pandas_basics_practice

November 21, 2020

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']

1. Create a DataFrame birds from this dictionary data which has the index labels.

[1]:

```
import pandas as pd
import numpy as np
data = {
    'birds':␣
 ↪['Cranes','Cranes','plovers','spoonbills','spoonbills','Cranes','plovers','Cranes','spoonbil
    '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']
df = pd.DataFrame(data,index=labels)
print(df)
```

```
        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.

[2]:

```
print(df.describe())
```

```
             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

```
[3]: print(df.head(2))
```

```
    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

```
[4]: print(df[['birds','age']])
```

```
        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']

```
[5]: 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

```
[6]: print(df[df['visits'] < 4])
```

```
        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

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

```
        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

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

```
    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)

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

```
        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

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

```
12
```

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

```
[11]: print(df.groupby('birds', as_index=False)['age'].mean() )
```

```
        birds  age
0      Cranes  3.5
1     plovers  3.5
2  spoonbills  6.0
```

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.

```
[12]: print("Original Dataframe : ")
      print(df,"\n")
      df.loc['k'] = ['parrot',2.5,3,'yes']
```

```python
print("Dataframe After appending a new row 'k' : ")
print(df,"\n")
df=df.drop('k')
print("Dataframe After deleting row 'k' : ")
print(df,"\n")
```

```
Original Dataframe :
        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

Dataframe After appending a new row '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      parrot  2.5       3      yes

Dataframe After deleting row '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)

[13]:

```
print(df.groupby('birds')['birds'].count())
```

```
birds
Cranes        4
plovers       2
spoonbills    4
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.

[14]:

```
print("Before sorting")
print(df)
df=df.sort_values(by=['age', 'visits'], ascending=[False, True])
print("After sorting")
print(df)
```

```
Before sorting
        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 sorting
        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

[15]:

```
print("Before replacing")
print(df)
df['priority']=df['priority'].map({'yes':1, 'no':0})
print("After replacing")
print(df)
```

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

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

```
[16]: df['birds'] = df['birds'].map(lambda x: 'trumpeters' if x=='Cranes' else x)
      print(df)
```

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