

REAL-TIME PLAYER ENGAGEMENT

Presented By:

Mai Phan Anh Tùng : 23020433

Phạm Hải Tiến : 2302042

Phạm Quốc Hùng : 23020373

Đại học Công nghệ - ĐHQG Hà Nội

Mục Lục

I: Giới thiệu (Introduction)

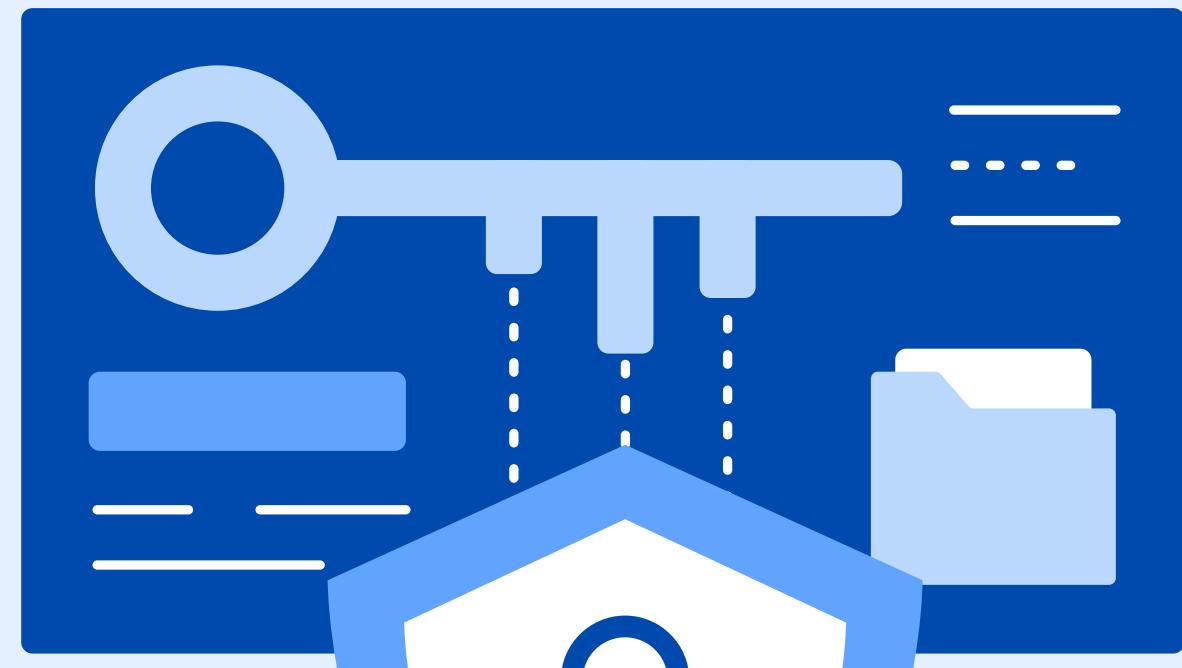
II: Phân Tích Dữ Liệu Nền (Base Analytics)

III: Kiến Trúc Hệ Thống Chính (Main System Architecture)

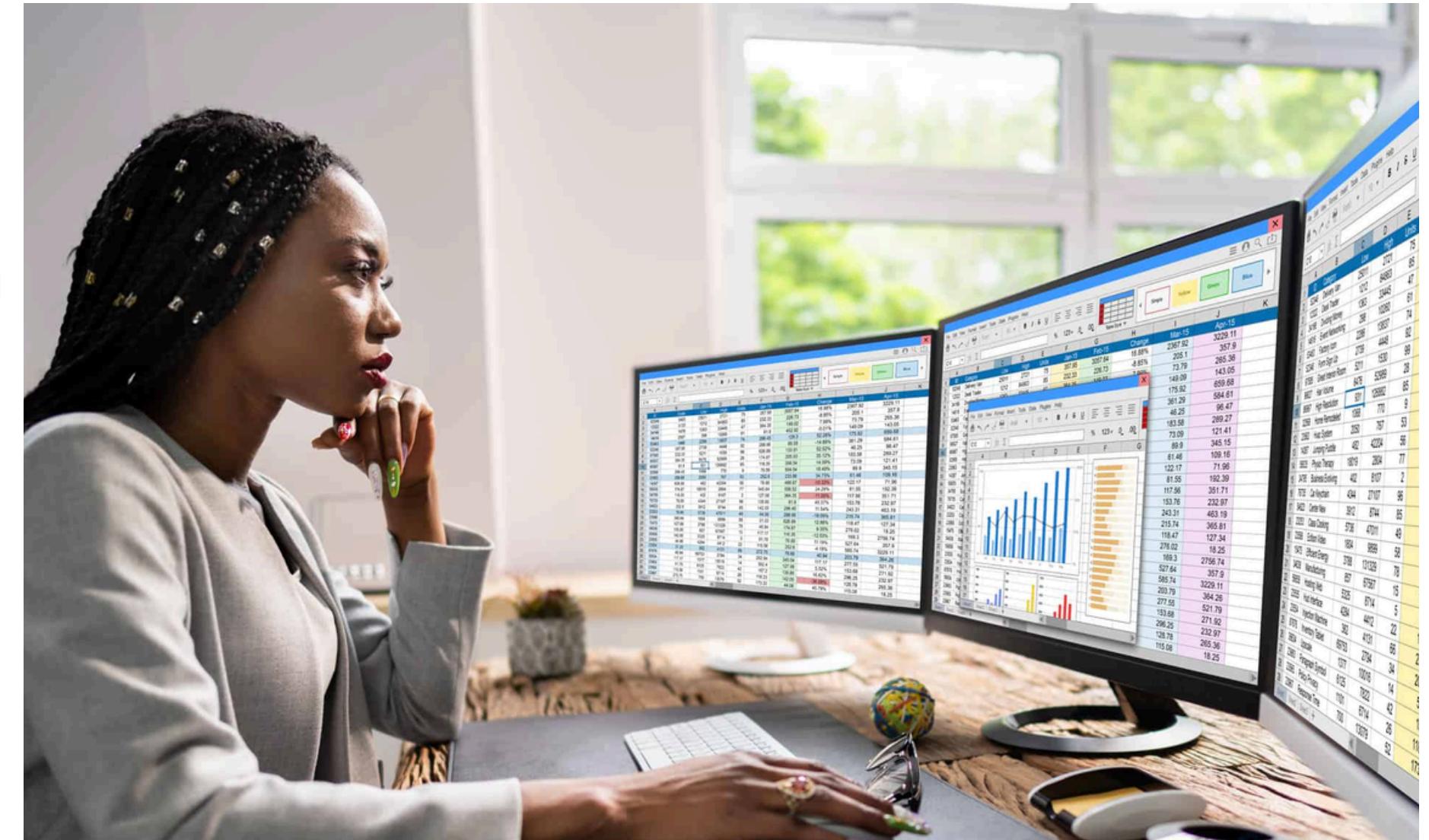
IV: Kết Luận (Conclusion)

I: Giới thiệu

VÌ SAO CẦN PHÂN TÍCH REAL-TIME TRONG GAME?



- Hiểu hành vi người chơi ngay lập tức
- Điều chỉnh gameplay theo thời điểm
- Cá nhân hoá trải nghiệm
- Tối ưu vận hành & máy chủ
- Cải thiện doanh thu





Marvel's Avengers là ví dụ điển hình cho thất bại do thiếu phân tích hành vi người chơi theo thời gian thực. Không có hệ thống theo dõi Engagement khiến người chơi nhanh chóng chán nản, dẫn đến lượng người chơi giảm mạnh chỉ sau vài tuần.



Fortnite áp dụng real-time player behavior analytics để tối ưu hoá độ khó, nội dung, sự kiện và vật phẩm theo từng loại người chơi, giúp tăng retention và tạo ra mô hình vận hành live-service cực kỳ hiệu quả.

BÀI TOÁN NGHIÊN CỨU

- Dự đoán mức độ tương tác người chơi real-time
- Theo dõi trạng thái người chơi
- Tự động cập nhật và phản hồi nhanh



THÁCH THỨC THU THẬP DỮ LIỆU

- Không có API telemetry real-time
- Vấn đề bảo mật, tải hệ thống, bản quyền
- Dữ liệu chỉ có dạng batch

GIẢI PHÁP

🎮 Player Engagement Analysis & Prediction 🚀

Notebook Input Output Logs Comments (29)

Show hidden cell

Version 1 of 1

Runtime

3s

Input

DATASETS

predict-online-gaming-behavior-data

Tags

Exploratory Data Analysis

Data Visualization

Data Analytics Classification

Multiclass Classification

Language

Python

Player Engagement Analysis & Prediction

Overview: Predict Online Gaming Behavior

This dataset provides a detailed view of player behavior and demographics in online gaming environments. It includes a variety of features that capture key aspects of gaming activity, player characteristics, and engagement levels, making it an excellent resource for analyzing player retention and predicting engagement patterns. 🌟

<https://www.kaggle.com/code/sulaniishara/player-engagement-analysis-prediction>

- Sử dụng dataset Kaggle
- Sinh sự kiện register/login/logout/purchase/level_up
- Giả lập dòng sự kiện streaming

II: Phân Tích Dữ Liệu Nền

Mục tiêu

- Hiểu đặc trưng hành vi người chơi trước khi xây dựng mô hình
- Xác định các biến quan trọng ảnh hưởng đến EngagementLevel
- Kiểm tra chất lượng dữ liệu (thiếu, lệch, nhiễu)
- Chuẩn hoá nền tảng để tối ưu pipeline học máy

Thành phần dữ liệu

- Nhân khẩu học: Age, Gender, Location
- Hành vi chơi: SessionsPerWeek, PlayTimeHours, AvgSessionDuration
- Tiến trình: PlayerLevel, AchievementsUnlocked
- Chi tiêu: InGamePurchases
- Độ khó game: GameDifficulty
- Nhãn: EngagementLevel

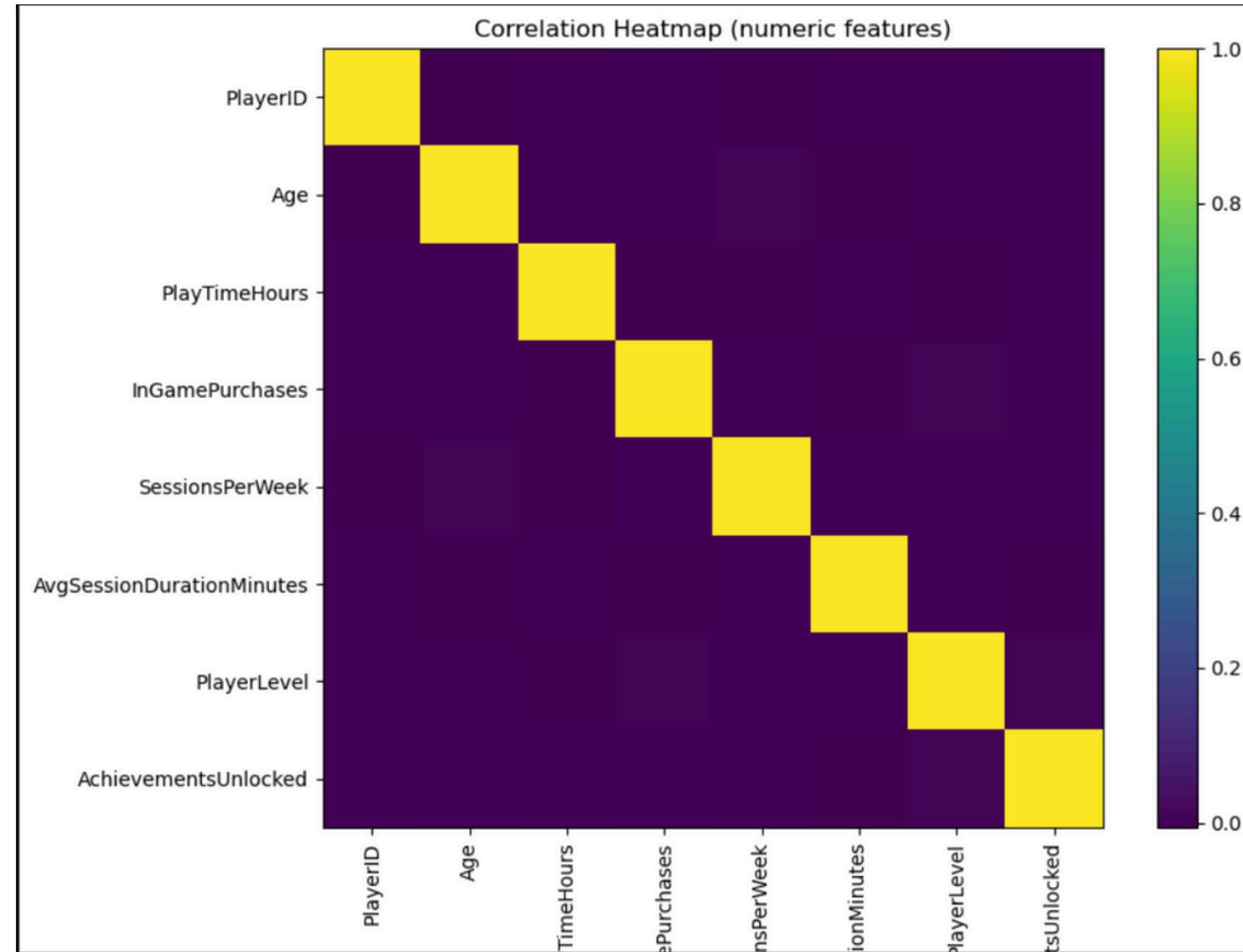
PlayerID	Age	Gender	Location	GameGenre	PlayTimeH	InGamePU	GameDifficulty	SessionsPerWeek	AvgSessionTime	PlayerLevel	AchievementCount	EngagementLevel
9000	43	Male	Other	Strategy	16.27112	0	Medium	6	108	79	25	Medium
9001	29	Female	USA	Strategy	5.525961	0	Medium	5	144	11	10	Medium
9002	22	Female	USA	Sports	8.223755	0	Easy	16	142	35	41	High
9003	35	Male	USA	Action	5.265351	1	Easy	9	85	57	47	Medium
9004	33	Male	Europe	Action	15.53194	0	Medium	2	131	95	37	Medium
9005	37	Male	Europe	RPG	20.56186	0	Easy	2	81	74	22	Low
9006	25	Male	USA	Action	9.752716	0	Hard	1	50	13	2	Low
9007	25	Female	Asia	RPG	4.401729	0	Medium	10	48	27	23	Medium
9008	38	Female	Europe	Simulation	18.15273	0	Easy	5	101	23	41	Medium
9009	38	Female	Other	Sports	23.94277	0	Easy	13	95	99	36	High
9010	17	Male	USA	Strategy	4.829916	0	Hard	8	95	14	12	High
9011	36	Female	Asia	Simulation	5.535981	1	Easy	16	124	62	31	High
9012	16	Male	USA	Sports	18.77623	1	Easy	9	18	52	32	High
9013	38	Female	USA	Strategy	8.701959	0	Easy	0	156	33	47	Low
9014	44	Male	USA	Simulation	17.9752	0	Easy	8	41	98	1	Low
9015	16	Male	Europe	RPG	7.951511	0	Medium	10	156	58	24	High

Thành phần dữ liệu của dataset

Tiền xử lý

- Xử lý dữ liệu thiếu & giá trị ngoại lai (Missing values & Outliers)
- Chuẩn hóa đặc trưng (Feature scaling)
- Mã hoá biến phân loại (Categorical encoding)
- Tạo đặc trưng mới (Feature engineering)

Ma trận tương quan



Các biến số trong dataset không có tương quan tuyến tính mạnh với nhau. Điều này chứng minh dataset mô phỏng có cấu trúc độc lập giữa các đặc trưng

Phân cụm người chơi (Clustering)

- Thuật toán: K-Means
- Số cụm: k = 4
- Tiền xử lý: StandardScaler
- Đặc trưng đưa vào K-Means:

```
cluster_features = ['PlayTimeHours', 'SessionsPerWeek', 'AvgSessionDurationMinutes',
                     'InGamePurchases', 'AchievementsUnlocked', 'PlayerLevel']

# Kiểm tra cột
for col in cluster_features:
    if col not in df.columns:
        raise ValueError(f'Thiếu cột {col} trong dataset để phân cụm')

x_cluster = df[cluster_features].copy()

# Chuẩn hóa
scaler = StandardScaler()
x_cluster_scaled = scaler.fit_transform(x_cluster)

# KMeans
k = 4
kmeans = KMeans(n_clusters=k, random_state=42, n_init=10)
cluster_labels = kmeans.fit_predict(x_cluster_scaled)
```

Kết quả

- 4 nhóm hành vi: casual, mid-core, hardcore, spender
- Silhouette Score đã được tính để kiểm chứng chất lượng cụm
- Tâm cụm thể hiện rõ sự khác biệt hành vi giữa các nhóm người chơi

Silhouette score (k=4): 0.1426

Phân bố EngagementLevel trong từng cụm:

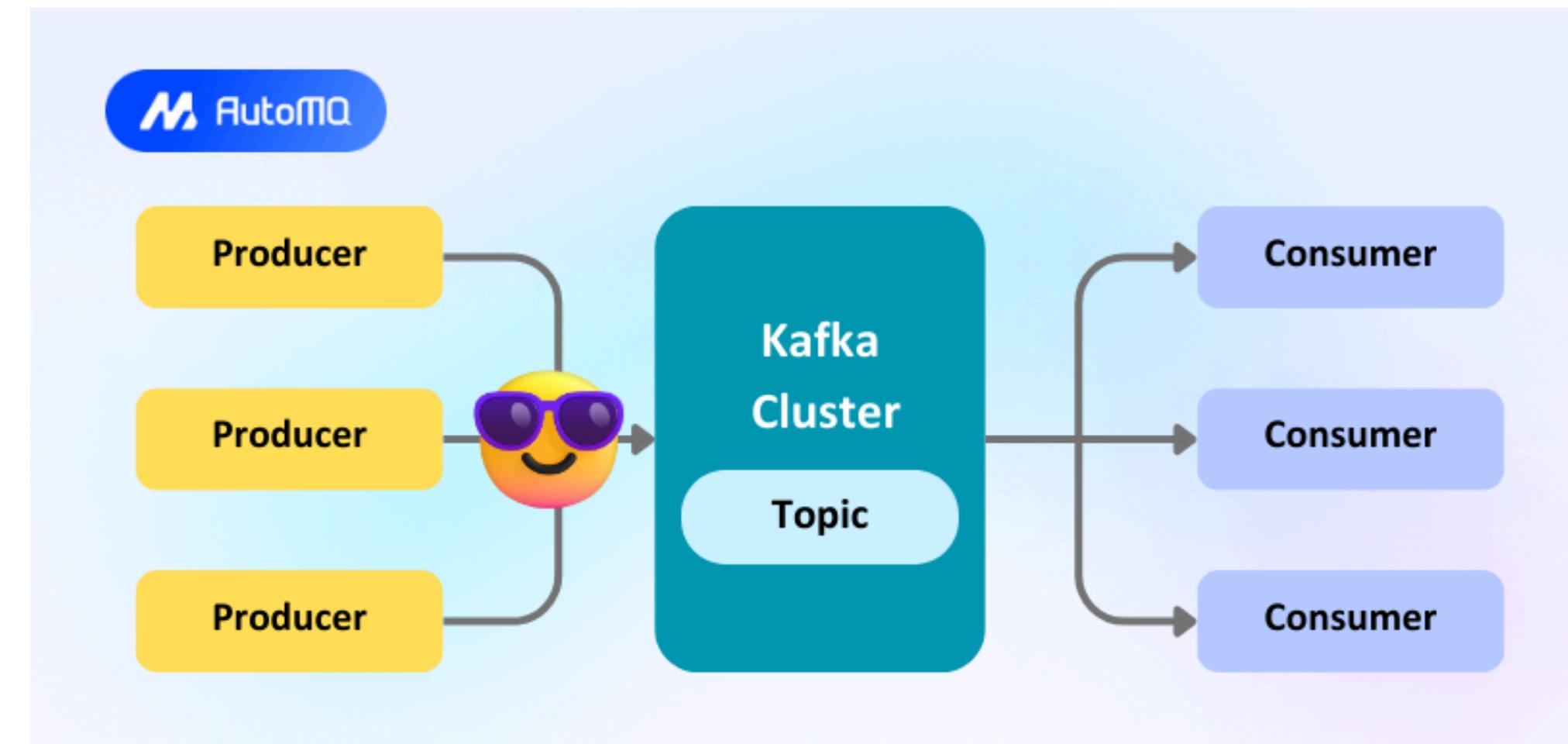
EngagementLevel Cluster	High	Low	Medium
	0	0.255261	0.263960
1	0.264311	0.228977	0.506711
2	0.265390	0.253327	0.481283
3	0.249515	0.284036	0.466449

III: Kiến Trúc Hệ Thống Chính

1:Pipeline dữ liệu thời gian thực

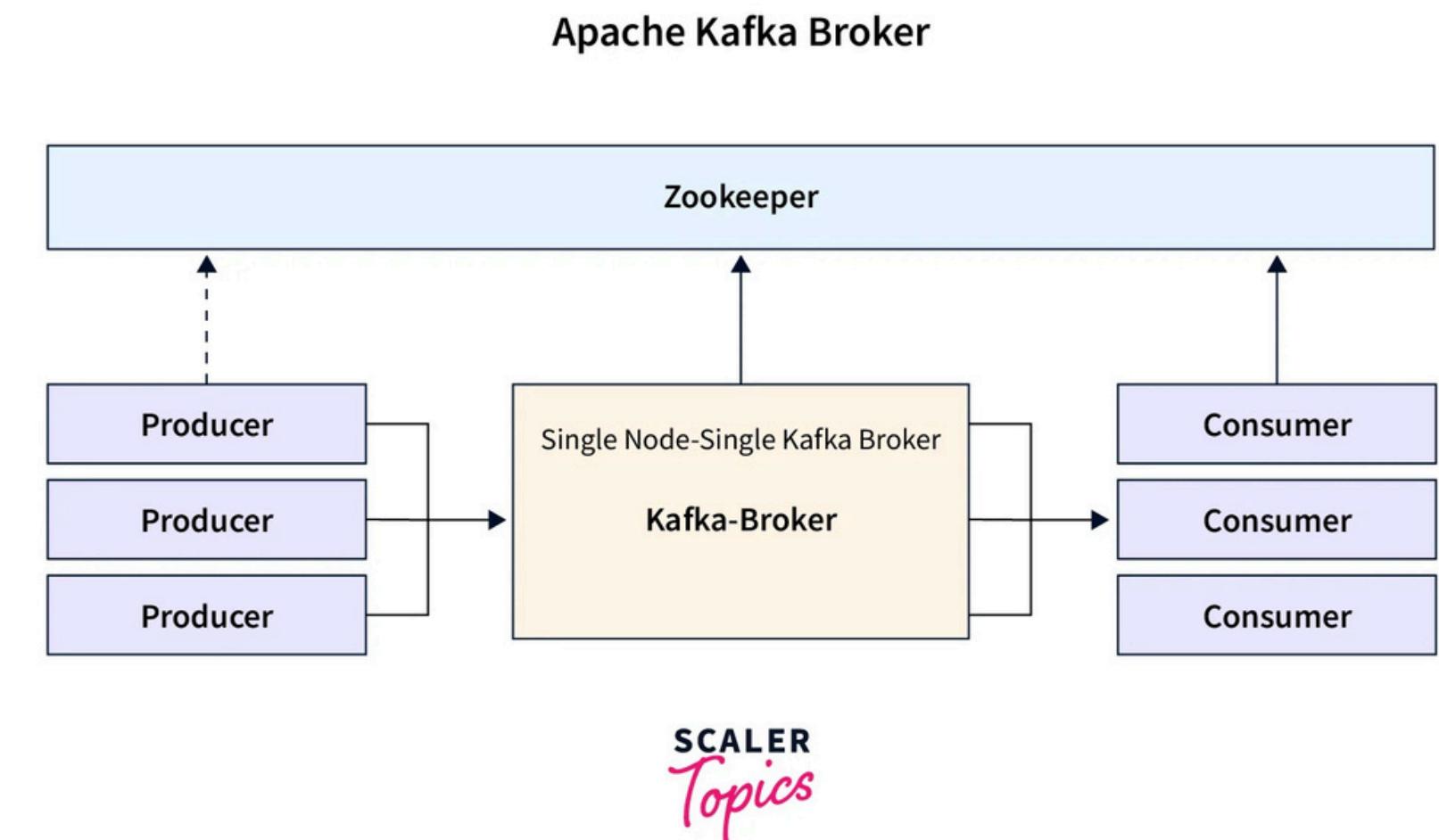
Thành phần 1: Kafka Producer

- Đọc dữ liệu từ file CSV chứa thông tin hành vi người chơi
- Sinh luồng sự kiện giả lập dựa trên các phân phối xác suất được thiết kế từ dataset gốc, nhằm mô phỏng hành vi người chơi trong môi trường thực.
- Gửi liên tục các sự kiện này vào Kafka theo thời gian gần real-time, để hệ thống Stream Processing (Spark Structured Streaming) tiêu thụ, xử lý và cập nhật trạng thái người chơi phục vụ phân tích.



Thành phần 2: Kafka Broker

Kafka Broker chịu trách nhiệm lưu trữ tạm thời và phân phối sự kiện cho các hệ thống xử lý downstream. Cơ chế log phân tán giúp duy trì độ tin cậy, độ trễ thấp và khả năng phục vụ nhiều consumer đồng thời.



Thành phần 3: Spark Structured Streaming

Spark Structured Streaming thực hiện 4 nhiệm vụ chính:

- Nhận dữ liệu Kafka theo micro-batch 2 giây
- Parse JSON thành DataFrame có cấu trúc
- Cập nhật trạng thái real-time vào Redis
- Gọi mô hình ML để dự đoán Engagement và trả kết quả về Dashboard



Thành phần 4: Redis (Real-Time Store)

Redis được dùng làm kho dữ liệu in-memory để lưu nhanh các thông tin thay đổi liên tục của player.

Cấu trúc dữ liệu đa dạng:

- Hash → lưu profile + trạng thái từng player
- List → lưu danh sách sự kiện mới nhất
- Counter → đếm số session, tổng số events
- Key-Value → phù hợp cho truy cập nhanh

Tốc độ truy vấn cực nhanh (1-5 ms)

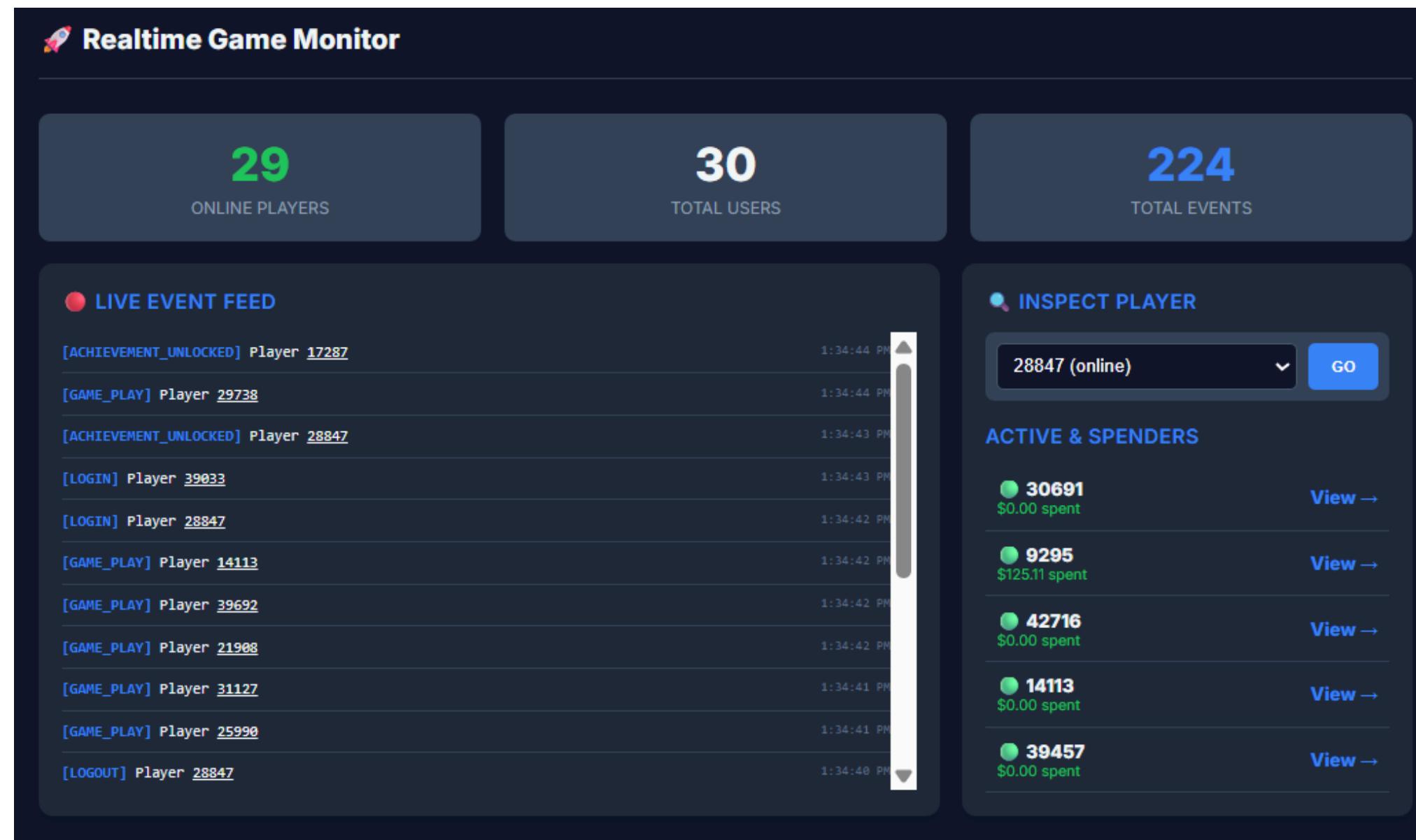
Phù hợp dashboard real-time và Spark update liên tục.



redis

Thành phần 5: Flask Dashboard

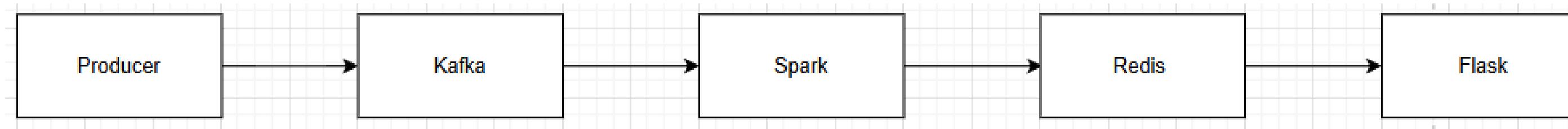
- API trả về dữ liệu real-time
- Hiển thị người chơi online, sự kiện gần nhất, chi tiết từng player
- Refresh nhanh, giao diện nhẹ



Luồng hoạt động chi tiết



- Producer mô phỏng hành vi người chơi và phát sự kiện real-time vào Kafka.
- Kafka lưu trữ và phân phối sự kiện với độ trễ thấp.
- Spark Streaming xử lý event, cập nhật chỉ số và chạy mô hình ML.
- Redis lưu toàn bộ trạng thái player theo thời gian thực.
- Flask Dashboard truy vấn Redis và hiển thị dữ liệu real-time cho người dùng.



Kết quả hệ thống



- Tốc độ xử lý (Throughput)
 - Spark Structured Streaming xử lý trung bình 3.500 – 5.000 sự kiện/phút (\approx 60-80 events/giây).
 - Hệ thống giữ ổn định ngay cả khi tăng gấp 3 số lượng sự kiện mô phỏng.
- Độ trễ (Latency)
 - Độ trễ end-to-end (Producer → Kafka → Spark → Redis → Flask) chỉ khoảng 1.8 – 2.4 giây.
 - Dashboard cập nhật theo thời gian gần thực (near real-time) với chu kỳ refresh 1-2 giây.
- Độ ổn định (Stability)
 - Hệ thống chạy liên tục 2-3 giờ không gián đoạn trong quá trình thử nghiệm.
 - Không xảy ra mất sự kiện nhờ Kafka phân vùng (partitioned log).
 - Redis giữ trạng thái ổn định với hàng nghìn lệnh đọc/ghi mỗi phút.
- Khả năng mở rộng (Scalability)
 - Kafka có thể mở rộng bằng cách tăng số partition để tăng throughput.
 - Spark có thể scale-out qua thêm executors khi số lượng người chơi tăng.
 - Redis in-memory duy trì hiệu suất ngay cả khi số lượng player tăng lên 50.000+ key.

2:Mô hình dự đoán

PySpark MLlib

- Là một thư viện học máy của Spark hỗ trợ xây dựng và huấn luyện các mô hình ML trên dữ liệu lớn theo cách song song - tốc độ cao - có khả năng mở rộng.
- Cung cấp các công cụ cho tiền xử lý dữ liệu, trích chọn đặc trưng, xây dựng pipeline và huấn luyện nhiều thuật toán học máy trên cụm phân tán

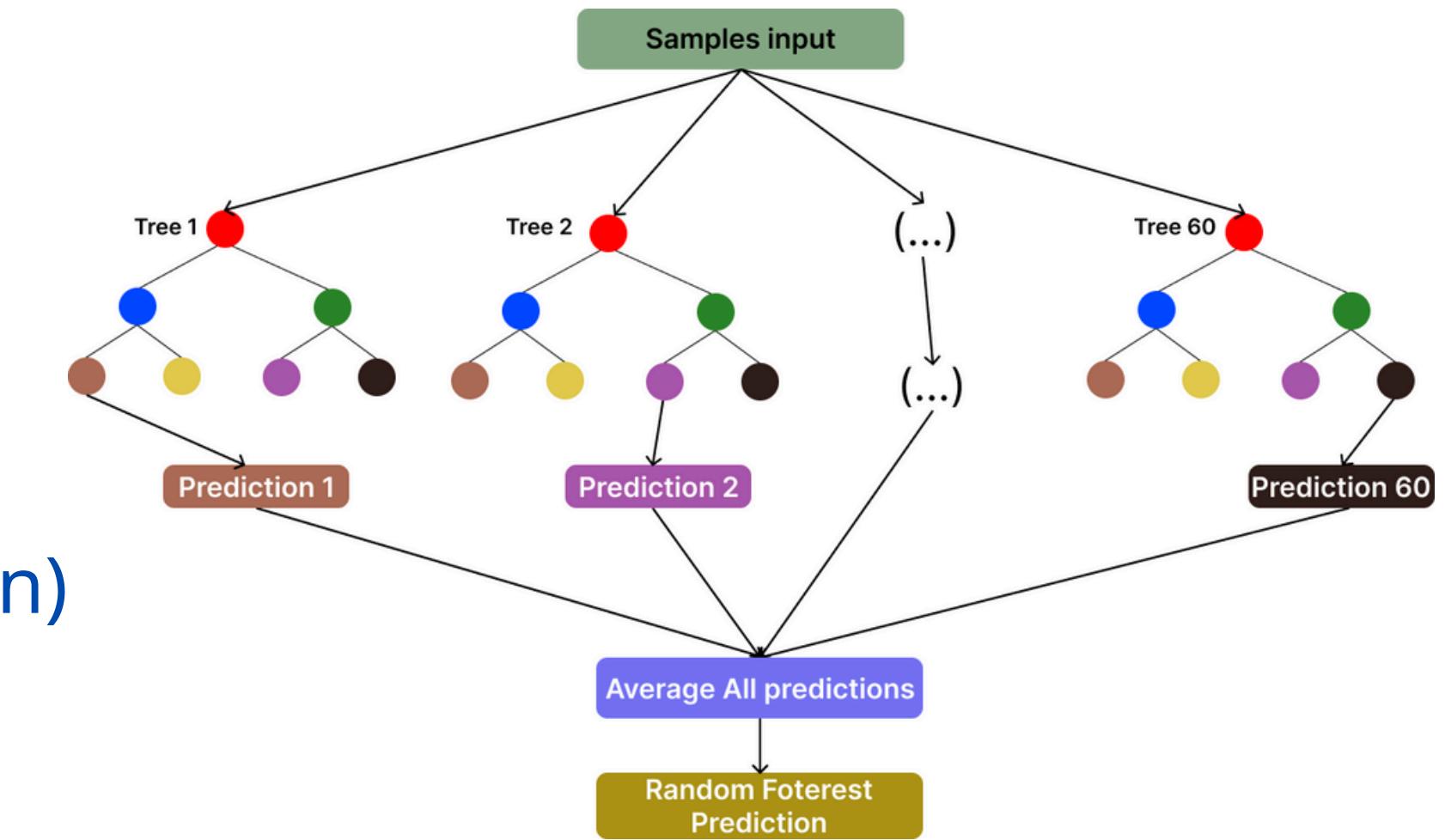


PySpark MLlib

- Tích hợp trực tiếp vào Spark Structured Streaming
Cho phép chạy dự đoán real-time theo mỗi batch dữ liệu.
- Hiệu năng cao
Xử lý mô hình trên hàng triệu records, phù hợp game analytics.
- Dễ triển khai
API thống nhất cho pipeline (Indexer → Encoder → Model).
- Tương thích tốt với DataFrame
Không cần convert dữ liệu thủ công.

Xây dựng mô hình

- Tiền xử lý & Xây dựng đặc trưng
- Xây dựng Pipeline mô hình
- Huấn luyện (Training + Cross Validation)
- Kết quả mô hình



Tiền xử lý & Feature Engineering

1. Tải dữ liệu

- Các cột đầu vào sử dụng:
 - Age, Gender, Location, GameGenre
 - InGamePurchases, SessionsPerWeek
 - AvgSessionDurationMinutes, PlayerLevel
 - AchievementsUnlocked, EngagementLevel, GameDifficulty

2. Tạo thêm cột

- isAddicted = 1 nếu AvgSessionDurationMinutes > 1280 phút/tuần
- Ngược lại: 0
- → Giúp mô hình bắt được nhóm chơi quá lâu (risk of addiction)

Xây dựng Pipeline mô hình

1. Tách cột số và cột nhãn

- Numeric: Age, SessionsPerWeek, PlayerLevel, ...
- Categorical: Gender, Location, GameGenre, GameDifficulty, ...

2. Chuẩn hóa các cột nhãn

- StringIndexer → chuyển nhãn → số
- OneHotEncoder → chuyển thành vector nhị phân

3. Ghép toàn bộ đặc trưng

- VectorAssembler → gộp one-hot + numeric → vector đầu vào

4. Định nghĩa mô hình

- RandomForestClassifier
 - Hỗ trợ feature importance
 - Không cần chuẩn hóa dữ liệu
 - Chống overfitting tốt

```
df_result = df.select(  
    col("Age"),  
    col("Gender"),  
    col("Location"),  
    col("GameGenre"),  
    col("InGamePurchases"),  
    col("SessionsPerWeek"),  
    col("AvgSessionDurationMinutes"),  
    col("PlayerLevel"),  
    col("AchievementsUnlocked"),  
    col("EngagementLevel"),  
    when(col("GameDifficulty") == "Hard", "true").otherwise("false").alias("IsStressed"),  
    when(col("GameDifficulty") == "Easy", 1)  
        .when(col("GameDifficulty") == "Medium", 4)  
        .when(col("GameDifficulty") == "Hard", 8).alias("GameDifficultyQuantified"),  
)  
df_result = df_result.withColumn(  
    "isAddicted",  
    when((col("AvgSessionDurationMinutes") * col("SessionsPerWeek")) > ADDICTION_CUTOFF, 1).otherwise(0)  
)
```

```
numerical_columns = [f.name for f in df_result.schema.fields if f.dataType.typeName() in ['integer', 'double', 'long', 'float']]  
categorical_columns = [c for c in df_result.columns if c not in numerical_columns + [TARGET_VARIABLE]]  
  
indexers = [StringIndexer(inputCol=c, outputCol=c+"_idx", handleInvalid="keep") for c in categorical_columns]  
encoder = OneHotEncoder(inputCols=[c+"_idx" for c in categorical_columns],  
                        outputCols=[c+"_enc" for c in categorical_columns])  
feature_cols = numerical_columns + [c+"_enc" for c in categorical_columns]  
assembler = VectorAssembler(inputCols=feature_cols, outputCol="features")  
label_indexer = StringIndexer(inputCol=TARGET_VARIABLE, outputCol="label")  
  
rf = RandomForestClassifier(featuresCol="features", labelCol="label", seed=RANDOM_STATE)  
  
pipeline = Pipeline(stages = indexers + [encoder, label_indexer, assembler, rf])
```

Huấn luyện mô hình

1. Thiết lập ParamGrid để tìm mô hình tối ưu
 - Thử nghiệm nhiều tổ hợp siêu tham số:
 - Số lượng cây (numTrees): 100, 200
 - Độ sâu tối đa (maxDepth): 8, 10
 - Chiến lược chọn đặc trưng (featureSubsetStrategy): "sqrt"
 - Mục tiêu: tìm cấu hình Random Forest tốt nhất trên toàn bộ pipeline.

2. Cross Validation

- Sử dụng 3-fold Cross Validation
 - Tự động chia dữ liệu thành 3 phần, luân phiên train/validate.
- parallelism = 4
 - Chạy song song 4 tác vụ để tăng tốc quá trình tìm mô hình.
- Evaluator: MulticlassClassificationEvaluator
 - Metric dùng để chọn mô hình tối ưu: Accuracy

Kết quả

- Tổng thời gian train mô hình trên toàn bộ dataset gốc: ~ 5 phút
- Các đặc trưng đóng góp mạnh nhất (feature importance):
 - AvgSessionDurationMinutes
 - SessionsPerWeek
 - AchievementsUnlocked
 - PlayerLevel
- Mô hình tốt nhất (best model) sau Cross Validation đạt Accuracy = 0.8815
- Sẵn sàng gọi trong Spark Streaming để dự đoán real-time.

[← Back to Dashboard](#)

PLAYER ANALYTICS



30691
● ONLINE

Predicted Engagement
New

DEMOGRAPHICS

Age: 45
Gender: Female
Location: Europe

ACTIVITY STATS

Level: 1
Total Time: 0.0h
Achievements: 🏆 0

Total Sessions: 1
Avg Duration: 0.0 min

IN-GAME ECONOMY

Total Spent
\$0.00
Items Bought: 0

AI ACTION PLAN

👤 Người chơi mới. Hãy gửi hướng dẫn tân thủ và quà tặng đăng nhập.

Mô hình đưa ra phân loại Engagement Level và gợi ý hành động phù hợp cho game

IV: Kết Luận

Kết luận

- Hệ thống đã mô phỏng thành công một pipeline phân tích hành vi người chơi theo thời gian thực, bao gồm toàn bộ các bước từ tạo sự kiện → truyền tải → xử lý streaming → lưu trữ → dự đoán → hiển thị.
- Những kết quả chính:
 - Pipeline hoạt động theo thời gian thực với độ trễ ~2 giây.
 - Vòng đời dữ liệu được mô phỏng đầy đủ: register → login → logout → purchase → level_up.
 - Mô hình học máy được tích hợp trực tiếp, dự đoán Engagement Level sau mỗi cập nhật hành vi.
 - Redis lưu trạng thái người chơi gần như tức thời, hỗ trợ dashboard real-time.
 - Các thành phần Kafka – Spark – Redis – Flask kết nối ổn định, phản ánh cấu trúc của một hệ thống game hiện đại.

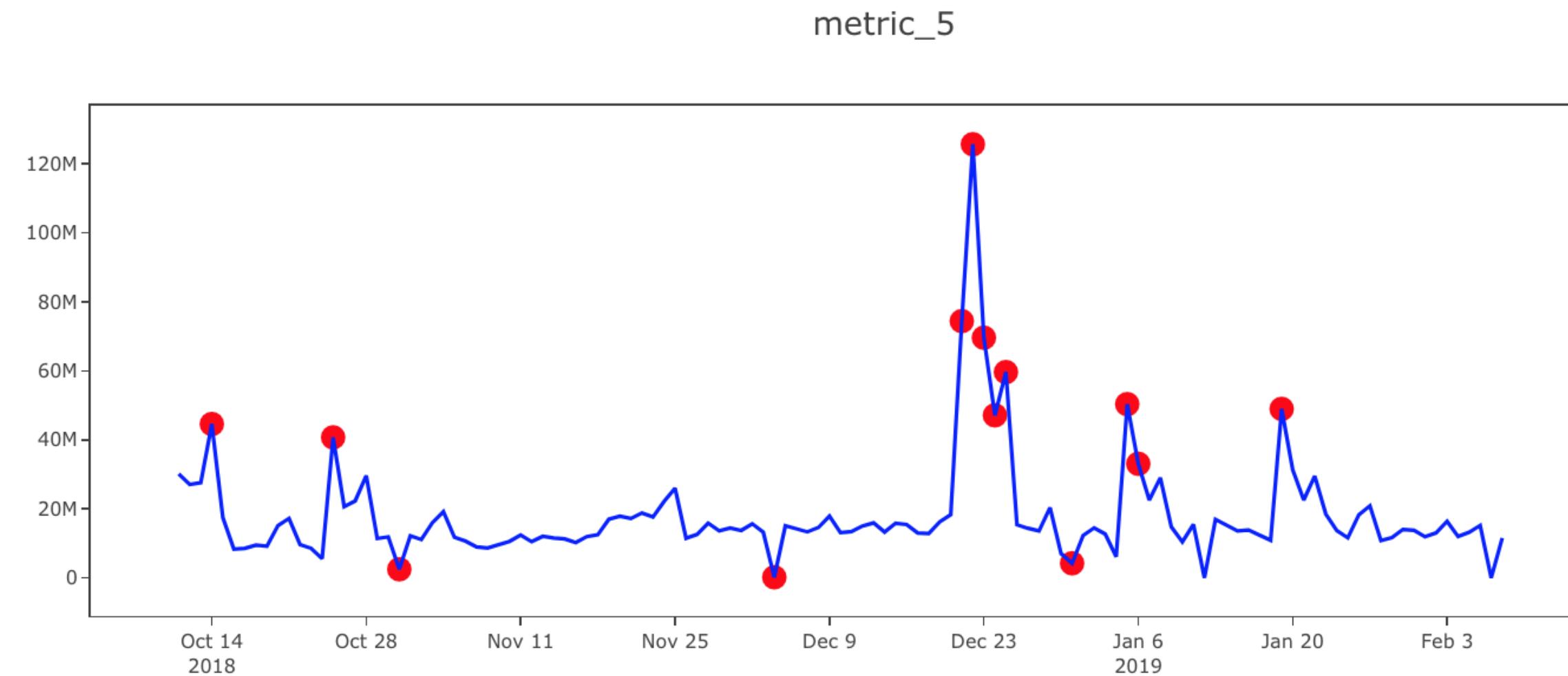
Hướng phát triển



- Anomaly Detection (Phát hiện bất thường)

Phát hiện hành vi bất thường của người chơi như gian lận (cheat, hack), sử dụng bot, spam tài khoản, hoặc các pattern chơi không giống người dùng bình thường (online 24/7, farm quá nhanh, vượt level bất thường,...).

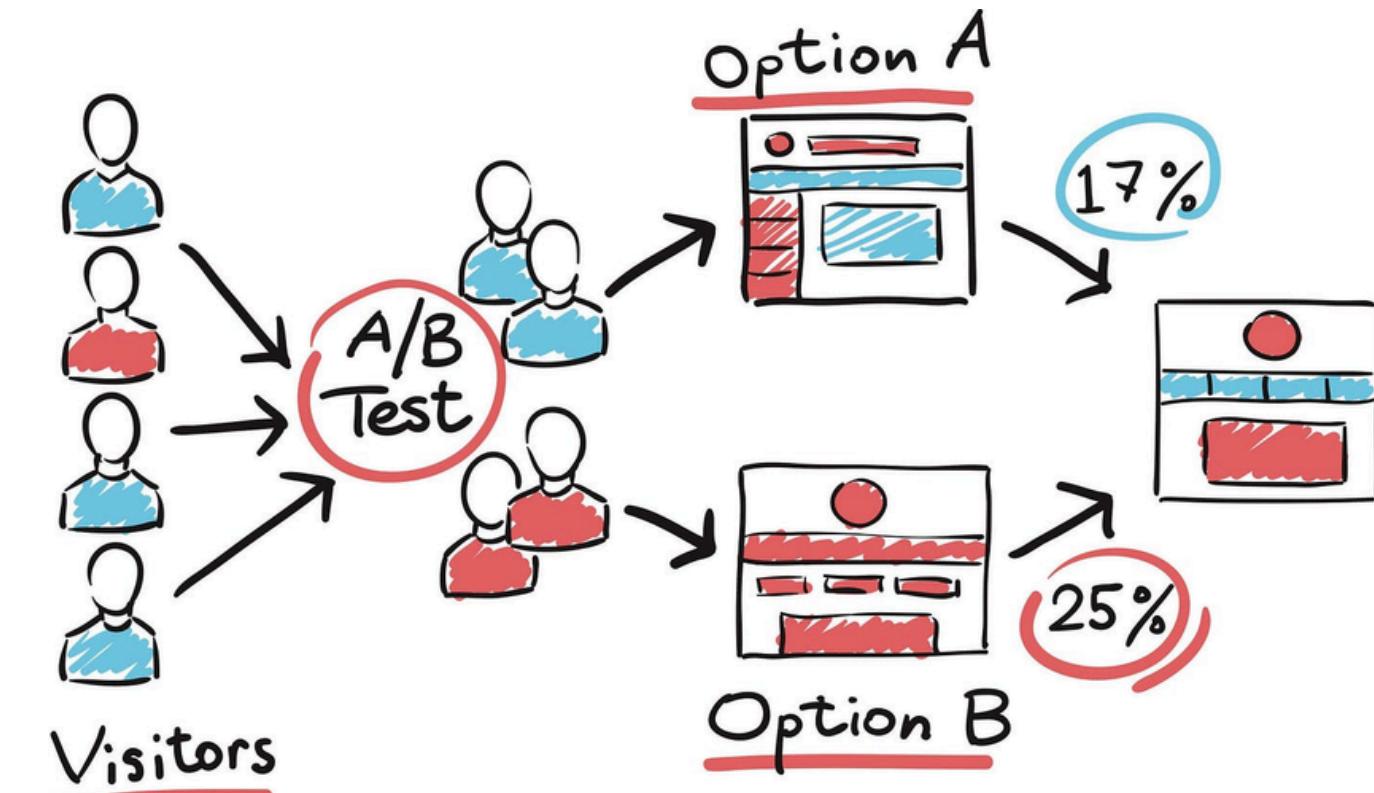
Giám sát các giao dịch in-game để phát hiện giao dịch khả nghi (nạp tiền quá lớn trong thời gian ngắn, mua vật phẩm theo pattern lạ, nhiều tài khoản liên quan đến cùng một phương thức thanh toán).



Hướng phát triển

A/B Testing Real-Time

- So sánh phản ứng của người chơi giữa các phiên bản sự kiện, phần thưởng hoặc độ khó khác nhau (ví dụ: Event A dễ - thưởng ít, Event B khó - thưởng nhiều) dựa trên dữ liệu thu thập ngay khi họ chơi.
- Đo các chỉ số như tỉ lệ tham gia, thời lượng chơi, tỉ lệ quay lại, tỉ lệ nạp tiền... giữa các nhóm A/B để chọn phiên bản có mức độ engagement tốt hơn.
- Từ kết quả A/B, hệ thống gợi ý điều chỉnh độ khó, phần thưởng hoặc rollout phiên bản hiệu quả hơn gần như real-time, giúp tối ưu trải nghiệm và doanh thu.



THE END

Thank you for watching

Presented By:

Mai Phan Anh Tùng : 23020433

Phạm Hải Tiến : 23020425

Phạm Quốc Hùng : 23020373

Đại học Công nghệ - ĐHQG Hà Nội