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

Answer:--->Supervised learning involves training a machine learning model on labeled data, where the correct output is provided for each example in the training set. The goal is to make predictions on new, unseen examples that are drawn from the same distribution as the training set. Common applications of supervised learning include image and speech recognition, natural language processing, and regression and classification tasks.

In semi-supervised learning, the training set contains a mix of labeled and unlabeled examples. The goal is still to make predictions on new, unseen examples, but the model can make use of the partial supervision provided by the labeled examples to improve its accuracy. Semi-supervised learning can be useful when it is expensive or time-consuming to label a large dataset, or when a small number of labeled examples can provide a sufficient signal to learn from.

Unsupervised learning involves training a machine learning model on a dataset where the correct output is not provided. The goal is to discover patterns or relationships in the data that can be used to represent the data in a more compact or meaningful way. Common applications of unsupervised learning include clustering, dimensionality reduction, and density estimation. Unsupervised learning can be useful for exploratory data analysis and for preprocessing raw data before applying supervised or semi-supervised learning methods.

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

Answer:-->Here are a few interesting examples to illustrate the widespread application of prediction algorithms. 1 - Email Spam. ... 2 - Handwritten Digit Recognition. ... 3 - Image segmentation. ... 4 - Speech Recognition. ... 5 - DNA Expression Microarray. ... 6 - DNA Sequence Classification.

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

Answer:--->Generally the biomedical data classification process can be divided into four phases, namely (1) data acquisition and segmentation, (2) data preprocessing, (3) feature extraction/dimension reduction, and (4) recognition and classification. Classification is the problem of identifying to which of a set of categories (subpopulations), a new observation belongs to, on the basis of a training set of data containing observations and whose categories membership is known.

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

Answer:--->Given a pair of (x,y) coordinates, we want a classifier that outputs either yellow or blue. We plot the labeled training data on a plane: An SVM takes these data points and outputs the hyperplane, which is simply a line in two-dimension, that best separates the tags. The line is the decision boundary.Support Vector Machine(SVM) is a supervised machine learning algorithm used for both classification and regression. Though we say regression problems as well its best suited for classification. The objective of SVM algorithm is to find a hyperplane in an N-dimensional space that distinctly classifies the data points.

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

Answer:--->The advantages of SVM and support vector regression include that they can be used to avoid the difficulties of using linear functions in the high-dimensional feature space, and the optimization problem is transformed into dual convex quadratic programs.It does not execute very well when the data set has more sound i.e. target classes are overlapping. In cases where the number of properties for each data point

outstrips the number of training data specimens, the support vector machine will underperform.SVM's are very good when we have no idea on the data. Works well with even unstructured and semi structured data like text, Images and trees. The kernel trick is real strength of SVM. With an appropriate kernel function, we can solve any complex problem.

6. Go over the kNN model in depth.

Answer:-->The abbreviation KNN stands for “K-Nearest Neighbour”. It is a supervised machine learning algorithm. The algorithm can be used to solve both classification and regression problem statements.

The number of nearest neighbours to a new unknown variable that has to be predicted or classified is denoted by the symbol ‘K’.

Let’s take a good look at a related real-world scenario before we get started with this awesome algorithm.

We are often notified that you share many characteristics with your nearest peers, whether it be your thinking process, working etiquettes, philosophies, or other factors. As a result, we build friendships with people we deem similar to us.

The KNN algorithm employs the same principle. Its aim is to locate all of the closest neighbours around a new unknown data point in order to figure out what class it belongs to. It’s a distance-based approach.

Loading Image Learn | Write | Earn Participate and become a part of 800+ data science authors Consider the diagram below; it is straightforward and easy for humans to identify it as a “Cat” based on its closest allies. This operation, however, cannot be performed directly by the algorithm.

KNN calculates the distance from all points in the proximity of the unknown data and filters out the ones with the shortest distances to it. As a result, it’s often referred to as a distance-based algorithm.

In order to correctly classify the results, we must first determine the value of K (Number of Nearest Neighbours).

In the following diagram, the value of K is 5. Since there are four cats and just one dog in the proximity of the five closest neighbours, the algorithm would predict that it is a cat based on the proximity of the five closest neighbors in the red circle’s boundaries.

KNN cat or dog classification Src: https://images.app.goo.gl/Lpd2apX1sf6DcQzW9

Here, ‘K’ is the hyperparameter for KNN. For proper classification/prediction, the value of K must be fine-tuned.

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

Answer:-->Hence, 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. Now plot the validation error curve to get the optimal value of K.Here we can see that that after around K>23 the error rate just tends to hover around 0.06-0.05 Let's retrain the model with that and check the classification report!1.Training error Rate 2.Validation Error Rate. If we observe the training error rate graph it can be seen that error increases for increasing value of K,also error is zero for K=1. This is because the closest point to any training data point is itself. Hence the prediction is always accurate with K=1

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

Answer:-->KNN is one of the simplest forms of machine learning algorithms mostly used for classification. It classifies the data point on how its neighbor is classified. KNN classifies the new data points based on the similarity measure of the earlier stored data points. For example, if we have a dataset of tomatoes and bananas.The k-NN algorithm does more computation on test time rather than train time. That is absolutely true. The idea of the kNN algorithm is to find a k-long list of samples that are close to a sample we want to classify. KNN is widely known as an ML algorithm that doesn't need any training on data. This is much different from eager learning approaches that rely on a training dataset to perform predictions on unseen data. With KNN, you don't need a training phase at all.

9. Create the kNN algorithm.

Answer:-->The k-nearest neighbor algorithm is imported from the scikit-learn package. Create feature and target variables. Split data into training and test data. Generate a k-NN model using neighbors value. Train or fit the data into the model. Predict the future.

First, import the KNeighborsClassifier module and create KNN classifier object by passing argument number of neighbors in KNeighborsClassifier() function. Then, fit your model on the train set using fit() and perform prediction on the test set using predict().

K-Nearest Neighbors Algorithm. The k-nearest neighbors algorithm, also known as KNN or k-NN, is a non-parametric, supervised learning classifier, which uses proximity to make classifications or predictions about the grouping of an individual data point.

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

Answer:--->A decision tree typically starts with a single node, which branches into possible outcomes. Each of those outcomes leads to additional nodes, which branch off into other possibilities. This gives it a treelike shape. There are three different types of nodes: chance nodes, decision nodes, and end nodes. A decision tree is a non-parametric supervised learning algorithm, which is utilized for both classification and regression tasks. It has a hierarchical, tree structure, which consists of a root node, branches, internal nodes and leaf nodes.Decision tree analysis involves visually outlining the potential outcomes, costs, and consequences of a complex decision. These trees are particularly helpful for analyzing quantitative data and making a decision based on numbers. A decision tree typically starts with a single node, which branches into possible outcomes. Each of those outcomes leads to additional nodes, which branch off into other possibilities. This gives it a treelike shape. There are three different types of nodes: chance nodes, decision nodes, and end nodes.

11. Describe the different ways to scan a decision tree.

Answer:--->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. Decision trees can be divided into two types; categorical variable and continuous variable decision trees. A decision tree is a very specific type of probability tree that enables you to make a decision about some kind of process. For example, you might want to choose between manufacturing item A or item B, or investing in choice 1, choice 2, or choice 3

12. Describe in depth the decision tree algorithm.

Answer:---->The tree depth is an INTEGER value. Maximum tree depth is a limit to stop further splitting of nodes when the specified tree depth has been reached during the building of the initial decision tree In general, the deeper you allow your tree to grow, the more complex your model will become because you will have more splits and it captures more information about the data and this is one of the root causes of overfitting in decision trees because your model will fit perfectly for the training data and will not be Decision Tree Decision Tree is one of the popular and most widely used Machine Learning Algorithms because of its robustness to noise, tolerance against missing information, handling of irrelevant, redundant predictive attribute values, low computational cost, interpretability, fast run time and robust predictors. I know, that's a lot 😂. But a common question I get asked from students is how to tune a Decision Tree. What should be the range of values I should try for the maximum depth, what should be the minimum number of samples required at a leaf node? These are very good questions that don't have a straightforward answer but what we can do is understand how changing one will affect your model. Like what does increasing the maximum depth really mean, what does changing the minimum sample leaves do to your model. So, in this article, I attempt to give you an introduction to these parameters and how they affect your model architecture and what it can mean to your model in general.

Let's look into Scikit-learn's decision tree implementation and let me explain what each of these hyperparameters is and how it can affect your model. Btw note that I assume you have a basic understanding of decision trees.

Since the decision tree is primarily a classification model, we will be looking into the decision tree classifier.

DecisionTreeClassifier criterion: string, optional (default="gini"):

The function to measure the quality of a split. Supported criteria are “gini” for the Gini impurity and “entropy” for the information gain.

If you ever wondered how decision tree nodes are split, it is by using impurity. Impurity is a measure of the homogeneity of the labels on a node. There are many ways to implement the impurity measure, two of which scikit-learn has implemented is the Information gain and Gini Impurity or Gini Index.

Information gain uses the entropy measure as the impurity measure and splits a node such that it gives the most amount of information gain. Whereas Gini Impurity measures the divergences between the probability distributions of the target attribute’s values and splits a node such that it gives the least amount of impurity.

According to the paper “Theoretical comparison between the Gini Index and Information Gain criteria” [3], the frequency of agreement/disagreement of the Gini Index and the Information Gain was only 2% of all cases, so for all intents and purposes you can pretty much use either, but the only difference is entropy might be a little slower to compute because it requires you to compute a logarithmic function:

Many of the researchers point out that in most of the cases, the choice of splitting criteria will not make much difference in the tree performance. Each criterion is superior in some cases and inferior in others, as the “No Free Lunch” theorem suggests.

splitter: string, optional (default=”best”)

The strategy used to choose the split at each node. Supported strategies are “best” to choose the best split and “random” to choose the best random split.

According to scikit-learn’s “best” and “random” implementation [4], both the “best” splitter and the “random” splitter uses Fisher-Yates-based algorithm to compute a permutation of the features array. You don’t really need to worry about the algorithm, the only difference is, in the “best” splitter it evaluate all splits using the criterion before splitting whereas the “random” splitter uses a random uniform function with min_feature_value, max_feature_value and random_state as inputs. We will look into what these are below but for now, let’s see how the splitter will affect the model.

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

Answer:--->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.increased test set error. Unlike other regression models, decision tree doesn't use regularization to fight against over-fitting. Instead, it employs tree pruning.Two approaches to avoiding overfitting are distinguished: pre-pruning (generating a tree with fewer branches than would otherwise be the case) and post-pruning (generating a tree in full and then removing parts of it). Results are given for pre-pruning using either a size or a maximum depth cutoff

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

Answer:--->They are very fast and efficient compared to KNN and other classification algorithms. Easy to understand, interpret, visualize. The data type of decision tree can handle any type of data whether it is numerical or categorical, or boolean. Normalization is not required in the Decision Tree. A significant advantage of a decision tree is that it forces the consideration of all possible outcomes of a decision and traces each path to a conclusion. It creates a comprehensive analysis of the consequences along each branch and identifies decision nodes that need further analysis.

Disadvantages of Decision Trees

One of the limitations of decision trees is that they are largely unstable compared to other decision predictors. A small change in the data can result in a major change in the structure of the decision tree, which can convey a different result from what users will get in a normal event

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

Answer:-->They can be used to solve both regression and classification problems. Decision tree uses the tree representation to solve the problem in which each leaf node corresponds to a class label and attributes are represented on the internal node of the tree.Decision trees are extremely useful for data analytics and machine learning because they break down complex data into more manageable parts. They're often used in these fields for prediction analysis, data classification, and regression.A small change in the data can result in a major change in the

structure of the decision tree, which can convey a different result from what users will get in a normal event. The resulting change in the outcome can be managed by machine learning algorithms, such as boosting and bagging.

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

Answer:--->The max_depth of a tree in Random Forest is defined as the longest path between the root node and the leaf node: Using the max_depth parameter, I can limit up to what depth I want every tree in my random forest to grow.The fundamental difference is that in Random forests, only a subset of features are selected at random out of the total and the best split feature from the subset is used to split each node in a tree, unlike in bagging where all features are considered for splitting a node. Save this answer.It is called a Random Forest because we use Random subsets of data and features and we end up building a Forest of decision trees (many trees). Random Forest is also a classic example of a bagging approach as we use different subsets of data in each model to make predictions.

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

Answer:--->Most of the features have shown negligible importance - the mean is about 5%, a third of them is of importance 0, a third of them is of importance above the mean. However, perhaps the most striking fact is the oob (out-of-bag) score: a bit less than 1%As noted above, only a subset of DTs is used for determining the OOB score.A prediction made for an observation in the original data set using only base learners not trained on this particular observation is called out-of-bag (OOB) prediction. These predictions are not prone to overfitting, as each prediction is only made by learners that did not use the observation for training.