INDIAN DATA CLUB

databricks

11

Statistical Analysis & ML Prep

Day 11 Challenge: From Raw Events to Feature Engineering in PySpark

14

14 DAYS AI CHALLENGE

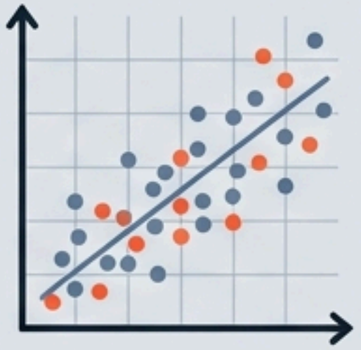
CODE BASICS

The Mission: Four Steps to Data Maturity

01. Understand

02. Investigate

03. Map



04. Prepare

01. Understand
Summary Statistics

02. Investigate
Hypothesis Testing

03. Map
Correlation Analysis

04. Prepare
Feature Engineering

Sanity Check: Understanding Data Distribution

Before modeling, we must quantify the dataset. We utilize the PySpark 'describe()' method on the 'price' column to extract key distributional metrics. Our goal is to identify scale, central tendency, and variance.

Execute

```
# Descriptive stats for price column
events.describe(['price']).show()
```

Action: Compute Count, Mean, StdDev, Min, Max.

The Scale of the Data

Output Results

```
events.describe(["price"]).show()
+---------+------------+
| summary |      price |
+---------+------------+
| count   | 67,501,979 |
| mean    |     292.46 |
| stddev  |     355.67 |
| min     |        0.0 |
| max     |    2574.07 |
+---------+------------+
```

Insight: High Variance detected.
The Standard Deviation (355.67) exceeds the Mean (292.46), indicating a highly skewed distribution. The dataset is massive: 67.5 Million records.

The Hypothesis: Behavior by Day

Do customers spend more on weekends than on weekdays?

Step 1

Extract day_of_week

Step 2

Flag Weekend vs. Weekday

Step 3

Compare Average Price

Implementing the Logic

PySpark Date Function

```
# events_with_day function
events_with_day = events.withColumn('day_of_week', F.dayofweek('event_time'))

# events_with_day comment
events_with_day = events_with_day.withColumn(
    'is_weekend',
    F.when(F.col('day_of_week').isin([1,7]), 'weekend').otherwise('weekday')
)
```

1 = Sunday,
7 = Saturday

The Verdict: Weekend vs. Weekday

WEEKEND

Mean Price:

$294.68

JetBrains Mono, Vermilion Orange

Count: ~24.7 Million events

WEEKDAY

Mean Price:

$291.17

JetBrains Mono, Cool Grey

Count: ~42.7 Million events

Hypothesis Confirmed: A slight observable increase in average spending occurs on weekends.

Mapping Relationships: Price vs. Category

Correlation Analysis

```
// Check for linear relationship
events.stat.corr('price', 'category_id')
```

The Query: Is there a linear relationship between the product category and the price? We use the Pearson Correlation Coefficient to find out.

Interpreting the Signal

-0.05718

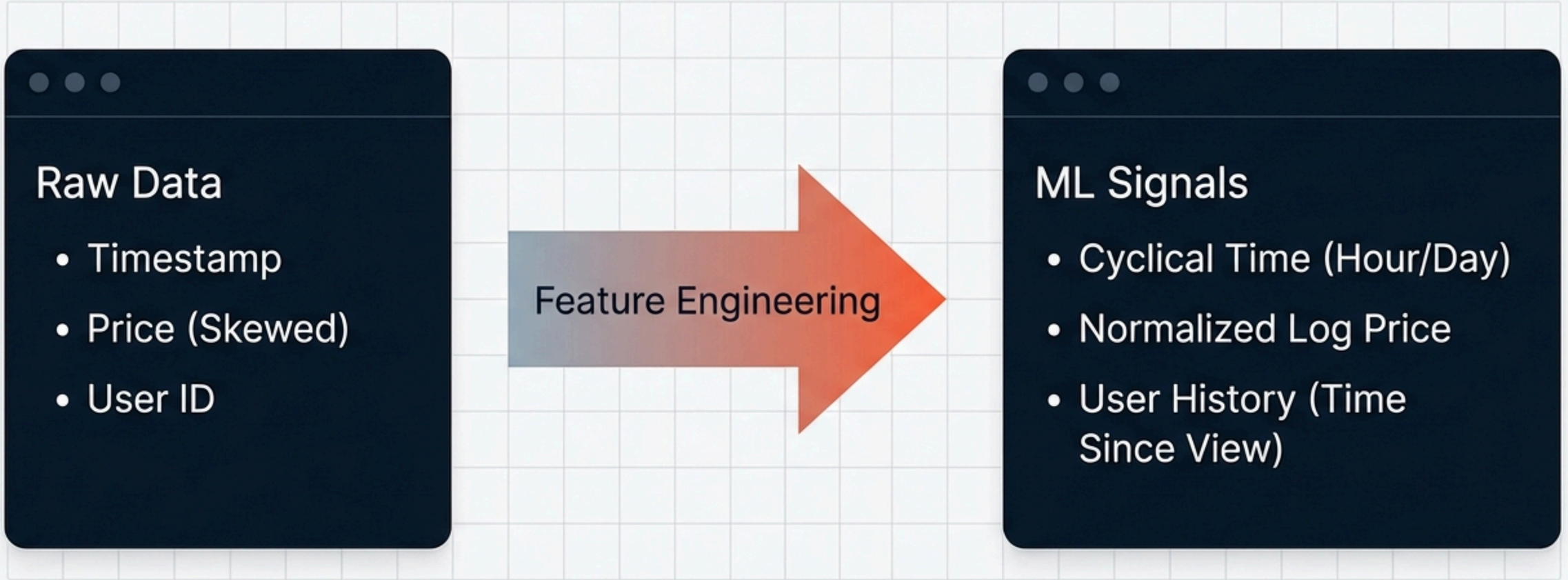
-0.05718421046857468

Result: Near Zero.

Interpretation: Effectively no linear correlation.

Reasoning: “category_id” is a nominal variable (random identifiers), not ordinal. A linear check is expected to fail.

From Raw Data to Model Features



Machine Learning models need descriptive signals, not just raw logs. We transform the data to capture patterns.

Capturing Cyclical Patterns

Hour of Day (0-23)

Day of Week (1-7)

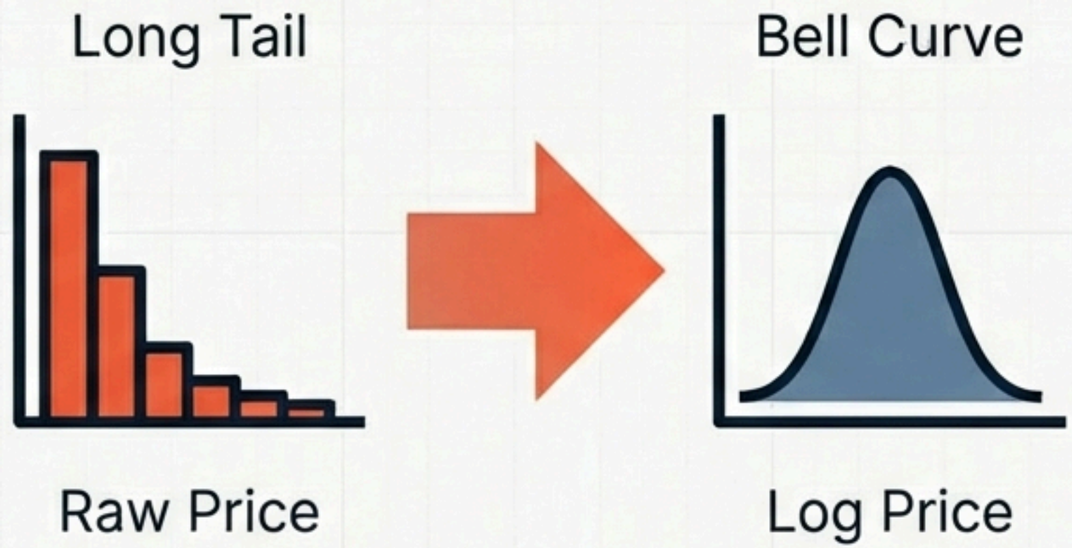
```
.withColumn('hour',
F.hour('event_time'))
.withColumn('day_of_week',
F.dayofweek('event_time'))
```

Extracts temporal context independent of the specific date, allowing the model to learn daily and weekly habits.

Normalizing the Target: Log Price

```
.withColumn('price_log',
F.log(F.col('price')+1))
```

Log transformation compresses the scale of variables with high variance (StdDev ~355). We add +1 to safely handle zero-values.



Advanced Feature: User Interaction History

```
.withColumn('time_since_first_view', \
    F.unix_timestamp('event_time') - \
    F.first('event_time').over(Window.partitionBy('user_id').orderBy
)
```

Group by User
Partitions the data to analyze each user's interactions independently.

Chronological Order
Sorts events within each user partition by time.

Find first-ever interaction
Identifies the timestamp of the user's initial event in the window.

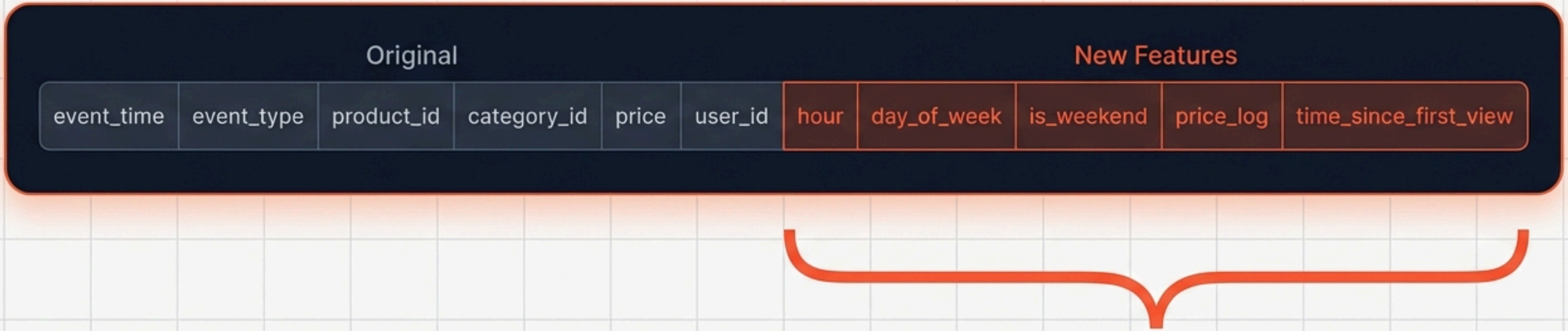
Calculate seconds elapsed
Computes the time difference between the current event and the first event in seconds.

Find first-ever interaction
Identifies the timestamp of the user's initial event in the window.

Reference: This complex feature engineering uses a Window Function to define the user scope and order.

The ML-Ready Dataset

Final Dataframe Schema



Engineered Inputs

Enriched with temporal, behavioral, and normalized features.

Challenge Complete: Day 11 Summary

Scale ✓

Verified 67M+ records & high variance.

Insight ✓

Confirmed Weekend spending lift (~$3).

Stats ✓

Disproved linear correlation with Category ID.

Eng ✓

Built complex features via Window functions.

READY FOR TRAINING

INDIAN DATA CLUB