Worksheet\_Set\_5

**MACHINE LEARNING**

. 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 a better measure of goodness of fit in regression compared to Residual Sum of Squares (RSS).R-squared measures the proportion of variation in the dependent variable that is explained by the independent variables in the model. It ranges from 0 to 1, with values closer to 1 indicating a better fit.

On the other hand, Residual Sum of Squares (RSS) measures the sum of the squared differences between the actual and predicted values of the dependent variable. While RSS can be used to compare different models, it is not as interpretable as R-squared and does not provide a single summary statistic for model fit.

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:- TSS (Total Sum of Squares) measures the total variance in the dependent variable. It is calculated as the sum of the squared differences between each observed dependent variable value and the mean of all observed dependent variable values.

ESS (Explained Sum of Squares) measures the explained variance in the dependent variable by the regression model. It is calculated as the sum of the squared differences between each predicted dependent variable value (from the regression model) and the mean of all observed dependent variable values.

RSS (Residual Sum of Squares) measures the residual variance in the dependent variable that is not explained by the regression model. It is calculated as the sum of the squared differences between each observed dependent variable value and the corresponding predicted dependent variable value (from the regression model).

The relationship between these three metrics is given by the equation:

TSS = ESS + RSS

What is the need of regularization in machine learning?

Ans:- Regularization is a technique used in machine learning to prevent overfitting, which occurs when a model is too complex and fits the training data too well, leading to poor performance on unseen data.

Regularization adds a penalty term to the loss function used for training the model. This term discourages the model from assigning too much importance to any one feature, thereby reducing the complexity of the model and making it more generalized.

. What is Gini–impurity index?

Ans:- Gini impurity is a measure of the probability of misclassifying a randomly chosen element from a set, if it were randomly labeled according to the class distribution in the set. It is commonly used as a criterion for splitting nodes in a decision tree classifier.

Gini impurity is calculated as the sum of the probabilities of misclassifying an element for all classes, and ranges from 0 (indicating perfect classification) to 0.5 (indicating a pure random classification). In a binary classification problem, the Gini impurity for a set is given by:

Gini Impurity = 1 - (p0^2 + p1^2)

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

Ans{:- Yes, unregularized decision trees are prone to overfitting.

Unregularized decision trees are constructed by repeatedly splitting the data based on the feature that provides the largest information gain. This process continues until some stopping criterion is met, such as a maximum tree depth or a minimum number of samples in a node.

Since the tree structure is determined by the training data, decision trees can easily overfit the data by creating a tree that is too complex and fits the training data too well. This can lead to poor performance on unseen data, as the tree may have learned idiosyncrasies in the training data that are not representative of the underlying relationship between the features and target variable.

. What is an ensemble technique in machine learning?

Ans:- An ensemble technique in machine learning is a method that combines multiple models to improve the overall performance of the model. The idea behind ensemble methods is that by combining the predictions of multiple models, the ensemble can produce a more accurate and robust prediction than any individual model.

There are several types of ensemble methods, including:

Bagging (Bootstrapped Aggregating): Bagging is an ensemble technique that trains multiple independent models on different subsets of the training data, with each subset chosen randomly with replacement. The predictions of the individual models are then combined to produce the final prediction.

Boosting: Boosting is an ensemble technique that trains multiple models in a sequential manner, with each model attempting to correct the mistakes made by the previous model. The predictions of the individual models are then combined to produce the final prediction.

Random Forests: Random forests is an ensemble technique that trains multiple decision trees on different subsets of the training data and features, and combines the predictions of the individual trees to produce the final prediction.

Stacking: Stacking is an ensemble technique that trains multiple models and uses their predictions as features for training a meta-model, which makes the final prediction.

What is the difference between Bagging and Boosting techniques?

Ans:- Bagging and Boosting are two ensemble learning methods used in machine learning to improve the performance of a base classifier.

Bagging stands for Bootstrap Aggregating and it involves training multiple models independently on different random subsets of the training data. The final prediction is obtained by combining the predictions of all models through voting or averaging. Bagging reduces overfitting by aggregating the results of several models, each of which overfits the data to some extent.

Boosting, on the other hand, trains multiple models sequentially, where each model tries to correct the mistakes of the previous model. Unlike Bagging, Boosting gives more weight to samples that are misclassified by previous models, so that the subsequent models focus more on those samples. The final prediction is obtained by combining the weighted predictions of all models. Boosting can lead to overfitting if not stopped at the right number of models.

In summary, Bagging aims to reduce overfitting by training models independently, while Boosting aims to improve the performance by correcting the mistakes of previous models.

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

Ans:- Out-of-bag (OOB) error is an estimate of the error of a random forest model that is calculated using only the samples that were not included in the training set for each individual tree. It provides an efficient way to evaluate the performance of a random forest model without the need for a separate validation set. The OOB error provides a measure of the generalization performance of the model, which is a good indicator of how well the model will perform on new, unseen data.

. What is K-fold cross-validation?

Ans:- K-fold cross-validation is a technique for evaluating the performance of machine learning models by dividing the data into "k" folds (or partitions), where k is a user-specified number. The model is trained on k-1 folds of the data and evaluated on the remaining one. This process is repeated k times, with each fold serving as the test set once. The performance of the model is then averaged across all k iterations to provide a more robust estimate of its generalization performance. K-fold cross-validation is a useful technique for reducing the variance of model performance estimates and can provide more reliable results than using a single train/test split.

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

Hyperparameter tuning is done to improve the performance of a machine learning model by finding the best combination of hyperparameters. The choice of hyperparameters can have a significant impact on the performance of a model, and the optimal values may not be obvious ahead of time. Hyperparameter tuning involves exploring a range of possible hyperparameter values and evaluating the performance of the model on a validation set for each combination. The goal is to find the set of hyperparameters that result in the best performance on the validation set, which is then used to make predictions on new, unseen data.

Hyperparameter tuning is an important step in the machine learning process, as it can result in a significant improvement in the performance of the model and lead to better results on new, unseen data

. 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 problems:

Oscillation: The model's cost function may oscillate back and forth across the minimum rather than converge to it, resulting in slow convergence or failure to converge altogether.

Overshooting the minimum: The model may overshoot the minimum and diverge instead of converging, especially when the cost function is highly non-linear.

Slow convergence: A large learning rate can cause the model to converge too quickly, jumping over the optimal solution and settling on a sub-optimal solution.

No convergence: If the learning rate is too large, the model may never converge to a minimum, instead constantly bouncing around and never settling on a solution.

To avoid these issues, it's important to carefully select a learning rate that is not too small (which would lead to slow convergence) or too large (which would result in instability). A commonly used technique is to start with a large learning rate and gradually decrease it over time, until the model starts to converge. This is known as learning rate decay.

In summary, the learning rate is a crucial hyperparameter in gradient descent, and a proper selection of the learning rate can greatly affect the speed and stability of convergence.

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

Ans:- Logistic regression is a linear classifier and is typically used for classification problems where the relationship between the features and the target is assumed to be linear. It models the relationship between the features and the target by using a linear combination of the input features and a set of weights, which are learned during training.

In cases where the relationship between the features and the target is non-linear, logistic regression may not produce accurate results as it is limited by its linear assumption. In such cases, other machine learning algorithms, such as decision trees, random forests, support vector machines, or neural networks, may be more appropriate as they can capture more complex non-linear relationships.

However, there are some techniques that can be used to extend logistic regression to handle non-linear relationships, such as adding polynomial features or transforming the input features in some way. These techniques can make logistic regression more flexible and allow it to capture non-linear relationships to some extent, but other machine learning algorithms may still be preferred for more complex data.

In summary, logistic regression is typically not suitable for classification of non-linear data, as it assumes a linear relationship between the features and the target. Other machine learning algorithms that can handle non-linear relationships may be more appropriate in these cases.

. Differentiate between Adaboost and Gradient Boosting?

AdaBoost and Gradient Boosting are two popular boosting algorithms used for improving the performance of machine learning models.

AdaBoost (Adaptive Boosting) is a meta-algorithm that combines multiple weak learners (e.g., simple decision trees) to form a strong learner that is capable of achieving higher accuracy than any of the individual weak learners. In AdaBoost, the weights of the training samples are adjusted after each iteration so that the samples that are misclassified receive higher weights, and the weak learners are trained on the updated sample weights. This process is repeated multiple times, and the final model is a weighted combination of all the weak learners.

Gradient Boosting, on the other hand, is a type of boosting algorithm that trains weak learners in a stagewise manner, where each weak learner tries to correct the mistakes of the previous one. The weak learners can be decision trees or any other machine learning model, and the objective is to minimize the loss function (e.g., mean squared error) by adding the weak learners sequentially. The gradient of the loss function is used to update the parameters of the weak learners in each iteration.

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

Ans:- The bias-variance tradeoff is a fundamental concept in machine learning that refers to the tradeoff between the model's ability to fit the training data well (low bias) and its ability to generalize well to new, unseen data (low variance).

Bias refers to the error that is introduced by approximating the real relationship between the features and the target. High bias models tend to oversimplify the problem, resulting in a model that is too simple to fit the training data well. This can lead to underfitting, where the model is not able to capture the underlying patterns in the data.

Variance, on the other hand, refers to the error that is introduced by the model's sensitivity to small fluctuations in the training data. High variance models tend to over-complicate the problem, resulting in a model that fits the training data too closely, but is not able to generalize well to new, unseen data. This can lead to overfitting, where the model becomes too specific to the training data and is not able to generalize to new data.

The goal in machine learning is to find a model that has low bias and low variance, as this would result in a model that fits the training data well and generalizes well to new data. However, this is not always possible, as increasing the model's complexity (reducing bias) usually results in increased variance, and reducing the model's complexity (reducing variance) usually results in increased bias.

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

Ans:- Kernels are a key component of Support Vector Machines (SVM) that are used to transform the input data into a higher-dimensional space where a linear boundary can be found to separate the classes. The choice of kernel can greatly affect the performance of an SVM model.

Linear Kernel: The linear kernel is the simplest and most straightforward kernel, which is used when the relationship between the features and the target is linear. The linear kernel computes the dot product of the input features and does not transform the input data into a higher-dimensional space.

Radial Basis Function (RBF) Kernel: The RBF kernel is a non-linear kernel that is widely used in SVM. The RBF kernel maps the input data into a higher-dimensional space and allows for the creation of non-linear boundaries between classes. It is based on the distance of each input feature from a fixed center, which is represented as a radial basis function.

Polynomial Kernel: The polynomial kernel is another non-linear kernel that is used in SVM. It transforms the input data into a higher-dimensional space by computing polynomials of the input features. The degree of the polynomial and the coefficients can be adjusted to control the complexity of the model.