MDA-710 Fall 2022

Homework 2 Intro to Programming II & Pandas & Handling Categorical Data

Due: September 29th @ 12AM

Submit on Blackboard

Earnable Points: 100pts
Base Points: 5ts

You can miss 5 points and still get a perfect score on this assignment

Programming II - Classes

Match the numbered parts in the image with the following terms/phrases. Only use each lettered term/phrase once. Each match is worth 2 points.

- A. Keyword used to define a class
- B. Parameter
- C. Defining a Property
- D. Used within a class to reference the current object
- 1. A
- 2. C
- 3. H
- 4. D
- 5. I

- E. Object
- F. Accessing a property
- G. Creating an object/instance
- H. Constructor
- I. Keyword used to define a function
- 6. B
- 7. G
- 8. E
- 9. F

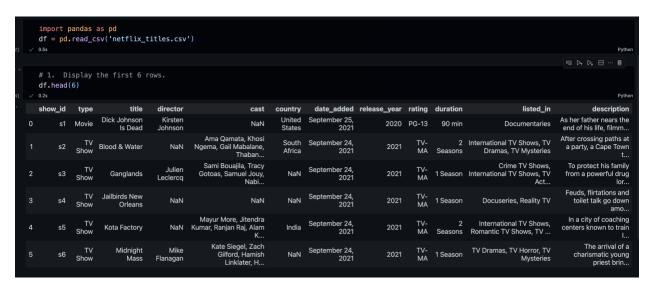
```
1 import numpy as np
3 → class Person:
      name = None
       age = np.nan
 5
      favorite_color = None
 6
7
8 +
      def __init__(self, name, age, favorite_color=None):
         self.name = name
9
         self.age = age
10
          self.favorite_color = favorite_color
11
12
       def __str__(self):
13 -
          return "Person\t" + self.name + "\t" + str(self.age) + "\t" + self.favorite_color
14
15
        def is_older_than(self, other_person):
16 +
          return self.age > other_person.age
17
18
19 danielle = Person("Danielle", 30, "pink")
20 sister = Person("Elise", 18)
21
22 print(danielle)
print(danielle.name + " " + ("is" if danielle.is_older_than(sister) else "is not") + " older than " + sister.name)
```

Pandas

For each question, answer with the code that would perform the given task. Add a screenshot of the output. Each question is worth 3 points except the last question, which is 5 points. (50 total points)

Using pandas, import the dataset "netflix titles" as a dataframe and

1. Display the first 6 rows.



2. Display rows 10 through 20



3. Display rows 10 through 20 and columns 1 through 2

```
# B. Display rows 10 through 20 and columns 1 through 2
  df[10:21][["show_id", "type"]]
✓ 0.2s
   show_id
               type
10
        s11 TV Show
11
        s12 TV Show
12
        s13
              Movie
13
       s14
              Movie
14
       s15 TV Show
15
       s16 TV Show
16
       s17
              Movie
17
       s18 TV Show
18
       s19
              Movie
19
       s20 TV Show
20
        s21 TV Show
```

4. Display the names of all the columns

```
# 4. Display the names of all the columns
   for col in df.columns:
       print(col)
✓ 0.1s
show id
type
title
director
cast
country
date_added
release_year
rating
duration
listed_in
description
```

5. Display the values only for the column 'title' for rows 10 through 20

```
# 5. Display the values only for the column 'title' for rows 10 through 20
   df[10:21]["title"]
                    Vendetta: Truth, Lies and The Mafia
10
11
                                        Bangkok Breaking
12
                                            Je Suis Karl
13
                        Confessions of an Invisible Girl
14
                        Crime Stories: India Detectives
15
                                       Dear White People
      Europe's Most Dangerous Man: Otto Skorzeny in ...
16
17
                                         Falsa identidad
18
                                               Intrusion
19
                                                  Jaguar
        Monsters Inside: The 24 Faces of Billy Milligan
20
Name: title, dtype: object
```

6. Display the standard deviation for the column 'release year'

```
# 6. Display the standard deviation for the column 'release_year'
df.std()["release_year"]
```

/var/folders/57/h9hkt3s54t1cljt_qy48dj540000gn/T/ipykernel_41947/348894
version, it will default to False. In addition, specifying 'numeric_onl
warning.

df.std()["release_year"]

8.819312130833966

7. Remove the 'description' column and show the results



8. Display all the unique values for 'type'

9. Display all the unique values for 'rating'

10. Display all rows for which the type is 'Movie'



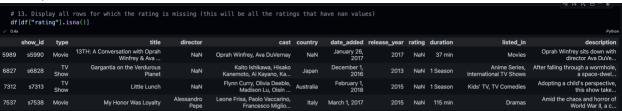
11. Display all row for which the type is 'Movie' and the rating is 'R'



12. Display all rows for which rating does not start with 'TV'



13. Display all rows for which the rating is missing (this will be all the ratings that have *nan* values)



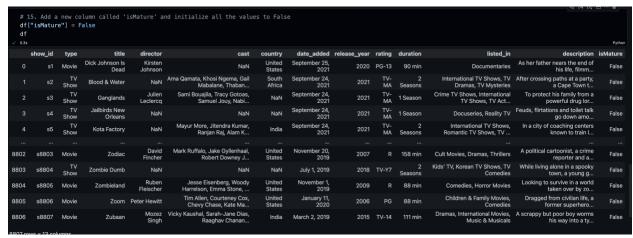
14. How many rows have missing ratings?

```
# 14. How many rows have missing ratings?

df[df["rating"].isna()].shape[0]

> 0.4s
```

15. Add a new column called 'isMature' and initialize all the values to False



16. For each row, set the column value for 'isMature' to True if the rating is any of the following: 'PG-13', 'TV-MA', 'TV-14', 'R', 'NC-17'



MCAR, MNAR, and MAR

(9 points) Notice that one of the values for 'rating' is 'nan'. This means the value is missing. Match each scenario for why the rating is missing with the correct missing data type (each type can only be used once):

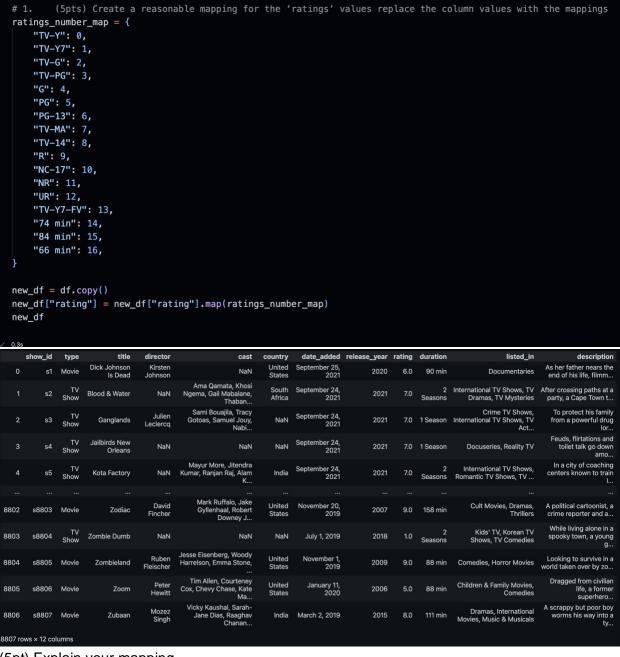
- 1. Movies that are shorter than an hour and are not released in the US or Europe will not have ratings.
 - MNAR Missing not at random
- We have no idea why the ratings are missing MCAR - Missing completely at random
- 3. When constructing the database, entering the ratings were optional MAR Missing at random

Handling Categorical Data

For each question, answer with the code that would perform the given task. Add a screenshot of the output.

Using pandas, import the dataset "netflix titles" as a dataframe and

1. (5pts) Create a reasonable mapping for the 'ratings' values replace the column values with the mappings



2. (5pt) Explain your mapping

I just took a list of all the unique ratings that are present and created a object with serial numbers.

```
ratings_number_map = {
    "TV-Y": 0,
    "TV-Y7": 1,
    "TV-G": 2,
    "TV-PG": 3,
    "G": 4,
    "PG": 5,
    "PG-13": 6,
    "TV-MA": 7,
    "TV-14": 8,
    "R": 9,
    "NC-17": 10,
    "NR": 11,
    "UR": 12,
    "TV-Y7-FV": 13,
    "74 min": 14,
    "84 min": 15,
    "66 min": 16,
```

- 3. After replacing the 'ratings' values with your mapping
 - a. (3pts) Drop all the 'ratings' with missing values (*dropna*) and find the average of the 'rating' column.

b. (3pts) Replace all the 'ratings' with missing values with 0 (*fillna*) and find the new average of the 'rating' column.

c. (4pts) Replace all the 'ratings' with missing values with the average of the 'rating' column (*fillna*) and find the new average of the 'rating' column

4. (3pt) How does the average change based on how you handle it?

Answer:

Dropping all the missing value and filling na values with the average gave same output as calculating average using pandas doesn't consider NA values while doing so.

Filling all na with 0 did reduce the average a bit.