manju-236-lab9

September 16, 2023

[3]:
```
#Q1.Array indexing and fancy indexing
import numpy as np

# Array Indexing
arr = np.array([1, 2, 3, 4, 5])
print(arr[0])  # Output: 1
print(arr[1:3])  # Output: [2 3]

# Fancy Indexing
arr = np.array([1, 2, 3, 4, 5])
print(arr[[0, 2]])  # Output: [1 3]
```

```
1
[2 3]
[1 3]
```

[1]:
```
#Q2.2D Array Slicing
import numpy as np

# create a 2D array
arr = np.array([[1, 2, 3], [4, 5, 6], [7, 8, 9]])

# slice the first two rows and the first two columns
slice_arr = arr[:2, :2]

print(slice_arr)
```

```
[[1 2]
 [4 5]]
```

[4]:
```
#Q3.5D array with ndim
import numpy as np

# create a 5D array with shape (1, 2, 3, 4, 5)
arr = np.array([[[[[1, 2, 3, 4, 5],
                   [6, 7, 8, 9, 10],
                   [11, 12, 13, 14, 15]],
                  [[16, 17, 18, 19, 20],
```

```
                    [21, 22, 23, 24, 25],
                    [26, 27, 28, 29, 30]],
                   [[31, 32, 33, 34, 35],
                    [36, 37, 38, 39, 40],
                    [41, 42, 43, 44, 45]],
                   [[46, 47, 48, 49, 50],
                    [51, 52, 53, 54, 55],
                    [56, 57, 58, 59, 60]]]]], ndmin=5)

print(arr)
```

```
[[[[[ 1  2  3  4  5]
    [ 6  7  8  9 10]
    [11 12 13 14 15]]

   [[16 17 18 19 20]
    [21 22 23 24 25]
    [26 27 28 29 30]]

   [[31 32 33 34 35]
    [36 37 38 39 40]
    [41 42 43 44 45]]

   [[46 47 48 49 50]
    [51 52 53 54 55]
    [56 57 58 59 60]]]]]
```

[5]:
```
#Q4. Reshape the array from 1-D to 2-D array.

import numpy as np

# 1-D array
arr = np.array([1, 2, 3, 4, 5, 6])

# Reshape to a 2-D array with 2 rows and 3 columns
new_arr = np.reshape(arr, (2, 3))
# or
# new_arr = arr.reshape((2, 3))

print(new_arr)
```

```
[[1 2 3]
 [4 5 6]]
```

[6]:
```
#Q5. Perform the Stack functions in Numpy arrays - Stack(), hstack(), vstack(),␣
 ↪and dstack().
```

```
import numpy as np

# Create some arrays to stack
arr1 = np.array([1, 2, 3])
arr2 = np.array([4, 5, 6])
arr3 = np.array([7, 8, 9])

# Stack the arrays using different functions
stacked = np.stack((arr1, arr2, arr3))
hstacked = np.hstack((arr1, arr2, arr3))
vstacked = np.vstack((arr1, arr2, arr3))
dstacked = np.dstack((arr1, arr2, arr3))

# Print the stacked arrays
print(stacked)
print(hstacked)
print(vstacked)
print(dstacked)
```

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

[7]:
```
#Q6. Perform the searchsort method in Numpy array.

import numpy as np

# Create a sorted array
arr = np.array([1, 3, 5, 7, 9])

# Search for the index where 6 should be inserted
index = np.searchsorted(arr, 6)

# Insert 6 into the array at the correct index
arr = np.insert(arr, index, 6)

# Print the sorted array with 6 inserted
print(arr)
```

```
[1 3 5 6 7 9]
```

[27]:
```python
#Q7. Create Numpy Structured array using your domain features.

import numpy as np

# Define the structured array data type
pet_dtype = np.dtype([
    ('name', 'U20'),
    ('species', 'U20'),
    ('age', int),
    ('price', float)
])

# Create an empty structured array with space for 10 pets
pet_array = np.empty(10, dtype=pet_dtype)

# Add data to the structured array
pet_array[0] = ('Fluffy', 'Cat', 3, 50.0)
pet_array[1] = ('Rex', 'Dog', 2, 100.0)
pet_array[2] = ('Nibbles', 'Hamster', 1, 10.0)

# Print the structured array
print(pet_array)
```

```
[('Fluffy', 'Cat', 3,  50.) ('Rex', 'Dog', 2, 100.)
 ('Nibbles', 'Hamster', 1,  10.) ('', '', 0,   0.) ('', '', 0,   0.)
 ('', '', 0,   0.) ('', '', 0,   0.) ('', '', 0,   0.) ('', '', 0,   0.)
 ('', '', 0,   0.)]
```

[35]:
```python
#Q8. Create Data frame using List and Dictionary.

import pandas as pd

# Create a list of dictionaries
pets = [
    {'name': 'Fluffy', 'species': 'Cat', 'age': 3, 'price': 50.0},
    {'name': 'Rex', 'species': 'Dog', 'age': 2, 'price': 100.0},
    {'name': 'Nibbles', 'species': 'Hamster', 'age': 1, 'price': 10.0}
]

# Create a DataFrame using pd.DataFrame()
df1 = pd.DataFrame(pets)

# Create a DataFrame using pd.DataFrame.from_dict()
df2 = pd.DataFrame.from_dict(pets)

# Print the resulting DataFrames
print(df1)
```

```
print(df2)
```

```
      name  species  age  price
0   Fluffy      Cat    3   50.0
1      Rex      Dog    2  100.0
2  Nibbles  Hamster    1   10.0
      name  species  age  price
0   Fluffy      Cat    3   50.0
1      Rex      Dog    2  100.0
2  Nibbles  Hamster    1   10.0
```

[36]:
```
#Q9. Create Data frame on your Domain area and perform the following operations␣
 ↪to find and eliminate the  missing data from the dataset.

import pandas as pd
import numpy as np

# Create a dictionary of pets
pets = {
    'name': ['Fluffy', 'Rex', 'Nibbles', np.nan, 'Buddy'],
    'species': ['Cat', 'Dog', 'Hamster', 'Fish', 'Dog'],
    'age': [3, 2, 1, np.nan, 5],
    'price': [50.0, 100.0, 10.0, np.nan, 200.0]
}

# Create a DataFrame from the dictionary
df = pd.DataFrame(pets)

# Check for missing data
print(df.isnull())

# Check for non-missing data
print(df.notnull())

# Drop rows with missing data
df = df.dropna()

# Fill missing data with a value
df = df.fillna(0)

# Replace missing data with a value
df = df.replace(np.nan, 0)

# Interpolate missing data
df = df.interpolate()

# Print the resulting DataFrame
```

```
print(df)
```

```
    name  species    age  price
0  False    False  False  False
1  False    False  False  False
2  False    False  False  False
3   True    False   True   True
4  False    False  False  False
    name  species    age  price
0   True     True   True   True
1   True     True   True   True
2   True     True   True   True
3  False     True  False  False
4   True     True   True   True
      name  species  age  price
0   Fluffy      Cat  3.0   50.0
1      Rex      Dog  2.0  100.0
2  Nibbles  Hamster  1.0   10.0
4    Buddy      Dog  5.0  200.0
```

[37]:
```
#Q10. Perform the Hierarchical Indexing in the above created dataset.

import pandas as pd

# Create a dictionary of pets
pets = {
    'name': ['Fluffy', 'Rex', 'Nibbles', 'Buddy'],
    'species': ['Cat', 'Dog', 'Hamster', 'Dog'],
    'age': [3, 2, 1, 5],
    'price': [50.0, 100.0, 10.0, 200.0]
}

# Create a DataFrame from the dictionary with hierarchical indexing
df = pd.DataFrame(pets, index=[['Kitten', 'Puppy', 'Hamster', 'Dog'], [1, 2, 3,␣
 ↪4]])

# Print the resulting DataFrame with hierarchical indexing
print(df)
```

```
              name  species  age  price
Kitten  1   Fluffy      Cat    3   50.0
Puppy   2      Rex      Dog    2  100.0
Hamster 3  Nibbles  Hamster    1   10.0
Dog     4    Buddy      Dog    5  200.0
```