Introduction to Computer Programming with R (FOR 6934)

Qing Zhao (School of Forest Resources & Conservation, qing.zhao@ufl.edu)
Daijiang Li (Department of Wildlife Ecology & Conservation, dli1@ufl.edu)
Denis Valle (School of Forest Resources & Conservation, drvalle@ufl.edu)

Class three

- Understand and manipulate strings
- Understand data index
- Visualize results of data selection

A character string can contain multiple parts

```
"Peter Parker"
```

```
## [1] "Peter Parker"
```

```
c("X1", "X2", "Y1", "Y2", "Y3")
```

```
## [1] "X1" "X2" "Y1" "Y2" "Y3"
```

Use paste() to create character strings with multiple parts

- paste()

Functions to extract parts from character strings

- strsplit()
- substr()

Sample code

Create character strings with multiple parts

```
a <- "Peter"
b <- "Parker"
paste(a, b, sep=' ')
```

```
## [1] "Peter Parker"
```

```
paste(a, b, sep='+')
```

```
## [1] "Peter+Parker"
```

Sample code

Create character strings with repeated parts

```
paste('X', 1:5, sep='_')
```

```
## [1] "X_1" "X_2" "X_3" "X_4" "X_5"
```

```
paste('X', 1:5, sep='')
```

```
## [1] "X1" "X2" "X3" "X4" "X5"
```

Sample code

Split character strings based on the common seperator

```
a <- c('Peter Parker', 'Mary Jane')
strsplit(a, split=' ')
```

```
## [[1]]
## [1] "Peter"  "Parker"
##
## [[2]]
## [1] "Mary" "Jane"
```

```
a <- c(paste('X', 1:2, sep='_'), paste('Y', 1, sep='_'))
a
```

```
## [1] "X_1" "X_2" "Y_1"
```

```
b1 <- strsplit(a, split='_')
b1
```

```
## [[1]]
## [1] "X" "1"
##
## [[2]]
## [1] "X" "2"
##
## [[3]]
## [1] "Y" "1"
```

```
b2 <- unlist(b1)
b2
```

```
## [1] "X" "1" "X" "2" "Y" "1"
```

```
b2[seq(from=1, to=6, by=2)]
```

```
## [1] "X" "X" "Y"
```

```
b2[seq(from=2, to=6, by=2)]
```

```
## [1] "1" "2" "1"
```

Sample code

Substract parts based on position

```
a
```

```
## [1] "X_1" "X_2" "Y_1"
```

```
substr(a, start=1, stop=1)
```

```
## [1] "X" "X" "Y"
```

```
substr(a, start=3, stop=3)
```

```
## [1] "1" "2" "1"
```

```
a2 <- paste('Z', 9:11, sep='')
a2
```

```
## [1] "Z9"  "Z10" "Z11"
```

```
strsplit(a2, split='')
```

```
## [[1]]
## [1] "Z" "9"
##
## [[2]]
## [1] "Z" "1" "0"
##
## [[3]]
## [1] "Z" "1" "1"
```

```
a2
```

```
## [1] "Z9"  "Z10" "Z11"
```

```
substr(a2, start=1, stop=1)
```

```
## [1] "Z" "Z" "Z"
```

```
substr(a2, start=2, stop=3)
```

```
## [1] "9"  "10" "11"
```

Sample code

substr() can be used on numeric values

```r
d <- c(20160101, 20160204, 20170212, 20170220)
d
```

```
## [1] 20160101 20160204 20170212 20170220
```

```r
substr(d, start=1, stop=4)
```

```
## [1] "2016" "2016" "2017" "2017"
```

Some comments

- Extract parts separated by the same separators (e.g. space, '-') with strsplit() function
- Extract parts based on their positions with substr() function

Data index

- Indices indicate positions of values in data structures
- Indices can be integers in square brackets
  - [3]
  - [1:5]
  - [2, 3]
- Indices can be names
  - [c('X1', 'Y3')]
- Indices can be found using which() function

Sample code

Use indices to find values in vectors

```r
a <- c(1:10)^2
a
```

```
##  [1]   1   4   9  16  25  36  49  64  81 100
```

```r
a[3]
```

```
## [1] 9
```

```r
a[5:10]
```

```
## [1]  25  36  49  64  81 100
```

Sample code

Minus can be used to eliminate some elements

```r
a
```

```
##  [1]   1   4   9  16  25  36  49  64  81 100
```

```r
a[-5]
```

```
## [1]   1   4   9  16  36  49  64  81 100
```

```r
a[-c(2,7)]
```

```
## [1]   1   9  16  25  36  64  81 100
```

Sample code

Use indices to find values in matrices

```
mat <- matrix(c('dog', 'cat', 'horse', 'gator', 'bird', 'fish'), 3, 2)
mat
```

```
##      [,1]    [,2]
## [1,] "dog"   "gator"
## [2,] "cat"   "bird"
## [3,] "horse" "fish"
```

```
mat[1,2]
```

```
## [1] "gator"
```

```
mat
```

```
##      [,1]    [,2]
## [1,] "dog"   "gator"
## [2,] "cat"   "bird"
## [3,] "horse" "fish"
```

```
mat[,1]
```

```
## [1] "dog"   "cat"   "horse"
```

```
mat[3,]
```

```
## [1] "horse" "fish"
```

```
mat
```

```
##      [,1]    [,2]
## [1,] "dog"   "gator"
## [2,] "cat"   "bird"
## [3,] "horse" "fish"
```

```
mat[c(1,3),2]
```

```
## [1] "gator" "fish"
```

```
mat[-1,1]
```

```
## [1] "cat"   "horse"
```

Sample code

Use names to index values

```
mat <- matrix(1:18, nrow=2, ncol=9)
colnames(mat) <- c(paste('X', 1:3, sep='_'), paste('Y', 1:6, sep='_'))
mat
```

```
##      X_1 X_2 X_3 Y_1 Y_2 Y_3 Y_4 Y_5 Y_6
## [1,]   1   3   5   7   9  11  13  15  17
## [2,]   2   4   6   8  10  12  14  16  18
```

```
mat[,c('X_2', 'Y_4')]
```

```
##      X_2 Y_4
## [1,]   3  13
## [2,]   4  14
```

Sample code

Paste() can be used for indexing data

```
mat
```

```
##      X_1 X_2 X_3 Y_1 Y_2 Y_3 Y_4 Y_5 Y_6
## [1,]   1   3   5   7   9  11  13  15  17
## [2,]   2   4   6   8  10  12  14  16  18
```

```
mat[,paste('Y', 2:5, sep='_')]
```

```
##      Y_2 Y_3 Y_4 Y_5
## [1,]   9  11  13  15
## [2,]  10  12  14  16
```

Sample code

Use logical values as indices

```
mat
```

```
##      X_1 X_2 X_3 Y_1 Y_2 Y_3 Y_4 Y_5 Y_6
## [1,]   1   3   5   7   9  11  13  15  17
## [2,]   2   4   6   8  10  12  14  16  18
```

```
mat[c(FALSE, TRUE),]
```

```
## X_1 X_2 X_3 Y_1 Y_2 Y_3 Y_4 Y_5 Y_6
##   2   4   6   8  10  12  14  16  18
```

Sample code

Indexing components in lists

```
a <- c('dog', 'cat')
b <- c(15, 1)
lst <- list(a, b)
lst
```

```
## [[1]]
## [1] "dog" "cat"
##
## [[2]]
## [1] 15  1
```

```
lst[[1]]
```

```
## [1] "dog" "cat"
```

```
lst[[1]][2]
```

```
## [1] "cat"
```

Sample code

Indexing components in lists using names

```
a <- c('dog', 'cat')
b <- c(15, 1)
lst <- list(animal=a, age=b)
lst
```

```
## $animal
## [1] "dog" "cat"
##
## $age
## [1] 15  1
```

```
lst[[1]]
```

```
## [1] "dog" "cat"
```

```
lst$animal
```

```
## [1] "dog" "cat"
```

```
lst$age[2]
```

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

Sample code

Use indices to change values

```
mat
```

```
##      X_1 X_2 X_3 Y_1 Y_2 Y_3 Y_4 Y_5 Y_6
## [1,]   1   3   5   7   9  11  13  15  17
## [2,]   2   4   6   8  10  12  14  16  18
```

```
mat[1,1] <- 100
mat
```

```
##      X_1 X_2 X_3 Y_1 Y_2 Y_3 Y_4 Y_5 Y_6
## [1,] 100   3   5   7   9  11  13  15  17
## [2,]   2   4   6   8  10  12  14  16  18
```

```
mat
```

```
##      X_1 X_2 X_3 Y_1 Y_2 Y_3 Y_4 Y_5 Y_6
## [1,] 100   3   5   7   9  11  13  15  17
## [2,]   2   4   6   8  10  12  14  16  18
```

```
mat[,'Y_6'] <- c(1000, 2000)
mat
```

```
##      X_1 X_2 X_3 Y_1 Y_2 Y_3 Y_4 Y_5  Y_6
## [1,] 100   3   5   7   9  11  13  15 1000
## [2,]   2   4   6   8  10  12  14  16 2000
```

Sample code

Find indices based on conditions

```
a <- c(1:10)^2
a
```

```
##  [1]   1   4   9  16  25  36  49  64  81 100
```

```
a == 16
```

```
##  [1] FALSE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
which(a == 16)
```

```
## [1] 4
```

```
a
```

```
##  [1]   1   4   9  16  25  36  49  64  81 100
```

```
a > 80
```

```
##  [1] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE  TRUE
```

```
which(a > 80)
```

```
## [1]  9 10
```

Sample code

Find values in vectors based on conditions

```
a
```

```
##  [1]   1   4   9  16  25  36  49  64  81 100
```

```
which(a < 50)
```

```
## [1] 1 2 3 4 5 6 7
```

```
a[which(a < 50)]
```

```
## [1]  1  4  9 16 25 36 49
```

Sample code

You can omit which() function, only use the condition

```
a
```

```
##  [1]   1   4   9  16  25  36  49  64  81 100
```

```
a < 50
```

```
##  [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE
```

```
a[a < 50]
```

```
## [1]  1  4  9 16 25 36 49
```

Sample code

Use indices to find values in NBA data

```
dat <- read.csv('c:/data/NBA.csv')
head(dat, n=3)
```

```
##                   Team Win Lose Rank Conference
## 1      Boston Celtics  53   29    1    Eastern
## 2 Cleveland Cavaliers  51   31    2    Eastern
## 3     Toronto Raptors  51   31    3    Eastern
```

```
tail(dat, n=3)
```

```
##                        Team Win Lose Rank Conference
## 28 Minnesota Timberwolves   31   51   13    Western
## 29      Los Angeles Lakers  26   56   14    Western
## 30            Phoenix Suns  24   58   15    Western
```

Sample code

Find values in data frames based on conditions

```
dat$Team <- as.character(dat$Team)
dat$Team[which(dat$Win > 60)]
```

```
## [1] "Golden State Warriors" "San Antonio Spurs"
```

```
dat$Team[which(dat$Win > 50 & dat$Conference == 'Western')]
```

```
## [1] "Golden State Warriors" "San Antonio Spurs"     "Houston Rockets"
## [4] "Los Angeles Clippers " "Utah Jazz"
```

```
dat$Team[which(dat$Win > 50 & dat$Conference == 'Eastern')]
```

```
## [1] "Boston Celtics"      "Cleveland Cavaliers" "Toronto Raptors"
```

```
length(dat$Team[which(dat$Win > 40 & dat$Conference == 'Western')])
```

```
## [1] 8
```

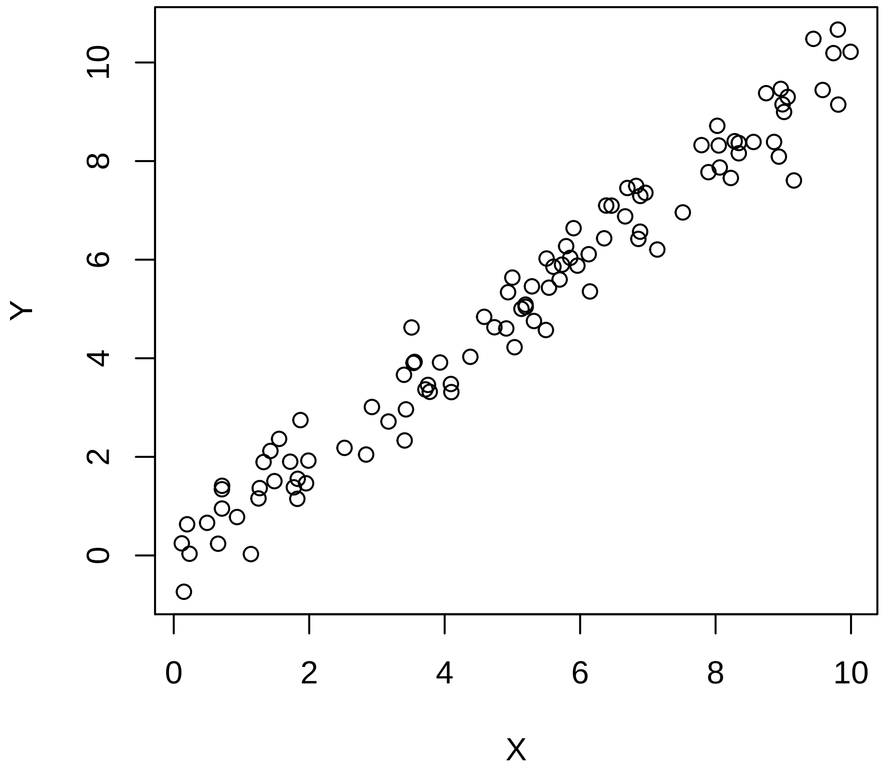
```
length(dat$Team[which(dat$Win > 40 & dat$Conference == 'Eastern')])
```

```
## [1] 9
```

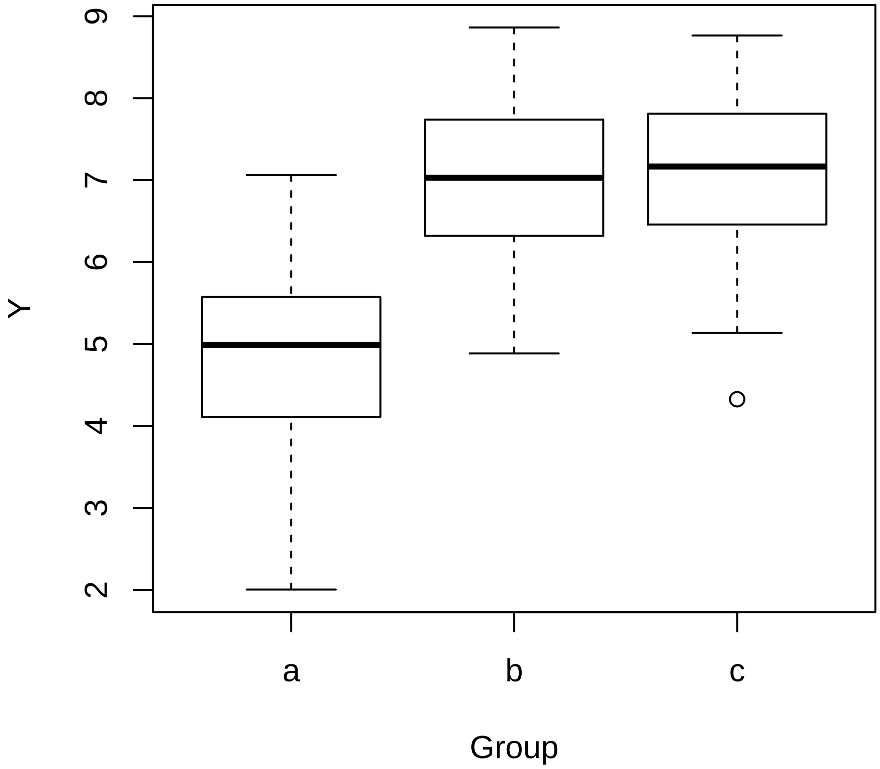
Graphing techniques

- plot() is useful to visualize the correlation between two numeric variables
- boxplot() is useful to visualize differences among multiple groups

Scatter plot



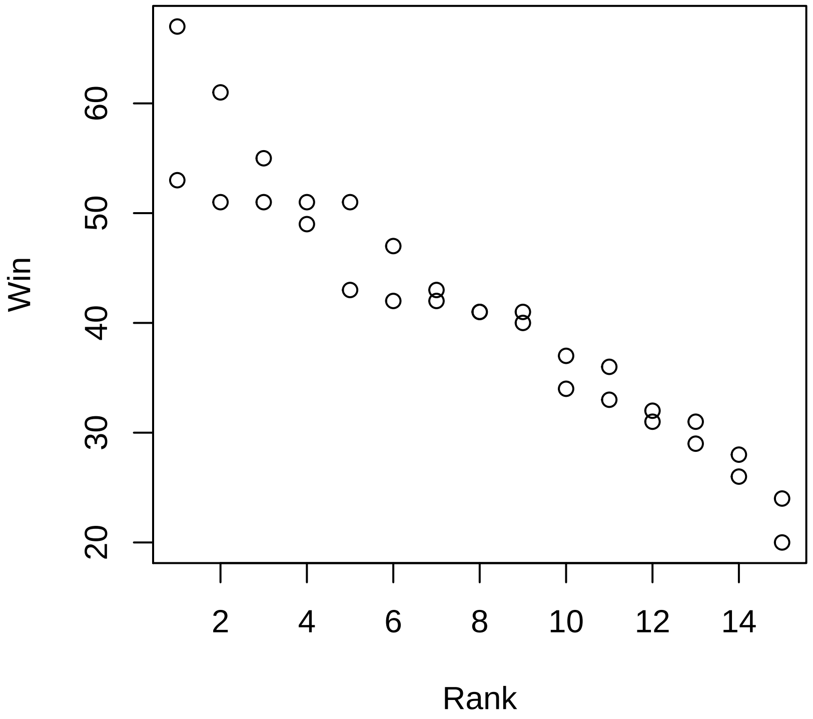
Box plot



Sample code

Scatter plot between wins and ranks

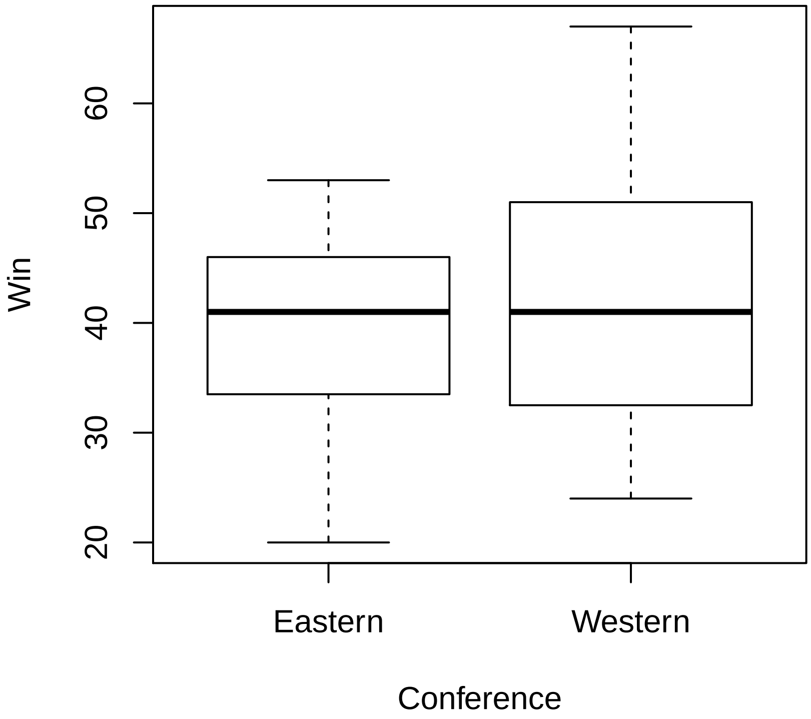
```
par(mar=c(7,5,1,3))
plot(dat$Win ~ dat$Rank, xlab='Rank', ylab='Win')
```



Sample code

Box plot of wins between east and west conferences

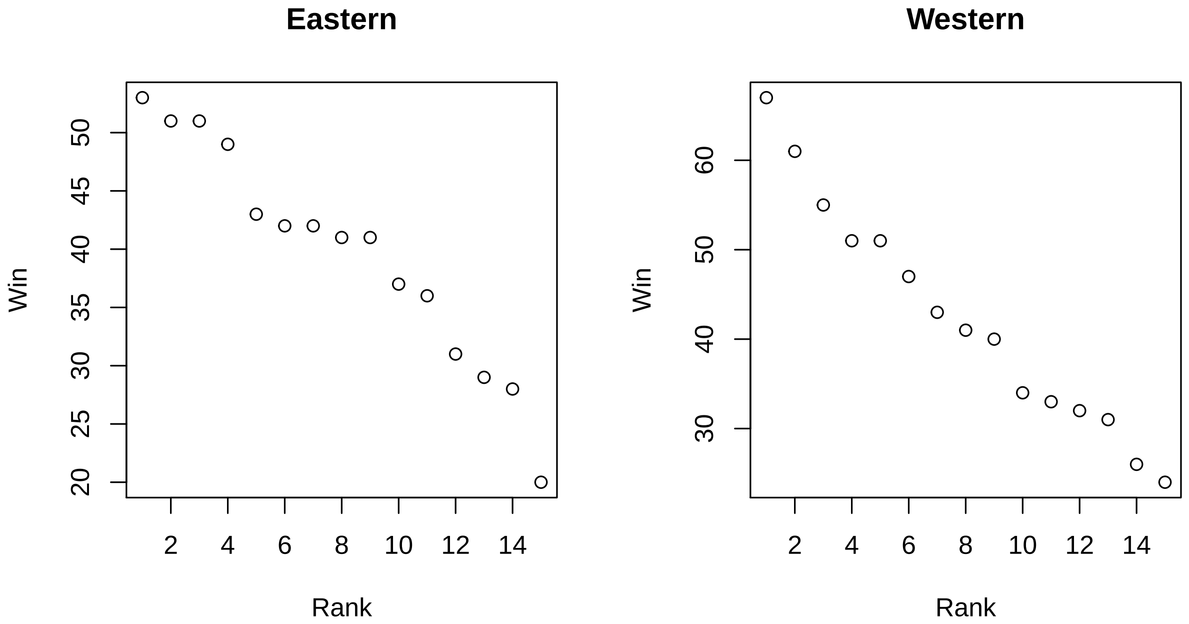
```
par(mar=c(7,5,1,3))
boxplot(dat$Win ~ dat$Conference, xlab='Conference', ylab='Win')
```



Sample code

Scatter plot between wins and ranks for each conference

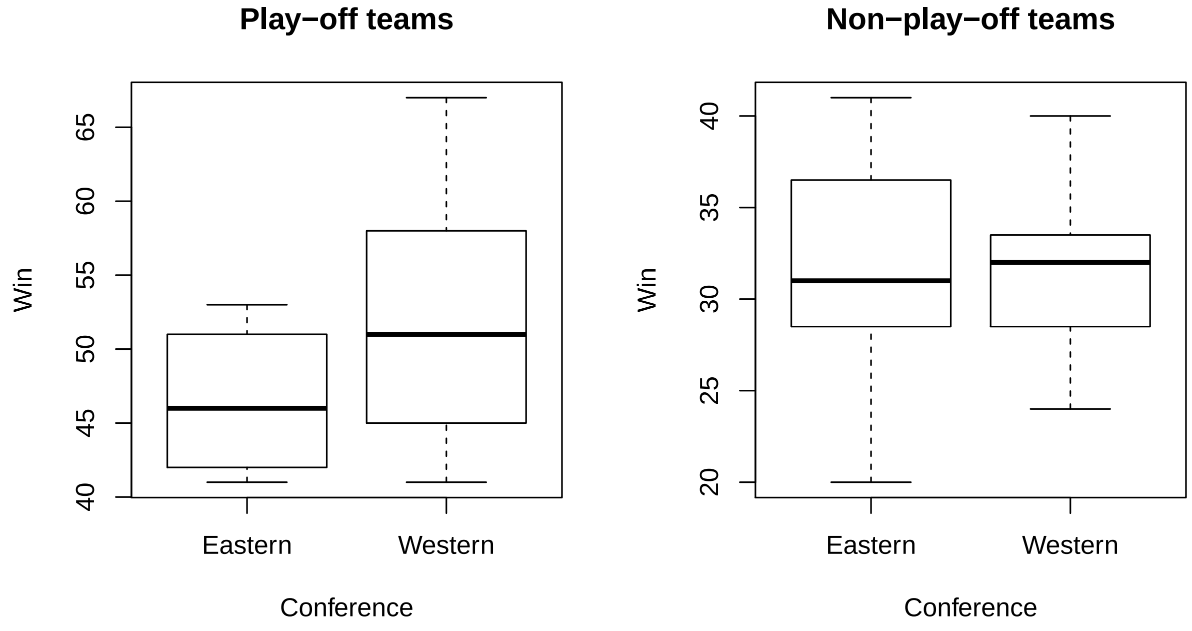
```
dat_east <- dat[which(dat$Conference == 'Eastern'),]
dat_west <- dat[which(dat$Conference == 'Western'),]
par(mfrow=c(1,2))
plot(dat_east$Win ~ dat_east$Rank, xlab='Rank', ylab='Win', main='Eastern')
plot(dat_west$Win ~ dat_west$Rank, xlab='Rank', ylab='Win', main='Western')
```



Sample code

Boxplot plot of wins between east and west conferences, play-off and non-play-off teams separately

```
dat_playoff <- dat[which(dat$Rank <= 8),]
dat_nonplayoff <- dat[which(dat$Rank >= 9),]
par(mfrow=c(1,2))
boxplot(dat_playoff$Win ~ dat_playoff$Conference,
        xlab='Conference', ylab='Win', main='Play-off teams')
boxplot(dat_nonplayoff$Win ~ dat_nonplayoff$Conference,
        xlab='Conference', ylab='Win', main='Non-play-off teams')
```



Summary

- Use paste(), strsplit() and substr() to manipulate strings
- Understand data index, use index to select data
- Combine indexing in graphing techniques

Thank you and see you next class