Machine learning

Q1 **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?**

R-squared is a commonly used measure of regression fit. It is a value between 0 and 1 that represents the proportion of variation in the dependent variable that is explained by the independent variables in the model. A larger R-squared value indicates a better fit.

Residual sum of squares (RSS) is another way to measure the goodness of fit of a regression. It is the sum of squares of the model residual (the difference between the observed and predicted values). A lower RSS value indicates a better match.

Both R-squared and RSS can be used to evaluate the fit of a regression model, but R-squared is generally easier to interpret and widely used because it is a relative measure (proportion of variation explained by the model) rather than an absolute measure. measure (sum of squares of the remainder). Additionally, R-squared is unitless, so models with different units can be easily compared.

Q2 **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, TSS (Total Sum of Squares), ESS (Explained Sum of Squares), and RSS (Residual Sum of Squares) are measures used to evaluate the goodness of fit of a model.

- TSS (Total Sum of Squares) is the sum of the squares of the difference between the observed values and the mean of the observed values. It measures the total variation in the dependent variable.
- ESS (Explained Sum of Squares) is the sum of the squares of the difference between the predicted values and the mean of the observed values. It measures the variation in the dependent variable that is explained by the independent variables in the model.
- RSS (Residual Sum of Squares) is the sum of the squares of the difference between the observed values and the predicted values. It measures the variation in the dependent variable that is not explained by the independent variables in the model.

The three metrics are related by the following equation:

TSS = ESS + RSS

This equation states that the total variation in the dependent variable is equal to the explained variation plus the residual variation.

R-squared is a commonly used measure of goodness of fit in regression, it is defined as the ratio of ESS over TSS. R-squared is a value between 0 and 1 that represents the proportion of the variation in the dependent variable that is explained by the independent variables in the model.

Q3 **What is the need of regularization in machine learning?**

Regularization is a technique used in machine learning to eliminate redundancy. Overfitting occurs when a model is so well trained on training data that it performs poorly on new, unseen data. This happens when the model is too complex with too many parameters, forcing the model to incorporate noise into the training data rather than the underlying model.

Regularization solves this problem by adding a penalty term to the objective function (also known as the cost function) that the model is trying to optimize. This penalty term, also known as the regularization term, prevents certain model parameters from becoming too large, effectively reducing model complexity.

There are two main types of regularization: L1 and L2 regularization. L1 regularization, also known as Lasso regularization, adds a penalty term to the objective function that is proportional to the absolute value of the parameter. L2 regularization, also known as Ridge regularization, adds a penalty term to the objective function that is proportional to the square of the parameter. Regularization can help improve model generalization performance by preventing overfitting and making the model more robust to unseen data. It also helps reduce model variance and improves model interpretability by reducing parameters to zero.

Q4 **What is Gini–impurity index?**

The Gini impurity index is a measure of impurity in a data set. It is often used in decision tree-based algorithms such as CART (Classification and Regression Trees) to determine the best split points when building a tree. The Gini impurity index is calculated by calculating the probability that a randomly selected item is mislabeled if it would have been randomly labeled according to the distribution of the classes together. It is defined as:

Impurity Gini = 1 - $\Sigma(p(i))^2$

where p(i) is the proportion of elements in class i.

A Gini impurity score of 0 indicates that all elements in the set belong to the same class, while a score of 1 indicates that the elements are evenly distributed across all classes. When building a decision tree, the algorithm will try to split the data so that the Gini impurity score is as low as possible.

Q5 Are unregularized decision-trees prone to overfitting? If yes, why?

yes, irregular decision trees are subject to redundancy. A decision tree is a model that recursively divides data into subsets based on specified criteria to create the cleanest possible subsets. This process continues until certain stopping conditions are met.

Unregulated decision trees are capable of creating very complex models with many branches and leaves. Each time a division is made, the tree grows deeper, increasing the number of branches and leaves. As a result, the tree may incorporate noise into the training data rather than the underlying model, causing the model to perform well on the training data but poorly on new, unseen data. Unregularized decision trees also have no mechanism to prevent overfitting, such as pruning or regularization. Therefore, they are more likely to overfit the data than regularized decision tree algorithms. In summary, irregular decision trees are prone to overfitting because they can build complex models to accommodate noise in the training data and lack a mechanism to prevent overfitting. Regular decision tree algorithms, such as random forests, use techniques such as bagging and random element selection to reduce redundancy.

Q6 **. What is an ensemble technique in machine learning?**

An ensemble in machine learning is a method of combining predictions from multiple models to improve the overall performance of the final prediction. The idea behind ensemble techniques is that by combining forecasts from multiple models, the resulting forecasts will be more robust and accurate than forecasts from a single model. There are several types of integration techniques, including:

Bagging: This is a method of training multiple models on different subsets of the training data and then combining their predictions. An example of this is random forests. Reinforcement: This is a method of sequentially training multiple models, where each model tries to correct the errors of the previous model. An example of this is AdaBoost. Stacking: This is an approach where multiple models are trained on the same data and then their predictions are used as input to another model called a meta-model. The metamodel will then make the final predictions. The overall technology can improve the performance of the model by reducing the model and/or deviation dispersion. They also clearly understand how different models contribute to the final forecast that helps improve the model's explanation. Ensemble methods also help to improve the generalization performance of the model by making it more robust to unseen data.

Q7 **What is the difference between Bagging and Boosting techniques?**

In machine learning, both are ensemble methods, but they differ in how they combine predictions from multiple models.

Bootstrap Aggregating is a technique for building multiple models by training them on different subsets of the training data created by random sampling and replacement. Once the models are trained, their predictions are combined to produce a final prediction. The idea behind packing is to reduce model variance by calculating predictions from multiple models. Bagging helps to reduce overcapacity caused by the large variance in calculating the predictions of different models. Random Forest is an example of an algorithm that uses packing techniques. On the other hand, advertising is a technology that creates more models when training the sequence. In each iteration, the model is concentrated on the previous model error samples. The idea of improvement is to reduce the model bias by training more models to correct the errors of previous models. Boosting helps reduce underfitting caused by high bias by focusing the trained model on the most complex samples. AdaBoost is an example of an algorithm that uses boosting techniques.

In summary, bagging is a method to reduce model variance by averaging multiple model predictions, while boosting is a method to reduce model bias by training multiple models to correct errors from previous models.

Q8 **What is out-of-bag error in random forests?**

In random forests, out-of-bag (OOB) error is a measure of model performance that is calculated using samples that were not used to train a particular tree.

When training a random forest, the data is randomly sampled with replacement to create multiple subsets of the data called bootstrap samples. Each tree in the forest is trained using a different bootstrap sample. Samples that are not included in the bootstrap samples for a given tree are called out-of-bag samples. The OOB error is the mean error of the out-of-bag samples of all trees in the forest.

One advantage of the OOB error is that it provides a measure of model performance without the need for a separate validation set. It can also be used as an estimate of the test error, since the OOB samples are not used in model training and can therefore be considered a form of cross-validation.

The OOB error can be calculated as the average error from misclassification (for classification problems) or the root mean square error (for regression problems) for all trees in the forest. It is important to note that the OOB error is a better estimate of the test error than the training error. This is because the OOB samples are different for each tree, resulting in more variability in the data.

Q9 **What is K-fold cross-validation?**

K-fold cross-validation is a technique for evaluating the performance of machine learning models. It is a resampling method that divides the data into k equal subsets or "folds" and then trains and evaluates the model k times, each time using a different fold as the validation set and the remaining k-1 times as the training set Skip. . The K-fold cross-validation process can be summarized as follows:

Divide the data into k subsets of equal size.

For each iteration (k times):

Use one of the k subsets as the validation set and the remaining k-1 subsets as the training set. Train the model in the training set.
Evaluate the model on the validation set. Record the evaluation metrics (e.g. precision, root mean squared error)
Calculates the average evaluation metric over all iterations.

The final result of k-fold cross-validation is the average of the evaluation metrics across all iterations. One of the main advantages of k-fold cross-validation is that it is data efficient because each sample is used only once for training and validation. In addition, it provides a more robust estimate of model performance by averaging the evaluation metrics across multiple validation sets. The value of K can be chosen based on the size of the data set and the level of precision required in the estimation.

Q10 **What is hyper parameter tuning in machine learning and why it is done?**
Hyperparameter tuning, also known as hyperparameter optimization, is the process of systematically searching for the best combination of hyperparameters for a machine learning model. Hyperparameters are parameters that are not learned from the data during training, but are set before training begins. Super -parameter examples include learning speed, number of hidden layers in the neural network and detection intensity.

Super -Reuse aims to find the best super -digital combination that will lead to the best model performance in invisible data. The hyperparameter tuning process involves training models multiple times with different combinations of hyperparameters and evaluating the performance of each model on a validation set. The hyperparameter combination that provides the best performance in the validation set is selected as the final hyperparameter set for the model. The hyperparameter tuning process is important in machine learning models because the performance of the model can be very sensitive to the choice of hyperparameters. Small changes in the hyperparameter value can cause large changes in model performance. Hyperparameter tuning allows you to find the best set of hyperparameters that gives the best model performance.

There are several ways to perform hyperparameter tuning, such as grid search, random search, and Bayesian optimization. These methods differ in how they search for the best set of hyperparameters, but they all aim to find the best combination of hyperparameters that will allow the model to achieve the best performance on unseen data.

Q11 **What issues can occur if we have a large learning rate in Gradient Descent?**

Gradient descent is an optimization algorithm for finding the minimum of a function that is often used to train machine learning models. The learning rate is a hyperparameter that controls the step size of the algorithm to move in the negative direction of the gradient, and is critical for the algorithm to converge to a minimum value. If the learning rate is set too high, the following problems may occur:

Oscillation: The algorithm will exceed the minimum and oscillate back and forth, preventing it from reaching convergence.

Difference: The algorithm will continue to update the parameters in the same direction, increasing the value of the cost function, resulting in differences. Slow convergence: A large learning rate will cause the algorithm to take larger steps, which will slow its convergence or even fail to reach the global minimum.

Bounce around the optimal value: A high learning rate can cause the algorithm to exceed the optimal value and bounce, preventing a stable solution from being reached. Higher variance: High learning rates cause the algorithm to generate solutions with higher variance, which causes the model to perform poorly on unseen data.

Q12 **Can we use Logistic Regression for classification of Non-Linear Data? If not, why?**

Logistic regression is a linear model often used for classification problems. It assumes that the relationship between the input variables and the output variables is linear. This means that it can only model linear decision boundaries and is not suitable for modeling non-linear relationships.

For non-linear data, logistic regression does not accurately capture the underlying patterns in the data and can lead to poor model performance. A common way to deal with non-linear data is to use a non-linear transformation of the input variable, such as a polynomial transformation, before applying logistic regression. However, this approach cannot capture complex nonlinear relationships in the data. In cases where logistic regression is not suitable for nonlinear data, other algorithms such as decision trees, k-nearest neighbors, support vector machines, or neural networks may be more appropriate. These algorithms can handle nonlinear decision boundaries and can better capture patterns underlying nonlinear data.

Q13 **Differentiate between Adaboost and Gradient Boosting.**

AdaBoost and gradient boosting are ensemble techniques used to improve the performance of machine learning models. But they differ in their approach and the way they combine multiple models. AdaBoost (Adaptive Boosting) is a boosting algorithm that builds multiple models sequentially. At each iteration, the algorithm focuses on samples that are misclassified in previous models. The idea behind AdaBoost is to reduce model bias by training multiple models to correct errors from previous models. AdaBoost adjusts the weights of the samples according to the errors of the previous model so that the next model pays more attention to samples that were previously misclassified.

Gradient boosting, on the other hand, is a boosting algorithm that creates multiple models in the same way as AdaBoost, but it performs an additional optimization step: minimizing the loss function. At each iteration, the algorithm fits a new model to the residuals of the previous model. The idea behind gradient boosting is to reduce model biases and outliers by adding new models to correct the errors of previous models and improve the overall performance of the ensemble.

In summary, both AdaBoost and gradient boosting are ensemble techniques used to improve the performance of machine learning models. AdaBoost focuses on reducing bias by adjusting sample weights according to the error of the previous model, while Gradient Boosting focuses on minimizing the loss function by fitting the new model to the rest of the previous model.

Q14 **What is bias-variance trade off in machine learning?**

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

Bias is the error caused by a simpler model approximating a real-world problem that can be extremely complex. High variance models tend to have simple structures, such as linear models, and make strong assumptions about the form of relationships between input and output variables. These models usually do not fit the data well, meaning they do not capture the underlying patterns in the data well.

Variance, on the other hand, refers to the error caused by the model's sensitivity to small fluctuations in the training data. High-variance models typically have complex structures, such as decision trees or neural networks, and allow a lot of flexibility in how they fit the data. These models tend to overfit the data, meaning they fit the noise in the training data rather than the underlying models.

The goal of machine learning is to find a good balance between bias and variance to create models that fit training data well and generalize well to new, unseen data. However, in practice, it is often difficult to find the optimal balance because models with large biases are poorly generalized and models with large variances are poorly fit.

Q15 **Give short description each of Linear, RBF, Polynomial kernels used in SVM.**

Support vector machines (SVM) is a type of algorithm that can be used for classification and regression problems. One of the most important features of the SVM is their ability to process non - linear multiple data by converting input data to a higher dimension space where they are linearly separated. This transformation is done using the kernel function.

Three commonly used kernel functions for SVMs:

Linear Kernel: The linear kernel is the simplest kernel function, it just calculates the dot product of the input vectors. Used when the data is already linearly separable in the original function space.

Radial Basis Function (RBF) kernel: The RBF kernel is a popular kernel function used when the data is not linearly separable. The RBF kernel maps the input data into a high-dimensional space where they become linearly separable. It is defined as the exponent of the negative Euclidean distance between the input vectors.

Polynomial nucleus: The polynomial nucleus is used to model polynomial coefficients in input data. It is defined as an entrance vector point product to increase to a certain function. The degree of the polynomial can be specified as a hyperparameter.

In summary, linear kernels are used when the data is already linearly separable, RBF kernels are used when the data is not linearly separable, and polynomial kernels are used to model polynomial relationships in the input data. These kernels are used in SVM to transform the input data into a high-dimensional space where they become linearly separable.

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

d Expected

2. Chisquare is used to analyse

a) Score

b) Rank

c) Frequencies

d) All of these

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

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

b Chisqared distribution

5. Which of the following distributions is Continuous

a) Binomial Distribution

b) Hypergeometric Distribution

c) F Distribution

d) Poisson Distribution

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

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

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

a Two tailed

9. Alternative Hypothesis is also called as?

a) Composite hypothesis

b) Research Hypothesis

c) Simple Hypothesis

d) Null Hypothesis

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

a np

SQL

1. Write SQL query to show all the data in the Movie table.

```
SELECT*

FROM movie;
```

2. Write SQL query to show the title of the longest runtime movie.

```
SELECT title

 FROM movie

 ORDER BY runtime DESC LIMIT 1;
```

3. Write SQL query to show the highest revenue generating movie title.

```
SELECT title

 FROM movie

 ORDER BY revenue DESC LIMIT 1;
```

4. Write SQL query to show the movie title with maximum value of revenue/budget.

```
SELECT title
 FROM movie
 ORDER BY budget DESC LIMIT 1;
```

5. Write a SQL query to show the movie title and its cast details like name of the person, gender, character name, cast order.

```
SELECT a.title, c.gender, b.character_name, b.cast_order, d.person_name FROM movie a
INNER JOIN movie_cast b ON a.movie_id=b.movie_id
INNER JOIN gender c ON c.gender_id=b.gender_id
INNER JOIN person d ON d.person_id= b.person_id;
```

6. Write a SQL query to show the country name where maximum number of movies has been produced, along with the number of movies produced.

```
SELECT a.country_name, count(a.country_name) AS count FROM country a
INNER JOIN production_country  b ON b.country_id=a.country_id
 GROUP BY country_name
 ORDER BY count DESC LIMIT 1;
```

7. Write a SQL query to show all the genre_id in one column and genre_name in second column.

```
SELECT *
FROM genre;
```

8. Write a SQL query to show name of all the languages in one column and number of movies in that particular column in another column.

```
SELECT a.movie_id, count(b.language_name) FROM movie_languages a
JOIN language b ON a.language_id=b.language_id
GROUP BY language_name
 ORER BY count(language_name) desc;
```

9. Write a SQL query to show movie name in first column, no. of crew members in second column and number of cast members in third column.

```
SELECT m.title AS movie_name, count(cr.person_id) AS no_of_crews,

count(ca.person_id) as No_of_cast  FROM movie m

INNER JOIN movie_crew cr ON cr.movie_id=m.movie_id

INNER JOIN movie_cast ca ON ca.movie_id=cr.movie_id

GROUP BY m.title;
```

10. Write a SQL query to list top 10 movies title according to popularity column in decreasing order.

```
SELECT title FROM movie

ORDER BY popularity DESC LIMIT 10;
```

11. Write a SQL query to show the name of the 3rd most revenue generating movie and its revenue.

```
SELECT title, revenue FROM movie ORDER BY revenue DESC LIMIT 1 OFFSET 2;
```

12. Write a SQL query to show the names of all the movies which have “rumoured” movie status.

```
Select title FROM movie

WHERE movie_status LIKE 'rumored';
```

13. Write a SQL query to show the name of the “United States of America” produced movie which generated maximum revenue.

```
SELECT title, revenue FROM movie a

INNER JOIN production_country b ON b.movie_id = a.movie_id

INNER JOIN country c ON c.country_id = b.country_id

WHERE country_name = 'United States of America'

ORDER BY revenue DESC LIMIT 1;
```

14. Write a SQL query to print the movie_id in one column and name of the production company in the second column for all the movies.

```
SELECT m.movie_id, pc.company_name FROM movie m
 INNER JOIN movie_company mc ON mc.movie_id = m.movie_id
 INNER JOIN production_company pc ON pc.company_id =mc.company_id;
```

15. Write a SQL query to show the title of top 20 movies arranged in decreasing order of their budget.

```
SELECT title FROM movie
ORDER BY budget DESC LIMIT 20;
```