# Real-Time Monitoring and Analytics of Arbitrage Opportunities in Cryptocurrency Markets

SP24: Big Data Technologies and Applications

Riddhi Kamleshkumar Vyas Raveena Kagne Priya Govindarajulu Soujanya Sankathala Tanvi Pagrut

***Abstract*—In the digital world of cryptocurrency, prices change between exchanges. There's a chance to make a profit through arbitrage. The project deals with streaming and studying real-time cryptocurrency market data to find potential arbitrage chances. As a result of this project, a dashboard is made that keeps looking at different cryptocurrency platforms like Bitcoin to find arbitrage chances. Apache Kafka and Apache Spark are used for stream processing. Special algorithms like Bloom filter are used to make sure analysis is done based on unique information and differential privacy techniques are used to preserve user's privacy. K-means clustering is used to find similar price points in different sets of data. This helps to find chances where one can buy crypto at a low price from one market and sell it for a higher price in another market. Locality-sensitive hashing is used for studying patterns among data. The results show this project's ability to quickly process and study data, providing timely insights into arbitrage chances, which are crucial for fast trading platforms. The findings also show the effectiveness of privacy techniques like differential privacy and Bloom filters in safeguarding transaction data. The main goal is to provide users with real-time insights into potential arbitrage chances, helping them to make informed decisions without relying on automated trading.**

***Index Terms*—cryptocurrency, Kafka, spark, Bloom filters, K-means**

## I. Introduction

### A. Motivation

A number of tools for handling high volumes of big data in finance, especially amid the new wave of cryptocurrencies, have been brought to the market in recent years to address these issues in data processing and analysis. The appeal of taking advantage of arbitrage opportunities across cryptocurrency marketplaces is very attractive to the retail investors and crypto traders – a possibility that can be exploited by monitoring the market dynamics and employing the appropriate analysis. The objective of this task is to tackle the issues arising from the size and speed of data which are produced in the digital cryptocurrency markets. Traditional data analysis tools were built for monitoring small data with large amounts of time between events. In this case, each millisecond can make the difference between profit and loss. To tackle this, this project is powered by Apache Kafka for high-throughput data ingestion as well as by Apache for stream processing in order to provide the required infrastructure to manage big data dimensions by its velocity and volume. Moreover, regard is to be given to the fact that the transaction support is among the key factors that ensures both the authenticity and security of transactions. The use of Bloom filters along with differential privacy enabled the team to maintain the security of users' trading information while allowing us to use the data in the process of finding arbitrage opportunities. It is technology versatile, but it is also necessary to maintain ethical standards and comply with strict industry regulations. The practical applicability of this project was another big motivation. The solution which it provides real time analysis of arbitrage opportunities. It focuses on reducing the latency in data processing, so that instead of having these insights delayed across the timeline, analysts will be able to deliver time-relevant insights. Lastly, the project wanted to harness the full potential of big data technologies, turning complex data flow into actionable insights/recommendations.

### B. Technology

Kafka for Data Streaming: Data is streamed in real-time from several exchanges, including Bitstamp, Bitfinex, and Kraken, using Kafka. This setup ensures the continuous and reliable delivery of trade data necessary for identifying arbitrage opportunities. Apache Spark for Data Processing and Analysis: Spark is used to process and analyze the incoming data streams. Bloom Filter: The Bloom filter is employed to efficiently check whether an element has been previously encountered in the stream. It helps to ensure that only unique transactions are processed and stored, thereby avoiding duplicate data entries and reducing unnecessary processing. Differential Privacy Algorithm: This technique adds some noise in user data to preserve privacy. It ensures that even if some data about individual transactions is leaked or exposed, it cannot be traced back to specific individuals or their activities. K-means clustering: K-means clustering is utilized to identify similar price points across different data streams, facilitating the detection of potential arbitrage opportunities. Locality Sensitive Hashing: This algorithm is used to reduce dimensions from high-dimensional data. The basic idea is to hash input items so that similar items map to the same buckets with high probability. InfluxDB: It is a high-performance time-series database designed for handling large volumes of time-stamped data. It's popular for its scalability, speed, and ease of use in storing and querying time-series data for various applications like monitoring, IoT, and analytics. MongoDB: MongoDB is a NoSQL database. It is used for analyzing the clusters to analyze price differences within each cluster in this project. Grafana: Grafana is an open-source analytics and monitoring platform that allows users to visualize and understand their

data through customizable dashboards. It supports various data sources and offers powerful features for real-time monitoring, alerting, and exploration of metrics.

## II. Methodology

### A. Data Collection and Streaming with Kafka

There are many free API's for cryptocurrency and we tried to get coin API and verified with code for API connection and data retrieval. Then we integrated a Kafka Producer with a WebSocket to collect and process real-time trade data for Bitcoin from CoinAPI. Data is categorized and published to specific Kafka topics based on the exchange origin, such as bitstamp_btc_usd, bitfinex_btc_usd, and kraken_btc_usd. Kafka effectively handles high throughput and low latency requirements, ensuring data is continuously ingested and made available for processing.

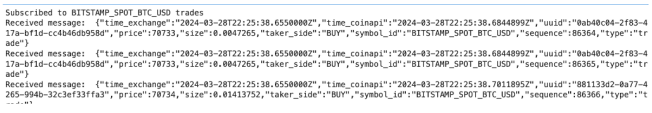


Fig. 1. CoinAPI data

### B. Data Processing and Analysis: Apache Spark, Bloom Filter and Differential Privacy Algorithms

Apache Spark Connection: A Spark session is initialized as an entry point for distributed data processing, configured to run locally with all available cores and to include the necessary Kafka package to interact with Kafka streams.

The use of a Bloom filter and differential privacy serves two key purposes: ensuring the uniqueness of data and protecting the privacy of individual transactions. These techniques are crucial for handling sensitive data in real-time streaming applications, particularly in fields like finance where data security and privacy are paramount.

**Bloom Filter**

*Purpose:* A SimpleFilter class represents a Bloom filter, a probabilistic data structure that is used to test whether an element is a member of a set. It's efficient in terms of space but allows for some false positives. This is used to quickly check the uniqueness of UUIDs without needing to keep all the data in memory, which would be impractical for large-scale data streams.

*Implementation:*

Initialization: A SimpleFilter class is defined, which sets up the Bloom filter with a specified size and number of hash functions. This setup impacts the probability of false positives and the efficiency of the filter.

Adding Items: When a new transaction comes in, the UUID of the transaction is added to the Bloom filter. Before adding, the filter checks if the UUID is already present using multiple hash functions to map the UUID to one or more positions in a bit array. If the positions are already set to True, it indicates a likely presence of the UUID (although false positives are possible).

Checking Uniqueness: For each transaction, the script checks if the UUID is already in the Bloom filter. If it is not, it processes the transaction further; otherwise, it skips processing, assuming it's a duplicate.

**Differential Privacy Algorithm**

*Purpose:* Differential privacy is applied to transaction UUIDs to anonymize them. This technique ensures that even if some data about individual transactions is leaked or exposed, it cannot be traced back to specific individuals or their activities.

*Implementation:*

Noise Addition: The function add_noise_to_uuid uses the Laplace distribution to generate noise, which is then added to the UUID. The noise-modified UUID is hashed to produce a new, anonymized identifier. This process ensures that each UUID is unique and cannot be traced back to the original without the exact noise value.

Hashing: After adding noise, the UUID is hashed using SHA-256 to ensure that the resultant UUID doesn't expose any original data. Hashing after adding noise also helps in further scrambling the data, thereby reinforcing privacy.

### C. K-means and LSH Implementation

In this project, we are using K-Means to identify clusters of similar prices, which could highlight common pricing trends across different times or different exchanges. Following this, LSH may be used to quickly query these clusters to find similar price points efficiently, aiding in the rapid detection of arbitrage opportunities. By combining these techniques, we leverage the strength of K-Means in finding meaningful groups and the efficiency of LSH in handling similarity queries at scale.

**K-means Clustering**

*Purpose:* K-Means is used to cluster transactions based on price to find groups of transactions with similar price points over time, which might help in identifying patterns or anomalies such as arbitrage opportunities.

*Implementation:*

Initialization: Start by choosing 'k' initial centroids, where 'k' is a user-defined number of clusters. The initial centroids are typically chosen randomly from the data points, or they might be chosen through other heuristics.

Assignment: Assign each data point to the nearest centroid. 'Nearest' might mean different things depending on the metric used, but it typically means the Euclidean distance.

Update: Once all points have been assigned to clusters, recalculate the centroids by taking the mean of all points assigned to each cluster's centroid.

Iterate: Repeat the assignment and update steps until the centroids no longer change significantly, indicating that the

model has converged, or until a maximum number of iterations is reached.

**Locality Sensitive Hashing**

*Purpose:* LSH is used after K-Means clustering. This might be to further refine the process of identifying and grouping similar data points (e.g., similar price points in different clusters), or to speed up the search for nearest neighbors within each cluster, thereby enhancing the performance of identifying arbitrage opportunities across different data segments.

*Implementation:*

Hash Functions: LSH uses multiple hash functions that are designed to increase the probability of collisions for similar items. Unlike cryptographic hash functions (which aim to minimize collisions), LSH hash functions aim to maximize collisions for similar items.

Buckets: Items are hashed to buckets with each hash function. Since similar items hash to the same bucket with high probability, each bucket potentially contains similar items.

Candidate Pairs: Items in the same bucket are considered as candidate pairs and can be compared further using a more accurate (and computationally expensive) similarity measure to verify their similarity.

### *D. Load Data into InfluxDB and Mongodb*

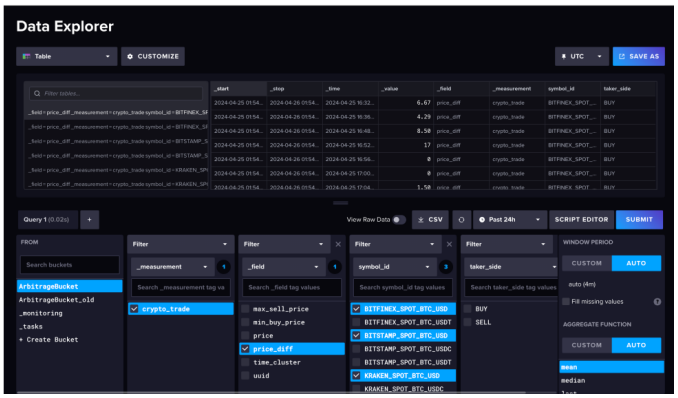


Fig. 2. InfluxDB Database

InfluxDB: We implemented a script that periodically loads JSON files into an InfluxDB database, specifically checking for new files every five minutes and then processing and uploading the data contained within them. The script is executed in a standalone manner, meaning it continuously monitors for new JSON files in the specified directory, processes each file by extracting data, and loads this data into InfluxDB. Once processed, files are moved to another directory to prevent re-processing, maintaining an organized flow of data from raw input files to database storage.

MongoDB: Implemented a python script that uses MongoDB with the pymongo library to automatically load JSON files into a MongoDB database. It also utilizes the APScheduler library to schedule this task periodically. This setup is ideal for scenarios where new data files are frequently generated and need to be added to a database for further processing or analysis. MongoDB leverages statistical and visualization techniques to extract meaningful insights from the data, such as trading dynamics and the effectiveness of hash-based data partitioning in the dataset.

### *E. ETL for K means clustering analysis and LSH for arbitrage opportunity*

For this project we performed ETL process for K-means clustering analysis and Locality-Sensitive Hashing (LSH) analysis to identify arbitrage opportunities using the trading data from MongoDB database. We extracted all the trades collections to retrieve relevant trade data such as symbol, price, and time of exchange. Cleaned and preprocessed the data for ensuring all timestamps and other fields are correctly formatted and normalized.For K-means created the features like 'price differences', 'buy/sell intensity'. For LSH, the hash values or the features used to compute hash values like price movements are correctly extracted. Following are the analysis done to understand the market dynamics and trade behavior, finding the arbitrage opportunities for earning profits and for monitoring and managing risk by identifying how trades are grouped and interacted.

**K-means clustering is applied to the trade data to identify clusters based on price movements or trading patterns.**

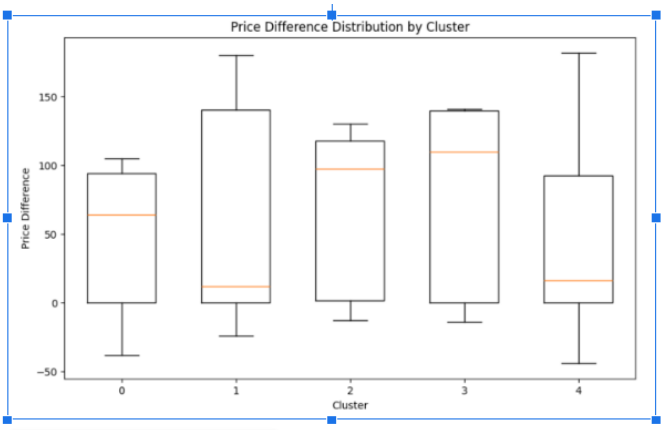


Fig. 3. Boxplot Visualization

The boxplot shows the variability and distribution of price differences across different clusters, which can be used to identify clusters with higher potential for arbitrage like larger price differences and also identifies the presence of any outliers or anomalies in the trading data.The boxplots show the central tendency and spread of the 'price_diff' data within each cluster. It calculates the 'price_diff' that is the difference between the buy and sell price within the same time cluster.

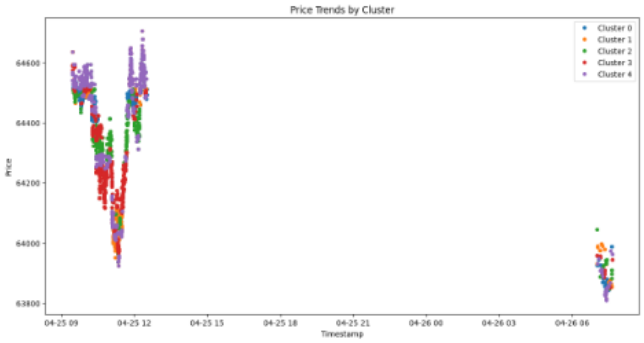


Fig. 4. Scatter plot Visualization

This is the scatter plot for the price trends of trades over time and categorizing them by the cluster each trade belongs to. Each cluster represents a grouping of trades based on certain characteristics determined by the K-means algorithm, for factors like price movement, volume, or time of day. Each dot represents the individual trade. It can be seen from the plot that there are concentrations of trade activity within specific time frames, and there are overlaps in price between the clusters which means clustering is not dependent on price but combined with other trade attributes.

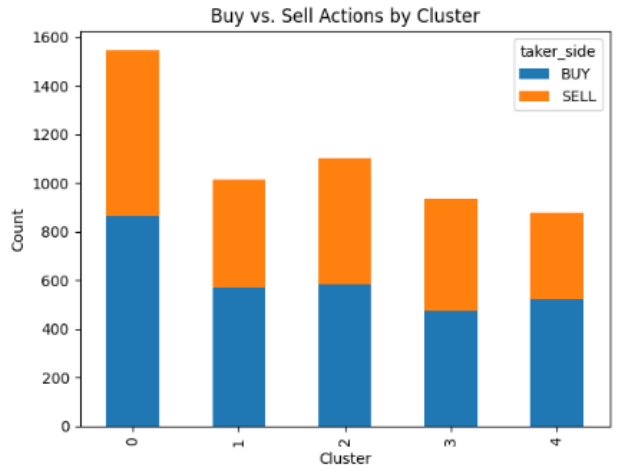


Fig. 5. Stacked Barplot Visualization

The stacked bar chart is created for analyzing the behavior of buy and sell actions within each cluster of trade data .It groups the data by 'time_cluster' and 'taker_side', which likely represent the cluster each trade belongs to and whether the trade was a buy or sell action, respectively.It counts the number of occurrences for each combination of cluster and taker side, creating a count for buy and sell actions within each cluster.Each bar represents a cluster, with the length of the bar indicating the total count of actions within that cluster, blue indicating the proportion of buy actions and orange indicating the proportion of sell actions.

**Apply Locality-Sensitive Hashing (LSH) to quickly identify similar trade patterns or price points, which could indicate potential arbitrage opportunities.**

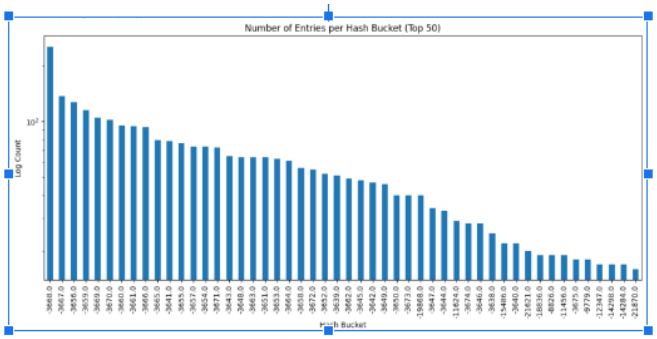


Fig. 6. Hash Bucket Distribution

This bar chart the analysis of the distribution of hash bucket counts from a Locality-Sensitive Hashing (LSH) process. It calculates the frequency of entries for each hash bucket and then displays the top 50 hash buckets in terms of entry count. Some hash buckets have significantly more entries than others. The first few hash buckets have the highest counts, with a sharp decrease as we move to the right, which indicates that certain hash bucket values have many more associated entries. Therefore, the particular hash values represent common or frequent trade characteristics in the dataset. The distribution helps in understanding the density and similarity of data points processed by LSH, which can be critical for tasks like identifying clusters of similar items in trading which could potentially be exploited for arbitrage opportunities.

### *F. Real Time Dashboard using Grafana*

We developed the real-time dashboard facilitated by Grafana for visualizing cryptocurrency trading data and enabling faster decision-making and identification of arbitrage opportunities across different exchanges. Grafana is a powerful tool for building dynamic dashboards that provide real-time insights into various metrics and trends. The dashboard retrieves data from InfluxDB, where processed trade data, including prices and timestamps, are stored. The dashboard refreshes every 5 minutes such that displayed visualizations are up-to-date with the continuous queries. Flux queries are utilized for querying InfluxDB in Grafana.

Following are the different elements or gauges of the real time grafana dashboard for cryptocurrency trading data : Minimum Buy Price Gauges: Three gauges show the lowest current buy prices for a particular cryptocurrency, such as Bitcoin (BTC), across three different exchanges: Bitstamp, Bitfinex, and Kraken.
Maximum Sell Price Gauges: Three gauges similar to the minimum buy price gauges depicting the highest current sell prices for the cryptocurrency on the same three exchanges. It shows the prices at which a seller will get profit if they want to sell at the highest current market price.
Latest Buy Trades Ticker: This horizontal ticker displays a

series of the most recent buy trades on all three exchanges with the associated prices. Each row represents a different trade, identified by the exchange, the trading pair (BTC/USD), the taker side as ”BUY”, trade price and trade time.
Latest Sell Trades Ticker: It is similar to the latest buy trades ticker but for sell trades. It shows the most recent sell trades across the exchanges, with the price at which each trade was executed and the exact time of the transaction.


Fig. 7. Grafana real-time dashboard

## G. Email Alerts

Implemented a python script to monitor a MongoDB database for arbitrage opportunities in cryptocurrency trades, and to send email alerts when such opportunities are detected. It involves querying the database, analyzing the data, and utilizing an email server to notify a recipient.

This script automates the process of detecting arbitrage opportunities by analyzing recent trades from a MongoDB database. When a profitable trade difference is identified based on predefined criteria, it sends an alert via email. This is a useful automation for traders or trading platforms looking to quickly capitalize on market inefficiencies without manual monitoring.

*Implementation:*
Email Sending (send_email): It uses Python’s smtplib and email.mime.text.MIMEText libraries to create and send emails. We set up the SMTP server details (like server address and port), sender’s email and password, and receiver’s email for it to work.
Arbitrage Opportunity Detection (process_opportunities: The system connects to MongoDB using pymongo’s MongoClient. It fetches recent trading data from the last 10 minutes. This data is processed into a DataFrame using pandas for analysis. It then identifies arbitrage opportunities by calculating price differences between buy and sell prices. If profitable opportunities are found, it sends email alerts with trade details.

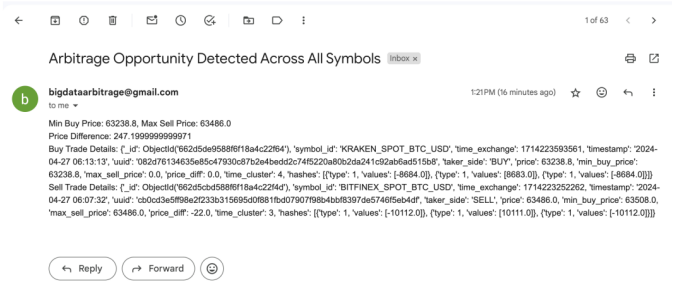


Fig. 8. Email Alert generated

## H. Fairness check on the implemented concepts

Fairness check explains why each is important for this project, what it specifically analyzes, and how to interpret the outputs obtained. This will provide a clearer understanding of how these checks contribute to the overall integrity and fairness of this project.

Data Aggregation by Symbol and Exchange: This fairness check ensures that data from different exchanges is being collected and represented proportionally in dataset. It’s crucial for preventing biases based on data source, which could skew trading decisions. To do so, we split the symbol_id to extract the exchange name and then group data by symbol_id while counting occurrences and collecting unique exchanges. The output shows the count of data points for each symbol_id, grouped by exchanges. For example, ’KRAKEN_SPOT_BTC_USD’ having a count of 4783 and associated with the ’KRAKEN’ exchange indicates that this particular trading pair from Kraken is well-represented in your data.

Fairness Check Across Clusters: This check aims to analyze if certain time clusters might systematically show more or less favorable trading conditions, which is essential to ensure that your trading strategy is not inadvertently biased towards specific times. To do so, we compute the average price difference per time_cluster, which gives insights into the arbitrage opportunities available in different time segments.

A bar chart with different average price differences for each cluster shows how these opportunities are distributed over time. Significant disparities, such as a much higher average price difference in one cluster, might indicate periods of high volatility or market inefficiencies that your system could be exploiting unevenly. This can inform adjustments to trading algorithms to ensure they operate fairly across all time periods, not just those with the most apparent opportunities.
Statistical Analysis for Fairness: This statistical test is conducted to verify if the observed variations in price differences across clusters are statistically significant and not just random fluctuations. An ANOVA test is used here to determine if there are statistically significant differences in the price differences across different time_clusters.

A very low p-value (e.g., 2.7177521964715263e-281) from the ANOVA test indicates extremely significant differences between clusters. This suggests that certain clusters consistently offer different arbitrage opportunities, which could lead to biased trading behaviors if not addressed.

Data Ingestion Check: To ensure that data is uniformly distributed across different symbols, which is vital for maintaining a balanced view of the market and avoiding skewed analyses based on incomplete data. Counting records per symbol_id to check how uniformly data is being ingested.

A histogram or bar chart showing each symbol and its count helps visualize which symbols are over or underrepresented. Disparities here could lead to an incomplete market analysis and biased trading decisions based on the more frequently updated symbols.

Analyze Data Characteristics: To examine if the characteristics of the data, such as average price differences vary significantly across different symbols, which could influence the fairness of the trading opportunities identified. Calculating the average price difference for each symbol to see if some symbols have inherently more price movement than others, which could affect arbitrage opportunities.

This analysis shows the average price difference by symbol. Large variations in these averages could suggest that some symbols are inherently more volatile or have more trading opportunities. If trading strategy disproportionately focuses on these symbols, it might not be treating all market opportunities fairly.

In conclusion, these fairness checks help ensure trading system is balanced across different dimensions such as time, market symbols, and data sources. They are crucial for avoiding systemic biases in automated trading environments, ensuring compliance with market standards, and maintaining the effectiveness and ethical integrity of your trading operations.

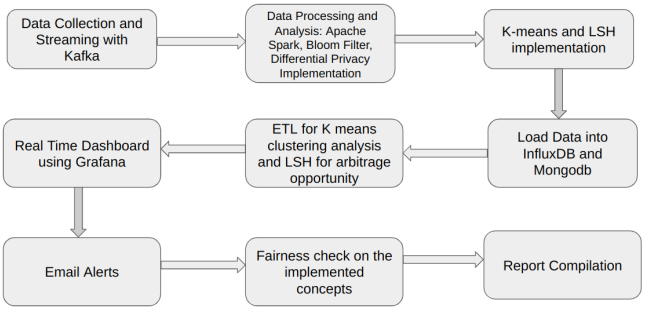


Fig. 9. Project Workflow

## III. RELATED WORK

[1] The paper by Levus et al. (2021) depicts the layout of an intelligent system that uses a client/server scheme based on Node.js to do arbitrage activities based on arbitrage chances in crypto markets. The authors conduct persistent monitoring of the price disparities on different exchanges to find arbitrage opportunities. The system is meant to work on a global scale, covering many crypto trading platforms, and thus it can accumulate a great volume of arbitrage deals. The authors build a WebSocket messaging protocol-based infrastructure to enable real-time communication with exchange APIs and to prevent data delays. On top of that, the paper concerns Telegram bots and web interfaces for sending users notices about arbitrage chances, which makes the system more interactive. This characteristic allows users to act on time to the alerts coming from the system. The backend, which relies on Node.js, treats the asynchronous data streams perfectly, but not the way Apache Spark does in its processing and real-time analysis. On the other hand, our project besides the detection of such opportunities using Apache Spark for high-performance data processing and real-time stream analytics also includes clever algorithms such as Bloom Filters, Differential Privacy, K-means, and Locality Sensitive Hashing. The technologies advance the system's performance in terms of data processing capacity, privacy, and arbitrage identification in which it uses the patterns generated from trading data. Integration of Grafana into our project made it more advanced in providing better data visualization and thus clients will have a more dynamic and insightful user experience. The improvements to the Node.js foundation and more sophisticated approaches to alerting personnel used by Levus et al. prove this system effective in terms of scalability, performance, and having a more rigorous approach to dealing with the numerous challenges in cryptocurrency arbitrage.

[2] The paper by Horvat et al. (2020) achieves the objective of handling simultaneously batch and data streams for cryptocurrency markets in real-time processing. This process is done to support the analysis of blockchain network trends and transactions for the market using the Lambda architecture technique. Such modular design facilitates the integration of components, as well as advanced analytical capacities like the detection of real-time events in the blockchain, market trends, and the social media influence on cryptocurrency reputations. Unlike the Lambda approach of separating the batch data processing and maintaining a fast speed layer, our Spark solution is built around a unified stream processing model that may minimize the system complexity as well as reduce the data latency, hence passing through these layers is not required. Additionally, our system comes with much more sophisticated visualization tools like Grafana, focusing on the dynamics of market behavior—an aspect that Lambda Architecture does not consider at all. In sum, our project consonantly focuses on scalability, performance, and user-friendly analysis along with Horvat et al.'s model and consequently, it offers an optimized solution for the challenges of real-time cryptocurrency arbitrage detection.

[3] The paper by Ayvaz and Shiha (2018) suggests a big data processing architecture for sentiment analysis of social media data which is adaptive and scalable. It depicts that the architecture has been tested for a particular use case, and it

discusses how cryptocurrencies can be analyzed using it. The Apache Spark complement in their method is used to process data while Apache Zeppelin is used for visualization to provide dynamic sentiment analysis and provide insights on the effect of social media sentiments and cryptocurrency prices. This technique is a significant aspect of knowing the present public opinions on market behaviors. Our approach is unlike Ayvaz and Shiha's general sentiment analysis framework by offering a solution for financial arbitrage that is more focused. In a contrary way, their study provides the generalized rule, which may further be applied beyond financial markets, potentially becoming a more adaptable tool dealing with sentiment analysis across various domains. Its ability to be just more adaptive gives it an edge over other approaches providing better real-time insight into the public sentiment in more portions of problems where instantaneous public opinion plays a critical role in decision-making.

[4] The paper by Barradas et al. (2022) describes a real-time big data architecture using the kappa design, enhanced for working with cryptocurrency and social media data, especially Twitter. This function employs k-means clustering on data to detect and demonstrate the correspondence between public sentiment on social networks like Twitter and cryptocurrency market behaviors. This strategy underscores the need to incorporate various data sources and consequently develop a holistic view that captures both transactional data and social media feeds. Unlike our project, which also employs real-time data processing to uncover arbitrage windows in cryptocurrency markets, Barradas et al.'s approach provides a solution that can be applied to a wider range of social media effects and financial operations which could be an advantage because of its double concern. While we rely on Apache Spark for streamlined processing and use Differential Privacy, as well as Bloom Filters to enhance the security and quality of data, Kappa architecture used in their research seems to offer superior integrating capacities to process diverse data in real-time, which is essential for applications that deal with the instant merging of analytics from multiple data flows. However, our specific solution, which is exclusively about market financial arbitrage, may be more finely tuned for financial systems, delivering specialized information with shorter latency and a simpler system, since there is no batch layer processing. This provides our system an advantage in the highly dynamic world of cryptocurrency trading, where both speed and data integrity are the ultimate priority.

[5] The research done by Raju and Taif (2020), looks at the present predictability of the price of Bitcoin using machine learning methods and analysis of the public sentiment. The study predominantly analyzes the sentiment feedback from Twitter and Reddit by unveiling the relationship between community perception and market price. The study employs various machine learning algorithms, such as those existing in LSTM networks, in building predictive models that quantify Bitcoin price rallies and/or dips to reflect the sentiment expressed in social media posts. The importance of the study is to explore extensively LSTM methods of predicting accuracy and then compare the results with ARIMA, the former forecasting model, which will give us an idea of what kind of impact public sentiment can have on the figures quoted in the market. On the contrary, ours adapts to the aspect of spotting accidental opportunities by using Apache Spark to analyze real-time data streams. Unlike the price predictions, the method that we employ uses complex data processing algorithms including differential privacy and locality-sensitive hashing for data enrichment of privacy protection and accuracy. These skills are not found in the sentiment analysis of the model. The study offers valuable insights into the effect of crowd-wise sentiment on price, while our project is the close-up solution for the arbitration market of high-speed environs because of its presence in real-time, scaling ability, and the specified financial data analysis.

[6] The paper by Mohapatra et al.(2019) proposes a new cryptocurrency price prediction platform entitled "KryptoOracle." The platform incorporates the sentiments of the Twitter community and is built on a Spark-based system. The platform pulls from many sources at the same time in a durable and fault-tolerant way. Sentiment analysis being a part of the system helps natural language processing queries to get real-time responses. Apart from this, an online learning model is employed which uses the new price and sentiment data to dynamically update its weights and to facilitate continuous learning and adaptation to market changes. The KryptoOracle platform is designed to help with accelerating decision-making and timely insights by using social media data to predict the movements of cryptocurrency prices efficiently. Similarly to KryptoOracle, our project has the primary goal of using real-time data for cryptocurrency market analysis. Although KcryptoOracle provides a market predictive feature through sentiment analysis, our project outskirts it with financial arbitrage detection which offers special tools for a niche but an essential part of cryptocurrency trading.

## IV. Experimental Setup

### A. Tools used

The data processing environment uses the hardware configuration of MacBook with Docker containers to emulate a distributed system. This way, one could have a local copy of the Apache Kafka and Apache Spark running virtually to ensure a formidable and replicable environment for each member of the team.

The programming configuration provides Apache Kafka for data ingestion and streaming. Kafka topics are essentially designed to take data from the cryptocurrency exchanges in real-time leading to quick processing and analysis. Apache Spark is picked for data processing and analysis tasks, it may work with large loads of data and give real-time analytics and processing that is aided with distinct features such as Bloom filters and differential privacy to be sure that data is unique and private. With Docker, Kafka and Spark instances are containerized thus a particular machine can run the whole software stack without any compatibility issues.

### B. Data Source

The project works with two key data sets from the real-time cryptocurrency market for its data. The main source is CoinAPI, which is a streaming service that gives cryptocurrency market data for multiple exchanges. This service is given through an API key provided access to it. We also use real-time data from exchanges, like Bitstamp, Bitfinex, and Kraken, which are streamed using WebSocket connections. Data should be processed according to its source and as such, particular Kafka topics are given for each exchange.

The project is particularly founded on Bitcoin trading pair live streaming like BTC/USD and mainly has access to live streaming data for analysis purposes. The data is comprehensive considering it contains transaction time, price, and exchange identifiers for identifying swings in rates. The analysis also includes data concerning stablecoins such as Tether and USD Coin. Both Tether and USD Coin are pegged to the US dollar but are separately managed. Among these, Tether (USDT) is managed by Tether, while Circle and Coinbase oversee USDC. Despite both stablecoins aiming to maintain a 1:1 on par with the US dollar, the respective management by other groups can cause minor value changes between them, creating several arbitrage opportunities. The variations in conversions are essential for the project because they specify the directions of profits in the dynamic cryptocurrency market.

### C. Evaluation and Performance Metrics

The evaluation metrics, as well as performance measures, are quite inevitably critical features that one cannot afford to ignore as he/she designs a real-time system to deal with high-speed cryptocurrency data. The main performance indicators are the delay of data processing and the volume of data stream delivery in the streaming channel of the pipeline. These metrics are indeed the most crucial part of the arbitrage trading system since the timely processing of data could boost the probability of finding arbitrage opportunities and overall profitable trading. The efficiency of the mechanism is measured through its capability to correctly detect and report anomalous market patterns. This task entails rating the opportunities detected and those that occurred to assess the forecast quality. Arbitrage detection accuracy is a paramount metric for work properness.

The scalability of the system so that it can handle real-time data is also another essential feature. The reliability of the system to operate without failures is also another feature that the system must possess. The advanced cryptocurrency data are collected at a fast rate, hence, there is a need to implement a system that will maintain its speed even with the high-volume data. Another important thing is for the system to be reliable, without breaks while functioning. In the end, by applying differential privacy and Bloom filters, it is imperative to measure how the system will balance between the protection of the user's privacy and the integrity and uniqueness of transaction data. Privacy and data integrity of patients are key criteria for the proper performance of the system, considered among the most important metrics in the assessment of its performance.

## V. Results and Analysis

**Application of KMeans, LSH, and Bloom Filters:** Our system leveraged the KMeans clustering model to group and identify trading sequences that could be used for arbitrage trading across several cryptocurrency markets. Locality-Sensitive Hashing(LSH) essentially enabled the efficiency of real-time trading by recognizing similar trading patterns which is an important factor in this sector to detect the arbitrage opportunities. Redundancy elimination and duplicate avoidance in data processing was made possible through the inclusion of Bloom filters. This ensured that only vital and relevant data is being used.

**Impact of Differential Privacy:** To ensure the privacy of the end-users, differential privacy methods were used. Through the implementation of these techniques, controlled noise was injected into our aggregated data, making it impossible for anyone to trace back any transaction to its origin. Through this implementation we not only complied with privacy rules but also instilled trust within our users.

**Real-Time Dashboard Performance:** The Grafana-powered real-time dashboard displays key metrics like the 'min buy price', the 'max sell price' or the recent trades for all the exchanges being monitored. In addition, the dashboard updated those metrics in real-time so the traders could make proper decisions based on timely updated data.

**Accuracy and Responsiveness of the Arbitrage Detection and Alert System:** The chosen algorithm for arbitrage detection was very accurate in recognizing the best trading opportunities when price fluctuations met predetermined levels. Precision was the primary success measure for the project as it directly affects the users trades profitability. The real-time email alert system, checking for arbitrage options alerted by the algorithm, was tested for its speed and reliability. Consistent alerts were sent that meet this time frame in which traders can implement their trading strategies and use the profit making opportunities to their advantage.

**Observations and Insights:** Market Dynamics: Insights gotten from the dashboard spurred the understanding that cryptocurrency exchanges are ever changing, and some perform better than others in adjusting to market shifts. Effectiveness of Privacy and Data Integrity Measures: Through the mechanisms of Bloom filters and differential privacy not only user data was protected but also the workload of the system was optimized.

**Conclusion:** The combination of KMeans, LSH, Bloom filters, and differential privacy algorithms helped our data processing pipeline to deal with large data sets efficiently and securely. The real-time dashboard gave an insight into the market with live data that users could use for prompt action. On the whole, the system proved to be quite proficient in discovering and indicating arbitrage situations expeditiously and, thus, providing traders the chance of converting these situations into their own profits.

## VI. Future Work and Conclusion

**Conclusion**

The project successfully demonstrated the application of real-time monitoring and analytics systems for detecting arbitrage opportunities across various cryptocurrency exchanges using data processing techniques like Kafka and Spark. By integrating multiple data sources from prominent exchanges like Bitstamp, Bitfinex, and Kraken, the system efficiently processed streaming trade data to identify arbitrage opportunities across different trading venues.

The system effectively handled data redundancy while maintaining user privacy with the use of the Bloom filter and differential privacy method. The arbitrage detection process is improved with the use of the K-Means clustering algorithm. It is used to segment large datasets into clusters based on the similarity of trade prices. By identifying clusters of trades with similar price points, the system could identify large price differences between exchanges or time intervals.

The integration of these machine learning techniques like K-Means clustering and Locality-Sensitive Hashing (LSH) further enabled the system to predict and take advantage of price differences in almost real time. This is crucial in the extremely volatile cryptocurrency market. This capability was complemented by the robust data infrastructure, including MongoDB and InfluxDB by simplifying the storage and retrieval of data. MongoDB is used for analyzing the clusters to analyze price differences within each cluster and Locality-Sensitive Hashing (LSH) efficiently identifies similar price points and forms clusters to find pairs or groups of trades that exhibit minimal price differences but are separated by time or exchange. This setup facilitated dynamic analysis and visualization of data using Grafana. The team faced various challenges during the project, including handling complicated data flows and preserving cost-effectiveness when using cloud services hence moving to temporary storage. It sets a new standard for financial analytics real-time data processing. This project has significant practical implications since it gives financial institutions and bitcoin traders an effective tool to improve their profitability and decision-making. Additionally, the project makes a major academic contribution to the field of real-time big-data processing.

**Future Work**

- Improved Analytical Features: Using machine learning models to forecast future price changes based on historical data may give arbitrage opportunities a competitive advantage. Moreover, including sentiment research from news and social media platforms may offer a more in-depth understanding of market trends.
- Expanding more data sources: The project currently combines information from Bitstamp, Bitfinex, and Kraken so adding more exchanges for cryptocurrencies and financial data feeds to increase the project's capacity for detecting arbitrage, data diversity, and trading opportunities.
- Utilize machine learning for anomaly identification to identify unusual trade patterns or possible manipulations of the market.
- Enhance the alert system to include automated trading actions when particular conditions are fulfilled or push messages to mobile devices.
- Faster Real-Time Processing: Although Apache Kafka and Spark are used in the project to process data in real-time, more improvements can be made to lower latency and boost throughput. This may include upgrading the current configuration or investigating novel stream processing technologies.
- Robust Error Handling and Recovery Mechanisms: Creating more complex error handling and recovery procedures will help to guarantee that the system keeps working even when there are failures, which will increase its robustness and dependency.
- Automated Trading Capabilities: Developing capabilities for automated trading based on predefined criteria and real-time data would be a logical extension of the existing project, which focuses on delivering data and insights for manual trading decisions.
- Collaborations and Partnerships: Forming partnerships with trading platforms or financial organizations may help provide integrated trading solutions, improve data access, and provide a competitive advantage.

## VII. Fairness

The phrase 'fairness' in our project is displayed by its equitability, impartiality, openness, and easy access. It starts off with democratic data gathering. Utilizing various exchanges through tools like Apache Kafka and subsequently working on data using Apache Spark, we seek to make information accessible that until now has been used by the elite few with digital assets and high-computational technologies. Keeping ethical concerns in mind, we endeavor to make sure our calculations - whether for the detection of arbitrage opportunities or for safeguarding uniqueness and privacy of data - are not introducing bias or an unfair advantage to any user group.

Differential privacy approaches we used indicate that we are devoted to protecting our user's privacy. An example is the segregation of clustering patterns by means of the K-Means clustering technique. This methodology enables us to place similar trade prices together, effectively leveling the playing field between all types of market participants, irrespectively small or big their trading volumes or market power. The usage of LSH algorithms also helped in maintaining the integrity of the data as well as speeding up matching similar price points which is integral in fast-moving cryptocurrency trading. LSH allows the fair identification of imported prices with similar trade prices that have the potential of yielding profitable arbitrage outcomes. Using LSH achieved reliable and unbiased processing of large amounts of data without bias to any particular dataset or user.

The infrastructure is also specifically designed to be resilient and equitable. With scalable cloud solutions, the platform addresses high-volume data without sacrificing the speed or

accuracy. Consequently, all users receive accurate data swiftly irrespective of the scale of their operations.

In conclusion, fairness is not just our guiding principle but a feature that spans all processes of the project, from collecting and processing to analytics and insights' distribution. We aaim to establish a level playing field for fair trade in the cryptocurrency market through our endeavor that is unbiased and transparent.

## VIII. Key Learnings

- Real-Time Data Processing: Using Spark for data processing and Kafka for message handling, the project successfully develops a pipeline for ingesting and processing real-time data from cryptocurrency markets. Using Kafka to manage real-time data feeds and Spark to execute real-time data transformations and aggregations, we learned how to handle streaming data effectively.
- Privacy and Security Measures: Utilized Bloom filters and differential privacy to ensure that the system processed unique data entries and protected the privacy of user transactions. The application of differential privacy methods safeguarded individual transaction details, addressing critical concerns regarding user confidentiality and data security.
- Machine Learning for Anomaly Detection: Using machine learning techniques like K-Means clustering and Locality-Sensitive Hashing (LSH) helps in identifying patterns and anomalies in data, such as potential arbitrage opportunities in market prices.
- Integrating InfluxDB and Grafana for data storage and visualization to provide users with actionable insights and make data-driven decisions in trading that could be exploited for profit.

## IX. Innovation

Utilizing Locality Sensitive Hashing (LSH) to improve the efficiency of matching similar price points across large datasets is an innovative strategy. This technique improves the speed and precision of arbitrage opportunity detection. The integration of Bloom filters and differential privacy to secure transaction details is particularly innovative. These technologies ensure that the system only processes unique data entries and maintains the confidentiality of user transactions, which are crucial in sensitive financial environments. This project provides help in exploiting arbitrage opportunities more quickly and accurately, significantly contributing to market efficiency. It allows for the immediate detection of price discrepancies across exchanges, which is innovative in its application to arbitrage opportunities It contributes to the application of big data technologies in the financial sector, demonstrating how real-time data streams can integrate with advanced data processing and analytics.

## X. Technical Difficulties

- The major difficulty we faced with GCP was its computational expense when executing multiple Spark sessions and handling extensive data across various local storage locations within the Google Cloud Platform environment. This issue impacted performance, scalability, and cost efficiency, ultimately resulting in system crashes. Consequently, we were unable to achieve our objectives with the free resources available. Running multiple Spark sessions simultaneously can lead to inefficient resource utilization because resources are not shared across sessions, potentially resulting in higher costs due to excessive consumption of computational resources. Therefore we moved to local storage.
- After loading data into InfluxDB, parsing does not align with the expected formatting i.e. data is stored in separate rows for each attribute instead of one single line.
- Flux queries are not user-friendly, and proper grouping does not work as expected. Flux, the data scripting and query language for InfluxDB is very complex and encountered difficulty in data grouping and aggregation.
- Email alerts do not work because they require the creation of app-specific passwords. After all, in Gmail it requires app-specific passwords for applications accessing the account providing an extra layer of security.
- The Email alert received had discrepancies across different exchanges indicating unrealistic values like minimum buy price were reflected as zero.

## XI. Prospects of winning competition/publication

Our project is implementing a real-time analysis and processing using Apache spark along with advanced methods like bloom filters, differential privacy, K means, and locality-sensitive hashing. These technologies offer system processing capacities, privacy protection, and permission slips for arbitrage generated by spotting a pattern in the trading data.

Moreover, the visualization of data by using Grafana in the project enabled us to add advanced features. In relation to this integration, we are now able to offer our clients a more modern interaction as well as contribute to providing them with more detailed and insightful information, hence helping us to deliver better-quality service.

In essence, the project involves the usage of the latest tech-powered solutions and algorithms to help detect opportunities and to improve the system performance based on data processing capacity, privacy, and arbiral opportunies. In the context of Grafana integration, we can mention that it has led our team to a new level of excellence and peak of the work. We can provide the most eligible customers with innovative and amazing service. We have high chances of getting the paper published in conference.

## XII. CRediT statement (Contributer Roles Taxonomy)

Contributions to the project were as follows:

- **Conceptualization:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut

- **Methodology:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut
- **Software:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut
- **Validation:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut
- **Formal Analysis:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut
- **Investigation:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut
- **Resources:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut
- **Data Curation:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut
- **Writing – Original Draft Preparation:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut
- **Writing – Review & Editing:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut
- **Visualization:** Riddhi Kamleshkumar Vyas, Soujanya Sankathala, Raveena Kagne, Priya Govindarajulu, Tanvi Prashant Pagrut

## REFERENCES


[1] Levus, R., Berko, A., Chyrun, L., Panasyuk, V., Hrubel, M. (2021). Intelligent System for Arbitrage Situations Searching in the Cryptocurrency Market. In MoMLeT+ DS (pp. 407-440).

[2] Horvat, Nebojša Ivković, Vladimir Todorović, Nikola Ivančević, Vladimir Gajic, Dusan Luković, Ivan. (2020). Big Data Architecture for Cryptocurrency Real-time Data Processing. https://www.researchgate.net/publication/343193189 Big Data Architecture for Cryptocurrency Real-time Data Processing.

[3] Ayvaz, S., Shiha, M. O. (2018). A scalable streaming big data architecture for real-time sentiment analysis. In Proceedings of the 2018 2nd International Conference on Cloud and Big Data Computing (ICCBDC '18) (pp. 47–51). Association for Computing Machinery. https://doi.org/10.1145/3264560.3266428.

[4] Barradas, A., Tejeda-Gil, A., Croda, R.M. (2022). Real-Time Big Data Architecture for Processing Cryptocurrency and Social Media Data: A Clustering Approach Based on k-Means. Algorithms, 15, 140.https://www.mdpi.com/1999-4893/15/5/140.

[5] Raju, S. M., Tarif, A. M. (2020). Real-time prediction of BITCOIN price using machine learning techniques and public sentiment analysis. arXiv preprint arXiv:2006.14473. https://doi.org/10.48550/arXiv.2006.1447.

[6] S. Mohapatra, N. Ahmed and P. Alencar, "KryptoOracle: A Real-Time Cryptocurrency Price Prediction Platform Using Twitter Sentiments," 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 2019, pp. 5544-5551. doi: 10.1109/BigData47090.2019.9006554.

## XIII. Appendix

TABLE I
DATA 228 Term Project Rubric

| Criteria | Link/Comment | Pts |
|---|---|---|
| Code Walkthrough | Slide for code walkthrough included for presentation. | 5 pts |
| Presentation Skills Includes time management | Includes time management | 5 pts |
| Discussion / Q&A | Slide for Q&A added, Time allocated for questions | 5 pts |
| Demo | Slide for demo (Real-time Dashboard) included for presentation | 5 pts |
| Report Format, completeness, language, plagiarism, whether turnItIn could process it (no unnecessary screenshots), etc | Yes we have validated all these factors while writing a report. | 7 pts |
| Version Control: Use of Git / GitHub or equivalent; must be publicly accessible | Publicly accessible Git/GitHub repository: https://github.com/riddhi-vyas/DATA228_bigdata_project.git | 3 pts |
| Lessons learned | Included in the report and presentation? Included in the report and presentation under the topic Key Learnings and Limitations. | 5 pts |
| Prospects of winning competition / publication | Yes, it is mentioned in report. | 3 pts |
| Innovation | Slide included in the presentation and added in the report. | 5 pts |
| Teamwork | The team connected using zoom meetings every week for follow-up. The screenshot attached. | 5 pts |
| Technical difficulty | Included the difficulties faced by the team under the same Key Learnings and Limitations in report and ppt | 5 pts |
| Practiced pair programming? | Used Google Colab for pair programming. Screenshot for all five team member access and google collab coding screenshot attached.https://colab.research.google.com/drive/1xDsDXmApLZifZawlL344NyvCEF0t-953?usp=sharing | 2 pts |
| Practiced agile / scrum (1-week sprints)? Submit evidence on Canvas - meeting minutes, sprint backlog, and any other artifacts. Use tools such as https://trello.com/en-US/pricing | Evidence submitted, Trello board for agile. Screenshot attached for evidence. Methodology: https://trello.com/invite/b/flR5VM72/ATTI05e03700150d2f725f07a2d5ae59f0399026BC91/big-data-project | 2 pts |
| Used Grammarly or other tools for language? Grammarly free version is sufficient; can use other tools as well. Submit report screenshot on Canvas. Hint: You can convert latex to pdf to word and then run Grammarly. | The team used a Grammarly app while writing the content. Aiming for a score of 90 before adding the content to the report. Grammarly screenshot attached. | 1 pts |
| Slides | | 3 pts |
| Used LaTeX? Upload .tex file (it should indicate that the IEEE LaTeX template was used and not generated from doc or other format). | LaTeX has used to write the report, providing a professional and consistent format for presenting the results of the project. https://www.overleaf.com/read/yyhjvzmncwgj#b045e0 | 2 pts |
| Used creative presentation techniques? Use of Generative AI is ok here. Try animation, effects, newer features such as those offered by prezi, etc. | Prezi is used for preparing presentations. Screenshot attached for prezi tool usage. | 2 pts |
| Literature Survey<br>1. Did not miss out on any important existing work that is relevant to the project.<br>2. Literature survey is organized into meaningful subsections<br>3. All references are cited and follow standard notation used in the template | Literature survey is added to the report. | 7 pts |
| Use of streaming algorithms: Use of Reservoir Sampling, Bloom Filter, Flajolet-Martin, DGIM, Computing Moments, algorithms for graph streams, etc | Bloom Filter is used as a streaming algorithm. | 5 pts |
| Use of Stream Processing Frameworks such as Spark, Flink, and Kafka | Kafka used for Data Streaming and Apache Spark used for Data Processing and Analysis. | 5 pts |
| Use of Locality Sensitive Hashing | LSH algorithm is used to find price Similarities by hashing prices along with K-means clustering | 5 pts |
| Use of Privacy techniques: K-Anonymity, L-Diversity, Differential Privacy, etc. | Differential Privacy technique is used to preserve privacy of user information such as identity. | 5 pts |
| Any other tools and techniques covered in the course not included in the other criteria Think Machine Unlearning, Explainable AI; Fairness; Federated Learning; Data Poisoning, Responsible AI, etc | Fairness check done on data ingestion and clusters check with analysis. This is explained in both slides and report. | 5 pts |
| Use of new tool(s) that were not used for any of the HW | K-means algorithm is used. This is explained in Slides and reports. | 3 pts |
| Elevator Pitch Video | Created and uploaded video on YouTube. https://youtu.be/gh-0u2vF0cQ?si=dCR8sl59j8hjlz77 | |