MACHINE LEARNING TECHNIQUES

Overview of machine learning

- Machine learning is a subfield of artificial intelligence (AI) that focuses on developing algorithms
- and models that enable computers to learn from and make predictions or decisions based on data, without being explicitly programmed.

- The overarching goal of machine learning is to allow computers to improve their performance on a particular task over time, through experience and exposure to relevant data.
- Here's an overview of the key concepts and components within machine learning:

- **Data -** Data is the foundation of machine learning. It can come in various forms, such as structured data (tables), unstructured data (text, images, audio), or time series data.
- The quality and quantity of data play a crucial role in the success of machine learning algorithms.

- **Features**: Features are specific characteristics or attributes of the data that are used as input for machine learning algorithms.
- Feature engineering involves selecting, transforming, and preprocessing these attributes to provide meaningful information to the algorithms.

- **Labels**: In supervised learning, the training data includes both features and corresponding labels (also called targets or outputs).
- The algorithm learns to map features to labels by identifying patterns in the data.

- **Algorithms/Models**: Machine learning algorithms are mathematical and statistical techniques that learn patterns from data. There are various types of algorithms, such as:
-
- **Supervised Learning**: Algorithms learn from labeled data to make predictions or classifications.

- Common algorithms include linear regression, decision trees, support vector machines, and neural networks.

- **Unsupervised Learning**: Algorithms work with unlabeled data to find patterns or groupings.
- Clustering and dimensionality reduction are examples of unsupervised learning.

- **Semi-Supervised Learning:** This approach uses a mix of labeled and unlabeled data to improve model performance.

- **Reinforcement Learning:** Algorithms learn through trial and error interactions with an environment to maximize a reward signal. This is often used in scenarios like game playing and robotics.

- **Training**: During the training phase, the algorithm uses the labeled data to adjust its internal parameters and optimize its performance.
- This involves finding the best parameters that minimize the difference between predicted outputs and actual labels.

- **Validation and Testing:** After training, the model's performance is evaluated on separate data (validation and test sets) to ensure it can generalize well to new, unseen data.
- This step helps to detect overfitting (when the model memorizes the training data instead of learning patterns).

- **Metrics**: Metrics are used to quantify how well a model performs.
- In classification tasks, metrics like accuracy, precision, recall, and F1-score are commonly used.
- For regression tasks, mean squared error (MSE) and root mean squared error (RMSE) are common metrics.

- **Hyperparameters**: These are settings that are not learned during training but affect how the algorithm learns.
- Examples include learning rate, number of hidden layers in a neural network, and tree depth in decision trees.

- **Deployment**: Once a model has been trained and evaluated, it can be deployed for making predictions on new, real-world data.
- This can involve integrating the model into software applications, websites, or other systems.

- **Iterative Process**: Machine learning is often an iterative process involving continuous improvement.
- Models might be retrained with new data, hyperparameters might be adjusted, and feature engineering might evolve as more insights are gained.

- Machine learning has a wide range of applications, including image and speech recognition, natural language processing, recommendation systems, fraud detection, autonomous vehicles, healthcare diagnostics, and much more.
- It's a rapidly evolving field with ongoing research and advancements.

Perspectives and Issues

- Machine learning has brought about significant advancements and transformative changes in various industries
- but it also presents several perspectives and challenges that researchers, practitioners, and policymakers need to address.

- Here are some key perspectives and issues in machine learning:

  Perspectives:

- **Technological Advancements:** Machine learning has the potential to revolutionize industries by automating tasks,

making predictions, and improving decision-making. It enables the development of innovative products and services.

- **Data-Driven Insights:** Machine learning allows organizations to extract valuable insights from large and complex datasets, helping them understand customer behavior, market trends, and other patterns that were previously difficult to identify.

- **Personalization**: ML enables personalized experiences in various domains, such as recommendations on streaming platforms, personalized medicine, and targeted marketing.

- **Automation and Efficiency:** ML-driven automation can streamline processes, reduce human intervention, and improve

efficiency in tasks ranging from manufacturing to customer support.

- **Scientific Discovery:** Machine learning aids researchers in making new scientific discoveries, such as in genomics, particle physics, and material science, by analyzing complex data patterns.

Issues:

- **Bias and Fairness:** ML models can inherit biases present in the training data, leading to biased decisions or predictions.

- Ensuring fairness and mitigating bias is a critical challenge, particularly in sensitive domains like criminal justice and lending.

- **Interpretability**: Many complex ML models are difficult to interpret, which can be problematic in cases where the

reasoning behind a decision is important, such as in medical diagnoses or legal contexts.

- **Data Privacy:** Handling sensitive personal data raises concerns about privacy and security.
- Techniques like differential privacy aim to protect individuals' privacy while still enabling effective data analysis.

- Data Quality and Quantity: ML models require large and high-quality datasets for effective training.
- Obtaining such data can be challenging, and noisy or incomplete data can adversely affect model performance.

- **Lack of Generalization:** Some ML models perform well on training data but struggle to generalize to new, unseen data.
- This issue, known as overfitting, requires careful model selection and regularization.

- **Ethical Considerations:** Decisions made by ML models can have ethical implications.

- For example, autonomous vehicles must make decisions in life-threatening situations, posing ethical dilemmas.
- **Algorithmic Transparency:** Understanding how an ML model arrives at a decision is crucial for accountability and trust.

- Lack of transparency can lead to skepticism and reluctance to adopt ML solutions.
- **Resource Intensity:** Training complex ML models can require significant computational resources, raising concerns about energy consumption and environmental impact.

- **Lack of Domain Knowledge**: ML models might not possess domain-specific knowledge, which can lead to incorrect or nonsensical predictions if they encounter data outside their training distribution.

- **Job Displacement**: Automation driven by ML can lead to job displacement in certain sectors, raising questions about the

future of work and the need for upskilling and reskilling programs.

- **Regulation and Governance:** As ML becomes more integrated into society, the need for regulations to ensure safety, fairness, and accountability becomes crucial.

- Addressing these issues requires collaboration between researchers, industry, policymakers, and ethicists.
- The responsible development and deployment of machine learning technologies involve a holistic approach that balances innovation with ethical and societal considerations.
- **Concept Learning**

- Concept learning is a fundamental process in machine learning and cognitive science.
- It refers to the ability of a machine or a cognitive system to acquire, understand, and generalize concepts or categories from examples or experiences.

- The goal of concept learning is to develop models that can identify and classify new instances into predefined categories based on their underlying characteristics.

- Here's how concept learning typically works:

- **Data Collection**: The process begins with collecting a set of examples or instances that belong to different categories.
- These instances are often represented as feature vectors, where each feature represents a specific characteristic or attribute of the instance.

- **Feature Extraction:** Relevant features are extracted from the instances.
- Feature extraction involves selecting the most informative attributes that distinguish one category from another.

- **Training Phase**: During the training phase, the machine learning algorithm processes the labeled examples to learn the underlying patterns or rules that differentiate the different categories.
- This involves adjusting the parameters of the algorithm's model to fit the training data.

- **Generalization**: After learning from the training data, the algorithm aims to generalize the learned concepts to correctly classify new, unseen instances.
- Generalization involves making predictions for instances that were not part of the training set.

- **Testing and Evaluation**: The trained model is tested on a separate dataset that it has not seen before (the test set). The accuracy, precision, recall, F1-score, and other metrics are used to evaluate the model's performance on this dataset.

- **Concept Evolution:** As new instances are encountered, the concept might evolve over time to accommodate new variations or characteristics.
- Concept learning can be further categorized into various types based on the nature of the learning process:

- **Inductive Learning:** This involves inferring general rules or concepts from specific instances.
- It's the process of going from specific examples to general concepts.

- **Deductive Learning:** This is the reverse of inductive learning. It involves deriving specific instances from general rules or concepts.
- **Abductive Learning:** This is a form of reasoning where the learner generates hypotheses that best explain the observed data.

- It's often used in diagnostic systems where the learner infers the most likely cause of observed symptoms.
- **Instance-Based Learning:** Instead of learning explicit rules, this approach stores specific instances in memory.
- When classifying new instances, the system compares them to stored instances to make decisions.

- Concept learning is integral to many machine learning tasks, such as image recognition, natural language processing, medical diagnosis, and recommendation systems.
- It enables machines to recognize and understand patterns, leading to applications that can mimic human-like categorization and decision-making processes.

Related Areas Of Machine Learning

- Machine learning is a multidisciplinary field that intersects with several related areas.
- These related areas often contribute concepts, techniques, and insights that enhance the capabilities and applications of

machine learning. Here are some important related areas of machine learning:

- **Artificial Intelligence (AI):** AI is the broader field encompassing machine learning.

- It focuses on creating systems that can perform tasks that typically require human intelligence, including problem-solving, reasoning, planning, and perception.
- **Deep Learning:** A subset of machine learning, deep learning focuses on using neural networks with multiple layers (deep architectures) to model complex patterns. Deep learning has

achieved remarkable success in areas like image recognition, natural language processing, and reinforcement learning.

- **Neural Networks:** Neural networks are computational models inspired by the human brain's structure.

- They consist of interconnected nodes (neurons) organized in layers, and they are used for various tasks like pattern recognition and function approximation.
- **Data Science:** Data science encompasses the entire data analysis process, including data collection, cleaning,

exploration, visualization, statistical analysis, and machine learning.

- Machine learning techniques are often employed to build predictive models in data science projects.

- **Natural Language Processing (NLP):** NLP focuses on enabling computers to understand, interpret, and generate human language.
- It involves tasks like sentiment analysis, machine translation, text summarization, and chatbot development.

- **Computer Vision:** Computer vision deals with teaching computers to interpret and understand visual information from the world, such as images and videos.
- Applications include object detection, image segmentation, facial recognition, and more.

- **Reinforcement Learning:** Reinforcement learning is a branch of machine learning where agents learn to take actions in an environment to maximize cumulative rewards.
- It's used in scenarios where agents need to learn optimal strategies through trial and error.

- **Unsupervised Learning:** Unsupervised learning involves learning patterns from unlabeled data. Clustering and dimensionality reduction are common tasks in this area, aiming to uncover underlying structures in the data.

- **Semi-Supervised Learning:** This combines labeled and unlabeled data to train models, bridging the gap between supervised and unsupervised learning.
- **Transfer Learning:** Transfer learning involves transferring knowledge from one task or domain to another. Pretrained

models can be fine-tuned for specific tasks, saving time and resources.

- **Explainable AI (XAI):** XAI focuses on making machine learning models interpretable and explainable, enhancing transparency and building trust in AI systems.

- **Ethics in AI: T**his area addresses the ethical implications of AI and machine learning, including issues of bias, fairness, accountability, and transparency.
- **Bayesian Learning:** Bayesian methods involve probabilistic reasoning and statistical inference to make predictions. They

are useful for handling uncertainty and incorporating prior knowledge.

- **Causal Inference:** Causal inference aims to identify cause-and-effect relationships from observational data, a crucial aspect in making informed decisions based on data.

- **Optimization:** Optimization techniques are essential for training machine learning models, as they help find the best parameters to minimize or maximize a certain objective function.

- **Time Series Analysis:**This area focuses on analyzing data points that are collected over time, with applications in finance, economics, weather forecasting, and more.
- **Quantum Machine Learning:** An emerging field that explores how quantum computing can enhance machine learning algorithms, particularly for complex problems.

- These related areas collectively contribute to the growth and development of machine learning, enabling the field to tackle a wide range of challenges and applications across various domains.

Applications

- Machine learning has a wide range of applications across various industries and domains.

- Its ability to analyze data, identify patterns, and make predictions has led to transformative advancements in technology.
- Here are some prominent applications of machine learning:

Image and Video Analysis:

- **Image Recognition:** Identifying and classifying objects, people, or scenes within images.
- **Object Detection:** Locating and classifying multiple objects within an image or video.
- **Facial Recognition:** Identifying and verifying individuals based on facial features.

- **Gesture Recognition:** Understanding and interpreting human gestures for interaction.

Natural Language Processing (NLP):

- **-Sentiment Analysis:**Determining the sentiment (positive, negative, neutral) of text or speech.

- - **Machine Translation:**Translating text or speech from one language to another.
- - **Text Summarization:** Creating concise summaries of lengthy text documents.
- - **Named Entity Recognition:** Identifying entities like names, dates, locations, and organizations in text.

- **Healthcare and Medicine:**
- **- Medical Diagnostics:** Diagnosing diseases and conditions based on medical data and imaging.
- **- Drug Discovery:** Identifying potential new drug candidates using molecular data analysis.

- **- Personalized Medicine:** Tailoring medical treatments based on an individual's genetic and health data.
- **- Health Monitoring:** Analyzing health data from wearable devices for disease prevention and management.
- **Finance and Trading:**

- **Algorithmic Trading**: Using machine learning to make stock trading decisions based on historical data and market trends.
- - **Credit Scoring:** Assessing credit risk by analyzing financial and behavioral data.

- **- Fraud Detection:** Identifying fraudulent activities in financial transactions.
- **- Portfolio Management:** Optimizing investment portfolios using predictive analytics.
- **Retail and E-Commerce:**

- - **Recommendation Systems:** Suggesting products or services to users based on their preferences and behavior.
- - **Demand Forecasting:** Predicting product demand to optimize inventory and supply chain management.
- - **Price Optimization:** Adjusting product prices to maximize sales and revenue.

- **- Customer Segmentation:** Dividing customers into groups based on similar traits for targeted marketing.

- **Autonomous Vehicles:**
- **Self-Driving Cars:**Enabling vehicles to navigate and make decisions without human intervention.

- **-Advanced Driver Assistance Systems (ADAS):** Assisting drivers with features like lane departure warnings and adaptive cruise control.

- **Manufacturing and Industry:**

- **- Predictive Maintenance:** Anticipating equipment failures and performing maintenance before they occur.
- **- Quality Control**: Identifying defects in manufacturing processes using image analysis.
- **- Supply Chain Optimization:** Optimizing logistics and supply chain operations for efficiency.

- **Energy and Utilities:**
- **- Energy Consumption Prediction:** Forecasting energy consumption patterns to optimize usage.
- **- Smart Grid Management:** Managing and optimizing electricity distribution in real-time.

- **- Anomaly Detection:** Detecting abnormalities and faults in industrial equipment.

- 9. Agriculture:
- **- Crop Health Monitoring:**Analyzing satellite or drone imagery to assess crop health and yield predictions.

- - **Precision Agriculture:**Optimizing resource allocation (water, fertilizers) based on data analysis.
- **- Livestock Monitoring:** Tracking the health and behavior of livestock using sensor data.
- **Entertainment and Content Creation:**

- - **Content Recommendation:** Recommending movies, music, articles, and other content to users.
- - **Music and Art Generation:** Creating music, artwork, and other creative content using machine learning models.
- These are just a few examples of the many applications of machine learning.

- As the field continues to evolve, its impact on diverse industries
- and domains is likely to grow, leading to further innovations and improvements in various aspects of our lives.

Software Tools

- There are several popular software tools and libraries used in machine learning for building, training, and deploying models.
- These tools provide a wide range of functionalities, from data preprocessing and model development to visualization and deployment.
- Here are some of the most commonly used ones:

- **Python**:
- - **NumPy and pandas**: Libraries for numerical and data manipulation tasks.
- **scikit-learn**: A versatile library for various machine learning algorithms and tools.

- - **TensorFlow**: An open-source deep learning framework developed by Google.
- - **Keras**:An easy-to-use high-level neural networks API that runs on top of TensorFlow.
- - **PyTorch**: A deep learning framework with dynamic computation graphs, popular in research and prototyping.

- - **Matplotlib and Seaborn:** Libraries for data visualization.
- **R**:
- - **caret**: Provides a unified interface for a wide range of machine learning algorithms.
- - **xgboost**: An efficient and scalable gradient boosting library.

- - **randomForest**: Implements random forest algorithms for classification and regression.
- - **ggplot2**: A powerful package for creating visually appealing graphics.

- **Jupyter Notebooks:** An interactive environment for writing and running code, especially useful for data exploration and analysis.
- **Apache Spark MLlib:** A scalable machine learning library for distributed computing.

- **Microsoft Azure Machine Learning:** A cloud-based platform for building, deploying, and managing machine learning models.
- **Amazon SageMaker:** A fully managed service that enables the building, training, and deployment of machine learning models at scale.

Google Cloud AI Platform: A cloud-based platform for building, training, and deploying machine learning models using Google Cloud resources.

- **IBM Watson Studio:** A collaborative environment for data scientists and developers to work on machine learning projects.

- **RapidMiner**: A platform for data science teams to build, deploy, and manage machine learning models.
- **KNIME**:An open-source platform for data analytics, reporting, and integration.
- **WEKA**: A suite of machine learning algorithms for data mining tasks, developed in Java.

- **Orange**: An open-source data visualization and analysis tool with machine learning components.
- Databricks: A unified analytics platform that includes Apache Spark for big data processing and machine learning.
- **H2O.ai:** A platform for building and deploying machine learning models with a focus on automation and scalability.

- **Caffe**: A deep learning framework popular for image classification tasks.
- **MLflow**: An open-source platform for managing the end-to-end machine learning lifecycle.

- These tools provide a variety of options for developers, data scientists, and researchers to work with data and build machine learning models.
- The choice of tool often depends on the specific task, the programming language preferred, the level of expertise, and the platform or infrastructure available.

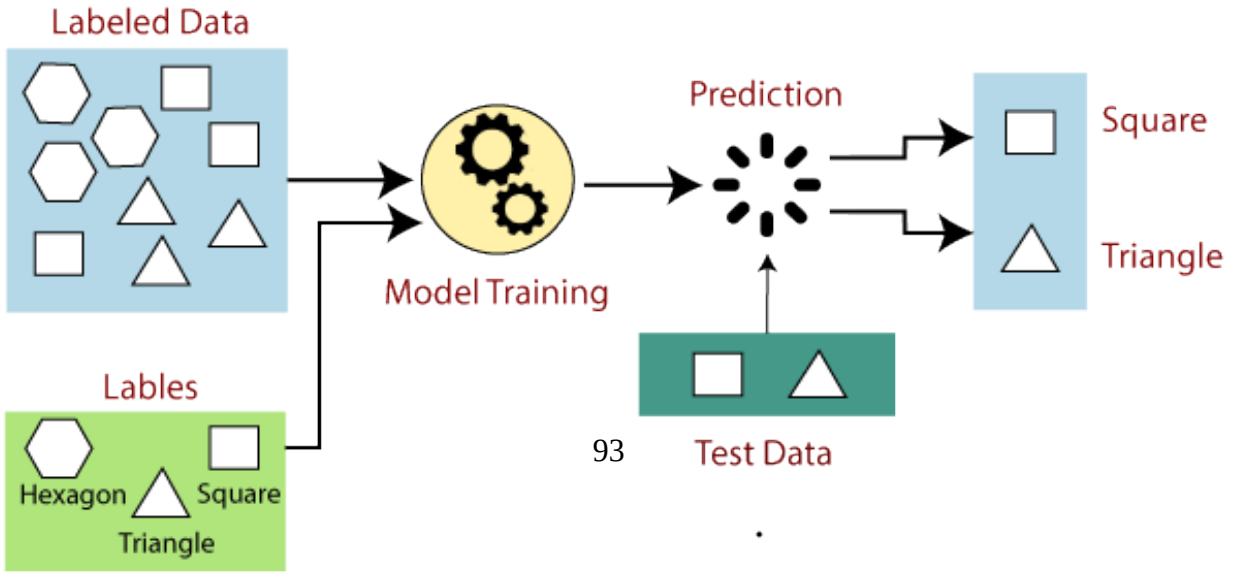
PART B Regression

Supervised Learning

- Supervised learning is the type of machine learning in which machines are trained using well "labelled" training data, and on basis of that data, machines predict the output.

- The labelled data means some input data is already tagged with the correct output.
- In supervised learning, models are trained using labelled dataset, where the model learns about each type of data.

- Once the training process is completed, the model is tested on the basis of test data (a subset of the training set), and then it predicts the output.



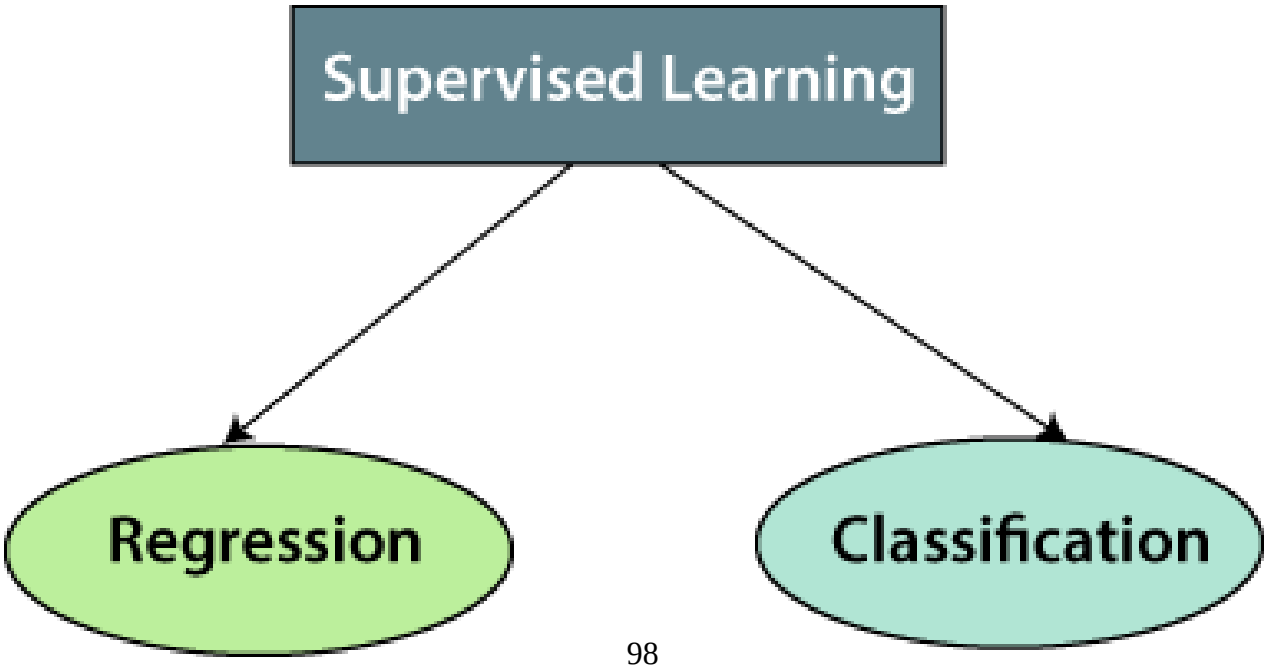
- If the given shape has four sides, and all the sides are equal, then it will be labelled as a Square.
- If the given shape has three sides, then it will be labelled as a triangle.
- If the given shape has six equal sides then it will be labelled as hexagon.

Steps Involved in Supervised Learning:

- First Determine the type of training dataset
- Collect/Gather the labelled training data.
- Split the training dataset into training dataset, test dataset, and validation dataset.

- Determine the input features of the training dataset, which should have enough knowledge so that the model can accurately predict the output.
- Determine the suitable algorithm for the model, such as support vector machine, decision tree, etc.
- Execute the algorithm on the training dataset.

- Sometimes we need validation sets as the control parameters, which are the subset of training datasets.
- Evaluate the accuracy of the model by providing the test set.
- If the model predicts the correct output, which means our model is accurate.


Supervised Learning
Regression
Classification

Regression

- Regression algorithms are used if there is a relationship between the input variable and the output variable.
- It is used for the prediction of continuous variables, such as Weather forecasting, Market Trends, etc.
- Below are some popular Regression algorithms which come under supervised learning:

- Linear Regression
- Regression Trees
- Non-Linear Regression
- Bayesian Linear Regression
- Polynomial Regression

Classification

- Classification algorithms are used when the output variable is categorical, which means there are two classes such as Yes-No, Male-Female, True-false, etc.

Techniques

- Random Forest
- Decision Trees

- Logistic Regression
- Support vector Machines

Advantages of Supervised learning:

- With the help of supervised learning, the model can predict the output on the basis of prior experiences.
- In supervised learning, we can have an exact idea about the classes of objects.

- Supervised learning model helps us to solve various real-world problems such as fraud detection, spam filtering, etc.

Disadvantages of supervised learning:

- Supervised learning models are not suitable for handling complex tasks.
- Supervised learning cannot predict the correct output if the test data is different from the training dataset.

- Training required lots of computation times.
- In supervised learning, we need enough knowledge about the classes of objects

- **Linear Regression**
- Linear regression is a statistical model used to predict the relationship between independent and dependent variables by examining two factors:
- Which variables, in particular, are significant predictors of the outcome variable?

- How significant is the regression line in terms of making predictions with the highest possible accuracy?
- To understand the terms “dependent” and “independent variable,” let’s take a real-world example.
- Imagine that we want to predict future crop yields based on the amount of rainfall, using data regarding past crops and rainfall amounts.

▪ Independent Variable

- The value of an independent variable does not change based on the effects of other variables.
- An independent variable is used to manipulate the dependent variable. It is often denoted by an “x.”
- In our example, the rainfall is the independent variable because we can’t control the rain, but the rain controls the

crop—the independent variable controls the dependent variable.

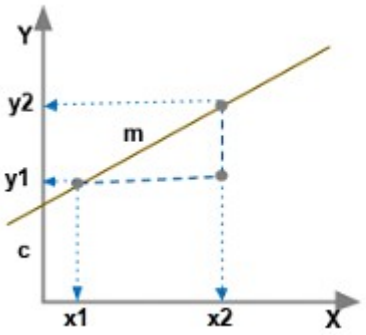
Dependent Variable

- The value of this variable changes when there is any change in the values of the independent variables, as mentioned before.
- It is often denoted by a “y.”

- In our example, the crop yield is the dependent variable, and it is dependent on the amount of rainfall.

Regression Equation

- The simplest linear regression equation with one dependent variable and one independent variable is:
- y = m*x + c
- Look at this graphic:



y —> Dependent Variable

x ---> Independent Variable

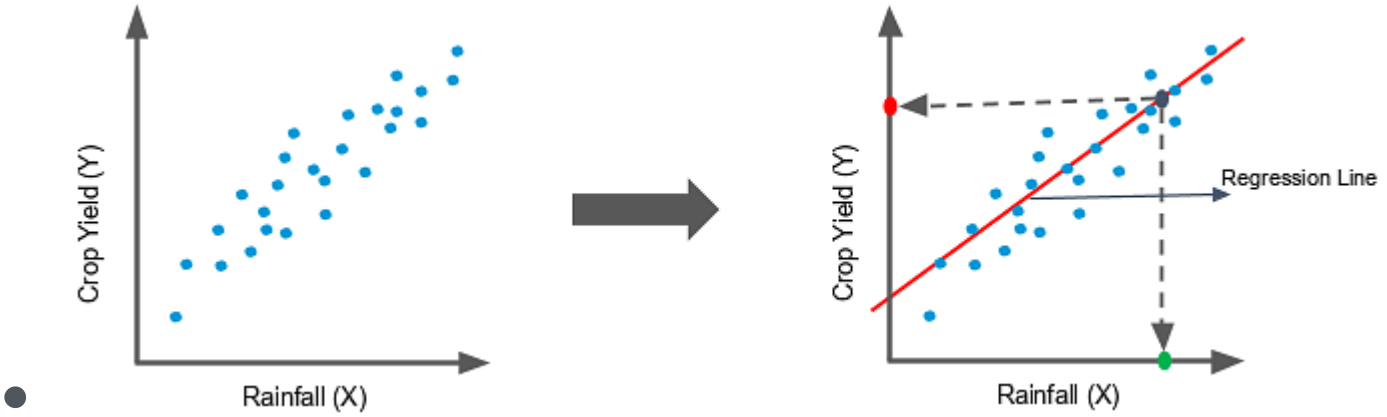
m —> Slope of the line

$m = \frac{y2 - y1}{x2 - x1}$

c ---> Coefficient of the line

- We have plotted two points, (x1,y1) and (x2,y2).
- Let’s discuss the example of crop yield used earlier and plot the crop yield based on the amount of rainfall.

- Here, rainfall is the independent variable and crop yield is the dependent variable.
- Consider these graphs:



- Here, we've drawn a line through the middle of the data.

- The red point on the y-axis is the crop yield you can expect for the amount of rainfall (x) represented by the green dot.
- If we have an idea about the amount of rainfall for a year, then we can predict how plentiful our crop will be.
- let us look at the reason behind the regression line.

Reasoning Behind the Regression Line

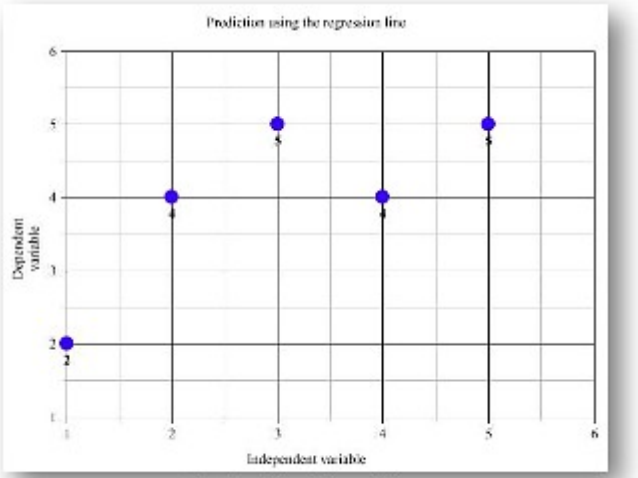
- Let’s consider a sample data set with five rows and find out how to draw the regression line.
- We’ll take two sets of data in which x is the independent variable and y is the dependent variable:

x	y

1	2
2	4
3	5

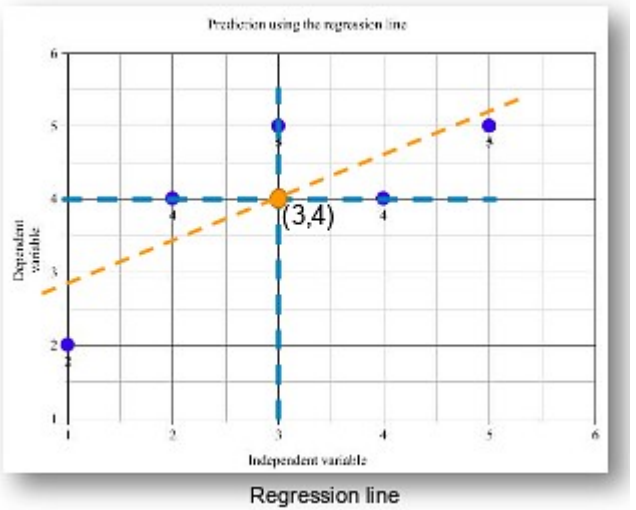
4	4
5	5

This is a graph with the data plotted:



Plotting the data points

- Next, we calculate the means, or average values, of x and y. The average of the x values is 3, and the average of the y values is 4.
- We plot both means on the graph to get the regression line.



- Now we'll discuss the regression line equation.

- The computation is:

	X	Y	(X^2)	(Y^2)	(X*Y)
	1	2	1	4	2
	2	4	4	16	8
	3	5	9	25	15
	4	4	16	16	16
	5	5	25	25	25
∑	∑ = 15	∑ = 20	∑ = 55	∑ = 86	∑ = 66

Y = m *X + c
= 0.6 *3 + 2.2
= 4

Linear equation is represented as Y = m X + c

$$m = \frac{((n * \sum XY) - (\sum X) * (\sum Y))}{((n * \sum X^2) - (\sum X)^2)} = \frac{((5 * 66) - (15 * 20))}{((5 * 55) - (225))} = 0.6$$

$$c = \frac{((\sum Y) * \sum (X^2) - (\sum X) * (\sum XY))}{((n * \sum X^2) - (\sum X)^2)} = 2.2$$

- We have calculated the values for x2, y2 and x*y to calculate the slope and intercept of the line. The calculated values are:

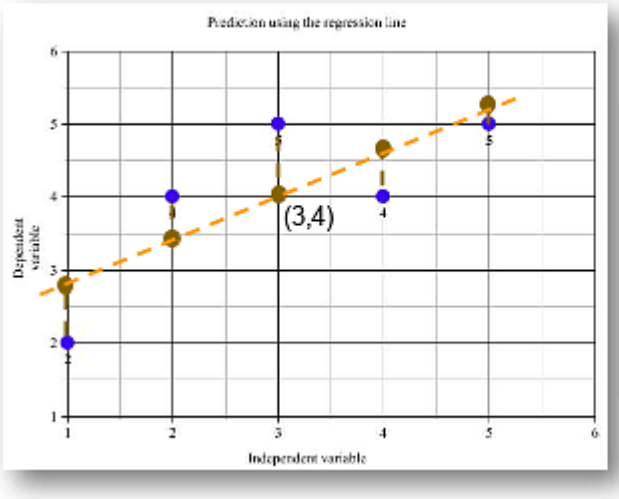
  m = 0.6

  c = 2.2

  The linear equation is:

  y = m*x + c

- Let’s find out the predicted values of y for corresponding values of x using the linear equation in which m = 0.6 and c = 2.2 and plot them.



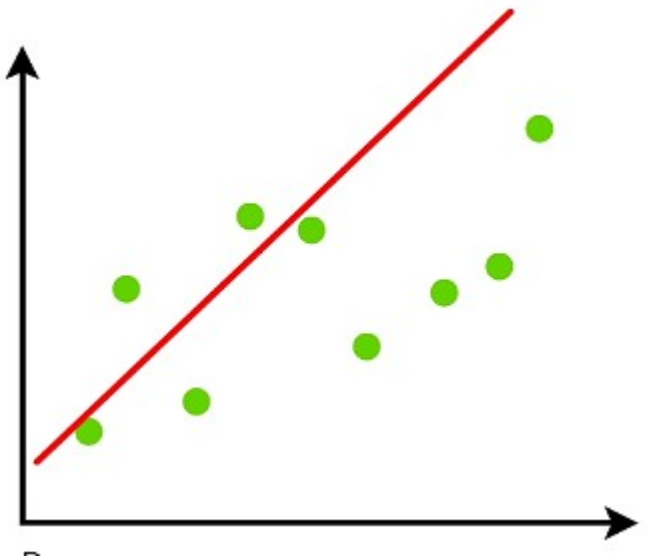
Y_{pred}
$Y=0.6*1+2.2=2.8$
$Y=0.6*2+2.2=3.4$
$Y=0.6*3+2.2=4$
$Y=0.6*4+2.2=4.6$
$Y=0.6*5+2.2=5.2$

- Here, the blue points represent the actual y values, and the brown points represent the predicted y values based on the model we created.
- The distances between the actual and predicted values are known as residuals or errors.
- The best-fit line should have the lowest sum of squares of these errors, also known as “e square.”

X	Y	Y_{pred}	$(Y-Y_{pred})$	$(Y-Y_{pred})^2$
1	2	2.8	-0.8	0.64
2	4	3.4	0.6	0.36
3	5	4	1	1
4	4	4.6	-0.6	0.36
5	5	5.2	-0.2	0.04
				$\sum = 2.4$

- You can observe that the sum of squared errors for this regression line is 2.4.

- We check this error for each line and determine the best-fit line having the lowest e square value.
- The graphical representation is:



- We keep the line moving through the data points to make sure the best-fit line has the least squared distance between the data points and the regression line.
- The above example shows the most commonly used formula for minimizing the distance.
- There are lots of ways to minimize the distance between the line and the data points, such as using the sum of

squared errors, the sum of absolute errors and the root mean square error.

- So far we have dealt with only two values, x and y.
- But it’s very rare in the real world to have only have two values when you’re calculating.
- Let’s talk about what happens when you have multiple inputs.

- While going through this Linear Regression in Python, let us stop by to learn Multiple Linear Regression and how it works by implementing in Python.

Multiple Linear Regression

- In simple linear regression, we have the equation:

- y = m*x + c

- For multiple linear regression, we have the equation:
- y = m1x1 + m2x2 + m3x3 +..... + c
- Here, we have multiple independent variables, x1, x2 and x3, and multiple slopes, m1, m2, m3 and so on.
- Implementation of Linear Regression

Example of Problem of MLR

- A venture capital firm is trying to figure out which companies it should invest in.
- We need to predict the profit of each company based on its expenses in research and development, marketing, administration and so on

Polynomial Regression

- Polynomial Regression is a regression algorithm that models the relationship between a dependent(y)
- and independent variable(x) as nth degree polynomial. The Polynomial Regression equation is given below:

- y= b0+b1x1+ b2x12+ b2x13+..... bnx1n
- It is also called the special case of Multiple Linear Regression in ML.
- Because we add some polynomial terms to the Multiple Linear regression equation to convert it into Polynomial Regression.

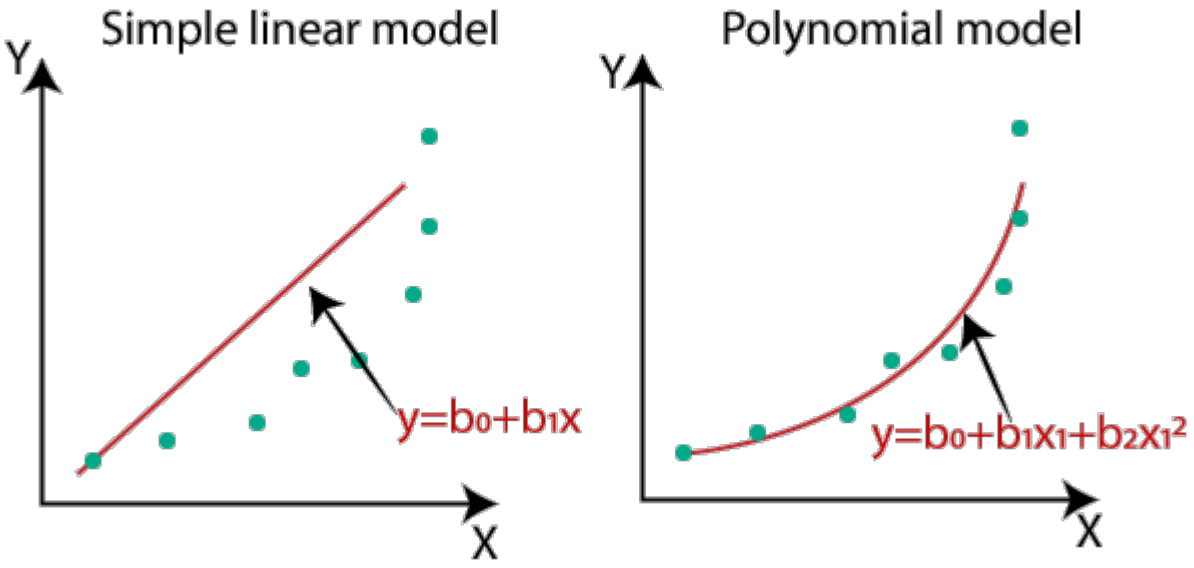
- It is a linear model with some modification in order to increase the accuracy.
- The dataset used in Polynomial regression for training is of non-linear nature.
- It makes use of a linear regression model to fit the complicated and non-linear functions and datasets.

- Hence, "In Polynomial regression, the original features are converted into Polynomial features of required degree (2,3,..,n) and then modeled using a linear model."
- Need for Polynomial Regression:
- The need of Polynomial Regression in ML can be understood in the below points:

- If we apply a linear model on a linear dataset, then it provides us a good result as we have seen in Simple Linear Regression
- but if we apply the same model without any modification on a non-linear dataset, then it will produce a drastic output.

- Due to which loss function will increase, the error rate will be high, and accuracy will be decreased.
- So for such cases, where data points are arranged in a non-linear fashion, we need the Polynomial Regression model.

- We can understand it in a better way using the below comparison diagram of the linear dataset and non-linear dataset.


Simple linear model
Y
X
y=b0+b1x
Polynomial model
Y
X
y=b0+b1x1+b2x1²

- in the above image, we have taken a dataset which is arranged non-linearly.
- So if we try to cover it with a linear model, then we can clearly see that it hardly covers any data point.
- On the other hand, a curve is suitable to cover most of the data points, which is of the Polynomial model.

- Hence, if the datasets are arranged in a non-linear fashion, then we should use the Polynomial Regression model instead of Simple Linear Regression.
- Equation of the Polynomial Regression Model:
- Simple Linear Regression equation:
- $y = b0+b1x$ .....(a)

- Multiple Linear Regression equation:
- $y= b0+b1x+ b2x2+ b3x3+....+ bnxn$ .....(b)
- Polynomial Regression equation: $y= b0+b1x + b2x2+ b3x3+....+ bnxn$ .....(c)
- When we compare the above three equations, we can clearly see that all three equations are Polynomial equations but differ by the degree of variables.

- The Simple and Multiple Linear equations are also Polynomial equations with a single degree, and the Polynomial regression equation is Linear equation with the nth degree.
- So if we add a degree to our linear equations, then it will be converted into Polynomial Linear equations.

APPLICATIONS OF POLYNOMIAL REGRESSION

- Engineering and Physics: In fields like engineering and physics, polynomial regression can be used to model complex relationships between variables.

- For instance, when studying the behavior of materials under stress or analyzing physical processes that don't follow a linear trend.
- Economics: Economic data often involves non-linear relationships.
- Polynomial regression can help in modeling economic trends that might exhibit curvature, such as

economic growth, inflation rates, and consumer demand over time.

- Biology and Medicine: Biological and medical data can have intricate patterns that aren't linear.
- Polynomial regression can be applied to model biological processes, drug interactions, and other medical phenomena.

- Environmental Science: Environmental data often involves intricate interactions between different factors.
- Polynomial regression can help in understanding complex relationships between variables in areas like climate science, ecology, and pollution studies.

- Finance: In finance, polynomial regression can be used to model financial data that shows nonlinear behavior, such as stock price movements or the relationship between interest rates and investment returns.
- Geology: Geoscientific data, like geological formations or the behavior of natural resources, might follow non-linear trends.

- Polynomial regression can be used to analyze such data to make predictions or understand underlying patterns.
- Market Research: In marketing, polynomial regression can be used to study consumer behavior and preferences, especially when trying to capture saturation effects or diminishing returns.

- Psychology and Social Sciences: Human behavior and social phenomena often exhibit complex patterns. Polynomial regression can be applied to model psychological or social variables that show nonlinear relationships.
- Astronomy and Astrophysics: Polynomial regression can be useful for fitting curves to astronomical data,

like light curves of variable stars or the orbits of celestial bodies.

- Image Analysis and Computer Vision: In some cases, polynomial regression can be employed for image analysis or computer vision tasks, where there might be non-linear relationships between image features and output variables.

- It's important to note that while polynomial regression can be effective in capturing non-linear relationships, it's also susceptible to overfitting when higher-degree polynomials are used.
- Overfitting occurs when the model fits the training data too closely and doesn't generalize well to new, unseen data. Regularization techniques and model evaluation are crucial to mitigate this issue.

Additionally, other advanced techniques like spline regression or machine learning algorithms might also be considered depending on the complexity of the data and the specific application.

Locally Weighted Linear Regression

- Locally Weighted Linear Regression

- Within the field of machine learning and regression analysis, Locally Weighted Linear Regression (LWLR) emerges as a notable approach that bolsters predictive accuracy through the integration of local adaptation.
- In contrast to conventional linear regression models, which presume a universal correlation among variables, LWLR acknowledges the significance of

localized patterns and relationships present in the data.

- In the subsequent discourse, we embark on an exploration of the fundamental principles, diverse applications, and inherent advantages offered by Locally Weighted Linear Regression.

- Our aim is to shed light on its exceptional capacity to amplify predictive prowess and furnish intricate understandings of intricate datasets.
- Fundamentally, LWLR manifests as a non-parametric regression algorithm that discerns the connection between a dependent variable and several independent variables.

- Notably, LWLR's distinctiveness emanates from its dynamic adaptability, which empowers it to bestow distinct weights upon individual data points contingent on their proximity to the target point under prediction.
- In essence, this algorithm accords greater significance to proximate data points, deeming them

as more influential contributors in the prediction process.

- Principles of Locally Weighted Linear Regression
- LWLR functions on the premise that the association between the dependent and independent variables adheres to linearity

- however, this relationship is allowed to exhibit variability across distinct sections within the dataset.
- This is achieved by employing an individual linear regression model for each prediction, employing a weighted least squares technique.
- The determination of weights is carried out through a kernel function, which bestows elevated weights

upon data points in close proximity to the target point and diminishes the weights for those that are farther away.

- Applications of Locally Weighted Linear Regression
- Time Series Analysis: LWLR is particularly useful in time series analysis, where the relationship between variables may change over time.

- By adapting to the local patterns and trends, LWLR can capture the dynamics of time-varying data and make accurate predictions.
- Anomaly Detection: LWLR can be employed for anomaly detection in various domains, such as fraud detection or network intrusion detection.

- By identifying deviations from the expected patterns in a localized manner, LWLR helps detect abnormal behavior that may go unnoticed using traditional regression models.
- Robotics and Control Systems: In robotics and control systems, LWLR can be utilized to model and predict the behavior of complex systems.

- By adapting to local conditions and variations, LWLR enables precise control and decision-making in dynamic environments.
- Benefits of Locally Weighted Linear Regression
- Improved Predictive Accuracy: By considering local patterns and relationships, LWLR can capture subtle

nuances in the data that might be overlooked by global regression models.

- This results in more accurate predictions and better model performance.
- Flexibility and Adaptability: LWLR can adapt to different regions of the dataset, making it suitable for complex and non-linear relationships.

- It offers flexibility in capturing local variations, allowing for more nuanced analysis and insights.
- Interpretable Results: Despite its adaptive nature, LWLR still provides interpretable results.
- The localized models offer insights into the relationships between variables within specific

regions of the data, aiding in the understanding of complex phenomena.

Numerical optimization

- Numerical optimization is a field of mathematics and computer science that deals with finding the best possible solution to a problem from a set of

possible solutions

- where the "best" solution is defined based on a certain objective function or criteria.
- This field has applications in various domains, including engineering, economics, machine learning, physics, and more.
- In numerical optimization, the goal is to find the input values (often referred to as parameters or

variables) that minimize or maximize the objective function.

- These input values are often subject to certain constraints, which can be equality or inequality conditions that the solution must satisfy.
- Here are some key concepts and methods commonly used in numerical optimization:

Objective Function: This is the function that

needs to be optimized.

- It could represent, for example, the cost to be minimized or the value to be maximized.

- Constraints: These are the conditions that the solution must satisfy.
- Constraints can be classified into equality constraints (where the constraint function is

required to be equal to a certain value)

- and inequality constraints (where the constraint function must be greater than or less than a certain value).
- Local and Global Optima: A local optimum is a solution that is the best within a certain neighborhood, but it might not be the best solution globally.

- A global optimum is the best solution across the entire feasible region.
- Search Methods: Numerical optimization algorithms use various strategies to search for optimal solutions.
- Some methods are gradient-based, utilizing information from the gradient (and sometimes the Hessian) of the objective function.

- Other methods are gradient-free, often referred to as direct search methods, which explore the parameter space without using gradient information.
- Gradient Descent: This is a widely used optimization algorithm that iteratively adjusts the parameter values in the direction of the negative gradient of the objective function, aiming to reach a local minimum.

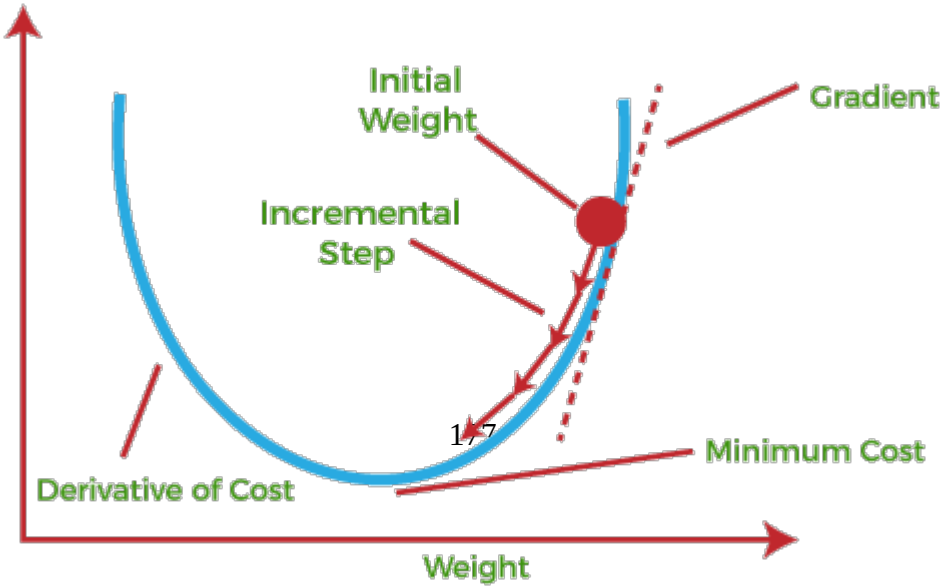
Gradient descent

- Gradient Descent is known as one of the most commonly used optimization algorithms to train machine learning models by means of minimizing errors between actual and expected results.

- Further, gradient descent is also used to train Neural Networks.
- n mathematical terminology, Optimization algorithm refers to the task of minimizing/maximizing an objective function f(x) parameterized by x.
- Similarly, in machine learning, optimization is the task of minimizing the cost function parameterized by the model's parameters.

What is Gradient Descent or Steepest Descent?

- Gradient descent was initially discovered by "Augustin-Louis Cauchy" in mid of 18th century.
- Gradient Descent is defined as one of the most commonly used iterative optimization algorithms of machine learning to train the machine learning and deep learning models


Initial
Weight
Gradient
Incremental
Step
Minimum Cost
Derivative of Cost
Weight

- If we move towards a negative gradient or away from the gradient of the function at the current point
- it will give the local minimum of that function.

- Whenever we move towards a positive gradient or towards the gradient of the function at the current point
- we will get the local maximum of that function.

What is Cost-function?

- he cost function is defined as the measurement of difference
- or error between actual values and expected values at the current

position and present in the form of a single real number.

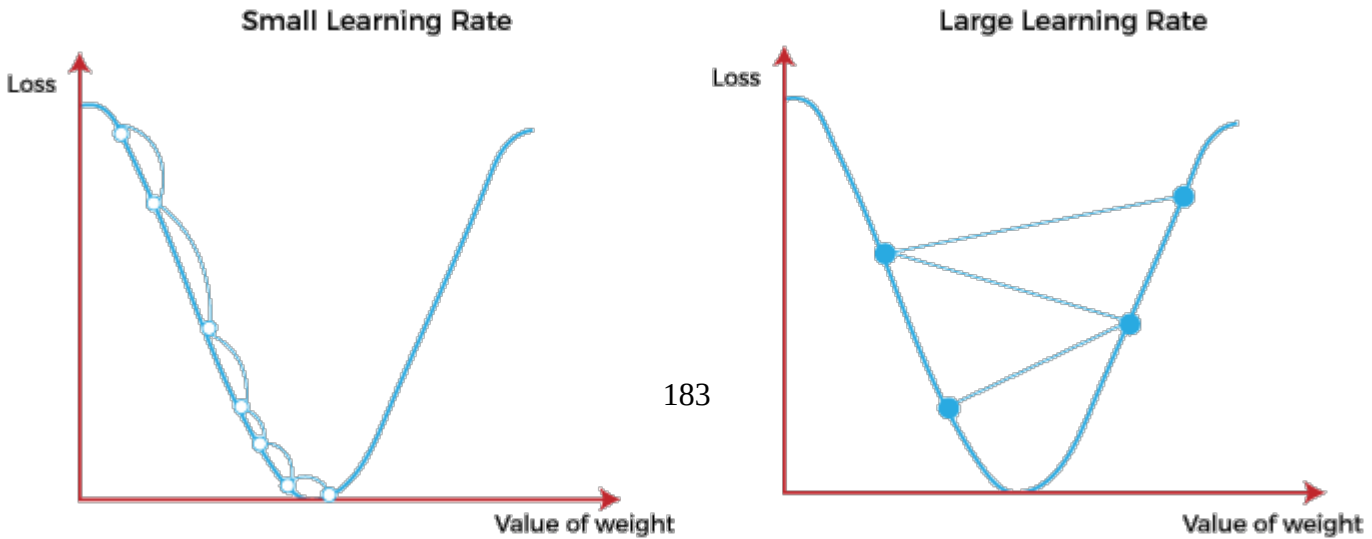
- It helps to increase and improve machine learning efficiency by providing feedback to this model
- so that it can minimize error and find the local or global minimum.
- The cost function is calculated after making a hypothesis with initial parameters and modifying these parameters using gradient descent algorithms over known data to reduce the cost function.

- Hypothesis:
- Parameters:
- Cost function:
- Goal:

Learning Rate:

- It is defined as the step size taken to reach the minimum or lowest point.

- This is typically a small value that is evaluated and updated based on the behavior of the cost function.
- If the learning rate is high, it results in larger steps but also leads to risks of overshooting the minimum.
- At the same time, a low learning rate shows the small step sizes, which compromises overall efficiency but gives the advantage of more precision.


Small Learning Rate
Loss
Value of weight
Large Learning Rate
Loss
Value of weight

Types of Gradient Descent

- Batch gradient descent (BGD) is used to find the error for each point in the training set and update the model after evaluating all training examples. This procedure is known as the training epoch. In simple words, it is a greedy approach where we have to sum

over all examples for each update.

-
- Advantages of Batch gradient descent:
- It produces less noise in comparison to other gradient descent.
- It produces stable gradient descent convergence.
- It is Computationally efficient as all resources are used for all training samples.

Stochastic gradient descent

- Stochastic gradient descent (SGD) is a type of gradient descent that runs one training example per iteration.
- Or in other words, it processes a training epoch for each example within a dataset and updates each training example's parameters one at a time.
- As it requires only one training example at a time, hence it is

easier to store in allocated memory.

Advantages of Stochastic gradient descent:

- It is easier to allocate in desired memory.
- It is relatively fast to compute than batch gradient descent.
- It is more efficient for large datasets.

Kernel methods

- A set of techniques known as kernel methods are used in machine learning to address classification, regression, and other prediction issues.

- They are built around the idea of kernels, which are functions that gauge how similar two data points are to one another in a high-dimensional feature space.
- Kernel methods' fundamental premise is used to convert the input data into a high-dimensional feature space,
- which makes it simpler to distinguish between classes or generate predictions.

Kernel Method in SVMs

- Support Vector Machines (SVMs) use kernel methods to transform the input data into a higher-dimensional feature space, which makes it simpler to distinguish between classes
- The kernel function in SVMs is essential in determining the decision boundary that divides the various classes
- Basically, kernel methods in SVMs are a powerful technique for solving classification and regression problems
- and they are widely used in machine learning because they can

handle complex data structures and are robust to noise and outliers.

Characteristics of Kernel Function

- Mercer's condition: A kernel function must satisfy Mercer's condition to be valid. This condition ensures that the kernel function is positive semi definite, which means that it is always greater than or equal to zero.
- Positive definiteness: A kernel function is positive definite if it is

always greater than zero except for when the inputs are equal to each other.

- Non-negativity: A kernel function is non-negative, meaning that it produces non-negative values for all inputs.
- Symmetry: A kernel function is symmetric, meaning that it produces the same value regardless of the order in which the inputs are given.
- Reproducing property: A kernel function satisfies the reproducing

property if it can be used to reconstruct the input data in the feature space.

- Smoothness: A kernel function is said to be smooth if it produces a smooth transformation of the input data into the feature space.
- Complexity: The complexity of a kernel function is an important consideration, as more complex kernel functions may lead to over fitting and reduced generalization performance.

Linear Kernel

- A linear kernel is a type of kernel function used in machine learning, including in SVMs (Support Vector Machines).
- It is the simplest and most commonly used kernel function, and it defines the dot product between the input vectors in the original feature space.

Polynomial Kernel

- A particular kind of kernel function utilised in machine learning, such as in SVMs, is a polynomial kernel (Support Vector

Machines).

- It is a nonlinear kernel function that employs polynomial functions to transfer the input data into a higher-dimensional feature space.

Gaussian (RBF) Kernel

- The Gaussian kernel, also known as the radial basis function (RBF) kernel, is a popular kernel function used in machine learning, particularly in SVMs (Support Vector Machines).

- It is a nonlinear kernel function that maps the input data into a higher-dimensional feature space using a Gaussian function

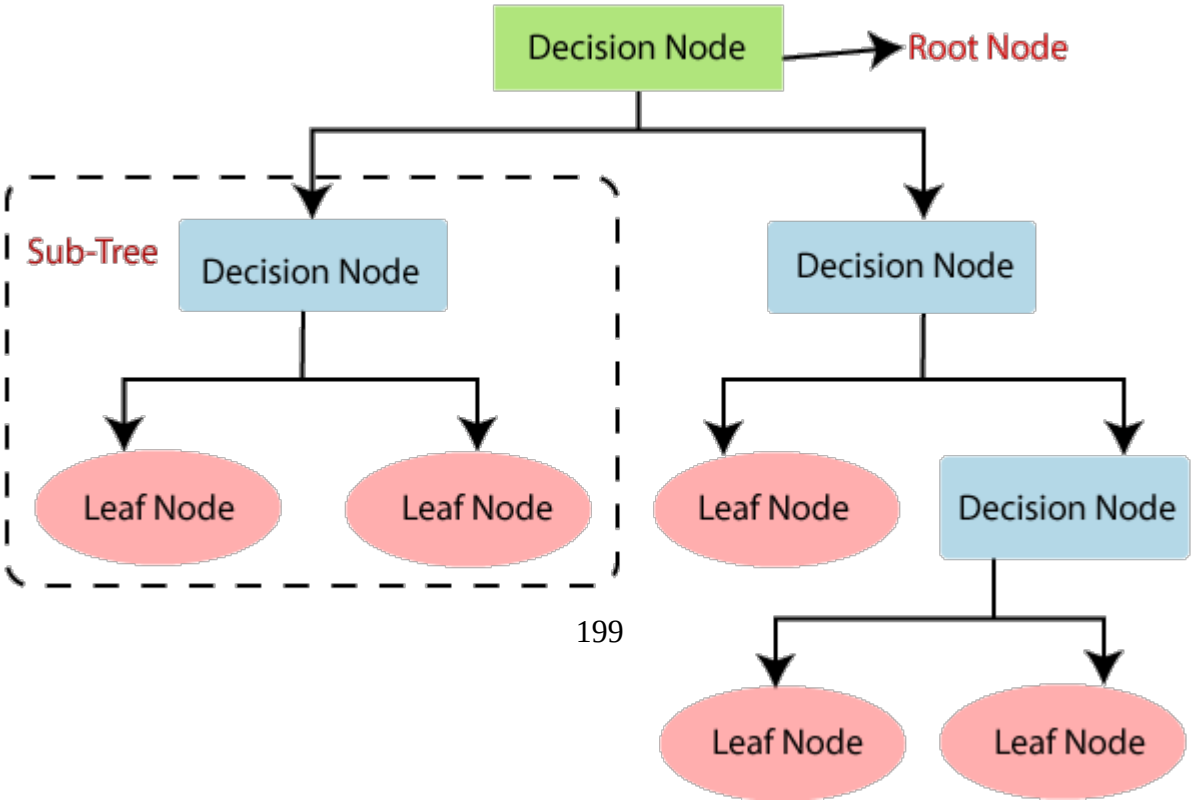
Laplace Kernel

- The Laplacian kernel, also known as the Laplace kernel or the exponential kernel, is a type of kernel function used in machine learning, including in SVMs (Support Vector Machines).
- It is a non-parametric kernel that can be used to measure the similarity or distance between two input feature vectors.

Decision Trees

- Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems,
- but mostly it is preferred for solving Classification problems.

- In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node.
- Decision nodes are used to make any decision and have multiple branches
- whereas Leaf nodes are the output of those decisions and do not contain any further branches.
- A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees


Decision Node
Root Node
Sub-Tree
Decision Node
Decision Node
Leaf Node
Leaf Node
Leaf Node
Decision Node
Leaf Node
Leaf Node

Why use Decision Trees?

- There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model.

- Below are the two reasons for using the Decision tree:
-
- Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.
- The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Decision Tree Terminologies

- Root Node: Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
- Leaf Node: Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.
- Splitting: Splitting is the process of dividing the decision node/root node into sub-nodes according to the given conditions.
- Branch/Sub Tree: A tree formed by splitting the tree.

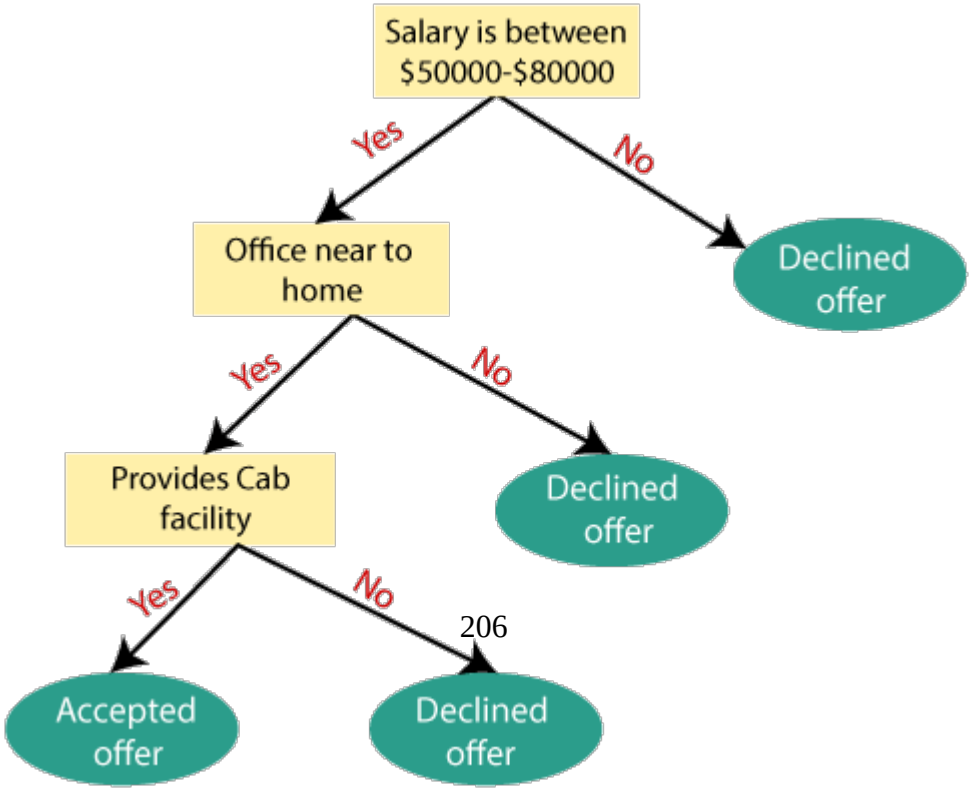
- Pruning: Pruning is the process of removing the unwanted branches from the tree.
- Parent/Child node: The root node of the tree is called the parent node, and other nodes are called the child nodes.

How does the Decision Tree algorithm Work?

- Step-1: Begin the tree with the root node, says S, which contains the complete dataset.

- Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- Step-3: Divide the S into subsets that contains possible values for the best attributes.
- Step-4: Generate the decision tree node, which contains the best attribute.
- Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3.

- Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.
- **Example**: Suppose there is a candidate who has a job offer and wants to decide whether he should accept the offer or Not.
- So, to solve this problem, the decision tree starts with the root node (Salary attribute by ASM).


Salary is between $50000-$80000
Yes
No
Office near to home
Declined offer
Yes
No
Provides Cab facility
Declined offer
Yes
No
Accepted offer
Declined offer

Attribute Selection Measures

- While implementing a Decision tree, the main issue arises that how to select the best attribute for the root node and for sub-nodes.

- So, to solve such problems there is a technique which is called as Attribute selection measure or ASM.

Information Gain:

- Information gain is the measurement of changes in entropy after the segmentation of a dataset based on an attribute.
- It calculates how much information a feature provides us about a class.
- According to the value of information gain, we split the node and

build the decision tree.

- ***Information Gain= Entropy(S)- [(Weighted Avg) *Entropy(each feature)***
- Entropy: Entropy is a metric to measure the impurity in a given attribute. It specifies randomness in data.

Gini Index:

- Gini index is a measure of impurity or purity used while creating a decision tree in the CART(Classification and Regression Tree)

algorithm.

- An attribute with the low Gini index should be preferred as compared to the high Gini index.
- It only creates binary splits, and the CART algorithm uses the Gini index to create binary splits.

Pruning: Getting an Optimal Decision tree

- Pruning is a process of deleting the unnecessary nodes from a tree in order to get the optimal decision tree.

- A too-large tree increases the risk of overfitting, and a small tree may not capture all the important features of the dataset.

Advantages of the Decision Tree

- It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
- It can be very useful for solving decision-related problems.
- It helps to think about all the possible outcomes for a problem.

- There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

- The decision tree contains lots of layers, which makes it complex.
- It may have an overfitting issue, which can be resolved using the Random Forest algorithm.
- For more class labels, the computational complexity of the decision tree may increase.

Nearest-Neighbor Classifiers

- K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
- K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is

most similar to the available categories.

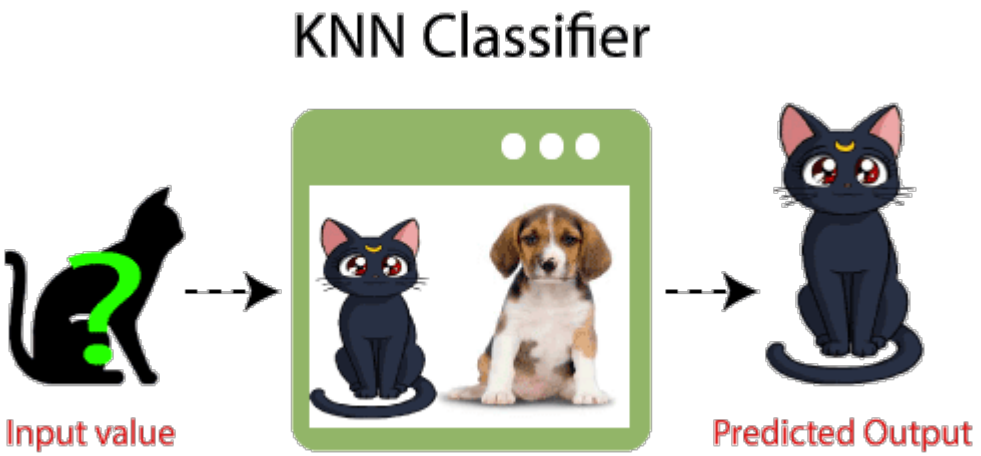
- K-NN algorithm stores all the available data and classifies a new data point based on the similarity.
- This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.
- K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems.
- K-NN is a non-parametric algorithm, which means it does not

make any assumption on underlying data.

- It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset.
- KNN algorithm at the training phase just stores the dataset and when it gets new data
- then it classifies that data into a category that is much similar to the new data.

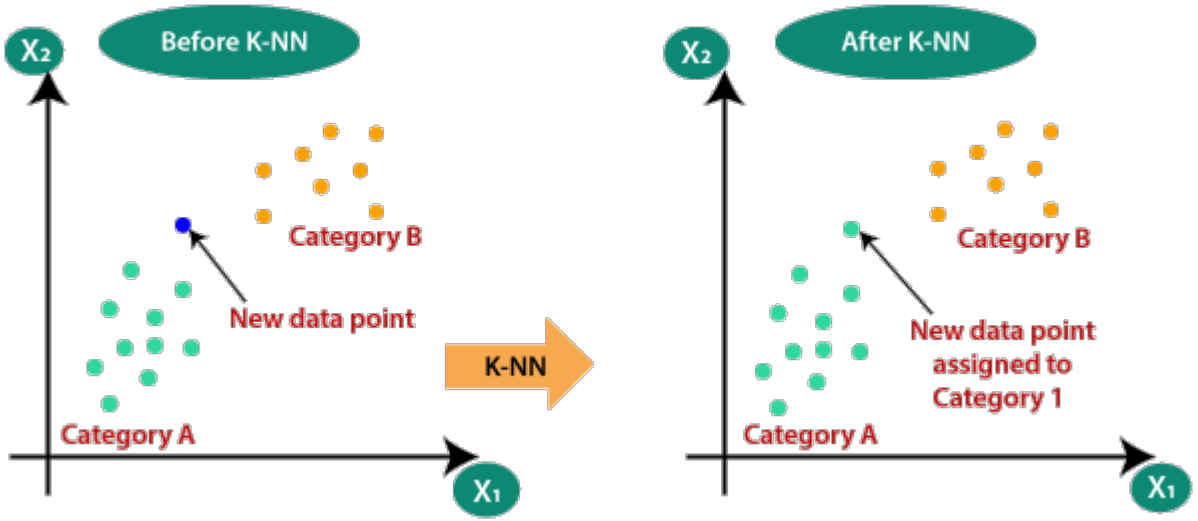
- **Example**: Suppose, we have an image of a creature that looks similar to cat and dog
- but we want to know either it is a cat or dog.
- So for this identification, we can use the KNN algorithm, as it works on a similarity measure.
- Our KNN model will find the similar features of the new data set to the cats and dogs images
- based on the most similar features it will put it in either cat or dog

category.



Why do we need a K-NN Algorithm?

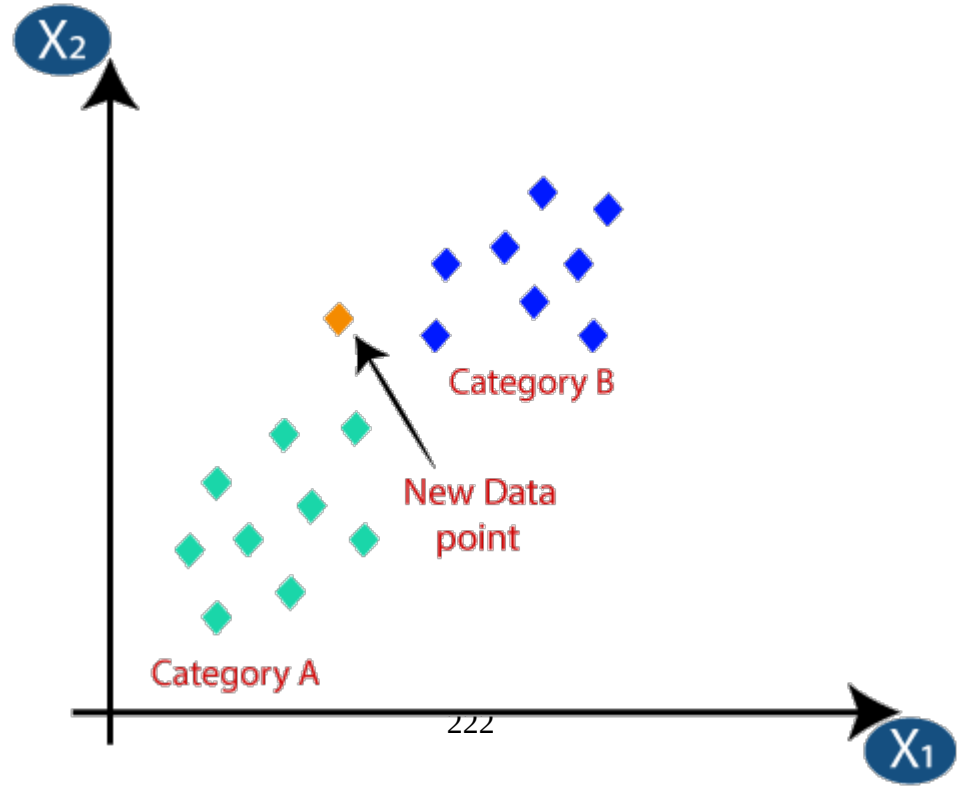
- Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories.
- To solve this type of problem, we need a K-NN algorithm. With the help of K-NN
- we can easily identify the category or class of a particular dataset. Consider the below diagram:


Before K-NN
X2
Category B
New data point
Category A
X1
K-NN
After K-NN
X2
Category B
New data point
assigned to
Category 1
Category A
X1

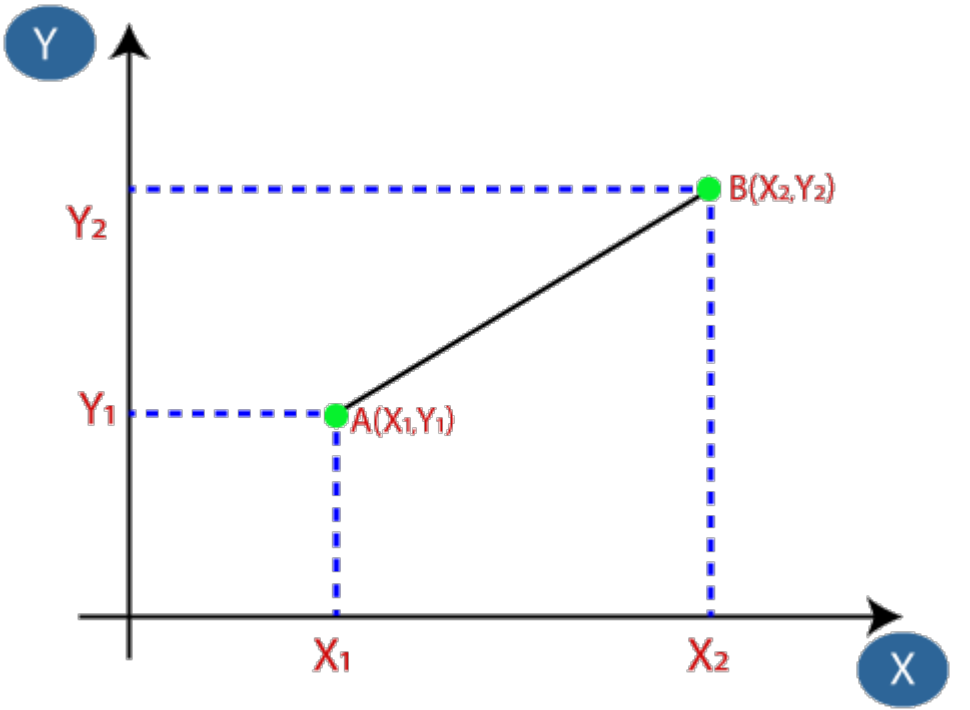
How does K-NN work?

- The K-NN working can be explained on the basis of the below algorithm:
- Step-1: Select the number K of the neighbors
- Step-2: Calculate the Euclidean distance of K number of neighbors
- Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

- Step-4: Among these k neighbors, count the number of the data points in each category.
- Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.
- Step-6: Our model is ready.
- Suppose we have a new data point and we need to put it in the required category. Consider the below image:

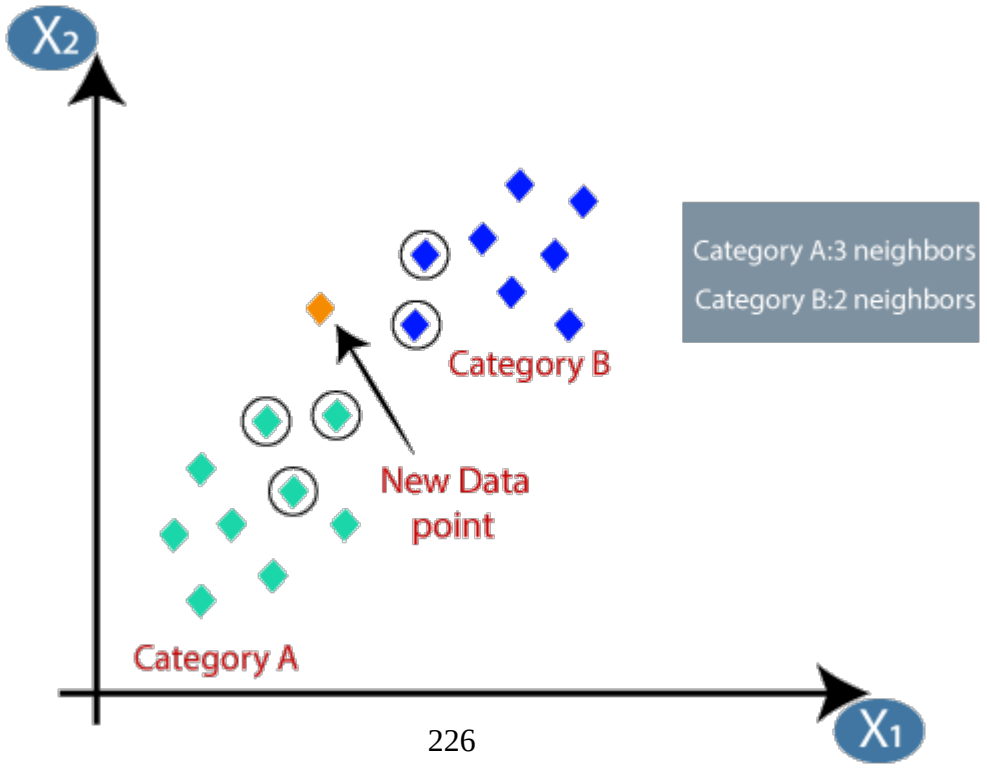

X2
Category B
New Data
point
Category A
X1

- Firstly, we will choose the number of neighbors, so we will choose the k=5.
- Next, we will calculate the Euclidean distance between the data points.
- The Euclidean distance is the distance between two points, which we have already studied in geometry.
- It can be calculated as:



Euclidean Distance between A_1 and $B_2 = \sqrt{(X_2-X_1)^2+(Y_2-Y_1)^2}$

- By calculating the Euclidean distance we got the nearest neighbors, as three nearest neighbors in category A
- and two nearest neighbors in category B. Consider the below image:


X2
X1
Category A:3 neighbors
Category B:2 neighbors
Category B
New Data point
Category A

- As we can see the 3 nearest neighbors are from category A, hence this new data point must belong to category A.

Advantages of KNN Algorithm:

- It is simple to implement.
- It is robust to the noisy training data
- It can be more effective if the training data is large.

Disadvantages of KNN Algorithm:

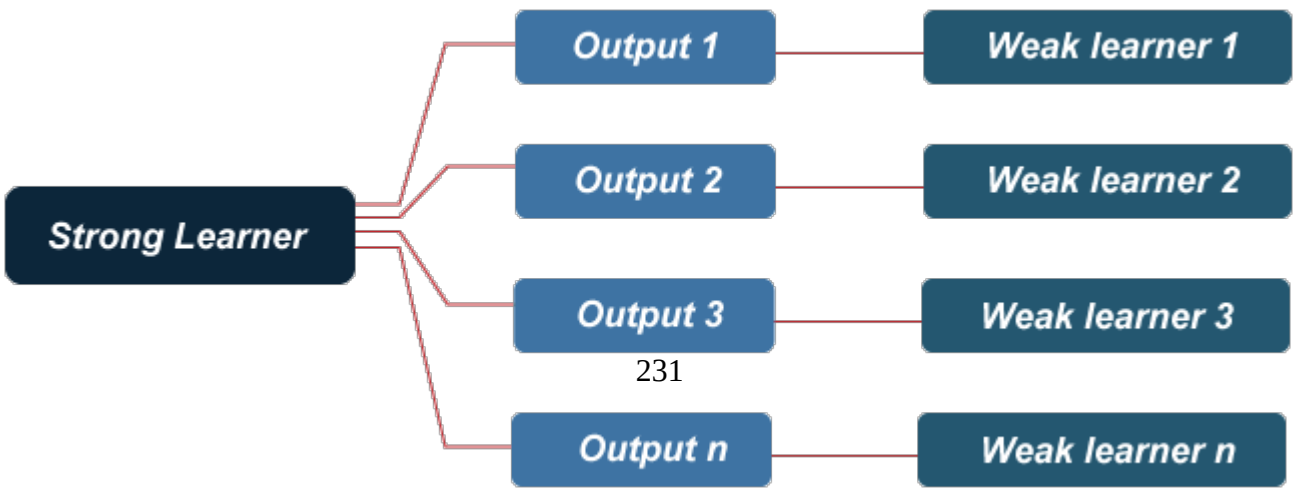
- Always needs to determine the value of K which may be complex some time.
- The computation cost is high because of calculating the distance between the data points for all the training samples.

What is Boosting?

- Boosting is an ensemble learning method that combines a set of weak learners into strong learners to minimize training errors.
- In boosting, a random sample of data is selected, fitted with a

model, and then trained sequentially.

- Boosting is an efficient algorithm that converts a weak learner into a strong learner. They use the concept of the weak learner and strong learner conversation through the weighted average values and higher votes values for prediction.


Strong Learner
Output 1
Output 2
Output 3
Output n
Weak learner 1
Weak learner 2
Weak learner 3
Weak learner n

Example

- Let's understand this concept with the help of the following example.
- Let's take the example of the email.

- How will you recognize your email, whether it is spam or not? You can recognize it by the following conditions:
- If an email contains lots of sources, that means it is spam.
- If an email contains only one file image, then it is spam.
- If an email contains the message "You Own a lottery of $xxxxx," it is spam.
- If an email contains some known source, then it is not spam.

- If it contains the official domain like educba.com, etc., it is not spam.
- The rules mentioned above are not that powerful to recognize spam or not; hence these rules are called weak learners.

Why is Boosting Used?

- To solve complicated problems, we require more advanced techniques.
- Suppose that, given a data set of images containing images of cats

and dogs

- you were asked to build a model that can classify these images into two separate classes.
- Like every other person, you will start by identifying the images by using some rules given below:
- The image has pointy ears: Cat
- The image has cat-shaped eyes: Cat

- The image has bigger limbs: Dog
- The image has sharpened claws: Cat
- The image has a wider mouth structure: Dog

How does the Boosting Algorithm Work?

- The basic principle behind the working of the boosting algorithm is to generate multiple weak learners and combine their

predictions to form one strict rule.

- These weak rules are generated by applying base Machine Learning algorithms on different distributions of the data set.
- After multiple iterations, the weak learners are combined to form a strong learner that will predict a more accurate outcome


Decision stumps

Here's how the algorithm works:

- Step 1: The base algorithm reads the data and assigns equal weight to each sample observation.

-

- Step 2: False predictions made by the base learner are identified. In the next iteration, these false predictions are assigned to the next base learner with a higher weightage on these incorrect predictions.

- Step 3: Repeat step 2 until the algorithm can correctly classify the output.

- Therefore, the main aim of Boosting is to focus more on miss-classified predictions.

  Types of Boosting

- **Adaptive boosting or AdaBoost**: This method operates iteratively, identifying misclassified data points and adjusting their weights to minimize the training error.

- The model continues to optimize sequentially until it yields the strongest predictor.
- **Gradient Boosting**: Gradient Boosting is also based on sequential ensemble learning.
- Here the base learners are generated sequentially so that the present base learner is always more effective than the previous one
- i.e., and the overall model improves sequentially with each iteration.

Benefits and Challenges of Boosting

- Ease of Implementation: Boosting can be used with several hyper-parameter tuning options to improve fitting.
- No data preprocessing is required, and boosting algorithms have built-in routines to handle missing data
- **Reduction of bias:** Boosting algorithms combine multiple weak learners in a sequential method, iteratively improving upon observations.

- This approach can help to reduce high bias, commonly seen in shallow decision trees and logistic regression models.
- **Computational Efficiency:** Since boosting algorithms have special features that increase their predictive power during training, it can help reduce dimensionality and increase computational efficiency.

Challenges of boosting include:

- Overfitting: There's some dispute in the research around whether

or not boosting can help reduce overfitting or make it worse.

- We include it under challenges because in the instances that it does occur, predictions cannot be generalized to new datasets.
- **Intense computation**: Sequential training in boosting is hard to scale up.
- **Vulnerability to outlier data**: Boosting models are vulnerable to outliers or data values that are different from the rest of the dataset.

- Because each model attempts to correct the faults of its predecessor, outliers can skew results significantly.
- **Real-time implementation:** You might find it challenging to use boosting for real-time implementation because the algorithm is more complex than other processes.
- Boosting methods have high adaptability, so you can use various model parameters that immediately affect the model's performance.

Applications of Boosting

- Healthcare: Boosting is used to lower errors in medical data predictions, such as predicting cardiovascular risk factors and cancer patient survival rates.
- **IT**: Gradient boosted regression trees are used in search engines for page rankings, while the Viola-Jones boosting algorithm is used for image retrieval.

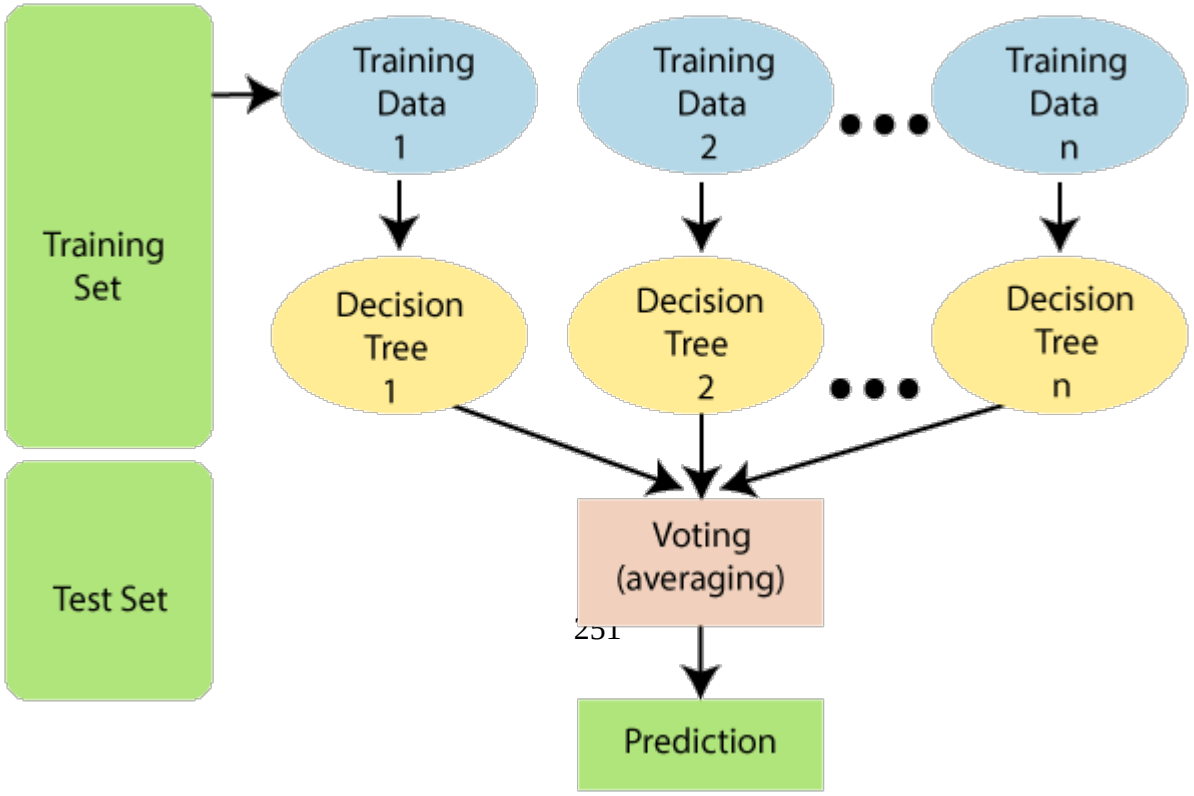
- **Finance**: Boosting is used with deep learning models to automate critical tasks, including fraud detection, pricing analysis, and more

Random Forests

- Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique.

- It can be used for both Classification and Regression problems in ML.
- It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model.
- As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that

dataset."


Training Set
Test Set
Training Data 1
Training Data 2
Training Data n
Decision Tree 1
Decision Tree 2
Decision Tree n
Voting (averaging)
Prediction

- **Assumptions for Random Forest**
- Since the random forest combines multiple trees to predict the class of the dataset, it is possible that some decision trees may predict the correct output, while others may not.
- But together, all the trees predict the correct output.
- Therefore, below are two assumptions for a better Random forest classifier:
- There should be some actual values in the feature variable of the

dataset so that the classifier can predict accurate results rather than a guessed result.

- The predictions from each tree must have very low correlations.

Why use Random Forest?

- Below are some points that explain why we should use the Random Forest algorithm:
- It takes less training time as compared to other algorithms.

- It predicts output with high accuracy, even for the large dataset it runs efficiently.
- It can also maintain accuracy when a large proportion of data is missing.

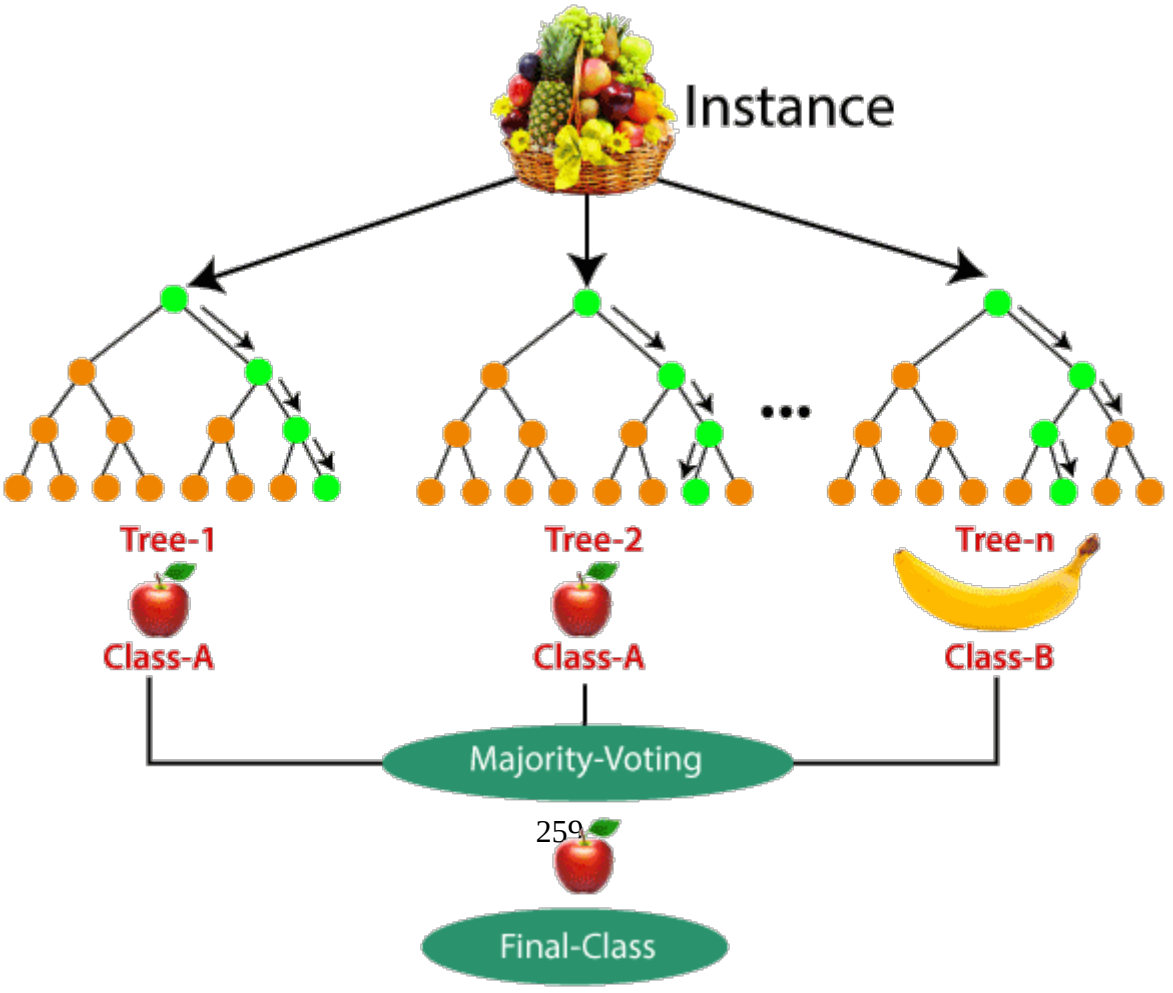
How does Random Forest algorithm work?

- Random Forest works in two-phase first is to create the random forest by combining N decision tree, and second is to make predictions for each tree created in the first phase.

- The Working process can be explained in the below steps and diagram:
- Step-1: Select random K data points from the training set.
- Step-2: Build the decision trees associated with the selected data points (Subsets).
- Step-3: Choose the number N for decision trees that you want to build.
- Step-4: Repeat Step 1 & 2.

- Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.
- The working of the algorithm can be better understood by the below example:
- **Example**: Suppose there is a dataset that contains multiple fruit images.
- So, this dataset is given to the Random forest classifier.

- The dataset is divided into subsets and given to each decision tree.
- During the training phase, each decision tree produces a prediction result
- and when a new data point occurs, then based on the majority of results, the Random Forest classifier predicts the final decision. Consider the below image:


Instance
Tree-1
Tree-2
Tree-n
Class-A
Class-A
Class-B
Majority-Voting
Final-Class

Applications of Random Forest

- There are mainly four sectors where Random forest mostly used:
- Banking: Banking sector mostly uses this algorithm for the

identification of loan risk.

- Medicine: With the help of this algorithm, disease trends and risks of the disease can be identified.
- Land Use: We can identify the areas of similar land use by this algorithm.
- Marketing: Marketing trends can be identified using this algorithm.

Advantages of Random Forest

- Random Forest is capable of performing both Classification and Regression tasks.
- It is capable of handling large datasets with high dimensionality.
- It enhances the accuracy of the model and prevents the overfitting issue.

Disadvantages of Random Forest

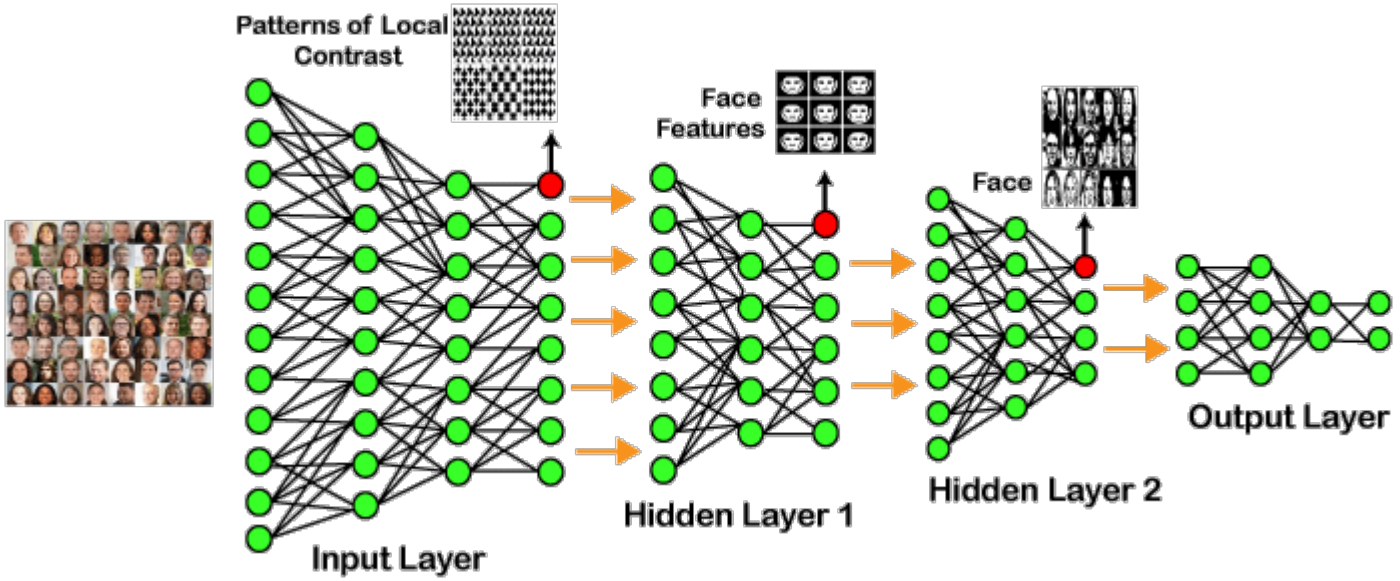
- Although random forest can be used for both classification and regression tasks, it is not more suitable for Regression tasks.

Deep Neural Networks

- Deep learning is based on the branch of machine learning, which is a subset of artificial intelligence.
- Since neural networks imitate the human brain and so deep learning will do. In deep learning, nothing is programmed explicitly
- Deep learning models are capable enough to focus on the accurate features themselves by requiring a little guidance from the

programmer and are very helpful in solving out the problem of dimensionality

- Deep learning is implemented with the help of Neural Networks
- and the idea behind the motivation of Neural Network is the biological neurons, which is nothing but a brain cell.
- **Example of Deep Learning**


Patterns of Local Contrast
Face Features
Face
Input Layer
Hidden Layer 1
Hidden Layer 2
Output Layer

- In the example given above, we provide the raw data of images to

the first layer of the input layer.

- After then, these input layer will determine the patterns of local contrast that means it will differentiate on the basis of colors, luminosity, etc.
- Then the 1st hidden layer will determine the face feature, i.e., it will fixate on eyes, nose, and lips, etc.
- And then, it will fixate those face features on the correct face template.

- So, in the 2nd hidden layer, it will actually determine the correct face here as it can be seen in the above image, after which it will be sent to the output layer

Architectures

Deep Neural Networks

- It is a neural network that incorporates the complexity of a certain level, which means several numbers of hidden layers are encompassed in between the input and output layers.

- They are highly proficient on model and process non-linear associations.

Deep Belief Networks

- A deep belief network is a class of Deep Neural Network that comprises of multi-layer belief networks.

Steps to perform DBN:

- With the help of the Contrastive Divergence algorithm, a layer of features is learned from perceptible units.

- Next, the formerly trained features are treated as visible units, which perform learning of features.
- Lastly, when the learning of the final hidden layer is accomplished, then the whole DBN is trained.

Recurrent Neural Networks

- It permits parallel as well as sequential computation, and it is exactly similar to that of the human brain (large feedback network of connected neurons).

- Since they are capable enough to reminisce all of the imperative things related to the input they have received, so they are more precise.

Types of Deep Learning Networks

Feed Forward Neural Network

- A feed-forward neural network is none other than an Artificial Neural Network, which ensures that the nodes do not form a cycle.
- In this kind of neural network, all the perceptrons are organized

within layers, such that the input layer takes the input, and the output layer generates the output.

- Since the hidden layers do not link with the outside world, it is named as hidden layers.
- Each of the perceptrons contained in one single layer is associated with each node in the subsequent layer.

Applications:

- Data Compression

- Pattern Recognition
- Computer Vision
- Sonar Target Recognition
- Speech Recognition
- Handwritten Characters Recognition

Recurrent Neural Network

- Recurrent neural networks are yet another variation of feed-

forward networks.

- Here each of the neurons present in the hidden layers receives an input with a specific delay in time.
- The Recurrent neural network mainly accesses the preceding info of existing iterations.
- For example, to guess the succeeding word in any sentence, one must have knowledge about the words that were previously used.

Applications:

- Machine Translation
- Robot Control
- Time Series Prediction
- Speech Recognition
- Speech Synthesis
- Time Series Anomaly Detection
- Rhythm Learning

- Music Composition

  Convolutional Neural Network

- Convolutional Neural Networks are a special kind of neural network mainly used for image classification, clustering of images and object recognition.
- DNNs enable unsupervised construction of hierarchical image representations.
- To achieve the best accuracy, deep convolutional neural networks

are preferred more than any other neural network.

Applications:

- Identify Faces, Street Signs, Tumors.
- Image Recognition.
- Video Analysis.
- NLP.
- Anomaly Detection.

- Drug Discovery.
- Checkers Game.
- Time Series Forecasting.

  Deep learning applications

  Self-Driving Cars

- In self-driven cars, it is able to capture the images around it by processing a huge amount of data, and then it will decide which

actions should be incorporated to take a left or right or should it stop.

- So, accordingly, it will decide what actions it should take, which will further reduce the accidents that happen every year.

Voice Controlled Assistance

- When we talk about voice control assistance, then Siri is the one thing that comes into our mind.
- So, you can tell Siri whatever you want it to do it for you, and it

will search it for you and display it for you.

Automatic Image Caption Generation

- Whatever image that you upload, the algorithm will work in such a way that it will generate caption accordingly.
- If you say blue colored eye, it will display a blue-colored eye with a caption at the bottom of the image.

Automatic Machine Translation

- With the help of automatic machine translation, we are able to convert one language into another with the help of deep learning.

Limitations

- It only learns through the observations.
- It comprises of biases issues.

Advantages

- It lessens the need for feature engineering.

- It eradicates all those costs that are needless.
- It easily identifies difficult defects.
- It results in the best-in-class performance on problems.

Disadvantages

- It requires an ample amount of data.
- It is quite expensive to train.
- It does not have strong theoretical groundwork.

Naive Bayes

- Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.
- It is mainly used in text classification that includes a high-dimensional training dataset.
- Naïve Bayes Classifier is one of the simple and most effective Classification algorithms which helps in building the fast machine

learning models that can make quick predictions.

- It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.
- Some popular examples of Naïve Bayes Algorithm are spam filtration, Sentimental analysis, and classifying articles.

Why is it called Naïve Bayes?

- **Naïve**: It is called Naïve because it assumes that the occurrence of a certain feature is independent of the occurrence of other features.

- Such as if the fruit is identified on the bases of color, shape, and taste, then red, spherical, and sweet fruit is recognized as an apple.
- Hence each feature individually contributes to identify that it is an apple without depending on each other.
- **Bayes**: It is called Bayes because it depends on the principle of Bayes' Theorem.

Bayes' Theorem:

- Bayes' theorem is also known as Bayes' Rule or Bayes' law, which

is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.

- The formula for Bayes' theorem is given as:

$$P(A|B)= \frac{P(B|A)P(A)}{P(B)}$$

- Where,

- P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.
- P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.
- P(A) is Prior Probability: Probability of hypothesis before observing the evidence.
- P(B) is Marginal Probability: Probability of Evidence.

Working of Naïve Bayes' Classifier:

- Working of Naïve Bayes' Classifier can be understood with the help of the below example:
- Suppose we have a dataset of weather conditions and corresponding target variable "Play".
- So using this dataset we need to decide that whether we should play or not on a particular day according to the weather conditions.
- So to solve this problem, we need to follow the below steps:

- Convert the given dataset into frequency tables.
- Generate Likelihood table by finding the probabilities of given features.
- Now, use Bayes theorem to calculate the posterior probability.

Advantages of Naïve Bayes Classifier:

- Naïve Bayes is one of the fast and easy ML algorithms to predict a class of datasets.

- It can be used for Binary as well as Multi-class Classifications.
- It performs well in Multi-class predictions as compared to the other Algorithms.
- It is the most popular choice for text classification problems.

Disadvantages of Naïve Bayes Classifier:

- Naive Bayes assumes that all features are independent or unrelated, so it cannot learn the relationship between features.

- Applications of Naïve Bayes Classifier:
- It is used for Credit Scoring.
- It is used in medical data classification.
- It can be used in real-time predictions because Naïve Bayes Classifier is an eager learner.
- It is used in Text classification such as Spam filtering and Sentiment analysis.

Types of Naïve Bayes Model:

- There are three types of Naive Bayes Model, which are given below:
- Gaussian: The Gaussian model assumes that features follow a normal distribution.
- This means if predictors take continuous values instead of discrete, then the model assumes that these values are sampled from the Gaussian distribution.

- Multinomial: The Multinomial Naïve Bayes classifier is used when the data is multinomial distributed. It is primarily used for document classification problems, it means a particular document belongs to which category such as Sports, Politics, education, etc.
- The classifier uses the frequency of words for the predictors.
- Bernoulli: The Bernoulli classifier works similar to the Multinomial classifier, but the predictor variables are the independent Booleans variables.

- Such as if a particular word is present or not in a document. This model is also famous for document classification tasks.

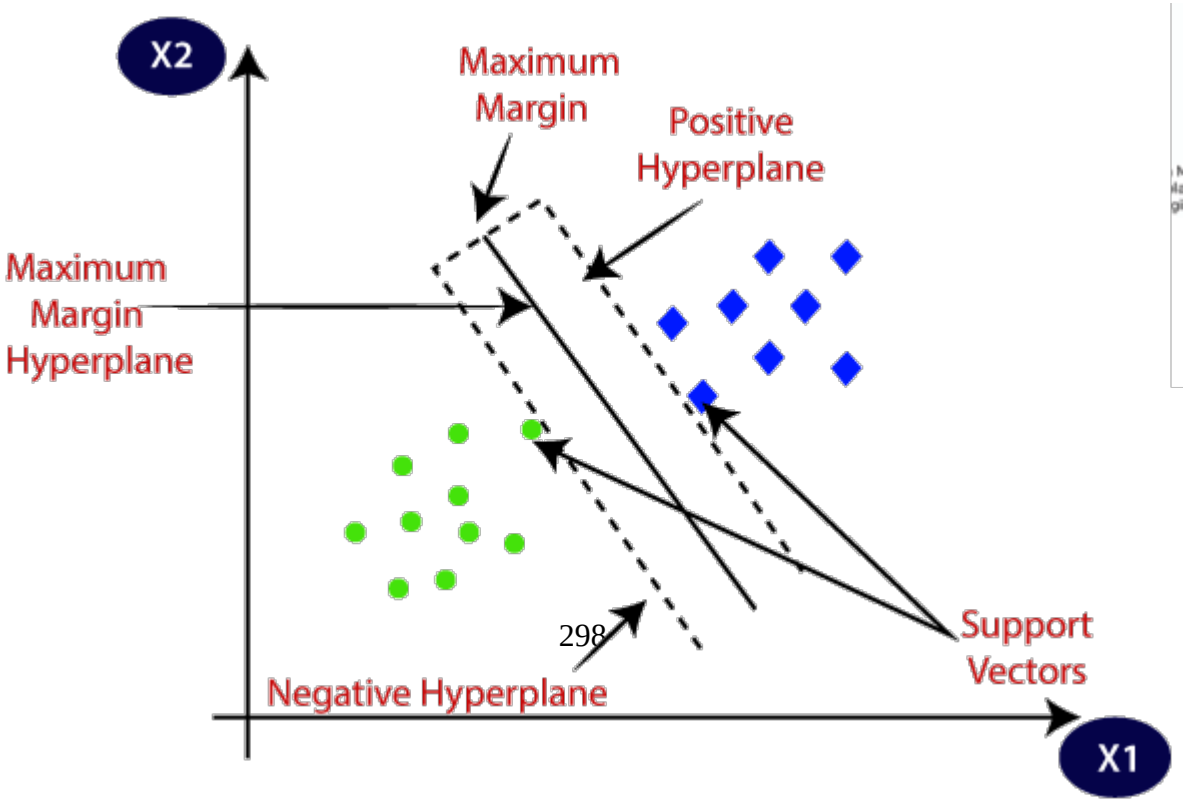
Support Vector Machines

- 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.

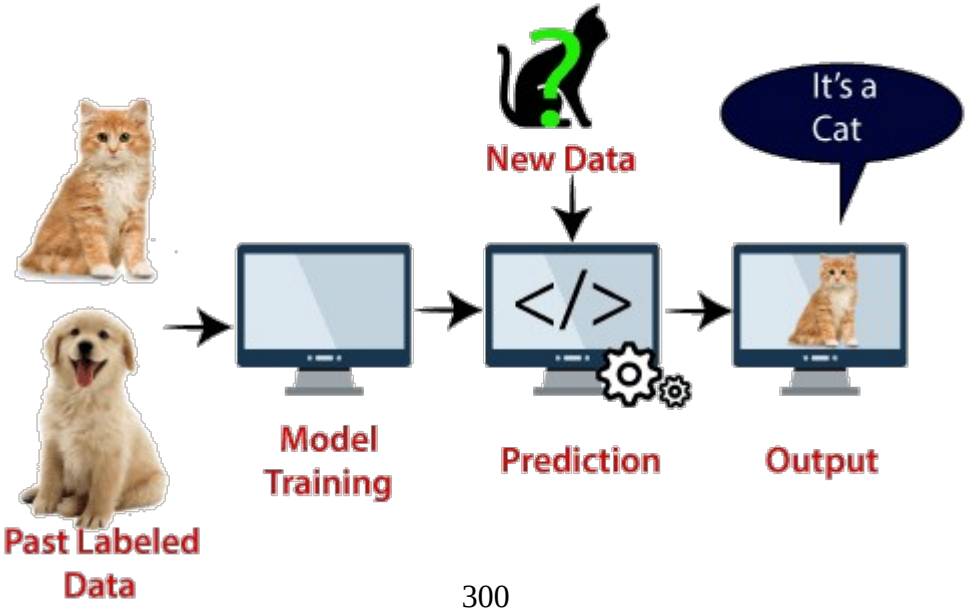
- VM chooses the extreme points/vectors that help in creating the hyperplane.
- These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine
- Consider the below diagram in which there are two different categories that are classified using a decision boundary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

hyperplane:


X2
Maximum
Margin
Positive
Hyperplane
Maximum
Margin
Hyperplane
Support
Vectors
Negative Hyperplane
X1

- uppose 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



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

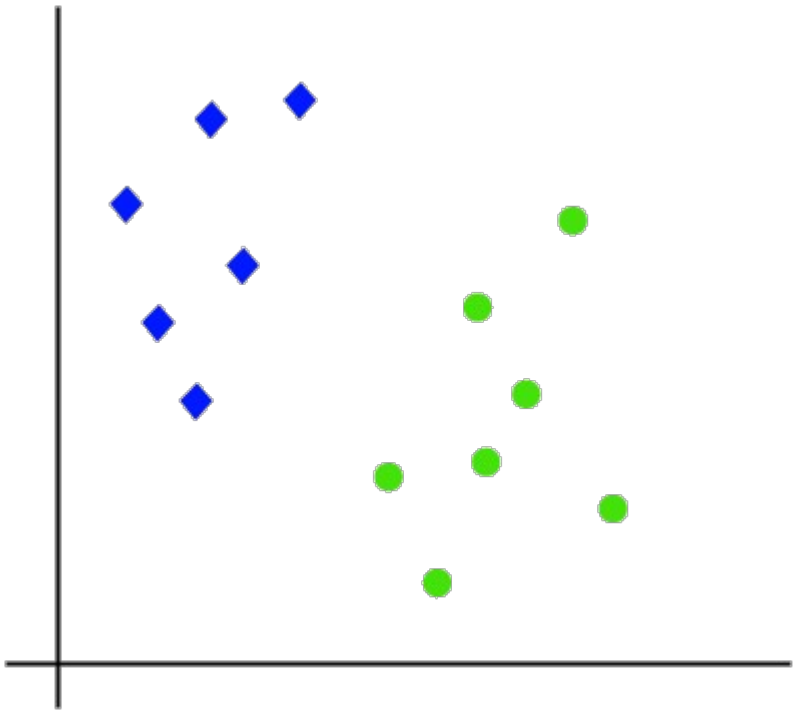
separated data

- 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.

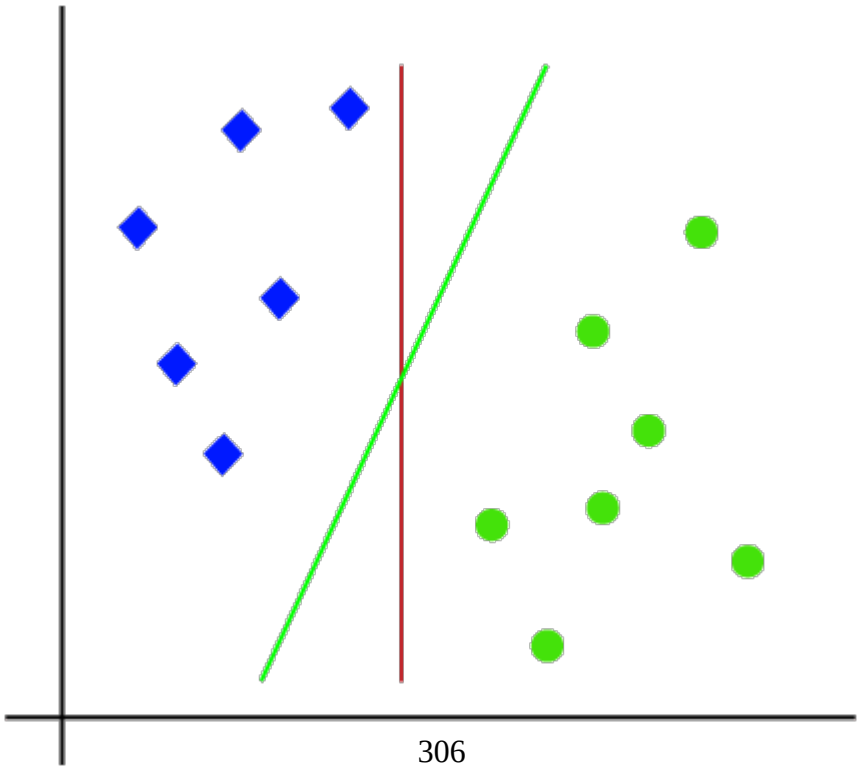
How does SVM works?

- 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.

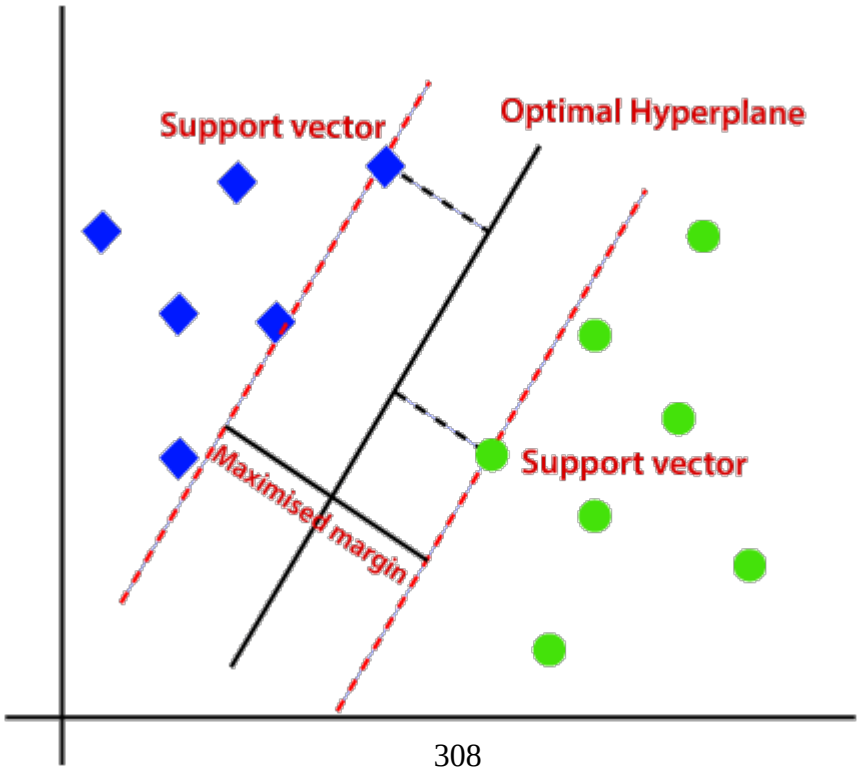
- We want a classifier that can classify the pair(x1, x2) of coordinates in either green or blue. Consider the below image:



- 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.


Support vector
Optimal Hyperplane
Maximised margin
Support vector

Model Selection

What is Model Selection?

- Model selection is the process of selecting the best model from all the available models for a particular business problem on the basis of different criterions such as robustness and model complexity.

Model Selection vs Variable Selection

- Variable selection is the process of selecting the best subset of predictors for a given problem and predictive model

- while model selection is done to select one specific model from the list of available predictive models for a given business problem.
- The set of best variables might vary according to the change in the predictive model used as different types of predictive modeling algorithms works differently.
- A specific set of features might yield very different results with different predictive models.

What Should a Model be Selected By?

- It is improbable to predict the best model for a given problem without experimenting with different models.
- It is possible to predict the best type of model that can be used for a given problem.
- For example, if you’re modeling a natural language processing problem it is highly likely that deep learning based predictive models will perform a lot better than statistical based models.

- The idea here is to select a model that suits our purpose and different criterions such as performance, robustness, complexity, etc. rather than searching for the best model.

What are Techniques for Model Selection?

- Model selection techniques can be widely classified as probabilistic measures and resampling methods.
- Probabilistic measures involve statistically scoring candidate models using performance on training dataset.

- **Random train/test split:** This is a resampling method. In this method the model is evaluated on the skill of generalization and predictive efficiency in an unseen set of data.
- The data points here are sampled without replacement. This involves splitting the data into train set and test set.
- The model that performs best on this test set is selected as the best model.
- **Cross validation:** It is a very popular resampling method for

model selection.

- In this method candidate models are trained and evaluated on multiple resampled train and test sets that are exclusive of each other.
- The data points here are sampled without replacement.
- The model performance across these different iterations are averaged to estimate the model performance.
- For example, K-Fold cross validation, Leave one out cross

validation

- **Bootstrap**: This is also a resampling method, and can be performed like random train/test split or cross validation.
- The only difference here is that the data points here are sampled with replacement.
- **AIC (Akaike Information Criterion)**:It is a probabilistic measure to estimate model performance on unseen data.
- It is not an absolute score and can be only used in comparison

with AIC scores of models from the same dataset.

- The model with the lowest AIC score is chosen as the best model.

Clustering

- Clustering or cluster analysis is a machine learning technique, which groups the unlabelled dataset.
- It can be defined as "A way of grouping the data points into different clusters, consisting of similar data points.
- The objects with the possible similarities remain in a group that has less or no similarities with another group."
- Let's understand the clustering technique with the real-world

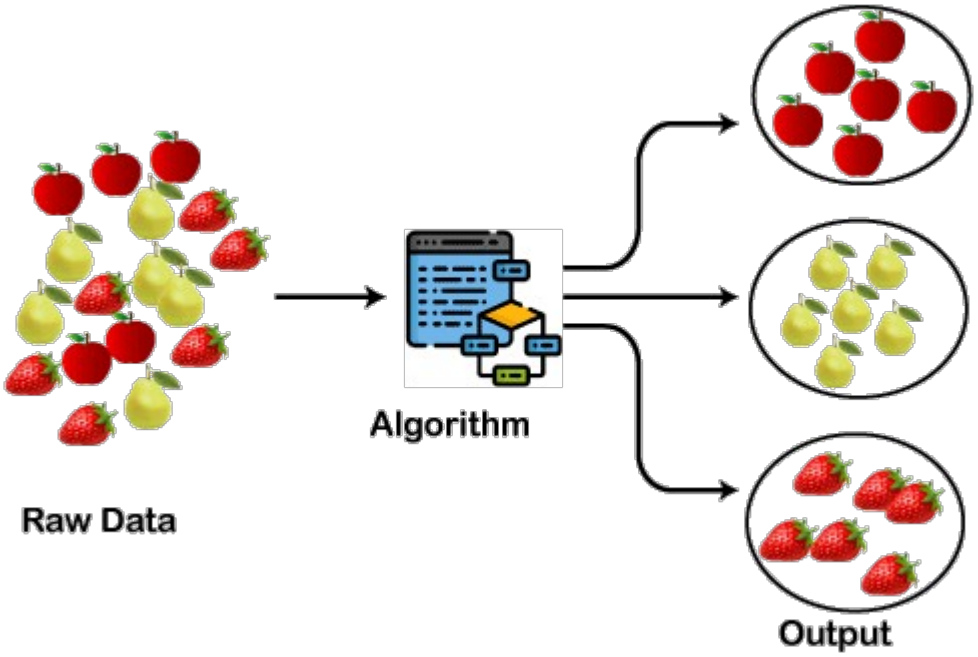
example of Mall:

- When we visit any shopping mall, we can observe that the things with similar usage are grouped together.
- Such as the t-shirts are grouped in one section, and trousers are at other sections, similarly, at vegetable sections, apples, bananas, Mangoes, etc., are grouped in separate sections, so that we can easily find out the things.
- The clustering technique also works in the same way.

- Other examples of clustering are grouping documents according to the topic.
- The clustering technique can be widely used in various tasks. Some most common uses of this technique are:
- Market Segmentation
- Statistical data analysis
- Social network analysis

- Image segmentation
- Anomaly detection, etc.
- Apart from these general usages, it is used by the Amazon in its recommendation system to provide the recommendations as per the past search of products.
- Netflix also uses this technique to recommend the movies and web-series to its users as per the watch history.
- The below diagram explains the working of the clustering

algorithm. We can see the different fruits are divided into several groups with similar properties.


Raw Data
Algorithm
Output

Applications of Clustering

- Below are some commonly known applications of clustering technique in Machine Learning:
- **In Identification of Cancer Cells**: The clustering algorithms are widely used for the identification of cancerous cells.
- It divides the cancerous and non-cancerous data sets into different groups.
- **In Search Engines**: Search engines also work on the clustering

technique.

- The search result appears based on the closest object to the search query.
- It does it by grouping similar data objects in one group that is far from the other dissimilar objects.
- The accurate result of a query depends on the quality of the clustering algorithm used.
- **Customer Segmentation:** It is used in market research to segment

the customers based on their choice and preferences.

- **In Biology:** It is used in the biology stream to classify different species of plants and animals using the image recognition technique.
- **In Land Use:** The clustering technique is used in identifying the area of similar lands use in the GIS database.
- This can be very useful to find that for what purpose the particular land should be used, that means for which purpose it is more

suitable

- There are many different types of clustering algorithms in machine learning, each with its own strengths and weaknesses.
- Some of the most common types include:
- **Centroid-based clustering:** These algorithms work by finding a set of centroids, which are representative points for each cluster.
- Data points are then assigned to the cluster with the closest centroid.

- K-means is a well-known example of a centroid-based clustering algorithm.
- **Density-based clustering:** These algorithms identify clusters based on the density of data points in a given region of the data space.
- Density-based clustering algorithms are more robust to outliers than centroid-based clustering algorithms.
- DBSCAN is a popular density-based clustering algorithm.

- **Distribution-based clustering:** These algorithms assume that the data points in each cluster follow a specific distribution.
- Gaussian mixture models (GMMs) are a common type of distribution-based clustering algorithm.
- **Hierarchical clustering:** These algorithms build a hierarchy of clusters by iteratively merging or splitting clusters.
- Hierarchical clustering algorithms can be either agglomerative (bottom-up) or divisive (top-down).

- **Fuzzy clustering**: These algorithms allow data points to belong to multiple clusters with different degrees of membership.
- Fuzzy clustering algorithms can be more flexible than hard clustering algorithms, which require each data point to belong to a single cluster.

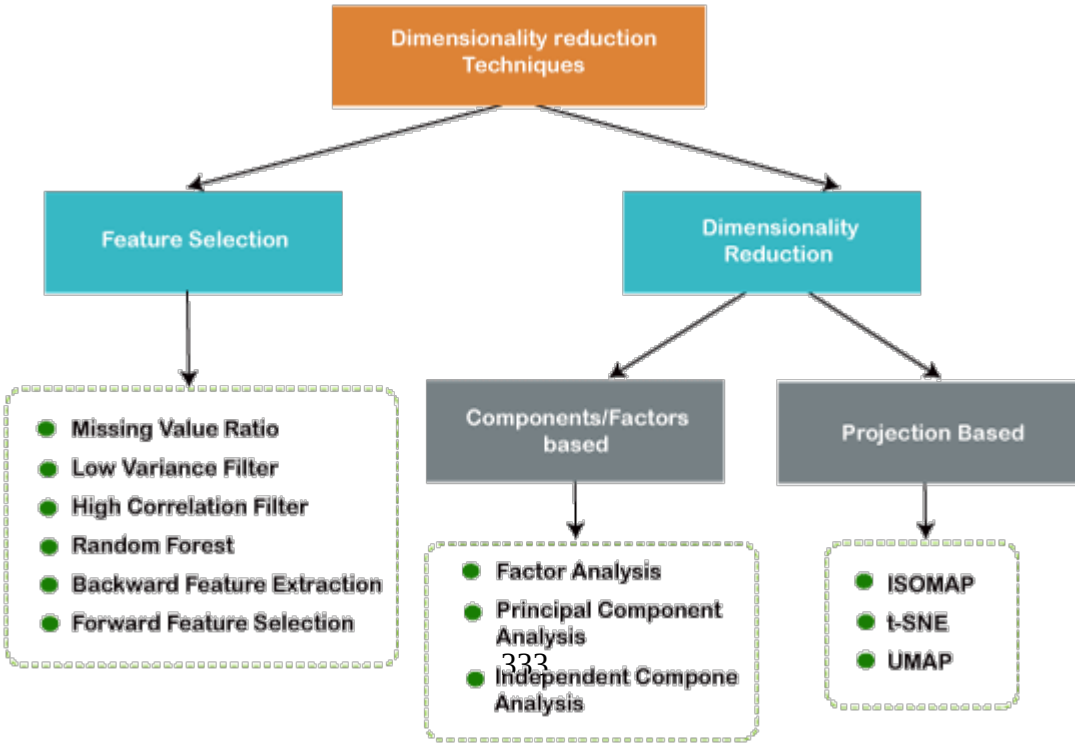
Dimensionality Reduction

What is Dimensionality Reduction?

- The number of input features, variables, or columns present in a given dataset is known as dimensionality, and the process to reduce these features is called dimensionality reduction.
- A dataset contains a huge number of input features in various cases, which makes the predictive modeling task more complicated.

- Because it is very difficult to visualize or make predictions for the training dataset with a high number of features, for such cases, dimensionality reduction techniques are required to use.
- Dimensionality reduction technique can be defined as, "It is a way of converting the higher dimensions dataset into lesser dimensions dataset ensuring that it provides similar information."
- These techniques are widely used in machine learning for obtaining a better fit predictive model while solving the

classification and regression problems.


Dimensionality reduction Techniques
Feature Selection
Dimensionality Reduction
Missing Value Ratio
Low Variance Filter
High Correlation Filter
Random Forest
Backward Feature Extraction
Forward Feature Selection
Components/Factors based
Projection Based
Factor Analysis
Principal Component Analysis
Independent Compone Analysis
ISOMAP
t-SNE
UMAP

Benefits of applying Dimensionality Reduction

- Some benefits of applying dimensionality reduction technique to the given dataset are given below:
- By reducing the dimensions of the features, the space required to store the dataset also gets reduced.
- Less Computation training time is required for reduced dimensions of features.
- Reduced dimensions of features of the dataset help in visualizing

the data quickly.

- It removes the redundant features (if present) by taking care of multicollinearity.

Disadvantages of dimensionality Reduction

- There are also some disadvantages of applying the dimensionality reduction, which are given below:
- Some data may be lost due to dimensionality reduction.

- In the PCA dimensionality reduction technique, sometimes the principal components required to consider are unknown.

Approaches of Dimension Reduction

- There are two ways to apply the dimension reduction technique, which are given below:

Feature Selection

- Feature selection is the process of selecting the subset of the relevant features and leaving out the irrelevant features present in

a dataset to build a model of high accuracy.

- n other words, it is a way of selecting the optimal features from the input dataset.

Feature Extraction:

- Feature extraction is the process of transforming the space containing many dimensions into space with fewer dimensions.
- This approach is useful when we want to keep the whole information but use fewer resources while processing the

information

Forward Feature Selection

- Forward feature selection follows the inverse process of the backward elimination process.
- It means, in this technique, we don't eliminate the feature; instead, we will find the best features that can produce the highest increase in the performance of the model.
- Below steps are performed in this technique:

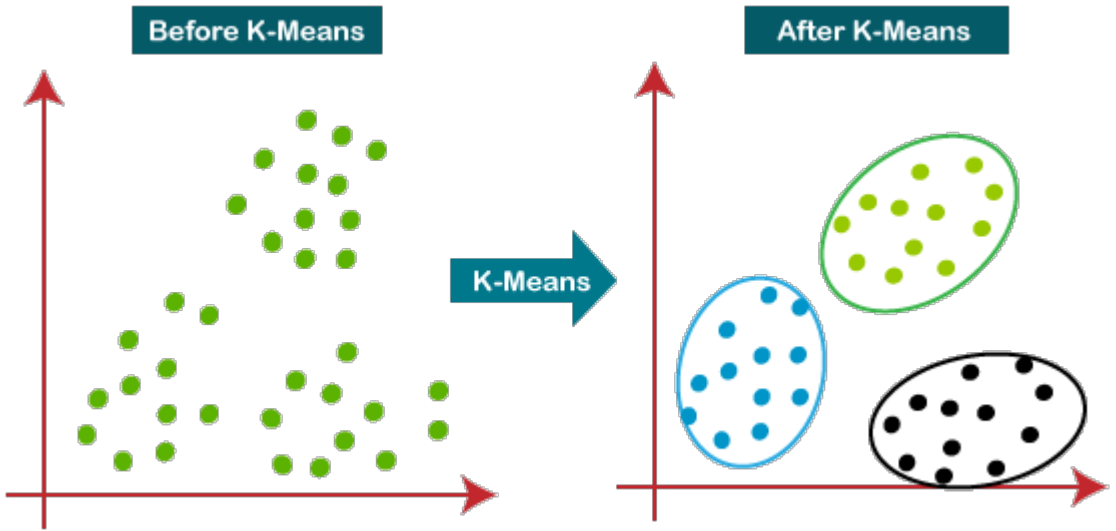
- We start with a single feature only, and progressively we will add each feature at a time.
- Here we will train the model on each feature separately.
- The feature with the best performance is selected.
- The process will be repeated until we get a significant increase in the performance of the model.

K-means

- K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science.
- -Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters.
- Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters

- and for K=3, there will be three clusters, and so on.
- It is an iterative algorithm that divides the unlabeled dataset into k different clusters
- in such a way that each dataset belongs only one group that has similar properties.
- The k-means clustering algorithm mainly performs two tasks:
- Determines the best value for K center points or centroids by an iterative process.

- Assigns each data point to its closest k-center.
- Those data points which are near to the particular k-center, create a cluster.
- Hence each cluster has datapoints with some commonalities, and it is away from other clusters.
- The below diagram explains the working of the K-means Clustering Algorithm:


Before K-Means
K-Means
After K-Means

How does the K-Means Algorithm Work?

- The working of the K-Means algorithm is explained in the below steps:

- Step-1: Select the number K to decide the number of clusters.

- Step-2: Select random K points or centroids. (It can be other from the input dataset).

- Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

- Step-4: Calculate the variance and place a new centroid of each cluster.

- Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

- Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

- Step-7: The model is ready.

Applications

- Image segmentation: K-means clustering can be used to segment images into different regions, such as the foreground and background.
- This can be useful for tasks such as object recognition and image classification.
- Customer segmentation: K-means clustering can be used to

segment customers into different groups based on their demographics, purchase history, and other factors.

- This information can then be used to target customers with relevant marketing messages.
- Text clustering: K-means clustering can be used to cluster text documents into different groups based on their content.
- This can be useful for tasks such as topic modeling and document summarization.

Advantages:

- K-means clustering is a simple and easy-to-understand algorithm.
- K-means clustering is a relatively fast algorithm, even for large datasets.
- K-means clustering is a versatile algorithm that can be used for a variety of tasks.

Disadvantages:

- K-means clustering requires the user to specify the number of clusters in advance.
- This can be difficult to do without prior knowledge of the dataset.
- K-means clustering is sensitive to the initial cluster centroids.
- If the initial cluster centroids are not chosen well, the algorithm may converge to a suboptimal solution.
- K-means clustering does not work well with data that has outliers. Outliers can cause the algorithm to converge to a suboptimal

solution.

Expectation maximization

- The EM algorithm is considered a latent variable model to find the local maximum likelihood parameters of a statistical model, proposed by Arthur Dempster, Nan Laird, and Donald Rubin in 1977.
- The EM (Expectation-Maximization) algorithm is one of the most commonly used terms in machine learning to obtain maximum likelihood estimates of variables that are sometimes observable

and sometimes not.

- However, it is also applicable to unobserved data or sometimes called latent.
- It has various real-world applications in statistics, including obtaining the mode of the posterior marginal distribution of parameters in machine learning and data mining applications.

What is an EM algorithm?

- The Expectation-Maximization (EM) algorithm is a statistical technique used for estimating the parameters of statistical models when there are missing or incomplete data.
- It is particularly useful in situations where the data is partially observed or where some variables are unobservable, making it challenging to perform maximum likelihood estimation directly.
- The EM algorithm consists of two main steps: the E-step

(Expectation step) and the M-step (Maximization step).

- These steps are iteratively performed until convergence is reached, typically when the change in estimated parameters between iterations becomes small.
- Here's a high-level overview of how the EM algorithm works:
- **Initialization**: Start with an initial guess for the model parameters.
- E-step (Expectation):

- In this step, you compute the expected value (or expectation) of the missing or latent variables given the current parameter estimates.
- This step involves calculating the probability distribution of the unobservable variables based on the current parameter values.
- M-step (Maximization):
- In this step, you update the model parameters to maximize the likelihood of the observed data, using the expected values

computed in the E-step.

- This step involves finding the parameter values that maximize the log-likelihood of the observed data, given the expected values from the E-step.

Iteration:

- Repeat the E-step and M-step until convergence is achieved. Convergence is typically determined by monitoring the change in parameter values between iterations.

- The EM algorithm is widely used in various fields, including machine learning, statistics, and data analysis.
- It is especially popular in clustering algorithms like the Gaussian Mixture Model (GMM) and in solving problems involving missing data.
- One key property of the EM algorithm is that it guarantees that the likelihood of the data increases (or at least remains constant) with each iteration, which makes it suitable for situations where the

likelihood function is not easily maximized directly due to missing or hidden variables.

Applications of EM algorithm

- The primary aim of the EM algorithm is to estimate the missing data in the latent variables through observed data in datasets.
- The EM algorithm or latent variable model has a broad range of real-life applications in machine learning. These are as follows:
- The EM algorithm is applicable in data clustering in machine

learning.

- It is often used in computer vision and NLP (Natural language processing).
- It is used to estimate the value of the parameter in mixed models such as the Gaussian Mixture Modeland quantitative genetics.
- It is also used in psychometrics for estimating item parameters and latent abilities of item response theory models.
- It is also applicable in the medical and healthcare industry, such as

in image reconstruction and structural engineering.

- It is used to determine the Gaussian density of a function.

Advantages of EM algorithm

- It is very easy to implement the first two basic steps of the EM algorithm in various machine learning problems, which are E-step and M- step.
- It is mostly guaranteed that likelihood will enhance after each iteration.

- It often generates a solution for the M-step in the closed form.

Disadvantages of EM algorithm

- The convergence of the EM algorithm is very slow.
- It can make convergence for the local optima only.
- It takes both forward and backward probability into consideration. It is opposite to that of numerical optimization, which takes only forward probabilities.

Gaussian mixture density estimation

- Gaussian Mixture Density Estimation (GMDE) is a statistical method used for modeling
- and estimating the probability density function of a dataset when the underlying distribution is assumed to be a mixture of multiple Gaussian (normal) distributions.
- It's a popular technique for density estimation and clustering in machine learning and statistics.

Here's how Gaussian Mixture Density Estimation works:

1. **Model Representation:**

- - Assume that the observed data points are generated from a mixture of K Gaussian distributions.
- - Each Gaussian component is defined by its mean (μ), variance (σ^2), and weight (π), where π represents the probability of a data point belonging to that component.
- - The probability density function (PDF) of a Gaussian

2. **Initialization:**

- - Initialize the parameters of the Gaussian mixture model. This typically involves randomly initializing the means, variances, and weights for each component.

3. Expectation-Maximization (EM):

- - Use the EM algorithm to iteratively update the model parameters to maximize the likelihood of the observed data.
- - E-step (Expectation step): Compute the posterior probabilities

(responsibilities) of each data point belonging to each Gaussian component. This is done using Bayes' theorem.

- - M-step (Maximization step): Update the parameters (means, variances, and weights) of each Gaussian component based on the computed responsibilities.

4. **Convergence:**

- - Repeat the E-step and M-step until convergence, typically determined by a small change in the parameters or the log-

likelihood of the data.

5. **Model Evaluation:**

- - Once the EM algorithm converges, you have the parameters of the Gaussian mixture model, which can be used for various purposes:
- - **Density estimation**: You can use the model to estimate the probability density of data points at different values.
- - **Clustering**: You can assign data points to clusters based on

their most likely Gaussian component.

- - **Anomaly detection**: Identify data points that have low probabilities under the model.
- Gaussian Mixture Density Estimation is a flexible and powerful technique that can capture complex data distributions by combining multiple Gaussian components.
- It is often used in applications such as image segmentation, speech recognition, and anomaly detection, where the underlying data

distribution is not known in advance and is likely to be a mixture of different modes or clusters.

Mixture of naive Bayes

- A Mixture of Naive Bayes (MNB) is a probabilistic model that combines multiple Naive Bayes classifiers into a single, more powerful model.
- It is a variant of the traditional Naive Bayes classifier that assumes that the data is generated from a mixture of multiple subpopulations
- each following its own Naive Bayes distribution.

- MNB models are particularly useful when the data cannot be accurately modeled by a single Naive Bayes distribution but can be better represented as a combination of several simpler models.
- Here's how Mixture of Naive Bayes works:

  1. **Data Assumption:**

- - MNB assumes that the dataset consists of several subpopulations or clusters, each with its own underlying distribution.

- - It assumes that within each cluster, the features are conditionally independent given the class label, just like in the traditional Naive Bayes model.

2. **Mixture Model:**

- - MNB introduces a hidden or latent variable that indicates the cluster to which each data point belongs.
- - The latent variable follows a discrete distribution (usually a categorical distribution), representing the probabilities of data

points belonging to each cluster.

- - For each cluster, there is a separate Naive Bayes classifier with its own set of class priors and conditional probability distributions for each feature given the class.

3. **Parameter Estimation:**

- - The parameters of the MNB model need to be estimated.
- This includes the class priors for each cluster and the conditional probabilities for each feature within each cluster.

4. **Inference:**

- - Given a new data point, the MNB model estimates the posterior probabilities of the data point belonging to each cluster.
- - These posterior probabilities are computed using the Bayes' theorem and the parameters estimated in step 3.

5. *Classification:*

- - To classify a data point, MNB typically assigns it to the cluster with the highest estimated posterior probability.

- - Once the cluster is determined, the corresponding Naive Bayes classifier within that cluster is used to classify the data point.
- Mixture of Naive Bayes models are useful in situations where the traditional Naive Bayes assumption of feature independence within a class doesn't hold or is too simplistic.
- By allowing for multiple subpopulations or clusters, MNB can capture more complex relationships between features and class labels.

- MNB models are particularly effective in text classification tasks, where the independence assumption of Naive Bayes is often violated, but the simplicity and interpretability of Naive Bayes are desired.
- They are also used in various other applications such as image classification and recommendation systems.

Hidden Markov Model (HMM)

- Hidden Markov Models (HMMs) are a type of probabilistic model that are commonly used in machine learning for tasks such as speech recognition, natural language processing, and bioinformatics.
- They are a popular choice for modelling sequences of data because they can effectively capture the underlying structure of the data, even when the data is noisy or incomplete.

Application

- Hidden Markov Models (HMMs) have a wide range of applications in various fields due to their ability to model sequential data and capture underlying hidden states.
- Some of the key applications of HMMs include:
- **Speech Recognition**: HMMs are widely used in speech recognition systems to model the probabilistic relationships between phonemes, words, or acoustic features.

- They help convert spoken language into text.
- Natural Language Processing (NLP):
- **Part-of-Speech Tagging**: HMMs are used to assign parts of speech (e.g., noun, verb, adjective) to words in text.
- **Named Entity Recognition**: HMMs can be applied to identify and classify entities (e.g., names of people, organizations) in text.
- **Bioinformatics**:

- **Gene Prediction**: HMMs are used to predict gene structures in DNA sequences.
- **Protein Structure Prediction**: They are used to predict protein secondary and tertiary structures from amino acid sequences.
- **Financial Modeling:**
- **Stock Market Analysis**: HMMs can be applied to model and predict stock price movements.
- **Credit Risk Assessment**: They are used to assess the

creditworthiness of individuals or companies based on historical data.

- **Machine Translation:** HMMs can be used in machine translation systems to align words or phrases between source and target languages and to generate translations.
- **Gesture Recognition**: In computer vision, HMMs can be applied to recognize and interpret human gestures in videos or image sequences.

- **Handwriting Recognition**: HMMs can be used to recognize handwritten characters or words, making them valuable in optical character recognition (OCR) systems.

Limitations

- **Independence Assumption**: HMMs assume that the observations (emissions) at each time step are conditionally independent given the hidden state. This is known as the "Markov property."
- In reality, many real-world processes may exhibit dependencies

that violate this assumption.

- **Fixed State Space:** HMMs assume a fixed, finite set of hidden states. If the number of states is not chosen appropriately, the model may either overfit (with too many states) or underfit (with too few states), leading to suboptimal performance.
- **Stationary Process**: HMMs assume that the transition probabilities between hidden states do not change over time (stationarity).

- In dynamic processes with changing characteristics, this assumption may not hold.
- **Limited Modeling of Long Dependencies:** HMMs are typically better suited for modeling short-term dependencies due to their first-order Markov property.
- Modeling long-range dependencies may require a large number of states or higher-order HMMs, which can be computationally expensive.

- **No Memory of Past States**: HMMs only consider the current hidden state when making predictions or generating observations. They do not have memory of past states, which can limit their ability to capture certain patterns.

Algorithm

- he hidden Markov algorithm (HMM algorithm) is a set of algorithms used to train and decode hidden Markov models (HMMs).

- HMMs are probabilistic models that can be used to model sequences of observations that are generated by an underlying sequence of hidden states.
- The HMM algorithm consists of two main steps:
- **Training**: The training step uses a set of observed sequences to estimate the parameters of the HMM.
- This includes the transition probabilities between hidden states and the emission probabilities of observations given hidden states.

- **Decoding**: The decoding step uses the trained HMM to predict the most likely sequence of hidden states that generated a given sequence of observations.
- There are two main types of HMM algorithms: the forward-backward algorithm and the Viterbi algorithm.
- The forward-backward algorithm is used to compute the probability of a sequence of observations given a HMM.
- This probability can be used to evaluate the performance of the

HMM or to calculate the likelihood of a new sequence of observations.

- The Viterbi algorithm is used to decode a sequence of observations and find the most likely sequence of hidden states that generated the observations.
- This algorithm is often used in speech recognition, natural language processing
- and other applications where it is necessary to infer the hidden

state of a system from a sequence of observations.

- Here is a simplified overview of the Viterbi algorithm:
- Initialize a vector V to store the probability of each hidden state at the start of the sequence.
- For each observation in the sequence:
- For each hidden state:
- Compute the probability of transitioning from the previous hidden

state to the current hidden state and emitting the current observation.

- Update the V vector to store the probability of the current hidden state given the sequence of observations up to the current point.
- The most likely sequence of hidden states is the sequence that corresponds to the highest probability in the V vector at the end of the sequence.
- HMM algorithms are a powerful tool for modeling and decoding

sequences of observations where the underlying hidden state is unknown.

Reinforcement Learning

- Reinforcement Learning is a feedback-based Machine learning technique in which an agent learns to behave in an environment by performing the actions and seeing the results of actions.
- For each good action, the agent gets positive feedback, and for each bad action, the agent gets negative feedback or penalty.
- In Reinforcement Learning, the agent learns automatically using feedbacks without any labeled data, unlike supervised learning.

- Since there is no labeled data, so the agent is bound to learn by its experience only.
- RL solves a specific type of problem where decision making is sequential, and the goal is long-term, such as game-playing, robotics, etc.
- The agent interacts with the environment and explores it by itself. The primary goal of an agent in reinforcement learning is to improve the performance by getting the maximum positive

rewards.

- The agent learns with the process of hit and trial, and based on the experience, it learns to perform the task in a better way.
- Hence, we can say that "Reinforcement learning is a type of machine learning method where an intelligent agent (computer program) interacts with the environment and learns to act within that."
- How a Robotic dog learns the movement of his arms is an example

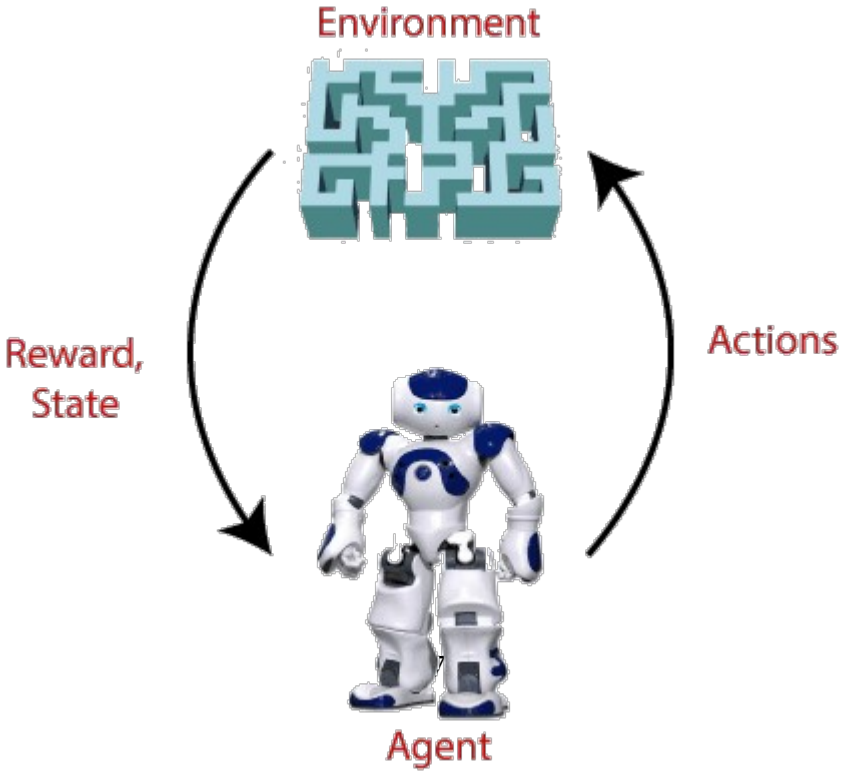
of Reinforcement learning.

- It is a core part of Artificial intelligence, and all AI agent works on the concept of reinforcement learning.
- Here we do not need to pre-program the agent, as it learns from its own experience without any human intervention.
- Example: Suppose there is an AI agent present within a maze environment, and his goal is to find the diamond.
- The agent interacts with the environment by performing some

actions, and based on those actions, the state of the agent gets changed, and it also receives a reward or penalty as feedback.

- The agent continues doing these three things (take action, change state/remain in the same state, and get feedback), and by doing these actions, he learns and explores the environment.
- The agent learns that what actions lead to positive feedback or rewards and what actions lead to negative feedback penalty.
- As a positive reward, the agent gets a positive point, and as a

penalty, it gets a negative point.



Terms used in Reinforcement Learning

- Agent(): An entity that can perceive/explore the environment and act upon it.
- Environment(): A situation in which an agent is present or surrounded by. In RL, we assume the stochastic environment, which means it is random in nature.
- Action(): Actions are the moves taken by an agent within the environment.

- State(): State is a situation returned by the environment after each action taken by the agent.
- Reward(): A feedback returned to the agent from the environment to evaluate the action of the agent.
- Policy(): Policy is a strategy applied by the agent for the next action based on the current state.
- Value(): It is expected long-term retuned with the discount factor and opposite to the short-term reward.

- Q-value(): It is mostly similar to the value, but it takes one additional parameter as a current action (a).

Key Features of Reinforcement Learning

- In RL, the agent is not instructed about the environment and what actions need to be taken.
- It is based on the hit and trial process.
- The agent takes the next action and changes states according to the feedback of the previous action.

- The agent may get a delayed reward.
- The environment is stochastic, and the agent needs to explore it to reach to get the maximum positive rewards.
- Approaches to implement Reinforcement Learning
- There are mainly three ways to implement reinforcement-learning in ML, which are:
- **Value-based:**

- The value-based approach is about to find the optimal value function, which is the maximum value at a state under any policy.
- Therefore, the agent expects the long-term return at any state(s) under policy π.
- **Policy-based:**
- Policy-based approach is to find the optimal policy for the maximum future rewards without using the value function.
- In this approach, the agent tries to apply such a policy that the

action performed in each step helps to maximize the future reward.

- The policy-based approach has mainly two types of policy:
- **Deterministic**: The same action is produced by the policy (π) at any state.
- **Stochastic**: In this policy, probability determines the produced action.
- **Model-based:** In the model-based approach, a virtual model is created for the environment, and the agent explores that

environment to learn it.

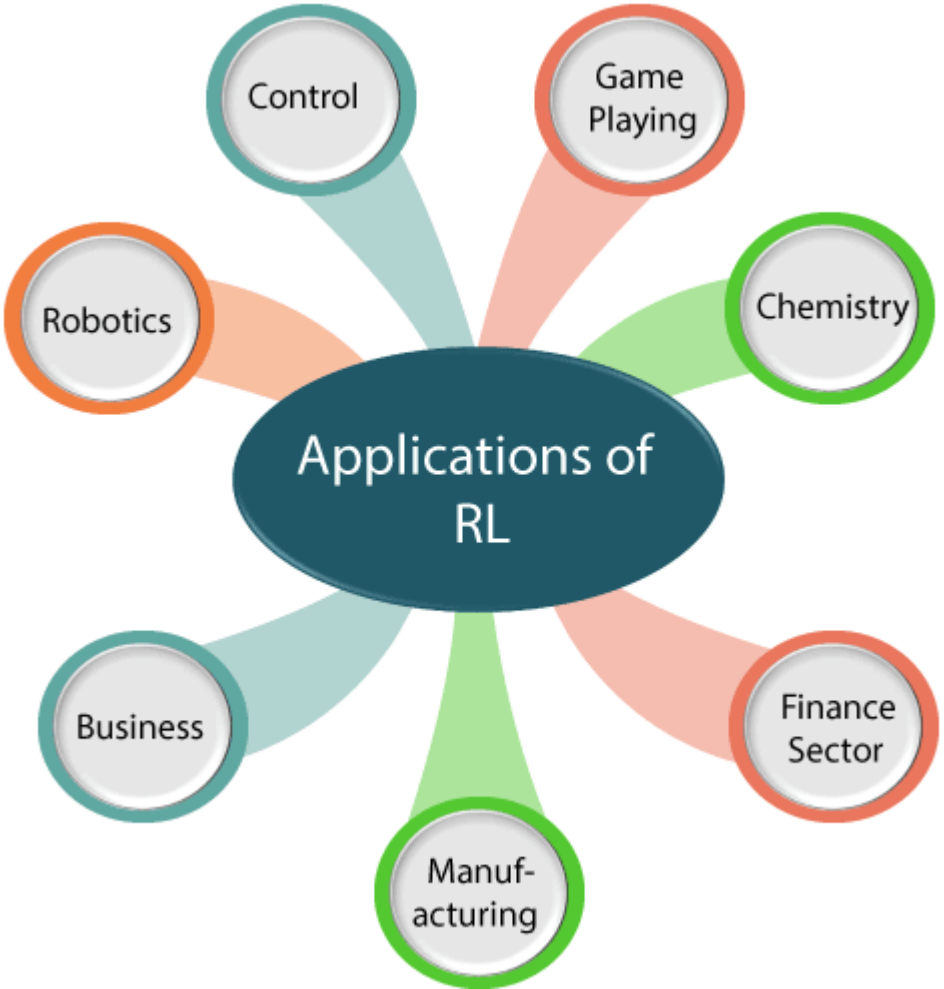
- There is no particular solution or algorithm for this approach because the model representation is different for each environment.

The difference table between RL and Supervised learning is given below:

Reinforcement Learning	Supervised Learning
RL works by interacting with the environment.	Supervised learning works on the existing dataset.
The RL algorithm works like the human brain works when making some decisions.	Supervised Learning works as when a human learns things in the supervision of a guide.
There is no labeled dataset is present	The labeled dataset is present.

No previous training is provided to the learning agent.	Training is provided to the algorithm so that it can predict the output.
RL helps to take decisions sequentially.	In Supervised learning, decisions are made when input is given.

Reinforcement Learning Applications


Applications of RL
Control
Game Playing
Chemistry
Robotics
Business
Finance Sector
Manuf-acturing

Robotics:

- RL is used in Robot navigation, Robo-soccer, walking, juggling, etc.

Control:

- RL can be used for adaptive control such as Factory processes, admission control in telecommunication, and Helicopter pilot is an example of reinforcement learning.

Game Playing:

- RL can be used in Game playing such as tic-tac-toe, chess, etc.

Chemistry:

- RL can be used for optimizing the chemical reactions.

Business:

- RL is now used for business strategy planning.

Manufacturing:

- In various automobile manufacturing companies, the robots use

deep reinforcement learning to pick goods and put them in some containers.

Finance Sector:

- The RL is currently used in the finance sector for evaluating trading strategies.