Data Analysis and Knowledge Discovery

Lecture 11

Faculty of Mathematics and Computer Science
Babeș-Bolyai University

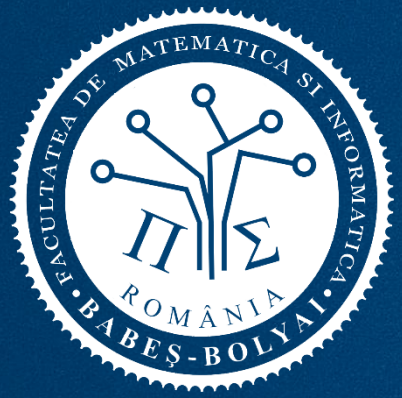
Sergiu Limboi, PhD Teaching Assistant

Motto:“When data never stops, your model must never sleep.”

Data Stream Mining

AGENDA

- Warm-Up
- Industry Reality
- What is Data Stream Mining?
- Stream Processing Architecture
- Stream Mining Techniques
- Algorithms for Stream Mining
- Teamwork Time
- Industry Case Studies
- LIVE DEMO
- Key Takeaways



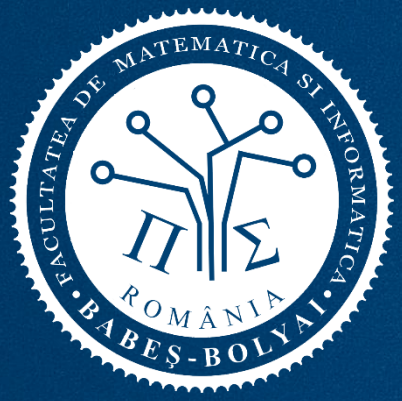
Warm-Up

Faculty of Mathematics and Computer Science

Warm-Up

Go to www.menti.com and enter the code **4938 8191**

or use the QR code



Industry Reality

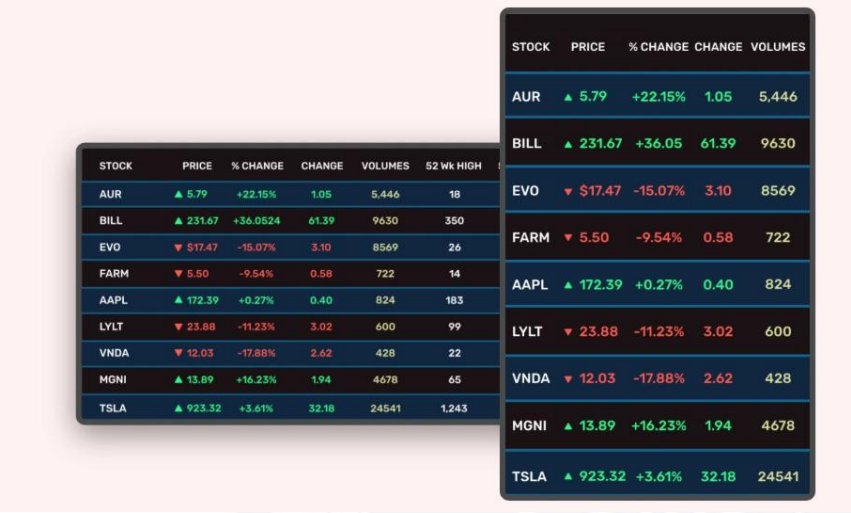
Industry Reality

- What if data is infinite, continuous, and fast?

- Data is too fast to store

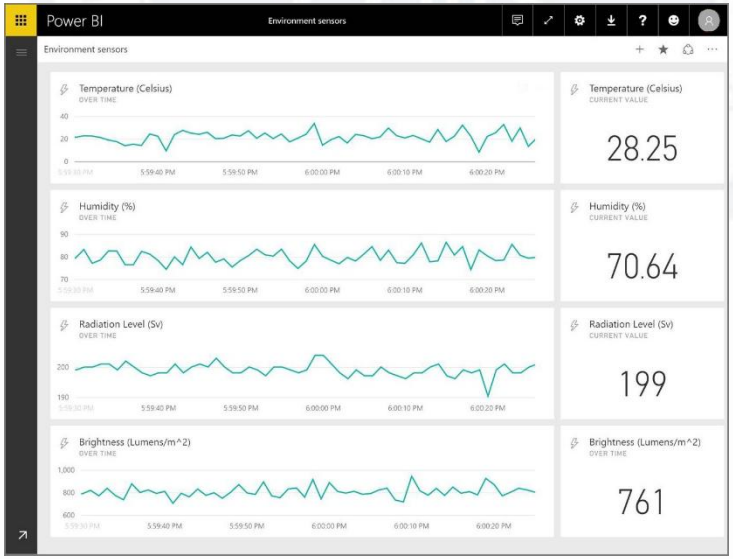
Before this Lecture	After this Lecture
Data is large	Data is infinite
Process later	Process now
Batch	Stream

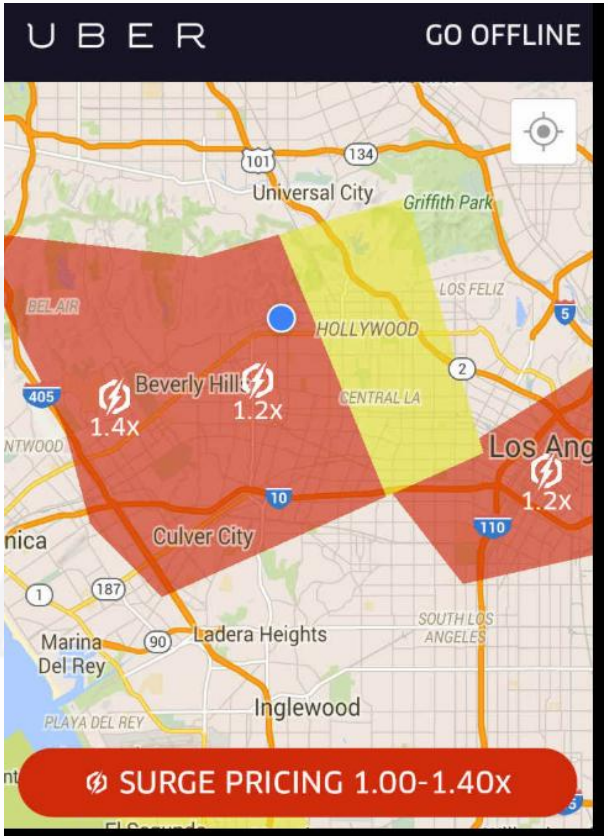
Industry Reality



STOCK	PRICE	% CHANGE	CHANGE	VOLUMES	52 Wk HIGH
AUR	▲ 5.79	+22.15%	1.05	5,446	18
BILL	▲ 231.67	+36.0524	61.39	9630	350
EVO	▼ $17.47	-15.07%	3.10	8569	26
FARM	▼ 5.50	-9.54%	0.58	722	14
AAPL	▲ 172.39	+0.27%	0.40	824	183
LYLT	▼ 23.88	-11.23%	3.02	600	99
VNDA	▼ 12.03	-17.88%	2.62	428	22
MGNI	▲ 13.89	+16.23%	1.94	4678	65
TSLA	▲ 923.32	+3.61%	32.18	24541	1,243

STOCK	PRICE	% CHANGE	CHANGE	VOLUMES
AUR	▲ 5.79	+22.15%	1.05	5,446
BILL	▲ 231.67	+36.05	61.39	9630
EVO	▼ $17.47	-15.07%	3.10	8569
FARM	▼ 5.50	-9.54%	0.58	722
AAPL	▲ 172.39	+0.27%	0.40	824
LYLT	▼ 23.88	-11.23%	3.02	600
VNDA	▼ 12.03	-17.88%	2.62	428
MGNI	▲ 13.89	+16.23%	1.94	4678
TSLA	▲ 923.32	+3.61%	32.18	24541





Industry Reality

What is Data Stream Mining?

What is a Data Stream?

- A data stream is a continuous, unbounded sequence of data points arriving over time.

- Key properties:
  - Continuous
    - Data keeps coming
    - Examples:
      - sensor readings every second
      - user clicks
      - transactions
  - Infinite
    - No “dataset size”
  - Ordered in time
    - Each event has a timestamp
    - Example: (10:01:02) user likes

What is a Data Stream?

- High velocity
  - Data arrives VERY FAST
  - Examples:
    - stock market → thousands/sec
    - TikTok → millions/sec
- Transient
  - If you don't process it → it's gone

- Batch data
- Stream data

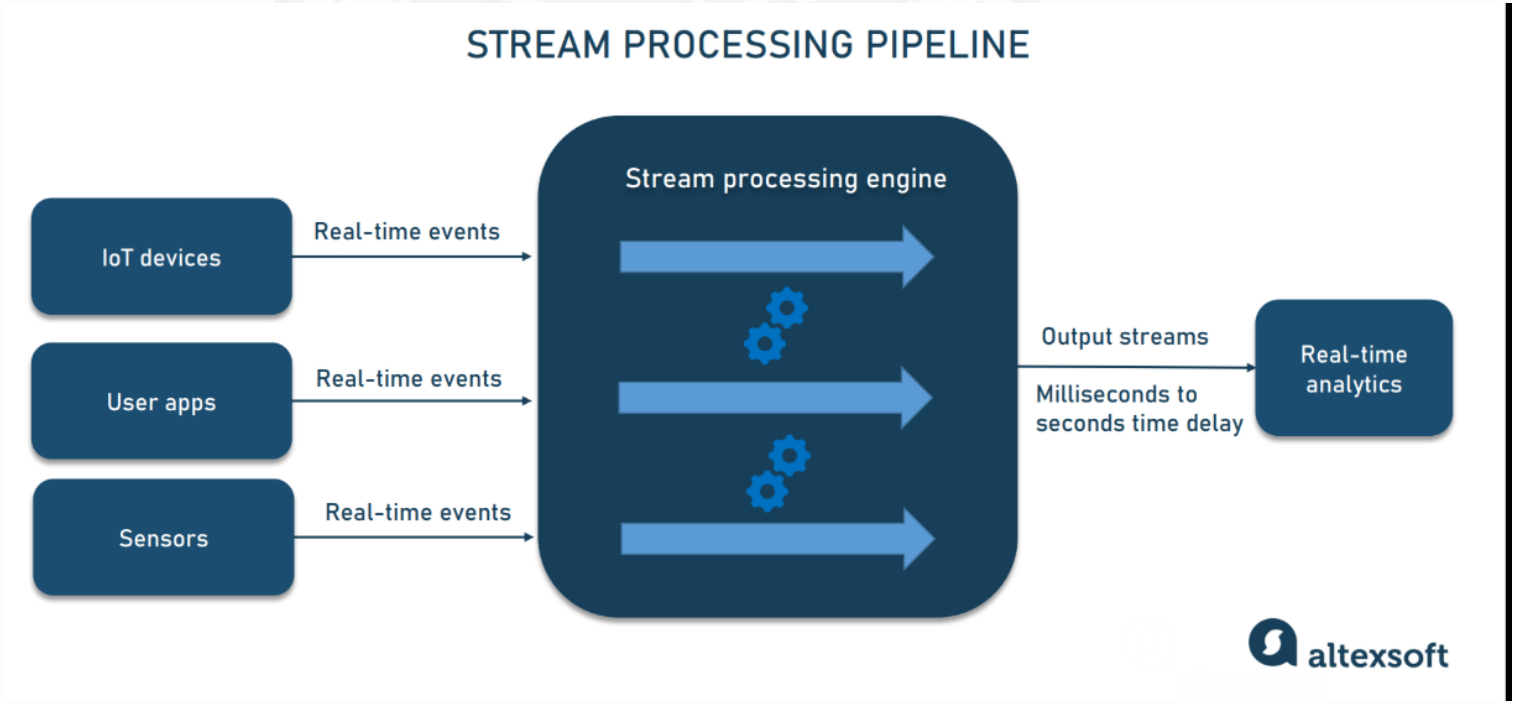
What is Data Stream Mining?

- Data stream mining is the process of extracting patterns, models, and knowledge from continuous data streams in real-time.
- Key properties:

Property	Explanation
Infinite	Data never stops
High speed	Arrives continuously
Real-time	Must process instantly
One-pass	No multiple scans
Memory constraint	Cannot store all

What is Data Stream Mining?

- Opposite of batch:
  - Batch: data → store → process
  - Stream: data → process → maybe store



Batch vs. Stream

Aspect	Batch	Stream
Data	Finite	Infinite
Storage	Full dataset	Partial/none
Processing	Offline	Real-time
Passes	Multiple	One-pass
Latency	Minutes/hours	Milliseconds

- Batch = think carefully
- Stream = react immediately

Processing Paradigm Shift

- In Batch:
  - Collect → Store → Process → Train → Predict
- In Stream:
  - Arrive → Process → Update → Decide
- No waiting
- No full dataset
- No reprocessing

Why Classical ML Fails?

❌ Cannot store data
❌ Cannot reprocess
❌ Cannot shuffle

- Imagine:
  - 1M events/sec
  - 24h → 86 billion events
  - Impossible to:
    - Save everything
    - Train classical models
- In streaming, you don't have the luxury of thinking twice.

Core Constraints

- One-pass constraint
  - You see data **only once**
  - Example: transaction arrives → must process now
    - cannot say “I’ll analyse later”
- Memory constraint
  - Cannot store everything
  - 1M events/sec → impossible to store
- Real-time constraint
  - Decisions must be immediate
  - Example: fraud detection → milliseconds
- Concept drift
  - Data changes over time
  - Example: TikTok trends today ≠ tomorrow

What do we “mine” in streams?

- Same problems, different constraints

Task	Batch	Stream
Classification	Static model	Evolving model
Clustering	Full dataset	Dynamic clusters
Frequent patterns	Exact	Approximate
Regression	Offline	Online

Example: Credit Card Transactions

- Streams look like:
  - (user1, 100$, 10:01)
  - (user2, 500$, 10:02)
  - (user1, 900$, 10:03)
  - ....
- Batch approach
  - store all data
  - train model later
- Stream approach
  - For each transaction:
    - read event
    - compute features
    - predict fraud
    - update model
  - All happens instantly

What is Data Stream Mining?

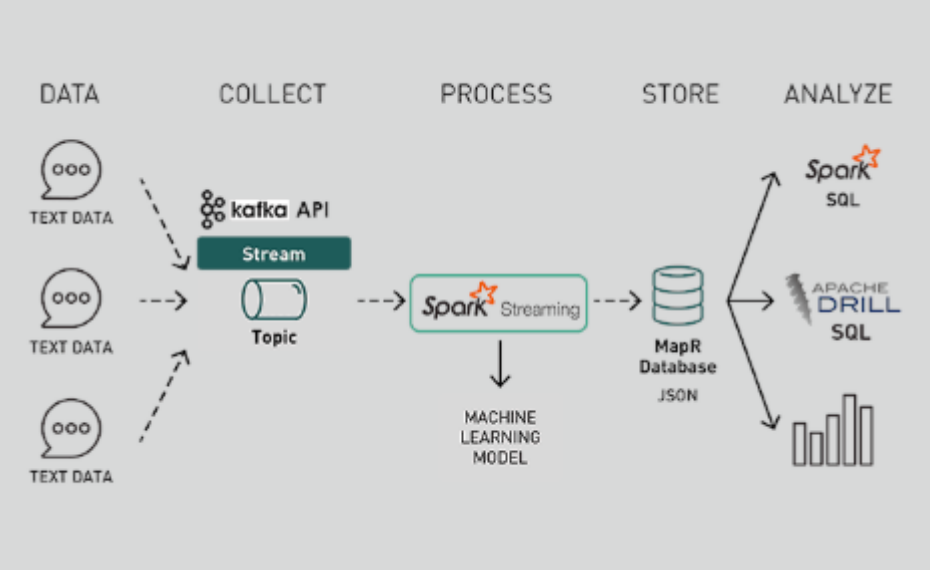
- “Streaming is not about data analysis. It is about data reaction.”

- Micro vs. Macro view
  - Micro (event-level)
    - Process one event
    - Update model
  - Macro (window-level)
    - analyse last 1 min / 1 hour

Stream Processing Architecture

Stream Processing Architecture

- A **stream architecture** is a system that can:
  - ingest data continuously
  - process it instantly
  - produce results in real-time



Stream Processing Architecture

- Layer 1: Data Sources (Producers)
  - Sources= Systems that **generate events**

Domain	Source
Uber	driver GPS
TikTok	user actions
Banking	transactions
IoT	sensors

- Data is not stored first
- It is **emitted as events**
- Everything becomes an **event**

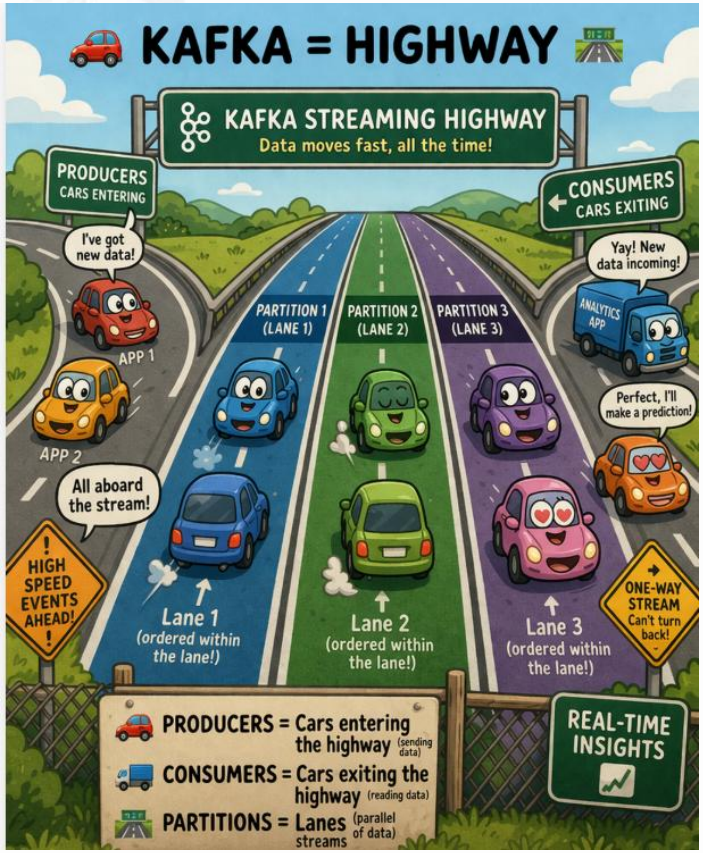
```
{
  "user_id": 123,
  "action": "like",
  "timestamp": 1710000000
}
```

Stream Processing Architecture

- Layer 2: Stream system/ Ingestion (Kafka)
  - Kafka = central nervous system of streaming
  - What it does:
    - receives data from producers
    - stores it temporarily
    - distributes it to consumers
  - Topic= logical stream
  - Example: transactions, clicks
  - Producer→ sends data
  - Consumer→ reads data
  - Partition : splits data → scalability

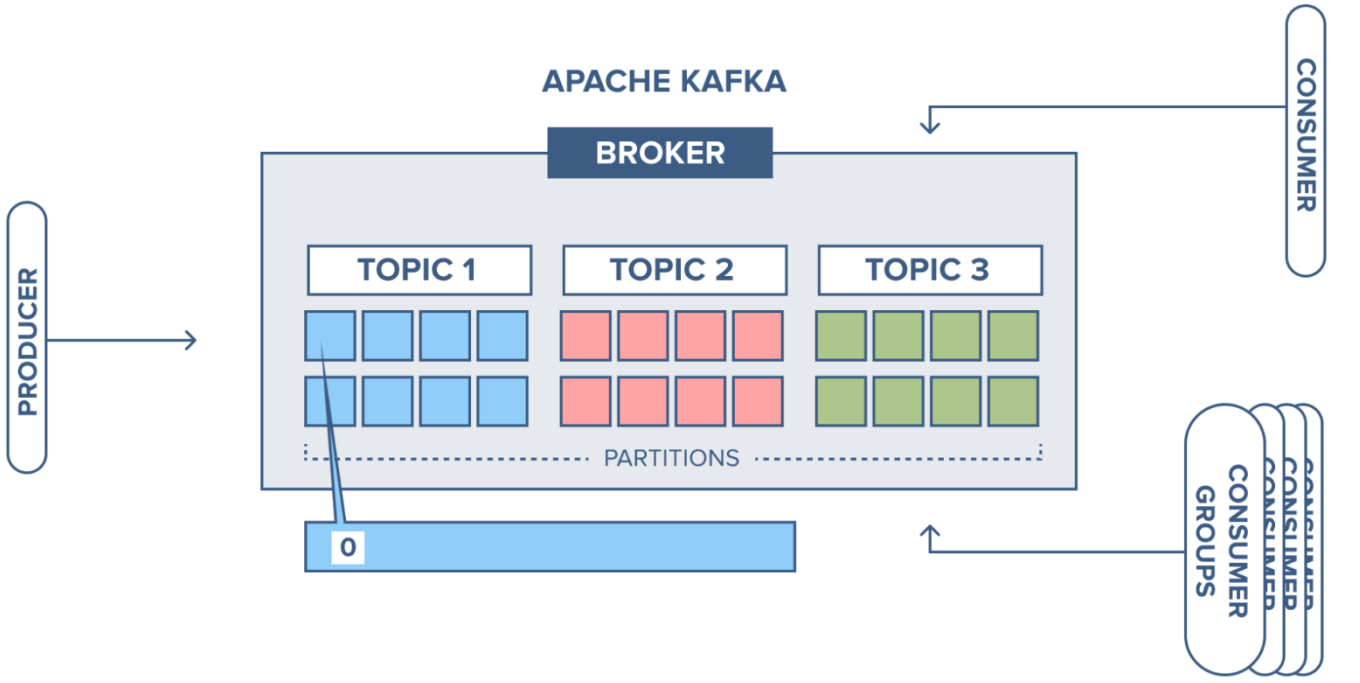
Stream Processing Architecture

- Layer 2: Stream system/ Ingestion (Kafka)
  - Why Kafka is critical:
    - handles millions of events/sec
    - fault-tolerant
    - scalable



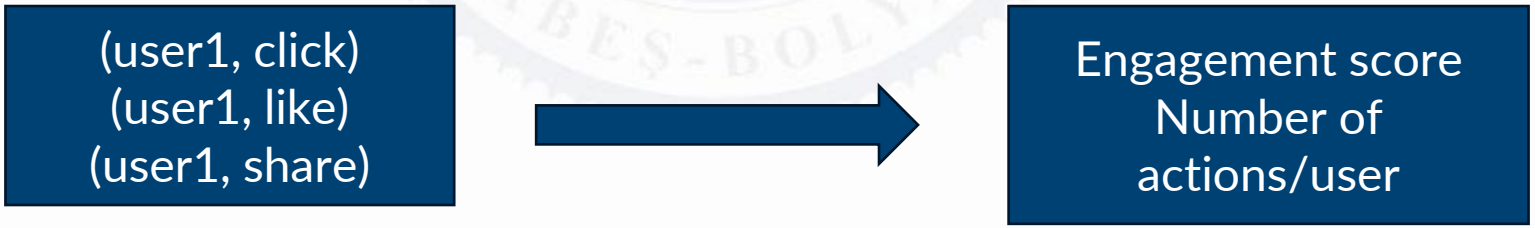
Stream Processing Architecture

- Layer 2: Stream system/ Ingestion (Kafka)



Stream Processing Architecture

- Layer 3: Processing engine
  - **Tools:**
    - Spark Streaming
    - Flink
  - **What happens here:**
    - filtering
    - aggregation
    - feature engineering
    - joining streams
    - transformations



Stream Processing Architecture

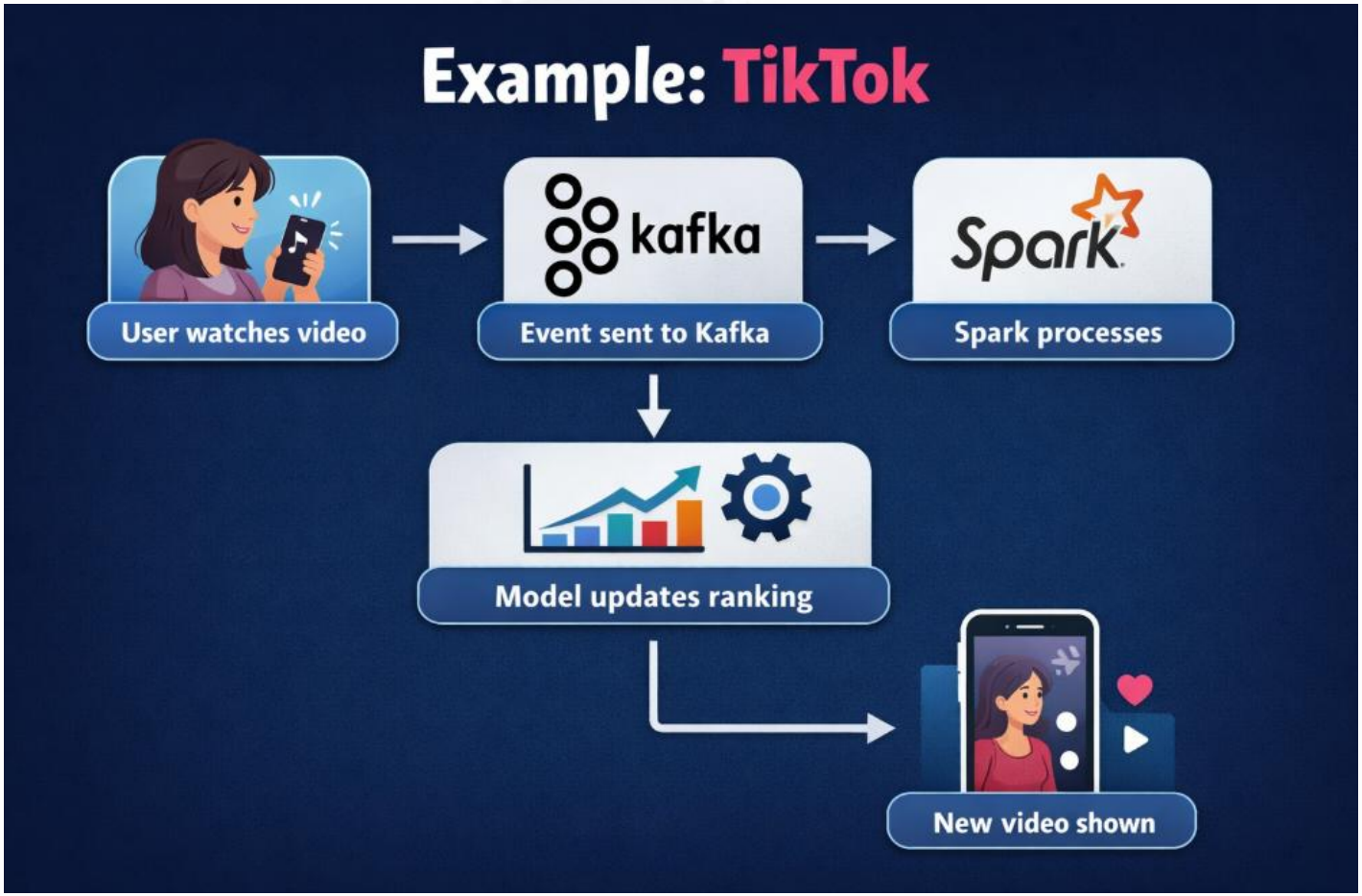
- Layer 4: Model / Intelligence Layer
  - Apply ML / logic
  - Types:
    - Rule-based
      - if amount > 1000 → fraud
    - Machine learning models
      - Classification
      - Regression
      - Clustering
    - Online models
      - continuously updated
  - Model must be **fast + adaptive**

Stream Processing Architecture

- Layer 5: Output Layer
  - Deliver results
  - Example:
    - TikTok → show next video
    - fraud → block transaction

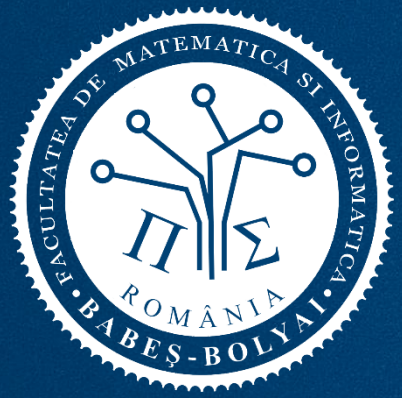
Type	Example
Dashboard	analytics
Alerts	fraud
API	recommendation
Storage	logs

Stream Processing Architecture



Key System Properties

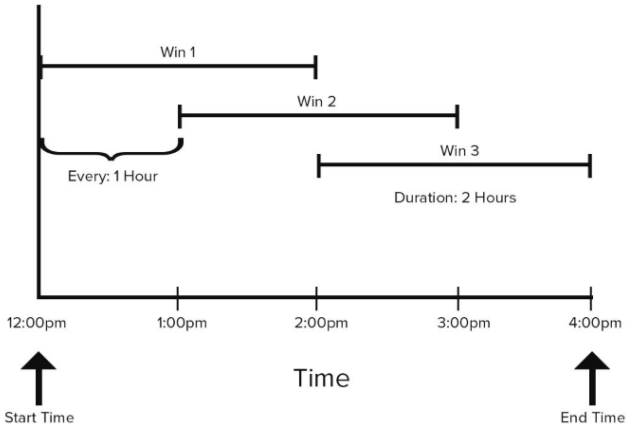
- Scalability
  - system must handle: millions of events
- Fault tolerance
  - System must survive:
    - crashes
    - network failures
- Low latency
  - Response must be: real-time (<100ms)
- Exactly-once / at-least-once
  - Data delivery guarantees
  - at-least-once → duplicates possible
  - exactly-once → harder but ideal



Stream Mining Techniques

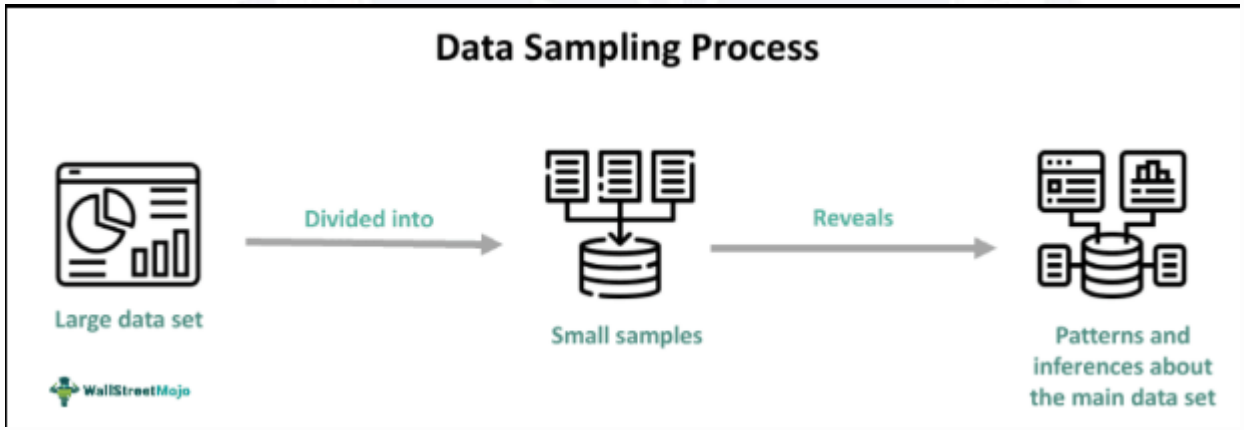
Stream Mining Techniques

- Sliding Window Model
  - Keep only recent data
  - Example: Last 1 hour transactions
  - **Types:**
    - Count-based (last 1000 events)
    - Time-based (last 1h)
    - Landmark window → window that starts at a **fixed point in the past (the landmark)** and keeps growing as new data arrives.



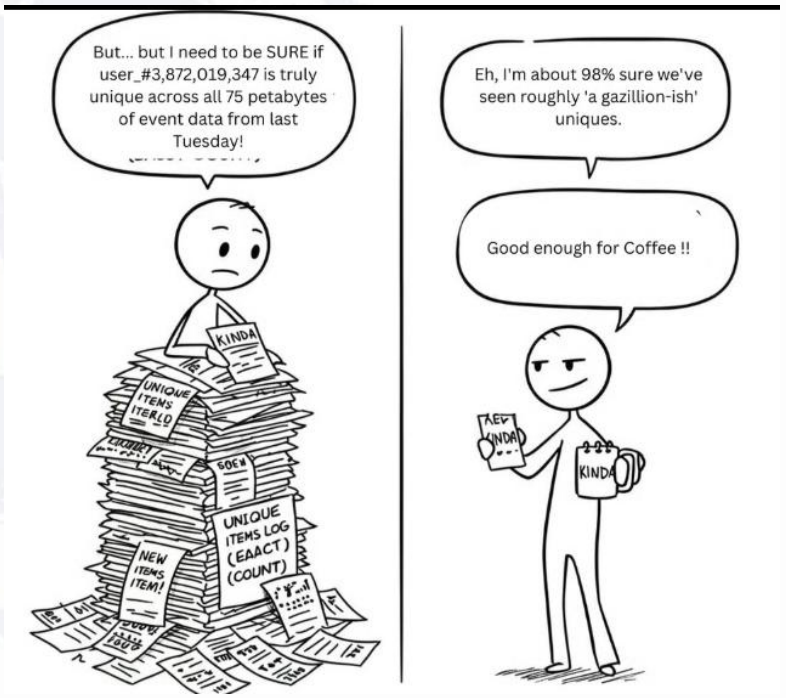
Stream Mining Techniques

- Sampling
  - Keep subset of data
  - Example: 1% of tweets



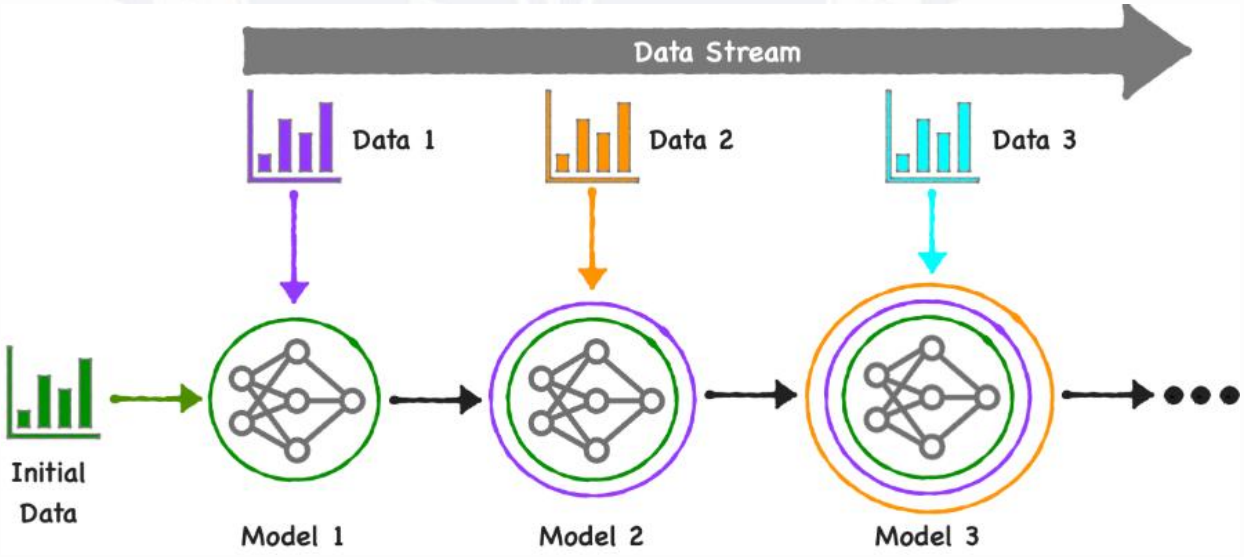
Stream Mining Techniques

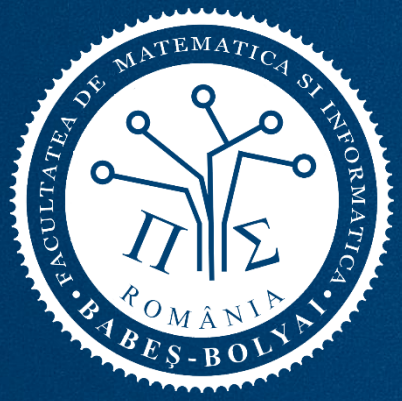
- Approximation (Sketches)
  - Estimate instead of exact
  - Example:
    - Count distinct users
    - Top-k items



Stream Mining Techniques

- Incremental Learning
  - Model updates continuously
  - Instead of:
    - train once
  - We do:
    - update model per new data





Algorithms for Stream Mining

Algorithms for Stream Mining

- Classical algorithms assume:
  - full dataset available
  - multiple passes
  - unlimited memory
- Stream algorithms must:
  - work incrementally
  - use limited memory
  - process one instance at a time
  - adapt to concept drift

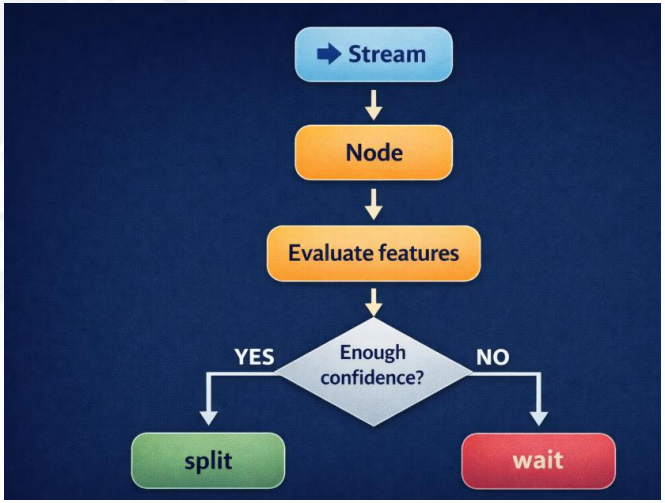
Algorithms for Stream Mining

- Categories of Stream algorithms:
  - Classification
  - Clustering
  - Frequent Pattern Mining
  - Regression

- SAME problems as ML… DIFFERENT constraints

Classification → Hoeffding Trees

- Build a decision tree **without seeing all data**
- Use **statistics** to decide splits early
- **Core concept: Hoeffding Bound**
  - It tells us: "After enough samples, we can confidently choose the best split."
- Instead of waiting for ALL data:
  - observe small sample
  - estimate best feature
  - split early



Classification → Hoeffding Trees

- Example: Predict fraud
  - Stream: (amount, location, time)
  - Tree learns:
    - If amount > 900 → suspicious
    - if location unusual → suspicious
  - Updates continuously

- Advantages
  - Fast
  - Memory efficient
  - Works in real-time

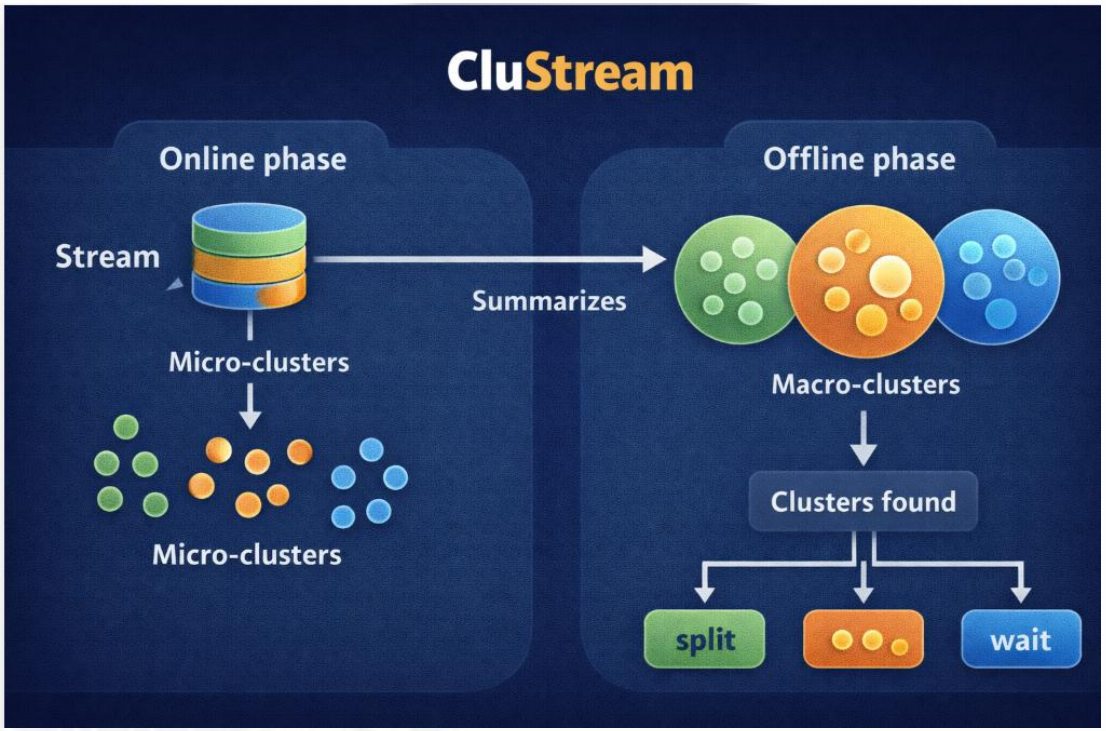
- Limitations
  - Approximate
  - Sensitive to drift

Clustering → CluStream

- Cluster data that **never stops arriving**
- Instead of storing all points: store **summaries (micro-clusters)**
- **Online phase**
  - incoming data → update micro-clusters
- **Offline phase**
  - group micro-clusters into real clusters
- TikTok users:
- Stream:
  - (user1, watches sports)
  - (user2, watches cooking)
  - (user3, watches sports)
- Micro-clusters: sports lovers , cooking lovers
- Later: merge into bigger segments

Clustering → CluStream

- **Advantages**
  - scalable
  - memory efficient
- **Limitations**
  - approximate clusters
  - parameter sensitive
- **DenStream (advanced)**
  - Handles:
    - noise
    - outliers



Frequent Pattern Mining → Lossy Counting

- Find frequent items in stream
- Issue: cannot store all items
- Solution: Approximate counting
- Idea
  - divide stream into buckets
  - track counts approximately
  - remove low-frequency items
- Stream: A, B, A, C, A, B, A, D
  - A → frequent
  - B → frequent
  - C, D → removed

Frequent Pattern Mining → Lossy Counting

- **Advantages**
  - low memory
  - Fast
- **Limitations**
  - not exact
  - threshold dependent
- You don’t find **exact frequency**
- You find **approximate frequent items**

Regression in Streams

- Predict numeric values in real-time

- Approach
  - incremental models
  - update weights continuously

- Uber:
  - predict ride demand
  - Model updates every second

Key Pattern Across ALL Algorithms

- Incremental learning
  - model = update(model, new_data)
- Approximation
  - exact → impossible
  - approximate → necessary
- Summarization
  - micro-clusters
  - Counts
  - sketches
- Adaptation
  - concept drift handling

Trade-offs

Factor	Batch	Stream
Accuracy	High	Approximate
Speed	Slow	Fast
Memory	Large	Small
Flexibility	High	Limited

- In streaming, we trade perfection for speed

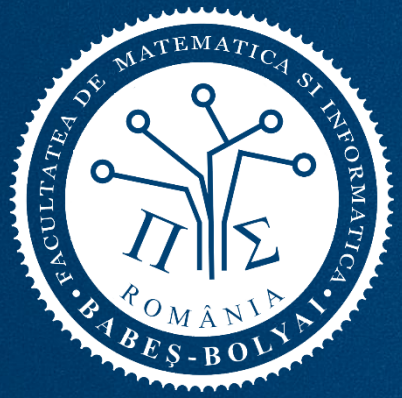
Teamwork Time

Teamwork Time — Real-Time Fraud Detection System

- **Duration: 10 minutes**
- You are part of a data science team at a bank or fintech company.
- Every second, the system receives many card transactions from users around the world.
- Your goal is to help design a system that can detect suspicious transactions **in real time**, before the payment is accepted.
- The bank wants:
  - very fast decisions
  - low fraud losses
  - few false alarms
  - a system that adapts when fraud patterns change

Teamwork Time — Real-Time Fraud Detection System

- Each team should discuss and propose answers for the following:
  - **What data comes in as a stream?**
    - Think about what arrives continuously, one event at a time.
  - **What features would you compute?**
    - Think about useful signals that help distinguish normal behaviour from fraud.
  - **What model type would you use?**
    - Would you use rules, ML, anomaly detection, incremental learning, or a combination?
  - **Batch or streaming?**
    - Which parts should be real-time and which parts can remain offline?
  - **How would you handle recent behaviour?**
    - Would you use a sliding window? If yes, what kind?
  - **How would the system adapt over time?**
    - How do you deal with changing fraud patterns?



Industry Case Studies

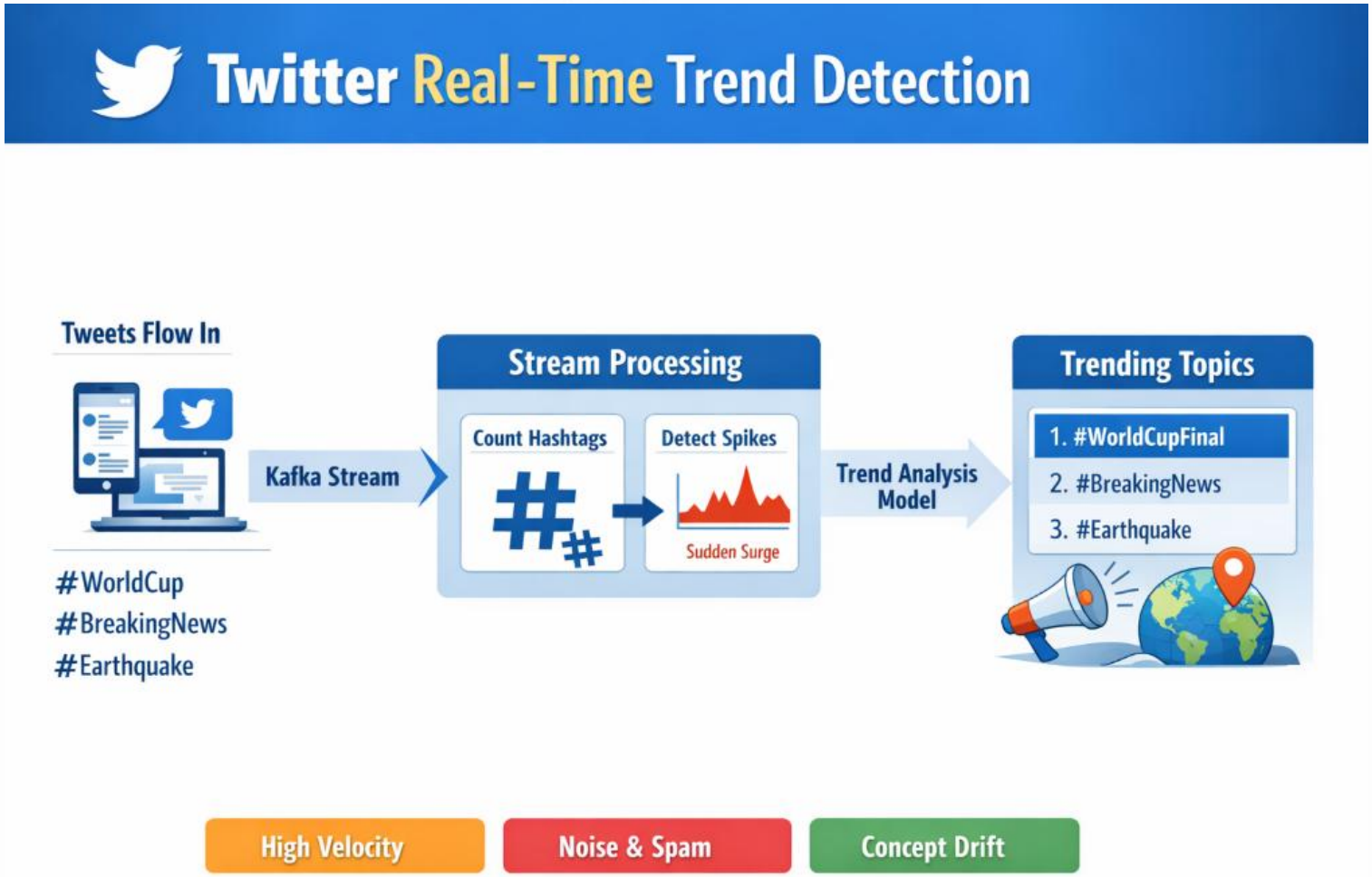
Case Study 1: Twitter (X) – Real-Time Trend Detection

- How does X detect trending topics in real-time?
-
- **Reality**
  - Millions of tweets per minute
  - Global scale
  - Must detect trends in **seconds**, not hours

- Sudden spike:
  - “World Cup Final”
  - “Earthquake”
  - “Breaking News”

- System must:
  - detect spike
  - rank it
  - show in “Trending”

Case Study 1: Twitter (X) – Real-Time Trend Detection



Case Study 2: Google Ads – Real-Time Bidding (RTB)

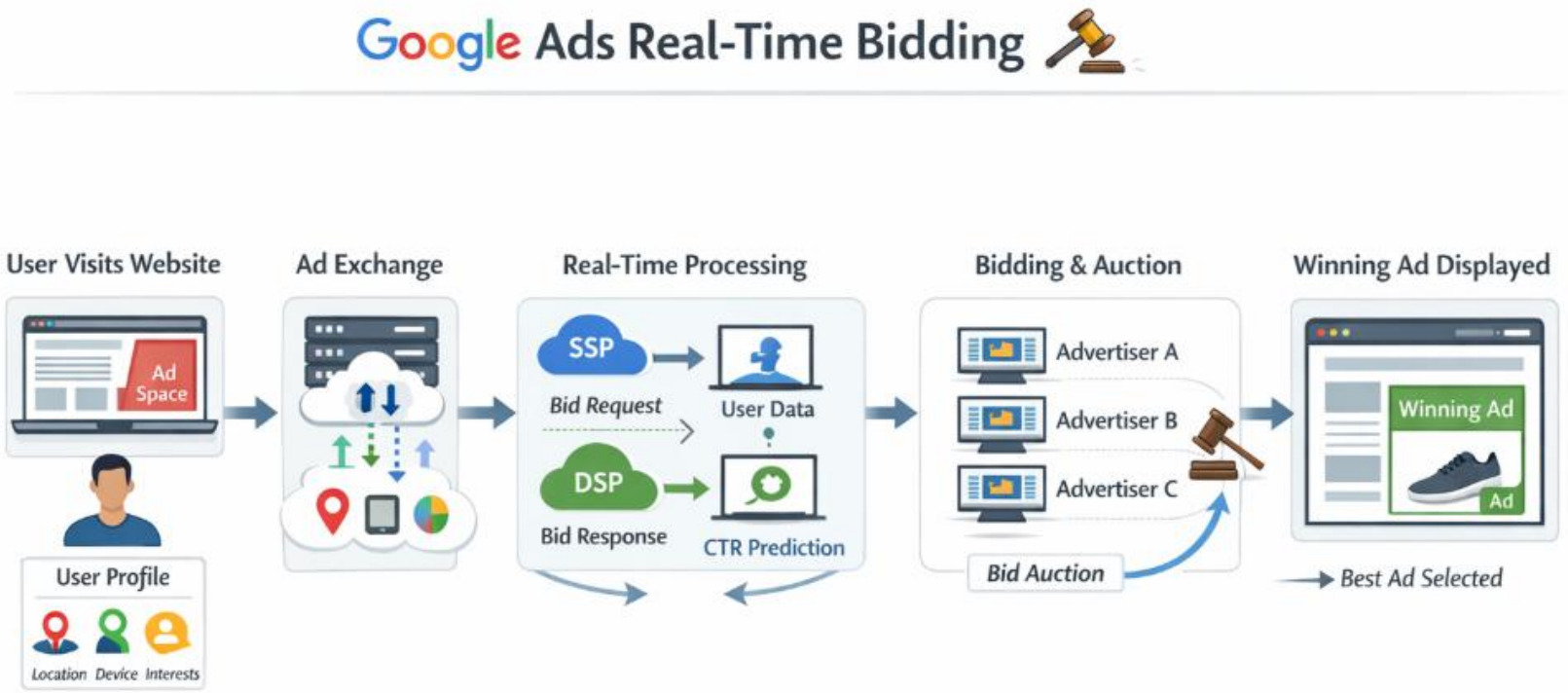
- How does Google decide **which ad to show in milliseconds when you open a webpage?**

- **Reality**
  - Every page load triggers an **auction**
  - Happens in **~100 milliseconds**
  - Thousands of advertisers compete

- This is one of the **fastest streaming ML systems in the world**

- You open a website:
- In < 0.1 seconds:
  - Your profile is analysed
  - Advertisers bid
  - Best ad is selected
  - Ad is displayed

Case Study 2: Google Ads – Real-Time Bidding (RTB)

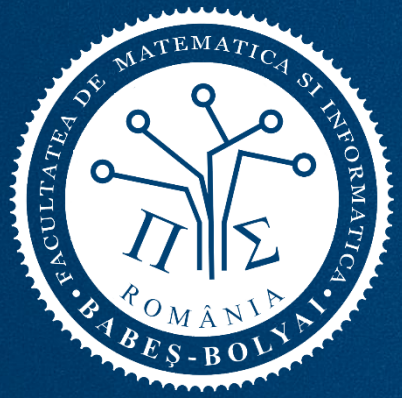



FACULTATEA DE MATEMATICA SI INFORMATICA
ROMÂNIA
BABEŞ-BOLYAI

LIVE DEMO

LIVE DEMO

- Kafka server hosted on AWS
- Producer→ data stream
- Consumer→ read data stream and apply ML techniques



Key Takeaways

Key takeaways

- Data streams = infinite + fast
- Cannot store everything
- One-pass algorithms
- Approximation is essential
- Models must adapt (concept drift)

Expectation

We process all data perfectly

Reality

We approximate everything and hope it works

Thank you for your attention — questions, thoughts, or challenges?

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE
BABEȘ-BOLYAI UNIVERSITY

1 Mihail Kogălniceanu Street,
Cluj-Napoca, Cluj, România

www.cs.ubbcluj.ro