Solved 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.

R^2 is a metric used for evaluating the performance of a regression model. The metric helps us to compare our current model with a constant baseline and tells us how much our model is better. The constant baseline is chosen by taking the mean of the data and drawing a line at the mean. R^2 is a scale-free score that implies it doesn't matter whether the values are too large or too small, the R^2 will always be less than or equal to 1.

The residual sum of squares (RSS) measures the level of variance in the error term, or residuals, of a regression model. The smaller the residual sum of squares, the better your model fits your data; the greater the residual sum of squares, the poorer your model fits your data.

R square is a better measure of goodness of fit model in regression because it squares the residual values, thereby, treating positive and negative discrepancies in the same way

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.

The Total Sum of Squares, denoted TSS, is the squared differences between the observed dependent variable and its mean. It is a measure of the total variability of the dataset.

The ESS (Explained Sum of Squares) is the sum of the differences between the predicted value and the mean of the dependent variable. If this value of ESS is equal to the Total Sum of Squares, it means our regression model captures all the observed variability and is perfect.

The residual sum of squares (RSS) also called sum of squares error, or SSE. The error is the difference between the observed value and the predicted value. The smaller the error, the better the estimation power of the regression**.**

The equation relating these three metrics with each other is-

$$TSS = ESS + RSS$$

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

Ans.

Regularization refers to techniques that are used to calibrate machine learning models in order to minimize the adjusted loss function and prevent overfitting or underfitting. Using Regularization, we can fit our machine learning model appropriately on a given test set and hence reduce the errors in it. Further, regularization also helps to reduce dimensionality of the training dataset, which in turn prevents over-fitting.

---

4. What is Gini–impurity index?

Ans.

Gini Index, also known as Gini impurity, calculates the amount of probability of a specific feature that is classified incorrectly when selected randomly. If all the elements are linked with a single class then it can be called pure. The Gini index varies between values 0 and 1, where 0 expresses the purity of classification, i.e. All the elements belong to a specified class or only one class exists there. And 1 indicates the random distribution of elements across various classes. The value of 0.5 of the Gini Index shows an equal distribution of elements over some classes. While designing the decision tree, the features possessing the least value of the Gini Index would get preferred. The Gini Index is determined by deducting the sum of squared of probabilities of each class from one, mathematically.

---

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

Ans.

Decision trees are prone to overfitting, especially when a tree is particularly deep. This is due to the amount of specificity we look at leading to smaller sample of events that meet the previous assumptions. This small sample could lead to unsound conclusions. Regularization in terms of decision tree means to control the growth of the tree. When decision tree becomes too large they tend to over-fit. To avoid over-fitting, we regularize the tree. Thereby, reducing the chances of overfitting. Hence, unregularized decision tress are prone to over fitting.

---

6. What is an ensemble technique in machine learning?

Ans.

Ensemble methods are techniques that create multiple models and then combine them to produce improved results. Ensemble methods usually produces more accurate solutions than a single model would. Ensemble techniques are classified into three types:

1. Bagging
2. Boosting
3. Stacking

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

Ans.

Bagging	Boosting
The simplest way of combining predictions that belong to the same type.	A way of combining predictions that belong to the different types.
Aim to decrease variance, not bias.	Aim to decrease bias, not variance.
Each model receives equal weight.	Models are weighted according to their performance.
Each model is built independently.	New models are influenced by the performance of previously built models.
Different training data subsets are selected using row sampling with replacement and random sampling methods from the entire training dataset.	Every new subset contains the elements that were misclassified by previous models.
Bagging tries to solve the over-fitting problem.	Boosting tries to reduce bias.
If the classifier is unstable (high variance), then apply bagging.	If the classifier is stable and simple (high bias) the apply boosting.
In this base classifiers are trained parallelly.	In this base classifiers are trained sequentially.
Example: The Random forest model uses Bagging.	Example: The AdaBoost uses Boosting techniques

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

Ans.

The out-of-bag (OOB) error is the average error for each calculated using predictions from the trees that do not contain in their respective bootstrap sample. This allows the RandomForestClassifier to be fit and validated whilst being trained. It is a useful measure to discriminate between different random forest classifiers. We could, for instance, vary the number of trees or the number of variables to be considered, and select the combination that produces the smallest value for this error rate

---

9. What is K-fold cross-validation?

Ans.

Cross-validation is a resampling procedure used to evaluate machine learning models on a limited data sample. 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. The general procedure is as follows:

1. Shuffle the dataset randomly.
2. Split the dataset into k groups
3. 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
4. Summarize the skill of the model using the sample of model evaluation scores

Importantly, each observation in the data sample is assigned to an individual group and stays in that group for the duration of the procedure. This means that each sample is given the opportunity to be used in the hold out set 1 time and used to train the model k-1 times.

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

Ans.

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

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

Ans.

In a situation with a very high learning rate in Gradient Descent, the algorithm may bypass the local minimum and overshoot, i.e., the optimal solution will be skipped.

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

Ans.

Logistic Regression has been used as a linear classifier, i.e. when the classes can be separated in the feature space by linear boundaries. On the other hand, Non-Linear Classification refers to separating those instances that are not linearly separable. In logistic regression, to differentiate between the two classes, an arbitrary line is drawn, ensuring that both the classes are on distinct sides. However, in case of non-linear data, in order to differentiate between the two classes, it is impossible to draw an arbitrary straight line to ensure that both the classes are on distinct sides. Hence, logistic regression is not advised to classify non-linear data.

13.Differentiate between Adaboost and Gradient Boosting.

Ans.

Adaboost	Gradient Boost
An additive model where shortcomings of previous models are identified by high-weight data points.	An additive model where shortcomings of previous models are identified by the gradient.
The trees are usually grown as decision stumps.	The trees are grown to a greater depth usually ranging from 8 to 32 terminal nodes.
Each classifier has different weights assigned to the final prediction based on its performance.	All classifiers are weighed equally and their predictive capacity is restricted with learning rate to increase accuracy.
It gives weights to both classifiers and observations thus capturing maximum variance within data.	It builds trees on previous classifier's residuals thus capturing variance in data.

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

Ans.

The bias–variance tradeoff is the property of a model that the variance of the parameter estimated across samples can be reduced by increasing the bias in the estimated parameters.

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

Ans.

Linear kernel is the most basic type of kernel, usually one dimensional in nature. It proves to be the best function when there are lots of features. The linear kernel is mostly preferred for text-classification problems as most of these kinds of classification problems can be linearly separated.

Linear kernel functions are faster than other functions.

Linear Kernel Formula

$$F(x, xj) = sum(x.xj)$$

Here, x, xj represents the data you're trying to classify.

Polynomial kernel is a more generalized representation of the linear kernel. It is not as preferred as other kernel functions as it is less efficient and accurate.

Polynomial Kernel Formula

F(x, xj) = (x.xj+1)^d

Here '.' shows the dot product of both the values, and d denotes the degree.
F(x, xj) representing the decision boundary to separate the given classes.

Gaussian Radial Basis Function (RBF) is one of the most preferred and used kernel functions in svm. It is usually chosen for non-linear data. It helps to make proper separation when there is no prior knowledge of data.

Gaussian Radial Basis Formula

F(x, xj) = exp(-gamma * ||x - xj||^2)

The value of gamma varies from 0 to 1. You have to manually provide the value of gamma in the code. The most preferred value for gamma is 0.1.