Data Science:

1. What is Data Science?

Data Science is a field of computer science that explicitly deals with turning data into information and extracting meaningful insights from it. The reason why Data Science is so popular is that the kind of insights it allows us to draw from the available data has led to some major innovations in several products and companies. Using these insights, we are able to determine the taste of a particular customer, the likelihood of a product succeeding in a particular market, etc.

Check out this comprehensive Data Science Course!

2. Differentiate between Data Analytics and Data Science

Data Analytics	Data Science
Data Analytics is a subset of Data Science.	Data Science is a broad technology that includes various subsets such as Data Analytics, Data Mining, Data Visualization, etc.
The goal of data analytics is to illustrate the precise details of retrieved insights.	The goal of data science is to discover meaningful insights from massive datasets and derive the best possible solutions to resolve business issues.
Requires just basic programming languages.	Requires knowledge in advanced programming languages, statistics, and special machine learning algorithms.
It focuses on just finding the solutions.	Data Science not only focuses on finding solutions but also predicts the future with past patterns or insights.
A data analyst's job is to analyze data in order to make decisions.	A data scientist's job is to provide insightful data visualizations from raw

	data that are easily understandable.

Become an expert in Data Scientist. Enroll now in PG program in Data Science and Machine Learning from MITxMicroMasters

3. How is Python Useful?

Python is widely recognized as an exceptionally advantageous programming language due to its versatility and simplicity. Its extensive range of applications and associated benefits have established it as a preferred choice among developers. Notably, Python stands out in terms of readability and user-friendliness.

Its syntax is meticulously designed to be intuitive and concise, enabling ease in coding, comprehension, and maintenance. Additionally, Python offers a comprehensive standard library that encompasses a diverse collection of pre-built modules and functions. This wealth of resources substantially minimizes the time and effort expended by developers, streamlining the execution of routine programming tasks.

4. How R is Useful in the Data Science Domain?

Here are some ways in which R is useful in the data science domain:

- **Data Manipulation and Analysis:** R offers a comprehensive collection of libraries and functions that facilitate proficient data manipulation, transformation, and statistical analysis.
- **Statistical Modeling and Machine Learning:** R offers a wide range of packages for advanced statistical modeling and machine learning tasks, empowering data scientists to build predictive models and perform complex analyses.
- **Data Visualization:** R's extensive visualization libraries enable the creation of visually appealing and insightful plots, charts, and graphs.

- **Reproducible Research:** R supports the integration of code, data, and documentation, facilitating reproducible workflows and ensuring transparency in data science projects.

5. What is Supervised Learning?

Supervised learning is a machine learning approach in which an algorithm learns from labeled training data to make predictions or classify new, unseen data. It involves the use of input data and corresponding output labels, allowing the algorithm to learn patterns and relationships. The goal is to generalize the learned patterns and accurately predict outputs for new input data based on the learned patterns.

6. What is Unsupervised Learning?

Unsupervised learning is a machine learning approach wherein an algorithm uncovers patterns and structures within unlabeled data, operating without explicit guidance or predetermined output labels. Its objective is to reveal hidden relationships, patterns, and clusters present in the data. Unlike supervised learning, the algorithm autonomously explores the data to identify inherent structures and draw inferences, proving valuable for exploratory data analysis and the discovery of novel insights.

7. What do you understand about Linear Regression?

Linear regression helps in understanding the linear relationship between the dependent and the independent variables. Linear regression is a supervised learning algorithm, which helps in finding the linear relationship between two variables. One is the predictor or the independent variable and the other is the response or the dependent variable. In linear regression, we try to understand how the dependent variable changes with respect to the independent variable. If there is only one independent variable, then it is called simple linear regression, and if there is more than one independent variable then it is known as multiple linear regression.

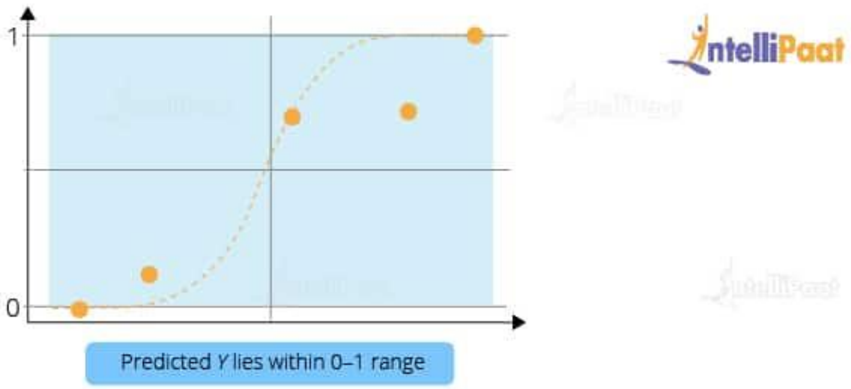
Interested in learning Data Science? Click here to learn more in this Data Science Course!

8. What do you understand by logistic regression?

Logistic regression is a classification algorithm that can be used when the dependent variable is binary. Let's take an example. Here, we are trying to determine whether it will rain or not on the basis of temperature and humidity.

Temperature
+
Humidity
=
Rain
IntelliPaat

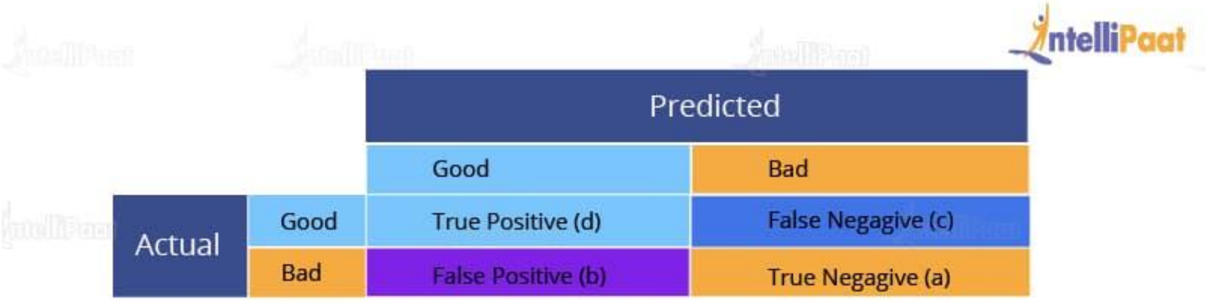
Temperature and humidity are the independent variables, and rain would be our dependent variable. So, the logistic regression algorithm actually produces an S shape curve.



So, basically in logistic regression, the Y value lies within the range of 0 and 1. This is how logistic regression works.

9. What is a confusion matrix?

The confusion matrix is a table that is used to estimate the performance of a model. It tabulates the actual values and the predicted values in a 2×2 matrix.



		Predicted	
		Good	Bad
Actual	Good	True Positive (d)	False Negagive (c)
	Bad	False Positive (b)	True Negagive (a)

True Positive (d): This denotes all of those records where the actual values are true and the predicted values are also true. So, these denote all of the true positives. False Negative (c): This denotes all of those records where the actual values are true, but the predicted values are false. False Positive (b): In this, the actual values are false, but the predicted values are true. True Negative (a): Here, the actual values are false and the predicted values are also false. So, if you want to get the correct values, then correct values would basically represent all of the true positives and the true negatives. This is how the confusion matrix works.

10. What do you understand about the true-positive rate and false-positive rate?

True positive rate: In Machine Learning, true-positive rates, which are also referred to as sensitivity or recall, are used to measure the percentage of actual positives which are correctly identified. Formula:

```
True Positive Rate = True Positives/Positives
```

False positive rate: False positive rate is basically the probability of falsely rejecting the null hypothesis for a particular test. The false-positive rate is calculated as the ratio between the number of negative events wrongly categorized as positive (false positive) upon the total number of actual events. Formula:

```
False-Positive Rate = False-Positives/Negatives.
```

Check out this comprehensive Data Science Course in India!

11. How is Data Science different from traditional application programming?

Data Science takes a fundamentally different approach to building systems that provide value than traditional application development.

In traditional programming paradigms, we used to analyze the input, figure out the expected output, and write code, which contains rules and statements needed to transform the provided input into the expected output. As we can imagine, these rules were not easy to write, especially, for data that even computers had a hard time understanding, e.g., images, videos, etc.

Data Science shifts this process a little bit. In it, we need access to large volumes of data that contain the necessary inputs and their mappings to the expected outputs. Then, we use data science algorithms, which use mathematical analysis to generate rules to map the given inputs to outputs.

This process of rule generation is called training. After training, we use some data that was set aside before the training phase to test and check the system's accuracy. The generated rules are a kind of black box, and we cannot understand how the inputs are being transformed into outputs.

However, If the accuracy is good enough, then we can use the system (also called a model).

As described above, in traditional programming, we had to write the rules to map the input to the output, but in Data Science, the rules are automatically generated or learned from the given data. This helped solve some really difficult challenges that were being faced by several companies.

Interested to learn Data Science skills? Check our Data Science course in Kottayam Now!

12. Explain the differences between supervised and unsupervised learning.

Supervised and unsupervised learning are two types of Machine Learning techniques. They both allow us to build models. However, they are used for solving different kinds of problems.

Supervised Learning	Unsupervised Learning
Works on the data that contains both inputs and the expected output, i.e., the labeled data	Works on the data that contains no mappings from input to output, i.e., the unlabeled data
Used to create models that can be employed to predict or classify things	Used to extract meaningful information out of large volumes of data
Commonly used supervised learning algorithms: Linear regression, decision tree, etc.	Commonly used unsupervised learning algorithms: K-means clustering, Apriori algorithm, etc.

13. What is the difference between long format data and wide format data?

Long Format Data	Wide Format Data
A long format data has a column for possible variable types and a column for the values of those variables.	Whereas, Wide data has a column for each variable.
Each row in the long format represents a one-time point per subject. As a result, each topic will contain many rows of data.	The repeated responses of a subject will be in a single row, with each response in its own column, in the wide format.
This data format is most typically used in R analysis and for writing to log files at the end of each experiment.	This data format is most widely used in data manipulations, and stats programmes for repeated measures ANOVAs and is seldom used in R analysis.
A long format contains values that do repeat in the first column.	A wide format contains values that do not repeat in the first column.
Use df.melt() to convert the wide form to long form	use df.pivot().reset_index() to convert the long form into wide form

14. Mention some techniques used for sampling. What is the main advantage of sampling?

Sampling is defined as the process of selecting a sample from a group of people or from any particular kind for research purposes. It is one of the most important factors which decides the accuracy of a research/survey result.

Mainly, there are two types of sampling techniques:

Probability sampling: It involves random selection which makes every element get a chance to be selected. Probability sampling has various subtypes in it, as mentioned below:

- Simple Random Sampling
- Stratified sampling
- Systematic sampling
- Cluster Sampling
- Multi-stage Sampling

Non- Probability Sampling: Non-probability sampling follows non-random selection which means the selection is done based on your ease or any other required criteria. This helps to collect the data easily. The following are various types of sampling in it:

- Convenience Sampling
- Purposive Sampling
- Quota Sampling
- Referral /Snowball Sampling

15. What is bias in data science?

Bias is a type of error that occurs in a data science model because of using an algorithm that is not strong enough to capture the underlying patterns or trends that exist in the data. In other words, this error occurs when the data is too complicated for the algorithm to understand, so it ends up building a model that makes simple assumptions. This leads to lower accuracy because of underfitting. Algorithms that can lead to high bias are linear regression, logistic regression, etc.

16. What is dimensionality reduction?

Dimensionality reduction is the process of converting a dataset with a high number of dimensions (fields) to a dataset with a lower number of dimensions. This is done by dropping some fields or columns from the dataset. However, this is not done haphazardly. In this process, the dimensions or fields are dropped only after making sure that the remaining information will still be enough to succinctly describe similar information.

17. Why is Python used for data cleaning in DS?

Data Scientists have to clean and transform huge data sets into a form that they can work with. It is important to deal with redundant data for better results by removing nonsensical outliers, malformed records, missing values, inconsistent formatting, etc.

Python libraries such as Matplotlib, Pandas, Numpy, Keras, and SciPy are extensively used for data cleaning and analysis. These libraries are used to load and clean the data and do effective analysis. For instance, you might decide to remove outliers that are beyond a certain standard deviation from the mean of a numerical column.

```
mean = df['Price'].mean()

std = df['Price'].std()

threshold = mean + (3 * std)  # Set a threshold for outliers

df = df[df['Price'] < threshold]  # Remove outliers
```

Hence, this is how the process of data cleaning is done using python libraries in the field of data science.

Learn more about Data Cleaning in a Data Science Tutorial!

18. Why is R used in Data Visualization?

R provides the best ecosystem for data analysis and visualization with more than 12,000 packages in Open-source repositories. It has huge community support, which means you can easily find the solution to your problems on various platforms like StackOverflow.

It has better data management and supports distributed computing by splitting the operations between multiple tasks and nodes, which eventually decreases the complexity and execution time of large datasets.

19. What are the popular libraries used in Data Science?

Below are the popular libraries used for data extraction, cleaning, visualization, and deploying DS models:

- TensorFlow: Supports parallel computing with impeccable library management backed by Google.
- SciPy: Mainly used for solving differential equations, multidimensional programming, data manipulation, and visualization through graphs and charts.
- Pandas: Used to implement the ETL(Extracting, Transforming, and Loading the datasets) capabilities in business applications.
- Matplotlib: Being free and open-source, it can be used as a replacement for MATLAB, which results in better performance and low memory consumption.
- PyTorch: Best for projects which involve ,machine learning algorithms and deep neural networks.

20. What are important functions used in Data Science?

Within the realm of data science, various pivotal functions assume critical roles across diverse tasks. Among these, two foundational functions are the cost function and the loss function.

Cost function: Also referred to as the objective function, the cost function holds substantial utility within machine learning algorithms, especially in optimization scenarios. Its purpose is to quantify the disparity between predicted values and actual values. Minimizing the cost function entails optimizing the model's parameters or coefficients, aiming to achieve an optimal solution.

Loss function: Loss functions possess significant significance in supervised learning endeavors. They evaluate the discrepancy or error between predicted values and actual labels. The selection of a specific loss function depends on the problem at hand, such as employing mean squared error (MSE) for regression tasks or cross-entropy loss for classification tasks. The loss function guides the model's optimization process during training, ultimately bolstering accuracy and overall performance.

21. What is k-fold cross-validation?

In k-fold cross-validation, we divide the dataset into k equal parts. After this, we loop over the entire dataset k times. In each iteration of the loop, one of the k parts is used for testing, and the other k − 1 parts are used for training. Using k-fold cross-validation, each one of the k parts of the dataset ends up being used for training and testing purposes.

22. Explain how a recommender system works.

A recommender system is a system that many consumer-facing, content-driven, online platforms employ to generate recommendations for users from a library of available content. These systems generate recommendations based on what they know about the users' tastes from their activities on the platform.

For example, imagine that we have a movie streaming platform, similar to Netflix or Amazon Prime. If a user has previously watched and liked movies from action and horror genres, then it means that the user likes watching movies of these genres. In that case, it would be better to recommend such movies to this particular user. These recommendations can also be generated based on what users with similar tastes like watching.

23. What is Poisson Distribution?

The Poisson distribution is a statistical probability distribution used to represent the occurrence of events within a specific interval of time or space. It is commonly employed to characterize infrequent events that happen independently and at a consistent average rate, such as quantifying the number of incoming phone calls received within a given hour.

24. What is a normal distribution?

Data distribution is a visualization tool to analyze how data is spread out or distributed. Data can be distributed in various ways. For instance, it could be with a bias to the left or the right, or it could all be jumbled up.

Data may also be distributed around a central value, i.e., mean, median, etc. This kind of distribution has no bias either to the left or to the right and is in the form of a bell-shaped curve. This distribution also has its mean equal to the median. This kind of distribution is called a normal distribution.

25. What is Deep Learning?

Deep Learning is a kind of Machine Learning, in which neural networks are used to imitate the structure of the human brain, and just like how a brain learns from information, machines are also made to learn from the information that is provided to them.

Deep Learning is an advanced version of neural networks to make the machines learn from data. In Deep Learning, the neural networks comprise many hidden layers (which is why it is called ‘deep’ learning) that are connected to each other, and the output of the previous layer is the input of the current layer.

26. What is CNN (Convolutional Neural Network)?

A Convolutional Neural Network (CNN) is an advanced deep learning architecture designed specifically for analyzing visual data, such as images and videos. It is composed of interconnected layers of neurons that utilize convolutional operations to extract meaningful features from the input data. CNNs exhibit remarkable effectiveness in tasks like image classification, object detection, and image recognition, thanks to their inherent ability to autonomously learn hierarchical representations and capture spatial relationships within the data, eliminating the need for explicit feature engineering.

27. What is an RNN (recurrent neural network)?

A recurrent neural network, or RNN for short, is a kind of Machine Learning algorithm that makes use of the artificial neural network. RNNs are used to find patterns from a sequence of data, such as time series, stock market, temperature, etc. RNNs are a kind of feedforward network, in which information from one layer passes to another layer, and each node in the network performs mathematical operations on the data. These operations are temporal, i.e., RNNs store contextual information about previous computations in the network. It is called recurrent because it performs the same operations on some data every time it is passed. However, the output may be different based on past computations and their results.

28. Explain selection bias.

Selection bias is the bias that occurs during the sampling of data. This kind of bias occurs when a sample is not representative of the population, which is going to be analyzed in a statistical study.

29. Between Python and R, which one will you choose for analyzing the text, and why?

Due to the following factors, Python will outperform R for text analytics:

- Python's Pandas module provides high-performance data analysis capabilities as well as simple-to-use data structures.
- Python does all sorts of text analytics more quickly.

30. Explain the purpose of data cleaning

Data cleaning's primary goal is to rectify or eliminate inaccurate, corrupted, improperly formatted, duplicate, or incomplete data from a dataset. This often yields better outcomes and a higher return on investment for marketing and communications efforts.

31. What do you understand from Recommender System? and State its application

Recommender Systems are a subclass of information filtering systems designed to forecast the preferences or ratings given to a product by a user.

The Amazon product suggestions page is an example of a recommender system in use. Based on the user's search history and previous orders, this area contains products.

32. What is Gradient Descent?

An iterative first-order optimization process called gradient descent (GD) is used to locate the local minimum and maximum of a given function. This technique is frequently applied in machine learning (ML) and deep learning (DL) to minimize a cost/loss function (for example, in linear regression).

33. What are the various skills required to become Data Scientist?

The following abilities are necessary to become a certified Data Scientist:

- Having familiarity with built-in data types like lists, tuples, sets, and related.
- N-dimensional NumPy array knowledge is required.
- Being able to use Pandas and Dataframes.
- Strong holdover performance in vectors with only one element.
- Hands-on experience with Tableau and PowerBI.

34. What is TensorFlow?

A free and open-source software library for machine learning and artificial intelligence is called TensorFlow. It enables programmers to build dataflow graphs, which are representations of the flow of data among processing nodes in a graph.

35. What is Dropout?

In Data Science, the term “dropout” refers to the process of randomly removing visible and hidden network units. By eliminating up to 20% of the nodes, they avoid overfitting the data and allow for the necessary space to be set up for the network’s iterative convergence process.

36. State any five Deep Learning Frameworks.

Some of the Deep Learning frameworks are:

- Caffe
- Keras
- TensorFlow
- Pytorch
- Chainer
- Microsoft Cognitive Toolkit

37. Define Neural Networks and its types

Neural Networks are computational models that derive their principles from the structure and functionality of the human brain. Consisting of interconnected artificial neurons organized in layers, Neural Networks exhibit remarkable capacities in learning and discerning patterns within datasets. Consequently, they assume a pivotal role in diverse domains including pattern recognition, classification, and optimization, thereby providing invaluable solutions in the realm of artificial intelligence.

There exist various types of Neural Networks, including:

- **Feedforward Neural Networks:** These networks facilitate a unidirectional information flow, progressing from input to output. They find frequent application in tasks involving pattern recognition and classification.
- **Convolutional Neural Networks (CNNs):** Specifically tailored for grid-like data, such as images or videos, CNNs leverage convolutional layers to extract meaningful features. Their prowess lies in tasks like image classification and object detection.
- **Recurrent Neural Networks (RNNs):** RNNs are particularly adept at handling sequential data, wherein the present output is influenced by past inputs. They are extensively utilized in domains such as language modeling and time series analysis.
- **Long Short-Term Memory (LSTM) Networks:** This variation of RNNs addresses the issue of vanishing gradients and excels at capturing long-term

dependencies in data. LSTM networks find wide-ranging applications in areas like speech recognition and natural language processing.

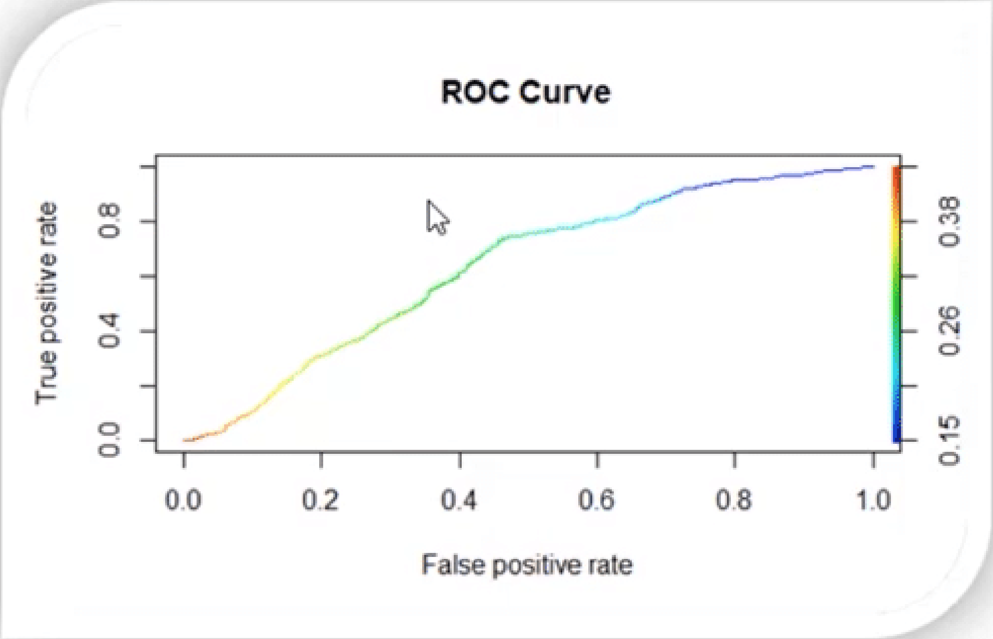
- **Generative Adversarial Networks (GANs):** GANs consist of a generator and a discriminator that is trained in a competitive manner. They are employed to generate new data samples and are helpful for tasks like image generation and text synthesis.

These examples represent only a fraction of the available variations and architectures tailored to specific data types and problem domains.

Intermediate Data Science Interview Questions

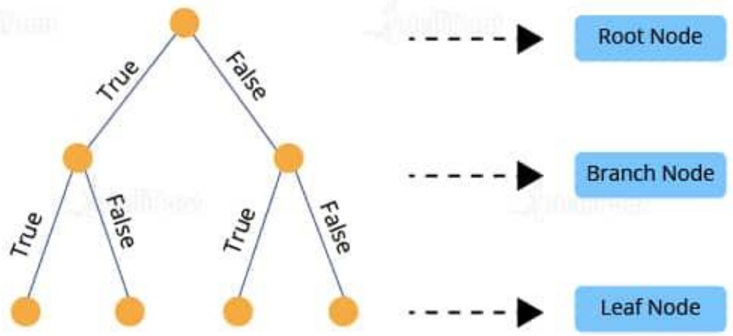
38. What is the ROC curve?

It stands for **Receiver Operating Characteristic**. It is basically a plot between a true positive rate and a false positive rate, and it helps us to find out the right tradeoff between the true positive rate and the false positive rate for different probability thresholds of the predicted values. So, the closer the curve to the upper left corner, the better the model is. In other words, whichever curve has greater area under it that would be the better model. You can see this in the below graph:



39. What do you understand by a decision tree?

A decision tree is a supervised learning algorithm that is used for both classification and regression. Hence, in this case, the dependent variable can be both a numerical value and a categorical value.

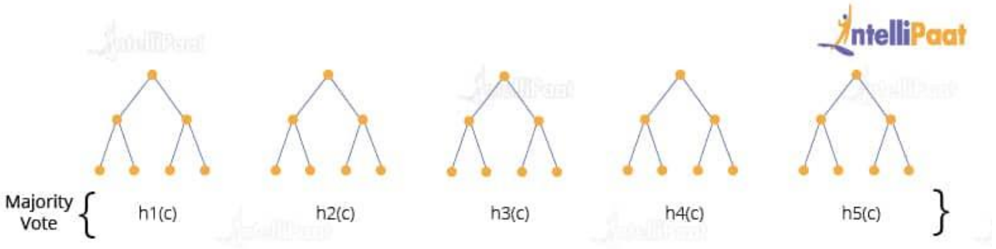


Here, each node denotes the test on an attribute, and each edge denotes the outcome of that attribute, and each leaf node holds the class label. So, in this case, we have a series of test conditions which give the final decision according to the condition.

Are you interested in learning Data Science from experts? Enroll in our Data Science Course in Bangalore now!

40. What do you understand by a random forest model?

It combines multiple models together to get the final output or, to be more precise, it combines multiple decision trees together to get the final output. So, decision trees are the building blocks of the random forest model.



41. Two candidates, Aman and Mohan appear for a Data Science Job interview. The probability of Aman cracking the interview is 1/8 and that of Mohan is 5/12. What is the probability that at least one of them will crack the interview?

The probability of Aman getting selected for the interview is 1/8

$P(A) = 1/8$

The probability of Mohan getting selected for the interview is 5/12

$P(B)=5/12$

Now, the probability of at least one of them getting selected can be denoted at the Union of A and B, which means

$P(A \cup B) = P(A) + P(B) - (P(A \cap B))$ ………………………(1)

Where P(A ∩ B) stands for the probability of both Aman and Mohan getting selected for the job.

To calculate the final answer, we first have to find out the value of P(A ∩ B)

So, P(A ∩ B) = P(A) * P(B)

1/8 * 5/12

5/96

Now, put the value of P(A ∩ B) into equation (1)

P(A U B) =P(A)+ P(B) – (P(A ∩ B))

1/8 + 5/12 -5/96

So, the answer will be 47/96.

42. How is Data modeling different from Database design?

Data Modeling: It can be considered as the first step towards the design of a database. Data modeling creates a conceptual model based on the relationship between various data models. The process involves moving from the conceptual stage to the logical model to the physical schema. It involves the systematic method of applying data modeling techniques.

Database Design: This is the process of designing the database. The database design creates an output which is a detailed data model of the database. Strictly speaking, database design includes the detailed logical model of a database but it can also include physical design choices and storage parameters.

43. What is precision?

Precision: When we are implementing algorithms for the classification of data or the retrieval of information, precision helps us get a portion of positive class values that are positively predicted. Basically, it measures the accuracy of correct positive predictions. Below is the formula to calculate precision:

$$precision = \frac{true\ positives}{true\ positives + false\ positives}$$

44. What is a recall?

Recall: It is the set of all positive predictions out of the total number of positive instances. Recall helps us identify the misclassified positive predictions. We use the below formula to calculate recall:

$$recall = \frac{true\ positives}{true\ positives\ \ + false\ negatives}$$

45. What is the F1 score and how to calculate it?

F1 score helps us calculate the harmonic mean of precision and recall that gives us the test's accuracy. If F1 = 1, then precision and recall are accurate. If F1 < 1 or equal to 0, then precision or recall is less accurate, or they are completely inaccurate. See below for the formula to calculate the F1 score:

$$\text{F1-score} \triangleq 2\frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

46. What is a p-value?

P-value is the measure of the statistical importance of an observation. It is the probability that shows the significance of output to the data. We compute the p-value to know the test statistics of a model. Typically, it helps us choose whether we can accept or reject the null hypothesis.

47. Why do we use p-value?

We use the p-value to understand whether the given data really describes the observed effect or not. We use the below formula to calculate the p-value for the effect ‘E’ and the null hypothesis ‘H0’ is true:

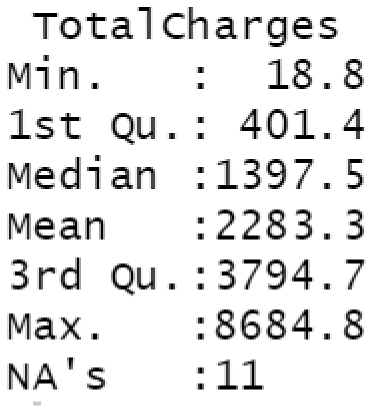
$$P\ Value = P(E \mid H0)$$

48. What is the difference between an error and a residual error?

An error occurs in values while the prediction gives us the difference between the observed values and the true values of a dataset. Whereas, the residual error is the difference between the observed values and the predicted values. The reason we use the residual error to evaluate the performance of an algorithm is that the true values are never known. Hence, we use the observed values to measure the error using residuals. It helps us get an accurate estimate of the error.

49. Why do we use the summary function?

The summary function in R gives us the statistics of the implemented algorithm on a particular dataset. It consists of various objects, variables, data attributes, etc. It provides summary statistics for individual objects when fed into the function. We use a summary function when we want information about the values present in the dataset. It gives us the summary statistics in the following form:



```
  TotalCharges
 Min.   :  18.8
 1st Qu.: 401.4
 Median :1397.5
 Mean   :2283.3
 3rd Qu.:3794.7
 Max.   :8684.8
 NA's   :11
```

Here, it gives the minimum and maximum values from a specific column of the dataset. Also, it provides the median, mean, 1st quartile, and 3rd quartile values that help us understand the values better.

50. How are Data Science and Machine Learning related to each other?

Data Science and Machine Learning are two terms that are closely related but are often misunderstood. Both of them deal with data. However, there are some fundamental distinctions that show us how they are different from each other.

Data Science is a broad field that deals with large volumes of data and allows us to draw insights from this voluminous data. The entire process of data science takes care of multiple steps that are involved in drawing insights out of the available data. This process includes crucial steps such as data gathering, data analysis, data manipulation, data visualization, etc.

Machine Learning, on the other hand, can be thought of as a sub-field of data science. It also deals with data, but here, we are solely focused on learning how to convert the processed data into a functional model, which can be used to map inputs to outputs, e.g., a model that can expect an image as an input and tell us if that image contains a flower as an output.

In short, data science deals with gathering data, processing it, and finally, drawing insights from it. The field of data science that deals with building models using algorithms is called machine learning. Therefore, machine learning is an integral part of data science.

51. Explain univariate, bivariate, and multivariate analyses.

When we are dealing with data analysis, we often come across terms such as univariate, bivariate, and multivariate. Let’s try and understand what these mean.

- Univariate analysis: Univariate analysis involves analyzing data with only one variable or, in other words, a single column or a vector of the data. This analysis allows us to understand the data and extract patterns and trends from it. Example: Analyzing the weight of a group of people.
- Bivariate analysis: Bivariate analysis involves analyzing the data with exactly two variables or, in other words, the data can be put into a two-column table. This kind of analysis allows us to figure out the relationship between the variables. Example: Analyzing the data that contains temperature and altitude.
- Multivariate analysis: Multivariate analysis involves analyzing the data with more than two variables. The number of columns of the data can be anything more than two. This kind of analysis allows us to figure out the effects of all other variables (input variables) on a single variable (the output variable).

Example: Analyzing data about house prices, which contains information about the houses, such as locality, crime rate, area, the number of floors, etc.

52. How can we handle missing data?

To be able to handle missing data, we first need to know the percentage of data missing in a particular column so that we can choose an appropriate strategy to handle the situation.

For example, if in a column the majority of the data is missing, then dropping the column is the best option, unless we have some means to make educated guesses about the missing values. However, if the amount of missing data is low, then we have several strategies to fill them up.

One way would be to fill them all up with a default value or a value that has the highest frequency in that column, such as 0 or 1, etc. This may be useful if the majority of the data in that column contains these values.

Another way is to fill up the missing values in the column with the mean of all the values in that column. This technique is usually preferred as the missing values have a higher chance of being closer to the mean than to the mode.

Finally, if we have a huge dataset and a few rows have values missing in some columns, then the easiest and fastest way is to drop those columns. Since the dataset is large, dropping a few columns should not be a problem anyway.

53. What is the benefit of dimensionality reduction?

Dimensionality reduction reduces the dimensions and size of the entire dataset. It drops unnecessary features while retaining the overall information in the data intact. Reduction in dimensions leads to faster processing of the data.

The reason why data with high dimensions is considered so difficult to deal with is that it leads to high time consumption while processing the data and training a model on it. Reducing dimensions speeds up this process, removes noise, and also leads to better model accuracy.

54. What is a bias-variance trade-off in Data Science?

When building a model using Data Science or Machine Learning, our goal is to build one that has low bias and variance. We know that bias and variance are both errors that occur due to either an overly simplistic model or an overly complicated model. Therefore, when we are building a model, the goal of getting high accuracy is only going to be accomplished if we are aware of the tradeoff between bias and variance.

Bias is an error that occurs when a model is too simple to capture the patterns in a dataset. To reduce bias, we need to make our model more complex. Although making the model more complex can lead to reducing bias, if we make the model too complex, it may end up becoming too rigid, leading to high variance. So, the tradeoff between bias and variance is that if we increase the complexity, the bias reduces and the variance increases, and if we reduce complexity, the bias increases and the variance reduces. Our goal is to find a point at which our model is complex enough to give low bias but not so complex to end up having high variance.

55. What is RMSE?

RMSE stands for the root mean square error. It is a measure of accuracy in regression. RMSE allows us to calculate the magnitude of error produced by a regression model. The way RMSE is calculated is as follows:

First, we calculate the errors in the predictions made by the regression model. For this, we calculate the differences between the actual and the predicted values. Then, we square the errors.

After this step, we calculate the mean of the squared errors, and finally, we take the square root of the mean of these squared errors. This number is the RMSE and a model with a lower value of RMSE is considered to produce lower errors, i.e., the model will be more accurate.

56. What is a kernel function in SVM?

In the SVM algorithm, a kernel function is a special mathematical function. In simple terms, a kernel function takes data as input and converts it into a required form. This transformation of the data is based on something called a kernel trick, which is what gives the kernel function its name. Using the kernel function, we can transform the data that is not linearly separable (cannot be separated using a straight line) into one that is linearly separable.

57. How can we select an appropriate value of k in k-means?

Selecting the correct value of k is an important aspect of k-means clustering. We can make use of the elbow method to pick the appropriate k value. To do this, we run the k-means algorithm on a range of values, e.g., 1 to 15. For each value of k, we compute an average score. This score is also called inertia or the inter-cluster variance.

This is calculated as the sum of squares of the distances of all values in a cluster. As k starts from a low value and goes up to a high value, we start seeing a sharp decrease in the inertia value. After a certain value of k, in the range, the drop in the inertia value becomes quite small. This is the value of k that we need to choose for the k-means clustering algorithm.

58. How can we deal with outliers?

Outliers can be dealt with in several ways. One way is to drop them. We can only drop the outliers if they have values that are incorrect or extreme. For example, if a dataset with the weights of babies has a value 98.6-degree Fahrenheit, then it is incorrect. Now, if the value is 187 kg, then it is an extreme value, which is not useful for our model.

In case the outliers are not that extreme, then we can try:

- A different kind of model. For example, if we were using a linear model, then we can choose a non-linear model
- Normalizing the data, which will shift the extreme values closer to other data points
- Using algorithms that are not so affected by outliers, such as random forest, etc.

59. How to calculate the accuracy of a binary classification algorithm using its confusion matrix?

In a binary classification algorithm, we have only two labels, which are True and False. Before we can calculate the accuracy, we need to understand a few key terms:

- True positives: Number of observations correctly classified as True
- True negatives: Number of observations correctly classified as False
- False positives: Number of observations incorrectly classified as True
- False negatives: Number of observations incorrectly classified as False

To calculate the accuracy, we need to divide the sum of the correctly classified observations by the number of total observations.

60. What is ensemble learning?

When we are building models using Data Science and Machine Learning, our goal is to get a model that can understand the underlying trends in the training data and can make predictions or classifications with a high level of accuracy.

However, sometimes some datasets are very complex, and it is difficult for one model to be able to grasp the underlying trends in these datasets. In such situations, we combine several individual models together to improve performance. This is what is called ensemble learning.

61. Explain collaborative filtering in recommender systems.

Collaborative filtering is a technique used to build recommender systems. In this technique, to generate recommendations, we make use of data about the likes and dislikes of users similar to other users. This similarity is estimated based on several varying factors, such as age, gender, locality, etc.

If User A, similar to User B, watched and liked a movie, then that movie will be recommended to User B, and similarly, if User B watched and liked a movie, then that would be recommended to User A.

In other words, the content of the movie does not matter much. When recommending it to a user what matters is if other users similar to that particular user liked the content of the movie or not.

62. Explain content-based filtering in recommender systems.

Content-based filtering is one of the techniques used to build recommender systems. In this technique, recommendations are generated by making use of the properties of the content that a user is interested in.

For example, if a user is watching movies belonging to the action and mystery genre and giving them good ratings, it is a clear indication that the user likes movies of this kind. If shown movies of a similar genre as recommendations, there is a higher probability that the user would like those recommendations as well.

In other words, here, the content of the movie is taken into consideration when generating recommendations for users.

63. Explain bagging in Data Science.

Bagging is an ensemble learning method. It stands for bootstrap aggregating. In this technique, we generate some data using the bootstrap method, in which we use an already existing dataset and generate multiple samples of the N size. This bootstrapped data is then used to train multiple models in parallel, which makes the bagging model more robust than a simple model.

Once all the models are trained, then it’s time to make a prediction, we make predictions using all the trained models and then average the result in the case of regression, and for classification, we choose the result, generated by models, that have the highest frequency.

64. Explain boosting in data science.

Boosting is one of the ensemble learning methods. Unlike bagging, it is not a technique used to parallelly train our models. In boosting, we create multiple models and sequentially train them by combining weak models iteratively in a way that training a new model depends on the models trained before it.

In doing so, we take the patterns learned by a previous model and test them on a dataset when training the new model. In each iteration, we give more importance to observations in the dataset that are incorrectly handled or predicted by previous models. Boosting is useful in reducing bias in models as well.

65. Explain stacking in data science.

Just like bagging and boosting, stacking is also an ensemble learning method. In bagging and boosting, we could only combine weak models that used the same learning algorithms, e.g., logistic regression. These models are called homogeneous learners.

However, in stacking, we can combine weak models that use different learning algorithms as well. These learners are called heterogeneous learners. Stacking works by training multiple (and different) weak models or learners and then using them together by training another model, called a meta-model, to make predictions based on the multiple outputs of predictions returned by these multiple weak models.

66. Explain how machine learning is different from deep learning.

A field of computer science, machine learning is a subfield of data science that deals with using existing data to help systems automatically learn new skills to perform different tasks without having rules to be explicitly programmed.

Deep Learning, on the other hand, is a field in machine learning that deals with building machine learning models using algorithms that try to imitate the process of how the human brain learns from the information in a system for it to attain new capabilities. In deep learning, we make heavy use of deeply connected neural networks with many layers.

67. What does the word 'Naive' mean in Naive Bayes?

Naive Bayes is a data science algorithm. It has the word 'Bayes' in it because it is based on the Bayes theorem, which deals with the probability of an event occurring given that another event has already occurred.

It has 'naive' in it because it makes the assumption that each variable in the dataset is independent of the other. This kind of assumption is unrealistic for real-world data. However, even with this assumption, it is very useful for solving a range of complicated problems, e.g., spam email classification, etc.

To learn more about Data Science, check out our Data Science Course in Hyderabad.

68. What is batch normalization?

One method for attempting to enhance the functionality and stability of the neural network is batch normalization. To do this, normalize the inputs in each layer such that the mean output activation stays at 0 and the standard deviation is set to 1.

69. What do you understand from cluster sampling and systematic sampling?

Cluster sampling is also known as the probability sampling approach where you can divide a population into groups, such as districts or schools, and then select a representative sample from among these groups at random. A modest representation of the population as a whole should be present in each cluster.

A probability sampling strategy called systematic sampling involves picking people from the population at regular intervals, such as every 15th person on a population list. The population can be organized randomly to mimic the benefits of simple random sampling.

70. What is the Computational Graph?

A directed graph with variables or operations as nodes is a computational graph. Variables can contribute to operations with their value, and operations can contribute their output to other operations. In this manner, each node in the graph establishes a function of the variables.

71. What is the difference between Batch and Stochastic Gradient Descent?

The differences between Batch and Stochastic Gradient Descent are as follows:

Batch	Stochastic Gradient Descent
Provides assistance in calculating the gradient utilizing the entire set of data.	Helps in calculating the gradient using only a single sample.
Takes time to converge.	Takes less time to converge.
The volume is substantial enough for analysis.	The volume is lower for analysis purposes.
Updates the weight infrequently.	Updates the weight more frequently.

72. What is an activation function?

An activation function is a function that is incorporated into an artificial neural network to aid in the network’s learning of complicated patterns in the input data. In contrast to a neuron-based model seen in human brains, the activation function determines what signals should be sent to the following neuron at the very end.

73. How Do You Build a random forest model?

The steps for creating a random forest model are as follows:

- Choose n from a dataset of k records.
- Create distinct decision trees for each of the n data values being taken into account. From each of them, a projected result is obtained.
- Each of the findings is subjected to a voting mechanism.
- The final outcome is determined by whose prediction received the most support.

74. Can you avoid overfitting your model? if yes, then how?

In actuality, data models may be overfitting. For it, the strategies listed below can be applied:

- Increase the amount of data in the dataset under study to make it simpler to separate the links between the input and output variables.
- To discover important traits or parameters that need to be examined, use feature selection.
- Use regularization strategies to lessen the variation of the outcomes a data model generates.
- Rarely, datasets are stabilized by adding a little amount of noisy data. This practice is called data augmentation.

75. What is Cross Validation?

Cross-validation is a model validation method used to assess the generalizability of statistical analysis results to other data sets. It is frequently applied when forecasting is the main objective and one wants to gauge how well a model will work in real-world applications.

In order to prevent overfitting and gather knowledge on how the model will generalize to different data sets, cross-validation aims to establish a data set to test the model during the training phase (i.e. validation data set).

76. What is variance in Data Science?

Variance is a type of error that occurs in a Data Science model when the model ends up being too complex and learns features from data, along with the noise that exists in it. This kind of error can occur if the algorithm used to train the model has high complexity, even though the data and the underlying patterns and trends are quite easy to discover. This makes the model a very sensitive one that performs well on the training dataset but poorly on the testing dataset, and on any kind of data that the model has not yet seen. Variance generally leads to poor accuracy in testing and results in overfitting.

77. What is pruning in a decision tree algorithm?

Pruning a decision tree is the process of removing the sections of the tree that are not necessary or are redundant. Pruning leads to a smaller decision tree, which performs better and gives higher accuracy and speed.

78. What is entropy in a decision tree algorithm?

In a decision tree algorithm, entropy is the measure of impurity or randomness. The entropy of a given dataset tells us how pure or impure the values of the dataset are. In simple terms, it tells us about the variance in the dataset.

```
Entropy(D) = - p * log2(p) - (1 - p) * log2(1 - p)
where:
Entropy(D) represents the entropy of the dataset D
p represents the proportion of positive class instances in D
log2 represents the logarithm to the base 2.
```

For example, suppose we are given a box with 10 blue marbles. Then, the entropy of the box is 0 as it contains marbles of the same color, i.e., there is no impurity. If we need to draw a marble from the box, the probability of it being blue will be 1.0. However, if we replace 4 of the blue marbles with 4 red marbles in the box, then the entropy increases to 0.4 for drawing blue marbles.

Additionally, In a decision tree algorithm, multi-class entropy is a measure used to evaluate the impurity or disorder of a dataset with respect to the class labels when there are multiple classes involved. It is commonly used as a criterion to make decisions about splitting nodes in a decision tree.

79. What information is gained in a decision tree algorithm?

When building a decision tree, at each step, we have to create a node that decides which feature we should use to split data, i.e., which feature would best separate our data so that we can make predictions. This decision is made using information gain, which is a measure of how much entropy is reduced when a particular feature is used to split the data. The feature that gives the highest information gain is the one that is chosen to split the data.

Let’s consider a practical example to gain a better understanding of how information gain operates within a decision tree algorithm. Imagine we have a dataset containing customer information such as age, income, and purchase history. Our objective is to predict whether a customer will make a purchase or not.

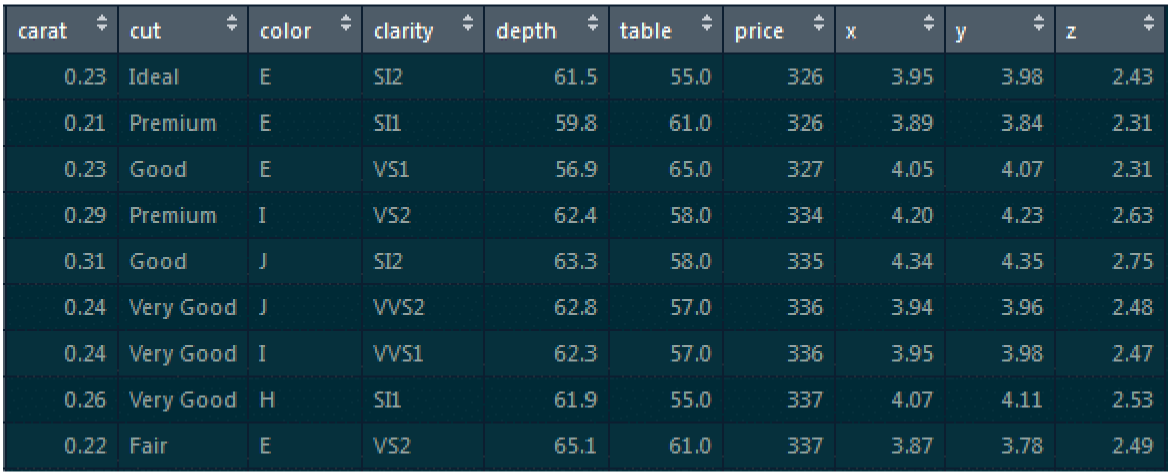
To determine which attribute provides the most valuable information, we calculate the information gain for each attribute. If splitting the data based on income leads to subsets with significantly reduced entropy, it indicates that income plays a crucial role in predicting purchase behavior. Consequently, income becomes a crucial factor in constructing the decision tree as it offers valuable insights.

By maximizing information gain, the decision tree algorithm identifies attributes that effectively reduce uncertainty and enable accurate splits. This process enhances the model's predictive accuracy, enabling informed decisions pertaining to customer purchases.

Explore this [Data Science Course in Delhi]() and master decision tree algorithm.

Advanced Data Science Interview Questions

80. From the below given 'diamonds' dataset, extract only those rows where the 'price' value is greater than 1000 and the 'cut' is ideal.



carat	cut	color	clarity	depth	table	price	x	y	z
0.23	Ideal	E	SI2	61.5	55.0	326	3.95	3.98	2.43
0.21	Premium	E	SI1	59.8	61.0	326	3.89	3.84	2.31
0.23	Good	E	VS1	56.9	65.0	327	4.05	4.07	2.31
0.29	Premium	I	VS2	62.4	58.0	334	4.20	4.23	2.63
0.31	Good	J	SI2	63.3	58.0	335	4.34	4.35	2.75
0.24	Very Good	J	VVS2	62.8	57.0	336	3.94	3.96	2.48
0.24	Very Good	I	VVS1	62.3	57.0	336	3.95	3.98	2.47
0.26	Very Good	H	SI1	61.9	55.0	337	4.07	4.11	2.53
0.22	Fair	E	VS2	65.1	61.0	337	3.87	3.78	2.49

First, we will load the **ggplot2** package:

```
library(ggplot2)
```

Next, we will use the **dplyr** package:

```
library(dplyr)// It is based on the grammar of data manipulation.
```

To extract those particular records, use the below command:

```
diamonds %>% filter(price>1000 & cut==”Ideal”)-> diamonds_1000_idea
```

81. Make a scatter plot between 'price' and 'carat' using ggplot. 'Price' should be on the y-axis, 'carat'

should be on the x-axis, and the 'color' of the points should be determined by 'cut.'

We will implement the scatter plot using **ggplot**.

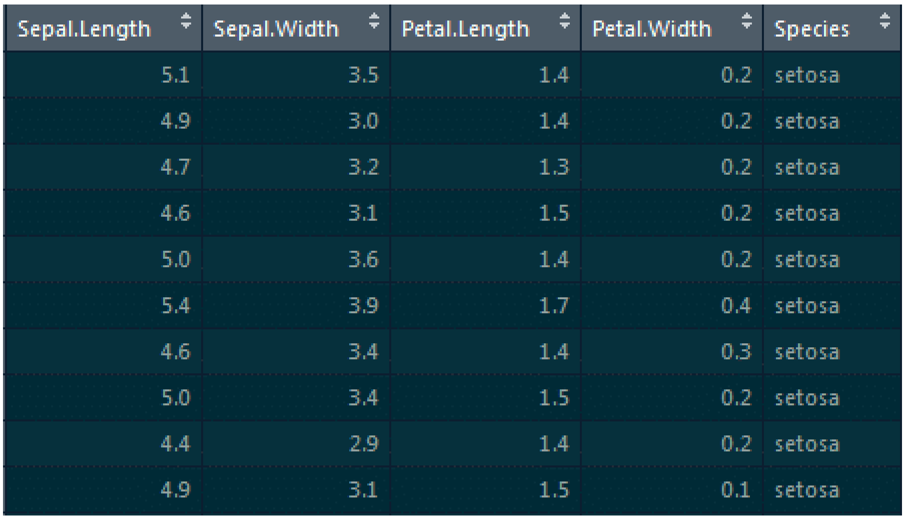
The ggplot is based on the grammar of data visualization, and it helps us stack multiple layers on top of each other.

So, we will start with the data layer, and on top of the data layer we will stack the aesthetic layer. Finally, on top of the aesthetic layer we will stack the geometry layer.

Code:

```
>ggplot(data=diamonds, aes(x=caret, y=price, col=cut))+geom_point()
```

82. Introduce 25 percent missing values in this 'iris' dataset and impute the 'Sepal.Length' column with 'mean' and the 'Petal.Length' column with 'median.'



Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa

To introduce missing values, we will be using the **missForest** package:

```
library(missForest)
```

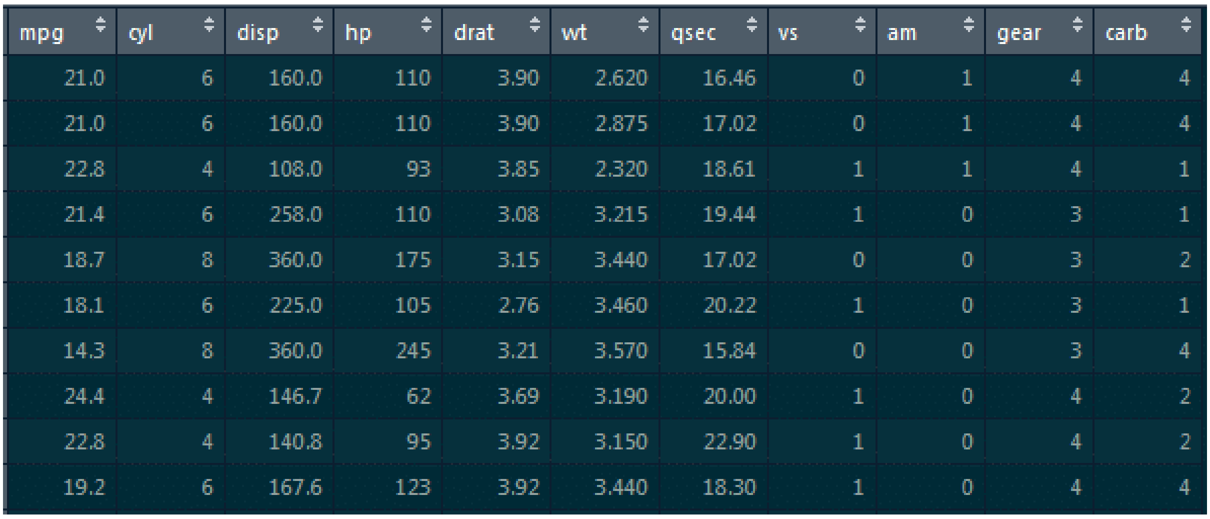
Using the prodNA function, we will be introducing 25 percent of missing values:

```
Iris.mis<-prodNA(iris,noNA=0.25)
```

For imputing the ‘Sepal.Length’ column with ‘mean’ and the ‘Petal.Length’ column with ‘median,’ we will be using the Hmisc package and the impute function:

```
library(Hmisc)
iris.mis$Sepal.Length<-with(iris.mis, impute(Sepal.Length,mean))
iris.mis$Petal.Length<-with(iris.mis, impute(Petal.Length,median))
```

83. Implement simple linear regression in R on this ‘mtcars’ dataset, where the dependent variable is ‘mpg’ and the independent variable is ‘disp.’



mpg	cyl	disp	hp	drat	wt	qsec	vs	am	gear	carb
21.0	6	160.0	110	3.90	2.620	16.46	0	1	4	4
21.0	6	160.0	110	3.90	2.875	17.02	0	1	4	4
22.8	4	108.0	93	3.85	2.320	18.61	1	1	4	1
21.4	6	258.0	110	3.08	3.215	19.44	1	0	3	1
18.7	8	360.0	175	3.15	3.440	17.02	0	0	3	2
18.1	6	225.0	105	2.76	3.460	20.22	1	0	3	1
14.3	8	360.0	245	3.21	3.570	15.84	0	0	3	4
24.4	4	146.7	62	3.69	3.190	20.00	1	0	4	2
22.8	4	140.8	95	3.92	3.150	22.90	1	0	4	2
19.2	6	167.6	123	3.92	3.440	18.30	1	0	4	4

Here, we need to find how ‘mpg’ varies w.r.t displacement of the column.

We need to divide this data into the training dataset and the testing dataset so that the model does not overfit the data.

So, what happens is when we do not divide the dataset into these two components, it overfits the dataset. Hence, when we add new data, it fails miserably on that new data.

Therefore, to divide this dataset, we would require the **caret** package. This caret package comprises the **createdatapartition()** function. This function will give the true or false labels.

Here, we will use the following code:

```
library(caret)
```

```
split_tag<-createDataPartition(mtcars$mpg, p=0.65, list=F)

mtcars[split_tag,]->train

mtcars[-split_tag,]->test

lm(mpg-data,data=train)->mod_mtcars

predict(mod_mtcars,newdata=test)->pred_mtcars

>head(pred_mtcars)
```

Explanation:

Parameters of the createDataPartition function: First is the column which determines the split (it is the mpg column).

Second is the split ratio which is 0.65, i.e., 65 percent of records will have true labels and 35 percent will have false labels. We will store this in a split_tag object.

Once we have the split_tag object ready, from this entire mtcars dataframe, we will select all those records where the split tag value is true and store those records in the training set.

Similarly, from the mtcars dataframe, we will select all those record where the split_tag value is **false** and store those records in the **test** set.

So, the split tag will have true values in it, and when we put '-' symbol in front of it, '-split_tag' will contain all of the false labels. We will select all those records and store them in the test set.

We will go ahead and build a model on top of the training set, and for the simple linear model we will require the **lm function**.

```
lm(mpg-data,data=train)->mod_mtcars
```

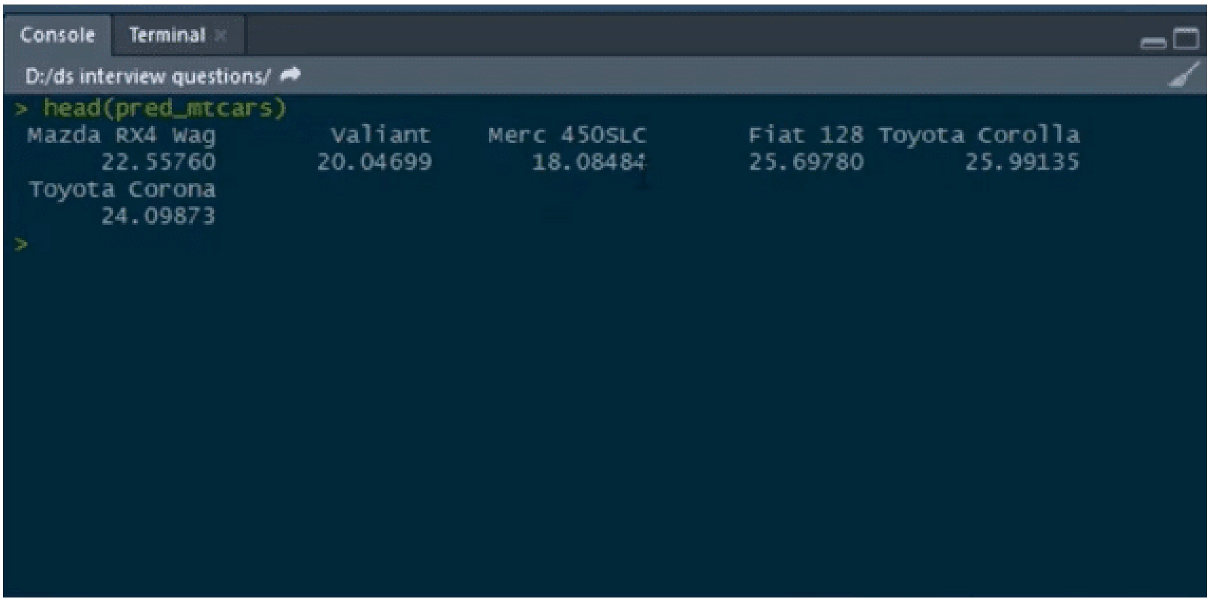
Now, we have built the model on top of the train set. It's time to predict the values on top of the test set. For that, we will use the predict function that takes in two parameters: first is the model which we have built and the second is the dataframe on which we have to predict values.

Thus, we have to predict values for the test set and then store them in pred_mtcars.

```
predict(mod_mtcars,newdata=test)->pred_mtcars
```

Output:

These are the predicted values of mpg for all of these cars.



```
> head(pred_mtcars)
 Mazda RX4 Wag          Valiant    Merc 450SLC         Fiat 128 Toyota Corolla
      22.55760         20.04699       18.08484         25.69780       25.99135
 Toyota Corona
      24.09873
>
```

So, this is how we can build a simple linear model on top of this mtcars dataset.

84. Calculate the RMSE values for the model building.

When we build a regression model, it predicts certain *y* values associated with the given *x* values, but there is always an error associated with this prediction. So, to get an estimate of the average error in prediction, RMSE is used.

Code:

```
cbind(Actual=test$mpg, predicted=pred_mtcars)->final_data

as.data.frame(final_data)->final_data

error<-(final_data$Actual-final_data$Prediction)
```

```
cbind(final_data,error)->final_data

sqrt(mean(final_data$error)^2)
```

Explanation: We have the actual and the predicted values. We will bind both of them into a single dataframe. For that, we will use the **cbind** function:

```
cbind(Actual=test$mpg, predicted=pred_mtcars)->final_data
```

Our actual values are present in the mpg column from the test set, and our predicted values are stored in the pred_mtcars object which we have created in the previous question. Hence, we will create this new column and name the column actual. Similarly, we will create another column and name it predicted which will have predicted values, and then store the predicted values in the new object which is final_data. After that, we will convert a matrix into a dataframe. So, we will use the as.data.frame function and convert this object (predicted values) into a dataframe:

```
as.data.frame(final_data)->final_data
```

We will pass this object which is final_data and store the result in final_data again. We will then calculate the error in prediction for each of the records by subtracting the predicted values from the actual values:

```
error<-(final_data$Actual-final_data$Prediction)
```

Then, store this result on a new object and name that object as **error**. After this, we will bind this error calculated to the same final_data dataframe:

```
cbind(final_data,error)->final_data //binding error object to this
final_data
```

Here, we bind the error object to this final_data, and store this into final_data again. **Calculating RMSE**:

```
Sqrt(mean(final_data$error)^2)
```

Output:

```
[1] 4.334423
```

Note: Lower the value of RMSE, the better the model. ***R and Python are two of the most important programming languages for Machine Learning Algorithms.***

85. Implement simple linear regression in Python on this ‘Boston’ dataset where the dependent variable is ‘medv’ and the independent variable is ‘lstat.’

Simple Linear Regression

```
import pandas as pd

data=pd.read_csv(‘Boston.csv’)     //loading the Boston dataset

data.head()  //having a glance at the head of this data

data.shape
```

Let us take out the dependent and the independent variables from the dataset:

```
data1=data.loc[:,[‘lstat’,’medv’]]

data1.head()
```

Visualizing Variables

```
import matplotlib.pyplot as plt

data1.plot(x=’lstat’,y=’medv’,style=’o’)

plt.xlabel(‘lstat’)

plt.ylabel(‘medv’)

plt.show()
```

Here, ‘medv’ is basically the median value of the price of the houses, and we are trying to find out the median values of the price of the houses with respect to to the lstat column.

We will separate the dependent and the independent variable from this entire dataframe:

```
data1=data.loc[:,[‘lstat’,’medv’]]
```

The only columns we want from all of this record are ‘lstat’ and ‘medv,’ and we need to store these results in data1.

Now, we would also do a visualization w.r.t to these two columns:

```
import matplotlib.pyplot as plt

data1.plot(x=’lstat’,y=’medv’,style=’o’)

plt.xlabel(‘lstat’)

plt.ylabel(‘medv’)

plt.show()
```

Preparing the Data

```
X=pd.Dataframe(data1[‘lstat’])

Y=pd.Dataframe(data1[‘medv’])

from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=100)

from sklearn.linear_model import LinearRegression

regressor=LinearRegression()

regressor.fit(X_train,y_train)
print(regressor.intercept_)
```

Output :

```
34.12654201
print(regressor.coef_)//this is the slope
```

Output :

```
[[-0.913293]]
```

By now, we have built the model. Now, we have to predict the values on top of the test set:

```
y_pred=regressor.predict(X_test)//using the instance and the predict
function and pass the X_test object inside the function and store
this in the y_pred object
```

Now, let’s have a glance at the rows and columns of the actual values and the predicted values:

```
Y_pred.shape, y_test.shape
```

Output :

```
((102,1),(102,1))
```

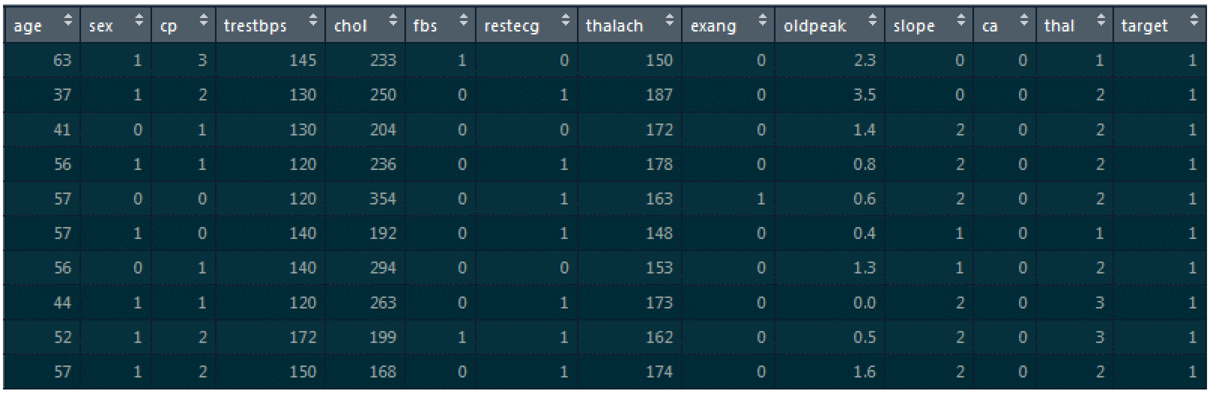
Further, we will go ahead and calculate some metrics so that we can find out the Mean Absolute Error, Mean Squared Error, and RMSE.

```
from sklearn import metrics import NumPy as np


print(‘Mean Absolute Error: ’, metrics.mean_absolute_error(y_test,
y_pred))


print(‘Mean Squared Error: ’, metrics.mean_squared_error(y_test,
y_pred))


print(‘Root Mean Squared Error: ’,
np.sqrt(metrics.mean_absolute_error(y_test, y_pred))
```

Output:

```
Mean Absolute Error: 4.692198


Mean Squared Error: 43.9198


Root Mean Squared Error: 6.6270
```

86. Implement logistic regression on this ‘heart’ dataset in R where the dependent variable is ‘target’ and the independent variable is ‘age.’



age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
37	1	2	130	250	0	1	187	0	3.5	0	0	2	1
41	0	1	130	204	0	0	172	0	1.4	2	0	2	1
56	1	1	120	236	0	1	178	0	0.8	2	0	2	1
57	0	0	120	354	0	1	163	1	0.6	2	0	2	1
57	1	0	140	192	0	1	148	0	0.4	1	0	1	1
56	0	1	140	294	0	0	153	0	1.3	1	0	2	1
44	1	1	120	263	0	1	173	0	0.0	2	0	3	1
52	1	2	172	199	1	1	162	0	0.5	2	0	3	1
57	1	2	150	168	0	1	174	0	1.6	2	0	2	1

For loading the dataset, we will use the **read.csv** function:

```
read.csv(“D:/heart.csv”)->heart

str(heart)
```

In the structure of this dataframe, most of the values are integers. However, since we are building a logistic regression model on top of this dataset, the final **target column is supposed to be categorical**. It cannot be an integer. So, we will go ahead and convert them into a factor. Thus, we will use the **as.factor** function and convert these integer values into categorical data.
We will pass on **heart$target** column over here and store the result in **heart$target** as follows:

```
as.factor(heart$target)->heart$target
```

Now, we will build a logistic regression model and see the different probability values for the person to have heart disease on the basis of different age values.

To build a logistic regression model, we will use the **glm** function:

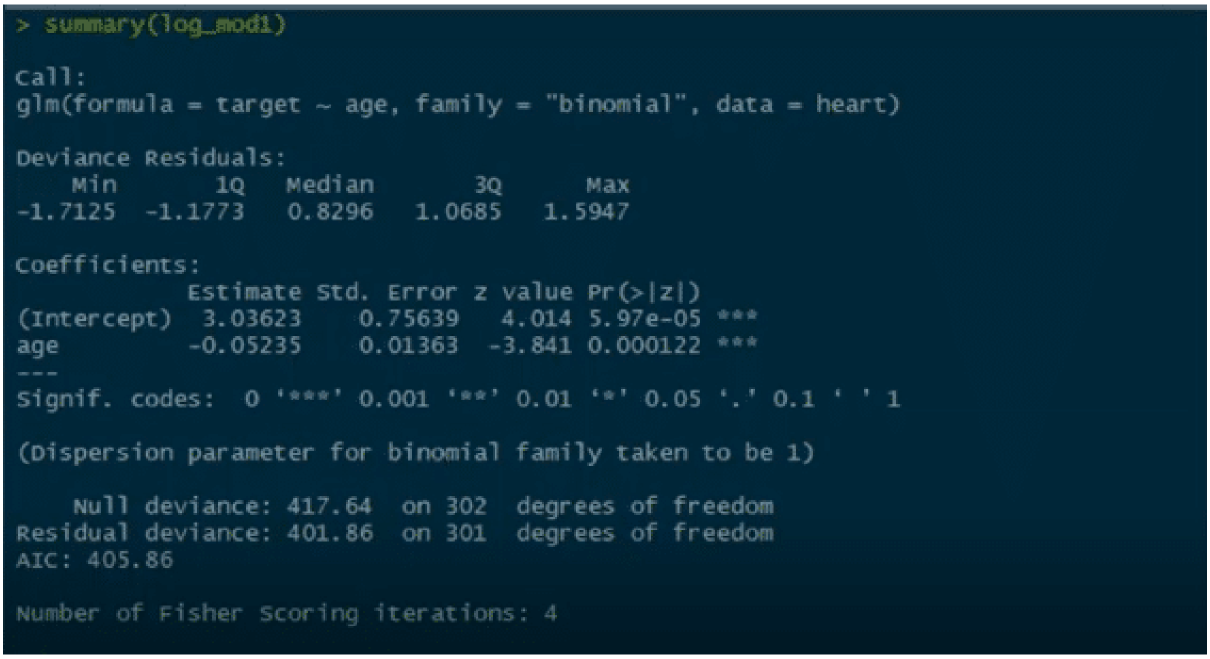
```
glm(target~age, data=heart, family=”binomial”)->log_mod1
```

Here, **target~age** indicates that the target is the dependent variable and the age is the independent variable, and we are building this model on top of the dataframe.

family=”binomial” means we are basically telling R that this is the logistic regression model, and we will store the result in **log_mod1**.

We will have a glance at the summary of the model that we have just built:

```
summary(log_mod1)
```



```
> summary(log_mod1)

Call:
glm(formula = target ~ age, family = "binomial", data = heart)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.7125  -1.1773   0.8296   1.0685   1.5947

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  3.03623    0.75639   4.014 5.97e-05 ***
age         -0.05235    0.01363  -3.841 0.000122 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 417.64  on 302  degrees of freedom
Residual deviance: 401.86  on 301  degrees of freedom
AIC: 405.86

Number of Fisher Scoring iterations: 4
```

We can see **Pr** value here, and there are three stars associated with this Pr value. This basically means that we can reject the null hypothesis which states that there is no relationship between the age and the target columns. But since we have three stars over here, this null hypothesis can be rejected. There is a strong relationship between the age column and the target column.

Now, we have other parameters like null deviance and residual deviance. Lower the deviance value, the better the model.

This null deviance basically tells the deviance of the model, i.e., when we don’t have any independent variable and we are trying to predict the value of the target column with only the intercept. When that’s the case, the null deviance is 417.64.

Residual deviance is wherein we include the independent variables and try to predict the target columns. Hence, when we include the independent variable which is age, we see that the residual deviance drops. Initially, when there are no independent variables, the null deviance was 417. After we include the age column, we see that the null deviance is reduced to 401.

This basically means that there is a strong relationship between the age column and the target column and that is why the deviance is reduced.

As we have built the model, it’s time to predict some values:

```
predict(log_mod1, data.frame(age=30), type=”response”)

predict(log_mod1, data.frame(age=50), type=”response”)

predict(log_mod1, data.frame(age=29:77), type=”response”)
```

Now, we will divide this dataset into train and test sets and build a model on top of the train set and predict the values on top of the test set:

```
>library(caret)

Split_tag<- createDataPartition(heart$target, p=0.70, list=F)

heart[split_tag,]->train

heart[-split_tag,]->test

glm(target~age, data=train,family=”binomial”)->log_mod2

predict(log_mod2, newdata=test, type=”response”)->pred_heart

range(pred_heart)
```

87. Build a ROC curve for the model built

The below code will help us in building the ROC curve:

```
library(ROCR)

prediction(pred_heart, test$target)-> roc_pred_heart

performance(roc_pred_heart, “tpr”, “fpr”)->roc_curve

plot(roc_curve, colorize=T)
```

Graph:

Go through this Data Science Course in Enakulam to get a clear understanding of Data Science!

88. Build a confusion matrix for the model where the threshold value for the probability of predicted values is 0.6, and also find the accuracy of the model.

Accuracy is calculated as:

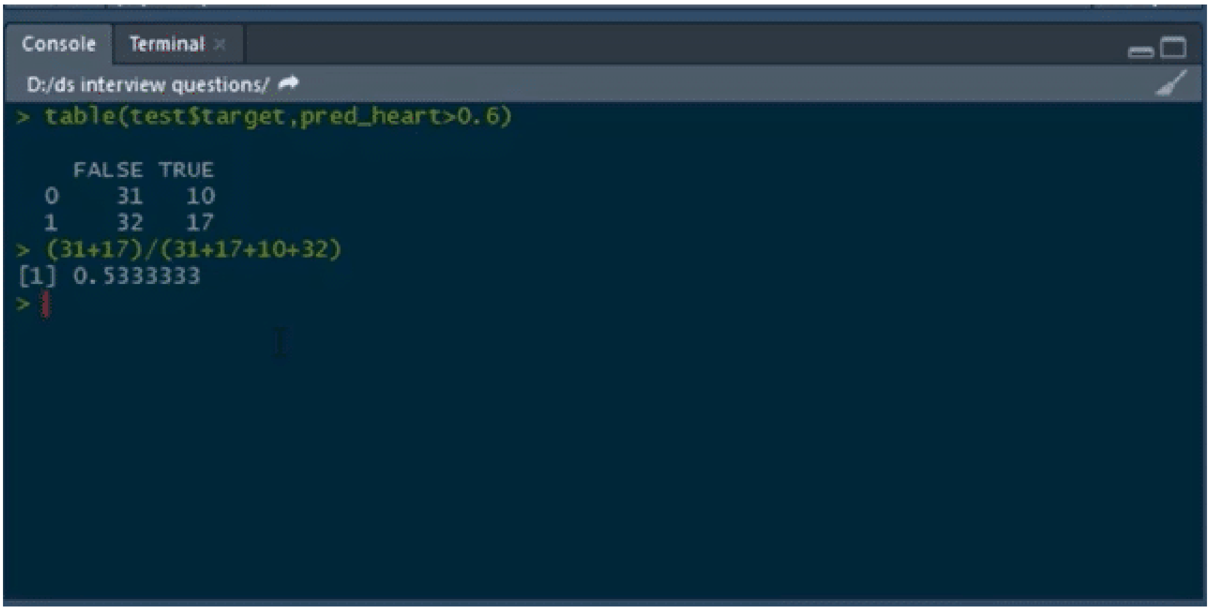
Accuracy = (True positives + true negatives)/(True positives+ true negatives + false positives + false negatives)

To build a confusion matrix in R, we will use the table function:

```
table(test$target,pred_heart>0.6)
```

Here, we are setting the probability threshold as 0.6. So, wherever the probability of pred_heart is greater than 0.6, it will be classified as 0, and wherever it is less than 0.6 it will be classified as 1.

Then, we calculate the accuracy by the formula for calculating **Accuracy**.

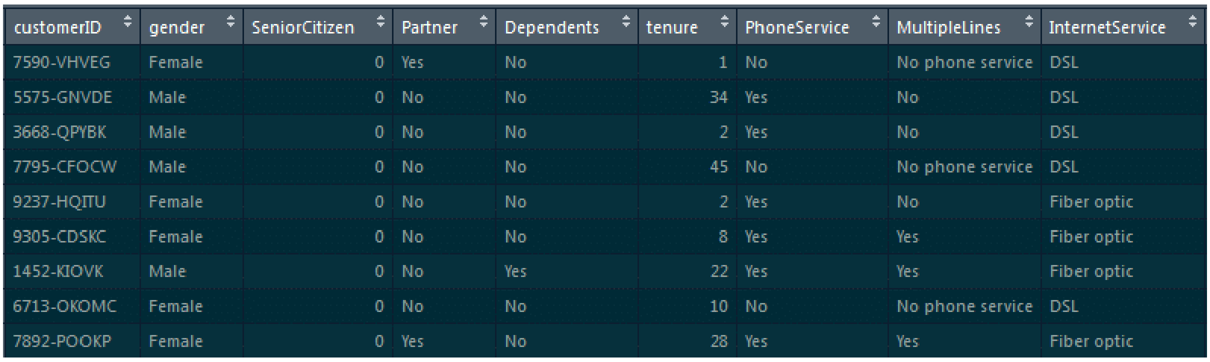


```
> table(test$target,pred_heart>0.6)

    FALSE TRUE
  0    31   10
  1    32   17
> (31+17)/(31+17+10+32)
[1] 0.5333333
>
```

89. Build a logistic regression model on the ‘customer_churn’ dataset in Python. The dependent variable is ‘Churn’ and the independent variable is ‘MonthlyCharges.’ Find the log_loss of the model.

First, we will load the pandas dataframe and the customer_churn.csv file:

```
customer_churn=pd.read_csv(“customer_churn.csv”)
```



customerID	gender	SeniorCitizen	Partner	Dependents	tenure	PhoneService	MultipleLines	InternetService
7590-VHVEG	Female	0	Yes	No	1	No	No phone service	DSL
5575-GNVDE	Male	0	No	No	34	Yes	No	DSL
3668-QPYBK	Male	0	No	No	2	Yes	No	DSL
7795-CFOCW	Male	0	No	No	45	No	No phone service	DSL
9237-HQITU	Female	0	No	No	2	Yes	No	Fiber optic
9305-CDSKC	Female	0	No	No	8	Yes	Yes	Fiber optic
1452-KIOVK	Male	0	No	Yes	22	Yes	Yes	Fiber optic
6713-OKOMC	Female	0	No	No	10	No	No phone service	DSL
7892-POOKP	Female	0	Yes	No	28	Yes	Yes	Fiber optic

After loading this dataset, we can have a glance at the head of the dataset by using the following command:

```
customer_churn.head()
```

Now, we will separate the dependent and the independent variables into two separate objects:

```
x=pd.Dataframe(customer_churn[‘MonthlyCharges’])

y=customer_churn[‘ Churn’]

#Splitting the data into training and testing sets

from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.3,
random_state=0)
```

Now, we will see how to build the model and calculate **log_loss**.

```
from sklearn.linear_model, we have to import LogisticRegression

l=LogisticRegression()

l.fit(x_train,y_train)

y_pred=l.predict_proba(x_test)
```

As we are supposed to calculate the log_loss, we will import it from **sklearn.metrics**:

```
from sklearn.metrics import log_loss
```

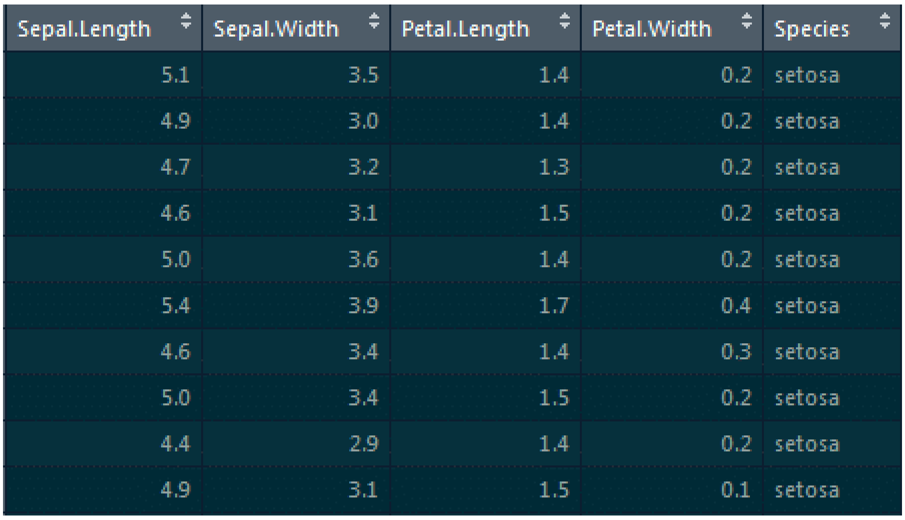
```
print(log_loss(y_test,y_pred)//actual values are in y_test and
predicted are in y_pred
```

Output:

```
0.5555020595194167
```

Become a master of Data Science by going through this online Data Science Course in Dehradun!

90. Build a decision tree model on 'Iris' dataset where the dependent variable is 'Species,' and all other columns are independent variables. Find the accuracy of the model built.



Sepal.Length	Sepal.Width	Petal.Length	Petal.Width	Species
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa

To build a decision tree model, we will be loading the **party** package:

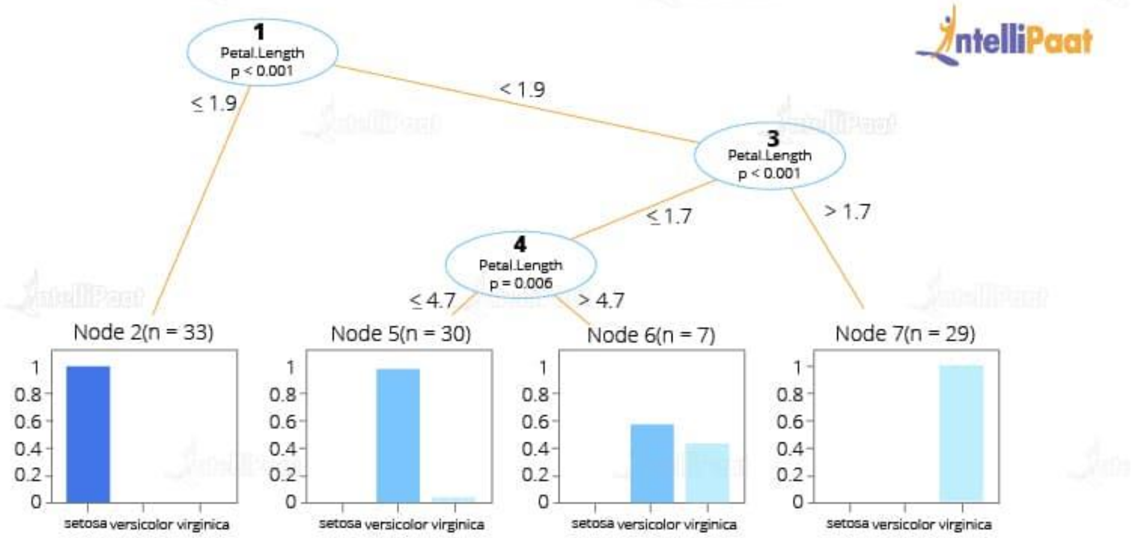
```
#party package

library(party)

#splitting the data

library(caret)

split_tag<-createDataPartition(iris$Species, p=0.65, list=F)

iris[split_tag,]->train
```

```
iris[~split_tag,]->test

#building model

mytree<-ctree(Species~.,train)
```

Now we will plot the model

```
plot(mytree)
```

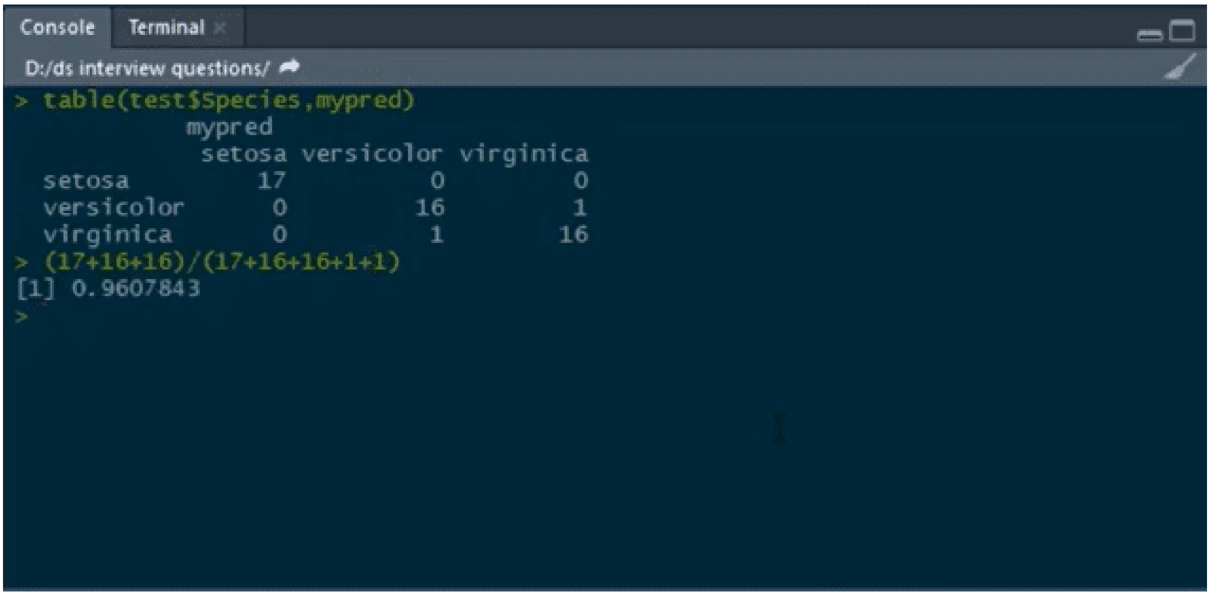
Model:



```
#predicting the values

predict(mytree,test,type=’response’)->mypred
```

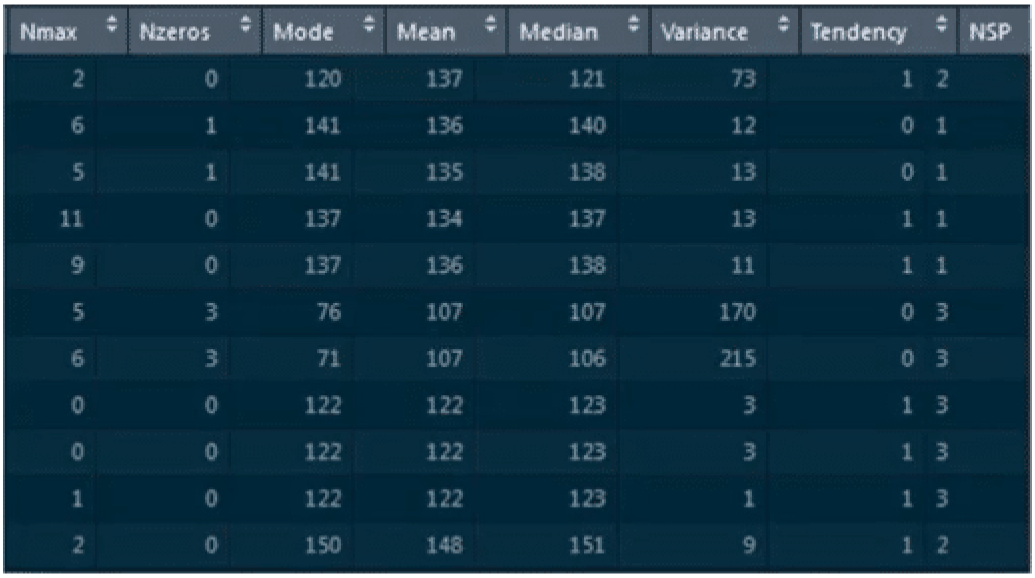
After this, we will predict the confusion matrix and then calculate the accuracy using the table function:

```
table(test$Species, mypred)
```



```
> table(test$Species,mypred)
            mypred
             setosa versicolor virginica
  setosa         17          0         0
  versicolor      0         16         1
  virginica       0          1        16
> (17+16+16)/(17+16+16+1+1)
[1] 0.9607843
>
```

91. Build a random forest model on top of this ‘CTG’ dataset, where ‘NSP’ is the dependent variable and all other columns are independent variables.



Nmax	Nzeros	Mode	Mean	Median	Variance	Tendency	NSP
2	0	120	137	121	73	1	2
6	1	141	136	140	12	0	1
5	1	141	135	138	13	0	1
11	0	137	134	137	13	1	1
9	0	137	136	138	11	1	1
5	3	76	107	107	170	0	3
6	3	71	107	106	215	0	3
0	0	122	122	123	3	1	3
0	0	122	122	123	3	1	3
1	0	122	122	123	1	1	3
2	0	150	148	151	9	1	2

We will load the CTG dataset by using **read.csv**:

```
data<-read.csv(“C:/Users/intellipaat/Downloads/CTG.csv”,header=True)


str(data)
```

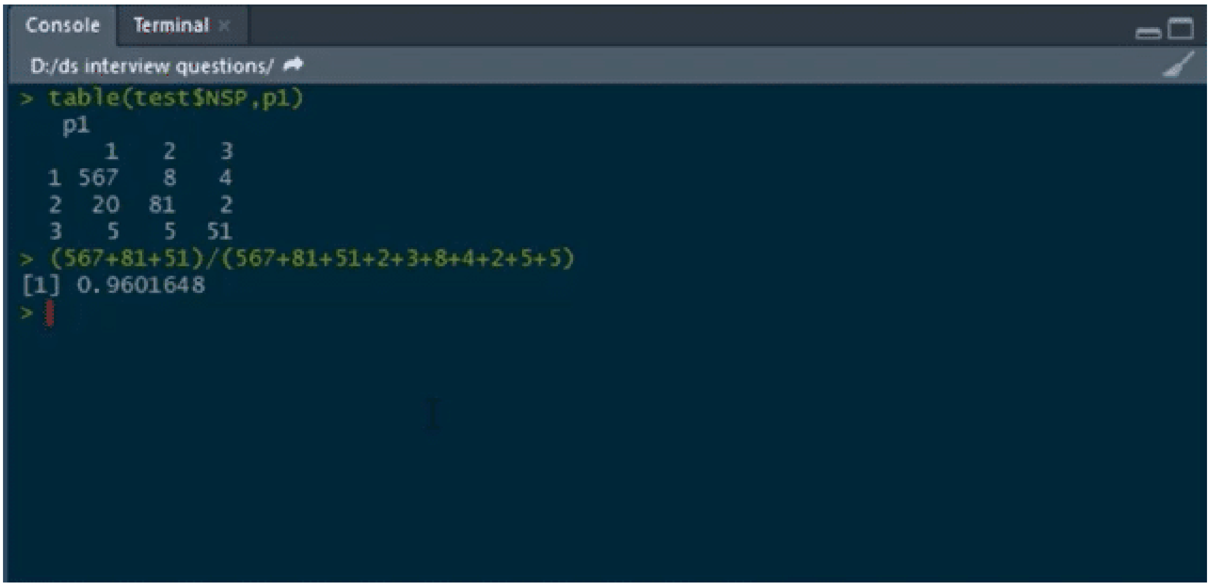
Converting the integer type to a factor

```
data$NSP<-as.factor(data$NSP)
```

```
table(data$NSP)

#data partition

set.seed(123)

split_tag<-createDataPartition(data$NSP, p=0.65, list=F)

data[split_tag,]->train

data[~split_tag,]->test

#random forest -1

library(randomForest)

set.seed(222)

rf<-randomForest(NSP~.,data=train)

rf

#prediction

predict(rf,test)->p1
```

Building confusion matrix and calculating accuracy:

```
table(test$NSP,p1)
```



```
> table(test$NSP,p1)
   p1
      1   2   3
  1 567   8   4
  2  20  81   2
  3   5   5  51
> (567+81+51)/(567+81+51+2+3+8+4+2+5+5)
[1] 0.9601648
> 
```

If you have any doubts or queries related to Data Science, get them clarified from Data Science experts on our [Data Science Community](#)!

92. Write a function to calculate the Euclidean distance between two points.

The formula for calculating the Euclidean distance between two points (x1, y1) and (x2, y2) is as follows:

```
√(((x1 - x2) ^ 2) + ((y1 - y2) ^ 2))
```

Code for calculating the Euclidean distance is as given below:

```
def euclidean_distance(P1, P2):
return (((P1[0] - P2[0]) ** 2) + ((P1[1] - P2[1]) ** 2)) ** .5
```

93. Write code to calculate the root mean square error (RMSE) given the lists of values as actual and predicted.

To calculate the root mean square error (RMSE), we have to:

1. Calculate the errors, i.e., the differences between the actual and the predicted values
2. Square each of these errors

3. Calculate the mean of these squared errors
4. Return the square root of the mean

The code in Python for calculating RMSE is given below:

```
def rmse(actual, predicted):
  errors = [abs(actual[i] - predicted[i]) for i in range(0, 
len(actual))]
  squared_errors = [x ** 2 for x in errors]
  mean = sum(squared_errors) / len(squared_errors)
  return mean ** .5
```

Check out this Machine Learning Course to get an in-depth understanding of Machine Learning.

94. Mention the different kernel functions that can be used in SVM.

In SVM, there are four types of kernel functions:

- **Linear kernel**
  In SVM (Support Vector Machines), a linear kernel is a type of kernel function used to transform input data into a higher-dimensional feature space. It is represented by the equation K(x, y) = x • y, where x and y are feature vectors. The linear kernel calculates the dot product between the two vectors to measure their similarity or dissimilarity.
- **Polynomial kernel**
  A polynomial kernel is a type of kernel function used to transform input data into a higher-dimensional feature space. It is represented by the equation K(x, y) = (x • y + c)^d, where x and y are feature vectors, c is a constant, and d is the degree of the polynomial. The polynomial kernel captures nonlinear relationships between data points by raising the dot product to a specified power.

- **Radial basis kernel**
  In SVM (Support Vector Machines), a radial basis kernel, also known as the Gaussian kernel, is a popular kernel function used for non-linear classification. It is represented by the equation K(x, y) = exp(-gamma * ||x – y||^2), where x and y are feature vectors, and gamma is a parameter that determines the influence of each training example. The radial basis kernel measures the similarity between data points based on their Euclidean distance in the feature space.
- **Sigmoid kernel**
  The sigmoid kernel is a type of non-linear kernel function commonly employed for classification tasks. It can be mathematically described by the equation K(x, y) = tanh(alpha * x * y + c), where x and y represent feature vectors, and alpha and c are parameters determining the sigmoid function’s shape. By utilizing the sigmoid kernel, Support Vector Machines (SVMs) can project data onto a higher-dimensional space, enabling the creation of non-linear decision boundaries for accurate classification.

95. How to detect if the time series data is stationary?

Time series data is considered stationary when variance or mean is constant with time. If the variance or mean does not change over a period of time in the dataset, then we can draw the conclusion that, for that period, the data is stationary.

96. Write code to calculate the accuracy of a binary classification algorithm using its confusion matrix.

We can use the code given below to calculate the accuracy of a binary classification algorithm:

```
def accuracy_score(matrix):
  true_positives = matrix[0][0]
  true_negatives = matrix[1][1]
  total_observations = sum(matrix[0]) + sum(matrix[1])
  return (true_positives + true_negatives) / total_observations
```

97. What does root cause analysis mean?

Root cause analysis is the process of figuring out the root causes that lead to certain faults or failures. A factor is considered to be a root cause if, after eliminating it, a sequence of operations, leading to a fault, error, or undesirable result, ends up working correctly. Root cause analysis is a technique that was initially developed and used in the analysis of industrial accidents, but now, it is used in a wide variety of areas.

98. What is A/B testing?

A/B testing is a kind of statistical hypothesis testing for randomized experiments with two variables. These variables are represented as A and B. A/B testing is used when we wish to test a new feature in a product. In the A/B test, we give users two variants of the product, and we label these variants as A and B.

The A variant can be the product with the new feature added, and the B variant can be the product without the new feature. After users use these two products, we capture their ratings for the product.

If the rating of product variant A is statistically and significantly higher, then the new feature is considered an improvement and useful and is accepted. Otherwise, the new feature is removed from the product.

Check out this Python Course to get deeper into Python programming.

99. Out of collaborative filtering and content-based filtering, which one is considered better, and why?

Content-based filtering is considered to be better than collaborative filtering for generating recommendations. It does not mean that collaborative filtering generates bad recommendations.

However, as collaborative filtering is based on the likes and dislikes of other users we cannot rely on it much. Also, users' likes and dislikes may change in the future.

For example, there may be a movie that a user likes right now but did not like 10 years ago. Moreover, users who are similar in some features may not have the same taste in the kind of content that the platform provides.

In the case of content-based filtering, we make use of users' own likes and dislikes which are much more reliable and yield more positive results. This is why platforms such as Netflix, Amazon Prime, Spotify, etc. make use of content-based filtering for generating recommendations for their users.

100. In the following confusion matrix, calculate precision and recall.

Total = 510	**Actual**		
Predicted		P	N
	P	156	11
	N	16	327

The formulae for precision and recall are given below.

```
Precision:
(True Positive) / (True Positive + False Positive)
Recall:
(True Positive) / (True Positive + False Negative)
Based on the given data, precision and recall are:
Precision: 156 / (156 + 11) = 93.4
Recall: 156 / (156 + 16) = 90.7
```

101. Write a function that when called with a confusion matrix for a binary classification model returns a dictionary with its precision and recall.

We can use the below for this purpose:

```
def calculate_precsion_and_recall(matrix):
  true_positive  = matrix[0][0]
  false_positive  = matrix[0][1]
  false_negative = matrix[1][0]
  return {
    'precision': (true_positive) / (true_positive + false_positive),
    'recall': (true_positive) / (true_positive + false_negative)
  }
```

102. What is reinforcement learning?

Reinforcement learning is a kind of Machine Learning, which is concerned with building software agents that perform actions to attain the most cumulative rewards.

A reward here is used for letting the model know (during training) if a particular action leads to the attainment of or brings it closer to the goal. For example, if we are creating an ML model that plays a video game, the reward is going to be either the points collected during the play or the level reached in it.

Reinforcement learning is used to build these kinds of agents that can make real-world decisions that should move the model toward the attainment of a clearly defined goal.

103. Explain TF/IDF vectorization.

The expression 'TF/IDF' stands for the Term Frequency–Inverse Document Frequency. It is a numerical measure that allows us to determine how important a word is to a document in a collection of documents called a corpus. TF/IDF is used often in text mining and information retrieval.

104. What are the assumptions required for linear regression?

There are several assumptions required for linear regression. They are as follows:

- The data, which is a sample drawn from a population, used to train the model should be representative of the population.
- The relationship between independent variables and the mean of dependent variables is linear.
- The variance of the residual is going to be the same for any value of an independent variable. It is also represented as X.
- Each observation is independent of all other observations.
- For any value of an independent variable, the independent variable is normally distributed.

105. What happens when some of the assumptions required for linear regression are violated?

These assumptions may be violated lightly (i.e., some minor violations) or strongly (i.e., the majority of the data has violations). Both of these violations will have different effects on a linear regression model.

Strong violations of these assumptions make the results entirely redundant. Light violations of these assumptions make the results have greater bias or variance.

106. How to deal with unbalanced binary classification?

Given below are the following points that will teach you to deal with unbalanced binary classification:

- Use other formulas to determine the model's performance, such as precision/recall, F1 score, etc.
- Re-sample the data using strategies such as undersampling (decreasing the sample size of the bigger class), oversampling (raising the sample size of the smaller class using repetition, SMOTE, and other similar strategies), and so on.
- K-fold cross-validation is used

- Use ensemble learning such that each decision tree only takes into account a portion of the bigger class and the complete sample of the smaller class.

107. Which cross-validation method would you use for a batch of time series data?

Instead of utilizing k-fold cross-validation, you should be aware that a time series is fundamentally organized by chronological order and is not made up of randomly dispersed data. Use approaches like forward-chaining, where you model on previous data and then look at forward-facing data, when dealing with time series data.

108. How can time-series data be declared as stationery?

The time series is considered stationary when its essential constituents don’t change over time. These variables might be variance or mean. Static time series exhibit no trends nor seasonal impacts. Data from stationary time series are required for data science models.

109. Difference between Point Estimates and Confidence Interval.

Point Estimates: A specific number known as the point estimate provides an estimate of the population parameter. The Maximum Likelihood estimator and the Method of Moments are two common techniques used to produce Population Parameter Point, estimators.

Confidence Interval: The confidence interval provides a range of values that most likely contain the population parameter. It even reveals the likelihood that the population parameter may be found in that specific period. The likelihood or similarity is represented by the Confidence Coefficient (or Confidence level), which is indicated by 1-alpha. The significance level is indicated by alpha.

110. Define the terms KPI, lift, model fitting, robustness, and DOE.

- KPI: KPI stands for Key Performance Indicator, which evaluates how successfully a company accomplishes its goals.
- Lift: Lift is a performance indicator for the target model compared to a random selection model. Lift measures how well the model predicts in comparison to no model.
- Model fitting: This describes how well the proposed model conforms to the available data.
- Robustness: This refers to how well the system can manage variations and changes.
- DOE: DOE refers to the task design with the goal of describing and explaining information variance under postulated circumstances to reflect variables.

111. What are LLMs?

Large Language Models, abbreviated as LLMs, are sophisticated artificial intelligence models designed to process and generate text that resembles human language based on the input they receive. They employ advanced techniques like deep learning, particularly neural networks, to comprehend and produce language patterns, enabling them to answer questions, engage in conversations, and provide information on a broad array of topics.

LLMs undergo training using extensive sets of textual data from diverse sources, including books, websites, and other text-based materials. Through this training, they acquire the ability to recognize patterns, comprehend context, and generate coherent and contextually appropriate responses.

Notable examples of LLMs, such as ChatGPT based on the GPT-3.5 architecture, have been trained on comprehensive and varied datasets to offer accurate and valuable information across different domains. These models possess natural language understanding capabilities and can undertake various tasks such as language translation, content generation, and text completion.

Their versatility allows them to assist users in diverse inquiries and tasks, making them valuable tools across numerous fields, including education, customer service, content creation, and research.

112. What is a Transformer in Machine Learning?

Within the realm of machine learning, the term "Transformer" denotes a neural network architecture that has garnered significant acclaim, primarily in the domain of natural language processing (NLP) tasks. Its introduction occurred in the seminal research paper titled "Attention Is All You Need," authored by Vaswani et al. in 2017. Since then, the Transformer has emerged as a fundamental framework in numerous applications within the NLP domain.

The Transformer architecture is purposefully designed to overcome the limitations encountered by conventional recurrent neural networks (RNNs) when confronted with sequential data, such as sentences or documents. Unlike RNNs, Transformers do not rely on sequential processing and possess the ability to parallelize computations, thereby facilitating enhanced efficiency and scalability.