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?

Answer-

RSS is generally considered a more comprehensive measure of goodness of fit. This is because RSS takes into account the magnitude of the residuals, whereas R-squared only looks at the proportion of variance explained. RSS is also less susceptible to issues like multicollinearity, which can affect R-squared.

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-

Total Sum of Squares (TSS): This measures the total variation in the dependent variable (y) around its mean. It's a measure of the total amount of variation in the data.

Explained Sum of Squares (ESS): This measures the variation in the dependent variable (y) that is explained by the independent variables (x) in the regression model. It's a measure of how well the model fits the data.

Residual Sum of Squares (RSS): This measures the variation in the dependent variable (y) that is not explained by the independent variables (x) in the regression model. It's a measure of the error or residuals in the model.
The equation relating these three metrics is: TSS = ESS + RSS

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

Answer-

Regularization is a crucial technique in machine learning that helps prevent overfitting, a common problem where a model becomes too complex and accurately fits the training data but fails to generalize well to new, unseen data. It Prevents Overfitting, Improves Model Stability, Reduces Model Complexity, Enhances Generalization, Improves Model Interpretability,etc.

4. What is Gini–impurity index?

Answer-

Gini impurity is a measurement used in decision trees to determine how the features of a dataset should split nodes to form the tree. It's a number between 0 and 0.5 that indicates the likelihood of new, random data being misclassified if it were given a random class label according to the class distribution in the dataset. The Gini impurity is used to build decision trees by selecting the feature and split point that results in the lowest Gini impurity. This process is repeated for each node in the tree until all nodes are pure, meaning they contain only one class.

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

Answer-

Unregularized decision-trees are prone to overfitting due to their complexity, sensitivity to noise and outliers, lack of generalization, and tendency to overfit to the training data. Regularization techniques, such as pruning and early stopping, can help mitigate overfitting in decision trees.

6. What is an ensemble technique in machine learning?

Answer-

Ensemble techniques in machine learning involve combining multiple models to improve performance. These techniques can be broadly categorized into three types: bagging, boosting, and stacking. Each of these techniques offers a unique approach to improving predictive accuracy.

Bagging: Bagging is an ensemble learning technique designed to improve the accuracy and stability of machine learning algorithms. It works by creating multiple datasets from the original data using bootstrap sampling and training a model on each dataset. The predictions from each model are then combined to produce a final prediction.

Boosting: Boosting is another ensemble learning technique that trains models sequentially, with each model focusing on the mistakes made by the previous model. This process continues until a desired level of accuracy is achieved.

Stacking: Stacking is a more advanced ensemble learning technique that involves training a meta-model to make predictions based on the predictions of other models. The meta-model is trained on the predictions made by the individual models, allowing it to learn how to combine their predictions to produce the best overall result.

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

Answer-

	Bagging	Boosting
Defination	Bagging is a parallel ensemble learning method that combines multiple models trained on different subsets of the same dataset	Boosting is a sequential ensemble learning method that combines multiple models trained on the same dataset, with each model focusing on correcting the mistakes of the previous model.
Objective	Reduce variance and prevent overfitting	Reduce bias and improve accuracy
Base Learners	Independent models trained in parallel	Sequentially trained weak learners
Weighting	Equal weight for all base learners	Weighted based on performance
Error Correction	Independent errors; no re-weighting	Emphasis on correcting mistakes
Training Speed	Parallel training; faster	Sequential training; slower
Final Model	Average or voting of base models	Weighted sum of base learners
Robustness	Less prone to overfitting	Prone to overfitting if not controlled

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

Answer-

The out-of-bag (OOB) error in random forests is a performance metric that provides an unbiased estimate of the model's performance on unseen data. It is calculated using the samples that are not used in the training of the model, which are known as out-of-bag samples. These samples are used to compute the OOB error for each individual tree in the forest, and the OOB error for the entire forest is computed by averaging the OOB errors of the individual trees.

9. What is K-fold cross-validation?

Answer-

K-fold cross-validation is a widely used statistical method in machine learning for estimating the performance of a model on unseen data. It involves splitting a dataset into k subsets or folds, where each fold is used as the validation set in tur n, while the remaining k-1 folds are used for training. This process is repeated k times, and performance metrics such as accuracy, precision, and recall are computed for each fold.

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

Answer-

Hyperparameter tuning in machine learning refers to the process of selecting the optimal values for the hyperparameters of a machine learning model. Hyperparameters are external parameters that are set before training a model, whereas model parameters are internal parameters that are learned during the training process.
Hyperparameter tuning is done to increase the efficiency of a model by tuning the parameters of the neural network. This is achieved by trying out different combinations of hyperparameters and evaluating their impact on the model's performance. The goal is to find the set of hyperparameters that results in the best performance of the model.

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

Answer-

If the learning rate is too large in Gradient Descent, several issues can occur:
Unstable training process: A large learning rate can cause the model to oscillate between different regions of the loss landscape, leading to an unstable training process.
Overfitting: A large learning rate can cause the model to learn too quickly, resulting in overfitting to the training data.
Convergence issues: A large learning rate can cause the model to converge too quickly, resulting in poor generalization to unseen data.
Exploding gradients: A large learning rate can cause the gradients to explode, making it difficult to train the model.
Divergence: A large learning rate can cause the model to diverge, resulting in a poor fit to the training data.

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

Answer-

No, we cannot use Logistic Regression for classification of non-linear data. Logistic Regression is a linear model that assumes a linear relationship between the independent variables and the log-odds of the dependent variable. It is not designed to handle non-linear relationships. Linear assumption, Loss function, Model interpretation, Overfitting, Non-convex optimization are some reasons why Logistic Regression is not suitable for non-linear data.

13. Differentiate between Adaboost and Gradient Boosting.

Answer-

	Adaboost	Gradient Boosting
Approach to Building the Ensemble	Adaboost builds the ensemble by iteratively training a series of weak learners, each one trying to correct the mistakes of the previous one.	Gradient Boosting builds the ensemble by training a series of weak learners, each one trying to fit the residual errors made by the previous one.
Loss Function	Adaboost uses an exponential loss function, which makes it more sensitive to outliers.	Gradient Boosting uses a differentiable loss function, which makes it less sensitive to outliers.
Iterative Process	Adaboost iteratively trains a series of weak learners, with each iteration focusing on the harder cases.	Gradient Boosting iteratively trains a series of weak learners, with each iteration trying to fit the residual errors made by the previous one.
Weak Learners	Adaboost is effective when used with weak learners, such as decision trees.	Gradient Boosting can be used with a wide range of weak learners, including decision trees, linear models, and neural networks.
Parallelization	Adaboost cannot be parallelized, as each iteration depends on the previous one.	Gradient Boosting can be parallelized, as each iteration can be trained independently.

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

Answer-

The bias-variance tradeoff is a fundamental concept in machine learning that refers to the balance between two types of errors that a model can make: bias and variance. The bias-variance tradeoff is about finding the right balance between these two types of errors. As a model becomes more complex (i.e., has more parameters), it can reduce its bias by capturing more of the underlying patterns in the data. However, this increased complexity can also lead to higher variance, as the model becomes more prone to overfitting.

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

Answer-

Linear Kernel: The linear kernel is a simple and intuitive kernel that maps the input data into a higher-dimensional space where a linear separator can be found. It is the simplest and most common kernel used in SVM. The linear kernel is defined as K(x, y) = x^T y, where x and y are the input vectors.

RBF (Radial Basis Function) Kernel: The RBF kernel is a non-linear kernel that maps the input data into a higher-dimensional space where a non-linear separator can be found. It is defined as K(x, y) = exp(-γ ||x-y||^2), where γ is a hyperparmeter that controls the width of the kernel. The RBF kernel is often used for problems where the data is not linearly separable.

Polynomial Kernel: The polynomial kernel is a non-linear kernel that maps the input data into a higher-dimensional space where a non-linear separator can be found. It is defined as K(x, y) = (x^T y + 1)^d, where d is the degree of the polynomial. The polynomial kernel is often used for problems where the data is not linearly separable and the relationship between the input features is non-linear.