Random Forests and Decision Trees
==
author: An alternative to predicting credit defaulters - which model performs better?
date:
autosize: true

Previously
==
- Previously we used logistic regression to predict whether or not someone was going to default on credit payment
- However, logistic regression is not the only model available
- In this section we will cover random forests and decision trees, an alternative to linear and logistic regression. We will compare models and decide which is more suited to our needs.

What is a tree?
==
- Given a dataset, we split the data recursively so that each subset is as homogenous as possible
- We measure homogeneity using the Gini index. Alternatively we can also use something called cross-entropy
- Both the Gini Index and cross-entropy measure the total variance across all classes in our data.
- A small value means out split is "purer"

Visualising Splits
==
`![alt text](TreeSplits.png)`

Source: Introduction to Statistical Learning

Pruning a tree
==
- The problem: too many splits lead to us overfitting on our data, too few and we underfit
- Usually the solution is to grow a large tree then prune it back to a smaller subtree

Let's use Decision Trees on our Default Dataset
==
Perform train/test split
```{r}
library(MASS)
library(ISLR)
library(caret)
set.seed(88)
train_Index <- createDataPartition(Default$default, p=0.8, list=FALSE)
train_credit <- Default[train_Index,]
test_credit <- Default[-train_Index,]
head(train_credit)
```

Fit Model
==
```{r}
model1 <- train(default ~ ., data=train_credit, method='rpart')
plot(model1)
# by default, our complexity parameter is 3
```

Playing with different values for the complexity parameter
==
```{r}
model2 <- train(default ~ balance, data=train_credit, method='rpart', tuneLength=6)
plot(model2)
```

==
```{r}
model2
```

Evaluating our model with test data
==
```{r}
pred <- predict(model2, test_credit)
confusionMatrix(pred, test_credit$default)
```

ROC Curve
==
```{r}
library(pROC)
pred_prob <- predict(model2, test_credit, type="prob")
rpart_ROC <- roc(predictor = pred_prob$No, response = test_credit$default, levels = rev(levels(test_credit$default)))
plot(rpart_ROC)
```

The problem with a single tree

- Decision Trees suffer from high variance
- Fitting our model to distinct data sets could lead to quite different results.
- In contrast, models such as linear regression tend to have low variance.
- To overcome high variance, we use methods such as Random Forests.
- Essentially, we take repeated samples from one training datset, fit a model to each sample, then average the predictions. The idea is that averaging in this way reduces variance.

Random Forests

```{r}
model_rf <- train(default ~ student + balance + income,
data=Default, method="rf")

pred = predict(model_rf, newdata=test_credit)
confusionMatrix(data=pred, test_credit$default)
```

Comparing Decision Trees and Random Forests to Logistic Regression and Linear Regression

- Classical approaches like linear regression assume a linear relationship between dependent and independent variables.
- If this assumption is true, a linear model will likely outperform a tree model
- However, if the relationship between variables is complex and non-linear, then a tree model might perform better.
- In terms of interpretability, trees are easier to visualise and understand
- Trees can handle qualitative predictors without having to create dummy variables.

```
![alt text](TreesandLinearModels.png)
```

Introduction to K-fold Cross Validation

When evaluating models, we often want to assess how well it performs in predicting the target variable on different subsets of the data. One such technique for doing this is k-fold cross-validation, which partitions the data into k equally sized segments (called ‘folds’). One fold is held out for validation while the other k-1 folds are used to train the model and then used to predict the target variable in our testing data. This process is repeated k times, with the performance of each model in predicting the hold-out set being tracked using a performance metric such as accuracy. The most common variation of cross validation is 10-fold cross-validation.

```{r}
# implement K-fold Cross Validation in caret
ctrl <- trainControl(method = "repeatedcv", number = 10,
savePredictions = TRUE)

model_with_cv <- train(default ~ student + balance + income,
data=Default, method="rf",
                 trControl = ctrl)

pred = predict(model_with_cv, newdata=test_credit)
confusionMatrix(data=pred, test_credit$default)
```