1. Review how decision trees are built and pruned.
2. Load the spam dataset from the kernlab package. Read the documentation of the dataset.
3. Split the data into a training and a test set.
4. Use the tree function from the tree package to train a decision tree to predict the type (spam/nonspam) of the emails:
   ```r
   tree.spam = tree(...) 
   summary(tree.spam)
   ```
5. Plot your estimated decision tree:
   ```r
   plot(tree.spam) 
   text(tree.spam, pretty=0)
   ```
6. Use the predict function to compute the classification error on the test set.
1. Construct a sequence of relevant pruned trees using CV and weakest link pruning:

```r
cv.spam = cv.tree(tree.spam, FUN = prune.misclass)
```

Note: `cv.spam$dev` contains the CV error of each tree. `cv.spam$size` contains the size of each tree.

2. Fit the pruned tree for which the CV error is minimal:

```r
prune.spam = prune.misclass(tree.spam, best=sizeminCV)
```

where `sizeminCV` is the size of the tree achieving minimum CV error.

3. Use the predict function to compute the prediction error of the pruned tree on the test set:

```r
yhat_prune = predict(prune.spam, ...)
```
1. Use the following commands to construct an aggregation of trees using the bagging technique:

```r
library(randomForest)

bag.spam = randomForest(type ~ ., data=train,
                          mtry = 57, importance=TRUE)
```

Note: because of the `mtry=57` argument (57 = number of variables), the random forest (topic to be discussed next lecture) reduces to a bootstrap aggregation of usual decision trees.

2. Use the `predict` function to compute the test error of the bagging model.