Machine Learning Assignment-5

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:- **<u>R-squared</u>** is a statistical measure that represents the proportion of the variance in the dependent variable that is explained by the independent variable(s) in a regression model. It is a key metric used to assess the goodness of fit of the model. The R-squared model ranges from 0 to 1, where 0 (0%) means no variability in dependent variable explained. 1 (100%) explains variability in the dependent variable.

   A better fit is indicated by higher R-squared value, R-squared is often expressed by percentage. For example, an R-squared of 0.80 means that 80% of the variability in the dependent variable is explained by the model.

   R-squared is sensitive to the number of predictors in the model, and adding more predictors may artificially increase R-squared.

   <u>Residual Sum of Squares (RSS):</u> is a statistical measure used to evaluate the goodness of fit of a regression model. It quantifies the total squared differences between the observed values and the values predicted by the model. A smaller RSS indicates a better fit because it means that the model predictions are closer to the actual observed values.

   Its primary use is for model comparison. A model with a lower RSS is considered a better-fitting model, as it suggests that the model's predictions are closer to the observed values.

   <u>Conclusion:</u> Neither R-squared nor Residual Sum of Squares is inherently "better" for measuring the goodness of fit in a regression model. They both offer valuable insights, but in different ways. We can pick one of these models based on what we are trying to achieve.

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: - **<u>Total Sum of Squares:</u>** TSS represents the total variability in the dependent variable (Y) without considering the model. It measures the squared differences between each observed data point and the mean of the dependent variable.

   Formula: TSS = $\sum_{i=1}^{n}(y_i - \bar{y})^2$

   y_i Observed value.
   $\bar{y}$ Mean dependent variable.
   n Number of data points

Explained Sum of Squares (ESS): Represents the variability in the dependent variable that is explained by the regression model. It measures the squared differences between the predicted values from the model and the mean of the dependent variable.

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

$\hat{y}_i$ Predicted value.
$\bar{y}$ Mean dependent variable.
n Number of data points.

Residual Sum of Squares (RSS): Represents the variability in the dependent variable that is not explained by the regression model. It measures the squared differences between the observed values and the predicted values from the model.

Formula: RSS = $\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

y_i Observed value.
$\hat{y}_i$ Predicted value.
n Number of data points.

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

Ans: Regularization is a technique used in machine learning to prevent overfitting and improve the generalization performance of a model.

Preventing Overfitting: Overfitting occurs when a model becomes too complex and fits the training data too closely. As a result, the model may perform poorly on new, unseen data. Regularization introduces a penalty term to the loss function, discouraging the model from fitting the training data too precisely.

Improving Generalizability: Generalizability refers to a model's ability to perform well on data it hasn't seen before. This is essential for any practical application of machine learning. By avoiding overfitting, regularization also improves the model's generalizability. Models that learn general patterns from the training data are more likely to generalize well to new data points.

Additional benefits:
Reduces variance: Regularization can help reduce the variance of a model, which means the model's predictions will be more consistent and less sensitive to small changes in the training data.

Improves interpretability: Some regularization techniques, like L1 regularization, can lead to sparser models with fewer features. This can make the model easier to interpret and understand.
Regularization is essential in machine learning because it helps prevent overfitting, improves generalizability, and can offer other benefits like reduced variance and improved interpretability.

4. What is Gini–impurity index?
   Ans: The Gini Impurity Index, also known simply as Gini Impurity or Gini ratio. It is a measure of impurity used in decision tree algorithms for classification. It tells you how likely it is that a randomly chosen data point would be misclassified if it were simply assigned a class label based on the most frequent class in the dataset or it quantifies the impurity or uncertainty of a set of elements, representing the probability of incorrectly classifying an element chosen randomly from the set.

   Overall, the Gini Impurity Index is a powerful tool for building decision trees and understanding the relationships between features and classes in your data.

5. Are unregularized decision-trees prone to overfitting? If yes, why?
   Ans: Yes, unregularized decision trees are prone to overfitting. Decision trees can continuously split and create more branches to perfectly fit the training data. While this might seem beneficial for capturing intricate patterns in the data, it can lead to poor generalization performance on new, unseen data. Overfitting occurs when a model learns the noise and specificities of the training data rather than the underlying true patterns.

6. What is an ensemble technique in machine learning?
   Ans: Ensemble techniques in machine learning powerful methods that combine predictions from multiple base models to create a single, more accurate and robust model, to improve overall performance and generalization. Ensemble methods are widely used across various machine learning tasks, and they can be applied to both classification and regression problems.

7. What is the difference between Bagging and Boosting techniques?
   Ans: Bagging and boosting are both powerful ensemble techniques in machine learning, but both have different approaches to combining multiple models and improving overall performance. The model differs in their training approaches, weighting schemes, and the way to handle errors.

   Bagging focuses on reducing variance. When we have a high-variance model and want to reduce its instability. Bagging model is used.

   Boosting aims to reduce bias and improve overall accuracy by adapting the training process based on the performance of previous models. When we need to improve both accuracy and generalization, especially for models with high bias we use Boosting technique.

8. What is out-of-bag error in random forests?
   Ans: In the realm of random forests, the "out-of-bag (OOB) error" is a unique and valuable tool for estimating the generalization performance of your model. It essentially refers to the average error of the trees in the forest on data points that were not trained on. This allows you to get a snapshot of how your model might perform on unseen data.

The out-of-bag error is calculated by comparing the predictions on the out-of-bag instances to their actual labels. The error is typically measured using the appropriate metric. The out-of-bag errors from all trees are then averaged to obtain the overall out-of-bag error for the Random Forest.

Out-of-bag error provides an unbiased estimate of the model's performance on unseen data, as it is based on instances that were not part of the training process for each tree.

Out-of-bag error is a powerful tool within random forests, offering a convenient and informative way to evaluate your model's generalization potential.

9. What is K-fold cross-validation?
   Ans: K-fold cross-validation is a method for evaluating a machine learning model performance on unseen data, it is used to assess the performance of a model and to reduce the variance that can result from a single train-test split. It involves splitting the dataset into k equal-sized folds, training the model on k-1 folds, and evaluating it on the remaining fold. This process is repeated k times, with each fold serving as the validation set once. The model performance metrics from each fold are then averaged to provide a more robust estimate of its overall performance.

10. What is hyper parameter tuning in machine learning and why it is done?
    Ans: Hyperparameter tuning in machine learning refers to the process of selecting the optimal hyperparameters for a model. The process of hyperparameter tuning involves exploring different combinations of hyperparameter values to find the set that results in the best model performance. It is a crucial step in the machine learning workflow. The aims is to find the best hyperparameter settings for a model, optimizing its performance and generalization ability on a specific task or dataset.

    This is typically done through a systematic search, and various techniques can be employed to efficiently navigate the hyperparameter space.

11. What issues can occur if we have a large learning rate in Gradient Descent?
    Ans: A large learning rate in Gradient Descent can lead to several issues that can impact model performance.

    Some issue and problems:

    1. Divergence: This is the most serious issue with a large learning rate, the updates to the model parameters can be too large, causing the weights to jump around erratically in the parameter space. This can lead to the model diverging from the optimal solution altogether, never reaching a minimum loss value and becoming unstable.

    2. Oscillations: Even if the model does not completely diverge, a large learning rate can cause it to oscillate around the minimum loss value instead of converging smoothly. This can be inefficient and result in longer training times without improving the model performance.

3. Overfitting: A large learning rate can make the model more sensitive to noise and irrelevant details in the training data. This can lead to overfitting, where the model performs well on the training data but poorly on unseen data.

4. Loss landscape exploration: With a large learning rate, the model might overshoot shallow local minima in the loss landscape, missing out on potentially better solutions deeper in the landscape. This can lead to suboptimal performance even if the model converges.

5. Numerical instability: Large learning rates can lead to numerical instability in some cases, especially for complex models or datasets with high variance. This can lead to unexpected behavior and errors during training.

In general, using a large learning rate is a gamble. It can potentially lead to faster convergence and better performance in some cases, but the risks mentioned above are significant. It's always best to start with a small learning rate and gradually increase it if necessary while monitoring the model's performance closely. Additionally, techniques like adaptive learning rates can help adjust the learning rate dynamically during training to address these issues.

12. Can we use Logistic Regression for classification of Non-Linear Data? If not, why?
Ans: Logistic regression is inherently a linear classifier, it can be adapted or extended to handle non-linear data to some extent. It models the relationship between the independent variables (features) and the log-odds of the target variable, using a logistic function to map the output into the range [0, 1]. Logistic Regression is effective for linearly separable data, it may struggle with complex, non-linear relationships in the data.

Logistic Regression is a powerful tool for linearly separable data and situations where interpretability is crucial, it may not be the best choice for complex non-linear classification tasks. For such cases, exploring non-linear models that can capture more intricate relationships in the data is generally recommended.

13. Differentiate between Adaboost and Gradient Boosting.
Ans: Difference between Adaboost and Gradient Boosting

Adaboost	Gradient Boosting
AdaBoost emphasizes correcting misclassified instances by adjusting data point weights	Gradient Boosting focuses on minimizing the residuals by optimizing a loss function through sequential training of weak learners.
AdaBoost works by sequentially training a series of weak learners on the dataset. Each learner is trained to correct the errors made by the previous ones	Gradient Boosting tends to be more robust and can handle complex relationships in the data.
The objective of AdaBoost is to combine weak learners to create a strong model	The main objective of Gradient Boosting is to minimize the residuals and improve the

with high accuracy. It focuses on reducing bias and improving overall accuracy	overall model fit. It focuses on reducing both bias and variance.
AdaBoost learners are trained sequentially, and each iteration places more emphasis on the instances misclassified by the previous learners. The final model is an ensemble of weighted weak learners.	Implementations of Gradient Boosting include XGBoost, Light GBM, and scikit-learn's Gradient Boosting Classifier/Gradient Boosting Regressor.
May overfit more easily	Often generalizes better
More sensitive to outliers	Less sensitive to outliers

14. What is bias-variance trade off in machine learning?
Ans: The bias-variance trade-off is a fundamental concept in machine learning that involves finding the right balance between bias and variance when building predictive models. It refers to the trade-off between a model ability to accurately capture the underlying patterns in the data (bias) and its sensitivity to variations or noise in the training data (variance). Balancing bias and variance leads to models that generalize well to new, unseen data.

The bias-variance trade-off is a key consideration in model development. It highlights the need to find a model that is complex enough to capture the underlying patterns in the data but not so complex that it overfits to noise. By actively managing these errors, you can avoid the pitfalls of underfitting and overfitting, leading to models that generalize well to real-world data.

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

Linear Kernel: Is the simplest kernel and represents the case where the decision boundary is a hyperplane in the input space. It is suitable for linearly separable data, where classes can be separated by a straight line, plane, or hyperplane. It calculates the dot product between data points, resulting in a linear decision boundary.

RBF (Radial Basis Function) Kernel: Also known as the Gaussian kernel, is versatile and effective for capturing non-linear relationships. It transforms the input space into a high-dimensional space, allowing the SVM to model complex decision boundaries. It projects data into an infinite-dimensional space, allowing for complex non-linear decision boundaries. It can capture a wide range of relationships in data.

Polynomial Kernel: is another option for handling non-linear relationships by transforming the input space into a higher-dimensional space. It introduces polynomial terms of the input features, allowing the SVM to capture non-linear patterns. But it requires careful tuning of parameters.