Advanced R Data Analysis Training

Trainer: Dr. Ravi Tiwari

Website:www.tertiarycourses.com.sg
Email: enquiry@tertiaryinfotech.com

Course Material available at:

https://github.com/rkrtiwari/rAdvanced

Agenda

Module 1: R Data Analysis Packages
- Data Analysis Components
- Data Analysis Steps
- R Data Analysis Packages

Module 2: Obtaining Data
- Reading Data from CSV file
- Reading Data from JSON file
- Reading Data from XML file
- Reading Data from Web
- Reading Data from APIs

Agenda

Module 3: Data Preprocessing
- Mutating Data
- Merging Data
- Reshaping Data
- Missing Data

Module 4: Data Visualization
- Using ggplot

Module 5: Advanced R Functions
- lapply
- sapply
- split
- tapply

Agenda

Module 6: Regression
- Univariate and Multivariate Linear Model Regression
- Polynomial Model Regression
- Generalized Regression Models

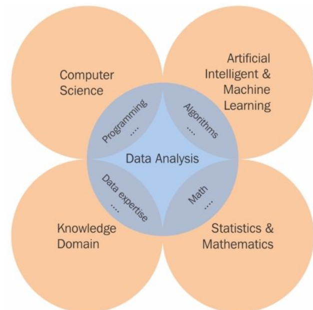
Module 7: Classification & Clustering
- Classification
- Clustering

Module 8: Time Series
- Creating Time Series
- Forecasting

Module 9: Shiny (Optional)

Module 1
Getting Started

Data Analysis Components



Data Analysis Steps

- Data Collection
- Data Processing
- Data Cleaning
- Data Visualization
- Modeling (eg Regression, Clustering...)
- Data Product

R Data Analysis Packages

Data Manipulation

dplyr: Data manipulation tasks

reshape2: Changing the data format

mice: Missing data Imputation

Data Analysis

glmnet: Regression

gam: Generalized Additive Model

rpart: Decision Tree

randomforest: Random Forest Analysis

R Data Analysis Packages

Data Visualization

ggplot2: Powerful visualization

shiny: Interactive data visualization

VIM: Missing data visualization

Install Packages

```
install.packages("dplyr")
install.packages("rpart")
install.packages("randomForest")
install.packages("mice")
install.packages("shiny")
install.packages("mice")
install.packages("ggplot2")
```

Module 2
Obtaining Data

Read Data from CSV File

```
data1 <- read.csv("data.csv", header = TRUE)
```

Read Data from json

```
data <- fromJSON(“data.json”)
```

Read Data from Web

```
url<-"http://archive.ics.uci.edu/ml/machine-lear
 ning-databases/wine/wine.data"
read.csv(url, nrows=5, header = FALSE)
```

Read Data from XML

```
library(XML)
data  <- xmlTreeParse(data.xml)
```

Challenge

Read the housing data from the following webpage
"https://archive.ics.uci.edu/ml/machine-learning-databases/housing/housing.data"
and store it in a dataframe named house

Time: 5 min

Module 3
Data Pre-Processing

Mutating Data

- Used to add a new column to a dataframe

```
mutate(mtcars, heavy = ifelse(wt > 3, "yes",
"no"))
```

Merging Data

- Joining two dataframes by one or more common key

INPUT

OBJECT ID#	Landuse Code
1	2
2	0
3	1

Join Fields

+

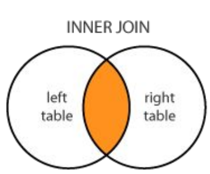
Landuse Code	Landuse Type
0	Unclassified
1	shrub
2	water

OUTPUT

OBJECT ID#	Landuse Code	Join Table Landuse Code	Join Table Landuse Type
1	2	2	water
2	0	0	Unclassified
3	1	1	shrub

Merging Data: Inner Join

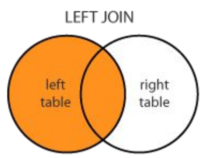
Returns all the rows where the join condition is met




```
merge(df1, df2, by = "CustomerId")
```

Merging Data: Left Join

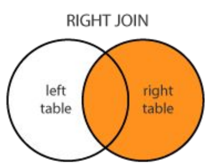
Returns all the rows from the left table, unmatched values gets NULL



merge(df1, df2, by = "CustomerId", all.x = TRUE)

Merging Data: Right Join

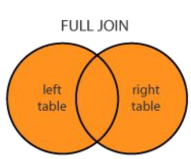
Returns all the rows from the right table, unmatched values gets NULL



merge(df1, df2, by = "CustomerId", all.y = TRUE)

Merging Data: Outer Join

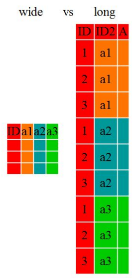
Returns all the rows, unmatched values gets NULL



merge(df1, df2, by = "CustomerId", all = TRUE)

Reshaping Data

- Data reshaping involves the rearrangement of the form of the data



Reshaping Data (Melting)

- For Reshaping, the data needs to be in the form where we have only id variables and its corresponding value (melted data)

```
aqm <- melt(airquality, id = c("Month", "Day"),
        measure.vars= c("Ozone", "Temp"))
```

Reshaping Data (Casting)

- Molten data can be casted into desired form

```
dcast(aqm, Month + Day ~ variable)
```

Reshaping Data (aggregation)

- When id variables do not identify unique observation, then an aggregation function is required

dcast(aqm, Month ~ variable, mean)

Reshaping Challenge

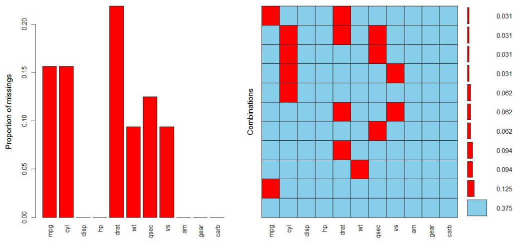
Find the mean value of mpg for each type of gears (3, 4, and 5) in mtcars dataset

Missing Data: Types

- The variable missingness is unrelated to the variable (Missing Completely At Random (MCAR))
- The variable missingness is related to the variable itself (Missing Not At Random (MNAR))
- The variable missingness is related to some other variable (Missing at Random)

Missing Data: Visualization

aggr(miss_mtcars, numbers=TRUE)



Missing Data: Treatment

- Complete Case Analysis

- Replace the missing data with non-missing data (imputation)
  - Mean Substitution
  - Regression
  - Stochastic Regression
  - Multiple Imputation

Complete case analysis

m1 <- lm(mpg ~ am + wt + qsec, data = miss_mtcars, **na.action = na.omit**)

Drawbacks:
- We lose statistical power as sample size is smaller
- With more variables, we are liable to lose more rows

Mean Substitution

```
mean_sub$qsec[is.na(mean_sub$qsec)] <-
 mean(mean_sub$qsec, na.rm = TRUE)
```

Drawback:

- It produces biased estimate of variance

Regression Imputation

Other column values are used to predict the value of the missing data in a given column

Drawback:

- It produces biased estimate of variance and covariance between different columns

Stochastic Regression Imputation

It adds a random (stochastic) value to the prediction of regression imputation

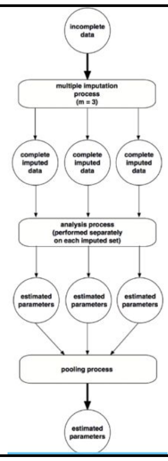
Drawback:

- Since, it produces only one imputed data set, it does not capture the full extent of uncertainty

Multiple Imputation

We generate multiple versions of the imputed data

```
imp <- mice(miss_mtcars, m=3)
imp$imp
```



Module 4
Data Visualization

Plotting Fundamental

Mapping of Data Properties to Visual Properties

Data Property: Numerical or Categorical

Visual Property: x and y position, color, shape, size, height of bars, etc.

ggplot Fundamental

Aes: Visual properties of geometry, such as x and y position, line color, point shapes, etc.

ggplot(mtcars) + aes(x=wt, y = mpg)

ggplot(mtcars) + aes(x=wt, y = mpg, col = factor(am))

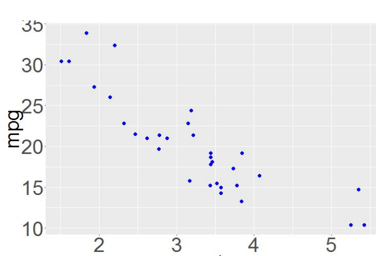
ggplot Fundamental

Geom: Geometric objects drawn to represent data. Such as bars, lines, and points

ggplot(mtcars) + aes(x=wt, y = mpg) + geom_point()

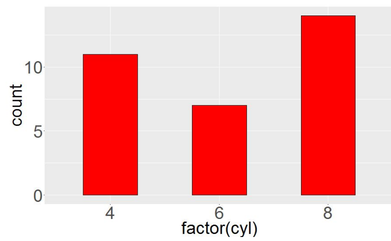
Geom_point

gplot(mtcars) + aes(x=wt, y=mpg) + geom_point(size=3, color = “blue”)



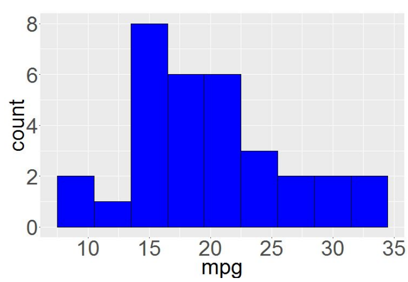
Geom_bar

ggplot(mtcars, aes(x = factor(cyl))) +
geom_bar(fill = ‘red’)



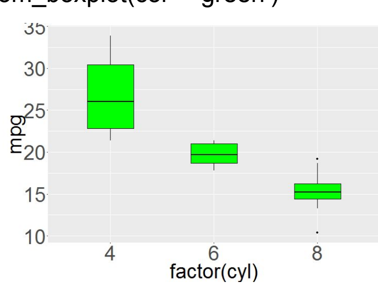
Geom_histogram

ggplot(mtcars, aes(x = mpg)) +
geom_histogram(binwidth = 3, fill = ‘blue’)



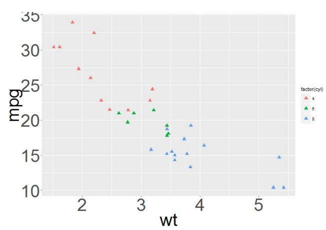
Geom_boxplot

ggplot(mtcars, aes(x = factor(cyl), y = mpg))
+ geom_boxplot(col = ‘green’)



aes color based on grouping

ggplot(mtcars) + aes(x=wt, y=mpg, color = factor(cyl)) + geom_point(size=3)



ggplot Fundamental

Scales: controls the mapping from the values in the data space to values in the aesthetics scale.

```
ggplot(mtcars) + aes(x=wt, y = mpg) +
  geom_point() +
scale_x_continuous(limits=c(1,6))
```

ggplot Fundamental

Guides: Aids the viewers in mapping visual properties back to data space. Such as tick marks, labels, legend

guides Example

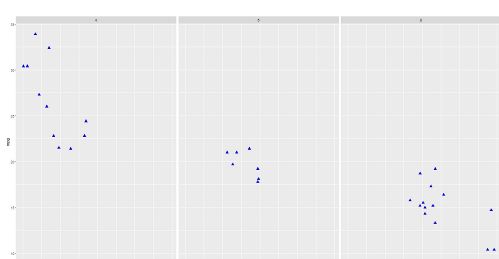
ggplot(mtcars) + aes(x=wt, y = mpg, col = factor(am), shape = factor(cyl)) + geom_point(size = 3) + guides(col = guide_legend('am'), shape=guide_legend('cyl'))

ggplot Fundamental

Facets: Plots subsets of data in separate panel

Facet Example

ggplot(mtcars) + aes(x=wt, y=mpg) + geom_point() + facet_wrap(~ cyl)



Challenge

1. Plot Day vs Ozone data for airquality dataset

2. Use different colors for different months

3. Use different panels for different months

Module 5
Advanced Functions

lapply

- Returns the result of applying the specified function to each element of the list

```
lapply(mtcars, mean)
```

sapply

- Same as lapply, but the results are returned as vector (s stands for simplify)

```
sapply(mtcars, mean)
```

Challenge

1. Find the mean of all the measurements in the iris dataset

tapply

- Returns the result of applying the specified function to specified groups in the data

```
tapply(mtcars$mpg, mtcars$cyl, mean)
```

aggregate

- Generalized form of tapply that can taken in multiple variables to be acted upon

aggregate(mtcars$mpg, by = list(mtcars$cyl), mean)

split

- Divides the data into specified groups

split(mtcars$mpg, mtcars$cyl)

Challenge

1. Find the mean of Petal.Length for each species in iris data set

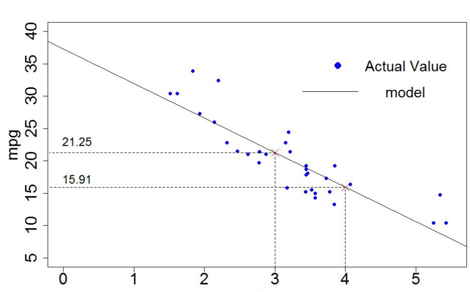
2. Find the mean of all four lengths for each species in the iris data set

Module 6
Regression

Linear Regression Definition

Involves finding a straight line that best describes the data

Linear Regression Example



Linear Regression - Univariate

```
m <- lm(mpg ~ wt, data = mtcars)
prediction <- predict(m, data.frame(wt = 3))
```

Linear Regression - Multivariate

```
m <- lm(mpg ~ wt + qsec, data = mtcars)
prediction <- predict(m,
data.frame(wt = 3, qsec = 20))
```

Challenge

Build a linear model to predict the median value of the home

Polynomial Regression

```
m <- lm(mpg ~ poly(wt,2), data = mtcars)
predict(m, data.frame(wt = 3))
```

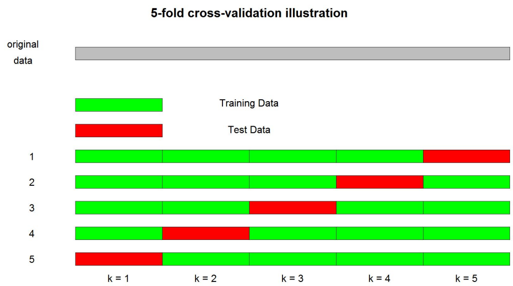
Generalized Additive Model

```
gam1 <- gam(mpg ~ s(wt,2) + disp, data =
mtcars)
predict(gam1, newdata = list(wt = 3, disp =
120))
```

Forward variable selection

```
m <- regsubsets(mpg ~ ., data = mtcars,
method = "forward")
ms <- summary(m)
which.max(ms$adjr2)
```

K-fold cross validation



Lasso Regularization

```
cv.out <- cv.glmnet(x,y, alpha=1, nfolds = 5)
bestlam  <- cv.out$lambda.min
lasso.pred <- predict(cv.out ,s=bestlam
,newx=x1)
```

Challenge

1. Build a linear model to predict the median value of the home
2. Use lasso method to create the best model that has 3 predictor variable

Time: 10 mins

Module 7
Classification &
Clustering

Steps in building a classifier

Step 1: Data partition into train and test data

Step 2: Model training on train data

Step 3: Model performance evaluation on test data

Decision Tree Classifier

1. m <- rpart(Species ~ ., data = train)
2. pred <- predict(m, test, type = "class")
3. table(pred, testSpecies)

Random Forest Classifier

1. m <- randomForest(Species ~ ., data=train)
2. pred <- predict(m, test)
3. table(pred, testSpecies)

Challenge

1. Build a random forest model to predict the median value of the home
2. What are the three most important variable

Clustering

Hierarchical Clustering

```
M <- dist(myIris)
hc <- hclust(M)
clusters <- cutree(hc, k = 3)
```

k-means Clustering

```
kmeans(myIris, centers = 3, nstart = 10)
```

Module 8
Time Series

Creating Time Series

```
apts <- ts(AirPassengers, frequency=12)
```

Decomposing Time Series

```
f <- decompose(apts)
plot(f)
```

Forecasting Time Series

```
fit <- arima(AirPassengers, order=c(0,1,1),
          list(order=c(0,1,1), period=12))

fore <- predict(fit, n.ahead=24)
```

Module 9
Shiny

Shiny illustration

runApp("shinyApp/")

Shiny Component:
ui.R (user interface)

ui <- fluidPage()

Shiny Component: server.R

server <- function(input, output) {}

Challenge

1. Create a shiny app that prints out the square of the input value

Time: 5 min

Alfred Ang
96983731
angch@tertiaryinfotech.com