LAB 3

Data Manipulation

- Filtering
  - attach(iris);setosa<-iris[Species=="setosa",]
  - mean.greater<-iris[Sepal.Length>=mean(Sepal.Length),]
  - setosa.greater<-iris[Sepal.Length>=median(Sepal.Length) & Species=="setosa",]; detach(iris)
  - iris_num1<-iris[,-c(5)]
  - iris[which(iris$Sepal.Length>5 & iris$Sepal.Length<5.5),]

  - Data.sepal<-subset(iris, select=c(Sepal.Length, Sepal.Width))
  - subset(iris, select=Sepal.Length:Petal.Width)
  - subset(iris,iris$Species=='setosa' & iris$Sepal.Length>5.2)
  - with(iris, subset(iris, Species=='setosa' & Sepal.Length>5.2))
  - attach(iris); subset(iris,Species=='setosa' & Sepal.Length>5.2); detach(iris)

Data Manipulation

- 정렬: order(), rev()
  - iris1<-iris[order(iris$Sepal.Length),]
  - iris2<-iris[rev(order(iris$Sepal.Length)),]
  - iris3<-iris[order(iris$Sepal.Length, iris$Sepal.Width),]

Data Manipulation

- apply()
  - Applies a simple function over dimensions of a data structure
  - So input is anything that has rows and columns (matrix, data frame, array, but not lists or vectors)

```
> args(apply)
function (X, MARGIN, FUN, ...)
NULL
```

Structure with a dimension

Dimension number (1 = rows, 2 = columns)

Function to apply

Additional arguments to the function

	Description	Input	Output
apply	Applys a function over dimensions of a data structure	Structure with a "dimension"	A single mode structure
lapply	Applys a function to elements of a list or vector	A list or vector	A list
sapply	Applys a function to elements of a list or vector	A list or vector	A "simplified list"
tapply	Applys a function to a vector by factor(s)	A Vector + Factor(s)	Depends on # factors
by	Applys a function to a data frame by factor(s)	A Data Frame + Factor(s)	A "by" object (which is a list)
aggregate	Applys a function to columns of a data frame by factor(s)	A Data frame + Factor(s)	A Data frame

Data Manipulation

- Sum / Mean of each attribute
  - colSums(iris[1:4]) # method 1
  - apply(iris[1:4],2,sum) # method 2
  - lapply(iris[1:4],sum) # method 3
  - sapply(iris[1:4],sum) # method 4
  - colMeans(iris[1:4]) # method 1
  - apply(iris[1:4],2,mean) # method 2
  - sapply(iris[1:4],mean) # method 3
- Filtering
  - iris_num2<-iris[, sapply(iris,is.numeric)]
  - iris_fac<-iris[, sapply(iris,is.factor)]
- Sum / Mean of each record
  - rowSums(iris[-5]) # method 1
  - apply(iris[-5],1,sum) # method 2
  - rowMeans(iris[-5]) # method 1
  - apply(iris[-5],1,mean) # method 2

Data Manipulation

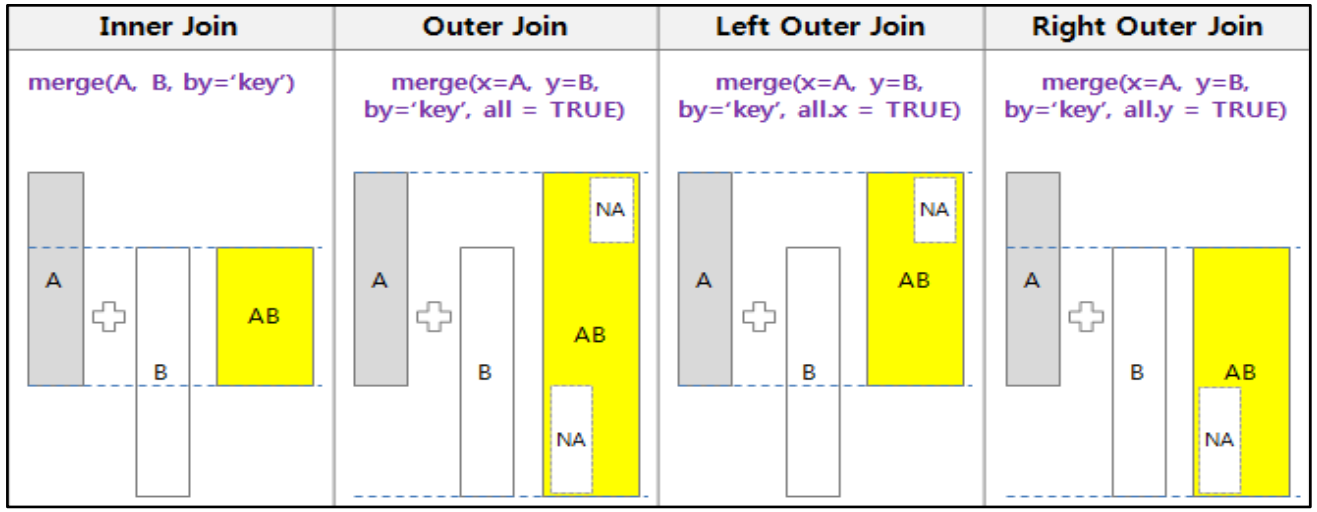
- Sum / Mean of values in each group (factor)
  - tapply(iris$Sepal.Length,iris$Species,sum)
  - by(iris$Sepal.Length, iris$Species, sum)

  - attach(iris); tapply(Sepal.Length,Species,sum)
  - with(iris,tapply(Sepal.Length,Species,sum))

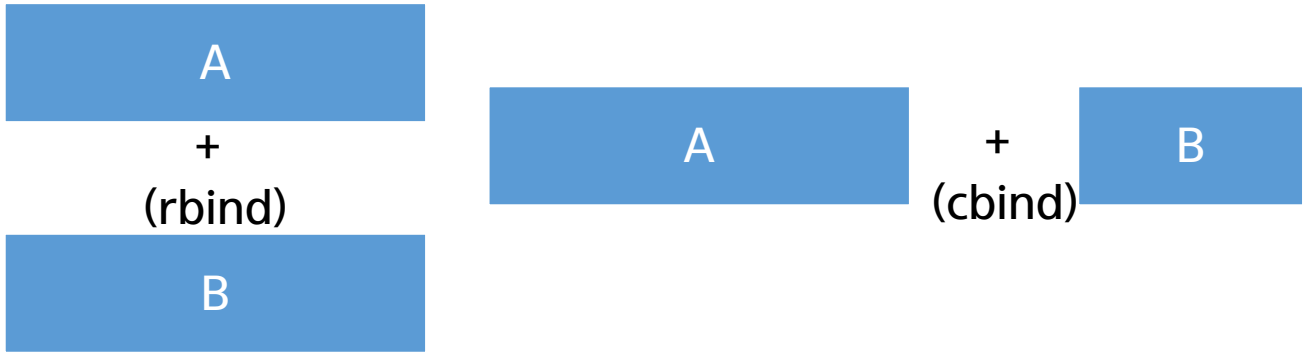
  - attach(iris); tapply(Sepal.Length,Species,mean)
  - with(iris,by(Sepal.Length,Species,mean))

Data Manipulation

- Merge two data frames
  - merge(df1, df2, by.x="", by.y="")



- rbind(), cbind()



Data Manipulation

- cbind()
  - iris_s1<-iris[,c(4,3)]; head(iris_s1)
  - iris_s2<-iris[,c(2,1)]; head(iris_s2)
  - iris_cbind<-cbind(iris_s1, iris_s2); head(iris_cbind)

- rbind()
  - iris_s1<-iris[c(51:100),]; head(iris_s1)
  - iris_s2<-iris[c(1:50),]; head(iris_s2)
  - iris_rbind<-rbind(iris_s1, iris_s2); head(iris_rbind); tail(iris_rbind)

Data Manipulation

- Merge()
  - d1<-data.frame(kids=c("jack","jill","jillian","john"),
    states = c("CA","MA","MA","HI"), stringsAsFactors = F)
  - d2<-data.frame(kids=c("jill","lillian","jack"), ages=c(10,7,12), stringsAsFactors = F)
  - d3<-data.frame(pals=c("jack","jill","lillian"), ages=c(12,10,7),stringsAsFactors=F)

  - (a.inner<-merge(d1,d2)) # inner join
  - (a.left<- merge(d1,d2,by="kids",all.x=T)) # left outer join
  - (a.right<- merge(d1,d2,by="kids",all.y=T)) # right outer join
  - (a.all<- merge(d1,d2,by="kids",all=T)) # outer join
  - (b.inner<-merge(d1,d3,by.x="kids",by.y="pals")) # inner join with different field names
  - (b.outer<-merge(d1,d3,by.x="kids",by.y="pals",all=T)) # outer join with different field names