Lab #3 – Trees and Random Forest

Useful commands

Command	Explanation
read.table()	Reads a file in table format and creates a data frame from it. Useful for text files (i.e. not csv)
tree(formula, data, type)	Fits a tree model to data; type controls the output response type ("class" for classification)
cv.tree()	Runs cross-validation to find size of tree that minimizes cross-validation error
prune.misclass(model,best)	Prunes a tree model to have number of leaves equal to best
randomForest(formula, data)	Runs random forest model
importance(rf.model)	Measures importance of each variable in random forest model
varImpPlot	Plots measures of importance for each variable

Example code for classification:

```
dat <-iris[c("Sepal.Length","Sepal.Width","Species")]
#scaling
dat$Sepal.Length <- as.vector(scale(dat$Sepal.length))
dat$Sepal.Width <- as.vector(scale(dat$Sepal.Width))
#create train and test sets
index <- sample(x=1:nrow(dat), size=.3*nrow(dat))
test <- dat[index,]
train <- dat[-index,]
test_pred <- test[c("Sepal.Length","Sepal.Width")] #predictors only
train_pred <- train[c("Sepal.Length","Sepal.Width")] #predictors only
test_species <- test$Species #true test classification
train_species <- train$Species #true train classification

#run tree model
tree_model <- tree(Species~., data = train)
summary(tree_model)
#visualize tree
plot(tree_model)
text(tree_model, pretty=0)
#predict on the test set
tree_preds <- predict(tree_model, newdata = x_test, type = "class")
#check accuracy of predictions
table(tree_preds, test_species)  #confusion matrix
mean(tree_preds ==test_species)  #accuracy
```

```
#cross-validation for pruning tree
cv_tree_model <- cv.tree(tree_model)
cv_tree_model
#plot deviance as function of sizs
plot(cv_tree_mod$size, cv_tree_mod$dev, type="b")
#prune tree to have 5 leaves
pruned_tree <- prune.misclass(tree_mod, best = 5)

#library for random forest
library(randomForest)
#run random forest
rf_model <- randomForest(Species~., data = train)
#predict on the test set
rf_pred <- predict(rf_model, newdata = x_test)
#check accuracy of predictions
table(rf_pred, test_species)   #confusion matrix
mean(rf_pred == test_species)  #accuracy
```

Exercise:

We will implement tree and random forest models, starting with the famous Spam dataset (classification) and our good old California Housing data set (regression).

Classification

1. Load the Spam data from the file *spam.data* using the *read.table* function
   * Hint: use *dat <- read.table("..../spam.data")*
2. Learn your data:
   a. https://web.stanford.edu/~hastie/ElemStatLearn/datasets/spam.info.txt
   b. What is the size of the data?
   c. Use the *str* function in order to take a glimpse at the data, what can you tell about it?
3. Split the data into train and test sets (does this sound familiar???)
   * Hint: the response (the y) is the last column of the data
4. We want to predict if a received mail is spam or not
5. **Trees**
   a. Load the *tree* library
   b. Fit a tree model using the *tree* function
      * Hint: notice that the formula is the same as in *lm()*
   c. Print summary of the *tree* model and examine the results
      * Hint: exactly like an *lm* model, i.e. *summary(model)*
   d. Plot the model using *plot(model)* and *text(model, pretty=0),* what can you say about the results?
   e. Use the model for prediction on the test set
      * Hint: use predict (exactly like an *lm* model) and set *type="class"*

f. Evaluate the fit by printing the accuracy and the confusion matrix (recall that a confusion matrix is the prediction vs the truth)

g. Recall that we prune the tree in order to avoid overfitting. The tree size is a tuning parameter and defines the complexity of the tree, thus it is determined by cross-validation. Run cross-validation to identify how to prune the tree.

   * Hint: use *cv.tree(model)*

h. Examine the output of the cv function (print the output from g). What is *size*? What is *k*?

   * Hint: use *?prune.misclass*

i. *dev* reports the deviance, which is a measure of how good the fit is. Lower deviance means a better fit. Below is a brief explanation from *Introduction to Statistical Learning*

   We see that the training error rate is 9 %. For classification trees, the deviance reported in the output of `summary()` is given by

$$-2 \sum_m \sum_k n_{mk} \log \hat{p}_{mk},$$

   where n_{mk} is the number of observations in the mth terminal node that belong to the kth class. A small deviance indicates a tree that provides a good fit to the (training) data. The *residual mean deviance* reported is simply the deviance divided by $n - |T_0|$, which in this case is $400 - 27 = 373$.

   Plot the deviance as a function of the size and the k.

   * Hint: use access the values by *model$....* and run *plot(x,y, type="b")*

j. Find the tree size that minimizes the corss-validation error

k. Prune the tree using the value you found in k. by running *prune.misclass(tree_mod, best = best_size)*

l. Repeat d.-f. with the pruned tree

6. **Random Forest:**
   a. Load *randomForset* package
   b. Fit a random forest model using the *randomForest* function.
   c. Use the model for prediction on the test set and evaluate the fit by its accuracy and the confusion matrix.

Regression

7. **Trees**
   a. Read California Housing data set stored in a CSV file *read.csv("path...../CAhousing.csv")*
   b. Split the data into train and test
   c. Everything that was done for the classification setting can be repeated in the regression setting in exactly the same way! Fit a regression tree to the data

      * Hint: same exact syntax as e. just <u>do not</u> set *type=class*

   d. Use the model for prediction on the test set and evaluate the fit using the root mean squared error (RMSE)

* Hint: sqrt(mean((pred-test)^2))

8. **Random Forest**
    a. Fit a random forest model in exactly the same way, but set *importance=TRUE*
    b. Use the model for prediction on the test set and evaluate the fit using the root mean squared error (RMSE)
    c. Run *importance* and *varImpPlot* on the received model. These are some measures for importance of variables. For more see *?importance* and ILSR pg. 330.

Good Luck!