Week 4 Assignment

Ben Arancibia

June 27, 2015

KJ 8.1

Recreate the simulated data from Exercise 7.2:

```
library(mlbench)
set.seed(200)
simulated <- mlbench.friedman1(200, sd = 1)
simulated <- cbind(simulated$x, simulated$y)
simulated <- as.data.frame(simulated)
colnames(simulated)[ncol(simulated)] <- "y"
```

a) Fit a random forest model to all of the predictors, then estimate the variable importance scores:

```
library(randomForest)
```

```
## randomForest 4.6-10
## Type rfNews() to see new features/changes/bug fixes.
```

```
library(caret)
```

```
## Warning: package 'caret' was built under R version 3.1.3
```

```
## Loading required package: lattice
```

```
## Warning: package 'lattice' was built under R version 3.1.3
```

```
## Loading required package: ggplot2
```

```
## Warning: package 'ggplot2' was built under R version 3.1.3
```

```
model1 <- randomForest(y ~ ., data = simulated, importance = TRUE, ntree = 1000)
rfImp1 <- varImp(model1, scale = FALSE)
```

Did the random forest model significantly use the uninformative predic- tors (V6 – V10)?

```
knitr::kable((round(rfImp1,2)))
```

	Overall
V1	8.91
V2	6.50

	Overall
V3	0.79
V4	7.94
V5	2.12
V6	0.18
V7	0.05
V8	-0.12
V9	-0.09
V10	-0.08

The model appears to place most importance on variables 1, 2, 4, and 5, and very little importance on 3 and 6 through 10.

b) Now add an additional predictor that is highly correlated with one of the informative predictors. For example:

```
simulated$duplicate1 <- simulated$V1 + rnorm(200) * .1
cor(simulated$duplicate1, simulated$V1)
```

```
## [1] 0.9304634
```

Fit another random forest model to these data. Did the importance score for V1 change? What happens when you add another predictor that is also highly correlated with V1?

```
model2 <- randomForest(y ~ ., data = simulated, importance = TRUE, ntree = 1000)
rfImp2 <- varImp(model2, scale = FALSE)

vnames <- c('V1', 'V2', 'V3', 'V4', 'V5', 'V6', 'V7', 'V8', 'V9', 'V10', 'duplicate1')

names(rfImp1) <- "Original"
rfImp1$Variable <- factor(rownames(rfImp1), levels = vnames)

names(rfImp2) <- "Extra"
rfImp2$Variable <- factor(rownames(rfImp2), levels = vnames)

rfImps <- merge(rfImp1, rfImp2, all = TRUE)
rownames(rfImps) <- rfImps$Variable
rfImps$Variable <- NULL

knitr::kable((round(rfImps,2)))
```

	Original	Extra
V1	8.91	6.18
V2	6.50	5.89
V3	0.79	0.60

	Original	Extra
V4	7.94	7.15
V5	2.12	2.06
V6	0.18	0.15
V7	0.05	-0.02
V8	-0.12	-0.04
V9	-0.09	-0.12
V10	-0.08	0.06
duplicate1	NA	3.70

When you add another highly correlated predictor the importance score for V1 drops.

c) Use the cforest function in the party package to fit a random forest model using conditional inference trees. The party package function varimp can calculate predictor importance. The conditional argument of that func- tion toggles between the traditional importance measure and the modified version described in Strobl et al. (2007). Do these importances show the same pattern as the traditional random forest model?

```
library(party)
```

```
## Warning: package 'party' was built under R version 3.1.3

## Loading required package: grid
## Loading required package: mvtnorm
## Loading required package: modeltools
## Loading required package: stats4
## Loading required package: strucchange

## Warning: package 'strucchange' was built under R version 3.1.3

## Loading required package: zoo

## Warning: package 'zoo' was built under R version 3.1.3

##
## Attaching package: 'zoo'
##
## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric
##
## Loading required package: sandwich

## Warning: package 'sandwich' was built under R version 3.1.3
```

```
set.seed(147)
cforest1 <- cforest(y ~ ., data = simulated[, 1:11], controls = cforest_control(ntree = 1000))
set.seed(147)
cforest2 <- cforest(y ~ ., data = simulated, controls = cforest_control(ntree = 1000))

cfImps1 <- varimp(cforest1)
cfImps2 <- varimp(cforest2)
cfImps3 <- varimp(cforest1, conditional = TRUE)
cfImps4 <- varimp(cforest2, conditional = TRUE)

cfImps1 <- data.frame(Original = cfImps1, Variable = factor(names(cfImps1), levels = vnames))

cfImps2 <- data.frame(Extra = cfImps2, Variable = factor(names(cfImps2), levels = vnames))

cfImps3 <- data.frame(CondInf = cfImps3, Variable = factor(names(cfImps3), levels = vnames))

cfImps4 <- data.frame("CondInf Extra" = cfImps4, Variable = factor(names(cfImps4), levels = vnames))

cfImps <- merge(cfImps1, cfImps2, all = TRUE)
cfImps <- merge(cfImps, cfImps3, all = TRUE)
cfImps <- merge(cfImps, cfImps4, all = TRUE)
rownames(cfImps) <- cfImps$Variable
cfImps$Variable <- factor(cfImps$Variable, levels = vnames)
cfImps <- cfImps[order(cfImps$Variable),]
cfImps$Variable <- NULL

knitr::kable((round(cfImps,2)))
```

	Original	Extra	CondInf	CondInf.Extra
V1	9.06	5.93	3.04	1.15
V2	6.85	6.25	4.01	3.55
V3	0.05	0.06	0.03	0.02
V4	8.70	8.59	4.87	4.81
V5	2.21	2.07	0.73	0.69
V6	0.02	-0.03	0.01	0.01
V7	0.10	0.05	0.03	0.01
V8	-0.06	-0.03	-0.01	-0.01
V9	-0.06	-0.07	-0.01	0.00
V10	-0.01	0.01	0.00	0.00
duplicate1	NA	3.70	NA	0.49

The conditional inference model has a similar pattern of importance as the random forest model from Part (a), placing most importance on predictors 1, 2, 4, and 5 and very little importance on 3, 6 through 10. Adding a highly correlated predictor has a detrimenal effect on the importance for $V1$.

d) Repeat this process with different tree models, such as boosted trees and Cubist. Does the same pattern occur?

Boosted Trees

```
library(ipred)
set.seed(147)
bagFit1 <- bagging(y ~ ., data = simulated[, 1:11], nbag = 50)

set.seed(147)
bagFit2 <- bagging(y ~ ., data = simulated, nbag = 50)
bagImp1 <- varImp(bagFit1)
```

```
## Loading required package: plyr
```

```
## Warning: package 'plyr' was built under R version 3.1.3
```

```
##
## Attaching package: 'plyr'
##
## The following object is masked from 'package:modeltools':
##
##     empty
##
## Loading required package: rpart
```

```
names(bagImp1) <- "Original"
bagImp1$Variable <- factor(rownames(bagImp1), levels = vnames)

bagImp2 <- varImp(bagFit2)
names(bagImp2) <- "Extra"
bagImp2$Variable <- factor(rownames(bagImp2), levels = vnames)

bagImps <- merge(bagImp1, bagImp2, all = TRUE)
rownames(bagImps) <- bagImps$Variable
bagImps$Variable <- NULL

knitr::kable((round(bagImps,2)))
```

	Original	Extra
V1	1.92	1.83
V2	2.30	2.21
V3	1.37	1.16
V4	2.77	2.72
V5	2.43	2.24
V6	1.00	0.87
V7	0.94	0.92
V8	0.58	0.49
V9	0.68	0.60
V10	0.86	0.74
duplicate1	NA	1.83

Cubist Trees

```
library(Cubist)
set.seed(147)
cbFit1 <- cubist(x = simulated[, 1:10], y = simulated$y, committees = 100)

cbImp1 <- varImp(cbFit1)
names(cbImp1) <- "Original"
cbImp1$Variable <- factor(rownames(cbImp1), levels = vnames)

set.seed(147)
cbFit2 <- cubist(x = simulated[, names(simulated) != "y"], y = simulated$y, committees = 100)

cbImp2 <- varImp(cbFit2)
names(cbImp2) <- "Extra"
cbImp2$Variable <- factor(rownames(cbImp2), levels = vnames)

cbImp <- merge(cbImp1, cbImp2, all = TRUE)
rownames(cbImp) <- cbImp$Variable
cbImp$Variable <- NULL

knitr::kable((round(cbImp,2)))
```

	Original	Extra
V1	71.5	47.5
V2	58.5	57.5
V3	47.0	49.0
V4	48.0	42.5
V5	33.0	27.0
V6	13.0	11.5
V7	0.0	0.5
V8	0.0	3.5
V9	0.0	0.0
V10	0.0	0.0
duplicate1	NA	37.5

KJ 8.6