ASSIGNMENT - 5

MACHINE LEARNING

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?**

Ans- R-squared is generally a better measure of the goodness of fit for a regression model than the residual sum of squares. Because

- Normalization: R-squared is a proportion (0 to 1) representing the proportion of variance explained by the model, making it easier to interpret and compare across different models and datasets.
- Contextual Meaning: It shows how much of the total variance in the dependent variable is explained by the model, providing a clearer measure of model effectiveness.
- Comparative Utility: It allows for straightforward comparison between models, whereas RSS is scale-dependent and less intuitive for such comparisons.

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- In regression analysis:

- **Total Sum of Squares (TSS)** measures the total variance in the dependent variable y around its mean:
  $TSS=\sum(yi-\bar{y})2$

- **Explained Sum of Squares (ESS)** measures the variance explained by the regression model:
  $ESS=\sum(\hat{y}i-\bar{y})2$

- **Residual Sum of Squares (RSS)** measures the variance not explained by the model (the model's error):
  $RSS=\sum(yi-\hat{y}i)2$

Relationship:

$TSS=ESS+RSS$

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

Ans- Regularization in machine learning is needed to:

1. **Prevent Overfitting**: By penalizing complex models, it helps avoid fitting noise in the training data, leading to better performance on unseen data.
2. **Improve Generalization**: It encourages simpler models that generalize better to new, unseen data.
3. **Reduce Complexity**: It helps in managing model complexity, making the model more interpretable and stable.

4- What is Gini–impurity index?

Ans- The Gini impurity index measures the purity of a node in a decision tree. It quantifies the likelihood of incorrectly classifying a randomly chosen element from the node if it were randomly labeled according to the distribution of classes in that node. The formula is:

$$Gini=1-\sum_{i=1}^{k}(p_i)^2$$

where p_i is the proportion of samples belonging to class i. A Gini index of 0 indicates perfect purity (all samples in the node belong to a single class), while higher values indicate more mixed classes.

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

Ans- Yes, unregularized decision trees are prone to overfitting because they can grow very large and complex, capturing noise and outliers in the training data. This excessive complexity leads to high variance and poor generalization to new, unseen data.

6- What is an ensemble technique in machine learning?

Ans- An ensemble technique in machine learning combines multiple models to improve performance and robustness. By aggregating predictions from various models, ensembles can reduce errors and enhance accuracy, stability, and generalization. Common methods include bagging (e.g., Random Forests), boosting (e.g., Gradient Boosting), and stacking.

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

Ans- Bagging and Boosting are both ensemble techniques but differ in their approach:

- **Bagging (Bootstrap Aggregating)** Builds multiple models independently on different subsets of the training data (created by sampling with replacement) and averages their predictions. It reduces variance and helps prevent overfitting. Example: Random Forests.

- **Boosting**: Builds models sequentially, where each new model corrects the errors of the previous ones. It focuses on difficult-to-predict cases and improves accuracy by combining the predictions of weak learners. Example: Gradient Boosting Machines (GBM).

In essence, bagging reduces variance by averaging multiple models, while boosting improves accuracy by focusing on mistakes from previous models.

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

Ans- In Random Forests, **Out-of-Bag (OOB) error** is an estimate of model performance calculated using data points that were not included in the bootstrap sample used to train each tree. For each tree, the OOB error is computed by predicting these excluded data points and comparing the predictions to the true values, providing an internal measure of the model's accuracy.

9- What is K-fold cross-validation?

Ans- **K-fold cross-validation** is a technique for evaluating a model's performance by dividing the dataset into (K) subsets or "folds". The model is trained (K) times, each time using (K-1) folds for training and the remaining fold for validation. The performance metrics from each fold are then averaged to provide a more reliable estimate of the model's accuracy.

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

Ans- Hyperparameter tuning in machine learning involves selecting the optimal values for hyperparameters settings or configurations for the learning algorithm that are not learned from the data but set before training. It is done to improve model performance, enhance accuracy, and ensure that the model generalizes well to new, unseen data by finding the best combination of hyperparameters for the given problem.

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

Ans- A large learning rate in Gradient Descent can cause:

1. Overshooting: The algorithm may jump over the optimal point, missing the minimum of the loss function.
2. Divergence: It can lead to unstable training, where the loss function increases instead of decreasing**.**
3. Poor Convergence: The model may fail to converge to the optimal solution, resulting in suboptimal performance.

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

Ans- Logistic Regression is typically not well-suited for classifying non-linear data directly because it assumes a linear relationship between the input features and the log-odds of the target variable. However, you can use it for non-linear classification by:

1. Feature Engineering: Transforming or creating new features that capture non-linear relationships (e.g., polynomial features).
2. Kernel Methods: Applying techniques like the kernel trick to map data into higher-dimensional spaces where linear separation might be possible.
3. Non-linear Extensions: Using logistic regression as part of more complex models like neural networks, where non-linearities are inherently handled.

Without such adjustments, standard logistic regression may struggle with non-linearly separable data**.**

13- Differentiate between Adaboost and Gradient Boosting?

Ans- AdaBoost and Gradient Boosting differ in:

- Error Correction:
- AdaBoost: Adjusts weights of misclassified samples to focus on difficult cases, combining weak learners through weighted voting.
- Gradient Boosting: Builds models sequentially to correct residual errors from previous models, optimizing a loss function via gradient descent.

- Model Aggregation:
- AdaBoost: Models are combined based on their weighted performance.
- Gradient Boosting: Adds new models to fit residuals from the existing ensemble.

- Sensitivity:

- **AdaBoost:** More sensitive to noisy data.
- **Gradient Boosting:** Generally more robust to noise due to systematic residual fitting.

14- What is bias-variance trade off in machine learning?
Ans- The **bias-variance trade-off** in machine learning is the balance between two types of errors:

- **Bias:** Error due to overly simplistic models that underfit the data, leading to poor performance on both training and test sets.
- **Variance:** Error due to overly complex models that overfit the training data, performing well on training data but poorly on new data.

The goal is to find a model that minimizes both bias and variance to achieve good performance and generalization.

15- Give short description each of Linear, RBF, Polynomial kernels used in SVM.
Ans. Linear Kernel: Computes the similarity between two vectors using a linear function, suitable for linearly separable data. Its formula is $K(x, x') = x \cdot x'$.

RBF (Radial Basis Function) Kernel: Measures similarity using a Gaussian function, effective for handling non-linear data by mapping it into a higher-dimensional space. Its formula is $K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$, where σ is a parameter controlling the width of the Gaussian.

Polynomial Kernel: Computes similarity using polynomial functions, allowing the model to learn non-linear relationships. Its formula is $K(x, x') = (x \cdot x' + c)^d$, where c is a constant and d is the polynomial degree.