Congratulations! You passed!

Grade received 100%

Latest Submission Grade 100%

To pass 80% or higher

Go to next item

1. What is one important reason why exploratory analysis is useful in Data Science?

1 / 1 point

○ It allows for more efficient loading in Pandas

○ It can help improve performance in production deployment pipelines

● It can help identify potential sources of bias and error in data, which can lead to more accurate models.

✓ **Correct**
Correct! Identifying pitfalls in data can allow you to correct and filter bad data to produce better models.

2. What is the correct syntax to load a CSV file from the following web url using Pandas: http://example.com/test.csv

1 / 1 point

○
```
import pandas as pd
df = pd.read_url("http://example.com/test.csv")
```

○
```
from pandas import load_csv
df = load_csv("http://example.com/test.csv")
```

●
```
import pandas as pd
df = pd.read_csv("http://example.com/test.csv")
```

✓ **Correct**
Correct!

3. What is the right syntax to replace every newline character from a data frame object in Pandas? **1 / 1 point**

○ `df.remove({"\n": " "}, regex=True)`

● `df.replace({"\n": " "}, regex=True)`

○ `df.remove_newline()`

✓ **Correct**
Correct!

4. What is one option you can use to make changes in a dataframe without having to re-assign the variable. The examples below will drop a `"value"` column. **1 / 1 point**

○ `df.drop(['value'], axis=1)`

○ `df.drop(['value'], axis=0, inplace=True)`

● `df.drop(['value'], axis=1, inplace=True)`

✓ **Correct**
Correct!

5. What is the correct syntax to sort the `cost` column in descending order? **1 / 1 point**

○ `df.sort_values(by="cost", descending=True)`

○ `df.sort_columns(by="cost", ascending=False)`

● `df.sort_values(by="cost", ascending=False)`

✓ **Correct**
Correct!

6. What are two Pandas methods that you can use to quickly get information about a data frame object? **1 / 1 point**

○
1. `head()`
2. `sample()`

○
1. `review()`
2. `expand()`

●
1. `describe()`
2. `info()`

✓ **Correct**
Correct!

7. What is the right syntax to apply the `normalize()` function to the `"numbers"` column in Pandas? **1 / 1 point**

● `df['numbers'].apply(normalize)`

○ `df['numbers'].apply_function(normalize)`

○ `df['numbers'].apply.normalize()`

✓ **Correct**
Correct!

8. How can you get all integers in a NumPy array divided by 5? **1 / 1 point**

○ `array.apply_division(5)`

● `array / 5`

○ `array = np.array([i/5 for i in array])`

✓ **Correct**
Correct!

9. What is a valid operation for reshaping an array created with **1 / 1 point**

```
array = np.ones((3, 4))
```

● `array.reshape((6,2))`

○ `array.reshape((6,3))`

○ `array.reshape((2,5))`

✓ **Correct**
Correct!

10. What would be the right way to get all items in an array that are bigger than 15? **1 / 1 point**

○ `array.apply(newaxis, 15)`

○ `array.shape(>15)`

● `array > 15`

✓ **Correct**
Correct!