CECS 551 Programming Assignment 3

Dr. Todd Ebert

Due Date: 3:30pm on March 20th

Preparatory Reading

Read Sections 7.3-7.6.

Presenting Your Work.

Submit via email a *single* R script named `assign3-StudentFirstName.R`. For example, if your first name is George, then you would submit the file `assign3-George.R`. Line 1 of the file should include your full name as a comment. Also, for each block of code that pertains to some exercise, say Exercise 1, preface the code with the comment (all in caps)

```
#EXERCISE 1
```

Each function should be named *exactly* as it appears in the exercise, and have an input signature that follows the order specified in the exercise.

Important: make sure your file produces no errors when sourced by the R interpreter. Otherwise your work will not be graded. Therefore, all answers, tables, explanations, etc. should be placed in comments, so as not to be confused with the actual R code.

Your email submission should be postmarked no later than the time shown on the due date. Submissions received after that time, but on the same day will lose one letter grade. Submissions emailed on a later day will either not be accepted, or will lose two letter grades.

Exercises

1. Write an R function called **partition** that takes as input a data frame `df`, and a real number $\alpha \in (0, 1)$ and returns a list consisting of two data frames `df1` and `df2`, where `df1` consists of $\lfloor \alpha r \rfloor$ randomly selected rows (without replacement) of `df`, and `df2` consists of the remaining unselected rows of `df`, where r denotes the number of rows of `df`. For example, if $r = 100$, and $\alpha = 0.4$, then `df1` will consist of 40 randomly selected rows of `df`, and `df2` will consist of the other 60 rows.

2. Implement the R function

   ```
   best_svm(df, alpha, degree, cost)
   ```

   which works as follows. First, `partition(df,alpha)` is called to obtain two data frames `df1` and `df2` from `df`, where `df1` will serve as the training set, and `df2` will serve as the test set. Now, for each (d, c) combination, where d is a member of vector `degree`, and c is a member of vector `cost`, function

   ```
   svm(Class~., data = df1,  kernel="polynomial", degree = d,
        type = "C-classification",cost=c)
   ```

   is called to create a support-vector machine model, where we assume that "Class" is the name of the data-frame attribute that represents the class label of each vector in the frame. The model is then tested against data set `df2`. Finally `best_svm` returns a list having the three attributes d, c, and `accuracy`, that give the d and c value of the model demonstrating highest accuracy against the test set. Note: make sure that each `df1` and `df2` remain fixed for each learning model.

3. Review and download the Car data set at

   ```
   https://archive.ics.uci.edu/ml/datasets/Car+Evaluation
   ```

   Place attribute names in line one of the file, making sure that `Class` is the name of the final attribute. Load the data set into a data frame `df`, and call `best_svm` on inputs `df`, $\alpha = 0.8$, degree $= (1, 2, 3, 4)$, and cost $= (0.1, 1.0, 10.0, 100.0, 1000.0, 10000.0)$. Call `best_svm` 10 different times on these inputs, and provide a table (as an R-comment) showing the d, c, and `accuracy` values of each call.

4. By setting the C-classification cost to 10^5 (meaning very little to no slack allowance), determine the least degree value d that can produce a (nonlinear) model that can attain 100% accuracy for the entire data set.

5. Repeat Exercise 2, but instead implement the function

   ```
   best.svm.cross(df, degree, cost, n)
   ```

   This function works similar to the previous one, but now, instead of the partitioning `df`, cross validation is performed on the entire data set using $n > 0$ as the fold. Thus, the call to `svm` is now

```
svm(Class~., data = df,  kernel="polynomial", degree = d,
     type = "C-classification",cost=c, cross = n)
```

Finally `best.svm.cross` returns a list having the three attributes d, c, and `accuracy`, that give the d and c value of the model demonstrating highest average cross-validation accuracy, along with the value of the average accuracy (obtained from the model via `model$tot.accuracy`).

6. Apply your function `best.svm.cross` to the Wisconsin Breast Cancer data set found at

   `https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+%28Original%29`

   Place attribute names in line one of the file, making sure that `Class` is the name of the final attribute. Load the data set into a data frame `df`, and call `best.svm.cross` on inputs `df`, degree $= (1, 2, 3, 4)$, cost $= (0.1, 1.0, 10.0, 100.0, 1000.0, 10000.0)$, and $n = 10$. In a comment, report on your findings.

7. Implement the R function

   ```
   bootstrap(df, model, p, n)
   ```

   that takes as input a data frame `df`, svm `model` trained on `df`, probability p, and positive integer n. Function `bootstrap` then makes n bootstrap samples of `df` and, for each sample S, determines the accuracy of `model` tested against S. The n accuracies are then sorted and used to create a confidence interval I so that, with probability p, the model accuracy lies within I. Finally `bootstrap` returns a list having the two attributes `lower` and `upper` that store the lower and upper bounds of the confidence interval.

8. Use function `bootstrap` on the Wisconsin Breast Cancer data set, $p = 0.90$, and $n = 100$. For the model, train an svm on the entire data set using the values for d and c as reported in Exercise 6. In a comment provide the 90% confidence interval for the accuracy of this model.