In this, we will learn to perform the following operations on a raw dataset:

* Dealing with missing data
* Dealing with categorical data
* Splitting the dataset into training and testing sets
* Scaling the features

**Dealing with Missing Values**

dataset$age = ifelse(is.na(dataset$age),ave(dataset$age, FUN = function(x) mean(x, na.rm = 'TRUE')),dataset$age)

dataset$salary = ifelse(is.na(dataset$salary), ave(dataset$salary, FUN = function(x) mean(x, na.rm = 'TRUE')), dataset$salary)

dataset$age = as.numeric(format(round(dataset$age, 0))) 🡪 Since we are not interested in having decimal places for age we will round it up using the above code. The argument 0 in the round function means no decimal places.

#### Dealing with Categorical Data

dataset$nation = factor(dataset$nation, levels = c('India','Germany','Russia'), labels = c(1,2,3))

dataset$purchased\_item = factor(dataset$purchased\_item, levels = c('No','Yes'),  labels = c(0,1))

#### Splitting the Dataset into Training and Testing Sets

#### install.packages('caTools') #install once library(caTools) # importing caTools library set.seed(123) split = sample.split(dataset$purchased\_item, SplitRatio = 0.8) training\_set = subset(dataset, split == TRUE) test\_set = subset(dataset, split == FALSE)

**Scaling the Features**

training\_set[,3:4] = scale(training\_set[,3:4])  
test\_set[,3:4] = scale(test\_set[,3:4])

#### Outliers removed using Script

#### outlierKD <- function(dt, var) {

#### var\_name <- eval(substitute(var),eval(dt))

#### tot <- sum(!is.na(var\_name))

#### na1 <- sum(is.na(var\_name))

#### m1 <- mean(var\_name, na.rm = T)

#### par(mfrow=c(2, 2), oma=c(0,0,3,0))

#### boxplot(var\_name, main="With outliers")

#### hist(var\_name, main="With outliers", xlab=NA, ylab=NA)

#### outlier <- boxplot.stats(var\_name)$out

#### mo <- mean(outlier)

#### var\_name <- ifelse(var\_name %in% outlier, NA, var\_name)

#### boxplot(var\_name, main="Without outliers")

#### hist(var\_name, main="Without outliers", xlab=NA, ylab=NA)

#### title("Outlier Check", outer=TRUE)

#### na2 <- sum(is.na(var\_name))

#### message("Outliers identified: ", na2 - na1, " from ", tot, " observations")

#### message("Proportion (%) of outliers: ", (na2 - na1) / tot\*100)

#### message("Mean of the outliers: ", mo)

#### m2 <- mean(var\_name, na.rm = T)

#### message("Mean without removing outliers: ", m1)

#### message("Mean if we remove outliers: ", m2)

#### response <- readline(prompt="Do you want to remove outliers and to replace with NA? [yes/no]: ")

#### if(response == "y" | response == "yes"){

#### dt[as.character(substitute(var))] <- invisible(var\_name)

#### assign(as.character(as.list(match.call())$dt), dt, envir = .GlobalEnv)

#### message("Outliers successfully removed", "\n")

#### return(invisible(dt))

#### } else{

#### message("Nothing changed", "\n")

#### return(invisible(var\_name))

#### }

#### }

#### source("https://goo.gl/4mthoF")

#### outlierKD(dat, variable)

#### How to Split the data:

#### Method 1:

#### # create training and test data

#### wbcd1\_train <- wbcd1\_n[1:469, ]

#### View(wbcd1\_train)

#### wbcd1\_test <- wbcd1\_n[470:569, ]

#### View(wbcd1\_test)

#### # create labels for training and test data

#### wbcd\_train\_labels <- wbcd1[1:469, 1]

#### View(wbcd\_train\_labels)

#### wbcd\_test\_labels <- wbcd1[470:569, 1]

#### View(wbcd\_test\_labels)

#### Method 2:

#### set.seed(3)

#### train<-order(runif(290))

#### test<- -train

#### View(train)

#### View(test)

#### # Divide the data into Training and Testing Data

#### training<-HouseVotes84[train,]

#### View(training)

#### testing<-HouseVotes84[test,]

#### View(testing)

#### Method 3:

#### # splitting the data based on species

#### iris\_setosa<-iris[iris$Species=="setosa",] # 50

#### iris\_versicolor <- iris[iris$Species=="versicolor",] # 50

#### iris\_virginica <- iris[iris$Species=="virginica",] # 50

#### iris\_train <- rbind(iris\_setosa[1:25,],iris\_versicolor[1:25,],iris\_virginica[1:25,])

#### iris\_test <- rbind(iris\_setosa[26:50,],iris\_versicolor[26:50,],iris\_virginica[26:50,])

#### View(iris\_train)

#### View(iris\_test)

#### Method 4:

#### library(caTools) iris$spl=sample.split(iris,SplitRatio=0.7)

#### train=subset(iris, iris$spl==TRUE) # where spl== TRUE means to add only those rows that have value true for spl in the training dataframe

#### View(train) # you will see that this dataframe has all values where iris$spl==TRUE

#### test=subset(iris, iris$spl==FALSE) # where spl== FALSE means to add only those rows that have value true for spl in the training dataframe

#### View(test) # you will see that this dataframe has all values where iris$spl==FALSE

#### Method 5:

Another easy method to split the dataset into training and test set is as follows;  
  
train= iris [1:100,] # this will put the first 100 rows into the training set

test= iris [101:150]  # this will put the remaining 50 rows into the test set

**Method 6:**

dt = sort(sample(nrow(data), nrow(data)\*.7))

train<-data[dt,]  
test<-data[-dt,]

**Method 7:**

library(caret)  
set.seed(3456)  
trainIndex <- createDataPartition(data$FD, p = .7,list = FALSE,times = 1)  
Train <- data[ trainIndex,]  
Valid <- data[-trainIndex,]