DAY 11

Statistical Analysis & ML Preparation

with PySpark

Databricks 14-Days AI Challenge

Agenda

- **Introduction**
- **Descriptive Statistics**
  - Measures of Central Tendency
  - Measures of Dispersion
- **Hypothesis Testing**
  - P-Value & Significance
  - Statistical Tests
- **A/B Test Design**
  - Sample Size Calculation
  - Implementation
- **Correlation Analysis**
- **Feature Engineering**
- **Best Practices**

Introduction

Why Statistical Analysis & ML Preparation?

Critical steps in the data science pipeline:

- **Understand Data:** Through descriptive statistics
- **Validate Assumptions:** Through hypothesis testing
- **Transform Data:** Through feature engineering

1. Descriptive Statistics

Understanding Your Data

Measures of Central Tendency

Mean (Arithmetic Average)

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Median

- Middle value when data is sorted
- Robust to outliers

Mode

- Most frequently occurring value

When to Use Each

Measure	**Best For**	**Outliers**
Mean	Normal dist.	Sensitive
Median	Skewed data	Robust
Mode	Categorical	Robust

Measures of Dispersion

Variance

$$s^2 = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2$$

Standard Deviation

$$s = \sqrt{s^2}$$

IQR (Interquartile Range)

$$IQR = Q3 - Q1$$

Skewness

- > 0: Right-skewed
- $= 0$: Symmetric
- < 0: Left-skewed

Kurtosis

- > 0: Heavy tails
- $= 0$: Normal-like
- < 0: Light tails

Descriptive Statistics in PySpark

```
from pyspark.sql import functions as F

# Basic descriptive statistics
events.describe(["price"]).show()

# Extended statistics with percentiles
events.select(
    F.count("price").alias("count"),
    F.mean("price").alias("mean"),
    F.stddev("price").alias("std_dev"),
    F.expr("percentile_approx(price, 0.25)").alias("Q1"),
    F.expr("percentile_approx(price, 0.50)").alias("median"),
    F.expr("percentile_approx(price, 0.75)").alias("Q3"),
    F.skewness("price").alias("skewness"),
    F.kurtosis("price").alias("kurtosis")
).show()

# Detect outliers using IQR method
lower_bound = Q1 - 1.5 * iqr
upper_bound = Q3 + 1.5 * iqr
```

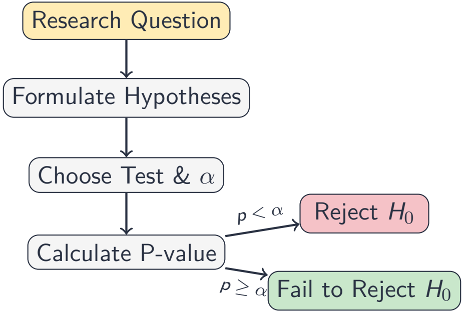
2. Hypothesis Testing

Making Data-Driven Decisions

Understanding Hypothesis Testing

Key Question

Is the observed pattern in the sample likely to exist in the population, or is it just due to random sampling variation?



Types of Hypotheses

Null Hypothesis (H_0)

- Default assumption
- No effect/difference
- Status quo

Alternative Hypothesis (H_1)

- What we're testing for
- Contradicts H_0

Test Types

Type	**Alternative**
Two-tailed	$\mu \neq \mu_0$
Left-tailed	$\mu < \mu_0$
Right-tailed	$\mu > \mu_0$

P-Value and Significance Level

Significance Level (α)

- $\alpha = 0.05$ (5%): Most common
- $\alpha = 0.01$ (1%): More stringent
- $\alpha = 0.10$ (10%): Exploratory

P-Value

$$P(\text{Observed}|H_0 \text{ is true})$$

Decision Rule

- $p < \alpha$: Reject H_0
- $p \geq \alpha$: Fail to reject H_0

Example:
If $p = 0.03$ and $\alpha = 0.05$:
$\rightarrow$ Reject H_0 (significant)

Types of Errors

Decision	H_0 **True**	H_0 **False**
Reject H_0	**Type I (α)** False Positive	✓Correct (Power)
Fail to Reject	✓Correct	**Type II (β)** False Negative

Statistical Power

$$\text{Power} = 1 - \beta$$

Factors Affecting Power:

- Sample size $\uparrow$
- Effect size $\uparrow$
- Variance $\downarrow$

Common Statistical Tests

Z-Test

$$z = \frac{\bar{x} - \mu_0}{\sigma/\sqrt{n}}$$

- Known population σ
- Large sample ($n > 30$)

T-Test

$$t = \frac{\bar{x} - \mu_0}{s/\sqrt{n}}$$

- Unknown population σ

Chi-Square Test

$$\chi^2 = \sum \frac{(O_i - E_i)^2}{E_i}$$

- Categorical data
- Independence tests

Two-Sample T-Test

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{s_1^2}{n_1} + \frac{s_2^2}{n_2}}}$$

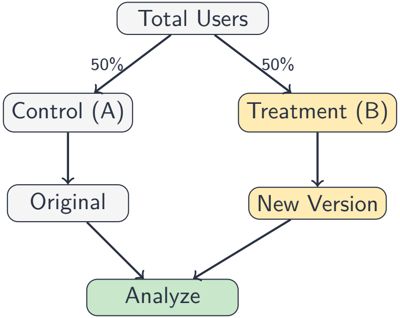
3. A/B Test Design

Controlled Experiments

What is A/B Testing?

Definition

A controlled experiment comparing two or more variants to determine which performs better.



Key Metrics & Parameters

Metric Types

- ▷ **Primary:** Main success indicator
- ▷ **Secondary:** Supporting metrics
- ▷ **Guardrail:** Must not worsen

Key Parameters

- ▷ Baseline rate (p_0)
- ▷ MDE (Min. Detectable Effect)
- ▷ Significance (α): 0.05
- ▷ Power ($1-\beta$): 0.80

Sample Size Calculation

Formula for Two-Proportion Z-Test:

$$n = \frac{2 \cdot (Z_{\alpha/2} + Z_{\beta})^2 \cdot \bar{p}(1 - \bar{p})}{(p_1 - p_0)^2}$$

Example Calculation

Given: $p_0 = 0.10$, MDE $= 0.02$, $\alpha = 0.05$, Power $= 0.80$

- $\bar{p} = \frac{0.10+0.12}{2} = 0.11$
- $n = \frac{2\times(1.96+0.84)^2\times0.11\times0.89}{0.0004} \approx 3,838$ per group

Total: $\sim$7,676 users needed

Common Pitfalls

Pitfall	Problem	Solution
Peeking	Inflated false positives	Pre-commit to duration
Multiple tests	Increased Type I errors	Bonferroni correction
Simpson's Paradox	Misleading aggregates	Segment analysis
Novelty effect	Temporary engagement	Run longer

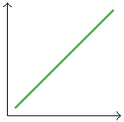
4. Correlation Analysis

Measuring Relationships

Types of Correlation

Positive

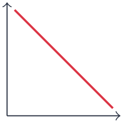
$r > 0$



$X \uparrow \Rightarrow Y \uparrow$

Negative

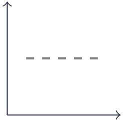
$r < 0$



$X \uparrow \Rightarrow Y \downarrow$

No Correlation

$r \approx 0$



No linear relation

Pearson Correlation Coefficient

Formula:

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2} \cdot \sqrt{\sum(y_i - \bar{y})^2}}$$

r Value	Interpretation
0.90 – 1.00	Very strong
0.70 – 0.89	Strong
0.50 – 0.69	Moderate
0.30 – 0.49	Weak
0.00 – 0.29	Very weak

Coefficient of Determination: r^2 = variance explained

Correlation in PySpark

```python
from pyspark.ml.stat import Correlation
from pyspark.ml.feature import VectorAssembler

# Method 1: Two columns
correlation = events.stat.corr("price", "conversion_rate")

# Method 2: Correlation matrix
numeric_columns = ["price", "quantity", "discount", "revenue"]

assembler = VectorAssembler(inputCols=numeric_columns, outputCol="features")
vector_df = assembler.transform(events.na.drop(subset=numeric_columns))

correlation_matrix = Correlation.corr(vector_df, "features", "pearson").head()
corr_array = correlation_matrix[0].toArray()

# Print correlation matrix
for i, col in enumerate(numeric_columns):
    print(f"{col}", end=" ")
    for j in range(len(numeric_columns)):
        print(f"{corr_array[i][j]:.3f}", end=" ")
```

5. Feature Engineering

Transforming Raw Data for ML

Types of Feature Engineering

Type	Description	Examples
Temporal	Time-based patterns	Hour, day of week, season
Aggregation	Summarize data	Total purchases, avg order
Transformation	Change distribution	Log, power, binning
Interaction	Combine features	Price $\times$ Quantity, ratios
Window	Rolling metrics	Moving avg, cumulative
Encoding	Categorical to numeric	One-hot, label encoding

Numerical Transformations

Log Transformation

$$x' = \log(x + 1)$$

- Handles skewed distributions
- Reduces outlier impact

Standardization (Z-score)

$$z = \frac{x - \mu}{\sigma}$$

Min-Max Scaling

$$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$$

- Scales to [0, 1] range

Square Root

$$x' = \sqrt{x}$$

Feature Engineering in PySpark

```python
from pyspark.sql import functions as F
from pyspark.sql.window import Window

# Temporal features
features = events.withColumn("hour", F.hour("event_time")) \
    .withColumn("day_of_week", F.dayofweek("event_date")) \
    .withColumn("is_weekend", F.dayofweek("event_date").isin([1, 7]).cast("int"))

# Log transformation
features = features.withColumn("price_log", F.log(F.col("price") + 1))

# Window features
user_window = Window.partitionBy("user_id").orderBy("event_time")
features = features.withColumn("event_sequence", F.row_number().over(user_window))

# Aggregation features
user_agg = events.groupBy("user_id").agg(
    F.count("*").alias("total_events"),
    F.avg("price").alias("avg_price"),
    F.sum(F.when(F.col("event_type") == "purchase", 1).otherwise(0)).alias("purchases")
)
```

Window Function Templates

Common Window Patterns:

Pattern	Use Case
`partitionBy().orderBy()`	Basic ordering within groups
`rowsBetween(-4, 0)`	Last 5 rows (rolling)
`unboundedPreceding, currentRow`	Cumulative calculations
`rangeBetween(-86400, 0)`	Last 24 hours (time-based)

Common Functions:

- `row_number()`, `rank()`, `lag()`, `lead()`
- `sum()`, `avg()`, `first()`, `last()`

6. Best Practices

& Quick Reference

Statistical Analysis Best Practices

- **Check Assumptions:** Verify normality before parametric tests
- **Use Appropriate Tests:** Match test to data type
- **Multiple Testing Correction:** Apply Bonferroni or FDR
- **Report Effect Size:** Not just p-values
- **Confidence Intervals:** Always include them
- **Sample Size:** Ensure adequacy before testing

Feature Engineering Best Practices

- **Domain Knowledge:** Create meaningful features
- **Avoid Data Leakage:** No future information
- **Handle Missing Values:** Impute appropriately
- **Scale Features:** Normalize for sensitive algorithms
- **Remove Redundancy:** Drop correlated features
- **Validate Transformations:** Check distributions

Common Pitfalls

- Data leakage using test data in training
- Overfitting with too many features
- Ignoring time order in temporal data

Quick Reference: Key Formulas

Metric	**Formula**
Mean	$\bar{x} = \frac{\sum x_i}{n}$
Variance	$s^2 = \frac{\sum (x_i - \bar{x})^2}{n-1}$
Pearson Correlation	$r = \frac{\text{Cov}(X,Y)}{\sigma_X \cdot \sigma_Y}$
Z-Score	$z = \frac{x-\mu}{\sigma}$
T-Statistic	$t = \frac{\bar{x}-\mu_0}{s/\sqrt{n}}$
A/B Sample Size	$n = \frac{2(Z_{\alpha/2}+Z_\beta)^2 \bar{p}(1-\bar{p})}{(p_1-p_0)^2}$

PySpark Statistical Functions

Function	Description	Example
`F.mean()`	Average	`df.select(F.mean("col"))`
`F.stddev()`	Std deviation	`df.select(F.stddev("col"))`
`F.variance()`	Variance	`df.select(F.variance("col"))`
`F.skewness()`	Skewness	`df.select(F.skewness("col"))`
`F.kurtosis()`	Kurtosis	`df.select(F.kurtosis("col"))`
`stat.corr()`	Correlation	`df.stat.corr("col1", "col2")`
`describe()`	Summary	`df.describe(["col"])`

Resources

- **Apache Spark ML Guide**
  https://spark.apache.org/docs/latest/ml-guide.html
- **Databricks Documentation**
  https://docs.databricks.com/
- **Spark by Examples**
  https://sparkbyexamples.com/
- **Databricks ML Lifecycle**
  https://docs.databricks.com/machine-learning/index.html
- **A/B Testing Guide**
  https://www.optimizely.com/optimization-glossary/ab-testing/

Thank You!

Day 11 Complete

Statistical Analysis & ML Preparation