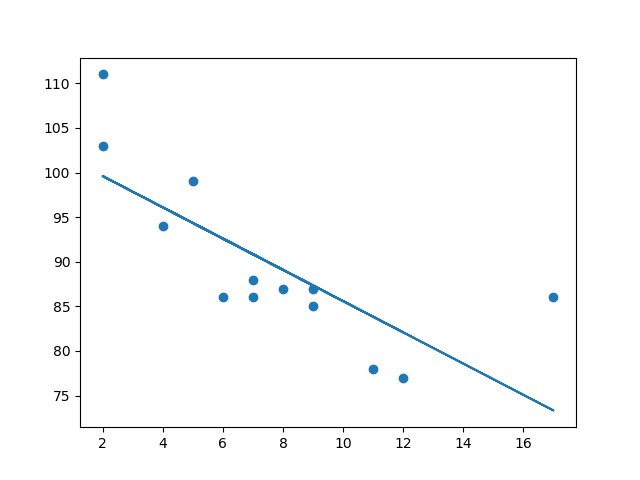
# Simple Linear Regression

The term regression is used when you try to find the relationship between variables. In Machine Learning, and in statistical modeling, that relationship is used to predict the outcome of future events.

Linear regression uses the relationship between the data-points to draw a straight line through all them.

This line can be used to predict future values.



**Linear regression** is also a type of [**supervised machine-learning algorithm**](https://www.geeksforgeeks.org/supervised-machine-learning/) that learns from the labelled datasets and maps the data points with most optimized linear functions which can be used for prediction on new datasets. It computes the linear relationship between the dependent variable and one or more independent features by fitting a linear equation with observed data.

The model’s equation offers **clear coefficients** that illustrate the **influence of each independent variable on the dependent variable**, enhancing our understanding of the underlying relationships. Its simplicity is a significant advantage; linear regression is transparent, easy to implement.

Our primary objective while using linear regression is to locate the best-fit line, which implies that the error between the predicted and actual values should be kept to a minimum.

The best Fit Line equation provides a straight line that represents the relationship between the dependent and independent variables. The slope of the line indicates how much the dependent variable changes for a unit change in the independent variable(s).



Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x)).

The model gets the best regression fit line by finding the best θ1 and θ2 values.

* **θ1:** intercept
* **θ2:** coefficient of x (also known as slope)

Once we find the best θ1 and θ2 values, we get the best-fit line.

Simple Linear Regression

[Simple linear regression](https://www.geeksforgeeks.org/simple-linear-regression-in-python/) is the simplest form of linear regression and it involves only one independent variable and one dependent variable. The equation for simple linear regression is:  
y=β0+β1X*y*=*β*0​+*β*1​*X*  
where:

* Y is the dependent variable
* X is the independent variable
* β0 is the intercept
* β1 is the slope

Multiple Linear Regression

[Multiple linear regression](https://www.geeksforgeeks.org/ml-multiple-linear-regression-using-python/) involves more than one independent variable and one dependent variable. The equation for multiple linear regression is:  
y=β0+β1X1+β2X2+………βnXn*y*=*β*0​+*β*1​*X*1+*β*2​*X*2+………*βn*​*Xn*  
where:

* Y is the dependent variable
* X1, X2, …, Xn are the independent variables
* β0 is the intercept
* β1, β2, …, βn are the slopes

**The goal of the algorithm is to find the best Fit Line equation that can predict the values based on the independent variables.**

In regression set of records are present with X and Y values and these values are used to learn a function so if you want to predict Y from an unknown X this learned function can be used. In regression we have to find the value of Y, So, a function is required that predicts continuous Y in the case of regression given X as independent features.

## Evaluation Metrics

**Mean Square Error (MSE)**

[Mean Squared Error (MSE)](https://www.geeksforgeeks.org/python-mean-squared-error/) is an evaluation metric that calculates the average of the squared differences between the actual and predicted values for all the data points. The difference is squared to ensure that negative and positive differences don’t cancel each other out.

MSE=1n∑i=1n(yi–yi^)2*MSE*=*n*1​∑*i*=1*n*​(*yi*​–*yi*​​)2

Here,

* n is the number of data points.
* yi is the actual or observed value for the ith data point.
* yi^*yi*​​ is the predicted value for the ith data point.

MSE is a way to quantify the accuracy of a model’s predictions. MSE is sensitive to outliers as large errors contribute significantly to the overall score.

**Mean Absolute Error (MAE)**

[Mean Absolute Error](https://www.geeksforgeeks.org/how-to-calculate-mean-absolute-error-in-python/)is an evaluation metric used to calculate the accuracy of a regression model. MAE measures the average absolute difference between the predicted values and actual values.

Mathematically, MAE is expressed as:

MAE=1n∑i=1n∣Yi–Yi^∣*MAE*=*n*1​∑*i*=1*n*​∣*Yi*​–*Yi*​​∣

Here,

* n is the number of observations
* Yi represents the actual values.
* Yi^*Yi*​​ represents the predicted values

Lower MAE value indicates better model performance. It is not sensitive to the outliers as we consider absolute differences.

**Root Mean Squared Error (RMSE)**

The square root of the residuals’ variance is the [Root Mean Squared Error](https://www.geeksforgeeks.org/root-mean-square-error-in-r-programming/). It describes how well the observed data points match the expected values, or the model’s absolute fit to the data.

In mathematical notation, it can be expressed as:  
RMSE=RSSn=∑i=2n(yiactual−yipredicted)2n*RMSE*=*nRSS*​​=*n*∑*i*=2*n*​(*yiactual*​−*yipredicted*​)2​​  
Rather than dividing the entire number of data points in the model by the number of degrees of freedom, one must divide the sum of the squared residuals to obtain an unbiased estimate. Then, this figure is referred to as the Residual Standard Error (RSE).

In mathematical notation, it can be expressed as:  
RMSE=RSSn=∑i=2n(yiactual−yipredicted)2(n−2)*RMSE*=*nRSS*​​=(*n*−2)∑*i*=2*n*​(*yiactual*​−*yipredicted*​)2​​

RSME is not as good of a metric as R-squared. Root Mean Squared Error can fluctuate when the units of the variables vary since its value is dependent on the variables’ units (it is not a normalized measure).

**Coefficient of Determination (R-squared)**

[R-Squared](https://www.geeksforgeeks.org/r-squared/) is a statistic that indicates how much variation the developed model can explain or capture. It is always in the range of 0 to 1. In general, the better the model matches the data, the greater the R-squared number.  
In mathematical notation, it can be expressed as:  
R2=1−(RSSTSS)*R*2=1−(*TSSRSS*​)

* [**Residual sum of Squares**](https://www.geeksforgeeks.org/residual-sum-of-squares/#:~:text=Residual%20sum%20of%20squares%20is%20used%20to%20calculate%20the%20variance,squares%2C%20the%20better%20the%20model.)**(RSS): The** sum of squares of the residual for each data point in the plot or data is known as the residual sum of squares, or RSS. It is a measurement of the difference between the output that was observed and what was anticipated.  
  RSS=∑i=2n(yi−b0−b1xi)2*RSS*=∑*i*=2*n*​(*yi*​−*b*0​−*b*1​*xi*​)2
* **Total Sum of Squares (TSS):**The sum of the data points’ errors from the answer variable’s mean is known as the total sum of squares, or TSS.  
  TSS=∑(y−yi‾)2*TSS*=∑​(*y*−*yi*​​)2

R squared metric is a measure of the proportion of variance in the dependent variable that is explained the independent variables in the model.

**Adjusted R-Squared Error**

Adjusted R2 measures the proportion of variance in the dependent variable that is explained by independent variables in a regression model. [Adjusted R-square](https://www.geeksforgeeks.org/ml-adjusted-r-square-in-regression-analysis/) accounts the number of predictors in the model and penalizes the model for including irrelevant predictors that don’t contribute significantly to explain the variance in the dependent variables.

Mathematically, adjusted R2 is expressed as:

AdjustedR2=1–((1−R2).(n−1)n−k−1)*AdjustedR*2=1–(*n*−*k*−1(1−*R*2).(*n*−1)​)

Here,

* n is the number of observations
* k is the number of predictors in the model
* R2 is coeeficient of determination

Adjusted R-square helps to prevent overfitting. It penalizes the model with additional predictors that do not contribute significantly to explain the variance in the dependent variable.

While evaluation metrics help us measure the performance of a model, regularization helps in improving that performance by addressing overfitting and enhancing generalization.

## Regularization Techniques

**Regularization techniques** are used in regression to **prevent overfitting**, where a model becomes too complex and starts memorizing the data instead of learning general patterns. These techniques add a penalty term to the loss function, discouraging the model from giving too much importance to any single feature.

### **1. Lasso Regression (L1 Regularization)**

* **What it does:** Adds a penalty proportional to the **absolute value of the coefficients**. This not only shrinks the coefficients but can force some to become exactly zero, effectively **selecting the most important features**.
* **Formula:**  
  Loss=MSE+λ∑i=1n∣βi∣Loss = \text{MSE} + \lambda \sum\_{i=1}^n |\beta\_i|Loss=MSE+λ∑i=1n​∣βi​∣
* **Real-Life Example:**  
  In the **house price prediction** scenario, if "fountain view" or "paint color" has negligible or no impact, Lasso will shrink their coefficients to zero, **removing them entirely** from the model. This simplifies the model and makes it interpretable.

### **2. Ridge Regression (L2 Regularization)**

* **What it does:** Adds a penalty proportional to the **square of the coefficients**. This shrinks the coefficients of less important features but doesn’t force them to zero.
* **Formula:**  
  Loss=MSE+λ∑i=1nβi2Loss = \text{MSE} + \lambda \sum\_{i=1}^n \beta\_i^2Loss=MSE+λ∑i=1n​βi2​  
  Where λ\lambdaλ is a tuning parameter (controls the strength of regularization), and βi\beta\_iβi​ are the feature coefficients.
* **Real-Life Example:**  
  Imagine you're predicting **house prices** based on 10 features (like area, number of rooms, location). If one feature, say "fountain view," is less important, Ridge will reduce its influence but not completely remove it. This keeps all features contributing but limits their impact.

### **3. Elastic Net (Combination of L1 & L2)**

* **What it does:** Combines the penalties of both Ridge (L2) and Lasso (L1). This is useful when you want the benefits of both approaches: feature selection (Lasso) and reduced multicollinearity (Ridge).
* **Formula:**  
  Loss=MSE+αλ∑i=1n∣βi∣+(1−α)λ∑i=1nβi2Loss = \text{MSE} + \alpha \lambda \sum\_{i=1}^n |\beta\_i| + (1 - \alpha) \lambda \sum\_{i=1}^n \beta\_i^2Loss=MSE+αλ∑i=1n​∣βi​∣+(1−α)λ∑i=1n​βi2​  
  α\alphaα: Balances the contribution of L1 and L2 regularization.
* **Real-Life Example:**  
  Elastic Net is great for datasets with many correlated features (e.g., predicting stock prices where multiple features are interrelated). It selects relevant features (like Lasso) while controlling for multicollinearity (like Ridge).

**Comparison with Real-Life Analogy**

Think of these techniques as controlling how much attention you give to each feature:

**Ridge**: Like gently turning down the volume on features that are too noisy but still important.

**Lasso**: Like muting features that contribute nothing valuable.

**Elastic Net**: Like combining the two approaches – lower the noise and completely remove irrelevant features if needed.