```r
# Title: R basics
# Objectives: variables, data types and others

# 1. Variables: numbers, strings, vectors, matrices
# Each variable has a name. A name can only contain numbers
# letters, '.' and '_'.
# 1) numbers
x <- 24 # Left arrow is to assign values to a variable
print(x)
x <- x * 2 # Arithmetic operation on numbers and
# change the value of the variable
print(x)

# 2) strings
myStr <- 'Less is more.'
print(myStr)

# 3) vector
myVector <- c(1,2,4,5) # c() creates a sequence of objects
myStrVector <- c('temp', 'humidity', 'cloud', 'wind')

print(myVector)
print(myStrVector)

myStrVector[1]
names(myVector) <- myStrVector # name the values in the vector
myVector[4] # Refer by index
myVector['cloud'] # Refer by name

# Other ways to create sequences
myRandom <- rnorm(50) # Generate a sequence of numbers from standard normal distribution
print(myRandom)
mySeq <- seq(1, 100) # Generate a sequence from 1-100, inclusive on both sides
print(mySeq)

# 4) Matrix
# Creates a 3X3 matrix
myM1 <- matrix( data = rnorm(9),nrow = 3, ncol = 3 )
print(myM1)

# 2. Functions
# A function is a predefined program that
# performs certain tasks.
# Calling a function use the function name
# + a pair of parentheses.
# Arguments are values passed to the function
# When there are multiple arguments, refer to
# them use names
myM2 <- matrix(c(1:16), 4, 4)
print(myM2)
# You can also use position to pass the arguments.
```

#2……

```r
# Install a package 'tidyverse'
install.packages('tidyverse')
# Import library
library(tidyverse)

?mpg # ? can be used to find the documentation
View(mpg)
# Find the datatype of variables using class()
class(mpg$drv)
class(mpg$hwy)
class(mpg$displ)
class(mpg$model)
# Let's examine variable cyl
class(mpg$cyl)
unique(mpg$cyl)
mpg.copy <- mpg
# Change the data type of cyl to factor
mpg.copy$cyl <- as.factor(mpg.copy$cyl)
class(mpg.copy$cyl)

# Explore a categorical variable
# bar plot
# Introduce ggplot2, which is a very popular
# plot package
install.packages('ggplot2')
library('ggplot2')
ggplot(data = mpg) + # Specify dataset
  geom_bar(mapping = aes(x = drv))

names(mpg)
lmcar.fit <- lm(hwy ~ drv + class + displ, data = mpg)
summary(lmcar.fit)
```

#3……

```r
# Topic: Logistic regression analysis for Default data
# 1. Import the dataset
default.df <- read.csv('Default.csv', stringsAsFactors = TRUE)
class(default.df$default)
class(default.df$student)

# 2. Explore with graphs
library(ggplot2)
names(default.df)
# Default vs. balance & income
ggplot(data = default.df) +
  geom_point(aes(x = balance, y = income, color = default, shape = default))
# It seems that higher balance is related to more default

# Make boxplots across different categories
ggplot(data = default.df) +
```

```r
  geom_boxplot(aes(x = default, y = balance, fill = default), width = 0.1)

# The relationship between default and student
table(default.df$default, default.df$student)

# 3. Use logistic regression to predict whether a person default
# glm() function will do

lg.fit <- glm(default ~ balance, data = default.df, family = binomial)
summary(lg.fit)

lg.fit.1 <- glm(default ~ student, data = default.df, family = binomial)
summary(lg.fit.1)
# contrasts function will tell you which case is coded as base case
contrasts(default.df$student)

# Fit the full model with all three variables
lg.fit.full <- glm(default ~ ., data = default.df, family = binomial)
summary(lg.fit.full)

# Use the full model to make prediction
# To predict probability, type = 'response'
lg.prob <- predict(lg.fit.full, data = default.df, type = 'response')
lg.prob[1:10]

# classification based on cutoff probability
lg.class <- ifelse(lg.prob > 0.5, 'Yes', 'No')
lg.class[1:10]

# Cross-tablate predicted class vs. true class
table(lg.class, default.df$default)
```

#4……

```r
# Title: Linear regression example
# Author:
# Date:

# The purpose of the project is to predict sales based
# on advertising spendings using linear regression method

# 1. Import the dataset
# Using full directory
# Advertising <- read.csv('C:\\Users\\gelin\\OneDrive - UNT System\\DSCI
5240\\Data\\Advertising.csv')
# The easy way is to save the data and the R file in
# the same directory
Advertising <- read.csv('Advertising.csv')

# 2. Explore the data
# Find the variable names
View(Advertising)
# We need to drop the X variable
# We can use the package dplyr
# First, install the package. You only need to do
```

```r
# it once
install.packages('dplyr')
# Then we need to import the package. You need
# to do it every R session
library(dplyr)
Advertising <- Advertising %>% select(-1)
# We can also use slicing a matrix
# Advertising <- Advertising[,-1]

# A. Explore an interval variable
mean(Advertising$sales)
median(Advertising$sales)
var(Advertising$sales)
min(Advertising$sales)
max(Advertising$sales)
# Using graphs - A histogram to explore the
# distribution of an interval variable
hist(Advertising$sales)
# And a boxplot
boxplot(Advertising$sales)
# B. Explore the relationship between TV and sales
cor(Advertising$sales, Advertising$TV)
# Check variable names
names(Advertising)
# To explore with a graph: scatterplot
plot(Advertising$TV, Advertising$sales)

# 3. Fit a simple linear model
# lm() function
lm.fit <- lm(sales ~ TV, data = Advertising)
summary(lm.fit)

# Draw the fitted line of the linear model
plot(Advertising$TV, Advertising$sales)
abline(lm.fit)

# With multiple variables
lm.fit.M <- lm(sales ~ TV + radio, data = Advertising)
summary(lm.fit.M)

# 4. To make predictions
predict(lm.fit.M, data = Advertising,
        interval= 'confidence')
# 20.555465 20.162781 20.948148
predict(lm.fit.M, data = Advertising,
        interval= 'prediction')
# 20.555465 17.2165165 23.894413
```

5.....

```r
happy <- read.csv('Happiness2019.csv')
View(happy)
names(happy)
```

```r
# lm.fit <- lm(Score ~ GDP.per.capita +)
# We can create a dataset that contains
# only the dependent and independent variables
# we want to use.
happy.fit <- happy[,c(3,4,5,6,7,8,9)]
lm.fit <- lm(Score ~ ., data = happy.fit)
summary(lm.fit)

ggplot(happy) +
  geom_bar(mapping = aes(y = Country.or.region, x = Score, fill = Overall.rank),
           stat = 'identity')
```

6…….

```r
# Topic: Logistic regression analysis for Default data
# Author:
# Date:

# Data Description
# We are interested in predicting whether an individual will default
# on his or her credit card payment, on the basis of
# annual income and monthly credit card balance.

# 1. Import the dataset
default.df <- read.csv('Default.csv', stringsAsFactors = TRUE)
class(default.df$default)
class(default.df$student)

# 2. Explore with graphs
library(ggplot2)
names(default.df)
# Default vs. balance & income
ggplot(data = default.df) +
  geom_point(aes(x = balance, y = income, color = default, shape = default))
# It seems that higher balance is related to more default

# Make boxplots across different categories
ggplot(data = default.df) +
  geom_boxplot(aes(x = default, y = balance, fill = default), width = 0.1)

# The relationship between default and student
table(default.df$default, default.df$student)

# 3. Use logistic regression to predict whether a person default
# glm() function will do

lg.fit <- glm(default ~ balance, data = default.df, family = binomial)
summary(lg.fit)

lg.fit.1 <- glm(default ~ student, data = default.df, family = binomial)
summary(lg.fit.1)
# contrasts function will tell you which case is coded as base case
contrasts(default.df$student)

# Fit the full model with all three variables
lg.fit.full <- glm(default ~ ., data = default.df, family = binomial)
```

```r
summary(lg.fit.full)

# Use the full model to make prediction
# To predict probability, type = 'response'
lg.prob <- predict(lg.fit.full, data = default.df, type = 'response')
lg.prob[1:10]
# About the type argument
# The default predictions are of log-odds (probabilities on logit scale)
# and type = "response" gives the predicted probabilities.
# The "terms" option returns a matrix giving the fitted values of each term
# in the model formula on the linear predictor scale.

# classification based on cutoff probability
lg.class <- ifelse(lg.prob > 0.5, 'Yes', 'No')
lg.class[1:10]

# Cross-tabulate predicted class vs. true class
# Confusion matrix
lg.cm <- table(lg.class, default.df$default)

# 4. Using LDA to classify
library(MASS)
# Fit an LDA model
lda.fit <- lda(default ~ ., data = default.df)
# Show the estimates
lda.fit
# Make predictions
lda.pred <- predict(lda.fit, data = default.df)
lda.pred$class[1:10] # The classification of Y
lda.pred$posterior[1:10,] # Posterior probability of Y in class Yes/No.
# The confusion matrix for the LDA model
lda.cm <- table(lda.pred$class,default.df$default)

# 5. Fit a QDA model

qda.fit <- qda(default ~ ., data = default.df)
# Show the estimate
qda.fit
# Make predictions
qda.pred <- predict(qda.fit, data = default.df)
qda.pred$class[1:10] # The classification of Y
qda.pred$posterior[1:10,] # Posterior probability of Y in class Yes/No.
# The confusion matrix for the LDA model
qda.cm <- table(qda.pred$class,default.df$default)

# 6. Naive Bayes
# The library is e1071.
library(e1071)
# Fit the model
nb.fit <- naiveBayes(default ~ ., data = default.df)
# Show the results
nb.fit
# predict() requires argument of 'newdata'
nb.class <- predict(nb.fit, newdata = default.df)
```

```r
nb.class[1:10]
# predict() with type = 'raw'
nb.pred <- predict(nb.fit, newdata = default.df, type = 'raw')
nb.pred[1:10,]
# The confusion matrix
nb.cm <- table(Predicted = nb.class, Actual = default.df$default)

# Comparison of the confusion matrix of different methods
lg.cm
lda.cm
qda.cm
nb.cm

library(caret)
```

7…….

```r
# Title:

# 1. Import the dataset
Covid <- read.csv('COVIDDeath.csv')
names(Covid)
# 2. A scatterplot to explore the relationship
# "Pneumonia.Deaths" vs. COVID.19.Deaths
plot(Covid$Pneumonia.Deaths, Covid$COVID.19.Deaths)
library(ggplot2)
ggplot(data = Covid) +
  geom_point(mapping = aes(x = Pneumonia.Deaths,
                           y = COVID.19.Deaths))
```