## 2.5 Support Vector Machine (SVM)

SVM algorithm is a method to find an optimal hyperplane (straight line for only 2 features) to classify the training data , extend to a pattern that is not linearly separable by transformations of original data to map into new space – the Kernel function [16]. As the name indicates, we need to find the support vectors first. Support vectors are the data points lie closest to the decision surface or hyperplane [16]. These data points are the most difficult points to be classified. The optimal decision surface or hyperplane fully depends on these support vectors. As shown in figure 2.5.1 left, there are two groups of training data, Red and Blue. There are several ways to draw the decision boundary (in dotted line). It could be an infinite number, but not all of them are suitable and optimal. In figure 2.5.1 right, this is the optimal hyperplane we can find based on these training data. The decision boundary is in solid line. How can we find it? SVM maximises the margin around the separating hyperplane. Those data

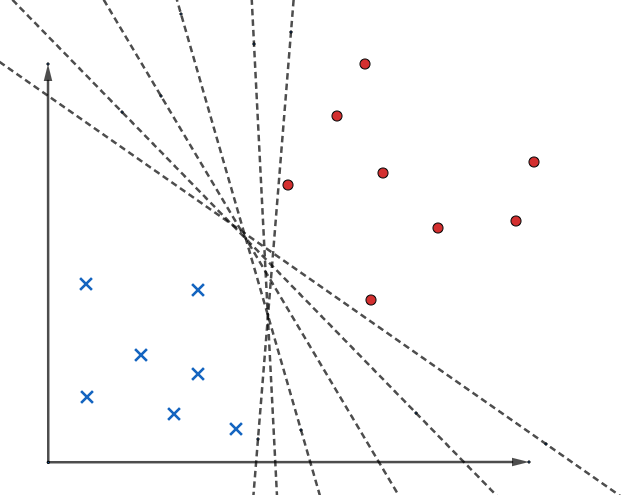
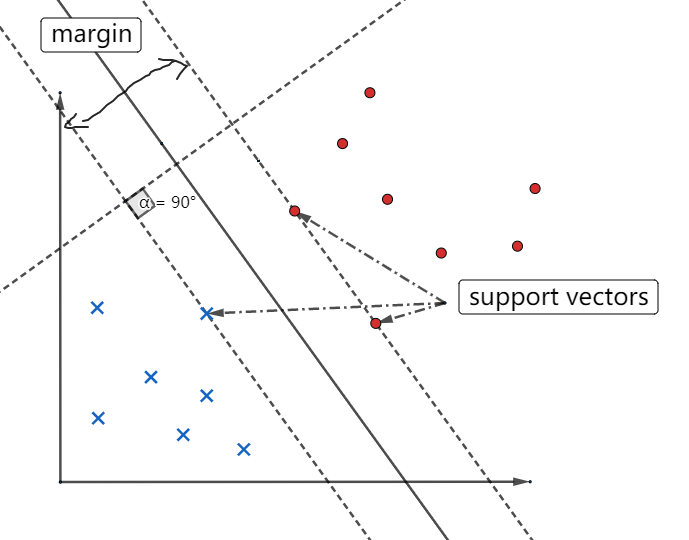
 

Figure 2.5.1: several decision boundaries

points lie on the dotted line are the support vectors, they just touch the boundary of the margin. Use decision rule (equation (2.5.1)) to determine the label of a new sample. If the result is greater than zero, this sample will be positive, negative for the result is less than zero. is the weight vector, and one element corresponding to each feature of the sample. is the sample in vector form. b is the bias.

We use the optimisation of maximising the margin to reduce the elements of weight vector that are nonzero to just a few that correspond to the important features that ‘matter’ in deciding hyperplane. These nonzero weights correspond to the support vector.

In figure 2.5.2 define the H1 and H2 are planes. We should maximise the width of the margin.

is the label of sample, it could be true or false, 1 or -1. When = +1, . For = -1, . These two conditions can be combined in (equation) 2.5.2 as a constrain.

Because the margin needs to be maximised, the distance between and should be maximised. The distance from point to line is . According the distance equation, we can represent the distance between and in the equation (2.5.3).

From the equation (2.5.3), we can know the width of margin. Because distance between H0 and H1

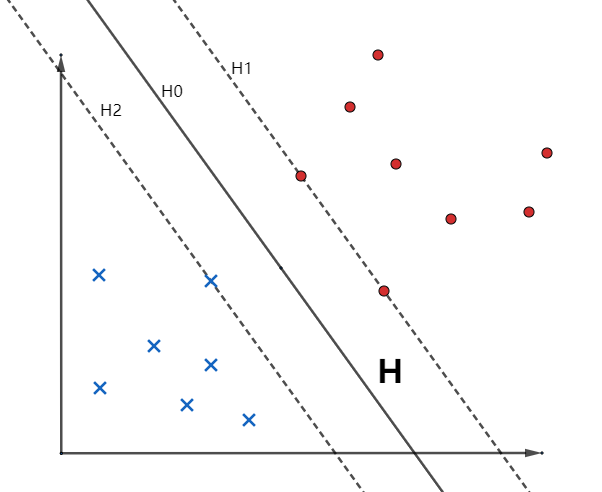


Figure 2.5.2

is equal to the distance between Ho and H2. Thereby, the width d is . If we want to maximise d, consequently, we need to minimise with constrain in equation (2.5.2). So, this is a constrained optimisation problem. It can be solved by the LaGrangian multiplier method. We can express this optimisation problem in equation (2.5.4). is for convenience, it does not affect the result, determines how fast the similarity metric decrease

Because we are dealing with minimisation, so the slope at that point should be zero, based on this condition, we can have two more equation, (2.5.5) and (2.5.6).

These two equations can be substituted into (equation) 2.5.4.

The equation (2.5.4) indicates this equation is the primal equation of this optimisation problem. By introducing the LaGrangian multiplier method into this problem, we will actually solve the optimisation problem by now solving for dual of the original problem [16]. Instead of minimising over and b, subject to constraint involving ’s, we can maximise over subject to the relation obtained previously for and b. Now, equation (2.5.7) can be converted as equation (2.5.8).

Notice that all we have are the dot products of and , if we take derivative with respect to and set it equals to 0 [16], we get the solution of , so we can solve : .

After we use the optimisation method to solve this problem, the LaGrangian Multiplier could be calculated. Once we get , weight vector can be determined, by choosing one element in , bias b can be found.

In summary, the input dataset T contains samples , xi has n features, is the label of each sample. The Lagrangian multiplier will be the output by solving the optimisation problem. Then calculate the weight vector through equation (2.5.5), also the bias b. The last step to build this model is to find the equation of hyperplane (). The method to predict new sample is just simply substituting the value of , according to the sign of the result to decide, as shown in equation (2.5.9).

In this project, we will use the simplified version of the Sequential Minimal Optimisation (SMO) to solve the optimisation problem. The details of SMO will be discussed in Chapter4 with the implementation based on the training data

## 4.3 Implementation of SVM [16]

There are 303 samples in the training data, each sample has 13 features. At the beginning of building a model, we should normalise the training data according to equation 4.1. All the calculations are in matrix form. X and are 303 13 matrices, is a matrix of the mean value of each feature. is also a matrix which contains the maximum value of each feature.

In this project, I used the simplified version of SMO to build SVM model. Because of the size of the training data, a simplified version could perform well in this project.

The Karush–Kuhn–Tucker (KKT) conditions (equation 4.2) can be used to validate the convergence of the optimal point. These three equations represent data points with correct classification, data points which are in the hyperplane and data points which are the support vectors lie on the boundary of the hyperplane.

Once the KKT conditions are satisfied, that will be the solution to this optimisation problem. SMO will do iteration to satisfy the KKT conditions. SMO selects two parameters, and , because of another constrain in equation 3.8, and optimises the objective value jointly for both these ’s [16]. Then adjusting bias b value under new parameters. It repeats the process until parameters converge. The difference between full version and a simplified version is the selection of parameters, the full version will try all possible pairs of parameters, there are possibilities. In the simplified version, we just iterate over all , from = 1 to m, and j will be a random number from 1 to n, except equals to .

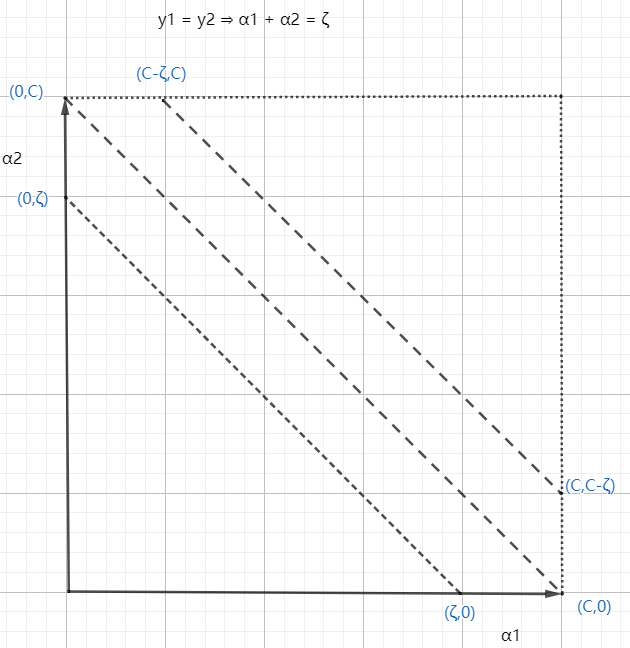
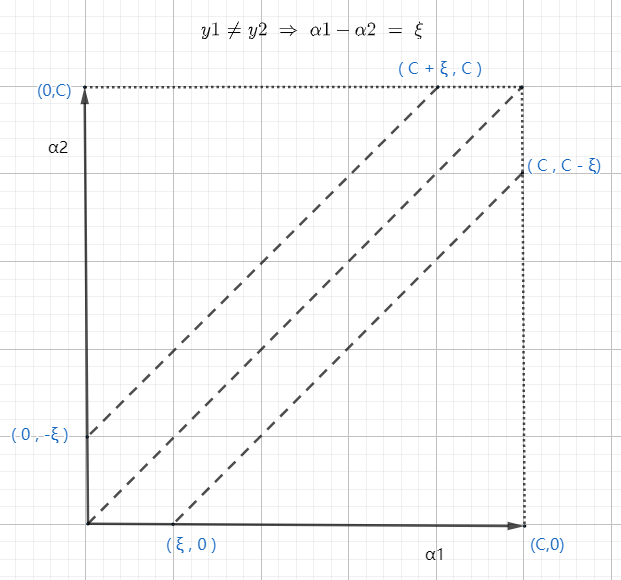


Figure 4.1

For example, now we select and , the constrain in equation 3.8 will becomes into equation 4.3.

There are two conditions we must consider, and . The figure 4.1 shows two conditions. From the diagram, we can find there is a range L < < H (equation 4.4).

The calculation of is given by equation 4.5.

Where

is the error between the result from equation 4.6 and the true label which is y value in training data. The new from equation 4.5 will be modified by equation 4.7, because of the range of

after we find the optimised the optimised can be calculated by equation 4.8

Finally, step is to computer threshold value b. We select b value such that KKT conditions is satisfied for and training sample. After optimisation, the new threshold value b can be computed by equation 4.8.

Where:

We can now predict a new sample by using this model. Bringing the value of each feature into the hyperplane function. If the result is greater than 0, this sample is a ‘positive’ sample, or it is ill. If result is less than 0, then this is a ‘negative’ sample, or it is healthy.

All the equation appeared in this section have been converted into Matlab code to build this SVM model.

## 5.2 SVM result

To test how accurate this SVM model is, I brought the training data back into the SVM model. To compare the predicted label of each sample with the true label I can calculate the accuracy. The accuracy is 83.5% which is slightly lower than the results from the two articles in section 2.2. It may be due to pre-processing of the data, like Principal Component Analysis (PCA). Or they used kernel function in the model.

Figure 5.1 shows the heatmap based on my training data. 0 and 1 mean healthy and ill, respectively. The x-axis is the true label; the y-axis is the predicted label. The misclassification occurs at (x=0, y=1) and (x=1, y=0). Another measurement is the critical error, it means classified ill people into the healthy group. It should be as lower as possible. In this model, the critical error is 3.96%.

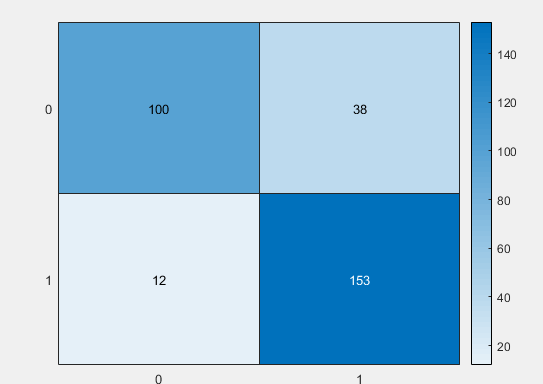


Figure 5.1