Linear Regression using R and Python

DR. ARNAB SADHU

ASSISTANT PROFESSOR

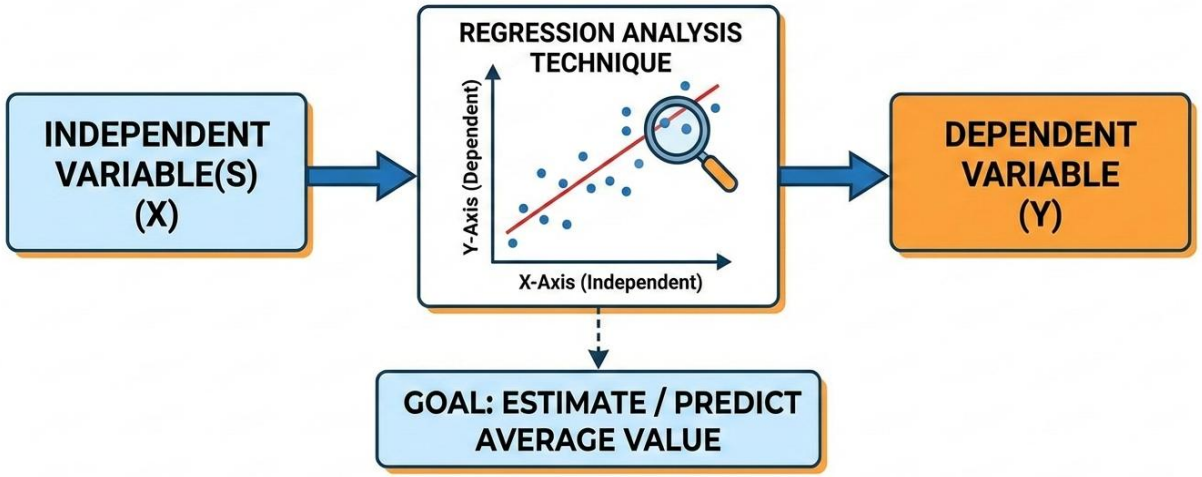
COMPUTER CENTRE

VIDYASAGAR UNIVERSITY

Regression analysis

- Regression Analysis is a technique of studying the dependence of one variable (called dependent variable), on one or more variables (called independent variables), with a view to estimate or predict the average value of the dependent variables in terms of the known or fixed values of the independent variables.

The dependent variable is variously known as explained variables, predicted, response and endogenous variables. While the independent variable is known as explanatory, predictor, regressor and exogenous variable.



A technique to study the dependence of Y on one or more X variables for prediction.

Regression analysis

- Estimate the relationship that exists, on the average, between the dependent variable and the explanatory variable
- Determine the effect of each of the explanatory variables on the dependent variable, controlling the effects of all other explanatory variables
- Predict the value of dependent variable for a given value of the explanatory variable

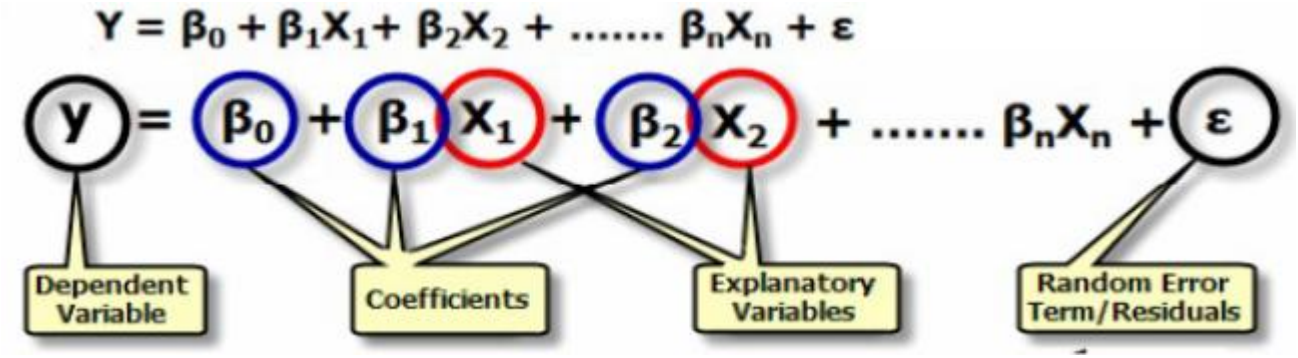
Assumptions of the Linear Regression Model

I. Linear Functional form
II. Fixed independent variables
III. Independent observations
IV. Representative sample and proper specification of the model (no omitted variables)
V. Normality of the residuals or errors
VI. Equality of variance of the errors (homogeneity of residual variance)
VII. No multicollinearity
VIII. No autocorrelation of the errors
IX. No outlier distortion

Regression analysis types

- **I. Linear Regression:** straight-line relationship of the form: y=mx+b
- **II. Non-linear Regression:** implies curved relationships – logarithmic relationships
- **III. Cross Sectional**: data gathered from the same time period
- **IV. Time Series:** Involves data observed over equally spaced points in time.

Model



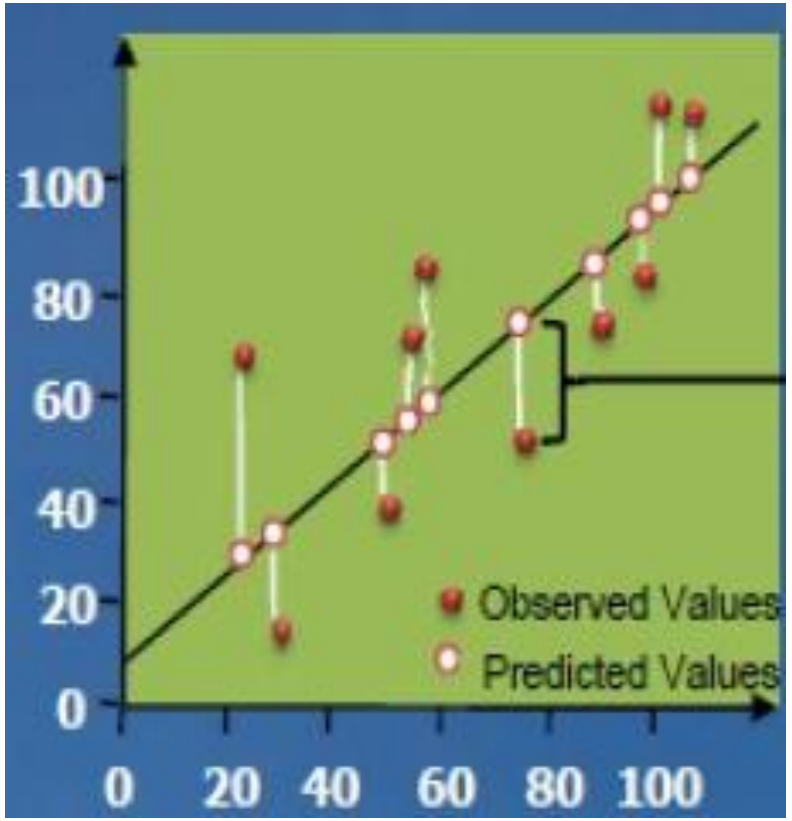
- **Dependent variable**: The single variable being explained/ predicted by the regression model
- **Independent variable**: The explanatory variable(s) used to predict the dependent variable.
- **Coefficients (β):** values, computed by the regression tool, reflecting explanatory to dependent variable relationships.
- **Residuals (ε):** the portion of the dependent variable that isn't explained by the model; the model under and over predictions.

Linear Regression types

- **A. Simple Linear Regression:** Single explanatory variable
- **B. Multiple Linear Regression:** Include any number of explanatory variable.

Simple Linear Regression Model

- Only one independent variable, x
- Relationship between x and y is described by a linear function
- Changes in y are assumed to be caused by changes in x



Mathematical equation

The general mathematical equation for a linear regression is −

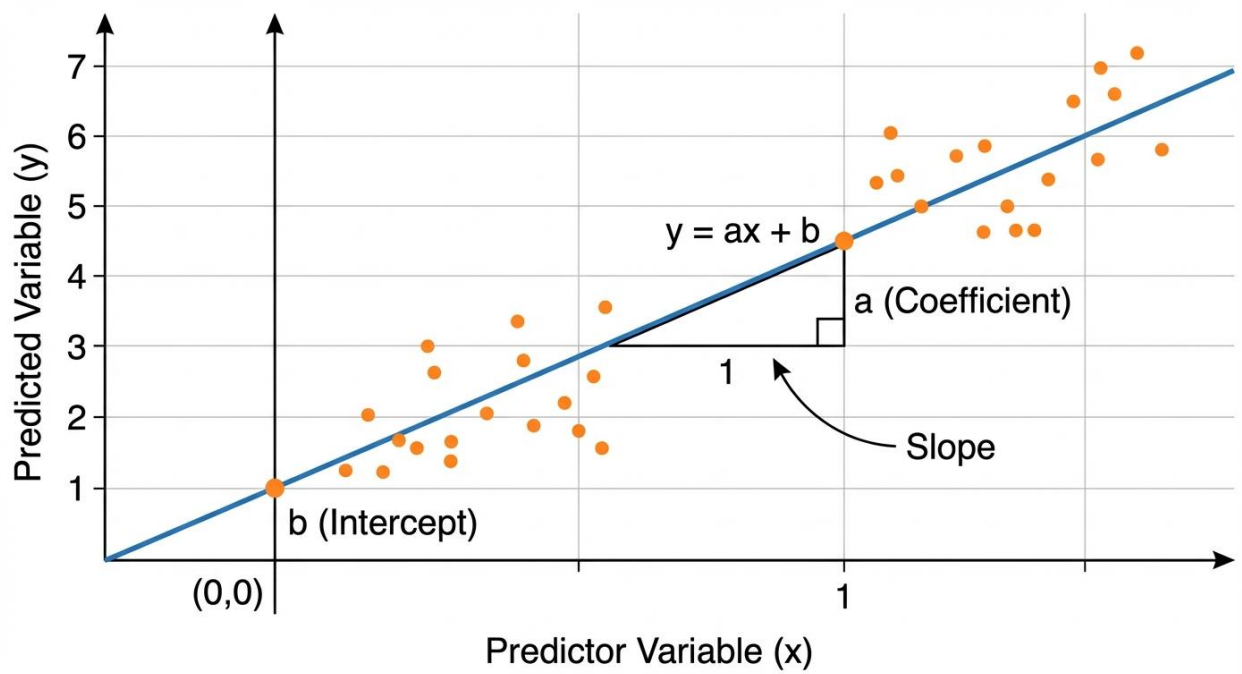
- $y = ax + b$

Where

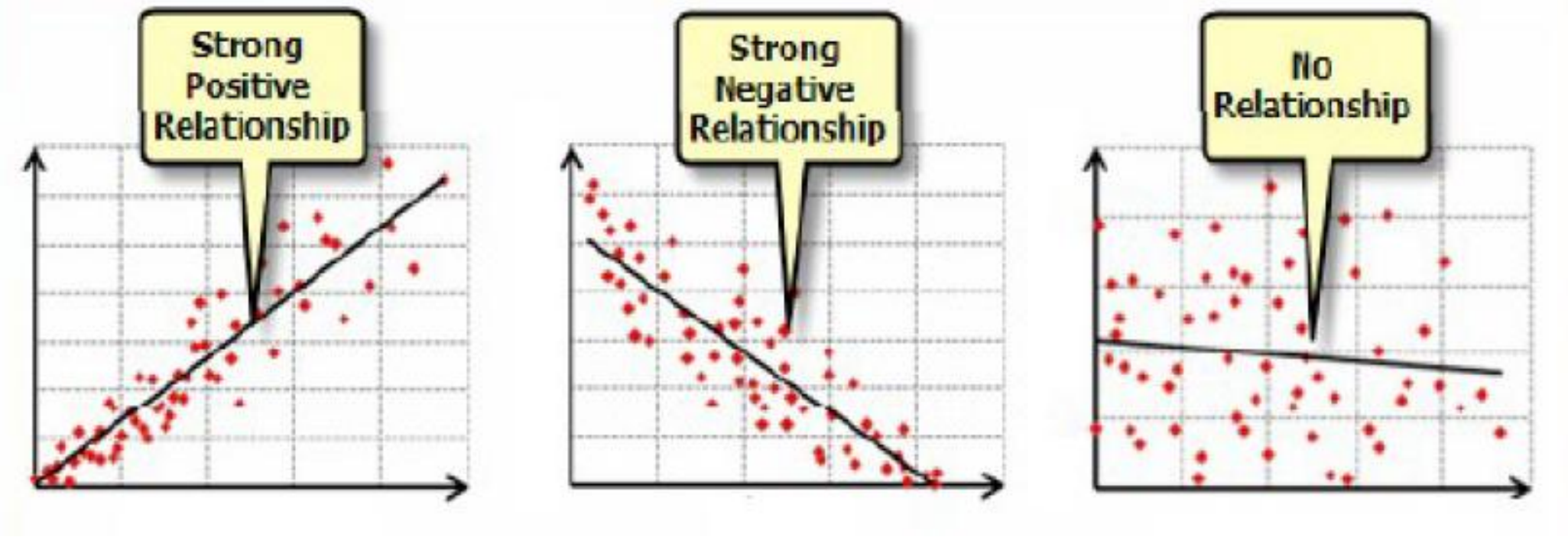
y is predicted variable

x is predictor variable

a is coefficients and b is intercept.



Types of linear regression model



Simple Linear Regression Example

- A real estate agent wishes to examine the relationship between the selling price of a home and its size (measured in square feet)
- A random sample of 10 houses is selected
  - Dependent variable (y) = house price in $1000s
  - Independent variable (x) = square feet

Steps to Establish a Regression in R

- A simple example of regression is predicting weight of a person when his height is known. To do this we need to have the relationship between height and weight of a person.
- The steps to create the relationship is −
  - Carry out the experiment of gathering a sample of observed values of height and corresponding weight.
  - Create a relationship model using the **lm()** functions in R.
  - Find the coefficients from the model created and create the mathematical equation using these
  - Get a summary of the relationship model to know the average error in prediction. Also called **residuals**.
  - To predict the weight of new persons, use the **predict()** function in R.

Steps to establish a Regression in Python

scikit-learn is widely used because of its simplicity, scalability, and integration with other machine learning tools. It offers a simple and efficient way to perform linear regression using the LinearRegression class.

- Step 1: Import the library and load your dataset.
- Step 2: Reshape the predictor variable if necessary (for single-variable regression).
- Step 3: Create a model instance using LinearRegression().
- Step 4: Fit the model to the training data.
- Step 5: Retrieve the model parameters: The .coeff_ attribute returns the slope, while .intercept_ provides the intercept.tercept.

Example: a Simple Linear Regression

- A real estate agent wishes to examine the relationship between the selling price of a home and its size (measured in square feet)
- A random sample of 10 houses is selected
  - Dependent variable (y) = house price in $1000s
  - Independent variable (x) = square feet

Example: a Simple Linear Regression

Sample data

House price (in \$1000s) y	Square Feet x
245	1400
312	1600
279	1700
308	1875
199	1100
219	1550
405	2350
324	2450
319	1425
255	1700

Background Math

The Line

- Our aim is to calculate the values **m** (slope) and **b** (y-intercept) in the equation of a line :

  - **y = mx + b**

- Where
  - **y** = how far up
  - **x** = how far along
  - **m** = Slope or Gradient (how steep the line is)
  - **b** = the Y Intercept (where the line crosses the Y axis)

Background Math cont..

Steps To find the line of best fit for **N** points:

- **Step 1**: For each (x,y) point calculate x^2 and xy
- **Step 2**: Sum all x, y, x^2 and xy, which gives us Σx, Σy, Σx^2 and Σxy
- **Step 3**: Calculate Slope **m**:
  - $\mathbf{m} = \frac{N\sum xy - \sum x \sum y}{N\sum x^2 - (\sum x)^2}$
- **Step 4**: Calculate Intercept **b**:
  - $\mathbf{b} = \frac{\sum y - m\sum x}{N}$
- **Step 5**: Assemble the equation of a line
  - $y = mx + b$

From the last example

y	x
245	1400
312	1600
279	1700
308	1875
199	1100
219	1550
405	2350
324	2450
319	1425
255	1700

- $N = 10$
- $\sum x = 17150, \sum y = 2865, \sum x^2 = 30983750, \sum xy = 5085975$
- $m = \frac{10*5085975-49134750}{10*30983750-294122500} = 0.1097$
- $b = \frac{2865-0.1097*17150}{10} = 98.24$
- So, $y = 0.1097 * x + 98.24$

In R:

```
> x_sq<-x^2
> xy<-x*y
> m = ((10*sum(xy))-(sum(x)*sum(y)))/(10*sum(x_sq)-sum(x)^2)
> b = (sum(y)-m*sum(x))/10
```

Corresponding R code

```
> x<-c(1400,1600,1700,1875,1100,1550,2350,2450,1425,1700)
> y<-c(245,312,279,308,199,219,405,324,319,255)
> relation<-lm(y~x)
> print(relation)
 Call:
 lm(formula = y ~ x)
 Coefficients:
 (Intercept)   x
  98.2483   0.1098
```

```
> print(summary(relation))
Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-49.388 -27.388  -6.388  29.577  64.333

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 98.24833   58.03348   1.693   0.1289
x            0.10977    0.03297   3.329   0.0104 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 41.33 on 8 degrees of freedom
Multiple R-squared:  0.5808,Adjusted R-squared:  0.5284
F-statistic: 11.08 on 1 and 8 DF,  p-value: 0.01039
```

Prediction

Function: predict()

- The basic syntax for predict() in linear regression is −
  - predict(object, newdata)
- Following is the description of the parameters used −
  - **object:** object is the formula which is already created using the lm() function.
  - **newdata:** newdata is the vector containing the new value for predictor variable.

Example

- Find price of a house with size in square feet = 1600

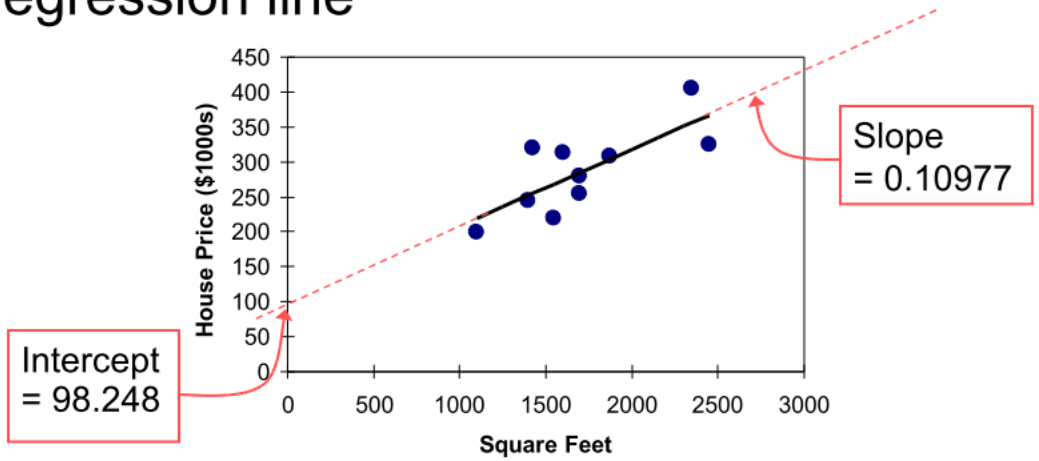
```
> relation<-lm(y~x)
> a<-data.frame(x=1600)
> result<-predict(relation,a)
> print(result)
```

- 273.8767
- Using math: y = mx+b = 0.1097*1600+98.24 = 273.8

Visualize the Regression Graphically

```
> plot(y,x,col="blue",main= "Height & Weight Regression",
abline(lm(x~y)),cex=1.3, pch=16, xlab="Weight in Kg", ylab="Height in
cm")
```

- House price model: scatter plot and regression line



$$\widehat{\text{house price}} = 98.24833 + 0.10977\ (\text{square feet})$$

Evaluating Linear Regression

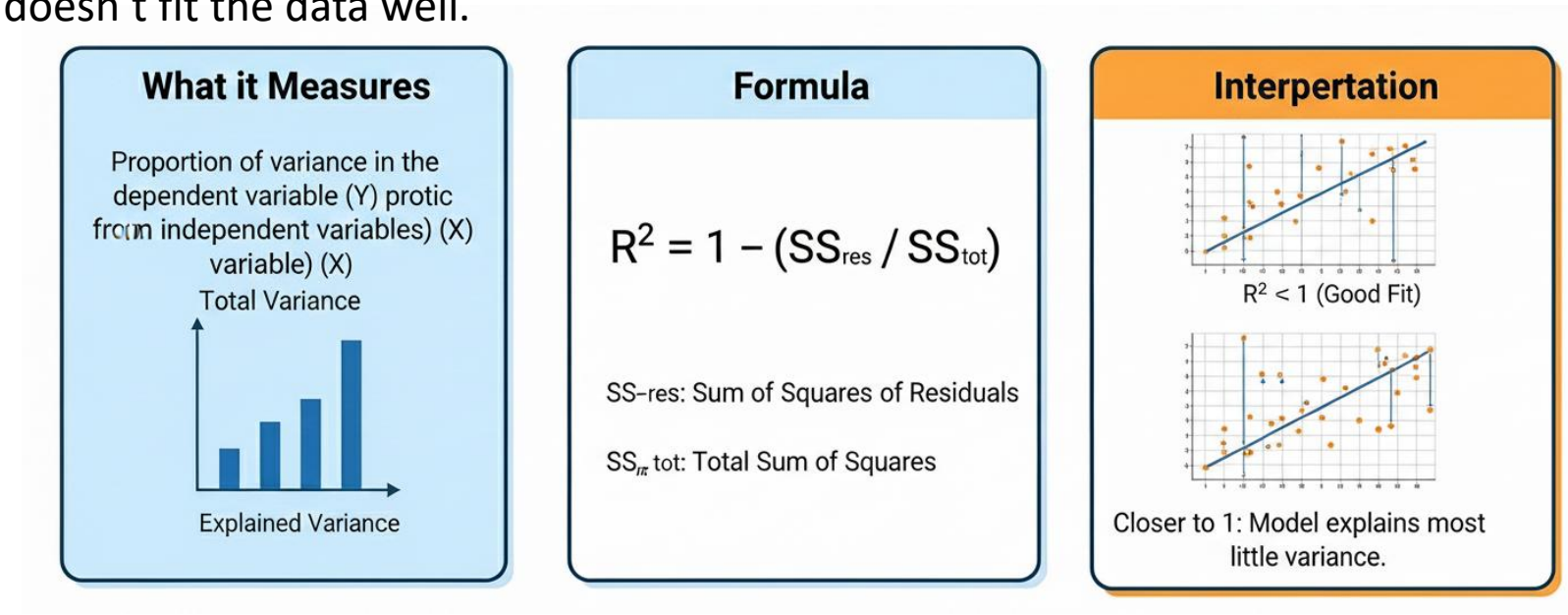
- R-SQUARED
- ADJUSTED R-SQUARED
- RMSE

R-squared (R^2)

This metric tells you how much of the variance in the dependent variable can be explained by the independent variables.

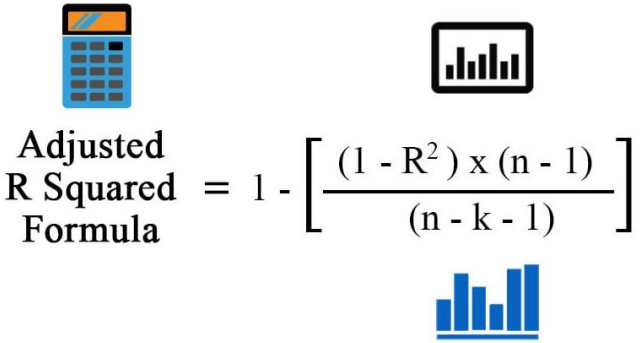
If R^2 is said 0.6, it means that the independent variable explains 60% of the variation in the dependent variable.

An R^2 value closer to 1 indicates that the model explains a large proportion of the variance, while a value closer to 0 means the model doesn't fit the data well.



Adjusted R-squared (Adj. R^2)

- The problem with R^2 is its value increases with adding more variables, irrespective of the significance of the variable.
- Adjusted r squared adjusts the R^2 value for the number of terms in the model.



Think of standard R^2 as a score that rewards you for **complexity**—the more ingredients you add to your recipe, the higher the score, even if the soup tastes the same.

Adjusted R^2 acts like a strict food critic. It balances two competing forces:

1. **Precision:** How well the model explains the data (like standard R^2).
2. **Simplicity:** A penalty for adding unnecessary complexity (number of variables).

If you add a new variable to your model, Adjusted R^2 asks: "*Did this new variable improve the prediction enough to justify the cost of making the model more complex?*"

- If the answer is **Yes** (it adds significant value), the score goes up. 📈
- If the answer is **No** (it's just noise), the score goes down. 📉

A Concrete Example: Predicting House Prices 🏠

Model A (1 Variable):

- **Predictor:** Square Footage (x_1).
- **Result:** This is a strong predictor.
- **R^2:** 0.70
- **Adjusted R^2:** 0.69

Model B (2 Variables):

- **Predictors:** Square Footage (x_1) + The Homeowner's Favorite Color (x_2).
- **Result:** The favorite color has statistically zero relationship to the house price. However, just by random chance, it might align slightly with some data points.
- **R^2:** 0.71 (It went up slightly! This is misleading.)
- **Adjusted R^2:** 0.67 (It went **down**.)

The Verdict: Adjusted R^2 correctly identifies that Model B is actually *worse* because it became more complicated without becoming significantly more accurate.

R^2 vs Adj- R^2

Feature	R-Squared (R^2)	Adjusted R-Squared ($Adj\ R^2$)
What it measures	The proportion of variance in the target variable explained by the predictors.	The proportion of variance explained, **corrected for the number of predictors** in the model.
Reaction to new variables	**Always increases** (or stays the same) when you add a new variable, even if it's useless.	**Increases only if** the new variable improves the model more than would be expected by chance.
Reaction to complexity	Ignores model complexity. It rewards "kitchen sink" models (throwing everything in).	Penalizes complexity. It decreases if you add variables that don't pull their weight.
Reliability	Can be misleading (over-optimistic) for models with many predictors.	More reliable for comparing models with different numbers of predictors.
Best Use Case	explaining how well the model fits the current dataset (descriptive).	deciding which features to keep when building a predictive model (feature selection).

RMSE

- Root Mean Squared Error (RMSE): RMSE measures the average magnitude of the errors in the model's predictions, with the same units as the dependent variable. A lower RMSE indicates better predictive performance.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (Predicted_i - Actual_i)^2}{N}}$$