**General Linear Model:**

1. What is the purpose of the General Linear Model (GLM)?

Ans:The purpose of the general linear model is to describe the general multivariate regression model in compact way of simultaneously writing several multiple linear regression models.

2. What are the key assumptions of the General Linear Model?

Ans: Following are the key assumptions of the general linear model:

(a)Linearity:

The linear regression model forces the prediction to be a linear combination of features, which is both its greatest strength and its greatest limitation. Linearity leads to interpretable models. Linear effects are easy to quantify and describe. They are additive, so it is easy to separate the effects. If you suspect feature interactions or a nonlinear association of a feature with the target value, you can add interaction terms or use regression splines.

(b)Normality:

It is assumed that the target outcome given the features follows a normal distribution. If this assumption is violated, the estimated confidence intervals of the feature weights are invalid.

(c)Homoscedasticity (constant variance):

The variance of the error terms is assumed to be constant over the entire feature space. Suppose you want to predict the value of a house given the living area in square meters. You estimate a linear model that assumes that, regardless of the size of the house, the error around the predicted response has the same variance. This assumption is often violated in reality. In the house example, it is plausible that the variance of error terms around the predicted price is higher for larger houses, since prices are higher and there is more room for price fluctuations. Suppose the average error (difference between predicted and actual price) in your linear regression model is 50,000 Euros. If you assume homoscedasticity, you assume that the average error of 50,000 is the same for houses that cost 1 million and for houses that cost only 40,000. This is unreasonable because it would mean that we can expect negative house prices.

(d)Independence:

It is assumed that each instance is independent of any other instance. If you perform repeated measurements, such as multiple blood tests per patient, the data points are not independent. For dependent data you need special linear regression models, such as mixed effect models or GEEs. If you use the “normal” linear regression model, you might draw wrong conclusions from the model.

(e)Fixed features:

The input features are considered “fixed”. Fixed means that they are treated as “given constants” and not as statistical variables. This implies that they are free of measurement errors. This is a rather unrealistic assumption. Without that assumption, however, you would have to fit very complex measurement error models that account for the measurement errors of your input features. And usually you do not want to do that.

(f)Absence of multicollinearity

You do not want strongly correlated features, because this messes up the estimation of the weights. In a situation where two features are strongly correlated, it becomes problematic to estimate the weights because the feature effects are additive and it becomes indeterminable to which of the correlated features to attribute the effects.

3. How do you interpret the coefficients in a GLM?

Ans: The interpretation of a coefficients in the linear regression model depends on the type of the corresponding feature:

(a)Numerical feature: Increasing the numerical feature by one unit changes the estimated outcome by its weight. An example of a numerical feature is the size of a house.

Binary feature: A feature that takes one of two possible values for each instance. An example is the feature “House comes with a garden”. One of the values counts as the reference category (in some programming languages encoded with 0), such as “No garden”. Changing the feature from the reference category to the other category changes the estimated outcome by the feature’s weight.

(b)Categorical feature with multiple categories: A feature with a fixed number of possible values. An example is the feature “floor type”, with possible categories “carpet”, “laminate” and “parquet”. A solution to deal with many categories is the one-hot-encoding, meaning that each category has its own binary column. For a categorical feature with L categories, you only need L-1 columns, because the L-th column would have redundant information (e.g. when columns 1 to L-1 all have value 0 for one instance, we know that the categorical feature of this instance takes on category L). The interpretation for each category is then the same as the interpretation for binary features..

(c)Intercept β : The intercept is the feature weight for the “constant feature”, which is always 1 for all instances. Most software packages automatically add this “1”-feature to estimate the intercept. The interpretation is: For an instance with all numerical feature values at zero and the categorical feature values at the reference categories, the model prediction is the intercept weight. The interpretation of the intercept is usually not relevant because instances with all features values at zero often make no sense. The interpretation is only meaningful when the features have been standardised (mean of zero, standard deviation of one). Then the intercept reflects the predicted outcome of an instance where all features are at their mean value

4. What is the difference between a univariate and multivariate GLM?

Ans:Univariate GLM have single feature and single y or target whereas multivariate GLM have multiple feature and single y or target.

5. Explain the concept of interaction effects in a GLM.

Ans: Interaction effect means that two or more features/variables combined have a significantly larger effect on a feature as compared to the sum of the individual variables alone. This effect is important to understand in GLM as we try to study the effect of several variables on a single response variable.

6. How do you handle categorical predictors in a GLM?

Ans: If our categorical predict have only two categories then we treat categorical predictor column as it is.But if categorical predictior has more than 2 categories then we create as many new columns as we have categories in the categorical predictor column.Each of these new column have 0 and 1 as value in them.We have two algorithms for doing this,one algorithm is ordinal encoding and another one is one hot encoding.The ordinal encoding is done when the categorical values in the predictors are ordinal in nature.When the categorical values in the predictors are nominal in nature then we do one hot encoding.

7. What is the purpose of the design matrix in a GLM?

Ans: The design matrix contains data on the independent variables (also called explanatory variables) in statistical models which attempt to explain observed data on a response variable (often called a dependent variable) in terms of the explanatory variables. The theory relating to such models makes substantial use of matrix manipulations involving the design matrix: see for example linear regression. A notable feature of the concept of a design matrix is that it is able to represent a number of different experimental designs and statistical models.

In the following expression:  
Y = XB + U

Y is the prediction vector

X is the design matrix

B is the matrix of coefficients

U is the matrix containing error values

8. How do you test the significance of predictors in a GLM?

Ans: We have many ways of testing the significance of predictors in a GLM.Following are some of the techniques to test the significance of predictors:  
1.Filter based technique:These techniques study the predictors individually on the basis of some statistical measures and try to find our significance of the predictor.Eg: ANOVA,Chisquare test,correlation,variance threshold etc.

2.Wrapper method: Wrapper method use predictive modelling to find out the significane of the predictors.Eg: Exhaustive feature selection,Forward selection,Backward elimination,Recursive feature elimination.

3.Embedded techniques: In an embedded method, feature selection is integrated or built into the classifier algorithm. During the training step, the classifier adjusts its internal parameters and determines the appropriate weights/importance given for each feature to produce the best classification accuracy.Eg: L1 reguliarized regression,decision trees,random feature importance.

9. What is the difference between Type I, Type II, and Type III sums of squares in a GLM?

Ans:

**Type I: sequential**

The SS for each factor is the incremental improvement in the error SS as each factoreffect is added to the regression model. In other words it is the effect as the factor wereconsidered one at a time into the model, in the order they are entered in the modelselection , for example A, B, C, and D in a 4-way ANOVA. The SS can also be viewed asthe reduction in residual sum of squares (SSE) obtained by adding that term to a fit thatalready includes the terms listed before it.

**Type 2: Hierarchical or partially sequential**SS is the reduction in residual error due to adding the term to the model after all otherterms except those that contain it, or the reduction in residual sum of squares obtained byadding that term to a model consisting of all other terms that do not contain the term inquestion. An interaction comes into play only when all involved factors are included inthe model. For example, the SS for main effect of factor A is not adjusted for anyinteractions involving A: AB, AC and ABC, and sums of squares for two-wayinteractions control for all main effects and all other two-way interactions, and so on.

**Type3: Marginal or orthogonal**SS gives the sum of squares that would be obtained for each variable if it were enteredlast into the model. That is, the effect of each variable is evaluated after all other factorshave been accounted for. Therefore the result for each term is equivalent to what isobtained with Type I analysis when the term enters the model as the last one in theordering.

10. Explain the concept of deviance in a GLM.

Ans:

The deviance is used to compare two models – in particular in the case of generalized linear models (GLM) where it has a similar role to residual sum of squares from ANOVA in linear models (RSS). Suppose in the framework of the GLM, we have two nested models, M1 and M2

**Regression:**

11. What is regression analysis and what is its purpose?

Ans: Regression analysis is a set of statistical processes for estimating the relationships between a dependent variable and one or more independent variables. The most common form of regression analysis is linear regression.

Regression analysis is primarily used for two conceptually distinct purposes. First, regression analysis is widely used for prediction and forecasting, where its use has substantial overlap with the field of machine learning. Second, in some situations regression analysis can be used to infer causal relationships between the independent and dependent variables

12. What is the difference between simple linear regression and multiple linear regression?

Ans: Simple linear regression has single input and single output column.While in case of multiple linear regression we have multiple input columns and single output column.

13. How do you interpret the R-squared value in regression?

Ans: R-squared is a goodness-of-fit measure for linear regression models. This statistic indicates the percentage of the variance in the dependent variable that the independent variables explain collectively.R-squared value lies between 0 and 100%.

Low value of R-squared means that the independent variable is not able to explain the variance in the dependent variable.

High value of R-squared means that the independent variable is able to explain the variance in the dependent variable.

14. What is the difference between correlation and regression?

Ans:

| **Basis of Difference** | **Correlation** | **Regression** |
| --- | --- | --- |
| **Meaning** | Correlation refers to a statistical measure that determines the association or co-relationship between two variables. | Regression depicts how an independent variable serves to be numerically related to any dependent variable. |
| **Utility** | Used for representing the linear relationship existing between two variables. | It is used for fitting the best line and estimating the value of one variable based on its relationship with the other. |
| **Dependent /Independent variables** | There is no difference between the two. Both variables are mutually dependent. | Both variables serve to be different in terms of regression analysis. One variable is independent, while the other is dependent. |
| **Indicator of** | It indicates the extent and way in which two variables make their movements together. | Regression depicts the impact of any unit change in the value of the known variable (x) on the value of the estimated variable (y). |
| **Objective** | To find the numerical value that defines and shows the relationship between variables. | To estimate the values of random variables based on the values shown by fixed variables. |
| **Purpose** | The primary purpose is to predict the most dependable forecasts. | The primary purpose is to predict/ estimate the value of any unknown variable by taking the help of the known variable. |
| **Scope** | Correlation analysis offers limited applications. | Regression analysis provides a broader scope of applications. |
| **Range** | Coefficients may range from -1.00 to +1.00. | If byx > 1, then bxy < 1 in regression analysis. |
| **Responding Nature** | The correlation coefficient serves to be independent of any change of Scale or shift in Origin. | The regression coefficient shows dependency on the change of Scale but is independent of its shift in Origin. |
| **Nature of Coefficient** | The correlation coefficient is mutual and symmetrical. | Regression coefficient fails to be symmetrical. |

15. What is the difference between the coefficients and the intercept in regression?

Ans: Both coefficients and intercept have different.The intercept term prevents the overall bias by forcing the residual mean to equal zero.  
Whereas the coefficient term decides how much a particular input variable affects the or contributes in the prediction.

16. How do you handle outliers in regression analysis?

Ans:Outliers can be handled in following way:  
(a)Trimming: Means we remove the outliers

Issue with trimming is if we have lot of outliers then our data will become thin(less data).

But good thing is that this algorithm is very fast.

(b)Capping: In this we decide lower limit and upper limit value, values less than lower limit value and greater than upper limit values are outliers.Values which are greater than upper limit value,we change those values to upper limit value.

(c)Treat outlier values as missing values and then use methods to fill those missing values.

(d)Apply discretization(makes classes in numerical data)

17. What is the difference between ridge regression and ordinary least squares regression?

Ans: The ridge or L2 regression has additional term in its cost function.It has (weight)^2 term in it.This additional term is not present in the least square regression cost function.This extra term in the ridge regression’s cost function helps to reduce the overfitting in the model.

18. What is heteroscedasticity in regression and how does it affect the model?

Ans: Heteroscedastic refers to a condition in which the variance of the residual term, or error term, in a regression model varies widely.  
Heteroscedasticity makes a regression model less dependable because the residuals should not follow any specific pattern. The scattering should be random around the fitted line for the model to be robust.

19. How do you handle multicollinearity in regression analysis?

Ans: We handle multicollinearity in following way:

**1.** **Collect more data**: In some cases, multicollinearity might be a result of a limited sample

size. Collecting more data, if possible, can help reduce multicollinearity and improve the

stability of the model.

**2.Remove one of the highly correlated variables**: If two or more independent variables are

highly correlated, consider removing one of them from the model. This step can help

eliminate redundancy in the model and reduce multicollinearity. Choose the variable to

remove based on domain knowledge, variable importance, or the one with the highest VIF.

**3. Combine correlated variables**: If correlated independent variables represent similar

information, consider combining them into a single variable. This combination can be done

by averaging, summing, or using other mathematical operations, depending on the context

and the nature of the variables.

**4. Use partial least squares regression (PLS**): PLS is a technique that combines features of both

principal component analysis and multiple regression. It identifies linear combinations of

the predictor variables (called latent variables) that have the highest covariance with the

response variable, reducing multicollinearity while retaining most of the predictive power.

20. What is polynomial regression and when is it used?

Ans: In statistics, polynomial regression is a form of regression analysis in which the relationship between the independent variable x and the dependent variable y is modelled as an nth degree polynomial in x. Polynomial regression fits a nonlinear relationship between the value of x and the corresponding conditional mean of y, denoted E(y |x). Although polynomial regression fits a nonlinear model to the data, as a statistical estimation problem it is linear, in the sense that the regression function E(y | x) is linear in the unknown parameters that are estimated from the data. For this reason, polynomial regression is considered to be a special case of multiple linear regression.

**Loss function:**

21. What is a loss function and what is its purpose in machine learning?

Ans: The loss function is the function that computes the distance between the current output of the algorithm and the expected output. It’s a method to evaluate how your algorithm models the data. It can be categorized into two groups. One for classification (discrete values, 0,1,2…) and the other for regression (continuous values).

A loss function is a measure of how good your prediction model does in terms of being able to predict the expected outcome(or value). We convert the learning problem into an optimization problem, define a loss function and then optimize the algorithm to minimize the loss function.

22. What is the difference between a convex and non-convex loss function?

Ans: Convex problems have only one local optimum point, which is also the global optimum point. The point is maximum or minimum based on the second order derivative of that function. On the other hand, non convex problems have multiple optimum points.

23. What is mean squared error (MSE) and how is it calculated?

Ans:

The Mean Squared Error measures how close a regression line is to a set of data points. It is a risk function corresponding to the expected value of the squared error loss.

Mean square error is calculated by taking the average, specifically the mean, of errors squared from data as it relates to a function.

The Mean Squared Error is calculated as:

MSE = (1/n) \* Σ(actual – forecast)^2

where:

n – sample size

actual – the actual data value

forecast – the predicted data valu

24. What is mean absolute error (MAE) and how is it calculated?

Ans: Mean Absolute Error (MAE) is a measure of the average size of the mistakes in a collection of predictions, without taking their direction into account. It is measured as the average absolute difference between the predicted values and the actual values and is used to assess the effectiveness of a regression model.

The MAE loss function formula:

**MAE = (1/n) Σ(i=1 to n) |y\_i – ŷ\_i|**

where:

n is the number of observations in the dataset.

y\_i is the true value.

ŷ\_i is the predicted value.

25. What is log loss (cross-entropy loss) and how is it calculated?

Ans: Cross-entropy loss, or log loss, measures the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss increases as the predicted probability diverges from the actual label.

H(P, Q) = – sum x in X P(x) \* log(Q(x)

Where P(x) is the probability of the event x in P, Q(x) is the probability of event x in Q

26. How do you choose the appropriate loss function for a given problem?

Ans:

If the problem is regression problem then we would choose either of MSE or MAE.If the data has outliers then we would choose MAE else we would choose MSE.This is because MSE is having square term so we can apply optimization algorithms on that.If we want to have best of both worlds then we can choose huber loss.  
If the problem is classification problem then we would choose cross entropy.If classification problem has 2 classes then we use binary cross entropy and if the problem has multiple classes then we use multiclass cross entropy.

27. Explain the concept of regularization in the context of loss functions.

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.

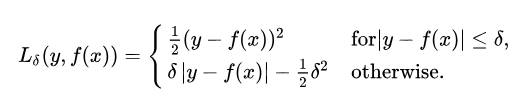
We have 3 types of regularizations:  
(a)L1 or lasso regularization: In this regularization we add (weight)^2 term to the loss function.

(b)L2 or ridge regularization:In this regularization we add |weight| term to the loss function.

(c)Elasticnet regularization: This is combination of L1 and L2 regularization.In this we add (m^2 + |m|) term to the loss function.

28. What is Huber loss and how does it handle outliers?

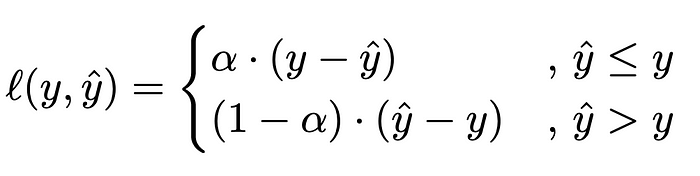
Ans: Huber loss is the combination of both MSE and MAE.



When the loss value is large,means when we are dealing with an outlier then huber loss changes from square formula to linear formula.In this way effect of the outlier is restricted and does not get enlarged as it does in case of MSE.

29. What is quantile loss and when is it used?

Ans: Loss function used to get error value in quantile regression is called as quantile loss.  
Following is the formula for quantile loss:



α is the quantile value, y is the output value and y^ is the predicted value.

30. What is the difference between squared loss and absolute loss?

Ans:Squared loss squares each error value and then takes average whereas the absolute loss just takes magnitude of error and takes average of the sum of all such error magnitudes.

Squared error is good for optimization as it is differentiable.But absolute error is not differentiable so not good for optimization.  
Squared error is not robust to outliers whereas absolute error is robust to outliers.

**Optimizer (GD):**

31. What is an optimizer and what is its purpose in machine learning?

Ans: Optimizers are algorithms or methods used to minimize an error function(loss function)or to maximize the efficiency of production. Optimizers are mathematical functions which are dependent on model’s learnable parameters i.e Weights & Biases. Optimizers help to know how to change weights and learning rate of neural network to reduce the losses.

Purpose of optimizers is to shape and mold your model into its most accurate possible form by futzing with the weights. The loss function is the guide to the terrain, telling the optimizer when it's moving in the right or wrong direction.

32. What is Gradient Descent (GD) and how does it work?

Ans: Gradient Descent is an optimization algorithm for finding a local minimum of a differentiable function. Gradient descent in machine learning is simply used to find the values of a function's parameters (coefficients) that minimize a cost function as far as possible.

Gradient Descent Algorithm iteratively calculates the next point using gradient at the current position, scales it (by a learning rate) and subtracts obtained value from the current position (makes a step).

33. What are the different variations of Gradient Descent?

Ans: There are three types of gradient descent learning algorithms: batch gradient descent, stochastic gradient descent and mini-batch gradient descent.

34. What is the learning rate in GD and how do you choose an appropriate value?

Ans: Learning rate (also referred to as step size or the alpha) is the size of the steps that are taken to reach the minimum. This is typically a small value, and it is evaluated and updated based on the behavior of the cost function.

Usually, we take the value of the learning rate to be 0.1, 0.01 or 0.001. The value of the step should not be too big as it can skip the minimum point and thus the optimisation can fail. It is a hyper-parameter and you need to experiment with its values.

35. How does GD handle local optima in optimization problems?

Ans: To handle local optima we use stochastic gradient descent. The randomness or noise introduced by SG allows to escape from local minima to reach a better minimum.

36. What is Stochastic Gradient Descent (SGD) and how does it differ from GD?

Ans: Stochastic gradient descent (often abbreviated SGD) is an iterative method for optimizing an objective function with suitable smoothness properties (e.g. differentiable or subdifferentiable). It can be regarded as a stochastic approximation of gradient descent optimization, since it replaces the actual gradient (calculated from the entire data set) by an estimate thereof (calculated from a randomly selected subset of the data).  
Gradient descent or batch gradient descent uses whole training data to update the weight once.In case of SGD does update in weight after each input row from the training data.

SGD converges much faster in comparison to Batch gradient descent as it updates weight more frequently.  
SGD requires less computation then Batch gradient descent.  
But issue with SGD is its convergence path is much noisier.

37. Explain the concept of batch size in GD and its impact on training.

Ans:  
The batch size is a hyperparameter of gradient descent that controls the number of training samples to work through before the model's internal parameters are updated.  
A smaller batch size allows the model to learn from each individual example but takes longer to train. A larger batch size trains faster but may result in the model not capturing the nuances in the data.

38. What is the role of momentum in optimization algorithms?

Ans: Momentum aids in the optimization process's convergence by keeping the optimizer going in the same direction as previously, even if the gradient changes direction or becomes zero. This means that the optimizer can take greater steps toward the cost function's minimum, which can help it get there faster.

39. What is the difference between batch GD, mini-batch GD, and SGD?

Ans:

|  |  |  |
| --- | --- | --- |
| **Gradient Descent/Batch Gradient Descent** | **Stochastic Gradient Descent** | **Mini-batch gradient descent** |
| Gradient Descent determines the cost function's gradient throughout the whole training dataset and updates the model's parameters based on the mean of all training examples across each epoch. | Stochastic gradient descent involves updating the model parameters and computing the gradient of the cost function for a single random training example at each iteration. | Mini-batch Gradient Descent updates the model parameters based on the mean gradient of the cost function with respect to the model parameters over a mini-batch, which is a smaller subset of the training dataset of equivalent size. |
| As each iteration of the approach requires computing the gradient of the cost function across the whole training dataset, GD takes some time to converge. | SGD adjusts the model parameters more often than GD, which causes it to converge more quickly | In order to strike a reasonable balance between speed and accuracy, the model parameters are changed more frequently than GD but less frequently than SGD. |
| Due to the requirement to retain the whole training dataset, GD consumes a lot of memory. | As just one training sample needs to be stored for each iteration, SGD requires less memory. | Just a percentage of the training samples had to be retained for each repetition, therefore the memory use is manageable. |
| GD is computationally expensive because the gradient of the cost function must be computed for the whole training dataset at each iteration. | As the cost function's gradient only needs to be calculated once for each repeat of training data, SGD is computationally efficient. | As the gradient of the cost function must be calculated for a portion of the training examples for each iteration, it is computationally efficient. |
| With little error, GD modifies the model's parameters based on the average of all training samples. | Due to the fact that SGD is updated using just one training sample, it has a lot of noise. | Mini-batch Gradient Descent has a significant amount of noise because the update is based on a small number of training examples. |

40. How does the learning rate affect the convergence of GD?

Ans:

If the learning rate is too high then cost function can jump over the minima means it will overshoot and will oscillate around the minima and thus our cost function may fail to converge.If learning rate is very small then it will take lot of time for our model to converge,so it will be computationally expensive and time consuming.

**Regularization**:

41. What is regularization and why is it used 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.

Regularization is used to prevent overfitting or underfitting of the model.Sometimes we use regularization for feature selection as well.

42. What is the difference between L1 and L2 regularization?

Ans: L1 or lasso regularization: In this regularization we add (weight)^2 term to the loss function.L2 or ridge regularization:In this regularization we add |weight| term to the loss function.  
If we increase value of then in case of L1 regularization the weight value can become 0 whereas in case of L2 regularization weights never become 0.This property of L1 regularization is used in feature selection to remove those features which are not that important.  
L1 regularization generates sparse matrix whereas L2 regularization does not generate sparse matrix.

43. Explain the concept of ridge regression and its role in regularization.

Ans: Ridge regression is a regularization technique, which is used to reduce the complexity of the model. It is also called as L2 regularization. In this technique, the cost function is altered by adding the penalty term to it. The amount of bias added to the model is called Ridge Regression penalty.

Ridge regression is also known as L2 regularization.In this regularization we add (weight)^2 term to the cost function.This is done so that we can avoid overfitting.The added term is also called as penalty.

44. What is the elastic net regularization and how does it combine L1 and L2 penalties?

Ans: Elastic net regularization is combination of both L1 and L2 regularization.

Penalty added in elastic net regularization is:  
 ((weight)^2 + |weight|)

45. How does regularization help prevent overfitting in machine learning models?

Ans: Regularization reduces the weight values and sometimes brings weight value to 0.This effectively reduces the contribution of some of the features in the output calculation.In this way it makes model less complex,more flexible and adds little bit randomness and all this reduces the overfitting in the model.

46. What is early stopping and how does it relate to regularization?

Ans: Early Stopping is a regularization technique for deep neural networks that stops training when parameter updates no longer begin to yield improves on a validation set. In essence, we store and update the current best parameters during training, and when parameter updates no longer yield an improvement (after a set number of iterations) we stop training and use the last best parameters. It works as a regularizer by restricting the optimization procedure to a smaller volume of parameter space.

47. Explain the concept of dropout regularization in neural networks.

Ans: Dropout is a regularization method that approximates training a large number of neural networks with different architectures in parallel.

During training, some number of layer outputs are randomly ignored or “dropped out.” This has the effect of making the layer look-like and be treated-like a layer with a different number of nodes and connectivity to the prior layer. In effect, each update to a layer during training is performed with a different “view” of the configured layer.

Dropout has the effect of making the training process noisy, forcing nodes within a layer to probabilistically take on more or less responsibility for the inputs.

This conceptualization suggests that perhaps dropout breaks-up situations where network layers co-adapt to correct mistakes from prior layers, in turn making the model more robust.

48. How do you choose the regularization parameter in a model?

Ans: Regularization parameter is a hyperparameter.So we tune it and we choose that value of regularization parameter for which the model has least value of MSE.

49. Whatis the difference between feature selection and regularization?

Ans: Feature selection is a technique in which we select the best subset of features from the available set of features.Highly correlated features and features which have least correlated with the target are removed.  
The main task of regularization is to solve the problem of overfitting.This is done by adding the penalty term in the cost function.Due to penalty term if features are being removed then it is secondary effect.  
Only L1 regularization is used for feature selection as in case of L1 regularization weight values gets reduced to 0.

50. What is the trade-off between bias and variance in regularized models?

Ans:Bias denotes how well our model has learned from the training data.Variance means difference between prediction values from training and prediction values from test data.  
High bias means model has trained very badly and has not learned much from the training data.High bias means underfitting.  
High variance means there is large difference between prediction values from test data and prediction value from training data.Condition of high variance and low bias means overfitting.  
Bias variance trade off says that as we try to reduce the bias,means as our model tries to learn more from the training data the variance value increases,means model tends to overfit.

As we try to reduce the increase the bias the variance value decreases.Means as we try to reduce the overfitting the making the model little bit more random and by adding more noise, the bias of the model increases.

**SVM**:

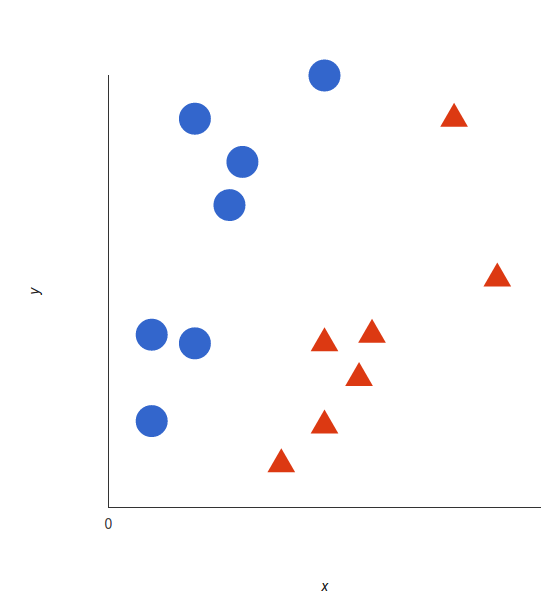
51. What is Support Vector Machines (SVM) and how does it work?

Ans: SVM is a powerful supervised algorithm that works best on smaller datasets but on complex ones. Support Vector Machine,

abbreviated as SVM can be used for both regression and classification tasks, but generally, they work best in classification

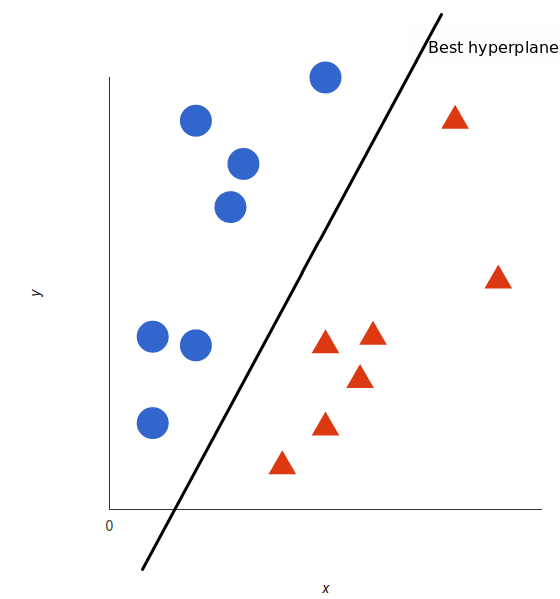
problems.

The basics of Support Vector Machines and how it works are best understood with a simple example. Let’s imagine we have two tags: *red* and *blue*, and our data has two [features](https://monkeylearn.com/blog/practical-explanation-naive-bayes-classifier/#feature-engineering): *x* and *y*. We want a classifier that, given a pair of *(x,y)* coordinates, outputs if it’s either *red* or *blue*. We plot our already labeled training data on a plane:



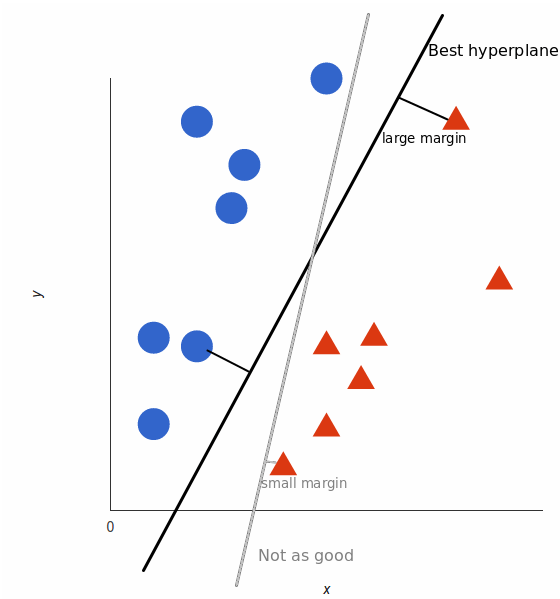
*Our labeled data*

A support vector machine takes these data points and outputs the hyperplane (which in two dimensions it’s simply a line) that best separates the tags. This line is the **decision boundary**: anything that falls to one side of it we will classify as *blue*, and anything that falls to the other as *red*.



*In 2D, the best hyperplane is simply a line*

But, what exactly is *the best* hyperplane? For SVM, it’s the one that maximizes the margins from both tags. In other words: the hyperplane (remember it's a line in this case) whose distance to the nearest element of each tag is the largest.



*Not all hyperplanes are created equal*

52. How does the kernel trick work in SVM?

Ans: If the data is not linearly separable in lower dimension then we use kernel trick on the data.

In kernel trick we convert our lower dimension data into higher dimesion data such that our data in higher dimension becomes linearly separable.Means we can convert our 1D data into 2D data such that the non linearly separable data becomes linearly separable data into the 2d dimension.

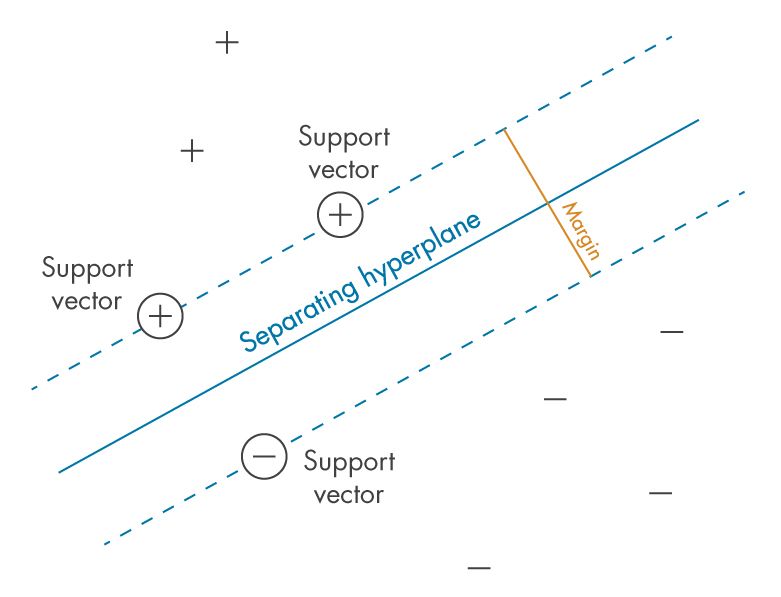
In higher dimension our data will be separable by a plan/hyperplane.

We find out the function which will transform our lower dimension data into higher dimension data,we call this function kernel.

Some examples of kernel functions are: RBF,polynomial,sigmoid.The whole transformation from lower to higher dimension is called as kernel transformation.

53. What are support vectors in SVM and why are they important?

Ans: Support vectors are data points that are closer to the hyperplane and influence the position and orientation of the hyperplane. Using these support vectors, we maximize the margin of the classifier. Deleting the support vectors will change the position of the hyperplane. These are the points that help us build our SVM.



54. Explain the concept of the margin in SVM and its impact on model performance.

Ans: It is the distance between the hyperplane and the observations closest to the hyperplane (support vectors). In SVM large margin is considered a good margin. There are two types of margins hard margin and soft margin.

a large margin can avoid the effect of random noise and reduce overfitting. Second, a larger margin will lead to a smaller VC dimension, reduce the number of potential classifiers, and, therefore, reduce the possibility of generalization error.

55. How do you handle unbalanced datasets in SVM?

Ans: An SVM classifier trained on an imbalanced dataset often produces models which are biased towards the majority class and have low performance on the minority class.  
To handle the unbalanced datasets we use weights in SVM.This type of SVM is called as weighted SVM or cost sensitive SVM.In weighted SVM we provide weights to the regularization parameter.This is done by giving argument to class\_weights parameter.The “balanced” argument to the class\_weights parameter uses the values of y to automatically adjust weights inversely proportional to class frequencies in the input data.We can even provide the dict object having weights for each category value.Usually the class with lower frequency is provided with higher weight and class with higher frequency is provided with lower weight.

56. What is the difference between linear SVM and non-linear SVM?

## Ans:

| **Linear SVM** | **Non-Linear SVM** |
| --- | --- |
| It can be easily separated with a linear line. | It cannot be easily separated with a linear line. |
| Data is classified with the help of hyperplane. | We use Kernels to make non-separable data into separable data. |
| Data can be easily classified by drawing a straight line. | We map data into high dimensional space to classify. |

57. What is the role of C-parameter in SVM and how does it affect the decision boundary?

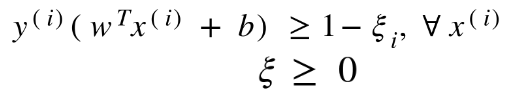
Ans: C is a regularization parameter that controls the trade off between the achieving a low training error and a low testing error that is the ability to generalize your classifier to unseen data.

If we keep high value of this parameter then it means we are asking our model to focus more on correctly classifying and focus less on maximizing distance between positive and negative hyperplane.So result of giving high value to ‘C’ parameter will be that positive and negative hyperplanes are very near to each other but there will be no misclassification of training datapoints.High value of ‘c’ means we are having hard SVM margin

If we give low value to ‘C’ then we are telling the model that more focus should be on maximizing the distance between the positive and negative hyperplane and less focus should be on correctly classifying the data points.So in this case we will have positive and negative hyperplanes at greater distance but there will be some datapoints which are misclassified.Low value of ‘c’ means we are having soft margin svm.

58. Explain the concept of slack variables in SVM.

Ans: In order to become a better classifier, SVM has to have more lenient constraints so that in the feasible region hyperplanes can be allowed to have a functional margin less than 1. To allow this to happen, a slack variable is introduced to each of the constraints of the optimization problem:



***‘ξ’***is the slack variable vector of *ξi’s (ξ*subscripted with *i)*. We define *ξi*for each example in the dataset. Each of its elements has to be atleast zero to really allow some points to have a functional margin less than or equal to 1(the second inequality in the above expression). It has a different value for each example in the training dataset and therefore, a hyperplane may have different minimum requirement for the values of functional margin for different points. This also allows the hyperplane to have a functional margin of less than 1 w.r.t. some data points.

59. What is the difference between hard margin and soft margin in SVM?

Ans: The difference between a hard margin and a soft margin in SVMs lies in the separability of the data. If our data is linearly separable, we go for a hard margin. However, if this is not the case, it won’t be feasible to do that. In the presence of the data points that make it impossible to find a linear classifier, we would have to be more lenient and let some of the data points be misclassified. In this case, a soft margin SVM is appropriate.

Sometimes, the data is linearly separable, but the margin is so small that the model becomes prone to overfitting or being too sensitive to outliers. Also, in this case, we can opt for a larger margin by using soft margin SVM in order to help the model generalize better

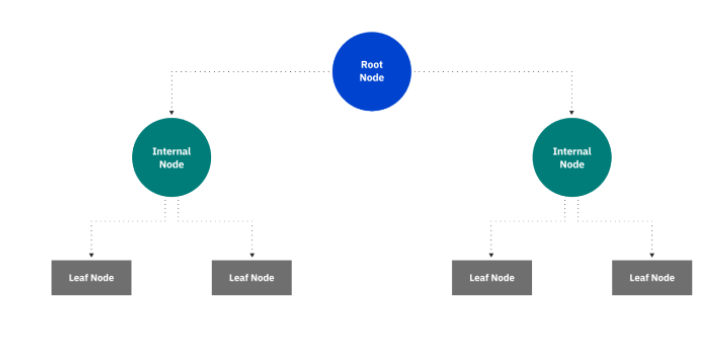
60. How do you interpret the coefficients in an SVM model?

Ans: The coefficients or weights in the SVM are nothing but coordinates of the vector which is perpendicular to the hyperplane.If we take the dot product of any point with the vector, you can tell on which side it is: if the dot product is positive, it belongs to the positive class, if it is negative it belongs to the negative class.

**Decision Trees:**

61. What is a decision tree and how does it work?

Ans: A decision tree is a non-parametric supervised learning algorithm, which is utilized for both classification and regression tasks. It has a hierarchical, tree structure, which consists of a root node, branches, internal nodes and leaf nodes.



A decision tree starts at a single point (or 'node') which then branches (or 'splits') in two or more directions. Each branch offers different possible outcomes, incorporating a variety of decisions and chance events until a final outcome is achieved.

62. How do you make splits in a decision tree?

Ans: We take each input column of the dataset and then we calculate the information gain for that column.The column with highest information gain value we will split that column.This is the way in which we do splitting of categorical columns.

But in case of numerical column,we first sort that numerical column in ascending order and then calculate information gain by taking 1st value of that column,then 2nd value and so on.The value of the column for which information gain value is higher ,we split the column from that value,means one child dataset will have column above that value(including that value) and another child dataset will have remaining values of that column.

63. What are impurity measures (e.g., Gini index, entropy) and how are they used in decision trees?

Ans: Entropy: Entropy is uncertainty/ randomness in the data, the more the randomness the higher will be the entropy. Information gain uses entropy to make decisions. If the entropy is less, information will be more.

Gini impurity: Gini impurity is a function that determines how well a decision tree was split. Basically, it helps us to determine which splitter is best so that we can build a pure decision tree. Gini impurity ranges values from 0 to 0.5.

64. Explain the concept of information gain in decision trees.

Ans:

Information gain is the basic criterion to decide whether a feature should be used to split a node or not. The feature with the optimal split i.e., the highest value of information gain at a node of a decision tree is used as the feature for splitting the node.

Information Gain is calculated for a split by subtracting the weighted entropies of each branch from the original entropy. When training a Decision Tree using these metrics, the best split is chosen by maximizing Information Gain.

65. How do you handle missing values in decision trees?

Ans: We handle missing values in decision tree in following way:  
(a)By removing the rows having missing values.  
(b)By imputing the missing values.In this we can do mean,median values to impute the missing values or we can use multiple imputation as well.  
(c)We can use surrogate split method as well.

66. What is pruning in decision trees and why is it important?

Ans: Pruning is a data compression technique in machine learning and search algorithms that reduces the size of decision trees by removing sections of the tree that are non-critical and redundant to classify instances.  
A fully grown decision tree is usually does overfitting.So to reduce the overfitting we apply pruning.

67. What is the difference between a classification tree and a regression tree?

Ans: The classification tree splits the response variable into mainly two classes Yes or No, also can be numerically categorized as 1 or 0. To apply recursive partitioning on the target category that can contain multiple variables, C4.5 algorithm is leveraged. In case of simple binary splits, CART algorithm is used. This is the reason why classification tree is applied when there is a need for categorical variable for categorical outcome. The regression trees are leveraged in case where the response variable is either continuous or numeric, but not categorical. Regression trees can be applied in case of prices, quantities, or data involving quantities etc.

The classification decision trees are built with unordered values with dependent variables. The regression decision trees take ordered values with continuous values

68. How do you interpret the decision boundaries in a decision tree?

Ans: The decision boundary of a decision tree is a way to visually represent how the tree partitions the feature space to make predictions. Unlike some other classifiers like support vector machines or logistic regression, decision trees do not have a straightforward mathematical representation of the decision boundary.

Since decision trees are non-linear classifiers, their decision boundaries can be quite complex and can include regions of arbitrary shape. The decision boundary is defined by the combination of decisions made at each internal node. It divides the feature space into different regions, where each region corresponds to a specific prediction or class label.

69. What is the role of feature importance in decision trees?

Ans: Feature importance in a decision tree is a measure of the significance or relevance of each feature in the tree's decision-making process. It indicates how much each feature contributes to splitting the data and making accurate predictions. The feature importance can be determined based on the structure and performance of the decision tree.

We can decide on feature importance by looking at decrease in gini impurity or entropy values for a particular feature.

It's important to note that feature importance in decision trees is relative to the specific tree and dataset it was trained on. The importance of features can vary between different trees or when different subsets of data are used. Therefore, it's recommended to assess feature importance across multiple trees.

70. What are ensemble techniques and how are they related to decision trees?

Ans: Ensemble techniques in machine learning involve combining multiple models or predictions to improve overall performance, robustness, and generalization. Rather than relying on a single model, ensemble methods aim to leverage the collective wisdom of multiple models to make more accurate and reliable predictions. Ensemble techniques can be applied to various types of machine learning algorithms.

Decision trees are commonly used as base models in many of the ensemble techniques.Ensemble techniques like gradient boosting,XGBoost,Random forest use decision tree as their base models.

**Ensemble Techniques:**

71. What are ensemble techniques in machine learning?

Ans: Ensemble techniques in machine learning involve combining multiple models or predictions to improve overall performance, robustness, and generalization. Rather than relying on a single model, ensemble methods aim to leverage the collective wisdom of multiple models to make more accurate and reliable predictions. Ensemble techniques can be applied to various types of machine learning algorithms.

Adaboost,voting classifier and regressor,Random forest,stacking,Gradient boosting are some examples of ensemble techniques.

72. What is bagging and how is it used in ensemble learning?

Ans: Bagging, short for Bootstrap Aggregating, is an ensemble technique that involves creating multiple instances of the same model trained on different subsets of the training data. Bagging is particularly effective when combined with decision tree models, but it can also be applied to other base models.

Here is an overview of how bagging ensemble works:

(a)Data Preparation: The training dataset is randomly divided into multiple subsets, typically of equal size. This sampling is performed with replacement, meaning that each subset can contain duplicate instances, and some instances may not be included.

(b)Base Model Training: For each subset of the training data, a base model (e.g., decision tree) is trained independently. Each model is exposed to a different subset of the data, resulting in slightly different models due to the variations in the training sets.

(c)Prediction Combination: Once all the base models are trained, predictions from each model are combined to produce the final ensemble prediction. For classification tasks, this can be done through majority voting, where the class with the highest number of votes is selected. For regression tasks, averaging the predictions is often used.

73. Explain the concept of bootstrapping in bagging.

Ans: Bootstrapping is a resampling technique used in ensemble learning to create multiple datasets by randomly sampling the original dataset with replacement. Each bootstrapped dataset is used to train a separate base model, and the predictions from these models are combined to form an ensemble prediction.

Bootstrapping is used in bagging in following way::

(a)Data Resampling: Given the original training dataset with N instances, bootstrapping involves randomly selecting N instances from the dataset with replacement. This means that some instances may be selected multiple times, while others may not be selected at all. Each bootstrapped dataset has the same size as the original dataset but may have different compositions.

(b)Base Model Training: Using each bootstrapped dataset, a separate base model (often the same model architecture) is trained independently. Each base model learns from a different bootstrapped dataset, which introduces diversity in the training process.

(c )Ensemble Prediction: Once all the base models are trained, their predictions are combined to produce the ensemble prediction. In bagging, this combination is typically done through averaging the predictions for regression tasks or majority voting for classification tasks.

74. What is boosting and how does it work?

Ans: Boosting is an ensemble learning technique that sequentially builds a series of base models, where each subsequent model focuses on correcting the mistakes made by the previous models. Boosting algorithms aim to create a strong learner by combining multiple weak learners.

Boosting works in following way:

(a)Base Model Training: Initially, a base model (e.g., decision tree, neural network) is trained on the training dataset. This base model is often a weak learner, which means it performs slightly better than random guessing but is not highly accurate on its own.

(b)Weighted Data Instances: Each instance in the training dataset is assigned an initial weight. Initially, all weights are usually set equally. However, as the boosting algorithm progresses, the weights are adjusted based on the performance of the previous models. Instances that are misclassified or have higher errors receive higher weights to receive more focus in subsequent iterations.

(c)Iterative Model Building: In subsequent iterations, new base models are trained on the updated training dataset, giving more importance to the misclassified instances. The process continues for a specified number of iterations or until a stopping criterion is met.

(d)Combining Predictions: The predictions of all the base models are combined to make the final ensemble prediction. The combination method varies based on the specific boosting algorithm. In binary classification, a common approach is to use weighted voting, where each base model's prediction is weighted based on its performance.

75. What is the difference between AdaBoost and Gradient Boosting?

Ans: Following are the differences between Adaboost and Gradient Boosting:

(a)Weight Updates:

AdaBoost: In AdaBoost, misclassified instances are assigned higher weights, which allows subsequent weak learners to focus more on those instances during training. Each subsequent weak learner is trained on a reweighted version of the dataset to prioritize the misclassified instances.

Gradient Boosting: In Gradient Boosting, the subsequent weak learners are trained to correct the residual errors made by the previous weak learners. The weights are not explicitly adjusted, and the focus is on minimizing the errors by updating the predictions based on the gradients of the loss function.

(b)Learning Approach:

AdaBoost: AdaBoost places more emphasis on difficult-to-classify instances by adapting the instance weights during training. It iteratively combines weak learners, with each learner providing an opinion in the final ensemble based on its performance.

Gradient Boosting: Gradient Boosting focuses on minimizing the errors of the ensemble by iteratively adding weak learners. Each weak learner is trained to improve the overall ensemble's performance by directly minimizing the loss function, typically through gradient descent.

(c) Loss Function Optimization:

AdaBoost: AdaBoost is primarily used for classification tasks and typically minimizes the exponential loss function. It aims to minimize the weighted sum of the exponential loss for misclassified instances.

Gradient Boosting: Gradient Boosting can be applied to both regression and classification tasks. It is flexible in terms of loss function selection and can optimize various loss functions, such as squared loss (regression) or logistic loss (classification), through gradient descent.

(d)Handling Outliers:

AdaBoost: AdaBoost is sensitive to outliers and noisy data. Since the weights of misclassified instances are increased in each iteration, outliers can have a significant influence on the ensemble's final predictions.

Gradient Boosting: Gradient Boosting is less sensitive to outliers and can handle them to some extent. The gradient-based optimization allows subsequent weak learners to focus on correcting the residual errors, which can help mitigate the impact of outliers.

76. What is the purpose of random forests in ensemble learning?

Ans: The purpose of Random Forest in ensemble learning is to combine the strengths of multiple decision trees to create a more robust and accurate predictive model. Random Forest is an ensemble method that utilizes bagging, a resampling technique, with decision trees as base models

Here are the key purposes and benefits of using Random Forest in ensemble learning:

(a)Reduction of Variance: Random Forest helps to reduce the variance associated with individual decision trees. By training multiple decision trees on different bootstrapped subsets of the data and averaging their predictions, Random Forest mitigates overfitting and provides more stable and reliable predictions.

(b)Handling of High-Dimensional Data: Random Forest can effectively handle datasets with a large number of features (high-dimensional data). Each decision tree in the Random Forest randomly selects a subset of features at each split, allowing it to capture different subsets of relevant features. This random feature selection helps to prevent individual decision trees from dominating based on a single influential feature.

(c)Robustness to Noisy Data: Random Forest is known for its robustness to noisy data and outliers. The averaging of predictions from multiple decision trees helps reduce the impact of individual noisy instances, as the majority consensus among the trees prevails.

(d)Feature Importance Estimation: Random Forest provides a measure of feature importance, indicating the relative significance of each feature in the prediction process. This information can be helpful in understanding the most influential features in the dataset.

77. How do random forests handle feature importance?

Ans: Random Forests handle feature importance by assessing the impact of each feature on the overall performance of the ensemble. The importance of a feature is determined based on how much it contributes to the reduction in impurity (e.g., Gini impurity) or the decrease in the chosen splitting criterion during the construction of the decision trees.

Here's a general overview of how Random Forests estimate feature importance:

(a)During Training: In each decision tree of the Random Forest, at each split point, a subset of features is randomly selected. The selected features are evaluated, and the one that optimally splits the data is chosen based on a specified criterion (e.g., Gini impurity, information gain). The improvement in the criterion resulting from the split is recorded.

(b)Aggregation of Importance: After training the ensemble of decision trees, the feature importance is calculated by aggregating the importance measures across all the trees. The aggregated importance is often normalized to ensure that the sum of feature importances adds up to 1 or scaled to a specific range.

(c)Evaluation Criterion: The specific metric used to measure feature importance can vary, but commonly used approaches include:  
(i)Gini importance  
(ii)Information gain

(iii)Mean decrease accuracy

78. What is stacking in ensemble learning and how does it work?

Ans: Stacking, also known as stacked generalization, is an ensemble learning technique that involves training multiple models (referred to as base models) and combining their predictions using a meta-model to make the final prediction. Stacking goes beyond simple averaging or voting by learning how to best combine the predictions of the base models through another model called the meta-model or aggregator.

Here's an overview of how stacking works:

(a)Base Model Training: Several different base models (e.g., decision trees, neural networks, support vector machines) are trained on the training dataset. Each base model learns to make predictions based on the input features.

(b)Prediction Generation: Once the base models are trained, they are used to generate predictions on a validation dataset that was not used during their training. Each base model generates predictions for the validation dataset.

(c)Meta-Model Training: The predictions from the base models (along with the original features) are used as inputs to train the meta-model. The meta-model learns to combine the predictions from the base models and generate the final prediction. This meta-model can be a simple model such as a linear regression, logistic regression, or even a more complex model like a neural network.

(d)Prediction Combination: Finally, when making predictions on new, unseen data, the trained base models are used to generate predictions, and these predictions are then fed into the trained meta-model to generate the ensemble's final prediction.

79. What are the advantages and disadvantages of ensemble techniques?

Ans: Ensemble techniques in machine learning offer several advantages, but they also come with certain disadvantages. Here's an overview of the advantages and disadvantages of ensemble techniques:

Advantages of Ensemble Techniques:

1. Improved Accuracy: Ensemble methods often achieve higher predictive accuracy compared to individual models. By combining predictions from multiple models, ensemble techniques can reduce bias, variance, and overfitting, leading to more robust and accurate predictions.

2. Enhanced Robustness: Ensembles are more resistant to outliers and noise in the data. Individual models may make errors or have limitations, but combining their predictions helps to mitigate these weaknesses, resulting in more reliable and robust performance.

3. Better Generalization: Ensemble techniques aim to generalize well to unseen data. By utilizing different learning algorithms or training subsets, ensembles can capture a wider range of patterns and relationships in the data, leading to improved generalization and better handling of complex datasets.

4. Feature Importance Estimation: Ensemble methods, such as Random Forests and Gradient Boosting, can provide insights into feature importance. They assess the contribution of each feature in the ensemble's predictions, helping to identify the most influential factors and aiding in feature selection or feature engineering.

5. Flexibility: Ensemble techniques are versatile and can be applied to various machine learning algorithms and tasks. They are not limited to specific model types, allowing for the integration of different models and approaches into a unified ensemble framework.

Disadvantages of Ensemble Techniques:

1. Increased Complexity: Ensembles introduce additional complexity in model creation, training, and prediction. Combining multiple models requires more computational resources, time, and effort compared to training and deploying a single model.

2. Overfitting Risk: Although ensembles can reduce overfitting, there is still a risk of overfitting, especially if the ensemble becomes too complex or if the base models are highly correlated. Careful model selection, regularization techniques, and proper validation are necessary to mitigate overfitting.

3. Interpretability: Ensemble models are generally less interpretable than individual models. The combination of multiple models may make it challenging to understand the underlying decision-making process or the specific contribution of each base model.

4. Training Time and Resource Requirements: Ensembles typically require more computational resources and time for training and prediction compared to individual models. The increased complexity and need for multiple models contribute to higher computational demands.

5. Model Dependency: Ensembles rely on the assumption that the base models are diverse and make independent errors. If the base models are too similar or suffer from similar biases or weaknesses, the ensemble may not provide significant improvements over individual models.

80. How do you choose the optimal number of models in an ensemble?

Ans:

Choosing the optimal number of models in an ensemble is a balancing act that involves considering factors such as performance, computational resources, and the risk of overfitting. There is no definitive rule for determining the exact number of models in an ensemble, but here are some approaches and considerations to guide the decision:

(a)Performance on Validation Set: Monitor the performance of the ensemble on a validation set as you increase the number of models. Initially, as you add more models, the performance may improve. However, there will be a point where adding more models does not significantly increase performance or may even start to decrease it. Identify this point and select the number of models where the performance plateaus.

(b)Performance on Test Set: Evaluate the ensemble's performance on an independent test set. This ensures that the ensemble's performance generalizes well to unseen data. Consider the stability of the ensemble's performance across different test sets to assess if adding more models provides consistent improvements.

(c )Computational Resources: Consider the available computational resources and the time required to train and make predictions with each model in the ensemble. As the number of models increases, the training and prediction time will also increase. Find a balance between performance and resource constraints.

(d)Overfitting Risk: Be cautious of the risk of overfitting when adding more models. Adding too many models can cause the ensemble to memorize the training data and perform poorly on new, unseen data. Use techniques like cross-validation or monitoring performance on validation sets to detect signs of overfitting.

(e)Ensemble Diversity: Assess the diversity of the base models in the ensemble. If the models are too similar, adding more models may not provide significant benefits. Ensure that the models in the ensemble are diverse in terms of algorithms, hyperparameters, and training data to maximize the ensemble's performance.

(f)Occam's Razor Principle: Consider the principle of Occam's Razor, which suggests selecting the simplest solution that achieves satisfactory performance. Adding more models increases complexity and may not lead to substantial improvements beyond a certain point.