Percentiles, Quartiles, and Boxplots in Data Science

Percentiles and Their Interpretation

A **percentile** indicates the value below which a certain percentage of the data falls.

Mathematical Definition

The **pth percentile** is the value such that p% of the data is less than or equal to that value, and (100-p)% is greater than or equal to it.

Example: The 75th percentile of test scores is 85 points.

- This means 75% of students scored 85 or below
- 25% of students scored 85 or above

Common Percentiles

- **10th percentile (P10)**: 10% of data below, 90% above
- **25th percentile (P25)**: 25% of data below, 75% above
- **50th percentile (P50)**: 50% of data below, 50% above (this is the **median**)
- **75th percentile (P75)**: 75% of data below, 25% above
- **90th percentile (P90)**: 90% of data below, 10% above

Computing Percentiles

Given sorted data, the pth percentile is found by:

```
Position = (p / 100) × (n + 1)
```

Where n is the number of data points.

Step-by-step example:

```
Data: [2, 4, 6, 8, 10, 12, 14, 16, 18, 20]  (n = 10)

Find the 75th percentile:
Position = (75 / 100) × (10 + 1) = 0.75 × 11 = 8.25

This means: between the 8th and 9th values
8th value = 16, 9th value = 18
Interpolate: 16 + 0.25 × (18 - 16) = 16 + 0.5 = 16.5

75th percentile = 16.5
```

This tells us 75% of the data is ≤ 16.5.

Percentile Ranks

The inverse question: "What percentile is this value?"

If a score of 85 is at the 75th percentile, then 75% scored lower.

Example: College admissions

```
SAT score: 1450
Percentile rank: 95th
Interpretation: 95% of test-takers scored below 1450
```

Practical Applications in ML and Data Science

1. Outlier detection:

- Values beyond the 1st or 99th percentile are extreme
- Using percentiles is more robust than using standard deviations

2. Data normalization:

- Percentile rank normalization: x_normalized = percentile_rank(x) / 100
- Maps any data to [0, 1] range, robust to outliers

3. Compression and storage:

- Store P1, P5, P10, ..., P90, P95, P99 instead of full data
- Useful for streaming data where you can't store everything

4. Performance targets:

- "99th percentile response time < 100ms"
- Captures tail behavior (most important for user experience)

5. Data quality monitoring:

- Track if 95th percentile price changes over time
- Detects data drift or anomalies

6. Imputation strategy:

- Fill missing values with median (50th percentile)
- Use percentiles for domain-specific imputation

Quartiles and Interquartile Range (IQR)

Quartiles divide data into four equal parts. They're percentiles at specific points: 25%, 50%, 75%, and 100%.

The Four Quartiles

- **Q0 (Minimum)**: 0th percentile, smallest value
- **Q1 (First Quartile)**: 25th percentile, lower quartile
- **Q2 (Median)**: 50th percentile, middle value
- **Q3 (Third Quartile)**: 75th percentile, upper quartile
- **Q4 (Maximum)**: 100th percentile, largest value

Each quartile contains approximately 25% of the data.

Computing Quartiles

Example with 12 data points:

```
Data (sorted): [2, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25]

Q0 (Min) = 2
Q1 (25th percentile) = 25% position = 0.25 × 13 = 3.25
  Between 3rd value (7) and 4th value (9)
  Q1 = 7 + 0.25 × (9 - 7) = 7 + 0.5 = 7.5

Q2 (Median) = 50% position = 0.5 × 13 = 6.5
  Between 6th value (13) and 7th value (15)
  Q2 = 13 + 0.5 × (15 - 13) = 14

Q3 (75th percentile) = 75% position = 0.75 × 13 = 9.75
  Between 9th value (19) and 10th value (21)
  Q3 = 19 + 0.75 × (21 - 19) = 19 + 1.5 = 20.5

Q4 (Max) = 25
```

Interquartile Range (IQR)

The **Interquartile Range** is the range of the middle 50% of data:

```
IQR = Q3 - Q1
```

From the example:

```
IQR = 20.5 - 7.5 = 13
```

This means 50% of the data falls within a range of 13 units.

Why IQR is Important

Robust to outliers:

- IQR only depends on Q1 and Q3 (ignores extremes)
- Doesn't change if you add extreme values
- Better than range for understanding typical spread

Example:

```
Dataset 1: [2, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25]
IQR = 13

Dataset 2: [2, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 1000]
Q1 = 7.5, Q3 = 20.5 (same as before!)
IQR = 13 (unchanged!)
```

Adding an extreme value (1000) doesn't change IQR, but it would dramatically change the range (2 to 1000).

Applications in ML

1. Outlier detection (IQR method):

```
Lower fence = Q1 - 1.5 × IQR
Upper fence = Q3 + 1.5 × IQR

Points outside these fences are potential outliers
```

Example:

```
Q1 = 7.5, Q3 = 20.5, IQR = 13
Lower fence = 7.5 - 1.5 × 13 = 7.5 - 19.5 = -12
Upper fence = 20.5 + 1.5 × 13 = 20.5 + 19.5 = 40

Any value < -12 or > 40 is a potential outlier
```

2. Data validation:

- Flag records with values outside expected IQR ranges
- Detect data entry errors automatically

3. Robust scaling:

```
x_scaled = (x - median) / IQR
```

Scale data using median and IQR instead of mean and std dev (more robust to outliers).

4. Anomaly detection:

- Real-time systems can track IQR of metrics
- Values far outside IQR indicate anomalies

Five Number Summary: Minimum, Q1, Median, Q3, Maximum

The **Five Number Summary** provides a complete picture of data distribution in five values:

```
Five Number Summary = [Minimum, Q1, Median, Q3, Maximum]
```

Or: **[Q0, Q1, Q2, Q3, Q4]**

Example

```
Data: [2, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25]

Five Number Summary:
- Minimum (Q0) = 2
- Q1 = 7.5
- Median (Q2) = 14
- Q3 = 20.5
- Maximum (Q4) = 25
```

What It Tells Us

Minimum and Maximum:

- Range of data: from 2 to 25
- Identify potential outliers at extremes

Q1 and Q3:

- Where the middle 50% of data lies: from 7.5 to 20.5
- Shows typical spread (IQR = 13)

Median:

- Center of distribution: 14
- 50% of data below, 50% above

Completeness

The Five Number Summary is "complete" in that it captures:

- **Location**: Where is the data? (median)
- **Spread**: How much variation? (IQR, range)

- **Skewness**: Is it symmetric?
  - If (Q2 - Q1) ≈ (Q3 - Q2): symmetric
  - If (Q2 - Q1) < (Q3 - Q2): right-skewed
  - If (Q2 - Q1) > (Q3 - Q2): left-skewed

Example from our data:

```
Q2 - Q1 = 14 - 7.5 = 6.5
Q3 - Q2 = 20.5 - 14 = 6.5
They're equal → symmetric distribution
```

Advantages Over Descriptive Statistics

Aspect	Mean/Std Dev	Five Number Summary
Robustness to outliers	Sensitive	Robust
Complete picture	No	Yes
Visual representation	Difficult	Easy (boxplot)
Nonparametric	No	Yes
Standard tests compatible	Yes	Limited

Nonparametric: The Five Number Summary works for any distribution shape, doesn't assume normality.

Boxplot as a Visual Representation of Distribution

A **boxplot** (box-and-whisker plot) visualizes the Five Number Summary graphically.

Components of a Boxplot

```
    *  ← Outlier (beyond upper fence)
    |
  ┌────┐
  │ □ │ ← Q3 (75th percentile)
  │   │
  │ ═ │ ← Median (Q2, 50th percentile)
  │   │
  └────┘
    │ ← Q1 (25th percentile)
──────┴────── ← Whiskers extend to Q1 - 1.5×IQR and Q3 + 1.5×IQR
```

|
◦ ← Outlier (beyond lower fence)

Detailed Explanation

The Box (the rectangle):

- **Bottom edge** = Q1 (25th percentile)
- **Middle line** = Median (Q2)
- **Top edge** = Q3 (75th percentile)
- Represents the **interquartile range (IQR)**: where 50% of data lies

The Whiskers (lines extending from box):

- **Lower whisker**: Extends from Q1 down to Q1 - 1.5×IQR
- **Upper whisker**: Extends from Q3 up to Q3 + 1.5×IQR
- Show "typical" data range (excluding outliers)

Outliers (individual points):

- Points beyond the whiskers (outside the fences)
- Plotted individually as dots or asterisks
- Q1 - 1.5×IQR to Q3 + 1.5×IQR is the "normal range"

Boxplot Construction Example

Given our data:

```
Data: [2, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25]

Q1 = 7.5
Median = 14
Q3 = 20.5
IQR = 13

Lower fence = 7.5 - 1.5 × 13 = -12
Upper fence = 20.5 + 1.5 × 13 = 40

Whisker endpoints:
- Lower whisker: max(minimum, lower fence) = max(2, -12) = 2
- Upper whisker: min(maximum, upper fence) = min(25, 40) = 25

No outliers (all data falls between fences)
```

Boxplot visualization:

```
 2 ──────┐
    │
 7.5 ┌──┴──┐
    │ 14 │  (median line)
20.5 └──┬──┘
    │
25 ─────┘
```

Interpreting Distribution Shape from Boxplot

Symmetric distribution:

```
    Median line is centered in box
    Whiskers are roughly equal length


    ─────────┐
       │
     ┌──────┼──────┐
     │   │   │
     └──────┼──────┘
       │
    ───────┘
```

Right-skewed distribution:

```
Median closer to Q1
Upper whisker longer than lower
More outliers above


───────┐
   │
 ┌───────┴───┐
 │  │   │
 └───┼───────┘
   │ ────────┐
   └───────┘ (outliers)
```

Left-skewed distribution:

```
Median closer to Q3
Lower whisker longer than upper
More outliers below


 ┌───────┐
 │ │ │ (outliers)
┌────┴───┴───┘
│ │  │
└───┼──────┘
  │
  └───────
```

Creating Multiple Boxplots for Comparison

Boxplots are most powerful when comparing groups:

```
Group A  Group B  Group C
  │      │      │
*  │      │      │
 │ │   ┌────┤    ┌────┤
┌────┼───┤  │  │  │ │
│ │ │  │  │  │ │
└────┼───┤  │  │  │ │
 │ │   └────┘    └───┤
 │ │      ┌──────┘
 │ │     *  │
  │       │
```

Visual insights:

- Group A has higher median than B and C
- Group C has more spread (larger IQR)
- Group A has outliers
- Group B is more compact

Practical Applications in Data Science

1. Data Exploration (EDA)

Step 1: Create boxplots for each numerical feature

```
boxplot(dataset.numerical_features)
```

Reveals:

- Which features have outliers
- Which are skewed
- Which have larger/smaller spreads
- Potential data quality issues

2. Feature Comparison Across Groups

Compare distributions by category:

```
Salary boxplot by Department:
         HR   Sales   IT   Finance
          |      |      |      |
       [box1]  [box2] [box3] [box4]
```

Insights:

- Which departments have higher salaries?
- Are there outliers in specific departments?
- Which departments have more variable pay?

3. Before/After Comparison

Compare treatment effects:

```
Blood Pressure Before vs After Treatment:

  Before      After
    |           |
  [box1]      [box2]
```

If the "After" box is lower, treatment is effective.

4. Outlier Detection and Cleaning

Use the boxplot to identify outliers:

```
Lower fence = Q1 - 1.5 × IQR
Upper fence = Q3 + 1.5 × IQR

Outliers = values outside [lower_fence, upper_fence]
```

Decision:

- Remove them (if data error)
- Transform them (log scale)
- Keep them (if legitimate extremes)
- Analyze separately (if important subgroup)

5. Quality Control in Manufacturing

Monitor production metrics:

```
Product Weight (in grams)
Target: 100g ± acceptable range

Boxplot of 1000 units:
  100.5 ├─┐
        │ □ │ (median)
  99.5  └─┘

Points outside fences → investigate production issue
```

6. Portfolio Risk Management

Compare investment returns:

```
Stock Returns (Annual %)

  Apple   Google   Tesla
    |       |        |
 [box1]  [box2]   [box3]
```

Tesla has longer whiskers → more volatile, higher risk.

Comparing Boxplots with Other Visualizations

Visualization	Best For	Limitations
Boxplot	Comparing distributions, identifying outliers	Hides data shape (bimodal not visible)
Histogram	Seeing exact shape of distribution	Hard to compare groups
Density plot	Smooth distribution shape	Can be misleading with small samples
Violin plot	Combining boxplot + density shape	More complex to interpret
Scatter plot	Raw data points	Overplotting when many points

The Ideal Approach

Use **multiple visualizations**:

1. **Boxplot**: Quick summary, identify outliers, compare groups
2. **Histogram**: See exact shape, detect multimodality
3. **Density plot**: Smooth distribution for larger datasets

Together they provide complete understanding.

Real-World Example: Website Response Time Analysis

Scenario: Your website has variable response times. You want to understand the distribution and identify problems.

Raw Data Summary

```
Response times (milliseconds):
[50, 55, 60, 65, 70, 75, 80, 85, 90, 100, 110, 150, 180, 200, 5000]
```

Computing Five Number Summary

```
Min = 50 ms
Q1 = 67.5 ms (25% of requests are faster)
Median = 85 ms (typical request)
Q3 = 147.5 ms (75% of requests are faster)
Max = 5000 ms (one extremely slow request)

IQR = 147.5 - 67.5 = 80 ms

Outlier fences:
Lower = 67.5 - 1.5 × 80 = -52.5 (irrelevant, all positive)
Upper = 147.5 + 1.5 × 80 = 267.5

Outliers: 5000 ms (beyond 267.5)
```

Boxplot Visualization

```
5000 *  ← Outlier (slow request, investigate!)
    |
200 |
    |
150 ┌──────┐
100 |    |
 85 |──|  ← Median
 70 |    |
 50 └──────┘
```

Insights and Actions

Findings:

- Typical request: 85 ms (good)
- 75% of requests: 50-148 ms (acceptable)
- One request: 5000 ms (problematic)

Actions:

- Investigate the 5000ms request (database issue? stuck process?)
- Track the 95th percentile response time: should be ~180 ms
- Set SLA: "99% of requests < 200 ms, 95% < 150 ms"

This is how **percentiles guide business decisions**.

Summary: From Five Numbers to Understanding

The progression from raw data to understanding:

1. **Percentiles**: Understand where specific values rank
2. **Quartiles**: Divide data into four comparable pieces
3. **IQR**: Measure typical spread robustly
4. **Five Number Summary**: Complete picture in five values
5. **Boxplot**: Visual representation for communication

These tools are:

- **Robust**: Resistant to outliers (unlike mean and std dev)
- **Intuitive**: Easy to understand and explain
- **Complete**: Capture location, spread, and shape
- **Practical**: Directly used for outlier detection, quality control, and decision-making

Master percentiles and boxplots, and you can communicate data insights clearly and make data-driven decisions confidently.