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?

ANS: 1. Scalability: R^2is scale-invariant, meaning it does not change if the scale of the data changes, whereas RSS is affected by the scale of the dependent variable. This makes R^2 a better choice when comparing models fitted on different scales.

2. Interpretability: R^2 has an intuitive interpretation as the proportion of variance explained, which is easier to understand than the sum of squared residuals. An R^2 of 0.75 means that 75% of the variance in the dependent variable is explained by the model, which is a straightforward interpretation.

3. Benchmarking: R^2 provides a clear benchmark. An R^2 of 0 indicates that the model explains none of the variability in the response data around its mean, while an R^2 of 1 indicates that the model explains all the variability.

4. Adjustment for model complexity: Adjusted R^2 takes into account the number of predictors in the model, which helps in assessing whether the addition of a new predictor really improves the model or is just adding complexity without significantly improving the fit.

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.

ANS: **TSS** (total sum of squares) is equal to **ESS** (explained sum of squares) plus **RSS** (residual sum of squares), we need to start with the definitions of these terms and then use some algebraic manipulations to arrive at the desired result.

Let us begin by defining the three terms:
TSS = $\sum(Yi - \bar{Y})^2$, where Yi is the actual value of the response variable for observation i, and $\bar{Y}$ is the mean of the response variable.

ESS = $\sum(\hat{Y}i - \bar{Y})^2$, where $\hat{Y}i$ is the predicted value of the response variable for observation i.

RSS = $\sum(Yi - \hat{Y}i)^2$, which is the sum of squared differences between the actual and predicted values of the response variable.

Now, we can expand the terms in the definition of TSS using the definition of $\hat{Y}i$:

TSS = $\sum(Yi - \bar{Y})^2 = \sum[(Yi - \hat{Y}i) + (\hat{Y}i - \bar{Y})]^2 = \sum(Yi - \hat{Y}i)^2 + \sum(\hat{Y}i - \bar{Y})^2 + 2\sum(Yi - \hat{Y}i)(\hat{Y}i - \bar{Y})$

Next, we can use the definition of RSS to simplify the first term on the right-hand side:

$\sum(Yi - \hat{Y}i)^2 = RSS$

Similarly, we can use the definition of ESS to simplify the second term:
$\sum(\hat{Y}i - \bar{Y})^2 = ESS$

Now, we need to simplify the third term using some algebraic manipulations. We can start by expanding the product:

$2\sum(Yi - \hat{Y}i)(\hat{Y}i - \bar{Y}) = 2\sum(Yi\hat{Y}i - Yi\bar{Y} - \hat{Y}i\bar{Y} + \hat{Y}i^2)$

Then, we can use the fact that the sum of the residuals $(Yi - \hat{Y}i)$ is zero, which can be shown as follows:

$\sum(Yi - \hat{Y}i) = \sum Yi - \sum\hat{Y}i = n\bar{Y} - n\bar{Y} = 0$

Using this fact, we can simplify the third term:

$2\sum(Yi - \hat{Y}i)(\hat{Y}i - \bar{Y}) = 2\sum(Yi\hat{Y}i - \hat{Y}i\bar{Y}) = 2(\sum Yi\hat{Y}i - \sum\hat{Y}i\bar{Y}) = 2(\sum\hat{Y}iYi - n\bar{Y}^2) = 2(ESS - n(\bar{Y} - \hat{Y})^2)$

where $\hat{Y}$ is the sample mean of the predicted values, which is equal to $\bar{Y}$.

Substituting these results back into the equation for TSS, we get:

$TSS = RSS + ESS + 2(ESS - n(\bar{Y} - \hat{Y})^2) = RSS + 2ESS - 2n(\bar{Y} - \hat{Y})^2 = RSS + 2ESS - 2n(\bar{Y} - \bar{Y})^2 = RSS + 2ESS - 0 = RSS + ESS$

Therefore, we have shown that TSS is equal to ESS plus RSS.

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

ANS: While developing machine learning models you must have encountered a situation in which the training accuracy of the model is high but the validation accuracy or the testing accuracy is low. This is the case which is popularly known as overfitting in the domain of machine learning. Also, this is the last thing a machine learning practitioner would like to have in his model. In this article, we will learn about a method known as Regularization in Python which helps us to solve the problem of overfitting. But before that let's understand what is the role of regularization in Python and what is underfitting and overfitting.

Role Of Regularization

In Python, Regularization is a technique used to prevent overfitting by adding a penalty term to the loss function, discouraging the model from assigning too much importance to individual features or coefficients.

Let's explore some more detailed explanations about the role of Regularization in Python:

1. **Complexity Control**: Regularization helps control model complexity by preventing overfitting to training data, resulting in better generalization to new data.

2. **Preventing Overfitting**: One way to prevent overfitting is to use regularization, which penalizes large coefficients and constrains their magnitudes, thereby preventing a model from becoming overly complex and memorizing the training data instead of learning its underlying patterns.
3. **Balancing Bias and Variance**: Regularization can help balance the trade-off between model bias (underfitting) and model variance (overfitting) in machine learning, which leads to improved performance.
4. **Feature Selection**: Some regularization methods, such as L1 regularization (Lasso), promote sparse solutions that drive some feature coefficients to zero. This automatically selects important features while excluding less important ones.
5. **Handling Multicollinearity**: When features are highly correlated (multicollinearity), regularization can stabilize the model by reducing coefficient sensitivity to small data changes.
6. **Generalization**: Regularized models learn underlying patterns of data for better generalization to new data, instead of memorizing specific examples.

4.What is Gini–impurity index?

ANS: By quantifying the impurity level of data nodes, Gini Impurity aids in identifying optimal splits, leading to more homogeneous subsets and ultimately more accurate predictions. With a range from 0 to 0.5, where lower values signify purer nodes, Gini Impurity serves as a crucial tool in decision tree algorithms.
Gini Impurity measures how well does a node splits the data set between the two outcomes. It aims to reduce the impurity score from the root node of the tree to the leaf node. The lower the score, the better the split is, as we have seen in our example a Gini Index of 0 denotes a pure node(all data points belong to one class).

Entropy and Gini criterion measure similar performance metrics. Calculating Gini Impurity is much faster as it is less expensive to compute, whereas Entropy does log calculation and is a more expensive computation. However, the results obtained from Entropy are slightly better. In the sklearn library, we have a parameter where we can provide criterion to choose the quality of the split. We can give “gini” or “entropy” as a criterion, here is the link for the documentation.

In my opinion, using Gini criterion to train large datasets is better. The results from both computations are very close, so it is not worth to compute Entropy.

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

ANS: Decision trees have a tendency to overfit to the training set due to their inherent nature of being able to create complex, highly specific models that can perfectly fit the training data. This can lead to poor generalization to unseen data. Several reasons contribute to decision trees' propensity to overfit:

High Complexity: Decision trees can grow to be very deep and complex, especially when they are not pruned or limited in some way. A deep tree can capture intricate patterns and noise in the training data, which may not be relevant or generalizable to new data.
Sensitive to Small Variations: Decision trees are sensitive to small variations in the training data. A small change in the training data can lead to a completely different tree structure, which can result in overfitting.
Lack of Generalization: Decision trees tend to memorize the training data instead of learning general patterns that can be applied to new, unseen data. This lack of generalization can cause the model to perform poorly on data it has not seen before.
Impurity Measures: Decision trees use impurity measures like Gini impurity or entropy to make splits at each node. These measures may favor splits that result in overfitting the training data by creating very specific rules.
Limited Sample Size: If the training dataset is small, decision trees can easily memorize the data, including noise, rather than learning the underlying patterns. This can lead to overfitting.
No Global Optimization: Decision trees are built in a greedy manner by selecting the best split at each node locally. This lack of global optimization can lead to suboptimal decisions at each step, which may result in overfitting.

6.What is an ensemble technique in machine learning?

ANS: Ensemble methods are techniques that aim at improving the accuracy of results in models by combining multiple models instead of using a single model. The combined models increase the accuracy of the results significantly. This has boosted the popularity of ensemble methods in machine learning.
Ensemble methods aim at improving predictability in models by combining several models to make one very reliable model.
The most popular ensemble methods are boosting, bagging, and stacking.
Ensemble methods are ideal for regression and classification, where they reduce bias and variance to boost the accuracy of models.

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

ANS :

BAGGING	BOOSTING
1.The simplest way of combining predictions that belong to the same type.	A way of combining predictions that belong to the different type.
2.Aim to decrease variance , not bias.	Aim to decrease bias, not variance.
3. Each model receives equal weight.	Models are weighted according to their performance.
4.Each model is built independently.	New models are influenced by the performance of previous built models.
5.Bagging tries to solve the over-fitting problems.	Boosting tries to reduce bias.
6.Example: The Random Forest Model uses Bagging.	Example: The AdaBoost uses Bossting techniques.

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

ANS: A random forest is an ensemble machine-learning model that is composed of multiple decision trees. A decision tree is a model that makes predictions by learning a series of simple decision rules based on the features of the data. A random forest combines the predictions of multiple decision trees to make more accurate and robust predictions.

Random Forests are often used for classification and regression tasks. In classification, the goal is to predict the class label (e.g., “cat” or “dog”) of each sample in the dataset. In regression, the goal is to predict a continuous target variable (e.g., the price of a house) based on the features of the data.

Random forests are popular because they are easy to train, can handle high-dimensional data, and are highly accurate. They also have the ability to handle missing values and can handle imbalanced datasets, where some classes are more prevalent than others.

To train a random forest, you need to specify the number of decision trees to use (the n_estimators parameter) and the maximum depth of each tree (the max_depth parameter). Other hyperparameters, such as the minimum number of samples required to split a node and the minimum number of samples required at a leaf node, can also be specified.

Once the random forest is trained, you can use it to make predictions on new data. To make a prediction, the random forest uses the predictions of the individual decision trees and combines them using a majority vote or an averaging technique.

9. What is K-fold cross-validation?

ANS: k-Fold Cross-Validation Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample.

If you have a machine learning model and some data, you want to tell if your model can fit. You can split your data into training and test set. Train your model with the training set and evaluate the result with test set. But you evaluated the model only once and you are not sure your good result is by luck or not. You want to evaluate the model multiple times so you can be more confident about the model design.

The procedure has a single parameter called k that refers to the number of groups that a given data sample is to be split into. As such, the procedure is often called k-fold cross-validation. When a specific value for k is chosen, it may be used in place of k in the reference to the model, such as k=10 becoming 10-fold cross-validation.

Cross-validation is primarily used in applied machine learning to estimate the skill of a machine learning model on unseen data. That is, to use a limited sample in order to estimate how the model is expected to perform in general when used to make predictions on data not used during the training of the model.

It is a popular method because it is simple to understand and because it generally results in a less biased or less optimistic estimate of the model skill than other methods, such as a simple train/test split.

Note that k-fold cross-validation is to evaluate the model design, not a particular training. Because you re-trained the model of the same design with different training sets.

The general procedure is as follows:

- Shuffle the dataset randomly.
- Split the dataset into k groups
- For each unique group:
- Take the group as a hold out or test data set
- Take the remaining groups as a training data set
- Fit a model on the training set and evaluate it on the test set
- Retain the evaluation score and discard the model

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

ANS: Hyperparameters directly control model structure, function, and performance. Hyperparameter tuning allows data scientists to tweak model performance for optimal results. This process is an essential part of machine learning, and choosing appropriate hyperparameter values is crucial for success.For example, assume you're using the learning rate of the model as a hyperparameter. If the value is too high, the model may converge too quickly with suboptimal results. Whereas if the rate is too low, training takes too long and results may not converge. A good and balanced choice of hyperparameters results in accurate models and excellent model performance.

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

ANS: f the learning rate is too high, the algorithm may overshoot the minimum, and if it is too low, the algorithm may take too long to converge. Overfitting: Gradient descent can overfit the training data if the model is too complex or the learning rate is too high.

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

ANS: Logistic regression has traditionally been used to come up with a hyperplane that separates the feature space into classes. But if we suspect that the decision boundary is nonlinear we may get better results by attempting some nonlinear functional forms for the logit function.

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

ANS: The bias-variance tradeoff is a fundamental concept in machine learning that describes the relationship between a model's accuracy, complexity, and how well it can predict unseen data. It's a tradeoff between a model's ability to represent data patterns accurately (low bias) and its susceptibility to changes in the training data (high variance).

Bias-Variance Tradeoff is crucial in machine learning because it directly impacts a model's predictive performance. A model with high bias will consistently produce predictions that are far from the actual values, while a model with high variance will produce widely varying predictions for different training datasets. In both cases, the model's ability to generalize to new, unseen data is compromised.

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

Linear Kernel: Decision Boundary: Form: The linear kernel produces a decision boundary that is a hyperplane in the feature space. This hyperplane separates data points from different classes in a linear fashion. Assumption: It assumes that the relationship between the features and the target variable is linear.

The radial basis function kernel, or RBF kernel, is a popular kernel function used in various kernelized learning algorithms. In particular, it is commonly used in support vector machine classification.

The polynomial kernel is a kernel function commonly used with support vector machines (SVMs) and other kernelized models, that represents the similarity of vectors (training samples) in a feature space over polynomials of the original variables, allowing learning of non-linear models.

STATISTICS

1. Using a goodness of fit, we can assess whether a set of obtained frequencies differ from a set of frequencies. Expected
2. Chisquare is used to analyse . Frequencies.
3. What is the mean of a Chi Square distribution with 6 degrees of freedom? 6
4. Which of these distributions is used for a goodness of fit testing? Chisqared distribution.
5. Which of the following distributions is Continuous . Binomial Distribution
6. A statement made about a population for testing purpose is called? Hypothesis.
7. If the assumed hypothesis is tested for rejection considering it to be true is called? Null Hypothesis .
8. If the Critical region is evenly distributed then the test is referred as? Two tailed.
9. Alternative Hypothesis is also called as? 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? np.