MACHINE LEARNING

ASSIGNMENT

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

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:- R-squared is generally a better measure of the goodness of fit for a regression model than the residual sum of squares (RSS).

R-squared, denoted as R^2, is a statistical measure that represents the proportion of the variance for the dependent variable that's explained by the independent variables in the model. It is dimensionless and ranges from 0 to 1, where a value closer to 1 indicates a better fit. R^2 is calculated using the formula:

$R^2 = 1 - \frac{RSS}{TSS}$

where RSS is the residual sum of squares, and TSS is the total sum of squares. TSS represents the total variance in the dependent variable.

The RSS, on the other hand, is the sum of the squared differences between the observed actual outcomes and the outcomes predicted by the regression model. It is calculated as:

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

wherey_i is the actual value and $\hat{y}_i$ is the predicted value from the model for the $i\text{-th}$ observation.

The reason why R^2 is often preferred over RSS as a measure of goodness of fit is due to its standardized nature:

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 :- In regression analysis, the following metrics are used to measure the variability in the data and how well the model fits the data:

1. **Total Sum of Squares (TSS)**:
   - TSS measures the total variability in the observed data. It represents the sum of the squared differences between each observed value and the mean of the observed values.

     TSS=∑i=1n(yi−y¯)2

     where yi is the observed value, and y¯ is the mean of the observed values.

2. **Explained Sum of Squares (ESS)**:
   - ESS measures the variability explained by the regression model. It represents the sum of the squared differences between the predicted values and the mean of the observed values.

ESS=∑i=1n(y^i−y¯)2

where y¯ is the predicted value.

3. **Residual Sum of Squares (RSS)**:
    - RSS measures the unexplained variability, or the variability in the observed data that the model does not capture. It represents the sum of the squared differences between the observed values and the predicted values.

      RSS=∑i=1n(yi−y^i)2

The relationship between these three metrics is given by:

TSS=ESS+RSS

This equation reflects the idea that the total variability in the data (TSS) is partitioned into the variability explained by the model (ESS) and the unexplained variability (RSS).

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

Answer:- Regularization is essential in machine learning to prevent overfitting, which occurs when a model learns the noise in the training data rather than the underlying pattern. Regularization techniques add a penalty to the model's complexity, encouraging simpler models that generalize better to new, unseen data. This results in improved performance and robustness of the model on test data. Common regularization methods include L1 (Lasso) and L2 (Ridge) regularization.

4. What is Gini–impurity index?

Answer:- The Gini impurity index is a measure of the impurity or diversity of a dataset, commonly used in decision tree algorithms to determine how a feature splits a dataset. It quantifies the likelihood of an incorrect classification of a randomly chosen element if it were randomly labeled according to the distribution of labels in the dataset.
The formula for Gini impurity is:

Gini impurity=1−i=1∑npi2

Where pi is the probability of an element being classified into class i .
A Gini impurity of 0 indicates that all elements belong to a single class, while a higher Gini impurity indicates more diversity and a mix of classes.

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

Answer:- Yes, unregularized decision trees are prone to overfitting. This happens because decision trees can grow very deep and complex, capturing all the nuances and noise in the training data. When a tree is fully grown without any restrictions, it will create a separate branch for almost every data point, leading to a model that performs well on the training data but poorly on new, unseen data. This overfitting results in high variance and poor generalization to test data. Regularization techniques, such as limiting the depth of the tree, setting a minimum number of samples per leaf, or pruning, are used to prevent overfitting by controlling the complexity of the tree.

6. What is an ensemble technique in machine learning?

Answer:- An ensemble technique in machine learning involves combining multiple models to improve overall performance. By aggregating the predictions of several models, ensembles can reduce errors and enhance accuracy compared to individual models. Common ensemble methods include bagging (e.g., Random Forest), boosting (e.g., AdaBoost, Gradient Boosting), and stacking. These techniques leverage the strengths of various models and mitigate their weaknesses, leading to more robust and reliable predictions.

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

Answer:- Bagging and boosting are both ensemble techniques used to improve the performance of machine learning models, but they differ in their approach and methodology:

1. Bagging (Bootstrap Aggregating):-
   Objective:- Reduce variance and prevent overfitting.
   Method:- Creates multiple subsets of the training data by random sampling with replacement (bootstrap samples). Each subset is used to train a separate model (often the same type, like decision trees), and the final prediction is made by averaging (for regression) or majority voting (for classification) across all models.
   Parallel Training:- Each model is trained independently in parallel.
   Example:- Random Forest, which uses bagging with decision trees.

2. Boosting:-
   Objective:- Reduce bias and improve predictive accuracy.
   Method:- Trains models sequentially, where each new model focuses on correcting the errors made by the previous models. The training data is weighted, with higher weights given to instances that were misclassified by previous models. The final prediction is made by a weighted combination of all models.
   Sequential Training:- Each model is trained in sequence, with adjustments based on the performance of previous models.
   Example:- AdaBoost, Gradient Boosting.

Key Differences:-
Training Process:- Bagging trains models independently and in parallel, while boosting trains models sequentially.
Focus:- Bagging reduces variance by averaging predictions of multiple models, while boosting reduces bias by focusing on and correcting errors of previous models.
Data Sampling:- Bagging uses bootstrap sampling to create different subsets of data, whereas boosting uses the entire dataset but adjusts weights based on previous model performance.

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

Answer:- Out-of-bag (OOB) error in random forests is an estimate of the prediction error for a model, based on the bootstrap sampling technique used to train the model. Here's how it works:

1. Bootstrap Sampling:- In random forests, each decision tree is trained on a different bootstrap sample, which is a random sample taken with replacement from the original training dataset. This means that about 63% of the original data is used for training each tree, while the remaining 37% is left out and not used in the training of that particular tree.

2. Out-of-Bag Data:- The data points not included in a given bootstrap sample are called out-of-bag data for that tree.

3. Error Estimation:- To estimate the OOB error, each data point is predicted by the trees that did not include it in their bootstrap sample. The OOB error is then calculated as the average prediction error over all the data points using these out-of-bag predictions.

The OOB error provides a robust and unbiased estimate of the model's performance without the need for a separate validation set. This is particularly useful as it makes efficient use of the available data.

9. What is K-fold cross-validation?

Answer:- In K-fold cross-validation, the data set is divided into a number of K-folds and used to assess the model's ability as new data become available. K represents the number of groups into which the data sample is divided.

K-fold cross-validation is a resampling technique used to evaluate the performance and generalizability of a machine learning model.
In summary, K-fold cross-validation helps in assessing how well a model will generalize to an independent dataset and is a standard method for model evaluation in machine learning.

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

Answer:- Hyperparameter tuning in machine learning involves selecting the best set of hyperparameters for a model to improve its performance. Hyperparameters are parameters that are set before the training process begins and are not learned from the data, such as learning rate, number of trees in a random forest, or the depth of a decision tree.

Why Hyperparameter Tuning is Done:
Improve Model Performance:- The right set of hyperparameters can significantly enhance the model's accuracy, precision, recall, or other performance metrics.
Prevent Overfitting/Underfitting:- Tuning helps find a balance between overfitting (model too complex) and underfitting (model too simple), ensuring that the model generalizes well to unseen data.
Optimize Resource Usage:- Efficient hyperparameter settings can reduce computational costs and training time while achieving better results.

Common Methods for Hyperparameter Tuning:-
Grid Search:- Exhaustively searches through a predefined set of hyperparameters. This method is simple but can be computationally expensive.

Random Search:- Samples hyperparameters randomly from a predefined range. It can be more efficient than grid search for large hyperparameter spaces.
Bayesian Optimization:- Uses probabilistic models to explore the hyperparameter space and find the optimal settings more efficiently than grid or random search.
Automated Machine Learning (AutoML):- Tools and libraries that automate the hyperparameter tuning process along with other aspects of model selection and training.

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

Answer:- A large learning rate in Gradient Descent can lead to several issues:-

Overshooting:- The model parameters may overshoot the optimal values, causing the algorithm to miss the minimum of the cost function. This can result in erratic behavior and divergence instead of convergence.
Divergence:- Instead of converging to a minimum, the model's cost function may start increasing, causing the algorithm to diverge. This happens because the updates are too large, pushing the parameters away from the optimal values.
Instability:- Large learning rates can cause the training process to be unstable, with the cost function fluctuating widely and failing to settle into a stable solution.
Poor Convergence:- Even if the algorithm converges, it may not converge to the global minimum or the optimal solution, leading to suboptimal performance.

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

Answer:- Logistic Regression has traditionally been used as a linear classifier, i.e. when the classes can be separated in the feature space by linear boundaries. That can be remedied however if we happen to have a better idea as to the shape of the decision boundary…
Logistic regression is known and used as a linear classifier. It is used to come up with a hyperplane in feature space to separate observations that belong to a class from all the other observations that do not belong to that class. The decision boundary is thus linear. Robust and efficient implementations are readily available (e.g. scikit-learn) to use logistic regression as a linear classifier.
Logistic Regression is primarily designed for binary classification problems with linear decision boundaries. This means it performs well when the classes can be separated by a straight line (or a hyperplane in higher dimensions).

Limitations with Non-Linear Data:-
Linear Decision Boundaries:- Logistic Regression models the probability of a class label using a linear decision boundary, which may not be suitable for datasets where the classes are not linearly separable.
Poor Performance on Complex Patterns:- For data with complex, non-linear relationships between features and class labels, a linear model may not capture the underlying patterns effectively.
Workarounds:-
Feature Engineering: -Transform the features to include non-linear combinations or polynomial terms. This can sometimes make the problem linearly separable in the transformed feature space.

Kernel Trick:- Use techniques such as the kernel trick (in methods like Support Vector Machines with non-linear kernels) to map data into higher-dimensional spaces where it may become linearly separable.
Non-Linear Models:- Consider other algorithms designed for non-linear classification, such as decision trees, random forests, or neural networks.

In summary, while Logistic Regression may struggle with non-linear data due to its linear nature, it can sometimes be adapted with feature transformations. For inherently non-linear patterns, other machine learning models might be more appropriate.

13. Differentiate between Adaboost and Gradient Boosting.

Answer:- The most significant difference is that gradient boosting minimizes a loss function like MSE or log loss while AdaBoost focuses on instances with high error by adjusting their sample weights adaptively.

Gradient boosting models apply shrinkage to avoid overfitting which AdaBoost does not do.

Gradient boosting also performs subsampling of the training instances while AdaBoost uses all instances to train every weak learner.

Overall gradient boosting is more robust to outliers and noise since it equally considers all training instances when optimizing the loss function.

AdaBoost is faster but more impacted by dirty data since it fixates on hard examples.

An additive model where shortcomings of previous models are identified by high-weight data points. But An additive model where shortcomings of previous models are identified by the gradient.

The trees are usually grown as decision stumps.but The trees are grown to a greater depth usually ranging from 8 to 32 terminal nodes.

Each classifier has different weights assigned to the final prediction based on its performance and .All classifiers are weighed equally and their predictive capacity is restricted with learning rate to increase accuracy.

It gives weights to both classifiers and observations thus capturing maximum variance within data. But It builds trees on previous classifier's residuals thus capturing variance in data.

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

Answer:- Bias-Variance Tradeoff is a fundamental concept in machine learning that deals with the balance between model bias and variance. In simpler terms, it refers to the tradeoff between a model's ability to accurately represent the underlying data patterns (low bias) and its susceptibility to fluctuations with changes in the training data (high variance).
Bias-Variance Tradeoff is crucial in machine learning because it directly impacts a model's predictive performance. A model with high bias will consistently produce predictions that are far from the actual values, while a model with high variance will

produce widely varying predictions for different training datasets. In both cases, the model's ability to generalize to new, unseen data is compromised.
By understanding and optimizing the bias-variance tradeoff, businesses can develop machine learning models that strike the right balance between simplicity and accuracy. This results in robust models that are less prone to overfitting and more likely to make accurate predictions on real-world data, leading to better decision-making and improved business outcomes.
Several related concepts and techniques contribute to understanding and managing the bias-variance tradeoff:

Regularization: A technique used to control model complexity and prevent overfitting by penalizing large parameter values.

Cross-Validation: A method to evaluate a model's performance on multiple data subsets, helping to estimate bias and variance.

Ensemble Methods: Combining multiple models to reduce variance and improve overall predictive performance.

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

Answer:- In Support Vector Machines (SVM), kernels are functions used to transform the input data into a higher-dimensional space where a linear decision boundary can be applied. Here’s a brief description of three commonly used kernels:-

1. Linear Kernel:-

Description:- The linear kernel is the simplest kernel function. It computes the dot product of the input vectors directly, without any transformation.

Formula:- (K(x, x') = x \cdot x' \)

Use Case:- Suitable for problems where the data is linearly separable. It is equivalent to using a linear SVM without mapping the data to a higher-dimensional space.

2. Polynomial Kernel:-

Description:- The polynomial kernel allows for learning non-linear decision boundaries by computing a polynomial function of the dot product of the input vectors. It can capture interactions between features up to a specified degree.

Formula:- (K(x, x') = (x \cdot x' + c)^d \)

Parameters:- (c \) (constant term) and \(d \) (degree of the polynomial).

Use Case:- Useful for capturing interactions and non-linear relationships in the data, where the relationship between features and classes is polynomial in nature.

3. RBF (Radial Basis Function) Kernel:-

Description: -The RBF kernel, also known as the Gaussian kernel, measures the similarity between input vectors based on their distance in the feature space. It maps data into an infinite-dimensional space, allowing it to capture complex patterns.

Formula:- (K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right) \)

Parameter:- (\sigma \) (the width of the Gaussian function).

Use Case:- Effective for problems where the decision boundary is highly non-linear. It works well in many cases and can handle complex patterns and relationships in the data.

Each kernel function has its strengths and is chosen based on the specific characteristics and requirements of the dataset.