Online Customer Feedback Analysis for E-Commerce

Pratik Pramod Revankar **Siddharth Lanka**

College of Engineering
University of Colorado at Boulder
Email: {Pratik.Revankar, Sai.Lanka}@colorado.edu

Abstract

Big Data Computing has been one of the emerging areas of research for the last decade. Ever since the emergence of big data processing tools, researchers have been trying to extend this work to target a few specific use cases of online stream processing. There have been considerable efforts to harness the benefit of stream processing; scalability, reliability and low-cost ownership - such as Spark, Flink which extend the idea of stream processing and provide other features such as time based windowing, stateful stream processing respectively. In this paper, we aim to build an online customer feedback analysis service that looks at customer feedback on an e-commerce store to derive useful metrics about products and make relevant business decisions utilizing the power of stream processing and seamless resource sharing channels.

Keywords: Big Data Architecture, Stream Processing, Message Queues, Marketing Analytics, Business Intelligence

1 Introduction

The Internet has brought a radical change in the way consumers procure goods and services, and how companies decide advertise and sell their products. Digital engagement with consumers has boosted the growth of the global economy, and the leading E-Commerce giants are serving as the digital marketplaces for trade and commerce, where brands can sell their products and strategize their marketing decisions to effectively reach and cater a wider population.

Marketing Analytics refers to the collection, management, and analysis of data to extract useful insights to support marketing decision-making. Appropriate data, analytical skill and support from top management team are necessary for effective deployment of marketing analytics (1).

Customer Satisfaction is a term frequently used in marketing, and is measure of how products and services provided by a particular brand or company meet or surpass customer expectations. There is sufficient evidence to suggest that customer satisfaction has an impact on the brands success in the market. Empirical studies and market surveys have shown that customer satisfaction, customer loyalty and profitability (2) are all inter-related. Mathematical frameworks, like the one proposed by Zahorik and Rust (3), also provide a way to determine which elements of customer satisfaction have the greatest impact on corporate performance, and to determine the financial value of various managerial actions that focus on improving aspects of service.

Big Data provides a new era of data exploration and utilization, and using Big Data Analytics, which is the process of collecting, organizing, and analyzing large sets of data to identify trends and patterns, we aim to provide and integratable framework for utilising customer satisfaction measure to perform marketing analytics.

Most companies leverage business intelligence tools to visualize streams of data ingested from various sources and make efficient marketing decisions. We propose a business intelligence tool that analyses online customer feedback to measure customer satisfaction. This will facilitate a data-driven approach to making business decisions and allow senior management at companies and brands to make effective, customised marketing strategies in real-time.

2 Related Work

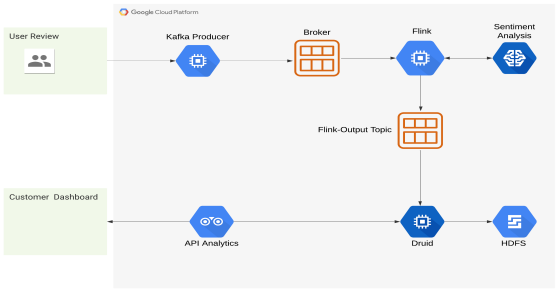
Marketing research has been a growing field, where new methods and tools have been proposed and developed to make smarter business decisions. One such proposed idea by a team of academics (Macdonald, E., Wilson, H.N. and Konus, U.) (4), is a research tool called real-time experience tracking (RET), which would allow to capture customer experiences almost immediately, before any memory loss or change in bias, unlike focus groups and surveys, which tap consumer's memory which could be unreliable, and the presence of observers may cause some changes in consumer behavior.

Another study examined whether a firm can attain sustained competitive advantage from its sensing, seizing and re-configuring capabilities, which are manifested by the use of marketing analytics, marketing decision- making, and product development management (5). The study involved conducting a survey with 221 UK firm managers, and their research model proposed in this study provided insights into how marketing analytics can be used to achieve sustained competitive advantage.

Customer feedback data contains rich information and new insights, as compared to traditional methods of data collection like surveys or interviews. A framework was proposed, based on artificial intelligence and machine learning techniques, to find the applicability of some key operations like opinion mining from customer feedback reviews, which could be useful for product service systems (6).

3 System Design

3.1 Architecture



3.2 API gateway

A REST endpoint is exposed to send the raw events which are further queued in Kafka for processing. This endpoint takes as input a customer review and various other event parameters, transforms this input into an event object based on some structure to be queued in Kafka. The other endpoint is used to query the processed events from the Time series database which has data about product metrics. Both these endpoints should be placed in nodes with a public IP as clients should be able to query these endpoints from the internet.

3.3 Kafka

Our usage of Kafka is to store the events generated in our system in a fault-tolerant way. The workers pick events off the queue(Source), process them and store these processed events back in a Kafka queue(Sink). We would use Kafka extensively for logging data and to get a general idea of the state of the system. These logs do not have to be maintained for long since they are only used for debugging purposes and Kafka gives us this flexibility of adding a retention period to the messages.

3.4 Worker

The worker node is responsible for reading messages off the queue in a pull based model. These workers should essentially scale based on the load in the queue. We plan to implement the worker logic using Flink to account for different type of events that could propagate throughout the system. While reading customer reviews could be a great start, collecting metrics about the product views and the amount of time spent on the page could also give us a lot of information about the product. Each of the event type will have it's own worker logic to handle and process these events. The worker is also responsible to query the input review against the trained sentiment analysis machine learning model stored in cloud storage.

4 Implementation

We have setup a multi-broker Kafka deployment following official guides from Confluent. This is mostly to make sure our events are replicated across brokers and partitions. Each broker is on it's own host in a virtual machine utilizing Compute Engine service by Google Cloud Platform(GCP). A multi-broker Kafka setup is needed along with a standalone zookeeper cluster to make sure events are not lost at any point since they are an integral part of our system. Zookeeper is used by Kafka to handle leader elections and to maintain cluster metadata.

The dataset used was the Amazon Customer Reviews, which initially had no labels for sentiment classification to train our model. The data only had a column for *star rating* which didn't have enough confidence to assign a sentiment label. We used a library called TextBlob to measure the polarity of the input text and get a score. Using these two features (star rating, polarity), we create the final labels for classification, using a K-Means clustering model. We then created a simple Bag-of-Words (BoW) representation for the customer review to use as input to train our machine learning model using the newly generated labels. We experimented with different models like Naive Bayes, SVM, Logistic Regression network to train on the augmented dataset and classify any new test review. Logistic regression had the best accuracy of 85.6 and was used as the classifier for sentiment analysis.

Our dataset is currently hosted on Cloud Storage and the sentiment analysis model is trained every day and a pickled model file is stored in Cloud Storage. We use the pickle module in Python to dump model related data into a pickle file and store in cloud storage. For any incoming review, we make use of the trained model to make predictions regarding the sentiment.

Once the events are gathered in Kafka, our Flink worker logic is responsible for handling state when dealing with user sessions and product viewed and added to cart events. To keep this logic simple we terminate user sessions every 30 minutes of idle time and a new session would be generated post that. The data source for our Flink worker is the source Kafka queue where all events with a predefined structure are sent to and the Flink worker runs appropriate code to handle the event types. If the event is a product review our ML model has to be queried to find the sentiment score of the given review. These raw events are finally modified into processed events with added metadata and queued back into a sink where Druid will stream these events into the database. We use Druid as a time series database and all events are aggregated and indexed based on a timestamp that every event occurring in our system will have. Timestamp is defined as the time at which the event is generated.

The data pipeline is equipped with a logging and monitoring service at the application level to gather the state of the system at any point. These logs can be streamed to a centralized service. This monitoring would come in handy when our machine learning model is trained each day by the Airflow DAG.

Finally we have a query layer on top of druid that takes as input multiple parameters to provide time based querying of product details. An example of a query could be: "Number of product views in the past 30 minutes". Druid has an aggregation layer built into the product that helps achieve such queries in real-time with sub minimal latencies. The next section talks about how we will visualize these events coming from Druid to give key stakeholders a view of the data in the system.

5 Visualization

For this project we, aim to have the following visualizations as part of a simple dashboard, which would be used by key stake holders like product managers,

product owners, etc.

5.1 Sentiment Score

This visualization gives the sentiment score for a particular product in a catalog based on latest review. This will be a simple number metric. Sentiment score will also be given for different product categories on the e-commerce platform.

5.2 Time-series progression of Product Sentiment

For any given product, we can visualize how the product has changed over time. Few metrics we could visualize in this manner is product viewed, product added to cart events and how the sentiment of the product is changing over time.

6 Evaluation

Our big data solution to identifying customer reviews on e-commerce platforms will be evaluated based on the following metrics.

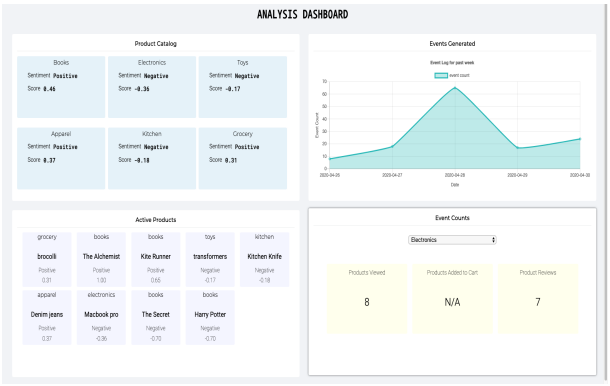


Figure 1: Analytics Dashboard

In figure 1 above, we present our analytics dashboard where the data is queried from our druid node. There are 4 cards in our UI and the following figures will mention them in detail.

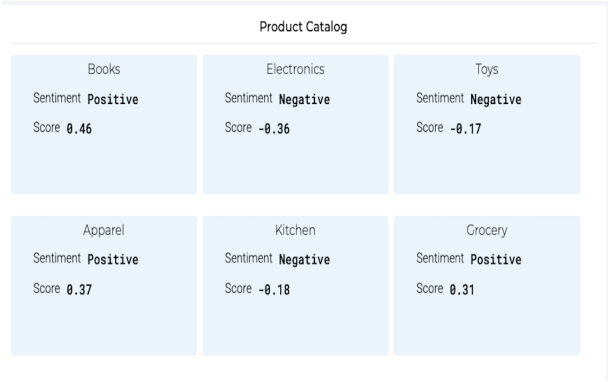


Figure 2: Product Catalog Card

In figure 2, this card maintains the average sentiment of all the products from a particular catalog. Average sentiment for product catalog is defined as sum of sentiment score of all reviews received divided by the total number of reviews for that particular catalog.

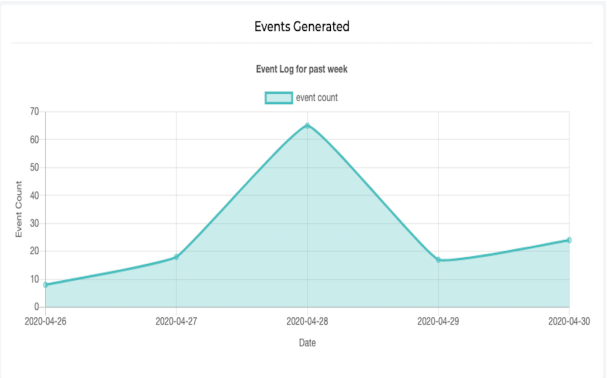


Figure 3: Events Generated Card

In figure 3, the card is a line chart of the total number of events generated in our system for the past week. The events tracked in our system are product_viewed, product_added_to_cart and product_review events. The x-axis represents the date and the y-axis represents the count of events.

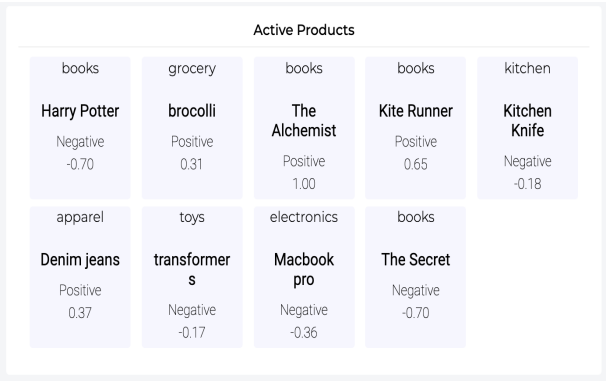


Figure 4: Active Products Card

Figure 4 represents the active products card where the data is presented for products that have maximum events generated in our system. A maximum of 10 products are returned and they are sorted by the total number of events generated for the product. The sentiment and the sentiment score represent what the customers feel about the product. The sentiment score is averaged for all reviews received for this product.

Figure 5 represents the event count card which has values for the number of events received for each type of event from product viewed, review and added to cart. The values represent the event count for each category.

The data generated by all the figures above will be extremely beneficial for the key decision maker in

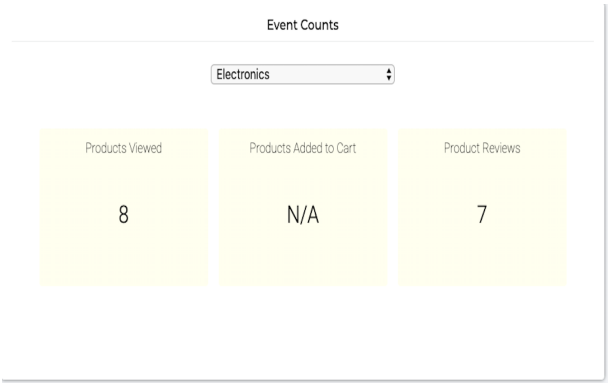


Figure 5: Event Count Card

an organisation to make data driven decisions based on the reviews, views a product has received. For instance, if a product has a negative sentiment score a decision could be made about how many items of the product are to be kept in the inventory. Since the reviews of this product are bad, the owner will try and clear sales of this product.

6.1 Stress Test

We stress tested our entire infrastructure by pushing 10,000 events to our kafka topic that we refer to as test. This was done sequentially using a for loop and the kafka message log in figures 6 and 7 represent the time at which the first and the last message was sent to the kafka topic respectively. It took about 7 minutes for the producer to send these asynchronous events to kafka.

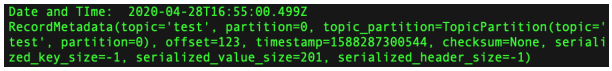


Figure 6: Timestamp of 1st kafka message

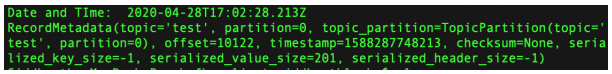


Figure 7: Timestamp of 10,000 message

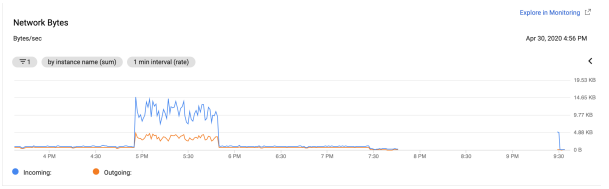


Figure 8: Druid Network Traffic

Our single node deployment of druid received lot of traffic from 4:50pm to 5:50pm. Figure 8 shows the network traffic graph from Google cloud platform. 4:50pm was when the first message was sent to kafka. Druid's indexer for kafka indexed these messages in about an hour. Druid can be configured for different workloads. This was just an experiment to test if our system could handle the load. In general, if there is a write heavy workload more number of real-time nodes have to be configured. The real-time nodes of druid are responsible for indexing from the sink provided. In our case since the sink is kafka, druid's kafka indexer indexes these processed events generated by our flink job and stores them in the druid database.

Our 10,000 events were generated for a book called harry potter. The event structure for this review is shown in the json below.

```
{
    "timestamp": now.strftime("%Y-%m-28T%H:%M:%S.%f")[:-3] + "Z",
    "event": "product_review",
    "product_id": "123475",
    "product_name": "Harry Potter",
    "product_review": "Bad book, I did not like it",
    "product_catalog": "books"
}
```

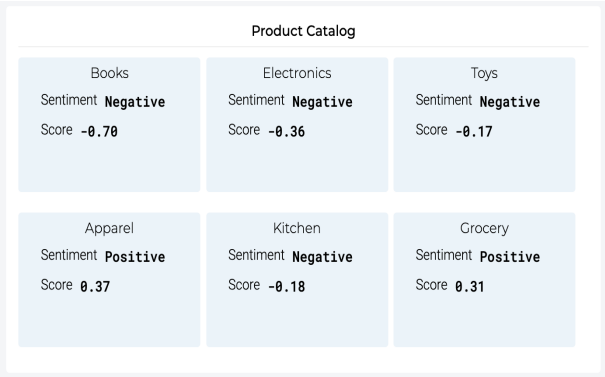


Figure 9: Product catalog after Stress test

Notice the drop in sentiment score for the product catalog by comparing figure 2 and Figure 9. The sentiment for product catalog books went from 0.46 to -0.70 after the stress test.

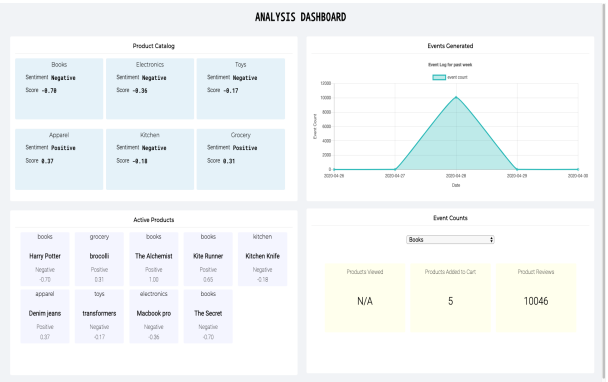


Figure 10: Dashboard after Stress test

The two figures 10 and 11 represent our UI and the events generated card after our stress start. Note that the event generated tooltip in figure 11.

6.2 Sentiment Analysis Model

The BoW method for the customer review sentiment classification had a baseline accuracy of 56%. Further evaluating with other machine learning models, SVM had an accuracy of 78% and Logistic regression had an accuracy of 85.6%. The Logistic Regression model

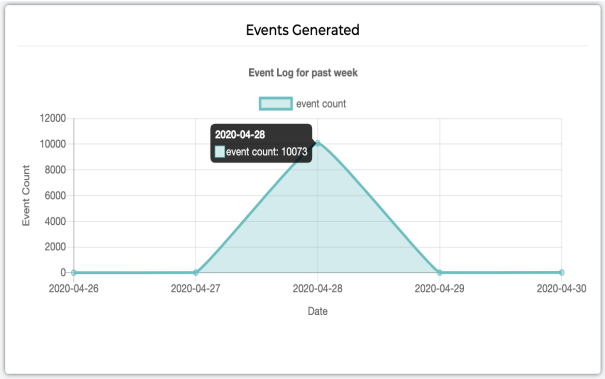


Figure 11: Product catalog after Stress test

was used as a sentiment classifier (Positive/Negative/Neutral) and TextBlob library was used to measure the sentiment polarity [-1,1], for an incoming customer review (text).

6.3 Scalability

A big data solution to a problem must be scalable and the idea of online processing helps build this idea further to build a solution at internet scale. Our API layer scales based on the number of requests it receives and is attached to an Auto Scaling Group in AWS which handles this for us. At the same time, as messages are being queued we want our workers to process these at a fair pace to avoid back-pressure. This can be done by running multiple worker nodes and since this is a workload which does not necessarily have to maintain state, it's easy to spawn new workers without any data inconsistencies.

6.4 Reliability

The architecture should be reliable in the sense that we should clearly identify each individual functional unit of the system and address how to provide redundancy and a failure-recovery strategy. The infrastructure in itself would be deployed on multiple availability zones to account for reliability on the deployment. Events are valuable in our system as they contribute to business decisions which means we should never miss an event thereby maintaining at least once semantics. Kafka addresses this in the form of partitions to provide redundancy in a multi-broker Kafka setup.

6.5 Latency

Our architecture involves multiple micro-services each entrusted with a particular job, we have to assess what sort of latency these services bring to the system. We also have to identify the throughput and assess the time taken from generating an event to actually providing valuable feedback through this event. The latencies measured are listed in the table below.

Service Name	Average Latency(ms)
kafka	37.47
zookeeper	43.723
druid	49.834

6.6 Security

We want to be data compliant in some sense and give the user the ability to know how his data is being used by the system. Although we do not collect the name of the user who posts the comment, we still want to give the user the flexibility that his data won't be used for any background processing.

7 Conclusion and Future Work

We present a Big Data system for processing events generated on e-commerce websites using kafka, flink and druid. Kafka makes it seamless to deal with events at web scale because of its asynchronous nature. Our components work well with each other as flink has scala apis to read and process events from kafka queue as both source and sink. Druid is especially useful as an analytics engine where you want a real-time view of the data in the system. As shown in our stress test, the entire infrastructure is scalable because of the asynchronous nature of the 3 main components of this Big Data System.

There are number of directions that we would like to pursue in future. Potential bottlenecks at the api layer on top of our sentiment analysis model can be addressed in future. This api can be placed behind a load balancer for scaling out. Currently, our sentiment analysis model is trained on static data and we don't have a mechanism to re-train our model based on events that are being generated in real-time. We could make use of Apache Airflow to create a directed acyclic workflow to periodically train our machine learning model based on a set time window. Our flink job right now does not handle user sessions and there could be a whole application around this to actually track users who viewed a product and added to cart in that particular session. We could add additional features to our dashboard and whole set of user segmentation and clustering logic once we start collecting user data. This could help in making data driven decisions when we cluster the users based on products viewed and liked. From a marketing point of view we could personalize their feed on the website to show similar products of their interest.

References

[1] Germann, F., Lilien, G.L. and Rangaswamy, A. *Performance implications of deploying marketing analytics.*, International Journal of Research in Marketing, 30(2), pp.114-128., 2013.

[2] Hallowell, R. *The relationships of customer satisfaction, customer loyalty, and profitability: an empirical study.*, International journal of service industry management, 1996.

[3] Zahorik, A.J. and Rust, R.T. *Customer satisfaction, customer retention, and market share.*, Journal of retailing, 69(2), pp.193-215., 1993.

[4] Emma K. Macdonald, Hugh N. Wilson, Umut Konus *Better customer insight-in real time*, Vol. 90. Harvard Business School Publishing, 2012.

[5] Cao, G., Duan, Y. and El Banna, A. *A dynamic capability view of marketing analytics: Evidence from UK firms.*, Industrial Marketing Management, 76, pp.72-83., 2019.

[6] Chen, Diandi, Zhang, Dawen, Tao, Fei and Liu, Ang. *Analysis of Customer Reviews for Product Service System Design based on Cloud Computing.*, Procedia CIRP.83.522-527.10.1016/j.procir.2019.03.116., 2019

[7] Xu, K., Liao, S.S., Li, J. and Song, Y. *Mining comparative opinions from customer reviews for competitive intelligence.*, Decision support systems, 50(4), pp.743-754. Vancouver., 2011