1. The sum of squares is a statistical technique used in regression analysis to determine the dispersion of data points. In a regression analysis, the goal is to determine how well a data series can be fitted to a function that might help to explain how the data series was generated. The sum of squares is used as a mathematical way to find the function that best fits (varies least) from the data. The RSS measures the amount of error remaining between the regression function and the data set after the model has been run. A smaller RSS figure represents a regression function that is well-fit to the data. The RSS, also known as the sum of squared residuals, essentially determines how well a regression model explains or represents the data in the model.

   $\text{RSS} = \sum^{n}_{i=1} (y^{i} - f(x_{i}))^{2}$

   Where:

   y_i = the ith value of the variable to be predicted

   $f(x_i)$ = predicted value of y_i

   n = upper limit of summation

   The residual standard error (RSE) is another statistical term used to describe the difference in standard deviations of observed values versus predicted values as shown by points in a regression analysis. It is a goodness-of-fit measure that can be used to analyze how well a set of data points fit with the actual model.

   RSE is computed by dividing the RSS by the number of observations in the sample less 2, and then taking the square root: $\text{RSE} = [\text{RSS}/(n-2)]^{1/2}$

2. The Total SS (TSS or SST) tells you how much variation there is in the dependent variable. Total SS = Σ(Yi – mean of $Y)^2$.
   Note: Sigma (Σ) is a mathematical term for summation or "adding up." It's telling you to add up all the possible results from the rest of the equation.
   Sum of squares is a measure of how a data set varies around a central number (like the mean). You might realize by the phrase that you're summing (*adding up*) squares—but squares of what? You'll sometimes see this formula:

   $$y = Y - \bar{Y}$$

   The Explained SS tells you how much of the variation in the dependent variable your model explained.
   Explained SS = Σ(Y-Hat – mean of $Y)^2$.
   The residual sum of squares tells you how much of the dependent variable's variation your model **did not explain**. It is the sum of the squared differences between the actual Y and the predicted Y:
   Residual Sum of Squares = Σ e2

   Total SS is related to the total sum and explained sum with the following formula:
   Total SS = Explained SS + Residual Sum of Squares

3. Regularization is one of the most important concepts of machine learning. It is a technique to prevent the model from overfitting by adding extra information to it. Sometimes the machine learning model performs well with the training data but does not perform well with the test data. It means the model is not able to predict the output when deals with unseen data by introducing noise in the output, and hence the model is called overfitted. This problem can be deal with the

help of a regularization technique. This technique can be used in such a way that it will allow to maintain all variables or features in the model by reducing the magnitude of the variables. Hence, it maintains accuracy as well as a generalization of the model. It mainly regularizes or reduces the coefficient of features toward zero. In simple words, "*In regularization technique, we reduce the magnitude of the features by keeping the same number of features.*"

Difference between Ridge Regression and Lasso Regression

Ridge regression is mostly used to reduce the overfitting in the model, and it includes all the features present in the model. It reduces the complexity of the model by shrinking the coefficients.

Lasso regression helps to reduce the overfitting in the model as well as feature selection.

4. 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.

Consider a dataset D that contains samples from k classes. The probability of samples belonging to class i at a given node can be denoted as pi. Then the Gini Impurity of D is defined as:

Gini(D)= 1- $\sum_{i=1}^{k} pi^2$

The node with uniform class distribution has the highest impurity. The minimum impurity is obtained when all records belong to the same class.

5. Overfitting refers to the condition when the model completely fits the training data but fails to generalize the testing unseen data. Overfit condition arises when the model memorizes the noise of the training data and fails to capture important patterns. A perfectly fit decision tree performs well for training data but performs poorly for unseen test data. If the decision tree is allowed to train to its full strength, the model will overfit the training data. There are various techniques to prevent the decision tree model from overfitting.

6. 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. This has been the case in a number of machine learning competitions, where the winning solutions used ensemble methods. In the popular Netflix Competition, the winner used an ensemble method to implement a powerful collaborative filtering algorithm.

7. 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.

Difference Between Bagging and Boosting: Bagging vs Boosting

	Bagging	**Boosting**
Basic Concept	Combines multiple models trained on different subsets of data.	Train models sequentially, focusing on the error made by the previous model.
Objective	To reduce variance by averaging out individual model error.	Reduces both bias and variance by correcting misclassifications of the previous model.

Data Sampling	Use Bootstrap to create subsets of the data.	Re-weights the data based on the error from the previous model, making the next models focus on misclassified instances.
Model Weight	Each model serves equal weight in the final decision.	Models are weighted based on accuracy, i.e., better-accuracy models will have a higher weight.
Error Handling	Each model has an equal error rate.	It gives more weight to instances with higher error, making subsequent model focus on them.
Overfitting	Less prone to overfitting due to average mechanism.	Generally not prone to overfitting, but it can be if the number of the model or the iteration is high.
Performance	Improves accuracy by reducing variance.	Achieves higher accuracy by reducing both bias and variance.
Common Algorithms	Random Forest	AdaBoost, XGBoost, Gradient Boosting Mechanism
Use Cases	Best for high variance, and low bias models.	Effective when the model needs to be adaptive to errors, suitable for both bias and variance errors.

8. In machine learning and data science, it is crucial to create a trustful system that will work well with the new, unseen data. Overall, there are a lot of different approaches and methods to achieve this generalization. Out-of-bag error is one of these methods for validating the machine learning model. This approach utilizes the usage of bootstrapping in the random forest. Since the bootstrapping samples the data with the possibility of selecting one sample multiple times, it is very likely that we won't select all the samples from the original data set. Therefore, one smart decision would be to exploit somehow these unselected samples, called out-of-bag samples. Correspondingly, the error achieved on these samples is called out-of-bag error. What we can do is to use out-of-bag samples for each decision tree to measure its performance. This strategy provides reliable results in comparison to other validation techniques such as train-test split or cross-validation.

9. K-fold cross-validation approach divides the input dataset into K groups of samples of equal sizes. These samples are called **folds**. For each learning set, the prediction function uses k-1 folds, and the rest of the folds are used for the test set. it is easy to understand, and the output is less biased than other methods.

The steps for k-fold cross-validation are:

- Split the input dataset into K groups
- For each group:
    - Take one group as the reserve or test data set.
    - Use remaining groups as the training dataset
    - Fit the model on the training set and evaluate the performance of the model using the test set.

10. Hyperparameter tuning is an essential part of controlling the behavior of a machine learning model. If we don't correctly tune our hyperparameters, our estimated model parameters produce suboptimal results, as they don't minimize the loss function. This means our model makes more

errors. In practice, key indicators like the accuracy or the confusion matrix will be worse. A learning algorithm learns or estimates model parameters for the given data set, 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.

For instance, if we work on natural language processing (NLP) models, we probably use neural networks, support-vector machines (SVMs), Bayesian networks, and Extreme Gradient Boosting (XGB) for tuning parameters.

11. The process of repeatedly nudging an input of a function by some multiple of the negative gradient is called gradient descent. It's a way to converge towards some local minimum of a cost function basically valley in a graph. According to Wikipedia, **Gradient descent** is a first-order iterative optimization algorithm for finding the minimum of a function. To find a local minimum or minimum cost of a function using gradient descent, one takes steps proportional to the negative of the **gradient** of the function at the current point.

    Gradient descent is a first-order iterative optimization algorithm for finding the minimum of a function.

    Gradient Descent is a simple optimization technique that could be used in many machine learning problems. It involves reducing the cost function. The cost function could be anything like least square methods, cross entropy. The **cost function** is the relation between calculated output and actual output.

12. Logistic Regression has traditionally been used as a linear classifier, i.e. when the classes can be separated in the feature space by linear boundaries. That can be remedied however if we happen to have a better idea as to the shape of the decision boundary…

    Logistic regression is known and used as a linear classifier. It is used to come up with a hyperplane 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.

13.

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

	terminal nodes. The week learners should stay a week in terms of nodes, layers, leaf nodes, and splits	
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 adapt and understand	The exponential loss provides maximum weights for the samples which are fitted in worse conditions.
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. While building the machine learning model, it is really important to take care of bias and variance in order to avoid overfitting and underfitting in the model. If the model is very simple with fewer parameters, it may have low variance and high bias. Whereas, if the model has a large number of parameters, it will have high variance and low bias. So, it is required to make a balance between bias and variance errors, and this balance between the bias error and variance error is known as **the Bias-Variance trade-off.**

For an accurate prediction of the model, algorithms need a low variance and low bias. But this is not possible because bias and variance are related to each other:

- If we decrease the variance, it will increase the bias.
- If we decrease the bias, it will increase the variance.

Bias-Variance trade-off is a central issue in supervised learning. Ideally, we need a model that accurately captures the regularities in training data and simultaneously generalizes well with the unseen dataset. Unfortunately, doing this is not possible simultaneously. Because a high variance algorithm may perform well with training data, but it may lead to overfitting to noisy data. Whereas, high bias algorithm generates a much simple model that may not even capture important regularities in the data. So, we need to find a sweet spot between bias and variance to make an optimal model.

Hence, the ***Bias-Variance trade-off is about finding the sweet spot to make a balance between bias and variance errors.***

15. VM 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.*** Introduce Kernel functions for sequence data, graphs, text, images, as well as vectors. The most used type of kernel function is **RBF.** Because it has localized and finite response along the entire x-axis.
    The kernel functions return the inner product between two points in a suitable feature space. Thus by defining a notion of similarity, with little computational cost even in very high-dimensional spaces.