**Due data:** 10/24/2018, end of the day. **Please submit the following 2 files via Canvas:**

1) For question 1 – 4, please submit in a word file or a PDF file;
2) For question 5, please submit a .ipynb file (Python jupiter notebook file).

## Question 1 (6 points):

**Please briefly explain the main idea of following learning algorithms:**

1) MAP Learning
2) Support Vector Machine
3) Bias-variation trade-off

## Question 2 (5 points):

Consider again the example application of Bayes rule in Section 6.2.1 of Tom Mitchell's textbook (or slide # 6 of Lecture 6). Suppose the doctor decides to order a second laboratory test for the same patient and suppose the second test returns a positive result as well. What are the posterior probabilities of *cancer* and $\neg$*cancer* following these two tests? Assume that the two tests are independent.

## Question 3 (6 points):

Consider the following set of training examples:

| Instance | Classification | $a_1$ | $a_2$ |
|---|---|---|---|
| 1 | + | T | T |
| 2 | + | T | T |
| 3 | − | T | F |
| 4 | + | F | F |
| 5 | − | F | T |
| 6 | − | F | T |

(*a*) What is the entropy of this collection of training examples with respect to the target function classification?

(*b*) What is the information gain of $a_2$ relative to these training examples?

## Question 4 (5 points):

Use **Naïve Bayes classifier** to classify the following new instance of the target concept *PlayTennis* that we discussed in Chapter 3:

**New instance**: <Outlook = sunny, Temperature = mild, Humidity = normal, wind = strong>

You are given the following 14 training examples:

| Day | *Outlook* | *Temperature* | *Humidity* | *Wind* | *PlayTennis* |
|---|---|---|---|---|---|
| D1 | Sunny | Hot | High | Weak | No |
| D2 | Sunny | Hot | High | Strong | No |
| D3 | Overcast | Hot | High | Weak | Yes |
| D4 | Rain | Mild | High | Weak | Yes |
| D5 | Rain | Cool | Normal | Weak | Yes |
| D6 | Rain | Cool | Normal | Strong | No |
| D7 | Overcast | Cool | Normal | Strong | Yes |
| D8 | Sunny | Mild | High | Weak | No |
| D9 | Sunny | Cool | Normal | Weak | Yes |
| D10 | Rain | Mild | Normal | Weak | Yes |
| D11 | Sunny | Mild | Normal | Strong | Yes |
| D12 | Overcast | Mild | High | Strong | Yes |
| D13 | Overcast | Hot | Normal | Weak | Yes |
| D14 | Rain | Mild | High | Strong | No |

[ Please refer to lecture slides Lecture 6, pages 28 – 30; and Mitchell textbook section 6.9.]

## Question 5: Programming Problem (18 points):

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
```

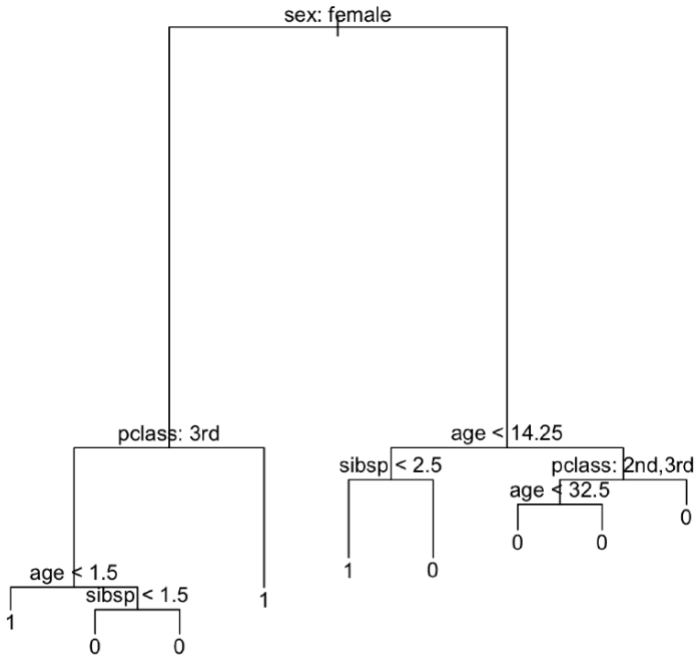
In python, you may have to do gridsearch and cross validation using GridSearchCV() to choose the best parameters. Try use different values for "max_leaf_nodes": [None, 1,2,3,4,5,6,7,8,9], (see reference 2).

**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.

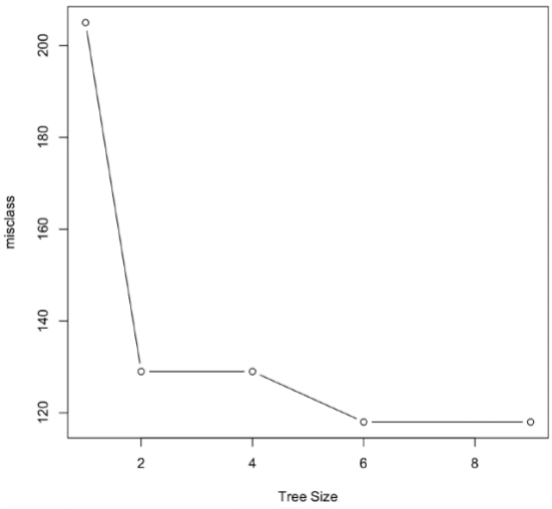
**Step 2:** fit a decision tree model using independent variables 'pclass + sex + age + sibsp' and dependent variable 'survived'. Plot the full tree. Make sure 'survived' is a qualitative variable taking 1 (yes) or 0 (no) in your code. You may see a tree similar to this one:



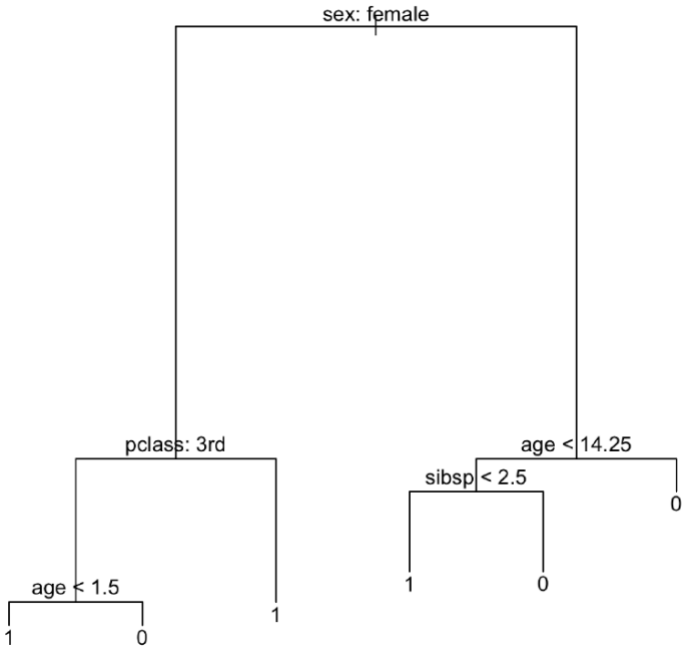

**Step 3:** check the performance of the full model: in-sample and out-of-sample accuracy, defined as:

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)

**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:** Report as many details as you can on the final pruned tree.

Required reports on: 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).