Due data: 10/27/2019, end of the day. **Please submit .ipynb file:**

Programming problem (40 points):

1) In this programming problem, you will get familiar with building a decision tree, using cross validation to prune a tree, evaluating the tree performance, and interpreting the result.

   Potential packages to use and short tutorials:
   (1)http://scikit-learn.org/stable/modules/tree.html
   (2)http://chrisstrelioff.ws/sandbox/2015/06/25/decision_trees_in_python_again_cross_validation.html

```
from sklearn import tree # tree library
tree.DecisionTreeClassifier() # for classification tree
tree.DecisionTreeRegressor() # for regression tree
# X: design matrix; Y: labels
fit(X, Y) # fit a tree
predict(X) # make prediction on test data
tree.export_graphviz(model) # visualize  tree
from sklearn.model_selection import KFold # K-fold cross validation
```

```
from sklearn.grid_search import GridSearchCV
In python, you may have to do gridsearch and cross validation using
```

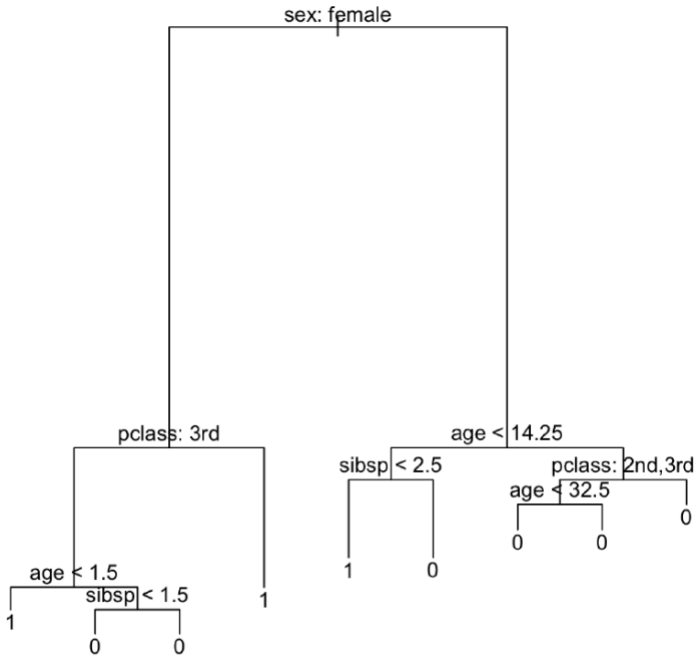
```
GridSearchCV() to choose the best parameters. Try use different values for
"max_leaf_nodes": [None, 1,2,3,4,5,6,7,8,9], (see reference 2).
```

classification tree

Use the titanic.csv dataset included in the assignment.

Step 1: read in Titanic.csv and observe a few samples, some features are categorical and others are numerical. Take a random 70% samples for training and the rest 30% for test.

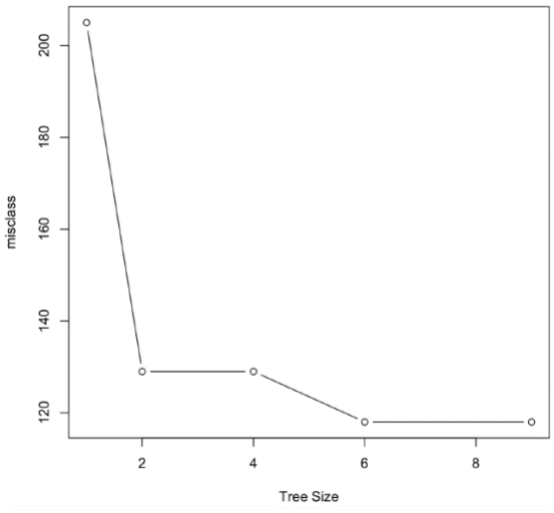
Step 2: fit a decision tree model using independent variables ‘pclass + sex + age + sibsp’ and dependent variable ‘survived’. Plot the full tree. Make sure ‘survived’ is a qualitative variable taking 1 (yes) or 0 (no) in your code. You may see a tree similar to this one:



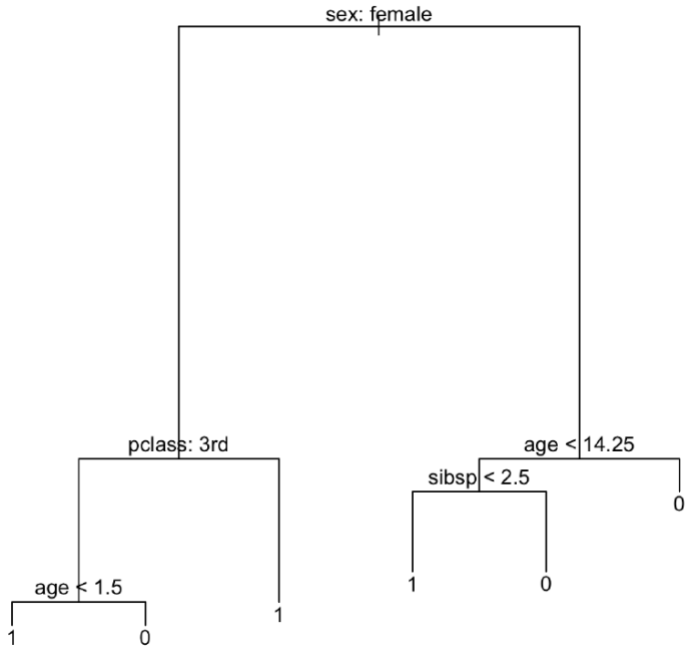
Step 3: print out the performance measures of the full model: in-sample and out-of-sample accuracy, defined as following:

in-sample percent survivors correctly predicted (on training set)
in-sample percent fatalities correctly predicted (on training set)
out-of-sample percent survivors correctly predicted (on test set)
out-of-sample percent fatalities correctly predicted (on test set)

Step 4: use cross-validation to find the best parameter to prune the tree. You should be able to plot a graph with the 'tree size' as the x-axis and 'number of misclassification' as the Y-axis. Find the minimum number of misclassification and choose the corresponding tree size to prune the tree. You may have a plot similar to:



Step 5: prune the tree with the optimal tree size. Plot the pruned tree. You may see a similar tree like this:



Step 6: For the final pruned tree, report its in-sample and out-of-sample accuracy, defined as

in-sample percent survivors correctly predicted (on training set)
in-sample percent fatalities correctly predicted (on training set)
out-of-sample percent survivors correctly predicted (on test set)
out-of-sample percent fatalities correctly predicted (on test set)

Check whether there is improvement in out-of-sample for the full tree (bigger model) and the pruned tree (smaller model).