1. **Recognize the differences between supervised, semi-supervised, and unsupervised learning.**

Solution - Input Data is provided to the model along with the output in the Supervised Learning. Only input data is provided in Unsupervised Learning. Output is predicted by the Supervised Learning. Hidden patterns in the data can be found using the unsupervised learning model.

Semi-supervised learning is a broad category of machine learning that uses labeled data to ground predictions, and unlabeled data to learn the shape of the larger data distribution.

2. **Describe in detail any five examples of classification problems.**

Solution - Five examples of classification problems in machine learning:

Email Spam Detection: The task is to classify incoming emails as either spam or non-spam (ham). By training a classification model on labeled email data, it can learn patterns and features indicative of spam emails, such as specific keywords, email headers, or email structure, and accurately classify new incoming emails.

Image Classification: In image classification, the goal is to classify images into predefined categories or classes. For instance, a model can be trained to recognize and classify images of animals, distinguishing between cats, dogs, and birds. The model learns from labeled training images and uses features like shapes, textures, and colors to classify unseen images accurately.

Sentiment Analysis: Sentiment analysis involves determining the sentiment expressed in text data, such as customer reviews, social media posts, or product feedback. The task is to classify the text as positive, negative, or neutral. By training a model on labeled text data, it can learn patterns in the text and sentiment-associated words or phrases to classify new, unseen text documents.

Credit Risk Assessment: In credit risk assessment, the objective is to classify loan applicants as either low risk or high risk based on various features like credit history, income, debt-to-income ratio, and other financial indicators. By training a model on historical loan data with known outcomes, it can learn to predict the risk level of new loan applications, helping financial institutions make informed lending decisions.

Medical Diagnosis: Classification models can be used in medical diagnosis to classify patients into different disease categories based on symptoms, medical test results, or medical imaging data. For example, a model can be trained to classify lung X-ray images into categories like normal, pneumonia, or lung cancer. By learning from labeled medical data, the model can assist doctors in accurate and timely diagnosis.

3. **Describe each phase of the classification process in detail.**

Solution - The classification process in machine learning typically involves several phases, from data preparation to model evaluation. Here are the main phases of the classification process:

Data Preparation: Data Collection: Gather relevant data for the classification task, ensuring it is representative and diverse.

Data Cleaning: Handle missing values, outliers, and inconsistencies in the dataset.

Feature Selection/Extraction: Identify and select relevant features or extract new features from the raw data that contribute to the classification task.

Data Transformation: Normalize or scale the features to ensure they are on a similar scale, which can be important for certain algorithms.

Data Splitting: Split the dataset into two or three subsets: training set, validation set, and test set. The training set is used to train the classification model. The validation set is used to tune hyperparameters and evaluate model performance during training. The test set is used to assess the final performance of the trained model on unseen data.

Model Selection: Select an appropriate classification algorithm based on the characteristics of the problem, available data, and performance requirements. Common algorithms include Logistic Regression, Decision Trees, Random Forests, Support Vector Machines (SVM), Naive Bayes, and Neural Networks.

Model Training: Train the chosen classification model on the training data. The model learns the relationships between the input features and the corresponding target labels using a specified learning algorithm. During training, the model adjusts its internal parameters to minimize the prediction errors on the training data.

Model Evaluation: Evaluate the trained model's performance on the validation set to assess its generalization ability and fine-tune hyperparameters. Common evaluation metrics include accuracy, precision, recall, F1-score, and area under the ROC curve (AUC-ROC). Perform cross-validation to obtain more robust performance estimates by training and evaluating the model on multiple different subsets of the data.

Model Tuning: Adjust hyperparameters of the classification algorithm to optimize the model's performance. Hyperparameters may include regularization parameters, learning rates, tree depths, kernel functions, or the number of hidden layers in a neural network. Grid search, random search, or Bayesian optimization can be used to find the optimal combination of hyperparameters**.**

Model Deployment: Once the model is trained, evaluate its final performance on the test set, which represents unseen data. Assess the model's accuracy and generalization ability to make predictions on new, real-world data. Deploy the trained model to make predictions on new instances in production, using it to classify new data according to the learned patterns.

4. **Go through the SVM model in depth using various scenarios.**

Solution - Support Vector Machines (SVM) is a powerful machine learning algorithm used for both classification and regression tasks.

Suppose we have a binary classification problem with linearly separable data points. In this scenario, SVM aims to find a hyperplane that separates the two classes with the largest margin. The steps involved are:

Data Preprocessing: Collect and preprocess the data, ensuring it is labeled and suitable for the classification task.

Feature Scaling: Normalize or standardize the features to ensure they are on a similar scale, which can help SVM converge faster.

Training the Model: Select the appropriate SVM variant: C-SVM or nu-SVM. Define the kernel function: Linear, Polynomial, Gaussian (RBF), or others, based on the characteristics of the data. Determine the regularization parameter C or nu, which controls the trade-off between achieving a

larger margin and minimizing misclassifications. Train the SVM model on the training data, optimizing the margin and misclassification errors.

Model Evaluation and Tuning: Evaluate the model's performance on a validation set or through cross-validation. Adjust hyperparameters like C, kernel parameters, or kernel type based on the performance metrics (e.g., accuracy, precision, recall) and validation results. Iterate on the training and evaluation steps until satisfactory performance is achieved.

Model Deployment: Test the final trained model on a separate test set to assess its performance on unseen data. Deploy the SVM model to make predictions on new instances, classifying them based on the learned hyperplane and decision boundaries**.**

5. **What are some of the benefits and drawbacks of SVM?**

Solution - Advantages of SVM Classifier:

- SVM works relatively well when there is a clear margin of separation between classes.
- SVM is more effective in high dimensional spaces and is relatively memory efficient.
- SVM is effective in cases where the dimensions are greater than the number of samples.

The Disadvantages of Support Vector Machine (SVM) are:

- Unsuitable to Large Datasets.
- Large training time.
- More features, more complexities.
- Bad performance on high noise.
- Does not determine Local optima.

6. **Go over the kNN model in depth.**

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

The K-NN working can be explained on the basis of the below algorithm:

- Step-1: Select the number K of the neighbors
- Step-2: Calculate the Euclidean distance of K number of neighbors
- Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.
- Step-4: Among these k neighbors, count the number of the data points in each category.
- Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.
- Step-6: Our model is ready.

7. **Discuss the kNN algorithm's error rate and validation error.**

Solution - Error rate initially decreases and reaches a minima. After the minima point, it then increase with increasing K. To get the optimal value of K, you can segregate the training and validation from the initial dataset

8. **For kNN, talk about how to measure the difference between the test and training results.**

Solution - The test data is the data we use to evaluate a model. For KNN the train data is the data that get's used to vote on the class label of a new data point (KNN doesn't really involve any training).

9. **Create the kNN algorithm.**

Solution – https://github.com/abhaykeni/Python-Assignments/blob/main/ML%20Assignments/KNNClassifier.ipynb

What is a decision tree, exactly? What are the various kinds of nodes? Explain all in depth.

Solution - A decision tree is a type of supervised machine learning used to categorize or make predictions based on how a previous set of questions were answered. The model is a form of supervised learning, meaning that the model is trained and tested on a set of data that contains the desired categorization.

There are three different types of nodes: chance nodes, decision nodes, and end nodes. A chance node shows the probabilities of certain results. A decision node shows a decision to be made, and an end node shows the outcome of a decision path.

10. **Describe the different ways to scan a decision tree.**

Solution – When scanning or traversing a decision tree, there are two commonly used methods: Depth-First Search (DFS) and Breadth-First Search (BFS). Each method offers a different approach to explore the nodes of the decision tree.

Depth-First Search (DFS):

In DFS, the tree is traversed in a depth-first manner, meaning it explores the tree by going as deep as possible before backtracking. There are three commonly used strategies within DFS:

Pre-order traversal: Visits the current node, then recursively visits the left subtree, and finally recursively visits the right subtree.

In-order traversal: Recursively visits the left subtree, visits the current node, and then recursively visits the right subtree. This strategy is commonly used for binary search trees to retrieve values in sorted order.

Post-order traversal: Recursively visits the left subtree, then recursively visits the right subtree, and finally visits the current node.

DFS is typically implemented using recursion or a stack data structure to keep track of nodes to be explored.

Breadth-First Search (BFS):

In BFS, the tree is traversed level by level, exploring the nodes at each level before moving to the next level. BFS uses a queue data structure to keep track of the nodes to be visited in a first-in-first-out (FIFO) manner.

Starting from the root node, the algorithm visits each node at the current level, enqueues their child nodes, and continues until all nodes have been visited. BFS is useful when you want to explore the tree level by level, such as finding the shortest path or searching for a node at a specific depth.

11. Describe in depth the decision tree algorithm.

Solution - In a decision tree, for predicting the class of the given dataset, the algorithm starts from the root node of the tree. This algorithm compares the values of root attribute with the record (real dataset) attribute and, based on the comparison, follows the branch and jumps to the next node.

For the next node, the algorithm again compares the attribute value with the other sub-nodes and move further. It continues the process until it reaches the leaf node of the tree. The complete process can be better understood using the below algorithm:

- Step-1: Begin the tree with the root node, says S, which contains the complete dataset.
- Step-2: Find the best attribute in the dataset using Attribute Selection Measure (ASM).
- Step-3: Divide the S into subsets that contains possible values for the best attributes.
- Step-4: Generate the decision tree node, which contains the best attribute.
- Step-5: Recursively make new decision trees using the subsets of the dataset created in step -3. Continue this process until a stage is reached where you cannot further classify the nodes and called the final node as a leaf node.

12. In a decision tree, what is inductive bias? What would you do to stop overfitting?

Solution -Before learning a model given a data and a learning algorithm, there are a few assumptions a learner makes about the algorithm. These assumptions are called the inductive bias. It is like the property of the algorithm.

Pruning refers to a technique to remove the parts of the decision tree to prevent growing to its full depth. By tuning the hyperparameters of the decision tree model one can prune the trees and prevent them from overfitting. There are two types of pruning Pre-pruning and Post-pruning.

14.Explain advantages and disadvantages of using a decision tree?

Solution –Some advantages of decision trees are:

- Simple to understand and to interpret.
- Requires little data preparation.
- The cost of using the tree (i.e., predicting data) is logarithmic in the number of data points used to train the tree.
- Able to handle both numerical and categorical data.
- Able to handle multi-output problems.

Disadvantages

- Overfitting is one of the practical difficulties for decision tree models. ...
- Decision trees cannot be used well with continuous numerical variables.
- A small change in the data tends to cause a big difference in the tree structure, which causes instability.

15. Describe in depth the problems that are suitable for decision tree learning.

Solution - Decision tree learning is suitable for a wide range of classification and regression problems. Here are some types of problems where decision tree learning is commonly used:

Binary Classification: Decision trees can effectively handle binary classification problems, where the goal is to assign instances to one of two classes. Examples include predicting whether a customer will churn or not, determining if an email is spam or not, or classifying a tumor as malignant or benign.

Multiclass Classification: Decision trees can be extended to handle multiclass classification problems, where the goal is to classify instances into more than two classes. Examples include classifying images into different categories, predicting the type of crop based on various attributes, or identifying different species of plants or animals.

Regression: Decision trees can also be used for regression tasks, where the goal is to predict a continuous or numerical value. Examples include predicting the price of a house based on its features, estimating the sales volume based on marketing expenditure, or forecasting the temperature based on weather conditions.

Feature Selection: Decision trees can help identify the most relevant features for a given problem. By analyzing the structure of the tree and the importance of different features, decision trees can assist in feature selection and dimensionality reduction tasks.

Interpretability and Explainability: Decision trees are inherently interpretable models, making them suitable for situations where interpretability and explainability are important. The decision paths and splits in the tree provide insights into the decision-making process, allowing users to understand why certain predictions are made.

Handling Nonlinear Relationships: Decision trees can capture nonlinear relationships between features and target variables. By recursively partitioning the feature space, decision trees can model complex decision boundaries, making them suitable for problems where linear models may not perform well.

Handling Missing Values: Decision trees can handle missing values in the data without requiring imputation. They can make decisions based on the available features at each node, allowing for robustness against missing data.

16. Describe in depth the random forest model. What distinguishes a random forest?

Solution – Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model**.**

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

The Working process can be explained in the below steps and diagram:

Step-1: Select random K data points from the training set.

Step-2: Build the decision trees associated with the selected data points (Subsets).

Step-3: Choose the number N for decision trees that you want to build.

Step-4: Repeat Step 1 & 2.

Step-5: For new data points, find the predictions of each decision tree, and assign the new data points to the category that wins the majority votes.

Random forest uses bootstrap replicas, that is to say, it subsamples the input data with replacement, whereas Extra Trees use the whole original sample

17. In a random forest, talk about OOB error and variable value.

Solution –

The out-of-bag (OOB) error is the average error for each calculated using predictions from the trees that do not contain in their respective bootstrap sample. This allows the RandomForestClassifier to be fit and validated whilst being trained.

The default method to compute variable importance is the mean decrease in impurity (or gini importance) mechanism: At each split in each tree, the improvement in the split-criterion is the importance measure attributed to the splitting variable, and is accumulated over all the trees in the forest separately for each variable