CSCE 587 Big Data Analytics Spring, 2017 Take Home Midterm Exam

Name ______________________________

Note: Submit your solutions to this exam using dropbox!

Part 1: Setting up the data.

Load the data set "icu". (You used this data set in your last homework assignments. As you may recall, the data set consists of 200 rows of data in which each row corresponds to one patient visiting the icu. Each row consists of 21 columns. When viewed as a classification problem, column 2 (`STA`) specifies the class of each observation/instance. The remaining columns are attributes that might be used to infer column 2. See https://www.umass.edu/statdata/statdata/data/icu.txt for a description of each feature.

1) Start by casting column 2 so that it is interpreted by R as a factor instead of an integer value.

2) Partition the set of 200 instances/observations from the previous step into a new **test set** comprised of 1/3 of the rows and a new **training set** comprised of 2/3 of the rows. We want to randomize the selection of rows into each partition. Call the test set **test_set**, and the training set **train_set**. Use the following steps to accomplish this:

a) Use **set.seed(555)** to set the seed for the random number generator. If you do not do this you will be massively penalized since your result will be hard to verify.

b) Create a vector called "index" with indices for all observations. Since there are 200 rows, your vector "index" should contains the values 1 to 200.

c) Next create a vector of the indices for the test_set by randomly sampling 1/3 of the indices in "index" using the method sample(). This method takes two arguments. The first argument is the vector "index" and the second argument is the number of samples you want to randomly select. You should be selecting 1/3, i.e., 66.

d) Using the vector of test_set indices you created in step 2c, extract the test set, assigning it to test_set.

e) Select the remaining 134 rows as the train_set

Turn in: Be sure to turn in your R code for step 1. Make sure that your R code for part 1 is clearly documented to indicate that it is for Part 1.

Part 2: Naive Bayes Analysis of the icu dataset

1) Using the `naiveBayes()` method from the package e1071, train a Naive Bayes model using the **train_ set** produced in part 1. Review the slides on Naïve Bayes (in particular the last several slides that cover the in-class lab). The arguments to `naiveBayes()` are the independent features and the dependent feature (column 2), the feature you are predicting. By default, consider all features except for column 2 to be independent features.
Review the documentation for Naïve Bayes to see how to specify a model formulae
http://ugrad.stat.ubc.ca/R/library/e1071/html/naiveBayes.html

2) Continuing with the training data set **train_set**, create the confusion matrix using the table command as in slide 61 of the Naïve Bayes slides. Display the confusion matrix.

3) Calculate the model accuracy from the model by summing up the correct classifications and dividing by the total number of observations. Recall that the `table()` command produces a confusion matrix which is a square matrix. The diagonal entries are the correct classifications. You can calculate the accuracy by summing the diagonal entries and dividing by the sum of all of the entries.

4) Repeat steps 2 & 3, using the testing data set, **test_set**.

Turn in: Be sure to turn in your R code for steps 1 - 4 as well as the overall accuracy you calculate in steps 3 & 4. Make sure that your R code for Part 2 is clearly documented to indicate that it is for Part 2.

Part 3: Decision Tree Analysis of the icu dataset

1) Using the `rpart()` method from the package `rpart`, train a Decision Tree model using the training data set **train_set** produced in part 1. Assign this decision tree model to the variable **tree1**, i.e.,
tree1 <- rpart(......)

Review the slides from the Decision Tree Lab. As in Part 2, the arguments to `rpart()` are the independent features and the dependent feature (column 2), the feature you are predicting. By default, consider all features except for column 2 to be independent features. In addition, use the parameter values `cp=0.05 and minsplit=2`.

2) Using the training data set **train_set** produced in part 1, create the confusion matrix using the table command along with **tree1**, the decision tree model you produced in step 1 of this part. Note: to make this work you will need to include a 3rd parameter to the `predict()` function. This parameter is type='class'

3) Calculate the model accuracy from the model by summing up the correct classifications and dividing by the total number of observations. Recall that the table() command produces a confusion matrix which is a square matrix. The diagonal entries are the correct classifications. You can calculate the accuracy by summing the diagonal entries and dividing by the sum of all of the entries.

4) Repeat steps 2 & 3, using the testing data set, **test_set**. DO NOT REPEAT STEP 1.

Turn in: Be sure to turn in your R code for steps 1 - 4 as well as the overall accuracy you calculate in steps 3 & 4. Make sure that your R code for Part 3 is clearly documented to indicate that it is for Part 3.