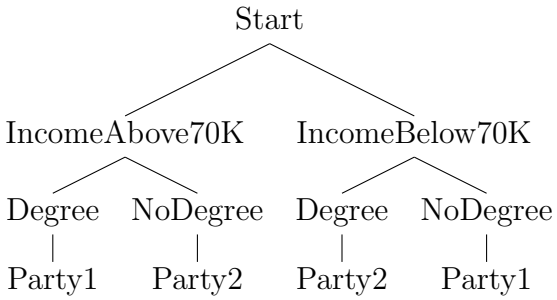
Random Forests
J. Eustace
November 2013

1 Random Forests

We begin with the concept of a **classification tree**. Imagine that we have a set of explanatory variables that we would like to use to predict an observed category.

In a classification tree, each variable is analyzed and split so that each subgroup is as homogenous as possible. In the following example, we would like to know which political party a person is likely to vote for, based on their income and education level.



The program in our example has identified that the first split happens when income hits $70,000. Another split occurs for those who have a degree versus those who do not. This is a binary example but it is easy to extend this to multiple splits or categories. Generally we would "train" our tree using a small sample in which the categories are known, and then apply the tree to unknown data for prediction.

What if we begin with 50 variables? In this case, we would risk overfitting if we tried to make a single tree that contained all of these variables. Therefore it is a better idea to use the **random forest** mthod.

Random forests are aptly named, as they made up of many decision trees. For each of the k trees, we select m variables and n cases (with replacement) at random, and use these to build the tree. Once our trees are trained, we can take an unknown case and feed it to each of the k trees, recording the category result each time. The mode of these k results will be our random forest prediction.

1.1 Implementation

- "randomForest" package in R
- "RandomForestClassifier in scikits learn" package in Python

1.2 Missing Data

The randomForest R package does not deal with missing data. For this reason, we will need to fill them in somehow before we use that method. Thankfully there is a function in the package called rfImpute that does this for us.

First we must change the empty cells in X to "NA", and save the data as a .dat file. We can put our observed value in to Y, and put the explanatory variables into dataframe X.

```
X.imputed <- rfImpute(X,Y,iter=5, ntree=100)
```

The unfortunate thing here is that we will need to fill in data for the test set, but do not have the observed variable in that case. In the titanic data, we can choose another variable such as class that can be used as an observed variable for the purpose of filling the missing data. Note that the observed data column must be one that does not have any missing cells.

1.3 Using R

The randomForest function takes a dataframe $X1$ and an observed vector Y. Note that if Y is categorical, we must use the as.factor() function to change the format first. Otherwise it will treat this as a regression problem.

The predict function takes a randomForest type variable and a new dataframe. To test the method on the Titanic data, I separated the last 100 rows of data before training the trees.

```
titanic.rf <- randomForest(X1, Y, ntree=100, replace=TRUE, importance=TRUE)
titanic.pred <- predict(titanic.rf, X2, type="response")
table(observed = Y2, predicted = titanic.pred )
```

predicted:		0	1
observed:	0	57	7
	1	10	26

The resulting table tells us that we should be around 83% accurate on the test data. This is pretty good, since according to Kaggle we will risk overfitting if our accuracy is much higher than 85% (and therefore we will do poorly on the hidden set).

1.4 Sources

1. Wikipedia: "Decision tree learning"
2. Wikipedia: "Random Forest"
3. Kaggle: Random Forests
   <https://www.kaggle.com/wiki/RandomForests>

1.5 R Code

```
#install.packages("randomForest")
#################################

library(randomForest)

data <- read.table('train.dat',  header=TRUE)
test_data <-read.table('test_data.dat',  header=TRUE)

PassengerId <- data$PassengerId
Survived <- data$Survived
Pclass <- data$Pclass
Sex <- data$Sex
Age <- data$Age
SibSp <- data$SibSp
Parch <- data$Parch
Fare <- data$Fare
Embarked <- data$Embarked
```

```
X <- data.frame(Pclass,Sex,Age,SibSp,Parch,Fare,Embarked)

PassengerId2 <- test_data$PassengerId
Pclass <- test_data$Pclass
Sex <- test_data$Sex
Age <- test_data$Age
SibSp <- test_data$SibSp
Parch <- test_data$Parch
Fare <- test_data$Fare
Embarked <- test_data$Embarked

X_test <- data.frame(Pclass,Sex,Age,SibSp,Parch,Fare,Embarked)
Y <- as.factor(Survived)

X.imputed <- rfImpute(X,Y,iter=5, ntree=100)

Y2 <- X_test[,1]
Y2 <- as.factor(Y2)
X.test.imputed <- rfImpute(X_test,Y2 ,iter=5, ntree=100)

X1 <-X.imputed[,2:8]
X2 <-X.test.imputed [,2:8]

titanic.rf <- randomForest(X1, Y, ntree=100, replace=TRUE, importance=TRUE)

titanic.pred <- predict(titanic.rf, X2, type="response")

write.csv(titanic.pred, file="titanic2.csv")
```