vishnu-265-lab9

September 16, 2023

Lab Exercise 9

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

[40]:

```
import numpy as np

#array indexing
arr=np.array([2,4,6,8,10,12])
print(arr[3])

#fancy indexing
selected_index = arr[[0,2,3,5]]
print(selected_index)
```

```
8
[ 2  6  8 12]
```

Q2. Execute the 2D array Slicing.

[41]:

```
import numpy as np
arr2d = np.array([[1,3,5,7],[2,4,6,8],[3,6,9,1]])
print("Array Elements :\n",arr2d)
#slicing
print("Slicing a Single element : ",arr2d[0,2])
print("Slicing Second Row : ",arr2d[1])
print("Slicing a Column : ",arr2d[:,1])
print("Third Row last element : ",arr2d[2,-1])
```

```
Array Elements :
 [[1 3 5 7]
 [2 4 6 8]
 [3 6 9 1]]
Slicing a Single element :  5
Slicing Second Row :  [2 4 6 8]
Slicing a Column :  [3 4 6]
Third Row last element :  1
```

Q3. Create the 5-Dimensional arrays using ‘ndmin’.

[42]:

```
import numpy as np
arr = np.array([10,20,30,40,50], ndmin=5)

print(arr)
print('Number of dimensions:', arr.ndim)
```

```
[[[[[10 20 30 40 50]]]]]
Number of dimensions: 5
```

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

[43]:

```
import numpy as np
arr=np.array([1,3,5,7,8,10])
print("1D array : ",arr)
new_arr = arr.reshape(2,3)
print("2D array : \n",new_arr)
```

```
1D array :  [ 1  3  5  7  8 10]
2D array :
 [[ 1  3  5]
 [ 7  8 10]]
```

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

[44]:

```
import numpy as np

arr1 = np.array([[10, 20], [30, 40]])
arr2 = np.array([[50, 60], [70, 80]])

stacked_arr = np.stack((arr1, arr2), axis=0)
hstacked_arr = np.hstack((arr1, arr2))
vstacked_arr = np.vstack((arr1, arr2))
dstacked_arr = np.dstack((arr1, arr2))

print('Original arrays:\n', arr1, '\n', arr2)
print('Stacked array:\n', stacked_arr)
print('Horizontally stacked array:\n', hstacked_arr)
print('Vertically stacked array:\n', vstacked_arr)
print('Depth-wise stacked array:\n', dstacked_arr)
```

```
Original arrays:
 [[10 20]
 [30 40]]
 [[50 60]
 [70 80]]
Stacked array:
 [[[10 20]
  [30 40]]
 [[50 60]
  [70 80]]]
Horizontally stacked array:
 [[10 20 50 60]
 [30 40 70 80]]
Vertically stacked array:
 [[10 20]
 [30 40]
 [50 60]
 [70 80]]
Depth-wise stacked array:
 [[[10 50]
  [20 60]]

 [[30 70]
  [40 80]]]
```

Q6. Perform the searchsort method in Numpy array.

[45]:

```python
import numpy as np

arr = np.array([45,85,23,92,103,76])
arr = np.sort(arr)
num = 56

index = np.searchsorted(arr, num)
print(arr)
print(f"Value {num} should be inserted at index {index}.")
```

```
[ 23  45  76  85  92 103]
Value 56 should be inserted at index 2.
```

Q7. Create Numpy Structured array using your domain features.

[46]:

```python
import numpy as np

# Define the data types for the structured array
dtype = [
    ('medication_id', 'int32'),
    ('name', 'U50'),           # U50 represents a Unicode string with up to 50␣
 ↪characters
    ('manufacturer', 'U50'),
    ('quantity', 'int32'),
    ('unit_price', 'float64'),
    ('expiration_date', 'datetime64[D]')  # Assuming expiration dates are␣
 ↪stored as date only
]

medication_data = np.array([
    (1, 'Aspirin', 'Bayer', 100, 0.5, '2023-12-31'),
    (2, 'Ibuprofen', 'Generic', 50, 0.3, '2024-06-30'),
    (3, 'Amoxicillin', 'Pfizer', 75, 1.2, '2023-11-15'),
    (4, 'Lisinopril', 'Novartis', 30, 0.8, '2024-04-20')
], dtype=dtype)

print(medication_data)
```

```
[(1, 'Aspirin', 'Bayer', 100, 0.5, '2023-12-31')
 (2, 'Ibuprofen', 'Generic',  50, 0.3, '2024-06-30')
 (3, 'Amoxicillin', 'Pfizer',  75, 1.2, '2023-11-15')
 (4, 'Lisinopril', 'Novartis',  30, 0.8, '2024-04-20')]
```

Q8. Create Data frame using List and Dictionary.

[47]:

```
import pandas as pd

# Create a list of dictionaries representing medication data
medication_data = [
    {'medication_id': 1, 'name': 'Aspirin', 'manufacturer': 'Bayer', 'quantity':
 ↪ 100, 'expiration_date': '2023-12-31'},
    {'medication_id': 2, 'name': 'Ibuprofen', 'manufacturer': 'Generic',␣
 ↪'quantity': 50,  'expiration_date': '2024-06-30'},
    {'medication_id': 3, 'name': 'Amoxicillin', 'manufacturer': 'Pfizer',␣
 ↪'quantity': 75,  'expiration_date': '2023-11-15'},
    {'medication_id': 4, 'name': 'Lisinopril', 'manufacturer': 'Novartis',␣
 ↪'quantity': 30, 'expiration_date': '2024-04-20'}
]

# Create a DataFrame from the list of dictionaries
df = pd.DataFrame(medication_data)

# Display the DataFrame
print(df)
```

```
   medication_id         name manufacturer  quantity expiration_date
0              1      Aspirin        Bayer       100      2023-12-31
1              2    Ibuprofen      Generic        50      2024-06-30
2              3  Amoxicillin       Pfizer        75      2023-11-15
3              4   Lisinopril     Novartis        30      2024-04-20
```

Q9. Create Data frame on your Domain area and perform the following operations to find and eliminate the missing data from the dataset. • isnull() • notnull() • dropna() • fillna() • replace() • interpolate()

[48]:

```
import pandas as pd
import numpy as np
```

```python
data = {
    'medication_id': [1, 2, 3, 4, 5],
    'name': ['Aspirin', 'Ibuprofen', 'Amoxicillin', 'Lisinopril', 'Paracetamol'],
    'quantity': [100, np.nan, 75, 30, np.nan],
    'unit_price': [0.5, 0.3, np.nan, 0.8, 1.0],
    'expiration_date': ['2023-12-31', '2024-06-30', None, '2024-04-20', '2023-09-30']
}

df = pd.DataFrame(data)

missing_data = df.isnull()
print("Missing Data:")
print(missing_data)

non_missing_data = df.notnull()
print("\nNon-Missing Data:")
print(non_missing_data)

df_dropped = df.dropna()
print("\nDataFrame after dropping rows with missing values:")
print(df_dropped)

mean_quantity = df['quantity'].mean()
df_filled = df.fillna({'quantity': mean_quantity})
print("\nDataFrame after filling missing quantity values:")
print(df_filled)

df_replaced = df.replace({'name': {'Paracetamol': 'Acetaminophen'}})
print("\nDataFrame after replacing 'Paracetamol' with 'Acetaminophen':")
print(df_replaced)

df_interpolated = df.interpolate()
print("\nDataFrame after linear interpolation for missing unit_price values:")
print(df_interpolated)
```

```
Missing Data:
   medication_id   name  quantity  unit_price  expiration_date
0          False  False     False       False            False
1          False  False      True       False            False
2          False  False     False        True             True
3          False  False     False       False            False
4          False  False      True       False            False

Non-Missing Data:
   medication_id  name  quantity  unit_price  expiration_date
```

```
0           True  True      True        True              True
1           True  True     False        True              True
2           True  True      True       False             False
3           True  True      True        True              True
4           True  True     False        True              True

DataFrame after dropping rows with missing values:
   medication_id        name  quantity  unit_price expiration_date
0              1     Aspirin     100.0         0.5      2023-12-31
3              4  Lisinopril      30.0         0.8      2024-04-20

DataFrame after filling missing quantity values:
   medication_id         name    quantity  unit_price expiration_date
0              1      Aspirin  100.000000         0.5      2023-12-31
1              2    Ibuprofen   68.333333         0.3      2024-06-30
2              3  Amoxicillin   75.000000         NaN            None
3              4   Lisinopril   30.000000         0.8      2024-04-20
4              5  Paracetamol   68.333333         1.0      2023-09-30

DataFrame after replacing 'Paracetamol' with 'Acetaminophen':
   medication_id           name  quantity  unit_price expiration_date
0              1        Aspirin     100.0         0.5      2023-12-31
1              2      Ibuprofen       NaN         0.3      2024-06-30
2              3    Amoxicillin      75.0         NaN            None
3              4     Lisinopril      30.0         0.8      2024-04-20
4              5  Acetaminophen       NaN         1.0      2023-09-30

DataFrame after linear interpolation for missing unit_price values:
   medication_id         name  quantity  unit_price expiration_date
0              1      Aspirin     100.0        0.50      2023-12-31
1              2    Ibuprofen      87.5        0.30      2024-06-30
2              3  Amoxicillin      75.0        0.55            None
3              4   Lisinopril      30.0        0.80      2024-04-20
4              5  Paracetamol      30.0        1.00      2023-09-30
```

```
C:\Users\91702\AppData\Local\Temp\ipykernel_18380\3099072653.py:35:
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_interpolated = df.interpolate()
```

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

[49]:

```python
# Convert 'quantity' and 'unit_price' columns to float
df['quantity'] = df['quantity'].astype(float)
df['unit_price'] = df['unit_price'].astype(float)

# Set hierarchical index using 'medication_id' and 'name'
```

```
df.set_index(['medication_id', 'name'], inplace=True)

# Display the DataFrame with hierarchical indexing
print(df)
```

```
                           quantity  unit_price expiration_date
medication_id name
1             Aspirin         100.0         0.5      2023-12-31
2             Ibuprofen         NaN         0.3      2024-06-30
3             Amoxicillin      75.0         NaN            None
4             Lisinopril       30.0         0.8      2024-04-20
5             Paracetamol       NaN         1.0      2023-09-30
```