Statistics in Python Cheat Sheet

Hamed Ahmadinia, Ph.D.

Hamed.Ahmadinia@metropolia.fi

Statistics in Python

A quick overview of key libraries for statistical analysis:

- **NumPy**: Efficient for numerical computations with arrays.
- **Pandas**: Ideal for working with tabular data using *DataFrames*.
- **SciPy**: Extends NumPy for scientific computations.
- **Statistics module**: A core Python library for basic statistics.

Min and Max

Finding Min/Max:

- **Built-in:** `min(iterable)`, `max(iterable)` - Find smallest/largest values in a list.
  Example: `min([3, 7, 2])` $\rightarrow$ 2, `max([3, 7, 2])` $\rightarrow$ 7
- **NumPy:** `np.min(arr)`, `np.max(arr)` - Optimized for arrays.
  Example: `np.min(np.array([3, 7, 2]))` $\rightarrow$ 2
- **Pandas:** `df.min()`, `df.max()` - Works on DataFrame columns.
  Example: `df['age'].min()` $\rightarrow$ Youngest age

Mean Values

Types of Means:
1. Arithmetic Mean (AM): The sum of all values divided by the count:

$$AM = \frac{\sum_{i=1}^{n} x_i}{n} \tag{1}$$

Example: $AM = \frac{3+4+5}{3} = 4$
2. Geometric Mean (GM): The nth root of the product of all values:

$$GM = \left(\prod_{i=1}^{n} x_i \right)^{\frac{1}{n}} \tag{2}$$

Example: $GM = \sqrt{4 \times 8} = 5.66$
3. Harmonic Mean (HM): The reciprocal of the arithmetic mean of reciprocals:

$$HM = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}} \tag{3}$$

Example: $HM = \frac{3}{\frac{1}{2}+\frac{1}{3}+\frac{1}{4}} = 2.77$
4. Weighted Mean: Each value has an associated weight:

$$WM = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} \tag{4}$$

Example: $WM = \frac{1(0.1)+2(0.3)+3(0.6)}{0.1+0.3+0.6} = 2.5$
Python Implementations:

- **Arithmetic Mean:** `np.mean(data)`
- **Geometric Mean:** `scipy.stats.gmean(data)`
- **Harmonic Mean:** `scipy.stats.hmean(data)`
- **Weighted Mean:** `np.average(data, weights=w)`

Median and Mode

1. Median: The middle value of a sorted dataset.
Formula:

$$\text{Median} = \begin{cases} x_{\frac{n+1}{2}}, & \text{if } n \text{ is odd} \\ \frac{x_{\frac{n}{2}} + x_{\frac{n}{2}+1}}{2}, & \text{if } n \text{ is even} \end{cases} \tag{5}$$

Example: Given data $[3, 1, 5, 7, 9]$, sort it to $[1, 3, 5, 7, 9]$. Median $= 5$.
For even-sized dataset $[3, 1, 5, 7]$, sorted $[1, 3, 5, 7]$, median $= \frac{3+5}{2} = 4$.
Python Code:

- **Using NumPy:** `np.median(data)`
- **Using Statistics Module:** `statistics.median(data)`

2. Mode: The most frequently occurring value.
Example: Data $[1, 2, 2, 3, 4, 4, 4, 5]$, mode $= 4$.
Python Code:

- **Using SciPy:** `scipy.stats.mode(data)`
- **Using Statistics Module:** `statistics.mode(data)`

Quantiles and IQR

1. Quantiles: These divide data into equal-sized subgroups.
Formula for Quantile:

$$Q_p = x_{(n-1)p+1} \tag{6}$$

where p is the quantile position (e.g., 0.25 for Q1, 0.50 for median, 0.75 for Q3).
Example: Given sorted data $[10, 20, 30, 40, 50]$: - $Q_1 = 20$ (25th percentile) - $Q_2 = 30$ (50th percentile, median) - $Q_3 = 40$ (75th percentile)
Python Code:

- **Using NumPy:** `np.quantile(data, [0.25, 0.5, 0.75])`
- **Using Pandas:** `df.quantile([0.25, 0.5, 0.75])`

2. Interquartile Range (IQR): Measures spread of the middle 50% of data.
Formula for IQR:

$$IQR = Q_3 - Q_1 \tag{7}$$

Example: If $Q_3 = 75$ and $Q_1 = 25$, then: $IQR = 75 - 25 = 50$.
Python Code:

- **Using SciPy:** `scipy.stats.iqr(data)`
- **Manual Calculation:** `np.percentile(data, 75) - np.percentile(data, 25)`

3. Boxplot: A graphical representation of quartiles and outliers.
Python Code:

- **Using Matplotlib:** `plt.boxplot(data)`
- **Using Seaborn:** `sns.boxplot(y=data)`

Boxplots

Boxplots: A graphical representation of data distribution using quartiles.

Key Components:

- **Median (Q2)**: The middle value.
- **Interquartile Range (IQR)**: Spread between Q1 (25%) and Q3 (75%).
- **Whiskers**: Extend to the smallest/largest values within 1.5 * IQR.
- **Outliers**: Points outside the whiskers.

Formula for Whiskers:

$$\begin{aligned}\text{Lower Whisker} &= Q_1 - 1.5 \times IQR, \\ \text{Upper Whisker} &= Q_3 + 1.5 \times IQR\end{aligned} \tag{8}$$

Example: Given sorted data $[100, 150, 200, 250, 300, 350, 400]$:

- Q1 = 175, Q2 (Median) = 250, Q3 = 325
- IQR $= Q3 - Q1 = 150$
- Whiskers: $175 - 1.5(150), 325 + 1.5(150)$

Python Code to Generate Boxplot:

```
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

data = np.random.normal(loc=300, scale=50, size=100)
plt.figure(figsize=(8,5))
sns.boxplot(y=data)
plt.title("Boxplot of Sales Data")
plt.show()
```

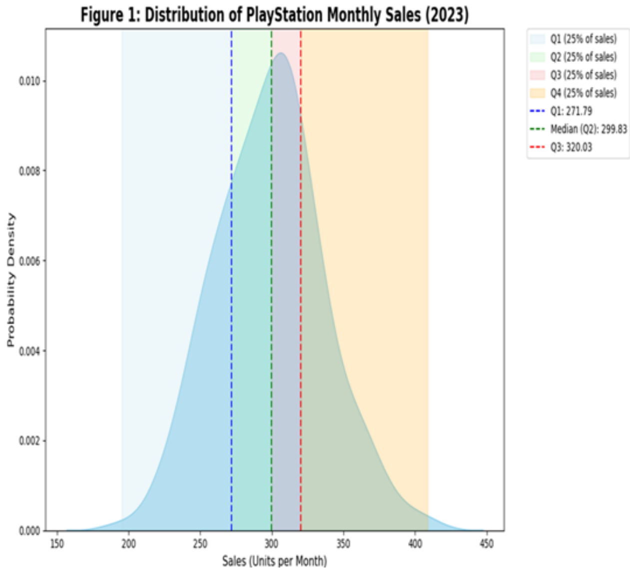

Figure 1: Example of a Boxplot Visualization

Website: ahmadinia.fi