module 4

July 25, 2021

0.1 Module 4

In this assignment, you will continue working on the movie data from IMDB.

- The data includes movies and ratings from the IMDB website
- Data File(s): imdb.xlsx

Data file contains 3 sheets:

- "imdb": contains records of movies and ratings scraped from IMDB website
- "countries": contains the country (of origin) names
- "directors": contains the director names

We have loaded and joined the data as "df" for you. Follow the instructions and finish the rest part.

[1]:
```
############################################################
### EXECUTE THIS CELL BEFORE YOU TO TEST YOUR SOLUTIONS ###
############################################################

import imp, os, sys
sol = imp.load_compiled("solutions", "./solutions.py")
sol.get_solutions("imdb.xlsx")
from nose.tools import assert_equal
from pandas.util.testing import assert_frame_equal, assert_series_equal
```

[2]:
```
# Loading the data
import pandas as pd

xls = pd.ExcelFile('imdb.xlsx')
df = xls.parse('imdb')
df_directors = xls.parse('directors')
df_countries = xls.parse('countries')

df = pd.merge(left=df, right=df_countries,
              how='inner', left_on='country_id',
              right_on='id')

df = pd.merge(left=df, right=df_directors,
```

```
                how='inner', left_on='director_id',
                right_on='id')

print("Finished.")
```

```
Finished.
```

[3]:
```
""" Q1:
Get the summary statistics for imdb_score and gross, then use the describe()␣
 ↪function to summarize this visually. Save the
result in a variable called score_gross_description and print it.
"""

# your code here

score_gross_description =  df[['imdb_score', 'gross']].describe()
score_gross_description
```

[3]:

	imdb_score	gross
count	178.000000	1.780000e+02
mean	8.294382	1.030402e+08
std	0.266960	1.242549e+08
min	8.000000	8.060000e+03
25%	8.100000	1.318510e+07
50%	8.200000	5.194371e+07
75%	8.475000	1.522436e+08
max	9.300000	6.232795e+08

[4]:
```
assert_frame_equal(score_gross_description, sol.score_gross_description)
print("Success!")
```

```
Success!
```

[5]:
```
"""Q2:
What is the average rating of the director Christopher Nolan's movies? Save␣
 ↪this value in a variable called nolan_mean and
print.
"""

# your code here
import numpy as np
nolan_mean = df[df['director_name'] == "Christopher Nolan"]['imdb_score'].mean()
nolan_mean
```

[5]: 8.6

[6]:
```
assert_equal(nolan_mean, sol.nolan_mean)
```

[7]:
```
"""Q3:
Create a series called 'directors' that contains each director's name and his␣
↪or her average rating.  Print out the type of your variable.
Use the 'directors' series to find the average rating for Steven Spielberg. ␣
↪Print the value.
"""

# your code here
directors = df.groupby("director_name")['imdb_score'].mean()
directors['Steven Spielberg']
```

[7]:
```
8.48
```

[8]:
```
assert_series_equal(directors, sol.directors)
print("Success!")
```

```
Success!
```

[9]:
```
"""Q4:
Select the non-USA movies made after 1960 by Hayao Miyazaki.
Save the result in a DataFrame called 'miyazaki', then print it.

Here are the steps:
1. Query the data ('df' DataFrame) based on the following conditions:
- Non-USA movies (country_id != 1)
- Movies made after 1960 (title_year > 1960)
- Movies made by director Hayao Miyazaki (director_id == 46)
2. Save the filtered data in a DataFrame called 'miyazaki' and print it"""


miyazaki = pd.DataFrame(df[df['country_id'] != 1])[df['title_year'] >␣
↪1960][df['director_id'] == 46]
miyazaki
```

[9]:

	movie_title	director_id	country_id	content_rating	title_year	imdb_score	gross	duration	id_x	country	id_y	director_name
152	Spirited Away	46	4	PG	2001	8.6	10049886	125	4	Japan	46	Hayao Miyazaki
153	Princess Mononoke	46	4	PG-13	1997	8.4	2298191	134	4	Japan	46	Hayao Miyazaki
154	Howl's Moving Castle	46	4	PG	2004	8.2	4710455	119	4	Japan	46	Hayao Miyazaki

[10]:
```
assert_frame_equal(miyazaki, sol.miyazaki)
print("Success!")
```

Success!

[11]:
```
"""Q5:
Create a Pivot Table that shows the median rating for each director, grouped by␣
↪their respective countries. Name your variable
'pivot_agg'
"""

# your code here
pivot_agg = pd.pivot_table(df, index = ['country','director_name'],values␣
↪=['imdb_score'], aggfunc = [np.median])
pivot_agg
```

[11]:

		median
		imdb_score
country	director_name	
Argentina	Juan Jose Campanella	8.20
Australia	George Miller	8.10
Brazil	Fernando Meirelles	8.70
	Jose Padilha	8.10
Canada	Denis Villeneuve	8.20
…		…
USA	Tony Scott	8.00
	Victor Fleming	8.15
	Wes Anderson	8.10
	Woody Allen	8.10
West Germany	Wolfgang Petersen	8.40

[125 rows x 1 columns]

[12]:
```
assert_frame_equal(pivot_agg, sol.pivot_agg)
print("Success!")
```

Success!

[13]:
```
"""Q6:
How long did the movie Gladiator aim to keep your attention? Save the series␣
↪with this information
in a variable called 'gladiator_duration', then print it.
"""
gladiator_duration = df[df['movie_title']=='Gladiator']['duration']
gladiator_duration
# your code here
```

[13]:
```
51    171
Name: duration, dtype: int64
```

[14]:
```
assert_series_equal(gladiator_duration, sol.gladiator_duration)
print("Success!")
```

```
Success!
```