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 - However, a smaller or lower value for the RSS is ideal in any model since it means there's less variation in the data set. In other words, the lower the sum of squared residuals, the better the regression model is at explaining the data.

   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.

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- Toxic shock syndrome (TSS) is a cluster of symptoms that involves many systems of the body. Certain bacterial infections release toxins into the bloodstream, which then spreads the toxins to body organs. This can cause severe damage and illness

   Employee self-service (ESS) is a widely used human resources technology that enables employees to perform many job-related functions, such as applying for reimbursement, updating personal information and accessing company benefits information -- which was once largely paper-based, or otherwise would have been maintained by management or administrative staff..

An RSS feed is a set of instructions residing on the computer server of a website, which is given upon request to a subscriber's RSS reader, or aggregator. The feed tells the reader when new material—such as a news article, a blog posting, or an audio or a video clip—has been published on the website. The aggregator monitors any number of sites' feeds and centrally organizes and displays the new material for the user. The user then has a single source where all of the latest content is automatically available.

Goodness of fit: **R2**. **TSS = ESS + RSS**, where TSS is Total Sum of Squares, ESS is Explained Sum of Squares and RSS is Residual Sum of Suqares. The aim of Regression Analysis is explain the variation of dependent variable Y.

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.

To avoid this, we use regularization in machine learning to properly fit the model to our test set. Regularization techniques help reduce the possibility of overfitting and help us obtain an optimal model.

This is a form of regression, that constrains/ regularizes or shrinks the coefficient estimates towards zero. In other words, this

technique **discourages learning a more complex or flexible model, so as to avoid the risk of overfitting**.

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.

For example, say you want to build a classifier that determines if someone will default on their credit card. You have some labelled data with features, such as bins for age, income, credit rating, and whether or not each person is a student. To find the best feature for the first split of the tree – the root node – you could calculate how poorly each feature divided the data into the correct class, default ("yes") or didn't default ("no"). This calculation would measure the impurity of the split, and the feature with the lowest impurity would determine the best feature for splitting the current node. This process would continue for each subsequent node using the remaining features.

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

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.

I recommend the 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.

6. What is an ensemble technique in machine learning?

ANS - The ensemble methods in machine learning combine the insights obtained from multiple learning models to facilitate accurate and improved decisions. These methods follow the same principle as the example of buying an air-conditioner cited above.

In learning models, noise, variance, and bias are the major sources of error. The ensemble methods in machine learning help minimize these error-causing factors, thereby ensuring the accuracy and stability of machine learning (ML) algorithms.

Similarly, ensemble methods in machine learning employ a set of models and take advantage of the blended output, which,

compared to a solitary model, will most certainly be a superior option when it comes to prediction accuracy.

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

ANS-

S.NO	Bagging	Boosting
1.	The simplest way of combining predictions that belongs to the same type.	A way of combining predictions that belong to the different types.
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 previously built models.
5.	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.
6.	Bagging tries to solve the over-fitting problem.	Boosting tries to reduce bias.
7.	If the classifier is unstable (high variance), then apply bagging.	If the classifier is stable and simple (high bias) the apply boosting.
8.	In this base classifiers are trained parallelly.	In this base classifiers are trained sequentially.

S.NO	Bagging	Boosting
9	Example: The Random forest model uses Bagging.	Example: The AdaBoost uses Boosting techniques

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

ANS- Out of bag (OOB) score is a way of validating the Random forest model. Below is a simple intuition of how is it calculated followed by a description of how it is different from validation score and where it is advantageous.

Out-of-bag (OOB) error, also called out-of-bag estimate, is a method of measuring the prediction error of random forests, boosted decision trees, and other machine learning models utilizing bootstrap aggregating (bagging). Bagging uses subsampling with replacement to create training samples for the model to learn from. OOB error is the mean prediction error on each training sample x_i, using only the trees that did not have x_i in their bootstrap sample.[1]

Bootstrap aggregating allows one to define an out-of-bag estimate of the prediction performance improvement by evaluating predictions on those observations that were not used in the building of the next base learner.

9. What is K-fold cross-validation?

ANS- Cross validation is an evaluation method used in machine learning to find out how well your machine learning model can predict the outcome of unseen data. It is a method that is easy to comprehend; works well for a limited data sample and also offers an evaluation that is less biased, making it a popular choice.

The data sample is split into ‘k’ number of smaller samples, hence the name: K-fold Cross Validation. You may also hear terms like fourfold cross validation, or tenfold cross validation, which essentially means that the sample data is being split into four or ten smaller samples respectively.

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

ANS -In machine learning, we need to differentiate between parameters and hyperparameters. A learning algorithm learns or estimates model parameters for the given data set, and then continues updating these values as it continues to learn. After learning is complete, these parameters become part of the model. For example, each weight and bias in a neural network is a parameter.

Hyperparameters, on the other hand, are specific to the algorithm itself, so we can’t calculate their values from the data. We use hyperparameters to calculate the model parameters. Different hyperparameter values produce different model parameter values for a given data set.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. Note that the learning algorithm optimizes the loss based on the input data and tries to find an optimal solution within

the given setting. However, hyperparameters describe this setting exactly.

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

ANS- The learning rate can see as step size, η. As such, gradient descent is taking successive steps in the direction of the minimum. If the step size η is too large, it can (plausibly) "jump over" the minima we are trying to reach, ie. we overshoot. This can lead to osculations around the minimum or in some cases to outright divergence. It is important to note that the step gradient descent takes is a function of step size η as well as the gradient values g. If we are in a local minimum with zero gradient the algorithm will not update the

Parameters p because the gradient is zero, similarly if p is in a "steep slope", even a small η will lead to a large update in p's values.

Particular for the case of divergence what happens is that as soon as an oversized step η is taken from an initial point $pi=0=0$, the gradient descent algorithm lands to a point $pi=1=1$ that is worse than $pi=0=0$ in terms of cost. At this new but cost function-wise worse point $pi=1=1$, when recalculating the gradients, the gradient values are increased, so the next (hopefully corrective) step is even larger. Nevertheless if this next step leads to a point $pi=2=2$ with even larger error because we overshoot again, we can be led to use even larger gradient values, leading ultimately to a vicious cycle of ever increasing gradient values and "exploding coefficients" pi.

In the code you provided you might wish add a $\text{print}(\text{gradient}(X, y, p))$ statement in the param_update function? If that is add, we can monitor the gradient in each iteration and see that in the case of a reasonably valued η the gradient values slowly decrease while in

the case of unreasonably large η the gradient values get steadily larger and larger.

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.

13. Differentiate between Adaboost and Gradient Boosting.

ANS-

Features	**Gradient boosting**	**Adaboost**
Model	**It identifies complex observations by huge residuals calculated in prior**	**The shift is made by up-weighting the observations that are miscalculated prior**

	iterations	
Trees	**The trees with week learners are constructed using a greedy algorithm based on split points and purity scores. The trees are grown deeper with eight to thirty-two terminal nodes. The week learners should stay a week in terms of nodes, layers, leaf nodes, and splits**	**The trees are called decision stumps.**
Classifier	**The classifiers are weighted precisely and their prediction capacity is constrained to learning rate and increasing accuracy**	**Every classifier has different weight assumptions to its final prediction that depend on the performance.**

Prediction	**It develops a tree with help of previous classifier residuals by capturing variances in data. The final prediction depends on the maximum vote of the week learners and is weighted by its accuracy.**	**It gives values to classifiers by observing determined variance with data. Here all the week learners possess equal weight and it is usually fixed as the rate for learning which is too minimum in magnitude.**
Short-comings	**Here, the gradients themselves identify the shortcomings.**	**Maximum weighted data points are used to identify the shortcomings.**
Loss value	**Gradient boosting cut down the error components to provide clear explanations and its concepts are easier to**	**The exponential loss provides maximum weights for the samples which are fitted in worse conditions.**

	adapt and understand	
Applications	**This method trains the learners and depends on reducing the loss functions of that week learner by training the residues of the model**	**Its focus on training the prior miscalculated observations and it alters the distribution of the dataset to enhance the weight on sample values which are hard for classification**

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

ANS -If our model is too simple and has very few parameters then it may have high bias and low variance. On the other hand if our model has large number of parameters then it's going to have high variance and low bias. So we need to find the right/good balance without overfitting and underfitting the data.

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.

Total Error

To build a good model, we need to find a good balance between bias and variance such that it minimizes the total error.

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

ANS - Linear SVM: Linear SVM is used for **linearly separable data**, which means if a dataset can be classified into two classes by using a single straight line, then such data is termed as linearly separable data, and classifier is used called as Linear SVM classifier.

RBF short for **Radial Basis Function Kernel** is a very powerful kernel used in SVM. Unlike linear or polynomial kernels, RBF is more complex and efficient at the same time that it can combine multiple polynomial kernels multiple times of different degrees to project the non-linearly separable data into higher dimensional space so that it can be separable using a hyperplane.

In machine learning, 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.

Intuitively, the polynomial kernel looks not only at the given features of input samples to determine their similarity, but also combinations of these. In the context of regression analysis, such combinations are known as interaction features. The (implicit) feature space of a polynomial kernel is equivalent to that of polynomial regression, but without the combinatorial blowup in the number of parameters to be learned. When the input features are binary-valued (booleans), then the features correspond to logical conjunctions of input features