pandas_basics_practice

June 11, 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]:
```
import pandas as pd
import numpy as np
```

[2]:
```
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.

[3]:
```
df = pd.DataFrame.from_dict(data)
df['labels'] = labels
df = df.set_index('labels')
df.head()
```

[3]:

labels	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

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

[4]:
```
df.describe(), df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 10 entries, a to j
Data columns (total 4 columns):
birds       10 non-null object
age         8 non-null float64
```

```
visits      10 non-null int64
priority    10 non-null object
dtypes: float64(1), int64(1), object(2)
memory usage: 400.0+ bytes
```

[4]:
```
(            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, None)
```

3. Print the first 2 rows of the birds dataframe

[5]:
```
df.head(2)
```

[5]:
```
         birds  age  visits priority
labels
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

[6]:
```
df[['age', 'birds']]
```

[6]:
```
        age       birds
labels
a       3.5      Cranes
b       4.0      Cranes
c       1.5     plovers
d       NaN  spoonbills
e       6.0  spoonbills
f       3.0      Cranes
g       5.5     plovers
h       NaN      Cranes
i       8.0  spoonbills
j       4.0  spoonbills
```

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

[7]:
```
df[['birds', 'age', 'visits']].iloc[[2, 3, 7]]
```

[7]:
```
             birds  age  visits
labels
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

[8]:
```
df[df['visits'] < 4]
```

[8]:
```
            birds  age  visits priority
labels
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

[9]:
```
indexes = df.isna()['age']

df[indexes][['birds', 'visits']]
```

[9]:
```
            birds  visits
labels
d       spoonbills       4
h           Cranes       2
```

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

[10]:
```
df.loc[(df['birds'] == 'Cranes') & (df['age'] < 4)]
```

[10]:
```
         birds  age  visits priority
labels
a       Cranes  3.5       2      yes
f       Cranes  3.0       4       no
```

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

[11]:
```
df.loc[df['age'].isin([2,3,4])]
```

[11]:
```
            birds  age  visits priority
labels
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

[12]:
```
df.loc[df['birds']== 'Cranes'].visits.sum()
```

[12]:
```
12
```

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

[13]:
```
df.groupby('birds').age.mean()
```

[13]:
```
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.

[14]:
```
data = {'birds' : ['penguin'], 'age' : [4], 'visits' : [5], 'priority' :␣
↪['yes'], 'labels' : ['k']}
df1 = pd.DataFrame.from_dict(data)
df1 = df1.set_index('labels')

df = df.append([df1])
df.loc['k'] = np.nan # filter based on index or unique ids, set as nan
df.dropna(inplace=True)
df.tail()
```

[14]:

labels	birds	age	visits	priority
e	spoonbills	6.0	3.0	no
f	Cranes	3.0	4.0	no
g	plovers	5.5	2.0	no
i	spoonbills	8.0	3.0	no
j	spoonbills	4.0	2.0	no

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

[15]:
```
df.birds.value_counts()
```

[15]:
```
spoonbills    3
Cranes        3
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.

[16]:
```
df.sort_values(['age'], ascending=False).sort_values(['visits'])
```

[16]:

labels	birds	age	visits	priority
g	plovers	5.5	2.0	no
j	spoonbills	4.0	2.0	no
a	Cranes	3.5	2.0	yes
i	spoonbills	8.0	3.0	no
e	spoonbills	6.0	3.0	no
c	plovers	1.5	3.0	no
b	Cranes	4.0	4.0	yes
f	Cranes	3.0	4.0	no

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

[17]:
```
df['priority'] = df['priority'].astype('category')
df['priority'] = df['priority'].cat.codes
df.head()
```

[17]:

labels	birds	age	visits	priority
a	Cranes	3.5	2.0	1

```
b           Cranes  4.0     4.0         1
c          plovers  1.5     3.0         0
e       spoonbills  6.0     3.0         0
f           Cranes  3.0     4.0         0
```

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

[18]:
```
indexes = df['birds'].isin(['Cranes'])
df['birds'][indexes] = 'trumpeters'
df
```

```
/usr/local/lib/python3.6/dist-packages/ipykernel_launcher.py:2:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
```

[18]:
```
             birds  age  visits  priority
labels
a       trumpeters  3.5     2.0         1
b       trumpeters  4.0     4.0         1
c          plovers  1.5     3.0         0
e       spoonbills  6.0     3.0         0
f       trumpeters  3.0     4.0         0
g          plovers  5.5     2.0         0
i       spoonbills  8.0     3.0         0
j       spoonbills  4.0     2.0         0
```