CSCE 587 Big Data Analytics Fall, 2019 Take Home Midterm Exam

Name _____

Note: Submit your solutions to this exam using dropbox!

Part 1: Setting up the data.

Install the package "titanic". It contains two data sets: `titanic_train` and `titanic_test`. We will only be using `titanic_train`.

The data set `titanic_train` consists of 891 rows of data in which each row corresponds to one of 891 passengers on the Titanic. Each row consists of 12 columns. When viewed as a classification problem, column 2 (`Survived`) specifies the class of each observation/instance. The remaining columns are attributes that might be used to infer column 2. We will focus on Columns 3, 5, 6, 7, & 8 (`Pclass, Sex, Age, SibSp,` and `Parch`). See https://cran.r-project.org/web/packages/titanic/titanic.pdf 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. Note: The integer values are 1 (survived) and 0 (did not survive).

2) Next, make your life easier by creating a subset consisting only of columns 2, 3, 5, 6, 7 & 8. The resulting set is comprised of 891 observations of 6 variables.

3) Randomly partition the set of 891 instances/observations from the previous step into three partitions of 297 observations each using the following steps:

    a. Set the seed for the random number generator using the value 587.
    b. We want to create 3 lists of indices that we can use to partition the data to allow us to perform 3-fold cross validation. If we assume that your dataframe is called "data", create 3 lists of indices as follows:

```
indices <- split(sample(nrow(data), nrow(data),replace=FALSE), as.factor(1:3))
```

    c. You can now create training and testing datasets using these indices to select partitions. For example, if you wanted to create a test set with the middle partition and use the other two partitions as training data you would use these two commands:

```
# take the second partition as the test set
test_set2 <- data[indices[[2]], ]
# take all but the second partition as the training set
train_set2 <- data[-indices[[2]], ]
```

Use this approach to create the following testing and training datasets:

1) **test_set1**, **train_set1** where partition 1 is the testing partition and other partitions are the training data.
2) **test_set2**, **train_set2** where partition 2 is the testing partition and other partitions are the training data.
3) **test_set3**, **train_set3** where partition 3 is the testing partition and other partitions are the training data.

Turn in: Be sure to turn in your R code for Part 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 titanic dataset

1) Using the naiveBayes() method from the package e1071, train a separate Naive Bayes model for each of the three training data sets: **train_set1**, **train_set2**, and **train_set3**, naming them **NB1**, **NB2**, and **NB3**, respectively. 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 (columns 2,3,4,5,6 of train_set) and the dependent feature (column 1), the feature you are predicting.
   Review the documentation for Naïve Bayes to see how to specify a model formula
   http://ugrad.stat.ubc.ca/R/library/e1071/html/naiveBayes.html

2) In this step you evaluate the performance of each model on the corresponding test set, e.g. **NB1** on **test_set1**, **NB2** on **test_set2**, etc. For each model and corresponding test set, create the confusion matrix using the table command as in slide 61 of the Naïve Bayes slides.

3) **Calculate** and **display** the sensitivity and specificity for each confusion matrix produced in the preceding step, e.g. **sensitivity1** and **specificity1** for the first confusion matrix, **sensitivity2** and **specificity2** for the second confusion matrix, and **sensitivity3** and **specificity3** for the third confusion matrix. **Note**: unlike the slides we saw in class, the rows are predictions and the columns are ground truth and the positive **and** negative rows and columns are switched, i.e.,

		Ground Truth	
		-	+
Prediction	-		
	+		

4) **Calculate** and **display** the average sensitivity and specificity

5) Repeat step 2 using your existing models from step 1. This time however, use the corresponding training data ***instead*** of the test data as the evaluation data, e.g. evaluate **NB1** using **train_set1** as the testing data. In this step we want to explore whether or not the models do much better when tested on the same data that was used for training.

6) Repeat step 3 for the confusion matrices produced in step 5.

7) **Calculate** and **display** the average sensitivity and specificity from the values you produced in step 6.

8) Compare the average sensitivity and specificity values produced in step 4 with those in step 7. What lesson can you draw from this comparison? **Explain!!!**

Turn in: Be sure to turn in your R code for steps 1 - 7 as well as the sensitivities and specificities produced in steps 3, 4, 6, and 7. Be sure that you address step 8 by analyzing and comparing the results from steps 4 and 7. Make sure that your R code for Part 2 is clearly documented to indicate that it is for Part 2.