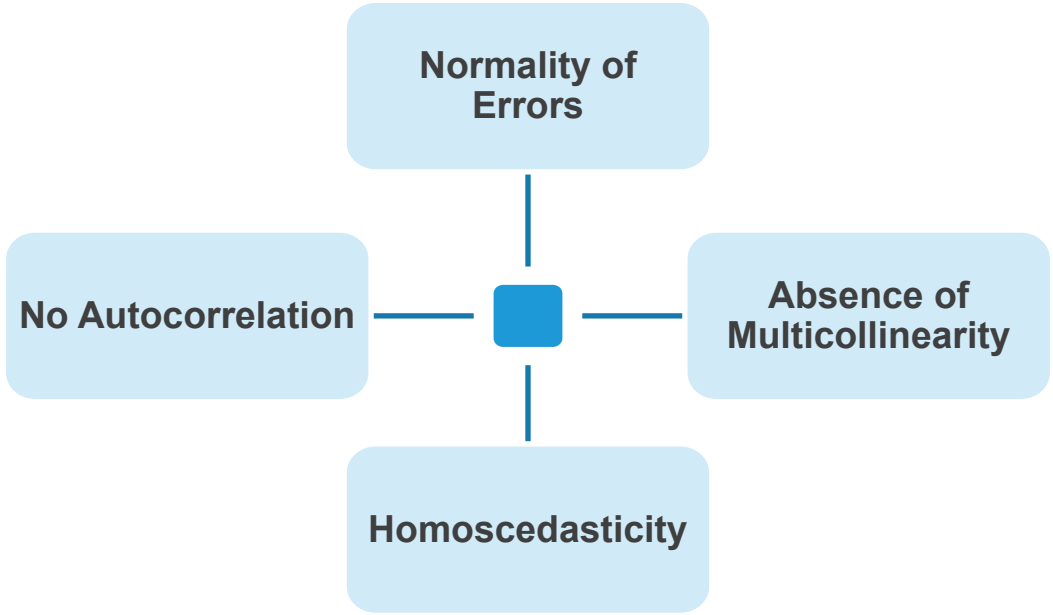
# Multiple Linear Regression

# 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

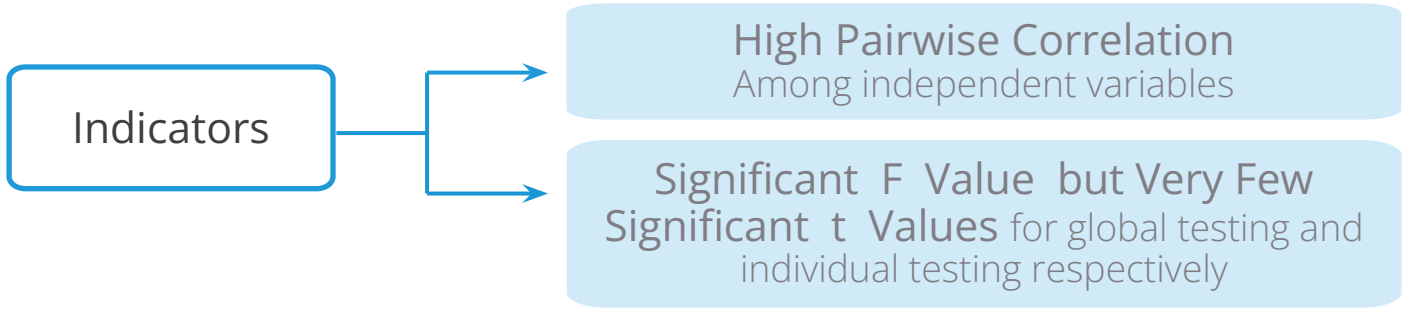
Multicollinearity has two serious consequences:

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 Python

```
#Importing the Data, Fitting Linear Model
import pandas as pd
perindex=pd.read_csv("Performance Index.csv")

import statsmodels.formula.api as smf
jpimodel=smf.ols('jpi~aptitude+tol+technical+general',data=perindex).fit()
```

```
#Variance Inflation Factor
from patsy import dmatrices
from statsmodels.stats.outliers_influence import variance_inflation_factor

# Break data into left and right hand side; y and X
y, X = dmatrices('jpi ~ aptitude + tol +  technical +general',
data=perindex, return_type="dataframe")
```

- *patsy is a library that helps in converting data frames into design matrices.*
- *dmatrices Construct two design matrices using specified formula.By convention, the first matrix is the "y" data, and the second is the "x" data.*
- *variance_inflation_factor() requires a design matrix as input to calculate vif.*

* We use the same dataset “Performance Index” which was used in previous ppt

# Detecting Multicollinearity in Python

```
# Calculating VIF & getting vif with their corresponding variable
# name

vif = pd.Series([variance_inflation_factor(X.values, i)for i in
range(X.shape[1])],index=X.columns)

vif
```

*variance_inflation_factor() calculates VIFs.*

```
# Output
```

```
Intercept     143.239081
aptitude        1.179906
tol             1.328205
technical       2.073907
general         2.024968
dtype: float64
```

*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.

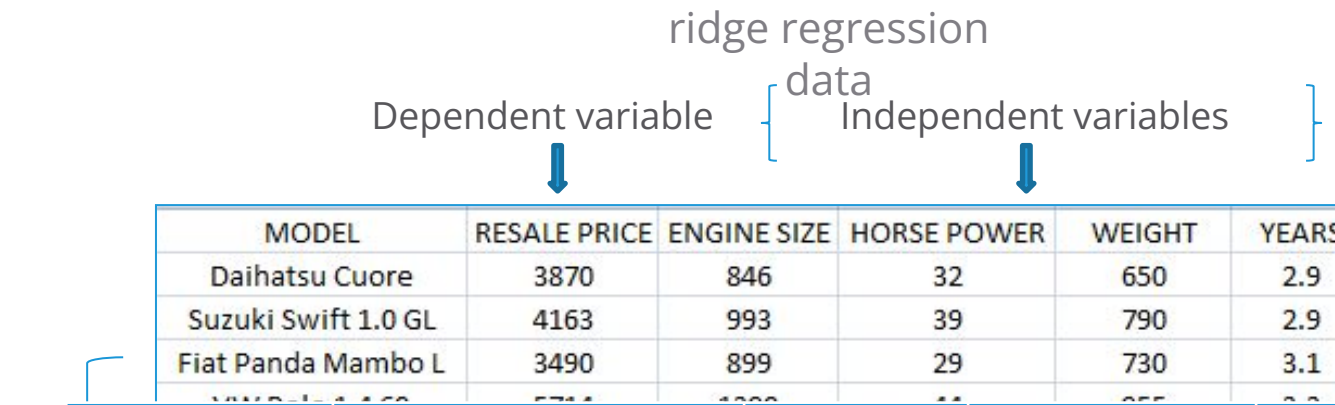
## Objective

- 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




| 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 |

| 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=pd.read_csv("ridge regression data.csv")
```

```
# Graphical representation of data
# Install and load package “seaborn”
import seaborn as sns
import matplotlib.pyplot as plt

sns.pairplot(ridgedata[['MODEL', 'RESALE PRICE', 'ENGINE SIZE', 'HORSE POWER','YEARS']]);plt.title('Fig.No. 2: ScatterPlot Matrix')
```

*pairplot() in the package seaborn is used to plot the scatter plot matrix*

# Scatter Plot Matrix

*Interpretation :*

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

```
# Output
```

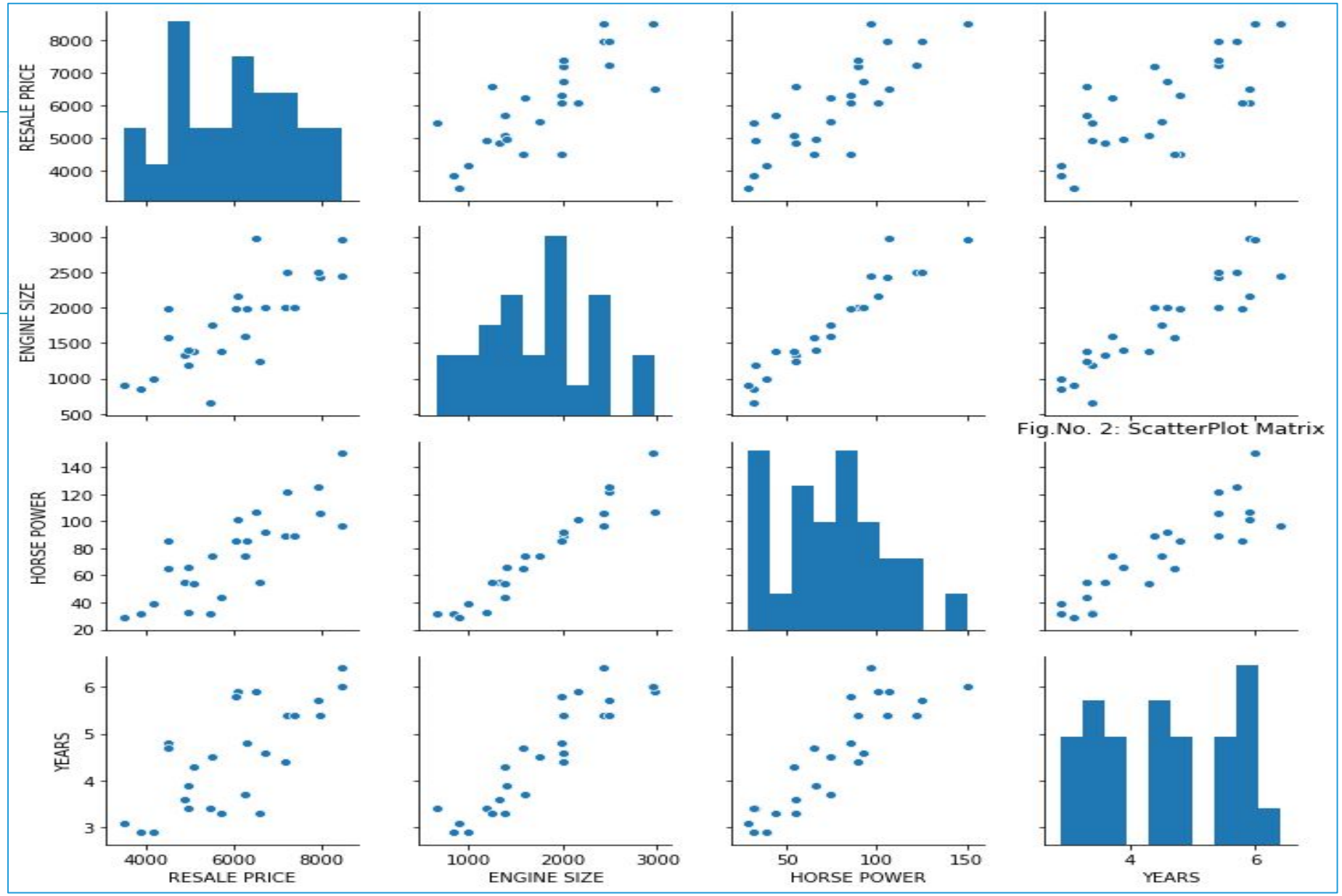



Fig.No. 2: ScatterPlot Matrix

# Detecting Multicollinearity in Python

```
#Importing the Data, Fitting Linear Model
ridgedata.columns = [c.replace(' ', '_') for c in ridgedata.columns]
model = smf.ols('RESALE_PRICE~ENGINE_SIZE+ HORSE_POWER + WEIGHT + YEARS',
data = ridgedata).fit()
```

*In pandas, the column names cannot contain spaces in between. Hence, before applying ols() remove spaces from column names wherever required.*

```
#Variance Inflation Factor
y, X = dmatrices('RESALE_PRICE~ENGINE_SIZE+ HORSE_POWER + WEIGHT +
YEARS', data=ridgedata, return_type="dataframe")
vif = pd.Series([variance_inflation_factor(X.values, i)for i in
range(X.shape[1])],index=X.columns)
vif
```

```
# Output
```

| | |
|---|---|
| Intercept | 26.193279 |
| ENGINE_SIZE | 15.759113 |
| HORSE_POWER | 12.046734 |
| WEIGHT | 9.113045 |
| YEARS | 13.978640 |
| dtype: float64 | |

*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 Python | • Variance Inflation Factor **`variance_inflation_factor()`** function in package **statsmodel** |
| Remedial Measures | • Drop variables<br>• Use Principal Component Regression<br>• Ridge regression |