1. What are the steps involved in constructing a Random Forest classifier?

What do you understand by Out-of-Bag (OOB) error?

Ans.

Several decision trees are made for a Random Forest classifier by using various random samples from the data. Every tree is given a subset for learning and predictions come from the common outcomes among most of the trees. It functions similarly to asking several people how they see things rather than counting on the recommendations of one model. The aim is to stop the network from overfitting and makes its predictions more accurate.

It's really practical to use the Out-of-Bag (OOB) error. Since every tree is given a random slice of the data, some data points are never seen by any tree. The data that are not used to build the tree are the OOB samples and we use them to test the tree's performance. Doing this gives us a good idea of how the model will perform on unknown samples.

2. What is k-fold cross-validation?

How does it work and why is it used in model evaluation?

Ans.

We use K-fold cross-validation to see if our model works well for cases it hasn't seen before. There are k parts (or k folds) of data and the model learns from k-1 of them, leaving the last part aside for testing. It is done k times, with the folder used for testing switching each time. The outcome is taken as the average of all the amounts produced in each run. It is helpful since it stops our model from doing well on one set of data but not on others. It explains how the model does overall, making its evaluation more complete.

3. What do underfitting and overfitting mean in machine learning?

Why do they occur, and what is the significance of the bias-variance tradeoff in selecting a good machine learning model?

Ans.

Two big issues that can happen in machine learning are underfitting and overfitting. When the model is too simple, it cannot recognize the real patterns which makes it fail on both the training and test data. In overfitting, the model picks up on all the little details in the training data and fails to give good results on fresh datasets it has not seen.

Because of this, the bias-variance tradeoff applies. Model bias is high when the model does not fit the data well (because it is too simple and this causes underfitting) and model variance is high when the model fits the data better than needed (which leads to overfitting). Trying to balance these two aspects allows the model to learn effectively and apply its learning to fresh examples.

4. Problem:
A patient undergoes a test that checks for 3 possible diseases: A, B, and C. Based on medical data:

- P(A) = 0.1
- P(B) = 0.3
- P(C) = 0.6

The test result comes back positive, and the conditional probabilities are:

- P(Positive | A) = 0.9
- P(Positive | B) = 0.8
- P(Positive | C) = 0.3
  If the test result is positive, what is the probability that the patient has disease A?

Ans:-

We calculate this by taking the probability of each disease * the chance the test is positive if the patient has it:

$P(\text{Positive}) = (0.1 \times 0.9) + (0.3 \times 0.8) + (0.6 \times 0.3)$

$= 0.09 + 0.24 + 0.18$

$= 0.51$

$P(A \mid \text{Positive}) = [\, P(\text{Positive} \mid A) \times P(A) \,] / P(\text{Positive})$

$= (0.9 \times 0.1) / 0.51$

$= 0.09/0.51 \approx 0.176$

So, in spite of a positive result, the chance the patient has disease A is only around 17.6% because disease C is diagnosed much more often and the test tends to mistakenly identify it in many cases.