|  |  |
| --- | --- |
|  |  |
| **Machine Learning** |  |
|  |  |
|  | DATE – 07/11/2022  Machine Learning |
|  | Zodex |

Table of Contents

[1. Introduction 1](#_Toc181480321)

[1.1 Machine Learning lifecycle 1](#_Toc181480322)

[1.2 Type of Machine Learning 1](#_Toc181480323)

[1.2.1 Supervised Machine Learning 1](#_Toc181480324)

[1.2.2 Unsupervised Learning 4](#_Toc181480325)

[1.2.3 Semi-Supervised Learning 5](#_Toc181480326)

[1.2.4 Reinforcement Machine Learning 6](#_Toc181480327)

[1.3 Model-Based vs Instance-Based Learning 6](#_Toc181480328)

[1.3.1 Model-Based Learning 6](#_Toc181480329)

[1.3.2 Instance-Based Learning 7](#_Toc181480330)

[2. Gradient Descent 8](#_Toc181480331)

[2.1 Cost Function 8](#_Toc181480332)

[2.2 Gradient Descent 8](#_Toc181480333)

[2.3 Types of Gradient Descent Algorithm 9](#_Toc181480334)

[2.4 Gradient Descent Learning Rate 10](#_Toc181480335)

[2.5 Learning Rate Schedule 11](#_Toc181480336)

[3. Supervised Machine Learning Algorithms 12](#_Toc181480337)

[3.1 Multicollinearity 12](#_Toc181480338)

[3.2 Linear Regression 12](#_Toc181480339)

[3.2.1 Simple Linear Regression 12](#_Toc181480340)

[3.2.2 Multiple Linear Regression 13](#_Toc181480341)

[3.2.3 Univariate Linear Regression 13](#_Toc181480342)

[3.2.4 Multivariate Regression 13](#_Toc181480343)

[3.2.5 Assumption in Linear Regression 14](#_Toc181480344)

[3.3 Evaluation Metrics for Linear Regression 15](#_Toc181480345)

[3.4 Polynomial Regression 17](#_Toc181480346)

[3.4.1 Assumptions of Polynomial Regression: 17](#_Toc181480347)

[3.5 Regularization 18](#_Toc181480348)

[3.5.1 Ridge (L2) Regularization (Ridge Regression) 18](#_Toc181480349)

[3.5.2 Lasso (L1) Regularization (Lasso Regression) 19](#_Toc181480350)

[3.5.3 Why Lasso Regression creates sparsity (some coefficients becoming exactly zero) 19](#_Toc181480351)

[3.5.4 Elastic Net (L1 and L2) Regularization (Elastic Net Regression) 20](#_Toc181480352)

[3.6 Logistic Regression 20](#_Toc181480353)

# Introduction

Machine learning is a branch of artificial intelligence that enables algorithms to uncover hidden patterns within datasets, allowing them to make predictions on new, similar data without explicit programming for each task. Traditional machine learning combines data with statistical tools to predict outputs, yielding actionable insights. This technology finds applications in diverse fields such as image and speech recognition, natural language processing, recommendation systems, fraud detection, portfolio optimization, and automating tasks.

## Machine Learning lifecycle

1. Study the Problems
2. Data Collection
3. Data Preparation

* Data cleaning
* Data Transformation
* Explanatory Data Analysis and Feature Engineering
* Split the dataset for training and testing.

1. Model Selection
2. Model building and Training
3. Model Evaluation
4. Model Tuning
5. Deployment
6. Monitoring and Maintenance

## Type of Machine Learning

There are several types of machine learning, each with special characteristics and applications. Some of the main types of machine learning algorithms are as follows:

1. Supervised Machine Learning
2. Unsupervised Machine Learning
3. Semi-Supervised Machine Learning
4. Reinforcement Learning

### Supervised Machine Learning

**Supervised machine learning** is a fundamental approach within the broader field of machine learning and artificial intelligence. It involves training algorithms using labeled datasets, where each input is paired with the correct output. **Supervised learning** allows the algorithm to learn the mapping from inputs to outputs, enabling it to make predictions or decisions when presented with new, unseen data.

#### Types of Supervised Machine Learning

1. **Classification**

[**Classification**](https://www.geeksforgeeks.org/getting-started-with-classification/)**deals with predicting categorical target variables, which represent discrete classes or labels. For instance, classifying emails as spam or not spam, or predicting whether a patient has a high risk of heart disease. Classification algorithms learn to map the input features to one of the predefined classes.**

**Here are some classification algorithms:**

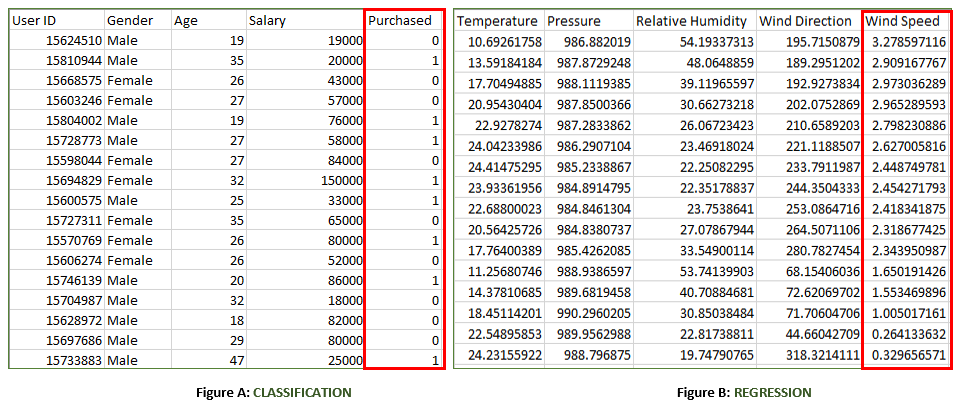
* [**Logistic Regression**](https://www.geeksforgeeks.org/understanding-logistic-regression/)
* [**Support Vector Machine**](https://www.geeksforgeeks.org/support-vector-machine-algorithm/)
* [**Random Forest**](https://www.geeksforgeeks.org/random-forest-regression-in-python/)
* [**Decision Tree**](https://www.geeksforgeeks.org/decision-tree/)
* [**K-Nearest Neighbors (KNN)**](https://www.geeksforgeeks.org/k-nearest-neighbours/)
* [**Naive Bayes**](https://www.geeksforgeeks.org/naive-bayes-classifiers/)

1. **Regression**

[**Regression**](https://www.geeksforgeeks.org/regression-classification-supervised-machine-learning/)**, on the other hand, deals with predicting continuous target variables, which represent numerical values. For example, predicting the price of a house based on its size, location, and amenities, or forecasting the sales of a product. Regression algorithms learn to map the input features to a continuous numerical value.**

**Here are some regression algorithms:**

* [**Linear Regression**](https://www.geeksforgeeks.org/ml-linear-regression/)
* [**Polynomial Regression**](https://www.geeksforgeeks.org/videos/polynomial-regression-algorithm-machine-learning/)
* [**Ridge Regression**](https://www.geeksforgeeks.org/videos/lasso-ridge-regression-algorithm-machine-learning/)
* [**Lasso Regression**](https://www.geeksforgeeks.org/videos/lasso-ridge-regression-algorithm-machine-learning/)
* [**Decision tree**](https://www.geeksforgeeks.org/decision-tree-introduction-example/)
* [**Random Forest**](https://www.geeksforgeeks.org/random-forest-regression-in-python/)



#### Supervised Machine Learning Algorithms

**Supervised learning** can be further divided into several different types, each with its own unique characteristics and applications. Here are some of the most common types of supervised learning algorithms:

* [**Linear Regression**](https://www.geeksforgeeks.org/ml-linear-regression/): Linear regression is a type of **supervised learning regression algorithm** that is used to predict a continuous output value. It is one of the simplest and most widely used algorithms in supervised learning.
* [**Logistic Regression**](https://www.geeksforgeeks.org/understanding-logistic-regression/): Logistic regression is a type of **supervised learning classification algorithm** that is used to predict a binary output variable. It is commonly used in machine learning applications where the output variable is either true or false, such as in fraud detection or spam filtering.
* [**Decision Trees**](https://www.geeksforgeeks.org/decision-tree/): Decision tree is a tree-like structure that is used to model decisions and their possible consequences. Each internal node in the [tree](https://www.geeksforgeeks.org/introduction-to-tree-data-structure-and-algorithm-tutorials/)represents a decision, while each leaf node represents a possible outcome. Decision trees can be used to model complex relationships between input features and output variables. A decision tree is a type of **machine learning algorithm** that is used for both classification and regression tasks.
* [**Random Forests**](https://www.geeksforgeeks.org/random-forest-regression-in-python/): Random forests again are made up of multiple decision trees that work together to make predictions. Each tree in the forest is trained on a different subset of the input features and data. The final prediction is made by aggregating the predictions of all the trees in the forest. Random forests are an ensemble **machine learning technique** that is used for both **classification and regression tasks in supervised learning**.
* [**Support Vector Machine(SVM)**](https://www.geeksforgeeks.org/support-vector-machine-algorithm/): The SVM algorithm creates a hyperplane to segregate n-dimensional space into classes and identify the correct category of new data points. The extreme cases that help create the hyperplane are called support vectors, hence the name Support Vector Machine. A Support Vector Machine is a type of supervised machine learning algorithm that is also used for both classification and regression tasks.
* [**K-Nearest Neighbors**](https://www.geeksforgeeks.org/k-nearest-neighbours/)**(KNN) :**KNN works by finding k training examples closest to a given input and then predicts the class or value based on the majority class or average value of these neighbors. The performance of KNN can be influenced by the choice of k and the distance metric used to measure proximity. However, it is intuitive but can be sensitive to noisy data and requires careful selection of k for optimal results. A K-Nearest Neighbors (KNN) is a type of algorithm that is used for both classification and regression tasks.
* [**Gradient Boosting**](https://www.geeksforgeeks.org/ml-gradient-boosting/): Gradient Boosting combines weak learners, like [decision trees](https://www.geeksforgeeks.org/decision-tree/), to create a strong model. It iteratively builds new models that correct errors made by previous ones. Each new model is trained to minimize residual errors, resulting in a powerful predictor capable of handling complex data relationships. A Gradient Boosting is a type of algorithm that is used for both classification and regression tasks.
* [**Naive Bayes Algorithm**](https://www.geeksforgeeks.org/naive-bayes-classifiers/): The **Naive Bayes algorithm** is a **supervised machine learning algorithm** based on applying [Bayes’ Theorem](https://www.geeksforgeeks.org/bayes-theorem/) with the “naive” assumption that features are independent of each other given the class label. Despite this simplifying assumption, Naive Bayes performs well for many real-world tasks, especially in text classification, spam detection, and document categorization.

### Unsupervised Learning

Unsupervised machine learning models, in contrast to [supervised learning](https://www.geeksforgeeks.org/supervised-machine-learning/), are given unlabelled data and allow discover patterns and insights on their own—without explicit direction or instruction.

#### Types of Unsupervised Machine Learning

There are mainly 3 types of Algorithms which are used for Unsupervised dataset.

* Clustering
* Association Rule Learning
* Dimensionality Reduction

1. **Clustering**

[Clustering](https://www.geeksforgeeks.org/clustering-in-machine-learning/) in unsupervised machine learning is the process of grouping unlabelled data into clusters based on their similarities. The goal of clustering is to identify patterns and relationships in the data without any prior knowledge of the data’s meaning.

Broadly this technique is applied to group data based on different patterns, such as similarities or differences, our machine model finds. These algorithms are used to process raw, unclassified data objects into groups. For example, in the above figure, we have not given output parameter values, so this technique will be used to group clients based on the input parameters provided by our data.

Some common clustering algorithms

* [K-means Clustering](https://www.geeksforgeeks.org/k-means-clustering-introduction/): Partitioning Data into K Clusters
* [Hierarchical Clustering](https://www.geeksforgeeks.org/ml-hierarchical-clustering-agglomerative-and-divisive-clustering/): Building a Hierarchical Structure of Clusters
* [Density-Based Clustering (DBSCAN)](https://www.geeksforgeeks.org/dbscan-clustering-in-ml-density-based-clustering/): Identifying Clusters Based on Density
* [Mean-Shift Clustering](https://www.geeksforgeeks.org/ml-mean-shift-clustering/): Finding Clusters Based on Mode Seeking
* [Spectral Clustering](https://www.geeksforgeeks.org/ml-spectral-clustering/): Utilizing Spectral Graph Theory for Clustering

1. **Association Rule Learning**

[Association rule learning](https://www.geeksforgeeks.org/association-rule/) is also known as association rule mining is a common technique used to discover associations in unsupervised machine learning. This technique is a rule-based ML technique that finds out some very useful relations between parameters of a large data set. This technique is basically used for market basket analysis that helps to better understand the relationship between different products. For e.g. shopping stores use algorithms based on this technique to find out the relationship between the sale of one product w.r.t to another’s sales based on customer behavior. Like if a customer buys milk, then he may also buy bread, eggs, or butter. Once trained well, such models can be used to increase their sales by planning different offers.

* [Apriori Algorithm](https://www.geeksforgeeks.org/apriori-algorithm/): A Classic Method for Rule Induction
* [FP-Growth Algorithm](https://www.geeksforgeeks.org/frequent-pattern-growth-algorithm/): An Efficient Alternative to Apriori
* [Eclat Algorithm](https://www.geeksforgeeks.org/ml-eclat-algorithm/): Exploiting Closed Item sets for Efficient Rule Mining
* [Efficient Tree-based Algorithms](https://www.geeksforgeeks.org/introduction-to-tree-data-structure-and-algorithm-tutorials/): Handling Large Datasets with Scalability

1. **Dimensionality Reduction**

Dimensionality reduction is the process of reducing the number of features in a dataset while preserving as much information as possible. This technique is useful for improving the performance of machine learning algorithms and for data visualization. Examples of dimensionality reduction algorithms include Dimensionality reduction is the process of reducing the number of features in a dataset while preserving as much information as possible.

* [Principal Component Analysis (PCA)](https://www.geeksforgeeks.org/principal-component-analysis-pca/): Linear Transformation for Reduced Dimensions
* [Linear Discriminant Analysis (LDA)](https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/): Dimensionality Reduction for Discrimination
* [Non-negative Matrix Factorization (NMF](https://www.geeksforgeeks.org/non-negative-matrix-factorization/)): Decomposing Data into Non-negative Components
* [Locally Linear Embedding (LLE)](https://www.geeksforgeeks.org/locally-linear-embedding-in-machine-learning/): Preserving Local Geometry in Reduced Dimensions
* Isomap: Capturing Global Relationships in Reduced Dimensions

### Semi-Supervised Learning

[Semi-Supervised learning](https://www.geeksforgeeks.org/ml-semi-supervised-learning/)is a machine learning algorithm that works between the [supervised and unsupervised](https://www.geeksforgeeks.org/supervised-unsupervised-learning/) learning so it uses both **labelled and unlabelled** data. It’s particularly useful when obtaining labelled data is costly, time-consuming, or resource-intensive. This approach is useful when the dataset is expensive and time-consuming. Semi-supervised learning is chosen when labelled data requires skills and relevant resources in order to train or learn from it.

We use these techniques when we are dealing with data that is a little bit labelled and the rest large portion of it is unlabelled. We can use the unsupervised techniques to predict labels and then feed these labels to supervised techniques. This technique is mostly applicable in the case of image data sets where usually all images are not labelled.

#### Types of Semi-Supervised Learning Methods

There are a number of different semi-supervised learning methods each with its own characteristics. Some of the most common ones include:

* **Graph-based semi-supervised learning:** This approach uses a graph to represent the relationships between the data points. The graph is then used to propagate labels from the labeled data points to the unlabeled data points.
* **Label propagation:** This approach iteratively propagates labels from the labeled data points to the unlabeled data points, based on the similarities between the data points.
* **Co-training:** This approach trains two different machine learning models on different subsets of the unlabeled data. The two models are then used to label each other’s predictions.
* **Self-training:** This approach trains a machine learning model on the labeled data and then uses the model to predict labels for the unlabeled data. The model is then retrained on the labeled data and the predicted labels for the unlabeled data.
* [**Generative adversarial networks (GANs)**](https://www.geeksforgeeks.org/generative-adversarial-network-gan/)**:** GANs are a type of deep learning algorithm that can be used to generate synthetic data. GANs can be used to generate unlabeled data for semi-supervised learning by training two neural networks, a generator and a discriminator.

### Reinforcement Machine Learning

[Reinforcement machine learning](https://www.geeksforgeeks.org/what-is-reinforcement-learning/)algorithm is a learning method that interacts with the environment by producing actions and discovering errors. **Trial, error, and delay** are the most relevant characteristics of reinforcement learning. In this technique, the model keeps on increasing its performance using Reward Feedback to learn the behavior or pattern. These algorithms are specific to a particular problem e.g. Google Self Driving car, AlphaGo where a bot competes with humans and even itself to get better and better performers in Go Game.

#### Reinforcement Learning Algorithms

* [**Q-learning:**](https://www.geeksforgeeks.org/q-learning-in-python/) Q-learning is a model-free RL algorithm that learns a Q-function, which maps states to actions. The Q-function estimates the expected reward of taking a particular action in a given state.
* [**SARSA (State-Action-Reward-State-Action):**](https://www.geeksforgeeks.org/sarsa-reinforcement-learning/) SARSA is another model-free RL algorithm that learns a Q-function. However, unlike Q-learning, SARSA updates the Q-function for the action that was actually taken, rather than the optimal action.
* [**Deep Q-learning**](https://www.geeksforgeeks.org/deep-q-learning/)**:** Deep Q-learning is a combination of Q-learning and deep learning. Deep Q-learning uses a neural network to represent the Q-function, which allows it to learn complex relationships between states and actions.

## Model-Based vs Instance-Based Learning

### Model-Based Learning

Model-based learning involves creating a mathematical model that can predict outcomes based on input data. The model is trained on a large dataset and then used to make predictions on new data. The model can be thought of as a set of rules that the machine uses to make predictions.

**Advantages of Model-Based Learning**

1. Faster predictions
2. More accurate predictions
3. Better understanding of data

**Disadvantages of Model-Based Learning**

1. Requires a large dataset
2. Requires expert knowledge

### Instance-Based Learning

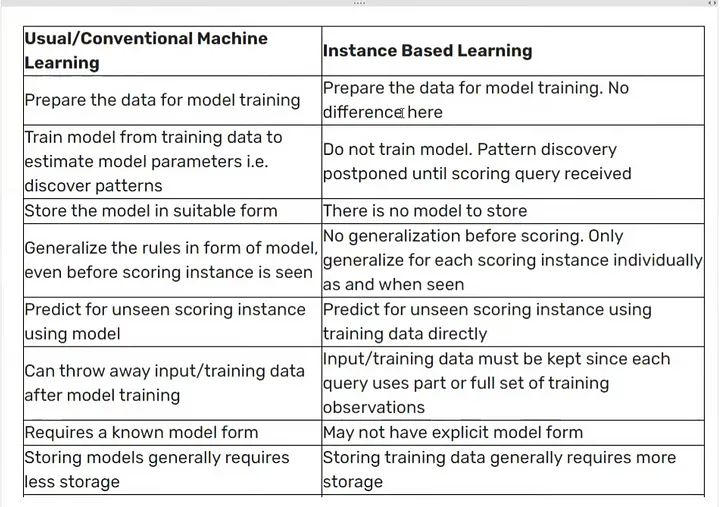
Instance-based learning involves using the entire dataset to make predictions. The machine learns by storing all instances of data and then using these instances to make predictions on new data. The machine compares the new data to the instances it has seen before and uses the closest match to make a prediction.

**Advantages of Instance-Based Learning**

1. No need for model creation
2. Can handle small datasets
3. More flexibility

**Disadvantages of Instance-Based Learning**

1. Slower predictions
2. Less accurate predictions
3. Limited understanding of data



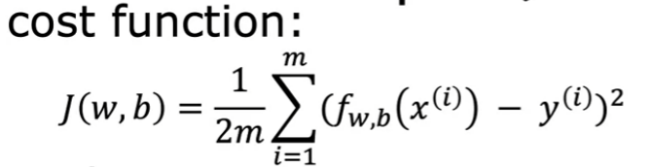
# Gradient Descent

A gradient is nothing but a derivative that defines the effects on outputs of the function with a little bit of variation in inputs.

Gradient Descent is a numerical optimization algorithm that aims to find the optimal parameters—weights and biases—of a neural network by minimizing a defined cost function.

## Cost Function

It is a function that measures the performance of a model for any given data. Cost Function quantifies the error between predicted values and expected values and presents it in the form of a single real number.



What is loss function?

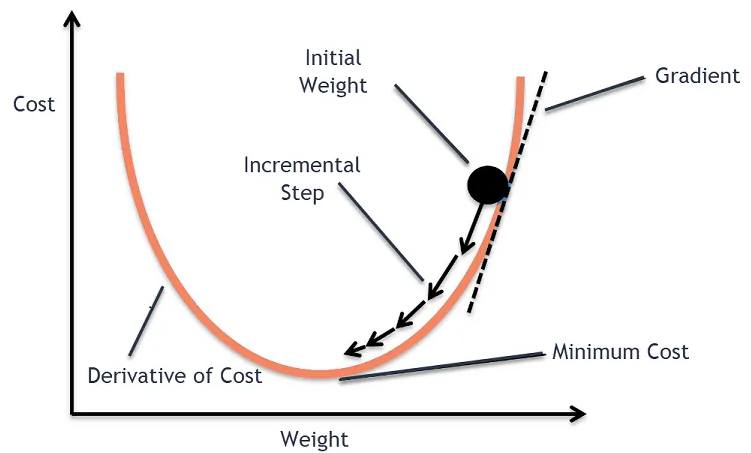
The loss function quantifies how much a model prediction y\_hat deviates from the ground truth y for one particular object So, when we calculate loss, we do it for a single object in the training or test sets.

Difference between cost and loss function?

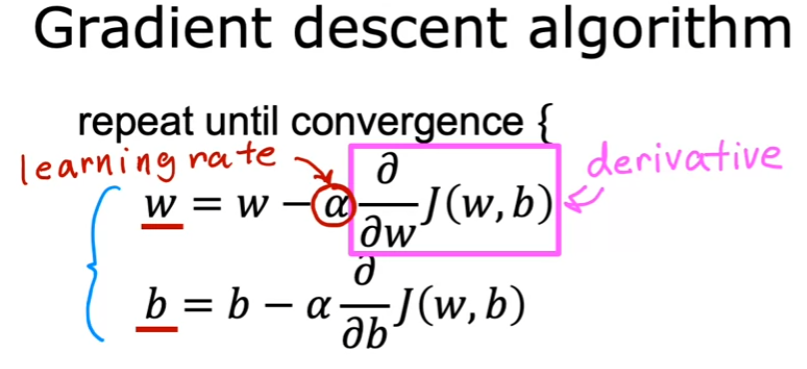
The loss functions are defined on a single training example. It means it measures how well your model performing on a single training example. But if we consider the entire training set and try to measure how well is our model performing on it, we define a function called the cost function.

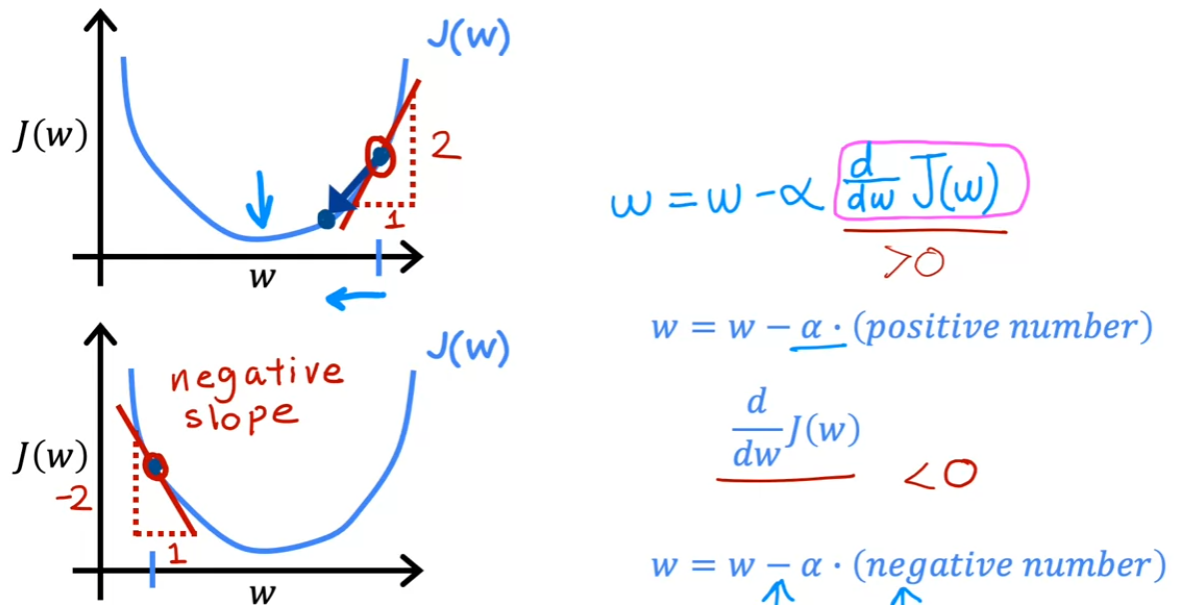
## Gradient Descent

Gradient Descent is a fundamental optimization algorithm in machine learning used to minimize the cost or loss function during model training.



Initially, we will assume values of w, b and will start moving to best fit w and b by subtracting derivative of cost function at assumed w, b (initial values) and we use alpha to converge slowly towards minima of cost function.





## Types of Gradient Descent Algorithm

The choice of gradient descent algorithm depends on the problem at hand and the size of the dataset. Batch gradient descent is suitable for small datasets, while stochastic gradient descent algorithm is more suitable for large datasets. Mini-batch is a good compromise between the two and is often used in practice.

1. **Batch Gradient Descent**

Batch gradient descent, also called vanilla gradient descent, calculates the error for each example within the training dataset, but it only gets updated after all training examples have been evaluated. This process is like a cycle and called a training epoch.

An advantage of batch gradient descent is its computational efficiency: it produces a stable error gradient and a stable convergence. But the stable error gradient can sometimes result in a state of convergence that isn’t the best the model can achieve. It also requires the entire training dataset to be in memory and available to the algorithm.

1. **Stochastic Gradient Descent**

By contrast, stochastic gradient descent (SGD) does this for each training example within the dataset, meaning it updates the parameters for each training example one by one. Depending on the problem, this can make SGD faster than batch gradient descent because SGD converges faster and take less no of epoch than BGD. One advantage is that frequent updates allow us to have a pretty detailed rate of improvement.

The trade-off is that these frequent updates can be more computationally expensive, especially in scenarios where each data point requires a lot of computation. Additionally, since SGD doesn’t use the full dataset for each update, the gradient may "jump around" more, leading to higher variance in the loss function and potentially more oscillations as it converges.

This noisy behaviour of SGD can sometimes be beneficial, as it helps escape local minima, but it also makes it less stable than BGD.

1. **Mini-Batch Gradient Descent**

Mini-batch gradient descent is the go-to method since it’s a combination of the concepts of SGD and batch gradient descent. It simply splits the training dataset into small batches and performs an update for each of those batches. This creates a balance between the robustness of stochastic gradient descent and the efficiency of batch gradient descent.

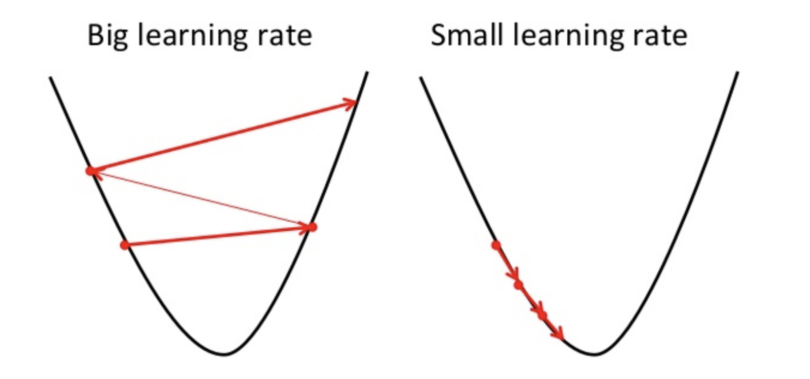
Common mini-batch sizes range between 50 and 256, but like any other machine learning technique, there is no clear rule because it varies for different applications. This is the go-to algorithm when training a neural network, and it’s the most common type of gradient descent within deep learning.

## Gradient Descent Learning Rate

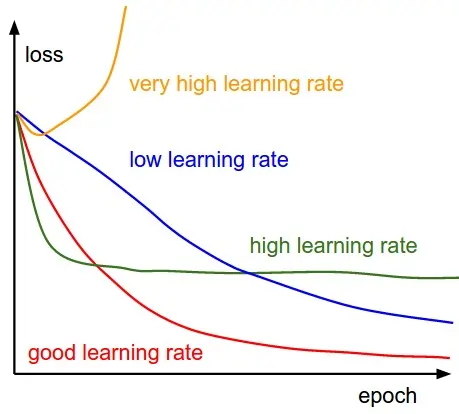
We have the direction we want to move in. Now, we must decide the size of the step we must take.

**\*It must be chosen carefully to end up with local minima.**

* If the learning rate is too high, we might OVERSHOOT the minima and keep bouncing without reaching the minima
* If the learning rate is too small, it takes more time but converges to the minimum and the training might turn out to be too long
* The learning rate is optimal, and the model converges to the minimum.



* The learning rate is higher than the optimal value. It overshoots.
* The learning rate is very large. It overshoots and diverges, moves away from the minima, and performance decreases in learning.



## Learning Rate Schedule

A learning rate schedule is a strategy in machine learning used to adjust the learning rate during training to improve convergence and model performance. The learning rate, which controls the step size when updating weights, plays a crucial role in how quickly and accurately a model learns. With a learning schedule, the learning rate changes over time based on specific rules, often helping to balance convergence speed and stability.

# Supervised Machine Learning Algorithms

## Multicollinearity

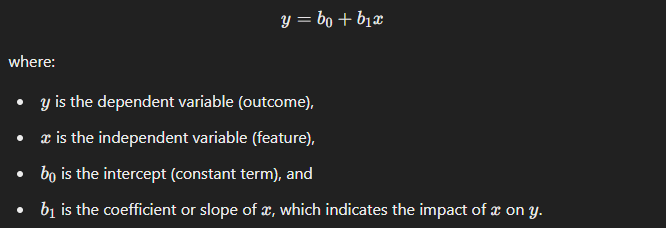
Multicollinearity is a statistical phenomenon that occurs when two or more independent variables in a regression model are highly correlated, indicating a strong linear relationship among the predictor variables. This issue complicates regression analysis by making it difficult to accurately determine the individual effects of each independent variable on the dependent variable.

## Linear Regression

Linear regression is a type of supervised machine learning algorithm that computes the linear relationship between the dependent variable and one or more independent features by fitting a linear equation to observed data.

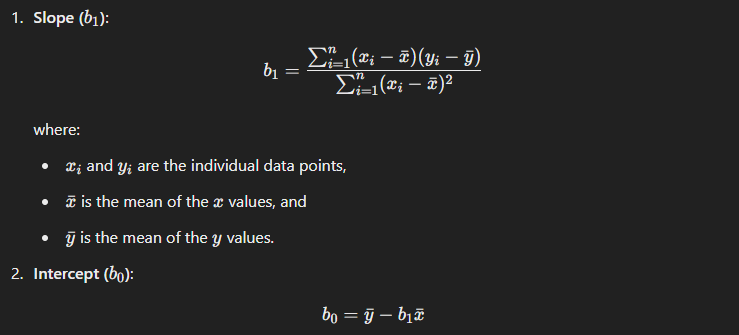
### Simple Linear Regression

A regression technique where we predict the outcome based on only one independent variable (or feature).



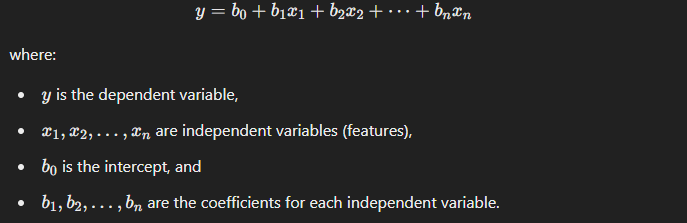
* **Formula for Weights (Coefficients)**

Using the Ordinary Least Squares (OLS) method, the goal is to find the weights (also known as coefficients) that minimize the cost function (sum of the squared errors between the predicted values and the actual values).

****

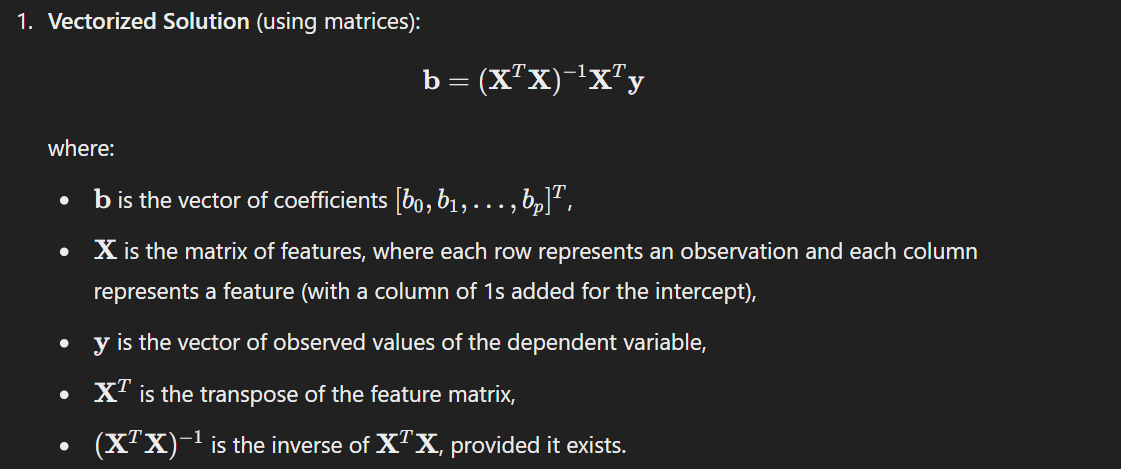
### Multiple Linear Regression

A regression technique where we predict the outcome based on two or more independent variables.



* **Formula for Weights (Coefficients) in Linear Regression Using OLS**

Using Ordinary Least Squares (OLS), we estimate the weights that minimize the sum of squared differences between the actual and predicted values. The weight vector (including the intercept) can be computed using the following matrix formula:



### Univariate Linear Regression

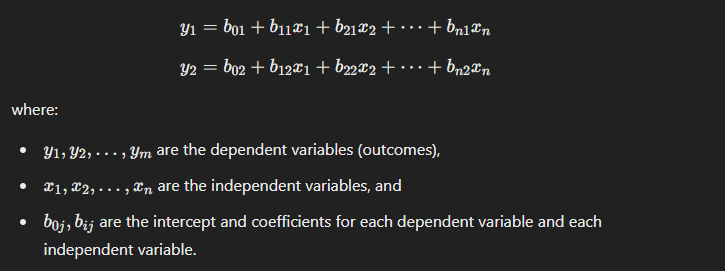
A regression where there is only one dependent variable.

**Example**: Predicting a single outcome, like "salary," based on one or multiple independent variables (such as years of experience and education level).

### Multivariate Regression

A regression where there are two or more dependent variables (outcomes).

**Example**: Suppose we want to predict both a person’s "salary" and "job satisfaction level" based on independent variables like years of experience, education level, and hours worked. Here, we are predicting multiple outcomes using potentially multiple features.



### Assumption in Linear Regression

* The Independent variables should be linearly related to the dependent variables.

This can be examined with the help of several visualization techniques like: Scatter plot or maybe you can use Heatmap or pair plot (to visualize every feature in the data in one particular plot).

* Every feature in the data is Normally Distributed.

This again can be checked with the help of different visualization Techniques, such as Q-Q plot, histogram and much more.

* There should be little or no multi-collinearity in the data.

The best way to check the Prescence of multi-collinearity is to perform VIF (Variance Inflation Factor).

* The mean of the residual is zero.

A residual is the difference between the observed y-value and the predicted y-value. However, having residuals closer to zero means the model is doing great.

* Residuals obtained should be normally distributed.

This can be verified using the Q-Q Plot on the residuals.

* Variance of the residual throughout the data should be same. This is known as homoscedasticity.

This can be checked with the help of residual vs fitted plot.

* There should be little or no Auto-Correlation is the data.

Auto-Correlation Occurs when the residuals are not independent of each other. This usually takes place in time series analysis.

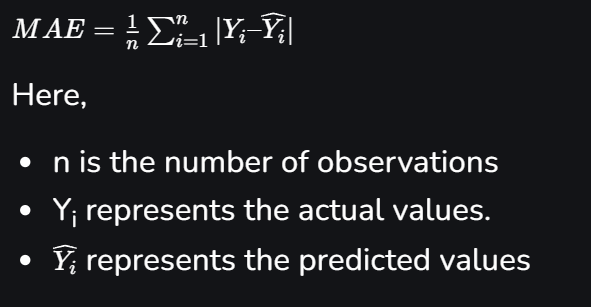
You can perform Durbin-Watson test or plot ACF plot to check for the autocorrelation. If the value of Durbin-Watson test is 2 then that means no autocorrelation, if value < 2 then there is positive correlation and if the value is between >2 to 4 then there is negative autocorrelation.

## Evaluation Metrics for Linear Regression

1. **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:

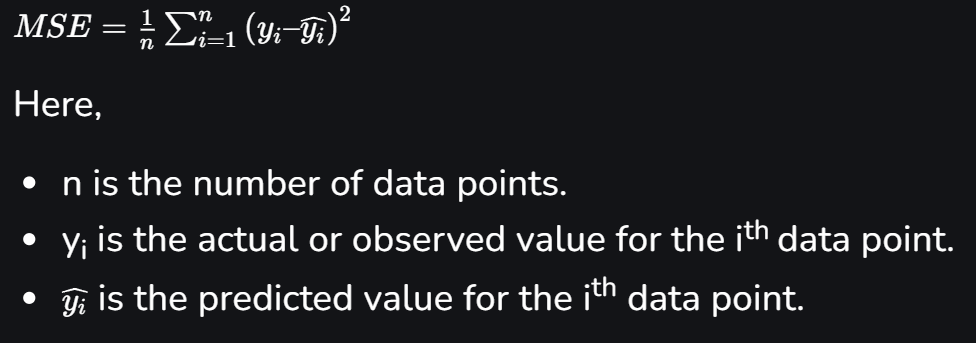


Lower MAE value indicates better model performance. It is not sensitive to the outliers as we consider absolute differences.

It is not used a lost function to calculate Gradient Descent because it is not differentiable at x=0.

1. **Mean Square Error (MSE)**

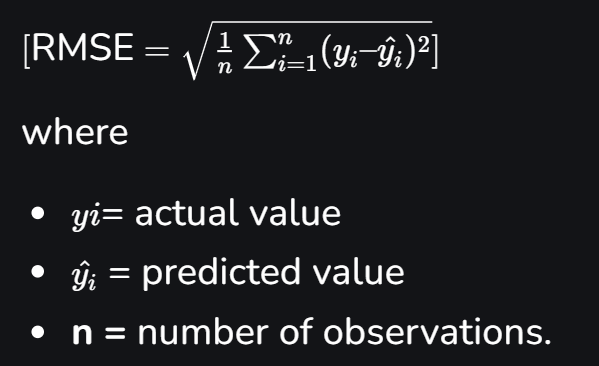
[Mean Squared Error (MSE)](https://www.geeksforgeeks.org/python-mean-squared-error/) is an evaluation metric that calculates the average of the squared differences between the actual and predicted values for all the data points. The difference is squared to ensure that negative and positive differences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



MSE 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.

1. **Root Mean Square Error**

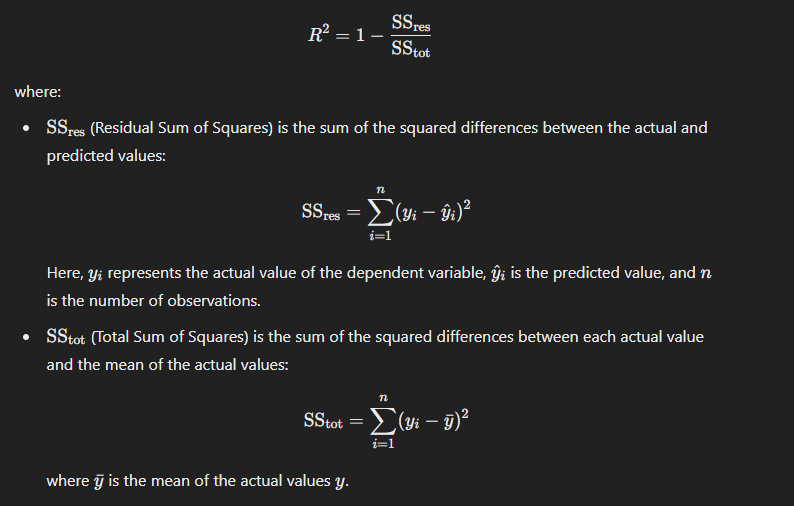
Root mean squared error (RMSE) is the square root of the mean of the square of all of the errors. RMSE is considered an excellent general-purpose error metric for numerical predictions. RMSE is a good measure of accuracy, but only to compare prediction errors of different models or model configurations for a particular variable and not between variables, as it is scale-dependent. It is the measure of how well a regression line fits the data points. The formula for calculating RMSE is:



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

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:

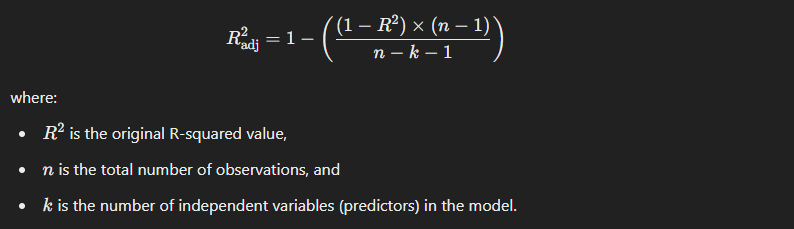


**Limitation:**

**R-squared increases or stays the same when additional independent variables are added**, regardless of whether they are actually relevant. This can give a false sense of model improvement even if the new variables don’t actually contribute to predictive power.

1. **Adjusted R-Squared Error**

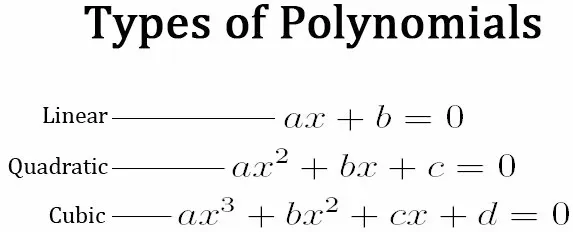
Adjusted R-squared compensates for this by adjusting the R-squared value based on the number of predictors and the sample size. It increases only if the newly added variable improves the model more than would be expected by chance.



Unlike R-squared, Adjusted R-squared can decrease if an added predictor does not improve the model significantly. This penalization helps us avoid overfitting and select the model with only meaningful predictors.

## Polynomial Regression

Polynomial Regression is a form of regression analysis in which the relationship between the independent variables and dependent variables are modelled in the nth degree polynomial.



Polynomial Regression does not require the relationship between the independent and dependent variables to be linear in the data set, this is also one of the main differences between the Linear and Polynomial Regression.

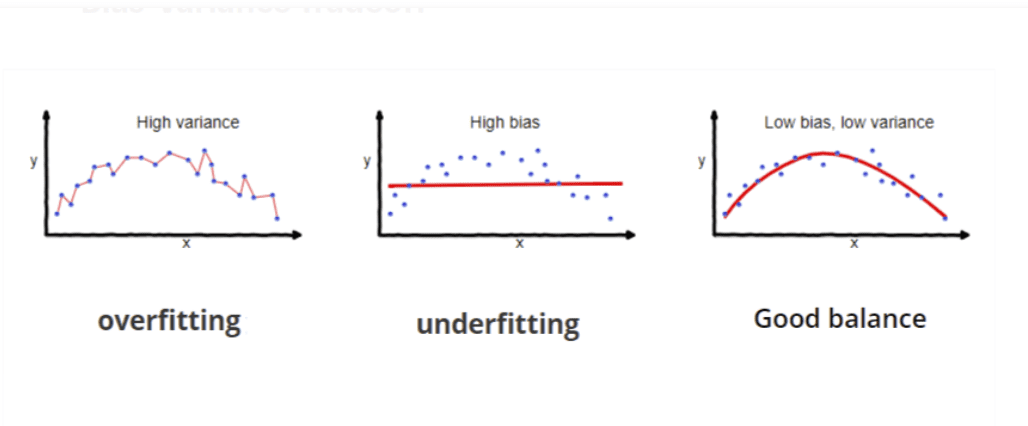
**As we increase the degree in the model, it tends to increase the performance of the model. However, increasing the degrees of the model also increases the risk of over-fitting and under-fitting the data.**

### Assumptions of Polynomial Regression:

* The behaviour of a dependent variable can be explained by a linear, or curvilinear, additive relationship between the dependent variable and a set of k independent variables (xi, i=1 to k).
* The relationship between the dependent variable and any independent variable is linear or curvilinear (specifically polynomial).
* The independent variables are independent of each other.
* The errors are independent, normally distributed with mean zero and a constant variance (OLS).

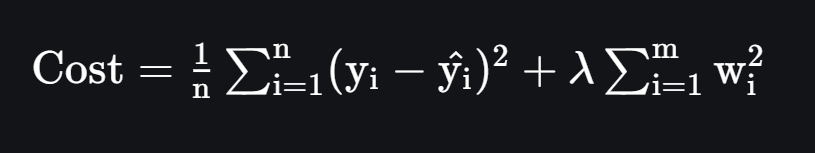
## Regularization

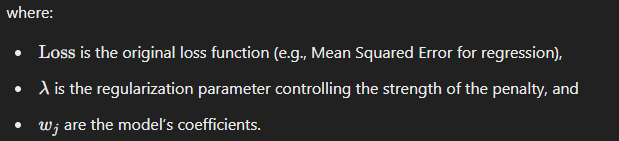
Regularization is a technique used in machine learning to prevent overfitting and improve the generalization performance of models. In essence, regularization adds a penalty term to the loss function, discouraging the model from learning overly complex patterns that may not generalize well to unseen data. This helps create simpler, more robust models.



### Ridge (L2) Regularization (Ridge Regression)

Ridge (or L2) regularization adds a penalty proportional to the square of the magnitude of coefficients. Unlike Lasso, Ridge regularization shrinks coefficients towards zero but does not make them exactly zero.





* **When to use**

Use Ridge when all features likely contribute to the outcome, especially if some of them are correlated (similar information). Ridge keeps all features but reduces their impact to avoid overfitting.

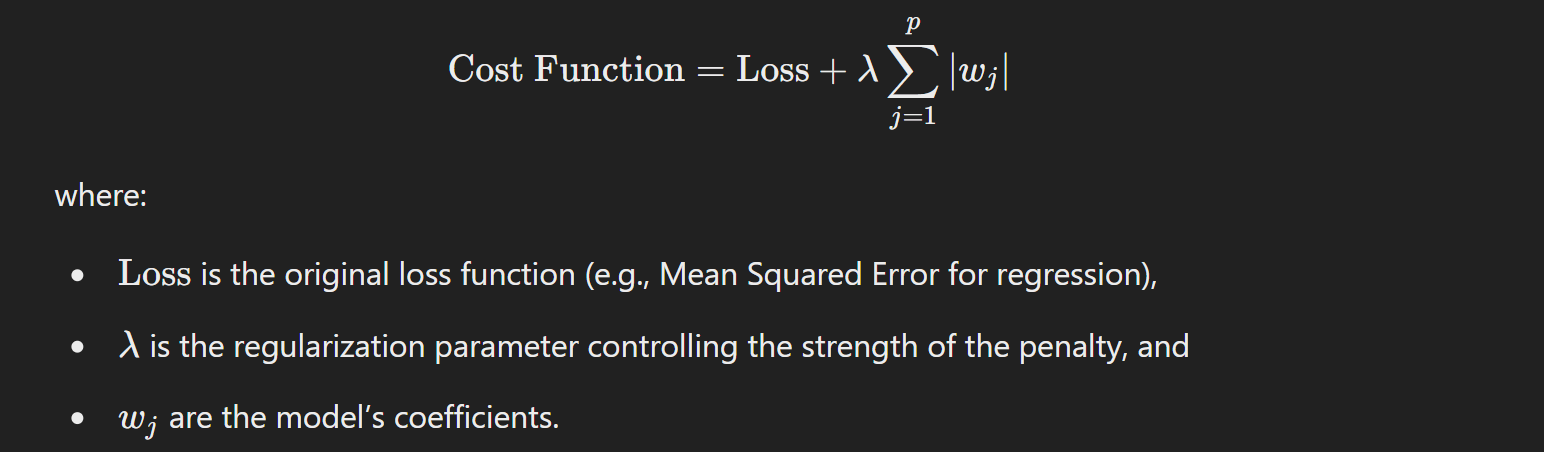
Good for: Stabilizing the model and making it less sensitive to small variations in data.

* **Effect of Lamda**

1. Lamda inc (0 – inf) => Weights dec (Original value – close to 0 but never 0).
2. Lamda inc => High value weights dec faster than low values weights.
3. Lamda inc => Bias inc and Variance dec.
4. Lamda inc => Loss function is parabolic for linear regression and loss function shrink and move towards 0 but not 0.

### Lasso (L1) Regularization (Lasso Regression)

Lasso (Least Absolute Shrinkage and Selection Operator) regularization, or L1 regularization, adds a penalty equal to the absolute value of the magnitude of coefficients. It encourages the model to reduce some of the coefficients to exactly zero, effectively performing feature selection by eliminating less important features.



* **When to use**

Use Lasso when you think only a few features are important, and you want the model to automatically select the best ones by shrinking irrelevant features' coefficients to zero.

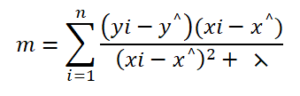
Good for: Feature selection. It gives you a simpler, easier-to-interpret model by focusing only on the most important features.

* **Effect of Lamda**

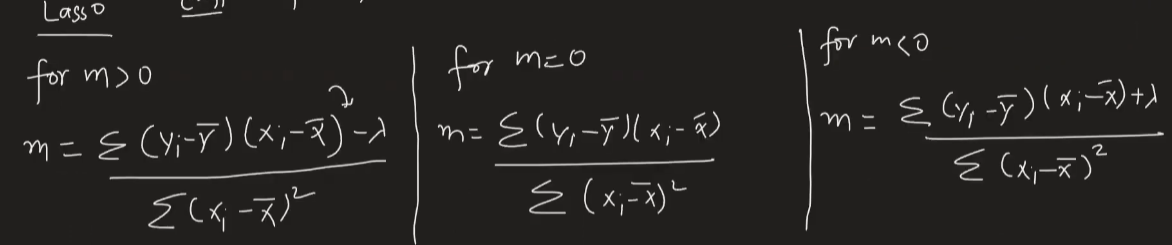
1. Lamda inc (0 – inf) => Weights dec (Original value – 0).
2. Lamda inc => High value weights dec faster than low values weights and for very high lamda all move towards zero and feature selection will happen only for middle value of lamda.
3. Lamda inc => Bias inc and Variance dec.
4. Lamda inc => Loss function is parabolic for linear regression and loss function shrink and move towards 0 and becomes 0 for large value of lamda.

### Why Lasso Regression creates sparsity (some coefficients becoming exactly zero)

* Formula for weight in case of Simple Ridge Regression is given below and as you can see lamda is in denominator which is why m will never be zero but can reach close to zero.

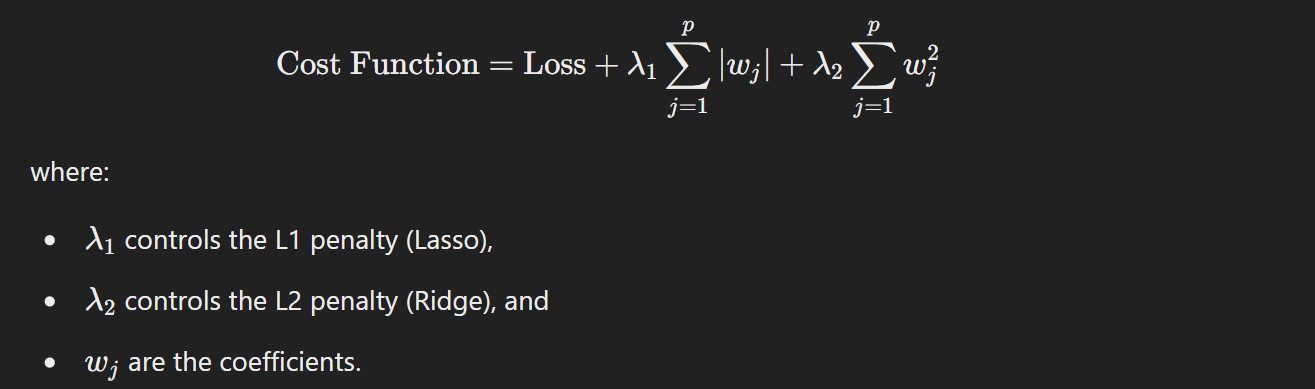


* Formula for weight in case of Simple Lasso regression is mention below and as you can see lamda is in numerator which will make m equal to 0 but not less than when lamda greater than or equal to
* Lasso regression creates sparsity in the dataset as there is a lambda term added or subtracted for different values of slope m and when the value of m reached zero but does not cross zero value.



### Elastic Net (L1 and L2) Regularization (Elastic Net Regression)

Elastic Net combines both L1 and L2 penalties, capturing the benefits of both regularizations. It adds both absolute and squared coefficient penalties, balancing the feature selection of Lasso with the stability of Ridge.



* **When to use**

Use Elastic Net when you have many features, some of which are important but also correlated. Elastic Net combines the strengths of Lasso (feature selection) and Ridge (keeping useful correlated features).

Good for: Complex datasets where you need both feature selection and stability.

## Logistic Regression