adhish-203-lab9

September 16, 2023

Adhish Bahl

2347203

Python Lab 9

Q1. Write a program to distinguish between Array Indexing and Fancy Indexing.

[85]:
```
import numpy as np

arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])
print("Original Array: \n", arr)

print("\nArray Indexing:")
print("Printing value for arr[2][0]: ", arr[2][0])

print("\nFancy Indexing:")
print("Printing value for arr[[1, 2], [2,1]]: ", arr[[1, 2], :])
print("Printing value for arr[[1, 2], [2,1]]: ", arr[[1, 2], [2, 1]])
```

```
Original Array:
 [[1 2 3]
 [4 5 6]
 [7 8 9]]

Array Indexing:
Printing value for arr[2][0]:  7

Fancy Indexing:
Printing value for arr[[1, 2], [2,1]]:  [[4 5 6]
 [7 8 9]]
Printing value for arr[[1, 2], [2,1]]:  [6 8]
```

Q2. Execute the 2D array Slicing.

[86]:
```
arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

print("\nPrinting value for arr[2, 1:]: ",arr[2, 1:])
print("\nPrinting value for arr[1, 1:2]: ",arr[1, 1:2])
print("\nPrinting value for arr[0, :3]: ",arr[0, :3])
```

```
print("\nPrinting value for arr[0:2, 2:6]:\n",arr[0:2, 2:6])
```

```
Printing value for arr[2, 1:]:  [8 9]

Printing value for arr[1, 1:2]:  [5]

Printing value for arr[0, :3]:  [1 2 3]

Printing value for arr[0:2, 2:6]:
 [[3]
 [6]]
```

Q3. Create the 5-Dimensional arrays using 'ndmin'.

[87]:
```
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9], ndmin=5)

print("Printing 5-D Array: ", arr)
print("Number of Dimensions:", arr.ndim)
```

```
Printing 5-D Array:  [[[[[1 2 3 4 5 6 7 8 9]]]]]
Number of Dimensions: 5
```

Q4. Reshape the array from 1-D to 2-D array.

[88]:
```
arr = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12])
newArr = arr.reshape(4, 3)
print("Printing 2-D array created using 1-D array:\n", newArr)
```

```
Printing 2-D array created using 1-D array:
 [[ 1  2  3]
 [ 4  5  6]
 [ 7  8  9]
 [10 11 12]]
```

Q5. Perform the Stack functions in Numpy arrays – Stack(), hstack(), vstack(), and dstack().

[89]:
```
a = np.array([1, 2, 3])
b = np.array([4, 5, 6])

print("Array a: ", a)
print("Array b: ", b)

c = np.stack((a, b))
print("\nPerforming stack operation on a and b:\n", c)

d = np.vstack((a, b))
print("\nPerforming vstack operation on a and b:\n", d)
```

```
e = np.hstack((a, b))
print("\nPerforming hstack operation on a and b:\n", e)

f = np.array([[1, 2], [3, 4]])
g = np.array([[5, 6], [7, 8]])

print("\nArray f:\n", f)
print("\nArray g:\n", g)

h = np.dstack((f, g))
print("\nPerforming dstack operation on f and g:\n", h)
```

```
Array a:  [1 2 3]
Array b:  [4 5 6]

Performing stack operation on a and b:
 [[1 2 3]
 [4 5 6]]

Performing vstack operation on a and b:
 [[1 2 3]
 [4 5 6]]

Performing hstack operation on a and b:
 [1 2 3 4 5 6]

Array f:
 [[1 2]
 [3 4]]

Array g:
 [[5 6]
 [7 8]]

Performing dstack operation on f and g:
 [[[1 5]
  [2 6]]

 [[3 7]
  [4 8]]]
```

Q6. Perform the searchsort method in Numpy array.

[90]:
```
a = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9])
value = 5

index = np.searchsorted(a, value)
```

```
print("Index value for", value, "is :", index)
```

```
Index value for 5 is : 4
```

Q7. Create Numpy Structured array using your domain features.

[91]:
```
sneakers = np.array([("Nike", "Air Jordan 1 Retro High OG", 17000),
                     ("Adidas", "Yeezy Boost 350 V2", 22000),
                     ("Converse", "Chuck Taylor All Star '70", 85000)],
                    dtype=[("brand", "U10"), ("model", "U30"), ("price", "i4")])

print("The structured array is as follows ==>\n", sneakers)
print("\nElements can be accessed as follows ==>\nValue for sneakers[1]: ",␣
 ↪sneakers[1])
print("Value for sneakers[0][1]: ", sneakers[0][1])
```

```
The structured array is as follows ==>
 [('Nike', 'Air Jordan 1 Retro High OG', 17000)
 ('Adidas', 'Yeezy Boost 350 V2', 22000)
 ('Converse', "Chuck Taylor All Star '70", 85000)]

Elements can be accessed as follows ==>
Value for sneakers[1]:  ('Adidas', 'Yeezy Boost 350 V2', 22000)
Value for sneakers[0][1]:  Air Jordan 1 Retro High OG
```

Q8. Create Data frame using List and Dictionary.

[92]:
```
import pandas as pd

sneakers = [
    {"brand": "Nike", "model": "Air Jordan 1 Retro High OG", "price": 17000},
    {"brand": "Adidas", "model": "Yeezy Boost 350 V2", "price": 22000},
    {"brand": "Converse", "model": "Chuck Taylor All Star '70", "price": 85000},
    {"brand": "Nike", "model": "Air Max 90", "price": 12000},
    {"brand": "Puma", "model": "Clyde All-Pro", "price": 11000}
]

print("Printing the ist of Dictioanry:\n", sneakers)

df = pd.DataFrame(sneakers)

print("\nPrinting Dataframe created using List and Dictionary:\n", df)
```

```
Printing the ist of Dictioanry:
 [{'brand': 'Nike', 'model': 'Air Jordan 1 Retro High OG', 'price': 17000},
{'brand': 'Adidas', 'model': 'Yeezy Boost 350 V2', 'price': 22000}, {'brand':
'Converse', 'model': "Chuck Taylor All Star '70", 'price': 85000}, {'brand':
'Nike', 'model': 'Air Max 90', 'price': 12000}, {'brand': 'Puma', 'model':
```

```
'Clyde All-Pro', 'price': 11000}]

Printing Dataframe created using List and Dictionary:
       brand                        model  price
0       Nike  Air Jordan 1 Retro High OG  17000
1     Adidas           Yeezy Boost 350 V2  22000
2  Converse   Chuck Taylor All Star '70  85000
3       Nike                  Air Max 90  12000
4       Puma               Clyde All-Pro  11000
```

Q9. Create Data frame on your Domain area and perform the following operations to find and eliminate the missing data from the dataset.

1. isnull()
2. notnull()
3. dropna()
4. fillna()
5. replace()
6. interpolate()

[93]:
```
sneakers = [
    {"brand": "Nike", "model": "Air Jordan 1 Retro High OG", "price": 17000},
    {"brand": "Adidas", "model": "Yeezy Boost 350 V2", "price": np.nan},
    {"brand": "Converse", "model": np.nan, "price": 85000},
    {"brand": np.nan, "model": "Air Max 90", "price": 12000},
    {"brand": "Puma", "model": "Clyde All-Pro", "price": 11000}
]

df = pd.DataFrame(sneakers)

print("Printing NULL values:\n", df.isnull())

print("\nPrinting not NULL values:\n", df.notnull())

df1 = pd.DataFrame(sneakers)
df1.dropna(inplace=True)
print("\nPrinting dataset after performing df1.dropna(inplace=True):\n", df1)

df.fillna(value={"model": "Unknown", "price": 0}, inplace=True)
print("\nPrinting dataset after performing df.fillna(value={\"model\":␣
 ↪\"Unknown\", \"price\": 0}, inplace=True):\n", df)

df.replace(to_replace=np.nan, value={"brand": "Unknown"}, inplace=True)
print("\nPrinting dataset after performing df.replace(to_replace=np.nan,␣
 ↪value={\"brand\": \"Unknown\"}, inplace=True):\n", df)

df.interpolate(method="linear", inplace=True)
```

```
print("\nPrinting dataset after performing df.interpolate(method=\"linear\",␣
 ↪inplace=True):\n", df)
```

```
Printing NULL values:
    brand  model  price
0  False  False  False
1  False  False   True
2  False   True  False
3   True  False  False
4  False  False  False

Printing not NULL values:
    brand  model  price
0   True   True   True
1   True   True  False
2   True  False   True
3  False   True   True
4   True   True   True

Printing dataset after performing df1.dropna(inplace=True):
   brand                       model    price
0  Nike  Air Jordan 1 Retro High OG  17000.0
4  Puma               Clyde All-Pro  11000.0

Printing dataset after performing df.fillna(value={"model": "Unknown", "price":
0}, inplace=True):
       brand                       model    price
0      Nike  Air Jordan 1 Retro High OG  17000.0
1    Adidas          Yeezy Boost 350 V2      0.0
2  Converse                     Unknown  85000.0
3       NaN                  Air Max 90  12000.0
4      Puma               Clyde All-Pro  11000.0

Printing dataset after performing df.replace(to_replace=np.nan, value={"brand":
"Unknown"}, inplace=True):
       brand                       model    price
0      Nike  Air Jordan 1 Retro High OG  17000.0
1    Adidas          Yeezy Boost 350 V2      0.0
2  Converse                     Unknown  85000.0
3   Unknown                  Air Max 90  12000.0
4      Puma               Clyde All-Pro  11000.0

Printing dataset after performing df.interpolate(method="linear", inplace=True):
       brand                       model    price
0      Nike  Air Jordan 1 Retro High OG  17000.0
1    Adidas          Yeezy Boost 350 V2      0.0
2  Converse                     Unknown  85000.0
```

```
3   Unknown                  Air Max 90  12000.0
4      Puma               Clyde All-Pro  11000.0
```

```
C:\Users\adhis\AppData\Local\Temp\ipykernel_13632\1349397664.py:25:
FutureWarning: DataFrame.interpolate with object dtype is deprecated and will
raise in a future version. Call obj.infer_objects(copy=False) before
interpolating instead.
  df.interpolate(method="linear", inplace=True)
```

Q10. Perform the Hierarchical Indexing in the above created dataset.

[94]:
```
sneakers = [
    {"brand": "Nike", "model": "Air Jordan 1 Retro High OG", "price": 17000},
    {"brand": "Adidas", "model": "Yeezy Boost 350 V2", "price": 22000},
    {"brand": "Converse", "model": "Chuck Taylor All Star '70", "price": 85000},
    {"brand": "Nike", "model": "Air Max 90", "price": 12000},
    {"brand": "Puma", "model": "Clyde All-Pro", "price": 11000}
]

df = pd.DataFrame(sneakers)

df.set_index(['brand', 'model'], inplace=True)

print("Printing dataframe after setting index: ", df)
```

```
Printing dataframe after setting index:
price
brand    model
Nike     Air Jordan 1 Retro High OG  17000
Adidas   Yeezy Boost 350 V2          22000
Converse Chuck Taylor All Star '70   85000
Nike     Air Max 90                  12000
Puma     Clyde All-Pro               11000
```