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: R-squared and Residual Sum of Squares (RSS) are both measures of the goodness of fit of a regression model, but they capture different aspects of the fit.

R-squared, also known as the coefficient of determination, measures the proportion of variance in the dependent variable that is explained by the independent variables in the model. It ranges from 0 to 1, with a higher value indicating a better fit. R-squared is useful for comparing different models or for determining the proportion of the variability in the dependent variable that is explained by the model.

On the other hand, Residual Sum of Squares (RSS) measures the total amount of unexplained variance in the dependent variable that remains after the model has been fit. It is the sum of the squared differences between the actual and predicted values of the dependent variable. A lower value of RSS indicates a better fit. RSS is useful for evaluating the accuracy of the predictions of the model.

In general, both measures are important and should be considered together when evaluating the goodness of fit of a model. However, R-squared is often considered to be a better measure of goodness of fit than RSS because it provides a single number that summarizes the proportion of variance in the dependent variable that is explained by the model, which is more interpretable and easier to compare across models.

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- This gives you the distance from the linear line drawn to each particular variable. You could also describe TSS as the dispersion of observed variables around the mean, or the variance. So, the goal of TSS is to measure the total variability of the dataset.

ESS-The explained sum of squares (ESS) is the sum of the squares of the deviations of the predicted values from the mean value of a response variable, in a standard regression model
RSS- 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.

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

4. What is Gini–impurity index?

   Ans: A measurement used to build Decision Trees to determine how the features of a dataset should split nodes to form the tree.
   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 unregularized decision-trees prone to overfitting? If yes, why?

Ans: Overfitting in Decision tree

Overfitting can be one problem that describes if your model no longer generalizes well Overfitting happens when any learning processing overly optimizes training set error at the cost test error. While it's possible for training and testing to perform equality well in cross validation, it could be as the result of the data being very close in characteristics, which may not be a huge problem. In the case of decision tree's they can learn a training set to a point of high granularity that makes them easily overfit. Allowing a decision tree to split to a granular degree, is the behavior of this model that makes it prone to learning every point extremely well — to the point of perfect classification — ie: overfitting.

Following steps to avoid overfitting:

- Use a test set that is not exactly like the training set, or different enough that error rates are going to be easy to see.

What is different enough? Enough to generalize what is being predicted. The problem should dictate this somewhat because if you are predicting on a baseline that is anomalous, you will need to approximate your validation set close enough. If the problem is more general such as spam classification, having enough data will usually have enough entropy to account for enough variance that should exemplify a disparity is cross validation. Additionally, you can use a one-hold-out dataset for extra assurance. You can be more scientific like using "Minimum Description Length principle" which is something related to the size of the error vs the size of the tree but that's getting a bit in the weeds.

- Ensure you have enough data.

It's possible that you don't have enough representitive data (more to my first points in this recommendation). There may not be enough examples to describe a specific case. Perhaps you're classifying houses vs apartments and you don't have enough data on apartments within a given range of values such as square feet and bedrooms, the model may not learn that apartments above 2 bedrooms and 2000 square feet in the city can be either house or apartment but is less likely to be an apartement if there are more houses in the dataset than there are in real life, the decision tree will consider the information gain in this range of variables to describe a split on assumptions it can only conclude with the data it observes. I can't think of a better example…

CHECK FOR CLASS IMBALANCE!

- Reduce the complexity of the decision tree model

There's a great quote for this: "Everything should be made as simple as possible, but no simpler."

Pruning a tree can be done before or after, but in sklearn, pre-pruning isn't possible, but you can choose to set the minimum amount of data that is required to create a leaf node, which will additionally qualify the conditions on which a split can occur. Additinally, one can set the max-

depth to control the complexity, limiting quantity of nodes quite a bit — you could adjust this paramter and check cross-validation each time after you've gridsearched a range that tends to perform well on the classification metric of your choice.

- Use decision trees in an ensemble

  Since decision trees are greedy, they are also prone to finding locally optimal solutions without considering a broader assumption about data. This is one reason (in my opinion), that makes these ideal for ensembles. Ensemble methods (bagging / random forrests, boosting, etc) that allows for weighting different aspects of decision trees (with bootstrapping, with random variable selection, with weighting of weak learners, with sample weight selection.. these are the main aspects that different ensembles mix and match to generalize better)

- Reduce the dimensionality of your data

Additionally, choose better features. Reducing the complexity of a model can also be done if you reduce the complexity of your data. The potential splits (ie: complexity), can be reduced before you even put your data to the model.

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 in machine learning
   usually produce more accurate solutions than a single model would.

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 RandomForestClassifier 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.

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

    Ans-
    Hyperparameter tuning is the process of selecting the optimal values for a machine learning model's hyperparameters. Hyperparameters are settings that control the learning process of the model, such as the learning rate, the number

of neurons in a neural network, or the kernel size in a support vector machine. The goal of hyperparameter tuning is to find the values that lead to the best performance on a given task.
In the context of machine learning, hyperparameters are configuration variables that are set before the training process of a model begins. They control the learning process itself, rather than being learned from the data. Hyperparameters are often used to tune the performance of a model, and they can have a significant impact on the model's accuracy, generalization, and other metrics.
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.

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

Ans- 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.

1. Can we use Logistic Regression for classification of Non-Linear Data? If not, why?
Ans
Logistic regression is a classification algorithm used to find the probability of event success and event failure. It is used when the dependent variable is binary(0/1, True/False, Yes/No) in nature. It supports categorizing data into discrete classes by studying the relationship from a given set of labelled data. It learns a linear relationship from the given dataset and then introduces a non-linearity in the form of the Sigmoid function.
Logistic regression is also known as Binomial logistics regression. It is based on sigmoid function where output is probability and input can be from -infinity to +infinity. Let's discuss some advantages and disadvantages of Linear Regression.

Advantages	Disadvantages
Logistic regression is easier to implement, interpret, and very efficient to train.	If the number of observations is lesser than the number of features, Logistic Regression should not be used, otherwise, it may lead to overfitting.
It makes no assumptions about distributions of classes in feature space.	It constructs linear boundaries.
It can easily extend to multiple classes(multinomial regression) and a natural probabilistic view of class predictions.	The major limitation of Logistic Regression is the assumption of linearity between the dependent variable and the independent variables.
It not only provides a measure of how appropriate a predictor(coefficient size)is, but also its direction of association (positive or negative).	It can only be used to predict discrete functions. Hence, the dependent variable of Logistic Regression is bound to the discrete number set.
It is very fast at classifying unknown records.	Non-linear problems can't be solved with logistic regression because it has a linear decision surface. Linearly separable data is rarely found in real-world scenarios.
Good accuracy for many simple data sets and it performs well when the dataset is linearly separable.	Logistic Regression requires average or no multicollinearity between independent variables.
It can interpret model coefficients as indicators of feature importance.	It is tough to obtain complex relationships using logistic regression. More powerful and compact algorithms such as Neural Networks can easily outperform this algorithm.
Logistic regression is less inclined to over-fitting but it can overfit in high dimensional datasets.One may consider Regularization (L1	In Linear Regression independent and dependent variables are related linearly. But Logistic Regression needs that independent

Advantages	Disadvantages
and L2) techniques to avoid over-fittingin these scenarios.	variables are linearly related to the log odds (log(p/(1-p)).

2. Differentiate between Adaboost and Gradient Boosting.

Ans-
AdaBoost AdaBoost or Adaptive Boosting is the first Boosting ensemble model. The method automatically adjusts its parameters to the data based on the actual performance in the current iteration. Meaning, both the weights for re-weighting the data and the weights for the final aggregation are re-computed iteratively. In practice, this boosting technique is used with simple classification trees or stumps as base-learners, which resulted in improved performance compared to the classification by one tree or other single base-learner.

Gradient Boosting
Gradient Boost is a robust machine learning algorithm made up of Gradient descent and Boosting. The word 'gradient' implies that you can have two or more derivatives of the same function. Gradient Boosting has three main components: additive model, loss function and a weak learner.
The technique yields a direct interpretation of boosting methods from the perspective of numerical optimisation in a function space and generalises them by allowing optimisation of an arbitrary loss function.
The Comparison
Loss Function: The technique of Boosting uses various loss functions. In case of Adaptive Boosting or AdaBoost, it minimises the exponential loss function that can make the algorithm sensitive to the outliers. With Gradient Boosting, any differentiable loss function can be utilised. Gradient Boosting algorithm is more robust to outliers than AdaBoost.
Flexibility
AdaBoost is the first designed boosting algorithm with a particular loss function. On the other hand, Gradient Boosting is a generic algorithm that assists in searching the approximate solutions to the additive modelling problem. This makes Gradient Boosting more flexible than AdaBoost.
Benefits
AdaBoost minimises loss function related to any classification error and is best used with weak learners. The method was mainly designed for binary classification problems and can be utilised to boost the performance of decision trees. Gradient Boosting is used to solve the differentiable loss function problem. The technique can be used for both classification and regression problems.
Shortcomings
In the case of Gradient Boosting, the shortcomings of the existing weak learners can be identified by gradients and with AdaBoost, it can be identified by high-weight data points.
Wrapping Up -Though there are several differences between the two boosting methods, both the algorithms follow the same path and share similar historic roots. Both the algorithms work for boosting the performance of a simple base-learner by iteratively shifting the focus towards problematic observations that are challenging to predict.
In the case of AdaBoost, the shifting is done by up-weighting observations that were misclassified before, while Gradient Boosting identifies the difficult observations by large residuals computed in the previous iterations.

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

Ans- Bias- The bias is known as the difference between the prediction of the values by the Machine Learning model and the correct value. Being high in biasing gives a large error in training as well as testing data. It recommended that an algorithm should always be low-biased to avoid the problem of underfitting. By high bias, the data predicted is in a straight line format, thus not fitting accurately in the data in the data set. Such fitting is known as the Underfitting of Data.
The variability of model prediction for a given data point which tells us the spread of our data is called the variance of the model. The model with high variance has a very complex fit to the training data and thus is not able to fit accurately on the data which it hasn't seen before. As a result, such models perform very well on training data but have high error rates on test data. When a model is high on variance, it is then said to as Overfitting of Data. Overfitting is fitting the training set accurately via complex curve and high order hypothesis but is not the solution as the error with unseen data is high.

While training a data model variance should be kept low.

Bias Variance Tradeoff

If the algorithm is too simple (hypothesis with linear equation) then it may be on high bias and low variance condition and thus is error-prone. If algorithms fit too complex (hypothesis with high degree equation) then it may be on high variance and low bias. In the latter condition, the new entries will not perform well. Well, there is something between both of these conditions, known as a Trade-off or Bias Variance Trade-off. This tradeoff in complexity is why there is a tradeoff between bias and variance. An algorithm can't be more complex and less complex at the same time. For the graph, the perfect tradeoff will be like this.

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

Ans-

Linear Kernel- It 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 It 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)

It 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.

12. .