MACHINE LEARNING

Rajib Dutta (duttarajib78@gmail.com) - Batch DS2402

March 16, 2024

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?

R-squared (R^2) and Residual Sum of Squares (RSS) are the two metrics used to assess the goodness of fit of regression models. While RSS is a general measure of goodness of fit for any regression model R^2 is specific to family of linear regression models only (i.e. where a linear relationship is assumed between dependent and independent variables).

1.1 R-squared (R^2):

R^2 measures the proportion of variance in dependent variable explained by the independent variables assuming dependent variable is linearly associated with the independent variables. It ranges from 0 to 1 where 0 indicates no linear association of the dependent variable with the independent variables and 1 indicates a perfect linear association. However, this measure is sensitive to the model complexity meaning that it keeps on increasing as model complexity increases thus making it a non ideal measure to detect model overfitting. However, there is another measure called Adjusted R-squared where the R^2 value is penalized by model complexity. Thus if an additional independent variable is not significantly associated with the dependent variable then including that variable into the linear regression model will increase the R^2 value but decrease the Adjusted R^2 value indicating a possible overfitting.

1.2 Residual Sum of Squares (RSS):

Residual sum of squares (RSS) is the total of squares of the prediction errors for the entire dataset on which the metric is computed. This metric is not just applicable for a linear regression model but for any regression model (linear as well as non-linear). This is a measure of on a quadratic scale how much the predictions are off in total from the ground reality and thus serves as a goodness of fit measure for any regression model. Lower value indicates a better fit.

Therefore, in conclusion if the model in question is a linear regression model then we can use R^2 to assess the goodness of fit even though Adjusted R^2 should ideally be used in conjunction with R^2 to assess possibility of any overfitting. Additionally, we normally use other metrics such as RSS or Mean absolute error (MAE) to complement the model assessment and/or selection process. However, if the model is not a linear regression model then we should not use R^2 and Adjusted R^2 to assess the goodness of fit; instead we should only use more general metrics such as RSS and/or MAE for this purpose. Hence, it is hard to compare as to which metric is better as they are used in very different contexts. What could be concluded is that R^2 is a very specialized metric only relevant for linear regression models while RSS is a general metric which has much wider use in general.

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.

As mentioned earlier, Total sum of square (TSS), Explained Sum of Squares (ESS) and Residual Sum of Squares (RSS) are three measures relevant to regression models in general. However, it can be mathematically proved that the following identity involving these three measures only holds for linear regression models and not regression models other than linear regression.

$$TSS = ESS + RSS$$

Each measures have been explained in brief as below.

2.1 Total Sum of Square (TSS)

Let's assume that we have a regression problem in hand meaning the target variable (y) is continuous in nature. In the absence of any model, our best prediction for y would be the average value of y i.e. $\bar{y}$. With this setup the total sum of squared error or also TSS would be as below.

$$TSS = \sum_{i \epsilon \mathbb{N}} (y_i - \bar{y})^2$$

where y_i is the i^{th} observation of y in the dataset.

2.2 Explained Sum of Squares (ESS)

Now let's assume that we have built a linear regression model of y on the selected features or independent variables. If $\hat{y}_i$is the prediction of y for the i^{th} observation then explained sum of squared ESS would be the following.

$$ESS = \sum_{i\epsilon\mathbb{N}} (\hat{y}_i - \bar{y})^2$$

This measures the amount of variance in the target that has been taken care or explained by the model.

2.3 Residual Sum of Squares (RSS)

Finally, this is the total squared of the prediction errors made by the model and is given as below.

$$RSS = \sum_{i\epsilon\mathbb{N}} (y_i - \hat{y}_i)^2$$

3 What is the need of regularization in machine learning?

In machine learning, the learning process is a mathematical optimization where some cost function which is a function of the model parameters computed on the training sample is minimized by tuning the model parameters. In this process it may so happen that the parameters are tuned to such an extent to achieve the minima in the cost function that the model learns too much details specific to the training dataset (or in other words the noise or randomness specific to the training sample). This leads to poor generalization or in other words extremely low prediction error on training sample but very high and inconsistent prediction error on unseen datasets also known as overfitting. One of the ways to deal with this is to make the model simpler. But if we keep the model too simple then the parameter optimization leads to a trained model where the prediction errors are way off on both training sample as well as out of sample or unseen data leading to underfitting. Theoretically, there exists a point between these two extremes where the prediction error are optimally low on both training sample and unseen data (out of sample). This point is called the right fit and is extremely hard to achieve manually. Hence, a technique called regularization was introduced to help achieve the point of right fit. In this technique a penalty term is added to the cost function where the penalty term is also a function of the model parameters. This leads to zero or very low estimated values of the model parameters associated with the predictors which have insignificant effects on the target and thus yielding a model close to the point of right fit. There are in general two types of regularization; L1 and L2. The regularization terms are as follows.

$L1 \rightarrow \lambda L1(\beta)$ where $L1(\beta)$ is the $L1$ norm of betas.

$L2 \rightarrow \lambda L2(\beta)$ where $L2(\beta)$ is the $L2$ norm of betas.

There are various other types of regularizations depending on the model type. For example, pruning is a popular regularization for decision tree models.

4 What is Gini–impurity index?

Gini-impurity index is a measure of heterogeneity in a dataset in terms of number of observations belonging to various classes. Hence, the Gini index would be highest if the observations are evenly distributed between the classes in the dataset. For example, if in a dataset of customers, half the customers are male and the other half is females then in terms of this gender information, the dataset is most heterogeneous as all the observations do not belong to one particular category but randomly fall into male and female categories with equal probability and hence has highest Gini index (0.5 in this example). On the other hand, on another dataset if we have all either male or female customers then that dataset is completely homogeneous in terms of this gender information and thus has lowest Gini index (0 in this case). It's possible range is $[0, 1]$. If g represents the gini index then it is expressed mathematically as below.

$$g = 1 - \sum_i p_i^2$$

where p_i being the probability of an observation falling into i^{th} class.

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

Yes, unregularized decision trees are prone to overfitting. The reason for that is the fact that the decision trees are grown through recursive splits on the nodes which results in best possible homogeneity measured using entropy or gini index. In worst case a fully grown decision tree results in leafs containing just one sample leading to 100% accuracy on the training sample but extremely poor accuracy on unseen dataset (or out of sample). Hence, pruning is performed which is preventing the tree to grow completely or chopping off the tree at a specific depth. This allows the tree to learn the noises present in the training sample less leading to slightly higher training error but lower out of sample error resulting in better generalization by avoiding overfitting. So correct pruning of decision trees though hyper-parameter tuning is the key to achieve the best fit model with optimum generalization capability.

6 What is an ensemble technique in machine learning?

Ensemble technique in machine learning is a method to combine multiple machine learning models with an aim to improve out of sample prediction accuracy. This leverages the principle of combining multiple expert opinion to make a final decision. This may employ different types of models (such as linear, tree based etc.) or it can use multiple models from the same realm but fitted on different subset of features on different datasets (as in the case of random forest). There are simpler ensemble methods such as majority voting for classification and averaging for regression. Also, many advanced ensemble algorithms exist and are widely used such as stacking, blending, bagging and boosting.

7 What is the difference between Bagging and Boosting techniques?

Bagging and boosting are two different techniques and use two separate principles. Below are the brief explanations of the two.

7.1 Bagging

It uses same algorithm (for example decision tree) to build several models but on different datasets most likely the bootstrap samples (with replacement or without replacement) of the original dataset. It also randomly selects features (with replacement or without replacement depending on the algorithm) from the features set to build individual models. Finally, it aggregates the predictions via majority voting for classification or weighted average for regression to derive the final predictions. The idea behind this method is that if we build multiple models on slightly different datasets and with different sets of features then it is unlikely that all models would make similar mistakes but would be smoothed out the errors made by some models by the others making the prediction more robust.

7.2 Boosting

This is a sequential approach to build a series of models where the observations for which the current model made errors are given greater weight so that the next model is more likely to make correct prediction for those observations. In this way the next model tries to correct the mistakes made by the previous model. In the end predictions from all individual models are combined using weighted mean and that combined prediction performs way better than individual model's predictions this making the combined model a strong predictor.

Hence the major difference is that bagging is a parallel algorithm where models could be trained simultaneously on the bags or the bootstrap samples while boosting is a sequential algorithm where the next model in the sequence can only be trained after the previous model has been already trained and predictions from the model have been made and weighted.

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

In random forest (RF) models, multiple decision tree (DT) models are fitted on bootstrapped sampled datasets. Hence, there would be bootstrap datasets where some data points from the parent dataset where the bootstrapped samples were generated would be left out. If the number of trees in the RF model is large enough then there is a good chance that one particular data point would be left out in multiple bootstrap samples. Thus that data point would serve as a out of bag sample for all the DT models fitted on those bootstrapped samples. We can use that out of bag sample as test sample for all such DT models. So, if we have say k such DT trees then we get k predictions for that out of bag sample. We then take a majority vote on those k predictions to make a final prediction in case of classification or average on k predicted values to come up with the final prediction in case of regression. This out of bag prediction can be done for all rows in the dataset which are left as out of bag in some bootstrapped sample. Then the prediction error computed on these out of bag predictions is called the out-of-bag error in RF models. The advantage is that as out of bag samples are not seen by the RF models where these are used as validation data points, there is no data leakage leading to more reliable predictions.

9 What is K-fold cross-validation?

In machine learning $k - fold$ cross validation is a model validation method which is widely used. In this technique, the entire training dataset is divided into k equal partitions where the data points are selected for each partition randomly without replacement meaning if a data point x_i has been included in j^{th} partition, it can not be re-selected for another partition say m^{th} partition. Then, the model is trained on $k - 1$ partitions and validated on one remaining partition. This process is repeated k times for each partition once resulting in k model scores. The average on these k scores serves as the final validation score for that model. This helps in model algorithm selection where there are multiple candidate algorithms to select from as well as the best hyper-parameter tuned model for same algorithm. Also this method is robust in assessing out of sample error as the performance has been scores on multiple (k) unseen validation datasets.

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

For a specific machine learning (ML) algorithm there could be many possible combinations for hyper-parameters to fit the model with. It's one of the best practices to create a grid of relevant hyper-parameters (or a number of possible combinations of hyper-parameter values forming a hyper-parameter space) and then fit and validate the model for each possible combination. Once that is done, we can select the best model which is the best as per some ore-defined selection criteria (e.g. we can choose the model and hence the corresponding combination of hyper-parameters which produces train and validation errors closest to each other among all scores produced by all the fitted models indicating no over-fitting). This practice of hyper-parameter tuning helps select the model which is more likely to generalize well within the realm of the hyper-parameter space chosen. Commonly, there are two methods of hyper-parameter tuning. First, grid search where one model for each possible combination of hyper-parameters is fitted. This is very comprehensive but time costly. Hence, probably not practical if the parameter space is very large. Second is random search where a predefined number of combinations are sampled at random from the parameter space and the model is fitted on each of the randomly sampled combinations. Another method is Bayesian search where the search uses the past searched model outcomes to sample the next best possible parameter combination. This method is more likely to sample the region of the parameter space which produced better model scores in the past searches as compared to other regions thus converging faster with an optimal parameter combination.

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

If we have a large learning rate in Gradient Descent, it is likely that in some iteration the region close to the local minima in the cost function is skipped and the parameter values jump to the region where further descent is necessary to get to the local minima. In the next iteration it is likely to happen again and hence the parameter values may oscillate around the region of local minima instead of converging to a value close to the local minima. So the optimization does not converge resulting in no optimally trained model.

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

Logistic regression estimates a linear decision boundary to separate two classes. In an n dimensional feature space the algorithm fits an n dimensional hyperplane which best separates the two classes. Hence, in the dataset in hand if the best possible decision boundary is a non linear hyper-surface or even a more complex shape then the linear decision boundary estimated by the logistic regression would be insufficient to adequately classify the data. We might need to fit more complex non-linear model for that (e.g. Decision Tree, Random Forest, Gradient Boosting etc.)

13 Differentiate between Adaboost and Gradient Boosting (GB).

Both Adaboost and Gradient Boosting are boosting algorithms where the predictions are gradually improved with subsequent estimators. However, these two methods have a few intrinsic differences as below.

1. In Adaboost the misclassified observations are assigned greater weights for the subsequent estimator so that it becomes more likely to be picked up during the training while in gradient boosting previous model's limitations are identified by the gradient.

2. Adaboost assigns weights to the observations for training the next estimator as well as the individual classifiers for making final prediction. On the other hand GB fits trees on the previous estimator's residuals.

3. Finally, in Adaboost the individual trees are decision stumps i.e. a tree with a depth of 1 with two leafs whereas in GB the trees are of greater depth with multiple leafs. Also for making final prediction Adaboost creates a weighted average while GB gives equal weights to all trees.

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

In machine learning (or statistical learning theory), the notion of bias-variance is closely related to the model generalization. Generally, if a very simple model is fitted to a dataset where the true underlying pattern is more complex then the model results in high training error as well as high out of sample error (the prediction error the model makes on unseen data), This implies that in general the model will make predictions which are significantly off the ground reality or in other words the models predictions are biased. On the other hand if a very complex model is fitted to the dataset then the model will learn the nish pattern specific to the training dataset resulting in very low training error but high and inconsistent out of sample error. This indicates that the predictions obtained from such a model will be all over i.e. with high variance. It has been proved that as model bias increases or synonymously model complexity decreases, the variance decreases and if variance increases or in other words, model complexity increases bias decreases. Hence, bias and variance counter each other and hence, we need to trade-off between these two to achieve the best result. Theoretically, there exists a sweet spot (some model complexity) between these two extremes for which both training error and out of sample error are optimally low meaning both bias and variance are low. The primary goal of machine learning is to find that complexity for which both bias and variance are optimally low. The model corresponding to that complexity generalizes the best.

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

Kernels are functions generally used in the context of SVM where there doesn't exist a linear decision boundary for the two classes. Kernels transform the inner product of any two given vectors in the feature space to a higher dimensional space where it might be possible to find a linear decision boundary which separates the two classes. Linear, RBF and Polynomial kernels are three types of kernels which are commonly used.

- Linear kernel is essentially the dot product between two given observations x_i and x_j in the feature space and can be written as

  $$K(x_i, x_j) = \sum_k x_{ik} \times x_{jk}$$

  where k is the dummy for number of dimension of the feature space.

- RBF kernel can transform vectors into a higher dimensional space and is given as

  $$K(x_i, x_j) = \exp(-\gamma \times \sum_k \|x_{ik} - x_{jk}^2\|)$$

- Polynomial kernel also transforms the dot product of two given vectors into a higher dimensional space and is given as

  $$K(x_i, x_j) = 1 + (\sum_k x_{ik} \times x_{jk})^d$$

  where d is the degree parameter of the polynomial kernel function.

Kernel is an important hyper-parameter that needs to be tuned when fitting the model.