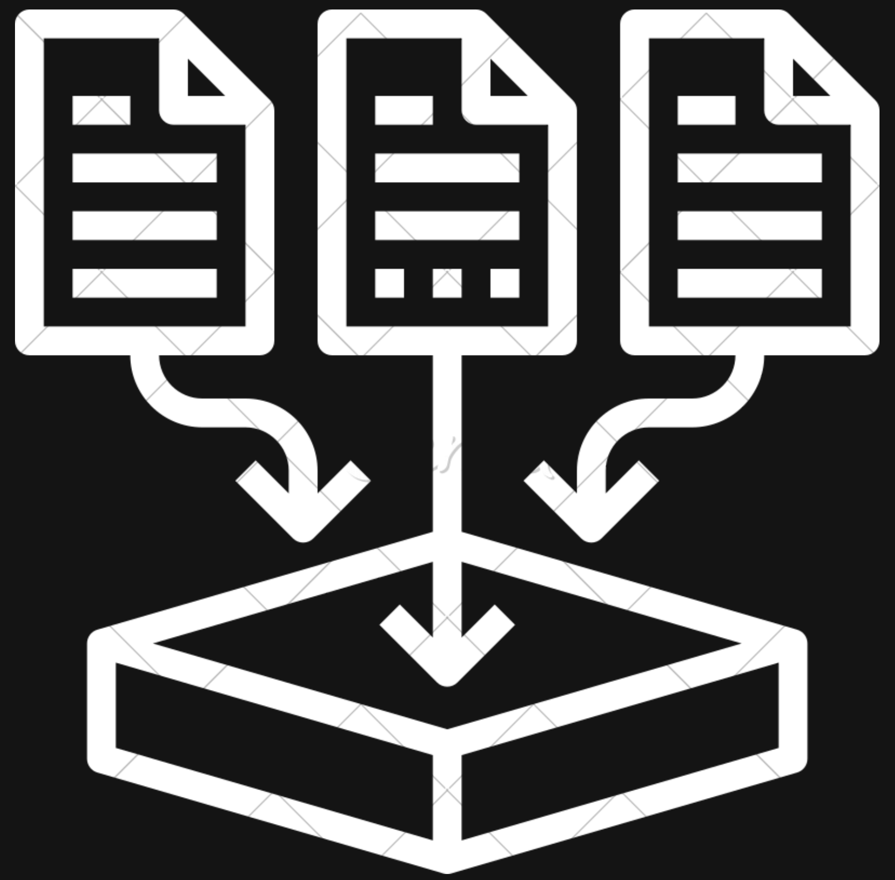
Intro to ML
Regression Models

But what is Machine learning?

Machine Learning

Instead of giving a computer a set of rules to follow, we provide it with a large amount of data and let it discover patterns and relationships within that data.



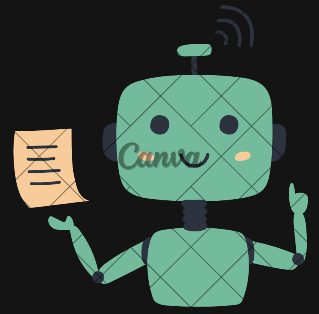
Content

- Applications of ML
- Types of ML
    - Supervised Learning
    - Unsupervised Learning
    - Reinforcement Learning
- Linear Regression
- Polynomial Regression

Applications of ML

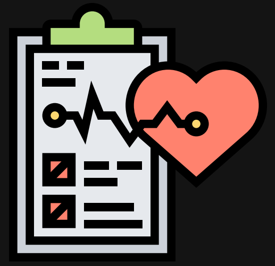
The applications of ML are limitless and increasing day by day , venturing into newer fields ! Some of them are -

Text generators (ChatGPT)



Virtual personal assistant

Diagnosis using Image detection



Autonomous Vehicles

Spam Detection

Speech 2 Text

Stock Market trading

And a loooooot more !!!

TYPES OF ML

The study of ML is broadly classified into three main branches based on the type of data and the actions to be performed on them. They are:

1. **Supervised Learning**
2. **Unsupervised Learning**
3. **Reinforcement Learning**

Supervised Learning

Learning from labeled examples to make predictions or classify new, unseen data.

Unsupervised Learning

Learning from unlabeled data to discover patterns, structures, or relationships without specific guidance or predefined labels.

Reinforcement Learning

Sequential decisions through trial and error, interacting with an environment and receiving rewards or penalties based on its actions.

Supervised Learning

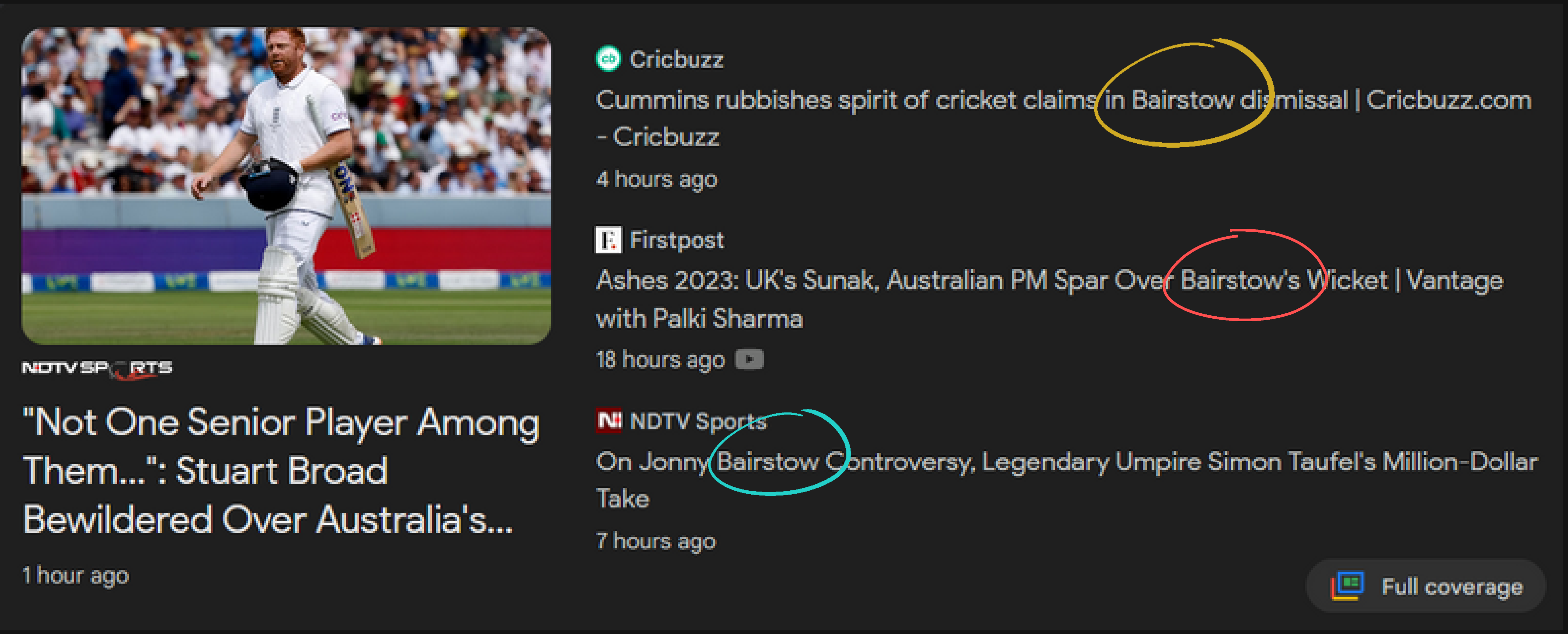
Suppose you are given a basket filled with different kinds of fruits. Now the first step is to train the machine with all the different fruits:

- If the shape of the object is rounded and has a depression at the top, is red in color, then it will be labeled as –Apple.
- If the shape of the object is a long curving cylinder having Green-Yellow color, then it will be labeled as –Banana.

Now suppose after training the data, you have given a new separate fruit, say Banana from the basket, and asked to identify it.
Since the machine has already learned the things from previous data and this time has to use it wisely. It will first classify the fruit with its shape and color and would confirm the fruit name as BANANA and put it in the Banana category.

Unsupervised Learning

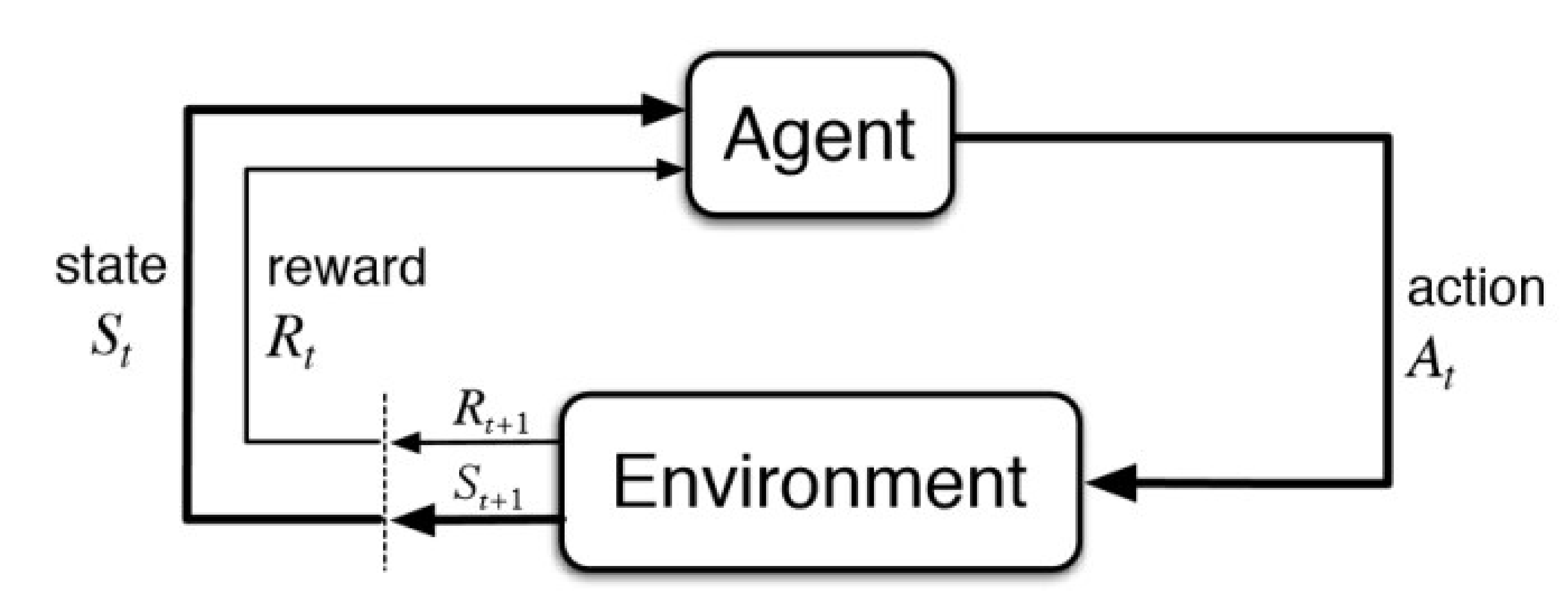
Unsupervised learning is the training of machine using information that is neither classified nor labelled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data.



Google news is a good example of Unsupervised learning. It scouts the internet for news articles which have similar key words/ news content
(The wicket of Jonny Bairstow in this case) . It then clubs together all these articles and presents to you

Reinforcement Learning

Reinforcement learning is a machine learning training method based on rewarding desired behaviors and/or punishing undesired ones. In general, a reinforcement learning agent is able to perceive and interpret its environment, take actions and learn through trial and error.



One good application is a home cleaning robot, where the robot learns optimal cleaning strategies by interacting with the environment, receiving feedback on their cleaning performance, and adjusting their actions to maximize cleaning efficiency and coverage. This allows the robots to adapt and improve over time, resulting in more effective and autonomous cleaning processes in our homes.

slido

Join at

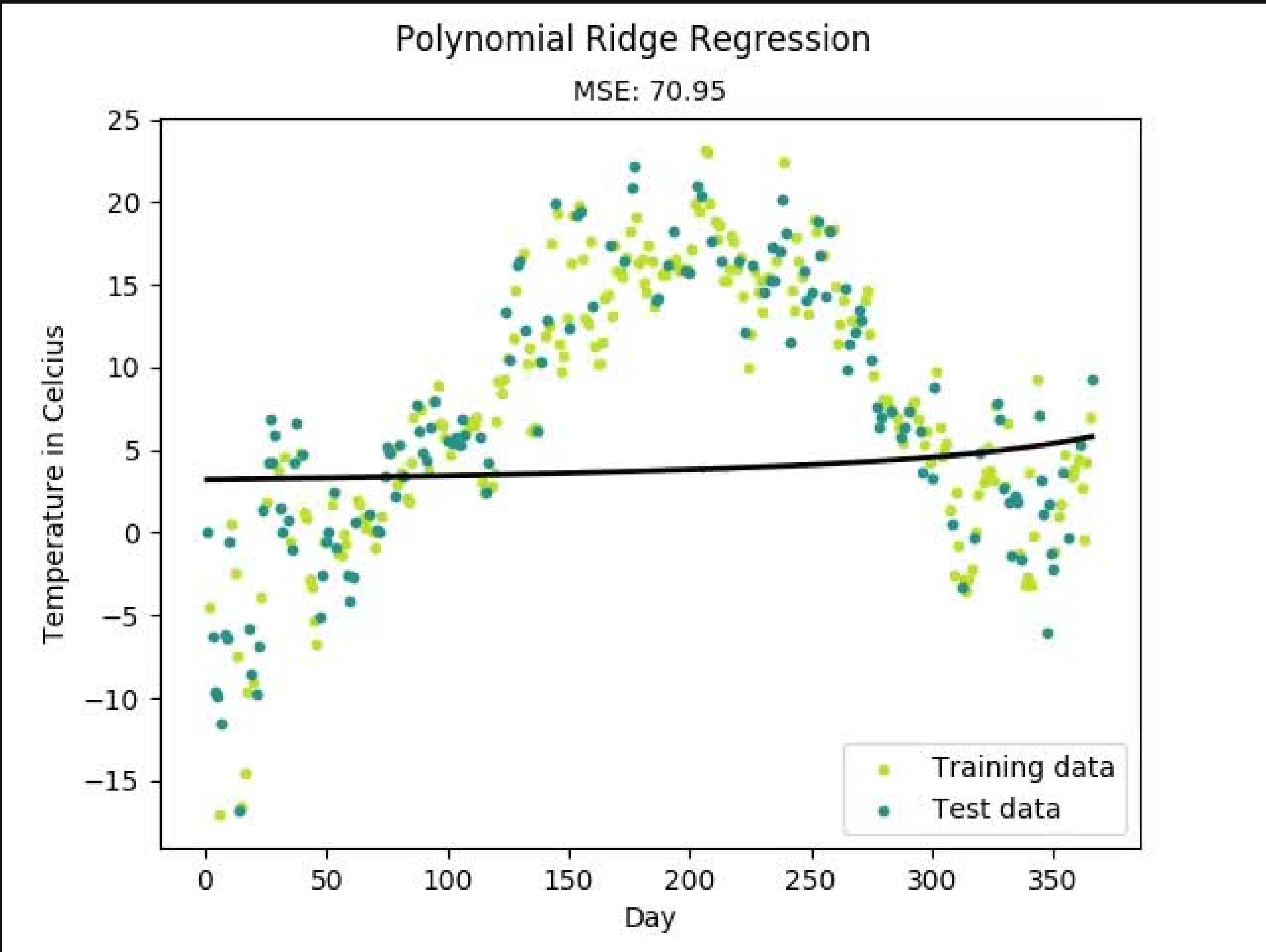
slido.com

#2630 823

Linear Regression

Regression is a type of supervised learning in ML that helps in mapping a predictive relationship between labels and data points. We try to predict a **continuous valued output**, where we try to map the input variables to some continuous function. The response variable takes continuous values here.

For example, if we want to predict housing prices based on the characteristics in the data given to us, we're dealing with a regression problem.



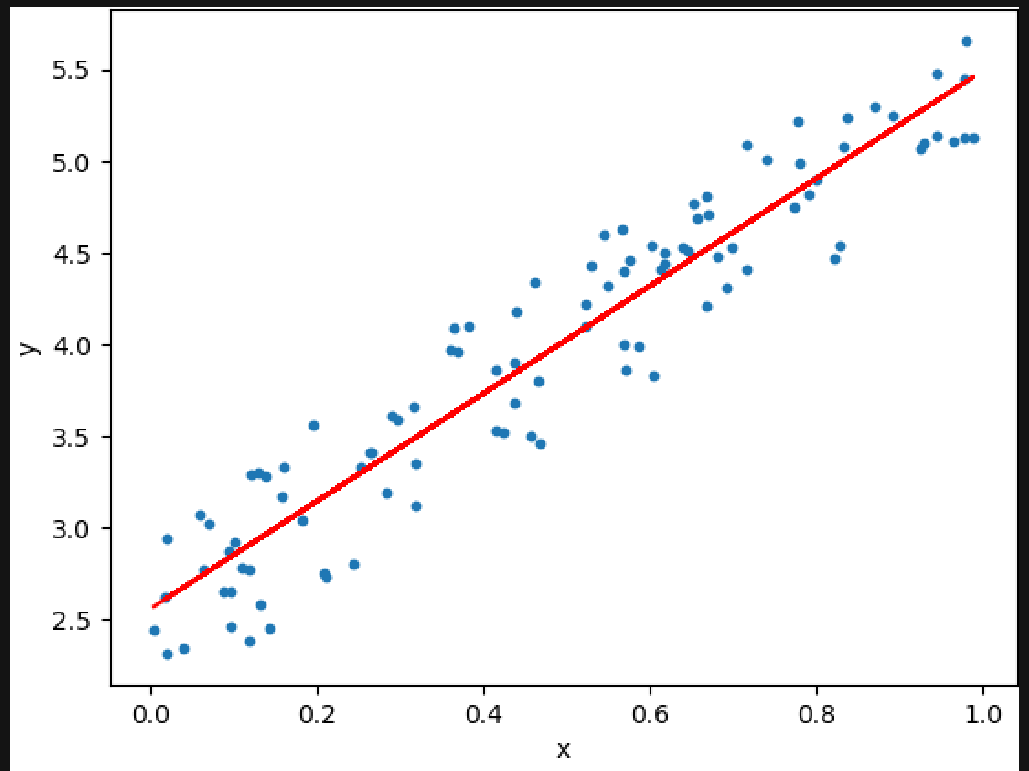
What is Regression?

LINEAR REGRESSION

Linear regression attempts to model the relationship between features by fitting a linear equation to observed data.

$$\hat{y} = wx + b$$

Where x is the input feature and w is the corresponding weight . b represents the constant term , also called the 'bias' term and $\hat{y}$ is the predicted value



Cost Function

In the best fit, we try minimizing the error between the predicted and true values. This is the criterion used to evaluate how well a model fits the data, and this criterion can be modelled as the cost function. Cost function determines the deviation from true values, we find and such that this function is minimum

Different types of Cost functions are employed depending on the model of ML used such as:

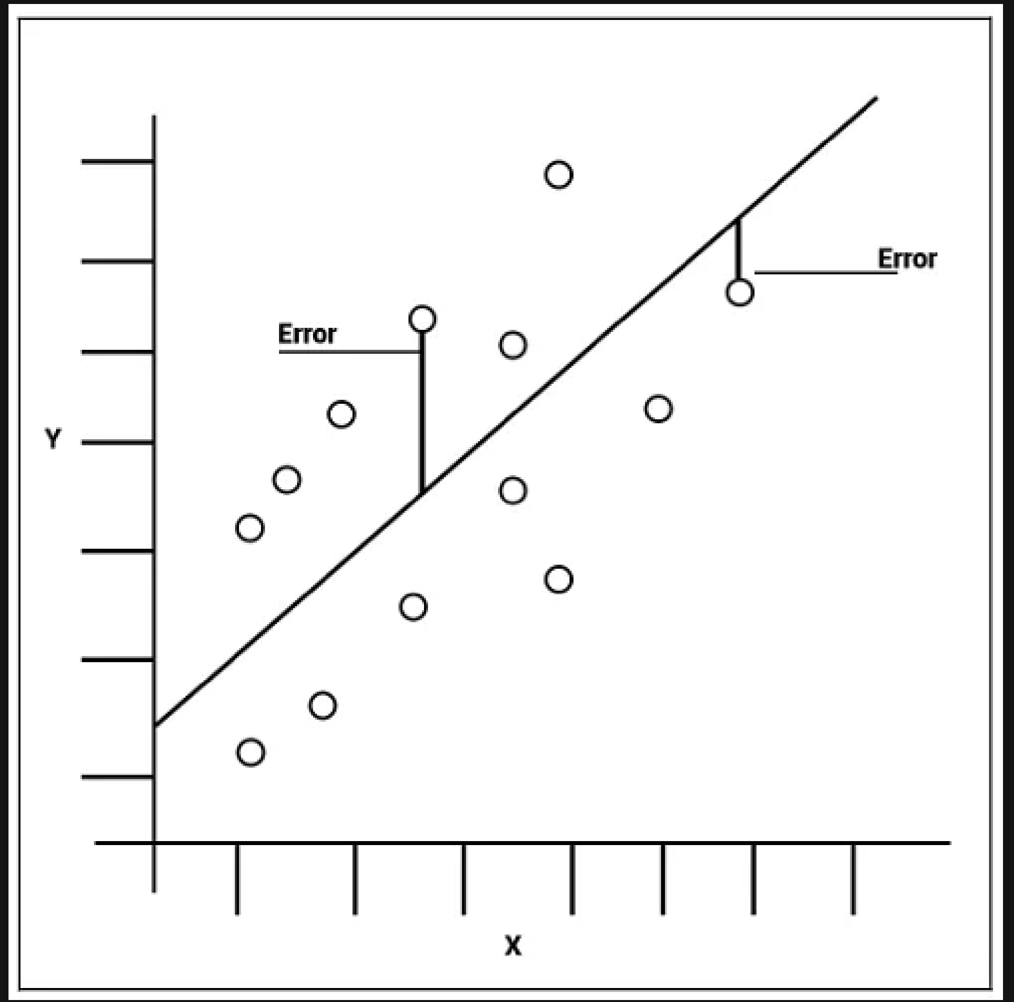
- Mean Squared Error
- Mean Average Error
- Binary Cross Entropy (Used in classification)
- Categorical Cross entropy(Used in classification)

Linear Regression mostly involves implementing MSE

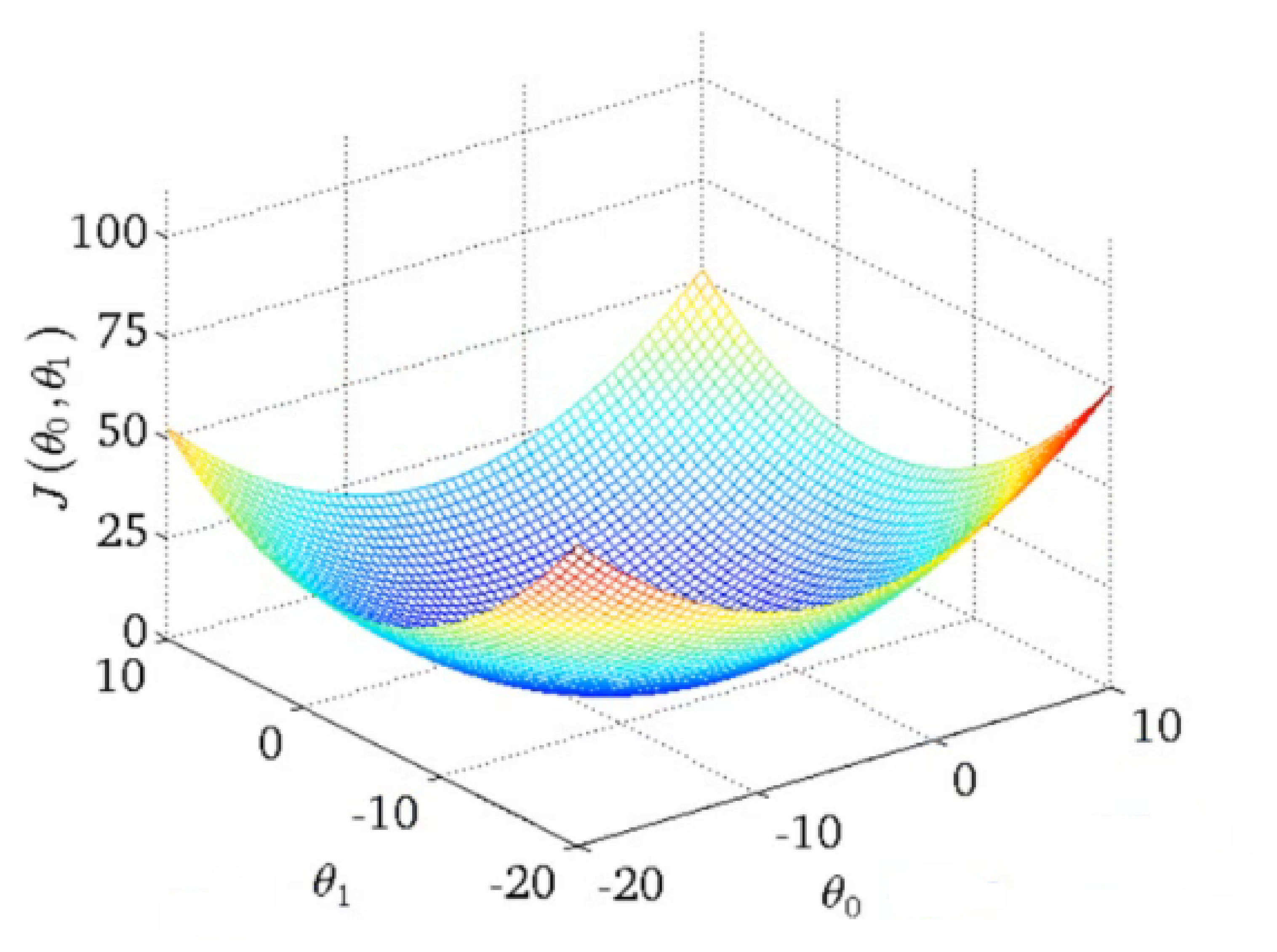
Mean Square Error

$$J(w,b) = \frac{1}{2m}\sum_{i=1}^{m}(\hat{y}_i - y_i)^2$$

- $J(w,b)$ represents the cost function (MSE) with respect to the parameters w and b.
- $\hat{y}_i$ is the hypothesis or predicted value for the i-th training example.
- y_i is the actual target value for the i-th training example.
- m is the number of training examples.



Cost Function Plotted (MSE)



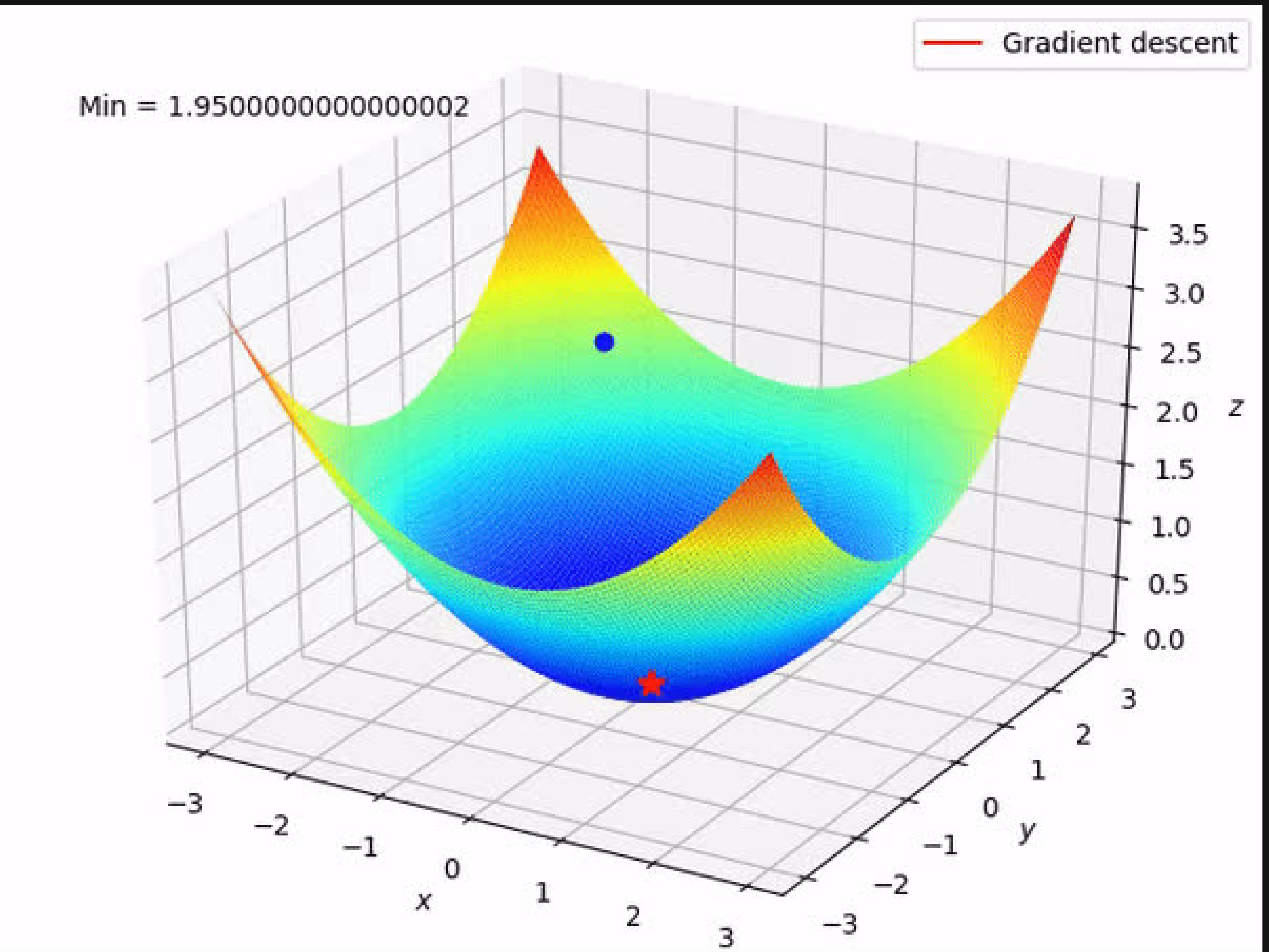
This graph is used to determine w and b which minimize the Cost function thereby giving the best fit line .

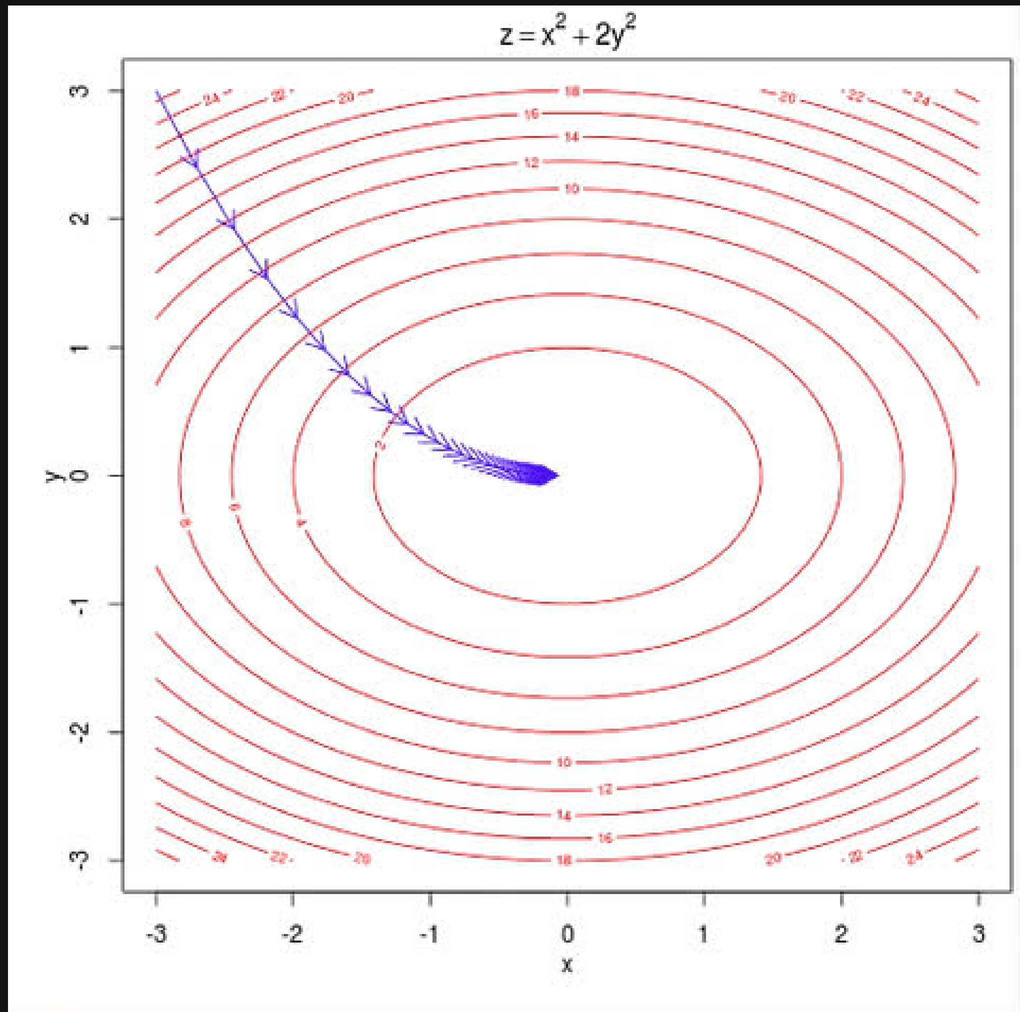
The MSE always is a convex function with one global minima which is extremely beneficial while implementing **Gradient Descent Algorithm**. which is used to determine the minima

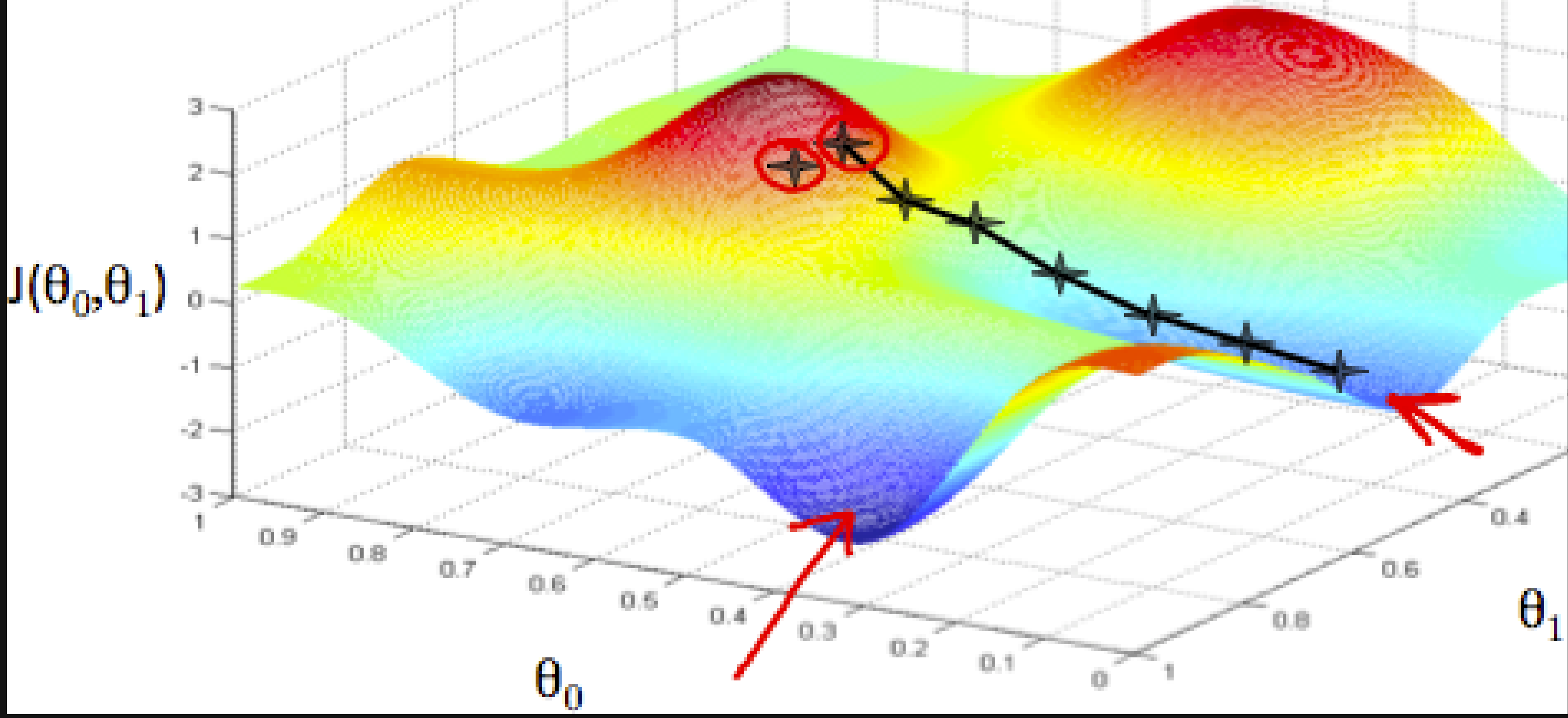
Gradient Descent Algorithm

Gradient Descent is an iterative optimization algorithm to find the minimum of a function

The way we do this is by taking the derivative of our cost function. The slope of the tangent is the derivative at that point and it will give us a direction to move towards. We make steps down the cost function in the direction with the steepest descent. The size of each step is determined by the parameter α, which is called the learning rate.







As we move closer and closer to the minima the slope would decrease (so α can be kept constant) and the person would take smaller and smaller steps. Hence, we can't reach the exact minimum but almost get there (closest possible).

Algorithm Implementation

$$w = w - \alpha \frac{\partial J(w,b)}{\partial w}$$

$$b = b - \alpha \frac{\partial J(w,b)}{\partial b}$$

$$\frac{\partial J(w,b)}{\partial w} = \frac{1}{m} \sum_{i=1}^{m} (\hat{y}_i - y_i) x_i$$

$$\frac{\partial J(w,b)}{\partial b} = \frac{1}{m} \sum_{i=1}^{m} (\hat{y}_i - y_i)$$

$$w = w - \alpha \tfrac{1}{m} \textstyle\sum_{i=1}^{m} ((w x_i + b) - y_i) x_i$$

$$b = b - \alpha \tfrac{1}{m} \textstyle\sum_{i=1}^{m} ((w x_i + b) - y_i)$$

Repeat till convergence

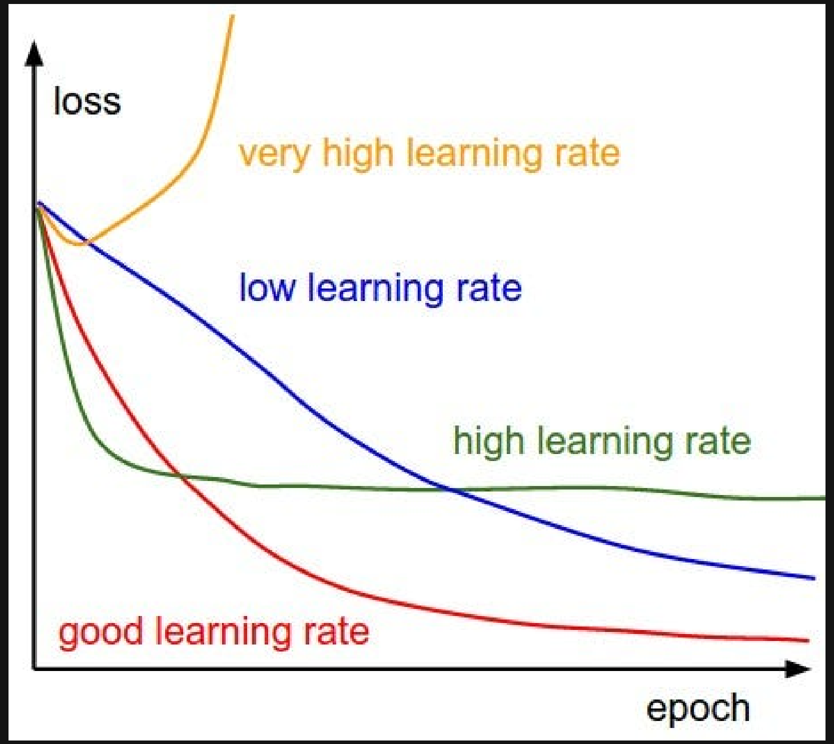
Word of caution , both w and b need to be updated **simultaneously** before the next iteration of cost function is calculated

The Learning Rate α

The learning rate is a hyperparameter in machine learning algorithms that determines the step size or the rate at which the model learns from the training data during the optimization process. It controls how quickly or slowly the model parameters are updated during gradient descent or other optimization algorithms.

Small α

A small **α** will result in a very slow convergence to minima



Large α

A Large **α** might lead to overshooting the minima and the cost function actually starts to uncrease after every iteration!!

A good way to determine an optimum α is to plot the cost function with time . If it is increasing , learning rate should be reduced and if the convergence is very slow , then it can be increased to achieve quicker convergence

We'll move to

Code Implementation .

slido

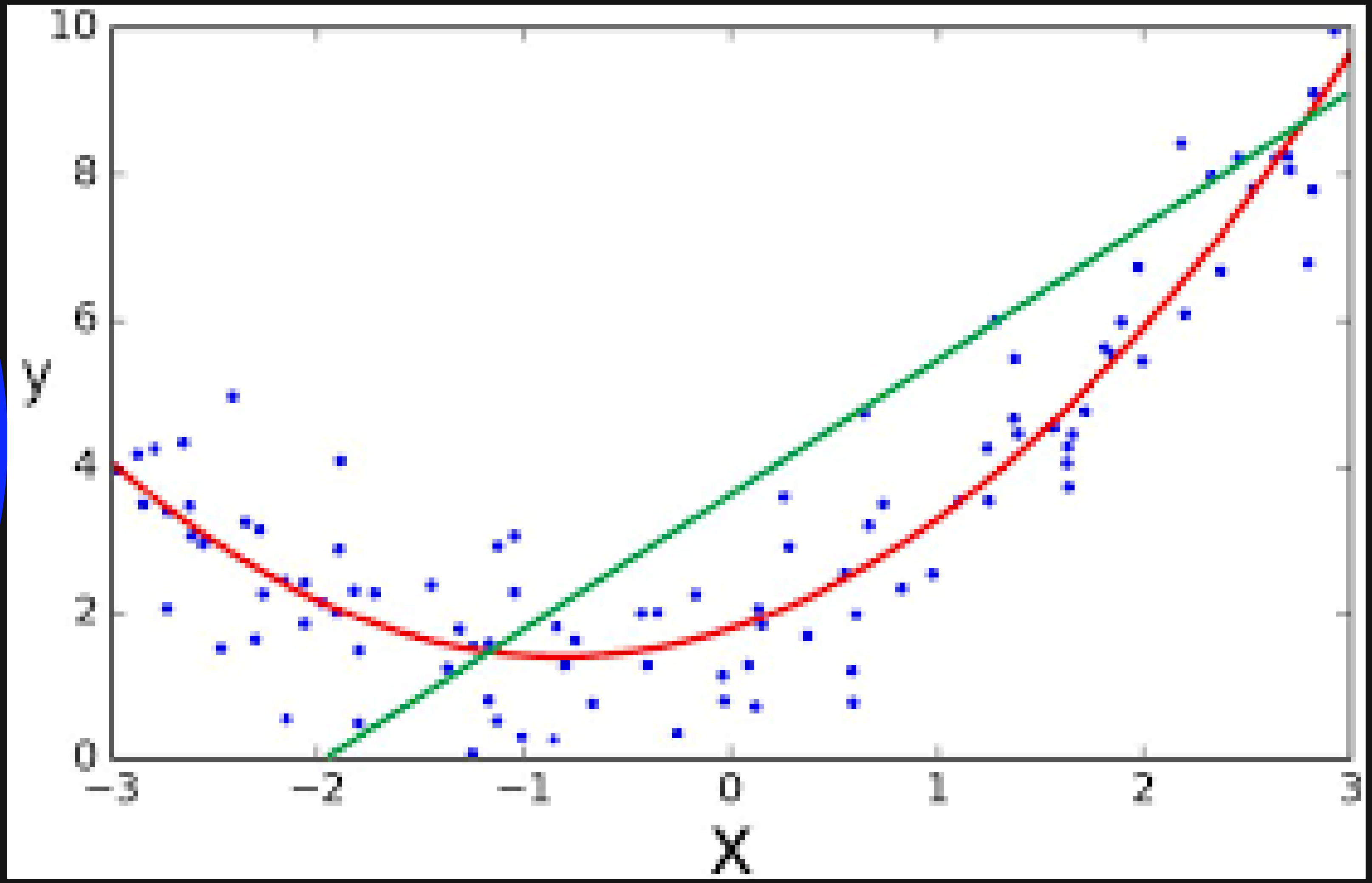
Join at

slido.com

#3801 219

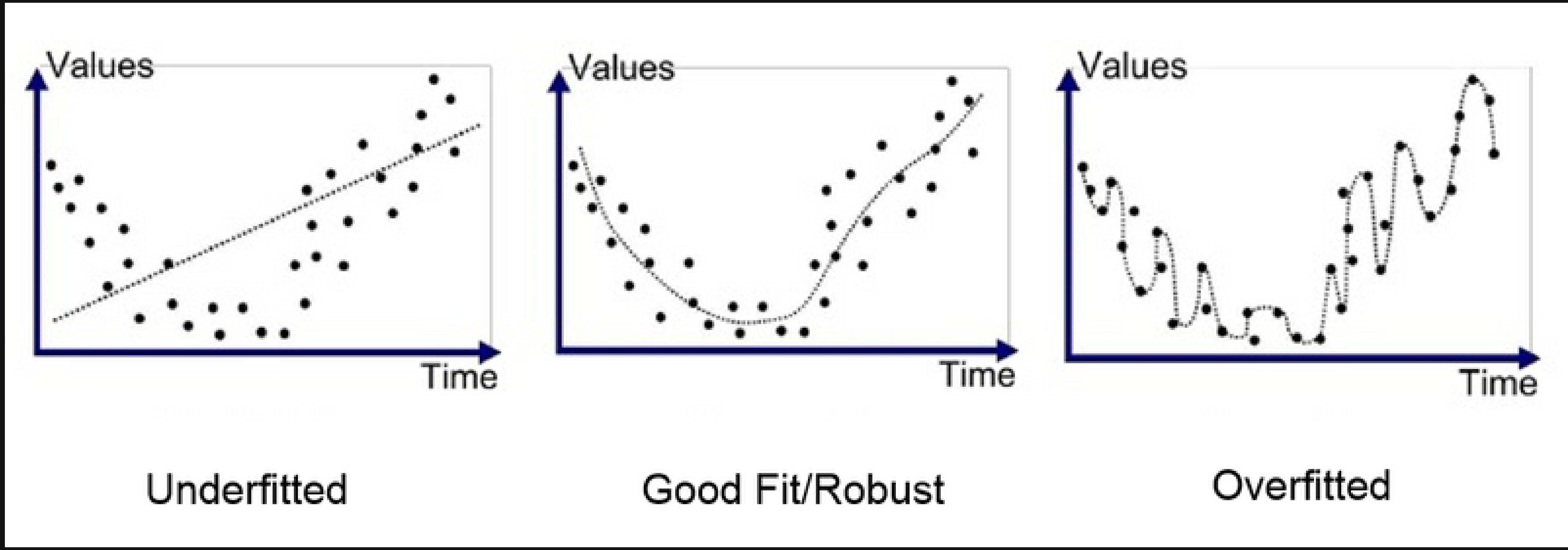
Polynomial Regression

In the case of polynomial regression, we can use any polynomial curve of a particular degree instead of restricting to a linear curve like we did in linear regression. We can actually use a linear model to fit nonlinear data. A simple way to do this is to add powers of each feature as new features, then train a linear model on this extended set of features. We use a generic polynomial expression for finding the best fit.



Adding features introduces a new problem called Overfitting and Underfitting

The Tussle between Bias and Variance

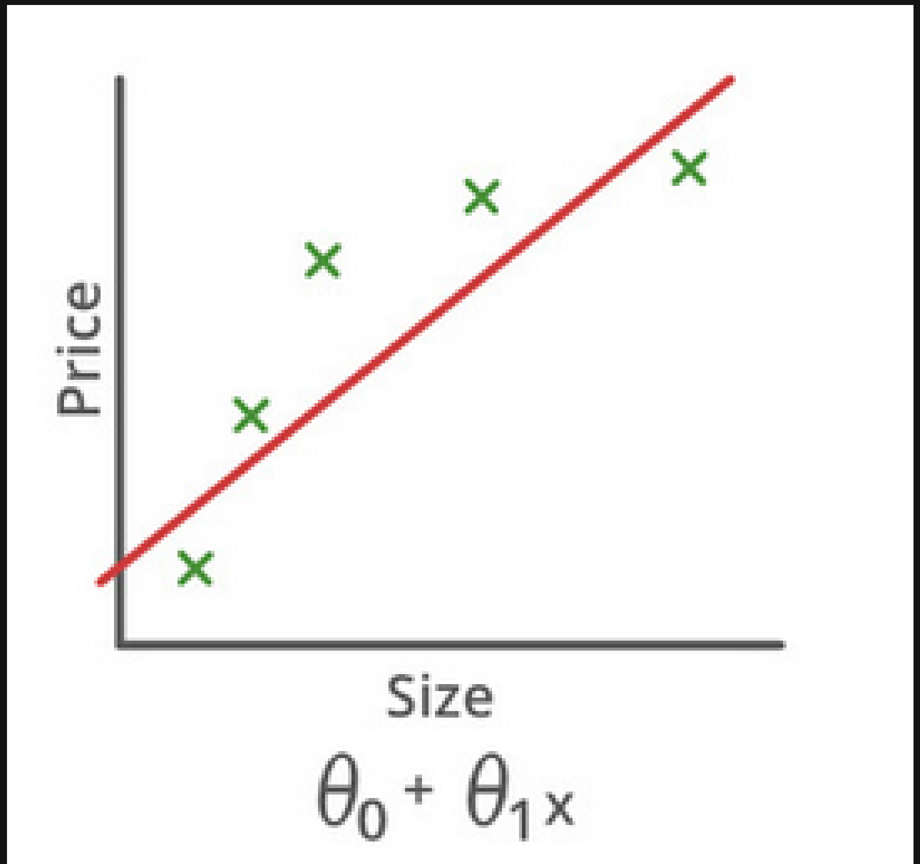


Consider a prediction model based on the dataset plotted in the above figures. The leftmost figure below shows the result of fitting a line to a dataset. We see that the data doesn’t really lie on straight line, so the fit doesn't really work well. Now, if we introduced an x^2 feature, then we obtain a slightly better fit to the data. The more number of features we add, it looks like the better a fit we get. But, adding too many features can be dangerous too since it would seem like it perfectly fits the given data but does not generalize well to predict new data.

Bias

Bias refers to the error in fitting the data due to a simple model where there is a dearth of features. The inability of a machine learning model to capture the true relationship in a given dataset is basically what bias is.

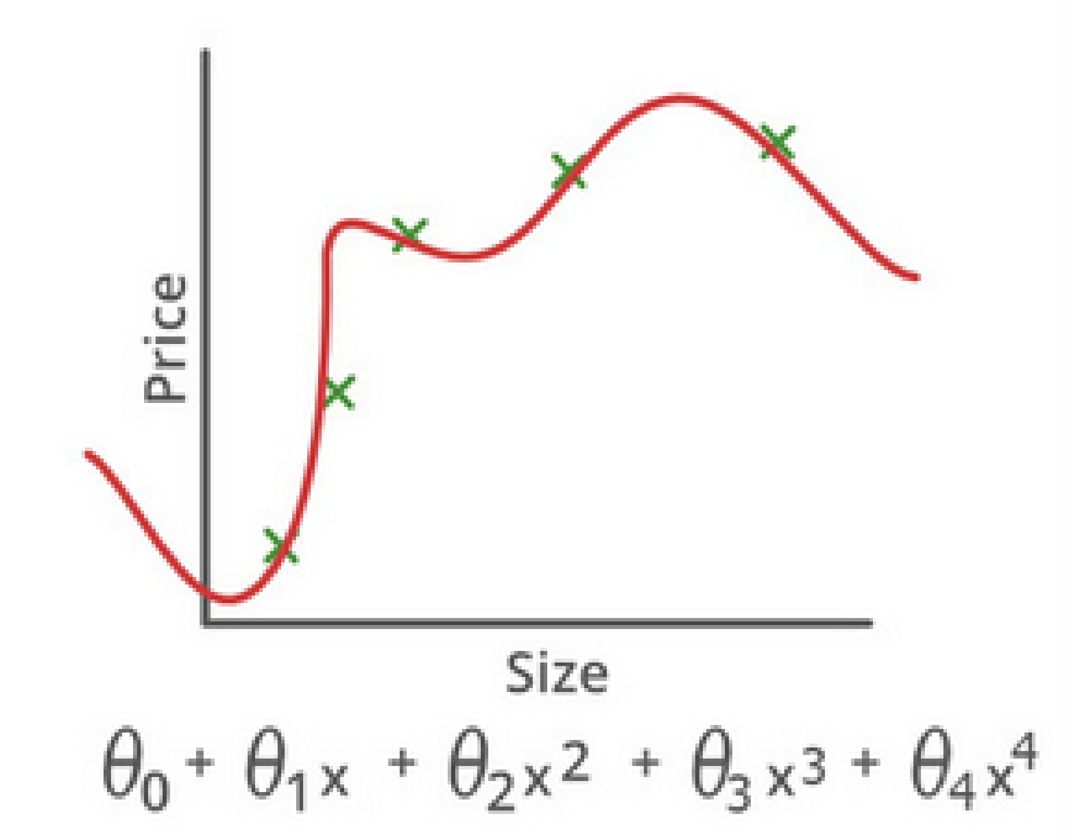
A high bias means that the model can't detect the patterns in the data and is under-fitting. The model in this case can't learn the present data properly. It's also called a case of **Underfitting .**



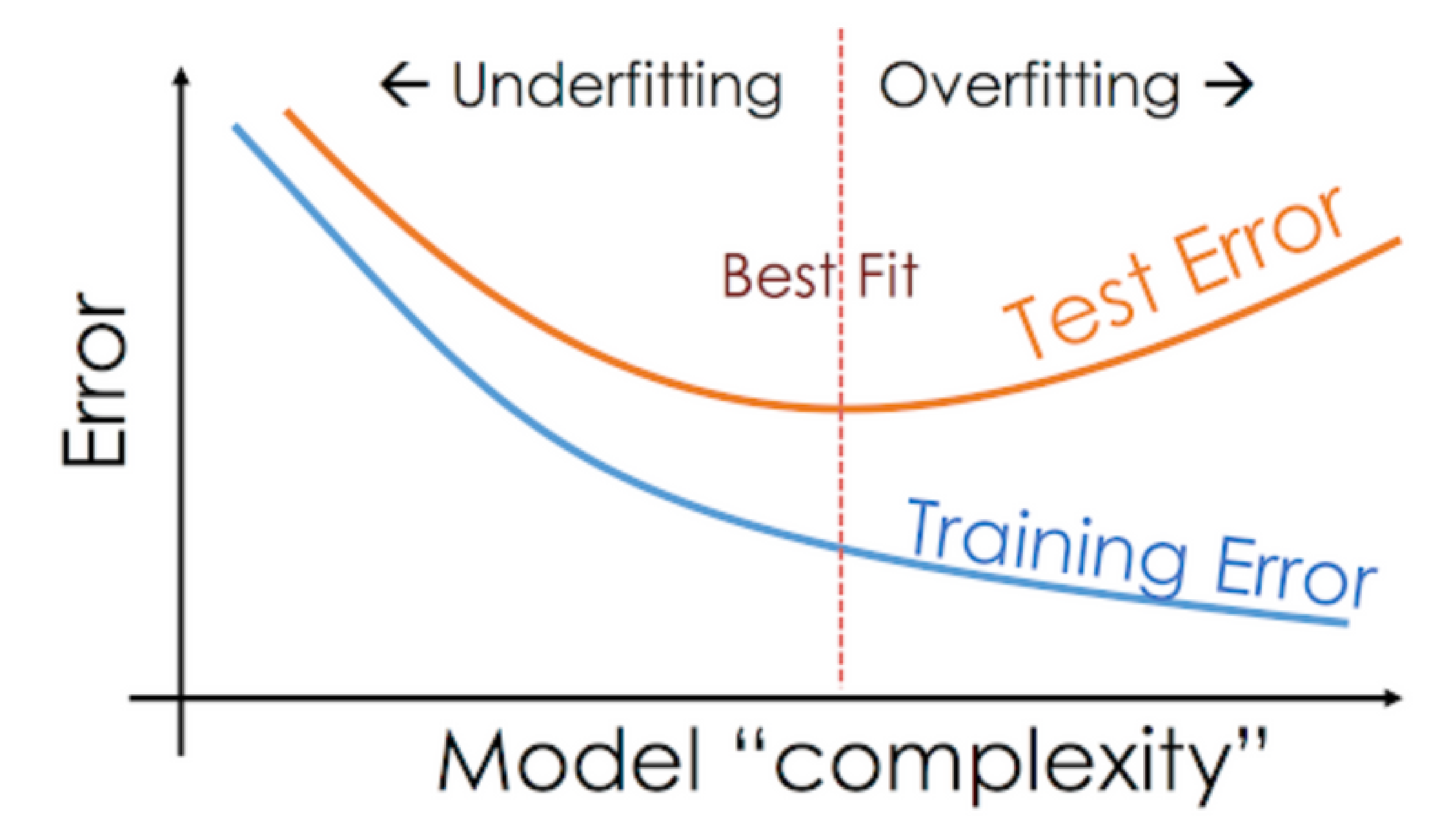
Variance

Variance refers to the error in fitting the data due to a complex model where there are too many features. Such a model does a great job in fitting the training data but it might fail to work the same way for testing data.

High variance means the model passes through most of the data points and it results in over-fitting (fitting the training set well but not the testing set) the data. The model in this case learns the present data too well and it's predictive power is poor.It's also called a case of **Overfitting**



Optimizing Bias and Variance



A high bias algorithm will have high Cost with both the training data and testing data . As we increase complexity , the algorithm will start accurately predicting training data and testing data.

Further increasing the complexity might cause an overfit or high variance , causing the algorithm to learn the training data a bit too well and perform poorly on testing data .

To prevent this , cost for every degree of x can be plotted for test data and the minima found can be the optimum power for the algorithm to balance bias and variance

We'll move to
Code Implementation.

slido

Join at

slido.com

#3206 471