MACHINE LEARNING-5

1-R-squared is a better measure of goodness of fit in regression compared to Residual Sum of Squares (RSS). R-squared is a proportion (between 0 and 1) that represents the proportion of the variance in the dependent variable that is explained by the independent variables. The higher the R-squared value, the better the model fits the data.

RSS, on the other hand, represents the squared difference between the observed and predicted values of the dependent variable. While RSS can be used to compare different models, it is not intuitive to interpret, and the magnitude of the RSS can be affected by the scale of the dependent variable.

In summary, R-squared provides a clear and easily interpretable summary of how well the model fits the data, and it is widely used in regression analysis.

2-TSS (Total Sum of Squares)

- The term sum of squares refers to a statistical technique used in regression analysis to determine the dispersion of data points. The sum of squares can be used to find the function that best fits by varying the least from the data. In a regression analysis, the goal is to determine how well a data series can be fitted to a function that might help to explain how the data series was generated. The sum of squares can be used in the financial world to determine the variance in asset values.

  The following is the formula for the total sum of squares.

  For a set X of n items: Sum of squares=∑i=0n(Xi−X¯)2

  where: Xi=The ith item in the set

  X=The mean of all items in the set

  (Xi−X) =The deviation of each item from the mean

  Sum of squares=i=0∑n(Xi−X)2

Residual Sum of Squares (RSS)

- The residual sum of squares (RSS) is a statistical technique used to measure the amount of variance in a data set that is not explained by a regression model itself. Instead, it estimates the variance in the residuals, or error term.

  RSS = $\sum^{n}_{i=1} (y^{i} - f(x_{i}))^{2}$

  Where:

  y_i = the ith value of the variable to be predicted

  $f(x_i)$ = predicted value of y_i

  n = upper limit of summation

ESS (Explained Sum of Squares)

- The **explained sum of squares** (**ESS**), alternatively known as the **model sum of squares** or **sum of squares due to regression** (**SSR** – not to be confused with the residual sum of squares (RSS) or sum of squares of errors), is a quantity used in describing how well a model, often a regression model, represents the data being modelled. In particular, the explained sum of squares measures how much variation there is in the modelled values and this is compared to the total sum of squares (TSS), which measures how much variation there is in the observed data, and to the residual sum of squares, which measures the variation in the error between the observed data and modelled values.

Mathematically, it is the sum of the squares of the difference between the predicted data and mean data.

Let $y_i = a + b_1x_{1i} + b_2x_{2i} + ... + \varepsilon_i$ is regression model, where:

y_i is the i th observation of the response variable

x_{ji} is the i th observation of the j th explanatory variable

a and b_i are coefficients

i indexes the observations from 1 to n

ε_i is the i th value of the error term

Then

$$\mathrm{ESS} = \sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2 .$$

This is usually used for regression models. The variation in the modeled values is contrasted with the variation in the observed data (total sum of squares) and variation in modeling errors (residual sum of squares). The result of this comparison is given by ESS as per the following equation:

ESS = total sum of squares – residual sum of squares

3-Regularization is one of the most important concepts of machine learning. It is a technique to prevent the model from overfitting by adding extra information to it.

Sometimes the machine learning model performs well with the training data but does not perform well with the test data. It means the model is not able to predict the output when deals with unseen data by introducing noise in the output, and hence the model is called overfitted. This problem can be deal with the help of a regularization technique.

This technique can be used in such a way that it will allow to maintain all variables or features in the model by reducing the magnitude of the variables. Hence, it maintains accuracy as well as a generalization of the model.

It mainly regularizes or reduces the coefficient of features toward zero. In simple words, "In regularization technique, we reduce the magnitude of the features by keeping the same number of features."

4-Gini Impurity is a measurement used to build Decision Trees to determine how the features of a dataset should split nodes to form the tree. More precisely, the Gini Impurity of a dataset is a number between 0-0.5, which indicates the likelihood of new, random data being misclassified if it were given a random class label according to the class distribution in the dataset.

The formula of the Gini Index is as follows:

$Gini=1-n\sum i=1(pi)2$

where,
‘pi’ is the probability of an object being classified to a particular class.

5-Yes, unregularized decision trees are prone to overfitting.

A decision tree builds a tree-like model of decisions and their consequences, where each internal node represents a test on an attribute, each branch represents the outcome of the test, and each leaf node represents a prediction or decision. When a tree is grown without any restrictions, it can continue to expand until each leaf node represents only a single data point. This leads to a model that fits the training data very well but is not able to generalize well to new data, i.e., it overfits the training data.Therefore, to avoid overfitting in decision trees, various regularization techniques such as pruning, limiting tree depth, or adding penalties for having too many splits or too many leaves can be used.

6-**Ensemble methods** is a machine learning technique that combines several base models in order to produce one optimal predictive model. In learning models, noise, variance, and bias are the major sources of error. The ensemble methods in machine learning help minimize these error-causing factors, thereby ensuring the accuracy and stability of machine learning (ML) algorithms.

With the help of example understanding the Ensemble method.

Imagine a group of blindfolded people playing the touch-and-tell game, where they are asked to touch and explore a mini donut factory that no one of them has ever seen before. Since they are blindfolded, their version of what a mini donut factory looks like will vary, depending on the parts of the appliance they touch. Now, suppose they are personally asked to describe what they touched. In that case, their individual experiences will give a precise description of specific parts of the mini donut factory. Still, collectively, their combined experiences will provide a highly detailed account of the entire equipment.

Similarly, ensemble methods in machine learning employ a set of models and take advantage of the blended output, which, compared to a solitary model, will most certainly be a superior option when it comes to prediction accuracy.

7-Bagging and Boosting are two types of Ensemble Learning. These two decrease the variance of a single estimate as they combine several estimates from different models. So the result may be a model with higher stability.

Bagging

Bagging stands for Bootstrap aggregating, which combines several models for better predictive results. In statistical classification and regression, bagging improves the stability and accuracy of machine learning algorithms by decreasing the variance and reducing the chances of overfitting.

Boosting

Boosting involves building a strong classifier from several weak classifiers using the weak models in series. The first step is to build a model from the training set. Then we create the second model, which tries to correct the error incurred while training the first. This process is continued while adding new models until the maximum number of models is reached, or the training data is finished.

Gradient Boosting or Adaboost is an implementation of boosting. AdaBoost stands for Adaptive Boosting and is a technique that combines multiple weak classifiers into a single strong classifier. Gradient boosting uses the gradient descent algorithm that minimizes any differentiable loss function

Difference Between Bagging and Boosting

Bagging	**Boosting**
The original dataset is divided into multiple subsets, selecting observations with replacement.	The new subset contains the components mis trained by the previous model.
This method combines predictions that belong to the same type.	This method combines predictions that belong to the different types.
Bagging decreases variance.	Boosting decreases bias.
Base classifiers are trained parallelly.	Base classifiers are trained sequentially.
The models are created independently.	The model creation is dependent on the previous ones.

8-Out-of-bag (**OOB**) **error**, also called out-of-bag estimate, is a method of measuring the prediction error of random forests, boosted decision trees, and other machine learning models utilizing bootstrap aggregating (bagging).

OOB (out-of-bag) errors are an estimate of the performance of a random forest classifier or regressor on unseen data.

The OOB error is computed using the samples that were not included in the training of the individual trees. This is different from the error computed using the usual training and validation sets, which are used to tune the hyperparameters of the random forest.

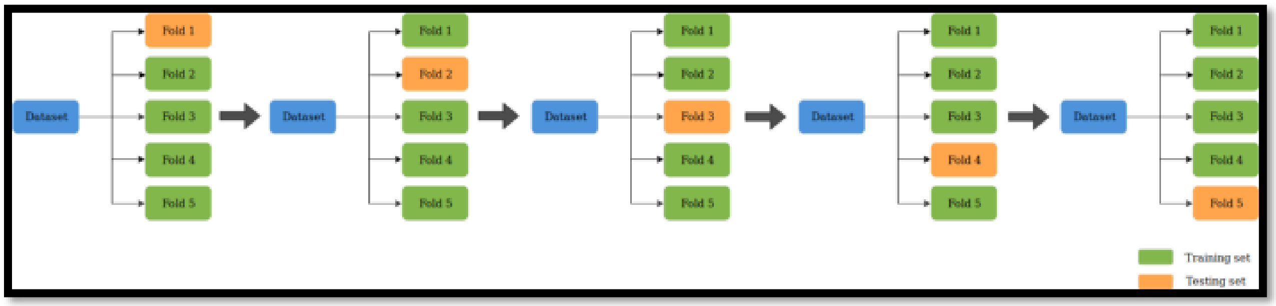
The OOB error can be useful for evaluating the performance of the random forest on unseen data. It is not always a reliable estimate of the generalization error of the model, but it can provide a useful indication of how well the model is performing.

To compute the OOB error, the samples that are not used in the training of an individual tree are known as "out-of-bag" samples. These samples are not used in the training of the tree, but they are used to compute the OOB error for that tree. The OOB error for the entire random forest is computed by averaging the OOB errors of the individual trees.

9-K-Fold CV is where a given data set is split into a ***K*** number of sections/folds where each fold is used as a testing set at some point.

Understanding with the example.

Let's take the scenario of 5-Fold cross validation(K=5). Here, the data set is split into 5 folds. In the first iteration, the first fold is used to test the model and the rest are used to train the model. In the second iteration, 2nd fold is used as the testing set while the rest serve as the training set. This process is repeated until each fold of the 5 folds have been used as the testing set.



10-Hyperparameter tuning is the process of selecting the best set of hyperparameters for a machine learning model. Hyperparameters are parameters that are set before training the model and are not learned from the data during training. Examples of hyperparameters include the learning rate in gradient descent, the number of hidden units in a neural network, or the depth of a decision tree.

Hyperparameter tuning is important because the performance of a machine learning model can be highly sensitive to the choice of hyperparameters. A good set of hyperparameters can result in a model that generalizes well and performs well on unseen data. On the other hand, a poor choice of hyperparameters can result in a model that overfits or underfits the data.

Hyperparameter tuning is usually done through a combination of trial and error, theoretical understanding of the model, and more systematic methods such as grid search, random search, or Bayesian optimization. The goal of hyperparameter tuning is to find the best set of hyperparameters that result in the highest performance on a validation set.

11-Having a large learning rate in gradient descent can result in the following issues:

Oscillation: A large learning rate can cause the model weights to oscillate and not converge to a minimum of the loss function.

Divergence: If the learning rate is too high, the updates to the weights can become too large, and the model may not converge to a minimum, but instead continue to increase, leading to a divergent solution.

Slow convergence: If the learning rate is too high, the model may not converge to the minimum in a reasonable amount of time. On the other hand, if the learning rate is too low, convergence may be slow.

Poor local minima: A large learning rate can cause the model to converge to a poor local minimum instead of the global minimum of the loss function, which would result in a suboptimal solution.

In summary, finding an appropriate learning rate is important for ensuring convergence and finding a good solution in gradient descent. A common approach is to start with a relatively large learning rate and gradually decrease it as the optimization progresses.

12-Logistic regression is well-suited for linearly separable binary classification problems. It models the relationship between the independent variables and the log odds of the dependent binary variable, and the decision boundary is a straight line in the feature space.

However, logistic regression may not be appropriate for non-linear classification problems, as it is limited to modeling linear relationships between the independent variables and the dependent variable. In non-linear problems, a more complex model, such as a support vector machine or a neural network, may be necessary to accurately model the relationship between the independent variables and the dependent variable.

Additionally, logistic regression assumes that the relationship between the independent variables and the dependent variable is the same for all observations. If the relationship is not the same for all observations, the model may not fit the data well and could lead to poor classification performance.

In summary, logistic regression is best suited for linearly separable binary classification problems and may not perform well for non-linear classification problems.

13-Gradient Boosting is the boosting algorithm that works on the principle of the stage wise addition method, where multiple weak learning algorithms are trained and a strong learner algorithm is used as a final model from the addition of multiple weak learning algorithms trained on the same dataset.

AdaBoost is a boosting algorithm, which also works on the principle of the stage wise addition method where multiple weak learners are used for getting strong learners.

Gradient boosting	**Adaptive Boosting**
This approach trains learners based upon minimizing the loss function of a learner (i.e., training on the residuals of the model)	This method focuses on training upon misclassified observations. Alters the distribution of the training dataset to increase weights on sample observations that are difficult to classify.
Weak learners are decision trees constructed in a greedy manner with split points based on purity scores (i.e., Gini, minimise loss). Thus, larger trees can be used with around 4 to 8 levels. Learners should still remain weak and so they should be constrained (i.e., the maximum number of layers, nodes, splits, leaf nodes)	The weak learners incase of adaptive boosting are a very basic form of decision tree known as stumps.
All the learners have equal weights in the case of gradient boosting. The weight is usually set as the learning rate which is small in magnitude.	The final prediction is based on a majority vote of the weak learners' predictions weighted by their individual accuracy

14-Whenever we discuss model prediction, it's important to understand prediction errors (bias and variance). There is a tradeoff between a model's ability to minimize bias and variance

What is bias?

Bias is the difference between the average prediction of our model and the correct value which we are trying to predict. Model with high bias pays very little attention to the training data and oversimplifies the model. It always leads to high error on training and test data.

What is variance?

Variance is the variability of model prediction for a given data point or a value which tells us spread of our data. Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn't seen before. As a result, such models perform very well on training data but has high error rates on test data.

Bias Variance Tradeoff

If our model is too simple and has very few parameters then it may have high bias and low variance. On the other hand if our model has large number of parameters then it's going to have high variance and low bias. So we need to find the right/good balance without overfitting and underfitting the data.

This tradeoff in complexity is why there is a tradeoff between bias and variance. An algorithm can't be more complex and less complex at the same time.

15-**Linear Kernel**

It is the most basic type of kernel, usually one dimensional in nature. It proves to be the best function when there are lots of features. The linear kernel is mostly preferred for text-classification problems as most of these kinds of classification problems can be linearly separated.

Linear kernel functions are faster than other functions.

Linear Kernel Formula

F(x, xj) = sum(x.xj)

Here, x, xj represents the data you're trying to classify.

Polynomial Kernel

It is a more generalized representation of the linear kernel. It is not as preferred as other kernel functions as it is less efficient and accurate.

Polynomial Kernel Formula

F (x, xj) = (x. xj+1) ^d

Here '.' shows the dot product of both the values, and d denotes the degree.

F(x, xj) representing the decision boundary to separate the given classes.

Gaussian Radial Basis Function (RBF)

It is one of the most preferred and used kernel functions in svm. It is usually chosen for non-linear data. It helps to make proper separation when there is no prior knowledge of data.

Gaussian Radial Basis Formula

F(x, xj) = exp(-gamma * ||x - xj||^2)

The value of gamma varies from 0 to 1. You have to manually provide the value of gamma in the code. The most preferred value for gamma is 0.1.