Machine Learning Assingment (Batch no. = DS2401)

Question 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?

Answer : 1) Interpretability = R-squard represents the proportion of the variance in the dependent variable that is predictable from the independent variables. It ranges between 0 and 1 where 1 indicates a perfect fit. On the other hand RSS is the sum of the squares of the residuals which doesn't have a clear interpretation on its own.

2) Normalization : R-squared is normalized and scale-independent making it easier to compare across different models.RSS being the sum of squared residuals is not normalized and can vary based on the scale of the data.

3) Comparative Analysis : R-squared allows for easy comparison of models. Higher R-squared values generally indicate a better fit whereas lower RSS values alone dont provide a clear indication of model performance without considering the scales of the data

4) Adjusted R-squared : There's also the adjusted R-squared which adjust for the number of predictoors in the model. This helps to penaliize overly complex models,providing a more accurate measure of goodness of fit.

5) Relationship with F-statistic : R-squared is directly related to the F-statistic which is used to test the overall significance of the regression model.A higher R-squared typically leads to a higher F-statistic indicating a better fit of the model

Question 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.

Answer : 1) Total Sum of Squares(TSS) : TSS represents the total variation in the dependent variable (Y) and is calculated as the sum od squared differences between each observed dependent variable value and the mean of the

dependent variable. TSS= $\Sigma(y_i - \bar{y})^2$

Where :

(1) yi is each observed dependent variable values.

(2) $\bar{y}$ is the mean of the dependent variable.

2) Explained Sum of Squares(ESS): ESS measures the variation in the dependent variable explained by the regression model. It represent the sum of squared difference between the predicted values of the dependent variable ($\hat{Y}$) and the mean of the dependent variable. ESS= $\Sigma(\hat{y}_i - \bar{y})^2$

Where:

(1) $\hat{y}_i$ is each predicted value of the dependent variable from the regression model.

(2) $\bar{y}$ is the mean of the dependent variable.

3) Residual Sum of Squares(RRS): RSS measures the unexplained varition in the dependent variable by the regression model. It represent the sum of squared differences between the observed values of the dependent variable(y) and the predicted values from the regression model ($\hat{y}$).

Where :

(1) y_i is each observed dependent variable value.

(2) $\hat{y}_i$ is each predicted value of the dependent variable from the regression model.

The relationship between these three metrics can be expressed using the equation:

TSS = ESS + RSS

this equation illustrates the the total variation in the dependent variable (TSS) is decomposed into the explained variation (ESS) and the unexplained variation (RSS)

by the regression model. In an ideal scenario where the modell perfectly fits the data ESS would equal TSS and RSS would be zero

Question 3 : What is the need of regularization in machine learning?

Answer : 1) Preventing Overfitting: Regularization helps to prevent overfitting by discouraging overly complex models that fit the training data too closely. By penalizing large coefficients or complex model structures regularization encourages models to focus on the most important features and avoid memorizing noise in the data.

2) Improving Generalization : Regularization improves the generalization ability of models allowing them to perform well on unseen data from the same distribution as the training data. By controlling the complexity of the model regularization helps to ensure that the learned patterns are more representative of the underlying true relationship in the data rather than capturing random fluctuation.

3) Handling Multicollinearity : Regularization techniques such as Ridge Regression (L2 regularization) can effectively handle multiconnllinearity which occurs when predictor variables in the model are highly correlated.By shrinking the coeficiennts of correlated variables regularization reduces the impact of mullticollinearity on the model's performance.

4) Feature Selection : Regularization can also act as a form of feature selection by shrinking the coefficients of less important feature towards zero. This helps to identify and prioriitize the most relevant features in the modell leading to simpler and more interpretable models.

5) Robustness to Noise : Regularization increases the model's robustness to noise in the training data by discouraging the model from fitting the noise too closely. This allows the model to focus on learning the underlying patterns that are more likely to generalize to new data.

Overall, regularization is essential in machine learning to improve model performance prevent overfitting enhance generalization ability handle

multicollinearity facilitate feature selection and increase robustness to noise in the training data

Question : 4 **What is Gini–impurity index?**

Answer : The Gini impurity index is a measure of how often a randomly chosen element from the set woulld be incorrectly labeled if it was randomly labeled according to the distribution of llabels in the subset.It's used in deciision tree algorithms particularly in classification problems to evalute the quality of a split.

Here's how it's calculated:

(1) For each class calculate the probability p(i|t) that a randomly chosen element in the set belongs to class i given that it's in node t.

(2) The Gini impurity G(t) of node t is then calculated as:

$G(t)=1-\sum_{i=1}^{n} p(i|t)^2$

Where n is the number of classes

A node with a Gini impurity of 0 is perfectly pure mening all its elements belong to the same class. A node with a Gini impurity of 0.5 is maximally impure meaning its elements are evenly distributed among all classes. In practice decision tree algorithms use the Gini impurity index to evalute potential splits in the data aiming to miniimize impurity in child nodes after splitting

Question : 5 **Are unregularized decision-trees prone to overfitting? If yes, why?**

Answer : Here's why unregularized decision trees are prone to overfitting:

(1) Flexible Structure : Decision trees have a highly flexible structure that can perfectly fit the training data by creating intricate decision boundaries. Each split in a decision tree further partitions the feature space potentially leading to overly complex decision rules that capture noise in the data.

(2) Memlrization of Training Data : Decision trees can memorize the training data by creating branches and leaves that perfectly classify each data point in the training set. This memorization can lead to poor generalization to new data as the as the model may fail to capture the true underlying patterns in the data.

(3) Sensitive to small Variations : Decision trees are sensitive to small variations in the training data. Even slight changes in the training data can result in different splits and decision rules leading to high variance in the model's predictions.

(4) Lack of Constraints : Unregularized decision trees do not impose any constraints on the commplexity of the tree. Without regularization techhniques such as pruning or limiting the depth of the tree decision trees can grow excessively deep and detailed resulting in overfitting

To mitigate overfitting in decission trees various regularization techniques can be applied such as pruning limiting the depth of the tree setting a minimum number of samples required to split a node or using ensemble methods like random Forest or Gradient Boosting which combine multiple decision tree to improve generalization performance.

Quetion : 6 **What is an ensemble technique in machine learning?**

Answer : There are several types of ensemble techniques including:

(1) Bagging (Bootstrap Aggregating): Bagging involves training multiple instance of the same base model on different subsets of the training data typically using bootstrapping (sampling with replacement) The final prediction is obtained by averaging or taking a majority vote among the prediction of the individual models.Random Forest is a popular ensemble algorithm based on bagging using decision trees as base learners.

(2) Boosting : Boosting sequentially trains a series of weak learners(models that are slightly better than random guessing) and assigns higher weights to the instance that are misclassified by previous models.

(3) Stacking : Stacking also known as stacked generalization combines multiple base models with a meta-models (or blender) that learns how to best combine the predictions of the base models. The base models make predictions on the training data and these predictions serve as input features for the meta-model which then produces the final prediction

(4) Voting : Voting methods combine the predictions of multiple base models by taking a simple majority vote (for classification) or averaging (for regression). There are different types of voting such as hard voting(simple majority) and soft voting(weighted average based on predicted probabilites

Question : 7 **What is the difference between Bagging and Boosting techniques?**

Answer : 1) Bagging (Bootstrap Aggregating):

(1) Bagging involves training multiple instances of a base model on different subsets of the training data usually sampled with replacement(bootstrap samples).

(2) Each model learns independently and the final prediction is made by averaging (for regression) or voting (for classification) the prediction of all models.

(3) bagging helps to reduce variance and prevent overfitting by creating diverse models that capture different aspect of the data.

(4) Random Forest is a well-known example of a bagging ensemble technique where the bas model is a decision tree.

2) Boosting : (1) Boosting involves sequentially training multiple instances of a base model where each subsequent model learns to correct the errors made by the previous models.

(2) Models are trained iteratively with each new model focusing more on the

instance that were misclassified by previous models.

(3) Boosting helps to reduce bias and improve predictive performance by focusing difficult-to-predict instances.

(4) Gradient Boosting Machines (GBM) AdaBoost and XGBoost are popular boosting algorithms used in practice.

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

Answer : Random Forests the out of bag (OOB) error also known as out of bag estimate is a method to estimate the performance of the model without the need for a seprate validation set.

Here's how it works:

(1) In each tree of the Random Forest a subset of the training data is randomly selected with replacement to build the tree.This means that some data points are not included in the training set for that particular tree.

(2) These excluded data points form the out of bag samples for that tree.

(3) The out of bag samples are then used to evaluate the prediction accuracy of the tree. Since these sample were not used in training the tree they serve as a natural validation set.

(4) After all trees have been built the OOB error is computed by aggregating the prediction errors from the out of bag samples across all trees in the forest.

Question 9 : What is K-fold cross-validation?

Answer : K-fold cross-validation is a technique used to evalute the performance of a machine learning model by partitioning the dateset into k equal-sized subsets or

"folds." The model is then trained on k-1 folds of the data and evaluated in the remaining fold. this process is repeated k times with each fold serving as the validation set exactly once

Here's how K-fold cross-validation works:

(1) Partitioning the Data: The dataset is divided into k equal-sized subsets (folds).

(2) Training and Validation: The model is trained k times each time using k-1 folds as the training set and the remaining fold as the validation set.

(3) Performance Evaluation: After training on k-1 folds the model's performance is evaluated on the validation fold.The performance metric(eg : accuracy , mean squared error) is recorded.

(4) Repeating the Process: Steps 2 and 3 are repeated k times each time using a diffrent fold as the validation set.

(5) Aggregating Results: The performance metrics from each fold are averaged to obtain a single estimate of the model's performance.

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

Answer : Hyperparameter tuning is done for several reasons:

(1) Optimizing Model Performance : Selecting the right hyperparameters can significantly impact the performance of a machine learning model. By tuning hyperparameters we aim to find the combination that yieldd the best performance on the validation or test set.

(2) Preventing Overfitting : Hyperparameters play a crucial role in controlling model complexity and preventing overfittimg. Tuning hyperparameters such as regularization strenght can help strike the right blance between bais and variance leading to a model that generalizes well to unseen data.

(3) Improving Efficiency : Selecting appropraite hyperparameters can lead to faster convergence and more efficient training. For example tuning the learning rate in gradient descent algorithms can help find the optimal trade off between convergence speed and stability.

(4) Generalization : Hyperparameters tuning helps ensure that the model's performance is not overly reliant on a specific set of hyperparameters values. by exploring a range of hyperparameters values we can build models that generalize better to unseen data and are more robust across different datasets.

(5) Exploring Model Variants : hyperparameters tuning allows for the exploration of different model architectures algorithms and optimization strategies. It enables data scientists and machine learning engineers to experiment with various configuration and select the one that best suits the problem at hand.

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

Answer : If we use a large learning rate in Gradient Descent several issues can arise:

(1) Overshooting the minimum : A large learning rate cn cause the algorithm to overshoot the minimum of the loss function leading to oscillations or divergence in the parameter updates. This can prevent the algorithm from converging to the optimal solution.

(2) Instability : Large lerning rates can make the optimization process unstable causing the parameter updates to fluctuate widely preventing the algorithms from settling into a stable configuration.

(3) Unstable Gradients : Large learning rates can amplify the gradients leading to extremely large updates in parameter values. This can result in numerical instability especially in deep neural network where gradients may become exponentially large or small (vanishing/exploding gradients).

(4) Poor Convergence : A large learning rate can hinder convergence by causing the optimization process to oscillate around the minimum or diverge entirely . This can result in longer training times or failure to find an optimal solution.

(5) Skipping the Minimum : In extreme cases a large learning rate can cause the optimization algorithm to skip over the minimum of the loss function entirely leading to suboptimal or non-convergent solutions.

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

Answer : Logistic Regression can be used for classification of non-linear data but it might not perform well when the decision boundary between classes is highly non-linear. It assumes a linear relationship between features and the log odds of the target variables so it may struggle to capture complex non - linear patterns in the data . Other methods like feature engineering kernel methods ensemble methods or neural networks may be more suitable for such scenarios.

Question 13 : Differentiate between Adaboost and Gradient Boosting.?

Answer :

Adaboost :

(1) Sequentially trains weak learners on modified version of the dataset.

(2) Minimizes the exponential loss function adjusting instance weight to emphasize misclassified samples.

(3) Primarily focuses on minimizing classification errors exponentially

(4) can be parallelized to some extent but relies on squential weight updates.

Gradient Boosting :

(1) Builds a sequence of weak learners to corrects errors made by the previous ensemble.

(2) Optimizes various loss functions (eg mean squared error , logistic loss) by fitting subsequent models to the residual errors

(3) More flexible in loss function choice and focusus in minimizing residual errors.

(4) Sequential nature makes full parallelization challenging although parallelism can be introduced for certain operations.

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

Answer : The bias-variance tradeoff is a fundamental concept in machine learning that relates to the balance between bias and variance in predictive models. It refers to the tradeoff between the model's ability to capture the true underlying patterns in the data (bias) and its sensitivity to small fluctuations or noise in the training data (variance).

(1) Bias : measures the error introduced by approximating a real-world problem with a simplified model. A high bias model tends to make strong asumption about the data which may lead to underfitting. In other words a biased model may oversimplify the underlying patterns in the data and may fail to capture important relationships.

(2) Variance : Variance measures the model's sensitivity to fluctuation or noise in the training data .A high variance model captures complex relationships in the training data but may also capture noise leading to overfitting. In other word a model with high variance may fit the training data too closely and may not generalize well to unseen data.

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

Answer :

1) Linear Kernel

(1) The linear kernel is the simplest kernel function used in SVMs.

(2) It computes the inner product of the features vectors in the orignal feature space

(3) The decision boundary generated by the linear kernal is a hyperplane that separates the classes.

2) RBF (Radial Basis Function) Kernel

(1) The RBF kernel is a popular choice for SVMs and is often used when the decision boundary is non-linear.

(2) It maps the orignal feature space into a higher-dimensional space using a non-linear transformation.

(3) The RBF kernel is characterized by a parameter γ, which determines the influence of each training example on the decision boundary.

3) Polynomial Kernel

(1) The polynomial kernel is used to capture non-linear relationship between features.

(2) It computes the inner product of the feature vectors in a higher-demensional space using polynomial functions.

(3) the polynomial kernel is characterized by two parameters the degree of the polynomial d and an optional bias term c.