# R Programming - Regression

Mahesh Divakaran

Amity University, Lucknow

2024-10-07

# Overview

- What is Regression
- Mtcars Dataset
- Data exploration (EDA)
- Visualizations
- Simple Linear Regression
- Multiple Linear Regression
- Logistic Regression
- Odds Ratio Interpretation

Section 1

# What is Regression?

# Definition and Purpose

- **Regression** is a statistical technique used to estimate the relationships between a dependent variable (response) and one or more independent variables (predictors).
- **Purpose**:
  - To understand the strength and direction of the relationship between variables.
  - To predict future outcomes based on the relationships observed.
- **Application**:
  - Widely used in fields like medicine, economics, engineering, and the social sciences for tasks like forecasting, risk assessment, and optimization.
- **Key Concepts**:
  - **Dependent Variable (Y)**: The outcome you're trying to predict or explain.
  - **Independent Variables (X)**: The predictors used to explain or influence Y.
  - **Error Term ($\epsilon$)**: Represents the unexplained variability in Y.

# Types of Regression: Linear and Non-Linear Regression

Regression can be broadly classified into two categories:

- **Linear Regression**: Models a straight-line relationship between the dependent and independent variables.
- **Non-Linear Regression**: Models a more complex, non-linear relationship between variables.

# Types of Linear Regression

- **Linear Regression** assumes that the relationship between the dependent and independent variables is a straight line. Key types include:

## 1. Simple Linear Regression

- **Definition**: Models the relationship between one independent variable (X) and one dependent variable (Y).
- **Formula**: $Y = \beta_0 + \beta_1 X + \epsilon$
- **Application**: Predicting outcomes based on a single predictor (e.g., predicting salary based on years of experience).

## 2. Multiple Linear Regression

- **Definition**: Models the relationship between two or more independent variables and a dependent variable.
- **Formula**: $Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_n X_n + \epsilon$
- **Application**: Predicting outcomes based on multiple predictors (e.g., predicting house prices based on size, location, and number of rooms).

# Types of Non-Linear Regression

- **Non-Linear Regression** models relationships that cannot be captured by a straight line. It is used when the relationship between variables is more complex. Key types include:

## 1. Logistic Regression

- **Definition**: Used when the dependent variable is binary (0/1 or Yes/No).
- **Formula**: $log(p/(1-p)) = \beta_0 + \beta_1 X1 + \ldots + \beta_n Xn$
- **Application**: Predicting probabilities, such as the likelihood of disease presence (yes/no).

## 2. Poisson Regression

- **Definition**: Used when the dependent variable represents count data (number of occurrences).
- **Formula**: $log(\lambda) = \beta_0 + \beta_1 X1 + \ldots + \beta_n Xn$
- **Application**: Modeling the number of events, such as customer complaints or number of accidents.

## 3. Polynomial Regression

- **Definition**: A form of regression that models the relationship as an nth-degree polynomial.
- **Formula**: $Y = \beta_0 + \beta_1 X + \beta_2 X^2 + \ldots + \beta_n X^n + \epsilon$
- **Application**: Capturing non-linear trends (e.g., modeling the growth curve of a population over time).

Section 2

# Introduction to the mtcars Dataset

# Overview of mtcars

- The **mtcars** dataset is built into R and contains data about various models of cars from the 1974 Motor Trend US magazine. - It includes 32 observations (cars) and 11 variables (attributes).

## Variables in the mtcars Dataset

- **mpg**: Miles per gallon (fuel efficiency)
- **cyl**: Number of cylinders in the car
- **disp**: Displacement (in cubic inches)
- **hp**: Horsepower
- **drat**: Rear axle ratio
- **wt**: Weight (in 1000 lbs)
- **qsec**: 1/4 mile time
- **vs**: Engine type (0 = V/S, 1 = straight)
- **am**: Transmission (0 = automatic, 1 = manual)
- **gear**: Number of forward gears
- **carb**: Number of carburetors

# Exploring the mtcars Dataset

- **dim()**: Returns the dimensions of the dataset (number of rows and columns).

```
  dim(mtcars)
#> [1] 32 11
```

- **summary()**: Provides a summary of each variable in the dataset, including statistics like min, max, mean, median, and quartiles.

```
summary(mtcars)
#>       mpg             cyl             disp             hp       
#>  Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0  
#>  1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5  
#>  Median :19.20   Median :6.000   Median :196.3   Median :123.0  
#>  Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7  
#>  3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0  
#>  Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0  
#>       drat             wt             qsec             vs        
#>  Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000  
#>  1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000  
#>  Median :3.695   Median :3.325   Median :17.71   Median :0.0000  
#>  Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375  
#>  3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000  
#>  Max.   :4.930   Max.   :5.424   Max.   :22.90   Max.   :1.0000  
#>        am              gear            carb      
#>  Min.   :0.0000   Min.   :3.000   Min.   :1.000  
#>  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000  
#>  Median :0.0000   Median :4.000   Median :2.000  
#>  Mean   :0.4062   Mean   :3.688   Mean   :2.812  
#>  3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000  
#>  Max.   :1.0000   Max.   :5.000   Max.   :8.000  
```

# Exploring the mtcars Dataset

- **glimpse()**: A function from the`dplyr`package that provides a transposed view of the dataset, showing data types and the first few entries for each variable.

```
library(dplyr)
glimpse(mtcars)
#> Rows: 32
#> Columns: 11
#> $ mpg  <dbl> 21.0, 21.0, 22.8, 21.4, 18.7, 18.1, 14.3, 24.4, 22.8, 19.2, 17.8,~
#> $ cyl  <dbl> 6, 6, 4, 6, 8, 6, 8, 4, 4, 6, 6, 8, 8, 8, 8, 8, 8, 4, 4, 4, 4, 8,~
#> $ disp <dbl> 160.0, 160.0, 108.0, 258.0, 360.0, 225.0, 360.0, 146.7, 140.8, 16~
#> $ hp   <dbl> 110, 110, 93, 110, 175, 105, 245, 62, 95, 123, 123, 180, 180, 180~
#> $ drat <dbl> 3.90, 3.90, 3.85, 3.08, 3.15, 2.76, 3.21, 3.69, 3.92, 3.92, 3.92,~
#> $ wt   <dbl> 2.620, 2.875, 2.320, 3.215, 3.440, 3.460, 3.570, 3.190, 3.150, 3.~
#> $ qsec <dbl> 16.46, 17.02, 18.61, 19.44, 17.02, 20.22, 15.84, 20.00, 22.90, 18~
#> $ vs   <dbl> 0, 0, 1, 1, 0, 1, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0,~
#> $ am   <dbl> 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 0,~
#> $ gear <dbl> 4, 4, 4, 3, 3, 3, 3, 4, 4, 4, 4, 3, 3, 3, 3, 3, 3, 4, 4, 4, 3, 3,~
#> $ carb <dbl> 4, 4, 1, 1, 2, 1, 4, 2, 2, 4, 4, 3, 3, 3, 4, 4, 4, 1, 2, 1, 1, 2,~
```

- **Dimensions**: The output from`dim(mtcars)`indicates there are 32 rows and 11 columns.
- **Summary Statistics**: The`summary(mtcars)`function gives insights into each variable, showing ranges and central tendencies.
- **Glimpse**: Using`glimpse(mtcars)`provides a quick overview of the dataset structure and types of variables.

Section 3

# Exploratory Data Analysis (EDA)

# Purpose of EDA

- To understand the distribution and relationships of variables in the mtcars dataset.
- To identify patterns, trends, and potential outliers.

# 1: Summary Statistics

```
# Summary statistics for mtcars
summary(mtcars)
#>       mpg             cyl             disp             hp
#>  Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0
#>  1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5
#>  Median :19.20   Median :6.000   Median :196.3   Median :123.0
#>  Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7
#>  3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0
#>  Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0
#>       drat             wt             qsec             vs
#>  Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000
#>  1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000
#>  Median :3.695   Median :3.325   Median :17.71   Median :0.0000
#>  Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375
#>  3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000
#>  Max.   :4.930   Max.   :5.424   Max.   :22.90   Max.   :1.0000
#>        am              gear            carb
#>  Min.   :0.0000   Min.   :3.000   Min.   :1.000
#>  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000
#>  Median :0.0000   Median :4.000   Median :2.000
#>  Mean   :0.4062   Mean   :3.688   Mean   :2.812
#>  3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000
#>  Max.   :1.0000   Max.   :5.000   Max.   :8.000
```

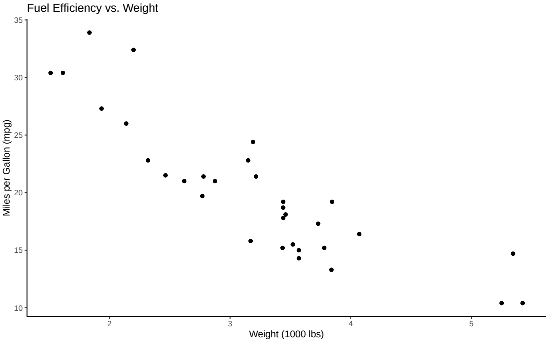
### Insights from Summary

- **mpg**: Ranges from 10.4 to 33.9, with a mean of 20.1.
- **cyl**: Most cars have 4 or 6 cylinders, with a maximum of 8.
- **hp**: Horsepower ranges from 52 to 335, indicating a wide variance in engine power.

# Visualization of Relationships

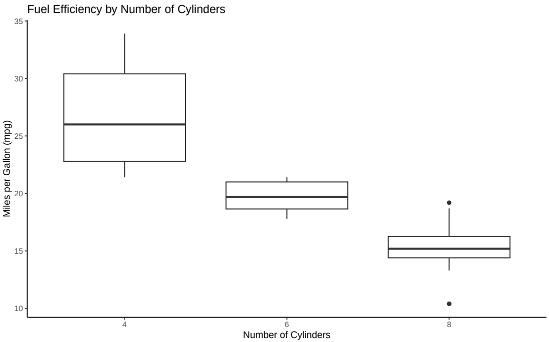
## Scatter Plot: mpg vs. wt

```
library(ggplot2)
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point() +
  labs(title = "Fuel Efficiency vs. Weight",
       x = "Weight (1000 lbs)",
       y = "Miles per Gallon (mpg)")
```



# Box Plot: mpg by Number of Cylinders

```
ggplot(mtcars, aes(x = factor(cyl), y = mpg)) +
  geom_boxplot() +
  labs(title = "Fuel Efficiency by Number of Cylinders",
       x = "Number of Cylinders",
       y = "Miles per Gallon (mpg)")
```




### Insights from Box Plot

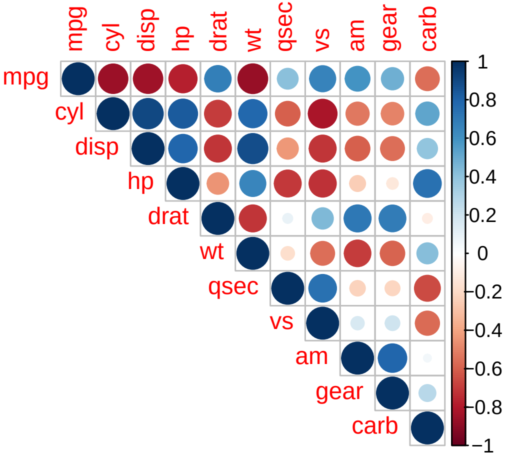
- **Cylinders and mpg**: Cars with 4 cylinders tend to have the highest mpg, while cars with 8 cylinders have the lowest, indicating a trend where more cylinders correspond to less fuel efficiency.

# Correlation Analysis

```
# Correlation matrix
cor_matrix <- cor(mtcars)
print(cor_matrix)
#>             mpg        cyl       disp         hp        drat         wt
#> mpg   1.0000000 -0.8521620 -0.8475514 -0.7761684  0.68117191 -0.8676594
#> cyl  -0.8521620  1.0000000  0.9020329  0.8324475 -0.69993811  0.7824958
#> disp -0.8475514  0.9020329  1.0000000  0.7909486 -0.71021393  0.8879799
#> hp   -0.7761684  0.8324475  0.7909486  1.0000000 -0.44875912  0.6587479
#> drat  0.6811719 -0.6999381 -0.7102139 -0.4487591  1.00000000 -0.7124406
#> wt   -0.8676594  0.7824958  0.8879799  0.6587479 -0.71244065  1.0000000
#> qsec  0.4186840 -0.5912421 -0.4336979 -0.7082234  0.09120476 -0.1747159
#> vs    0.6640389 -0.8108118 -0.7104159 -0.7230967  0.44027846 -0.5549157
#> am    0.5998324 -0.5226070 -0.5912270 -0.2432043  0.71271113 -0.6924953
#> gear  0.4802848 -0.4926866 -0.5555692 -0.1257043  0.69961013 -0.5832870
#> carb -0.5509251  0.5269883  0.3949769  0.7498125 -0.09078980  0.4276059
#>             qsec         vs          am       gear        carb
#> mpg   0.41868403  0.6640389  0.59983243  0.4802848 -0.55092507
#> cyl  -0.59124207 -0.8108118 -0.52260705 -0.4926866  0.52698829
#> disp -0.43369788 -0.7104159 -0.59122704 -0.5555692  0.39497686
#> hp   -0.70822339 -0.7230967 -0.24320426 -0.1257043  0.74981247
#> drat  0.09120476  0.4402785  0.71271113  0.6996101 -0.09078980
#> wt   -0.17471588 -0.5549157 -0.69249526 -0.5832870  0.42760594
#> qsec  1.00000000  0.7445354 -0.22986086 -0.2126822 -0.65624923
#> vs    0.74453544  1.0000000  0.16834512  0.2060233 -0.56960714
#> am   -0.22986086  0.1683451  1.00000000  0.7940588  0.05753435
#> gear -0.21268223  0.2060233  0.79405876  1.0000000  0.27407284
#> carb -0.65624923 -0.5696071  0.05753435  0.2740728  1.00000000
```

# Correlation Analysis

```
# Visualization of correlation matrix
library(corrplot)
corrplot(cor_matrix,  method = 'circle', type = 'upper')
```



# Correlation Analysis

## Insights from Correlation Matrix

- **Strong Correlations**:
  - Positive correlation between **hp** (horsepower) and **mpg** (miles per gallon).
  - Negative correlation between **wt** (weight) and **mpg**.

## Conclusion

- EDA highlights key relationships and patterns within the dataset, informing subsequent modeling approaches.

Section 4

# Linear Regression

# Simple Linear Regression

## Definition

- Simple Linear Regression models the relationship between a dependent variable $Y$ and one independent variable $X$.
- The goal is to predict or explain $Y$ as a linear function of $X$.

## Formula

- The mathematical model for simple linear regression is: $Y = \beta_0 + \beta_1 X + \epsilon$
    - $\beta_0$: Intercept (the value of $Y$ when $X = 0$)
    - $\beta_1$: Slope (the change in $Y$ for a unit increase in $X$)
    - $\epsilon$: Error term (accounts for the variability in $Y$ that is not explained by $X$)

## Application

- Used for tasks such as predicting fuel efficiency based on vehicle weight (e.g., using `mpg` as $Y$ and `wt` as $X$ from the mtcars dataset).

# Regression of mpg on wt

```
# Simple Linear Regression in R
model <- lm(mpg ~ wt, data = mtcars)
model
#>
#> Call:
#> lm(formula = mpg ~ wt, data = mtcars)
#>
#> Coefficients:
#> (Intercept)           wt
#>      37.285       -5.344
```

# Regression of mpg on wt

```
# Simple Linear Regression in R
summary(model)
#>
#> Call:
#> lm(formula = mpg ~ wt, data = mtcars)
#>
#> Residuals:
#>     Min      1Q  Median      3Q     Max
#> -4.5432 -2.3647 -0.1252  1.4096  6.8727
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept)  37.2851     1.8776  19.858  < 2e-16 ***
#> wt           -5.3445     0.5591  -9.559 1.29e-10 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 3.046 on 30 degrees of freedom
#> Multiple R-squared:  0.7528, Adjusted R-squared:  0.7446
#> F-statistic: 91.38 on 1 and 30 DF,  p-value: 1.294e-10
```

# Regression of mpg on wt

```
# Plot
ggplot(mtcars, aes(x = wt, y = mpg)) +
  geom_point() +
  geom_smooth(method = "lm", col = "blue") +
  labs(title = "Simple Linear Regression: mpg vs wt",
       x = "Weight (1000 lbs)",
       y = "Miles per Gallon (mpg)")
```

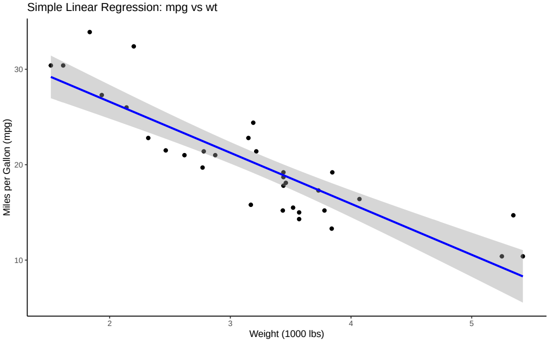
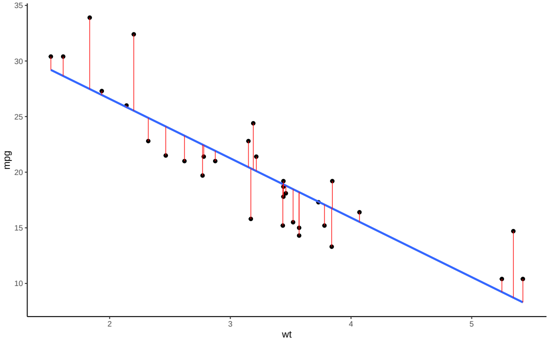



**Figure 1:** Regression of mpg on wt

# Fitted values and residuals

```r
library(broom)
ggplot(augment(model), aes(wt, mpg)) +
  geom_point() +
  stat_smooth(method = lm, se = FALSE) +
  geom_segment(aes(xend = wt, yend = .fitted), color = "red", size = 0.3)
```



Section 5

# Multiple Linear Regression

# Multiple Linear Regression with Two Factors

## Definition

- Multiple Linear Regression models the relationship between a dependent variable $Y$ and two or more independent variables.

## Formula

- The model for multiple linear regression with two independent variables $(X_1$ and $X_2)$ is:

$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \epsilon$

- $ _1 $: The effect of $X_1$ on $Y$, holding $X_2$ constant.
- $ _2 $: The effect of $X_2$ on $Y$, holding $X_1$ constant.

## Application

- We can extend the mpg model by adding an additional factor, such as horsepower (hp) and weight (wt) as predictors of mpg.

# Regression of mpg on wt and hp

```
# Multiple Linear Regression in R
model2 <- lm(mpg ~ wt + hp, data = mtcars)
model2
#>
#> Call:
#> lm(formula = mpg ~ wt + hp, data = mtcars)
#>
#> Coefficients:
#> (Intercept)           wt           hp
#>    37.22727     -3.87783     -0.03177
```

# Regression of mpg on wt and hp

```
# Multiple Linear Regression in R
summary(model2)
#>
#> Call:
#> lm(formula = mpg ~ wt + hp, data = mtcars)
#>
#> Residuals:
#>    Min     1Q Median     3Q    Max
#> -3.941 -1.600 -0.182  1.050  5.854
#>
#> Coefficients:
#>             Estimate Std. Error t value Pr(>|t|)
#> (Intercept) 37.22727    1.59879  23.285  < 2e-16 ***
#> wt          -3.87783    0.63273  -6.129 1.12e-06 ***
#> hp          -0.03177    0.00903  -3.519  0.00145 **
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> Residual standard error: 2.593 on 29 degrees of freedom
#> Multiple R-squared:  0.8268, Adjusted R-squared:  0.8148
#> F-statistic: 69.21 on 2 and 29 DF,  p-value: 9.109e-12
```

# Regression of mpg on wt and hp

## Interpretation

- **Coefficients**:
  - The coefficient for **wt** shows the change in mpg for every unit change in weight, holding horsepower constant.
  - The coefficient for **hp** shows the change in mpg for every unit change in horsepower, holding weight constant.
- **Multiple R-squared**: Indicates how well the model explains the variability in mpg.

Section 6

# Assumptions of Linear Regression

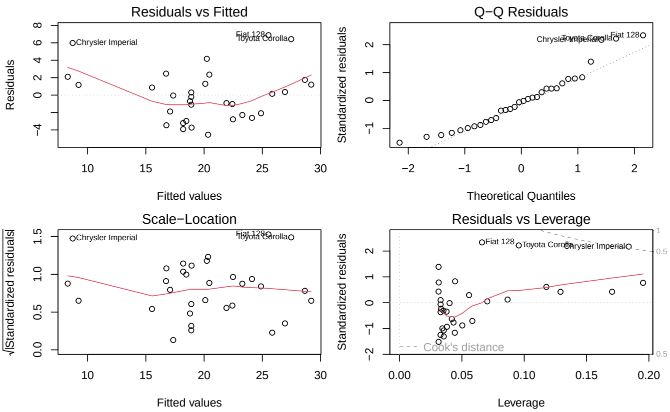
# Key Assumptions

1. **Linearity**:
   - The relationship between the independent and dependent variables must be linear.
   - This can be checked visually using scatter plots or residual plots.
2. **Independence**:
   - Observations should be independent of each other.
   - This is important for the validity of statistical inferences.
3. **Homoscedasticity**:
   - The residuals (errors) should have constant variance across all levels of the independent variables.
   - Can be checked using residual vs. fitted value plots.
4. **Normality of Residuals**:
   - The residuals should follow a normal distribution.
   - This can be checked using a Q-Q plot or a histogram of residuals.
5. **No Multicollinearity (for Multiple Regression)**:
   - Independent variables should not be highly correlated with each other.
   - This can be checked using a correlation matrix or the Variance Inflation Factor (VIF).

# Diagnostic Plots in R

```r
# Diagnostic plots for model2
par(mfrow = c(2, 2), mar = c(4, 4, 2, 1) )
plot(model)
```

# Diagnostic Plots in R



# Interpretation

- Residuals vs Fitted: Checks for linearity and homoscedasticity.
- Q-Q Plot: Checks for normality of residuals.
- Scale-Location Plot: Checks for homoscedasticity.
- Residuals vs Leverage: Identifies influential points.-

# No Autocorrelation

- **Definition**: The residuals should not be correlated with each other.
- This assumption is crucial when working with time series or spatial data, where observations might be dependent over time or space.
- **How to check**:
  - Plot residuals over time (for time series data).
  - Use statistical tests like the **Durbin-Watson Test**.

```
# Durbin-Watson test for autocorrelation
library(lmtest)
dwtest(model2)
#>
#>  Durbin-Watson test
#>
#> data:  model2
#> DW = 1.3624, p-value = 0.02061
#> alternative hypothesis: true autocorrelation is greater than 0
```

# Checking Linearity

- The relationship between the independent variables and the dependent variable should be linear.

**How to check:**

1. **Scatter Plot**: Plot each predictor against the response variable.
2. **Residuals vs Fitted Values**: Check for non-random patterns in the residuals.

# Checking Linearity

```
# Checking Linearity using Residuals vs Fitted Plot
model <- lm(mpg ~ wt + hp, data = mtcars)
plot(model, which = 1)
```

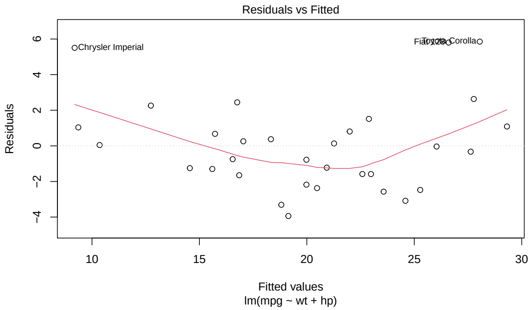



**Figure 2:** Residuals vs Fitted Values Plot

# Checking Linearity

## Interpretation

- If the residuals are randomly scattered around zero with no clear pattern, the linearity assumption is likely satisfied.

# Checking Independence

- Observations should be independent of one another. ### How to check:

1. **Durbin-Watson Test**: Used for detecting autocorrelation in residuals, particularly in time-series data.

```
# Checking Independence using Durbin-Watson Test
library(lmtest)
dwtest(model)
#>
#>  Durbin-Watson test
#>
#> data:  model
#> DW = 1.3624, p-value = 0.02061
#> alternative hypothesis: true autocorrelation is greater than 0
```

## Interpretation

- A Durbin-Watson statistic close to 2 indicates no autocorrelation in the residuals.

# Checking Homoscedasticity

- The residuals should have constant variance across all levels of the independent variables.

## How to check:

1. **Residuals vs Fitted Plot**: Look for a “funnel” shape, indicating heteroscedasticity.
2. **Breusch-Pagan Test**: A formal statistical test for heteroscedasticity.

# Checking Homoscedasticity

```r
# Checking Homoscedasticity using Residuals vs Fitted Plot
plot(model, which = 3)
```

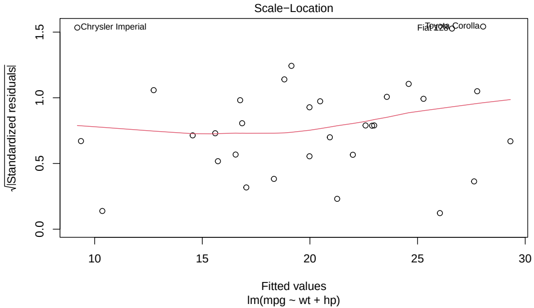



**Figure 3:** Residuals vs Fitted Plot

# Checking Homoscedasticity

```
# Breusch-Pagan test
library(lmtest)
bptest(model)
#>
#>  studentized Breusch-Pagan test
#>
#> data:  model
#> BP = 0.88072, df = 2, p-value = 0.6438
```

## Interpretation

- If the residuals fan out or show a pattern, the assumption of homoscedasticity may be violated.
- The Breusch-Pagan test p-value should be greater than 0.05 to satisfy this assumption.

# Checking Normality of Residuals

- The residuals should follow a normal distribution.

## How to check:

1. **Q-Q Plot**: Visually assess the normality of residuals.
2. **Shapiro-Wilk Test**: A statistical test for normality.

# Checking Normality of Residuals

```r
# Checking Normality using Q-Q Plot
plot(model, which = 2)
```

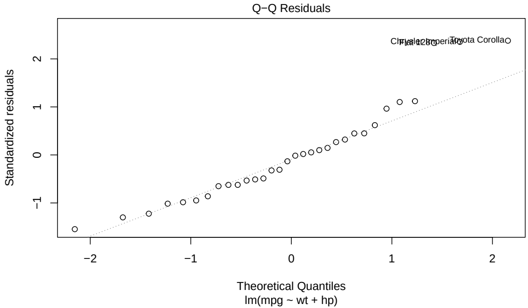



**Figure 4:** Q-Q Plot

# Checking Normality of Residuals

```
# Shapiro-Wilk test
shapiro.test(residuals(model))
#>
#>  Shapiro-Wilk normality test
#>
#> data:  residuals(model)
#> W = 0.92792, p-value = 0.03427
```

## Interpretation

- If the points in the Q-Q plot roughly follow a straight line, the residuals are approximately normally distributed.
- A p-value $> 0.05$ in the Shapiro-Wilk test suggests the residuals are normally distributed.

# Checking for Multicollinearity

- Independent variables should not be highly correlated with each other.

### How to check:

1. **Variance Inflation Factor (VIF)**: Detects the presence of multicollinearity.
2. **Correlation Matrix**: Visually inspect correlations between variables.

```
# Checking Multicollinearity using VIF
library(car)
vif(model)
#>       wt       hp
#> 1.766625 1.766625
```

# Checking for Multicollinearity

```
# Correlation matrix of predictors
cor(mtcars[, c("wt", "hp", "mpg")])
#>             wt         hp        mpg
#> wt   1.0000000  0.6587479 -0.8676594
#> hp   0.6587479  1.0000000 -0.7761684
#> mpg -0.8676594 -0.7761684  1.0000000
```

## Interpretation

- VIF values greater than 2 indicate problematic multicollinearity.
- High correlations between predictors in the correlation matrix suggest multicollinearity.

Section 7

# Logistic Regression

# Logistic Regression

- Logistic regression is used when the dependent variable is categorical (binary or dichotomous).
- It models the probability that a given outcome occurs, based on one or more independent variables.

## Formula

- The logistic regression model is:

$$\log\left(\frac{p}{1-p}\right) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_n X_n$$

  - Where p is the probability of the outcome of interest.

## Example Application

- We'll model whether a car has high mpg (above 20) based on weight (`wt`) and horsepower (`hp`).

# Logistic Regression Example (mtcars)

## Binary Outcome: High vs. Low mpg

Logistic regression is used when the dependent variable is binary (e.g., 0/1 or Yes/No). Since mtcars doesn't contain a direct binary outcome variable, we can create one for demonstration purposes, such as whether a car has high vs. low miles-per-gallon (mpg $>$ 20 or not).

```r
# Create binary outcome for mpg
mtcars$mpg_high <- ifelse(mtcars$mpg > 20, 1, 0)
```

```r
# Logistic regression model
log_model <- glm(mpg_high ~ wt + hp, data = mtcars, family = binomial)
log_model
#>
#> Call:  glm(formula = mpg_high ~ wt + hp, family = binomial, data = mtcars)
#>
#> Coefficients:
#> (Intercept)           wt           hp
#>     894.228     -202.865       -2.021
#>
#> Degrees of Freedom: 31 Total (i.e. Null);  29 Residual
#> Null Deviance:       43.86
#> Residual Deviance: 1.116e-08     AIC: 6
```

# Logistic Regression Example (mtcars)

```
summary(log_model)
#>
#> Call:
#> glm(formula = mpg_high ~ wt + hp, family = binomial, data = mtcars)
#>
#> Coefficients:
#>               Estimate Std. Error z value Pr(>|z|)
#> (Intercept)    894.228 365884.162   0.002    0.998
#> wt            -202.865  84688.218  -0.002    0.998
#> hp              -2.021    858.062  -0.002    0.998
#>
#> (Dispersion parameter for binomial family taken to be 1)
#>
#>     Null deviance: 4.3860e+01  on 31  degrees of freedom
#> Residual deviance: 1.1156e-08  on 29  degrees of freedom
#> AIC: 6
#>
#> Number of Fisher Scoring iterations: 25
```

**Interpretation**

- Coefficients: The log-odds of high mpg decrease as weight or horsepower increase.
- Odds Ratio: The exponentiated coefficients represent the odds of having high mpg given a one-unit increase in weight or horsepower.

# Key Assumptions of Logistic Regression

1. **Linearity of the Logit**
   - The relationship between the independent variables and the log-odds of the dependent variable should be linear.
2. **Independence of Errors**
   - Observations should be independent of each other.
3. **No Multicollinearity**
   - Independent variables should not be highly correlated with each other.
4. **Adequate Sample Size**
   - A sufficient number of events for each predictor is necessary for stable estimates.

We'll explore how to check these assumptions and interpret the results.

# Linearity of the Logit

- The relationship between the log-odds of the outcome and each predictor should be linear.

## How to check:

1. **Box-Tidwell Test**: A statistical test to check linearity of the logit.

```
# Checking Linearity of Logit using Box-Tidwell Test
mtcars$log_wt <- log(mtcars$wt)
log_model_test <- glm(mpg_high ~ wt + I(wt * log_wt) + hp, data = mtcars, family = binomial)
summary(log_model_test)
#>
#> Call:
#> glm(formula = mpg_high ~ wt + I(wt * log_wt) + hp, family = binomial,
#>     data = mtcars)
#>
#> Coefficients:
#>                  Estimate Std. Error   z value Pr(>|z|)
#> (Intercept)     1.239e+16  1.959e+08  63272684   <2e-16 ***
#> wt             -5.784e+15  1.360e+08 -42540630   <2e-16 ***
#> I(wt * log_wt)  2.173e+15  6.084e+07  35709419   <2e-16 ***
#> hp             -2.302e+13  2.364e+05 -97357716   <2e-16 ***
#> ---
#> Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
#>
#> (Dispersion parameter for binomial family taken to be 1)
#>
#>     Null deviance:  43.86  on 31  degrees of freedom
#> Residual deviance: 288.35  on 28  degrees of freedom
#> AIC: 296.35
#>
```

# Independence of Errors

- Residuals (errors) should be independent across observations.

## How to check:

1. **Durbin-Watson Test**: Used to check for autocorrelation in residuals (as in linear regression).

```
# Checking independence of errors using Durbin-Watson Test
dwtest(log_model)
#>
#>  Durbin-Watson test
#>
#> data:  log_model
#> DW = 1.3853, p-value = 0.0244
#> alternative hypothesis: true autocorrelation is greater than 0
```

## Interpretation

- A Durbin-Watson statistic close to 2 indicates that the errors are independent.

# No Multicollinearity

- Independent variables should not be highly correlated.

## How to check:

1. **Variance Inflation Factor (VIF)**: Detects multicollinearity.
2. **Correlation Matrix**: Visualize correlations between predictors.

```
# Checking Multicollinearity using VIF
vif(log_model)
#>       wt       hp
#> 4.234545 4.234545
```

# No Multicollinearity

```
# Correlation matrix for predictors
cor(mtcars[, c("wt", "hp")])
#>           wt        hp
#> wt 1.0000000 0.6587479
#> hp 0.6587479 1.0000000
```

### Interpretation

- VIF values greater than 5 suggest problematic multicollinearity.
- High correlations in the correlation matrix also indicate multicollinearity.

# Adequate Sample Size

- Logistic regression requires a sufficient number of events (positive cases) per predictor.

## How to check:

1. **Events per Variable (EPV)**: A rule of thumb is at least 10 events (cases where the outcome is 1) per independent variable.

```
# Check how many events we have for the binary outcome
table(mtcars$mpg_high)
#>
#>  0  1
#> 18 14
```

## Interpretation

- Ensure that the number of events (cases where mpg_high == 1) divided by the number of predictors is greater than 10.

# Model Diagnostics – Residuals and Influential Points

## Diagnostic Plots

- Even though logistic regression doesn't assume homoscedasticity or normality of residuals, checking residuals for extreme values and influential points is still useful.

## How to check:

1. **Residuals vs Fitted Plot**: Look for extreme residuals.
2. **Cook's Distance**: Identify influential points.

# Model Diagnostics – Residuals and Influential Points

```
# Residuals vs Fitted Plot for Logistic Regression
plot(log_model, which = 1)
```

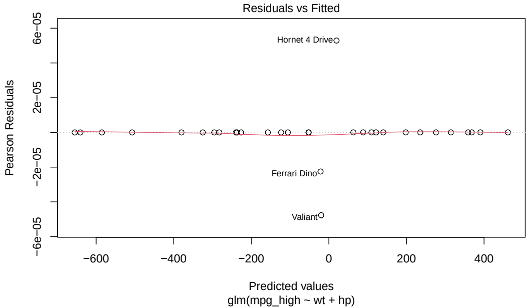
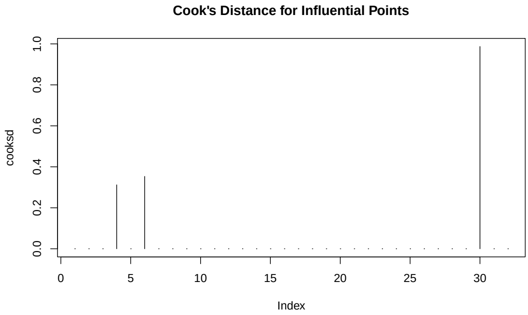



**Figure 5:** Residuals vs Fitted Plot for Logistic Model

# Model Diagnostics – Residuals and Influential Points

## Cook's Distance to identify influential points

```
cooksd <- cooks.distance(log_model)
plot(cooksd, type = "h", main = "Cook's Distance for Influential Points")
```




### Interpretation

- Residuals vs Fitted Plot: Look for residuals that significantly deviate from the main cluster.
- Cook's Distance: Points with high Cook's Distance ($> 1$) could be influential.

# Formula

Odds Ratio $= e^{\beta}$

- This tells us the change in odds for a one-unit change in the predictor.

```
# Calculate Odds Ratios
exp(coef(log_model))
#>  (Intercept)           wt           hp
#>          Inf 7.884361e-89 1.325092e-01
```

## Interpretation

- For each unit increase in weight (wt), the odds of having high mpg decrease by the calculated odds ratio.
- For each unit increase in horsepower (hp), the odds of having high mpg also decrease by the calculated odds ratio.

# Thank You!

- Thank you for your attention!
- Feel free to reach out with any questions.

**Contact:**


- **Email**: mahesh.divakaran01@gmail.com
- **LinkedIn**: linkedin.com/in/imaheshdivakaran