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

In a supervised learning model, the algorithm learns on a labeled dataset, providing an answer key that the algorithm can use to evaluate its accuracy on training data. An unsupervised model, in contrast, provides unlabeled data that the algorithm tries to make sense of by extracting features and patterns on its own.

Semi-supervised learning takes a middle ground. It uses a small amount of labeled data bolstering a larger set of unlabeled data. And reinforcement learning trains an algorithm with a reward system, providing feedback when an artificial intelligence agent performs the best action in a particular situation.

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

* **Customer behavior prediction**: Customers can be classified into different categories based on their buying patterns, web store browsing patterns etc. For example, classification models can be used to determine whether a customer is likely to purchase more items or not. If the classification model predicts a greater likelihood that they are about to make more purchases, then you might want to send them promotional offers and discounts accordingly. Or if it has been determined that they will probably fall off of their purchasing habits soon, maybe save them for later by making their information readily available.
* **Document classification**: A multinomial classification model can be trained to classify documents in different categories.
* **Image classification**: A multinomial classification model can be trained to classify images into different categories. For example, in order to classify images of dogs and cats for use within machine vision systems, machine learning techniques can help automate this process based on pre-classified images of dogs and cats.rent categories
* **Web text classification**: Classifies web text or assign tag to web text based on pre-determined categories learned from the past data. For example, classification models can be used to automatically classify web text into one of the following categories: Sports, Entertainment, or Technology.
* **Ad click-through rate prediction**: Binary classification models can be used to predict whether one or more ads on the website will be clicked or not. Such models are used to optimize the ad inventory on websites by selecting which ads will have a better chance of being clicked. A machine learning classification model can be built using historical data about what types of users do or don’t click on certain ads, along with information like demographics and content within each web page where an ad appears; then it is used to predict the chances that a user will click on an ad.
* **Product categorization**: A multinomial classification can be used to categorize the products sold by different retailers in the same categories irrespective of categories assigned to the product by the respective retailers. This use case is relevant for eCommerce aggregators.
* **Malware classification**: A multinomial classification can be used to classify the new/emerging malwares on the basis of comparable features of similar malware. Malware classification is very useful for security experts to take appropriate actions for combating/preventing malware. Machine learning classification algorithms such as Naïve Bayes, k-NN and tree-based models can be used for malware classification.
* **Image sentiment analysis**: Machine learning binary classification models can be built based on machine learning algorithms to classify whether the image contains a positive or negative emotion/sentiment or not. This use case is relevant in the field of social media analytics where machine learning techniques are applied to understand users’ opinions and sentiments on different topics.
* **Customer churn prediction**: A binary classification model can be used to classify whether a customer will churn or not in the near future. The application of the customer churn classification model can be found in different business scenarios like up-selling/cross-selling to existing customers, identifying at-risk accounts in the customer base, etc. More commonly, telecommunications companies have been found to use machine learning classification models for churn prediction.
* **Customer behavior assessment for promotional offers**: A binary classification model can be used to classify whether an account is customer-friendly or not in the context of a specific business scenario like upselling, cross-selling etc. For example, based on past data about how customers respond to certain types of offers; machine learning techniques can be used to predict whether a given customer will respond positively or negatively to the offer.
* **Anomaly detection problems such as fraud detection:** Anomaly detection models can be built using machine learning classification algorithms like Naïve Bayes, k-NN etc. The application of these machine learning anomaly detection models is very wide and includes use cases such as finding unusual patterns in financial transactions that may indicate fraud, finding machine problems by detecting unusual machine readings, and monitoring machine parameters to detect abnormalities.
* **Credit card fraud detection**: A binary classification model can be used for credit card fraud detection where the historical transactions data of a customer is analyzed using machine learning algorithms like Naïve Bayes, k-NN etc. Based on past fraudulent or non-fraudulent transaction data and machine learning classification models, it can be predicted whether the given credit card will result in fraudulent transactions or not
* **Deduction validation classification**: A binary classification model can be used to classify whether a deduction claimed by the buyer on a given invoice is a valid or invalid deduction. This would be useful in account receivables to classify whether the given invoice will be paid in full or partial based on deduction validation classification
* **Credit-worthiness assessment**: A machine learning classification model can be trained to predict the probability of default for a customer based on past transaction data and historical information about customers who have defaulted/not defaulted in their payments. Credit card companies, financial institutions like banks, etc
* **Blocked order release recommendation**: A binary classification model can be built to classify whether an order placed by the customer should be blocked or not based on the buyer credit exposure. This use case is very prevalent in account receivables where machine learning classification models are used to predict whether a given order should be blocked or not. This would help the business save costs by identifying high-risk customers.
* **Sentiment analysis**: A machine learning binary classification model can be trained to identify the sentiment (positive/negative) of a given text document based on classification algorithms like Naïve Bayes, SVM etc. This would help determine whether the sentiment expressed in a document such as an email is positive or negative for business purposes like identifying whether a customer is satisfied or dissatisfied with the service provided.

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

The System Model consists of four main phases: **Pre-processing, segmentation, feature extraction, and classifying location**.

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

* **Identify the right hyper-plane (Scenario-1):**Here, we have three hyper-planes (A, B, and C). Now, identify the right hyper-plane to classify stars and circles.  
  You need to remember a thumb rule to identify the right hyper-plane: “Select the hyper-plane which segregates the two classes better”. In this scenario, hyper-plane “B” has excellently performed this job.
* **Identify the right hyper-plane (Scenario-2):**Here, we have three hyper-planes (A, B, and C) and all are segregating the classes well. Now, How can we identify the right hyper-plane?

Here, maximizing the distances between nearest data point (either class) and hyper-plane will help us to decide the right hyper-plane. This distance is called as **Margin**. Let’s look at the below snapshot:[[](https://www.analyticsvidhya.com/wp-content/uploads/2015/10/SVM_4.png)](https://www.analyticsvidhya.com/wp-content/uploads/2015/10/SVM_4.png)Above, you can see that the margin for hyper-plane C is high as compared to both A and B. Hence, we name the right hyper-plane as C. Another lightning reason for selecting the hyper-plane with higher margin is robustness. If we select a hyper-plane having low margin then there is high chance of miss-classification.

* **Identify the right hyper-plane (Scenario-3):**Hint:Use the rules as discussed in previous section to identify the right hyper-plane

**[](https://www.analyticsvidhya.com/wp-content/uploads/2015/10/SVM_5.png)**Some of you may have selected the hyper-plane **B**as it has higher margin compared to **A.**But, here is the catch, SVM selects the hyper-plane which classifies the classes accurately prior to maximizing margin. Here, hyper-plane B has a classification error and A has classified all correctly. Therefore, the right hyper-plane is **A.**

* **Can we classify two classes (Scenario-4)?:**Below, I am unable to segregate the two classes using a straight line, as one of the stars lies in the territory of other(circle) class as an outlier.  **[](https://www.analyticsvidhya.com/wp-content/uploads/2015/10/SVM_61.png)**As I have already mentioned, one star at other end is like an outlier for star class. The SVM algorithm has a feature to ignore outliers and find the hyper-plane that has the maximum margin. Hence, we can say, SVM classification is robust to outliers.  
  **[](https://www.analyticsvidhya.com/wp-content/uploads/2015/10/SVM_71.png)**
* **Find the hyper-plane to segregate to classes (Scenario-5):**In the scenario below, we can’t have linear hyper-plane between the two classes, so how does SVM classify these two classes? Till now, we have only looked at the linear hyper-plane.**[](https://www.analyticsvidhya.com/wp-content/uploads/2015/10/SVM_8.png)**SVM can solve this problem. Easily! It solves this problem by introducing additional feature. Here, we will add a new feature z=x^2+y^2. Now, let’s plot the data points on axis x and z:  
  [[](https://www.analyticsvidhya.com/wp-content/uploads/2015/10/SVM_9.png)](https://www.analyticsvidhya.com/wp-content/uploads/2015/10/SVM_9.png)In above plot, points to consider are:
  + All values for z would be positive always because z is the squared sum of both x and y
  + In the original plot, red circles appear close to the origin of x and y axes, leading to lower value of z and star relatively away from the origin result to higher value of z.

In the SVM classifier, it is easy to have a linear hyper-plane between these two classes. But, another burning question which arises is, should we need to add this feature manually to have a hyper-plane. No, the SVM  algorithm has a technique called the [**kernel**](https://en.wikipedia.org/wiki/Kernel_method)**trick**. The SVM kernel is a function that takes low dimensional input space and transforms it to a higher dimensional space i.e. it converts not separable problem to separable problem. It is mostly useful in non-linear separation problem. Simply put, it does some extremely complex data transformations, then finds out the process to separate the data based on the labels or outputs you’ve defined.

When we look at the hyper-plane in original input space it looks like a circle:  
[](https://www.analyticsvidhya.com/wp-content/uploads/2015/10/SVM_10.png)

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

* **Pros:**
  + It works really well with a clear margin of separation
  + It is effective in high dimensional spaces.
  + It is effective in cases where the number of dimensions is greater than the number of samples.
  + It uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
* **Cons:**
  + It doesn’t perform well when we have large data set because the required training time is higher
  + It also doesn’t perform very well, when the data set has more noise i.e. target classes are overlapping
  + SVM doesn’t directly provide probability estimates, these are calculated using an expensive five-fold cross-validation. It is included in the related SVC method of Python scikit-learn library.

6. Go over the kNN model in depth.

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.

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

With the training accuracy of **93%** and the test accuracy of 86%, our model might have shown overfitting here. Why so? When the value of K or the number of neighbors is too low, the model picks only the values that are closest to the data sample, thus forming a very complex decision boundary as shown above.

9. Create the kNN algorithm.

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

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

* Decision Tree is a **Supervised learning technique**that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where**internal nodes represent the features of a dataset, branches represent the decision rules** and **each leaf node represents the outcome.**
* In a Decision tree, there are two nodes, which are the **Decision Node** and**Leaf Node.** Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.
* The decisions or the test are performed on the basis of features of the given dataset.
* ***It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.***
* It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.
* In order to build a tree, we use the **CART algorithm,** which stands for **Classification and Regression Tree algorithm.**
* A decision tree simply asks a question, and based on the answer (Yes/No), it further split the tree into subtrees.
*  **Root Node:** Root node is from where the decision tree starts. It represents the entire dataset, which further gets divided into two or more homogeneous sets.
*  **Leaf Node:** Leaf nodes are the final output node, and the tree cannot be segregated further after getting a leaf node.

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

12. Describe in depth the decision tree 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).** Like Information Gain and Entropy.
* **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.

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

* At the beginning, the whole training set is considered as the **root.**
* Feature values are preferred to be categorical. If the values are continuous then they are discretized prior to building the model.
* Records are **distributed recursively** on the basis of attribute values.
* Order to placing attributes as root or internal node of the tree is done by using some statistical approach which are below mention.

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?

Advantages of the Decision Tree

* It is simple to understand as it follows the same process which a human follow while making any decision in real-life.
* It can be very useful for solving decision-related problems.
* It helps to think about all the possible outcomes for a problem.
* There is less requirement of data cleaning compared to other algorithms.

Disadvantages of the Decision Tree

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

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

Decision tree learning is generally best suited to problems with the following characteristics:

* Instances are represented by **attribute-value pairs**.
  + There is a finite list of attributes (e.g. hair colour) and each instance stores a value for that attribute (e.g. blonde).
  + When each attribute has a small number of distinct values (e.g. blonde, brown, red) it is easier for the decision tree to reach a useful solution.
  + The algorithm can be extended to handle real-valued attributes (e.g. a floating point temperature)
* The target function has **discrete output values**.
  + A decision tree classifies each example as one of the output values.
    - Simplest case exists when there are only two possible classes (**Boolean classification**).
    - However, it is easy to extend the decision tree to produce a target function with more than two possible output values.
  + Although it is less common, the algorithm can also be extended to produce a target function with real-valued outputs.
* Disjunctive descriptions may be required.
  + Decision trees naturally represent disjunctive expressions.
* The training data may contain errors.
  + Errors in the classification of examples, or in the attribute values describing those examples are handled well by decision trees, making them a robust learning method.
* The training data may contain missing attribute values.
  + Decision tree methods can be used even when some training examples have unknown values (e.g., humidity is known for only a fraction of the examples).

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

Step 1: In Random forest n number of random records are taken from the data set having k number of records.

Step 2: Individual decision trees are constructed for each sample.

Step 3: Each decision tree will generate an output.

Step 4: Final output is considered based on ***Majority Voting or Averaging***for Classification and regression respectively.

|  |  |
| --- | --- |
| **Decision trees** | **Random Forest** |
| 1. Decision trees normally suffer from the problem of overfitting if it’s allowed to grow without any control. | 1. Random forests are created from subsets of data and the final output is based on average or majority ranking and hence the problem of overfitting is taken care of. |
| 2. A single decision tree is faster in computation. | 2. It is comparatively slower. |
| 3. When a data set with features is taken as input by a decision tree it will formulate some set of rules to do prediction. | 3. Random forest randomly selects observations, builds a decision tree and the average result is taken. It doesn’t use any set of formulas. |

**Important Features of Random Forest**

**1. Diversity-**Not all attributes/variables/features are considered while making an individual tree, each tree is different.

**2. Immune to the curse of dimensionality-** Since each tree does not consider all the features, the feature space is reduced.

**3. Parallelization-**Each tree is created independently out of different data and attributes. This means that we can make full use of the CPU to build random forests.

**4.  Train-Test split-**In a random forest we don’t have to segregate the data for train and test as there will always be 30% of the data which is not seen by the decision tree.

**5.  Stability-**Stability arises because the result is based on majority voting/ averaging.

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

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**.

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