STAT305

Summer Research Project

2006– 2007

The Application of the Support Vector Machine to the Classification

Olivia Son

Department of Mathematics and Statistics
University of Canterbury

STAT305

The application of the Support Vector Machine to the Classification

Olivia Son

Deptartment of Mathematics & Statistics

The University of Canterbury

Abstract

A classification technique, known as Support Vector Machine (SVM) is applied to tobacco data from the SYFT technologies Ltd. The SVM is used to classify illegal from others. Decision tree is performed prior to SVM and these two classification methods are compared by misclassification rate for the accuracy of classification performance.

Introduction

During the past few years, the number of classification techniques has increased with the rapid growth of technology. Amongst all these new techniques, one that has become increasingly successful in discriminative approaches to pattern classification as well as in regression problems is a technique called Support Vector Machine (SVM).

The foundation of Support Vector Machines (SVM) has been developed by Vapnik in 1995 and is increasingly gaining reputation due to its many attractive features.

The project is created from a set of tobacco data that is provided from the SYFT technologies Ltd. The SYFT Ltd creates a mass spectrometer which analyses the gas composition and volatile compounds to detect illegal shipment. The data are divided into four categories, 'legal', 'illegal', 'general cargo', and 'possible'. 'Legal' contains legal tobacco shipment, 'illegal' contains illegal tobacco and 'general cargo' contains neither of illegal and legal shipment. The last class 'possible' is the data that do not belong to any of other classes. The main task of the project is to detect 'illegal' from other classes. The way to approach this is performing the classification that enables to separate illegal from other classes.

Initially, it was assumed that illegal and legal are not well separated as illegal would tend to be hidden in legal. However, by looking at the scatter plot of the data, it is found that the illegal is well separated from legal.

In order to separate illegal and general cargo which are not well separated, there is a need to apply a good classification technique.

Another classification technique called ‘decision tree' was attempted first. While decision tree has a number of properties, such as not requiring any prior assumptions resembling a type of probability distribution of the class and other attributes, it is also limited in a way that it can only perform the classification linearly along the direction of the axis. In order to cope with this problem, a classification method which is more flexible and allows non-linear classification was required

Firstly, the classification technique called decision tree has been attempted. While decision tree has a number of properties, such as not requiring any prior assumptions resembling a type of probability distribution of the class and other attributes, it is also limited in a way that it can only perform the classification linearly along the direction of the axis. In order to cope with this problem, a classification method which is more flexible and allows non-linear classification was required. Therefore, Support Vector Machine (SVM) is chosen as such a method and the project is designed to apply the Support Vector Machine to classification on to the tobacco data.

The performance of classification is evaluated by misclassification rate. It is predicted that the SVM will perform better classification than the method of decision tree due to its property especially ability to classifying nonlinearly.

In the following introduction, the method of SVM will be dealt in more details in methodology section. How the SVM being conducted is dealt in application section, and the result of the application is reported. Finally, the conclusion and the future work are reported at the end of the report.

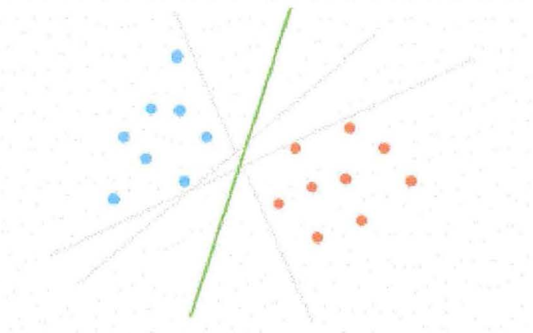
Methodology

Support Vector Machine (SVM)

Support Vector Machine was invented by Vladimir Vapnik. It is a method to create a model from a set of training data. It illustrates the decision boundary by using a subset of the training examples, known as Support Vectors. The reason this method is used for the project is because SVM can classify data non-linearly and has the ability to handle large feature spaces since complexity does not depend on the dimensionality of the feature space. Moreover, the sparseness of solutions when dealing with large data sets is only because the support vectors are used to find the separating hyperplane.

Linear Classifier

To illustrate the basic idea of SVM, the simplest case of SVM of linear classifier is introduced.

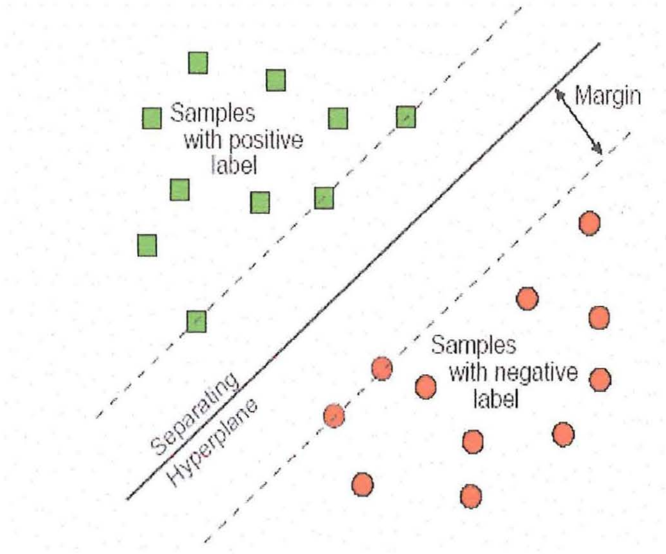


Optimal Separating Hyperplane

Here, there is an infinite number of possible linear classifiers (hyperplanes) that can separate the data. Even though all possible linear classifiers have misclassification rate of zero, it does not mean that all hyperplanes will work equally well. Thus, amongst all the possible linear classifiers, we need to find the hyperplane that classifies the data in the best way. In order to find the optimal separating hyperplane, the concept of a maximal margin hyperplane is used. The advantage of using maximal margin is that it ignores other training data as it only needs support vectors which defines the hyperplane and works very well empirically.

Maximising Margin

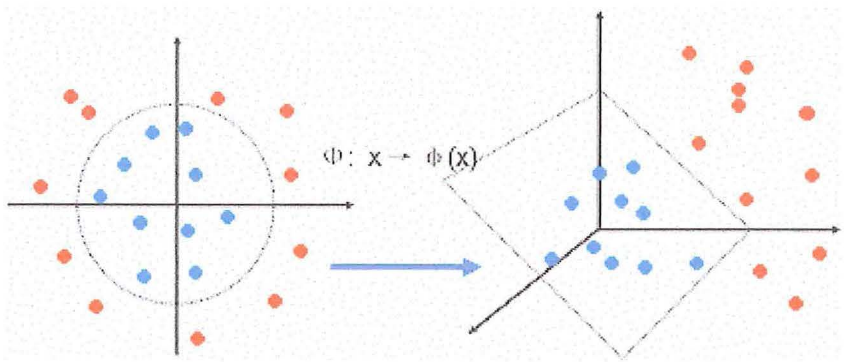
Margin of a linear classifier is the width that the boundary could be increased by before hitting the nearest datapoints. Support vectors are these nearest datapoints that the margin pushes up against.



Hyperplane with small margins is more flexible and is able to fit many training data. Thus, the smaller the margins of the hyperplane are, the greater the capacity. However, according to the SRM principle, the margins with higher capacity have the greater generalising error. To minimize the generalisation errors, the hyperplane must have the maximum margin. Hence, the hyperplane that has the maximum margin has the minimum generalisation error, which means that the hyperplane with the maximum margin is the optimal separating hyperplane.

Support vector to feature Space

So far, the case of linearly separable data is considered. However, the primary reason for using the SVM for this project is to enable SVM to separate the data non-linearly. When the data is not linearly separable, SVM maps the data into a higher dimensional space called 'feature space' by using functions known as kernels. Then SVM creates a separating hyperplane in the feature space to classify the data in the space. This kernel transferring function is powerful in that it does not need to represent the space explicitly all we need is simply define the kernel function



There are infinitely many kernels functions, but the four main types that are commonly used in Support Vector Machines are:

Linear: u'*v
Polynomial: (gamma*u'*v + coef0)^degree
Radial basis function(RBF): exp(-gamma*|u-v|^2)
Sigmoid (feed-forward neural network): tanh(gamma*u'*v + coef0)

A polynomial is a popular method for non-linear modeling. However, RBF is most popular kernel amongst four functions due to the number of advantages.

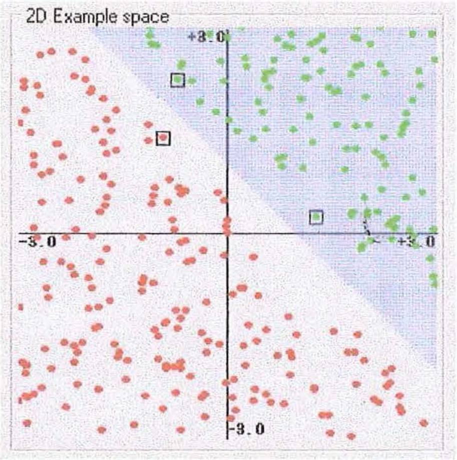
Firstly, RBF has a finite number of responses across the x-axis. It also enables to transfer original data into a higher dimension feature space non-linearly, so it can deal with non-linear relationship of the class and attributes. The RBF kernel transfers samples non-linearly into a higher dimensional space, so it can deal with the case where the relation between the class labels and the attributes is non-linear unlike linear kernel. Furthermore, for certain parameters, the sigmoid kernel behaves like RBF and also REF kernel has less hyperparameters than the polynomial kernel.

Thus it is recommended that RBF kernel is the most suitable model to use.

Now Consider how each kernel models performed as classifier. Each of kernels model demonstrates classification of non-linear data which is the result of transformation by its function.

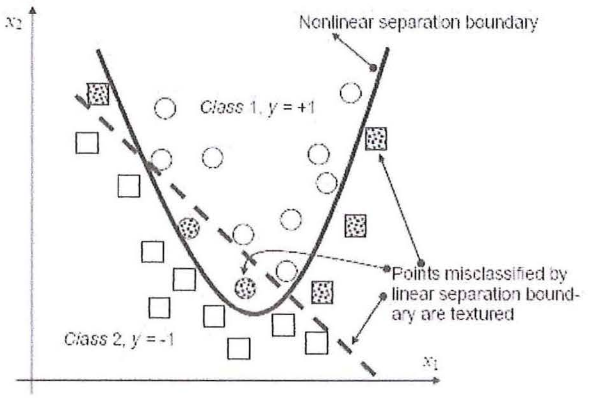
Kernel functions supported by DTREG:

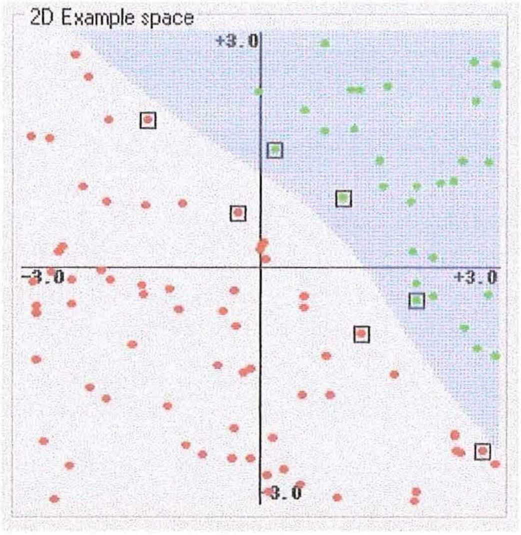
Linear: u'*v



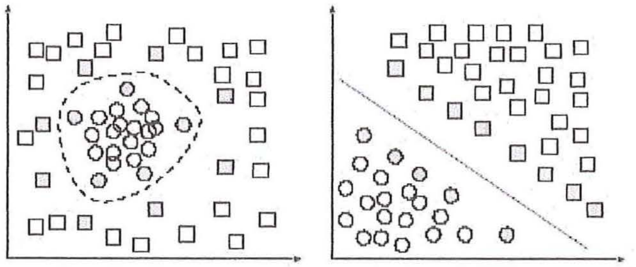
(This example was generated by pcSVMdemo.)

Polynomial: (gamma*u'*v + coef0)^degree


x_2
Nonlinear separation boundary
Class 1, y = +1
Points misclassified by linear separation boundary are textured
Class 2, y = -1
x_1



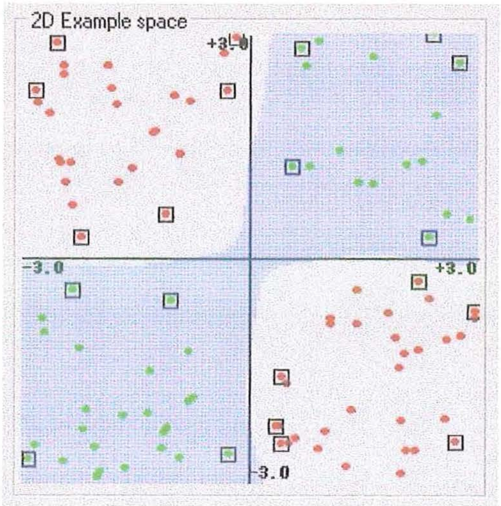
Radial basis function: exp(-gamma*|u-v|^2)

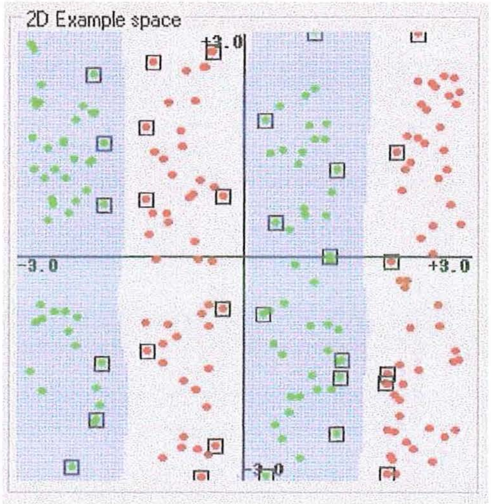


(a) Radial Basis Function

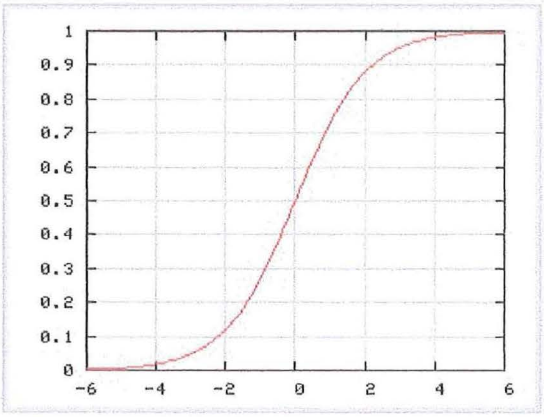
(b) RBF mapping

Separable classification with Radial Basis kernel functions in different space. Left: original space. Right: feature space.

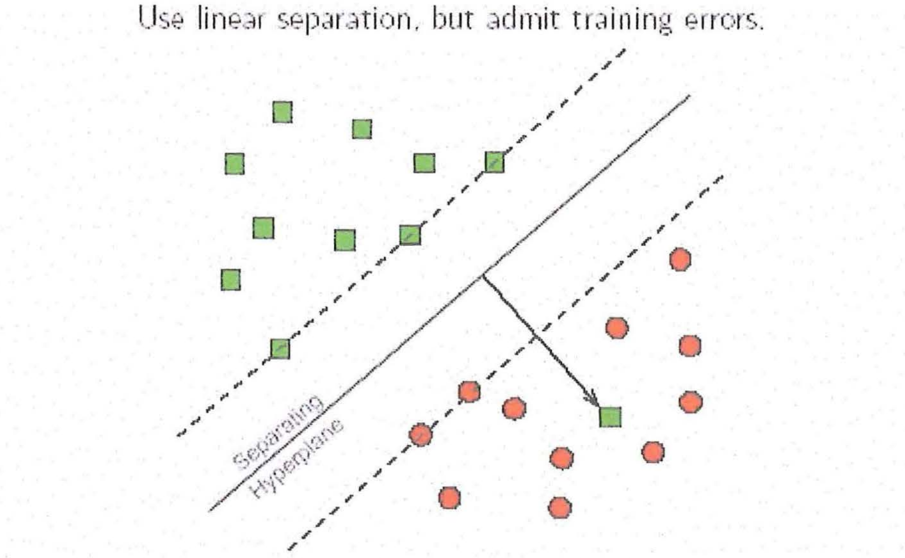




Sigmoid (feed-forward neural network): tanh(gamma*u'*v + coef0)



Even though, SVM is used to produce a hyperplane that ideally separates the feature vectors completely into two non-overlapping groups, perfect separation may not be possible, or it may produce a classifier with a large number of feature vector dimensions that the model does not generalize well to other data, which is known as over fitting.


Use linear separation, but admit training errors.
Separating Hyperplane
Penalty of error: distance to hyperplane multiplied by *error cost C*.

To allow some flexibility in separating the data, SVM has a cost parameter called C, which enables the balance between training errors and rigid margins. It creates a soft margin which allows some misclassifications. If the value of C increases, then the cost of misclassification also increases and leads to a more accurate model which may not generalize well.

Another parameter that is used in kernel functions is gamma which is the width of kernel function.

Default SVM used in R

To apply SVM to the tobacco data, the package of e11071 in the software R is used. The default setting of SVM used in R is as follows. Firstly, the kernel function used is RBF kernel and the parameter gamma is defined by the formula gamma = if (is.vector(x)) 1 else 1 / ncol(x). The parameter C is 1 in default. The type of SVM used is nu-classification. Resubstitution method is used as misclassification rate.

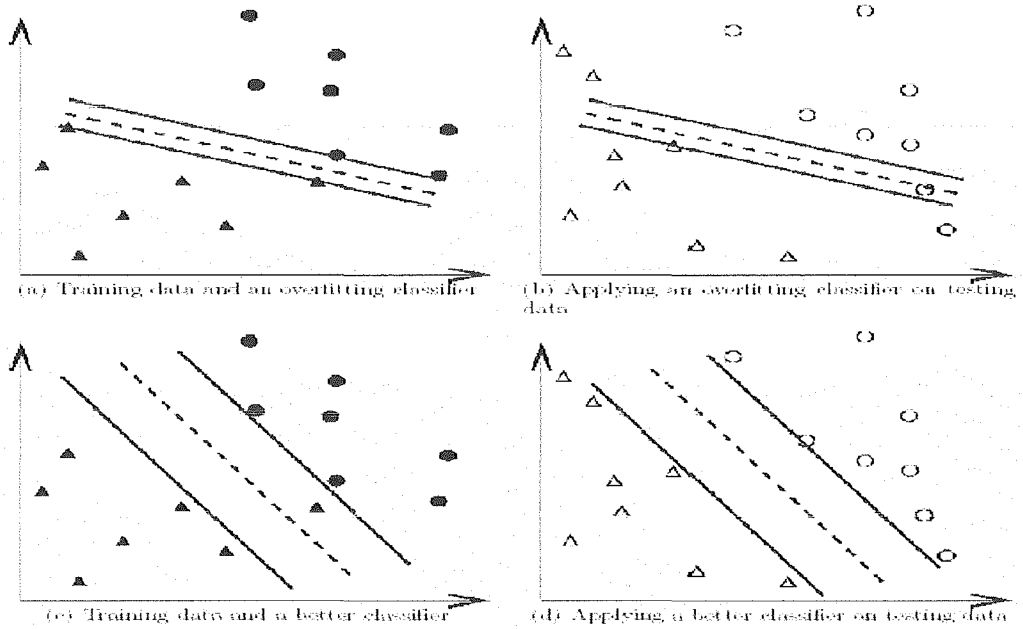
Misclassification Rate

The performance of the classification work is evaluated by Miscassification Rate. In the Misclassification rate, there are two types, Resubstitution and Cross Validation Rate.

As mentioned before, traing set of data is used to define the classifier then the testing set of data is used to assess how well the defined classifier performs. For the Resubstitution, the same data is used twice as training and testing set. Thus resubstitution has a high risk of underestimating the misclassification rate. However, Cross-validation is the leave-one-out form in that it removes each observation in turn to define the classifier. It then tests if the leave-one-out classifiers correctly classifies the removed observation.

The diagram below, contrasts the resubstitution and Cross-validation. Testing accuracy the classifier in Figures 1(a) and 1(b) is not sensible because it over-fits the training data. Training and testing data in

Figure1(a) and 1(b) as the training and validation sets in cross-validation, the accuracy is not also sensible. However, classifiers in 1(c) and 1(d) which do not over-fit the training.



(a) Training data and an overfitting classifier

(b) Applying an overfitting classifier on testing data

(c) Training data and a better classifier

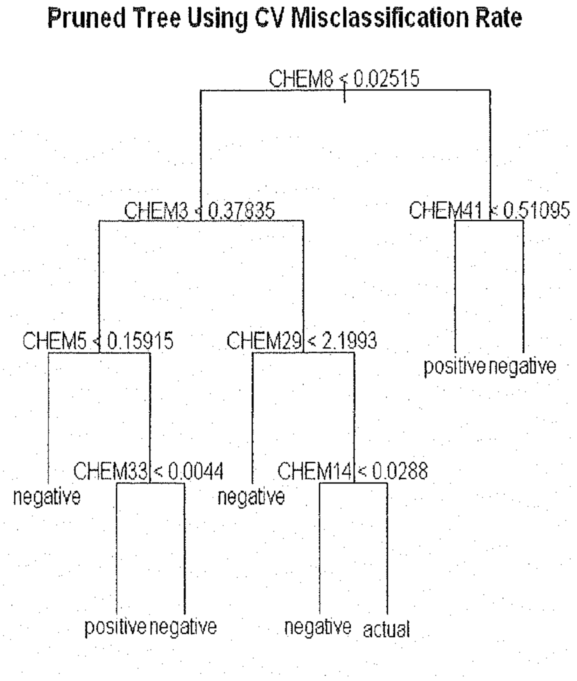
(d) Applying a better classifier on testing data

Figure 1: An overfitting classifier and a better classifier (● and ▲: training data; ○ and △: testing data).

Thus it is recommended that cross validation is better to use for mislclassification error rate.

Application

The Summary of Decision Tree



Classification Matrix				
datamcpred	Illegal	General Cargo	Legal	Possible
Illegal	11	3	0	0
General Cargo	0	442	2	0
Legal	0	1	89	0
Possible	0	0	0	0

Before attempt SVM, the Decision tree method is tried for this data. The actual which is illegal is well classified with the misclassification rate of 0.19.

It is shown that the misclassification error rate is 0.0091 which is very small.

Demonstration of SVM on IRIS data

Before conducting the SVM to the tabacco data, the SVM example on Iris data is observed. The iris data is an established data that is used for demonstrating how to perform classification algorithms. This iris data have four features of the iris, and the aim is to classify the groups of iris based on these four features. In order to visualize the SVM performance, only two features of petal length and the petal width are taken into account since they contain the most information about the class.

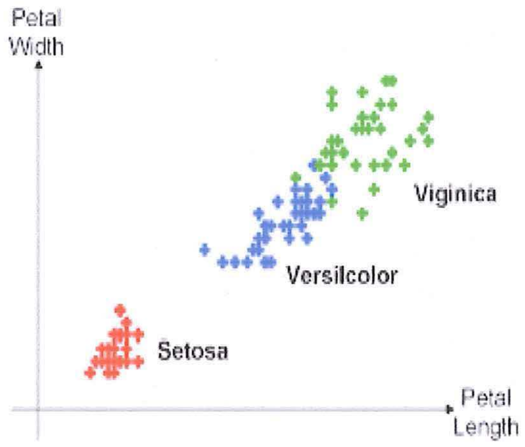
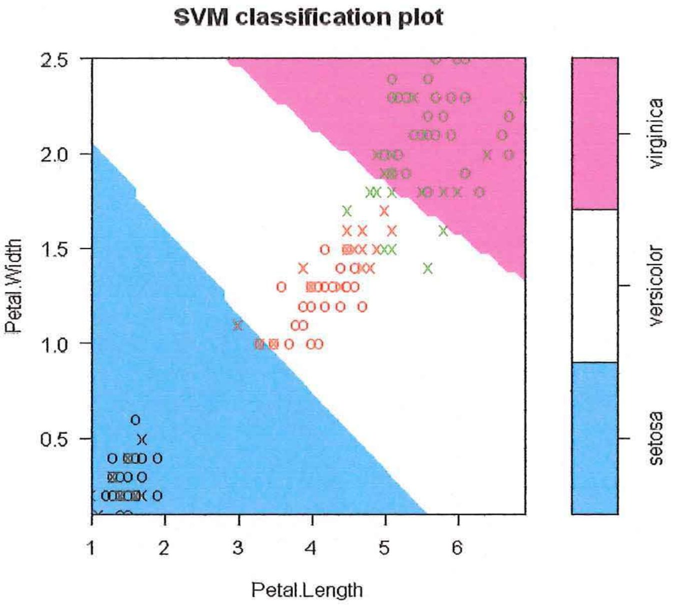


FIGURE 4.1: Iris data set



As it is illustrated in Figure 4.1, the Versilcolor and Setosa classes can easily be separated by a linear classifier, whereas, the separation between Versilcolor and Viginica is not simple. However, it illustrated that the SVM performs well as there is only misclassification of 2 between versicolor and verginica

```
pred          setosa versicolor virginica
  setosa          50          0         0
  versicolor       0         48         2
  virginica        0          2        48
```

Then this method is tried on the Tobacco Data

Default application of SVM to tobacco data

Misclassification rate shown,

Classification Matrix				
pred	Illegal	General Cargo	Legal	Possible
Illegal	8	0	0	0
General Cargo	13	548	78	0
Legal	0	0	13	0
possible	0	0	0	0

Misclassification rate

= #misclassified / #total
= (13+ 78)/(8+ 13+ 548+ 78+ 13)
= 91/660
= 0.138

Now actual which is illegal did not have large misclassification rate with others, however, we found that there is high misclassification rate between legal and general cargo, thus even the prime aim of the research is to classify illegal from others, we need to improve the SVM.

In order to improve the SVM, it is suggested to change the parameter of gamma and cost or attempt other kernel functions.

The value of Default gamma is 0.02, so the values of gamma 0.2 and 0.002 are attempted. However, no improvement was observed by the same classification matrix, hence there is no change in misclassification rate. The only support vector has changed from 230 to 330 when gamma of 0.2 is attempted and to 331 when gamma of 0.002 attempted.

By default, the value of the cost parameter is 1, so the values of 10 and 100 are attempted. However, no improvement was observed again by the same classification matrix, hence the same misclassification rate. The only support vector has changed from 230 to 330 when cost of 10 and 100 attempted.

Result

The misclassification rate of 0.0091 for decision tree and 0.138 for SVM indicate that Support Vector Machine does not perform as well as the decision tree for this set of data 0.138.

Changing the default settings of gamma to 0.2 and 0.002 and cost to 10 and 100 do not improve the performance of the Support Vector Machine for the Classification.

Future Work

There are a number of suggestions to improve the SVM for classification. The accuracy of an SVM model is largely dependent on the selection of the model parameters. The attempt of changing the parameters of gamma and cost in a wide range of the value and observing its affect on misclassification matrix will enable to find the Optimal Parameter Values.

Secondly, it is suggested that using the Cross-Validation method as misclassification rate will test the accuracy of classification performance more precisely.

Lastly, even though the REB is the most well-know kernel, it is also suggested to attempt other kernel functions for this data, since another kernel function maybe possible to perform better than REB.

Reference

Pg.150-151, 168-169, 256-257 Tan, P. (2006). Introduction to Data Mining. Boston: Pearson Education Inc.

Pg.371-389 Hastie, T. (2001). The Elements of Statistical Learning; Data Mining, Inference, and Prediction.

http://www-users.cs.umn.edu/~kumar/dmbook/ch4.pdf

http://www.iro.umontreal.ca/~pift6080/documents/papers/svm_tutorial.ppt

http://www.ecs.soton.ac.uk/~srg/publications/pdf/SVM.pdf

http://www.dtreg.com/svm.htm

http://en.wikipedia.org/wiki/Support_vector_machine

http://www.iro.umontreal.ca/~pift6080/documents/papers/svm_tutorial.ppt

http://www.csie.ntu.edu.tw/~cjlin/libsvm

http://www.csie.ntu.edu.tw/~cjlin/papers/guide/guide.pdf

http://www.statsoft.com/textbook/stsvm.html

Appendices

```
> library(e1071)
Loading required package: class
## classification mode
# default with factor response:
> model <- svm(category ~ ., data = datain)
> x <- datain[,2:51]
> y <- datain[,1]
> print(model)
Call:
 svm(formula = category ~ ., data = datain)
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.02
Number of Support Vectors:  236
> summary(model)
Call:
 svm(formula = category ~ ., data = datain)
Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.02
Number of Support Vectors:  236
 ( 21 125 90 )
Number of Classes:  3
Levels:
 actual negative positive possible
# test with train data
> pred <- predict(model, x)
# (same as:)
> pred <- fitted(model)
```

```
# Check accuracy:
> table(pred, y)
          y
pred        actual negative positive possible
  actual        8       0       0       0
  negative     13     548      78       0
  positive      0       0      13       0
  possible      0       0       0       0
```

```
> ## Default S3 method:
> svm(x, y = NULL, scale = TRUE, type = NULL, kernel =
+  "radial", degree = 3, gamma = 0.2,
+  coef0 = 0, cost = 1, nu = 0.5,
+  class.weights = NULL, cachesize = 40, tolerance = 0.001, epsilon = 0.1,
+  shrinking = TRUE, cross = 0, probability = FALSE, fitted = TRUE, subset, na.action =
na.omit)

Call:
svm.default(x = x, y = NULL, scale = TRUE, type = NULL, kernel = "radial",
    degree = 3, gamma = 2, coef0 = 0, cost = 1, nu = 0.5, class.weights = NULL,
    cachesize = 40, tolerance = 0.001, epsilon = 0.1, shrinking = TRUE,
    cross = 0, probability = FALSE, fitted = TRUE, subset, na.action = na.omit)


Parameters:
   SVM-Type:  one-classification
 SVM-Kernel:  radial
       gamma:  0.2
          nu:  0.5

Number of Support Vectors:  331

> # test with train data
>   pred <- predict(model, x)
> # (same as:)
> pred <- fitted(model)
>
> # Check accuracy:
>   table(pred, y)
          y
```

```
> # S3 method for class 'formula':
> svm(formula, data = NULL, subset, na.action =
+  na.omit, scale = TRUE)
Error in as.vector(x, mode) : cannot coerce to vector
> ## Default S3 method:
> svm(x, y = NULL, scale = TRUE, type = NULL, kernel =
+  "radial", degree = 3, gamma = 0.002,
+  coef0 = 0, cost = 1, nu = 0.5,
+  class.weights = NULL, cachesize = 40, tolerance = 0.001, epsilon = 0.1,
+  shrinking = TRUE, cross = 0, probability = FALSE, fitted = TRUE, subset, na.action =
na.omit)

Call:
svm.default(x = x, y = NULL, scale = TRUE, type = NULL, kernel = "radial",
    degree = 3, gamma = 0.002, coef0 = 0, cost = 1, nu = 0.5, class.weights = NULL,
    cachesize = 40, tolerance = 0.001, epsilon = 0.1, shrinking = TRUE,
    cross = 0, probability = FALSE, fitted = TRUE, subset, na.action = na.omit)
Parameters:
   SVM-Type:  one-classification
 SVM-Kernel:  radial
       gamma:  0.002
          nu:  0.5

Number of Support Vectors:  331
```

```
> # test with train data
>   pred <- predict(model, x)
> # (same as:)
> pred <- fitted(model)
>
> # Check accuracy:
```

```
>  table(pred, y)
          y
pred        actual negative positive possible
  actual       8        0        0        0
  negative    13      548       78        0
  positive     0        0       13        0
  possible     0        0        0        0
```

```
> # adopt new cost = 100
>
> ## S3 method for class 'formula':
> svm(formula, data = NULL, subset, na.action =
+  na.omit, scale = TRUE)
Error in as.vector(x, mode) : cannot coerce to vector
> ## Default S3 method:
> svm(x, y = NULL, scale = TRUE, type = NULL, kernel =
+  "radial", degree = 3, gamma = if ( is.vector(x)) 1 else 1 / ncol(x),
+  coef0 = 0, cost = 100, nu = 0.5,
+  class.weights = NULL, cachesize = 40, tolerance = 0.001, epsilon = 0.1,
+  shrinking = TRUE, cross = 0, probability = FALSE, fitted = TRUE, subset, na.action =
na.omit)

Call:
svm.default(x = x, y = NULL, scale = TRUE, type = NULL, kernel = "radial",
    degree = 3, gamma = if (is.vector(x)) 1 else 1/ncol(x), coef0 = 0,
    cost = 100, nu = 0.5, class.weights = NULL, cachesize = 40, tolerance = 0.001,
    epsilon = 0.1, shrinking = TRUE, cross = 0, probability = FALSE,
    fitted = TRUE, subset, na.action = na.omit)



Parameters:
   SVM-Type:  one-classification
 SVM-Kernel:  radial
       gamma:  0.02
          nu:  0.5
```

```
Number of Support Vectors:  330
```

```
> # adopt new cost = 10
>
> ## S3 method for class 'formula':
> svm(formula, data = NULL, subset, na.action =
+ na.omit, scale = TRUE)
Error in as.vector(x, mode) : cannot coerce to vector
> ## Default S3 method:
> svm(x, y = NULL, scale = TRUE, type = NULL, kernel =
+ "radial", degree = 3, gamma = if ( is.vector(x)) 1 else 1 / ncol(x),
+ coef0 = 0, cost = 10, nu = 0.5,
+ class.weights = NULL, cachesize = 40, tolerance = 0.001, epsilon = 0.1,
+ shrinking = TRUE, cross = 0, probability = FALSE, fitted = TRUE, subset, na.action =
na.omit)

Call:
svm.default(x = x, y = NULL, scale = TRUE, type = NULL, kernel = "radial",
    degree = 3, gamma = if (is.vector(x)) 1 else 1/ncol(x), coef0 = 0,
    cost = 10, nu = 0.5, class.weights = NULL, cachesize = 40, tolerance = 0.001,
    epsilon = 0.1, shrinking = TRUE, cross = 0, probability = FALSE,
    fitted = TRUE, subset, na.action = na.omit)

Parameters:
   SVM-Type:  one-classification
 SVM-Kernel:  radial
      gamma:  0.02
         nu:  0.5
Number of Support Vectors:  330
```