TRIBHUVAN UNIVERSITY

INSTITUTE OF SCIENCE AND TECHNOLOGY

HIMALAYA COLLEGE OF ENGINEERING

CHYASAL,LALITPUR

Lab Report No:- 7

Title:- Dataframes manipulation & Operations in python using pandas.

Submitted by:-

Name:- Surajan Shrestha.

Roll No:- 38

Submitted To:- Rubas Mali

Department Of CSIT

Checked by:-

Date of submission:- 2081/06/04

TITLE: Dataframe Manipulation and operations in python using pandas.

OBJECTIVE:

↳ The objective of this lab is to demonstrate how to perform key operations on Dataframes in python using the pandas library. we will work with a dataset containing information about birds, including their species, age, number of visits, and priority and perform various operations such as indexing, slicing, filtering and modifying data in Dataframes.

THEORY:

→ Pandas is a powerful python library used for data manipulation and analysis. It provides data structures like series and Dataframe for handling and analyzing data effectively. Dataframe is a two dimensional, size-mutable and heterogeneous tabular data structure with labeled axes (rows & columns).

Csv (comma separated values) is a widely-used format for storing data. pandas can read csv files into a Dataframes using the read_csv() function. Dataframes allows users to load, manipulate, and analyze data in tabular format.

The iloc[] function in pandas is used for accessing data based on integer-location-based indexing. It helps to retrieve specific rows & columns from a Dataframe by their index and column positions.

with pandas, various operations can be performed on Dataframes such as data filtering, indexing, Statistical analysis, data aggregation and handling missing values.

OBSERVATIONS:

1, CSV files & Dataframes:

→ To work with CSV files, you can load them into a pandas Dataframe using read_csv().

Example:
```
import pandas as pd
df = pd.read_csv('example.csv')
print(df.head())
```

2, Dataframes Creation from Dictionary:

```
→ data = {'Name': ['surajan', 'milan', 'sid'],
          Age: [22, 42, 16],
          'city': ['New york', 'paris', 'nepal']}
  df = pd.Dataframe(data)
  print(df)
```

3, iloc[]:

```
→ import pandas as pd
  data = {'Name': ['surajan', 'milan', 'sid'],
          Age: [22, 42, 16],
          'city': ['Nepal', 'paris', 'Japan']}
  df = pd.Dataframe(data)
  print(df.iloc[0])
  print(df.iloc[[1,2], [0,1]])
```

lab Tasks:

```
/* codes */
import pandas as pd
import numpy as np
# Data provided
data = {'birds':['cranes','cranes','plovers','spoonbills',
        'spoonbills','cranes','plovers','cranes','spoonbills',
        'spoonbills'],
    'age':[3.5, 4, 1.5, np.nan, 6, 3, 5.5, np.nan, 8, 4],
    'visits':[2, 4, 3, 4, 3, 4, 2, 2, 3, 2],
    'priority':['yes','yes','no','yes','no','no','no','yes',
                'no','no']}
    labels = ['a','b','c','d','e','f','g','h','i','J']
```

1. create a Dataframes birds from this dictionary data which has the index labels.

```
birds = pd.Dataframe(data, index=labels)
```

OUTPUT:

```
     birds        age    visits    priority
a    Cranes       3.5    2         yes
b    cranes       4.0    4         yes
c    plovers      1.5    3         no
d    spoon bills  NaN    4         yes
e    spoon bills  6.0    3         no
f    cranes       3.0    4         no
g    plovers      5.5    2         no
h    cranes       NaN    2         yes
i    Spoonbills   8.0    3         no
J    spoonbills   4.0    2         no
```

#2. Display a summary of the basic information about birds Dataframe and its data.

→
```
summary = birds.info()
print (summary)
```

OUTPUT:

```
<class 'pandas.core.frame.Dataframe'>
 Index: 10 entries, a to J
 Data Columns (total 4 columns):
  #      Column      Non-Null  Count      Dtype
 ---     -----      -------------        -----
  0      birds       10  non-null         object
  1      age          8  non-null         float64
  2      visits      10  non-null         int64
  3      priority    10  non-null         object
 dtypes: float64(1),  int64 (1),  object (2)
 memory usage : 400.0+ bytes
 None
```

#3. print the first 2 rows of the birds dataframe.

→ ~~birds~~ = first_two_rows = birds.head(2)
```
print ( first_two_rows )
```

OUTPUT:

```
     birds     age     visits   priority
a    Cranes    3.5       2       yes
b    cranes    4.0       4       yes
```

#4. print all the rows with only 'birds' and 'age' columns from the data frame.

```
→ birds_age = birds [['birds', 'age']]
  print (birds_age)
```

OUTPUT:

```
        birds        age
a      Cranes        3.5
b      cranes        4.0
c      plovers       1.5
d      Spoonbills    NaN
e      Spoonbills    6.0
f      cranes        3.0
g      plovers       5.5
h      cranes        NaN
i      spoonbills    8.0
j      Spoonbills    4.0
```

#5. Select [2,3,7] rows and in columns ['birds', 'age', 'visits']

```
→ selected_rows = birds.iloc[[2,3,7],[0,1,2]]
```

#6. select the rows where the number of visits is less than 4

```
→ visits_less_than_4 = birds[birds['visits']<4]
```

#7. select the rows with columns ['birds', 'visits'] where the age is missing i.e NaN

```
→ missing_age = birds[birds['age'].isna()][[
  'birds', 'visits']]
```

#8. Select the rows where the birds is a Cranes & the age is less than 4.

→ Cranes_age_less_than_4 = birds [(birds['birds'] == 'cranes') & (birds ['age']<4)]

#9. select the rows the age is between 2 and 4 (inclusive)

→ age_between_2_and_4 = birds [birds ['age']. betwee (2,4)]

#10. find the total number of visits of the bird cranes.

→ total_visits_cranes = birds [birds['birds'] == 'Cranes'] ['visits'].sum ()

#11. calculate the mean age for each different birds in dataframe.

→ mean_age_per_bird = birds.groupby('birds') ['age'] mean()

#12. Append a new row 'k' to dataframe with your choice of values for each column. Then delete that row to return original Dataframe.

→ birds.loc ['k'] = ['swans', 5, 3, 'no']
birds.drop ('k', inplace = True)

#13. find the number of each type of birds in dataframe (counts)

→ bird_counts = birds['birds'].value_Counts()

#14. Sort dataframe (birds) first by values in the 'age' in descending order, then by value in the 'visits' column in ascending order.

→ sorted_birds = birds.sort_values (by= ['age', 'visits'], ascending = [false, True])

15. Replace the priority column values with 'yes' should be 1 and 'no' should be 0

→ birds['priority'] = birds ['priority'].replace ({'yes':1, 'no'= 0})

#16. In the 'birds' column, change the 'cranes' entries to 'trumpeters'.

→ birds ['birds']=birds['birds'].replace('cranes', 'trumpeters')

DISCUSSION:

→ In this lab, we performed various Dataframe operations using pandas, starting with creating a Dataframe from a dictionary and assigning the custom index labels. we explored data selection, filtering and slicing with iloc[], handled missing data and conducted operations like summing the entries, group-by for calculating means and appending/deleting rows. we also sorted Dataframes by multiple columns and replaced categorical values with binary ones. These operations are essential for flexible data manipulation & analysis in real-world scenarios.

Check

Conclusion:

→ This lab covered key pandas Dataframe operations like indexing, filtering, and modifying data. we learned how to efficiently manipulate and analyze tabular data, handle missing values and aggregate information. pandas offers versatile tools for managing real-world data with ease.