**ISQA 8080 Assignment 3 Due: By Tuesday, Nov. 5 2019, 5:30 PM**

**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**NOTES:**

1. Use R for the calculations and implementation.
2. Submit all documents in a zip file and upload it to Canvas. Name your Zip Folder with your name, A3, and the course # (Example: LastName-A3-ISQA 8080).

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. **Decision Trees – Creating Classification Trees, and Tree Pruning (45 points)**

This problem involves the Census data set which is provided as part of the assignment files. **Note: This is a new version of the data, do not use the one provided with assignment 2!**

Similar to assignment 2, you will develop a model to predict whether a person in the US Census earns more than $50K (High Income) or not (Low Income).

1. Create summary statistics for the data. In particular, how many missing values do you see (overall / per variable)?
2. Now, handle the missing data of the census data set based on the strategies we discussed in class. For numerical variables, you might consider one of the different imputation options (bagImpute or similar) to deal with missing values. Briefly describe how you handled missing data in numerical and in categorical variables. Show the resulting summary information for the Census dataset after handling the missing data.
3. Let’s start building a prediction model. Split the model into 80% training set and 20% test set. Use a unique random seed to create the split (you can use set.seed(sample(10000,1)) for this).

Create a basic classification tree on the training set (e.g., using the rpart package). For this, utilize cross-validation on the training set, and use the complexity parameter cp = 0.0 (specify this using tuneGrid()). Visualize the tree and print the visualization here (note: depending on the actual complexity, the visualization could be hard to read due to the large number of nodes).

1. Use the created tree to predict the target variable on the 20% test set. Create two confusion matrices, one each for the cross-validated training set and the test set, as well as 2 ROC curves. Compare them against each other. What are the respective accuracies, sensitivities, specificities, and AUCs?
2. Now, prune the tree. Use parameter tuning for the complexity parameter cp on the training set (remember: use cross-validation on the 80% training data) to determine the optimal tree complexity/ size. Use this pruned tree to create the test set predictions for the 20% test data.

Show the tree (you can use the cex parameter in rpart.plot to increase the text size in case it’s too small, e.g., cex = 0.7).

How do the accuracies, sensitivities, specificities, and AUCs compare to the unpruned tree for the training and test set?

1. Finally, let’s compare the classification performance against a random classifier as a baseline. The random classifier works as follows:
   1. Calculate the overall probability *p* of income == “High” in your training set. I.e., out of all observations in the training set, how many of them are “High”?
   2. For each observation in the test set, create a prediction of the Income variable using the calculated probability *p*. To do this, create a vector of random numbers between 0 and 1. The vector needs to have the same length as you have observations in your test set. E.g., use runif(nrow(test.set)) for this. Convert the random numbers into predictions using the probability *p*.

Example: if *p* = 45%, predict “High” if a random number is <*p*, otherwise “Low”.

* 1. Create the confusion matrix for the test set and the random classifier prediction. Calculate the accuracy, sensitivity, specificity, and AUC for the random classifier.
  2. Compare them against the previous decision tree. What do you observe?

1. **Decision Trees – Bagging and Boosting (55 points)**

Building on the previous problem, you will create a bagged and a boosted decision tree model for the same Census data set.

1. In your own words, describe the concepts of bagging and boosting as types of ensembles of decision trees. How do they differentiate themselves?

1. In which category does Random Forest fall? How does Random Forest address the problem of correlation between the created trees?
2. Using the same 80% / 20% training and test set split, we will start with building a Random Forest model on the cross-validated training set to predict the income. Use mtry = 10 as parameter, i.e., use 10 predictors (including dummy variables) in each split when building the model (mtry can be included via tuneGrid).

Once you created the random forest model, create the confusion matrix and ROC curve for the training set. Show them here.

1. Now, create another random forest model with the parameter mtry = . Create the confusion matrix and ROC curve for the training set. Show them here. Which model works better, the one with mtry = 11, or the one with mtry = ?
2. Create a variable importance plot (relative influence plot) on the model with mtry = Which variables seem to be the most important in predicting the Income?
3. Now, create a boosted decision tree using the xgbTree model in R. Create the confusion matrix and ROC curve for the training set. Show the results.
4. Compare the performance of the Random Forest and the Gradient Boosting models on the cross-validated training set. Which one seems to work better for the given training data, i.e., which one would you choose? How do they compare against the original decision tree in part 1?
5. Finally, for the model that you select based on the cross-validated training data performance, calculate the predictions for the 20% test set. Create the confusion matrix and ROC curve for the test set. What is the estimated performance of the model that you have selected on unseen data?