MACHINE LEARNING WORKSHEET-5

Q.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. = R-squared is considered a better measure of goodness of fit in regression compared to the Residual Sum of Squares (RSS). R-squared is a value between 0 and 1 that represents the proportion of variability in the response variable (y) that is explained by the predictor variables (x). It provides a simple, interpretable measure of the fit of the model. On the other hand, RSS represents the sum of squared differences between the observed and predicted response values, but it does not provide a single value that can be compared to other models. In practice, R-squared is often used to compare different regression models, while RSS is used to assess the magnitude of the residuals and to check for overfitting. In summary, R-squared is generally preferred when evaluating the goodness of fit of a regression model because it provides a clear summary of the model's performance.

Q.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): It measures the total variance in the response variable (Y). It is calculated as the sum of squared differences between each observed response value (yi) and the mean of response values (μ). Mathematically, it is represented as:

TSS = $\Sigma(yi - \mu)^2$

ESS (Explained Sum of Squares): It measures the amount of variation in the response variable (Y) that can be explained by the predictor variable (X) in the regression model. It is calculated as the sum of squared differences between each predicted response value (ŷi) and the mean of response values (μ). Mathematically, it is represented as:

ESS = $\Sigma(\hat{y}i - \mu)^2$

RSS (Residual Sum of Squares): It measures the amount of variation in the response variable (Y) that cannot be explained by the predictor variable (X) in the regression model. It is calculated as the sum of squared differences between each observed response value (yi) and the predicted response value (ŷi). Mathematically, it is represented as:

RSS = $\Sigma(yi - \hat{y}i)^2$

The three metrics are related to each other by the following equation:

TSS = ESS + RSS

In other words, the total variance in the response variable (TSS) is equal to the sum of the variance explained by the predictor variable (ESS) and the variance not explained by the predictor variable (RSS).

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

ANS. = Regularization is a technique used in machine learning to prevent overfitting of the model. Overfitting occurs when a model is trained too well on the training data and memorizes the noise and fluctuations in the data instead of learning the underlying patterns. This leads to poor performance on unseen data. Regularization adds a penalty term to the loss function, which discourages the model from learning the noise and fluctuations in the data. This results in a simpler and more generalizable model. The most common forms of regularization are L1 (Lasso) and L2 (Ridge) regularization. The need for regularization arises when the number of parameters in the model is large compared to the number of samples in the training data.

Q.4] What is Gini–impurity index?

ANS. = The Gini impurity index is a measure of how impure a set of randomly chosen elements is. It is widely used as an evaluation criterion in decision trees and random forests to determine the best split point for each node in a tree. The Gini impurity index takes into account the proportion of the data in each class for a particular node and calculates a score that represents the probability of a randomly chosen element belonging to a certain class. The goal is to minimize the Gini impurity index for a node, which results in a set of elements with high class purity and therefore, more accurate predictions. The lower the Gini-impurity, the purer the sub-group, and the more certain the classification of items in the sub-group.

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

ANS. = Yes, unregularized decision trees are prone to overfitting. This is because they can easily capture the noise in the training data, which leads to an overly complex tree with many branches and leaves. Overfitting occurs when a model is too complex and tries to fit the noise in the data instead of capturing the underlying relationship between the inputs and the outputs. This results in poor generalization performance on new unseen data, which can lead to high prediction errors. To avoid overfitting in decision trees, techniques like pruning, limiting the maximum depth of the tree, or using regularization techniques such as Gini-impurity index can be applied.

Q.6] What is an ensemble technique in machine learning?

ANS. = An ensemble technique in machine learning is a method that combines the predictions of multiple models to produce a more accurate and robust prediction. The idea behind ensemble methods is that the combination of multiple models can result in better performance compared to a single model, as the individual models can compensate for each other's weaknesses and overcome their limitations. Some common ensemble techniques in machine learning include bagging, boosting, random forests, and stacking.

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

ANS. = Bagging (Bootstrapped Aggregating) and Boosting are two popular ensemble techniques in machine learning.

Bagging uses a combination of bootstrapping and aggregating to improve the performance and stability of a single model. It trains multiple models independently on different bootstrapped samples of the training data, and then combines the models to make a prediction. The idea behind bagging is to reduce the variance in the predictions made by a single model.

Boosting, on the other hand, trains a sequence of models in a stage-wise manner. The idea behind boosting is to reduce the bias in the predictions made by a single model. Each model in the sequence focuses on correcting the mistakes made by its predecessor, until the final model has improved the accuracy of predictions. Boosting can also be thought of as assigning higher weights to the samples that are misclassified by the previous model, and training the next model to focus on these samples.

In summary, bagging reduces the variance in the predictions of a single model, while boosting reduces the bias in the predictions of a single model.

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

ANS. = The Out-of-bag (OOB) error is a measure of accuracy of a random forest algorithm, used to evaluate the performance of a model without using cross-validation. It is the error rate of the model on the observations that are not used during the training of each tree in the forest. Each tree in a random forest is trained on a different subset of the data, obtained by bootstrapping the original dataset, and the OOB error is calculated as the average error rate over all the trees on the instances that were not used for training them. This provides a natural way to evaluate the model without the need for a separate validation set, and can be a useful estimate of the test error for small sample sizes.

Q.9] What is K-fold cross-validation?

ANS. = K-fold cross-validation is a model evaluation technique in which the original dataset is divided into k equal parts or folds. The model is trained on k-1 folds and tested on the remaining one fold. This process is repeated k times, each time with a different fold being used as the test set. Finally, the average performance of the model is obtained over all k iterations. The goal of this technique is to maximize the training set while ensuring that all the data is used for testing and evaluation, thereby reducing the risk of overfitting and providing a more reliable estimate of the model's performance.

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

ANS. = Hyperparameter tuning is the process of choosing the best set of hyperparameters for a machine learning model to optimize its performance on a given task. The hyperparameters of a model are its configuration options that are set prior to training, and they control the learning process of the model. Examples of hyperparameters in a machine learning model include the learning rate in gradient descent, the number of hidden units in a neural network, or the maximum depth of a decision tree.

Hyperparameter tuning is done to improve the performance of a machine learning model by selecting the best set of hyperparameters for a given task. This is necessary because the performance of a machine learning model can be greatly impacted by the choice of hyperparameters, and it is often not possible to determine the optimal set of hyperparameters by intuition or simple experimentation. Instead, hyperparameter tuning is used to systematically evaluate a range of hyperparameter values and select the set that results in the best performance on the given task.

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

ANS. = Having a large learning rate in Gradient Descent can result in the following issues:

1] Overstepping: The algorithm takes large steps and may overshoot the optimal solution.

2] Oscillations: The learning rate may be too large, causing the algorithm to oscillate instead of approaching the minimum of the cost function.

3] Slow Convergence: The algorithm may converge slowly if the learning rate is too large, resulting in long training times.

4] Divergence: The algorithm may diverge and never reach a minimum if the learning rate is too high, causing the cost function to increase instead of decrease.

Therefore, it's important to choose a learning rate that is appropriate for the problem and dataset, so the Gradient Descent algorithm can converge to the optimal solution effectively.

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

ANS. = No, We can’t use Logistic Regression for classification of non- Linear data. Logistic Regression is a linear classifier and cannot handle non-linear data well. The hypothesis function of Logistic Regression is a linear function and cannot model complex non-linear relationships between the features and the target variable. It is based on the assumption that the relationship between the features and the target variable is a logistic curve, which can be approximated by a straight line. This makes it unsuitable for handling non-linear data as it cannot capture the non-linear relationship between features and target variables. In such cases, more complex models like decision trees, support vector machines or artificial neural networks should be used instead.

Q.13] Differentiate between Adaboost and Gradient Boosting?

Ans. = AdaBoost and Gradient Boosting are both ensemble learning methods that combine multiple weak models to create a strong model. However, they differ in how they create and combine these weak models.

AdaBoost (Adaptive Boosting) trains a series of weak models, with each model focusing on the samples that the previous model misclassified. The weight of each sample is adjusted based on its classification performance, so that the next model will give more importance to the samples that are harder to classify. The final prediction is made by combining the predictions of all the weak models, with each model having a weight proportional to its classification accuracy.

Gradient Boosting, on the other hand, trains a series of weak models, with each model trying to correct the mistakes of the previous model. The weak models are decision trees, and the combination of these trees forms a final prediction. Unlike AdaBoost, Gradient Boosting does not adjust the weight of samples, but instead uses the gradient descent optimization algorithm to minimize the loss function between the actual and predicted target values.

In summary, AdaBoost focuses on modifying the weight of samples to prioritize difficult to classify samples, while Gradient Boosting focuses on correcting the mistakes of previous models by using gradient descent optimization to minimize the loss function.

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

ANS. = The bias-variance trade-off in machine learning refers to the balance between a model's ability to fit the training data (bias) and its ability to generalize to new, unseen data (variance).

Bias is the error introduced by oversimplifying the underlying relationships in the data. High-bias models make strong assumptions about the data and tend to have a high training error, but low generalization error.

Variance, on the other hand, is the error introduced by making the model too complex, so that it fits the training data too closely and is highly sensitive to the noise and fluctuations in the data. High-variance models have low training error but high generalization error.

The goal in machine learning is to find a model that has the right balance between bias and variance, so that it generalizes well to new data. This involves choosing a model that is neither too simple (high bias) nor too complex (high variance). The trade-off between bias and variance can be adjusted by changing the model's complexity, the size of the training set, and the choice of regularization method.The bias-variance trade-off is a key consideration in the model selection and evaluation process, as it helps practitioners choose a model that strikes a balance between accuracy and generalization.

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

ANS. = Support Vector Machines (SVMs) are a type of machine learning algorithm used for classification and regression tasks. One of the key features of SVMs is their use of kernel functions, which are used to map the input data into a higher-dimensional feature space, where the data can be separated by a hyperplane. There are several different types of kernel functions that can be used with SVMs which are classified as:

1. Linear kernel: The linear kernel is the simplest type of kernel function and is used when the data is linearly separable in the original feature space. The linear kernel computes the dot product of the input data, resulting in a linear decision boundary.

2. RBF (Radial Basis Function) kernel: The RBF kernel is used when the data is not linearly separable in the original feature space. The RBF kernel maps the input data into a higher-dimensional feature space where a non-linear decision boundary can be found. The RBF kernel is defined as the Gaussian function of the Euclidean distance between two points.

3. Polynomial kernel: The polynomial kernel is used to capture non-linear relationships between the input features. The polynomial kernel maps the input data into a higher-dimensional feature space and computes the dot product of the input data raised to a power. The degree of the polynomial kernel determines the degree of non-linearity in the decision boundary.

In general, the choice of kernel function depends on the nature of the data and the problem being solved. The linear kernel is a good starting point for linearly separable data, while the RBF and polynomial kernels can be used for non-linear data.