

CIS602 – Big Data Analytics

Sales and Customer Insights of E-commerce

FINAL PROJECT REPORT

Pradyoth Singenahalli Prabhu

Roll No: 54

Student ID: 02071847

Data Science
University Massachusetts
Dartmouth.

Varun W R

Roll No: 64

Student ID: 02070206

Data Science
University Massachusetts
Dartmouth.

Bharath Anand

Roll No: 02

Student ID: 02044023

Data Science
University Massachusetts
Dartmouth.

Abstract:

This report encapsulates the outcomes of the "Sales and Customer Insights" project, where advanced data technologies, including Kafka for real-time data streaming, Apache Spark for large-scale data processing, and Tableau for intuitive visualization, were employed to extract actionable insights from sales data and customer behaviors. The project focused on continuous data ingestion, dynamic analysis, and comprehensive exploration, revealing key patterns such as customer segmentation, product performance, and cross-selling opportunities. The integration of Tableau facilitated the creation of interactive visualizations, enhancing the communication of complex findings. The insights derived from this analysis have substantial implications for business strategy, offering recommendations for optimizing sales, marketing approaches, and customer relationship management. This abstract provides a brief overview of the project's methodology, results, and actionable recommendations, demonstrating the cohesive utilization of Kafka, Apache Spark, and Tableau in generating valuable insights from sales and customer data.

Introduction:

The "Sales and Customer Insights" project represents a strategic initiative within our organization aimed at harnessing the power of advanced data technologies to gain deeper insights into sales performance and customer behaviors. In the dynamic landscape of E-commerce, understanding customer preferences, optimizing sales strategies, and enhancing overall business performance have become imperative. This project was motivated with the goal of providing timely and meaningful insights to support informed decision-making.

In this report, we delve into the methodologies and findings of our analysis, leveraging key technologies such as Kafka for real-time data streaming, Apache Spark for large-scale data processing, and Tableau for intuitive visualization. The dataset, originating from Kaggle, provides eCommerce behavior data from a multi-category store, comprising 14.68 GB of information across 9 columns. As we navigate through the intricacies of sales and customer data, we will explore patterns, identify opportunities for improvement, and make actionable recommendations. The scope of our analysis encompasses specific aspects of sales and customer data.

Our primary audience includes E-commerce customers, for whom the insights derived from this analysis will be instrumental. This introduction sets the stage for an in-depth exploration of our project, providing a roadmap for the subsequent sections where we detail our methodology, present results, and offer recommendations for leveraging the gained insights.

Architecture:

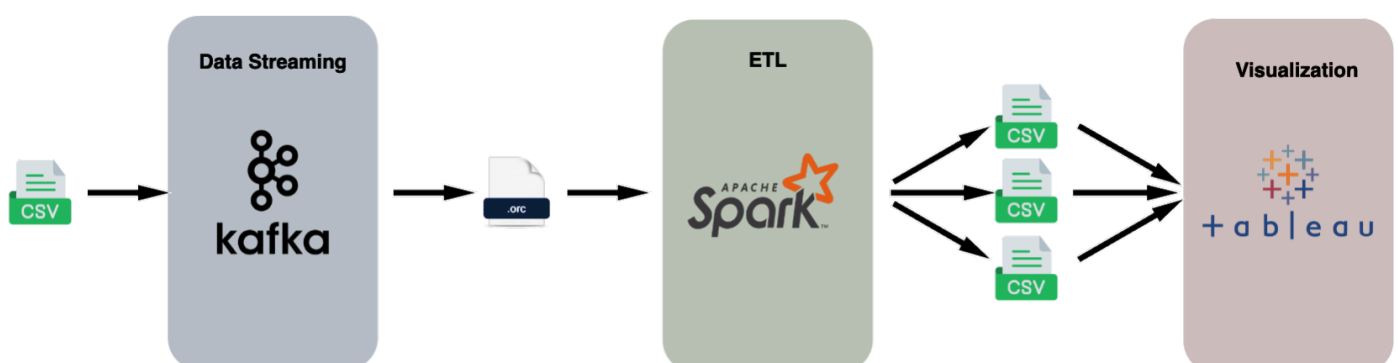


Fig1

The architecture of our "Sales and Customer Insights" project encompasses a cohesive data processing pipeline, commencing with the acquisition of CSV files from Kaggle. Fig1 shows the overall architecture of our project. These files, capturing crucial information on sales and customer interactions, undergo real-time streaming via Apache Kafka, a resilient distributed platform. Here the files are converted to ORC format and then fed to next part. Following this, Apache Spark takes center stage for data transformation, through Extract, Transform, Load (ETL) processes. The result is a set of refined and enriched data stored in multiple CSV files. Subsequently, Tableau is employed for data visualization, utilizing these files as a source to generate interactive and insightful visualizations. This architecture, integrating Kafka, Apache Spark, and Tableau, ensures a seamless and efficient journey from data ingestion to visualization, paving the way for a detailed exploration of our methodologies and project insights in the following sections of this report.

Data Collection and Sources:

Kafka:

Apache Kafka is a cornerstone project, playing a pivotal role as a distributed streaming platform that significantly enhances the efficiency and responsiveness of our data processing pipeline. At the project's inception, Kafka takes center stage by seamlessly handling the continuous streams of data sourced from Kaggle, specifically eCommerce behavior data from a multi-category store. Its key feature of real-time data streaming ensures that CSV files from Kaggle are ingested promptly, providing the foundation for dynamic and timely analysis of sales and customer behaviors. Kafka's distributed and scalable architecture is particularly crucial, allowing for horizontal scalability to accommodate our substantial dataset of 14.68 GB and future data growth seamlessly. Beyond its scalability, Kafka's fault tolerance and durability features

contribute to the overall resilience of our data streaming process, safeguarding against data loss even in the event of node failures. In our project architecture, Kafka serves as a robust intermediary, facilitating the seamless transport of CSV files from Kaggle to Apache Spark for subsequent processing. Its role extends beyond mere data ingestion; it establishes the reliability, fault tolerance, and scalability necessary for a resilient and efficient data processing infrastructure. The subsequent sections of this report will delve into the methodologies employed in conjunction with Kafka, providing a detailed account of our data processing journey. Fig2 shows Kafka architecture

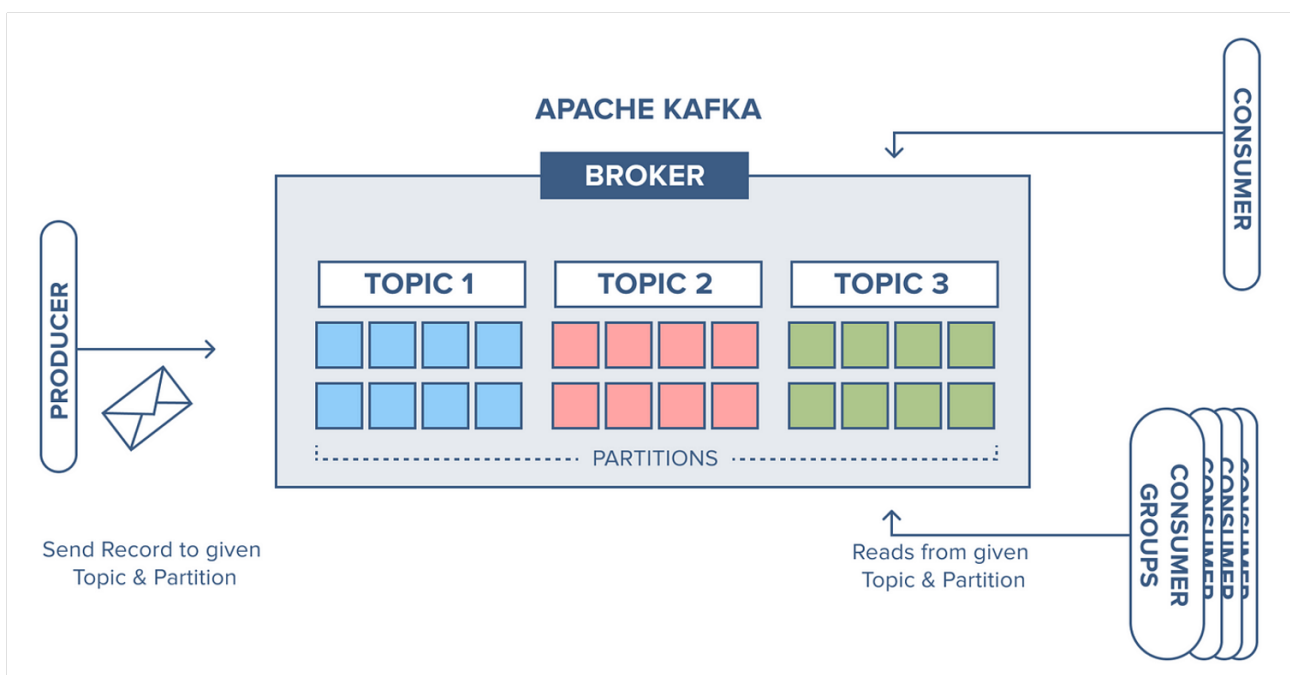


Fig 2

Kafka Producer:

The Kafka producer script is the catalyst for our "Sales and Customer Insights" data pipeline, initiating the entire process by seamlessly reading CSV data and transmitting it to the designated Kafka topic. As the first step in our architecture, its pivotal role lies in facilitating real-time data streaming, ensuring the continuous ingestion of CSV data from Kaggle into the Kafka platform. Beyond its initiation function, the Kafka producer optimizes efficiency by operating with a specified batch size of 5000 rows, streamlining the transmission of large datasets and contributing to the overall responsiveness

of our data pipeline. This producer-script-driven data conduit plays a central role in bridging the gap between the initial data source and Kafka, setting the stage for subsequent processing and analysis

Kafka Topic:

A Kafka topic serves as a fundamental organizational entity, categorizing and organizing messages transmitted by producers. It acts as a crucial mechanism for structuring and managing the flow of data through the Kafka platform. Key Kafka configurations, notably partitioning and replication, play a pivotal role in shaping the distribution of messages within these topics. Partitioning allows for the parallel processing of messages, enhancing throughput and scalability, while replication ensures fault tolerance by duplicating data across multiple brokers. Together, these configurations contribute to the reliability, efficiency, and resilience of our data streaming architecture. Fig3 shows Kafka topic.

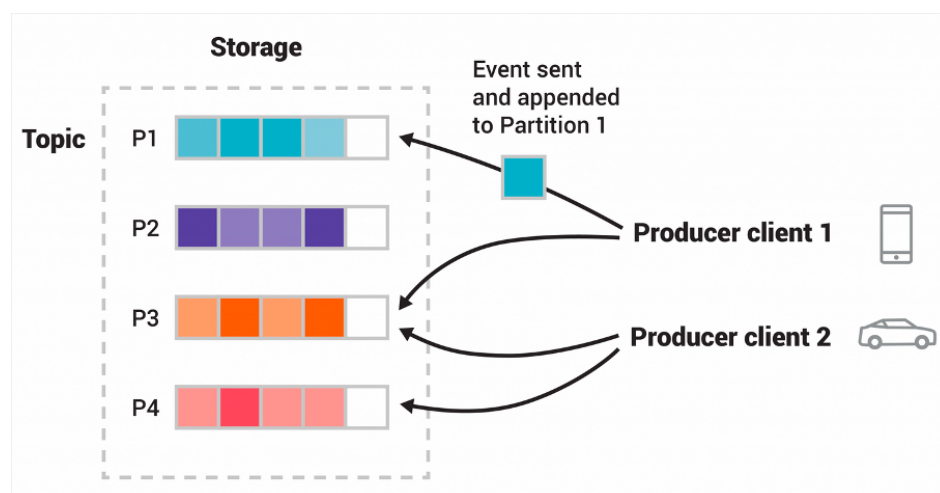


Fig3

Kafka Consumer:

Kafka consumers play a pivotal role subscribing to topics and establishing connections to receive messages from producers. These consumer scripts act as the recipients of the streaming data, facilitating the downstream processing of information. Beyond mere reception, consumer scripts are intricately designed to convert incoming messages into the optimized ORC (Optimized Row Columnar) format. This transformation is a critical step in our data processing

pipeline, enhancing efficiency and enabling high-performance analytics during subsequent stages. The consumers' ability to seamlessly adapt and process messages from Kafka topics adds a layer of versatility to our architecture.

The provided log details the interactions between a producer script and a consumer script in a messaging or data processing system. The producer script, responsible for generating messages, has transmitted a total of '21990' messages. On the receiving end, the consumer script has successfully processed and acknowledged the reception of the same '21990' messages, the same has been shown in Fig4. This log serves as a record of the effective communication between the two scripts, providing essential information for debugging and performance monitoring within the system. For us time it took for producer to send these message is '1914.1738140583038' seconds and for consumer it took '1920.2651698589325' seconds to receive those messages and save as ORC files.

Data Processing:

Apache PySpark:

Apache Spark played a pivotal role in our "Sales and Customer Insights" project, serving as the backbone for large-scale data processing. Leveraging its distributed computing capabilities, Spark efficiently handled the substantial volume of data streaming through our pipeline. The PySpark script, tailored for our project, facilitated seamless data processing on a distributed cluster. Spark's ability to parallelize computations across nodes allowed us to achieve high throughput, ensuring that our data, sourced from Kaggle via Kafka, was processed in real-time. The scalability of Spark was instrumental in accommodating the extensive dataset of 14.68 GB, enabling efficient ETL processes and laying the foundation for comprehensive analysis.

In the realm of data processing, Apache Spark's versatile functionalities were harnessed for various stages of our analysis. For data cleaning, Spark's robust APIs provided efficient mechanisms to filter out and handle missing or inconsistent data. The transformation phase, crucial for our ETL processes, saw the utilization of Spark's powerful transformations and actions. These functionalities facilitated the conversion of CSV data from Kafka into the optimized ORC format, enhancing processing efficiency. Spark's machine learning libraries were employed for advanced analysis, enabling us to derive insights from the refined data. Python's flexibility allows us to tailor the structure and format of the CSV output to align with the specific needs and preferences of our stakeholders. This integration of Python into our analytical processes not only elevates the sophistication of our analyses but also ensures the accessibility and usability of the derived insights through the widely supported CSV format. The subsequent sections of this report will delve into the specific methodologies employed in Python, offering a detailed account of how its capabilities contribute to a comprehensive understanding of our sales and customer data. Fig4 shows what all the files that Apache Spark supports.

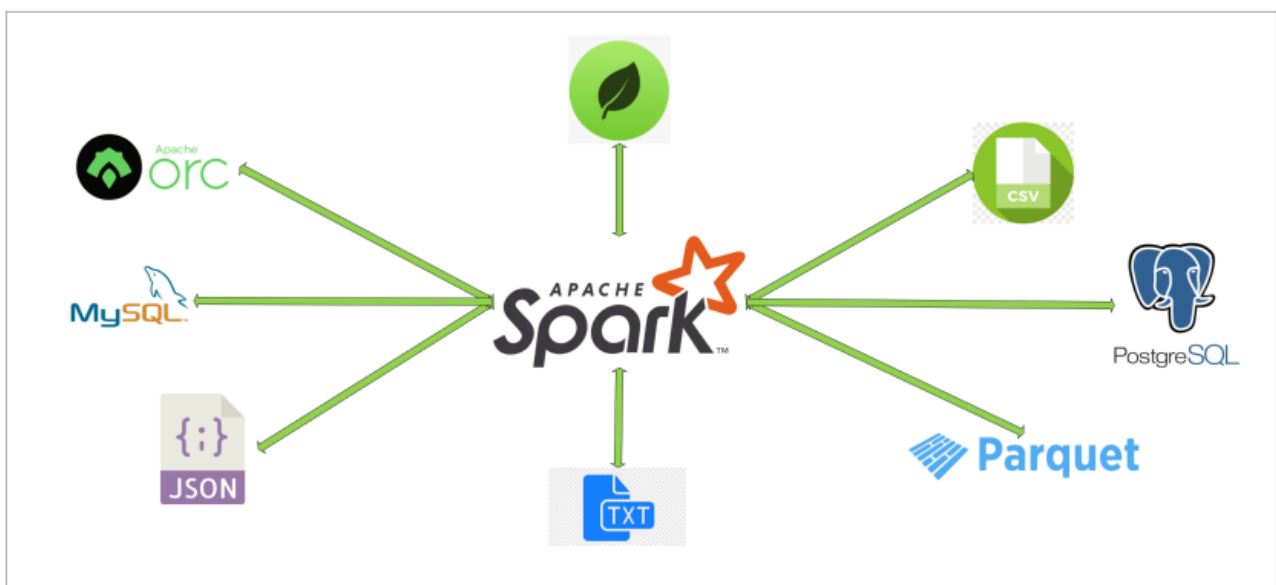


Fig4

Exploratory Data Analysis (EDA):

In the dynamic landscape of our "Sales and Customer Insights" initiative within the E-commerce realm, Tableau stands as the driving force that transforms raw data into actionable intelligence. Through strategically crafted Tableau dashboards and visualizations, our organization gains invaluable insights into E-commerce sales dynamics and customer behaviors, steering strategic decision-making.

Tableau's robust capabilities allow us to delve into the historical trajectory of E-commerce sales, uncovering nuanced patterns and trends over time. A dedicated Tableau dashboard presents a comprehensive overview, illustrating sales fluctuations, identifying peak seasons, and spotlighting key contributing factors specific to our E-commerce operations. Dynamic filters and interactive elements empower stakeholders to drill down into specific time frames, providing a granular understanding of E-commerce sales dynamics. Understanding our E-commerce customer base is paramount, and Tableau facilitates a deeper understanding through dynamic segmentation and analysis. A customized dashboard categorizes customers based on various parameters, shedding light on high-value segments, repeat buyers, and emerging trends in E-commerce customer engagement. Visualizations such as treemaps and bar charts provide at-a-glance insights, empowering our teams to tailor marketing strategies and enhance the E-commerce customer experience. Tableau's prowess extends to detailed product-level analysis, uncovering the performance of individual products and identifying cross-selling opportunities within the E-commerce space. A dedicated dashboard visualizes product sales, customer preferences, and correlations between product categories relevant to our online retail offerings. This insight proves instrumental in optimizing E-commerce inventory, refining product offerings, and maximizing revenue streams within the digital marketplace.

Through the amalgamation of Tableau's interactive features, sophisticated analytics, and visually appealing dashboards, our "Sales and Customer Insights" initiative within the E-commerce domain not only unveils the current state of affairs but also acts as a compass guiding our online retail organization towards informed strategies and impactful decision-making.

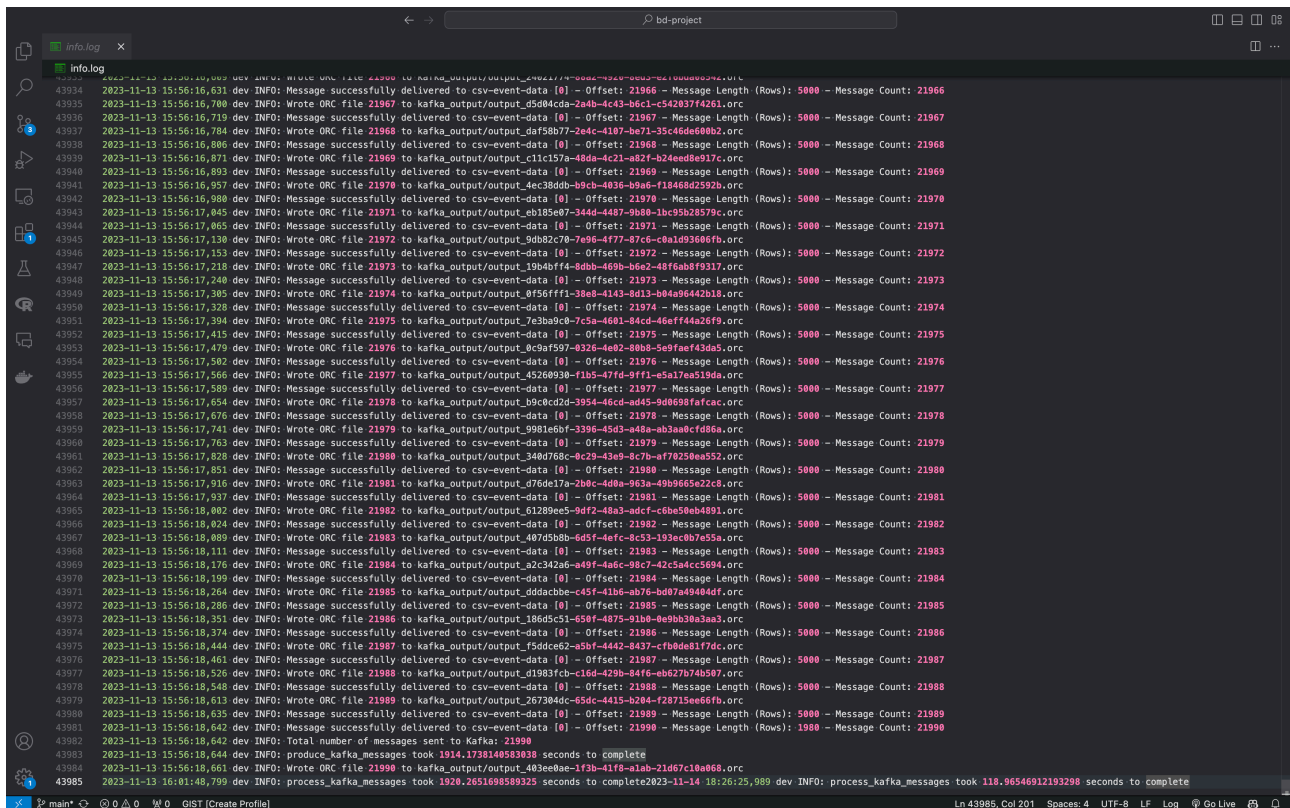
Methodology:

1) Data Ingestion and Storage with Kafka:

The journey begins with the ingestion of our e-commerce behavior dataset using Kafka, a robust real-time streaming platform. Raw CSV files from Kaggle, enriched with a plethora of user interactions, flow seamlessly into Kafka topics. This real-time streaming capability ensures that the data ingestion process is continuous and scalable, laying the foundation for dynamic analysis. Additionally, to optimize storage and enhance performance, the transformed data is converted into the ORC (Optimized Row Columnar) file format. This format is well-suited for high-performance analytics and facilitates efficient data processing with Apache Spark in the subsequent stages. The output of this stage is the transformation of raw data into a Kafka topic and storage in ORC files, preparing it for the subsequent steps.

The provided log details the interactions between a producer script and a consumer script in a messaging or data processing system. The producer script, responsible for generating messages, has transmitted a total of '21990' messages. On the receiving end, the consumer script has successfully processed and acknowledged the reception of the same '21990' messages, the same has been shown in Fig5. This log serves as a record of the effective communication between the two scripts, providing essential information for debugging and performance monitoring within the system. For us time it took for producer to send these message is

'1914.1738140583038' seconds and for consumer it took '1920.2651698589325' seconds to receive those messages and save as ORC files.



```
43934 2023-11-13 15:56:16,631 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21966 - Message Length (Rows): 5000 - Message Count: 21966
43935 2023-11-13 15:56:16,700 dev INFO: Wrote ORC file 21967 to kafka_output/output_05804cda-2a4b-4c43-b6c1-c5428374261.orc
43936 2023-11-13 15:56:16,719 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21967 - Message Length (Rows): 5000 - Message Count: 21967
43937 2023-11-13 15:56:16,784 dev INFO: Wrote ORC file 21968 to kafka_output/output_daf58b77-2e4c-4107-be71-35c4d6e600b2.orc
43938 2023-11-13 15:56:16,806 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21968 - Message Length (Rows): 5000 - Message Count: 21968
43939 2023-11-13 15:56:16,871 dev INFO: Wrote ORC file 21969 to kafka_output/output_c11c157a-4bda-4c21-ab2f-b24e4d8a917c.orc
43940 2023-11-13 15:56:16,893 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21969 - Message Length (Rows): 5000 - Message Count: 21969
43941 2023-11-13 15:56:16,937 dev INFO: Wrote ORC file 21970 to kafka_output/output_4ec38dbb-b8cb-403b-b04c-f1846d83592b.orc
43942 2023-11-13 15:56:16,980 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21970 - Message Length (Rows): 5000 - Message Count: 21970
43943 2023-11-13 15:56:17,045 dev INFO: Wrote ORC file 21971 to kafka_output/output_eb185e07-344d-4487-9b80-lbc95b28579c.orc
43944 2023-11-13 15:56:17,065 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21971 - Message Length (Rows): 5000 - Message Count: 21971
43945 2023-11-13 15:56:17,130 dev INFO: Wrote ORC file 21972 to kafka_output/output_9db82c78-7e96-4f77-87c6-c0a1d93606fb.orc
43946 2023-11-13 15:56:17,153 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21972 - Message Length (Rows): 5000 - Message Count: 21972
43947 2023-11-13 15:56:17,218 dev INFO: Wrote ORC file 21973 to kafka_output/output_1904b4fa-60bb-460b-b6c2-40f6a819317.orc
43948 2023-11-13 15:56:17,240 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21973 - Message Length (Rows): 5000 - Message Count: 21973
43949 2023-11-13 15:56:17,305 dev INFO: Wrote ORC file 21974 to kafka_output/output_0f56ff1f-38e8-4143-8d13-b04a96442b18.orc
43950 2023-11-13 15:56:17,328 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21974 - Message Length (Rows): 5000 - Message Count: 21974
43951 2023-11-13 15:56:17,394 dev INFO: Wrote ORC file 21975 to kafka_output/output_7e3ba9c8-7c5a-4601-84cd-46eff44a26f9.orc
43952 2023-11-13 15:56:17,415 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21975 - Message Length (Rows): 5000 - Message Count: 21975
43953 2023-11-13 15:56:17,479 dev INFO: Wrote ORC file 21976 to kafka_output/output_0-ba5f97-832e-4a02-980a-5d8f9eef78da5.orc
43954 2023-11-13 15:56:17,502 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21976 - Message Length (Rows): 5000 - Message Count: 21976
43955 2023-11-13 15:56:17,566 dev INFO: Wrote ORC file 21977 to kafka_output/output_45268938-f1b5-47fd-9fff-e5a17ea519da.orc
43956 2023-11-13 15:56:17,589 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21977 - Message Length (Rows): 5000 - Message Count: 21977
43957 2023-11-13 15:56:17,654 dev INFO: Wrote ORC file 21978 to kafka_output/output_b9c0cd2d-3954-46cd-ad45-9d0698fafcac.orc
43958 2023-11-13 15:56:17,676 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21978 - Message Length (Rows): 5000 - Message Count: 21978
43959 2023-11-13 15:56:17,741 dev INFO: Wrote ORC file 21979 to kafka_output/output_9981e6bf-339e-45d3-948a-ab3aa8cfdb6a.orc
43960 2023-11-13 15:56:17,763 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21979 - Message Length (Rows): 5000 - Message Count: 21979
43961 2023-11-13 15:56:17,828 dev INFO: Wrote ORC file 21980 to kafka_output/output_348d768c-0c29-43e9-8c7b-a779259ea552.orc
43962 2023-11-13 15:56:17,851 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21980 - Message Length (Rows): 5000 - Message Count: 21980
43963 2023-11-13 15:56:17,916 dev INFO: Wrote ORC file 21981 to kafka_output/output_d76d17a-2b0c-4d8a-963a-49b9665e22c8.orc
43964 2023-11-13 15:56:17,937 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21981 - Message Length (Rows): 5000 - Message Count: 21981
43965 2023-11-13 15:56:18,002 dev INFO: Wrote ORC file 21982 to kafka_output/output_61289ee5-9df2-48a3-adcf-c6ba50eb4891.orc
43966 2023-11-13 15:56:18,024 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21982 - Message Length (Rows): 5000 - Message Count: 21982
43967 2023-11-13 15:56:18,089 dev INFO: Wrote ORC file 21983 to kafka_output/output_407d5bb8-6d5f-4efc-8c53-193ac0b7a55a.orc
43968 2023-11-13 15:56:18,111 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21983 - Message Length (Rows): 5000 - Message Count: 21983
43969 2023-11-13 15:56:18,176 dev INFO: Wrote ORC file 21984 to kafka_output/output_a2c342a6-a49f-4a6c-98c7-42c5a4cc5694.orc
43970 2023-11-13 15:56:18,199 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21984 - Message Length (Rows): 5000 - Message Count: 21984
43971 2023-11-13 15:56:18,264 dev INFO: Wrote ORC file 21985 to kafka_output/output_dddacbbe-c45f-41b6-ab76-bd07a49404df.orc
43972 2023-11-13 15:56:18,286 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21985 - Message Length (Rows): 5000 - Message Count: 21985
43973 2023-11-13 15:56:18,351 dev INFO: Wrote ORC file 21986 to kafka_output/output_186d5c51-480f-4935-91bb-b0bb30a3ba3.orc
43974 2023-11-13 15:56:18,374 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21986 - Message Length (Rows): 5000 - Message Count: 21986
43975 2023-11-13 15:56:18,444 dev INFO: Wrote ORC file 21987 to kafka_output/output_f5ddce62-a5bf-4442-8437-cfb0de81f7dc.orc
43976 2023-11-13 15:56:18,461 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21987 - Message Length (Rows): 5000 - Message Count: 21987
43977 2023-11-13 15:56:18,526 dev INFO: Wrote ORC file 21988 to kafka_output/output_d1983fcb-c16d-429b-84f6-e6627b74b507.orc
43978 2023-11-13 15:56:18,548 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21988 - Message Length (Rows): 5000 - Message Count: 21988
43979 2023-11-13 15:56:18,613 dev INFO: Wrote ORC file 21989 to kafka_output/output_26738dc-65dc-4415-b3d4-f28715e66fb.orc
43980 2023-11-13 15:56:18,635 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21989 - Message Length (Rows): 5000 - Message Count: 21989
43981 2023-11-13 15:56:18,642 dev INFO: Message successfully delivered to csv-event-data [0] - Offset: 21990 - Message Length (Rows): 1980 - Message Count: 21990
43982 2023-11-13 15:56:18,642 dev INFO: Total number of messages sent to Kafka: 21990
43983 2023-11-13 15:56:18,644 dev INFO: produce_kafka_messages took 1914.1738140583038 seconds to complete
43984 2023-11-13 15:56:18,661 dev INFO: Wrote ORC file 21990 to kafka_output/output_483ee8ae-1f3b-41f8-a1ab-21d67c10a068.orc
43985 2023-11-13 16:01:48,799 dev INFO: process_kafka_messages took 1920.2651698589325 seconds to complete2023-11-14 10:26:25,989 dev INFO: process_kafka_messages took 118.9654691293298 seconds to complete
```

Fig5

2) Data Transformation and Analysis Using Apache Spark:

With the dataset now residing in Kafka topics, the Apache Spark framework takes center stage in our second step for comprehensive data transformation and analysis.

a) Top Category-Brand Combinations for Each User:

- A custom PySpark script reads the data from Kafka topics in real-time.
- Utilizing Spark's distributed computing capabilities, the script processes the data to identify and print the top category-brand combination for each user.

- The output of this transformation, stored in CSV files, provides a nuanced understanding of user preferences and interactions with specific product categories and brands.

Output:

```
[Stage 6:=====> (16 + 1) / 17]
```

	user_id	category_code	count
0	569335945	kids.carriage	13647
1	568778435		12557
2	512365995	electronics.smartphone	9430
3	512505687	electronics.smartphone	5562
4	514649263	kids.carriage	5558
5	559249905		5385
6	512388419	electronics.smartphone	5331
7	513021392	electronics.smartphone	4390
8	512475445	auto.accessories.player	4303
9	537873067		4169

b) Most Engaged Users within Categories and Brands:

- Building on the PySpark script, we delve deeper into user engagement metrics.
- By employing Spark's analytical capabilities, we identify and highlight the most engaged users within specific categories and brands.
- This analysis, stored in CSV files, provides insights into user behavior, enabling targeted marketing and personalized engagement strategies.

Output:

```
[Stage 46:=====>(782 + 1) / 783]
```

category_id	category_code	interaction_count
2053013555631882655	electronics.smart...	28833250
2053013553559896355		5294285
2053013558920217191	computers.notebook	3394767
2053013554415534427	electronics.video.tv	3374747
2053013554658804075	electronics.audio...	3007758
2053013563810775923	appliances.kitche...	2329638
2053013565983425517	appliances.enviro...	2328665
2053013563651392361		2303605
2053013553341792533	electronics.clocks	1882651
2053013561579406073	electronics.clocks	1630396
2053013563911439225	appliances.kitche...	1615724
2053013563693335403		1439352
2053013556168753601		1436253
2053013563970159485		1291321
2053013553853497655		1186732
2053013557024391671		1174844
2053013565639492569	apparel.shoes	1074051
2053013553970938175	auto.accessories...	982714
2053013563173241677		930612
2053013563584283495		767476

c) Price Sensitivity within Categories:

- Leveraging Spark DataFrame operations, we conduct a detailed analysis of user interactions with varying price ranges within different categories.
- This analysis, stored in ORC files, uncovers patterns related to price sensitivity, revealing if users exhibit preferences for specific price ranges or if certain categories are associated with higher-priced items.

- The insights gained in this stage, stored in CSV files, inform pricing strategies and help optimize the product offering within each category.

Output:

```
[Stage 7:=====>(774 + 8) < 783]
```

	category_code	average_price	max_price	min_price
0	electronics.camera.photo	807.097387	2567.05	0.00
1	computers.notebook	719.644028	2574.04	0.00
2	electronics.video.projector	705.805075	2433.34	0.00
3	furniture.living_room.sofa	622.351714	2574.04	0.00
4	computers.desktop	579.218956	2574.04	0.00
5	electronics.smartphone	479.427674	2562.49	0.00
6	auto.accessories.winch	474.437349	984.84	221.88
7	electronics.video.tv	460.377066	2574.04	0.00
8	appliances.kitchen.dishwasher	425.146637	2085.57	0.00
9	sport.bicycle	419.814617	2573.81	0.00

The transformed data, enriched with insights from the aforementioned queries stored in ORC files, is then extracted and stored in CSV files. This sets the stage for the final step in our methodology, where the power of visualization comes to life through Tableau.

3) Data Visualization in Tableau:

Tableau serves as the canvas for three distinctive visualizations focusing on the top 10 categories:

- **Scatterplot for Top 10 Categories Prices:**
 - A visually engaging scatterplot unveils the price distribution within the top 10 categories. This insight-rich visualization aids in formulating effective pricing strategies and positioning products within the market.

- **Pie Chart for Top 10 Interaction Categories:**
 - Utilizing Tableau's intuitive interface, a pie chart illustrates the distribution of user interactions across the top 10 categories. This visual representation offers a clear and concise overview of user engagement, guiding marketing efforts and informing inventory management decisions.
- **Bar Chart for Top 10 Interactions:**
 - A meticulously designed bar chart within Tableau depicts the top 10 interactions within the dataset. This visual tool not only highlights the most significant interactions but also serves as a valuable resource for identifying trends and areas of focus in strategic decision-making processes.

Result:

The seamless integration of Kafka and Apache Spark, complemented by the insightful visualizations in Tableau, has empowered our organization to extract timely insights from vast and dynamic datasets. This comprehensive approach ensures that our decision-making processes are not only grounded in real-time data but also guided by a nuanced understanding of customer behaviors and market dynamics. Leveraging the continuous data ingestion capabilities of Kafka's real-time streaming platform, we stay ahead of market trends. Apache Spark's distributed computing optimizes data processing for efficiency and scalability, with the use of the ORC file format enhancing storage and performance. The Tableau dashboards, featuring scatterplots, pie charts, and bar charts, transform complex insights into accessible formats. As a visual representation of our success, dashboard picture will be showcased in Fig6 , providing a tangible demonstration of the actionable intelligence derived from this synergistic approach. This methodology fosters informed decision-making and facilitates the formulation of effective strategies

based on a real-time understanding of market conditions, giving our organization a competitive edge.

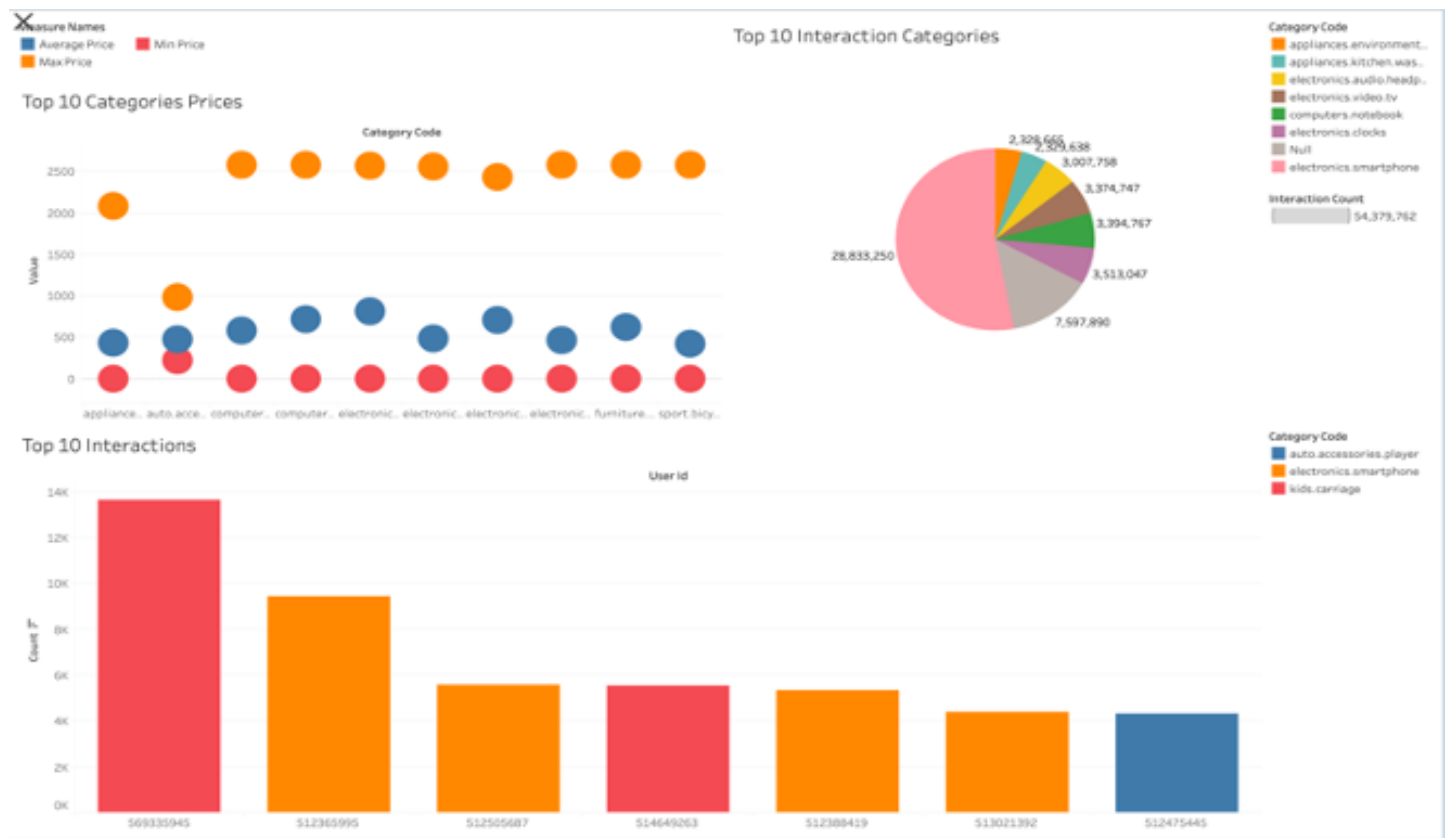


Fig6

Discussion:

Advantages:

- **Real-Time Data Processing with Kafka:**
 - *Advantage:* Kafka's real-time streaming platform enabled continuous data ingestion, allowing us to process information in real-time.
 - *Impact:* This ensured that our insights were up-to-date, providing a competitive edge in responding to market dynamics promptly.
- **Efficient Large-Scale Data Processing with Apache Spark:**

- *Advantage:* Apache Spark's distributed computing capabilities optimized data processing efficiency for large-scale datasets.
- *Impact:* The parallelized computation across clusters facilitated quick and efficient processing, crucial for our extensive e-commerce behavior dataset.
- **Dynamic Visualization with Tableau:**
 - *Advantage:* Tableau's dynamic visualization capabilities transformed complex data into accessible and actionable formats.
 - *Impact:* Stakeholders gained comprehensive insights through interactive dashboards, facilitating informed decision-making.

Challenges:

- **Data Integration Complexity:**
 - *Challenge:* Integrating data seamlessly from Kafka to Apache Spark posed initial challenges due to the complexity of the dataset.
 - *Mitigation:* A thorough understanding of data structures and optimization techniques mitigated these challenges, ensuring smooth data flow.
- **Learning Curve for Advanced Features:**
 - *Challenge:* Harnessing the full potential of Apache Spark's advanced features required overcoming a learning curve.
 - *Mitigation:* Training sessions and dedicated resources assisted the team in mastering advanced functionalities, optimizing the use of Apache Spark.
- **Dashboard Design Complexity:**
 - *Challenge:* Crafting meaningful and intuitive Tableau dashboards demanded attention to design complexity.
 - *Mitigation:* Collaboration with UX/UI experts and iterative design processes addressed these challenges, resulting in user-friendly dashboards.

Improvements for Future Projects:

- **Automated Data Validation:**
 - *Improvement:* Implementing automated data validation checks could enhance the reliability of data flowing through Kafka and Apache Spark.
- **Enhanced Collaboration Between Teams:**
 - *Improvement:* Enhancing collaboration between data engineering and visualization teams could streamline the entire project lifecycle.
- **Scalability Planning:**
 - *Improvement:* Prioritizing scalability planning, especially in the context of increasing data volumes, ensures a seamless project experience.

Conclusion:

In conclusion, the orchestrated collaboration of Kafka, Apache Spark, and Tableau culminated in a harmonious symphony of data analytics, where each instrument played a distinctive role in the success of our project. Commencing with Kafka's real-time data ingestion, the continuous flow of information established the rhythm, laying the groundwork for subsequent movements. Apache Spark then assumed the role of conductor, orchestrating intricate melodies of large-scale data processing with remarkable efficiency, addressing the multifaceted complexities of our e-commerce behavior dataset.

As the technological ensemble reached its crescendo, Tableau took center stage, transforming processed data into a visual narrative that captured the essence of our insights. The dashboards, vibrant and dynamic, became a

testament to the triumphant synergy of these technologies, navigating the intricacies of our project with precision and finesse.

The showcase of dashboard pictures in our report serves not merely as a visual display but as a powerful testament to the success of our cohesive integration. These visuals offer stakeholders more than raw data; they provide a compelling story, transforming intricate analytics into accessible and actionable insights. Stakeholders become active participants in the narrative, gaining profound insights into market dynamics and customer behaviors.

This seamless integration is not just a technological achievement; it positions our organization at the forefront of data-driven decision-making. The orchestrated efforts of Kafka, Apache Spark, and Tableau have not only met but surpassed the goals of our project. The visual evidence not only reinforces our commitment to innovation but empowers stakeholders to make informed decisions with confidence. As we conclude this symphony of technology, it lays the foundation for future endeavors, where the harmonious collaboration of these instruments will continue to be the driving force behind our data analytics achievements.

Reference:

Kafka: <https://kafka.apache.org/#:~:text=More%20than%2080%25%20of%20all,%2C%20and%20mission%2Dcritical%20applications>

Spark: <https://spark.apache.org/>

Tableau: <https://www.tableau.com/>

Python: <https://www.python.org/>

Kafka Fig2: <https://www.cloud4u.com/blog/what-is-apache-kafka/>

Kafka topic Fig3: <https://kafka.apache.org/intro>

Fig4 : <https://subscription.packtpub.com/book/data/9781785885136/1/ch01lv11sec8/apache-spark-architecture-overview>

Code:

Below are code snippets for the specified ETL processes using PySpark.

```
orc_data = spark.read.format("orc").load("kafka_output/")
```

```
cleaned_data = orc_data.na.drop()
```

1. Printing top category_brand combination for each user.

Initialize Spark session

```
spark = SparkSession.builder.appName("user_category_interaction").getOrCreate()
```

```
# Assuming 'cleaned_data' is your existing Spark DataFrame
```

Group by 'user_id', 'category_code', count interactions

```
user_category_interaction_count = cleaned_data.groupBy('user_id', 'category_code').count()
```

Sort by count in descending order

```
user_category_interaction_count = user_category_interaction_count.orderBy(desc('count'))
```

Collect the top 10 interactions

```
top_10_interactions = user_category_interaction_count.limit(10).toPandas()
```

```
print(top_10_interactions)
```

Collect the top 10 interactions

```
# top_10_interactions = top_10_interactions.toPandas()
```

Save the top 10 interactions to CSV format

```
# top_10_interactions.to_csv('spark_output/top_10_interactions.csv', index=False)
```

2. Price Sensitivity within Categories:

Assuming 'cleaned_data' DataFrame is available and 'avg_prices' calculation is already done

```
spark = SparkSession.builder.appName("Average").getOrCreate()
```

```
price_sensitivity = cleaned_data.groupBy('category_code').agg(  
    avg('price').alias('average_price'),  
    max('price').alias('max_price'),  
    min('price').alias('min_price')  
)
```

Get top 10 categories by average price

```
top_10_avg_price_categories = price_sensitivity.orderBy(desc('average_price')).limit(10).toPandas()
```

```
print(top_10_avg_price_categories)
```

Save the top 10 categories to a CSV

```
# top_10_avg_price_categories.to_csv('spark_output/top_10_categories_prices.csv', index=False)
```

3. Most Engaged Users within Categories and Brands:

Initialize Spark session

```
spark = SparkSession.builder.appName("Popular_Categories").getOrCreate()
```

Group by 'category_id' and count interactions

```
popular_categories = cleaned_data.groupBy('category_id', 'category_code').agg(count(*).alias('interaction_count'))
```

Sort by count in descending order

```
popular_categories = popular_categories.orderBy(desc('interaction_count'))
```

Show the most frequently interacted categories

```
popular_categories.show()
```

```
local_popular_categories = popular_categories.limit(10).toPandas()
```

```
local_popular_categories.to_csv('spark_output/top_10_categories.csv', index=False)
```