session2_pandas_solutions

September 23, 2019

1 [MIRACUM 2019][Session 2] Solutions

- Kim Hee (Graduate research assistant)
- Universitätsmedizin Mannheim, Mannheim (UMM)

- This is prepared for a turorial `Data analysis tools (Datenanalysewerkzeuge)`
- Task 1 - 13 are taken from: 10 minutes to pandas. A tutorial in depth is availalbe in the Cookbook

1.1 Prerequisite

- import python packages
- create some dataframe objects

[ ]:
```
import pandas as pd
pd.set_option('display.max_columns', 999)
import numpy as np
import matplotlib.pyplot as plt
```

[ ]:
```
# Creating a DataFrame by passing a array, with a datetime index and labeled␣
↪columns:
dates = pd.date_range('20130101', periods=6)
dates
```

[ ]:
```
df = pd.DataFrame(np.random.randn(6, 4), index=dates, columns=list('ABCD'))
df
```

Creating a DataFrame by passing a dict of objects that can be converted to series-like.

[ ]:
```
df2 = pd.DataFrame({'A': 1.,
                    'B': pd.Timestamp('20130102'),
                    'C': pd.Series(1, index=list(range(4)), dtype='float32'),
                    'D': np.array([3] * 4, dtype='int32'),
                    'E': pd.Categorical(["test", "train", "test", "train"]),
                    'F': 'foo'})
df2
```

1.2 Viewing data

1.2.1 Task 1. print the data types of `df` and `df2`

- hint: `.dtypes`

[ ]:
```
df.dtypes
```

[ ]:
```
df2.dtypes
```

1.2.2 Task 2. shows a quick statistic summary of `df` and `df2`

- hint: `.describe()`

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

[ ]:
```
df2.describe()
```

1.2.3 Task 3. view the top and bottom row of `df`

- hints: `.head(1)` and `.tail(1)`

[ ]:
```
df.head(1)
```

[ ]:
```
df.tail(1)
```

Task 4. display the index and columns of `df`
* hints: `.index` and `.columns`

[ ]:
```
df.index
```

[ ]:
```
df.columns
```

1.2.4 Task 5. sort by value on a single column `A` of `df`

- hint: `.sort_values(by='COLUMN', ascending=True|False)`

[ ]:
```
df.sort_values(by='A', ascending=True)
```

1.3 Selection

1.3.1 Task 6. selecting a single column `A` and `B`

- hint: `df['COLUMN']`

[ ]:
```
df[['A','B']]
```

1.3.2 Task 7. slices by index date from 20130102 to 20130104

- hint: `df['INDEX_BEGINNIG_VALUE':'INDEX_ENDING_VALUE']`

[ ]:
```
df['20130102':'20130104']
```

1.3.3 Task 8. selection by label: slice `df` where date from 20130102 to 20130104 and columns of A and B

- hint: `.loc['INDEX_BEGINNIG_VALUE':'INDEX_ENDING_VALUE', ['COLUMN_1', 'COLUMN_2']]`

[ ]:
```
df.loc['20130102':'20130104', ['A', 'B']]
```

1.3.4 Task 9. selection by position: slice `df` from the first to third rows and columns

- hint: `.iloc[ROW_BEGINNING_POSITION:ROW_ENDING_POSITION, COLUMN_BEGINNING_POSITION:COLUMN_ENDING_POSITION]`

[ ]:
```
df.iloc[0:3, 0:3]
```

1.4 Merge

1.4.1 Task 10. double the `df` (concatenate two dataframes)

- hint: `pd.concat([DF_1, DF_2])`

[ ]:
```
pd.concat([df, df])
```

1.4.2 Task 11. join two dataframes below and save the result to a new dataframe `df3`

- hint: 'pd.merge(DF_1, DF_2, on='KEY')

[ ]:
```
left = pd.DataFrame({'A': ['foo', 'bar', 'foo', 'bar'],
                     'B': ['one', 'one', 'two', 'two']})
left
```

[ ]:
```
right = pd.DataFrame({'A': ['foo', 'bar', 'foo', 'bar', 'foo', 'foo'],
                      'C': np.random.randn(6),
                      'D': np.random.randn(6)})
right
```

[ ]:
```
df3 = pd.merge(left, right, on='A')
df3
```

1.5 Grouping

1.5.1 Task 12. Grouping `df` by `A` and `B` and then applying the sum() function

- hint: `.groupby('COLUMN').sum()`

[ ]:
```
df3.groupby(['A', 'B']).sum()
```

1.6 Plotting

1.6.1 Task 13. plot `df` with labels

- hint: `.plot()`

[ ]:
```
df.plot()
```

1.7 Data I/O

1.7.1 Task 14. read a csv file `resources/johnsnow_pumps.csv`

- hint: `pd.read_csv('FILE_PATH')`

[ ]:
```
pd.read_csv('resources/johnsnow_pumps.csv')
```

1.7.2 Task 15. create a database driver object and make a database connection

[ ]:
```
import pandas.io.sql as psql
import psycopg2 # database driver for PostgreSQL
DB_IP = "172.31.0.2"
conn = psycopg2.connect(f"postgres://postgres:postgres@{DB_IP}:5432/mimic")
```

1.7.3 Task 16. show all public tables in PostgreSQL

- hint: `SELECT * FROM pg_catalog.pg_tables WHERE schemaname = 'public'`

[ ]:
```
psql.read_sql("SELECT * \
                FROM pg_catalog.pg_tables \
               WHERE schemaname = 'public'", conn)
```

1.7.4 Task 17. print the first five records from admissions table

[ ]:
```
a = psql.read_sql("SELECT * FROM admissions", conn)
a.head()
```

2 Repeat the tasks 1 - 13 to admissions table

2.0.1 Task 18. select admissions table where insurance is Private

[ ]:
```
a[a['insurance'] == 'Private']
```

2.0.2 Task 19. sort admissions table by admittime

[ ]:
```
a.sort_values(by = 'admittime')
```

2.0.3 Task 20. group admissions table by marital_status and then applying the size() function

[ ]:
```
a.groupby(['marital_status']).size()
```

2.0.4 Task 21. plot the output of Task 20 to a pie chart

[ ]:
```
a.groupby(['marital_status']).size().plot(kind="pie")
```

2.0.5 Task 21. group admissions table by admission_type and then applying the size() function

[ ]:
```
a.groupby(['admission_type']).size()
```

2.0.6 Task 22. plot the output of Task 21 to a horizontal bar chart

[ ]:
```
a.groupby(['admission_type']).size().plot(kind="barh")
```

2.0.7 Task 23. Join `admissions` table and `patients` table based on `subject_id` column

```
[ ]: a = psql.read_sql("SELECT * FROM admissions", conn)
     p = psql.read_sql("SELECT * FROM patients", conn)
     ap = pd.merge(a, p, on = 'subject_id' , how = 'inner')
```

2.0.8 Task 24. group the joined table by `admission_type` and `gender` and then applying the `size()` function

```
[ ]: ap.groupby(['admission_type','gender']).size()
```

2.0.9 Task 25. reshape the output of `Task 24`

- hint: apply `.unstack()` function

```
[ ]: ap.groupby(['admission_type','gender']).size().unstack()
```

2.0.10 Task 26. plot the output of `Task 25` to a `stacked horizontal bar chart`

- hint: `.plot(kind="barh", stacked=True)`

```
[ ]: ap.groupby(['admission_type','gender']).size().unstack().plot(kind="barh",␣
     ↪stacked=True)
```

3 Good job! Repeat the tasks to other tables