ps2_answers

October 3, 2023

1 BA222 Problem Set 2

Ethan Chang !pip3.12 install numpy

[1]:
```
import pathlib
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

1.1 Part 1: Python Basics

1.1.1 Question 1: Compound Interest Calculator

[2]:
```
def compound(K: float, r: float, T: float) -> float:
    """ Returns the dollar value of money (including the principal `K`),
        that will be had at the end of `T` years using rate `r`"""
    return K * (1 + r) ** T

assert(np.round(compound(1000, 0.05, 10), 2) == 1628.89)
assert(np.round(compound(100, 0.04, 20), 2) == 219.11)
```

[3]:
```
if compound(5000, 0.06, 10) > compound(2500, 0.06, 25):
    print("$5,000 compounding for 10 years at a 6% interest rate is better")
else:
    print("$2,500 compounding for 25 years at a 6% interest rate is better")
```

```
$2,500 compounding for 25 years at a 6% interest rate is better
```

1.1.2 Question 2: Checking First and Last

[4]:
```
def FirstLastSame(arr: list) -> bool:
    """ Returns True if the first and last elements of `arr` are the same """
    return arr[0] == arr[-1]

assert(FirstLastSame([1, 0, 1]) == True)
assert(FirstLastSame([10, 9, 8, 7, 6]) == False)
```

1.2 Part 2: Numpy and Pandas

1.2.1 Question 3: Numpy Simulation and Drop-Shipping Business

[5]:
```
obs_price: np.ndarray = np.random.normal(20, 2, 1000)
obs_quant: np.ndarray = np.random.randint(100, 201, 1000)
obs_unit_cost: np.ndarray = np.random.normal(3, 0.5, 1000)
```

[6]:
```
def profit(price: float, quantity: int, unit_cost: float, fixed_cost = 500,␣
 ↪shipping = 5) -> float:
    """ Returns the profit given the price, quantity, and unit cost """
    return price * quantity - (unit_cost + shipping) * quantity - fixed_cost
```

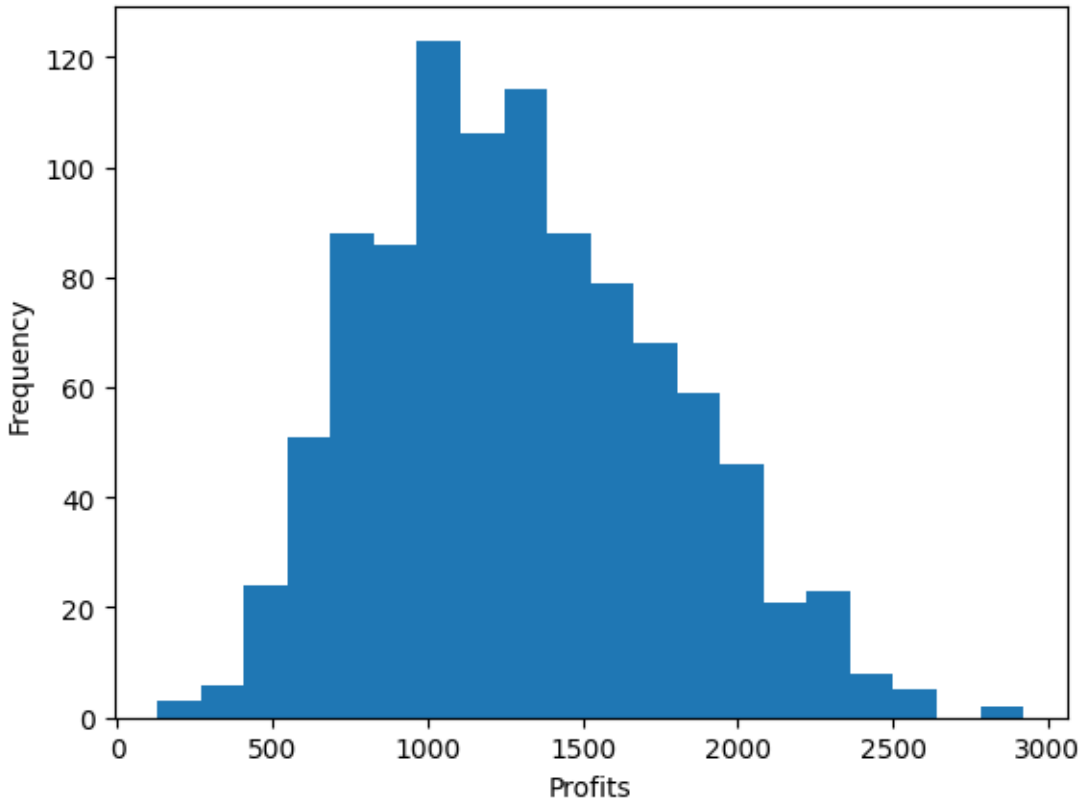
[7]:
```
profits = [profit(price, quantity, unit_cost) for price, quantity, unit_cost \
    in zip(obs_price, obs_quant, obs_unit_cost, strict=True)]

pd.DataFrame({'profits': profits}).describe()
```

[7]:

	profits
count	1000.000000
mean	1304.971753
std	476.968217
min	130.191879
25%	954.161890
50%	1267.403548
75%	1637.583221
max	2920.778185

[8]:
```
plt.hist(profits, bins=20)
plt.xlabel('Profits')
plt.ylabel('Frequency')
plt.show()
```



The distribution is roughly normal, with a possible minor skew to the right.

1.2.2 Question 4: Real Data Analysis in Pandas

[9]:
```
congress = pd.read_csv(f'{pathlib.Path.cwd().parents[1]}/CSVs/CongressTerms.
↪csv')
congress.head()
```

[9]:

	congress	chamber	firstname	lastname	birthday	state	party	Incumbent(Dummy)	termstart	age
0	107	house	Benjamin	Gilman	12/6/1922	NY	R	1	1/3/2001	78.1
1	107	house	Ralph	Hall	5/3/1923	TX	D	1	1/3/2001	77.7
2	107	house	Henry	Hyde	4/18/1924	IL	R	1	1/3/2001	76.7
3	107	house	Ralph	Regula	12/3/1924	OH	R	1	1/3/2001	76.1
4	107	house	Carrie	Meek	4/29/1926	FL	D	1	1/3/2001	74.7

[10]:
```
num_obs = len(congress)
print(f'There are {num_obs} observations in the dataset')

num_house = len(congress[congress['chamber'] == 'house'])
print(f'There are {num_house} House observations and {num_obs - num_house}␣
 ↪Senate observations')
```

```
There are 3822 observations in the dataset
There are 3098 House observations and 724 Senate observations
```

[11]:
```
# mean age of senators in the 107th session of congress
congress[(congress['congress'] == 107) & (congress['chamber'] ==␣
 ↪'senate')]['age'].mean()
```

[11]:
```
58.89903846153846
```

[12]:
```
congress['age'].idxmax()
```

[12]:
```
congress                  107
chamber                senate
firstname                  J.
lastname             Thurmond
birthday            12/5/1902
state                      SC
party                       R
Incumbent(Dummy)            1
termstart            1/3/2001
age                      98.1
Name: 442, dtype: object
```

The value seems reasonable as James Strom Thurmond Sr. lived until he was 100.

[13]:
```
congress.query("chamber == 'house'").query('congress == 110')['firstname'].
 ↪value_counts().head(1)
```

[13]:
```
firstname
John    30
Name: count, dtype: int64
```

[ ]:
```

```