MACHINE LEARNING

Q1 to Q15 are subjective answer type questions, Answer them briefly.

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:
   Both R-squared and Residual Sum of Squares (RSS) are measures of goodness of fit in regression analysis, but they capture different aspects of the model's performance.

   R-squared (also known as the coefficient of determination) measures the proportion of variation in the dependent variable that is explained by the independent variables in the model. In other words, it indicates how well the model fits the data, with values ranging from 0 to 1. Higher R-squared values indicate a better fit, as they mean that a larger proportion of the variation in the dependent variable is explained by the independent variables in the model.

   On the other hand, RSS measures the total sum of squared differences between the actual values of the dependent variable and the predicted values by the model. It represents the amount of unexplained variation in the data, and lower RSS values indicate a better fit, as they mean that the model is able to explain more of the variation in the data.

   Therefore, both measures are useful in evaluating the goodness of fit of a model, but they serve different purposes. R-squared is a useful measure to assess the overall fit of the model and to compare different models, while RSS is useful to identify the degree of the error in the model's predictions.

   In general, a good model should have both a high R-squared value and a low RSS value, indicating that it explains a large proportion of the variation in the dependent variable and has a low degree of error in its predictions. However, in some cases, one measure may be more important than the other, depending on the research question and the nature of the data being analyzed.

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 = ESS + RSS, where TSS is Total Sum of Squares, ESS is Explained Sum of Squares and RSS is Residual Sum of Squares. The aim of Regression Analysis is explaining the variation of dependent variable Y

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

   Ans:
   While training a machine learning model, the model can easily be overfitted or under fitted. To avoid this, we use regularization in machine learning to properly fit a model onto our test set. Regularization techniques help reduce the chance of overfitting and help us get an optimal model.

   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.

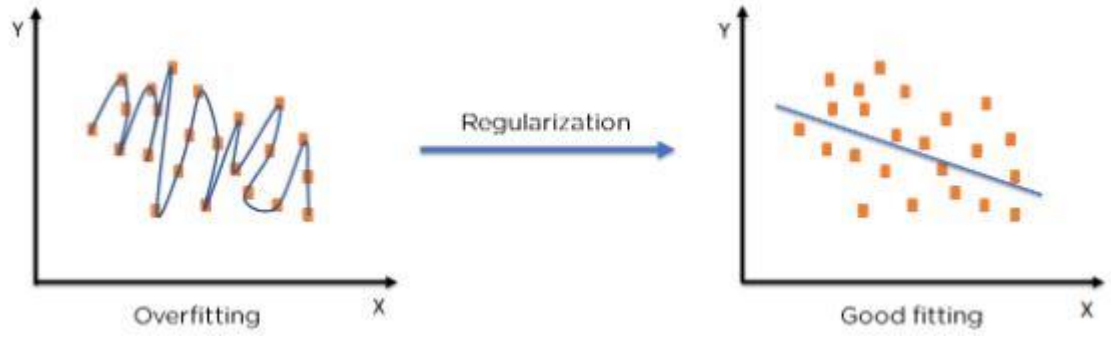

Figure: Regularization on an over-fitted model

Using Regularization, we can fit our machine learning model appropriately on a given test set and hence reduce the errors in it.

4. What is Gini–impurity index?

Ans:
Gini Impurity is a measurement used to build Decision Trees to determine how the features of a dataset should split nodes to form the tree. More precisely, the Gini Impurity of a dataset is a number between 0-0.5, which indicates the likelihood of new, random data being misclassified if it were given a random class label according to the class distribution in the dataset.

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

Ans:
Decision trees are prone to overfitting when they capture noise in the data. Pruning and setting appropriate stopping criteria are used to address this assumption.

Decision trees have a tendency to overfit to the training set because they can keep growing deeper and more complex until they perfectly classify the training data. This can lead to the tree capturing noise in the data, rather that the underlying relationships, and thus performing poorly on new, unseen data.

Regularization techniques such as pruning, setting a minimum number of samples required to split a node, or limiting the maximum depth of the tree can help mitigate overfitting in decision trees.

Regularization techniques aim to simplify the tree and prevent it from becoming overly complex, thus improving its ability to generalize to new data.

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.

In this ensemble technique, machine learning professionals use a number of models for making predictions about each data point. The predictions made by different models are taken as separate votes. Subsequently, the prediction made by most models is treated as the ultimate prediction.

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

Ans:
Bagging and boosting are different ensemble techniques that use multiple models to reduce error and optimize the model. The bagging technique combines multiple models trained on different subsets of data, whereas boosting trains the model sequentially, focusing on the error made by the previous model.

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 Random Forest Classifier to be fit and validated whilst being trained.

9. What is K-fold cross-validation?

Ans:
K-fold cross-validation is a technique for evaluating predictive models. The dataset is divided into k subsets or folds. The model is trained and evaluated k times, using a different fold as the validation set each time.

Performance metrics from each fold are averaged to estimate the model's generalization performance. This method aids in model assessment, selection, and hyperparameter tuning, providing a more reliable measure of a model’s effectiveness.

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

Ans:
Hyperparameters in Machine learning are those parameters that are explicitly defined by the user to control the learning process. These hyperparameters are used to improve the learning of the model, and their values are set before starting the learning process of the model. The only way to determine these is through multiple experiments, where you pick a set of hyperparameters and run them through your model. This is called hyperparameter tuning.

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.

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

Ans:
When the learning rate is too large, gradient descent can suffer from divergence. This means that weights increase exponentially, resulting in exploding gradients which can cause problems such as instabilities and overly high loss values.

The learning rate is an important hyperparameter that greatly affects the performance of gradient descent. It determines how quickly or slowly our model learns, and it plays an important role in controlling both convergence and divergence of the algorithm. When the learning rate is too large, gradient descent can suffer from divergence. This means that weights increase exponentially, resulting in exploding gradients which can cause problems such as instabilities and overly high loss values. On the other hand, if the learning rate is too small, then gradient descent can suffer from slow convergence or even stagnation—which means it may not reach a local minimum at all unless many iterations are performed on large datasets.

In order to avoid these issues with different learning rates for each parameter/variable, we use adaptive techniques such as Adagrad and Adam which adjust their own learning rates throughout training based on real-time observations of parameters during optimization (i.e., they control exploration/exploitation trade-offs). These adaptive measures ensure better results than standard gradient descent while avoiding potential pitfalls in terms of either massive gains or slow losses due to misconfigured static global learning rates like those used with traditional gradient descent algorithms.

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

Ans:

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 (e.g. scikit-learn) 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 **github.**

Logistic Regression is a type of **Linear Regression** that uses a **Sigmoid *Activation Function***, which is a type of ***Nonlinear Activation Function***. **Linear Regression** is a method of predicting the outcome of a ***continuous variable*** based on the relationship between the ***independent variables*** and the ***dependent variable***. By replacing the ***Linear Activation Function*** with a **Sigmoid *Activation Function***, **Linear Regression** can become a **Logistic Regression** model.

The equation of **Linear Regression** can be written as —

$$F(x) = w^T x + b$$

Equation of Linear Regression

If the *equation* of Linear Regression is passed through a Sigmoid Activation Function, then it becomes Logistic Regression.

$$\hat{y} = \sigma(w^T x + b)$$

13. Differentiate between Adaboost and Gradient Boosting.

Ans:

Adaboost is computed with a specific loss function and becomes more rigid when comes to few iterations. But in gradient boosting, it assists in finding the proper solution to additional iteration modeling problem as it is built with some generic features.

Adaboost is effective when it's implied in week learners and when it is related to few classifications' errors it reduces the loss function. It was developed for problems that require binary classification and can be used to improve the efficiency of decision trees. In gradient boosting, it is used to crack the problems with differential loss functions. It can be implied in both regression problems and classification issues. Though there are a few differences in these two boosting techniques, both follow a similar path and have the same historic roots. Both boost the performance of a single learner by persistently shifting the attention towards problematic remarks which are challenging to compute and predict.

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

Ans:

On the other hand, in certain cases, it struggles to grasp the intricacies of the data and thus fails to provide an accurate prediction. Striking a balance between accuracy and the ability to make predictions beyond the training data in an ML model is called the bias-variance tradeoff.

In statistics and machine learning, the bias–variance tradeoff describes the relationship between a model's complexity, the accuracy of its predictions, and how well it can make predictions on previously unseen data that were not used to train the model.

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

Ans:

In machine learning, 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.

Kernel Function is a method used to take data as input and transform it into the required form of processing data. “Kernel” is used due to a set of mathematical functions used in Support Vector Machine providing the window to manipulate the data. So, Kernel Function generally transforms the training set of data so that a non-linear decision surface is able to transform to a linear equation in a higher number of dimension spaces.

SVM algorithms use a set of mathematical functions that are defined as the kernel. The function of kernel is to take data as input and transform it into the required form. Different SVM algorithms use different types of kernel functions. These functions can be different types. For example, ***linear, nonlinear, polynomial, radial basis function (RBF), and sigmoid.***