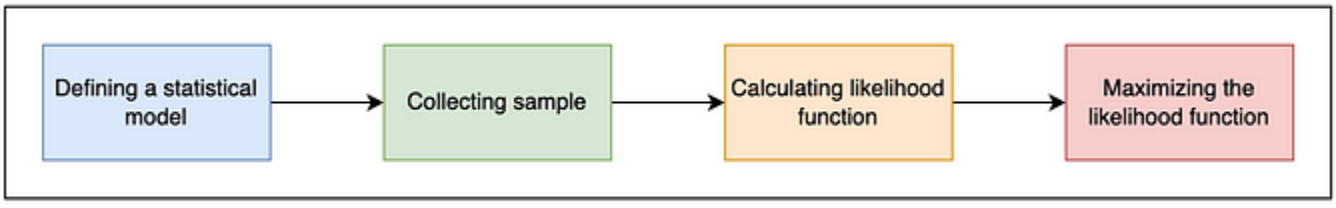
1.Maximum likelihood estimation (MLE)

Maximum likelihood estimation (MLE) is a statistical approach that determines the models' parameters in machine learning. The idea is to find the values of the model parameters that maximize the likelihood of observed data such that the observed data is most probable.

Let's look at an example to understand MLE better. Assume that we want to estimate the average height of a city's population. However, because of the sheer size of the population, we cannot calculate the true average height of the population. So, we estimate the average height as follows:

- **Defining a statistical model:** We start by assuming that the height of the population follows a normal distribution. This implies that few people have a shorter or taller height than average.
- **Collecting the sample:** We then collect a sample of heights from the population and find the average height based on that sample.
- **Calculating the likelihood function:** Given the population's average height, we look at the likelihood of observing heights. The likelihood function represents the probability of observing the provided data given the parameters in our model. In our case, the model's parameters are the normal distribution's mean and standard deviation. Due to computational reasons, the log-likelihood function is often used instead of the likelihood function.
- **Maximizing the likelihood function:** MLE aims to find the average height that maximizes the log-likelihood function of obtaining the observed sample and makes the observed heights most probable.



Estimation process

We can now model the average height with a normal distribution whose parameters are selected by maximizing the likelihood function.

Importance of MLE in machine learning

In supervised machine learning, we use labeled data that trains the model's parameters. The training data consists of input features and the corresponding output labels. During the training phase, we aim to find the model parameters that best capture the patterns in the labeled data.

MLE helps fine-tune the machine learning models. In the training phase, we adjust the model's parameters to maximize the likelihood of the labeled data. Alternatively, we can use a negative log-likelihood that represents the loss function. A **loss function** quantifies the difference between predicted and actual values and is defined as follows:

L = |y\{y}^|

Here, y represents the actual output and y^ represents the estimated value. We aim to minimize this loss function L during training to reach an accurate and effective model. Note that minimizing the negative log-likelihood is equivalent to maximizing the likelihood, and this is a common objective in the training of probabilistic models.

Reference-https://learningdaily.dev/understanding-maximum-likelihood-estimation-in-machine-learning-22b915c3e05a

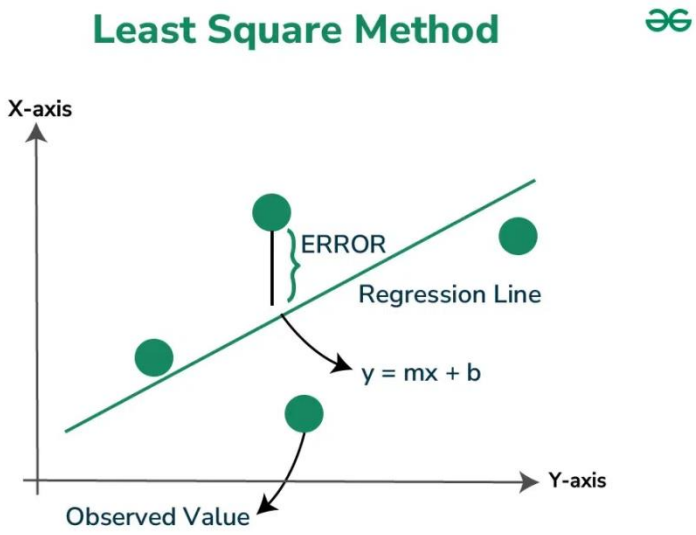
2.Least Square method

Least Square method is a fundamental mathematical technique widely used in **data analysis, statistics, and regression modeling** to identify the **best-fitting curve or line** for a given set of data points. This method ensures that the overall error is reduced, providing a highly accurate model for predicting future data trends.

In statistics, when the data can be represented on a cartesian plane by using the independent and dependent variable as the x and y coordinates, it is called **scatter data.** This data might not be useful in making interpretations or predicting the values of the dependent variable for the independent variable. So, we try to get an **equation of a line that fits best to the given data points** with the help of the **Least Square Method**.

Least Square Method is used to derive a generalized linear equation between two variables. when the value of the dependent and independent variable is represented as the x and y coordinates in a 2D cartesian coordinate system. Initially, known values are marked on a plot. The plot obtained at this point is called a scatter plot.
Then, we try to represent all the marked points as a straight line or a **linear equation**. The equation of such a line is obtained with the help of the Least Square method. This is done to get the value of the dependent variable for an independent variable for which the value was initially unknown. This helps us to make predictions for the value of dependent variable.

Least Square Method Definition
Least Squares method is a statistical technique used to find the equation of best-fitting curve or line to a set of data points by minimizing the sum of the squared differences between the observed values and the values predicted by the model.



This method aims at minimizing the sum of squares of deviations as much as possible. The line obtained from such a method is called a **regression line** **or** **line of best fit.**

Formula for Least Square Method

Least Square Method formula is used to find the best-fitting line through a set of data points. For a simple linear regression, which is a line of the form $y=mx+c$, where y is the dependent variable, x is the independent variable, a is the slope of the line, and b is the y-intercept, the formulas to calculate the slope (m) and intercept (c) of the line are derived from the following equations:

1. **Slope (m) Formula:** $m = n(\sum xy) - (\sum x)(\sum y) \,/\, n(\sum x2) - (\sum x)2$
2. **Intercept (c) Formula:** $c = (\sum y) - a(\sum x) \,/\, n$

Where:

- n is the number of data points,
- $\sum xy$ is the sum of the product of each pair of x and y values,
- $\sum x$ is the sum of all x values,
- $\sum y$ is the sum of all y values,
- $\sum x2$ is the sum of the squares of x values.

The steps to find the line of best fit by using the least square method is discussed below:

- **Step 1:** Denote the independent variable values as xi and the dependent ones as yi.
- **Step 2:** Calculate the average values of xi and yi as X and Y.
- **Step 3:** Presume the equation of the line of best fit as y = mx + c, where m is the slope of the line and c represents the intercept of the line on the Y-axis.
- **Step 4:** The slope m can be calculated from the following formula:

$$m = [\Sigma\,(X - xi) \times (Y - yi)] \,/\, \Sigma(X - xi)2$$

- **Step 5:** The intercept c is calculated from the following formula:

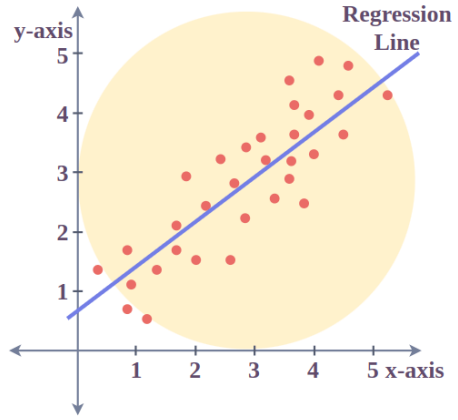
$$c = Y - mX$$

Thus, we obtain the line of best fit as y = mx + c, where values of m and c can be calculated from the formulae defined above.

These formulas are used to calculate the parameters of the line that best fits the data according to the criterion of the least squares, minimizing the sum of the squared differences between the observed values and the values predicted by the linear model.

Least Square Method Graph

Let us have a look at how the data points and the line of best fit obtained from the Least Square method look when plotted on a graph.



The red points in the above plot represent the data points for the sample data available. **Independent variables are plotted as x-coordinates and dependent ones are**

plotted as y-coordinates. The equation of the line of best fit obtained from the Least Square method is plotted as the red line in the graph.
We can conclude from the above graph that how the **Least Square method helps us to find a line that best fits the given data points** and hence can be used to make further predictions about the value of the dependent variable where it is not known initially.

Limitations of the Least Square Method

The Least Square method assumes that the data is evenly distributed and doesn't contain any outliers for deriving a line of best fit. But, this method doesn't provide accurate results for unevenly distributed data or for data containing outliers.

Check: Least Square Regression Line

Least Square Method Solved Examples

Problem 1: Find the line of best fit for the following data points using the Least Square method: (x,y) = (1,3), (2,4), (4,8), (6,10), (8,15).

Solution:

Here, we have x as the independent variable and y as the dependent variable. First, we calculate the means of x and y values denoted by X and Y respectively.

X = (1+2+4+6+8)/5 = 4.2

Y = (3+4+8+10+15)/5 = 8

Reference-https://www.geeksforgeeks.org/least-square-method/

3.Robust linear regression

Robust linear regression is a type of regression analysis designed to overcome the limitations of ordinary least squares (OLS) regression when the data contains outliers or is not normally distributed. Unlike OLS, which minimizes the sum of squared residuals, robust linear regression minimizes a different function of the residuals to reduce the influence of outliers on the regression model.

Why Use Robust Linear Regression?

1. **Outliers**: OLS regression is highly sensitive to outliers. A few extreme values can significantly affect the estimated coefficients.
2. **Non-normal errors**: If the residuals are not normally distributed, OLS estimations might be biased or inefficient.
3. **Data with heavy tails**: Data that do not follow a normal distribution (heavy-tailed distributions) can lead to unreliable OLS estimates.

Applications

- Robust regression is useful in real-world applications where data may not be perfectly clean, such as in economics, engineering, finance, and environmental science.
- It is particularly useful when you expect a high level of noise or outliers in the data that could distort the results from standard linear regression methods.

Summary

Robust linear regression methods are essential when dealing with real-world data that may contain outliers or is not normally distributed. By using robust methods like Huber Regression, RANSAC, or LAD, you can obtain more reliable estimates in the presence of outliers or non-normal errors.

4.RIDGE REGRESSION

Introduction to Ridge Regression

Ridge regression, also known as **Tikhonov regularization**, is a technique used to analyze multiple regression data that suffer from multicollinearity. Multicollinearity occurs when predictor variables are highly correlated, making it difficult to estimate the relationship between each predictor and the response variable accurately. Ridge regression addresses this problem by adding a penalty to the regression model to shrink the regression coefficients, which helps reduce model complexity and prevent overfitting.

Key Concepts of Ridge Regression

1. **Ordinary Least Squares (OLS) Regression Limitations**:
    - **Multicollinearity**: When two or more predictors in a model are correlated, it becomes difficult to determine their individual effect on the response variable. This leads to large variances in the estimated regression coefficients, making the model unstable.
    - **Overfitting**: In cases where the model has many predictors, OLS regression can overfit the data, capturing noise rather than the underlying trend.
2. **Ridge Regression Solution**:
    - **Regularization**: Ridge regression introduces a regularization parameter that adds a penalty for large coefficients in the regression model. This penalty helps to reduce the magnitude of the coefficients, leading to a simpler, more generalizable model.
    - **L2 Penalty**: Ridge regression uses an L2 penalty, which is the sum of the squares of the coefficients.

Ridge regression is one of the types of linear regression in which a small amount of bias is introduced so that we can get better long-term predictions.

Ridge regression is a regularization technique, which is used to reduce the complexity of the model. It is also called as **L2 regularization**.

In this technique, the cost function is altered by adding the penalty term to it. The amount of bias added to the model is called **Ridge Regression penalty**. We can calculate it by multiplying with the lambda to the squared weight of each individual feature.

The equation for the cost function in ridge regression will be:

$$\sum_{i=1}^{M}\left(y_i - y'_i\right)^2 = \sum_{i=1}^{M}\left(y_i - \sum_{j=0}^{n}\beta_j * x_{ij}\right)^2 + \lambda\sum_{j=0}^{n}{\beta_j}^2$$

- In the above equation, the penalty term regularizes the coefficients of the model, and hence ridge regression reduces the amplitudes of the coefficients that decreases the complexity of the model.
- As we can see from the above equation, if the values of **λ tend to zero, the equation becomes the cost function of the linear regression model.** Hence, for the minimum value of λ, the model will resemble the linear regression model.
- A general linear or polynomial regression will fail if there is high collinearity between the independent variables, so to solve such problems, Ridge regression can be used.
- It helps to solve the problems if we have more parameters than samples.

Advantages of Ridge Regression:

- **Reduced Variance**: By shrinking the coefficients, ridge regression reduces the model variance, making it less sensitive to small changes in the training data.
- **Improved Prediction**: Ridge regression often improves prediction accuracy, especially when dealing with multicollinearity or high-dimensional data.

Conclusion

Ridge regression is a powerful technique for handling multicollinearity and overfitting in regression models. By adding an L2 penalty to the regression, it reduces the magnitude of the coefficients, making the model more robust and generalizable. The choice of the regularization parameter λ\lambda λ is crucial and is typically selected using cross-validation to balance model complexity and prediction accuracy.

Reference-https://www.javatpoint.com/regularization-in-machine-learning

5. Bayesian Linear Regression

Linear regression is a popular regression approach in machine learning. Linear regression is based on the assumption that the underlying data is normally distributed and that all relevant predictor variables have a linear relationship with the outcome. But In the real world, this is not always possible, it will follows these assumptions, Bayesian regression could be the better choice.
Bayesian regression employs prior belief or knowledge about the data to "learn" more about it and create more accurate predictions. It also takes into account the data's uncertainty and leverages prior knowledge to provide more precise estimates of the data. As a result, it is an ideal choice when the data is complex or ambiguous.
Bayesian regression uses a Bayes algorithm to estimate the parameters of a linear regression model from data, including prior knowledge about the parameters. Because of its probabilistic character, it can produce more accurate estimates for regression parameters than ordinary least squares (OLS) linear regression, provide a measure of uncertainty in the estimation, and make stronger conclusions than OLS. Bayesian regression can also be utilized for related regression analysis tasks like model selection and outlier detection.

Bayesian Regression
Bayesian regression is a type of linear regression that uses Bayesian statistics to estimate the unknown parameters of a model. It uses Bayes' theorem to estimate the likelihood of a set of parameters given observed data. The goal of Bayesian regression is to find the best estimate of the parameters of a linear model that describes the relationship between the independent and the dependent variables.
The main difference between traditional linear regression and Bayesian regression is the underlying assumption regarding the data-generating process. Traditional linear regression assumes that data follows a Gaussian or normal distribution, while Bayesian regression has stronger assumptions about the nature of the data and puts a prior probability distribution on the parameters. Bayesian regression also enables more flexibility as it allows for additional parameters or prior distributions, and can be used to construct an arbitrarily complex model that explicitly expresses prior beliefs about the data. Additionally, Bayesian regression provides more accurate predictive measures from fewer data points and is able to construct estimates for uncertainty around the estimates. On the other hand, traditional linear regressions are easier to implement and generally faster with simpler models and can provide good results when the assumptions about the data are valid.
Bayesian Regression can be very useful when we have insufficient data in the dataset or the data is poorly distributed. The output of a Bayesian Regression model is obtained from a probability distribution, as compared to regular regression techniques where the output is just obtained from a single value of each attribute.

Some Dependent Concepts for Bayesian Regression
The important concepts in Bayesian Regression are as follows:

Bayes Theorem
Bayes Theorem gives the relationship between an event's prior probability and its posterior probability after evidence is taken into account. It states that the conditional probability of an event is equal to the probability of the event given certain conditions multiplied by the prior probability of the event, divided by the probability of the conditions.

i.e .

Where P(A|B) is the probability of event A occurring given that event B has already occurred, P(B|A) is the probability of event B occurring given that event A has already occurred, P(A) is the probability of event A occurring and P(B) is the probability of event B occurring.

Maximum Likelihood Estimation (MLE)

MLE is a method used to estimate the parameters of a statistical model by maximizing the likelihood function. it seeks to find the parameter values that make the observed data most probable under the assumed model. MLE does not incorporate any prior information or assumptions about the parameters, and it provides point estimates of the parameters

Maximum A Posteriori (MAP) Estimation

MAP estimation is a Bayesian approach that combines prior information with the likelihood function to estimate the parameters. It involves finding the parameter values that maximize the posterior distribution, which is obtained by applying Bayes' theorem. In MAP estimation, a prior distribution is specified for the parameters, representing prior beliefs or knowledge about their values. The likelihood function is then multiplied by the prior distribution to obtain the joint distribution, and the parameter values that maximize this joint distribution are selected as the MAP estimates. MAP estimation provides point estimates of the parameters, similar to MLE, but incorporates prior information.

Need for Bayesian Regression

There are several reasons why Bayesian regression is useful over other regression techniques. Some of them are as follows:

1. Bayesian regression also uses the prior belief about the parameters in the analysis. which makes it useful when there is limited data available and the prior knowledge are relevant. By combining prior knowledge with the observed data, Bayesian regression provides more informed and potentially more accurate estimates of the regression parameters.
2. Bayesian regression provides a natural way to measure the uncertainty in the estimation of regression parameters by generating the posterior distribution, which captures the uncertainty in the parameter values, as opposed to the single point estimate that is produced by standard regression techniques. This distribution offers a range of acceptable values for the parameters and can be used to compute trustworthy intervals or Bayesian confidence intervals.
3. In order to incorporate complicated correlations and non-linearities, Bayesian regression provides flexibility by offering a framework for integrating various prior distributions, which makes it capable to handle situations where the basic assumptions of standard regression techniques, like linearity or homoscedasticity, may not be true. It enables the modeling of more realistic and nuanced relationships between the predictors and the response variable.
4. Bayesian regression facilitates model selection and comparison by calculating the posterior probabilities of different models.
5. Bayesian regression can handle outliers and influential observations more effectively compared to classical regression methods. It provides a more robust approach to regression analysis, as extreme or influential observations have a lesser impact on the estimation.

Reference-https://www.geeksforgeeks.org/implementation-of-bayesian-regression/

6.Linear Models for Classification – Discriminant Functions

Introduction to Linear Models for Classification

Linear models for classification are used to separate data points of different classes using a linear decision boundary. These models are particularly useful when the relationship between the input features and the class labels is approximately linear. One of the fundamental concepts in linear classification is the **discriminant function**, which helps decide the class to which a data point belongs.

Discriminant Functions

A **discriminant function** is a function that assigns a score to each data point, allowing the classification model to decide which class the data point belongs to based on these scores. The basic idea is to create a linear function that outputs a different score for each class, and then assign the class with the highest score to the data point.

Types of Discriminant Functions

1. **Linear Discriminant Function**:

   The simplest form of a discriminant function is linear. It assumes that the decision boundary between classes is a linear function of the input features.

2. **Fisher's Linear Discriminant**:

   Fisher's Linear Discriminant is a specific type of linear discriminant function that is particularly useful for binary classification problems.

   It aims to find a projection (a linear combination of features) that maximizes the separation between two classes by maximizing the ratio of between-class variance to within-class variance.

3. **Quadratic Discriminant Analysis (QDA)**:

   QDA extends the concept of a linear discriminant function by allowing for a **quadratic decision boundary**.

   Unlike linear discriminant functions that assume a linear boundary, QDA allows for more complex boundaries, which can provide better classification accuracy for certain datasets.

4. **Logistic Regression**:

   Although typically associated with regression, logistic regression is a linear classifier that uses a logistic function to model the probability that a data point belongs to a particular class.The decision boundary is linear, and logistic regression is commonly used for binary classification problems.

7& 8.Probabilistic Generative and Discriminative Models

Machine learning algorithms today rely heavily on probabilistic models, which take into consideration the uncertainty inherent in real-world data. These models make predictions based on probability distributions, rather than absolute values, allowing for a more nuanced and accurate understanding of complex systems. One common approach is Bayesian inference, where prior knowledge is combined with observed data to make predictions. Another approach is maximum likelihood estimation, which seeks to find the model that best fits observational data.

What are Probabilistic Models?

Probabilistic models are an essential component of machine learning, which aims to learn patterns from data and make predictions on new, unseen data. They are statistical models that capture the inherent uncertainty in data and incorporate it into their predictions. Probabilistic models are used in various applications such as image and speech recognition, natural language processing, and recommendation systems. In recent years, significant progress has been made in developing probabilistic models that can handle large datasets efficiently.

Categories Of Probabilistic Models

These models can be classified into the following categories:

- Generative models
- Discriminative models.
- Graphical models

Generative models:

Generative models aim to model the joint distribution of the input and output variables. These models generate new data based on the probability distribution of the original dataset. Generative models are powerful because they can generate new data that resembles the training data. They can be used for tasks such as image and speech synthesis, language translation, and text generation.

Discriminative models

The discriminative model aims to model the conditional distribution of the output variable given the input variable. They learn a decision boundary that separates the different classes of the output variable. Discriminative models are useful when the focus is on making accurate predictions rather than generating new data. They can be used for tasks such as image recognition, speech recognition, and sentiment analysis.

Graphical models

These models use graphical representations to show the conditional dependence between variables. They are commonly used for tasks such as image recognition, natural language processing, and causal inference.

Naive Bayes Algorithm in Probabilistic Models

The Naive Bayes algorithm is a widely used approach in probabilistic models, demonstrating remarkable efficiency and effectiveness in solving classification problems. By leveraging the power of the Bayes theorem and making simplifying assumptions about feature independence, the algorithm calculates the probability of the target class given the feature set. This method has found diverse applications across various industries, ranging from spam filtering to medical diagnosis. Despite its simplicity, the Naive Bayes algorithm has proven to be highly robust, providing rapid results in a multitude of real-world problems.

Naive Bayes is a probabilistic algorithm that is used for classification problems. It is based on the Bayes theorem of probability and assumes that the features are conditionally independent of each other given the class. The Naive Bayes Algorithm is used to calculate the probability of a given sample belonging to a particular class. This is done by calculating the posterior probability of each class given the sample and then selecting the class with the highest posterior probability as the predicted class.
The algorithm works as follows:

1. Collect a labeled dataset of samples, where each sample has a set of features and a class label.
2. For each feature in the dataset, calculate the conditional probability of the feature given the class.
3. This is done by counting the number of times the feature occurs in samples of the class and dividing by the total number of samples in the class.
4. Calculate the prior probability of each class by counting the number of samples in each class and dividing by the total number of samples in the dataset.
5. Given a new sample with a set of features, calculate the posterior probability of each class using the Bayes theorem and the conditional probabilities and prior probabilities calculated in steps 2 and 3.
6. Select the class with the highest posterior probability as the predicted class for the new sample.

Importance of Probabilistic Models

- Probabilistic models play a crucial role in the field of machine learning, providing a framework for understanding the underlying patterns and complexities in massive datasets.
- Probabilistic models provide a natural way to reason about the likelihood of different outcomes and can help us understand the underlying structure of the data.
- Probabilistic models help enable researchers and practitioners to make informed decisions when faced with uncertainty.
- Probabilistic models allow us to perform Bayesian inference, which is a powerful method for updating our beliefs about a hypothesis based on new data. This can be particularly useful in situations where we need to make decisions under uncertainty.

Advantages Of Probabilistic Models

- Probabilistic models are an increasingly popular method in many fields, including artificial intelligence, finance, and healthcare.
- The main advantage of these models is their ability to take into account uncertainty and variability in data. This allows for more accurate predictions and decision-making, particularly in complex and unpredictable situations.
- Probabilistic models can also provide insights into how different factors influence outcomes and can help identify patterns and relationships within data.

Disadvantages Of Probabilistic Models

There are also some disadvantages to using probabilistic models.

- One of the disadvantages is the potential for overfitting, where the model is too specific to the training data and doesn't perform well on new data.
- Not all data fits well into a probabilistic framework, which can limit the usefulness of these models in certain applications.
- Another challenge is that probabilistic models can be computationally intensive and require significant resources to develop and implement.

Reference- https://www.geeksforgeeks.org/probabilistic-models-in-machine-learning/

9.Laplacian Approximation

Introduction to Laplacian Approximation

The **Laplacian approximation** is a technique used in statistics and machine learning to simplify complex probability distributions, especially in the context of Bayesian inference. Bayesian inference involves updating our beliefs about a parameter (like the average height of people in a city) based on observed data (like a sample of people's heights).

However, the exact calculations for updating these beliefs can be very complicated, particularly when dealing with large datasets or complex models. The Laplacian approximation provides a way to make these calculations easier by approximating the complex "posterior" distribution with a simpler one, usually a bell-shaped curve (Gaussian distribution).

Key Concepts of Laplacian Approximation

1. **Bayesian Inference**:
    - Bayesian inference is a method of statistical inference in which we update our beliefs or knowledge about a parameter based on new data.
    - We start with a prior belief about what the parameter might be (the **prior**), and after seeing the data, we update this belief to a new one (the **posterior**).
2. **The Challenge of Complex Distributions**:
    - In many real-world situations, the updated belief (posterior) after seeing the data can be very complex and not easy to calculate exactly.
    - This complexity can make it difficult to answer questions about the data, like predicting future events or understanding the distribution of the parameter.
3. **Simplifying the Problem**:
    - The Laplacian approximation simplifies this problem by assuming that, around the most likely value of the parameter (where the posterior peaks), the shape of the distribution is approximately like a bell curve.
    - Even if the true distribution is not exactly a bell curve, this assumption makes the problem much more manageable because bell curves (Gaussians) are mathematically simple and well-understood.
4. **How Laplacian Approximation Works**:
    - Imagine you have a complicated, lumpy landscape (which represents the complex posterior distribution). You want to find the highest point (the most likely value of the parameter) and understand the surrounding area.
    - The Laplacian approximation finds the highest point and then approximates the area around it as a smooth, rounded hill (a bell curve).
    - By doing this, it allows us to quickly estimate probabilities and make inferences without having to deal with all the lumps and bumps of the actual landscape.

Example of Laplacian Approximation

Scenario: Let's say you want to estimate the average weight of apples in an orchard. You start with a belief (prior) about the average weight based on previous knowledge. Then, you collect a sample of apples and weigh them (data). After weighing the apples, you want to update your belief about the average weight of all apples in the orchard (posterior).

- **Before Seeing the Data**: You might believe that the average weight of apples is around 150 grams, but you are not certain. This is your prior belief.
- **After Seeing the Data**: You weigh 20 apples, and the results suggest that the average weight might be closer to 160 grams. You now have a new belief that is more informed by the data. This is your posterior belief.

Now, suppose the exact shape of this updated belief is complex and difficult to describe. The Laplacian approximation helps by assuming that around the most likely average weight (160 grams in this case), the shape of the distribution is roughly bell-shaped. This makes further calculations, like predicting the weight of a new apple, much simpler.

By using the Laplacian approximation, we can quickly approximate probabilities and make decisions without getting bogged down in complex mathematics. It's like drawing a smooth curve that fits well around the most likely values and using that simple curve for analysis instead of the true, more complex curve.

10.Bayesian Logistic Regression

Introduction to Bayesian Logistic Regression

Bayesian logistic regression is a method that combines the principles of Bayesian inference with logistic regression, a popular technique used for binary classification tasks. In traditional logistic regression, we aim to find the parameters (weights) that best separate two classes (e.g., spam vs. not spam emails) by maximizing the likelihood of the observed data.

In Bayesian logistic regression, instead of finding a single best estimate for these parameters, we consider them as random variables with a probability distribution. This approach allows us to incorporate prior knowledge about the parameters and quantify uncertainty in our predictions, which can be very useful in situations where data is limited or noisy.

Key Concepts of Bayesian Logistic Regression

1. **Logistic Regression**:
    - Logistic regression is a type of regression analysis used when the dependent variable (output) is binary (0 or 1, true or false).
    - It models the probability that a given input belongs to a particular class using the logistic function, which outputs a value between 0 and 1.
2. **Bayesian Inference**:
    - Bayesian inference is a method of statistical inference that updates the probability for a hypothesis as more evidence or information becomes available.
    - In Bayesian logistic regression, we start with a prior distribution over the model parameters (what we believe about the parameters before seeing the data) and update this to a posterior distribution after observing the data.
3. **Prior Distribution**:
    - The **prior** represents our beliefs about the parameters before we see any data. For example, we might believe that all weights are centered around zero with some uncertainty.
    - Choosing the right prior can help guide the model, especially when data is scarce. Common choices for priors in logistic regression are Gaussian distributions (which express a belief that the parameters are likely around zero but could be anywhere).
4. **Posterior Distribution**:
    - After observing the data, we update our prior beliefs to form the **posterior distribution**, which combines our prior beliefs with the likelihood of the observed data under different parameter values.
    - The posterior distribution reflects both our initial beliefs and the new information from the data, allowing us to make more informed predictions that take into account uncertainty.
5. **Likelihood Function**:
    - The **likelihood** represents the probability of observing the data given a set of parameter values. In logistic regression, this is based on the logistic function.
    - The likelihood is used in combination with the prior to compute the posterior.
6. **Posterior Predictive Distribution**:
    - Instead of making a single prediction, Bayesian logistic regression provides a distribution over possible predictions, accounting for uncertainty in the parameter estimates.

- This distribution can be used to compute predictive probabilities, making the model's predictions more robust and reliable.

Benefits of Bayesian Logistic Regression

- **Incorporates Prior Knowledge**: Allows incorporating prior beliefs about the model parameters, which is helpful when data is limited or when we have domain knowledge.
- **Quantifies Uncertainty**: Provides a probabilistic framework that quantifies uncertainty in predictions, which is valuable for risk assessment and decision-making in critical applications.
- **Avoids Overfitting**: By regularizing parameter estimates through priors, Bayesian logistic regression can prevent overfitting, especially in high-dimensional datasets or when the number of features exceeds the number of observations.

Example of Bayesian Logistic Regression

Let's illustrate Bayesian logistic regression using a simple example of binary classification. We will use synthetic data to classify points into two categories.

Scenario: Suppose we have a dataset of students' study hours and their pass/fail status in an exam. can model the probability of passing the exam based on the number of study hours using Bayesian logistic regression.

Conclusion

Bayesian logistic regression provides a powerful framework for binary classification tasks by combining the strengths of logistic regression with the flexibility and uncertainty quantification of Bayesian inference. This approach is particularly valuable when dealing with small datasets, noisy data, or when prior knowledge about the parameters is available.

11.Kernel Functions, Using Kernels in Generalized Linear Models (GLMs), and Kernel Trick

Introduction to Kernel Functions

Kernel functions are mathematical functions used in machine learning, particularly in algorithms that rely on measuring the similarity between data points, such as support vector machines (SVMs) and certain types of regression models. The kernel function calculates the inner product of two data points in a higher-dimensional space without explicitly computing the coordinates in that space. This property, known as the **kernel trick**, enables complex, non-linear patterns to be learned from the data.

Key Concepts of Kernel Functions

1. **Kernel Function**:
    - A kernel function is a function that computes a dot product between two vectors in a (potentially very high-dimensional) feature space. It allows operations in this high-dimensional space without explicitly mapping the data to that space.
    - Common kernel functions include:
        - **Linear Kernel**: Computes the standard inner product between two vectors.
        - **Polynomial Kernel**: Computes a polynomial combination of the input data points.
        - **Gaussian (RBF) Kernel**: Computes similarity based on the distance between two points, where closer points have a higher similarity.
        - **Sigmoid Kernel**: Similar to neural network activation functions, often used in support vector machines.
2. **Kernel Trick**:
    - The kernel trick is a technique that allows linear classifiers to learn non-linear boundaries by implicitly mapping input features into high-dimensional feature spaces. This is achieved by replacing the inner product (dot product) in algorithms with a kernel function.
    - For example, instead of computing $\phi(x) \cdot \phi(y)$\phi(x) \cdot \phi(y)$\phi(x) \cdot \phi(y)$ (where ϕ\phi$ is a mapping function to a high-dimensional space), we compute $K(x,y)$K(x, y)$K(x,y)$, the kernel function, directly in the input space.
    - This avoids the computational cost of transforming data into high-dimensional space explicitly, making algorithms efficient even in very high-dimensional spaces.
3. **Mercer's Theorem**:
    - Mercer's theorem provides the theoretical foundation for kernel methods, stating that any continuous, symmetric, and positive semi-definite function can be used as a kernel function, effectively representing a dot product in some (potentially infinite-dimensional) feature space.

Using Kernels in Generalized Linear Models (GLMs)

Generalized Linear Models (GLMs) are a broad class of models that include linear regression, logistic regression, and Poisson regression. They can be extended to incorporate non-linear relationships using kernels, enabling them to model more complex patterns in data.

1. **GLMs and Feature Space**:
    - Traditional GLMs operate on the original feature space, applying a linear combination of the input features to predict the target variable.
    - By introducing kernel functions, GLMs can be transformed into models that operate in a high-dimensional feature space, allowing them to learn non-linear relationships.
2. **Kernelized GLMs**:
    - In a kernelized GLM, the input data is implicitly mapped to a high-dimensional space using a kernel function. The model then fits a linear combination of the mapped features.
    - For instance, in kernelized logistic regression, the decision boundary becomes non-linear in the original feature space but remains a linear function in the high-dimensional space induced by the kernel.
    - This technique allows GLMs to model more complex patterns without changing the underlying linear structure of the model in the feature space.
3. **Advantages of Using Kernels in GLMs**:
    - **Flexibility**: Enables the model to capture non-linear relationships.
    - **Computational Efficiency**: The kernel trick allows the use of high-dimensional feature spaces without explicitly computing transformations, reducing the computational burden.
    - **Application to Different Data Types**: Kernels can be designed for various types of data, including sequences, graphs, and text, extending the applicability of GLMs.

Example: Kernelized Logistic Regression

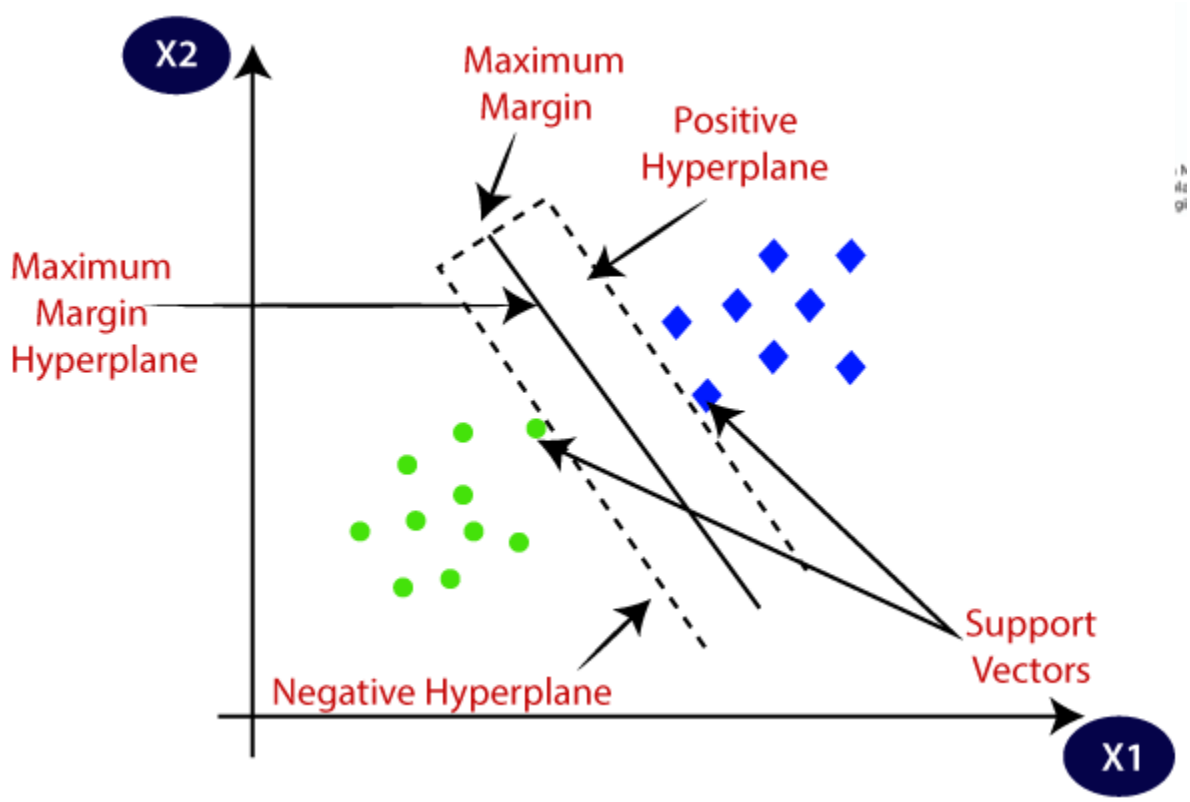
To illustrate the use of kernels in GLMs, let's consider a simple example of binary classification using kernelized logistic regression.

12.Support Vector Machine Algorithm

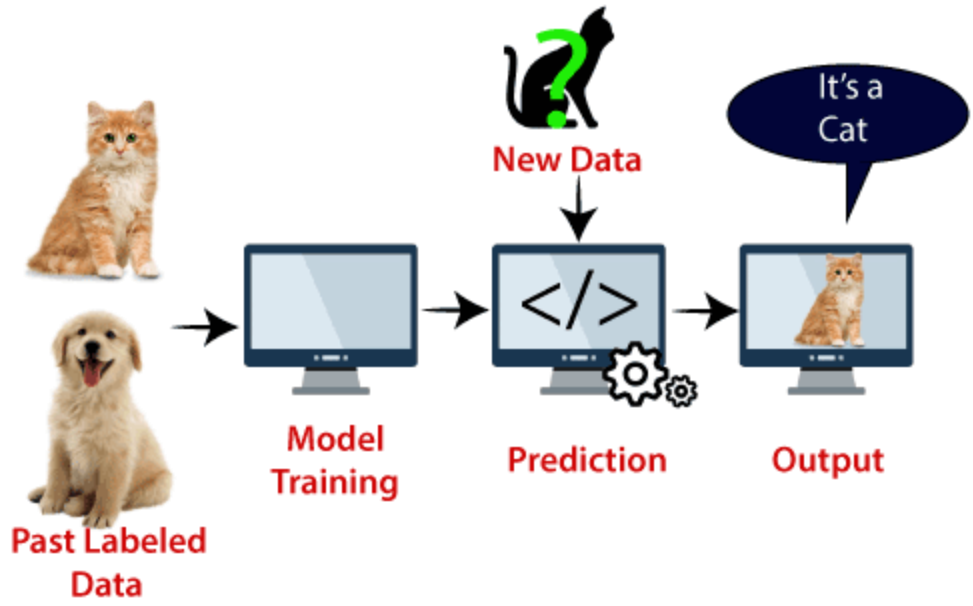
Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane:



Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat. Consider the below diagram:



SVM algorithm can be used for **Face detection, image classification, text categorization,** etc.

Types of SVM

SVM can be of two types:

- **Linear SVM:** Linear SVM is used for linearly separable data, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.
- **Non-linear SVM:** Non-Linear SVM is used for non-linearly separated data, which means if a dataset cannot be classified by using a straight line, then such data is termed as non-linear data and classifier used is called as Non-linear SVM classifier.

Hyperplane and Support Vectors in the SVM algorithm:

Hyperplane: There can be multiple lines/decision boundaries to segregate the classes in n-dimensional space, but we need to find out the best decision boundary that helps to classify the data points. This best boundary is known as the hyperplane of SVM.

The dimensions of the hyperplane depend on the features present in the dataset, which means if there are 2 features (as shown in image), then hyperplane will be a straight line. And if there are 3 features, then hyperplane will be a 2-dimension plane.

We always create a hyperplane that has a maximum margin, which means the maximum distance between the data points.

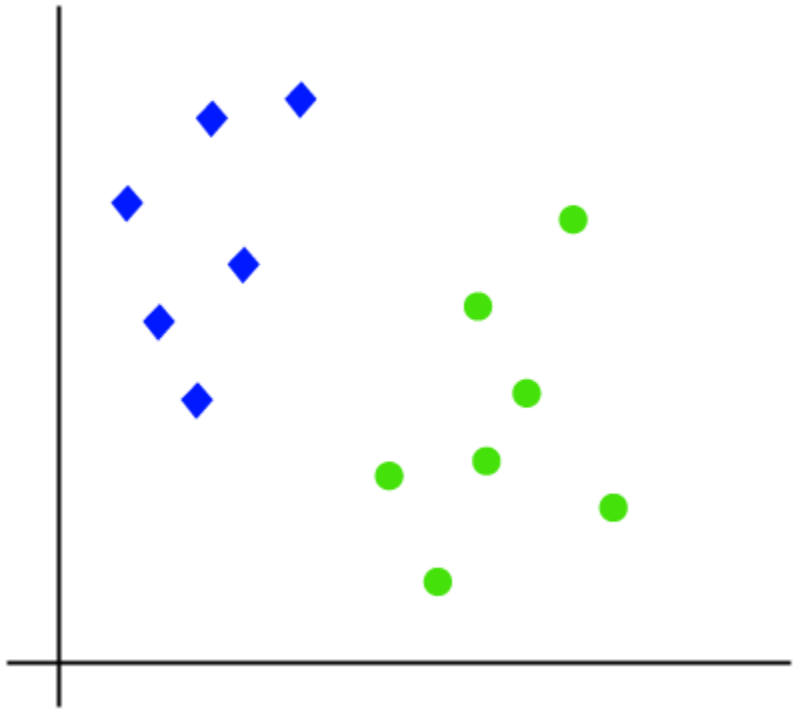
Support Vectors:

The data points or vectors that are the closest to the hyperplane and which affect the position of the hyperplane are termed as Support Vector. Since these vectors support the hyperplane, hence called a Support vector.

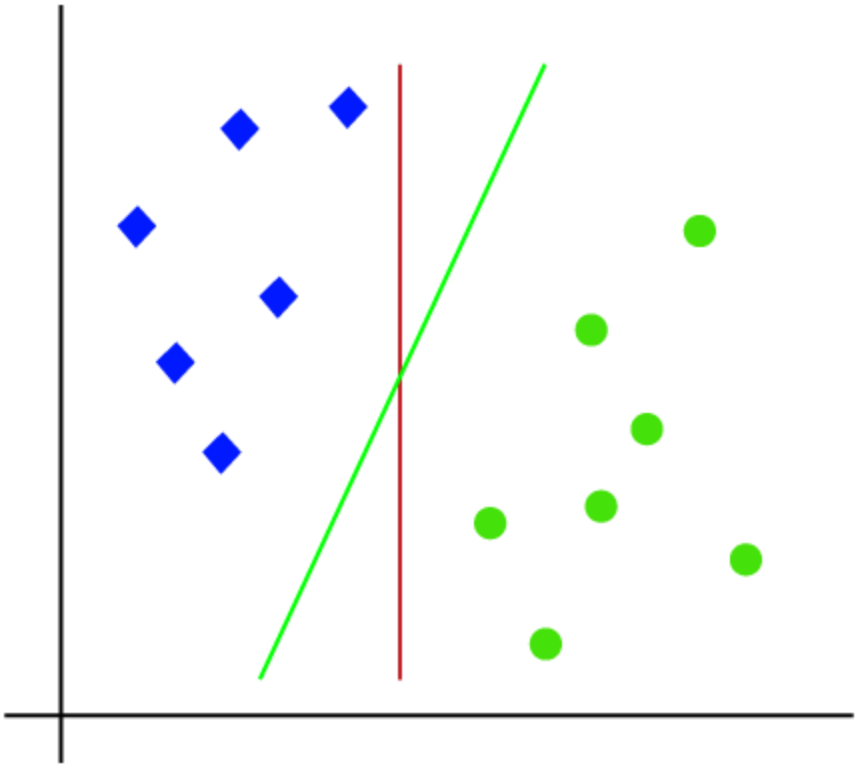
How does SVM works?

Linear SVM:

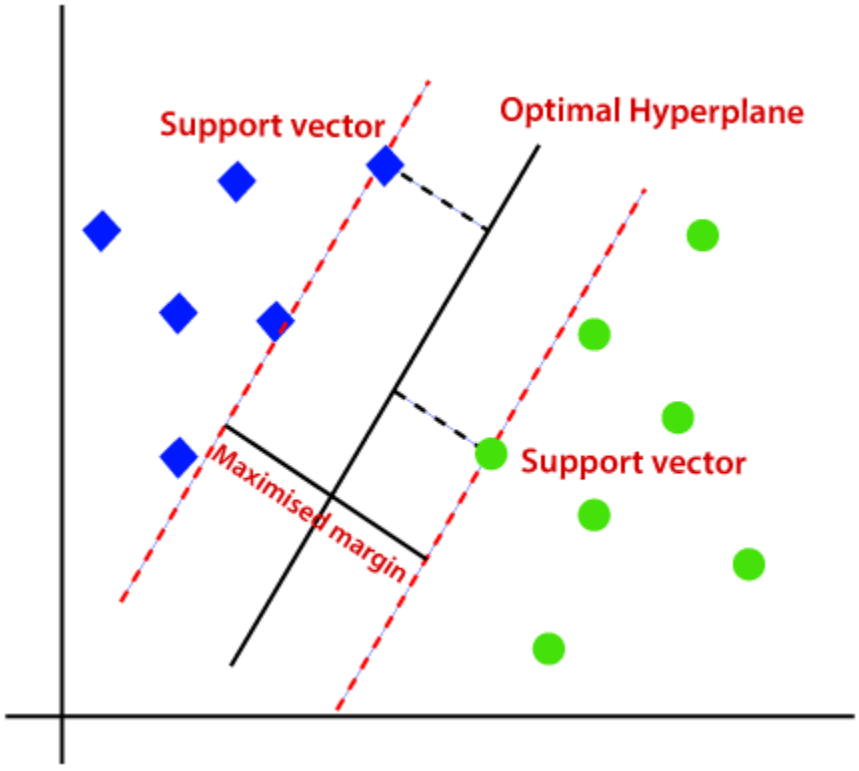
The working of the SVM algorithm can be understood by using an example. Suppose we have a dataset that has two tags (green and blue), and the dataset has two features x1 and x2. We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. Consider the below image:



So as it is 2-d space so by just using a straight line, we can easily separate these two classes. But there can be multiple lines that can separate these classes. Consider the below image:

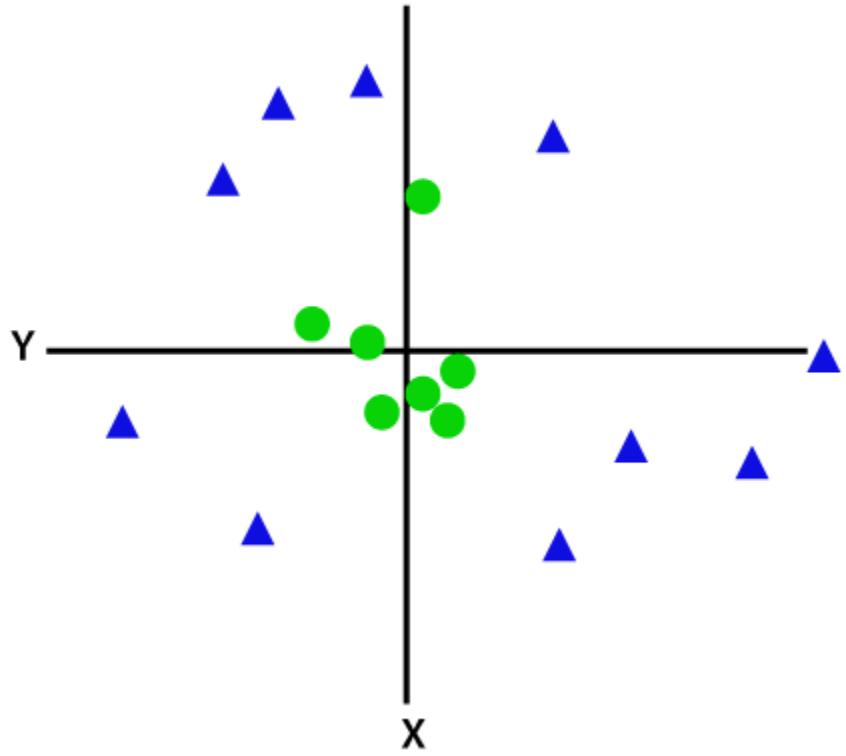


Hence, the SVM algorithm helps to find the best line or decision boundary; this best boundary or region is called as a **hyperplane**. SVM algorithm finds the closest point of the lines from both the classes. These points are called support vectors. The distance between the vectors and the hyperplane is called as **margin**. And the goal of SVM is to maximize this margin. The **hyperplane** with maximum margin is called the **optimal hyperplane**.



Non-Linear SVM:

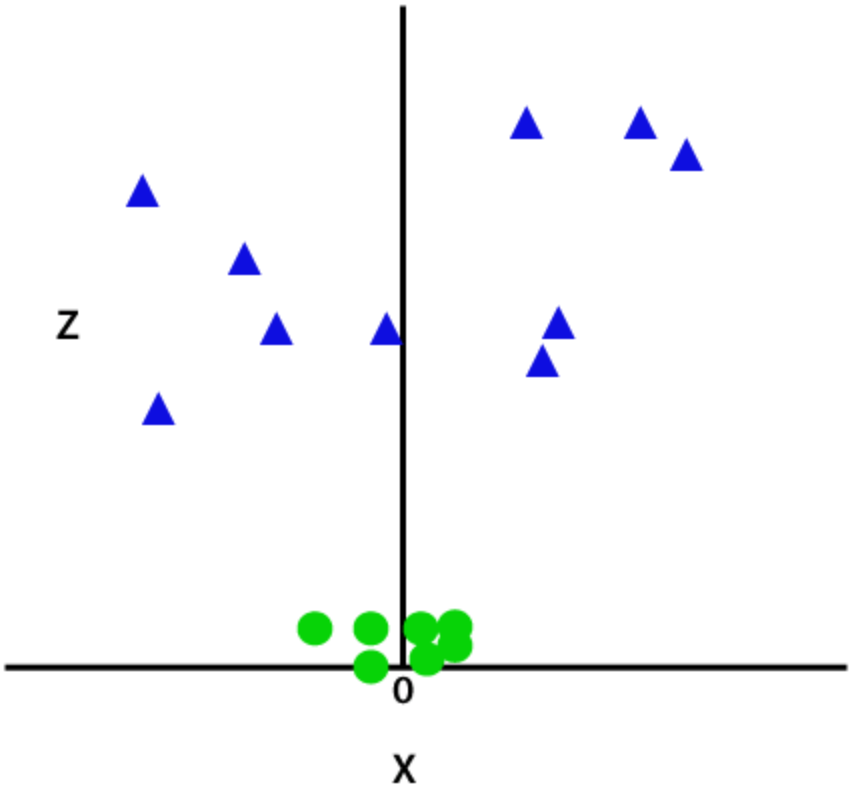
If data is linearly arranged, then we can separate it by using a straight line, but for non-linear data, we cannot draw a single straight line. Consider the below image:



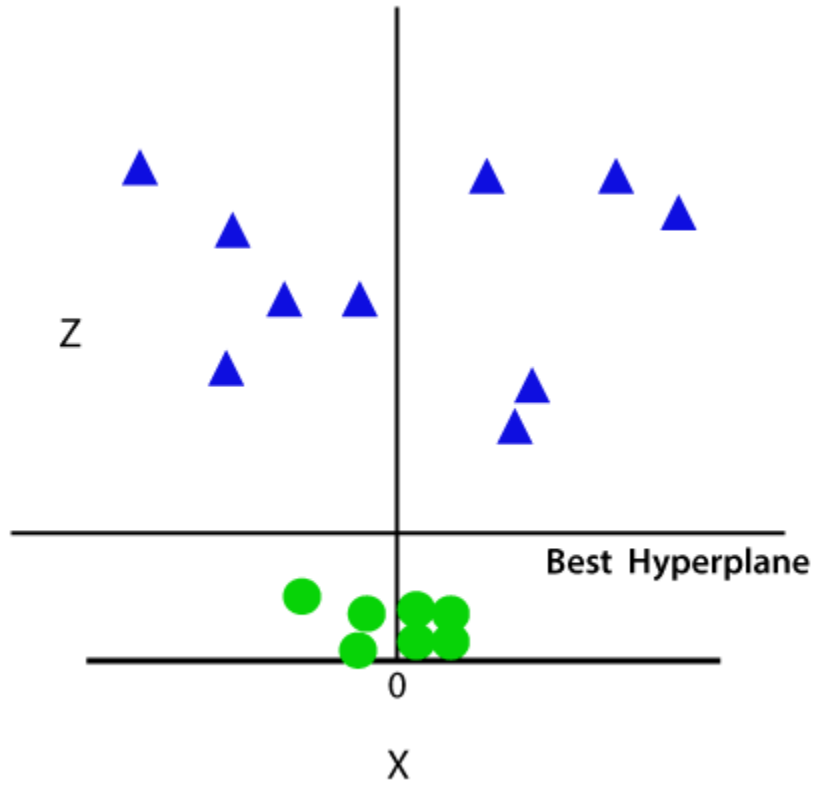
So to separate these data points, we need to add one more dimension. For linear data, we have used two dimensions x and y, so for non-linear data, we will add a third dimension z. It can be calculated as:

$z=x^2 +y^2$

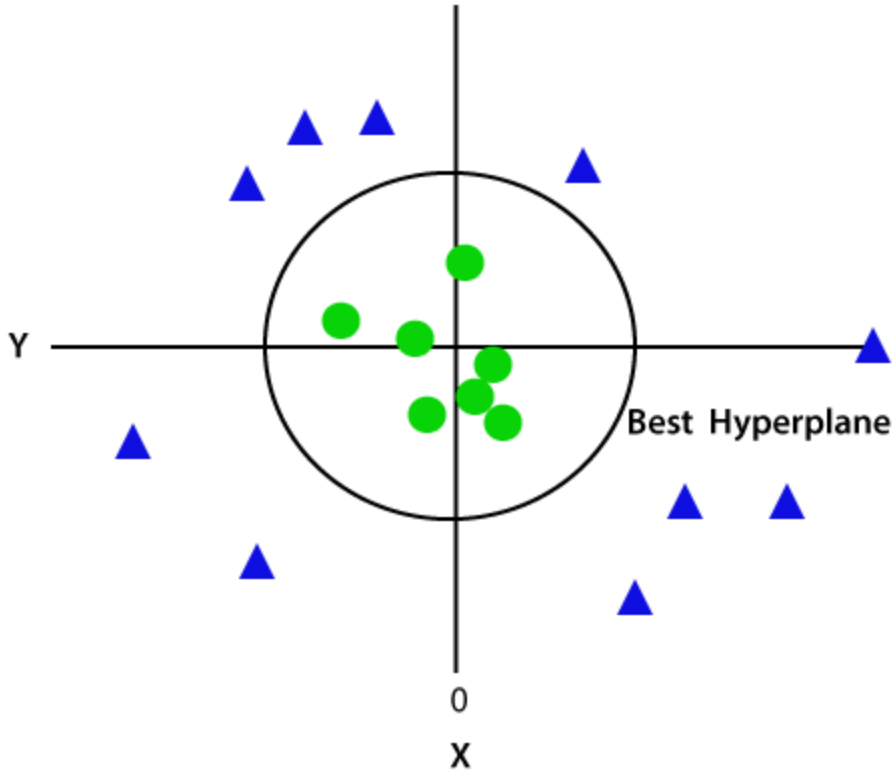
By adding the third dimension, the sample space will become as below image:



So now, SVM will divide the datasets into classes in the following way. Consider the below image:



Since we are in 3-d Space, hence it is looking like a plane parallel to the x-axis. If we convert it in 2d space with z=1, then it will become as:



Hence we get a circumference of radius 1 in case of non-linear data.

Reference- https://www.javatpoint.com/machine-learning-support-vector-machine-algorithm