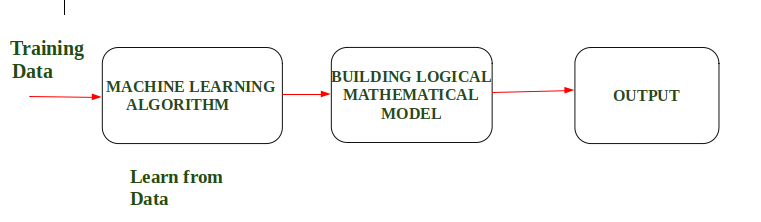
**Q1.How will u design a learning system? Explain with example?**

 When we fed the Training Data to Machine Learning Algorithm, this algorithm will produce a mathematical model and with the help of the mathematical model, the machine will make a prediction and take a decision without being explicitly programmed. Also, during training data, the more machine will work with it the more it will get experience and the more efficient result is produced.



**Example :**In Driverless Car, the training data is fed to Algorithm like how to Drive Car in Highway, Busy and Narrow Street with factors like speed limit, parking, stop at signal etc. After that, a Logical and Mathematical model is created on the basis of that and after that, the car will work according to the logical model. Also, the more data the data is fed the more efficient output is produced.

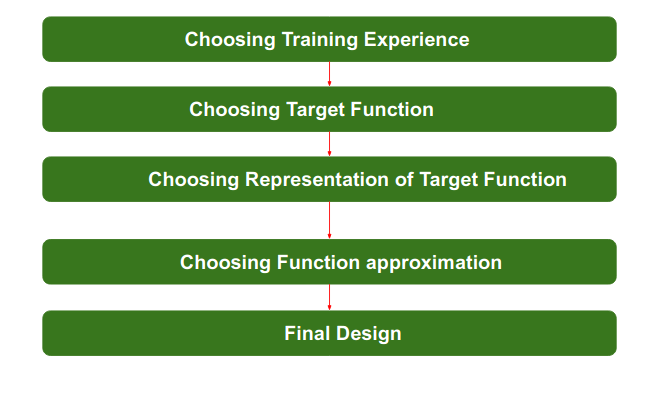
**Designing a Learning System in Machine Learning :**

According to Tom Mitchell, “A computer program is said to be learning from experience (E), with respect to some task (T). Thus, the performance measure (P) is the performance at task T, which is measured by P, and it improves with experience E.”

**Example:**In Spam E-Mail detection,

* **Task, T:** To classify mails into Spam or Not Spam.
* **Performance measure, P:** Total percent of mails being correctly classified as being “Spam” or “Not Spam”.
* **Experience, E:** Set of Mails with label “Spam”

**Steps for Designing Learning System are:**



**Step 1) Choosing the Training Experience:**The very important and first task is to choose the training data or training experience which will be fed to the Machine Learning Algorithm. It is important to note that the data or experience that we fed to the algorithm must have a significant impact on the Success or Failure of the Model. So Training data or experience should be chosen wisely.

Below are the attributes which will impact on Success and Failure of Data:

* The training experience will be able to provide direct or indirect feedback regarding choices. For example: While Playing chess the training data will provide feedback to itself like instead of this move if this is chosen the chances of success increases.
* Second important attribute is the degree to which the learner will control the sequences of training examples. For example: when training data is fed to the machine then at that time accuracy is very less but when it gains experience while playing again and again with itself or opponent the machine algorithm will get feedback and control the chess game accordingly.
* Third important attribute is how it will represent the distribution of examples over which performance will be measured. For example, a Machine learning algorithm will get experience while going through a number of different cases and different examples. Thus, Machine Learning Algorithm will get more and more experience by passing through more and more examples and hence its performance will increase.

**Step 2- Choosing target function:**The next important step is choosing the target function. It means according to the knowledge fed to the algorithm the machine learning will choose NextMove function which will describe what type of legal moves should be taken.  For example : While playing chess with the opponent, when opponent will play then the machine learning algorithm will decide what be the number of possible legal moves taken in order to get success.

**Step 3- Choosing Representation for Target function:**When the machine algorithm will know all the possible legal moves the next step is to choose the optimized move using any representation i.e. using linear Equations, Hierarchical Graph Representation, Tabular form etc. The NextMove function will move the Target move like out of these move which will provide more success rate. For Example : while playing chess machine have 4 possible moves, so the machine will choose that optimized move which will provide success to it.

**Step 4- Choosing Function Approximation Algorithm:**An optimized move cannot be chosen just with the training data. The training data had to go through with set of example and through these examples the training data will approximates which steps are chosen and after that machine will provide feedback on it. For Example : When a training data of Playing chess is fed  to algorithm so at that time it is not machine algorithm will fail or get success and again from that failure or success it will measure while next move what step should be chosen and what is its success rate.

**Step 5- Final Design:**The final design is created at last when system goes from number of examples  , failures and success , correct and incorrect decision and what will be the next step etc. Example: DeepBlue is an intelligent  computer which is ML-based won chess game against the chess expert Garry Kasparov, and it became the first computer which had beaten a human chess expert.

**Q2. List and explain prespective and issues in machine learning?**

There are several perspectives and issues in machine learning that researchers and practitioners need to consider to ensure the development of robust and ethical systems. Here are some of the main perspectives and issues:

1. Technical perspective: This perspective focuses on the technical aspects of machine learning, such as algorithms, models, and data preprocessing. One of the main issues in this perspective is the bias in machine learning models, which can lead to unfair outcomes and discrimination. Another issue is the lack of interpretability of some models, which makes it difficult to understand how they make decisions.

2. Human-centered perspective: This perspective focuses on the impact of machine learning on people and society. One of the main issues in this perspective is the potential loss of jobs due to automation, which can lead to economic and social inequality. Another issue is the privacy and security of personal data, which can be compromised by machine learning systems.

3. Ethical perspective: This perspective focuses on the ethical considerations of machine learning, such as fairness, accountability, and transparency. One of the main issues in this perspective is the responsibility of developers to ensure that their systems do not perpetuate discrimination or harm to individuals or groups. Another issue is the need for transparency in machine learning systems, particularly in decision-making processes.

4. Legal perspective: This perspective focuses on the legal implications of machine learning, such as intellectual property, liability, and regulation. One of the main issues in this perspective is the ownership of data and models, particularly in collaborative environments. Another issue is the liability of developers and users for the outcomes of machine learning systems.

In conclusion, machine learning presents several perspectives and issues that need to be considered to ensure the development of robust, ethical, and fair systems. By addressing these perspectives and issues, researchers and practitioners can contribute to the development of a responsible and beneficial use of machine learning in various domains.

**Q3. Differentiate between batch and online learning?**

|  | Online machine learning | Offline machine learning |
| --- | --- | --- |
| Complexity | More complex because the model keeps evolving over time as more data becomes available. | Less complex because the model is fed with more consistent data sets periodically. |
| Computational power | More computational power is required because of the continuous feed of data that leads to continuous refinement. | Fewer computational power is needed because data is delivered in batches; the model isn’t continuously refining itself. |
| Use in production | Harder to implement and control because the production model changes in real-time according to its data feed. | Easier to implement because offline learning provides engineers with more time to perfect the model before deployment. |
| Applications | Used in applications where new data patterns are constantly required (e.g., weather prediction tools) | Used in applications where data patterns remain constant and don’t have sudden concept drifts (e.g., image classification) |

**Q4. Explain the relation between AI, AL ,Dl ,Ds ?**

AI (Artificial Intelligence) is a broad field that encompasses various subfields, including AL (Artificial Learning), DL (Deep Learning), and DS (Data Science). These subfields are closely related and are used in various applications of AI.

Data Science (DS) is concerned with the collection, cleaning, and analysis of data to extract insights and inform decision-making. DS involves the use of statistical and computational techniques to analyze and model data. DS provides the foundational knowledge and skills needed for AI applications.

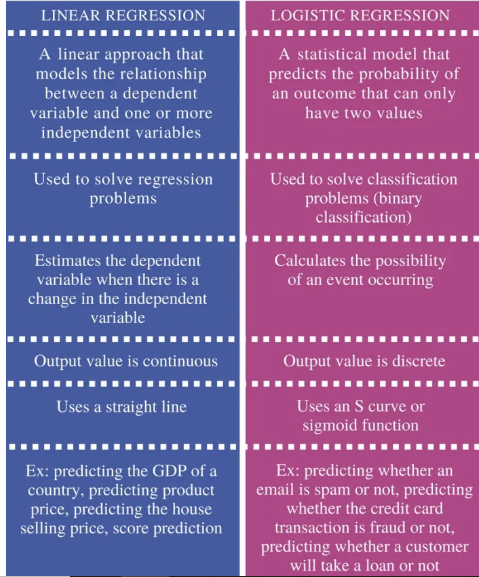
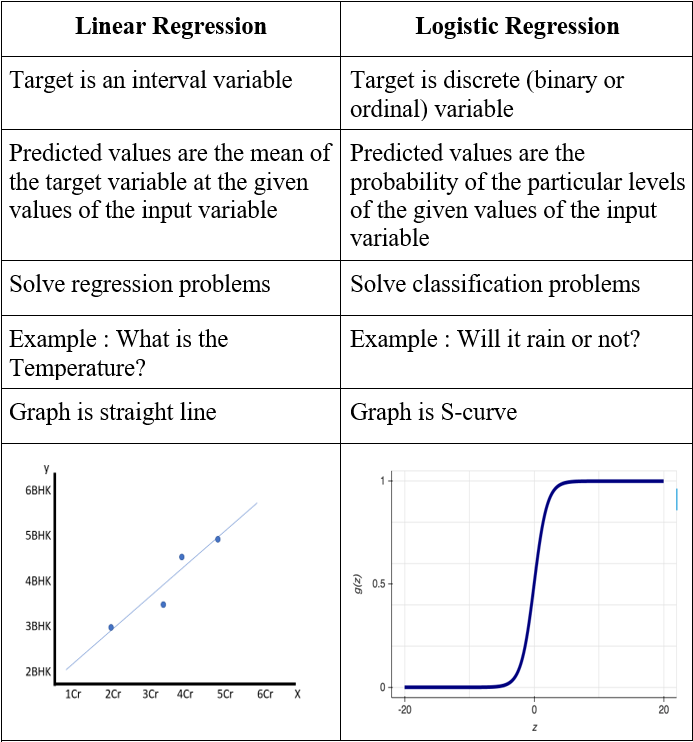
Artificial Learning (AL) is a subfield of AI that focuses on creating algorithms and models that can learn from data. AL involves the use of statistical and machine learning techniques to train models on data and make predictions or decisions. AL is based on the idea that machines can learn from data and improve their performance over time.

Deep Learning (DL) is a subfield of AL that is concerned with the creation of neural networks that can learn from data. DL involves the use of artificial neural networks with multiple layers to perform complex tasks such as image recognition, speech recognition, and natural language processing. DL is based on the idea that neural networks can simulate the behavior of the human brain and learn to recognize patterns in data.

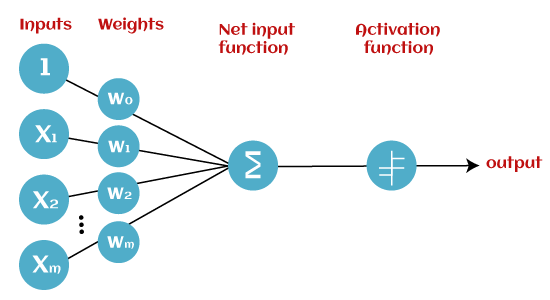
Therefore, DS provides the necessary tools and techniques for collecting and analyzing data, while AL and DL provide the methods for creating models that can learn from that data. AI applications often involve a combination of these subfields to create intelligent systems that can perform complex tasks such as natural language processing, computer vision, and autonomous decision-making.

In conclusion, AI, AL, DL, and DS are closely related and work together to create intelligent systems. DS provides the necessary foundation for AI applications, while AL and DL provide the methods for creating models that can learn from data and make predictions or decisions. By combining these subfields, researchers and practitioners can create powerful and intelligent systems that can perform complex tasks and improve over time.

**Q5. Differentiate between linear and logistic regression?**

****

**Q6. Explain the perceptron update rule?**

**Perceptron is also understood as an Artificial Neuron or neural network unit that helps to detect certain input data computations in business intelligence**.

* **Input Nodes or Input Layer:**

This is the primary component of Perceptron which accepts the initial data into the system for further processing. Each input node contains a real numerical value.

* **Wight and Bias:**

Weight parameter represents the strength of the connection between units. This is another most important parameter of Perceptron components. Weight is directly proportional to the strength of the associated input neuron in deciding the output. Further, Bias can be considered as the line of intercept in a linear equation.

* **Activation Function:**

These are the final and important components that help to determine whether the neuron will fire or not. Activation Function can be considered primarily as a step function.

Types of Activation functions:

* Sign function
* Step function, and
* Sigmoid function

## Simple Ensemble Techniques

In this section, we will look at a few simple but powerful techniques, namely:

1. Max Voting
2. Averaging
3. Weighted Averaging

#### 2.1 Max Voting

The max voting method is generally used for classification problems. In this technique, multiple models are used to make predictions for each data point. The predictions by each model are considered as a ‘vote’. The predictions which we get from the majority of the models are used as the final prediction.

For example, when you asked 5 of your colleagues to rate your movie (out of 5); we’ll assume three of them rated it as 4 while two of them gave it a 5. Since the majority gave a rating of 4, the final rating will be taken as 4. **You can consider this as taking the mode of all the predictions.**

The result of max voting would be something like this:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Colleague 1 | Colleague 2 | Colleague 3 | Colleague 4 | Colleague 5 | Final rating |
| 5 | 4 | 5 | 4 | 4 | 4 |

#### 2.2 Averaging

Similar to the max voting technique, multiple predictions are made for each data point in averaging. In this method, we take an average of predictions from all the models and use it to make the final prediction. Averaging can be used for making predictions in regression problems or while calculating probabilities for classification problems.

For example, in the below case, the averaging method would take the average of all the values.

i.e. (5+4+5+4+4)/5 = 4.4

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Colleague 1 | Colleague 2 | Colleague 3 | Colleague 4 | Colleague 5 | Final rating |
| 5 | 4 | 5 | 4 | 4 | 4.4 |

#### 2.3 Weighted Average

This is an extension of the averaging method. All models are assigned different weights defining the importance of each model for prediction. For instance, if two of your colleagues are critics, while others have no prior experience in this field, then the answers by these two friends are given more importance as compared to the other people.

The result is calculated as [(5\*0.23) + (4\*0.23) + (5\*0.18) + (4\*0.18) + (4\*0.18)] = 4.41.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Colleague 1 | Colleague 2 | Colleague 3 | Colleague 4 | Colleague 5 | Final rating |
| weight | 0.23 | 0.23 | 0.18 | 0.18 | 0.18 |
| rating | 5 | 4 | 5 | 4 | 4 | 4.41 |

**Q7. Differentiate between Markov Models and Hidden Markov Models?**

****

**Q8. Application of clustering?**

Clustering is a machine learning technique used to group similar data points together based on their similarities or distance from each other in a high-dimensional space. Clustering has a wide range of applications in various fields, including data mining, image processing, bioinformatics, social network analysis, and many more. Some of the common applications of clustering are discussed below.

1. Image segmentation: Clustering is widely used in image processing to segment an image into different regions based on their similarities. This is useful in various applications such as object recognition, tracking, and classification.

2. Customer segmentation: Clustering is used in marketing to group customers with similar characteristics or behavior. This helps companies to target their marketing campaigns more effectively and personalize their offerings.

3. Anomaly detection: Clustering can be used to identify anomalies or outliers in a dataset. This is useful in fraud detection, network intrusion detection, and other applications where identifying unusual patterns is important.

4. Recommendation systems: Clustering can be used to group similar items or products together based on their features, and then recommend similar items to users based on their preferences.

5. Natural language processing: Clustering can be used in text analysis to group similar documents or sentences together based on their semantic similarity. This is useful in applications such as document classification, topic modeling, and sentiment analysis.

Overall, clustering is a powerful machine learning technique with a wide range of applications. It is an unsupervised learning technique, meaning that it can be applied to datasets where there are no pre-existing labels or categories.

**Q9. Problems on Partitioning clustering, fuzzy clustering?**

Here are some common problems associated with partitioning clustering and fuzzy clustering:

Partitioning clustering:

1. Sensitivity to initial conditions: Partitioning clustering algorithms such as K-means are sensitive to the initial selection of cluster centers, which can result in suboptimal or different results if the algorithm is run multiple times.

2. Determining the optimal number of clusters: It can be difficult to determine the optimal number of clusters in partitioning clustering. If the number of clusters is set too high or too low, the resulting clusters may not be meaningful or useful.

3. Scalability: Partitioning clustering algorithms can be computationally expensive and may not scale well to large datasets.

Fuzzy clustering:

1. Determining the optimal fuzziness parameter: In fuzzy clustering, the fuzziness parameter determines the degree to which a data point can belong to multiple clusters. However, determining the optimal fuzziness parameter can be challenging and may require domain knowledge or trial-and-error.

2. Interpretability: Fuzzy clustering can be more difficult to interpret compared to partitioning clustering, as data points can belong to multiple clusters with varying degrees of membership.

3. Computational complexity: Fuzzy clustering algorithms can be computationally expensive and may not scale well to large datasets.

4. Overlapping clusters: Fuzzy clustering can result in overlapping clusters, which may not be desired or interpretable in some applications.

Overall, both partitioning clustering and fuzzy clustering have their advantages and limitations, and the choice of which clustering method to use will depend on the specific characteristics of the dataset and the goals of the analysis.

**Q10.Explain different type of clustering?**

Clustering is a machine learning technique used to group similar data points together based on their similarities or distance from each other in a high-dimensional space. There are several types of clustering algorithms, including:

1. Partitioning Clustering: Partitioning clustering algorithms, such as K-means and K-medoids, partition a dataset into k number of clusters by assigning each data point to a single cluster. The partitioning process aims to minimize the sum of squared distances between the data points and their assigned cluster centers.

2. Hierarchical Clustering: Hierarchical clustering algorithms, such as agglomerative and divisive clustering, build a hierarchy of clusters by recursively merging or splitting clusters based on their similarities. Hierarchical clustering can result in a dendrogram, which shows the hierarchical relationship between clusters.

3. Density-based Clustering: Density-based clustering algorithms, such as DBSCAN and OPTICS, group data points together based on their density. Data points that are close together and have a high density are considered to be part of the same cluster.

4. Fuzzy Clustering: Fuzzy clustering algorithms, such as Fuzzy C-means, assign a degree of membership to each data point for each cluster, rather than assigning each point to a single cluster. Fuzzy clustering allows data points to belong to multiple clusters with varying degrees of membership.

5. Model-based Clustering: Model-based clustering algorithms, such as Gaussian Mixture Models, assume that the data points are generated from a probabilistic model and aim to estimate the parameters of the model to identify the underlying clusters.

6. Subspace Clustering: Subspace clustering algorithms, such as CLIQUE and PROCLUS, identify clusters in high-dimensional datasets where the data points may only be similar in a subset of dimensions, or subspaces.

**Q11. Explain DBSCAN?**

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a density-based clustering algorithm that groups together data points that are in high-density regions, while leaving points that are in low-density regions as noise. DBSCAN is particularly useful for datasets with complex and irregularly-shaped clusters.

DBSCAN works by defining a neighborhood around each data point, based on a distance metric and a neighborhood radius parameter. Data points that are within the neighborhood radius are considered to be neighbors. DBSCAN then identifies core points, which are data points with a minimum number of neighbors within the neighborhood radius. These core points form the centers of clusters.

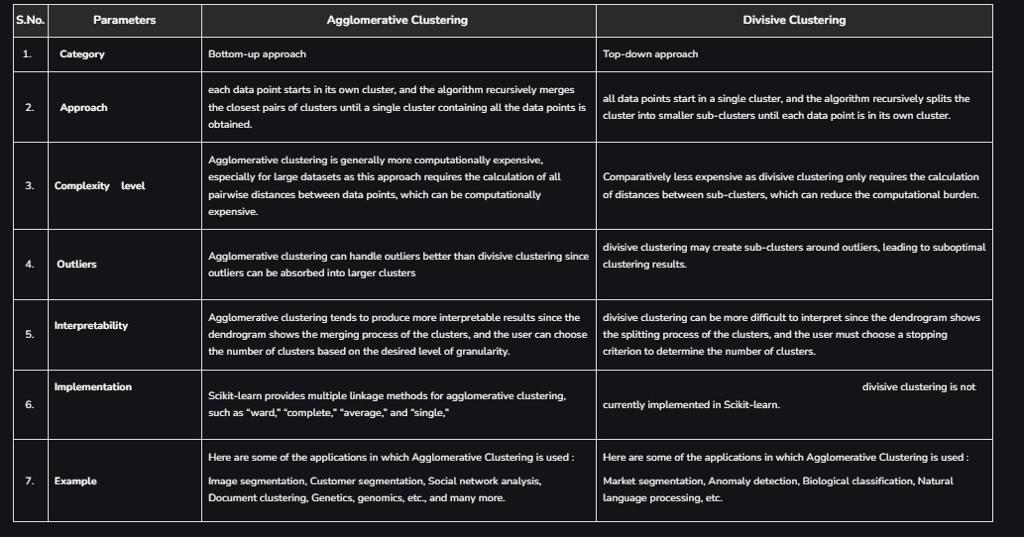
DBSCAN then expands the clusters by adding neighboring points to the cluster, as long as they meet the minimum number of neighbors criterion. If a data point is not a core point, but is within the neighborhood of a core point, it is considered to be part of the same cluster.

DBSCAN can also identify noise points, which are data points that do not belong to any cluster. These are data points that are not within the neighborhood radius of any core point.

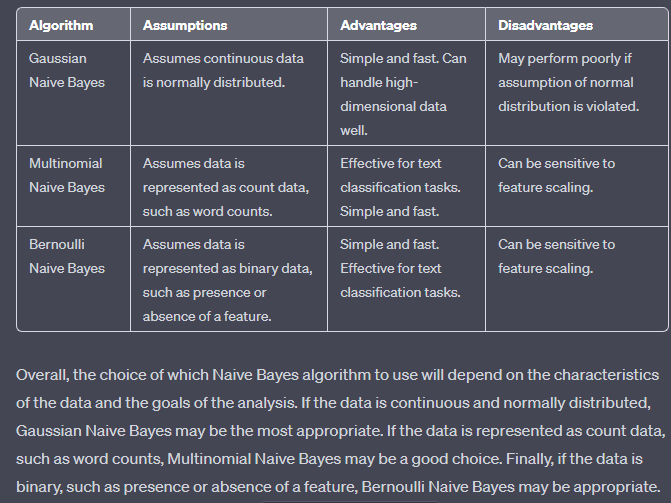
The performance of DBSCAN can be affected by the choice of distance metric, neighborhood radius, and minimum number of neighbors. These parameters can be adjusted to optimize the clustering results for a given dataset.

DBSCAN has several advantages over other clustering algorithms. It can identify clusters of arbitrary shape, and can handle datasets with noise and outliers. DBSCAN is also relatively efficient, with a time complexity of O(n log n). However, DBSCAN can be sensitive to the choice of parameters and can be computationally expensive for very large datasets.

**Q12. Differentiate between agglomerative and divisive algorithm?**

****

**Q13. Differentiate between different type of Naïve Bayes algorithm?**

****

**Q14. Explain collaborative filtering?**

Collaborative filtering is a technique used in recommendation systems to provide personalized recommendations to users based on their past behavior and the behavior of similar users. The basic idea behind collaborative filtering is that if two users have similar preferences for a set of items, they are likely to have similar preferences for other items as well.

Collaborative filtering works by analyzing a dataset of user-item interactions, such as ratings or purchases, and identifying patterns of similarity between users or items. There are two main types of collaborative filtering:

1. User-based collaborative filtering: This approach identifies similar users based on their past behavior and recommends items that the similar users have rated highly. For example, if user A and user B have both rated several movies highly, and user A has not yet rated a new movie, user-based collaborative filtering would recommend that movie to user A based on the fact that user B has rated it highly.

2. Item-based collaborative filtering: This approach identifies similar items based on their past behavior and recommends items that are similar to items that the user has already rated highly. For example, if a user has rated several action movies highly, item-based collaborative filtering would recommend other action movies to the user based on their similarity to the movies the user has already enjoyed.

**Q15. Define-Recall, precise, error, confusion matrix, spectral clustering**

1. Recall (also known as sensitivity): A performance metric that measures the proportion of true positives (i.e., correctly identified positive cases) out of all actual positive cases. Mathematically, recall is calculated as: true positives / (true positives + false negatives).

2. Precision: A performance metric that measures the proportion of true positives out of all predicted positive cases. Mathematically, precision is calculated as: true positives / (true positives + false positives).

3. Error: A measure of the deviation between the predicted and actual values in a model. There are many types of error, including mean squared error, mean absolute error, and root mean squared error, among others.

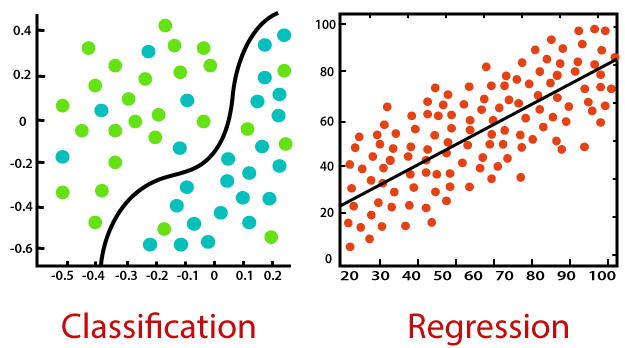
4. Confusion matrix: A table used to evaluate the performance of a classification model. A confusion matrix shows the number of true positives, true negatives, false positives, and false negatives for each class in a classification problem.

5. Spectral clustering: A clustering technique that uses the eigenvalues and eigenvectors of a similarity matrix to group together data points. Spectral clustering is particularly useful for datasets with complex and irregularly-shaped clusters, and can identify clusters that are not easily separable using other clustering techniques.

**Q16. Compare Classification with regression with an example.?**

The main difference between Regression and Classification algorithms that Regression algorithms are used to **predict the continuous** values such as price, salary, age, etc. and Classification algorithms are used to **predict/Classify the discrete values** such as Male or Female, True or False, Spam or Not Spam, etc.

|  |  |
| --- | --- |
| **Regression Algorithm** | **Classification Algorithm** |
| In Regression, the output variable must be of continuous nature or real value. | In Classification, the output variable must be a discrete value. |
| The task of the regression algorithm is to map the input value (x) with the continuous output variable(y). | The task of the classification algorithm is to map the input value(x) with the discrete output variable(y). |
| Regression Algorithms are used with continuous data. | Classification Algorithms are used with discrete data. |
| In Regression, we try to find the best fit line, which can predict the output more accurately. | In Classification, we try to find the decision boundary, which can divide the dataset into different classes. |
| Regression algorithms can be used to solve the regression problems such as Weather Prediction, House price prediction, etc. | Classification Algorithms can be used to solve classification problems such as Identification of spam emails, Speech Recognition, Identification of cancer cells, etc. |
| The regression Algorithm can be further divided into Linear and Non-linear Regression. | The Classification algorithms can be divided into Binary Classifier and Multi-class Classifier. |



**Example:** The best example to understand the Classification problem is Email Spam Detection. The model is trained on the basis of millions of emails on different parameters, and whenever it receives a new email, it identifies whether the email is spam or not. If the email is spam, then it is moved to the Spam folder

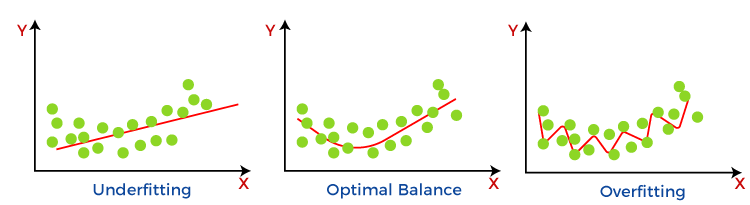
**Example:** Suppose we want to do weather forecasting, so for this, we will use the Regression algorithm. In weather prediction, the model is trained on the past data, and once the training is completed, it can easily predict the weather for future days.

**Q17. Distinguish between overfitting and underfitting. How it can affect model generalization?**

Overfitting and underfitting are two common problems in machine learning that can affect the generalization performance of a model.

Overfitting occurs when a model learns the noise in the training data instead of the underlying pattern, resulting in a model that is too complex and fits the training data too well. This means that the model may not generalize well to new, unseen data. Signs of overfitting include high training accuracy but low test accuracy, as well as large differences between the training and test accuracy. Overfitting can occur when a model is too complex, when there is insufficient training data, or when the model is trained for too many epochs.

On the other hand, underfitting occurs when a model is too simple and cannot capture the underlying pattern in the data, resulting in a model that does not fit the training data well. This means that the model may also not generalize well to new, unseen data. Signs of underfitting include low training and test accuracy, as well as no significant difference between the training and test accuracy. Underfitting can occur when a model is too simple, when there is insufficient training data, or when the model is trained for too few epochs.

Both overfitting and underfitting can affect the generalization performance of a model, which is its ability to perform well on new, unseen data. Overfitting can cause a model to perform well on the training data but poorly on new data, while underfitting can cause a model to perform poorly on both the training and test data. To address these issues, techniques such as regularization, early stopping, and increasing the amount of training data can be used to prevent overfitting, while increasing the model complexity and improving the quality of training data can help address underfitting. 

**Q18. Explain the basic elements of a Hidden Markov Model (HMM). List any two applications of HMM.**

A Hidden Markov Model (HMM) is a statistical model used to represent sequences of observations, where the underlying states that generate the observations are unknown. HMMs are commonly used in speech recognition, handwriting recognition, and natural language processing. The basic elements of an HMM are:

1. States: The states represent the underlying system or process that generates the observations. In an HMM, the states are hidden or unobserved.

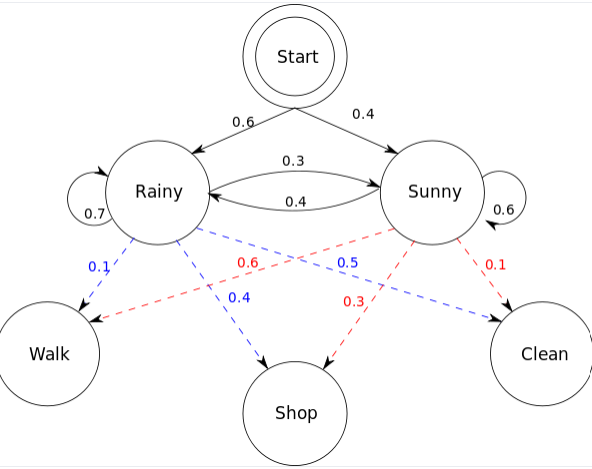
2. Observations: The observations are the visible outputs that are produced by the system or process. In an HMM, the observations are the only information available to the observer.

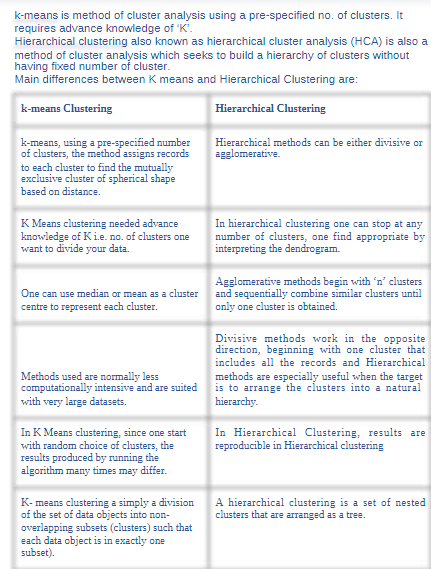
3. Transition probabilities: The transition probabilities represent the probability of transitioning from one state to another. These probabilities are typically represented in a transition matrix.

4. Emission probabilities: The emission probabilities represent the probability of observing a given observation given a particular state. These probabilities are typically represented in an emission matrix.

Two applications of HMM are:

1. Speech recognition: HMMs are commonly used in speech recognition systems to model the relationship between spoken words and the corresponding acoustic signals. In this application, the states correspond to phonemes, while the observations correspond to acoustic features such as frequency and amplitude.

2. Handwriting recognition: HMMs can also be used in handwriting recognition to model the relationship between handwritten characters and the corresponding strokes or pen movements. In this application, the states correspond to different pen movements, while the observations correspond to the strokes themselves.

**Q19. Compare K means clustering with Hierarchical Clustering Techniques.?**

**Discuss any four examples of machine learning applications.**

Machine learning has numerous applications across various fields. Here are four examples of machine learning applications:

1. Image and Speech Recognition: Machine learning algorithms can be used for image and speech recognition. For example, facial recognition software can be trained to recognize faces in photos, while speech recognition software can be trained to transcribe audio into text. Applications of this technology include virtual assistants like Siri, Alexa, and Google Assistant.

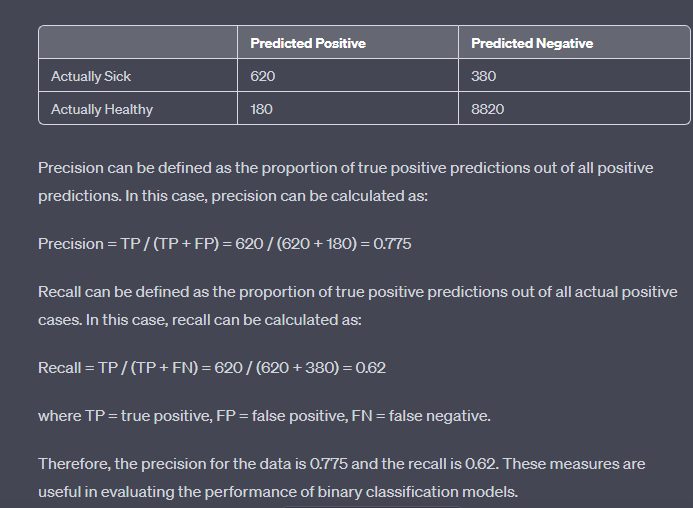
2. Fraud Detection: Machine learning algorithms can be used to identify fraudulent transactions in real-time, enabling companies to quickly detect and prevent financial fraud. The algorithms can learn patterns and behaviors associated with fraudulent activity and flag any transactions that deviate from those patterns.

3. Predictive Maintenance: Machine learning can be used to predict equipment failure and prevent downtime. By analyzing data from sensors and other sources, algorithms can learn to identify patterns that indicate an imminent equipment failure. This can help companies avoid costly repairs and minimize downtime.

4. Personalized Marketing: Machine learning algorithms can be used to personalize marketing efforts. By analyzing data on consumer behavior, companies can develop models to predict which products or services a consumer is likely to purchase, and tailor marketing efforts accordingly. This can lead to increased sales and customer satisfaction.

These are just a few examples of the many applications of machine learning. Machine learning has the potential to revolutionize many industries and improve the way we live and work.

**Q20. Suppose 10000 patients get tested for flu; out of them, 9000 are actually healthy and 1000 are actually sick. For the sick people, a test was positive for 620 and negative for 380. For the healthy people, the same test was positive for 180 and negative for 8820. Construct a confusion matrix for the data and compute the precision and recall for the data.**

**Q**

**Q21. Explain DBSCAN algorithm for density based clustering. List out its advantages compared to K-means.?**

DBSCAN (Density-Based Spatial Clustering of Applications with Noise) is a popular clustering algorithm used in machine learning and data mining. The main idea behind DBSCAN is to group together data points that are closely packed together in high-density regions and separate them from the low-density regions. Here is an explanation of how the DBSCAN algorithm works:

1. Choose a point from the dataset that has not been visited before and identify all the points within its ε-neighborhood (a circle centered on the chosen point with radius ε).

2. If the number of points within the ε-neighborhood is greater than or equal to the minimum number of points required to form a dense region (minPts), then a new cluster is formed.

3. If the number of points within the ε-neighborhood is less than minPts, the point is labeled as noise.

4. If the point belongs to a dense region, its ε-neighborhood is also explored to find all the points that belong to the same cluster. This is done iteratively until all the points in the cluster have been identified.

5. The algorithm continues this process for all the points in the dataset until all the points have been visited.

DBSCAN has several advantages over other clustering algorithms like K-means, such as its ability to discover clusters of arbitrary shape and its robustness to noise and outliers. However, it also has some disadvantages, such as its sensitivity to the choice of parameters ε and minPts.

The advantages of DBSCAN algorithm are:

1. Can discover clusters of arbitrary shapes, unlike K-means which can only discover spherical clusters.

2. Can handle noise and outliers effectively by labeling them as noise points and separating them from the clusters.

3. Does not require the number of clusters to be specified in advance, which can be an advantage when the number of clusters is not known beforehand.

The disadvantages of DBSCAN algorithm are:

1. The performance of the algorithm can be sensitive to the choice of parameters ε and minPts.

2. The algorithm can struggle to identify clusters of varying densities, especially if the parameter values are not set correctly.

3. The algorithm may not work well with high-dimensional data due to the curse of dimensionality.

**Q22.Describe the significance of soft margin hyperplane and explain how they are computed.**

In machine learning, support vector machines (SVMs) are a popular type of supervised learning algorithm that can be used for classification or regression tasks. SVMs work by finding a hyperplane that maximally separates the classes of data points in feature space. The soft margin hyperplane is a variant of the SVM algorithm that allows for some misclassification errors in the training data.

The significance of soft margin hyperplanes is that they can handle datasets that are not linearly separable by allowing some misclassifications. In contrast, the standard hard margin hyperplane SVM requires the data to be completely separable, which may not always be the case in real-world applications. By allowing some misclassification errors, the soft margin hyperplane SVM can still achieve good generalization performance on unseen data.

The computation of a soft margin hyperplane SVM involves the following steps:

1. Given a training dataset with input features X and binary labels y, the SVM aims to find a hyperplane that separates the data points into two classes, such that the margin between the hyperplane and the closest data points (known as support vectors) is maximized.

2. In the soft margin SVM, a slack variable ξi is introduced for each data point i. This variable measures the degree of misclassification for that data point. The objective of the soft margin SVM is to minimize the sum of the slack variables subject to the constraint that the hyperplane correctly separates the data points.

3. The optimization problem can be formulated as a quadratic programming (QP) problem, which can be solved efficiently using standard QP solvers.

4. The hyperplane in the soft margin SVM is defined by a weight vector w and a bias term b, similar to the hard margin SVM. However, the soft margin SVM also has an additional parameter C, known as the regularization parameter, which controls the trade-off between the margin size and the degree of misclassification. A smaller value of C results in a wider margin and more misclassifications, while a larger value of C results in a narrower margin and fewer misclassifications.

5. Once the SVM has been trained on the training data, it can be used to predict the labels of new data points by computing their distance to the hyperplane. Data points that fall on one side of the hyperplane are predicted to belong to one class, while data points that fall on the other side are predicted to belong to the other class.

**Q23. Illustrate K means clustering algorithm with an example.**

K-Means clustering is an unsupervised learning algorithm. There is no labeled data for this clustering, unlike in supervised learning. K-Means performs the division of objects into clusters that share similarities and are dissimilar to the objects belonging to another cluster.

The term ‘K’ is a number. You need to tell the system how many clusters you need to create. For example, K = 2 refers to two clusters. There is a way of finding out what is the best or optimum value of K for a given data.

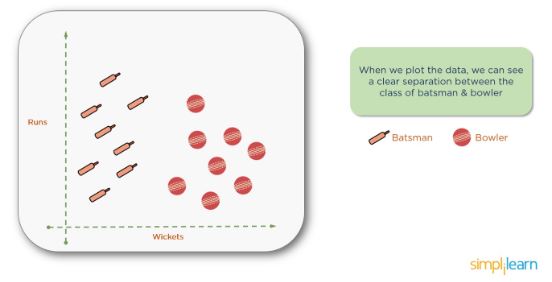
For a better understanding of k-means, let's take an example from cricket. Imagine you received data on a lot of cricket players from all over the world, which gives information on the runs scored by the player and the wickets taken by them in the last ten matches. Based on this information, we need to group the data into two clusters, namely batsman and bowlers.

### **Solution:**

Assign data points

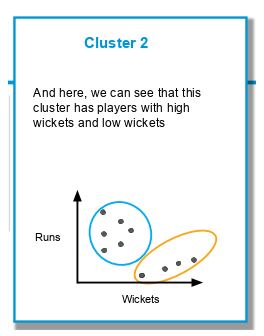
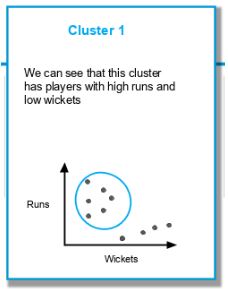
Here, we have our data set plotted on ‘x’ and ‘y’ coordinates. The information on the y-axis is about the runs scored, and on the x-axis about the wickets taken by the players.

If we plot the data, this is how it would look:



### **Perform Clustering**

We need to create the clusters, as shown below:



Considering the same data set, let us solve the problem using K-Means clustering (taking K = 2).

The first step in k-means clustering is the allocation of two centroids randomly (as K=2). Two points are assigned as centroids. Note that the points can be anywhere, as they are random points. They are called centroids, but initially, they are not the central point of a given data set.



The next step is to determine the distance between each of the randomly assigned centroids' data points. For every point, the distance is measured from both the centroids, and whichever distance is less, that point is assigned to that centroid. You can see the data points attached to the centroids and represented here in blue and yellow.



The next step is to determine the actual centroid for these two clusters. The original randomly allocated centroid is to be repositioned to the actual centroid of the clusters.



This process of calculating the distance and repositioning the centroid continues until we obtain our final cluster. Then the centroid repositioning stops.



As seen above, the centroid doesn't need anymore repositioning, and it means the algorithm has converged, and we have the two clusters with a centroid.

**Q24. State the mathematical formulation of the SVM problem. Give an outline of the method for solving the problem.**

The SVM problem is formulated as a constrained optimization problem, as follows:

Given a training set of n data points {xi, yi}, where xi is the i-th data point and yi is its corresponding binary class label (+1 or -1), the SVM seeks to find a hyperplane in the feature space that maximally separates the two classes of data points. The hyperplane is defined by a weight vector w and a bias term b, such that the decision function for a new data point x is:

f(x) = sign(w · x + b)

The goal of the SVM is to find the optimal values of w and b that maximize the margin between the hyperplane and the closest data points of each class. The margin is defined as the distance between the hyperplane and the closest data point of either class.

The optimization problem can be formulated as:

minimize 1/2 ||w||^2

subject to:

yi(w · xi + b) >= 1, for i = 1, 2, ..., n

This is a convex quadratic optimization problem with linear constraints. The objective function is the L2-norm of the weight vector, which penalizes large values of w and encourages a larger margin. The constraints ensure that each data point is correctly classified and lies on the correct side of the hyperplane.

The solution to the SVM problem can be obtained using the following steps:

1. Convert the problem to its dual form: The dual form of the SVM problem involves expressing the optimization problem in terms of Lagrange multipliers, which allows for the use of the kernel trick to handle non-linearly separable data.

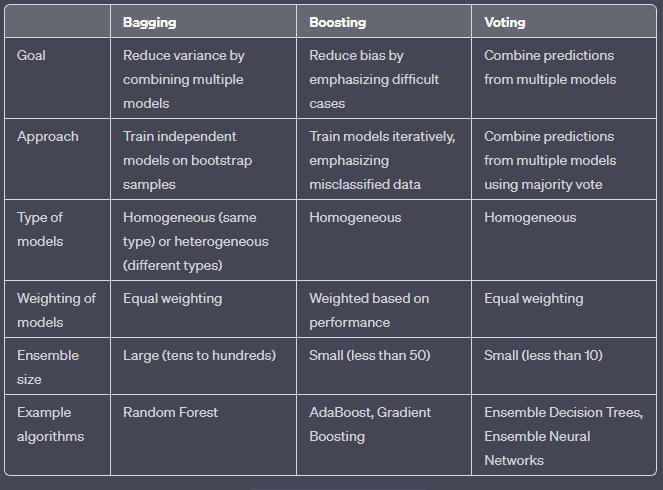
2. Solve the dual problem: The dual problem can be solved using quadratic programming (QP) techniques, which involve solving a system of linear equations involving the Lagrange multipliers.

3. Compute the weight vector and bias term: Once the optimal Lagrange multipliers have been obtained, the weight vector and bias term can be computed using a formula that involves the support vectors, which are the data points that lie on the margin or are misclassified.

4. Use the trained SVM to classify new data points: Once the SVM has been trained, it can be used to classify new data points by computing their distances to the hyperplane.

In summary, the SVM problem is formulated as a constrained optimization problem, which can be solved using the dual form of the problem and quadratic programming techniques. The solution involves computing the weight vector and bias term, which define the hyperplane that separates the data points, and can be used to classify new data points.

**Q25. Differentiate between bagging, boosting and voting.?**



Bagging, boosting, and voting are all ensemble methods that combine multiple models to improve the accuracy and robustness of predictions. Bagging involves training multiple independent models on bootstrap samples of the data to reduce variance, while boosting involves iteratively training models to emphasize difficult cases and reduce bias. Voting combines the predictions from multiple models using a majority vote to improve accuracy.

In bagging, the models can be homogeneous (i.e., of the same type) or heterogeneous (i.e., of different types). The models are trained independently on bootstrap samples of the data, and their predictions are combined using equal weighting.

In boosting, the models are typically homogeneous and trained iteratively, with each new model emphasizing the misclassified data points from the previous model. The models are weighted based on their performance, with poorly performing models receiving more weight to improve their accuracy.

In voting, the models are typically homogeneous and their predictions are combined using a majority vote. The ensemble size is typically small, as the accuracy of the ensemble typically levels off after a few models are added.

Examples of algorithms that use these techniques include Random Forest for bagging, AdaBoost and Gradient Boosting for boosting, and Ensemble Decision Trees and Ensemble Neural Networks for voting.

**Q26. Explain regression with an example.?**

Regression is a type of supervised learning algorithm used to predict a continuous numerical value, such as the price of a house or the weight of an object, based on input features. In regression, the goal is to learn a function that maps the input features to a continuous output value.

For example, consider a dataset of houses with features such as the number of bedrooms, the square footage, and the age of the house. The goal of a regression algorithm would be to learn a function that predicts the price of the house based on these features.

A simple regression model is linear regression, which assumes a linear relationship between the input features and the output value. The linear regression model can be expressed as:

y = b0 + b1\*x1 + b2\*x2 + ... + bn\*xn

where y is the predicted output value, b0 is the intercept term, b1 to bn are the coefficients for the input features x1 to xn, respectively.

To train the linear regression model, we start by splitting the dataset into training and testing sets. We use the training set to fit the model by finding the values of the coefficients b0 to bn that minimize the sum of squared errors between the predicted and actual output values. This is typically done using an optimization algorithm such as gradient descent.

Once the model is trained, we can use the testing set to evaluate its performance by calculating the mean squared error or another evaluation metric.

For example, suppose we have a dataset of houses with features such as the number of bedrooms, square footage, and age, and we want to predict the price of the house. We can train a linear regression model on a subset of the data and use it to predict the prices of the remaining houses. The model might predict the price of a house with 3 bedrooms, 1500 square feet, and 10 years old to be $250,000. We can then evaluate the model's performance by comparing its predictions to the actual prices of the houses in the testing set. If the model has a low mean squared error, we can say that it is a good predictor of house prices based on these features.

**Q27. A patient takes a lab test and the result comes back positive. It is known that the test returns a correct positive result in only 98% of the cases and a correct negative result in only 97% of the cases. Furthermore, only 0.008 of the entire population has this disease.**

**1. What is the probability that this patient has cancer?**

**2. What is the probability that he does not have cancer?**

**3. What is the diagnosis?**

1. To calculate the probability that the patient has cancer given a positive test result, we can use Bayes' theorem:

P(cancer|positive) = P(positive|cancer) \* P(cancer) / P(positive)

where P(cancer|positive) is the probability that the patient has cancer given a positive test result, P(positive|cancer) is the probability of a positive test result given that the patient has cancer (0.98), P(cancer) is the prior probability of the patient having cancer (0.008), and P(positive) is the overall probability of a positive test result, which can be calculated as:

P(positive) = P(positive|cancer) \* P(cancer) + P(positive|no cancer) \* P(no cancer)

P(positive) = 0.98 \* 0.008 + (1 - 0.97) \* (1 - 0.008)

P(positive) = 0.01676

Plugging in the values, we get:

P(cancer|positive) = 0.98 \* 0.008 / 0.01676 = 0.465

So the probability that the patient has cancer given a positive test result is 0.465 or 46.5%.

2. To calculate the probability that the patient does not have cancer given a positive test result, we can use the complement rule:

P(no cancer|positive) = 1 - P(cancer|positive) = 1 - 0.465 = 0.535

So the probability that the patient does not have cancer given a positive test result is 0.535 or 53.5%.

3. Based on the calculations, the patient has a 46.5% chance of having cancer given a positive test result. However, it is important to note that the test has a relatively low accuracy and a high false positive rate, which means that there is a significant chance of a misdiagnosis. Therefore, it would be advisable for the patient to undergo further tests and consultations with medical professionals to determine the correct diagnosis.

**Q28. Discuss the issues involved in decision tree learning.?**

Decision tree learning is a popular and widely used machine learning technique for classification and regression tasks. However, there are several issues involved in decision tree learning that can affect the accuracy and generalization of the resulting models. Some of the major issues are:

1. Overfitting: One of the main issues in decision tree learning is overfitting, which occurs when the tree is too complex and captures noise in the training data rather than the underlying patterns. Overfitting can be addressed by using techniques such as pruning, limiting the tree depth, or using ensemble methods.

2. Bias-variance trade-off: Decision tree models can have high bias (underfitting) or high variance (overfitting), which can affect the model's accuracy and generalization. Techniques such as regularization, cross-validation, and boosting can help to address the bias-variance trade-off and improve the model's performance.

3. Missing values: Decision tree learning can be sensitive to missing values in the training data. There are various strategies for handling missing values, including imputation, ignoring the missing values, or treating them as a separate category.

4. Feature selection: The accuracy of decision tree models can be affected by the choice of features used for training. Selecting irrelevant or redundant features can lead to a suboptimal model. Feature selection techniques such as information gain, chi-squared, or correlation-based methods can help to identify the most relevant features for the task.

5. Handling continuous data: Decision trees work best with discrete or categorical data, but can also be applied to continuous data. However, this requires discretizing the data, which can lead to information loss and affect the accuracy of the model.

6. Handling multi-class classification: Decision trees are typically designed for binary classification tasks, but can also be extended to multi-class classification. However, this requires choosing an appropriate splitting criterion and handling imbalanced classes, which can affect the accuracy of the model.

**Q29. Explain how Support Vector Machine can be used for classification of linearly separable data.**

Support Vector Machines (SVMs) are a powerful machine learning technique that can be used for classification of linearly separable data. The basic idea behind SVMs is to find a hyperplane in the feature space that separates the data into two classes with the maximum margin. The margin is defined as the distance between the hyperplane and the closest data points of each class.

To illustrate how SVM can be used for classification of linearly separable data, consider the following example:

Suppose we have a dataset of two classes, positive (+1) and negative (-1), with two features x1 and x2, and the data can be perfectly separated by a linear hyperplane. The goal is to find the hyperplane that best separates the two classes.

The SVM algorithm works by finding the hyperplane that maximizes the margin, subject to the constraint that all data points are correctly classified. Mathematically, this can be formulated as an optimization problem:

minimize: 1/2 \* ||w||^2

subject to: yi (w.T xi + b) >= 1

where w is the weight vector, b is the bias term, xi is a feature vector, yi is the label (+1 or -1), and ||w|| is the Euclidean norm of the weight vector.

The optimization problem is solved using a technique called Lagrange multipliers, which results in a set of weights w and bias b that define the optimal hyperplane. The hyperplane is given by the equation:

w.T x + b = 0

where w is the weight vector and b is the bias term.

Once the optimal hyperplane is found, new data points can be classified by evaluating which side of the hyperplane they lie on. Points on one side of the hyperplane are classified as positive (+1), and points on the other side are classified as negative (-1).

In summary, SVM can be used for classification of linearly separable data by finding the hyperplane that maximizes the margin between the two classes, subject to the constraint that all data points are correctly classified. The resulting hyperplane can be used to classify new data points by evaluating which side of the hyperplane they lie on.

**Q30. Describe the random forest algorithm to improve classifier accuracy.?**

Random forest is an ensemble learning algorithm used for classification and regression. It is a combination of multiple decision trees, where each tree is trained on a different subset of the data and using a different subset of the features.

The random forest algorithm works as follows:

1. Randomly select a subset of the training data with replacement (known as bootstrap aggregating or bagging).

2. Randomly select a subset of the features for each tree to use, ensuring that each tree uses a different subset of features.

3. Grow a decision tree using the selected subset of data and features, using a splitting criterion such as Gini impurity or entropy to determine the best split at each node.

4. Repeat steps 1-3 to create a forest of decision trees.

5. To classify a new data point, pass it through each decision tree in the forest and count the number of times it is classified as each class.

6. The final classification is the class with the highest number of votes.

Random forest has several advantages over a single decision tree. It reduces overfitting by averaging the predictions of many decision trees, each trained on a different subset of the data. It can also handle missing data and outliers.

In summary, the random forest algorithm improves classifier accuracy by combining multiple decision trees trained on different subsets of the data and features, reducing overfitting and increasing robustness.

**Q31. Differentiate between boosting and bagging?**

| **S.NO** | **Bagging** | **Boosting** |
| --- | --- | --- |
| **1.** | **The simplest way of combining predictions that  belong to the same type.** | **A way of combining predictions that  belong to the different types.** |
| **2.** | **Aim to decrease variance, not bias.** | **Aim to decrease bias, not variance.** |
| **3.** | **Each model receives equal weight.** | **Models are weighted according to their performance.** |
| **4.** | **Each model is built independently.** | **New models are influenced  by the performance of previously built models.** |
|  |  |  |
| **5.** | **Different training data subsets are selected using row sampling with replacement and random sampling methods from the entire training dataset.** | **Every new subset contains the elements that were misclassified by previous models.** |
| **6.** | **Bagging tries to solve the over-fitting problem.** | **Boosting tries to reduce bias.** |
| **7.** | **If the classifier is unstable (high variance), then apply bagging.** | **If the classifier is stable and simple (high bias) the apply boosting.** |
| **8.** | **In this base classifiers are trained parallelly.** | **In this base classifiers are trained sequentially.** |
| **9** | **Example: The Random forest model uses Bagging.** | **Example: The AdaBoost uses Boosting techniques** |

**Bagging**: It is a homogeneous weak learners’ model that learns from each other independently in parallel and combines them for determining the model average.

**Boosting**: It is also a homogeneous weak learners’ model but works differently from Bagging. In this model, learners learn sequentially and adaptively to improve model predictions of a learning algorithm.

## Bagging

**B**ootstrap **A**ggregating, also known as bagging, is a machine learning ensemble meta-algorithm designed to improve the stability and accuracy of machine learning algorithms used in statistical classification and regression. It decreases the [variance](https://www.geeksforgeeks.org/mathematics-mean-variance-and-standard-deviation/)and helps to avoid [overfitting](https://www.geeksforgeeks.org/underfitting-and-overfitting-in-machine-learning/). It is usually applied to [decision tree methods](https://www.geeksforgeeks.org/decision-tree/). Bagging is a special case of the model averaging approach.

**Implementation Steps of Bagging**

* **Step 1:** Multiple subsets are created from the original data set with equal tuples, selecting observations with replacement.
* **Step 2:** A base model is created on each of these subsets.
* **Step 3:**Each model is learned in parallel with each training set and independent of each other.
* **Step 4:**The final predictions are determined by combining the predictions from all the models.

## Boosting

Boosting is an ensemble modeling technique that attempts to build a strong classifier from the number of weak classifiers. It is done by building a model by using weak models in series. Firstly, a model is built from the training data. Then the second model is built which tries to correct the errors present in the first model. This procedure is continued and models are added until either the complete training data set is predicted correctly or the maximum number of models is added

**Algorithm**

1. *Initialise the dataset and assign equal weight to each of the data point.*
2. *Provide this as input to the model and identify the wrongly classified data points.*
3. *Increase the weight of the wrongly classified data points and decrease the weights of correctly classified data points. And then normalize the weights of all data points.*
4. *if (got required results)  
     Goto step 5  
   else  
     Goto step 2*
5. *End*

**Q32. Explain different type boosting?**

Boosting is an ensemble learning technique that combines multiple weak classifiers to create a strong classifier. The basic idea of boosting is to train a sequence of weak classifiers, each of which is trained to correct the mistakes made by the previous classifiers. There are several different types of boosting algorithms, including:

1. AdaBoost (Adaptive Boosting): In AdaBoost, each instance is assigned a weight and a series of weak classifiers are trained in sequence. After each classifier is trained, the weights of the misclassified instances are increased, so that they are more likely to be correctly classified by the next classifier. The final classification is determined by a weighted vote of all the classifiers.

2. Gradient Boosting: In gradient boosting, a series of decision trees are trained in sequence, each of which is trained to correct the mistakes made by the previous tree. The objective function to be optimized is the negative gradient of the loss function, so that each subsequent tree focuses on the instances that were incorrectly classified by the previous trees.

3. XGBoost (Extreme Gradient Boosting): XGBoost is an optimized implementation of gradient boosting that uses a combination of tree-based and linear models to improve accuracy and speed. It includes several advanced features, such as regularization, early stopping, and custom loss functions.

4. LightGBM (Light Gradient Boosting Machine): LightGBM is a gradient boosting framework that uses a histogram-based approach to split the data, which reduces the memory usage and improves the training speed. It also includes several advanced features, such as categorical feature support, parallel learning, and GPU acceleration.

In summary, boosting is a powerful technique for improving the accuracy of machine learning models by combining multiple weak classifiers. AdaBoost, gradient boosting, XGBoost, and LightGBM are different types of boosting algorithms that vary in their optimization objectives, learning algorithms, and advanced features.

**Q33. Explain different ensembling learning?**

Ensemble learning is a technique of combining multiple machine learning models to improve the overall performance of the system. There are several types of ensemble learning, including:

1. Bagging (Bootstrap Aggregating): Bagging involves training multiple models on different subsets of the training data, selected randomly with replacement (known as bootstrapping). Each model is trained independently and produces a prediction, which is combined using a simple voting scheme.

2. Boosting: Boosting is a technique that trains a sequence of models, each of which is trained to correct the errors of the previous models. Each subsequent model focuses on the instances that were incorrectly classified by the previous models, resulting in a more accurate final prediction.

3. Stacking: Stacking involves training multiple models and combining their predictions using a meta-model. The meta-model is trained on the output of the individual models and produces the final prediction.

4. Blending: Blending is a technique that is similar to stacking, but instead of using a meta-model, it combines the predictions of multiple models using a weighted average.

5. Random Forest: Random forest is an ensemble learning technique that combines multiple decision trees, where each tree is trained on a different subset of the data and features. The final prediction is based on the majority vote of the individual trees.

In summary, ensemble learning is a powerful technique for improving the performance of machine learning models. Bagging, boosting, stacking, blending, and random forest are different types of ensemble learning that vary in their training and combining mechanisms.

**Q34. Short notes**

**a) Out of bag evaluation b)random forest c)random patches d)random subspaces e)Naïve Bayes**

a) Out-of-bag evaluation: Out-of-bag (OOB) evaluation is a technique used in bagging algorithms, such as random forest, to estimate the performance of the model without the need for a separate validation set. In OOB evaluation, each instance in the training data has a probability of being included in the bootstrap sample. The instances that are not selected in the bootstrap sample are referred to as OOB instances. The model is trained on the bootstrap sample and the OOB instances are used for evaluation. This provides an estimate of the model's performance on new, unseen data.

b) Random forest: Random forest is an ensemble learning algorithm that combines multiple decision trees to improve the accuracy and reduce the variance of the model. Each tree is trained on a bootstrap sample of the data, and at each node, a random subset of features is considered for splitting. The final prediction is based on the majority vote of the individual trees. Random forest is a powerful algorithm that can handle high-dimensional data, nonlinear relationships, and missing values.

c) Random patches: Random patches is a variation of the bagging algorithm, where subsets of both instances and features are selected randomly to train each model in the ensemble. This approach is useful for high-dimensional data, where there are more features than instances. By randomly selecting a subset of features and instances, it reduces the variance of the model and avoids overfitting.

d) Random subspaces: Random subspaces is another variation of the bagging algorithm, where subsets of features are selected randomly to train each model in the ensemble. This approach is useful for data where each instance has a large number of features, but only a subset of those features are relevant for prediction.

e) Naïve Bayes: Naïve Bayes is a probabilistic algorithm based on Bayes' theorem, which assumes that all features are independent of each other. It is a simple yet powerful algorithm that is commonly used for classification tasks, especially for text classification. Naïve Bayes calculates the probability of each class given the input features and selects the class with the highest probability as the prediction. Naïve Bayes is fast and requires very little training data, but its assumption of independence may not hold in some real-world scenarios.

**Q35. Define inductive bias? Explain inductive bias in decision tree learning?**

Inductive bias refers to the set of assumptions and biases that a machine learning algorithm uses to make predictions about new, unseen data based on its training data. It is the prior knowledge or beliefs that are built into the learning algorithm, which guide it towards selecting one hypothesis over another.

In decision tree learning, the inductive bias refers to the assumptions that are made about the structure of the decision tree based on the training data. These assumptions can include the following:

1. Occam's razor: The principle that the simplest explanation is the most likely one. In decision tree learning, this means that simpler trees are preferred over more complex ones, all other things being equal.

2. Minimum description length: The idea that the best hypothesis is the one that requires the shortest description, in terms of the number of bits needed to encode it.

3. Attribute selection bias: The idea that some attributes are more important than others for making accurate predictions, and therefore should be given more weight when constructing the decision tree.

4. Tree pruning bias: The bias towards removing nodes and branches that do not contribute much to the accuracy of the tree.

The inductive bias in decision tree learning is important because it can affect the performance of the algorithm on new, unseen data. An overly complex or overly simple tree can lead to overfitting or underfitting, respectively, which can result in poor generalization to new data. Therefore, it is important to choose an appropriate inductive bias that balances simplicity and accuracy in the decision tree.

**Q36. Explain maximum likelihood and least squared hypothesis?**

Maximum likelihood and least squares are two common methods for estimating the parameters of a statistical model.

Maximum likelihood is a method for estimating the parameters of a probability distribution by finding the values of the parameters that maximize the likelihood function, which is the probability of observing the data given the parameters. In other words, it is the method of finding the most probable distribution that could have generated the observed data. The likelihood function is typically expressed as the product of the individual probabilities of each data point, assuming that each data point is independently and identically distributed. Maximum likelihood is commonly used in the context of parametric models, where the distribution of the data is assumed to belong to a certain family of distributions with a finite number of parameters.

Least squares is a method for estimating the parameters of a linear regression model by finding the values of the parameters that minimize the sum of the squared differences between the observed values and the predicted values of the model. In other words, it is the method of finding the line that best fits the data by minimizing the sum of the squared errors between the predicted values and the observed values. The method is widely used for modeling continuous variables and is based on the assumption that the errors are normally distributed with a mean of zero and a constant variance.

Both maximum likelihood and least squares methods are widely used in machine learning and statistical modeling, and each has its own strengths and weaknesses depending on the specific problem and the assumptions made about the data. The choice between the two methods depends on the nature of the problem and the underlying assumptions.

**Q37. What are the requirement of clustering algorithm?**

The requirements of a clustering algorithm are:

1. Similarity measure: A clustering algorithm must define a way to measure the similarity or distance between data points in the dataset. This similarity measure is used to determine which data points are grouped together into a cluster.

2. Number of clusters: The number of clusters that the algorithm should generate must be specified or determined automatically by the algorithm.

3. Scalability: The clustering algorithm should be able to handle large datasets with many dimensions and data points.

4. Robustness: The clustering algorithm should be able to handle noise and outliers in the data without affecting the clustering result significantly.

5. Interpretability: The clustering algorithm should produce results that are easy to interpret and understand.

6. Efficiency: The clustering algorithm should be efficient in terms of time and space complexity, especially for large datasets.

7. Validity: The clustering algorithm should produce clusters that are meaningful and useful for the specific problem domain.

Overall, the requirements of a clustering algorithm depend on the specific problem domain, the characteristics of the data, and the goals of the analysis. Therefore, different clustering algorithms may be more appropriate for different situations and datasets.

**Q38. What is the goal of the support vector machine (SVM)? How to compute the margin?**

The goal of the support vector machine (SVM) is to find the optimal hyperplane that separates the data points of different classes with the maximum margin. The SVM is a supervised learning algorithm that is commonly used for classification tasks.

The margin is the distance between the hyperplane and the closest data points of each class, also known as the support vectors. The SVM algorithm computes the margin by maximizing the distance between the hyperplane and the support vectors. This results in a hyperplane that has the largest possible margin and is therefore most robust to noise and generalizes better to unseen data.

To compute the margin, the SVM algorithm solves a constrained optimization problem, where the objective is to maximize the margin subject to the constraint that all data points are correctly classified. The optimization problem can be formulated as:

maximize 2/||w|| subject to y\_i(w^T x\_i + b) >= 1 for all i

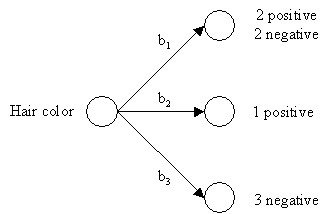
where w is the weight vector of the hyperplane, b is the bias term, x\_i is the i-th data point, y\_i is its label (+1 or -1), and ||w|| is the Euclidean norm of w.

The solution to this optimization problem can be found using techniques such as quadratic programming or gradient descent. The resulting hyperplane can then be used to predict the class of new data points by simply computing their position relative to the hyperplane.

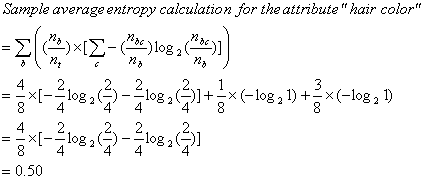
**Q39. For a Sun Burn dataset given below, construct a decision tree**

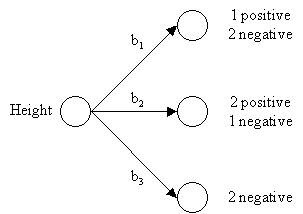
| **Name** | **Hair** | **Height** | **Weight** | **Location** | **Class** |
| --- | --- | --- | --- | --- | --- |
| **Sunita(Sara)** | **blonde** | **average** | **light** | **no** | **yes** |
| **Anit(Dana)** | **blonde** | **tall** | **average** | **yes** | **no** |
| **Kavita(Alex)** | **brown** | **short** | **average** | **yes** | **no** |
| **Sushma(Annie)** | **blonde** | **short** | **average** | **no** | **yes** |
| **Xavier(Emilie)** | **red** | **average** | **heavy** | **no** | **yes** |
| **Balaji(Pete)** | **brown** | **tall** | **heavy** | **no** | **no** |
| **Ramesh(John)** | **brown** | **average** | **heavy** | **no** | **no** |
| **Swetha(Katie)** | **blonde** | **short** | **light** | **yes** | **no** |
|  |  |  |  |  |  |

1. Perform average entropy calculations on the complete data set for each of the four attributes:

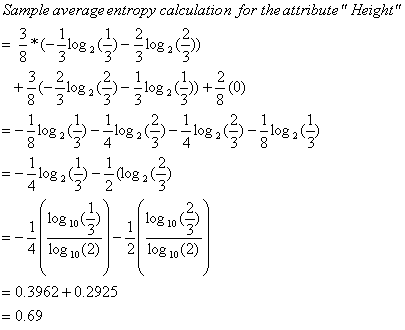


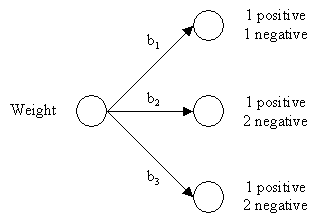
|  |  |
| --- | --- |
| *b1* = blonde *b2* = red *b3* = brown | Average Entropy = 0.50 |



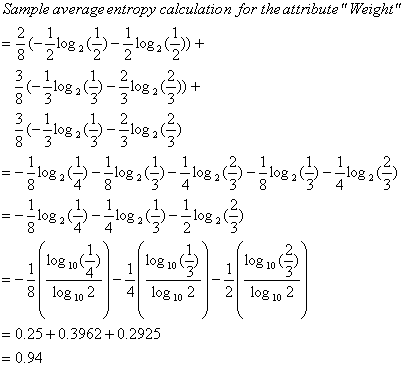


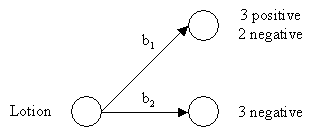
|  |  |
| --- | --- |
| *b1* = short *b2* = average *b3* = tall | Average Entropy = 0.69 |



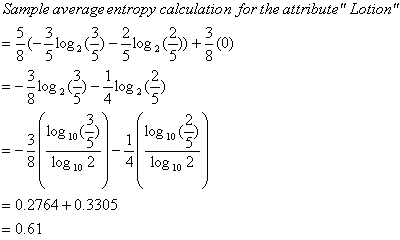


|  |  |
| --- | --- |
| *b1* = light *b2* = average *b3* = heavy | Average Entropy = 0.94 |





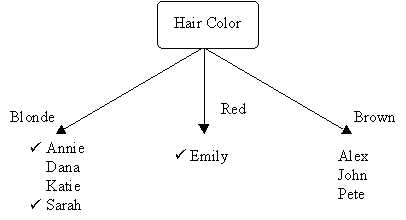
|  |  |
| --- | --- |
| *b1* = no *b2* = yes | Average Entropy = 0.61 |



**Results**

|  |  |
| --- | --- |
| ***Attribute*** | ***Average Entropy*** |
| Hair Color | **0.50** |
| Height | 0.69 |
| Weight | 0.94 |
| Lotion | 0.61 |

The attribute "hair color" is selected as the first test because it minimizes the entropy.



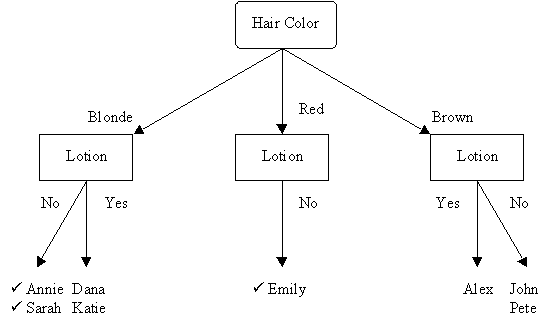
1. Similarily, we now choose another test to separate out the sunburned individuals from the blonde haired inhomogeneous subset, {Sarah, Dana, Annie, and Katie}.

**Results**

|  |  |
| --- | --- |
| ***Attribute*** | ***Average Entropy*** |
| Height | 0.50 |
| Weight | 1.00 |
| Lotion | **0.00** |

The attribute "lotion" is selected because it minimizes the entropy in the blonde hair subset.

Thus, using the "hair color" and "lotion" tests together ensures the proper identification of all the samples.



This is the completed decision tree.

**Q40. Explain machine learning life cycle?**

Machine learning life cycle involves seven major steps, which are given below:

* **Gathering Data**
* **Data preparation**
* **Data Wrangling**
* **Analyse Data**
* **Train the model**
* **Test the model**
* **Deployment**

## 1. Gathering Data:

Data Gathering is the first step of the machine learning life cycle. The goal of this step is to identify and obtain all data-related problems.In this step, we need to identify the different data sources, as data can be collected from various sources such as **files**, **database**, **internet**, or **mobile devices**. It is one of the most important steps of the life cycle. The quantity and quality of the collected data will determine the efficiency of the output. The more will be the data, the more accurate will be the prediction.

This step includes the below tasks:

* **Identify various data sources**
* **Collect data**
* **Integrate the data obtained from different sources**

By performing the above task, we get a coherent set of data, also called as a **dataset**. It will be used in further steps.

## 2. Data preparation

After collecting the data, we need to prepare it for further steps. Data preparation is a step where we put our data into a suitable place and prepare it to use in our machine learning training.In this step, first, we put all data together, and then randomize the ordering of data.

This step can be further divided into two processes:

* **Dataexploration:**  
  It is used to understand the nature of data that we have to work with. We need to understand the characteristics, format, and quality of data.  
  A better understanding of data leads to an effective outcome. In this, we find Correlations, general trends, and outliers.
* **Datapre-processing:**  
  Now the next step is preprocessing of data for its analysis.

## 3. Data Wrangling

Data wrangling is the process of cleaning and converting raw data into a useable format. It is the process of cleaning the data, selecting the variable to use, and transforming the data in a proper format to make it more suitable for analysis in the next step. It is one of the most important steps of the complete In real-world applications, collected data may have various issues, including:

* **Missing Values**
* **Duplicate data**
* **Invalid data**
* **Noise**

So, we use various filtering techniques to clean the data.

It is mandatory to detect and remove the above issues because it can negatively affect the quality of the outcome.

## 4. Data Analysis

Now the cleaned and prepared data is passed on to the analysis step. This step involves:

* **Selection of analytical techniques**
* **Building models**
* **Review the result**

The aim of this step is to build a machine learning model to analyze the data using various analytical techniques and review the outcome. It starts with the determination of the type of the problems, where we select the machine learning techniques such as **Classification**, **Regression**, **Cluster analysis**, **Association**, etc. then build the model using prepared data, and evaluate the model.

## 5. Train Model

Now the next step is to train the model, in this step we train our model to improve its performance for better outcome of the problem.We use datasets to train the model using various machine learning algorithms. Training a model is required so that it can understand the various patterns, rules, and, features.

## 6. Test Model

Once our machine learning model has been trained on a given dataset, then we test the model. In this step, we check for the accuracy of our model by providing a test dataset to it.Testing the model determines the percentage accuracy of the model as per the requirement of project or problem.

## 7. Deployment

The last step of machine learning life cycle is deployment, where we deploy the model in the real-world system.If the above-prepared model is producing an accurate result as per our requirement with acceptable speed, then we deploy the model in the real system. But before deploying the project, we will check whether it is improving its performance using available data or not. The deployment phase is similar to making the final report for a project.

**Q41. Define ML? Explain with specific example?**

Machine Learning (ML) is a branch of Artificial Intelligence (AI) that involves the development of algorithms and statistical models that enable computer systems to automatically improve their performance on a task by learning from experience, without being explicitly programmed. In other words, ML enables computer systems to identify patterns in data, learn from them, and make decisions or predictions based on that learning. This technology has numerous applications, including natural language processing, image and speech recognition, recommendation systems, predictive analytics, and fraud detection, among others. The goal of ML is to enable computers to perform tasks that would normally require human intelligence and intervention, making processes more efficient and accurate.

One example of ML is image recognition. Imagine you want to build a system that can identify different objects in a photo. First, you would need a large dataset of labeled images (i.e., images with objects identified and labeled). Then, you would use an ML algorithm to train a model on this data. The algorithm would learn to identify patterns in the images that correspond to different objects, such as shapes, colors, and textures. Once the model is trained, you could use it to classify new images by feeding them into the model and receiving a prediction of what objects are in the photo.

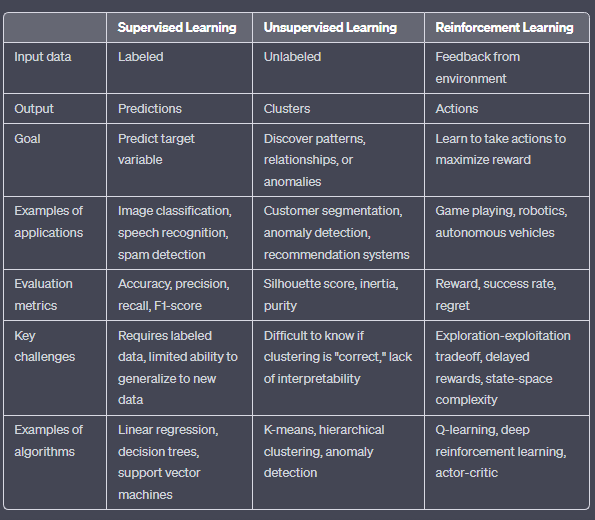
For instance, let's say you train your model on a dataset of labeled images of cats and dogs. After training, you could feed a new image of a cat into the model, and it would correctly predict that the image contains a cat. Similarly, if you feed a new image of a dog, the model would correctly predict that the image contains a dog.

This is just one example of how ML can be used to recognize images. There are many other applications of ML in various domains, such as natural language processing, speech recognition, fraud detection, and predictive maintenance.

**Q42. Explain different type of Machine learning System?**

### Comparison Table

| **Criteria** | **Supervised ML** | **Unsupervised ML** | **Reinforcement ML** |
| --- | --- | --- | --- |
| Definition | Learns by using labelled data | Trained using unlabelled data without any guidance. | Works on interacting with the environment |
| Type of data | Labelled data | Unlabelled data | No – predefined data |
| Type of problems | Regression and classification | Association and Clustering | Exploitation or Exploration |
| Supervision | Extra supervision | No supervision | No supervision |
| Algorithms | Linear Regression, Logistic Regression, SVM, KNN etc. | K – Means, C – Means, Apriori | Q – Learning, SARSA |
| Aim | Calculate outcomes | Discover underlying patterns | Learn a series of action |
| Application | Risk Evaluation, Forecast Sales | Recommendation System, Anomaly Detection | Self Driving Cars, Gaming, Healthcare |

****

**Q43. Explain different type of bias and variance with a suitable graph ?**

## What is Bias?

In general, a machine learning model analyses the data, find patterns in it and make predictions. While training, the model learns these patterns in the dataset and applies them to test data for prediction. **While making predictions, a difference occurs between prediction values made by the model and actual values/expected values**, **and this difference is known as bias errors or Errors due to bias**. It can be defined as an inability of machine learning algorithms such as Linear Regression to capture the true relationship between the data points. Each algorithm begins with some amount of bias because bias occurs from assumptions in the model, which makes the target function simple to learn. A model has either:

* **Low Bias:** A low bias model will make fewer assumptions about the form of the target function.
* **High Bias:** A model with a high bias makes more assumptions, and the model becomes unable to capture the important features of our dataset. **A high bias model also cannot perform well on new data.**

Generally, a linear algorithm has a high bias, as it makes them learn fast. The simpler the algorithm, the higher the bias it has likely to be introduced. Whereas a nonlinear algorithm often has low bias.

Some examples of machine learning algorithms with low bias **are Decision Trees, k-Nearest Neighbours and Support Vector Machines**. At the same time, an algorithm with high bias is **Linear Regression, Linear Discriminant Analysis and Logistic Regression.**

### Ways to reduce High Bias:

High bias mainly occurs due to a much simple model. Below are some ways to reduce the high bias:

* Increase the input features as the model is underfitted.
* Decrease the regularization term.
* Use more complex models, such as including some polynomial features.

## What is a Variance Error?

The variance would specify the amount of variation in the prediction if the different training data was used. In simple words, **variance tells that how much a random variable is different from its expected value**

**Low variance** means there is a small variation in the prediction of the target function with changes in the training data set. At the same time, **High variance** shows a large variation in the prediction of the target function with changes in the training dataset.

A model that shows high variance learns a lot and perform well with the training dataset, and does not generalize well with the unseen dataset. As a result, such a model gives good results with the training dataset but shows high error rates on the test dataset.

Since, with high variance, the model learns too much from the dataset, it leads to overfitting of the model. A model with high variance has the below problems:

* A high variance model leads to overfitting.
* Increase model complexities.

Usually, nonlinear algorithms have a lot of flexibility to fit the model, have high variance.

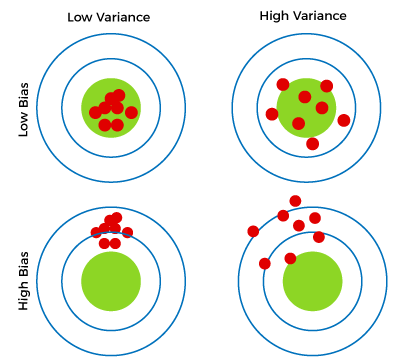
Some examples of machine learning algorithms with low variance are, **Linear Regression, Logistic Regression, and Linear discriminant analysis**. At the same time, algorithms with high variance are **decision tree, Support Vector Machine, and K-nearest neighbours.**

### Ways to Reduce High Variance:

* Reduce the input features or number of parameters as a model is overfitted.
* Do not use a much complex model.
* Increase the training data.
* Increase the Regularization term.

## Different Combinations of Bias-Variance

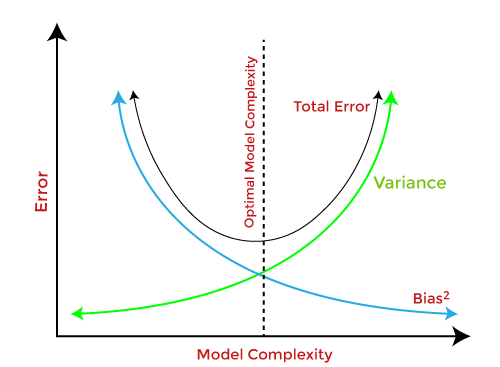
There are four possible combinations of bias and variances, which are represented by the below diagram



1. **Low-Bias, Low-Variance:**  
   The combination of low bias and low variance shows an ideal machine learning model. However, it is not possible practically.
2. **Low-Bias, High-Variance:** With low bias and high variance, model predictions are inconsistent and accurate on average. This case occurs when the model learns with a large number of parameters and hence leads to an **overfitting**
3. **High-Bias, Low-Variance:** With High bias and low variance, predictions are consistent but inaccurate on average. This case occurs when a model does not learn well with the training dataset or uses few numbers of the parameter. It leads to **underfitting** problems in the model.
4. **High-Bias, High-Variance:**  
   With high bias and high variance, predictions are inconsistent and also inaccurate on average.

## Bias-Variance Trade-Off

While building the machine learning model, it is really important to take care of bias and variance in order to avoid overfitting and underfitting in the model. If the model is very simple with fewer parameters, it may have low variance and high bias. Whereas, if the model has a large number of parameters, it will have high variance and low bias. So, it is required to make a balance between bias and variance errors, and this balance between the bias error and variance error is known as **the Bias-Variance trade-off.**



For an accurate prediction of the model, algorithms need a low variance and low bias. But this is not possible because bias and variance are related to each other:

* If we decrease the variance, it will increase the bias.
* If we decrease the bias, it will increase the variance.

**Q44. Explain different data preprocessing technique?**

The steps in data preprocessing in machine learning are:

1. Consolidation after [acquisition of the data](https://www.analytixlabs.co.in/blog/data-acquisition/)
2. Data Cleaning:
   1. Convert the data types if any mismatch present in the data types of the variables
   2. Change the format of the date variable to the required format
   3. Replace the special characters and constants with the appropriate values
3. Detection and treatment of missing values
4. Treating for negative values, if any present depending on the data
5. Outliers detection and treatment
6. Transformation of variables
7. Creation of new derived variables
8. Scale the numerical variables
9. Encode the categorical variables
10. Split the data into training, validation, and test set

**Steps Involved in Data Preprocessing:**

**1. Data Cleaning:**   
The data can have many irrelevant and missing parts. To handle this part, data cleaning is done. It involves handling of missing data, noisy data etc.

* **(a). Missing Data:**   
  This situation arises when some data is missing in the data. It can be handled in various ways.   
  Some of them are:
  1. **Ignore the tuples:**   
     This approach is suitable only when the dataset we have is quite large and multiple values are missing within a tuple.
  2. **Fill the Missing values:**   
     There are various ways to do this task. You can choose to fill the missing values manually, by attribute mean or the most probable value.
* **(b). Noisy Data:**   
  Noisy data is a meaningless data that can’t be interpreted by machines.It can be generated due to faulty data collection, data entry errors etc. It can be handled in following ways :
  1. **Binning Method:**   
     This method works on sorted data in order to smooth it. The whole data is divided into segments of equal size and then various methods are performed to complete the task. Each segmented is handled separately. One can replace all data in a segment by its mean or boundary values can be used to complete the task.
  2. **Regression:**   
     Here data can be made smooth by fitting it to a regression function.The regression used may be linear (having one independent variable) or multiple (having multiple independent variables).
  3. **Clustering:**   
     This approach groups the similar data in a cluster. The outliers may be undetected or it will fall outside the clusters.

**2. Data Transformation:**   
This step is taken in order to transform the data in appropriate forms suitable for mining process. This involves following ways:

1. **Normalization:**   
   It is done in order to scale the data values in a specified range (-1.0 to 1.0 or 0.0 to 1.0)
2. **Attribute Selection:**   
   In this strategy, new attributes are constructed from the given set of attributes to help the mining process.
3. **Discretization:**   
   This is done to replace the raw values of numeric attribute by interval levels or conceptual levels.
4. **Concept Hierarchy Generation:**   
   Here attributes are converted from lower level to higher level in hierarchy. For Example-The attribute “city” can be converted to “country”.

**3. Data Reduction:**   
Since data mining is a technique that is used to handle huge amount of data. While working with huge volume of data, analysis became harder in such cases. In order to get rid of this, we uses data reduction technique. It aims to increase the storage efficiency and reduce data storage and analysis costs.

The various steps to data reduction are:

1. **Data Cube Aggregation:**   
   Aggregation operation is applied to data for the construction of the data cube.
2. **Attribute Subset Selection:**   
   The highly relevant attributes should be used, rest all can be discarded. For performing attribute selection, one can use level of significance and p- value of the attribute.the attribute having p-value greater than significance level can be discarded.
3. **Numerosity Reduction:**   
   This enable to store the model of data instead of whole data, for example: Regression Models.
4. **Dimensionality Reduction:**   
   This reduce the size of data by encoding mechanisms.It can be lossy or lossless. If after reconstruction from compressed data, original data can be retrieved, such reduction are called lossless reduction else it is called lossy reduction. The two effective methods of dimensionality reduction are:Wavelet transforms and PCA (Principal Component Analysis).

**Not answered Question :-**

**Q45. Explain different type of distribution?**

**Q46. Numerical on SVM, knn, K-means , decision tree, confusion matrix?**

**Q47. collaborative filtering numerical also?**

**Q48. Use K Means clustering to cluster the following data into two groups. Assume**

**cluster centroid are m1=2 and m2=4. The distance function used is Euclidean**