Machine Learning

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: Both R-squared and Residual Sum of Squares (RSS) are usually used for measuring the goodness of fit of model in regression analysis, but they have slightly different purposes.

- R-squared is generally considered as a better measure of goodness of fit of model in regression analysis because it helps us in understanding the overall performance of the model in explaining the variation of the dependent variable. Moreover, it ranges from 0 to 1 making it easy to compare different models. But it does not reveal anything about the magnitude of errors in the predictions made by the model.

- Residual Sum of Squares (RSS) measures the magnitude of errors or the difference between the observed values and the values predicted by the model for the dependent variable. Its value depends on the scale of the dependent variable which makes it difficult to compare different models. It is useful when we want to measure the accuracy of individual predictions.

The choice between these two measures depends on the goals of the model analysis.

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: TSS (Total Sum of Squares), ESS (Explained Sum of Squares) and RSS (Residual Sum of Squares) are all metrics used while measuring the goodness of fit of a model in regression analysis. They are explained as below:

- TSS (Total Sum of Squares): It measures the total variance in the dependent variable that the model attempts to explain. TSS is the sum of the squared differences between each observed value of the dependent variable and the mean value of the dependent variable.

- ESS (Explained Sum of Squares): It measures the variance in the dependent variable that is explained by the model built. ESS it the sum of the squared differences between the predicted values of the dependent variable and the mean of the dependent variable.

- RSS (Residual Sum of Squares): It measures the variance in the dependent variable that is unexplained by the model. RSS is the sum of the squared differences between the observed values and the values predicted by the model for the dependent variable.

The sum of the explained variance and the unexplained variance of the dependent variable for the regression model is the total variance of the dependent variable. This relationship is represented by the following equation:

TSS = RSS + ESS

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

Answer: Regularization is used in machine learning to prevent overfitting of the model on the training data and improve the generalization of the model. Regularization adds a penalty term to the model's objective function thus decreasing overfitting. It improves the performance and stability of the model. The benefits and advantages of regularization are:

- Prevents overfitting: Regularization procedures add penalty to the model's objective function coefficients. This discourages complex models, promotes simple models and prevents overfitting.

- Overcome multicollinearity problem: When independent variables in a regression model are highly correlated with each other, this leads to the problem of multicollinearity. Regularization reduces the effects of multicollinearity by shrinking the coefficients towards zero.

- Generalization to unseen data: Regularization reduces the model's complexity and helps it to generalize to unseen data. This enables the model to perform better on new and unseen data.

- Resilience to outliers: Regularization makes the models more resilient to outliers and noise by penalizing large coefficients influenced by outliers. This results in better predictions by model even with noisy data.

- Balance between bias and variance: Regularization adds a penalty term to the objective function of the model and makes a balance between the bias which is introduced by simplifying the model and the variance in the model.

4. What is Gini–impurity index?

Answer: The Gini impurity index is mostly used in decision tree algorithms for classification problems. It quantifies how often a randomly chosen element from the set would be incorrectly classified. For a given node t with K classes in the decision tree, the Gini impurity index is calculated as below:

$$Gini(t) = 1 - \sum_{i=1}^{K} p(i)^2$$

In the above formula p(i) represents the probability of choosing a class i at node t. The Gini impurity index ranges from 0 to 1 where 0 indicates perfect purity which means all elements belong to the same class and 1 indicates maximum impurity which means elements are evenly distributed across all classes. The Gini impurity index is typically used in CART (Classification and Regression Trees) algorithms.

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

Answer: Yes. Unregularized decision trees are prone to overfitting. This is because of the following reasons.

1. High variance in training data: Decision trees can capture very small details and patterns in the training data. When the training data has high variance with outliers and noise, the decision trees can become highly complex and deep. This will lead to overfitting the training data and reduce the model's ability to generalize to unseen data.

2. Lack of pruning: When no pruning is applied on decision trees, they will continue to grow deeper and wider until they perfectly fit the training data. This will lead to capturing of all the noise and irrelevant features in the training data and result in overfitting.

3. Memorization of data: Decision trees can memorize training data by creating branches and leaves to perfectly fit every instance of the training data. In unregularized decision trees, this will lead to unwanted memorization of irrelevant training data. This will lead to overfitting of training data and poor performance on unseen data.

4. Sensitivity: Decision trees are often sensitive to small changes in the training data. This will create different tree structures for even for slightly varying data sets. In unregularized decision trees, this sensitivity will lead to capturing of the noise and random fluctuations in the training data and will finally lead to overfitting.

6. What is an ensemble technique in machine learning?

Answer: An ensemble technique in machine learning refers to the process of combining multiple machine learning models called as base models or weak learners to build a more powerful model with higher accuracy. The ensemble technique aims to mitigate the errors or biases that may exist in individual models by leveraging the collective wisdom of multiple models. The idea behind ensemble techniques is that by combining different models, the strengths of one model can compensate for the weaknesses of another model thereby improving the accuracy and stability of the algorithm. Some examples of ensemble techniques are Bagging or Bootstrap Aggregating, Boosting, Stacking, Voting, Blending, etc. The most popular algorithms that utilize ensemble techniques are Random Forest, AdaBoost (Adaptive Boosting), Gradient Boosting Machines (GBM), XGBoost (Extreme Gradient Boosting) and CatBoost.

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

Answer: Bagging (also called as Bootstrap Aggregating) and Boosting are ensemble techniques used in machine learning. They follow different approaches and methods for combining multiple models. The main differences between them are:

1. Training approach: In Bagging, the same base learning model is trained on different subsets of training data independently. The final prediction is made by aggregating the predictions of all the base models. In Boosting, the base models are trained

sequentially with each subsequent model focusing on the elements that were misclassified by the previous model. The final prediction is made by adding the weighted predictions of all the base models.

2. Weightage of base models: In Bagging, each base model is given equal weightage in the final predictions. In regression, the final prediction is made by averaging the predictions of all the base model and in classification a majority vote is used. In Boosting, the base models are given weightage according to their performance. The weighted predictions of all the base models are combined to get the final predictions.

3. Parallel or sequential: Since the base models are trained independently in Bagging, each base model can be trained in parallel. This makes bagging algorithms highly scalable. Bagging is sequential as each subsequent model depends on inputs from the previous model. So, boosting can't be done in parallel.

4. Overfitting: Bagging is effective in reducing overfitting as it averages out the predictions of multiple models. It tends to produce more stable and less overfitted data. Boosting, on the other hand, tries to improve model performance by sequentially correcting the errors made by previous models. This may sometimes lead to overfitting.

5. Variance and bias: Bagging aims to decrease variance whereas Boosting aims to decrease the bias in the classifier.

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

Answer: In Random Forests, the data points are randomly selected with replacement from the original data set to create multiple subsets of training data for training each decision tree. For a decision tree, the data points from the original dataset that are not included in its training are called as out-of-bag (OOB) samples.

In Random Forests, the out-of-bag (OOB) error is an estimate of the model's performance on unseen data. The out-of-bag (OOB) error is calculated by evaluating each data point in the data set using all the decision trees for which it is an OOB sample. It is computed as the average prediction error over all OOB samples. For example, mean squared error for regression and misclassification rate for classification. The advantage of analysing with OOB error is that it provides an unbiased estimate of the model's performance without the need for cross-validation.

9. What is K-fold cross-validation?

Answer: K-fold cross-validation is a commonly used method for evaluating a machine learning model's performance. It estimates how well the model will generalize to an unseen data set. By ensuring that every data point is used for both training and validation, it provides a robust estimate of a model's performance.

In K-fold cross-validation, the original data set is randomly split into K equally sized subsets or folds with approximately the same size and a roughly equal distribution of the target variable. Then, one of the K folds is used as the validation set and the remaining K-1 folds are used as the training set. The model is trained on the training set and validated on the validation set. This is repeated K times for each fold and K performance metrics are obtained. These

metrics are then averaged to obtain the overall performance metric of the model. One of the advantages of K-fold cross-validation is that it helps in detecting overfitting.

K-fold cross-validation can also be used for hyperparameter tuning. Different sets of hyperparameters can be tested in each iteration, and the set with the best performance can be selected.

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

Answer: Hyperparameter tuning is the process of optimizing the hyperparameters of a machine learning model that control the model training process, to improve its performance. Hyperparameters are settings that control the model training process such as the learning rate of the model, the kernel size in a support vector machine or the number of neurons in a neural network, etc. Hyperparameter tuning is done for the following reasons:

1. The hyperparameters chosen during the model training process can significantly affect the model's performance. Tuning these hyperparameters will lead improved accuracy and performance of the model.

2. Hyperparameters determine the complexity of a model and its ability to capture the patterns in the data. Hyperparameter tuning attempts to make a balance between capturing too much of patterns which is called as overfitting and not capturing enough of patterns which is called as underfitting.

3. Optimizing the hyperparameters leads to generalization of the model to new and unseen data.

4. Tuning the hyperparameters that control the use of computational resources required for training a model leads to optimum utilization of available computational resources.

5. Tunning the hyperparameters can lead to models that are easily interpretable and more stable.

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

Answer: A large learning rate in Gradient Descent can cause the following problems:

1. A large learning rate in Gradient Descent can cause the algorithm to overshoot the minimum of the loss function.

2. In some cases, a large learning rate may prevent the optimization algorithm from converging altogether. It may even diverge.

3. The optimization algorithm may become unstable and erratic with the loss bouncing around without converging to a stable value.

4. With an unstable optimization algorithm, the resulting model may lack generalization to unseen data. This is a result of the model parameters not tuned properly to capture the patterns in the data.

5. A large learning rate can make the optimization algorithm highly sensitive to initial values of the model parameters.

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

Answer: Logistic Regression is a linear classification algorithm. It models the relationship between the features and the label using a linear decision boundary which is a hyperplane that separates feature classes. Therefore, Logistic Regression can not be used for classification of non-linear data. If we use Logistic Regression directly on non-linear data, it will fail to separate the classes using a straight line or a hyperplane and it would result in poor performance of the model.

However, it is possible to use Logistic Regression indirectly on non-linear data. This is done by using techniques such as feature engineering or transforming the input features into higher dimensions using methods like polynomial features or kernel methods. The original feature space is mapped into a higher dimensional space where the feature classes become linearly separable. Now, Logistic Regression can be effectively used on the transformed features.

13. Differentiate between Adaboost and Gradient Boosting.

Answer: AdaBoost (Adaptive Boosting) and Gradient Boosting are both popular ensemble machine learning techniques used in a variety of problems. The basic differences between the two are as follows:

1. Approach: AdaBoost works on the principle of improving the predictions by giving more weights to misclassified data points. Gradient Boosting, on the other hand, improves the predictions by fitting each new model to the errors or residuals of the previous model.

2. Loss function and outliers: AdaBoost tries to minimise the exponential loss function which makes the algorithm sensitive to outliers. Gradient Boosting uses various loss functions, such as squared error loss for regression and exponential loss for classification. It is more robust to outliers than AdaBoost.

3. Weights assignment: AdaBoost increases the weights of incorrectly classified instances so that subsequent weak learners pay more attention to them. Gradient Boosting doesn't assign different weights to data points but rather updates the model parameters based on the gradient of the loss function with respect to the model's prediction.

4. Base learners: AdaBoost typically uses decision trees with a single node and two leaves called decision stumps as its base learners whereas Gradient Boosting uses various base learners such as decision trees, regression trees or even simpler models like linear regression.

5. Classification or regression problems: AdaBoost was primarily designed for binary classification problems and can improve the efficiency of decision trees. Gradient Boosting is used to crack the problems with differential loss functions. It can be used in both classification and regression problems.

6. Parallel processing: AdaBoost is less parallelizable because each subsequent model depends on the performance of the previous one. Gradient Boosting can be parallelized to a greater extent than AdaBoost as each new model is independent of the others and its only fitted to the residuals of the previous model.

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

Answer: The bias-variance tradeoff is a fundamental concept in machine learning which is related to the performance of the machine learning models.

- Bias: Bias represents the error introduced by a model's assumptions by attempting top simplify it. A model with high bias tends to be too simplified and is unable to capture the true underlying patterns in the data. This often leads to underfitting which means the model will perform poorly on both training and new or unseen data as it oversimplifies the problem.

- Variance: Variance refers to a model's sensitivity to the noise, fluctuations or outliers in the data. A model with high variance will be too complex and will memorize the noise in the training data. High variance leads to overfitting and the model will perform very well on the training data but poorly on new or unseen data as it has captured the noise in the training data.

- Bias-variance tradeoff: Bias-variance tradeoff is about finding the correct balance between simplicity and complexity in a model. A model is built in such a way that neither it is too simple (high bias) nor too complex (high variance). The goal of machine learning is to strike a balance between bias and variance so that it performs well on the training data as well as it generalizes well on unseen data. This is usually achieved through various techniques like hyperparameter tuning, model selection, cross validation and regularization.

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

Answer: The short descriptions of Linear, RBF and Polynomial kernels used in SVM (Support Vector Machine) are as below:

- Linear Kernel: A linear kernel is the simplest and most commonly used function in SVM. It calculates the dot product between the input feature vectors which is a measure of their similarity or distance in the original feature space. In the case of linear kernels, the decision boundary is a linear hyperplane which separates the classes in the feature space. So, a linear kernel is particularly useful in the case of linearly separable data or when the number of features is high.

- RBF Kernel: The RBF (Radial Basis Function) kernel also called as Gaussian kernel is a non-linear kernel function and a popular choice in SVMs when the data is not

linearly separable. It maps the input features into a high-dimensional space using a Gaussian function. It has two parameters: gamma (γ) and C, where gamma is a parameter that controls the width of the Gaussian function and C controls the trade-off between maximizing the margin and minimizing the classification error. One advantage of the RBF kernel is that it can capture complex relationships in the data without the need for feature engineering. However, the choice of the gamma parameter can be challenging, as a smaller value may result in underfitting, while a larger value may result in overfitting.

- Polynomial kernel: A polynomial kernel function is a non-linear kernel function that uses polynomial functions to transfer the input data into a higher-dimensional feature space. It computes the dot product of the input vectors elevated to the degree of the polynomial. The decision boundary of an SVM with a polynomial kernel can capture more complex relationships. But the degree of the polynomial should be properly selected as a high degree can lead to overfitting and a low degree might not adequately represent the underlying relationships in the data. The polynomial kernel has the benefit of detecting both linear and nonlinear relationships in the data.

Statistics

1: Using a goodness of fit, we can assess whether a set of obtained frequencies differ from a set of frequencies.
a) Mean
b) Actual
c) Predicted
d) Expected

Answer: d) Expected

2: Chisquare is used to analyse
a) Score
b) Rank
c) Frequencies
d) All of these

Answer: c) Frequencies

3: What is the mean of a Chi Square distribution with 6 degrees of freedom?
a) 4
b) 12
c) 6
d) 8

Answer: c) 6

4: Which of these distributions is used for a goodness of fit testing?
a) Normal distribution
b) Chisqared distribution
c) Gamma distribution
d) Poission distribution

Answer: b) Chisquared distribution

5: Which of the following distributions is Continuous
a) Binomial Distribution
b) Hypergeometric Distribution
c) F Distribution
d) Poisson Distribution

Answer: c) F Distribution

6: A statement made about a population for testing purpose is called?
a) Statistic
b) Hypothesis
c) Level of Significance
d) TestStatistic

Answer: b) Hypothesis

7: If the assumed hypothesis is tested for rejection considering it to be true is called?
a) Null Hypothesis
b) Statistical Hypothesis
c) Simple Hypothesis
d) Composite Hypothesis

Answer: a) Null Hypothesis

8: If the Critical region is evenly distributed then the test is referred as?
a) Two tailed
b) One tailed
c) Three tailed
d) Zero tailed

Answer: a) Two tailed

9: Alternative Hypothesis is also called as?
a) Composite hypothesis
b) Research Hypothesis

c) Simple Hypothesis
d) Null Hypothesis

Answer: b) Research Hypothesis

10: In a Binomial Distribution, if ‘n’ is the number of trials and ‘p’ is the probability of success, then the mean value is given by _____
a) np
b) n

Answer: a) np