**MACHINE LEARNING WORKSHEET – 5**

***Q1 to Q15 are subjective answer type questions, Answer them briefly.***

**Q1. 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?**

**Answer :**

R-squared is a better measure of goodness of fit model in regression than Residual Sum of Squares (RSS) because R-squared measures the proportion of the variance in the dependent variable that is explained by the independent variables in the regression model and it is a relative measure of goodness of fit, and it tells us how well the independent variables in the model explain the variation in the dependent variable.

On the other hand RSS measures the sum of the squared residuals, which are the differences between the observed values of the dependent variable and the predicted values of the dependent variable based on the regression model. It measures the absolute fit of the model to the data, and it does not provide a measure of how well the independent variables in the model explain the variation in the dependent variable.

**Q2. 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.**

**Answer :**

In regression analysis, TSS,ESS and RSS are three important measures used to evaluate the performance of a regression model.

TSS measures the total variation in the dependent variable that can be explained by the regression model. It is calculated as the sum of squared differences between the actual values of Y and the mean of Y.

ESS measures the variation in the dependent variable that can be explained by the independent variables in the regression model. It is calculated as the sum of squared differences between the predicted values of Y and the mean of Y.

RSS measures the variation in the dependent variable that is not explained by the independent variables in the regression model. It is calculated as the sum of squared differences between the actual values of Y and the predicted values of Y.

Equation which relates the above three : **TSS = ESS + RSS**

**Q3. What is the need of regularization in machine learning?**

**Answer :**

* Prevent overfitting: Regularization helps to prevent overfitting by reducing the complexity of a model. By adding a penalty term to the loss function, the model is encouraged to generalize better and perform well on unseen data.
* Improve generalization: Regularization can improve the generalization performance of a model by reducing the variance in the model's predictions. This helps to avoid overfitting and improves the ability of the model to generalize well to new data.
* Handle multicollinearity: Regularization can also be used to handle multicollinearity, which is a common problem in linear regression. By adding a penalty term to the model's loss function, the coefficients of highly correlated features are reduced, which helps to improve the stability and interpretability of the model.
* Handle high-dimensional data: Regularization is particularly useful when dealing with high-dimensional data, where there are many features and few observations. Regularization helps to reduce the number of features and prevent overfitting in such cases

**Q4. What is Gini–impurity index?**

**Answer :**

The Gini impurity index is a commonly used measure of impurity or diversity in decision trees and other machine learning algorithms. It measures the probability of incorrectly classifying a randomly chosen element in a dataset if it were labelled randomly according to the distribution of classes in the dataset.

Mathematically, the Gini impurity index is defined as:

G = 1 - ∑(pi)^2

**Q5. Are unregularized decision-trees prone to overfitting? If yes, why?**

**Answer :**

Unregularized decision trees are particularly prone to overfitting when the dataset is noisy or has a large number of features. Without any constraints on the size or shape of the tree, it can easily create highly specific decision rules that capture noise or outliers in the data. This results in a tree that performs well on the training data but poorly on new, unseen data.

To solve this problem, various regularization techniques can be applied to decision trees to prevent overfitting. These techniques include pruning, limiting the depth of the tree, reducing the minimum number of samples required to split a node, and using ensemble methods like bagging or boosting.

**Q6. What is an ensemble technique in machine learning?**

**Answer:**

Ensemble learning is a machine learning technique that involves combining multiple models to improve the performance of a predictive model. These techniques are commonly used to increase the accuracy, stability, and robustness of a ML model by combining the predictions of multiple models. There are two types of Ensemble Techniques :

* Bagging: It is a technique where multiple models are trained on different random subsets of the training data. These models are then combined by taking the average of their predictions. Examples of bagging techniques include Random Forest, Extra Trees, and Bagging Regressor.
* Boosting: Boosting is a technique where multiple models are trained sequentially, with each new model trying to correct the errors of the previous model. Boosting focuses on training weak models that can be combined to form a strong model. Examples of boosting techniques include AdaBoost, Gradient Boosting, and XGBoost.

**Q7. What is the difference between Bagging and Boosting techniques?**

**Answer:**

There are several key differences between the Bagging and Boosting techniques:

* Training process: Bagging trains each model independently on different subsets of the training data, while boosting trains each model sequentially, with each new model focusing on correcting the errors of the previous model.
* Weighting: In Bagging, each model has an equal weight in the final prediction. In contrast, in Boosting, each model's weight is determined by its performance on the training data.
* Sample selection: In Bagging, each model is trained on a random subset of the training data, with replacement. In Boosting, each new model is trained on the instances that were misclassified by the previous models.
* Model type: Bagging can be used with any type of model, while boosting is most commonly used with decision trees.
* Goal: Bagging reduces variance and overfitting, while boosting reduces bias and underfitting.

**Q8.**  **What is out-of-bag error in random forests?**

**Answer:**

Out-of-bag error is a metric used to estimate the generalization error of a random forest model. In a random forest, each tree is trained on a randomly selected subset of the training data, with replacement. This means that each tree sees a different subset of the training data, and some instances are left out of each subset.

**Q9. What is K-fold cross-validation?**

**Answer:**

K-fold cross-validation is a technique used to evaluate the performance of a machine learning model on a limited dataset. It involves splitting the available data into K folds or partitions, where K is typically set to 5 or 10. In each iteration, one fold is used as the test set, and the remaining K-1 folds are used as the training set to fit the model. The model is then evaluated on the test set and the performance metrics are recorded. This process is repeated K times, with each fold being used as the test set exactly once. K-fold cross-validation is a powerful technique for evaluating the performance of a machine learning model and for selecting the best hyperparameters to improve its performance.

**Q10. What is hyper parameter tuning in machine learning and why it is done?**

**Answer:**

Hyperparameter tuning is the process of finding the best values for the hyperparameters of a machine learning algorithm that optimize its performance on a specific task or dataset. Hyperparameters are different from model parameters as they cannot be learned from data directly, but instead are set prior to training the model. Hyperparameter tuning is important because the performance of a machine learning algorithm can be highly sensitive to the choice of hyperparameters. Choosing suboptimal hyperparameters can result in poor model performance, longer training times, and overfitting or underfitting of the model. Overall, hyperparameter tuning is an important step in the machine learning workflow, as it helps to improve the performance of the model and ensure that it is well-suited for the specific task and dataset at hand.

**Q11. What issues can occur if we have a large learning rate in Gradient Descent?**

**Answer:**

* Overshooting the minimum: A large learning rate can cause the gradient descent algorithm to take large steps in the direction of the gradient, potentially overshooting the minimum and causing the algorithm to diverge instead of converging to the optimal solution.
* Slow convergence: If the learning rate is too large, the algorithm may oscillate back and forth across the minimum instead of converging to it, which can significantly slow down the convergence rate of the algorithm.
* Unstable updates: A large learning rate can result in unstable updates, where the parameter values oscillate wildly and fail to converge. This can result in the algorithm failing to find a good solution, or finding a suboptimal solution.
* Poor generalization: If the learning rate is too large, the algorithm may overfit to the training data and fail to generalize well to new data. This is because the algorithm may focus too heavily on fitting the training data, and ignore the underlying patterns that are present in the data.

**Q12. Can we use Logistic Regression for classification of Non-Linear Data? If not, why?**

**Answer:**

Logistic Regression is a linear classifier, which means that it can only model linear decision boundaries between classes. Therefore, it may not be suitable for classification of non-linear data.If the data is non-linear, the decision boundary between classes cannot be modeled as a straight line. In such cases, Logistic Regression may fail to accurately model the relationship between the input features and the target variable, resulting in poor performance.

**Q13. Differentiate between Adaboost and Gradient Boosting.**

**Answer:**

* Iteration process: Adaboost trains a sequence of weak learners iteratively, and each new weak learner is trained on the instances that were previously misclassified. In contrast, Gradient Boosting trains the weak learners in a sequential manner, with each new weak learner trying to correct the errors of the previous learner.
* Overfitting: Adaboost can be prone to overfitting when the base model is too complex or the learning rate is too high, while Gradient Boosting is less prone to overfitting due to its use of more complex base models.
* Learning rate: Adaboost uses a learning rate parameter that controls the contribution of each weak learner to the final model, while Gradient Boosting uses a shrinkage parameter that controls the step size of each iteration.

**Q14.** **What is bias-variance trade off in machine learning?**

**Answer:**

Bias refers to the difference between the expected (or average) prediction of the model and the true value. It is a measure of how much the model's predictions deviate from the correct answer. A model with high bias underfits the data and fails to capture the underlying patterns and relationships between the input features and the target variable.

Variance, on the other hand, refers to the variability of the model's predictions for different input data points. It measures how much the predictions vary for different training sets. A model with high variance overfits the data and captures noise and random fluctuations in the training set, resulting in poor generalization to new data.

**Q15. Give short description each of Linear, RBF, Polynomial kernels used in SVM.**

**Answer:**

Linear kernel: The linear kernel is the simplest kernel function, and it is used when the data can be separated by a hyperplane. The linear kernel simply computes the dot product between each pair of input data points, which is equivalent to measuring the similarity between the data points in the original feature space. The decision boundary in the transformed feature space is a linear hyperplane.

RBF (Radial Basis Function) kernel: The RBF kernel is a popular choice for SVMs because it can model complex, nonlinear decision boundaries. The RBF kernel transforms the input data into an infinite-dimensional feature space by measuring the similarity between each pair of data points using a Gaussian function. The RBF kernel has two hyperparameters: the kernel width (also known as the gamma parameter) controls the width of the Gaussian function and determines the smoothness of the decision boundary, while the regularization parameter (C) controls the trade-off between achieving a low training error and a smooth decision boundary.

Polynomial kernel: The polynomial kernel function transforms the input data into a higher dimensional feature space by computing the dot product of the input data points raised to a certain degree. The degree of the polynomial is a hyperparameter that determines the complexity of the decision boundary. The polynomial kernel can model nonlinear decision boundaries, but it can be prone to overfitting if the degree is too high or the regularization parameter (C) is too small.