Zeru-Zhou-project3

January 30, 2022

1 Project 3 – Zeru Zhou

TA Help: NA

Collaboration: NA

- Get help from Dr. Ward's videos

1.1 Question 1

[1]:

```
from block_timer.timer import Timer
import pandas as pd


Intruder = pd.read_csv('/depot/datamine/data/noaa/2020_sampleB.csv',␣
 ↪names=["station_id", "date", "element_code", "value", "mflag", "qflag",␣
 ↪"sflag", "obstime"])
intruder_ids = Intruder["station_id"].dropna().tolist()
unique_intruder = list(set(intruder_ids))


Original = pd.read_csv('/depot/datamine/data/noaa/2020_sample.csv',␣
 ↪names=["station_id", "date", "element_code", "value", "mflag", "qflag",␣
 ↪"sflag", "obstime"])
original_ids = Original["station_id"].dropna().tolist()
unique_ids = list(set(original_ids))

with Timer():
    # compare the two lists
    for i in unique_intruder:
        if i not in unique_ids:
            print(i)
```

```
USFAKEROW22
```

```
Total time 45.45118 seconds.
```

[2]:

```
with Timer():
    print(set(intruder_ids) - set(original_ids))
```

```
{'USFAKEROW22'}
```

```
Total time 2.04778 seconds.
```

As we can see, doing set calculation is much time efficient than checking with a loop. It is 22 times faster than the original method.

1.2 Question 2

[4]:

```python
import pandas as pd

mydf = pd.read_csv("/depot/datamine/data/iowa_liquor_sales/clean_sample.csv",␣
 ↪sep=";")
```

[3]:

```python
sales_list = mydf['Sale (Dollars)'].dropna().tolist()
```

[4]:

```python
from block_timer.timer import Timer
with Timer(title = "List Loop"):
    value = 0.0
    for i in sales_list:
        value += i
    print(f'Average Sales is {value/len(sales_list)}')
```

```
Average Sales is 126.70881221986102
```

```
[List Loop] Total time 0.57098 seconds.
```

[5]:

```python
with Timer(title = "Series Loop"):
    value = 0.0
    for idx, val in mydf['Sale (Dollars)'].dropna().iteritems():
        value += val
    print(f'Average Sales is {value/len(sales_list)}')
```

```
Average Sales is 126.70881221986102
```

```
[Series Loop] Total time 2.09563 seconds.
```

Method with list loop is 4 times faster than Series loop.

1.3 Question 3

[6]:

```python
with Timer(title="Loops"):

    # calculate the mean
    mean = sum(sales_list)/len(sales_list)

    # calculate the std deviation
    # you can use **2 to square a value and
    # **0.5 to square root a value
    Std_sqr = 0
    for value in sales_list:
        Std_sqr += (value - mean)**2 / len(sales_list)
    My_std = Std_sqr ** 0.5

    # calculate the list of z-scores
    zscores = []
    for value in sales_list:
        zscores.append((value - mean) / My_std)


    # print the first 5 z-scores
    print(zscores[:5])
```

```
[0.08084600884959506, 0.44521166882119256, -0.24055833528157824,
0.07453606558553277, -0.24055833528157824]
```

```
[Loops] Total time 2.70004 seconds.
```

[8]:
```
with Timer(title="Vectorization"):
    print(((mydf['Sale (Dollars)'] - mydf['Sale (Dollars)'].mean())/mydf['Sale␣
 ↪(Dollars)'].std()).iloc[0:5])
```

```
0    0.080846
1    0.445212
2   -0.240558
3    0.074536
4   -0.240558
Name: Sale (Dollars), dtype: float64
```

```
[Vectorization] Total time 0.56405 seconds.
```

The results are exactly the same. Using Series is 5 times faster than using for loops here.

1.4 Question 4

[3]:
```
import pandas as pd
from collections import defaultdict
```

[5]:
```
mydf = mydf.loc[:, ('Store Number', 'Volume Sold (Gallons)')]
```

[9]:
```
volumn_dict = defaultdict(int)
```

[10]:
```
for idx, val in mydf.iterrows():
    volumn_dict[val['Store Number']] += val['Volume Sold (Gallons)']
```

The volumn dictionary is created with keys and values correspond to these 2 columns.

1.5 Question 5

[13]:

```
for key in volumn_dict:
    if volumn_dict[key] < 100000:
        continue
    elif volumn_dict[key] > 149999:
        print(f'High: {key}')
    else:
        print(f'Low: {key}')
```

```
Low: 2190.0
High: 4829.0
High: 2633.0
High: 2512.0
Low: 3494.0
Low: 2625.0
High: 3420.0
Low: 3952.0
High: 3385.0
Low: 3354.0
Low: 3814.0
```

As code above, we got the output we want.

1.6 Pledge

By submitting this work I hereby pledge that this is my own, personal work. I've acknowledged in the designated place at the top of this file all sources that I used to complete said work, including but not limited to: online resources, books, and electronic communications. I've noted all collaboration with fellow students and/or TA's. I did not copy or plagiarize another's work.

> As a Boilermaker pursuing academic excellence, I pledge to be honest and true in all that I do. Accountable together – We are Purdue.