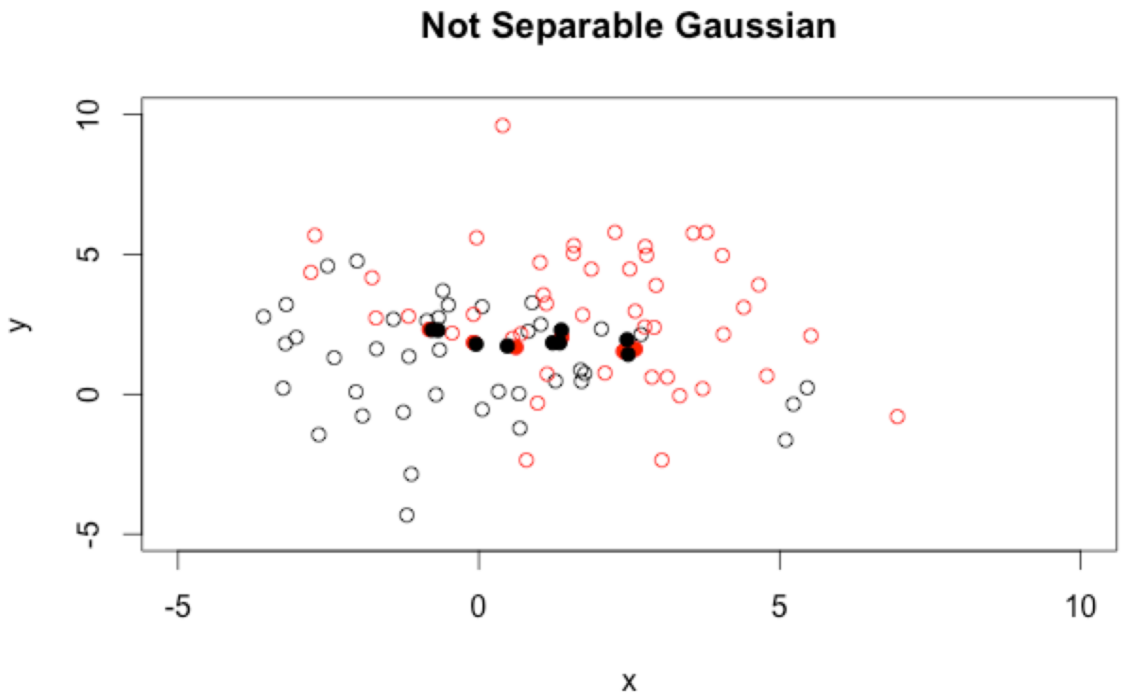
The confusion matrix is as follow:

```
> confusionMatrixLinear
     [,1] [,2]
[1,]   50    0
[2,]    0   50
> confusionMatrixLinearNotSeparable
     [,1] [,2]
[1,]   34   16
[2,]   16   34
```

Part5:

The result of Gaussian kernel based support vector machine algorithm is as follow:



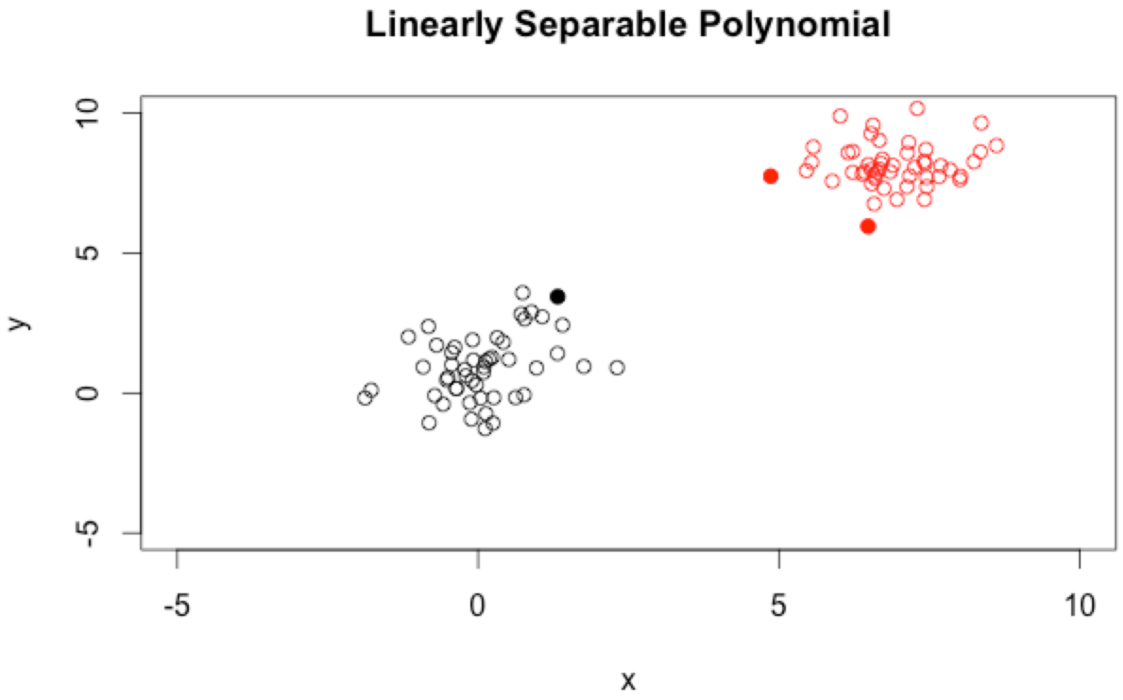


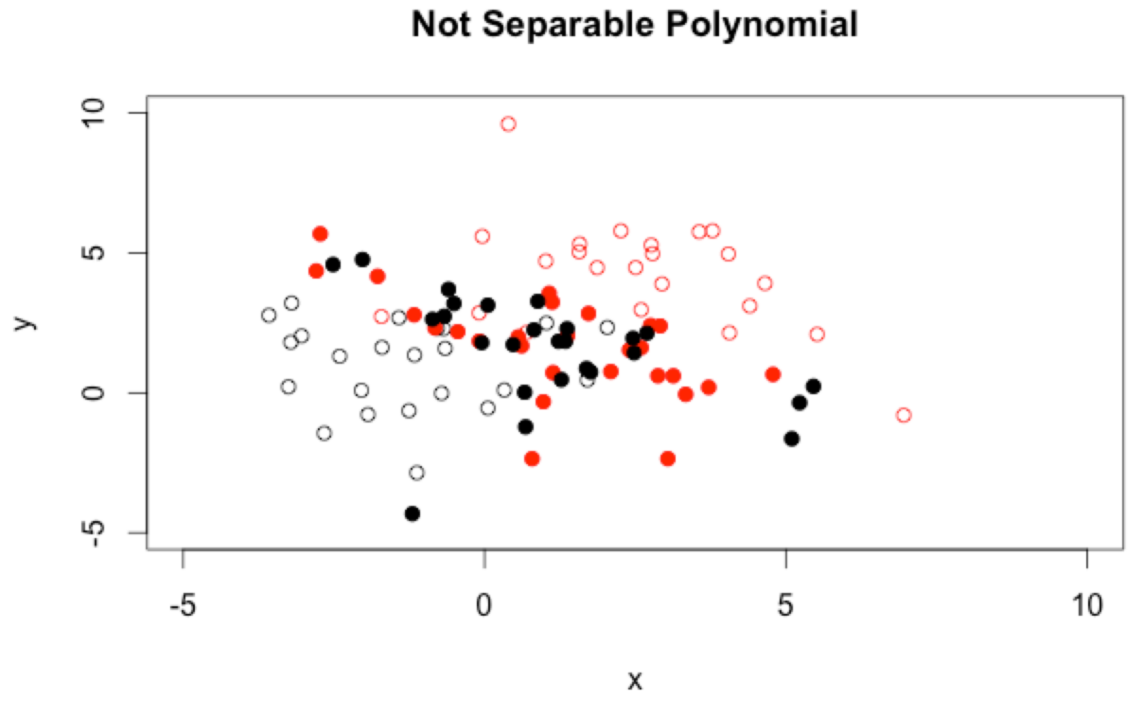
Note that the solid plots show the locations of the support vectors.

The confusion matrix is as follow:

```
> confusionMatrixLinear
     [,1] [,2]
[1,]   50    0
[2,]    0   50
> confusionMatrixLinearNotSeparable
     [,1] [,2]
[1,]   29   15
[2,]   21   35
```

The result of Gaussian kernel based support vector machine algorithm is as follow:





The confusion matrix is as follow:

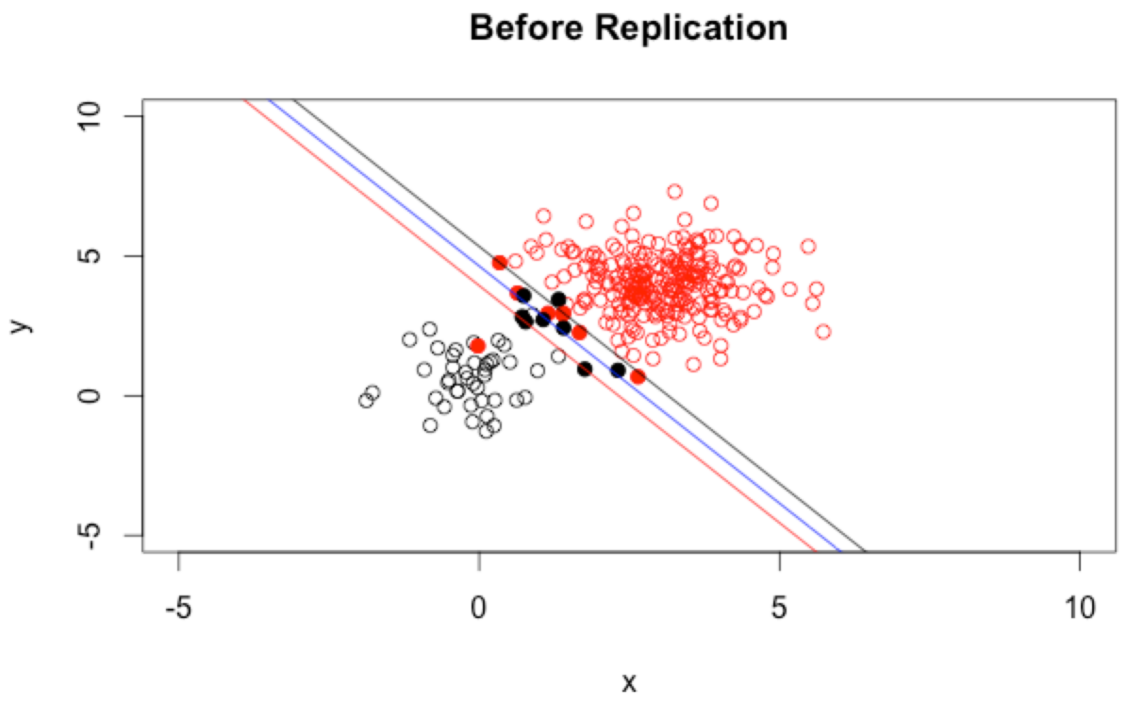
```
> confusionMatrixLinear
     [,1] [,2]
[1,]   50    0
[2,]    0   50
> confusionMatrixLinearNotSeparable
     [,1] [,2]
[1,]   34   16
[2,]   16   34
```

Part6:

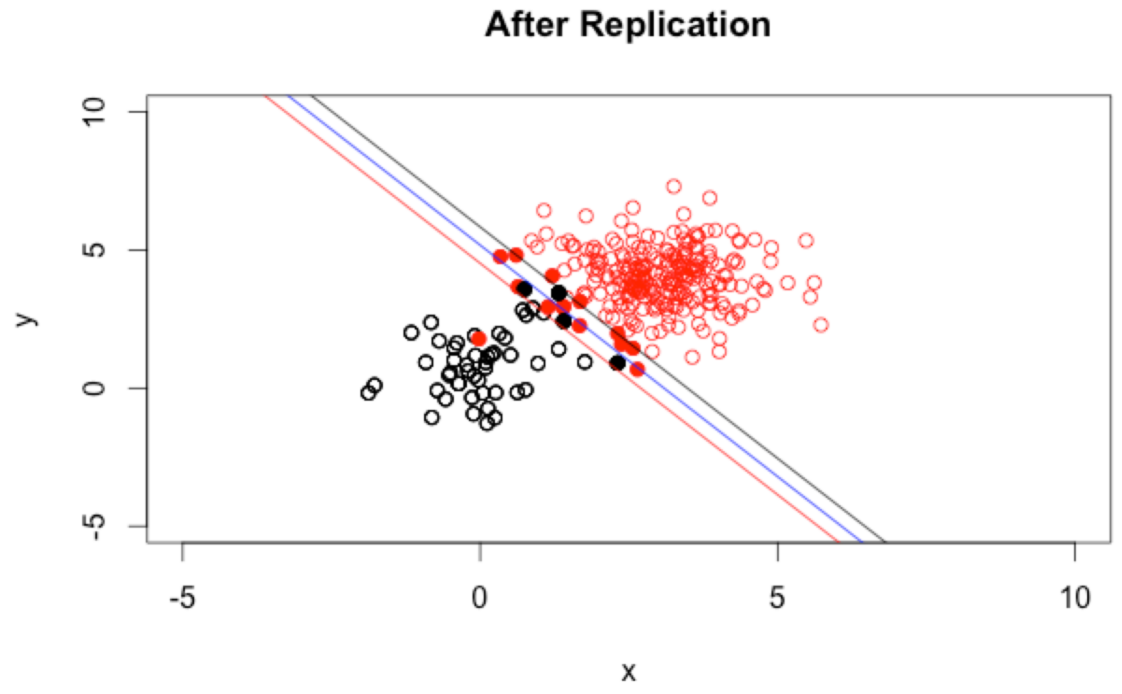
A solution to tackle the problem that one dataset is substantially larger than the other.

The idea is to replicate the smaller data set until the size becomes the same as the other one.

The result before the replication is as follow:



After the replication, it becomes:



We can see that the margins become much better after the replication. Therefore, the solution works well in tackling the problem.