Interview Query

Top 49 Python Data Science Interview Questions (Updated for 2024)

Written by **IQ Team**

Published May 28, 2024

Estimated reading time: 26 minutes

Table of contents

Introduction

Python interview questions feature prominently in data science technical interviews. You will likely be asked questions covering key Python coding concepts during a typical interview. Start your practice with these newly updated Python data science interview questions covering statistics, probability, string parsing, NumPy/matrices, and Pandas.

Python data science interview questions range from asking you what's the difference between a list and a tuple and asking you to **find all bigrams in a sentence** or to **implement the K-means algorithm from scratch**.

The most commonly covered topics in Python data science interview questions include:

- **Basic Python Interview Questions**
- **String Manipulation Interview Questions**
- **Statistics and Probability Interview Questions**
- **Python Pandas Interview Questions**

Why Is Python Asked in Data Science Interviews?

Python has reigned as the dominant language in data science over the past few years, taking over former strongholds such as R, Julia, Spark, and Scala. That is thanks in large part to its wide breadth of data science libraries (modules) supported by a strong and growing data science community.

One of the main reasons why Python is now the preferred language of choice is because Python libraries can extend their use to the full stack of data science. While each data science language has its own specialties, such as R for data analysis and modeling within academia and Spark and Scala for big data ETLs and production, Python has produced an ecosystem of libraries that all fit nicely together.

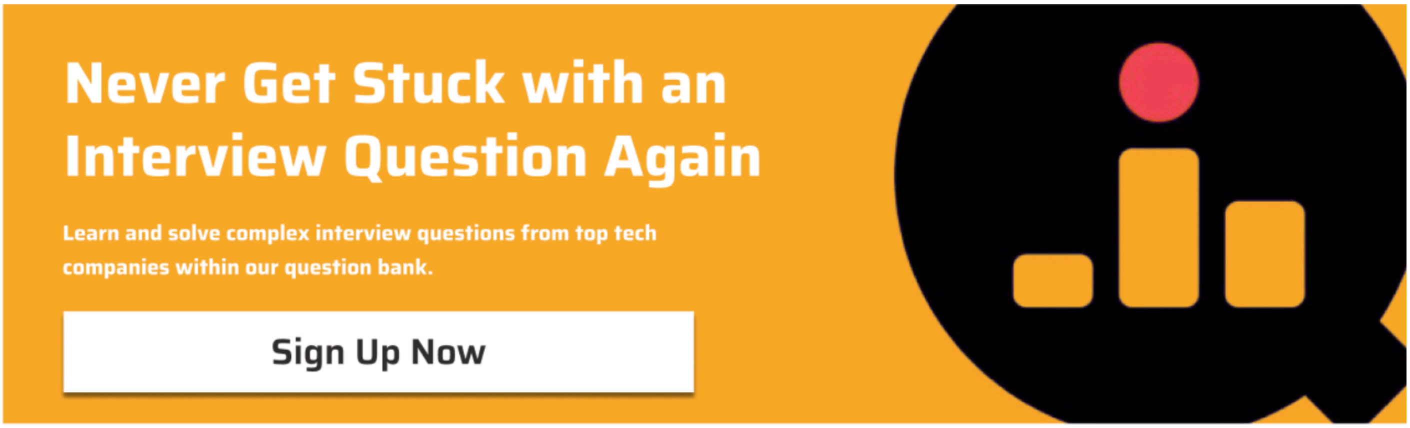
At the end of the day, it's much easier to program and perform full-stack data science without having to switch languages. This means running exploratory data analysis, creating graphs and visualization, building the model, and implementing the deployment, all in one language.

Who Gets Asked Python Questions?

Data scientists, data engineers, machine learning engineers, and data analysts face Python questions in interviews. However, the difficulty of the question is dependent on the role.

Here's how **Python questions differ between data analysts and data scientists:**

- **Data Analyst** - Data analyst Python questions are easier and are typically scripting-focused. In general, most questions will be easy Pandas and Python questions.
- **Data Scientist/Data Engineer** - More than two-thirds of data scientists use Python every day. Questions include basic concepts like conditions and branching, loops, functions, and object-oriented programming. Data engineer interview questions also focus heavily on common libraries like NumPy, SciPy and Pandas, as well as advanced concepts like regression, K-means clustering and classification.


Never Get Stuck with an Interview Question Again
Learn and solve complex interview questions from top tech companies within our question bank.
Sign Up Now

Basic Python Interview Questions

Although there are plenty of advanced technical questions, be sure you can quickly and competently answer basic questions like "What data types are used in Python?" and "What is a Python dictionary?" You don't want to get caught stumbling on an answer for a basic Python syntax question.

If possible, direct your responses back to work experiences or Python data science projects you have worked on.

1. What built-in data types are used in Python?

Python uses several built-in data types, including:

- Number (int, float and complex)

- String (str)
- Tuple (tuple)
- Range (range)
- List (list)
- Set (set)
- Dictionary (dict)

In Python, data types are used to classify or categorize data, and every value has a data type.

2. How are data analysis libraries used in Python? What are some of the most common libraries?

A key reason Python is such a popular data science programming language is because there is an extensive collection of data analysis libraries available. These libraries include functions, tools and methods for managing and analyzing data. There are Python libraries for performing a wide range of data science functions, including for the processing of image and textual data, data mining and data visualization. The most widely used Python data analysis libraries include:

- Pandas
- NumPy
- SciPy
- TensorFlow
- SciKit
- Seaborn
- Matplotlib

3. How is a negative index used in Python?

Negative indexes are used in Python to assess and index lists and arrays from the end of your string, moving backwards towards your first value. For example, n-1 will show the last item in a list, while n-2 will show the second to last. Here's an example of a negative index in Python:

```
b = "Python Coding Fun"
print(b[-1])
>> n
```

4. What is the difference between lists and tuples in Python?

Lists and tuples are classes in Python that store one or more objects or values. Key differences include:

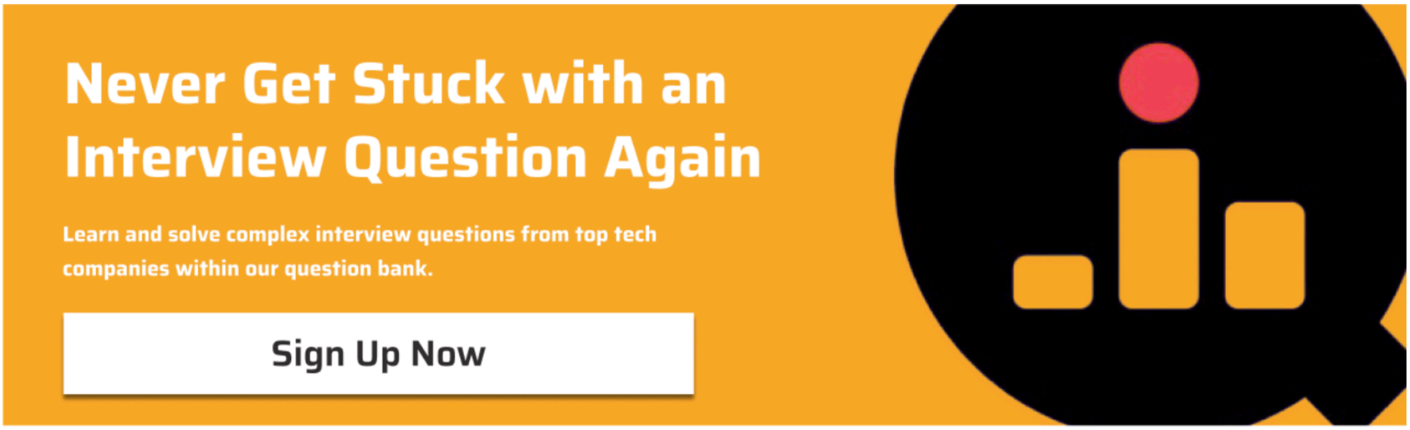
- **Syntax** – Lists are enclosed in square brackets and tuples are enclosed in parentheses.

- **Mutable vs. Immutable** – Lists are mutable, which means they can be modified after being created. Tuples are immutable, which means they cannot be modified.
- **Operations** – Lists have more functionalities available than tuples, including insert and pop operations, as well as sorting.
- **Size** – Because tuples are immutable, they require less memory and are subsequently faster.

5. What library would you prefer for plotting Seaborn or Matplotlib?

Seaborn and Matplotlib are two of the most popular visualization libraries in Python. One thing to note is that Seaborn is built on top of Matplotlib. However, Seaborn tends to offer more customization, thanks to its built-in tools. Therefore, Seaborn can make the work faster, and you could switch to Matplotlib for fine-tuning.

NOTE: This question asks about preferences. The library you choose might be dependent on the task or how familiar you are with the tool. In other words, there is no right or wrong answer; rather, the interviewer wants to understand how proficient you are at creating visualizations in Python.



6. Is Python an object-oriented programming language?

Yes and no. Python combines features of both object-oriented programming (OOP) and aspect-oriented programming. One reason it can’t be considered a true OOP language is that it doesn’t support strong encapsulation, which is the only basic feature of an OOP that Python does not support.

7. What is the difference between a series and a dataframe in Pandas?

Series only support a single list with index, whereas a dataframe supports one or more series. In other words:

- **Series** is a one-dimensional array that supports any datatype (including integers, strings, floats, etc.). In a series, the axis labels are the index.
- **A dataframe** is a two-dimensional data structure with columns that can support different data types. It is similar to a SQL table or a dictionary of series objects.

8. How would you find duplicate values in a dataset for a variable in Python?

You can check for duplicates using the Pandas duplicated() method. This will return a boolean series which is TRUE only for unique elements.

```
DataFrame.duplicated(subset=None,keep='last')
```

In this example, keep determines what to do with duplicates. You can use

- **First** - Considers the first value unique and the rest as duplicates.
- **Last** - Considers the last value unique and the rest as duplicates.
- **False** - Considers all same values as duplicates.

9. What is a lambda function in Python?

Sometimes called an “anonymous function,” the lambda function is just like a normal function but is not defined with the keyword. They are defined with the keyword. Lambda functions are restricted to a single line expression, and can take in multiple parameters, just like normal functions.

Here is an example of both normal and lambda functions for the argument (x) and the expression (x+x)

Normal function:

```
def function_name(x)
return x+x
```

Lambda function:

```
lambda x: x+x
```

10. Is memory de-allocated when you exit Python?

No. Modules with circular references to other objects are not always freed. It is also impossible to free some of the memory reserved by the C library.

11. What is a compound datatype?

Compound data structures are single variables that represent multiple values. Some of the most common in Python are:

- **Lists** - A collection of values where the order is important.
- **Tuples** - A sequence of values where the order is important.
- **Sets** - A collection of values where membership in the set is important.

12. What is list comprehension in Python? Provide an example.

List comprehension is used to define and create a list based on an existing list. For example, if we wanted to separate all the letters in the word “retain,” and make each letter a list item, we could use list comprehension:

```
r_letters = [ letter for letter in 'retain' ]
print( r_letters)
```

Output:

```
['r', 'e', 't', 'a', 'i', 'n']
```

13. What is tuple unpacking? Why is it important?

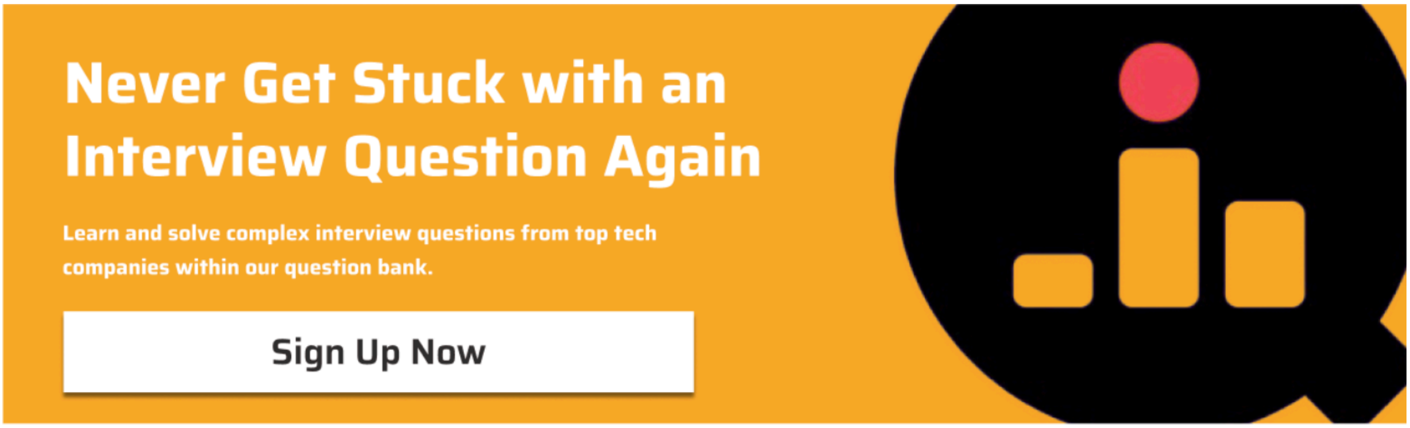
The short answer: Unpacking refers to the practice of assigning elements of a tuple to multiple variables. You use the * operator to assign elements of an unpacking assignment to assign it a value.

With unpacking, you can swap variables without using a temporary variable. For example:

```
x = 20
y = 30

print(f'x={x}, y={y}')

x, y = y, x

print(f'x={x}, y={y}')
```

Output:

```
x=20, y=30
x=30, y=20
```



14. What’s the difference between / and // in Python?

Both / and // are division operators. However, / does float division, dividing the first operand by the second. / returns the value in decimal form. // does floor division, dividing the first operand by the second, but returns the value in natural number form.

- / **example:** 9 / 2 returns 4.5
- // **example:** 9 / 2 returns 4

15. How do you convert integers to strings?

The most common way to convert an integer to a string in Python is with the built-in **str() function**. This function converts any data type into a string; however, there are other ways you can do this. You can turn to the **f-string function**, by using **"%s" keywords** or with the **.format function**.

16. What are arrays in Python?

Arrays store multiple values in one single variable. For example, you could create an array "faang" which included Facebook, Apple, Amazon, Netflix and Google.

Example:

```
faang = ["facebook", "apple", "amazon", "netflix", "google"]
print(faang)
```

Output:

```
['facebook', 'apple', 'amazon', 'netflix', 'google']
```

17. What's the difference between mutable and immutable objects?

In Python, mutable or immutable refers to whether or not the object's value can change. Mutable objects can change those values, while immutable objects cannot. Mutable data types include lists, sets, dictionaries and byte arrays. Immutable data types include numeric data types (boolean, float, etc.), strings, frozensets and tuples.

18. What are some of the limitations of Python?

Python is limited in a few key ways, including:

- **Speed** - Studies have shown that Python is slower than languages like Java and C++. However, there are options to make Python faster, like a custom runtime.
- **V2 vs V3** - Python 2 and Python 3 are incompatible.
- **Mobile development** - Python is great for desktop and server applications, but weaker for mobile development.

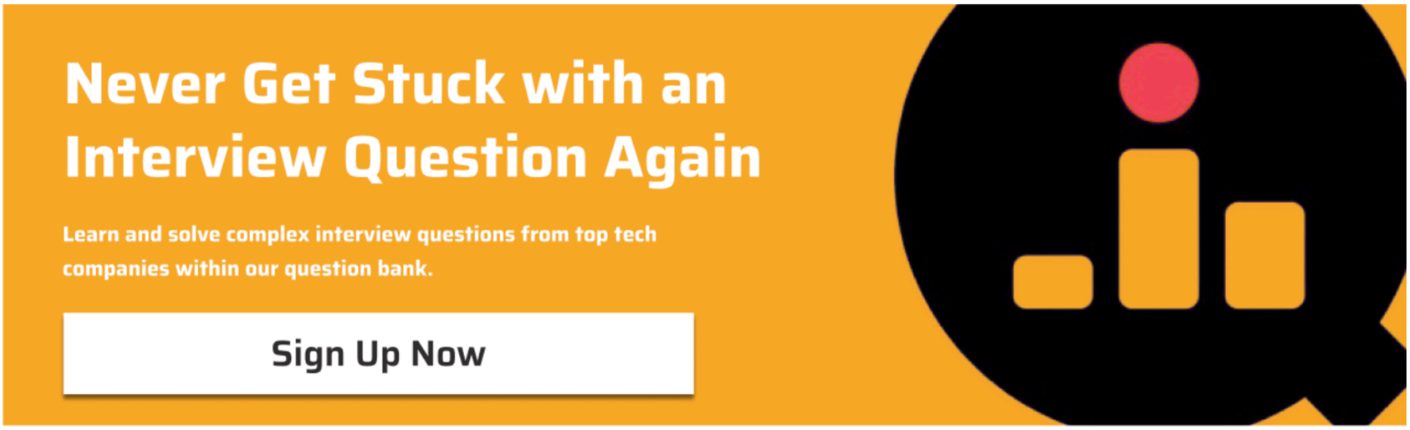
- **Memory consumption** - Python is not great for memory intensive applications.

19. Explain the zip() and enumerate() functions.

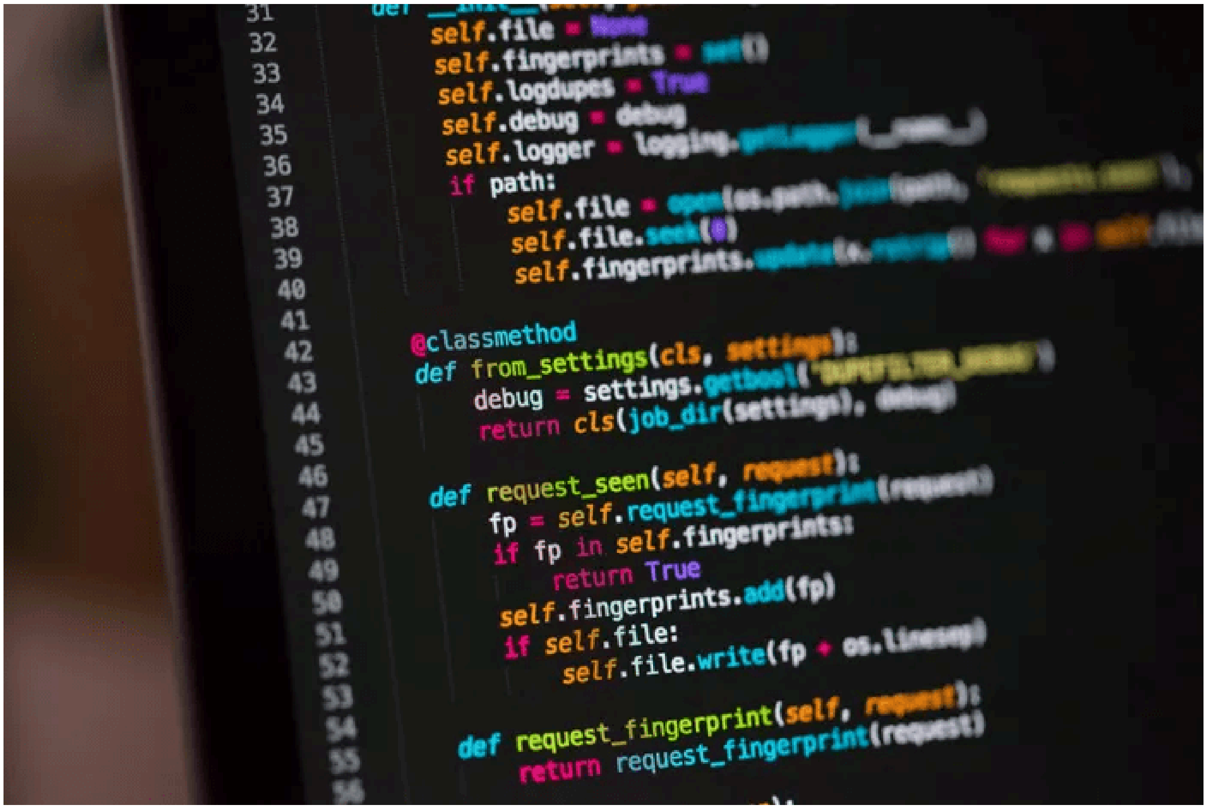
The **enumerate() function** returns the indexes of all items in lists, dictionaries, sets and other iterables. The **zip() function** combines multiple iterables.

20. Define PYTHONPATH.

PYTHONPATH tells Python Interpreter where to locate module files imported into a program. The role is similar to PATH. PYTHONPATH includes both the source library directory and the source code directories.



String Manipulation Python Interview Questions



String parsing questions in Python data science interviews are probably one of the most common. These types of questions focus on how well you can manipulate text data, which always needs to be thoroughly cleaned and transformed into a dataset.

These types of questions are common for companies that process a lot of text like Twitter, LinkedIn, Indeed or Netflix.

21. Write a function that can take a string and return a list of bigrams.

Example:

```
sentence = """
Have free hours and love children?
"""
```

```
output = [('have', 'free'),
('free', 'hours'),
('hours', 'and'),
('and', 'love'),
('love', 'children?')]
```

When separating a sentence into bigrams, the first thing we need to do is split the sentence into individual words. We would need to loop through each word of the sentence and append bigrams to the list. How many loops would we need, for instance, if the amount of words in a sentence was equal to k?

22. Given two strings A and B, return whether or not A can be shifted some number of times to get B.

Example:

```
A = 'abcde'
B = 'cdeab'
can_shift(A, B) == True
A = 'abc'
B = 'acb'
can_shift(A, B) == False
```

This problem is relatively simple if we work out the underlying algorithm that allows us to easily check for string shifts between the strings A and B. First off, we have to set baseline conditions for string shifting. Strings A and B must both be the same length and consist of the same letters. We can check for the former by setting a condition statement when the length of A and B are equivalent.

23. Given two strings, string1 and string2, determine if there exists a one to one character mapping between each character of string1 to string2.

Example:

```
string1 = 'qwe'
string2 = 'asd'
string_map(string1, string2) == True
#q = a, w = s, and e = d
```

Note: This example would return False if the letters were repeated; for example, string1 = ‘donut’ and string2 =’fatty’. This is because the letter t from `fatty` attempts to map to two different outcomes (t = n or t = u)..

24. Given a string, return the first recurring character in it, or “None” if there is no recurring character.

Example:

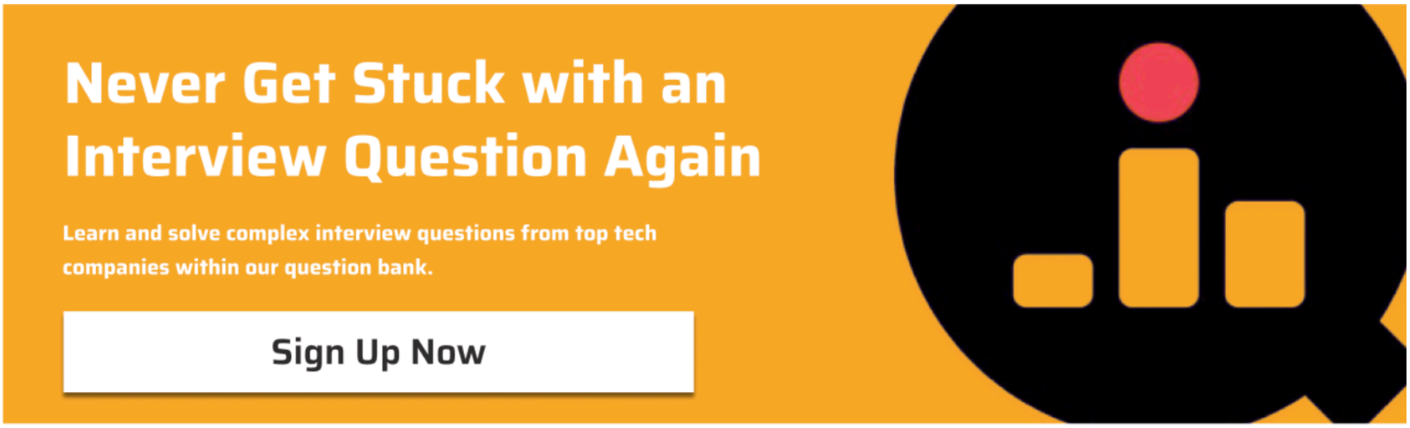
```
input = "interviewquery"
output = "i"
```

Given that we have to return the first index of the second repeating character, we should be able to go through the string in one loop, save each unique character, and then just check if the character exists in that saved set. If it does, return the character.

25. Given two strings, string1 and string2, write a function is_subsequence to find out if string1 is a subsequence of string2.

Hint: Notice that in the subsequence problem set, one string in this problem will need to be traversed to check for the values of the other string. In this case, it is string2.

The idea to solve this should then be simple. We traverse both strings from one side to the other side going from leftmost to rightmost. If we find a matching character, we move ahead in both strings. Otherwise, we move ahead only in string2.


Never Get Stuck with an Interview Question Again
Learn and solve complex interview questions from top tech companies within our question bank.
Sign Up Now

Python Statistics and Probability Interview Questions

Python statistics and probability questions test your ability to translate stats and probability concepts into code. Both types require knowledge of the mathematical concepts, as well as intermediate Python skills.

- **Statistics Python questions** - These questions take the form of random sampling from a distribution, generating histograms and computing different statistical metrics such as standard deviation, mean or median.
- **Probability Python questions** - Probability questions typically focus on concepts like Binomial or Bayes Theorem. Since most probability questions are focused on calculating chances based on a condition, almost all questions can be proven by writing Python code.

26. Write a function to generate N samples from a normal distribution and plot them on the histogram.

This is a relatively simple Python problem that requires setting up a distribution and then generating and plotting n samples from it. We can do this with the SciPy library for scientific computing.

First, **declare a standard normal distribution**, e.g. mean=0 and standard deviation = 1. Then we generate samples through the **rvs(n) function.**

27. Write a function that takes in a list of dictionaries with both a key and list of integers, and returns a dictionary with the standard deviation of each list.

```
input = [
    {
        'key': 'list1',
        'values': [4,5,2,3,4,5,2,3],
    },
    {
        'key': 'list2',
        'values': [1,1,34,12,40,3,9,7],
    }
]
```

```
output = {'list1': 1.12, 'list2': 14.19}
```

Hint: Remember the equation for standard deviation. To be able to fulfill this function, we need to use the equation, where we take the sum of the square of the data value minus the mean, over the total number of data points, all within a square root.

$$\sigma = \sqrt{\frac{\sum(x_i - \mu)^2}{N}}$$

28. Given a list of stock prices in ascending order by datetime, write a function that outputs the max profit by buying and selling at a specific interval.

```
stock_prices = [10,5,20,32,25,12]
dts = [
    '2019-01-01',
    '2019-01-02',
    '2019-01-03',
    '2019-01-04',
    '2019-01-05',
    '2019-01-06',
]
```

```
def max_profit(stock_prices,dts) → 27
```

There are many ways you could go about solving this problem. A good first step is thinking about what our goal is: if we want the maximum profit, then ideally we want to buy at the lowest possible price and sell at the highest possible price. However, since we cannot go back in time, we have a constraint that our sell date must be after our buy date.

29. Amy and Brad take turns rolling a fair six-sided die. Whoever rolls a "6" first, wins the game. Amy starts by rolling first.

What's the probability that Amy wins on her first roll? Let's play out the scenario. If she loses, then Brad must lose his first roll for Amy to have a chance to win again.

You know the probability that Amy wins on her first roll is ⅙. What is then the probability of Amy winning on the 3rd roll? 5th roll?

30. Write a function to simulate the overlap of two computing jobs and output an estimated cost.

More context. Every night between 7 p.m. and midnight, two computing jobs from two different sources are randomly started, with each job lasting an hour. When the jobs run simultaneously at any point in their computations, they cause a failure in some of the company's other nightly jobs, resulting in downtime for the company that costs $1,000.

The CEO needs a single number representing the annual (365 days) cost of this problem.

Hint. We can model this scenario by implementing two random number generators across a spectrum of 0 to 300 minutes, modeling the time in minutes between 7 p.m. and midnight.

Python Pandas Interview Questions

While Pandas has many roles in data science, like analytics-type questions, in most Python interviews, Pandas Interview questions are related to data cleaning. These questions include on-hot encoding variables, using the Pandas apply() function to group different variables, and text cleaning different columns.

31. Given a dataset of test scores, write Pandas code to return cumulative bucketed scores of <50, <75, <90, <100.

```
def bucket_test_scores(df):
    bins = [0, 50, 75, 90, 100]
    labels=['<50','<75','<90' , '<100']
    df['test score'] = pd.cut(df['test score'], bins,labels=labels)
```

32. Given two dataframes (one with addresses and the other with various cities and states), write a function to create a single dataframe with complete addresses.

Hint. In this question, we are given a dataframe full of addresses (in the form of strings) and asked to interpolate state names (more strings) into those addresses.

We will need to match our state names with the cities that they contain. That is going to require us to perform a simple merge of our two dataframes. But before we can do that, we need to split df_addresses such that we can isolate the city part of the address to use in our merge.

33. Given a dataframe of students' favorite colors and test scores, write a function to select only those rows (students) where their favorite color is green or red and their test grade is above 90.

We need to filter our dataframe by two conditions: grade and favorite color. We can filter our dataframe by grade by setting our dataframe equal to itself with the condition that the grade column is greater than 90:

```
students_df = students_df[students_df["grade"] > 90]
```

Now we want to do the same process for favorite color, but the problem is that we have two possible categories for inclusion in the filtered dataframe. How can we write code to include both possibilities in our final dataframe?

34. Given a dataframe with rainfall data (day of the week and rainfall inches), write a function to find the median amount of rainfall for the days on which it rained.

There are two steps to solve the problem:

- **Step 1.** Remove all days with no rain.
- **Step 2.** Calculate the attribute median of the dataframe.

35. You are given a dataframe with prices of cheeses. However, the dataframe is missing values in the price column. Write a function to impute the median price in place of missing values.

This problem uses two built-in Pandas methods.

```
dataframe.column.median()
```

This returns the median of a column in a dataframe.

```
dataframe.column.fillna('value')
```

This applies value to all NaN values in a given column.

36. Write a function that returns the maximum number in the list.

Given a list of integers, write a function that returns the maximum number in the list. If the list is empty, return `None`.

Example 1:

Input:

```
nums = [1, 7, 3, 5, 6]
```

Output:

```
find_max(nums) → 7
```

Example 2:

Input:

```
nums = []
```

Output:

```
find_max(nums) → None
```

Python Data Manipulation Interview Questions

Data manipulation questions are a common type of Python data engineer interview question. They cover techniques that would be transforming data outside of NumPy or Pandas. This is common when designing ETLs, when transforming data between raw json and database reads.

Many times these types of transformations will require grouping, sorting or filtering data using lists, dictionaries and other Python data structure types. These questions test your general knowledge of Python data munging outside of actual Pandas formatting.

37. Given a list of timestamps in sequential order, return a list of lists grouped by week (seven days) using the first timestamp as the starting point.

This question sounds like it should be a SQL question, doesn't it? Weekly aggregation implies a form of GROUP BY in a regular SQL or Pandas question. In either case, aggregation on a dataset of this form by week would be pretty trivial.

But as a scripting question, this task is trying to pry out if the candidate is comfortable dealing with unstructured data, as data scientists may be forced to deal with a lot of unstructured data depending on their specific role or company.

In this function, we have to do a few things:

1. Loop through all of the datetimes.
2. Set a beginning timestamp as our reference point.
3. Check if the next timestamp in the array is more than seven days ahead. a. If so, set the new timestamp as the reference point. b. If not, continue to loop through and append the last value.

38. Given a dictionary consisting of many roots and a sentence, stem all the words in the sentence with the root forming it.

This Python question explores the concept of stemming, which is the heuristic of chopping off the end of a word to clean and bucket it into an easier feature set.

Input:

```
roots = ["cat", "bat", "rat"]
sentence = "the cattle was rattled by the battery"
```

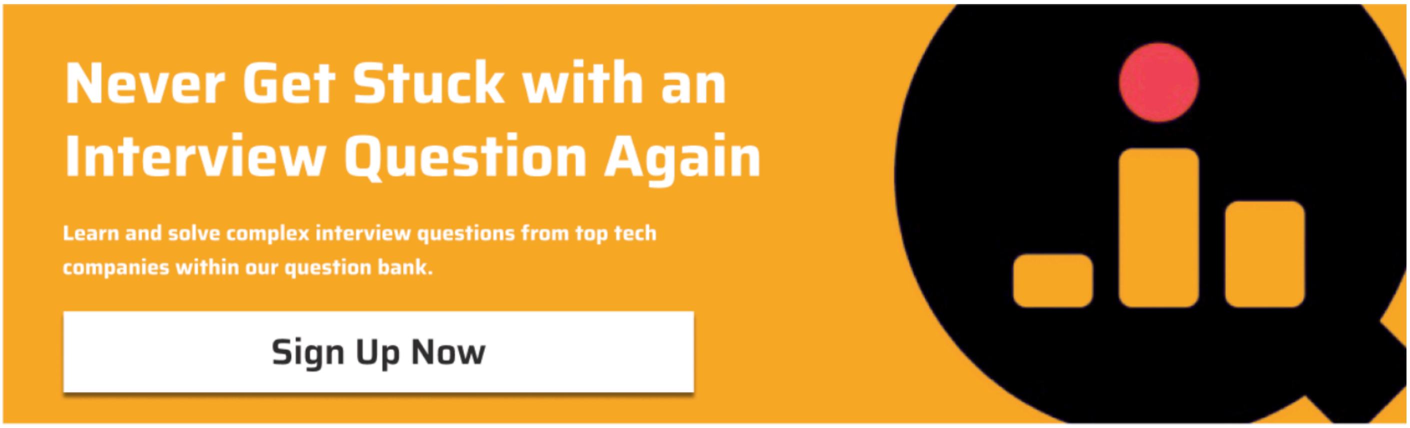
Output:

```
"the cat was rat by the bat"
```

39. Given two dictionaries (friends_added and friends_removed), write a function to list the pairs of friends with corresponding beginning and ending timestamps.

Hint. You are only looking for friendships that have an end date. Because of this, every friendship that will be in our final output is contained within the friends_removed list.

If you start by iterating through the friends_removed dictionary, you will already have the id pair and the end date of each listing in our final output. Next, you just need to find the corresponding start date for each end date.


Never Get Stuck with an Interview Question Again
Learn and solve complex interview questions from top tech companies within our question bank.
Sign Up Now

Matrices and NumPy Python Interview Questions

Many data science problems deal with working with the NumPy library and matrices. Matrices and NumPy interview questions are not as common as the others but still show up, especially for specialized roles like in computer vision interviews. This involves working with the NumPy library to run matrix multiplication, calculating the Jacobian determinant, and transforming matrices in some way or form.

40. What is NumPy used for? What are its benefits?

NumPy is an open-source library that is used to analyze data, and includes support for Python's multi-dimensional arrays and matrices. NumPy is used for a variety of mathematical and statistical operations.

41. Compute the inverse of a matrix in NumPy.

You can find the inverse of any square matrix with the **numpy.linalg.inv(array) function**. In this case, the 'array' would be the matrix to be inverted.

42. Write a function to return a 5-by-5 matrix that contains the portion of employees employed in each department compared to the total number of employees at each company.

More context. Let's say we have a five-by-five matrix num_employees where each row is a company and each column represents a department. Each cell of the matrix displays the number of employees working in that particular department at each company.

To reconstruct the new array, loop through every cell in a department and divide by the total number of employees of the whole company, which is the sum of the whole row.

43. Given an array filled with random values, write a function rotate_matrix to rotate the array by 90 degrees in the clockwise direction.

There are two approaches to this problem. The first would be to analyze how exactly a 90-degree clockwise rotation changes the index of each entry in the matrix. The second is to think of a series of simpler matrix transformations that amount to a 90-degree clockwise rotation when performed in succession.

Python Data Structures and Algorithms Interview Questions

Python data structures interview questions assess your ability to use Python coding in algorithms. In general there are two types of questions, algorithmic coding problems and writing algorithms from scratch.

44. Write a function shortest_transformation to find the length of the shortest transformation sequence from begin_word to end_word through the elements of word_list.

Input:

```
begin_word = "same",
end_word = "cost",
word_list = ["same","came","case","cast","lost","last","cost"]
```

Output:

```
def shortest_transformation(begin_word, end_word, word_list) → 5
```

Since the transformation sequence would be:

```
'same' → 'came' → 'case' → 'cast' → 'cost'
```

Generally, shortest path algorithms require the solution to recursively try every possible matching path from the start to the end.

In this question, we have a few constraints.

1. Every word in word_list is of the same length.

2. The max difference between two words in the path is only one letter change.

45. Given a dictionary with keys of letters and values of a list of letters, write a function closest_key to find the key with the input value closest to the beginning of the list.

Input:

```
dictionary = {
    'a' : ['b','c','e'],
    'm' : ['c','e'],
}
input = 'c'
```

Output:

```
closest_key(dictionary, input) → 'm'
```

With this question, ask: Is your computed distance always positive? Negative values for distance (for example, between ‘c’ and ‘a’ instead of ‘a’ and ‘c’) will interfere with getting an accurate result.

46. Given two strings, string1 and string2, write a function max_substring to return the maximal substring shared by both strings.

Input:

```
string1 = 'mississippi'

string2 = 'mossyistheapple'
```

The idea is that we need to try every matching substring of string1 and string2. So, for example, if we have string1 = abbc, string2 = acc, we can take the first letter of string1, a, and look for a match in string2. Once we find one, we are left with the same problem with a smaller portion of the two strings. The remaining part of string1 will be bbc and string2 cc, and we repeat the process.

- In the second iteration, we don’t find a match _b_bc with cc.
- In the third iteration, we don’t find a match b_b_c with cc.
- Finally, we have a match bb_c_ with _c_c.
- We finished string1, and the result is ac.

Python Machine Learning Interview Questions

Python machine learning questions tend to focus on model deployment and model building, and, in particular, assess your ability to use Python coding in algorithms. In general, there are two types of questions: algorithmic coding problems and writing algorithms from scratch.

47. Develop a k-means clustering algorithm in Python from the ground up

You are provided with:

- A two-dimensional NumPy array `data_points` consisting of an arbitrary number of data points (rows) `n` and an arbitrary number of columns `m`.
- The number of clusters, `k`.
- The initial centroids value for the data points in each cluster, `initial_centroids`.

Return a list of the cluster to which each point belongs in the original list data_points, maintaining the same order (as an integer).

Example

```
#Input
data_points = [(0,0),(3,4),(4,4),(1,0),(0,1),(4,3)]
k = 2
initial_centroids = [(1,1),(4,5)]


#Output
```

```
k_means_clustering(data_points,k,initial_centroids) → [0,1,1,0,0,1]
```

48. Build a K-nearest neighbors classification model from scratch with the following conditions:

- Use Euclidean distance (aka, the *“2 norm”*) as your closeness metric.
- Your function should be able to handle data frames of arbitrarily many rows and columns.
- If there is a tie in the class of the k nearest neighbors, rerun the search using k-1 neighbors instead.
- You may use pandas and numpy but *NOT* scikit-learn.

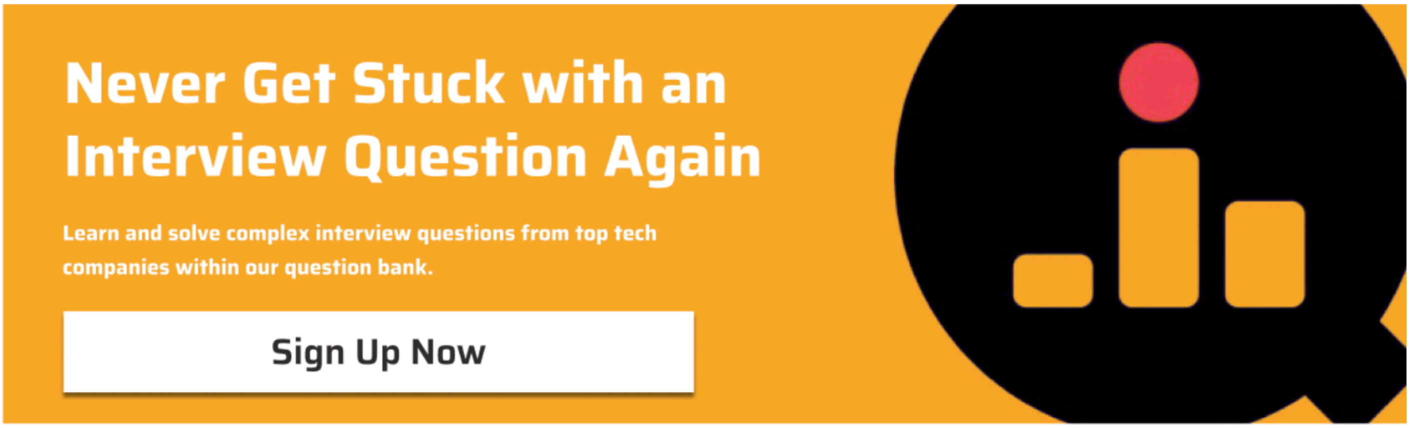
Example Output:

```
def kNN(k,data,new_point) → 2
```

49. Build a random forest model from scratch.

The model should have these conditions:

- The model takes as input a dataframe df and an array `new_point` with a length equal to the number of fields in the df.
- All values of both df and `new_point` are 0 or 1, i.e., all fields are dummy variables, and there are only two classes.
- Rather than randomly deciding what subspace of the data each tree in the forest will use like usual, make your forest out of decision trees that go through every permutation of the value columns of the data frame and split the data according to the value seen in `new_point` for that column.
- Return the majority vote on the class of new_point.
- You may use pandas and NumPy, but *NOT* scikit-learn.

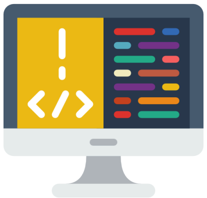


Learn more about Python Interview Questions

This course is designed to help you learn everything you need to know about working with data, from basic concepts to more advanced techniques.

(Python Interview) Learning Path

You only have **6 lessons** remaining in **Python Questions: Hard** course. Currently you are on **Max Width**.



Max Width

Given an array of `words` and a `max_width` parameter, write a function `justify` to format the text such that each line has exactly `max_width` characters. Pad extra spaces ' ' when necessary so that each line has exactly `max_width` characters.

Extra spaces between words should be distributed as evenly as possible. If the number of spaces on a line does not divide evenly between words, place excess spaces on the right-hand side of each line.

Note: You may assume that there is no word in `words` that is longer than `max_width`.

Continue Lesson

Example:

Input:

hide remaining sections in this course

```
words = ["This", "is", "an", "example", "of", "text", "justification."]
max_width = 16

def justify(words, max_width): ->
[
   "This    is    an",
   "example  of text",
   "justification.  "
]
```

□ Check Normality (Question)

□ **Pool Matching** (Question)

More Python Learning Resources

Continue your prep with an Interview Query. We offer a variety of Python resources, including:

- **500+ Real Data Science Interview Questions**
- **Data Science Learning Path**
- **Data Science Challenges**
- **Data Science Take-Homes**
- **Python Learning Path**

Streamlining your recruitment process for Python-savvy data science roles? Let OutSearch.ai's AI-driven platform help you find candidates who not only excel in Python but are perfect for your team's dynamic. Consider checking out the site!

Related articles

From Capital One to New Heights: Jayandra Lade's Data Science Journey

October Data Science Job Market Report (2024)

Navigating the New Age of Interviews: A Data Scientist's Experience

From Finance to Data Engineering: Hanna Lee's Success Story

How to Create a Stand Out Data Analyst Portfolio (2024 Guide)