a0-python

● Graded

Student

Tagore Kosireddy

Total Points

50 / 50 pts

Autograder Score

30.0 / 30.0

Passed Tests

Public Tests

q1 (17/17)

q3a (4/4)

q3b (9/9)

Question 2

Q2

12 / 12 pts

✔ **– 0 pts** Correct

Question 3

Q3a-Manual

1 / 1 pt

✔ **– 0 pts** Correct

Question 4

Q3b Manual

3 / 3 pts

✔ **– 0 pts** Correct

Question 5

General

4 / 4 pts

✔ **– 0 pts** To Everyone:
When submitting on Gradescope, make sure to check:
1. Your notebook runs in the Autograder
2. Your notebooks looks correct in its formatting.

✔ **– 0 pts** Stylistically, you should break your commands that are running over 80 characters long into separate lines. Future assignments will lose points

✔ **– 0 pts** Code is not commented

Add good comments to your code.

Autograder Results

Autograder Output

```
  _________        __       __
 /   _____ \    __|  |__ __|  |__
|  /     \  |  |__    __|__    __|   _______   _  _____
| |       | |     |  |      |  |    |  ___  | | |/ ____|
| |       | |     |  |      |  |    | |___| | |  /
| |       | |     |  |      |  |    |  _____| | |
|  \_____/  |     |  |_     |  |_   | |_____  | |
 \_________/      \__|      \__|    \_______| |_|
                                               v5.2.3






------------------------------- GRADING SUMMARY --------------------------------

No discrepancies found while verifying scores against the log.


Total Score: 30.000 / 30.000 (100.000%)

           name  score  max_score
0  Public Tests    NaN        NaN
1            q1   17.0       17.0
2           q3a    4.0        4.0
3           q3b    9.0        9.0
```

Public Tests

q1 results: All test cases passed!

q3a results: All test cases passed!

q3b results: All test cases passed!

q1 (17/17)

q1 results: All test cases passed!

q3a (4/4)

q3a results: All test cases passed!

q3b (9/9)

q3b results: All test cases passed!

Submitted Files

▾ a0.ipynb Download

a0 - Python Assignment

Upon successful completion of this assignment, a student will be able to:

- Correctly setup Python environment on Campus Linux Machines
- Gain experience in formatting text using Markdown
- Submit assignment to Gradescope.

You will also become familar with the general format of the programming assignments. Typically, you will be provided information for the question and be asked to fill in a function or code in a cell(s).

Sometimes, you will also be asked to respond to your analysis or describe the results (filling in a Markdown cell).

Submission Guidelines

- Please restrict the code you write to those cells provided in the Jupyter notebook.
  While you are debugging you may add extra code cells to help you debug and develop your answers, but before submitting you should clean up your code limit the code you enter to those cells.
- Suppress or remove any extra print statements that you add.
  There are certain variables being displayed or printed in the default code you are supplied with the lab. You should suppress other print statements. Note, while you are working on your solution you may add your own print statements for debugging, but these should be commented our or removed before submission.

Q0 - Setup Python Environment

You must setup a Python environment following the instructions on Canvas.

Q1 - Confirm Setup

The following code looks to see whether your notebook is run on Gradescope (GS), Colab (COLAB), or the linux Python environment you were asked to setup.

In [22]:

```
import re
import os
import platform
import sys

# flag if notebook is running on Gradescope
if re.search(r'amzn', platform.uname().release):
    GS = True
else:
    GS = False

# flag if notebook is running on Colaboratory
try:
  import google.colab
  COLAB = True
except:
  COLAB = False

# flag if running on Linux lab machines.
cname = platform.uname().node
if re.search(r'(guardian|colossus|c28)', cname):
    LLM = True
else:
    LLM = False

print("System: GS - %s, COLAB - %s, LLM - %s" % (GS, COLAB, LLM))
```

```
System: GS - False, COLAB - False, LLM - True
```

Python Version

Check to make sure the correct version of Python was run.

In [23]:

```
pver = sys.version
print(pver)
```

```
3.10.13 (main, Sep 11 2023, 13:44:35) [GCC 11.2.0]
```

Notebook/Package Setup

It is good practice to list all imports needed at the top of the notebook. You can import modules in later cells as needed, but listing them at the top clearly shows all which are needed to be available / installed.

Because we are testing the environment, we do not follow this best practice here.

In [24]:

```
# Import standard DS packages
import pandas as pd
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt
%matplotlib inline

# Package for Autograder
import otter
grader = otter.Notebook()
```

Developing on Colab

If you are doing development on Colab, the otter-grader package is not available, so you will need to install it with pip (uncomment the cell directly below). Make sure this code gets commented again before running on the Campus Linux machines.

In [25]:

```
# Only uncomment if you developing on Colab
# if COLAB == True:
#     print("Installing otter:")
#     !pip install otter-grader==5.2.3
```

Environment Setup

The python environment that is running is:

In [26]:

```
env1 = sys.executable
print(env1)
```

```
/home/campus19/trkosire/.conda/envs/cs5831/bin/python
```

In [27]:

```
env2 =!conda info | grep 'active env'
print(env2)
```

```
['     active environment : cs5831', '    active env location : /home/campus19/trkosire/.cond
```

Make sure that the environment you set up for the class is what is being used to execute your notebook. For example, the default name should be "cs4821" or "cs5831".

Next, we are going to look at many of the packages installed.

In [28]:

```
import importlib
from packaging.version import Version, parse

OK = '\x1b[42m[ OK ]\x1b[0m'
FAIL = "x1b[41m[FAIL]\x1b[0m"

def import_version(pkg, req_ver, fail_msg=""):
    mod = None
    try:
        mod = importlib.import_module(pkg)
        ver = mod.__version__
        if Version(ver) != req_ver:
            print(FAIL, "%s version %s required, but %s installed."
                  % (lib, req_ver, ver))
        else:
            print(OK, '%s version %s' % (pkg, ver))
    except ImportError:
        print(FAIL, '%s not installed. %s' % (pkg, fail_msg))
    return (mod, Version(ver), req_ver)

requirements = {'pandas': parse("2.1.4"), 'numpy': parse("1.26.3"),
                'matplotlib': parse("3.8.0"), 'scipy': parse("1.11.4"),
                'seaborn': parse('0.12.2'), 'plotly': parse("5.9.0"),
                'plotly': parse("5.9.0"), 'sklearn': parse("1.3.0"),
                'nltk': parse("3.8.1"), 'spacy': parse('3.7.2'),
                'imblearn': parse('0.11.0'), 'mlxtend': parse('0.23.1'),
                'dill': parse('0.3.7'), 'otter': parse('5.2.3')
                }

pks = []
for lib, required_version in list(requirements.items()):
    pks.append(import_version(lib, required_version))
```

```
[42m[ OK ][0m pandas version 2.1.4
[42m[ OK ][0m numpy version 1.26.3
[42m[ OK ][0m matplotlib version 3.8.0
[42m[ OK ][0m scipy version 1.11.4
[42m[ OK ][0m seaborn version 0.12.2
[42m[ OK ][0m plotly version 5.9.0
[42m[ OK ][0m sklearn version 1.3.0
```

```
[42m[ OK ][0m nltk version 3.8.1
[42m[ OK ][0m spacy version 3.7.2
[42m[ OK ][0m imblearn version 0.11.0
[42m[ OK ][0m mlxtend version 0.23.1
[42m[ OK ][0m dill version 0.3.7
[42m[ OK ][0m otter version 5.2.3
```

In [29]:

```
grader.check("q1")
```

Out [29]:

```
q1 results: All test cases passed!
```

Q2

Enter the following formatted text into the Markdown cell below.

In the same Markdown cell, with each element on a newline:

- Add a first level heading with "a0-Python"
- Add a second level heading with the due date formatted as: `Due: DayOfWeek, Mon. Day, Year` replacing DayOfWeek, Mon. - Month, Day, and Year with the correct information.
- Add boldface course "CS 4821" or "CS 5841"
- On the next line, add the semester "Spring 2024" in italics
- Copy over the course learning objectives from the syllabus as an unordered list, e.g., "Upon successful completion of this course, ..."
- Create a table with the grading scheme A-F from the online syllabus
- Add a link to the Office of Academic and Community Conduct Resource Center for Students

Enter your answer in this Markdown block

a0-Python

Due: Tuesday, Janurary, 23, 2024

CS 5841

Spring 2024

Learning Objectives Upon successful completion of this course, students will be able to:

- Describe advanced data mining concepts and techniques
- Apply the techniques of clustering, classification, association finding, and visualization on sample and real-world data sets
- Select and apply data mining tools to real-world problems
- Formulate a real-world data mining problem: find, get, and clean the data, select and implement appropriate methods, evaluate the performance of the methods, etc.
- Discuss and communicate the results and methods of applying data mining techniques to a problem

Grading Scheme

Percentage	Letter Grade
> 93%	A
[88 - 93%)	AB
[83 - 88%)	B
[78 - 83%)	BC
[73 - 78%)	C
[68 - 73%)	CD
[60 - 68%)	D
< 60%	F

Office of Academic and Community Conduct Resource Center for Students

Q3 - Pandas

We are going to look at a data set based on Amazon's top 50 bestselling books from 2009 to 2019 (available from Kaggle).

Q3a - Load Data

Load in the csv file, `amazon-bestsellers.csv` into a `pandas` DataFrame using the `read_csv` function.

Pandas Resources:

- https://pandas.pydata.org/
- https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.read_csv.html

The data's columns are:

- `Name`: name of the book
- `Author`: author of the book
- `User Rating`: Amazon user rating
- `Reviews`: number of written reviews on Amazon
- `Price`: the price of the book (as of 10/13/2020)
- `Year`: the year it ranked on the best seller.
- `Genre`: whether it is fiction or not (from Goodreads)

In [30]:

```
df = pd.read_csv("amazon-bestsellers.csv")  # modify this code to properly read the data
# use the column names provided above
df.head()
```

Out [30]:

```
                                          Name  \
0                   10-Day Green Smoothie Cleanse   
1                               11/22/63: A Novel   
2         12 Rules for Life: An Antidote to Chaos   
3                          1984 (Signet Classics)   
4  5,000 Awesome Facts (About Everything!) (Natio...   

                     Author  User Rating  Reviews  Price  Year        Genre  
0                  JJ Smith          4.7    17350      8  2016  Non Fiction  
1              Stephen King          4.6     2052     22  2011      Fiction  
2        Jordan B. Peterson          4.7    18979     15  2018  Non Fiction  
3             George Orwell          4.7    21424      6  2017      Fiction  
4  National Geographic Kids          4.8     7665     12  2019  Non Fiction  
```

In [31]:

```
grader.check("q3a")
```

Out [31]:

```
q3a results: All test cases passed!
```

Q3b - DataFrame Statistics

Find the name of the book with the minimum number of reviews.

Find the name of the book with the maximum rating breaking ties with the highest price.

Find the average price of all the non-fiction books.

Helpful Resource for anyone getting started using `pandas`:
[Jake VanderPlas Python Data Science Handbook Chapter on Pandas]

In [32]:

```
min_review_val = df.Reviews.min()
min_review_df = df[ df['Reviews'] == min_review_val]
book_min_reviews = min_review_df['Name'].iloc[0]
max_rating_val = df['User Rating'].max()
max_rating_price_df_1 = df[ df['User Rating'] == max_rating_val]
highest_price = max_rating_price_df_1.Price.max()
max_rating_price_df_2 = max_rating_price_df_1 [ max_rating_price_df_1['Price'] == highest_price]
book_max_rating = max_rating_price_df_2['Name'].iloc[0]
non_fiction_df = df[df.Genre == 'Non Fiction']
ave_price_nonfiction = non_fiction_df.Price.mean()
ave_price_nonfiction
print(book_min_reviews)
print(book_max_rating)
print(ave_price_nonfiction)
```

```
Divine Soul Mind Body Healing and Transmission System: The Divine Way to Heal You, Hu
Hamilton: The Revolution
14.841935483870968
```

In [33]:

```
grader.check("q3b")
```

Out [33]:

```
q3b results: All test cases passed!
```

Checks before submitting

Make sure to run this notebook on the campus linux machines.

FAQ:

- Q: I notice my zip file is growing as I repeatedly run the notebook?

- A: This assignment is graded from a log file called `.OTTER_LOG`. You can delete the `.OTTER_LOG` file, before rerunning the entire notebook `Cell -> Run All`.
- Q: I am getting different package numbers?
- A: Make sure you follow the instructions to set up the Python environment on the campus linux machines.

Important submission steps: 1. Run the tests and verify that they all pass. 2. Choose **Save Notebook** from the **File** menu, then **run the final cell**. 3. Click the link to download the zip file. 4. Then submit the zip file to the corresponding assignment on Gradescope.

It is your responsibility to make sure your work is saved before running the last cell.

Submission

Make sure you have run all cells in your notebook in order before running the cell below, so that all images/graphs appear in the output. The cell below will generate a zip file for you to submit. **Please save before exporting!**

NOTE the submission must be run on the campus linux machines. See the instruction in the Canvas assignment.

In [ ]:

```
# Save your notebook first, then run this cell to export your submission.
grader.export(pdf=False, run_tests=True)
```

▾ .OTTER_LOG Download

```
1 Large file hidden. You can download it using the button above.
```

▾ __zip_filename__ Download

```
1 a0_2024_01_23T11_43_05_374291.zip
```

a0.otter Download

```
{
    "notebook": "a0.ipynb",
    "save_environment": true,
    "ignore_modules": []
}
```