

# Regression and Classification with R \*

Yanchang Zhao

<http://www.RDataMining.com>

5 May 2017

---

\*Chapters 4 & 5, in *R and Data Mining: Examples and Case Studies*.

<http://www.rdatamining.com/docs/RDataMining-book.pdf>

Introduction

Linear Regression

Generalized Linear Regression

Decision Trees with Package party

Decision Trees with Package rpart

Random Forest

Online Resources

- ▶ Basics of regression and classification
- ▶ Building a linear regression model to predict CPI data
- ▶ Building a generalized linear model (GLM)
- ▶ Building decision trees with package *party* and *rpart*
- ▶ Training a random forest model with package *randomForest*

---

†Chapter 4: Decision Trees and Random Forest & Chapter 5: Regression, in book *R and Data Mining: Examples and Case Studies*.

<http://www.rdatamining.com/docs/RDataMining.pdf>

- ▶ Regression: to predict a continuous value, such as the volume of rain
- ▶ Classification: to predict a categorical class label, such as weather: rainy, sunny, cloudy or snowy

- ▶ Regression is to build a function of *independent variables* (also known as *predictors*) to predict a *dependent variable* (also called *response*).
- ▶ For example, banks assess the risk of home-loan applicants based on their age, income, expenses, occupation, number of dependents, total credit limit, etc.
- ▶ Linear regression models
- ▶ Generalized linear models (GLM)

- ▶ MAE: Mean Absolute Error

$$\text{MAE} = \frac{1}{n} \sum_{i=1}^n |\hat{y}_i - y_i| \quad (1)$$

- ▶ MSE: Mean Squared Error

$$\text{MSE} = \frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2 \quad (2)$$

- ▶ RMSE: Root Mean Squared Error

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^n (\hat{y}_i - y_i)^2} \quad (3)$$

where  $y_i$  is actual value and  $\hat{y}_i$  is predicted value.

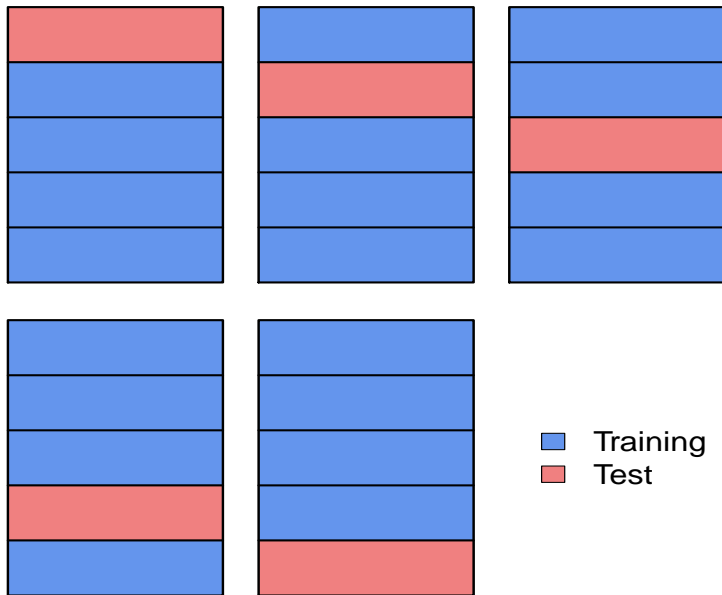
- ▶ A model is over complex and performs very well on training data but poorly on unseen data.
- ▶ To evaluate models with out-of-sample test data, i.e., data that are not included in training data





- ▶ Split data into  $k$  subsets of equal size
- ▶ Reserve one set for test and use the rest for training
- ▶ Average performance of all above

# 5-Fold Cross Validation



Introduction

Linear Regression

Generalized Linear Regression

Decision Trees with Package party

Decision Trees with Package rpart

Random Forest

Online Resources

- ▶ Linear regression is to predict response with a linear function of predictors as follows:

$$y = c_0 + c_1x_1 + c_2x_2 + \cdots + c_kx_k,$$

where  $x_1, x_2, \dots, x_k$  are predictors,  $y$  is the response to predict, and  $c_0, c_1, \dots, c_k$  are coefficients to learn.

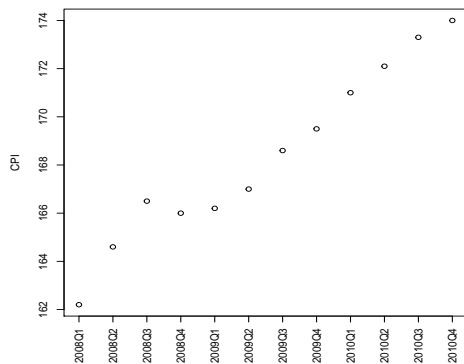
- ▶ Linear regression in R: `lm()`
- ▶ The Australian Consumer Price Index (CPI) data: quarterly CPIs from 2008 to 2010 <sup>‡</sup>

---

<sup>‡</sup>From Australian Bureau of Statistics, <http://www.abs.gov.au>.

# The CPI Data

```
year <- rep(2008:2010, each = 4)
quarter <- rep(1:4, 3)
cpi <- c(162.2, 164.6, 166.5, 166, 166.2, 167, 168.6, 169.5, 171,
        172.1, 173.3, 174)
plot(cpi, xaxt = "n", ylab = "CPI", xlab = "")
# draw x-axis, where 'las=3' makes text vertical
axis(1, labels = paste(year, quarter, sep = "Q"), at = 1:12, las = 3)
```



```
## correlation between CPI and year / quarter
cor(year, cpi)

## [1] 0.9096316

cor(quarter, cpi)

## [1] 0.3738028

## build a linear regression model with function lm()
fit <- lm(cpi ~ year + quarter)
fit

##
## Call:
## lm(formula = cpi ~ year + quarter)
##
## Coefficients:
## (Intercept)      year      quarter
## -7644.488      3.888      1.167
```

With the above linear model, CPI is calculated as

$$\text{cpi} = c_0 + c_1 * \text{year} + c_2 * \text{quarter},$$

where  $c_0$ ,  $c_1$  and  $c_2$  are coefficients from model fit.

What will the CPI be in 2011?

```
cpi2011 <- fit$coefficients[[1]] +  
  fit$coefficients[[2]] * 2011 +  
  fit$coefficients[[3]] * (1:4)  
cpi2011  
  
## [1] 174.4417 175.6083 176.7750 177.9417
```

With the above linear model, CPI is calculated as

$$\text{cpi} = c_0 + c_1 * \text{year} + c_2 * \text{quarter},$$

where  $c_0$ ,  $c_1$  and  $c_2$  are coefficients from model fit.

What will the CPI be in 2011?

```
cpi2011 <- fit$coefficients[[1]] +  
  fit$coefficients[[2]] * 2011 +  
  fit$coefficients[[3]] * (1:4)  
cpi2011  
  
## [1] 174.4417 175.6083 176.7750 177.9417
```

An easier way is to use function `predict()`.



More details of the model can be obtained with the code below.

```
attributes(fit)

## $names
## [1] "coefficients" "residuals" "effects"
## [4] "rank"         "fitted.values" "assign"
## [7] "qr"          "df.residual" "xlevels"
## [10] "call"        "terms"      "model"
##
## $class
## [1] "lm"

fit$coefficients

## (Intercept)      year      quarter
## -7644.487500    3.887500    1.166667
```

Function residuals(): differences between observed values and fitted values



```
# differences between observed values and fitted values
residuals(fit)
```

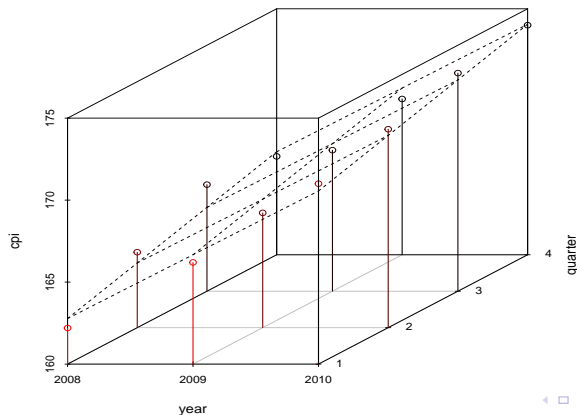
```
##           1           2           3           4           5
## -0.57916667  0.65416667  1.38750000 -0.27916667 -0.46666667
##           6           7           8           9          10
## -0.83333333 -0.40000000 -0.66666667  0.44583333  0.37916667
##           11          12
##  0.41250000 -0.05416667
```

```
summary(fit)
```

```
##
## Call:
## lm(formula = cpi ~ year + quarter)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.8333 -0.4948 -0.1667  0.4208  1.3875
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
```

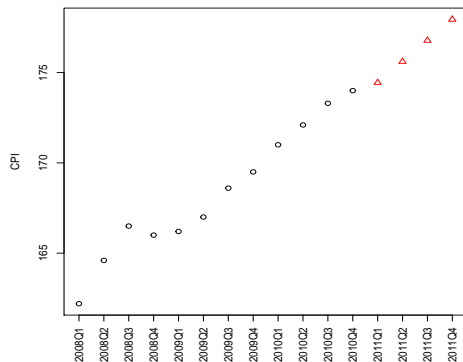
# 3D Plot of the Fitted Model

```
library(scatterplot3d)
s3d <- scatterplot3d(year, quarter, cpi, highlight.3d = T, type = "h",
  lab = c(2, 3)) # lab: number of tickmarks on x-/y-axes
s3d$plane3d(fit) # draws the fitted plane
```



# Prediction of CPIs in 2011

```
data2011 <- data.frame(year = 2011, quarter = 1:4)
cpi2011 <- predict(fit, newdata = data2011)
style <- c(rep(1, 12), rep(2, 4))
plot(c(cpi, cpi2011), xaxt = "n", ylab = "CPI", xlab = "", pch = style,
     col = style)
axis(1, at = 1:16, las = 3, labels = c(paste(year, quarter, sep = "Q"),
    "2011Q1", "2011Q2", "2011Q3", "2011Q4"))
```



Introduction

Linear Regression

**Generalized Linear Regression**

Decision Trees with Package party

Decision Trees with Package rpart

Random Forest

Online Resources

- ▶ Generalizes linear regression by allowing the linear model to be related to the response variable via a link function and allowing the magnitude of the variance of each measurement to be a function of its predicted value
- ▶ Unifies various other statistical models, including linear regression, logistic regression and Poisson regression
- ▶ Function `glm()`: fits generalized linear models, specified by giving a symbolic description of the linear predictor and a description of the error distribution

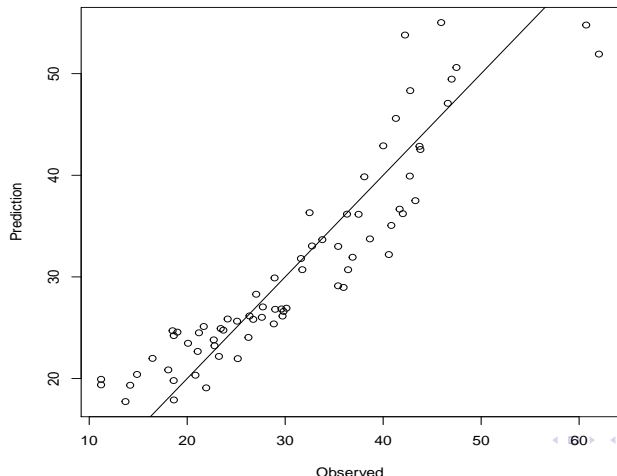
# Build a Generalized Linear Model

```
data("bodyfat", package="TH.data")
myFormula <- DEXfat ~ age + waistcirc + hipcirc + elbowbreadth +
              kneebreadth
bodyfat.glm <- glm(myFormula, family = gaussian("log"), data = bodyfat)
summary(bodyfat.glm)

##
## Call:
## glm(formula = myFormula, family = gaussian("log"), data = b...
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -11.5688  -3.0065   0.1266   2.8310  10.0966
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.734293   0.308949   2.377  0.02042 *
## age          0.002129   0.001446   1.473  0.14560
## waistcirc    0.010489   0.002479   4.231 7.44e-05 ***
## hipcirc      0.009702   0.003231   3.003  0.00379 **
## elbowbreadth 0.002355   0.045686   0.052  0.95905
## kneebreadth  0.063188   0.028193   2.241  0.02843 *
```

# Prediction with Generalized Linear Regression Model

```
pred <- predict(bodyfat.glm, type = "response")  
plot(bodyfat$DEXfat, pred, xlab = "Observed", ylab = "Prediction")  
abline(a = 0, b = 1)
```





Introduction

Linear Regression

Generalized Linear Regression

Decision Trees with Package party

Decision Trees with Package rpart

Random Forest

Online Resources

```
str(iris)

## 'data.frame': 150 obs. of 5 variables:
## $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
## $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1...
## $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1...
## $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0...
## $ Species : Factor w/ 3 levels "setosa","versicolor",....

# split data into two subsets: training (70%) and test (30%); set
# a fixed random seed to make results reproducible
set.seed(1234)
ind <- sample(2, nrow(iris), replace = TRUE, prob = c(0.7, 0.3))
train.data <- iris[ind == 1, ]
test.data <- iris[ind == 2, ]
```

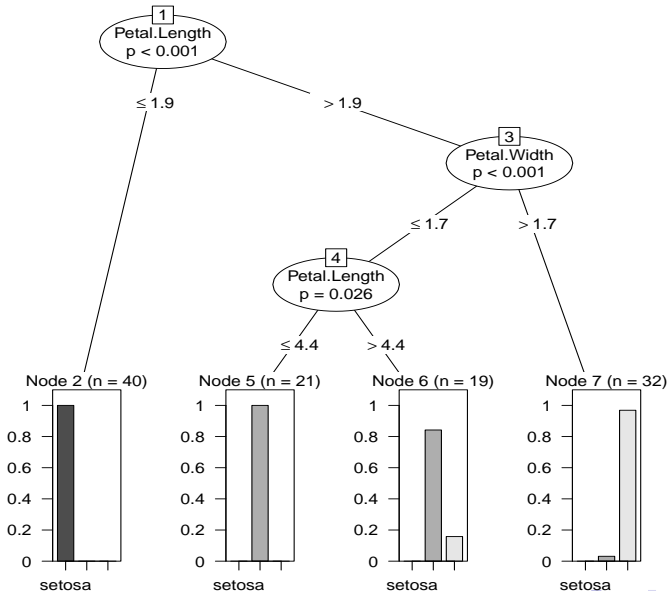
- ▶ Control the training of decision trees: MinSplit, MinBucket, MaxSurrogate and MaxDepth
- ▶ Target variable: Species
- ▶ Independent variables: all other variables

```
library(party)
myFormula <- Species ~ Sepal.Length + Sepal.Width + Petal.Length +
  Petal.Width
iris_ctree <- ctree(myFormula, data = train.data)
# check the prediction
table(predict(iris_ctree), train.data$Species)
```

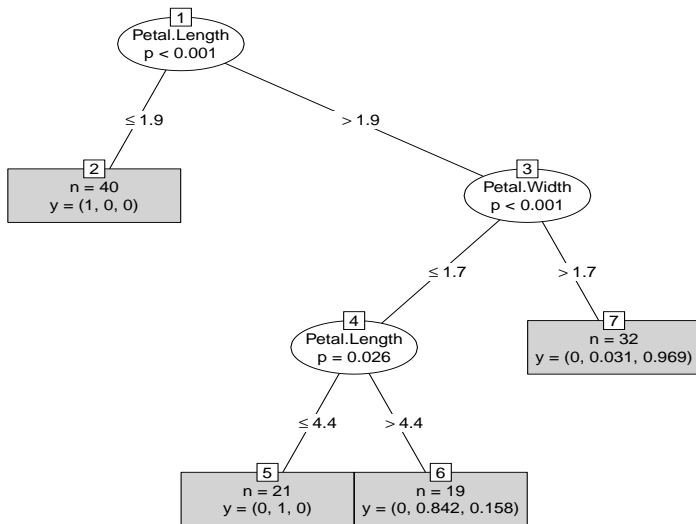
```
##
##          setosa versicolor virginica
## setosa          40           0         0
## versicolor       0           37         3
## virginica        0           1        31
```

```
print(iris_ctree)

##
##   Conditional inference tree with 4 terminal nodes
##
## Response:   Species
## Inputs:    Sepal.Length, Sepal.Width, Petal.Length, Petal.Width
## Number of observations:  112
##
## 1) Petal.Length <= 1.9; criterion = 1, statistic = 104.643
##   2)* weights = 40
## 1) Petal.Length > 1.9
##   3) Petal.Width <= 1.7; criterion = 1, statistic = 48.939
##     4) Petal.Length <= 4.4; criterion = 0.974, statistic = ...
##       5)* weights = 21
##       4) Petal.Length > 4.4
##         6)* weights = 19
##     3) Petal.Width > 1.7
##       7)* weights = 32
```



```
plot(iris_ctree, type = "simple")
```



```
# predict on test data  
testPred <- predict(iris_ctree, newdata = test.data)  
table(testPred, test.data$Species)
```

```
##  
## testPred      setosa versicolor virginica  
##   setosa          10           0           0  
##   versicolor       0           12          2  
##   virginica        0           0          14
```

Introduction

Linear Regression

Generalized Linear Regression

Decision Trees with Package party

Decision Trees with Package rpart

Random Forest

Online Resources



# The bodyfat Dataset

```
data("bodyfat", package = "TH.data")
```

```
dim(bodyfat)
```

```
## [1] 71 10
```

```
# str(bodyfat)
```

```
head(bodyfat, 5)
```

```
##      age DEXfat waistcirc hipcirc elbowbreadth kneebreadth
## 47  57  41.68    100.0    112.0         7.1         9.4
## 48  65  43.29     99.5    116.5         6.5         8.9
## 49  59  35.41     96.0    108.5         6.2         8.9
## 50  58  22.79     72.0     96.5         6.1         9.2
## 51  60  36.42     89.5    100.5         7.1        10.0
##      anthro3a anthro3b anthro3c anthro4
## 47      4.42      4.95      4.50      6.13
## 48      4.63      5.01      4.48      6.37
## 49      4.12      4.74      4.60      5.82
## 50      4.03      4.48      3.91      5.66
## 51      4.24      4.68      4.15      5.91
```

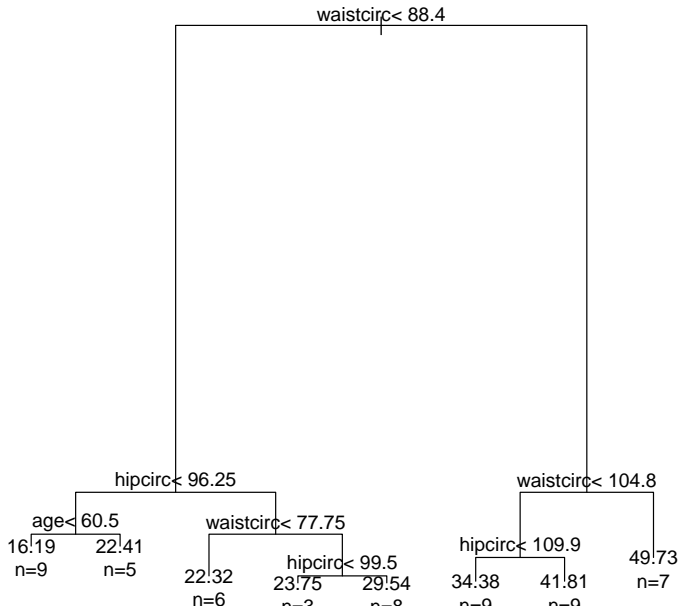
# Train a Decision Tree with Package rpart

```
# split into training and test subsets
set.seed(1234)
ind <- sample(2, nrow(bodyfat), replace=TRUE, prob=c(0.7, 0.3))
bodyfat.train <- bodyfat[ind==1,]
bodyfat.test <- bodyfat[ind==2,]
# train a decision tree
library(rpart)
myFormula <- DEXfat ~ age + waistcirc + hipcirc + elbowbreadth +
              kneebreadth
bodyfat_rpart <- rpart(myFormula, data = bodyfat.train,
                      control = rpart.control(minsplit = 10))
# print(bodyfat_rpart$cptable)
print(bodyfat_rpart)
plot(bodyfat_rpart)
text(bodyfat_rpart, use.n=T)
```

# The rpart Tree

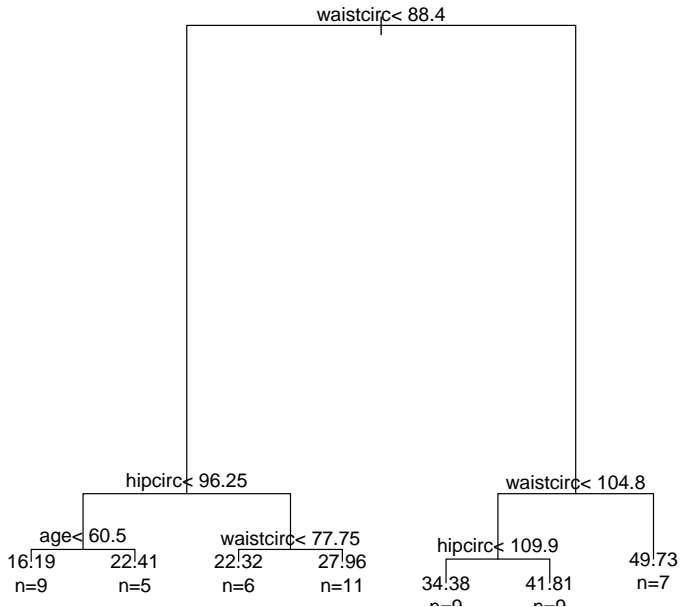
```
## n= 56
##
## node), split, n, deviance, yval
##      * denotes terminal node
##
## 1) root 56 7265.0290000 30.94589
##    2) waistcirc< 88.4 31  960.5381000 22.55645
##      4) hipcirc< 96.25 14  222.2648000 18.41143
##        8) age< 60.5 9    66.8809600 16.19222 *
##        9) age>=60.5 5    31.2769200 22.40600 *
##      5) hipcirc>=96.25 17  299.6470000 25.97000
##        10) waistcirc< 77.75 6    30.7345500 22.32500 *
##        11) waistcirc>=77.75 11  145.7148000 27.95818
##          22) hipcirc< 99.5 3    0.2568667 23.74667 *
##          23) hipcirc>=99.5 8    72.2933500 29.53750 *
##    3) waistcirc>=88.4 25 1417.1140000 41.34880
##      6) waistcirc< 104.75 18  330.5792000 38.09111
##        12) hipcirc< 109.9 9    68.9996200 34.37556 *
##        13) hipcirc>=109.9 9    13.0832000 41.80667 *
##      7) waistcirc>=104.75 7  404.3004000 49.72571 *
```

# The rpart Tree

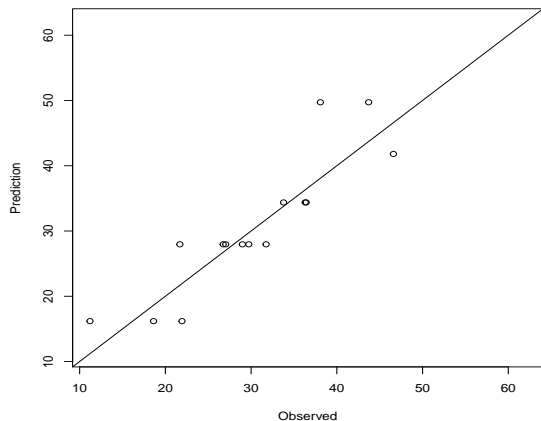


```
# select the tree with the minimum prediction error
opt <- which.min(bodyfat_rpart$cptable[, "xerror"])
cp <- bodyfat_rpart$cptable[opt, "CP"]
# prune tree
bodyfat_prune <- prune(bodyfat_rpart, cp = cp)
# plot tree
plot(bodyfat_prune)
text(bodyfat_prune, use.n = T)
```

# Selected Tree



```
DEXfat_pred <- predict(bodyfat_prune, newdata = bodyfat.test)
xlim <- range(bodyfat$DEXfat)
plot(DEXfat_pred ~ DEXfat, data = bodyfat.test, xlab = "Observed",
     ylab = "Prediction", ylim = xlim, xlim = xlim)
abline(a = 0, b = 1)
```



Introduction

Linear Regression

Generalized Linear Regression

Decision Trees with Package party

Decision Trees with Package rpart

**Random Forest**

Online Resources



- ▶ Package *randomForest*
  - ▶ very fast
  - ▶ cannot handle data with missing values
  - ▶ a limit of 32 to the maximum number of levels of each categorical attribute
  - ▶ extensions: *extendedForest*, *gradientForest*
- ▶ Package *party*: `cforest()`
  - ▶ not limited to the above maximum levels
  - ▶ slow
  - ▶ needs more memory

```
# split into two subsets: training (70%) and test (30%)
ind <- sample(2, nrow(iris), replace=TRUE, prob=c(0.7, 0.3))
train.data <- iris[ind==1,]
test.data <- iris[ind==2,]
# use all other variables to predict Species
library(randomForest)
rf <- randomForest(Species ~ ., data=train.data, ntree=100,
                   proximity=T)
```

```
table(predict(rf), train.data$Species)
```

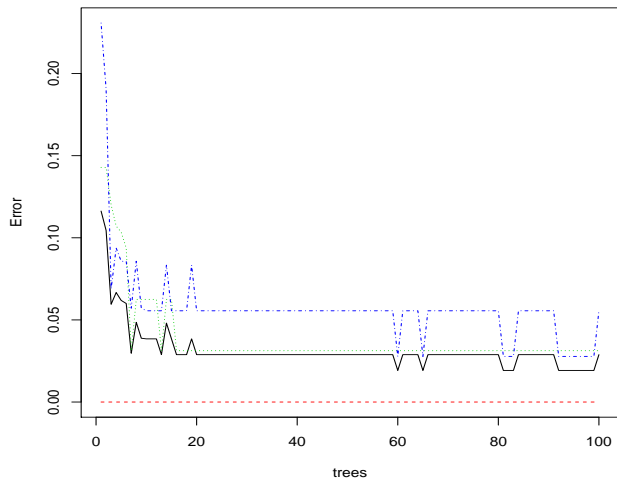
```
##
##           setosa versicolor virginica
## setosa           36           0           0
## versicolor        0           31           2
## virginica         0           1           34
```

```
print(rf)
```

```
##
## Call:
## randomForest(formula = Species ~ ., data = train.data, ntr...
##           Type of random forest: classification
##           Number of trees: 100
## No. of variables tried at each split: 2
##
##           OOB estimate of  error rate: 2.88%
## Confusion matrix:
##           setosa versicolor virginica class.error
## setosa           36           0           0 0.00000000
## versicolor        0           31           1 0.03125000
## virginica         0           2           34 0.05555556
```

# Error Rate of Random Forest

```
plot(rf, main = "")
```

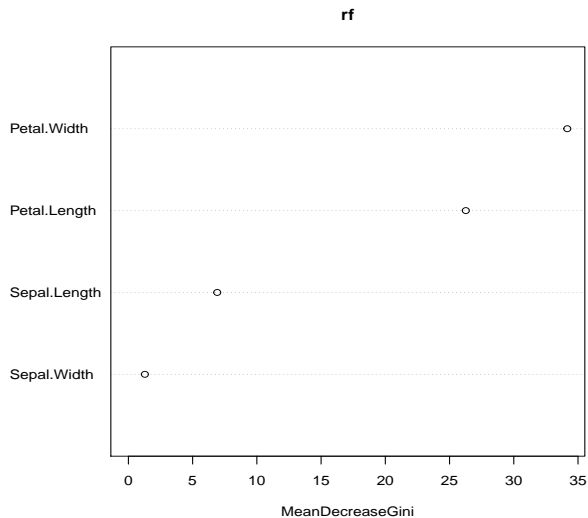


```
importance(rf)
```

```
##              MeanDecreaseGini
## Sepal.Length          6.913882
## Sepal.Width           1.282567
## Petal.Length          26.267151
## Petal.Width           34.163836
```

# Variable Importance

```
varImpPlot(rf)
```



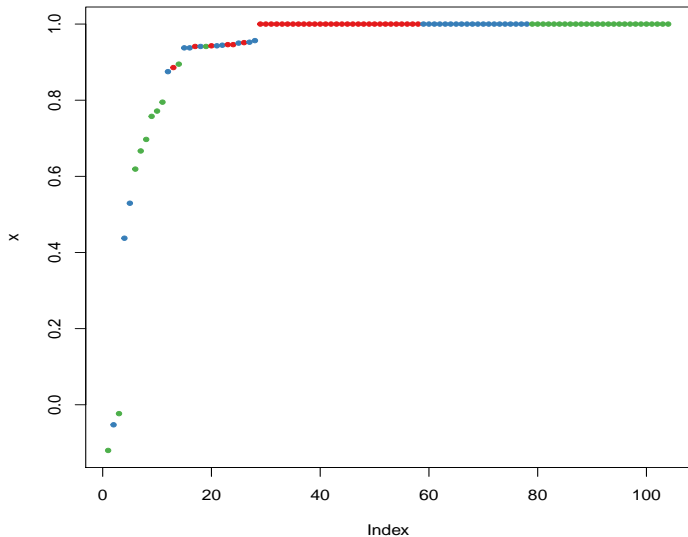
The margin of a data point is as the proportion of votes for the correct class minus maximum proportion of votes for other classes. Positive margin means correct classification.

```
irisPred <- predict(rf, newdata = test.data)
table(irisPred, test.data$Species)

##
## irisPred      setosa versicolor virginica
##   setosa      14          0           0
##   versicolor   0          17          3
##   virginica    0           1          11

plot(margin(rf, test.data$Species))
```

# Margin of Predictions





Introduction

Linear Regression

Generalized Linear Regression

Decision Trees with Package party

Decision Trees with Package rpart

Random Forest

Online Resources

- ▶ Chapter 4: Decision Trees and Random Forest & Chapter 5: Regression, in book *R and Data Mining: Examples and Case Studies*

<http://www.rdatamining.com/docs/RDataMining.pdf>

- ▶ R Reference Card for Data Mining

<http://www.rdatamining.com/docs/R-refcard-data-mining.pdf>

- ▶ Free online courses and documents

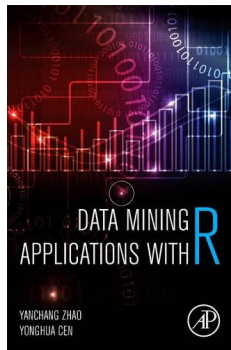
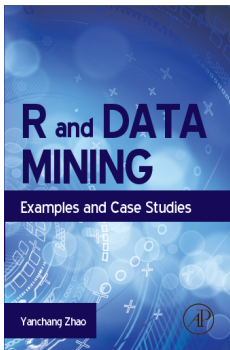
<http://www.rdatamining.com/resources/>

- ▶ RDataMining Group on LinkedIn (24,000+ members)

<http://group.rdatamining.com>

- ▶ Twitter (3,000+ followers)

@RDataMining



Thanks!

Email: [yanchang\(at\)RDataMining.com](mailto:yanchang(at)RDataMining.com)

Twitter: @RDataMining