📚 Module 32-34: Data Analysis & Distribution - Complete Notes

🎯 What You'll Learn

Master **outlier detection, distribution shapes, and data visualization** — essential for data quality and analysis.

📖 Concept Explained

Outliers

Values that are unusually far from the rest.

```
Data: [10, 12, 14, 15, 18, 100]
                              ↑
                        OUTLIER (doesn't fit)

Detection Methods:
- IQR: < Q1-1.5*IQR or > Q3+1.5*IQR
- Z-Score: |z| > 3 (more than 3 std devs from mean)
```

Distribution Shapes

```
NORMAL (Bell curve):
    Mean ≈ Median
    Symmetric

RIGHT-SKEWED:
    Mean > Median
    Long tail on right (high outliers)
    Example: Response times

LEFT-SKEWED:
    Mean < Median
    Long tail on left (low outliers)
```

💻 Programming Connection

Code Examples

```python
# Example 1: Outlier Detection - Z-Score

from statistics import mean, stdev

def find_outliers_zscore(data, threshold=3):
    """Find outliers using z-score method"""
    avg = mean(data)
```

```
    std = stdev(data)

    outliers = []
    for x in data:
        z = (x - avg) / std if std > 0 else 0
        if abs(z) > threshold:
            outliers.append({"value": x, "z_score": round(z, 2)})

    return outliers

data = [100, 102, 98, 105, 500, 101]
print(find_outliers_zscore(data))
```

```
# Example 2: Impact of Outliers

def outlier_impact(data):
    """Show how outliers affect statistics"""
    from statistics import median

    avg_with = mean(data)
    med_with = median(data)

    # Remove outliers
    clean = [x for x in data if x not in find_outliers_zscore(data)]

    return {
        "original_mean": round(avg_with, 2),
        "original_median": med_with,
        "clean_mean": round(mean(clean), 2) if clean else None,
        "clean_median": median(clean) if clean else None,
        "outliers_removed": len(data) - len(clean)
    }

print(outlier_impact([100, 102, 98, 105, 500, 101]))
```

```
# Example 3: Detect Distribution Shape

def identify_distribution(data):
    """Identify if data is skewed"""
    from statistics import median

    avg = mean(data)
    med = median(data)

    diff = avg - med

    if abs(diff) / med < 0.05:
        shape = "symmetric"
    elif diff > 0:
        shape = "right-skewed"
```

```
    else:
        shape = "left-skewed"

    return {
        "mean": round(avg, 2),
        "median": med,
        "shape": shape,
        "recommendation": "Use median" if shape != "symmetric" else "Mean is fine"
    }

# Right-skewed (response times often look like this)
print(identify_distribution([50, 55, 60, 65, 70, 150, 200]))
```

```
# Example 4: Text-Based Distribution Visualization

def text_histogram(data, bins=5):
    """Create simple text histogram"""
    min_val, max_val = min(data), max(data)
    bin_width = (max_val - min_val) / bins

    counts = [0] * bins
    for x in data:
        idx = min(int((x - min_val) / bin_width), bins - 1)
        counts[idx] += 1

    max_count = max(counts)

    for i, count in enumerate(counts):
        low = min_val + i * bin_width
        high = low + bin_width
        bar = '█' * int(count / max_count * 20)
        print(f"{low:6.1f}-{high:6.1f} {bar} ({count})")

import random
data = [random.gauss(100, 15) for _ in range(100)]
text_histogram(data, 5)
```

```
# Example 5: Five-Number Summary

def five_number_summary(data):
    """Min, Q1, Median, Q3, Max"""
    sorted_d = sorted(data)
    n = len(sorted_d)

    return {
        "min": sorted_d[0],
        "Q1": sorted_d[n // 4],
        "median": sorted_d[n // 2],
        "Q3": sorted_d[3 * n // 4],
        "max": sorted_d[-1]
```

```
    }

data = list(range(10, 101, 5))
print(five_number_summary(data))
```

🧪 SDET/Testing Application

```
# SDET Scenario: Performance Anomaly Detection

def detect_performance_anomaly(current, history):
    """Check if current value is anomalous"""
    avg = mean(history)
    std = stdev(history)

    z = (current - avg) / std if std > 0 else 0
    is_anomaly = abs(z) > 2

    return {
        "current": current,
        "history_mean": round(avg, 2),
        "z_score": round(z, 2),
        "anomaly": is_anomaly,
        "status": "🚨 ANOMALY" if is_anomaly else "✅ Normal"
    }

history = [100, 102, 98, 105, 101, 99, 103, 97]
print(detect_performance_anomaly(150, history))  # 🚨 ANOMALY
print(detect_performance_anomaly(105, history))  # ✅ Normal
```

```
# SDET Scenario: Data Quality Report

def data_quality_report(data, label="Data"):
    """Comprehensive data quality analysis"""
    outliers = find_outliers_zscore(data, threshold=2)
    dist = identify_distribution(data)
    summary = five_number_summary(data)

    return {
        "label": label,
        "count": len(data),
        "summary": summary,
        "distribution": dist["shape"],
        "outliers_count": len(outliers),
        "outliers": [o["value"] for o in outliers],
        "data_quality": "Good" if len(outliers) < len(data) * 0.05 else "Review
needed"
    }
```

```
response_times = [100, 105, 110, 95, 102, 500, 98, 107]
print(data_quality_report(response_times, "Response Times"))
```

🔑 Key Takeaways

✅ **Outliers distort mean** — Use median for skewed data
✅ **IQR method** — Q1 - 1.5×IQR to Q3 + 1.5×IQR
✅ **Z-score > 3** — Unusual value
✅ **Right-skewed** — Mean > Median (high outliers)
✅ **Don't blindly remove outliers** — Investigate first!

💾 *Save as:* `Module_32_34_Analysis_Distribution.md`

Phase 6 Complete! 🔥🚀