annmary-211-lab9

September 16, 2023

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

[2]:
```
import numpy as np
arr1=np.array([1,2,3,4,5,6,7,8,9])
#Array(Simple) Indexing
ai=arr1[3]
print("Array Indexing: ",ai)
#Fancy Indexing
fi=arr1[[2,4,6,8]]
print("Fancy Indexing: ",fi)
```

```
Array Indexing:  4
Fancy Indexing:  [3 5 7 9]
```

Q2. Execute the 2D array Slicing.

[3]:
```
import numpy as np
arr=np.array([[1,2,3,4,5,6],[7,8,9,10,11,12]])
print(arr[1][1:5])
```

```
[ 8  9 10 11]
```

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

[4]:
```
import numpy as np
arr1=np.array([9,8,7,6],ndmin=5)
print("The 5 dimensional array using ndmin is : ",arr1)
```

```
The 5 dimensional array using ndmin is :  [[[[[9 8 7 6]]]]]
```

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

[5]:
```
import numpy as np
arr=np.array([0,9,8,7,6,5,4,3,2,1])
print("The original array :",arr)
reshapearr=arr.reshape(2,5)
print("Array after reshapeing : \n",reshapearr)
```

```
The original array : [0 9 8 7 6 5 4 3 2 1]
Array after reshapeing :
```

```
[[0 9 8 7 6]
 [5 4 3 2 1]]
```

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

[6]:
```
import numpy as np
arr1=np.array([00,11,22,33,44,55,66,77,88,99])
print("First array : ",arr1)
arr2=np.array([10,20,30,40,50,60,70,80,90,100])
print("Second array : ",arr2)

#Using stack
sarr=np.stack((arr1,arr2))
print("\nArrays after using stack : \n",sarr)

#Using hstack
harr=np.hstack((arr1,arr2))
print("\nArrays after using hstack : \n",harr)

#Using vstack
varray=np.vstack((arr1,arr2))
print("\nArrays after using vstack : \n",varray)

#Using dstack
darray=np.dstack((arr1,arr2))
print("\nArrays after using dstack :\n",darray)
```

```
First array :  [ 0 11 22 33 44 55 66 77 88 99]
Second array :  [ 10  20  30  40  50  60  70  80  90 100]

Arrays after using stack :
 [[  0  11  22  33  44  55  66  77  88  99]
 [ 10  20  30  40  50  60  70  80  90 100]]

Arrays after using hstack :
 [  0  11  22  33  44  55  66  77  88  99  10  20  30  40  50  60  70  80
  90 100]

Arrays after using vstack :
 [[  0  11  22  33  44  55  66  77  88  99]
 [ 10  20  30  40  50  60  70  80  90 100]]

Arrays after using dstack :
 [[[  0  10]
  [ 11  20]
  [ 22  30]
  [ 33  40]
  [ 44  50]
```

```
 [ 55  60]
 [ 66  70]
 [ 77  80]
 [ 88  90]
 [ 99 100]]]
```

Q6. Perform the searchsort method in Numpy array.

[7]:
```
import numpy as np
arr=np.array([12,23,34,45,56,67,78])

sarr=np.searchsorted(arr,[45,56,78,12])
print("Search Sort : ",sarr)

sarray=np.searchsorted(arr,23,side='left')
print("(using side) : ",sarray)
```

```
Search Sort :  [3 4 6 0]
(using side) :  1
```

Q7. Create Numpy Structured array using your domain features.

[10]:
```
import numpy as np

# Define structured array with field names and data types
courses = np.array([
    (1, 'Introduction to Python', 'John Doe', '2023-09-01', '2023-09-30', 50,␣
 ↪49.99),
    (2, 'Web Development Fundamentals', 'Jane Smith', '2023-10-01',␣
 ↪'2023-10-31', 75, 79.99),
    (3, 'Machine Learning Basics', 'Alice Johnson', '2023-11-01', '2023-11-30',␣
 ↪30, 99.99)
], dtype=[
    ('course_id', 'int'),
    ('course_name', 'U50'),
    ('instructor', 'U50'),
    ('start_date', 'datetime64[D]'),  # Use datetime64[D] for date-only format
    ('end_date', 'datetime64[D]'),    # Use datetime64[D] for date-only format
    ('enrollment_count', 'int'),
    ('price', 'float')
])

# Accessing structured array elements
print("Course Names:", courses['course_name'])
print("Instructors:", courses['instructor'])
```

```
Course Names: ['Introduction to Python' 'Web Development Fundamentals'
 'Machine Learning Basics']
Instructors: ['John Doe' 'Jane Smith' 'Alice Johnson']
```

Q8. Create Data frame using List and Dictionary.

[12]:
```
import pandas as pd
#Dataframe using Dictionary
df={
    'Books':['Aarachar','Harry Potter 1','Percy Jackson','Chemmeen'],
    'Year':[2012,1997,2010,1956]
}
mybooks=pd.DataFrame(df)
print(mybooks)

#DataFrame using list
mylist=[2012,1997,2010,1956]
mydata=pd.DataFrame(mylist,index=['a','b','c','d'],
                    columns=['Years'])
print("\n\n",mydata)
```

```
           Books  Year
0        Aarachar  2012
1  Harry Potter 1  1997
2   Percy Jackson  2010
3        Chemmeen  1956


    Years
a   2012
b   1997
c   2010
d   1956
```

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()

[13]:
```
import pandas as pd
import numpy as np

# Create a sample DataFrame
data = {
    'course_id': [1, 2, 3, 4, 5],
    'course_name': ['Python Basics', 'Web Development', 'Machine Learning',␣
 ↪'Data Science', 'SQL Fundamentals'],
    'instructor': ['John Doe', 'Jane Smith', 'Alice Johnson', 'Bob Brown', 'Eva␣
 ↪White'],
    'start_date': ['2023-09-01', None, '2023-10-01', '2023-10-15',␣
 ↪'2023-11-01'],
    'end_date': ['2023-09-30', '2023-11-30', '2023-11-30', None, '2023-12-15'],
    'enrollment_count': [50, 75, None, 60, 40],
```

```
    'price': [49.99, 79.99, 99.99, None, 59.99]
}

df = pd.DataFrame(data)

# Display the initial DataFrame
print("Initial DataFrame:")
print(df)

# Check for missing data
print("\nCheck for missing data (isnull()):")
print(df.isnull())

# Check for non-missing data
print("\nCheck for non-missing data (notnull()):")
print(df.notnull())

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

# Fill missing values with a specific value
df_filled = df.fillna({'start_date': '2023-01-01', 'end_date': '2023-12-31',␣
 ↪'enrollment_count': 0, 'price': 0.0})
print("\nDataFrame after filling missing values:")
print(df_filled)

# Replace missing values with a specific value
df_replaced = df.replace(np.nan, 'N/A')
print("\nDataFrame after replacing missing values:")
print(df_replaced)

# Interpolate missing values
df_interpolated = df.interpolate()
print("\nDataFrame after interpolating missing values:")
print(df_interpolated)
```

```
Initial DataFrame:
   course_id       course_name     instructor  start_date    end_date  \
0          1     Python Basics       John Doe  2023-09-01  2023-09-30
1          2   Web Development     Jane Smith        None  2023-11-30
2          3  Machine Learning  Alice Johnson  2023-10-01  2023-11-30
3          4      Data Science      Bob Brown  2023-10-15        None
4          5  SQL Fundamentals      Eva White  2023-11-01  2023-12-15

   enrollment_count  price
```

```
0              50.0  49.99
1              75.0  79.99
2               NaN  99.99
3              60.0    NaN
4              40.0  59.99

Check for missing data (isnull()):
   course_id  course_name  instructor  start_date  end_date  enrollment_count  \
0      False        False       False       False     False             False
1      False        False       False        True     False             False
2      False        False       False       False     False              True
3      False        False       False       False      True             False
4      False        False       False       False     False             False

   price
0  False
1  False
2  False
3   True
4  False

Check for non-missing data (notnull()):
   course_id  course_name  instructor  start_date  end_date  enrollment_count  \
0       True         True        True        True      True              True
1       True         True        True       False      True              True
2       True         True        True        True      True             False
3       True         True        True        True     False              True
4       True         True        True        True      True              True

   price
0   True
1   True
2   True
3  False
4   True

DataFrame after dropping rows with missing data:
   course_id       course_name instructor  start_date    end_date  \
0          1     Python Basics   John Doe  2023-09-01  2023-09-30
4          5  SQL Fundamentals  Eva White  2023-11-01  2023-12-15

   enrollment_count  price
0              50.0  49.99
4              40.0  59.99

DataFrame after filling missing values:
   course_id       course_name     instructor  start_date    end_date  \
0          1     Python Basics       John Doe  2023-09-01  2023-09-30
```

```
1          2   Web Development     Jane Smith  2023-01-01  2023-11-30
2          3  Machine Learning  Alice Johnson  2023-10-01  2023-11-30
3          4      Data Science      Bob Brown  2023-10-15  2023-12-31
4          5  SQL Fundamentals      Eva White  2023-11-01  2023-12-15

   enrollment_count  price
0              50.0  49.99
1              75.0  79.99
2               0.0  99.99
3              60.0   0.00
4              40.0  59.99

DataFrame after replacing missing values:
   course_id       course_name     instructor  start_date    end_date  \
0          1     Python Basics       John Doe  2023-09-01  2023-09-30
1          2   Web Development     Jane Smith         N/A  2023-11-30
2          3  Machine Learning  Alice Johnson  2023-10-01  2023-11-30
3          4      Data Science      Bob Brown  2023-10-15         N/A
4          5  SQL Fundamentals      Eva White  2023-11-01  2023-12-15

  enrollment_count  price
0             50.0  49.99
1             75.0  79.99
2              N/A  99.99
3             60.0    N/A
4             40.0  59.99

DataFrame after interpolating missing values:
   course_id       course_name     instructor  start_date    end_date  \
0          1     Python Basics       John Doe  2023-09-01  2023-09-30
1          2   Web Development     Jane Smith        None  2023-11-30
2          3  Machine Learning  Alice Johnson  2023-10-01  2023-11-30
3          4      Data Science      Bob Brown  2023-10-15        None
4          5  SQL Fundamentals      Eva White  2023-11-01  2023-12-15

   enrollment_count  price
0              50.0  49.99
1              75.0  79.99
2              67.5  99.99
3              60.0  79.99
4              40.0  59.99
```

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

[14]:
```python
import pandas as pd

# Create a sample DataFrame
data = {
```

```
    'course_id': [1, 2, 3, 4, 5],
    'course_name': ['Python Basics', 'Web Development', 'Machine Learning',␣
 ↪'Data Science', 'SQL Fundamentals'],
    'instructor': ['John Doe', 'Jane Smith', 'Alice Johnson', 'Bob Brown', 'Eva␣
 ↪White'],
    'start_date': ['2023-09-01', None, '2023-10-01', '2023-10-15',␣
 ↪'2023-11-01'],
    'end_date': ['2023-09-30', '2023-11-30', '2023-11-30', None, '2023-12-15'],
    'enrollment_count': [50, 75, None, 60, 40],
    'price': [49.99, 79.99, 99.99, None, 59.99]
}

df = pd.DataFrame(data)

# Set hierarchical index with 'course_id' and 'course_name' as levels
df.set_index(['course_id', 'course_name'], inplace=True)

# Display the DataFrame with hierarchical indexing
print("DataFrame with Hierarchical Indexing:")
print(df)
```

```
DataFrame with Hierarchical Indexing:
                             instructor  start_date    end_date  \
course_id course_name
1         Python Basics         John Doe  2023-09-01  2023-09-30
2         Web Development     Jane Smith        None  2023-11-30
3         Machine Learning  Alice Johnson  2023-10-01  2023-11-30
4         Data Science         Bob Brown  2023-10-15        None
5         SQL Fundamentals      Eva White  2023-11-01  2023-12-15

                           enrollment_count  price
course_id course_name
1         Python Basics                50.0  49.99
2         Web Development              75.0  79.99
3         Machine Learning              NaN  99.99
4         Data Science                 60.0    NaN
5         SQL Fundamentals             40.0  59.99
```

[ ]: