ITCS 6156 Fall 2016
Homework 4
Points: 100

This assignment is to be completed individually. No group work is allowed.

The assignment requires [R] and [RStudio] installed on your machine, with packages [kernlab], [ggplot2] and [ROCR]

Download the data and code from the course's Canvas page.

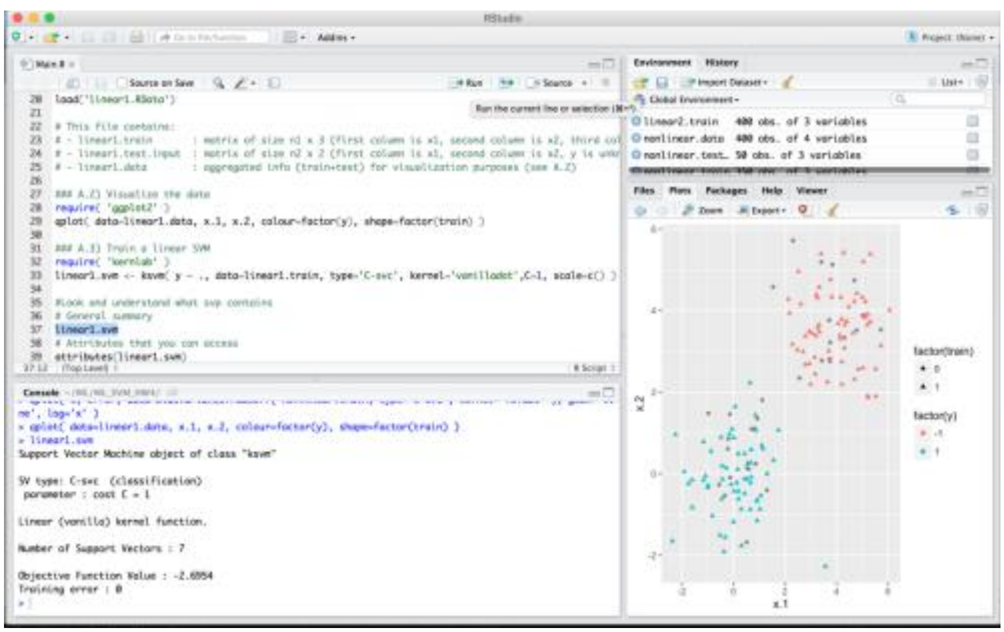
Description

The objective of this exercise is for you to gain practical experience in how to manipulate a SVM in R, with the package kernlab. You will test your model and observe the effect of changing the parameter C and the kernel.

Requirements

For this assignment you need to step through the script in the file "Main.R" that is inside the "ML_SVM_HW4" folder. To run the script and answer the questions for this assignment you are expected to examine and run each line of code in this script.

1. Open "Main.R" in RStudio.
2. Step through the script. Highlighting each line of code and hit "Run".
3. Outputs are displayed in the "Console" and the "Plots" tabs.
4. Record your observations.
5. Save your plots. Click "Export" under the "Plots" tab to save as image/PDF.



Note: you must set the working directory path to your own file settings by updating line 9 setwd('/<File Path>/ML_SVM_HW4/')

Part A: Linear dataset.
A.1/A.2) Visualization of the first dataset:

load('linear1.RData')
This loads 3 new variables are in the environment now. You can type ls() to check.

- linear1.train : a matrix of size n1x3 (first column is *x1*, second column is *x2* and third column is the output *y*).
- linear1.test.input : a matrix of size n2 x 2 (same as the train but without the output *y*).
- linear1.data : the combined dataset (for visualization).

Then visualize the dataset:

require('ggplot2')
qplot(data=linear1.data, x.1, x.2, colour=factor(y), shape=factor(train))

Q1: Save your plot. How can you characterize this dataset?

To classify this 2D dataset we train a linear SVM on the training set which will be our predictor for the testing set.

A.3/A.4/A.5) Training the SVM:
A.3) Train a linear SVM
require('kernlab')
linear1.svm <- ksvm(y ~ ., data=linear1.train, type='C-svc', kernel='vanilladot', C=100, scale=c())

A.4) Plot the model
plot(linear1.svm, data=linear1.train)

A.5) Adding points of test on the graph
points(linear1.test.input[sample.int(nrow(linear1.test.input),10),], pch=4)

Q2: Save your plot. What are the black points in the figure?
Q3: What is the parameter C in the svm?

A.6/A.7 Testing predictions on test set
A.6) Prediction
linear1.prediction <- predict(linear1.svm, linear1.test.input)

A.7) Look at accuracy
load('linear1Sol.RData')
# contains linear1.test.output

```
print(paste0('Accuracy: ', 100*sum( linear1.prediction == linear1.test.output ) / length(linear1.test.output)), '%')
```

Q4: Try different values for the parameter C in the svm. How is the accuracy of the model affected by varying C?

Let's consider a dataset a bit more complex:
A.9 to A.14) Do the exact same thing as the first part on the linear2 dataset (non-separable dataset).

Q5: Is the accuracy a sufficient method to assess the performance of a model?

A.15) Separate positive and negative examples :

```
### A.15) A confusion matrix gives more information than just accuracy
print('Confusion Matrix: ');print(table( linear2.prediction, linear2.test.output, dnn= c("prediction","reality") ))
```

A.16) ROC Curves (See wikipedia):

```
linear2.prediction.score <- predict( linear2.svm, linear2.test.input, type='decision' )
require( 'ROCR' )
## ROC
linear2.roc.curve <- performance( prediction( linear2.prediction.score, linear2.test.output ), measure='tpr', x.measure='fpr' )
plot( linear2.roc.curve )
```

Q6: How would you characterize the performance of this model? Is this good? bad? surprising?

Part B: Non-Linear dataset.
This part deals with the 'nonlinear' dataset.

B.1/B.2/B.3) Trying linear SVM on a dataset where it is not appropriate.
Let's try a different kernel

Q7: How can you characterize this dataset?
Q8: How would you characterize the performance of this model? Is this good? bad? surprising?

B.4) Lets try a better kernel - RBF Kernel:

```
nonlinear.svm <- ksvm( y ~ ., data=nonlinear.train, type='C-svc', kernel='rbf', kpar=list(sigma=1), C=100, scale=c() )
plot( nonlinear.svm, data=nonlinear.train )
```

Q9: How would you characterize the performance of this model? Is this good? bad? surprising?

B.5) You can interactively see the impact of C and kernel type on the model

```
require('manipulate')
manipulate( plot( ksvm( y ~ ., data=nonlinear.train, type='C-svc', kernel=k,
C=2^c.exponent, scale=c() ), data=nonlinear.train ), c.exponent=slider(-10,10),
k=picker('Gaussian'='rbfdot', 'Linear'='vanilladot',
'Hyperbolic'='tanhdot','Spline'='splinedot', 'Laplacian'='laplacedot') )
```

Q10: What is the kernel and C value you set for your model here?

Q11: How would you characterize the performance of this model? Is this good? bad? surprising?
Include the plots you used for this assessment.

B.6) Visualization of the impact of C on the prediction accuracy (**Bias-Variance Tradeoff**):

```
### B.6) Bias-Variance Tradeoff
BiasVarianceTradeoff <- function( dataset, cross=10, c.seq=2^seq(-10, 10), ... ) {
  err <- sapply( c.seq, function( c )
          {
                  cross( ksvm( y ~ ., data=dataset, C=c, cross=cross, ...) )
          })
  return(data.frame( c=c.seq, error=err ))
}

qplot( c, error, data=BiasVarianceTradeoff( nonlinear.train, type='C-svc', kernel='rbfdot'
), geom='line', log='x' )
```

Q12: Plot the performance versus C curve for the nonlinear SVM with Guassian kernel. How would you use this plot to optimize the choice for C?
Include the plots you used for this assessment.

<u>Assignment Submissions</u>

What to submit using Canvas (Email submissions will <u>NOT</u> be accepted):

1. **HW4_report.pdf** – PDF document with your write for the answers to the questions in this assignment (Q1-Q12).
2. **INFO.pdf** – PDF document with the following assignment information:
    a) Explanation of status and stopping point, if incomplete.
    b) Explanation of additional functions and analysis, if any.