Practical - 11

- Shivam Tawari
A - 58

Aim: Write a program in R for implementing a classification (SVM).

Theory:

Classification:
Classification is the process of predicting a categorical label of a data object based on its features and properties.

In classification, we locate identifiers or boundary conditions that correspond to a particular label or category. Then try to place various unknown objects into those categories, by using the identifiers.

There are several classification algorithm that are listed ~~below~~ below:

1. Logistic Regression
2. Decision Tree
3. Support Vector Machine (SVM)
4. Naive Bayes Classifier
5. K- Nearest Neighbour.

Support Vector Machines (SVM):

Support Vector Machine is one of the most popular Supervised Learning algorithms which is used for Classification as well as Regression problems.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. The best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, hence it is called support vector machine.

Support Vectors:

Support vectors are the data points, which are closest to the hyperplane. These points will define the ~~sq~~ separating line better by calculating margins.

Code:

```
install. packages ("e1071")
set . seed (10111)
x = matrix (rnorm (40), 20, 2)
y = rep (c(-1, 1), c (10, 10))
x [y = = 1, ] = x [y == 1, ] + 1
plot (x, col = y+3 , pch = 19)
library (e1071)
dat = data.frame (x, y = as.factor (y))
svmfit = svm (y ~ ., data = dat, kernel = "linear",
              cost = 10 , scale = FALSE)
print (svmfit)
plot (svmfit, dat)
make. grid = function (x, n= 75) {
    grange = apply (x, 2, range)
    x1 = seq (from = grange [1, 1],
              to = grange [2, 1], length = n)
    x2 = seq (from =grange [1, 2],
              to = grange [2, 2], length = n)
    expand.grid (X1 = x1, X2 = x2) }
xgrid = make . grid (x)
xgrid [1 : 10,].
ygrid = predict (svmfit , xgrid)
plot (xgrid, col = c ("red", "blue") [as.numeric(ygrid)],
              pch = 20, cex = .2)
points (x, col = y+3 , pch = 19)
```

The points are more relevant to the construction of the classifier.

Hyperplane :
A hyperplane is a decision plane which separates between a set of objects having different class memberships.

Conclusion : Hence, we have successfully implemented a program in R for classification (SVM).

Code:

Untitled1*

Run Source

```r
# Shivam Tawari A-58
install.packages("e1071")
set.seed(10111)
x = matrix(rnorm(40), 20, 2)
y = rep(c(-1, 1), c(10, 10))
x[y == 1,] = x[y == 1,] + 1
plot(x, col = y + 3, pch = 19)
library(e1071)
dat = data.frame(x, y = as.factor(y))
svmfit = svm(y ~ ., data = dat, kernel = "linear", cost = 10, scale = FALSE)
print(svmfit)
plot(svmfit, dat)
make.grid = function(x, n = 75) {
  grange = apply(x, 2, range)
  x1 = seq(from = grange[1,1], to = grange[2,1], length = n)
  x2 = seq(from = grange[1,2], to = grange[2,2], length = n)
  expand.grid(X1 = x1, X2 = x2)
}
xgrid = make.grid(x)
xgrid[1:10,]
ygrid = predict(svmfit, xgrid)
plot(xgrid, col = c("red","blue")[as.numeric(ygrid)], pch = 20, cex = .2)
points(x, col = y + 3, pch = 19)
points(x[svmfit$index,], pch = 5, cex = 2)
```

Output:

Console Terminal Jobs

/cloud/project/

```
> set.seed(10111)
> x = matrix(rnorm(40), 20, 2)
> y = rep(c(-1, 1), c(10, 10))
> x[y == 1,] = x[y == 1,] + 1
> plot(x, col = y + 3, pch = 19)
> library(e1071)
> dat = data.frame(x, y = as.factor(y))
> svmfit = svm(y ~ ., data = dat, kernel = "linear", cost = 10, scale = FALSE)
> print(svmfit)

Call:
svm(formula = y ~ ., data = dat, kernel = "linear", cost = 10,
    scale = FALSE)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  10

Number of Support Vectors:  6
```

```
> plot(svmfit, dat)
> make.grid = function(x, n = 75) {
+   grange = apply(x, 2, range)
+   x1 = seq(from = grange[1,1], to = grange[2,1], length = n)
+   x2 = seq(from = grange[1,2], to = grange[2,2], length = n)
+   expand.grid(X1 = x1, X2 = x2)
+ }
> xgrid = make.grid(x)
> xgrid[1:10,]
           X1         X2
1  -1.3406379 -0.5400074
2  -1.2859572 -0.5400074
3  -1.2312766 -0.5400074
4  -1.1765959 -0.5400074
5  -1.1219153 -0.5400074
6  -1.0672346 -0.5400074
7  -1.0125540 -0.5400074
8  -0.9578733 -0.5400074
9  -0.9031927 -0.5400074
10 -0.8485120 -0.5400074
> ygrid = predict(svmfit, xgrid)
> plot(xgrid, col = c("red","blue")[as.numeric(ygrid)], pch = 20, cex = .2)
> points(x, col = y + 3, pch = 19)
> points(x[svmfit$index,], pch = 5, cex = 2)
```