Streamlined Website Insights: Leveraging Kafka, Redis, MongoDB, Flink, Elasticsearch, and Kibana

Group 2:
- Vidushi
- Kamakshi
- Akanksha
- Spandana

Motivation & Impact

Motivation:
- [illegible]

Impact:
- [illegible]

Project Overview

- [illegible]

Introduction

- The landscape of e-commerce analytics is rapidly evolving, driven by advancements in real-time data processing and machine learning technologies.
- This project outlines our novel approach to capturing and analyzing data from a simulated furniture e-commerce website, where user interactions are synthesized to reflect realistic browsing, shopping, and purchasing behaviors.
- Our goal is to deliver a dual-layered view into user engagement and product performance by utilizing a hybrid model that combines the analytical depth of offline machine learning approaches with the immediacy of real-time stream processing.
- This project is an example that illustrates how transformative the hybrid approach of real-time analysis with offline analysis can be in e-commerce.

Explainable AI + Differential Privacy

[illegible]

Stream Processing

Integration with Flink - Elastic Search- Kibana

[illegible]

Tools Used

Overview of Data Simulated

- User Details
  Fields: userId, name, email, gender.
  Device Info: deviceType, operatingSystem, browserType.
- Session Dynamics
  Browsing: productIds, productNames, productPrices, productCategories.
  Engagement: actions, pageViews, sessionDuration, clicks.
- Purchasing Behavior
  Transactions: purchaseMade, amountSpent, paymentMethodType.
  Reviews: Sentiments (positive, negative, neutral).
- Referrals & Location
  Referrer Sources: Google, Facebook, etc.
  Geographic Data: country, state, city, zipCode.

The data generated from hypothetical website is streamed into Kafka and then to redis where bloom filter is executed.

Bloom Filter & Hyperlog log

- Consume user session data from the Kafka topic user_session_info.
- Bloom Filter: Utilize to quickly check the presence of user IDs and prevent processing duplicates.
- HyperLogLog: Employ to count distinct user IDs without storing them, optimizing memory usage.
- Sessions from users with "unknown" device types or referrers are skipped to ensure data quality.
- New valid users (identified in Bloom Filter) are added for tracking and counting.
- Messages for new and returning users are produced back to Kafka under valid_customer_session if they pass validation checks.

Batch Processing

MongoDB- LSH

- [illegible]

Technical difficulties ,Key Learnings

Technical Difficulties:
- We encountered significant challenges in synchronizing data across multiple platforms including Redis, MongoDB, Flink, Kafka, Elasticsearch, and Kibana, necessitating a complete redesign of our data pipeline architecture.
- Additionally, scaling the system to handle large volumes of synthetic data required optimization of our data processing methods.
- Implementing differential privacy while maintaining accurate analytics was also challenging, as it involved securing user information while extracting valuable insights.

Key Learnings:
- Effectively merging real-time data processing with thorough offline analysis.
- Learnt how to seamlessly integrate technologies such as Apache Kafka, Redis, MongoDB, Apache Flink, Elasticsearch, and Kibana was key to creating a robust analytics platform.
- Implementing differential privacy techniques within machine learning models taught us the importance of protecting user data while still being able to extract meaningful insights.
- We also looked into scalability problems, which resulted in early design thoughts on how to handle vast amounts of data efficiently.

Introduction

- The landscape of e-commerce analytics is rapidly evolving, driven by advancements in real-time data processing and machine learning technologies.
- This project outlines our novel approach to capturing and analyzing data from a simulated furniture e-commerce website, where user interactions are synthesized to reflect realistic browsing, shopping, and purchasing behaviors.
- Our goal is to deliver a dual-layered view into user engagement and product performance by utilizing a hybrid model that combines the analytical depth of offline machine learning approaches with the immediacy of real-time stream processing.
- This project is an example that illustrates how transformative the hybrid approach of real-time analysis with offline analysis can be in e-commerce.

Motivation & Impact

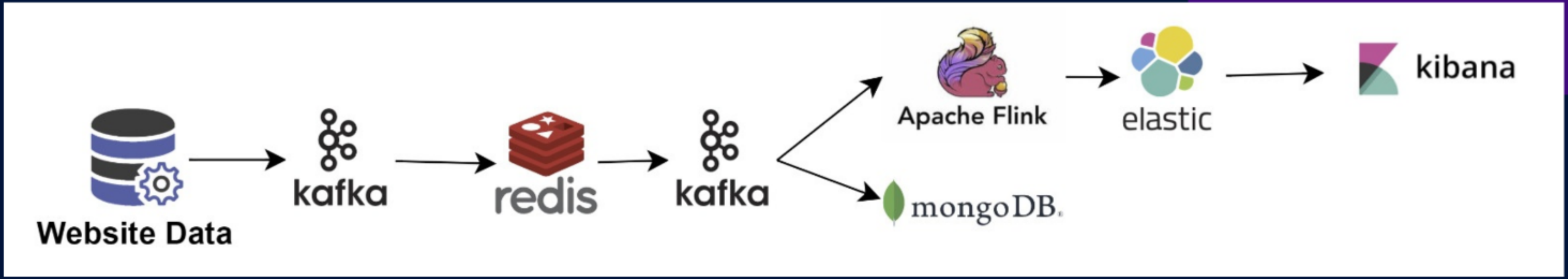
Motivation:

- In the rapidly evolving landscape of e-commerce, businesses are increasingly reliant on technology to understand consumer behavior, anticipate needs, and enhance user satisfaction.
- However, the challenge often lies in balancing the need for immediate response with the requirement for deep, strategic insight.
- Our motivation stems from the need to bridge this gap—integrating real-time responsiveness with comprehensive analytical depth.

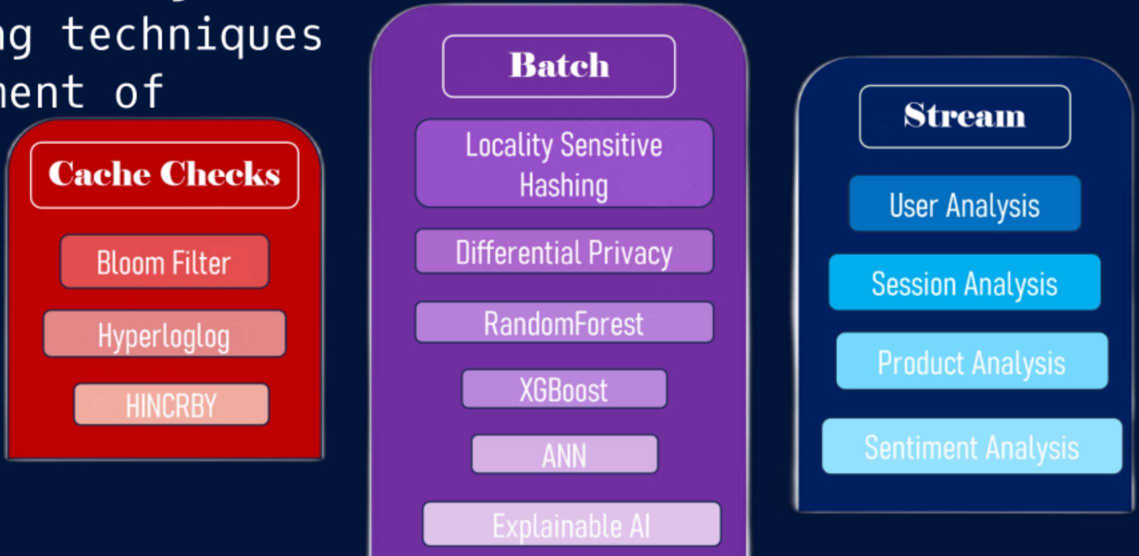
Impact:

- The implementation of our hybrid analytical framework stands to transform e-commerce platforms by providing a more holistic view of the customer journey.
- The dual benefits include enhanced user experiences through personalized interactions and improved business outcomes through optimized product offerings and marketing strategies.
- This approach not only satisfies the immediate expectations of today's online shoppers but also looks ahead, predicting future trends. This sets a new benchmark for responsiveness and understanding in the e-commerce world.

Project Overview



- Initially, website data encapsulating user interactions is collected and streamed in real-time using Apache Kafka.
- This stream is then interfaced with Redis, where Bloom Filter & Hyperlog log are implemented and then valid user data is again loaded into another kafka topic.
- Following this, the valid users data is stored in MongoDB for advanced offline batch processing, including techniques such as Locality Sensitive Hashing and development of models.
- Parallelly, the data is channeled into Apache Flink and Elasticsearch, to generate real time insights and the visual representations are crafted using Kibana.



Streamlined Website Insights: Leveraging Kafka, Redis, MongoDB, Flink, Elasticsearch, and Kibana

Group 2:
- Vidushi
- Kamakshi
- Akanksha
- Spandana

Introduction

- The landscape of e-commerce analytics is rapidly evolving, driven by advancements in real-time data processing and machine learning technologies.
- This project outlines our novel approach to capturing and analyzing data from a simulated furniture e-commerce website, where user interactions are synthesized to reflect realistic browsing, shopping, and purchasing behaviors.
- Our goal is to deliver a dual-layered view into user engagement and product performance by utilizing a hybrid model that combines the analytical depth of offline machine learning approaches with the immediacy of real-time stream processing.
- This project is an example that illustrates how transformative the hybrid approach of real-time analysis with offline analysis can be in e-commerce.

Tools Used

Overview of Data Simulated

- User Details
  Fields: userId, name, email, gender.
  Device Info: deviceType, operatingSystem, browserType.
- Session Dynamics
  Browsing: productId, productNames, productPrices, productCategories.
  Engagement: actions, pageViews, sessionDuration, clicks.
- Purchasing Behavior
  Transactions: purchaseMade, amountSpent, paymentMethodType.
  Reviews: Sentiments (positive, negative, neutral).
- Referrals & Location
  Referrer Sources: Google, Facebook, etc.
  Geographic Data: country, state, city, zipCode.

[illegible]

Bloom Filter & Hyperlog log

[illegible]

Batch Processing

MongoDB- LSH

[illegible]

Technical difficulties ,Key Learnings

Technical Difficulties:
- We encountered significant challenges in synchronizing data across multiple platforms including Redis, MongoDB, Flink, Kafka, Elasticsearch, and Kibana, necessitating a complete redesign of our data pipeline architecture.
- Additionally, scaling the system to handle large volumes of synthetic data required optimization of our data processing methods.
- Implementing differential privacy while maintaining accurate analytics was also challenging, as it involved securing user information while extracting valuable insights.

Key Learnings:
- Effectively merging real-time data processing with thorough offline analysis.
- Learnt how to seamlessly integrate technologies such as Apache Kafka, Redis, MongoDB, Apache Flink, Elasticsearch, and Kibana was key to creating a robust analytics platform.
- Implementing differential privacy techniques within machine learning models taught us the importance of protecting user data while still being able to extract meaningful insights.
- We also looked into scalability problems, which resulted in early design thoughts on how to handle vast amounts of data efficiently.

Overview of Data Simulated

- User Details
  Fields: userId, name, email, gender.
  Device Info: deviceType, operatingSystem, browserType.
- Session Dynamics
  Browsing: productIds, productNames, productPrices, productCategories.
  Engagement: actions, pageViews, sessionDuration, clicks.
- Purchasing Behavior
  Transactions: purchaseMade, amountSpent, paymentMethodType.
  Reviews: Sentiments (positive, negative, neutral).
- Referrals & Location
  Referrer Sources: Google, Facebook, etc.
  Geographic Data: country, state, city, zipCode.

Bloom Filter & Hyperlog log

The data generated from hypothetical website is streamed into kafka and then to redis where bloom filter is executed.

- Consume user session data from the Kafka topic user_session_info.
- Bloom Filter: Utilize to quickly check the presence of user IDs and prevent processing duplicates.
- HyperLogLog: Employ to count distinct user IDs without storing them, optimizing memory usage.
- Sessions from users with 'unknown' device types or referrers are skipped to ensure data quality.
- New valid users (identified by Bloom filter) are added for tracking and counting.
- Messages for new and returning users are produced back to Kafka under valid_customer_session if they pass validation checks.

MongoDB- LSH

Batch Processing

- The valid user session information is dumped into Mongodb and then processing is done.
- Implemented Locality-Sensitive Hashing (LSH) to identify similar users.
- It works by hashing similar input items into the same "buckets" with high probability, allowing for the identification of items that are similar to a query item quickly.
- This implementation of Locality-Sensitive Hashing (LSH) not only improves the efficiency of our system but also enhances user experience by quickly identifying and connecting users with similar interests or behaviors.

Streamlined Website Insights: Leveraging Kafka, Redis, MongoDB, Flink, Elasticsearch, and Kibana

Group 2:
- Vidushi
- Kamakshi
- Akanksha
- Spandana

Motivation & Impact

[illegible]

Project Overview

[illegible]

Introduction

- The landscape of e-commerce analytics is rapidly evolving, driven by advancements in real-time data processing and machine learning technologies.
- This project outlines our novel approach to capturing and analyzing data from a simulated furniture e-commerce website, where user interactions are synthesized to reflect realistic browsing, shopping, and purchasing behaviors.
- Our goal is to deliver a dual-layered view into user engagement and product performance by utilizing a hybrid model that combines the analytical depth of offline machine learning approaches with the immediacy of real-time stream processing.
- This project is an example that illustrates how transformative the hybrid approach of real-time analysis with offline analysis can be in e-commerce.

Explainable AI + Differential Privacy

[illegible]

Integration with Flink - Elastic Search- Kibana

[illegible]

Tools Used

Overview of Data Simulated

- User Details
  Fields: userId, name, email, gender.
  Device Info: deviceType, operatingSystem, browserType.
- Session Dynamics
  Browsing: productIds, productNames, productPrices, productCategories.
  Engagement: actions, pageViews, sessionDuration, clicks.
- Purchasing Behavior
  Transactions: purchaseMade, amountSpent, paymentMethodType.
  Reviews: Sentiments (positive, negative, neutral).
- Referrals & Location
  Referrer Sources: Google, Facebook, etc.
  Geographic Data: country, state, city, zipCode.

Bloom Filter & Hyperlog log

[illegible]

MongoDB- LSH

[illegible]

Technical difficulties ,Key Learnings

Technical Difficulties:
- We encountered significant challenges in synchronizing data across multiple platforms including Redis, MongoDB, Flink, Kafka, Elasticsearch, and Kibana, necessitating a complete redesign of our data pipeline architecture.
- Additionally, scaling the system to handle large volumes of synthetic data required optimization of our data processing methods.
- Implementing differential privacy while maintaining accurate analytics was also challenging, as it involved securing user information while extracting valuable insights.

Key Learnings:
- Effectively merging real-time data processing with thorough offline analysis.
- Learnt how to seamlessly integrate technologies such as Apache Kafka, Redis, MongoDB, Apache Flink, Elasticsearch, and Kibana was key to creating a robust analytics platform.
- Implementing differential privacy techniques within machine learning models taught us the importance of protecting user data while still being able to extract meaningful insights.
- We also looked into scalability problems, which resulted in early design thoughts on how to handle vast amounts of data efficiently.

Explainble AI + Differential Privacy

- Developed two machine learning models and one deep learning model aimed at predicting customer purchasing behavior.
- These models are meticulously crafted to analyze various customer data points and predict whether a customer is likely to make a purchase.
- In our model, we use differential privacy to add 'noise' to the data in a controlled way. This is done by the Laplace mechanism that we applied to the numerical features in our dataset.
- Differential Privacy is crucial when working with user data because it helps to protect individual privacy without sacrificing the utility of the dataset as a whole.
- In our project, we used SHAP (SHapley Additive exPlanations), which is an approach to explain the output of machine learning models.
- We apply SHAP to interpret the predictions of our models, helping us to understand which features are most influential in predicting whether a user will make a purchase.

Stream Processing

Integration with Flink - Elastic Search- Kibana

- Consumes user session data from the valid_customer_session Kafka topic.
- Flink processes streaming data from Kafka, handling user session data as it flows in real-time.
- By using Flink, Converts each message into a session summary, identifying key user activities and product interactions.
- Analyzes sentiments in product reviews classifying them into Elasticsearch's product_sentiment index.
- Elasticsearch allows for the efficient indexing and storage of large volumes of data processed by Flink enabling real-time insights and visualization through Kibana.

Streamlined Website Insights: Leveraging Kafka, Redis, MongoDB, Flink, Elasticsearch, and Kibana

Group 2:
- Vidushi
- Kamakshi
- Akanksha
- Spandana

Motivation & Impact

[illegible]

Project Overview

[illegible]

Introduction

- The landscape of e-commerce analytics is rapidly evolving, driven by advancements in real-time data processing and machine learning technologies.
- This project outlines our novel approach to capturing and analyzing data from a simulated furniture e-commerce website, where user interactions are synthesized to reflect realistic browsing, shopping, and purchasing behaviors.
- Our goal is to deliver a dual-layered view into user engagement and product performance by utilizing a hybrid model that combines the analytical depth of offline machine learning approaches with the immediacy of real-time stream processing.
- This project is an example that illustrates how transformative the hybrid approach of real-time analysis with offline analysis can be in e-commerce.

Explainable AI + Differential Privacy

[illegible]

Stream Processing

Integration with Flink - Elastic Search- Kibana

[illegible]

Tools Used

Overview of Data Simulated

- User Details
  Fields: userId, name, email, gender.
  Device Info: deviceType, operatingSystem, browserType.
- Session Dynamics
  Browsing: productIds, productNames, productPrices, productCategories.
  Engagement: actions, pageViews, sessionDuration, clicks.
- Purchasing Behavior
  Transactions: purchaseMade, amountSpent, paymentMethodType.
  Reviews: Sentiments (positive, negative, neutral).
- Referrals & Location
  Referrer Sources: Google, Facebook, etc.
  Geographic Data: country, state, city, zipCode.

The data generated from hypothetical website is streamed into Kafka and then to Redis where Bloom filter is executed.

Bloom Filter & Hyperlog log

- Consumes user session data from the Kafka topic user_session_info.
- Bloom Filter: Utilize to quickly check the presence of user IDs and prevent processing duplicates.
- HyperLogLog: Employ to count distinct user IDs without storing them, optimizing memory usage.
- Sessions from users with 'unknown' device types or referrers are skipped to ensure data quality.
- New valid users (identified by Bloom Filter) are added for tracking and counting.
- Messages for new and returning users are produced back to Kafka under valid_customer_session if they pass validation checks.

Batch Processing

MongoDB- LSH

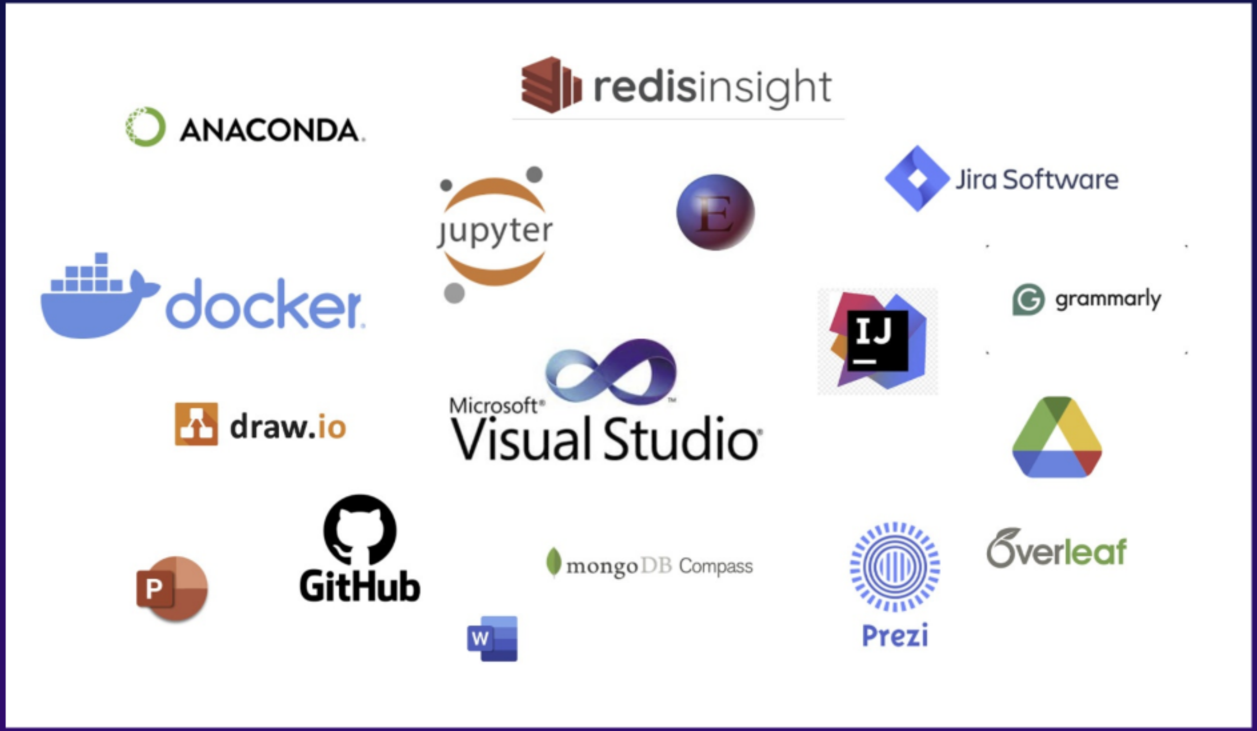
- The valid user session information is dumped into MongoDB and then processing is done.
- Implemented Locality-Sensitive Hashing (LSH) to identify similar users.
- It works by hashing similar input items into the same "buckets" with high probability, allowing for the identification of items that are similar to a query item quickly.
- This implementation of Locality-Sensitive Hashing (LSH) not only improves the efficiency of our system but also enhances user experience by quickly identifying and connecting users with similar interests or behaviors.

Technical difficulties ,Key Learnings

Technical Difficulties:
- We encountered significant challenges in synchronizing data across multiple platforms including Redis, MongoDB, Flink, Kafka, Elasticsearch, and Kibana, necessitating a complete redesign of our data pipeline architecture.
- Additionally, scaling the system to handle large volumes of synthetic data required optimization of our data processing methods.
- Implementing differential privacy while maintaining accurate analytics was also challenging, as it involved securing user information while extracting valuable insights.

Key Learnings:
- Effectively merging real-time data processing with thorough offline analysis.
- Learnt how to seamlessly integrate technologies such as Apache Kafka, Redis, MongoDB, Apache Flink, Elasticsearch, and Kibana was key to creating a robust analytics platform.
- Implementing differential privacy techniques within machine learning models taught us the importance of protecting user data while still being able to extract meaningful insights.
- We also looked into scalability problems, which resulted in early design thoughts on how to handle vast amounts of data efficiently.

Tools Used


redisinsight
ANACONDA.
jupyter
E
Jira Software
docker
IJ
grammarly
draw.io
Microsoft® Visual Studio
P
GitHub
mongoDB Compass
W
Prezi
Overleaf

Technical difficulties ,Key Learnings

Technical Difficulties:

- We encountered significant challenges in synchronizing data across multiple platforms including Redis, MongoDB, Flink, Kafka, Elasticsearch, and Kibana, necessitating a complete redesign of our data pipeline architecture.
- Additionally, scaling the system to handle large volumes of synthetic data required optimization of our data processing methods.
- Implementing differential privacy while maintaining accurate analytics was also challenging, as it involved securing user information while extracting valuable insights.

Key Learnings:

- Effectively merging real-time data processing with thorough offline analysis.
- Learnt how to seamlessly integrate technologies such as Apache Kafka, Redis, MongoDB, Apache Flink, Elasticsearch, and Kibana was key to creating a robust analytics platform.
- Implementing differential privacy techniques within machine learning models taught us the importance of protecting user data while still being able to extract meaningful insights.
- We also looked into scalability problems, which resulted in early design thoughts on how to handle vast amounts of data efficiently.

Streamlined Website Insights: Leveraging Kafka, Redis, MongoDB, Flink, Elasticsearch, and Kibana

Group 2:
- Vidushi
- Kamakshi
- Akanksha
- Spandana

Motivation & Impact

[illegible]

Project Overview

[illegible]

Introduction

- The landscape of e-commerce analytics is rapidly evolving, driven by advancements in real-time data processing and machine learning technologies.
- This project outlines our novel approach to capturing and analyzing data from a simulated furniture e-commerce website, where user interactions are synthesized to reflect realistic browsing, shopping, and purchasing behaviors.
- Our goal is to deliver a dual-layered view into user engagement and product performance by utilizing a hybrid model that combines the analytical depth of offline machine learning approaches with the immediacy of real-time stream processing.
- This project is an example that illustrates how transformative the hybrid approach of real-time analysis with offline analysis can be in e-commerce.

Explainable AI + Differential Privacy

[illegible]

Integration with Flink - Elastic Search- Kibana

[illegible]

Tools Used

Overview of Data Simulated

- User Details
  Fields: userId, name, email, gender.
  Device Info: deviceType, operatingSystem, browserType.
- Session Dynamics
  Browsing: productIds, productNames, productPrices, productCategories.
  Engagement: actions, pageViews, sessionDuration, clicks.
- Purchasing Behavior
  Transactions: purchaseMade, amountSpent, paymentMethodType.
  Reviews: Sentiments (positive, negative, neutral).
- Referrals & Location
  Referrer Sources: Google, Facebook, etc.
  Geographic Data: country, state, city, zipCode.

The data generated from hypothetical website is streamed into Kafka and then to redis where Bloom filter is executed.

Bloom Filter & Hyperlog log

[illegible]

Batch Processing

MongoDB- LSH

[illegible]

Technical difficulties ,Key Learnings

Technical Difficulties:
- We encountered significant challenges in synchronizing data across multiple platforms including Redis, MongoDB, Flink, Kafka, Elasticsearch, and Kibana, necessitating a complete redesign of our data pipeline architecture.
- Additionally, scaling the system to handle large volumes of synthetic data required optimization of our data processing methods.
- Implementing differential privacy while maintaining accurate analytics was also challenging, as it involved securing user information while extracting valuable insights.

Key Learnings:
- Effectively merging real-time data processing with thorough offline analysis.
- Learnt how to seamlessly integrate technologies such as Apache Kafka, Redis, MongoDB, Apache Flink, Elasticsearch, and Kibana was key to creating a robust analytics platform.
- Implementing differential privacy techniques within machine learning models taught us the importance of protecting user data while still being able to extract meaningful insights.
- We also looked into scalability problems, which resulted in early design thoughts on how to handle vast amounts of data efficiently.