Worksheets_set5

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?

R-squared (R^2) and Residual Sum of Squares (RSS) are both commonly used measures of goodness of fit in regression analysis, but they have different purposes.

R-squared is a statistical measure that represents the proportion of variance in the dependent variable that is explained by the independent variable(s) in the model. It ranges from 0 to 1, where a value of 1 indicates that the model explains all the variation in the dependent variable, and a value of 0 indicates that the model does not explain any of the variation in the dependent variable. R-squared is often used to evaluate the overall performance of a model, and a higher R-squared indicates a better fit.

On the other hand, the Residual Sum of Squares (RSS) measures the difference between the actual values of the dependent variable and the predicted values from the model. It represents the total amount of unexplained variation in the dependent variable. The goal in regression analysis is to minimize the RSS, which means that the model is making accurate predictions.

While both measures are useful in evaluating the performance of a regression model, they have different strengths and weaknesses. R-squared is a more intuitive measure, as it provides an easy-to-understand percentage of how well the model fits the data. However, it can be misleading if the model is overfitting the data, as it can give a high R-squared even if the model is not a good predictor of new data.

On the other hand, the RSS is a more robust measure that reflects the accuracy of the model in making predictions, regardless of the complexity of the model. However, it is harder to interpret than R-squared and does not provide an intuitive understanding of the model's performance.

In summary, R-squared and RSS are both important measures of goodness of fit in regression analysis, and they should be used together to provide a comprehensive evaluation of the model's performance. R-squared can provide an intuitive understanding of the model's overall performance, while RSS can provide a more accurate reflection of the model's accuracy in making predictions.

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.

In regression analysis, the Total Sum of Squares (TSS), Explained Sum of Squares (ESS), and Residual Sum of Squares (RSS) are metrics used to assess the goodness of fit of a regression model.

TSS represents the total variability in the response variable, which can be decomposed into two parts: the explained variability (ESS) and the unexplained variability (RSS). ESS measures the amount of variability in the response variable that is explained by the regression model, while RSS measures the amount of variability that is not explained by the model.

The equation relating these three metrics is as follows:

TSS = ESS + RSS

where TSS is the total sum of squares, ESS is the explained sum of squares, and RSS is the residual sum of squares.

Mathematically, TSS can be expressed as the sum of the squared differences between each observation and the mean of the response variable. ESS can be expressed as the sum of the squared differences between the predicted values and the mean of the response variable, while RSS can be expressed as the sum of the squared differences between the observed values and the predicted values.

In summary, TSS measures the total variability in the response variable, ESS measures the amount of variability in the response variable that is explained by the model, and RSS measures the amount of variability that is not explained by the model. The equation TSS = ESS + RSS is used to relate these three metrics.

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

Regularization is a technique used in machine learning to prevent overfitting of a model to the training data. Overfitting occurs when a model is too complex, and it captures the noise or random fluctuations in the training data instead of the underlying patterns or relationships.

Regularization techniques add a penalty term to the loss function that the model is minimizing during training. The penalty term discourages the model from fitting the training data too closely, which helps it to generalize better to unseen data.

There are various types of regularization techniques, such as L1 regularization, L2 regularization, and dropout. L1 regularization adds a penalty proportional to the absolute value of the model's parameters, while L2 regularization adds a penalty

proportional to the square of the parameters. Dropout randomly drops out some of the neurons during training, which helps to prevent the model from relying too heavily on any particular set of neurons.

In summary, regularization is needed in machine learning to prevent overfitting and improve the generalization performance of the model.

4. What is Gini–impurity index?

The Gini impurity index is a measure of the impurity or randomness of a set of categorical data. It is commonly used in decision trees to determine the best split for a given node.

The Gini impurity measures the probability of misclassification of a randomly chosen element in the set. It ranges from 0 to 1, where 0 indicates a perfectly pure set (all elements belong to the same class) and 1 indicates a completely impure set (the elements are equally distributed among all classes).

To calculate the Gini impurity index, we sum the probabilities of each class being chosen squared, and subtract the sum from 1. Mathematically, it can be expressed as follows:

Gini impurity = $1 - \Sigma (p_i)^2$

where p_i is the probability of an element belonging to the ith class.

In decision trees, the Gini impurity index is used to determine the best split for a given node. The split with the lowest Gini impurity is chosen as it results in the highest purity in the resulting nodes.

Decision trees are a type of machine learning algorithm that create a model by recursively splitting the data into smaller subsets, based on the values of the input features, until a stopping criterion is met. This process continues until a set of decision rules is created that can be used to make predictions.

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

Unregularized decision trees are constructed by splitting the data into subsets in a way that maximizes the information gain at each node. This can lead to the creation of complex decision rules that are tailored to the training data, but may not generalize well to new data. As a result, unregularized decision trees are prone to overfitting, where the model fits the training data too closely and does not perform well on new, unseen data.

Regularization techniques can be used to reduce the likelihood of overfitting. One such technique is called pruning, which involves removing branches from the tree that do not improve the model's performance on a validation set. Another technique is to limit the depth of the tree, or the number of samples required to create a split. These approaches help to simplify the model and prevent it from overfitting to the training data.

6. What is an ensemble technique in machine learning?

Ensemble Methods, what are they? Ensemble methods is a machine learning technique that combines several base models in order to produce one optimal predictive model. To better understand this definition lets take a step back into ultimate goal of machine learning and model building. This is going to make more sense as I dive into specific examples and why Ensemble methods are used.

I will largely utilize Decision Trees to outline the definition and practicality of Ensemble Methods (however it is important to note that Ensemble Methods do not only pertain to Decision Trees).

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

Bagging is a technique for reducing prediction variance by producing additional data for training from a dataset by combining repetitions with combinations to create multi-sets of the original data. Boosting is an iterative strategy for adjusting an observation's weight based on

the previous classification. It attempts to increase the weight of an observation if it was erroneously categorized. Boosting creates good predictive models in general.

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

Out-of-bag (OOB) error in random forests is an estimate of the performance of a random forest model on unseen data.

Random forests are an ensemble learning method that combines multiple decision trees to make predictions. Each decision tree is trained on a bootstrap sample of the original dataset, which means that some data points are not included in the sample. The OOB data points are the ones that are not included in the bootstrap sample for a given tree.

The OOB error is calculated by using the OOB data points to evaluate the predictions of the decision tree that was trained without them. For each OOB data point, the majority vote of the decision trees that were not trained on that point is used to make a prediction. The OOB error is then the average difference between the predicted value and the true value for all OOB data points.

By using the OOB error as an estimate of model performance, random forests can avoid the need for a separate validation dataset. This can be particularly useful in situations where the dataset is small, and it is not possible to set aside a portion of the data for validation without losing too much information.

9. What is K-fold cross-validation?

K-fold Cross-Validation is when the dataset is split into a K number of folds and is used to evaluate the model's ability when given new data. K refers to the number of groups the data sample is split into. For example, if you see that the k-value is 5, we can call this a 5-fold cross-validation. Each fold is used as a testing set at one point in the process.

In machine learning, we need to differentiate between parameters and hyperparameters. A learning algorithm learns or estimates model parameters for the given data set, then continues updating these values as

it continues to learn. After learning is complete, these parameters become part of the model. For example, each weight and bias in a neural network is a parameter.

Hyperparameters, on the other hand, are specific to the algorithm itself, so we can't calculate their values from the data. We use hyperparameters to calculate the model parameters. Different hyperparameter values produce different model parameter values for a given data set.

Hyperparameter tuning consists of finding a set of optimal hyperparameter values for a learning algorithm while applying this optimized algorithm to any data set. That combination of hyperparameters maximizes the model's performance, minimizing a predefined loss function to produce better results with fewer errors. Note that the learning algorithm optimizes the loss based on the input data and tries to find an optimal solution within the given setting. However, hyperparameters describe this setting exactly.

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

Logistic regression is known and used as a linear classifier. It is used to come up with a hyper*plane* 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 to use logistic regression as a linear classifier.

While logistic regression makes core assumptions about the observations such as IID (each observation is independent of the others and they all have an identical probability distribution), the use of a linear decision boundary is *not* one of them. The linear decision

boundary is used for reasons of simplicity following the Zen mantra – when in doubt simplify. In those cases where we suspect the decision boundary to be nonlinear, it may make sense to formulate logistic regression with a nonlinear model and evaluate how much better we can do. That is what this post is about. Here is the outline. We go through some code snippets here but the full code for reproducing the results can be downloaded from.

13. Differentiate between Adaboost and Gradient Boosting.

Adaboost and gradient boosting are types of ensemble techniques applied in machine learning to enhance the efficacy of week learners. The concept of boosting algorithm is to crack predictors successively, where every subsequent model tries to fix the flaws of its predecessor. Boosting combines many simple models into a single composite one. By attempting many simple techniques, the entire model becomes a strong one, and the combined simple models are called week learners. So the adaptive boosting and gradient boosting increases the efficacies of these simple model to bring out a massive performance in the machine learning algorithm.

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

It is important to understand prediction errors (bias and variance) when it comes to accuracy in any machine learning algorithm. There is a tradeoff between a model's ability to minimize bias and variance which is referred to as the best solution for selecting a value of Regularization constant. Proper understanding of these errors would help to avoid the overfitting and underfitting of a data set while training the algorithm.

Bias

The bias is known as the difference between the prediction of the values by the ML model and the correct value. Being high in biasing gives a large error in training as well as testing data. Its recommended that an algorithm should always be low biased to avoid the problem of underfitting.

By high bias, the data predicted is in a straight line format, thus not fitting

accurately in the data in the data set. Such fitting is known as Underfitting of Data. This happens when the hypothesis is too simple or linear in nature. Refer to the graph given below for an example of such a situation.