8/4/19

DATA ANALYTICS

-> Analytics : The science of using data to build models that lead to better decisions that add value to individuals, to companies, to institutions.

→ R :→ A software environment for data analysis, statistical computing & graphics.
→ A programming language.

→ Why use R?

1. Nice graphics and visualizations
2. Free — open source project
3. Easy to re-run previous work

→ R console:

1. To get info about a particular function, we can type ?[fn_name] eg: ?sqrt.
   It results document.
2. To save our value to a variable, we can type var_name = sqrt(2) (or)
   var_name <— sqrt(2)
   result is stored in var_name.
3. Variable naming rules:— should not use spaces, shouldn't start with number.
4. ls() → to see list of variable names used in the R session.

→ R only allows 1 data type in 1 vector.

Eg:- country = c("India", "USA", "Brazil")

number = c("75", "60")

c → Combine

If we type country[1], we get India.

→ seq(0, 100, 2)

↓ ↓ ↓

starting num | last num | increment value.

→ All of the data in a single object →

data.frame.

Eg: CountryData = data.frame(Country, number ~~Lifeexpectancy~~)

↓

To combine two ~~for~~ vectors.

→ To add another field / vector use '$'

Eg: CountryData $ population = c(100, 200, 500)

→ To combine two data frames, we use "rbind" function.

Newfunction_name = rbind(dataframe1_name, dataframe2_name)

→ getwd() - To find the path to the folder in console.

→ WHO = read.csv("WHO.csv") - To read csv file

→ str(WHO) - To get the structure of file

→ summary(WHO) - To get the numerical summary of file

→ WHO-Europe = subset(WHO, Region == "Europe")

Dataset — Variable to be matched

→ write.csv(WHO-Europe, "WHO-Europe.csv")

dataset — file name

- To save the data into the csv file

→ rm(WHO-Europe) - To remove the data set

→ WHO $ Under15 - Will print the under15 data set from WHO file

→ mean( ) - To find mean

→ sd ( ) - To get the standard deviations

→ which.min ( ) - will returns the min values index

→ which.max ( ) - will returns the max values index or observations

→ plot(WHOGNI, WHOFertilityRate) - To plot graphs

x-axis — y-axis

→ Outliers = subset(WHO, GNI > 10000 & FertilityRate > 2.5)

- It will returns the set with those specifications

→ nrow(Outliers) - will give the number of countries in the subset.

→ Outliers[c("Country", "GNI", "FertilityRate")]

- it will give the table from the subset with those combinations.

→ plot (WHOGNI , WHOFertilityRate) - scatter plot

→ hist (WHO$CellularSubscribers) - histogram plot

→ boxplot (WHO$LifeExpectancy ~ WHO$Region) - box plot

↑
we can also label the graph using xlab = " " xlim = c(
ylab = " " breaks = c
main = " "

→ table (WHO$Region) - prints the number of observations of each regions

→ tapply (WHO$Over60 , WHO$Region , mean)

↑ splits with this ↑ calculates this with 1st parameter

→ tapply (WHO$LiteracyRate , WHO$Region , min , na.rm = TRUE)

↑ to overcome the non entry values i.e NA

==

→ USDA = read.csv("USDA.csv")

→ names (USDA) - will give the names of the variables present in the file

→ match ("CAVIAR" , USDA$Description) - will give that particular index from Descriptions

→ HighSodium = USDA$Sodium > mean(USDA$Sodium, na.rm = TRUE)

↑ gives the output as TRUE FALSE

HighSodium = as.numeric( )

↑ To get the o/p in numerical i.e, 1 - TRUE
0 - FALSE

9/4/18

* Linear Regression :

Ashenfelter - tested the quality of wine using a set of independent variables

↑ Age, weather (Harvest Rain, Winter Rain) temperature

⇒ One variable regression model

$$y_i = \beta_0 + \beta_1 x_i + e_i \quad \leftarrow \text{error}$$

y_i ↑ dependent variable; β_0 ↑ intercept co-efficient; β_1 ↑ regression co-eff; x_i ↑ Independent variable

√ (for linear regression model)

- Sum of Squared Errors SSE = $e_1^2 + e_2^2 + \cdots e_n^2$
- Root mean square error RMSE = $\sqrt{\frac{SSE}{n}}$
- Total sum of squares (for base line model) =

↑ line will be ⊥ to x-axis
point is found by taking the mean of all datasets

- $R^2 = 1 - \frac{SSE}{SST}$ $\quad 0 \le SSE \le SST$, $0 \le SST$

if $R^2 = 0$ means no improvement over baseline

$R^2 = 1$ means a perfect predictive model

⇒ Multiple linear Regression model with K variables

$$y_i = \beta_0 + \beta_1 x_1^i + \beta_2 x_2^i + \dots + \beta_k x_k^i + e^i$$

where y_i = dependent variable

x_j^i = j^{th} independent variable for i^{th} observation

e^i = error term

```
wine = read.csv("wine.csv")
-> model1 = lm (Price ~ AGST , data = wine)
                ↑
            linear model
```

-> model1 $ residuals — will give the error terms

-> SSE = sum (model1 $ residuals ^ 2)

-> model2 = lm (Price ~ AGST + HarvestRain , data = wine)

* Correlation :

A measure of the linear relationship between variables

if correlation = +1 → perfect positive linear relationship

0 → no linear relationship

-1 → perfect negative linear relationship

-> cor (wine $ WinterRain , wine $ Price)

* Predictive ability :

{ training data - data that we already used
{ test data - new data

wineTest = read.csv("wine_test.csv")

~~→ predictTest =~~

→ model4 = lm(Price ~ AGST + HarvestRain + WinterRain + Age, data = wine)

→ predictTest = predict(model4, newdata = wineTest)

- which predicts the model with new data

→ SSE = sum((wineTest$Price - predictTest)^2)
actual value - prediction value

SST = sum((wineTest$Price - mean(wine$Price))^2)
actual value - mean of training set

$R^2 = 1 - SSE/SST$

12/4/19

* Logistic Regression :

Predicts the probability of happening

Eg: Poor care = 1

Good care = 0

$$P(y=0) = 1 - P(y=1)$$

- Independent variables $x_1, x_2 \dots x_k$
- Logistic Response Function

$$P(y=1) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_k x_k)}}$$

- $\text{Odds} = \frac{P(y=1)}{P(y=0)}$

$$\text{Odds} = e^{\beta_0 + \beta_1 x_1 + \dots + \beta_k x_k}$$

$$\underbrace{\log(\text{Odds})}_{\text{logit}} = \beta_0 + \beta_1 x_1 + \dots + \beta_k x_k$$

quality = read.csv(" ")

→ Install Package - install.packages("caTools")

library(caTools) - to load the package in current R session.

- set.seed(88) - to initialize the random no. generator
- split = sample.split(quality$PoorCare, SplitRatio = 0.75)

↑ outcome variable ↑ percentage of data to go in training set

→ split will give TRUE's & FALSE's for GoodCare & PoorCare

- qualityTrain = subset(quality, split == TRUE)
- qualityTest = subset(quality, split == FALSE)

- QualityLog = glm(PoorCare ~ OfficeVisits + Narcotics, data = qualityTrain, family = binomial)

↑ general Logistic Regression model

- predictTrain = predict(QualityLog, type = "response", newdata = qualityTest)
  - will give the prediction probabilities

- tapply(predictTrain, qualityTrain$PoorCare, mean)
  - mean of prediction of true outcomes

→ Threshold value - to get the binary prediction
  • inplace of probabilities.

→ Confusion Matrix / Classification Matrix

	Predicted=0	Predicted=1
Actual=0	True negatives (TN)	False positives (FP)
Actual=1	False negatives (FN)	True positives (TP)

0 - Goodcare 1 - Poorcare
TN - Actual goodcare predicted goodcare
TP - Actual poorcare predicted poorcare

Sensitivity = $\frac{TP}{TP + FN}$ True Positive Rate

Specificity = $\frac{TN}{TN + FP}$ True Negative Rate

- A model with high threshold ↑ → Sensitivity ↓
  Specificity ↑

table (quality Train $ PoorCare , predict Train > 0.5)

↑ Row = true outcome

↑ Column = Predictions

Overall Accuracy = $(TN + TP) / N$

Overall Error Rate = $(FP + FN) / N$

False Negative Error Rate = $FN / (FN + TP)$

False Positive Error Rate = $FP / (FP + TN)$

Base line model accuracy = $(TN + FP) / N$

- install . packages ("ROCR")
- ROCRPredict = prediction (predictTest , test $ TenYearCHD)
  ↓ predicted value ↓ Actual value
- as . numeric (performance (ROCRpred , "auc") @ y . values)
  - will give the auc value