Homework: *R* for Classification, Linear SVM
Dolores Romero Morales

The aim of this homework is to revise classification using a dataset you are familiar with (newhousing.txt). This file can be found on the course directory. This file has thirteen explanatory variables. The last column is the class membership ('pricelevel'). There are two classes: 'below' and 'above'.

Note: Please recall that prior to starting this homework you need to ensure that you have uploaded the corresponding R package. To build Support Vector Machines we have used the R package 'e1071'. You already installed it during the PC Workshop. Now you only need to upload it with the following command:

```
library('e1071')
```

Step 1. Read the newhousing.txt file into a data frame and get the dimension of the dataset.

Answer:

```
mynewhousing <- read.table(file.choose(),header=T,stringsAsFactors=TRUE)
dim(mynewhousing)
```

Note: It is important to get familiar with the data you will be working with. It is important to know which are the explanatory variables (columns 1 to 13), and the class membership (column 14).

Step 2. Get a summary report for all the explanatory variables and the class split, i.e., the number of observations in each class. Recall that the class membership is given by the last column.

Answer:

```
summary(mynewhousing)
priceleveltable <- table(mynewhousing$pricelevel)
aboveinmynewhousing <- priceleveltable[1]/nrow(mynewhousing)
belowinmynewhousing <- priceleveltable[2]/nrow(mynewhousing)
```

Step 3. Reshuffle the dataset and take a subsample of 400 observations. Call this the *minihousing*. Call the remaining dataset the *testhousing*.

Answer:

```
set.seed(1)
reshuffle <- mynewhousing[sample(nrow(mynewhousing)),]
minihousing <- reshuffle[1:400,]
testhousing <- reshuffle[401: nrow(reshuffle),]
```

Note:

It is not asked in the question, but if you want the class split in the minihousing and the testinghousing you can use these couple of lines. Please note that you will be able to check that the class split in the minihousing, in the testhousing and in the whole dataset is similar.

```
priceleveltable <- table(minihousing$pricelevel)
aboveinminihousing <- priceleveltable[1]/nrow(minihousing)
belowinminihousing <- priceleveltable[2]/nrow(minihousing)
priceleveltable <- table(testhousing$pricelevel)
aboveintesthousing <- priceleveltable[1]/nrow(testhousing)
belowintesthousing <- priceleveltable[2]/nrow(testhousing)
```

Step 4. Using the *minihousing* sample tune the Support Vector Machine with the linear kernel, for values of the tradeoff parameter equal to 2^k, k=-12,…,12. Report the best value of the parameter found.

Answer:

```
set.seed(1000)
tunedmodellinear <- tune.svm(pricelevel~.,data = minihousing, cost=10^(-12:12),
kernel="linear")
```

Note1:

It is important that you understand that we have taken a wider range of values of the parameter (cost) in the model. By taking a wider range, you are doing a more thorough parameter tuning, and therefore, a better training of your final Linear SVM model.

Note2:

You may have found some "WARNINGS". These warnings come from the optimization software that needs to be called to train the SVM model. Do not worry about them. But if you are curious, this is saying that the software imposes a maximum number of iterations and that this maximum has been reached.

Note3:

It is not been asked in the question, but it would be good if you got a summary of the tuning process that you have just performed. For that, you can call, as usual, the summary function. Please see the output of the summary function below. In particular, it reports

- the best value of cost found in the chosen grid: 0.1
- the corresponding best 10-fold cross validation error: 0.1825

```
summary(tunedmodellinear)

Parameter tuning of 'svm':

- sampling method: 10-fold cross validation

- best parameters:
 cost
  0.1

- best performance: 0.1825

- Detailed performance results:
    cost  error dispersion
1  1e-12 0.4450 0.06324555
2  1e-11 0.4450 0.06324555
3  1e-10 0.4450 0.06324555
4  1e-09 0.4450 0.06324555
5  1e-08 0.4450 0.06324555
6  1e-07 0.4450 0.06324555
7  1e-06 0.4450 0.06324555
8  1e-05 0.4450 0.06324555
9  1e-04 0.4450 0.06324555
10 1e-03 0.2875 0.05682576
```

```
11 1e-02 0.2175 0.08083763
12 1e-01 0.1825 0.06876894
13 1e+00 0.1950 0.07619420
14 1e+01 0.2000 0.08164966
15 1e+02 0.2050 0.08644202
16 1e+03 0.2025 0.08616038
17 1e+04 0.2050 0.07619420
18 1e+05 0.2250 0.10929064
19 1e+06 0.4225 0.11986682
20 1e+07 0.5725 0.17967950
21 1e+08 0.4150 0.16758083
22 1e+09 0.3800 0.13934370
23 1e+10 0.4050 0.18737959
24 1e+11 0.3950 0.16865481
25 1e+12 0.3950 0.16865481
```

Step 5. Use the best value of the parameter found in the previous step to build a model in *minihousing*. Test the model in *testhousing* and report the classification accuracy.

Answer:

```
bestcost <- tunedmodellinear$best.parameters[[1]]
set.seed(100000)
finalmodellinear <- svm(pricelevel~.,data=minihousing,cost=bestcost,kernel="linear")
myprediction <- predict(finalmodellinear, testhousing[,-14])
classificationtable <- table(myprediction,testhousing[,14])
acctestfinalmodellinear <- sum(diag(classificationtable))/sum(classificationtable)
```

Note:

acctestfinalmodellinear is roughly 76.5%.