Practice Datasets

May 28, 2024

1 Jack bReacher

- Fake Corporate Dataset
- Purpose: Simulating real-time datasets for fake corporate data breaches.
- Data Handling Skills: Work with simulated/mock objects for better data exploration
- Extensions: Change the distribution, use a different sampling amount, etc.
  - Skies the limit here

[1]:

```python
from CyberHerder import make_pandas_df
```

```
/usr/lib/python3/dist-packages/scipy/__init__.py:146: UserWarning: A NumPy
version >=1.17.3 and <1.25.0 is required for this version of SciPy (detected
version 1.26.4
  warnings.warn(f"A NumPy version >={np_minversion} and <{np_maxversion}"
```

[2]:

```python
print(f'Making a new set \n {make_pandas_df().iloc[:, :1]}')
```

```
16
Making a new set
                                              0
IPv4                                 247.31.0.0
IPv6              51e:aeaf:d505:6:4:49:e40:bfe4
MAC-eui48                     fa-52-e0-ba-35-e8
MAC-eui64               2e-31-2b-19-c9-19-29-29
Phone Num                            9403781620
Current Position                     20-02-2020
Onboarded                            20-03-2010
Postal Code                              579462
UID                                  pHPNKSVgFR
Credit Card                    6811221813603135
GPS                                  (-37,-135)
Email                     HUGHES@sigcryptal.edu
Pin                                      384183
Username                                 GRAHAM
Password                                pokemon
Company                          Liberty-Crypto
Salary (Mean)                      96933.333333
```

1.1 Mario's Candy Shop

- Fake Candy Store Dataset
- Purpose: Simulating real-time datasets for the local business.
- Data Handling Skills: Work with simulated/mock objects for better data exploration
- Extensions: Change the distribution, use a different sampling amount, etc.
    - Skies the limit here
- Note: hard to see but the true mean is slightly different from the simulated mean. Shows that mocks/digital twins are great in the supply chain.

[6]:
```
from marios_candy_shop import Shop, Delicacy
from scipy.stats import norm

import numpy as np
```

[7]:
```
np.random.seed(42)
```

[8]:
```
candy_types = ('Chocolate Bar', 'Gumdrops', 'Gobstopper')
candy_variances = (1, 0.3, 3)
```

[9]:
```
candy_weights = [norm.rvs(4, 1, size=10), norm.rvs(3, 0.3, size=10), norm.
 ↪rvs(5, 3, size=10)]
candy_weights
```

[9]:
```
[array([4.49671415, 3.8617357 , 4.64768854, 5.52302986, 3.76584663,
        3.76586304, 5.57921282, 4.76743473, 3.53052561, 4.54256004]),
 array([2.86097469, 2.86028107, 3.07258868, 2.42601593, 2.48252465,
        2.83131374, 2.69615066, 3.0942742 , 2.72759278, 2.57630889]),
 array([9.39694631, 4.3226711 , 5.20258461, 0.72575544, 3.36685183,
        5.33276777, 1.54701927, 6.12709406, 3.19808393, 4.12491875])]
```

[11]:
```
delicacies = [Delicacy(candy_types[i], candy_variances[i], candy_weights[i][y])␣
 ↪for i in range(3) for y in range(10)]
print(len(delicacies))
print(delicacies[0])
```

```
30
Chocolate Bar weighs 4.496714153011233 and sells for 1
```

[12]:
```
print(f'True mean of sample: {np.mean([x.weight for x in delicacies])}')
```

```
True mean of sample: 3.8484443157718897
```

[13]:
```
# Draw some random sample with replacement and append mean to mean_lengths.
sample_means, sims = [], 1000
for i in range(sims):
    temp_sample = np.random.choice(delicacies, replace=True, size=4)
    sample_means.append(np.mean([x.weight for x in delicacies]))
```

```python
# Calculate bootstrapped mean and 95% confidence interval.
boot_mean = np.mean(sample_means)
boot_95_ci = np.percentile(sample_means, [2.5, 97.5])
print("Bootstrapped Mean Length = {}, 95% CI = {}".format(boot_mean, boot_95_ci))
```

```
Bootstrapped Mean Length = 3.8484443157718906, 95% CI = [3.84844432 3.84844432]
```