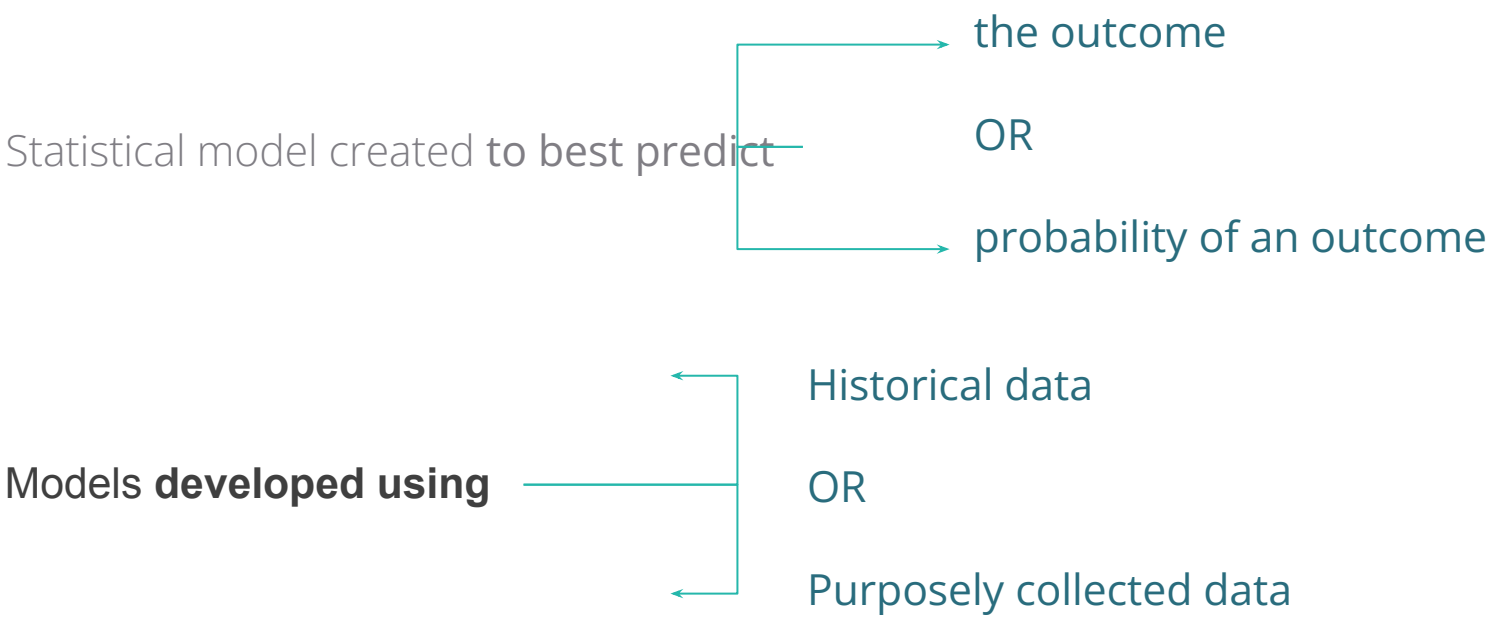
# Introduction to Predictive Modelling

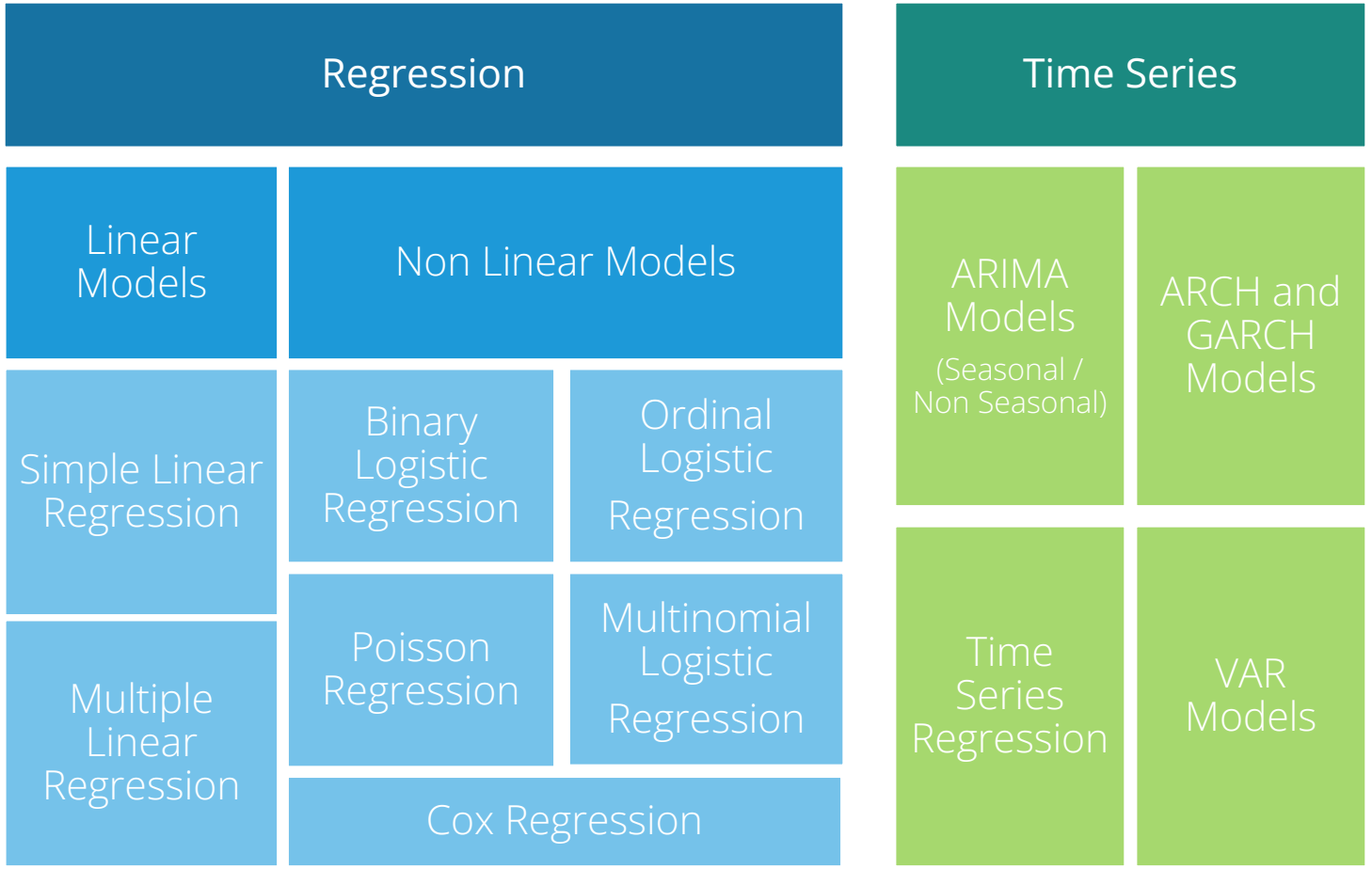
# Contents

# What is Predictive modelling?

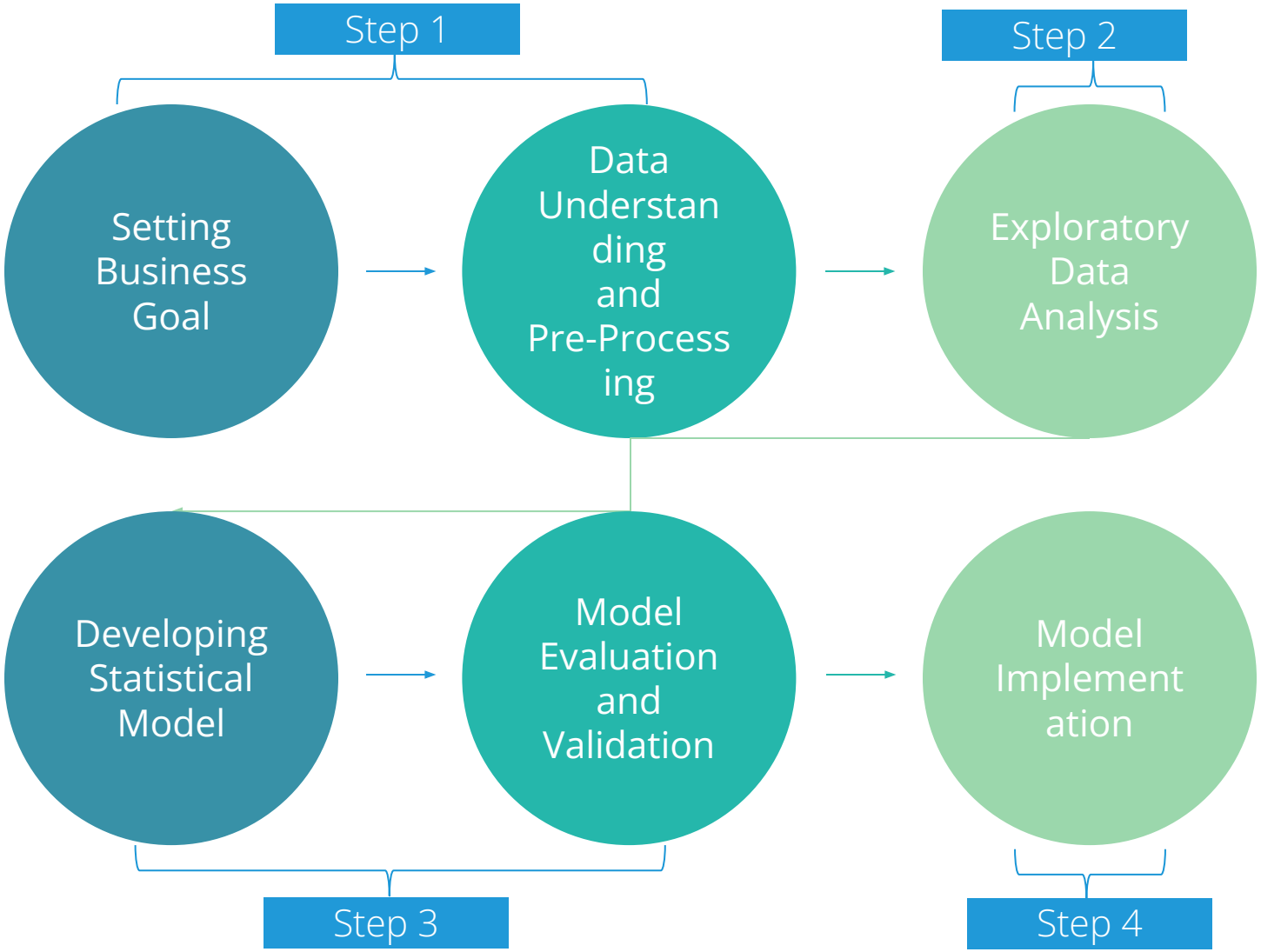



Predictive analytics is used in **financial services, insurance, telecommunications, retail, travel, healthcare, pharmaceuticals, sports and several other fields**

# Predictive modelling – Important Techniques



# Predictive modelling – General Approach



# Step 1 – Data Understanding and Pre-processing

**Data Understanding and Pre Processing**

- Data Understanding
  - Understanding data dimension, variable types, variable relationships
- Converting raw data to usable data

  Data cleaning by checking for and handling:
  - Missing values
  - Inconsistencies
- Transforming variables
- Feature engineering
  - Using domain knowledge to create new features or variables which can be used in the model

Pre-Processing:

- Grouping / Factoring / Segmentation / Reduction

# Step 2 – Exploratory Data Analysis

## Exploratory Data Analysis

- **Performing exploratory data analysis** using:
  - Frequency tables
  - Cross tables
  - Descriptive statistics
  - Visualizations
  - Correlation matrix

# Step 3 – Model Identification , Selection and Validation

## Model Selection and Validation

- **Model identification and selection** is based on:
  - Study objective
  - Type of dependent variable
  - Checking different statistics / decision criteria based on models (Eg. p-value, $R^2$, AIC, etc.) i.e diagnostic checks
  - Automatic search procedures
- **Cross Validation**
  - Splitting data into training data and test data
  - Checking predictive ability of model on new data
  - Comparison of results with theoretical expectations, empirical results and simulation results

# Step 4 – Model Implementation

Model Implementation

- **Drawing inferences from the model results** by :
  - Building equations using only the coefficients of significant variables
  - Mapping the model chosen with the existing system
- Fitting the model on new data and generating predictions

- **Observing values of Predictors :**
  - In a spreadsheet or
  - Web application or any other user interface or
  - Integrating the current systems

# Get an Edge!

Any predictive model is developed on historical data. Sample size and data dimension are key determinants for a good model

- If sample size is too small, model may not give good insight about the relationship among the variables.
- Also, if the data has large number of variables (columns) but few observations (rows), we are essentially trying to learn too much from a small sample. Results from models developed using such data will be erratic. **The rule of thumb for appropriate sample dimension is that observations should be 10 times the number of variables.** For instance, if we wish to study the relationship of 8 variables, then we must have more than 80 observations.

# Quick Recap

In this session, we gained knowledge on the concept of **predictive modelling :**

| | |
|---|---|
| Predictive modelling | • Used to predict the outcome or the probability of an outcome<br>• Models developed using historical data or purposely collected data |
| Important Statistical models | • Regression Models- Linear and non-linear<br>• Time Series Models |
| General Approach in Predictive modelling | • Data understanding and pre-processing<br>• Exploratory data analysis<br>• Model selection and validation<br>• Model Implementation |

# Multiple Linear Regression

## Introduction

# Content

# Multiple Linear Regression

- Multiple linear regression is used to explain the relationship between one continuous dependent variable and two or more independent variables.
- The independent variables can be continuous or categorical.
- Multiple Linear Regression is used when we want to predict the value of a variable based on the values of two or more other variables.
- The variable we want to predict is called the dependent variable
- The variables used to predict the value of dependent variable are called independent variables (or explanatory variables/predictors).
- Multiple linear regression requires the model to be linear in the parameters.
- Example: The price house in USD can be dependent variable and  area of house, location of house , air quality index in the area, distance from airport etc. can be independent variables.

# Statistical Model

$$Y = b_0 + b_1X_1 + b_2X_2 + \ldots + b_pX_p + e$$

where,

| | | |
|---|---|---|
| Y | : | Dependent Variable |
| $X_1, X_2, \ldots, X_p$ | : | Independent Variables |
| $b_0, b_1, \ldots, b_p$ | : | Parameters of Model |
| e | : | Random Error Component |

- Independent variables can either be **Continuous or Categorical**
- Multiple linear regression **requires the model to be linear in the parameters**
- Parameters of the model are estimated by Least Square Method.
- The **least squares (LS)** criterion states that the **sum of the squares of errors** (or residuals) **is minimum**.
- Mathematically, following quantity is minimized to estimate parameters using least square method.

$$\hat{}$$

- Error ss= $\Sigma (Yi - \hat{Yi})2$

# Case Study – Modeling Job Performance Index

## Background

- A company conducts different written tests before recruiting employees. The company wishes to see if the scores of these tests have any relation with post-recruitment performance of those employees.

## Objective

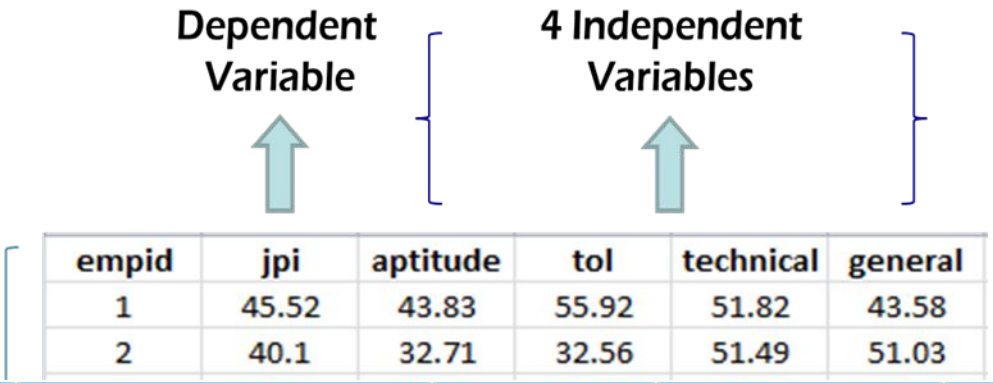
- To predict employees' job performance index after probationary period, based on scores of tests conducted at the time of recruitment

## Available Information

- Sample size is 33
- Independent Variables: Scores of tests conducted before recruitment on the basis of four criteria – **Aptitude, Test of Language, Technical Knowledge, General Information**
- Dependent Variable: **Job Performance Index** calculated after an employee finishes probationary period (6 months)

# Data Snapshot

Performance Index




| empid | jpi | aptitude | tol | technical | general |
|---|---|---|---|---|---|
| 1 | 45.52 | 43.83 | 55.92 | 51.82 | 43.58 |
| 2 | 40.1 | 32.71 | 32.56 | 51.49 | 51.03 |

| Columns | Description | Type | Measurement | Possible values |
|---|---|---|---|---|
| empid | Employee ID | integer | - | - |
| jpi | Job performance Index | numeric | - | positive values |
| aptitude | Aptitude score | numeric | - | positive values |
| tol | Test of Language | numeric | - | positive values |
| technical | Technical Knowledge | numeric | - | positive values |
| general | General Information | numeric | - | positive values |

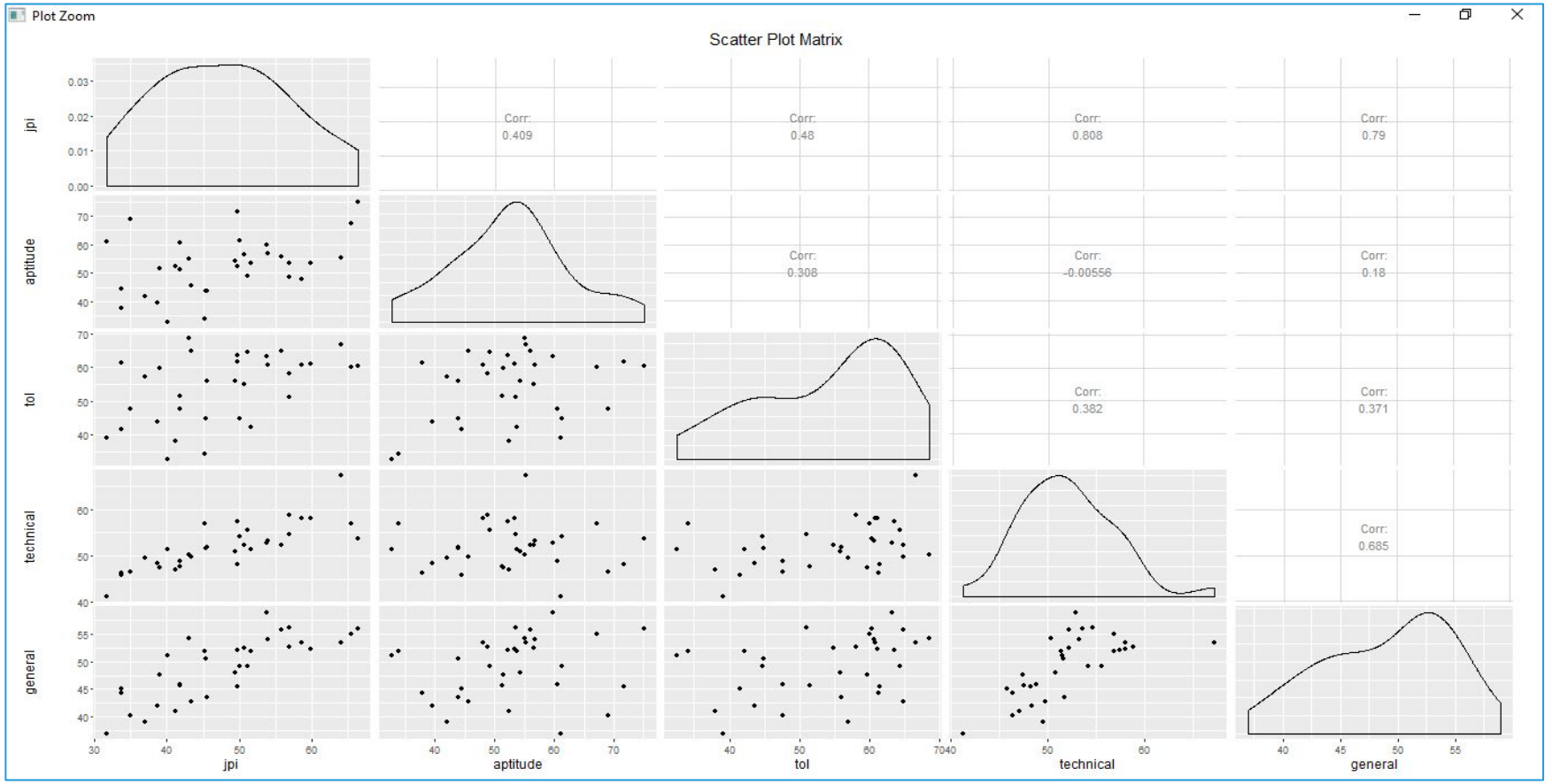
# Graphical Representation of Data

- It is always recommended to have a general look at your data and behavior of all the variables before moving to modeling.
- This helps you in making intuitive inferences about the data, which can be statistically validated by your final model.
- The simplest way of doing this is creating a scatter plot matrix, which will give bivariate relationships between variables.

```
#Importing the Data
perindex<-read.csv("Performance Index.csv",header=TRUE)

#Graphical Representation of the Data
library(GGally)

ggpairs(perindex[,c("jpi","aptitude","tol","technical","general")],
        title="Scatter Plot Matrix",
        columnLabels=c("jpi","aptitude","tol","technical","general"))
```

# Scatter Plot Matrix

**`ggpairs()`** function in package **`GGally`** gives a Generalised Pairs Plot which not only visualises bivariate scatter relationships, but also gives their quantified representation, in the form of Correlation Coefficients, along with Distribution for each variable



# Simple v/s Multiple Linear Regression

- Using simple linear regression to solve such a problem is not wrong. For instance, we can study the impact of aptitude on job performance, then see the impact of technical expertise on job performance and so on. But is this approach efficient? Certainly not!

Why is Multiple Linear Regression a better method than Simple Linear Regression?

Single predictor provides inadequate information about the response variable

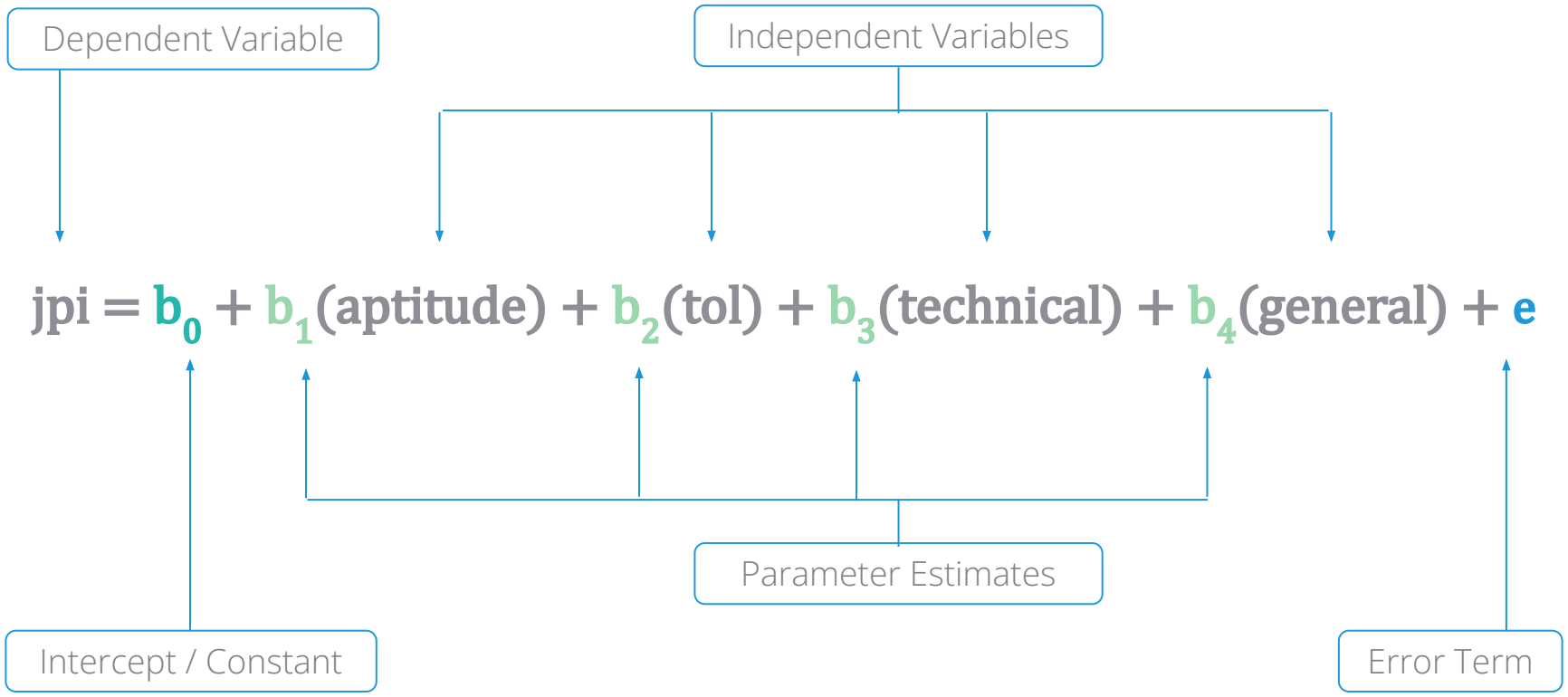
Simultaneous study of multiple variables is essential as the response is always influenced by more than one variables

# Questions that MLR Answers

Multiple linear regression analysis of employee performance indices answers the following questions:

1. Do tests conducted at the time of recruitment determine a candidate's performance in the first six months in that job?
2. Which of the four test scores is most significant in determining job performance?
3. Can any of the tests be discontinued?
4. Can performance of newly recruited candidates be estimated based on the scores of tests conducted at the time of their recruitment?

# Model for the Case Study



# Parameter Estimation using Least Square Method

| Parameters | Coefficients |
|---|---|
| Intercept | -54.2822 |
| aptitude | 0.3236 |
| tol | 0.0334 |
| technical | 1.0955 |
| general | 0.5368 |

E(jpi)= -54.2822 + 0.3236 (aptitude) + 0.0334 (tol) + 1.0955 (technical) + 0.5368 (general) + e

# Parameter Estimation Using lm function in R

```
#Model Fit
jpimodel<-lm(jpi~aptitude+tol+technical+general, data=perindex)
jpimodel
```

- *lm() fits a linear regression.*
- *~ separates dependent and independent variables*
- *Left hand side of tilde(~) represents the dependent variable and right-hand side shows independent variables*
- *+ separates multiple independent variables.*

```
#Output
Coefficients:
(Intercept)     aptitude          tol    technical      general
  -54.28225      0.32356      0.03337      1.09547      0.53683
```

- *Coefficients are the model parameters.*
- *Signs of each parameter represent their relationship with the dependent variable.*

***** **~. in lm() function uses all variables except the dependent variable. This is helpful when the data has a large number of predictors.**

# Interpretation of Partial Regression Coefficients

- For every unit increase in the independent variable (X), the expected value of the dependent variable (Y) will change by the corresponding parameter estimate (b), keeping all the other variables constant

| Parameters | Coefficients |
|---|---|
| Intercept | -54.2822 |
| aptitude | 0.3236 |
| tol | 0.0334 |
| technical | 1.0955 |
| general | 0.5368 |

- From the parameter estimates table, we observe that the parameter estimate for Aptitude Test is 0.3236

  We can infer that for one unit increase in aptitude test score, the expected value of job performance index will increase by 0.3236 units

# Quick Recap

| | |
|---|---|
| Understand the Data | • Ensure the data is complete and consistent<br>• Identify dependent and independent variables |
| **Simple Data Check** | • Use **`pairs()`** function to yield a simple scatter plot<br>• Use **`ggpairs()`** function from **`GGally`** package for a more nuanced plot (Recommended) |
| Fit a Model | • Run a regression and obtain ordinary least square estimates of parameters<br>• lm() function fits a linear regression model |

# Multiple Linear Regression

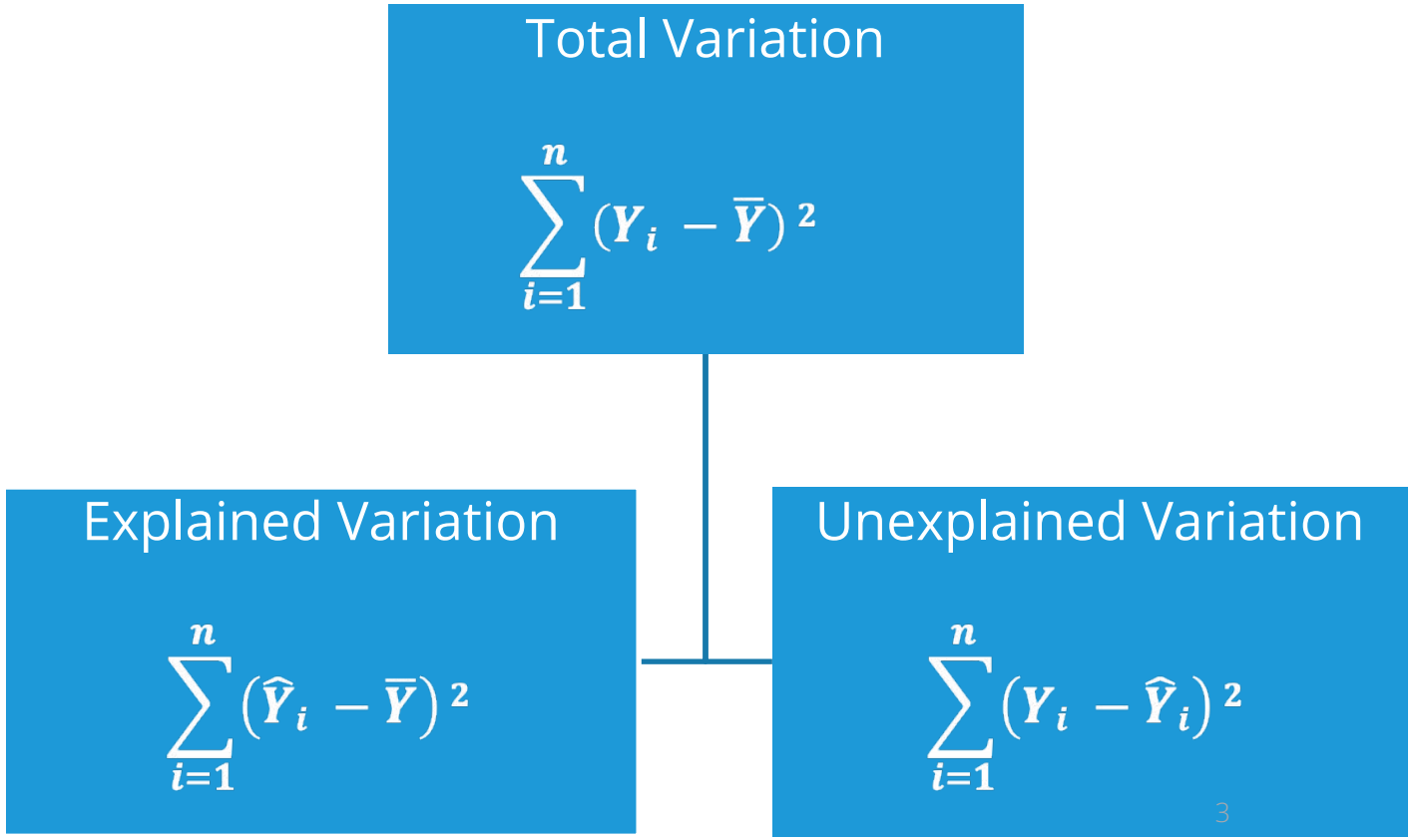
# Introduction – Part II

# Content

1. Global Testing – ANOVA
2. Individual Testing – t Test
3. Measure of Goodness of Fit – R Squared
4. Fitted values and Residuals
5. Predictions for New Dataset
6. Standardizing Coefficients

# Partitioning Total Variance

- Total Variation in dependent variables Y can be split into two: Explained and Unexplained.
- Explained variation is the summation of squared difference between estimated values of Y and the mean value of Y. Whereas, sum of squared difference between the actual values of Y and estimated values is considered to be unexplained.



# Global Testing – Using F Test

Testing whether at least one variable is significant

| | |
|---|---|
| **Objective** | To test the **null hypothesis** that **all the parameters are simultaneously equal to zero** |

Null Hypothesis ($H_0$): $b_1 = b_2 = \ldots = b_p = 0$

Alternate Hypothesis ($H_1$): At least one coefficient is not zero

| | |
|---|---|
| **Test Statistic** | $F = \frac{\text{Mean Square of Regression}}{\text{Mean Square of Error}}$ |
| **Decision Criteria** | Reject the null hypothesis if p-value < 0.05 |

# Global Testing – Using F Test

## ANOVA Table

| Source | DF (Degrees of Freedom) | SS (Sum of Squares) | MSS =SS/DF (Mean Sum of Squares) | F Value | Pr > F |
|---|---|---|---|---|---|
| Regression(Explained) | p=4 | 2510.007 | 627.5017 | 49.8129 | <0.0001 |
| Error(Unexplained) | n-p-1=28 | 352.7208 | 12.5972 | | |
| Total | n-1=32 | 2862.728 | | | |

Reject the null hypothesis **since p-value < 0.05**

At least one variable has significant impact on performance index

* **Note : This slide is in continuation of the previous slide. So the data considered is the same, “Performance Index” data.**

# Individual Testing – Using t Test

Testing **which variable is significant**

| | |
|---|---|
| **Objective** | To test the **null hypothesis** that **parameters of individual variables are equal to zero** |

Null Hypothesis ($H_0$): $b_i = 0$

Alternate Hypothesis ($H_1$): $b_i \neq 0$

where i = 1,2,…,p

| | |
|---|---|
| **Test Statistic** | $t = \frac{\text{Estimated } b_i}{\text{Standard Error of Estimated } b_i}$ |
| **Decision Criteria** | Reject the null hypothesis **if p-value < 0.05** |

# Individual Testing – Using t Test

| Parameters | Coefficients | Standard Error | t statistic | p-value |
|---|---|---|---|---|
| Intercept | -54.2822 | 7.3945 | -7.3409 | 0.0000 |
| aptitude | 0.3236 | 0.0678 | 4.7737 | 0.0001 |
| tol | 0.0334 | 0.0712 | 0.4684 | 0.6431 |
| technical | 1.0955 | 0.1814 | 6.0395 | 0.0000 |
| general | 0.5368 | 0.1584 | 3.3890 | 0.0021 |

p-values for aptitude, technical and general are < 0.05

p-value for test of language (tol) is > 0.05

Therefore, tol is the only insignificant variable

# Measure of Goodness of Fit – R Squared

$R^2$ is the proportion of variation in the dependent variable which is explained by the independent variables. Note that $R^2$ always increases if variable is added in the model

$$R^2 = \frac{\text{Explained Variation}}{\text{Total Variation}} = \frac{\sum_{i=1}^{n} (\hat{Y}_i - \bar{Y})^2}{\sum_{i=1}^{n} (Y_i - \bar{Y})^2}$$

The adjusted R-squared is a modified version of R-squared that has been adjusted for the number of predictors in the modelling.

$$R_a^2 = 1 - \frac{n-1}{n-p-1}(1 - R^2)$$

The adjusted R-squared is a modified version of R-squared that has been adjusted for the number of predictors in the model.

Normally, $R^2$ greater than 0.7 is considered as a benchmark for accepting goodness of fit of a model.

# Understanding Summary Output

```
#Model Summary
summary(jpimodel)
```

*summary() generates a detailed description of the model.*

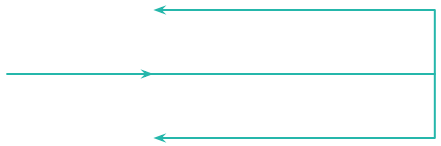
```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -54.28225    7.39453  -7.341 5.41e-08 ***
aptitude      0.32356    0.06778   4.774 5.15e-05 ***
tol           0.03337    0.07124   0.468   0.6431
technical     1.09547    0.18138   6.039 1.65e-06 ***
general       0.53683    0.15840   3.389   0.0021 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 3.549 on 28 degrees of freedom
Multiple R-squared:  0.8768,    Adjusted R-squared:  0.8592
F-statistic: 49.81 on 4 and 28 DF,  p-value: 2.467e-12
```

*Interpretation :*

- *Reject null hypothesis that no variables are significant as p-value is<0.05*
- *aptitude, technical, general are significant variables (p-values<0.05)*
- *tol is not significant (p-value>0.05)*

# Summary of Findings

**Significant variables**



**Aptitude**
**Technical knowledge**
**General information**

Out of four dependent variables, **three affect job performance index positively**

---

$R^2$ → **0.88**

88% of the variation in job performance index is explained by the model & 12% is unexplained variation

# Fitted Values and Residuals

- Fitted values (also called 'Predicted Values') are calculated using **estimated model** parameters and by substituting values of independent variables. The model now will include only the significant variables.

Estimated Model:

E(jpi)= **-54.40644** + **0.33335***aptitude + **1.11663***technical + **0.54316***general

| Values of Independent Variables for First Employee | |
|---|---|
| **aptitude** | 43.83 |
| **tol** | 55.92 |
| **technical** | 51.82 |
| **general** | 43.58 |

# Fitted Values and Residuals

| Values of Independent Variables for First Employee | |
|---|---|
| aptitude | 43.83 |
| technical | 51.82 |
| general | 43.58 |

aptitude technical general

jpi= **-54.40644** + **0.33335***43.83 + **1.11663***51.82 + **0.54316***43.58

**Predicted jpi= 41.73850**

**Residual**= Observed jpi – Predicted jpi = 45.52 - 41.73850 = 3.781497

# Fitted Values and Residuals

```
#Model Fitting after eliminating the insignificant variable
jpimodel_new<-lm(jpi~aptitude+technical+general,data=perindex)
jpimodel_new
```

*The insignificant variable tol is not included in the new model*

```
#Output
Coefficients:
(Intercept)     aptitude    technical      general
   -54.4064       0.3333       1.1166       0.5432
```

*Estimated values of the model parameters using the new model*

* To get the fitted values and the residuals values, the model should include only the significant variables

# Fitted Values and Residuals

```
#Adding Fitted Values and Residuals to the Original Dataset
perindex$pred<-fitted(jpimodel_new)

perindex$resi<-residuals(jpimodel_new)
```

*fitted() and residuals() fetch fitted values and residuals respectively.*

#Output

| | empid | jpi | aptitude | tol | technical | general | pred | resi |
|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 45.52 | 43.83 | 55.92 | 51.82 | 43.58 | 41.73850 | 3.781497 |
| 2 | 2 | 40.10 | 32.71 | 32.56 | 51.49 | 51.03 | 41.70973 | -1.609731 |
| 3 | 3 | 50.61 | 56.64 | 54.84 | 52.29 | 52.47 | 51.36215 | -0.752151 |
| 4 | 4 | 38.97 | 51.53 | 59.69 | 47.48 | 47.69 | 41.69149 | -2.721486 |
| 5 | 5 | 41.87 | 51.35 | 51.50 | 47.59 | 45.77 | 40.71145 | 1.158549 |
| 6 | 6 | 38.71 | 39.60 | 43.63 | 48.34 | 42.06 | 35.61699 | 3.093010 |

*Interpretation :*

- *pred values are calculated based on the values of the model parameters*
- *resi is the difference between the actual jpi values and the pred values.*
- *Lower the residuals, lesser is the difference between fitted and observed and better is the model.*

# Predictions for New Dataset

- New data set should have all the independent variables used in the model
- Column names of all common variables in the new and old datasets should be identical
- Note that missing values will be taken as 0 (which can be incorrect)

```
#Importing New Dataset
perindex_new<-read.csv("Performance Index new.csv", header=TRUE)

perindex_new$pred<-predict(jpimodel_new,perindex_new)


head(perindex_new)
```

*predict() returns predicted values. Fitted model is the first argument and new dataset object is the second argument. This ensures R uses parameters from the fitted model for predictions on new data.*

| | empid | jpi | tol | technical | general | aptitude | pred |
|---|---|---|---|---|---|---|---|
| 1 | 34 | 66.35 | 59.20 | 57.18 | 54.98 | 66.74 | 61.55258 |
| 2 | 35 | 56.10 | 64.92 | 52.51 | 55.78 | 55.45 | 53.00898 |
| 3 | 36 | 48.95 | 63.59 | 57.76 | 52.08 | 51.73 | 55.62154 |
| 4 | 37 | 43.25 | 64.90 | 50.13 | 42.75 | 45.09 | 39.82060 |
| 5 | 38 | 41.20 | 51.50 | 47.89 | 45.77 | 50.85 | 40.87977 |
| 6 | 39 | 50.24 | 55.77 | 51.13 | 47.98 | 53.86 | 46.70139 |

# Predictions with Confidence Interval

```
#Predictions with Confidence Interval
predict(jpimodel_new, perindex_new, interval="confidence")
```

*interval ="confidence" generates 95% confidence intervals by default*

```
#Output
       fit      lwr      upr
1 61.55258 59.00956 64.09559
2 53.00898 50.67792 55.34004
3 55.62154 53.65401 57.58906
4 39.82060 37.73390 41.90730
5 40.87977 39.23364 42.52590
6 46.70139 45.41627 47.98650
```

**Q. Why are confidence intervals needed for predictions?**

**A.** The point estimate is the best guess of the true value of the parameter, while the interval estimate gives a measure of accuracy of that point estimate by providing an interval that contains plausible values.

* **If you wish to specify the level of tolerance/confidence, use level= argument in the predict() function. For example, to calculate 90% confidence intervals, level=0.90**

# Standardized Coefficients

How to determine relative importance of predictors?

One possible answer is standardized regression coefficient

Predictors can have very different types of units, which make comparing the regression coefficients meaningless. One solution is to standardize all variables before performing regression analysis.

**standardization refers to the process of subtracting the mean ( $\mu$ ) from each value and dividing by the standard deviation ($\sigma$ ).**

$$Z = \frac{X - \mu}{\sigma}$$

| | X1 | X2 | Standardized X1 | Standardized X2 |
|---|---|---|---|---|
| | 32 | 1052 | -0.20 | -1.74 |
| | 37 | 1237 | 0.46 | -1.06 |
| | 25 | 1672 | -1.12 | 0.54 |
| | 39 | 1724 | 0.72 | 0.74 |
| | 23 | 1555 | -1.38 | 0.11 |
| | 41 | 1423 | 0.99 | -0.37 |
| | 43 | 1870 | 1.25 | 1.27 |
| | 28 | 1661 | -0.72 | 0.50 |
| | | | | |
| Mean | 33.5 | 1524.25 | | |
| SD | 7.60 | 271.69 | | |

# Standardized Coefficient - R code

**Generation of standardized parameter estimate**

```
#Install and load package lm.beta

install.packages("lm.beta")
library(lm.beta)

lm.beta(jpimodel_new)
```

- *lm.beta function in "lm.beta" is used to generate the standardized parameter estimate*

#Output

```
Standardized Coefficients::
(Intercept)     aptitude    technical      general
  0.0000000    0.3543742    0.5880966    0.3236793
```

*Interpretation:*

- *technical has highest impact on job performance index followed by aptitude*

# Quick Recap

| | |
|---|---|
| Check Variable Significance | • Undertake global and individual testing |
| Measure Goodness of Fit | • Check R-squared, Adjusted R-squared to see how much variation is explained by the model<br>• Generally, R-squared greater than 0.6 is considered to be a good indicator |
| Summary Output | • Summary of **lm()** output is exhaustive and gives t statistics, p-value, $R^2$ to draw fundamental conclusions about the model |

# Quick Recap

| | |
|---|---|
| Fitted Values and Errors | • **`fitted()`** and **`residuals()`** are used to fetch fitted values and residuals respectively |
| Predictions | • **`predict()`** function predicts values for new data<br>• Predictions can be obtained as either point estimates or as confidence intervals |
| Standardizing Coefficients | • **lm.beta()** function in package **lm.beta** gives the standardized coefficients.<br>• It is used to compare the relative importance of independent variables when the variables are in different metric units |

# Multiple Linear Regression

# Using Categorical Variables

# Content

1. Categorical Variables
2. Dummy Variables
3. MLR using Categorical Independent Variables
4. Changing the base category

# Categorical Variables (Nominal Scale)

- Categorical variables with a nominal scale define sub-groups in the data.
  Example: Region, department etc.

- Even if these variables are represented using numerical values such as 1,2,3,4 etc, the numbers do not have any mathematical meaning.

- A Regression model can include these variables in the model

# What Are Dummy Variables?

- Regression analysis requires numerical variables.
- So when there are categorical variables in a regression model , we create dummy variables for them.
- A dummy variable is a binary variable which takes the values 1 or zero.
- Dummy variables are useful because they enable us to use a single regression equation to represent multiple groups. This means that we do not need to write out separate equation models for each subgroup.

# Case Study – Predicting Restaurant Sales

## Background

- A city-based association of restaurants and cafes records all sorts of transactions and descriptive data for the purpose of industry-level analysis. The association wishes to find out if this data can be used to determine sales of restaurants.

## Objective

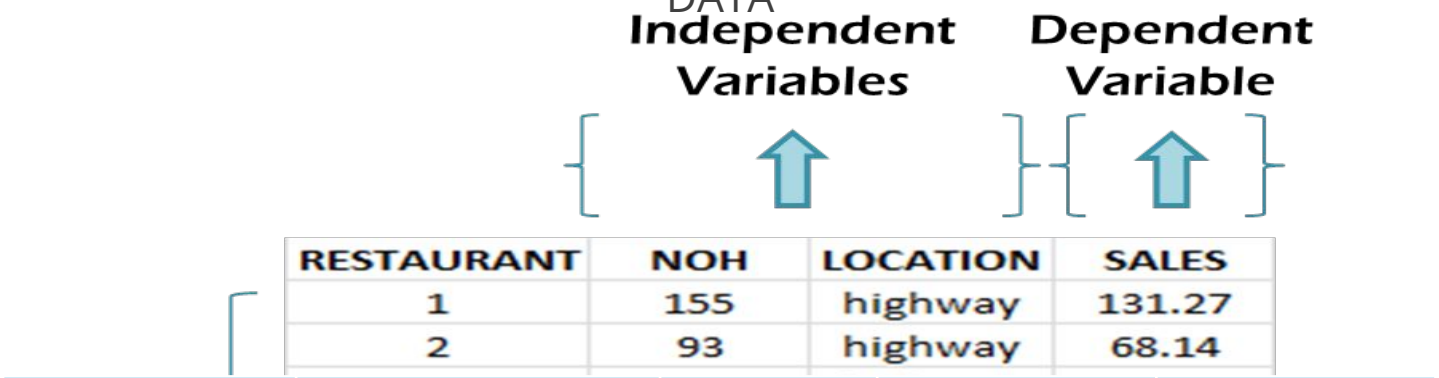
- To predict sales of restaurants

## Available Information

- Sample size is 16
- Independent Variables: **Location of the Restaurant** – Categorical Variable with 3 Categories – Mall, Street and Highway and **Number of Households in the Area**
- Dependent Variable: **Sales of the Restaurant**

# Data Snapshot

RESTAURANT SALES DATA




| Columns | Description | Type | Measurement | Possible values |
|---|---|---|---|---|
| RESTAURANT | Restaurant Number | numeric | - | - |
| NOH | Number of Households in the Vicinity of the Restaurant | numeric | - | positive values |
| LOCATION | Whether the Restaurant is Situated in a Mall, on a Street or on a Highway | Categorical | mall, street, highway | 3 |
| SALES | Annual Sales of the Restaurant | numeric | - | positive values |

# MLR using Categorical Independent Variables

If there is a categorical independent variable with K categories we must define only K – 1 dummy variables

In the case study,

| | |
|---|---|
| Dependent Variable | Sales of a Restaurant |
| Independent Variables | 1. Location of the Restaurant<br>2. Number of Households in the Area |

- Location is a categorical variable with 3 categories – Mall, Street and Highway
  Therefore, there will be 3-1=2 dummy variables
- The category for which dummy variable is not defined is called 'Base Category'

# Data Snapshot – With Dummy Variables

Y : Sales of a Restaurant

X1 : No. of Households

X2 : 1 if location is 'Mall' and 0 otherwise

X3 : 1 if location is 'Street' and 0 otherwise

- The location is "highway" if MALL=0 and STREET=0

| RESTAURANT | NOH | LOCATION | SALES | MALL | STREET |
|---|---|---|---|---|---|
| 1 | 155 | highway | 135.27 | 0 | 0 |
| 2 | 93 | highway | 72.74 | 0 | 0 |
| 3 | 128 | highway | 114.95 | 0 | 0 |
| 4 | 114 | highway | 102.93 | 0 | 0 |
| 5 | 158 | highway | 131.77 | 0 | 0 |
| 6 | 183 | highway | 160.91 | 0 | 0 |
| 7 | 178 | mall | 179.86 | 1 | 0 |
| 8 | 215 | mall | 220.14 | 1 | 0 |
| 9 | 172 | mall | 179.64 | 1 | 0 |
| 10 | 197 | mall | 185.92 | 1 | 0 |
| 11 | 207 | mall | 207.82 | 1 | 0 |
| 12 | 95 | mall | 113.51 | 1 | 0 |
| 13 | 224 | street | 203.98 | 0 | 1 |
| 14 | 199 | street | 174.48 | 0 | 1 |
| 15 | 240 | street | 220.43 | 0 | 1 |
| 16 | 100 | street | 93.19 | 0 | 1 |

# Why Not K Dummy Variables?

Can there be as many dummy variables as categories?

- If k dummy variables are created for k categories, there will be perfect multicollinearity – The Dummy Variable Trap
- In order to avoid falling into this trap, model with k categories and k dummy variables must have no intercept
- In such a model, coefficients will directly represent mean value of that variable

However, it is desirable to stick to the rule of k categories = k – 1 Dummy Variables

# Statistical Model Using Dummy Variables

Basic Multiple Linear Regression Model

$$Y = b_0 + b_1X_1 + b_2X_2 + \ \ldots \ + b_pX_p + e$$

where,

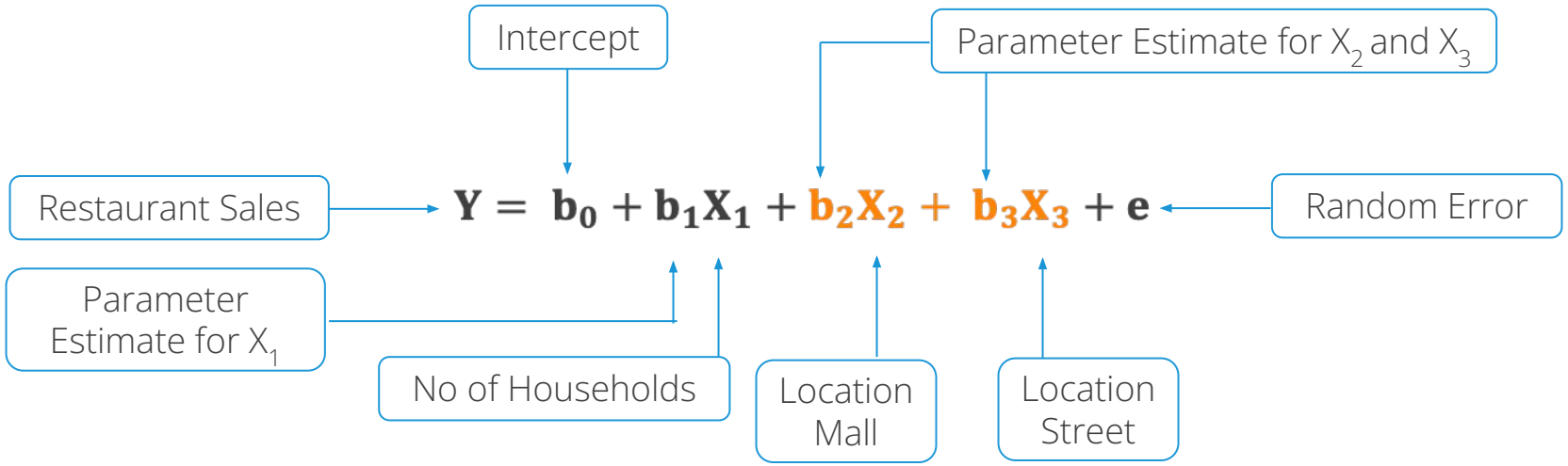
Y : Dependent Variable

$X_1, X_2, \ldots, X_p$ : Independent Variables

$b_0, b_1, \ldots, b_p$ : Parameters of Model

e : Random Error Component

---



# Interpretation of Results

Regression Coefficients of categorical dummy variables are interpreted

**relative to the base category**

- The positive beta coefficient of mall ($b_2$) implies that if the restaurant is located in a mall, the sale amount will be higher than sale amount of restaurant on highway by $b_2$. units.
- If the coefficient is negative $b_2$ then it implies that restaurant located in mall will have lower sales than restaurant on highway by $b_2$ units.
- The same applies to street v/s highway
- Remember, dummy variable inferences are useful only if the variable is significant

# MLR with Categorical Dummy Variables in R

```
#Importing the Data
restaurantsales<-read.csv("RESTAURANT SALES DATA.csv",header=TRUE)

str(restaurantsales)
```

- *str() shows class and levels of variables in the data.*

```
'data.frame':   16 obs. of  4 variables:
 $ RESTAURANT: int  1 2 3 4 5 6 7 8 9 10 ...
 $ NOH       : int  155 93 128 114 158 173 178 215 152 197 ...
 $ LOCATION  : Factor w/ 3 levels "highway","mall",..: 1 1 1 1 1 1 2 2 2 2 ...
 $ SALES     : num  131.3 68.1 115 102.9 131.8 ...
```

```
levels(restaurantsales$LOCATION)

[1] "highway" "mall"    "street"
```

- *levels() to check categorical variable's levels and their order*

```
#Fitting Multiple Linear Regression Model
salesmodel<-lm(SALES~NOH+LOCATION,data=restaurantsales)

summary(salesmodel)
```

*summary() generates a detailed description of the model.*

# MLR with Categorical Dummy Variables in R

#Output

```
Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      2.18923    8.59214   0.255    0.803
NOH              0.83828    0.05618  14.920 4.13e-09 ***
LOCATIONmall    37.05241    5.81407   6.373 3.54e-05 ***
LOCATIONstreet   7.15367    6.73141   1.063    0.309
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 9.398 on 12 degrees of freedom
Multiple R-squared:  0.9701,    Adjusted R-squared:  0.9627
F-statistic:   130 on 3 and 12 DF,  p-value: 2.05e-09
```

*Interpretation:*

- *R orders factor levels alphabetically and takes the first level as the base category by default*
- *NOH and Mall are significant variables*
- *Beta coefficient for mall is 37.05241. This implies that if a restaurant is located in a mall, its sales will be more than the restaurant located on highway by 37.05241.*
- *Street too has a positive coefficient, implying that sales of restaurant located on street will be 7.15367 times higher than highway.*

# Changing the Base Category in R

```
#Changing the Base Category to “mall”
restaurantsales$LOCATION<-relevel(restaurantsales$LOCATION,ref="mall")
```

*relevel() reorders levels of a factor variables.*
*ref= is used to specify changed reference (base) level.*

```
#Fitting Model on Data with Reordered Levels
salesmodel2<-lm(SALES~NOH+LOCATION, data=restaurantsales)

summary(salesmodel2)
```

# Changing the Base Category in R

#Output

```
Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)       39.24164   10.50204   3.737 0.002840 **
NOH                0.83828    0.05618  14.920 4.13e-09 ***
LOCATIONhighway  -37.05241    5.81407  -6.373 3.54e-05 ***
LOCATIONstreet   -29.89874    6.12294  -4.883 0.000377 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 9.398 on 12 degrees of freedom
Multiple R-squared:  0.9701,    Adjusted R-squared:  0.9627
F-statistic:   130 on 3 and 12 DF,  p-value: 2.05e-09
```

*Interpretation:*

- *Mall has now become the base category*
- *Coefficient of highway is just negative of coefficient of mall observed in previous model (Degree of association between mall and highway is the same, changed sign indicates that relativity has reversed)*
- *Note that street is also significant*

# Quick Recap

In this session, we learnt how to handle categorical variables in multiple linear regression by introducing Dummy Variables

| | |
|---|---|
| Number of Dummy Variables | • The number of dummy variables must be one less than the number of levels in the categorical variable |
| Interpretation | • The coefficient attached to the dummy variables must always be interpreted in relation to the base, or reference group—that is, the group that receives the value of zero. The base chosen will depend on the purpose of research at hand |
| Dummy Variables v/s Data Observations | • If a model has several qualitative variables with several classes, introduction of dummy variables can consume a large number of degrees of freedom. Therefore, one should always weigh the number of dummy variables to be introduced against the total number of observations available for analysis |
| Dummy Variables in R | • R automatically assigns dummies to categorical variables in **`lm()`**<br>• Use **`relevel()`** to change the base category for modeling |

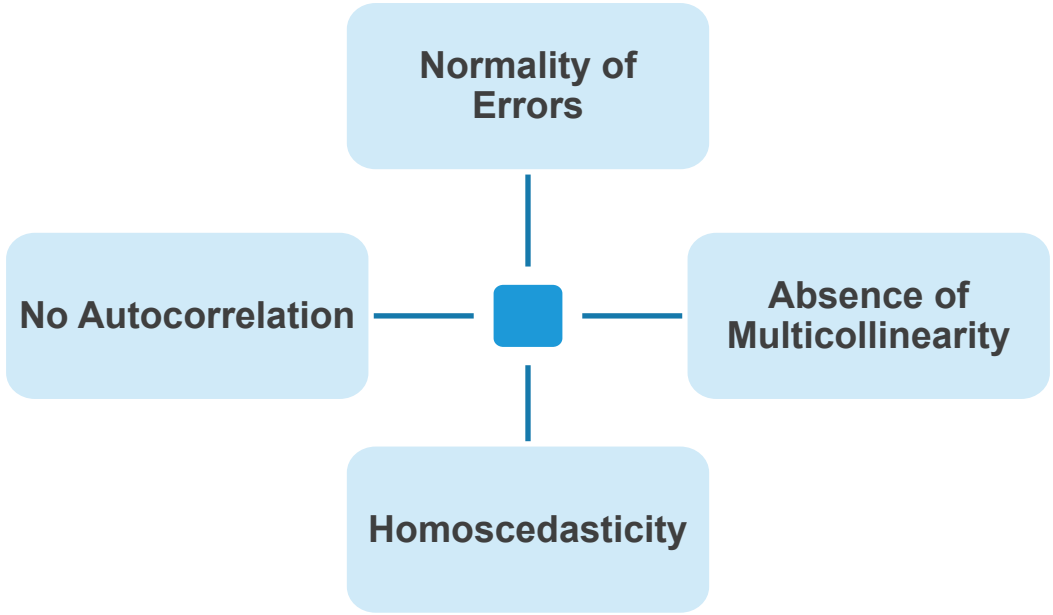
# Multiple Linear Regression

# Multicollinearity Problem

# Contents

# Key Assumptions of Multiple Linear Regression

Multiple Linear Regression **makes four key assumptions**




Violations of these assumptions may result in biased variable relationships, over or under-estimation of parameters (i.e. biased standard errors), and unreliable confidence intervals and significance tests

# Problem of Multicollinearity

Multicollinearity exists if there is strong linear relationship among the independent variables

Multicollinearity has two serious consequences:

1. Highly Unstable Model Parameters

As standard errors of their estimates are inflated

2. Model Fails to Accurately Predict for Out of Sample Data

Therefore, it is important to check for Multicollinearity in regression analysis

# Detecting Multicollinearity Through VIF

VIF (Variance Inflation Factor) Method:

Dependent Variable : Y

Independent variables : X1, X2, X3, X4

| Dependent Variable | Independent Variables | $R^2$ | $1 - R^2$ = Tolerance | VIF = 1/(Tolerance) |
|---|---|---|---|---|
| X1 | X2, X3, X4 | | | |
| X2 | X1, X3, X4 | | | |
| X3 | X1, X2, X4 | | | |
| X4 | X1, X2, X3 | | | |

Any VIF > 5, indicates presence of Multicollinearity

# Detecting Multicollinearity in R

```
#Importing the Data, Fitting Linear Model

perindex<-read.csv("Performance Index.csv", header=TRUE)
jpimodel<-lm(jpi~aptitude+tol+technical+general, data=perindex)

#Variance Inflation Factor
#Install and load package “car”.

install.packages("car")
library(car)

vif(jpimodel)
```

*car stands for Companion to Applied Regression and consists of several useful functions for advance regression analysis.*

*vif() in package car calculates VIFs.*

* Continuing with same dataset “Performance Index”

# Detecting Multicollinearity in R

```
# Output
```

| aptitude | tol | technical | general |
|---|---|---|---|
| 1.179906 | 1.328205 | 2.073907 | 2.024968 |

*Interpretation :*
*All VIFs are less than 5, Multicollinearity is not present.*

# Multicollinearity – Remedial Measures

The problem of Multicollinearity can be solved by different approaches:

- Drop one of the independent variables, which is explained by others
- Use Principal Component Regression in case of severe Multicollinearity
- Use Ridge Regression

* Dropping a variable may not be a good idea if many VIFs are large.
Principal Component Method will be discussed in detail under Data Reduction and Segmentation

# Case Study - Modelling Resale Price of Cars

## Background

- A car garage has old cars for resale. They keep records for different models of cars and their specifications.

## Objective

- To predict the resale price based on the information available about the engine size, horse power, weight and years of use of the cars

## Available Information

- Records -26
- Independent Variables: ENGINE SIZE, HORSE POWER, WEIGHT AND YEARS
- Dependent Variable: RESALE PRICE

# Data Snapshot

ridge regression data

Dependent variable

Independent variables

| MODEL | RESALE PRICE | ENGINE SIZE | HORSE POWER | WEIGHT | YEARS |
|---|---|---|---|---|---|
| Daihatsu Cuore | 3870 | 846 | 32 | 650 | 2.9 |
| Suzuki Swift 1.0 GL | 4163 | 993 | 39 | 790 | 2.9 |
| Fiat Panda Mambo L | 3490 | 899 | 29 | 730 | 3.1 |

Observations

| Columns | Description | Type | Measurement | Possible values |
|---|---|---|---|---|
| MODEL | Model of the car | character | - | - |
| RESALE PRICE | Resale price | numeric | Euro | positive values |
| ENGINE SIZE | Size of the engine | numeric | cc | positive values |
| HORSE POWER | Power of the engine | numeric | kW | positive values |
| WEIGHT | Weight of the car | numeric | kg | positive values |
| YEARS | Number of years in use | numeric | - | positive values |

# Correlation Matrix

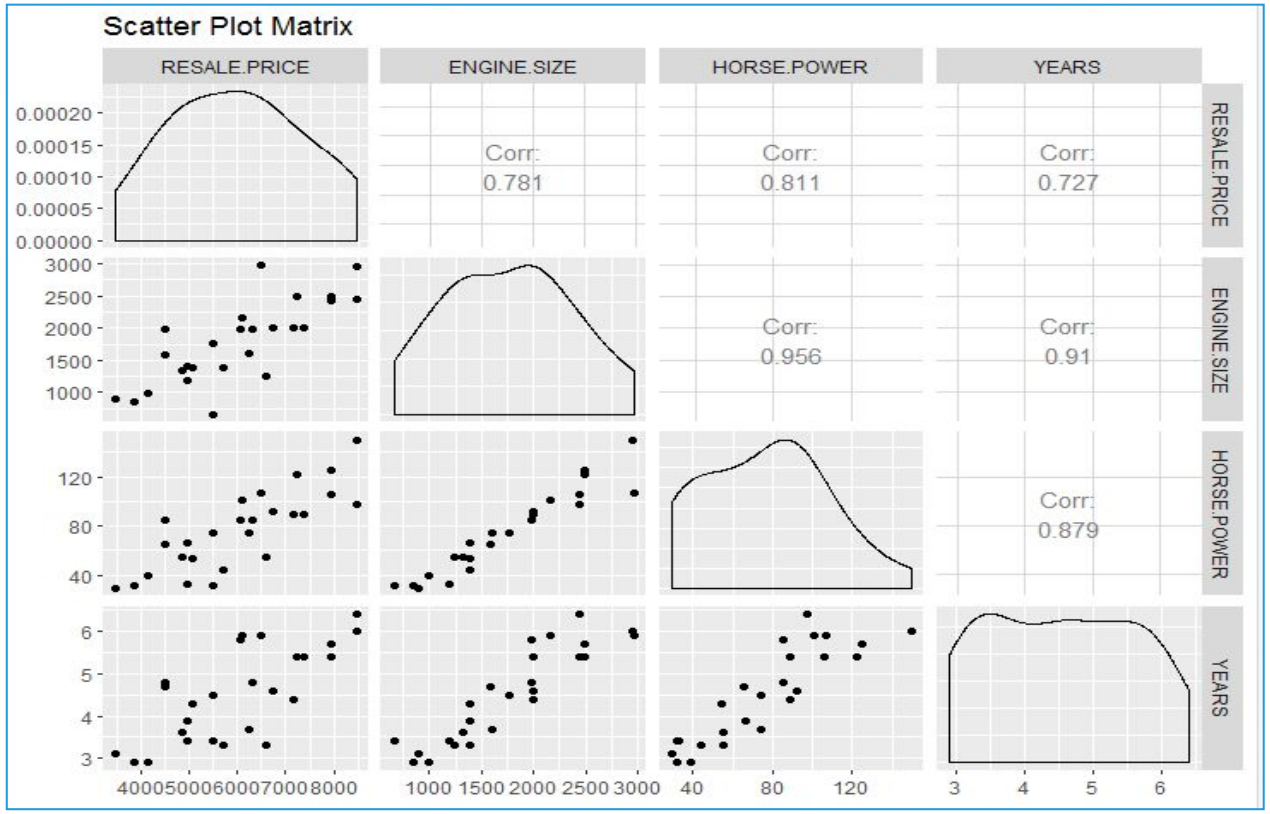
```
# Importing the Data
ridgedata<-read.csv("ridge regression data.csv", header=TRUE)
```

```
# Graphical representation of data
# Install and load package “GGally”
install.packages("GGally")
library(GGally)

ggpairs(ridgedata[,c("RESALE.PRICE","ENGINE.SIZE","HORSE.POWER","YEARS
")],title="Scatter Plot Matrix", columnLabels = c("RESALE.PRICE",
"ENGINE.SIZE","HORSE.POWER","YEARS"))
```

- *ggpairs() in the package GGally is used to plot the scatter plot matrix.*

# Correlation Matrix

```
# Output
```




*Interpretation :*

- *The independent variables have high positive correlation among themselves .*

# Detecting Multicollinearity in R

```
#Fitting Linear Model
model<-lm(RESALE.PRICE ~ ENGINE.SIZE +HORSE.POWER +WEIGHT+ YEARS,
data=ridgedata)

#Variance Inflation Factor
library(car)
vif(model)

# Output
```

| ENGINE.SIZE | HORSE.POWER | WEIGHT | YEARS |
|---|---|---|---|
| 15.759113 | 12.046734 | 9.113045 | 13.978640 |

*Interpretation:*

- *VIF values for all the variables are greater than 5, hence we can conclude that there exist Multicollinearity between the independent variables.*

# Quick Recap

This session explained the **problem of Multicollinearity, along with its consequences and remedial measures:**

| | |
|---|---|
| Multicollinearity Exists | • When independent variables have strong linear relationship |
| Results in | • Unstable model parameters<br>• Inaccurate predictions for out of sample data |
| Indicators | • High pairwise correlation<br>• Significant F value but very few significant t values |
| Checking in R | • Variance Inflation Factor **`vif()`** function in package **car** |
| Remedial Measures | • Drop variables<br>• Use Principal Component Regression<br>• Ridge regression |

# Multiple Linear Regression

## Normality and

## Homoscedasticity Assumptions

# Contents

# Normality and Homoscedasticity

- The  errors in Multiple Linear Regression are assumed to follow Normal Distribution.

- If Normality of Errors is not true then statistical tests and associated P values based on F and t distribution are not reliable.

- **Homoscedasticity** describes a situation in which variance of  error term is same across all values of the independent variables.

- In the absence of Homoscedasticity ( Or presence of Heteroscedasticity) the standard errors of parameter estimates are incorrect.

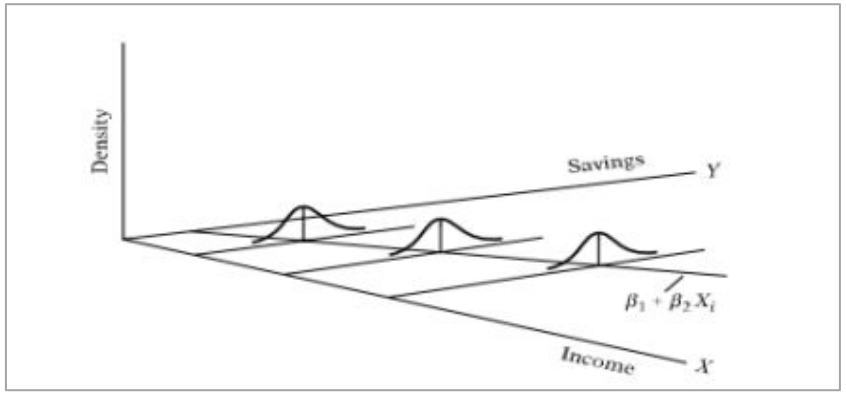
# Assumption of Homoscedasticity

- Variance of error term must be constant across the independent variables (defined by X values )

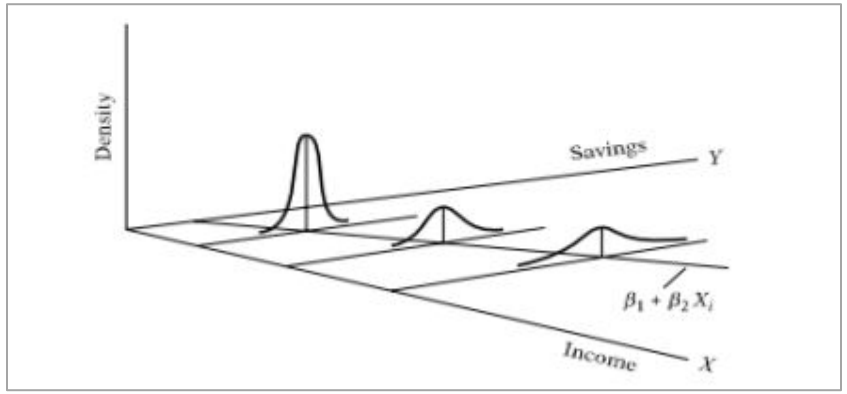
$$V\left(e_i/x_i\right) = \sigma^2 \quad indicates\ homoscedasticity$$

$$V\left(e_i/x_i\right) = \sigma_i^2 \quad indicates\ heteroscedasticity$$

Homoscedastic Errors

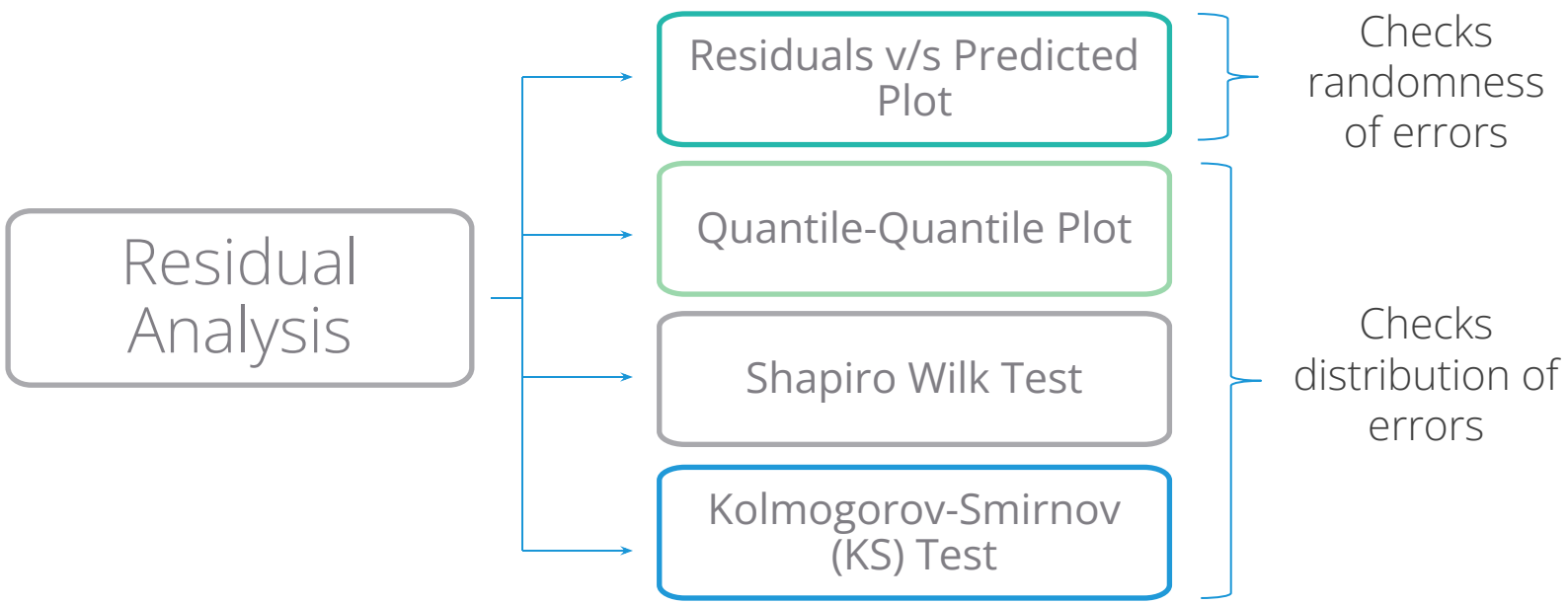



Heteroscedastic Errors



# Residual Analysis

Observed Value – Predicted value = Residual



# Residual Analysis for Performance Index Data

Continuing with the "**Performance Index** " data,

- **Model** job performance index ( **jpi** ) based on aptitude score ( **aptitude** ), test of language ( **tol** ), technical knowledge ( **technical** ) and general information ( **general** )
- Get the fitted values and thus the residuals.
- Analyse the  distribution of residuals

# Residual v/s Predicted Plot in R

```
#Importing the Data, Fitting Linear Model and Calculate Fitted Values
and Residuals
perindex<-read.csv("Performance Index.csv",header=TRUE)
jpimodel<-lm(jpi~aptitude+tol+technical+general, data=perindex)
perindex$pred<-fitted(jpimodel)
perindex$resi<-residuals(jpimodel)
```

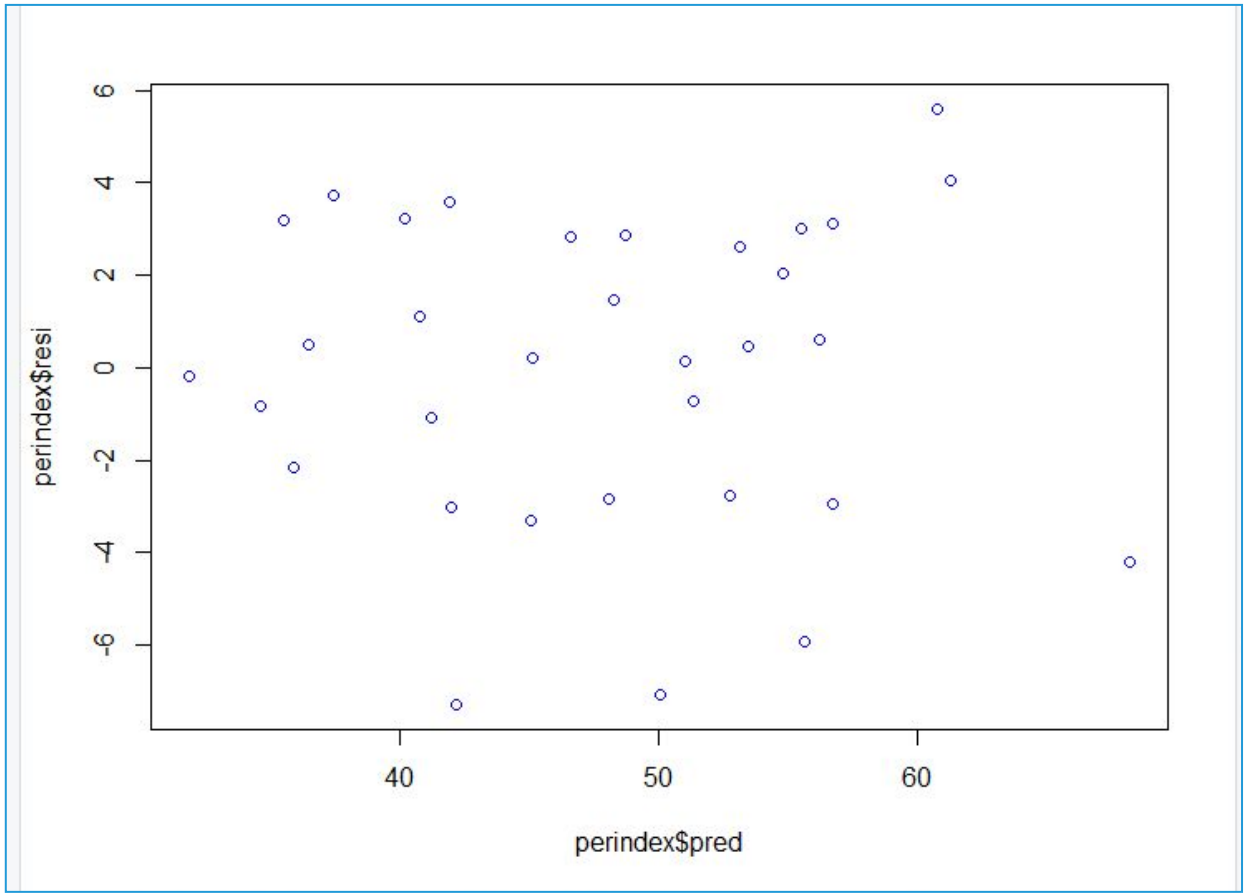
- *lm() fits a linear regression.*
- *fitted() and residuals() fetch fitted values and residuals respectively.*

```
#Residuals v/s Predicted Plot
plot(perindex$pred,perindex$resi,col="blue")
```

*plot() is used to plot predicted values against residuals.*

# Residual v/s Predicted Plot in R

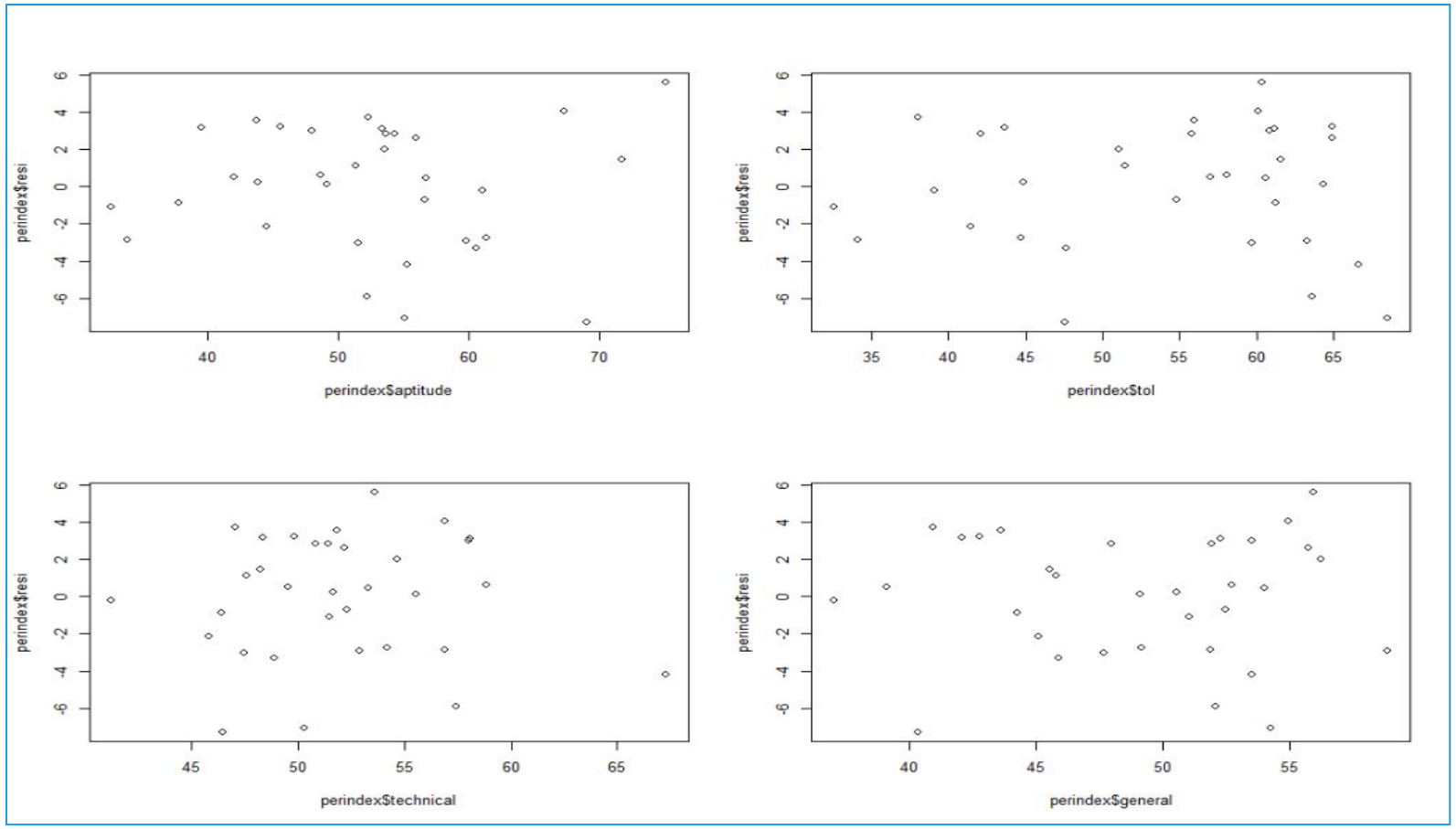
```
# Output
```




*Interpretation:*

- *Residuals in our model are randomly distributed which indicates presence of Homoscedasticity*

# Residual v/s Independent variables Plot in R




*Interpretation:*

- *Residuals in our model are randomly distributed which indicates presence of Homoscedasticity*

# QQ Plot in R

- **The Quantile-Quantile (QQ) Plot** is a powerful graphical tool for assessing normality.
- Quantiles are calculated using sample data and plotted against expected quantiles under Normal distribution.

High Correlation between Sample Quantiles and Theoretical Quantiles → Normality

- If the data are truly sampled from a Gaussian (Normal) distribution, the **QQ plot will be linear.**

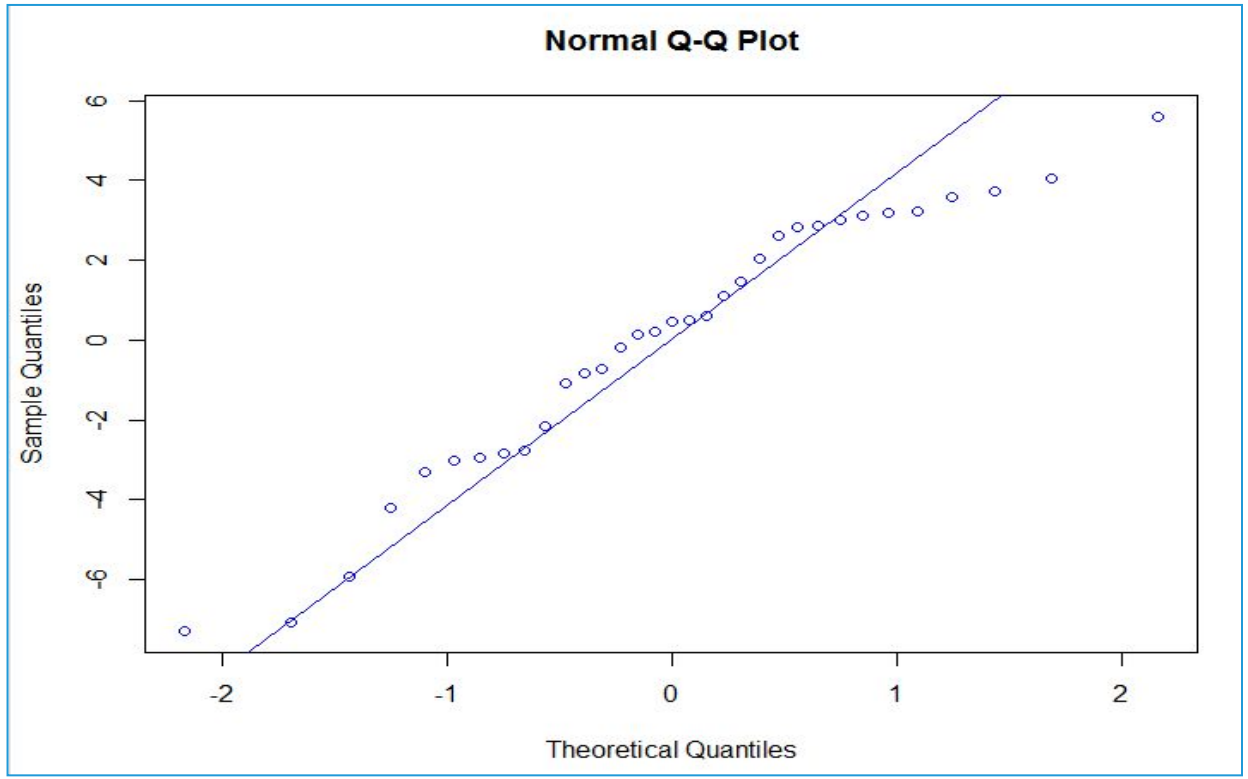
# QQ Plot in R

```
#QQ Plot
qqnorm(perindex$resi,col="blue")

qqline(perindex$resi,col="blue")
```

- *qqnorm() produces a plot with theoretical quantiles on x axis against the sample quantiles on y axis.*
- *Column for which normality is being tested is specified in the first argument.*
- *qqline() adds a line which passes through the first and third quartiles.*

# QQ Plot in R

```
# Output
```




*Interpretation:*

- *Most of these points are close to the line except few values indicating no serious deviation from Normality.*

# Shapiro Wilk Test

| Objective | To **correlate**, sample ordered values with expected Normal scores in order **to test normality of the sample** |
|---|---|

**Null Hypothesis ($H_0$): Sample is drawn from Normal Population**

**Alternate Hypothesis ($H_1$): Not $H_0$**

| Test Statistic | |
|---|---|
| Decision Criteria | Reject the null hypothesis if p-value < 0.05 |

# Shapiro Wilk Test in R

```
# Shapiro Wilk Test
shapiro.test(perindex$resi)
```

*shapiro.test() from basic stats package, returns correlation coefficient w and p-value.*

```
# Output
```

```
        Shapiro-Wilk normality test

data:  perindex$resi
W = 0.94986, p-value = 0.1318
```

*Interpretation:*

- *p-value>0.05, Do not reject $H_0$. Normality can be assumed.*

# Absence of Normality – Remedial Measure

Mathematical Transformation of the dependent variable is used as a remedial measure in case of serious departure from Normality.

Typically Log Transformation is used. However, there is general transformation called as Box Cox Transformation given as :

- Box Cox transformation

$$Y^* = \frac{Y^\lambda - 1}{\lambda} \qquad \lambda \neq 0$$

$$= \log Y \qquad \lambda = 0$$

Where Y is the response variable

- R can automatically detect the optimum λ using **`boxcox()`** in package MASS

# Quick Recap

This session explained in detail **normality of errors**. Here's a quick recap:

| | |
|---|---|
| Normality Assumption | • Error terms should be normally distributed |
| Homoscedasticity | • Errors should have constant variance across X values |
| Residual v/s Predicted Plot | • Ideally should be randomly distributed |
| QQ Plot | • Used to check if errors follow Normal distribution |
| Shapiro Wilk Test | • Test for Normality assessment of errors |
| Box Cox Transformation | • Transforming non normal response to normal |

# Multiple Linear Regression

# Influential Observations

# Contents

# Outliers in Regression Model

- A regression outlier is an observation that has an unusual value of the dependent variable Y for given X value.
  Here X value may not be unusual.

- A regression outlier will have a large residual.

- The data may have an unusual X value — i.e., it is far from the mean of X.

- Regression outlier or unusual X value may not affect overall model. If both are true simultaneously then it is most likely to influence overall model.
  It is important to develop model which is not influenced by one or few observations.

# Influential Observation

- An **influential observation** is an observation whose deletion from the dataset would noticeably change the result of the calculation.

- In regression analysis an influential data point is one whose deletion has a large effect on the parameter estimates or predictions.

# Handling of  Influential Observations

A single (or a few) observation may have a large influence on the results of regression analysis

It is important to develop a model which is not influenced by just few observations

The problem is encountered more commonly in small sample data

# Detecting Influential Observations

Continuing with the "**Performance Index** " data, we **Model** job performance index ( **jpi** ) based on aptitude score ( **aptitude** ), test of language ( **tol** ), technical knowledge ( **technical** ) and general information ( **general** ).

**Use two methods** to identify the influential observations: **"Cook's Distance Method" and "DFBETAs"**.

# Cook's Distance Method

Cook's distance **measures the effect of deleting a given observation.**

Let Di be the Cook's distance for observation i.

$$D_i = \frac{\sum_{j=1}^{n} \left(\hat{Y}_j - \hat{Y}_{j(i)}\right)^2}{p\ MSE}$$

$\hat{Y}_j$ = prediction from the full regression model for observation j

$\hat{Y}_{j(i)}$ = prediction of $j^{th}$ observation from a refitted model after removing $i^{th}$ observation

$MSE$ = mean square error of the regression model

p = number of fitted parameters in the model

Cut off to indicate influential observation,

- Simple operational guideline $D_i > 1$
- Alternative $D_i > 4/n$, where n is the number of observations

* It is recommended to check model performance by excluding highly influential observation

# DFBETAs

DFBETA Statistics

DFBETA **measures the difference** in each parameter estimate with and without a specific observation. There is a DFBETA for each data point and for each parameter estimate.

Large values of DFBETAs

Indicate

**Observations are influential** in estimating a given parameter

Cut off to indicate influential observation,

- general cut off value recommended is 2
- size adjusted cut off is taken to be $2/\sqrt{n}$

# Influential Observations in R

```
#Importing the Data
perindex<-read.csv("Performance Index.csv",header=T)
jpimodel<-lm(jpi~aptitude+tol+technical+general,data=perindex)

#Finding Influential Observations
influ<-influence.measures(jpimodel)
influ
```

- **influence.measures()** produces a class "inf" object
- tabular display showing the DFBETAs for each model variable, DFFITS, covariance ratios, Cook's distances and the diagonal elements of the hat matrix.

# Influential Observations in R

```
# Output
```

| | dfb.1_ | dfb.aptt | dfb.tol | dfb.tchn | dfb.gnrl | dffit | cov.r | cook.d | hat | inf |
|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0.12274 | -1.49e-01 | 0.129300 | 1.11e-01 | -0.23193 | 0.3688 | 1.088 | 2.71e-02 | 0.1056 | |
| 2 | -0.06975 | 1.17e-01 | 0.133588 | 2.98e-02 | -0.09549 | -0.2299 | 1.653 | 1.09e-02 | 0.2914 | * |
| 3 | 0.00730 | -8.66e-03 | 0.004774 | 1.49e-02 | -0.02638 | -0.0473 | 1.255 | 4.63e-04 | 0.0515 | |
| 4 | -0.15696 | 9.47e-02 | -0.173853 | 2.14e-01 | -0.08110 | -0.3002 | 1.152 | 1.82e-02 | 0.1009 | |
| 5 | 0.05386 | -1.09e-02 | 0.008672 | -4.09e-02 | 0.00366 | 0.0778 | 1.248 | 1.25e-03 | 0.0564 | |
| 6 | 0.24456 | -1.68e-01 | -0.058830 | 6.42e-03 | -0.12035 | 0.3513 | 1.153 | 2.48e-02 | 0.1190 | |
| 7 | -0.02188 | -1.73e-02 | 0.012271 | 6.43e-03 | 0.01387 | -0.0382 | 1.633 | 3.02e-04 | 0.2660 | * |
| 8 | -0.16820 | 1.32e-02 | 0.047431 | 1.75e-01 | -0.06910 | 0.2772 | 1.124 | 1.55e-02 | 0.0839 | |
| 9 | -0.00894 | 7.48e-04 | 0.010029 | -1.28e-02 | 0.02059 | 0.0354 | 1.283 | 2.60e-04 | 0.0682 | |
| 10 | 0.11205 | -2.25e-01 | 0.240141 | -1.75e-01 | 0.07832 | -0.3408 | 1.230 | 2.35e-02 | 0.1420 | |
| 11 | -0.18055 | 7.95e-02 | 0.074018 | 1.21e-01 | -0.05749 | -0.2328 | 1.261 | 1.11e-02 | 0.1175 | |
| 12 | -0.34053 | 3.23e-01 | -0.062977 | 1.49e-01 | 0.04455 | 0.4753 | 1.048 | 4.44e-02 | 0.1294 | |
| 13 | -0.00279 | 1.52e-01 | 0.050720 | -2.02e-02 | -0.06082 | 0.2144 | 1.411 | 9.46e-03 | 0.1819 | |
| 14 | 0.03535 | -2.71e-02 | 0.033243 | 1.67e-02 | -0.05554 | 0.0779 | 1.503 | 1.26e-03 | 0.2052 | |
| 15 | -0.06100 | 1.62e-05 | -0.079422 | -3.44e-02 | 0.14787 | 0.2052 | 1.256 | 8.62e-03 | 0.1057 | |
| 16 | -0.02601 | -5.63e-02 | 0.145740 | -1.91e-01 | 0.22252 | 0.3179 | 1.246 | 2.05e-02 | 0.1406 | |
| 17 | 0.00576 | 5.23e-02 | -0.223037 | -4.56e-02 | 0.14885 | 0.3033 | 1.186 | 1.86e-02 | 0.1132 | |
| 18 | -0.47081 | 7.16e-01 | -0.106108 | -1.02e-01 | 0.32416 | 0.9688 | 0.836 | 1.73e-01 | 0.2138 | |
| 19 | -0.00256 | -4.07e-03 | 0.008451 | 5.08e-03 | -0.00541 | 0.0141 | 1.324 | 4.13e-05 | 0.0941 | |
| 20 | -0.05213 | -1.83e-01 | 0.123094 | 7.18e-05 | 0.05859 | -0.2879 | 1.101 | 1.66e-02 | 0.0813 | |

**Interpretation:**

Higher the cook's distance, more is the influence of observation on the model.

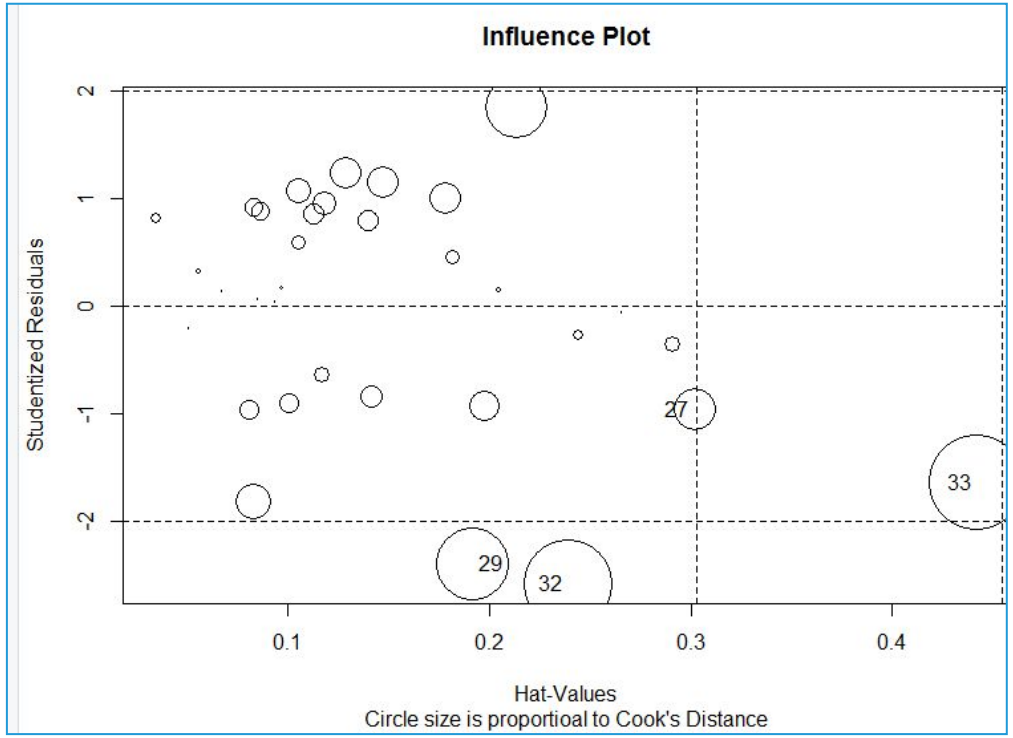
# Influential Plot in R

```
#Influence Plot

install.packages("car")
library(car)

influencePlot(jpimodel,
              id.method="identify",
              main="Influence Plot",
              sub="Circle size is proportioal to Cook's Distance")
```

**influencePlot()** creates a “bubble” plot of Studentized residuals by hat values, with the areas of the circles representing the observations proportional to Cook’s distances.
**id.method="**identify**"** enables interactive point identification.
**main =** Title for plot
**sub =** X axis label

# Influence Plot in R

```
# Output
```




**Interpretation:**
The data points 27, 29, 32, 33 are detected as influential observations.

# Quick Recap

In this session, we learnt what are **influential observations in regression analysis:**

| | |
|---|---|
| Influential Observations | • Having a few observations influence the results of regression analysis is not desirable |
| How to Calculate Influential Observations | • Such influential observations can be calculated via two most widely used methods. Cook's Distance and DFBetas |
| Cook's Distance | • Cook's distance measures the effect of deleting a given observation. |
| DFBetas | • DFBETA measures the difference in each parameter estimate with and without the influential point. |
| Influential Observations in R | • **`influence.measures()`** produces object giving influential observations by different measures<br>• **`influencePlot()`** creates a "bubble" plot of Studentized residuals by hat values, with the areas of the circles representing the observations proportional to Cook's distances |

# Multiple Linear Regression

# Cross Validation - I

# Contents

# Cross Validation in Predictive Modeling

Cross Validation is a
process of evaluating a model on
'Out of Sample' data

- **Model performance measures** such as R-squared or Root Mean Squared Error (RMSE) **tend to be optimistic on 'In Sample Data'**

- **Model performance on Out of sample data** gives a more realistic picture of model performance.

Cross validation is important because although a model is built on historical data, ultimately it is to be used on future data. However good the model, if it fails on out of sample data then it defeats the purpose of predictive modelling.

# Cross Validation in Predictive Modeling

There are different approaches to cross validation. The five most significant are:

- Hold-Out Validation
- K-Fold Cross Validation
- Repeated K-Fold Cross Validation
- Leave-One-Out Cross Validation (LOOCV)
- Resampling Validation Method (Bootstrap Method)

# Introduction To Package Caret in R

- The **caret** package (short for **C**lassification **A**nd **Re**gression **T**raining) is a **set of functions that attempt to streamline the process for creating predictive models**
- The package contains tools for:
  - Data splitting
  - Pre-processing
  - Feature selection
  - Model tuning using re-sampling
  - Variable importance estimation

# Case Study – Modelling Motor Insurance Claims

## Background

- A car insurance company collects range of information from its customers at the time of buying and claiming insurance. The company wishes to check if any of this information can be used to model and predict claim amount

## Objective

- To model motor insurance claim amount based on vehicle related information collected at the time of registering and claiming insurance

## Available Information

- Sample size is 1000
- Independent Variables: Vehicle Information – **Vehicle Age, Engine Capacity, Length and Weight of the Vehicle**
- Dependent Variable: **Claim Amount**

# Data Snapshot

Motor_Claims

Independent variables

Dependent variable

Observations

| vehage | CC | Length | Weight | claimamt |
|---|---|---|---|---|
| 4 | 1495 | 4250 | 1023 | 72000 |
| 2 | 1061 | 3495 | 875 | 72000 |
| 2 | 1405 | 3675 | 980 | 50400 |
| 7 | 1298 | 4090 | 930 | 39960 |
| 2 | 1495 | 4250 | 1023 | 106800 |
| 1 | 1086 | 3565 | 854 | 69592.8 |

| Columns | Description | Type | Measurement | Possible values |
|---|---|---|---|---|
| vehage | Age of the vehicle at the time of claim | integer | Years | positive values |
| CC | Engine capacity | numeric | cc | positive values |
| Length | Length of the vehicle | numeric | mm | positive values |
| Weight | Weight of the vehicle | numeric | kg | positive values |
| claimamt | Claim amount | numeric | INR | positive values |

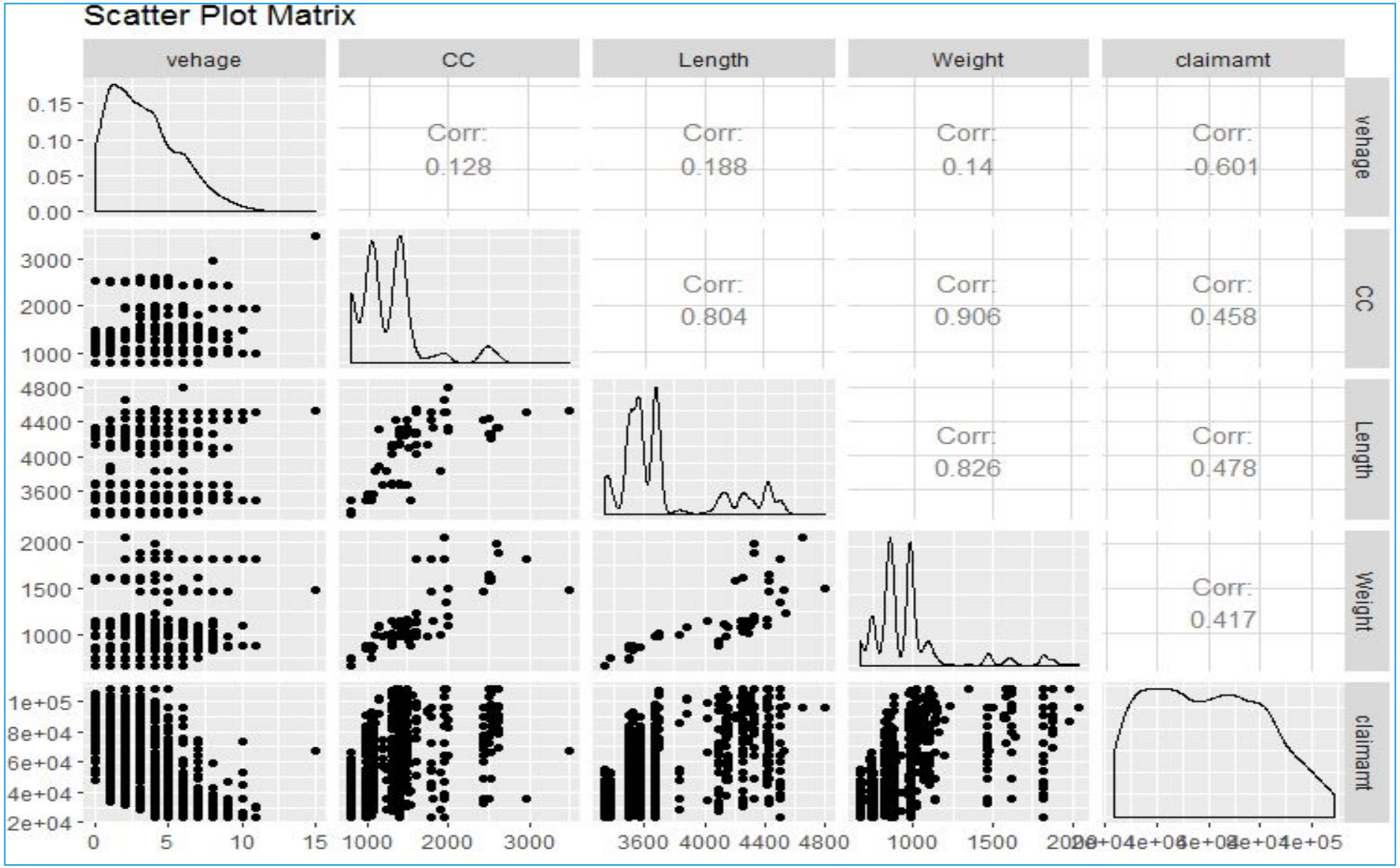
# Data Visualization

```
#Importing the Data
motor<-read.csv("Motor_Claims.csv",header=TRUE)

# Install package “GGally”, if not installed
previously
# plotting correlation matrix
library(GGally)

ggpairs(motor[,c("vehage","CC","Length","Weight","claimamt")],
title="Scatter Plot Matrix" ,
columnLabels = c("vehage","CC","Length","Weight","claimamt"))
```

*Using ggpairs in library GGally to get a scatter plot of the variables in the data set*

# Scatter Plot

```
# Output
```




*Interpretation : Correlation between some of the independent variables are high suggesting a chance of multicollinearity.*

# Detecting Multicollinearity

```
# Linear regression model

motor_model<-lm(claimamt~Length+CC+vehage+Weight, data=motor)
summary(motor_model)
```

```
# Output

Residuals:
   Min     1Q Median     3Q    Max
-45577  -8007     39   7852  40561

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -54765.128   5569.375  -9.833  < 2e-16 ***
Length         35.461      1.990  17.824  < 2e-16 ***
CC             15.413      2.114   7.292 6.23e-13 ***
vehage       -6637.213    154.098 -43.071  < 2e-16 ***
Weight         -16.255      3.678  -4.420 1.10e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 11360 on 995 degrees of freedom
Multiple R-squared:  0.7379,    Adjusted R-squared:  0.7368
F-statistic: 700.3 on 4 and 995 DF,  p-value: < 2.2e-16
```

*Interpretation:*
*All the independent variables in the model are significant.*

# Detecting Multicollinearity

```
# Obtaining vif

library(car)
vif(motor_model)
```

*vif in library car gives the VIFs of the independent variables in the regression model.*

```
# Output showing VIF

   Length        CC   vehage   Weight
3.396171 5.881428 1.038357 6.552811
```

*Interpretation:*
*CC and Weight have VIF >5*

# Re- Modelling

```
# New model
motor_model1<-lm(claimamt~Length+CC+vehage,data=motor)
summary(motor_model1)
```

*New model after removing weight to adjust for multicollinearity*

```
# Output of the new model
```

```
Residuals:
   Min     1Q Median     3Q    Max
-47069  -7673    -14   7783  40447

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -49195.196   5475.151  -8.985  < 2e-16 ***
Length          32.065      1.852  17.312  < 2e-16 ***
CC               8.689      1.481   5.867 6.02e-09 ***
vehage       -6638.076    155.525 -42.682  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 11470 on 996 degrees of freedom
Multiple R-squared:  0.7327,    Adjusted R-squared:  0.7319
F-statistic: 910.3 on 3 and 996 DF,  p-value: < 2.2e-16
```

*Interpretation:*
*All the independent variables in the model are significant.*

* **Dropping one independent variable is one of the remedial measures to adjust for multicollinearity(when not many variables are multicollinear). As weight had the maximum VIF value, it is excluded from the model to adjust for multicollinearity.**

# VIF of New Model

```
# VIF

vif(motor_model1)
```

*Getting VIFs of the independent variables in the new model*

```
# VIFs of variables in the new model
```

```
  Length       CC   vehage
2.889718 2.833931 1.038355
```

*Interpretation:*
*All VIF s are <5.*

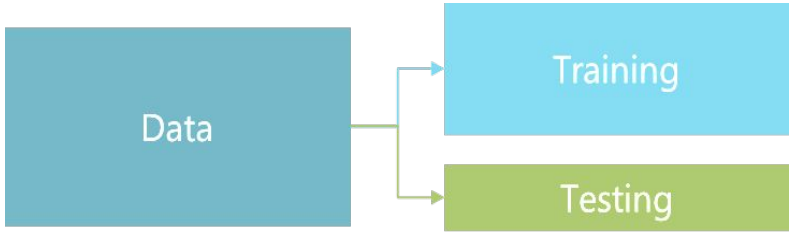
# RMSE of the Model

```
# RMSE of the model

motor$res<-residuals(motor_model1)
RMSEmotor<-sqrt(mean(motor$res**2))
RMSEmotor

# Output

[1] 11444.51
```

*Interpretation :*
*RMSE for the model.*

* **RMSE of this model will be used later to measure the performance of the model on cross validation**

# Hold-Out Validation




In Hold-Out validation method, available data is split into two non-overlapped parts: 'Training Data' and 'Testing Data'

- The model is,
  - Developed using training data
  - Evaluated using testing data

- Training data should have more sample size. Typically 70%-80% data is used for model development

* **Note : This ppt is in continuation of previous ppt. So the data considered is the same, "Motor_Claims" data.**

# Hold Out Validation in R

```
# Creation of Datasets for Validation
# Install & load package “caret”
motor <- read.csv("Motor_Claims.csv", header=TRUE)

install.packages("caret")
library(caret)

Index <- createDataPartition(motor$claimamt,p=0.8,list=FALSE)

head(index)
dim(index)
```

- *createDataPartition() generates list of observation numbers to be included in training data.*
- *p= is the percentage of data that goes into training data.*
- *list= specifies if results should be in a list format or matrix.*

```
# Output of first 6 rows
of index
```

```
     Resample1
[1,]         1
[2,]         2
[3,]         3
[4,]         5
[5,]         6
[6,]         7
```

```
# Output dim(index)
```

```
[1] 800   1
```

**Note : While splitting data, observations are selected randomly, so the output will vary.**

# Hold Out Validation in R

```
traindata<-motor[index,]
testdata<-motor[-index,]

dim(traindata)
dim(testdata)
```

*training and testing data sets for the Motor Insurance Data*

```
# Output

[1] 800   5
```

*Dimension of training set*

```
[1] 200   5
```

*Dimension of testing set*

# Hold Out Validation in R

```
# RMSE of training data
motor_trn_model<-lm(claimamt~Length+CC+vehage,data=traindata)

traindata$res<-residuals(motor_trn_model)
head(traindata)

RMSEtrain<-sqrt(mean(traindata$res**2))
RMSEtrain
```

```
# RMSE for testing data
testdata$pred<-predict(motor_trn_model,testdata)

testdata$res<-(testdata$claimamt-testdata$pred)

RMSEtest<-sqrt(mean(testdata$res**2))
RMSEtest
```

* For testing data, since we work with predictors, we calculate residuals as observed values minus the predicted values

# Hold Out Validation in R

```
# Output
```

| | vehage | CC | Length | Weight | claimamt | res |
|---|---|---|---|---|---|---|
| 1 | 4 | 1495 | 4250 | 1023 | 72000.0 | -1603.964 |
| 2 | 2 | 1061 | 3495 | 875 | 72000.0 | 12821.441 |
| 3 | 2 | 1405 | 3675 | 980 | 50400.0 | -17651.055 |
| 5 | 2 | 1495 | 4250 | 1023 | 106800.0 | 19996.145 |
| 6 | 1 | 1086 | 3565 | 854 | 69592.8 | 1397.857 |
| 7 | 4 | 796 | 3495 | 740 | 38400.0 | -5059.737 |

*res column gives the residual for the training set data*

```
[1] 11444.51
```

*RMSE for the original data with all data points*

```
[1] 11345.8
```

*RMSE for the training data*

```
[1] 11868.11
```

*RMSE for testing data*

*Interpretations :*
*Comparing RMSE of training and testing data shows not much difference between the two and also are in line with the RMSE of the original model. Thus we can say that the model is stable.*

* There is no thumb rule for comparison of model performance. The only criterion is performance measure for training and testing data should not be too diverse

# Quick Recap

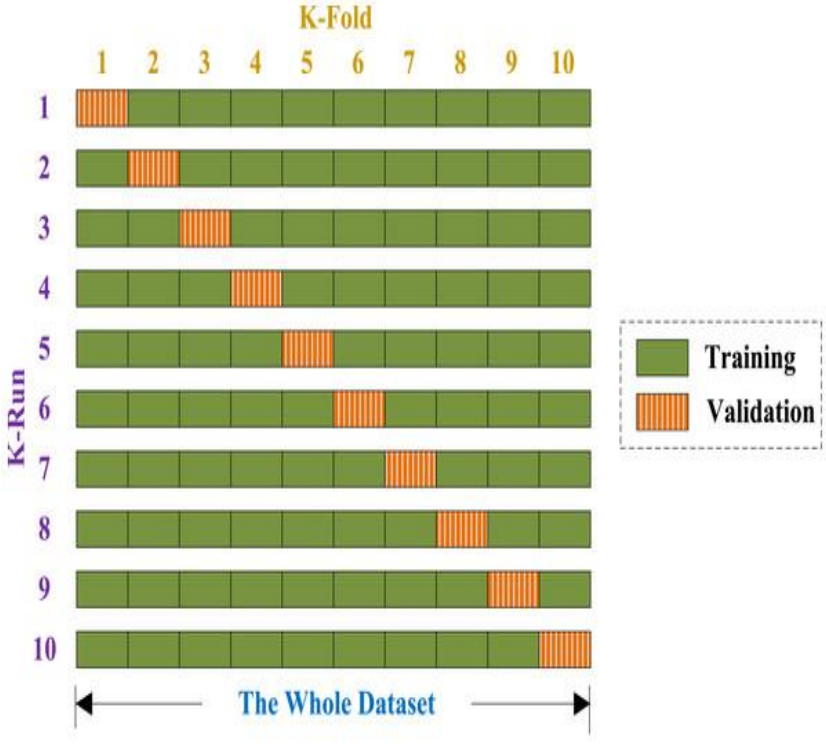
| | |
|---|---|
| Cross Validation – Meaning and Need | • Process of evaluating the model on 'Out of Sample' data<br>• Important because although a model is built on historical data, ultimately it is to be used on future data |

# Multiple Linear Regression

# Cross Validation - II

# Contents

# K-Fold Cross Validation




- In k-fold cross-validation the data is first partitioned into k equally (or nearly equally) sized segments or folds
- Then k iterations of training and testing are performed such that each time one fold is kept aside for testing and model is developed using k-1 folds

- Model performance measure is aggregate measure based on above iterations

# K-Fold Cross Validation in R

```
#Creating ‘k’ Folds
kfolds<-trainControl(method="cv",number=4)
```

- *trainControl() controls the computational nuances of the train function.*
- *method=“cv” tells R to use Cross Validation method.*
- *number= specifies the number of folds.*

```
#Testing
model<-
train(claimamt~vehage+CC+Length,data=motor,method="lm",trControl=k
folds)
model
```

- *train() fits predictive models over different tuning parameters.*
- *It performs a number of classification and regression routines, fits each model and calculates a resampling based performance measure.*
- *method=lm fits a linear regression*
- *trControl= specifies the train function.*

# K-Fold Cross Validation in R

```
# Output
```

```
Linear Regression

1000 samples
   3 predictor

No pre-processing
Resampling: Cross-Validated (4 fold)
Summary of sample sizes: 751, 749, 750, 750
Resampling results:

  RMSE      Rsquared   MAE
  11445.19  0.7319599  9004.326
```

*Interpretation :*

- *$R^2$ of the original model is 73.19%*
- *RMSE for the original is model is 11444.51*
- *Comparing the RMSE values, we can say that the model is stable*

# Repeated K-Fold Cross Validation

- As the name suggests, repeated k-fold cross validation technique **undertakes cross validation and repeats the process m-number of times**
- This ensures that more robust measure of model performance is generated
- **K-fold** is repeated m times with different randomization in each repetition

For instance,

- Five repeats of 10-fold cross validation will generate 50 total resamples.
- These results are again averaged to produce a single estimate
- This is not the same as 50-fold cross validation

# Repeated K-Fold Cross Validation in R

```
#Creating ‘k’ Folds and ‘m’ repeats

kfolds<-trainControl(method="repeatedcv",number=4,repeats=5)
```

- *trainControl() control the computational nuances of the train function.*
- *method=“repeatedcv” tells R to use Repeated Cross Validation method.*
- *number= specifies the number of folds.*
- *repeats= specifies the number of repeats.*

```
#Testing

model<-
train(claimamt~vehage+CC+Length,data=motor,method="lm",trControl=k
folds)

model
```

# Repeated K-Fold Cross Validation in R

```
# Output
```

```
Linear Regression

1000 samples
   3 predictor

No pre-processing
Resampling: Cross-Validated (4 fold, repeated 5 times)
Summary of sample sizes: 749, 749, 751, 751, 749, 751, ...
Resampling results:

  RMSE      Rsquared   MAE
  11527.31  0.7305732  9043.351
```

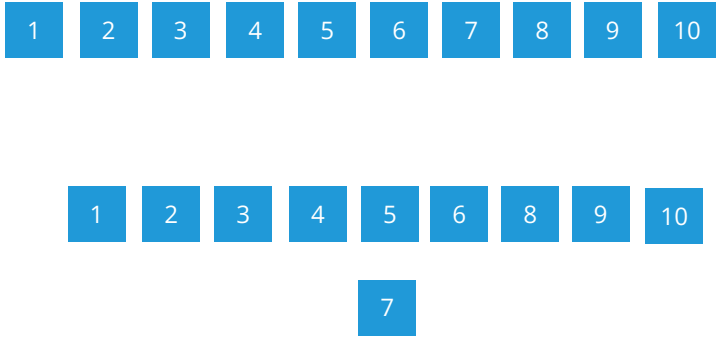
*Interpretation :*

- *$R^2$ of the original model is 73.19%*
- *RMSE for the original is 11444.51*
- *RMSE values of the cross validated model indicates stability.*

# Leave One Out Cross Validation (LOOCV)

- **LOOCV** is a special case of k-fold cross validation where **k equals the sample size (n)**
- Each time one observation is kept aside and the model is developed on the remaining data.
- The left out observation is predicted using the model.
- This process is repeated n times
- RMSE is computed based on these predicted residuals

- Sample size is 10 and one observations (say 7) is chosen to be kept aside
- The model is developed on the new sample with n=9 and observation 7 is predicted



# LOOCV in R

```
#Creating ‘k’ Folds
kfolds<-trainControl(method="LOOCV")
```

- *trainControl() control the computational nuances of the train function.*
- *method=“LOOCV” tells R to use Leave One Out Cross Validation process.*
- *No other arguments need to be included in the command.*

```
#Testing
model<-
train(claimamt~vehage+CC+Length,data=motor,method="lm",trControl=k
folds)
model
```

# LOOCV in R

```
# Output
```

```
Linear Regression

1000 samples
   3 predictor

No pre-processing
Resampling: Leave-One-Out Cross-Validation
Summary of sample sizes: 999, 999, 999, 999, 999, 999, ...
Resampling results:

  RMSE      Rsquared   MAE
  11515.85  0.7294088  9039.582
```

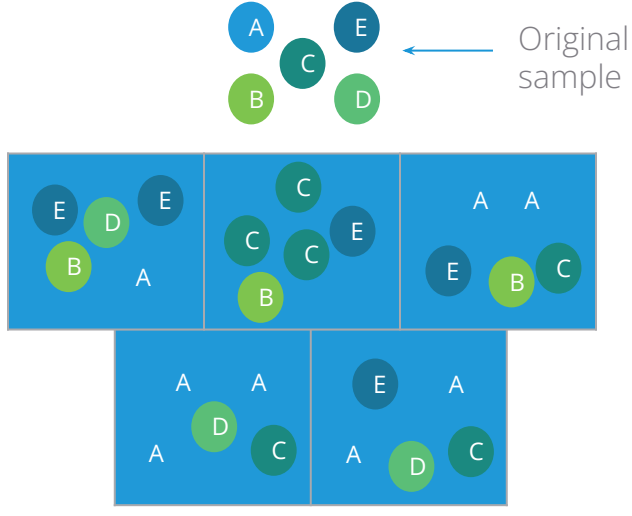
*Interpretation :*

- *$R^2$ of the original model is 73.19%*
- *Not much difference in RMSE values compared to that of the original model*
- *Thus model can be considered stable*

# Resampling Validation (Bootstrapping)

- Bootstrapping is a technique by which a **random sample with replacement is repeatedly drawn from the original sample and the size of the random sample is the same as original** (Therefore having some observations appear more than once)
- Bootstrapping essentially allows us to simulate repeated statistical experiments
- Generally, 100-500 bootstrap samples are considered adequate for evaluating precision of a model

- Suppose our sample size is 5
- Number of bootstrap samples is 5
- 5 samples of size 5 are drawn
- Model is fit on each of the samples and average of all results is considered to measure model performance



# Bootstrapping in R

```
#Creating ‘k’ Folds

kfolds<-trainControl(method="boot")
```

- *trainControl() control the computational nuances of the train function.*
- *method=“boot” tells R to use bootstrapping approach to resampling.*

```
#Testing

model<-
train(claimamt~vehage+CC+Length,data=motor,method="lm",trControl=k
folds)
model
```

# Bootstrapping in R

```
# Output
```

```
Linear Regression

1000 samples
   3 predictor

No pre-processing
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 1000, 1000, 1000, 1000, 1000, 1000, ...
Resampling results:

  RMSE   Rsquared   MAE
  11400  0.7354236  8967.069
```

*Interpretation :*

- *$R^2$ of the original model is 73.19%*
- *RMSE for the original is 11444.51*
- *Comparing the RMSE values of the cross validated model, we can say model is stable.*

# Quick recap

This session illustrated five most important **model validation techniques:**

| | |
|---|---|
| K-Fold Cross Validation | • Data is first partitioned into **k** equally (or nearly equally) sized segments or folds<br>• Then k iterations of training and testing are performed such that each time one fold is kept aside for testing and model is developed using **k-1** folds |
| Repeated K-Fold Cross Validation | • This is an extension of k-fold method wherein the process is repeated **m** number of times |

# Quick Recap

| | |
|---|---|
| Leave One Out Cross Validation (LOOCV) | • Special case of k-fold cross validation where **k** equals the sample size (n), observation number **i** is kept aside and the model is developed on remaining data after which error is calculated<br>• This process is repeated **n** times, for all i's<br>• RMSE is computed based on these predicted residuals |
| Resampling Cross Validation (Bootstrapping) | • Random sample with replacement is repeatedly drawn from the original sample and the size of the random sample is same as original<br>• Model is fit on each sample and average results are considered for measuring model validity |
| Validation in R | • Package **caret** has all functions needed to carry out different types of validation<br>• **`createDataPartition()`, `trainControl()`** and **`train()`** are the most important functions<br>• Depending on the validation technique, **`method=`** needs to be specified in **`trainControl()`** |