



MAPÚA UNIVERSITY

SCHOOL OF ELECTRICAL, ELECTRONICS, AND COMPUTER ENGINEERING

Experiment 7: API and Web Scraping

CPE106L (Software Design Laboratory)

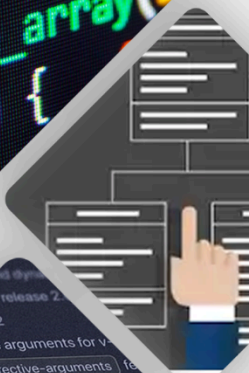
Danikka Villaron

Jessa Abigaile Tongol

Mark Vincent De Villa

Matthew Benedict Nepomuceno

Group No.: 2
Section: E03



PreLab

Insights and Reflections

**<De Villa> Fundamentals Of Python: First Programs 3rd Edition (Chapter 11)
9780357881132**

The book "Fundamentals Of Python: First Programs" (3rd ed.) introduced the idea of data analysis and visualization in Chapter 11. Chapter 11.1 presented methods to compute and determine significant statistical values in a dataset like the mean, median, mode, standard deviation, minimum, and maximum values utilizing the Python language and NumPy library. Chapter 11.2 presents the methods to visualize the relationship between values in a dataset, creating pie charts, bar charts, scatter plots, line plots, and histograms using Python. Chapter 11.3 presents other libraries and methods to write, access, clean, and visualize datasets, specifically with the "pandas" and "matplotlib" Python libraries.

<Villaron> Getting started - Matplotlib 3.8.0 documentation

The Getting Started guide for Matplotlib gives a quick rundown on how to install the library and create a basic plot. For installation, it recommends using either `pip install matplotlib` or `conda install -c conda-forge matplotlib`. It also provided a way to make a simple plot and a guide to create a basic example using numpy for data generation and Matplotlib's pyplot module for plotting. The shown script makes a sine wave plot by creating an array of x values over a set range and calculating their sine values for y.

PostLab

Programming Problems

PE #5: Visit the website of the U.S. Bureau of Labor Statistics at <https://www.bls.gov/data/home.htm> and download the data for the average price of bread, as shown earlier in this chapter (there will be data for more recent years added since these words were written). You can also use the breadprice.csv file here >> [Chapter 11 Data files](#). Write a program in a file named breadprice.py that loads the data set and cleans it as you did earlier in this chapter. Then include code to display a line plot of the average price for each year in the table.

```
breadprice.py > ...
import pandas as pd
import matplotlib.pyplot as plt

file_path = "breadprice.csv"
df = pd.read_csv(file_path)

print("Initial Data:")
print(df.head())

df['Year'] = df['Year'].astype(int)
df['Average Price'] = df.iloc[:, 1:].mean(axis=1)

avg_price_per_year = df[['Year', 'Average Price']].set_index('Year')

plt.figure(figsize=(10, 5))
plt.plot(avg_price_per_year.index, avg_price_per_year['Average Price'], marker='o', linestyle='-', color='b')
plt.xlabel("Year")
plt.ylabel("Average Bread Price")
plt.title("Average Bread Price per Year")
plt.grid(True)
plt.show()
```

Figure 1.1 breadprice.py program

In this program, we imported two libraries, pandas and matplotlib.pyplot, which help us handle data and make graphs. After importing necessary libraries, we read the breadprice.csv file into a table called df. This allows for easier data representation. We printed the first few rows to see what the data looks like and converted the Year column to integers to ensure it's in the correct format. After these, we calculated the average bread price for each year by taking the mean of the other columns (which we assume are different types of bread prices) and add it as a new column called Average Price and created a new table called avg_price_per_year that has Year as the index and Average Price as column. After all the process, the last part of the code plots the data using the matplotlib.

Initial Data:

	Year	Jan	Feb	Mar	Apr	May	Jun	Jul	Aug	Sep	Oct	
0	2012	1.423	1.442	1.395	1.426	1.412	1.403	1.427	1.487	1.481	1.422	
1	2013	1.422	1.411	1.412	1.409	1.401	1.439	1.434	1.408	1.419	1.358	
2	2014	1.365	1.388	1.359	1.388	1.481	1.400	1.413	1.396	1.405	1.414	
3	2015	1.479	1.435	1.440	1.454	1.463	1.467	1.447	1.420	1.432	1.418	
4	2016	1.425	1.407	1.416	1.406	1.382	1.333	1.349	1.341	1.329	1.343	

	Nov	Dec
0	1.418	1.436
1	1.382	1.385
2	1.420	1.466
3	1.409	1.428
4	1.362	1.362

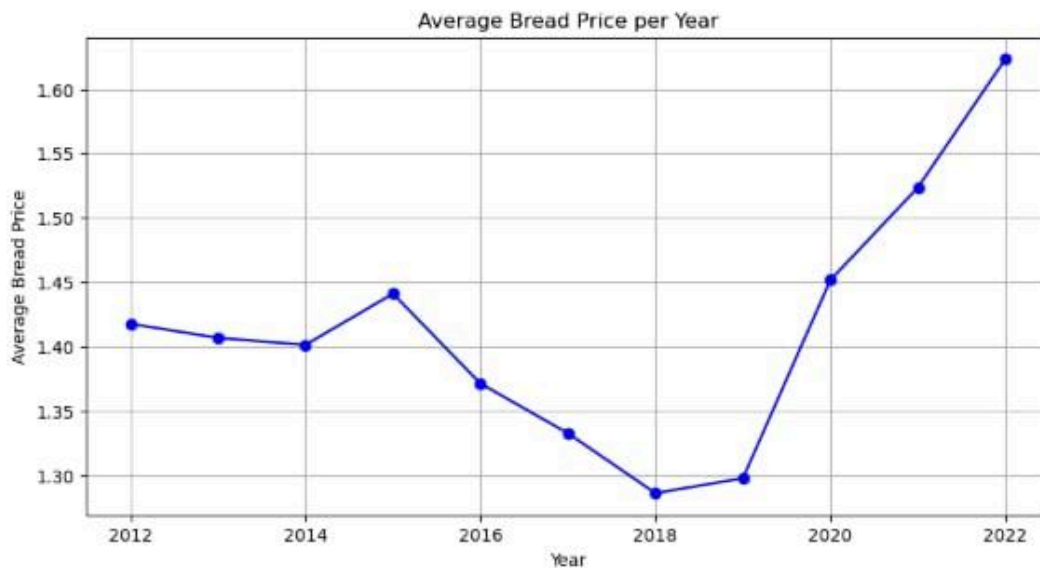


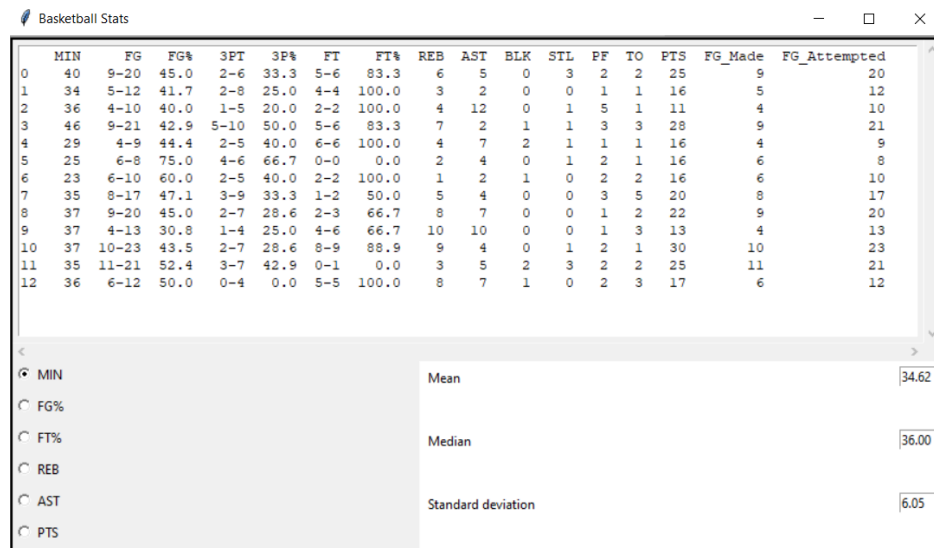
Figure 1.2 Output in Jupyter Lab

Desired output that plots the average price for each year in the table.

PE #6: The columns labeled FG, 3PT, and FT of the data set in the Analyzing Basketball Statistics case study (Download the files here >> [CaseStudy2](#)) do not show a single integer value but instead show values with the format <makes-attempts>, which is not suitable for the kind of data analysis performed on the other columns. For example, analysts might like to view the mean of free throws attempted as well as mean of the free throw percentage. You can correct this problem with a cleaning step that, for each such column:

- removes it from the data frame
- creates two new columns from this series, where the first column includes the numbers of makes and the second column includes the number of attempts
- inserts the new pairs of columns into the data frame at the appropriate positions, with the appropriate column headings (for example, FTM and FTA)

Define a function named **cleanStats** in the file hoopstatsapp.py. This function expects a data frame as an argument and returns the frame cleaned according to the steps listed previously. You should call this function after the frame is loaded from the CSV file and before it is passed to the **HoopStatsView** constructor.



	MIN	FG	FG%	3PT	3P%	FT	FT%	REB	AST	BLK	STL	PF	TO	PTS	FG_Made	FG_Attempted
0	40	9-20	45.0	2-6	33.3	5-6	83.3	6	5	0	3	2	2	25	9	20
1	34	5-12	41.7	2-8	25.0	4-4	100.0	3	2	0	0	1	1	16	5	12
2	36	4-10	40.0	1-5	20.0	2-2	100.0	4	12	0	1	5	1	11	4	10
3	46	9-21	42.9	5-10	50.0	5-6	83.3	7	2	1	1	3	3	28	9	21
4	29	4-9	44.4	2-5	40.0	6-6	100.0	4	7	2	1	1	1	16	4	9
5	25	6-8	75.0	4-6	66.7	0-0	0.0	2	4	0	1	2	1	16	6	8
6	23	6-10	60.0	2-5	40.0	2-2	100.0	1	2	1	0	2	2	16	6	10
7	35	8-17	47.1	3-9	33.3	1-2	50.0	5	4	0	0	3	5	20	8	17
8	37	9-20	45.0	2-7	28.6	2-3	66.7	8	7	0	0	1	2	22	9	20
9	37	4-13	30.8	1-4	25.0	4-6	66.7	10	10	0	0	1	3	13	4	13
10	37	10-23	43.5	2-7	28.6	8-9	88.9	9	4	0	1	2	1	30	10	23
11	35	11-21	52.4	3-7	42.9	0-1	0.0	3	5	2	3	2	2	25	11	21
12	36	6-12	50.0	0-4	0.0	5-5	100.0	8	7	1	0	2	3	17	6	12

Summary statistics for MIN:

- Mean: 34.62
- Median: 36.00
- Standard deviation: 6.05

Figure 2.1 Original Output

This shows the original, unchanged file where the pairs in FG, 3PT, and FT aren't separated yet

The screenshot shows a window titled "Basketball Stats" containing a table of 13 rows of player statistics. The columns are: MIN, FGM, FGA, FG%, 3PTM, 3PTA, 3P%, FTM, FTA, FT%, REB, AST, BLK, STL, PF, TO, and PTS. Below the table is a summary panel with a list of statistics on the left and their corresponding values on the right.

	MIN	FGM	FGA	FG%	3PTM	3PTA	3P%	FTM	FTA	FT%	REB	AST	BLK	STL	PF	TO	PTS
0	40	9	20	45.0	2	6	33.3	5	6	83.3	6	5	0	3	2	2	25
1	34	5	12	41.7	2	8	25.0	4	4	100.0	3	2	0	0	1	1	16
2	36	4	10	40.0	1	5	20.0	2	2	100.0	4	12	0	1	5	1	11
3	46	9	21	42.9	5	10	50.0	5	6	83.3	7	2	1	1	3	3	28
4	29	4	9	44.4	2	5	40.0	6	6	100.0	4	7	2	1	1	1	16
5	25	6	8	75.0	4	6	66.7	0	0	0.0	2	4	0	1	2	1	16
6	23	6	10	60.0	2	5	40.0	2	2	100.0	1	2	1	0	2	2	16
7	35	8	17	47.1	3	9	33.3	1	2	50.0	5	4	0	0	3	5	20
8	37	9	20	45.0	2	7	28.6	2	3	66.7	8	7	0	0	1	2	22
9	37	4	13	30.8	1	4	25.0	4	6	66.7	10	10	0	0	1	3	13
10	37	10	23	43.5	2	7	28.6	8	9	88.9	9	4	0	1	2	1	30
11	35	11	21	52.4	3	7	42.9	0	1	0.0	3	5	2	3	2	2	25
12	36	6	12	50.0	0	4	0.0	5	5	100.0	8	7	1	0	2	3	17

MIN	Mean	19.62
FG%	Median	17.00
FT%	Standard deviation	5.91
REB		
AST		
PTS		

Figure 2.2 Modified Output

Figure 2.2 shows the modified CSV file in a GUI, with the difference from the original being FC, 3PT, and FT had their data separated into two columns in the format of attempts and makes, and moves the new columns in the index of the old columns.

```

hoopstatsapp.py > cleanStats
1  """
2  File: hoopstatsapp.py
3
4  The application for analyzing basketball stats.
5  """
6
7  from hoopstatsview import HoopStatsView
8  import pandas as pd
9  import os
10
11  def main():
12      """Creates the data frame and view and starts the app."""
13      frame = pd.read_csv("newbrogdonstats.csv")
14      cleanStats(frame)
15      HoopStatsView(frame).mainloop()
16
17  def cleanStats(oldFrame):
18      """Separates FG, 3PT, and FT into two columns, and exports a new data set"""
19      if os.path.exists("newbrogdonstats.csv"):
20          framing = pd.read_csv("newbrogdonstats.csv")
21          if 'FG' in framing.columns:
22              #FG Col
23              framing[["FGM", "FGA"]] = framing["FG"].str.split("-", expand=True).astype(int)
24              fg_index = framing.columns.get_loc("FG")
25              framing = framing.drop(columns=["FG"])
26
27              framing.insert(fg_index, "FGM", framing.pop("FGM"))
28              framing.insert(fg_index+1, "FGA", framing.pop("FGA"))
29
30          if '3PT' in framing.columns:
31              #3PT Col
32
33              framing[["3PTM", "3PTA"]] = framing["3PT"].str.split("-", expand=True).astype(int)
34              pt_index = framing.columns.get_loc("3PT")
35              framing = framing.drop(columns=["3PT"])
36
37              framing.insert(pt_index, "3PTM", framing.pop("3PTM"))
38              framing.insert(pt_index+1, "3PTA", framing.pop("3PTA"))
39
40          if 'FT' in framing.columns:
41              #FT Colc
42
43              framing[["FTM", "FTA"]] = framing["FT"].str.split("-", expand=True).astype(int)
44              ft_index = framing.columns.get_loc("FT")
45              framing = framing.drop(columns=["FT"])

```

Figure 2.3.1 Modified hoopstatsapp.py (Part 1)

```

46         framing.insert(ft_index, "FTM", framing.pop("FTM"))
47         framing.insert(ft_index+1, "FTA", framing.pop("FTA"))
48
49     else:
50         framing = pd.read_csv(oldFrame)
51         if 'FG' in framing.columns:
52             #FG Col
53             framing[["FGM", "FGA"]] = framing["FG"].str.split("-", expand=True).astype(int)
54             fg_index = framing.columns.get_loc("FG")
55             framing = framing.drop(columns=["FG"])
56
57             framing.insert(fg_index, "FGM", framing.pop("FGM"))
58             framing.insert(fg_index+1, "FGA", framing.pop("FGA"))
59
60         if '3PT' in framing.columns:
61             #3PT Col
62
63             framing[["3PTM", "3PTA"]] = framing["3PT"].str.split("-", expand=True).astype(int)
64             pt_index = framing.columns.get_loc("3PT")
65             framing = framing.drop(columns=["3PT"])
66
67             framing.insert(pt_index, "3PTM", framing.pop("3PTM"))
68             framing.insert(pt_index+1, "3PTA", framing.pop("3PTA"))
69
70         if 'FT' in framing.columns:
71             #FT Colc
72
73             framing[["FTM", "FTA"]] = framing["FT"].str.split("-", expand=True).astype(int)
74             ft_index = framing.columns.get_loc("FT")
75             framing = framing.drop(columns=["FT"])
76
77             framing.insert(ft_index, "FTM", framing.pop("FTM"))
78             framing.insert(ft_index+1, "FTA", framing.pop("FTA"))
79
80         framing.to_csv("newbrogdonstats.csv", index = False)
81
82     if __name__ == "__main__":
83         main()
84

```

Figure 2.3.2 Modified hoopstatsapp.py (Part 2)

The new function, cleanStats() allows the program to create a new CSV file so the program can access the cleanbrogdonstats.csv and extract its contents, separate the FG and FT columns then we can finally input these into the new CSV file. This file is then used as the new file the program will get its data from to put into the GUI.