UNIT 15 DATA ANALYSIS AND R

Structure **Page Nos.**

15.1 INTRODUCTION

This unit deals with the concept of data analysis and how to leverage it by using R programming. The unit discusses various tests and techniques to operate on data in R and how to draw insights from it. The unit covers the Chi-Square Test, its significance and the application in R with the help of an example. The unit also familiarises with the concept of Regression Analysis and its types including- Simple Linear and Multiple Linear Regression and afterwards, Logistic Regression. It is further substantiated with examples in R that explain the steps, functions and syntax to use correctly. It also explains how to interpret the output and visualise the data. Subsequently, the unit explains the concept of Time Series Analysis and how to run it on R. It also discusses about the Stationary Time Series, extraction of trend, seasonality, and error and how to create lags of a time series in R.

15.2 OBJECTIVES

After going through this Unit, you will be able to:-

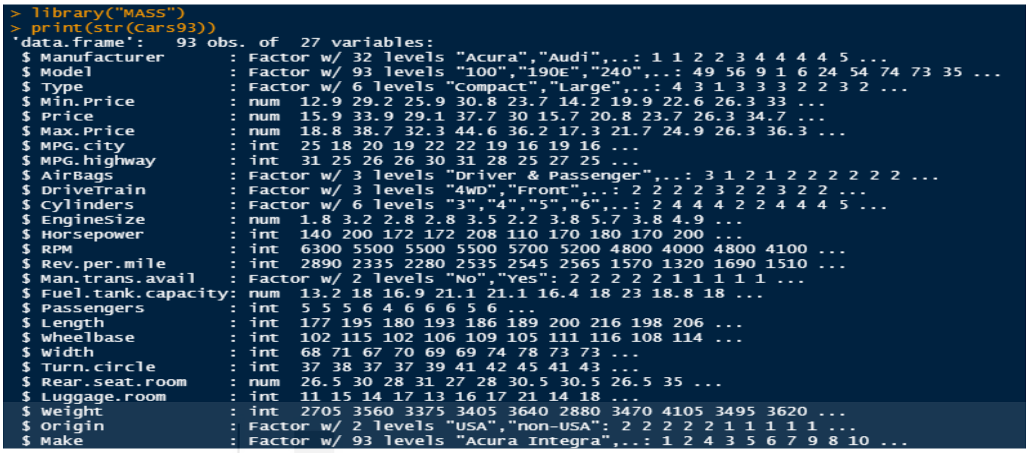
- Run tests and techniques on data and interpret the results using R;
- explain the correlation between two variables in a dataset by running Chi-Square Test in R;
- explain the concept of Regression Analysis and distinguish between their types- simple Linear and Multiple Linear;
- build relationship models in R to plot and interpret the data and further use it to predict the unknown variable values;
- explain the concept of Logistic Regression and its application on R;
- explain about the Time Series Analysis and the special case of Stationary Time Series;
- explain about extraction of trend, seasonality, and error and how to create lags of a time series in R.

15.3 CHI-SQUARE TEST

The Chi-Square test is a statistical tool for determining if two categorical variables are significantly correlated. Both variables should come from the same population and be categorical in nature, such as – top/bottom, True/False, Black/White.Syntax of a chi-square test: chisq.test(data)

EXAMPLE:

Let's consider R's built in "MASS" library that contains Cars93 dataset that represents the sales of different models of car.

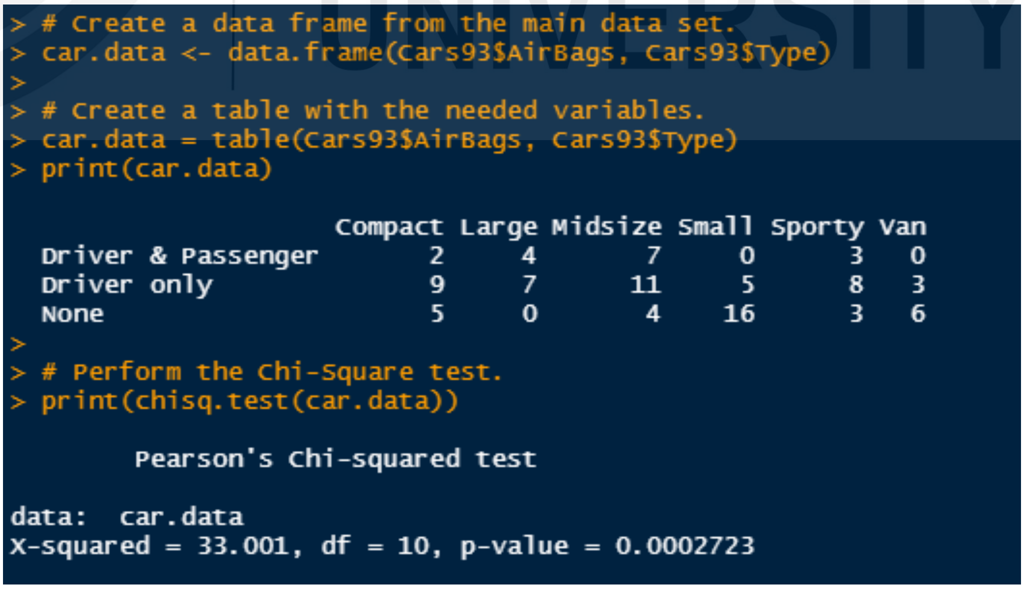


```
> library("MASS")
> print(str(Cars93))
'data.frame':   93 obs. of  27 variables:
 $ Manufacturer      : Factor w/ 32 levels "Acura","Audi",..: 1 1 2 2 3 4 4 4 4 5 ...
 $ Model             : Factor w/ 93 levels "100","190E","240",..: 49 56 9 1 6 24 54 74 73 35 ...
 $ Type              : Factor w/ 6 levels "Compact","Large",..: 4 3 1 3 3 3 2 2 3 2 ...
 $ Min.Price         : num  12.9 29.2 25.9 30.8 23.7 14.2 19.9 22.6 26.3 33 ...
 $ Price             : num  15.9 33.9 29.1 37.7 30 15.7 20.8 23.7 26.3 34.7 ...
 $ Max.Price         : num  18.8 38.7 32.3 44.6 36.2 17.3 21.7 24.9 26.3 36.3 ...
 $ MPG.city          : int  25 18 20 19 22 22 19 16 19 16 ...
 $ MPG.highway       : int  31 25 26 26 30 31 28 25 27 25 ...
 $ AirBags           : Factor w/ 3 levels "Driver & Passenger",..: 3 1 2 1 2 2 2 2 2 2 ...
 $ DriveTrain        : Factor w/ 3 levels "4WD","Front",..: 2 2 2 2 3 2 2 3 2 2 ...
 $ Cylinders         : Factor w/ 6 levels "3","4","5","6",..: 2 4 4 4 2 2 4 4 4 5 ...
 $ EngineSize        : num  1.8 3.2 2.8 2.8 3.5 2.2 3.8 5.7 3.8 4.9 ...
 $ Horsepower        : int  140 200 172 172 208 110 170 180 170 200 ...
 $ RPM               : int  6300 5500 5500 5500 5700 5200 4800 4000 4800 4100 ...
 $ Rev.per.mile      : int  2890 2335 2280 2535 2545 2565 1570 1320 1690 1510 ...
 $ Man.trans.avail   : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 1 1 1 1 1 ...
 $ Fuel.tank.capacity: num  13.2 18 16.9 21.1 21.1 16.4 18 23 18.8 18 ...
 $ Passengers        : int  5 5 5 6 4 6 6 6 5 6 ...
 $ Length            : int  177 195 180 193 186 189 200 216 198 206 ...
 $ Wheelbase         : int  102 115 102 106 109 105 111 116 108 114 ...
 $ Width             : int  68 71 67 70 69 69 74 78 73 73 ...
 $ Turn.circle       : int  37 38 37 37 39 41 42 45 41 43 ...
 $ Rear.seat.room    : num  26.5 30 28 31 27 28 30.5 30.5 26.5 35 ...
 $ Luggage.room      : int  11 15 14 17 13 16 17 21 14 18 ...
 $ Weight            : int  2705 3560 3375 3405 3640 2880 3470 4105 3495 3620 ...
 $ Origin            : Factor w/ 2 levels "USA","non-USA": 2 2 2 2 2 1 1 1 1 1 ...
 $ Make              : Factor w/ 93 levels "Acura Integra",..: 1 2 4 3 5 6 7 9 8 10 ...
```

Figure 15.1: Description of sample data set

As you can see, we have various variables that can be considered as categorical variable. Let's consider "Airbags" and "Type" for our model. You want to check, if there is a correlation in these two categorical variables. Chi-square test is a good indicator for such information. To perform the chi-square test, you may perform the following steps:

- First, you need to extract this data from the dataset (see Figure 15.2).
- Next, create the table of the data(See Figure 15.2) and
- Perform chi square test on the table (See Figure 15.2)



```
> # Create a data frame from the main data set.
> car.data <- data.frame(Cars93$AirBags, Cars93$Type)
>
> # Create a table with the needed variables.
> car.data = table(Cars93$AirBags, Cars93$Type)
> print(car.data)

                     Compact Large Midsize Small Sporty Van
  Driver & Passenger       2     4       7     0      3   0
  Driver only              9     7      11     5      8   3
  None                     5     0       4    16      3   6
>
> # Perform the Chi-Square test.
> print(chisq.test(car.data))

        Pearson's Chi-squared test

data:  car.data
X-squared = 33.001, df = 10, p-value = 0.0002723
```

Figure 15.2: Chi-square testing

The result shows the p value 0.0002723 which is less than 0.05 which indicates strong correlation. In addition, the value of chi square is also high. Thus, the variable type of car is strongly related to number of air bags.

Chi-square test is one of the most useful test in finding relationships between categorical variables.

How can you find the relationships between two scale or numeric variables using R? One such technique, which helps in establishing a model-based relationship is regression, which is discussed next.

15.4 LINEAR REGRESSION

Regression analysis is a common statistical technique for establishing a relationship model between two variables. One of these variables is known as a predictor variable, and its value is derived via experimentation. The response variable, whose value is generated from the predictor variable, is the other variable.

A regression model that employs a straight line to explain the relationship between variables is known as linear regression. In Linear Regression these two variables are related through an equation, where exponent (power) of both these variables is one. It searches for the value of the regression coefficient(s) that minimises the total error of the model to find the line of best fit through your data.

The general equation for a linear regression is –

$$y = a + b \times x$$

In the equation given above:

- y is called response/dependent variable, whereas x is a independent/predictor variable.
- The a and b values are the coefficients used in the equation, which are to be predicted.

The objective of the regression model is to determine the values of these two constants.

There are two main types of linear regression:

- *Simple Linear Regression*: This kind of regression uses only one independent variable, as shown in the equation above.
- *Multiple Linear Regression*: However, if you add more independent variables like: $y = a + b \times x_1 + c \times x_{2+\ldots}$, then it is called multiple regression.

Steps for Establishing a Linear Regression:

A basic example of regression is guessing a person's weight based on his/her height. To do so, you need to know the correlation between a person's height and weight.

The steps to establishing a relationship are as follows:

1. Carry out an experiment in which you collect a sample of observed height and weight values.
2. Create a relationship model using the **lm()** functions in R.
3. Find the coefficients from the model you constructed and use them to create a mathematical equation.

4. To find out the average error in prediction, get a summary of the relationship model. Also known as residuals, as shown in Figure 15.3
5. The **predict()** function in R can be used to predict the weight of new person. A sample regression line and residual are shown in Figure 15.3

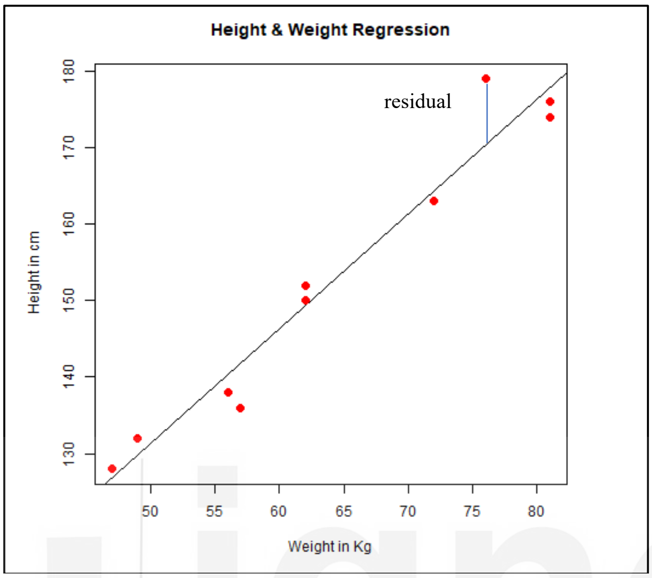
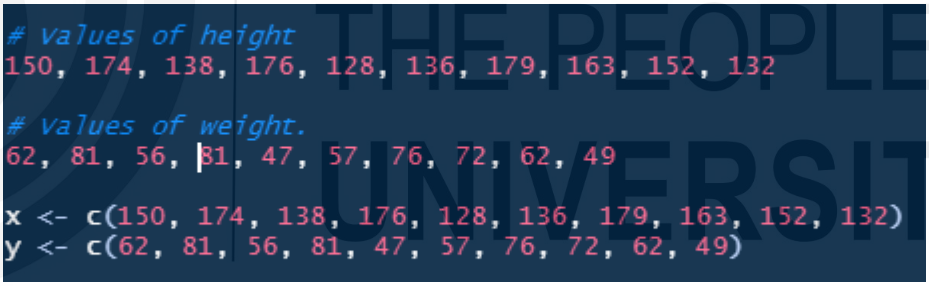


Figure 15.3: An example of regression mode and residual

Input Data

Below is the sample data with the observations between weight and height, which is experimentally collected and is input in the Figure 15.4



```
# Values of height
150, 174, 138, 176, 128, 136, 179, 163, 152, 132

# Values of weight.
62, 81, 56, 81, 47, 57, 76, 72, 62, 49

x <- c(150, 174, 138, 176, 128, 136, 179, 163, 152, 132)
y <- c(62, 81, 56, 81, 47, 57, 76, 72, 62, 49)
```

Figure 15.4: Sample data for linear regression

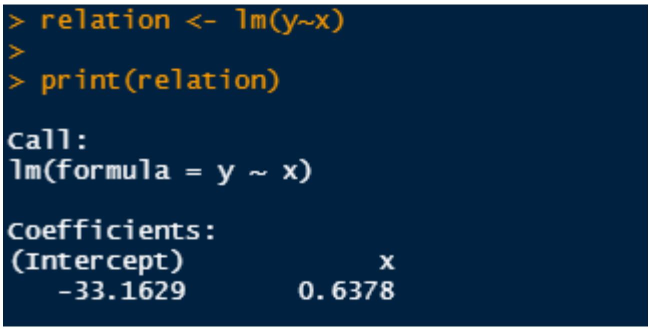
lm() function create the relation model between the variable i.e. predictor and response.

Syntax: lm(formula, data),where

formula: presenting the relation between x and y.

data: data on which the formula needs to be applied.

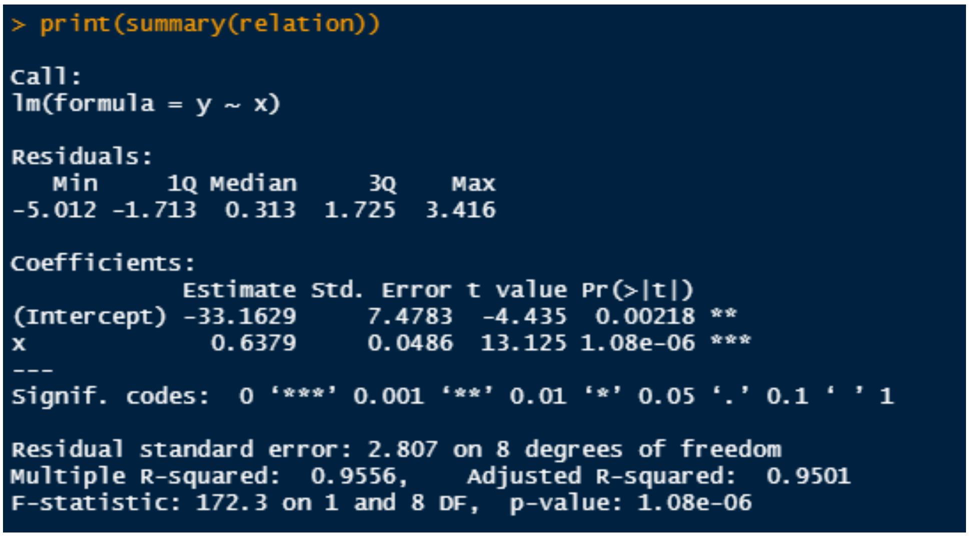
Figure 15.5 shows the use of this function.



```
> relation <- lm(y~x)
>
> print(relation)

Call:
lm(formula = y ~ x)

Coefficients:
(Intercept)            x
   -33.1629       0.6378
```

Figure 15.5: Use of lm function in linear regression

Summary of the relationship:



```
> print(summary(relation))

Call:
lm(formula = y ~ x)

Residuals:
   Min     1Q Median     3Q    Max 
-5.012 -1.713  0.313  1.725  3.416 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) -33.1629     7.4783  -4.435  0.00218 ** 
x             0.6379     0.0486  13.125 1.08e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.807 on 8 degrees of freedom
Multiple R-squared:  0.9556,	Adjusted R-squared:  0.9501 
F-statistic: 172.3 on 1 and 8 DF,  p-value: 1.08e-06
```

Figure 15.6: Results of regression

The results of regression as presented by R includes the following:

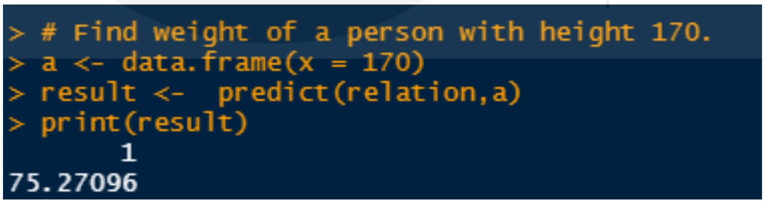
1. Five-point summary (Minimum, First Quartile, Median, Third Quartile, and Maximum). This shows the spread of the residual. You may observe that about 50% of residuals are in the range -1.713 to +1.725, which shows a good model fit.
2. The t value and Pr values for the intercept (that is b in the equation y = ax +b) and x (that is a in the equation y = ax +b).
3. The F-statistics is very high with a low p-value, indicating statistical difference between group means.

Predict function:

Function which will be used to predict the weight of the new person.

Syntax: Predict(object, newdata),

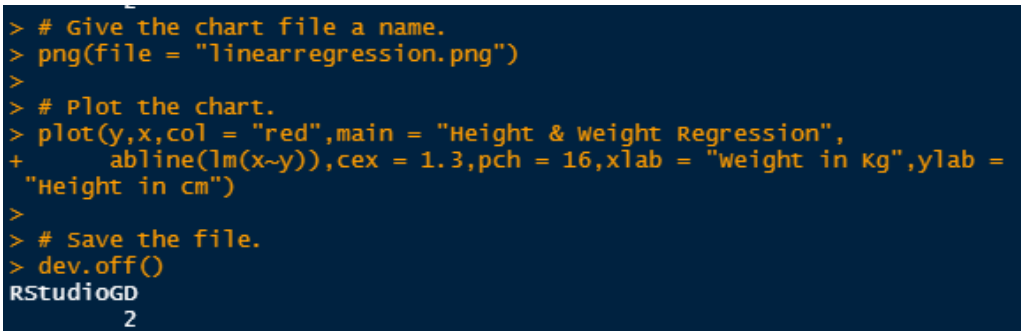
object is the formula already formulated using lm() function.
newdata is the vector containing new value for predictor variable.



```
> # Find weight of a person with height 170.
> a <- data.frame(x = 170)
> result <-  predict(relation,a)
> print(result)
       1 
75.27096 
```

Figure 15.7: The Predict function

Plot for Visualization: Finally, you may plot these values by setting the plot title and axis titles (see Figure 15.8). The linear regression line is shown in Figure 15.3.



```
> # Give the chart file a name.
> png(file = "linearregression.png")
>
> # Plot the chart.
> plot(y,x,col = "red",main = "Height & Weight Regression",
+      abline(lm(x~y)),cex = 1.3,pch = 16,xlab = "Weight in Kg",ylab =
 "Height in cm")
>
> # Save the file.
> dev.off()
RStudioGD
        2
```

Figure 15.8: Making a chart of linear regression

Linear regression has one response variable and one predictor variables, however, in many practical cases there can be more than one predictor variables. This is the case of multiple regression and is discussed next.

15.5 MULTIPLE REGRESSION

The relationship between two or more independent variables and a single dependent variable is estimated using multiple linear regression. When you need to know the following, you can utilize multiple linear regression.

- The degree to which two or more independent variables and one dependent variable are related (e.g. how baking soda, baking temperature, and amount of flour added affect the taste of cake).
- The dependent variable's value at a given value of the independent variables (e.g. the taste of cake for different amount of baking soda, baking temperature, and flour).

The general equation for multiple linear regression is –

y = a + b1X1 + b2X2 +...bnXn

where,

- **y** is response variable.
- **a, b1, b2...bn** are coefficients.
- **X1, X2, ...Xn** are predictor variables.

The **lm()** function in R is used to generate the regression model. Using the input data, the model calculates the coefficient values. Using these coefficients, you can then predict the value of the response variable for a given collection of predictor variables.

lm() Function:

The relationship model between the predictor and the response variable is created using this function.

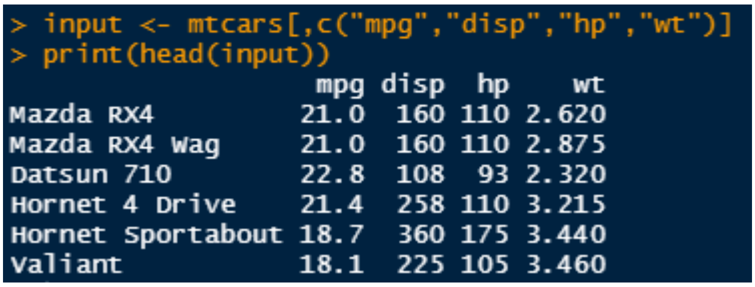
Syntax: The basic syntax for **lm()** function in multiple regression is –

lm(y ~ x1+x2+x3...,data),

The relationship between the response variable and the predictor variables is represented by a **formula**. The vector on which the formula will be applied is called **data**.

INPUT Data

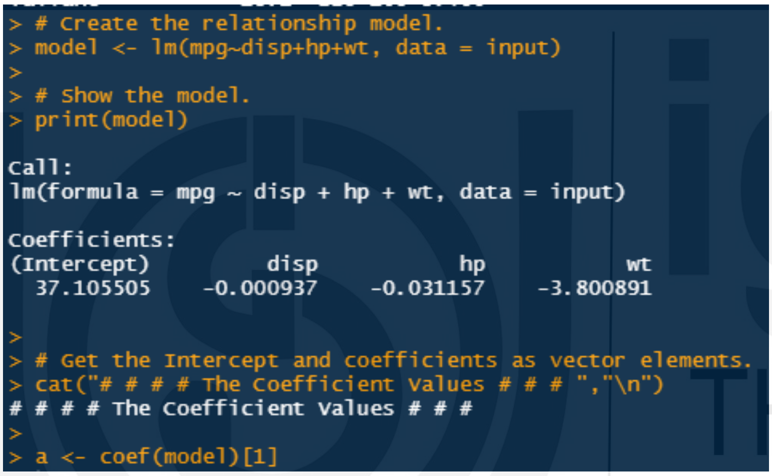
Let's take the R inbuilt data set "mtcars", which gives comparison between various car models based on the mileage per gallon (mpg), cylinder displacement ("disp"), horse power("hp"), weight of the car("wt") & more.
The aim is to establish relationship of mpg (response variable) with predictor variable (disp, hp, wt). The head function, as used in Figure 15.9, shows the first 5 rows of the dataset.



```
> input <- mtcars[,c("mpg","disp","hp","wt")]
> print(head(input))
                   mpg disp  hp    wt
Mazda RX4         21.0  160 110 2.620
Mazda RX4 Wag     21.0  160 110 2.875
Datsun 710        22.8  108  93 2.320
Hornet 4 Drive    21.4  258 110 3.215
Hornet Sportabout 18.7  360 175 3.440
Valiant           18.1  225 105 3.460
```

Figure 15.9: Sample data for Multiple regression

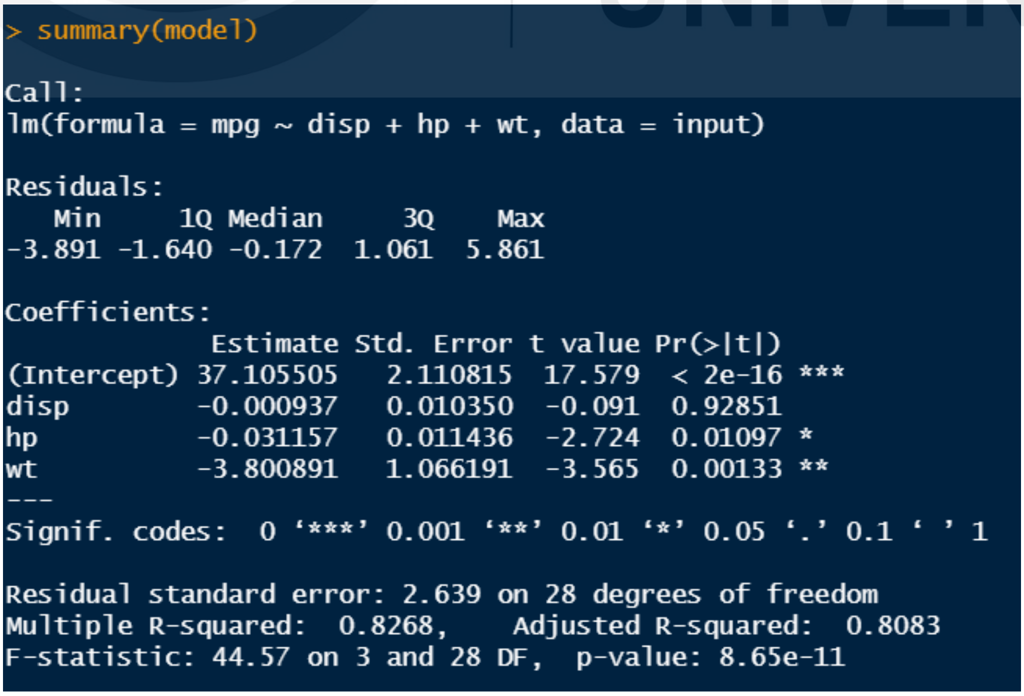
Creating Relationship model & getting the coefficients



```
> # Create the relationship model.
> model <- lm(mpg~disp+hp+wt, data = input)
>
> # Show the model.
> print(model)

Call:
lm(formula = mpg ~ disp + hp + wt, data = input)

Coefficients:
(Intercept)         disp           hp           wt
  37.105505    -0.000937    -0.031157    -3.800891

>
> # Get the Intercept and coefficients as vector elements.
> cat("# # # # The Coefficient Values # # # ","\n")
# # # # The Coefficient Values # # #
>
> a <- coef(model)[1]
```

Figure 15.10: The Regression model

Please note that the *input* is the name of a variable, which was created in Figure 15.9.



```
> summary(model)

Call:
lm(formula = mpg ~ disp + hp + wt, data = input)

Residuals:
   Min     1Q Median     3Q    Max 
-3.891 -1.640 -0.172  1.061  5.861 

Coefficients:
             Estimate Std. Error t value Pr(>|t|)    
(Intercept) 37.105505   2.110815  17.579  < 2e-16 ***
disp        -0.000937   0.010350  -0.091  0.92851    
hp          -0.031157   0.011436  -2.724  0.01097 *  
wt          -3.800891   1.066191  -3.565  0.00133 ** 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.639 on 28 degrees of freedom
Multiple R-squared:  0.8268,	Adjusted R-squared:  0.8083 
F-statistic: 44.57 on 3 and 28 DF,  p-value: 8.65e-11
```

Figure 15.11: Display of various output parameters

The results of regression as presented by R includes the following:

1. Five-point summary (Minimum, First Quartile, Median, Third Quartile, and Maximum). This shows the spread of the residual. You may observe that about 50% of residuals are in the range -1.640 to +1.061, which shows a good model fit.
2. Low p values mean the model is statistically significant.

Creating Equation for Regression Model: Based on the intercept & coefficient values one can create the mathematical equation as follows:

$$Y = a + b \times x_{disp} + c \times x_{hp} + d \times x_{wt}$$

or

$$Y = 37.15 - 0.000937 \times x_{disp} - 0.0311 \times x_{hp} - 3.8008 \times x_{wt}$$

The same equation will be applied in predicting new values.

Check your Progress 1

1. What is linear regression?

   …………………………………………………………………………..

   …………………………………………………………………………..
2. What does chi-square test answers?

   …………………………………………………………………………

   …………………………………………………………………………
3. Difference between linear and multiple regression?

   …………………………………………………………………………

   …………………………………………………………………………

15.6 LOGISTIC REGRESSION

In R Programming, logistic regression is a classification algorithm for determining the probability of event success and failure. When the dependent variable is binary (0/1, True/False, Yes/No), logistic regression is utilised. In a binomial distribution, the logit function is utilised as a link function.

Binomial logistic regression is another name for logistic regression. It is based on the sigmoid function, with probability as the output and input ranging from -∞ to +∞.The sigmoid function is given below:

$$g(z) = \frac{1}{1+ e^{-z}} \text{ Where } z = a + b \times x$$

The general equation for logistic regression is –

$$g(z) = \frac{1}{1 + e^{-(a+b_1\times x_1+b_2\times x_2+b_3\times x_3+\ldots)}}$$

where, **y** is called as the response variable, and x_i are predictors.
The *a* and b_i are coefficients.

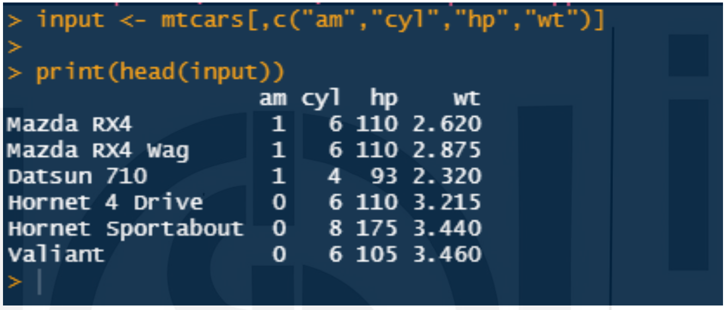
The glm() function is used to construct the regression model.

Syntax:

glm (formula, data,family)

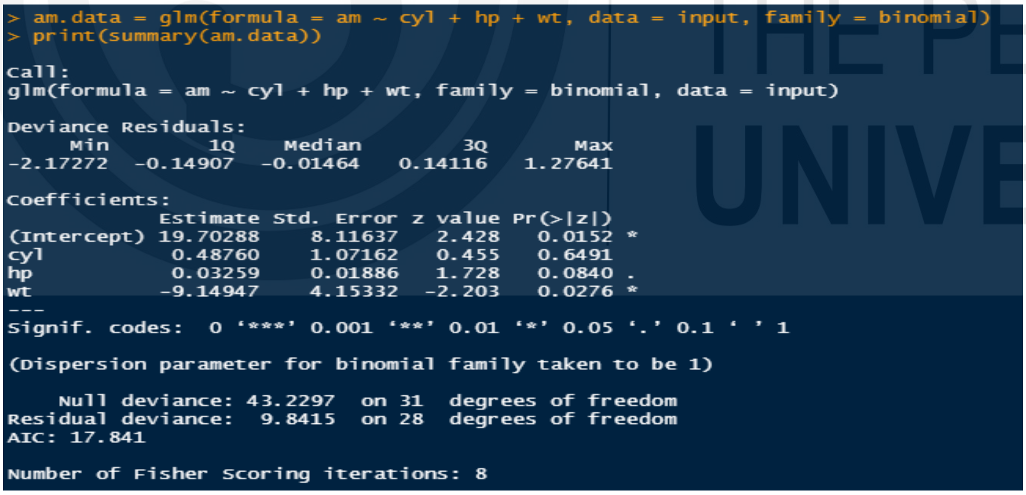
- The symbol expressing the relationship between the variables is a formula.
- The data set containing the values of these variables is known as data.
- family is a R object that specifies the model's details. For logistic regression, it has a binomial value.

Input Data: Let's take the R inbuilt data set “mtcars”, which provides details of various car models & engine specifications. The transmission mode of the car i.e. whether the car is manual or automatic is described by the column *am* having a binary value as 0 or 1. You can create the model between columns “am” (Outcome/ dependent/ response variable) and three others – hp, wt and cyl (predictor variables).This model is aimed at determining, if car would have manual or automatic transmission, given the horse power (hp), weight (wt) and number of cylinders (cyl) in the car.



```
> input <- mtcars[,c("am","cyl","hp","wt")]
>
> print(head(input))
                  am cyl  hp    wt
Mazda RX4          1   6 110 2.620
Mazda RX4 Wag      1   6 110 2.875
Datsun 710         1   4  93 2.320
Hornet 4 Drive     0   6 110 3.215
Hornet Sportabout  0   8 175 3.440
Valiant            0   6 105 3.460
> |
```

Figure 15.12: The sample data set for logistic regression



```
> am.data = glm(formula = am ~ cyl + hp + wt, data = input, family = binomial)
> print(summary(am.data))

Call:
glm(formula = am ~ cyl + hp + wt, family = binomial, data = input)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.17272  -0.14907  -0.01464   0.14116   1.27641

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 19.70288    8.11637   2.428   0.0152 *
cyl          0.48760    1.07162   0.455   0.6491
hp           0.03259    0.01886   1.728   0.0840 .
wt          -9.14947    4.15332  -2.203   0.0276 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 43.2297  on 31  degrees of freedom
Residual deviance:  9.8415  on 28  degrees of freedom
AIC: 17.841

Number of Fisher Scoring iterations: 8
```

Figure 15.13: The logistic regression model

The null deviance demonstrates how well a model with an intercept term can predict the dependent variable, whereas the residual deviance represents how well a model with n predictor variables can predict the dependent variable. Deviance is measure of goodness of fit of a model.

In the summary as the p-value is more than 0.05 for the variables "cyl" (0.0152) and "hp" (0.0276), we will consider them insignificant in contributing to the value of the variable "am". Only weight (wt) impacts the "am" value in this regression model.

15.7 TIME SERIES ANALYSIS

A Time Series is any metric that is measured at regular intervals. It entails deriving hidden insights from time-based data (years, days, hours, minutes) in order to make informed decisions. When you have serially associated data, time series models are particularly beneficial. Weather data, stock prices, industry projections, and so on are just a few examples.

A time series is represented as follows:

A data point, say (Y_t), at a specific time *t* (indicated by subscript *t*) is defined as the either sum or product of the following three components:

Seasonality (S_t), Trend (T_t); and Error (e_t) (also known as, **White Noise**).

Input: Import the data set and then use ts() function.
The steps to use the function are given below. However, it is pertinent to note here that the input values used in this case should ideally be a numeric vector belonging to the "numeric" or "integer" class.
The following functions will generate quarterly data series from 1959:
ts(inputData, frequency =4, start = c(1959,2)) #frequency 4 => QuarterlyData
The following function will generate monthly data series from 1990
ts(1:10, frequency =12, start = 1990) #freq 12 => MonthlyData
The following function will generate yearly data series from 2009 to 2014.
ts(inputData, start=c(2009), end=c(2014), frequency=1) # YearlyData

In case, you want to use Additive Time Series, you use the following:

$$Y_t = S_t + T_t + e_t$$

However, for Multiplicative Time Series, you may use:

$$Y_t = S_t \times T_t \times e_t$$

The additive time series can be converted from multiplicative time series by taking using the log function on the time series as represented below:

$$additiveTS = log(multiplcativeTS)$$

15.7.1 Stationary Time Series

A time series is considered "stationary" if the following criteria are satisfied:

1. When the mean value of a time series remains constant over a period of time and hence, the trend component is removed Over time, the variance does not increase.
2. Seasonality has a minor impact.

This means it has no trend or seasonal characteristics, making it appear to be random white noise regardless of the time span viewed.

Steps to convert a time series as stationary
Each data point in a time series is differentiated by subtracting it from the one before it. It is a frequent technique for making a time series immobile. To make a stationary series out of most time series patterns 1 or 2 differencing is required.

15.7.2 Extraction of trend, seasonality and error

Using decompose() and forecast::stl, the time series is separated into seasonality, trend, and error components (). You may use the following set of commands to do so.

```
timeSeriesData = EuStockMarkets[,1]
resultofDecompose = decompose(timeSeriesData, type=”mult”)
plot(resultofDecompose)
resultsofSt1 = stl(timeSeriesData, s.window = “periodic”)
```

15.7.3 Creating lags of a time-series

A lag of time series is generated when the time basis is shifted by a given number of periods. Moreover, the state of a time series a few periods ago, however, may still have an effect on its current state. Hence, in the time series models, the delays of a time series are typically used as explanatory variables.

```
lagTimeSeries = lag(timeSeriesData, 3) #Shifting to 3 periods earlier
library(DataCombine)
mydf = as.data.frame(timeSeriesData)
mydf = slide(mydf, “x”, NewVar = “xLag1”, slideBy = -1) #create lag1
variable
mydf = slide(mydf, “x”, NewVar = “xLag1”, slideBy = 1)
```

Check your Progress 2

1. What is logistic regression?

   ……………………………………………………………………………..

2. What are the uses of Time-Series analysis?

   …………………………………………………………………………….

3. Differentiate between linear regression and logistic regression?

   …………………………………………………………………………….

15.8 SUMMARY

This unit introduces the concept of data analysis and examine its application using R programming. It explains about the Chi-Square Test that is used to determine if two categorical variables are significantly correlated and further study its application on R. The unit explains the Regression Analysis, which is a common statistical technique for establishing a relationship model between two variables- a predictor variable and the response variable. It further explains the various models in Regression Analysis including Linear and Logistics Regression Analysis. In Linear Regression the two variables are related through an equation of degree is one and employs a straight line to explain the relationship between variables. It is categorised into two types- Simple Linear Regression which uses only one independent variable and Multiple Linear Regression which uses two or more independent variables. Once familiar with the Regression, the unit proceeds to explain about the logistic regression, which is a classification algorithm for determining the probability of event success and failure. It is also known as Binomial logistic regression and is based on the sigmoid function, with probability as the output and input ranging from $-\infty$ to $+\infty$. At the end, the unit introduces the concept of time series analysis and help understand its application and usage on R. It also discusses the special case of Stationary Time Series and how to make a time series stationary. This section further explains how to extract the trend, seasonality and error in a time series in R and the creating lags of a time series.

15.9 ANSWERS

Check your Progress 1

1. A regression model that employs a straight line to explain the relationship between variables is known as linear regression. In Linear Regression these two variables are related through an equation, where exponent (power) of both these variables is one. It searches for the value of the regression coefficient(s) that minimises the total error of the model to find the line of best fit through your data.
2. The Chi-square test of independence determines whether there is a statistically significant relationship between categorical variables. It's a hypothesis test that answers the question—do the values of one categorical variable depend on the value of other categorical variables?
3. Linear regression considers 2 variables whereas multiple regression consists of 2 or more variables.

Check your Progress 2

1. Logistic regression is an example of supervised learning. It is used to calculate or predict the probability of a binary (yes/no) event occurring.
2. Time series analysis is used to identify the fluctuation in economics and business. It helps in the evaluation of current achievements. Time series is used in pattern recognition, signal processing, weather forecasting and earthquake prediction.
3. The problems pertaining to regression are solved using linear regression; however, the problems pertaining to classification are solved using the logistic regression. The linear regression yields a continuous result, whereas logistic regression yields discrete results.

15.10 REFERENCES AND FURTHER READINGS

1. De Vries, A., & Meys, J. (2015). *R for Dummies*. John Wiley & Sons.
2. Peng, R. D. (2016). *R programming for data science* (pp. 86-181). Victoria, BC, Canada: Leanpub.
3. Schmuller, J. (2017). *Statistical Analysis with R For Dummies*. John Wiley & Sons.
4. Field, A., Miles, J., & Field, Z. (2012). *Discovering statistics using R*. Sage publications.
5. Lander, J. P. (2014). *R for everyone: Advanced analytics and graphics*. Pearson Education.
6. Lantz, B. (2019). *Machine learning with R: expert techniques for predictive modeling*. Packt publishing ltd.
7. Heumann, C., & Schomaker, M. (2016). *Introduction to statistics and data analysis*. Springer International Publishing Switzerland.
8. Davies, T. M. (2016). *The book of R: a first course in programming and statistics*. No Starch Press.
9. https://www.tutorialspoint.com/r/index.html
10. https://data-flair.training/blogs/chi-square-test-in-r/
11. http://r-statistics.co/Time-Series-Analysis-With-R.html
12. http://r-statistics.co/Logistic-Regression-With-R.html