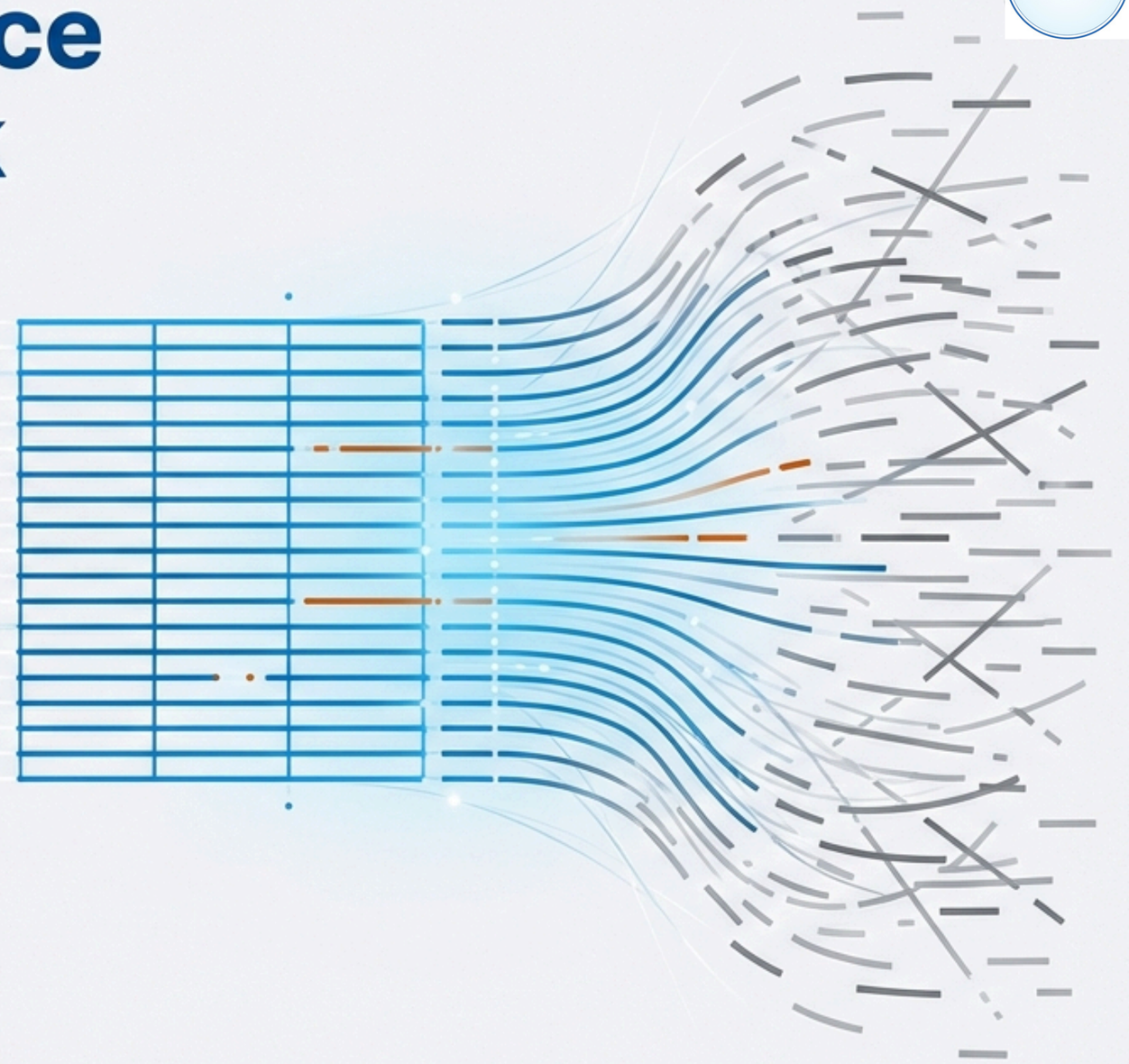
Unlocking E-Commerce Insights with PySpark

A workflow for extracting actionable intelligence from raw data using Databricks.



CASE STUDY: EXPLORATORY DATA ANALYSIS ON 2019-NOV.CSV

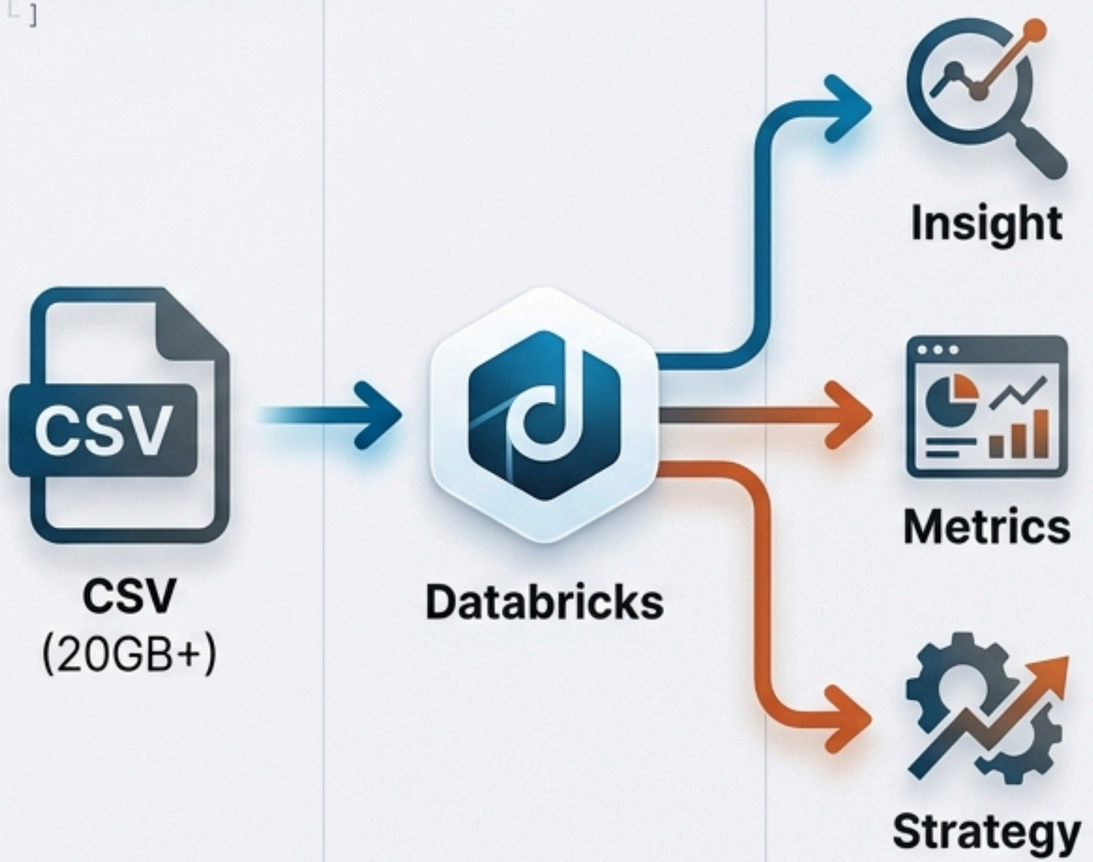
From Raw Clicks to Business Strategy

We are analyzing a large-scale event log containing user interactions from a major online store. The raw file, “2019-Nov.csv”, captures millions of views, cart additions, and purchases.

THE CHALLENGE: The data is massive and unstructured. To uncover trends and brand dominance, we require a tool capable of handling scale efficiently.

THE SOLUTION: We will utilize PySpark within a Databricks notebook environment to Load, Inspect, Transform, and Visualize the data.

```
LOAD ...
TRANSFORM ... ....,
VISUALIZE ...
]
```




```
VISUALIZE
.. ..... .... .. ........ )
df.show()
}
```

INDIAN DATA CLUB

Ingesting the Raw Data

```
events = spark.read.csv('/Volumes/workspace/ecommerce/ecommerce_data/2019-Nov.csv',
         header=True,
         inferSchema=True)
```

CRITICAL PARAMETER: inferSchema=True

By enabling schema inference, PySpark automatically scans the raw data to identify data types (integers, strings, timestamps). This eliminates the need to manually define the structure of the file, accelerating the initial setup.

Discovering the Schema

Verifying the structure of the dataframe

```
events.printSchema()
```

```
root
 |-- event_time: timestamp (nullable = true)
 |-- event_type: string (nullable = true)
 |-- product_id: integer (nullable = true)
 |-- category_id: long (nullable = true)
 |-- category_code: string (nullable = true)
 |-- brand: string (nullable = true)
 |-- price: double (nullable = true)
 |-- user_id: integer (nullable = true)
 |-- user_session: string (nullable = true)
```

Focusing on Key Metrics

Code

```
display(
  events.select(
    'event_type',
    'product_id',
    'price'
  ).limit(10)
)
```

Result

event_type	product_id	price
view	1003461	489.07
view	5000088	293.65
view	17302664	28.31
view	3601530	712.87
view	1004775	183.27

The .limit(10) function allows us to preview the data doter the data contents without triggering a memory-intensive full load.

Aggregating User Actions

Grouping by event type reveals the distribution of user behavior.

```
display(events.groupBy('event_type').count())
```

Dominant Action: ~63 Million Views

event_type	count
view	63,556,110
cart	3,028,930
purchase	916,939

The E-Commerce Conversion Funnel

INTERPRETATION:

While traffic volume is immense, the drop-off is significant. Only 1.4% of product views result in a final transaction, providing a clear baseline for optimization.

Advanced Chaining for Market Leaders

Breaking down complex data transformations into logical, actionable steps.

```
top_brands = (
  events.filter('brand IS NOT NULL')
  .groupBy('brand')
  .count()
  .orderBy('count', ascending=False)
  .limit(5)
)
```

CLEAN:
Remove records with missing brand names.

ORGANIZE:
Group data by manufacturer.

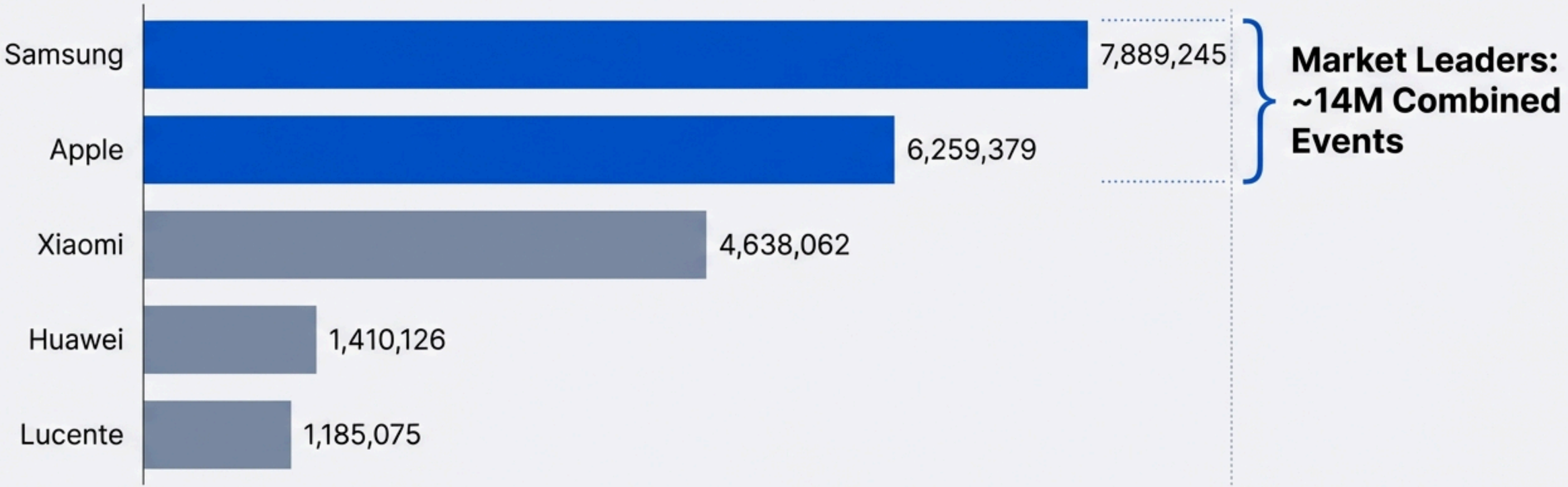
TALLY:
Calculate total events per brand.

RANK:
Sort from highest to lowest.

FOCUS:
Isolate only the top 5 performers.

Samsung and Apple Dominate the Market

Analysis of the top 5 brands by event volume.



Syntax Summary

Key PySpark functions and operations used throughout the analysis.

`spark.read.csv()` JetBrains Mono

Loads data from file path. Use 'inferSchema=True' to detect types.

`printSchema()` JetBrains Mono

Displays the tree structure of columns and data types.

`select()` JetBrains Mono

Projects specific columns to narrow the analysis scope.

`groupBy().count()` JetBrains Mono

Aggregates data based on categorical values.

`filter()` JetBrains Mono

Excludes rows based on specific conditions (e.g., removing nulls).

`orderBy()` JetBrains Mono

Sorts the resulting DataFrame in ascending or descending order.

The Data Analysis Loop

This workflow transforms raw 2019 e-commerce logs into a clear hierarchy of market leadership and conversion metrics.