1. R-squared or Residual Sum of Squares (RSS) which one of these two is a better measure of goodness of fit model in regression and why?

ANS: 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 terms. The sum of squares is used as a mathematical way to find the function that best fits (varies least) from the data.

- The RSS measures the amount of error remaining between the regression function and the data set after the model has been run. A smaller RSS figure represents a regression function that is well-fit to the data.
- The residual sum of squares (RSS) measures the level of variance in the error term, or residuals, of a regression model.
- The smaller the residual sum of squares, the better your model fits your data; the greater the residual sum of squares, the poorer your model fits your data.
- A value of zero means your model is a perfect fit.

R-squared :-

R-squared is a measure of how well a linear regression model "fits" a dataset. Also commonly called the coefficient of determination, R-squared is the proportion of the variance in the response variable that can be explained by the predictor variable. The value for R-squared can range from 0 to 1. A value of 0 indicates that the response variable cannot be explained by the predictor variable at all. A value of 1 indicates that the response variable can be perfectly explained without error by the predictor variable.

2. What are TSS (Total Sum of Squares), ESS (Explained Sum of Squares) and RSS (Residual Sum of Squares) in regression. Also mention the equation relating these three metrics with each other.

ANS: TSS (Total Sum of Squares): In statistical data analysis the total sum of squares (TSS or SST) is a quantity that appears as part of a standard way of presenting results of such analyses. It is defined as being the sum, over all observations, of the squared differences of each observation from the overall mean.

Total Sum of Squares is defined and given by the following function:

Formula:

$\text{Sum of Squares} = \sum(\text{Sum of Squares} = \sum(x_i - \bar{x})^2$

Where –

x_i = frequency , $\bar{x}$ = mean

ESS (Explained Sum of Squares): Explained sum of square (ESS) or Regression sum of squares or Model sum of squares is a statistical quantity used in modeling of a process This is usually used for regression models.This is usually used for regression models.

ESS gives an estimate of how well a model explains the observed data for the process. It tells how much of the variation between observed data and predicted data is being explained by the model proposed. 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

The residual sum of squares (RSS):

The residual sum of squares (RSS) calculates the degree of variance in a regression model. It estimates the level of error in the model's prediction. The smaller the residual sum of squares, the better your model fits your data; the larger the residual sum of squares, the worse. It is the sum of squares of the observed data minus the predicted data.

Formula:

$$residual\ sum\ of\ squares = \sum_{i=1}^{n}(y_i - \hat{y_i})^2$$

3. What is the need of regularization in machine learning?

ANS: Regularization: Regularization is one of the most important concepts of machine learning. It is a technique to prevent the model from over fitting 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."

Regularization Techniques: There are mainly two types of regularization techniques, which are given below:

- **Ridge Regression**
- **Lasso Regression**

Ridge Regression: Ridge regression is one of the types of linear regression in which a small amount of bias is introduced so that we can get better long-term predictions. Ridge regression is a regularization technique, which is used to reduce the complexity of the model. It is also called as **L2 regularization**.

Lasso Regression: Lasso regression is another regularization technique to reduce the complexity of the model. It stands for **Least Absolute and Selection Operator.** It is similar to the Ridge Regression except that the penalty term contains only the absolute weights instead of a square of weights. Since it takes absolute values, hence, it can shrink the slope to 0, whereas Ridge Regression can only shrink it near to 0.It is also called as **L1 regularization**

4. What is Gini–impurity index?

ANS: Gini Impurity tells us what is the probability of misclassifying an observation.The Gini impurity measure is one of the methods used in decision tree algorithms to decide the optimal split from a root node, and subsequent splits.

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.

Let Ginx represent the gini index.

$$Ginx = \bar{p}_1 \cdot (1 - p_1) + p_2 \cdot (1 - p_2)$$
$$equivalently,$$
$$Ginx = 2 \cdot p_1 p_2$$

Where p1, p2 are class 1 , 2 probabilities, respectively. The equation above will give us the gini impurity measure for a sub split

Note: p1 + p2 =1

5. Are unregularized decision-trees prone to overfitting? If yes, why?

ANS: overfitting: When a model learns the information and noise in the training to the point where it degrades the model's performance on fresh data, this is known as overfitting data. A model that overfits the training data is referred to as overfitting.

Yes unregularized decision-trees prone to overfitting,Overfitting happens when any learning processing overly optimizes training set error at the cost test error. While it's possible for training and testing to perform equality well in cross validation, it could be as the result of the data being very close in characteristics, which may not be a huge problem. In the case of decision tree's they can learn a training set to a point of high granularity that makes them easily overfit. Allowing a decision tree to split to a granular degree, is the behavior of this model that makes it prone to learning every point extremely well to the point of perfect classification ie: overfitting.

There are the following steps to avoid overfitting:

1) Use a test set that is not exactly like the training set, or different enough that error rates are going to be easy to see.

2) Ensure you have enough data.

3) Reduce the complexity of the decision tree model.

4) Use decision trees in an ensemble.

5) Reduce the dimensionality of your data.

6. What is an ensemble technique in machine learning?

ANS: Ensemble methods are techniques that create multiple models and then combine them to produce improved results. Ensemble methods usually produces more accurate solutions than a single model would. This has been the case in a number of machine learning competitions, where the winning solutions used ensemble methods**.**

The ensemble methods in machine learning combine the insights obtained from multiple learning models to facilitate accurate and improved decisions. These methods follow the same principle as the example of buying an air-conditioner cited above.

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.

Ensemble Techniques:

Here is a list of ensemble learning techniques, starting with basic ensemble methods and then moving on to more advanced approaches.

Simple Ensemble Methods:

Mode: In statistical terminology, "mode" is the number or value that most often appears in a dataset of numbers or values. In this ensemble technique, machine learning professionals use a number of models for making predictions about each data point. The predictions made by different models are taken as separate votes.

The Mean/Average: In the mean/average ensemble technique, data analysts take the average predictions made by all models into account when making the ultimate prediction.

Advanced Ensemble Methods:

Bagging (Bootstrap Aggregating): The primary goal of "bagging" or "bootstrap aggregating" ensemble method is to minimize variance errors in decision trees. The objective here is to randomly create samples of training datasets with replacement (subsets of the training data). The subsets are then used for training decision trees or models.

Boosting: An iterative ensemble technique, "boosting," adjusts an observation's weight based on its last classification.In the boosting ensemble method, data scientists train the first boosting algorithm on an entire dataset and then build subsequent algorithms by fitting residuals from the first boosting algorithm, thereby giving more weight to observations that the previous model predicted inaccurately.

7. What is the difference between Bagging and Boosting techniques?

ANS: Bagging:

Bagging is a learning technique that helps in improving the performance, implementation, and accuracy of machine learning algorithms. Or in other words, we can say that it is basically a machine learning ensemble meta-algorithm crafted to enhance the stability and accurateness of algorithms utilised in statistical classification and regression. It is also known as Bootstrap aggregating**.**

Boosting:

Boosting is an ensemble technique that iteratively alters the weight of observation based on the last classification. It helps in enhancing the power of a machine learning program by counting more complicated or qualified algorithms. This process can also help in reducing bias and variance in machine learning**.**

Difference between Bagging and Boosting:

1. Bagging is a learning approach that aids in enhancing the performance, execution, and precision of machine learning algorithms. And Boosting is an approach that iteratively modifies the weight of observation based on the last classification.

2 Bagging is the easiest method of merging predictions that belong to the same type. And Boosting is a method of merging predictions that belong to different types.

3. Here In Bagging, every model has equal weight. And In Boosting , the weight of the models depends on their performance.

4. In bagging, each model is assembled independently. And In boosting, the new models are impacted by the implementation of earlier built models.

5. Bagging helps in solving the over-fitting issue. And Boosting helps in reducing the bias.

6. In the case of bagging, if the classifier is unstable, then we apply bagging. In the case of boosting, If the classifier is stable, then we apply boosting.

8. What is out-of-bag error in random forests?

ANS: Random Forest is one of the machine learning algorithms that use bootstrap aggregation. Random Forest aggregates the result of several decision trees. Decision Trees are known to work well when they have small depth otherwise they overfit. When they are used ensemble in Random Forests, this weakness of decision trees is mitigated.

Random Forest works by taking a random sample of small subsets of the data and applies a decision tree classification to them. The prediction of the Random Forest is then a combination of the individual prediction of the decision trees either by summing or taking a majority vote or any other suitable means of combining the results**.**

Out-of-Bag Error in Random Forest:

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). Bagging uses subsampling with replacement to create training samples for the model to learn from.Out-of-bag error and cross-validation (CV) are different methods of measuring the error estimate of a machine learning model.Bootstrap aggregating allows one to define an out-of-bag estimate of the prediction performance improvement by evaluating predictions on those observations that were not used in the building of the next base learner.

Calculating out-of-bag error:

Since each out-of-bag set is not used to train the model, it is a good test for the performance of the model. The specific calculation of OOB error depends on the implementation of the model, but a general calculation is as follows.

1) Find all models (or trees, in the case of a random forest) that are not trained by the Out-of-bag instance.

2) Take the majority vote of these models' result for the OOB instance, compared to the true value of the OOB instance.

3)Compile the OOB error for all instances in the OOB dataset.

9. What is K-fold cross-validation?

ANS: Cross-validation is a statistical method used to estimate the skill of machine learning models.It is commonly used in applied machine learning to compare and select a model for a given predictive modeling problem because it is easy to understand, easy to implement, and results in skill estimates that generally have a lower bias than other methods.

Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.

The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation.

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

In this tutorial, you will discover a gentle introduction to the k-fold cross-validation procedure for estimating the skill of machine learning models.

After completing this tutorial, you will know:

a) That k-fold cross validation is a procedure used to estimate the skill of the model on new data.

b) There are common tactics that you can use to select the value of k for your dataset.

c) There are commonly used variations on cross-validation such as stratified and repeated that are available in scikit-learn.

It is a popular method because it is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.

The general procedure is as follows:

1. Shuffle the dataset randomly.

2. Split the dataset into k groups

3. For each unique group:

a) Take the group as a hold out or test data set

b) Take the remaining groups as a training data set

c) Fit a model on the training set and evaluate it on the test set

d) Retain the evaluation score and discard the model

4. Summarize the skill of the model using the sample of model evaluation scores.

10. What is hyper parameter tuning in machine learning and why it is done?

ANS: Hyperparameters are the knobs or settings that can be tuned before running a training job to control the behavior of an ML algorithm. They can have a big impact on model training as it relates to training time, infrastructure resource requirements (and as a result cost), model convergence and model accuracy**.** Model parameters are learnt as part of training process, whereas the values of hyperparameters are set before running the training job and they do not change during the training.

Different types of Hyperparameters:

Hyperparameters can be broadly divided into 3 categories —

1) .Model hyperparameters : defines the fundamental construct of a model itself for ex. attributes of a neural network architecture like filter size, pooling, stride, padding.

2). Optimizer hyperparameters : are related to how the model learn the patterns based on data. These types of hyperparameters include optimizers like gradient descent and stochastic gradient descent (SGD), Adam, RMSprop, Adadelta and so on. More details are available here on Keras Optimizer page.

3). Data hyperparameters : are related to the attributes of the data, often used when you don’t have enough data or enough variation in data.

In such cases data augmentation techniques like cropping, resizing, binarization etc. are involved.

The purpose of a hyper-Parameter model is to provide a fast learning environment. It is an important ingredient of many current Machine Learning Software packages. Many of the newer packages come with wide tuning capabilities so that they can be effectively used in many situations where traditional tuning may not be feasible.

11. What issues can occur if we have a large learning rate in Gradient Descent?

ANS: Every time we train a deep learning model, or any neural network for that matter, we're using gradient descent (with backpropagation). We use it to minimize a loss by updating the parameters/weights of the model.

The parameter update depends on two values: a gradient and a learning rate. The learning rate gives you control of how big (or small) the updates are going to be. A bigger learning rate means bigger updates and, hopefully, a model that learns faster.

The learning rate can seen as step size, η. As such, gradient descent is taking successive steps in the direction of the minimum. If the step size η is too large, it can (plausibly) "jump over" the minima we are trying to reach, ie. we overshoot. This can lead to osculations around the minimum or in some cases to outright divergence.The learning rate can seen as step size, η.

. As such, gradient descent is taking successive steps in the direction of the minimum. If the step size η is too large, it can (plausibly) "jump over" the minima we are trying to reach, ie. we overshoot. This can lead to osculations around the minimum or in some cases to outright divergence.

12. Can we use Logistic Regression for classification of Non-Linear Data? If not, why?

ANS: Logistic Regression is a statistical technique used to model the relationship between a dependent variable and one or more independent variables. The dependent variable in a Logistic Regression model is binary, meaning it can take on only two values (e.g. yes/no, true/false). The independent variables, on the other hand, can be continuous or categorical. Logistic Regression models the relationship between the dependent variable and independent variables by using the logistic function to produce a probability that the dependent variable is a certain value, which can then be thresholded to make a final binary prediction.

Logistic Regression is not suitable for complex non-linear relationships between the dependent variable and independent variables. It is also not recommended for multi-class classification problems, as it can only handle binary classification**.**

13. Differentiate between Adaboost and Gradient Boosting.

ANS: Adaboost: Adaboost increases the performance of all the available machine learning algorithms and it is used to deal with weak learners. It gains accuracy just above the arbitrary chances of classifying the problem. The adaptable and most used algorithm in AdaBoost is decision trees with a single level.

Gradient Boosting: The gradient boosting depends on the intuition which is the next suitable possible model, when get combined with prior models that minimize the cumulative predicted errors. The crucial idea of gradient boosting is to fix the targeted outcomes for the next model to reduce the error.

Differentiate between Adaboost and Gradient Boosting:

1) Gradient boosting and Adaboost are ensemble methods applied in machine learning(ML) modeling to improve the efficacy of weak learners to increase algorithm performance. Both use a combination of weak learners to predict a target variable. However, they do that differently.

2) While gradient boosting trains the learners and reduces the weak learners' loss functions by training the model's residues, Ada boost focuses on training the prior miscalculated observations and alters the data distribution to improve sample weight values.

3) And while boosting algorithms are less prone to overfitting, they can get complex and overfit the training data.

4) In Gradient Boosting the classifiers are weighted precisely and their prediction capacity is constrained to learning rate and increasing accuracy.And in Adaboost Every classifier has different weight assumptions to its final prediction that depend on the performance.

5) Gradient boosting cut down the error components to provide clear explanations and its concepts are easier to adapt and understand And in Adaboost The exponential loss provides maximum weights for the samples which are fitted in worse conditions.

So, when it comes to Adaptive boosting the approach is done by up-lifting the weighted observation which is misclassified prior and used to train the model to give more efficacy. In gradient boosting, the complex observations are computed by large residues left on the previous iteration to increase the performance of the existing model.

14. What is bias-variance trade off in machine learning?

ANS: Bias: The bias is known as the difference between the prediction of the values by the ML model and the correct value. Being high in biasing gives a large error in training as well as testing data. Its recommended that an algorithm should always be low biased to avoid the problem of underfitting.

By high bias, the data predicted is in a straight line format, thus not fitting accurately in the data in the data set. Such fitting is known as Underfitting of Data. This happens when the hypothesis is too simple or linear in nature. Refer to the graph given below for an example of such a situation.

Variance: The variability of model prediction for a given data point which tells us spread of our data is called the variance of the model. The model with high variance has a very complex fit to the training data and thus is not able to fit accurately on the data which it hasn't seen before.When a model is high on variance, it is then said to as Overfitting of Data.

It is important to understand prediction errors (bias and variance) when it comes to accuracy in any machine learning algorithm. There is a tradeoff between a model's ability to minimize bias and variance which is referred to as the best solution for selecting a value of Regularization constant. Proper understanding of these errors would help to avoid the overfitting and underfitting of a data set while training the algorithm.

Bias is the simplifying assumptions made by the model to make the target function easier to approximate. Variance is the amount that the estimate of the target function will change given different training data. Trade-off is tension between the error introduced by the bias and the variance.

15. Give short description each of Linear, RBF, Polynomial kernels used in SVM.

ANS: Support Vector Machines: Support Vector Machine (SVM) is a relatively simple Supervised Machine Learning Algorithm used for classification and/or regression. It is more preferred for classification but is sometimes very useful for regression as well. Basically, SVM finds a hyper-plane that creates a boundary between the types of data.

Linear in Support Vector Machine : Linear SVM or Simple SVM is used for data that is linearly separable. A dataset is termed linearly separable data if it can be classified into two classes using a single straight line, and the classifier is known as the linear SVM classifier. It's most commonly used for tasks involving linear regression and classification.

RBF(radial basis function) in SVM: Kernels in SVM classification refer to the function that is responsible for defining the decision boundaries between the classes. Apart from the classic linear kernel which assumes that the different classes are separated by a straight line, a RBF (radial basis function) kernel is used when the boundaries are hypothesized to be curve-shaped.

RBF kernel is used in SVM because it helps the SVM to become non linear rather than linear.RBF kernel function is similar to Normal distribution.

Polynomial kernels used in SVM: In machine learning, the polynomial kernel is a kernel function commonly used with support vector machines (SVMs) and other kernelized models, that represents the similarity of vectors (training samples) in a feature space over polynomials of the original variables, allowing learning of non-linear models.