dplyr: powerful collection of R packages that are data tools for transforming and visualizing data

Pipe Operator: %>%: together functions consectively

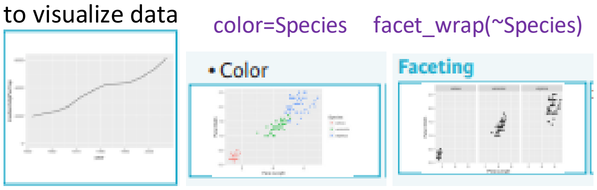
Functions (use logical sense to determine sequential):

```
iris %>%
  group_by(Species) %>%
  summarize(medianSL=median(Sepal.Length),
            maxSL=max(Sepal.Length))
```

filter() allows you to select a subset of rows in a data frame
arrange() sorts dataset in ascending or descending order
mutate() allows you to update or create new columns
summarize() turn many observations into single data point
group_by() allows you to summarize within groups
distinct(),n_distinct() Subset distinct/unique rows
na_if() Convert values to NA
top_n() If n is positive, chose top rows, otherwise if negative

ggplot: implements grammar of graphics, can use it to visualize data

color=Species facet_wrap(~Species)



aes() Construct aesthetic mappings
geom_abline(), geom_hline() Ref lines: horizontal, vertical
labs(), xlab(), ylab() Modify axis, legend, and plot labels
lims(), xlim(), ylim() Set scale limits
theme() Modify theme such as grey, light, dark

```
by_year <- gapminder %>%
  group_by(year) %>%
  summarize(medianGdpPerCap=median(gdpPercap))
ggplot(by_year, aes(x=year,
                    y=medianGdpPerCap))+
  geom_line()+
  expand_limits(y=0)
```

Date and Time in R: Abbreviations

%d	Day of month (decimal)
%m	Month (decimal)
%b	Month (abbv.)
%B	Month (full name)
%y	Year (2 digit)
%Y	Year (4 digit)
%W	Decimal week of the year (starting on Monday)

%a	Weekday (abbv.)
%A	Full weekday
%w	Decimal Weekday (0=Sunday)
%j	Day of year (decimal)
%W	Week of year (starting Sun)

Transform date: Example code

End-result format

> format(as.Date(x), "%Y/%m/%d")
> strptime(as.character(X$date), "%d/%m/%Y")

Time-zone

Difference of times: > difftime(time1, time2, tz, units = c("auto", "secs", "mins", "hours", "days","weeks"))

Interval: > interval(start = NULL, end = NULL, tzone = tz(start))

Print Current Date/Time: Sys.Date(), Sys.time()

Lubridate function: library(lubridate)

```
> ymd_hms("2012-12-31 23:59:59")
## [1] "2012-12-31 23:59:59 UTC"
> ldate <- mdy_hms("12/31/2012 23:59:59")
> ldate + seconds(1)
## [1] "2013-01-01 UTC"
```

Shiny by RStudio

Three components: Input, Output, Server

```
> library(shiny) ## Template
> ui <- fluidPage(
numericInput(inputId = "n",
"Sample size", value = 25),
plotOutput(outputId = "hist") )
> server <- function(input, output)
{ output$hist <- renderPlot({
hist(rnorm(input$n)) }) }
> shinyApp(ui = ui, server = server)
```

submitButtion()	dataTableOutput()
checkboxGroupInput()	htmlOutput()
dateRangeinput()	imageOutput()
fileInput()	tableOutput()
numericInput()	plotOutput()
radioButtons()	uiOutput()
sliderInput()	verbatimTextOutput()
textInput()	textOutput()

Association Pattern Mining Rules in R: arules, eclat

```
# Read the data with read_transactions command
> txn <- read.transactions(file="baskets_rdb.csv",
rm.duplicates=F, format="single",sep=",")
# Look at the transactions data  > inspect(txn)
# Numerical and Graphical Analyze
> itemFrequency(txn); itemFrequencyPlot(txn)
# Running the Algorithm; finding patterns
> pattern <- apriori(txn, parameter = list(sup = 0.5))
> rules  <- eclat(data = dataset, parameter = list
(support = 0.003, minlen = 2))
> inspect(sort(rules, by = 'support')[1:10])
```

Support: fraction of transactions that contain both X and Y

$supp(X \to Y) = (\sigma(X \cup Y))/(|T|)$

Confidence: fraction of Y in transactions that contain X

$conf(X \to Y) = (\sigma(X \cup Y))/(\sigma(X))$

$P(A \cap B) = P(A) \times P(B) \Rightarrow$ statistical indep.

$P(A \cap B) > P(A) \times P(B) \Rightarrow$ +ve correlated

$P(A \cap B) < P(A) \times P(B) \Rightarrow$ -ve correlated

$Lift = (P(Y|X))/(P(Y))$

$Interest = (P(X,Y)) / (P(X)P(Y))$

Apriori Principle: If an itemset is freq, then all of its subsets must also be freq.

For m data points, there are $m(m-1)/2$ pairs.

- f_{00} = number of pairs that are from different classes and clustered into different groups
- f_{01} = number of pairs that are from different classes and clustered into the same group
- f_{10} = number of pairs that are from the same class and clustered into different groups
- f_{11} = number of pairs that are from the same class and clustered into the same group

$$\text{Rand Statistic} = \frac{f_{00}+f_{11}}{f_{00}+f_{01}+f_{10}+f_{11}} \qquad \text{Jaccard Coefficient} = \frac{f_{11}}{f_{01}+f_{10}+f_{11}}$$

graph based view for cohesion and proximity, same formula for prototype-based view

$$cohesion(C_i) = \sum_{\mathbf{x},\mathbf{y} \in C_i} proximity(\mathbf{x},\mathbf{y}) \qquad separation(C_i, C_j) = \sum_{\mathbf{x} \in C_i, \mathbf{y} \in C_j} proximity(\mathbf{x},\mathbf{y})$$

PCA Example Code

```
X <- data[,3:ncol(data)];X <- scale(X,center=F,scale=F)
PCA_results <- princomp(X)
names(PCA_results) ## Names that are available
V <- PCA_results$loadings
Y <- PCA_results$scores #transformed-each col is PC
# Check the variances
s1 <- apply(X,2,var) # variances of each column in X
s2 <- apply(Y,2,var) # variances of each column in Y
```

SVD Example Code

```
X <- data[,3:ncol(data)];X <- scale(X,center=F,scale=F)
SVD_results <- svd(X)
names(SVD_results)
lambdas2 <- SVD_results$d     # singular values of x
U <- SVD_results$u  # columns contain the left singular vectors (n by 1) of x
V <- SVD_results$v  # columns contain the right singular vectors (p by 1) of x
Y2 = as.matrix(X) %*% V        # transformed
```

Hierarchical Clustering

```
data(iris) # load internal sample data
mydata <- iris[,1:4]  # select variables of interest
mydata  <- scale(mydata) # standardize variables
d <- dist(mydata, method = "euclidean") #matrix
fit <- hclust(d, method="ward.D2")
plot(fit)       # display dendogram
groups2 <- cutree(fit, k=3)
```

Clusterboost Example Code   * library(fpc)

```
pmatrix <- scale(iris[,-5])   #scale variables
cboot.hclust <- clusterboot(pmatrix,
clustermethod=hclustCBI,method="ward.D", k=3)
summary(cboot.hclust$result)
cboot.hclust$result$partition
table(cboot.hclust$result$partition,iris[,5])
cboot.hclust$bootmean  # clusterwise means
```

$$\text{Precision} = \frac{tp}{tp+fp}$$

$$\text{Recall} = \frac{tp}{tp+fn}$$

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

An ROC curve (receiver operating characteristic curve) is graph showing the performance of a classification model at all classification thresholds. This curve plots two parameters: TPR and FPR

$$\text{Accuracy} = \frac{tp+tn}{tp+tn+fp+fn}$$

Confusion Matrix

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)