prabhudayala@gmail.com_op2

July 24, 2019

Consider the following Python dictionary data and Python list labels:

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']

[27]:
```
import pandas as pd
import numpy as np
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 DataFrame birds from this dictionary data which has the index labels.

[28]:
```
df=pd.DataFrame(data,index=labels)
df
```

[28]:

	birds	age	visits	priority
a	Cranes	3.5	2	yes
b	Cranes	4.0	4	yes
c	plovers	1.5	3	no
d	spoonbills	NaN	4	yes
e	spoonbills	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.

[29]:
```
df.describe()
```

[29]:

	age	visits
count	8.000000	10.000000
mean	4.437500	2.900000

```
std     2.007797   0.875595
min     1.500000   2.000000
25%     3.375000   2.000000
50%     4.000000   3.000000
75%     5.625000   3.750000
max     8.000000   4.000000
```

3. Print the first 2 rows of the birds dataframe.

[30]:
```
df.head(2)
```

[30]:
```
    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 dataframe

[31]:
```
print(df[['birds','age']].to_string(index=False))
```

```
      birds  age
     Cranes  3.5
     Cranes  4.0
    plovers  1.5
 spoonbills  NaN
 spoonbills  6.0
     Cranes  3.0
    plovers  5.5
     Cranes  NaN
 spoonbills  8.0
 spoonbills  4.0
```

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

[32]:
```
print(df[['birds','age','visits']].iloc[[2,3,7]])
```

```
        birds  age  visits
c     plovers  1.5       3
d  spoonbills  NaN       4
h      Cranes  NaN       2
```

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

[33]:
```
df[df['visits']<4]
```

[33]:
```
         birds  age  visits priority
a      Cranes  3.5       2      yes
c     plovers  1.5       3       no
e  spoonbills  6.0       3       no
g     plovers  5.5       2       no
h      Cranes  NaN       2      yes
i  spoonbills  8.0       3       no
j  spoonbills  4.0       2       no
```

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

[34]:
```
df[['birds','visits']][df['age'].isnull()]
```

[34]:
```
         birds  visits
d  spoonbills       4
h      Cranes       2
```

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

[35]:
```
df[(df['birds']=='Cranes') & (df['age']<4)]
```

[35]:
```
    birds  age  visits priority
a  Cranes  3.5       2      yes
f  Cranes  3.0       4       no
```

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

[36]:
```
df[(df['age']>=2) & (df['age']<=4)]
```

[36]:
```
         birds  age  visits priority
a      Cranes  3.5       2      yes
b      Cranes  4.0       4      yes
f      Cranes  3.0       4       no
j  spoonbills  4.0       2       no
```

10. Find the total number of visits of the bird Cranes

[37]:
```
df['visits'][df['birds']=='Cranes'].sum()
```

[37]:
```
12
```

11. Calculate the mean age for each different birds in dataframe.

[38]:
```
df.groupby('birds')['age'].mean()
```

[38]:
```
birds
Cranes        3.5
plovers       3.5
spoonbills    6.0
Name: age, dtype: float64
```

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

[42]:
```
print("original")
print(df)
data_temp = {'birds': 'Cranes1',
        'age': 3.5,
        'visits': 2,
        'priority': 'yes'}
df_temp111=pd.DataFrame(data_temp, index=['k'])
df=df.append(df_temp111)
print("after adding k")
print(df)
df=df.drop('k')
print("after dropping k")
print(df)
```

```
original
        birds  age  visits priority
a      Cranes  3.5       2      yes
b      Cranes  4.0       4      yes
c     plovers  1.5       3       no
d  spoonbills  NaN       4      yes
e  spoonbills  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
after adding k
        birds  age  visits priority
a      Cranes  3.5       2      yes
b      Cranes  4.0       4      yes
c     plovers  1.5       3       no
d  spoonbills  NaN       4      yes
e  spoonbills  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
k     Cranes1  3.5       2      yes
after dropping k
        birds  age  visits priority
a      Cranes  3.5       2      yes
b      Cranes  4.0       4      yes
c     plovers  1.5       3       no
d  spoonbills  NaN       4      yes
e  spoonbills  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
```

13. Find the number of each type of birds in dataframe (Counts)

[43]:
```
df['birds'].value_counts()
```

[43]:
```
spoonbills    4
Cranes        4
plovers       2
Name: birds, dtype: int64
```

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

[44]:
```
df.sort_values(by = ['age', 'visits'], ascending = [False, True])
```

[44]:

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

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

[45]:
```
def sample(x):
    if x=='no':
        return 0
    else:
        return 1
df.priority=df.priority.apply(sample)
df
```

[45]:

	birds	age	visits	priority
a	Cranes	3.5	2	1
b	Cranes	4.0	4	1
c	plovers	1.5	3	0
d	spoonbills	NaN	4	1
e	spoonbills	6.0	3	0
f	Cranes	3.0	4	0
g	plovers	5.5	2	0
h	Cranes	NaN	2	1
i	spoonbills	8.0	3	0
j	spoonbills	4.0	2	0

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

[46]:
```
def sample1(x):
    if x=='Cranes':
        return 'trumpeters'
    else:
        return x
df.birds=df.birds.apply(sample1)
df
```

[46]:

	birds	age	visits	priority
a	trumpeters	3.5	2	1
b	trumpeters	4.0	4	1
c	plovers	1.5	3	0
d	spoonbills	NaN	4	1

```
e  spoonbills  6.0       3         0
f  trumpeters  3.0       4         0
g     plovers  5.5       2         0
h  trumpeters  NaN       2         1
i  spoonbills  8.0       3         0
j  spoonbills  4.0       2         0
```