International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING
ISSN:2147-6799 www.ijisae.org Original Research Paper

Applying AWS and the Kafka Framework for Real-Time Weather Data Analysis

M. Jahir Pasha[1], V. Vijaya Chandra Rao[2], P. Bhasha[3], D. William Albert[4], V. Sujatha[5], K. K. Baseer[6*]

Submitted: 28/06/2023 **Revised**: 07/08/2023 **Accepted**: 26/08/2023

Abstract: The weather forecasting assesses the change that is now taking place in the state of the atmosphere, making it an important and crucial process in people's everyday lives. Big data analysis is a technique for examining enormous quantity of information in order to discover undiscovered prototype and vital details that may improve outcomes The concept of Big Data has captured the attention of many spheres of society, including the climatology institution. Big data analytics will thereby develop climate prediction outcomes and help forecasters produce more precise weather predictions. Numerous big data techniques and technologies have been created to manage and analyse the substantial amount of weather data from diverse sources in order to accomplish this goal and provide useful responses. Issues with traditional data management approaches and technology solved by using analytics for big data in prediction of weather. The data is processed and analysed using cloud computing tools including Kafka, AWS S3,AWS Crawler, AWS Glue Data Catalogue, and Athena AWS. This framework helps in dealing and analyzing real-time information in other segments like Stock exchange, e-business functionalities, gaming, etc. By using this framework useful features and hidden patterns in the real-time data can be extracted and analyzed. In addition, Machine Learning Model can be trained on the data that we generated..

Keywords*: Weather data, Big data analysis, Kafka, Amazon AWS, Amazon S3, Amazon Athena, Visualization, Zookeeper*

1. Introduction

Analyses of weather data are very crucial to human existence. Preparing for natural catastrophes such as floods and droughts, as well as the agriculture and tourism industries, benefits greatly from precise weather forecasts. In the media world, public sector, and industrial farming, climatology is very valuable commercially. Large databases of imagery from both radar and satellite sources are used in this data-driven field, necessitating the use of effective technology for data storage, computation, and extraction in order to uncover hidden insights.Weather forecasting is a critical field that helps to protect lives and property by providing advanced warning of severe weather events. With the help of big data analytics and Apache Kafka, weather forecasting can be improved by processing and analyzing large amounts of data in a timely and efficient manner Weather forecasting relies heavily on observational data, which is collected from various sources such as weather stations, radar, and satellites. Remote sensing technologies such as radar, lidar, and satellite imagery are used to collect this data. Machine learning techniques such as neural networks and decision trees can be used to analyze chronological weather data and identify patterns that can be used to make more accurate predictions.It requires the use of powerful computer systems to run complex models and simulations. High-performance computing technologies such as supercomputers and cloud computing are used to support these calculations.Prediction of weather often involves the use of complex data and models, which can be difficult to understand. Visualization tools such as maps, charts, and animations are used to help forecasters make sense of this data.Real-time streaming data pipelines and applications are created using Kafka. The data is processed and analysed using cloud computing tools including Kafka, AWS S3 (basic storage service), AWS Crawler, AWS Glue Data Catalog, and AWS Athena.

[1]*Associate Professor, Department Of Computer Science & Engineering. G. Pullaiah College of Engineering and Technology,Kurnool, India. Email Id: jahir4444@gmail.com*
[2] *Assistant Professor, Department Of Computer Science & Engineering. G. Pullaiah College of Engineering and Technology,Kurnool, India .Email Id: vijayachandraraocse@gpcet.ac.in*
[3]*Assistant Professor, Department of Data Science,School of Computing, MOHAN BABU UNIVERSITY (Erstwhile Sree Vidyanikethan Engineering College (Autonomous))Tirupati – 517102, Andhra Pradesh. Email Id: bhasha.p@vidyanikethan.edu*
[4]*Professor, Ashoka Women's Engineering College Kurnool. Email Id: dr.albertdwgtl@gmail.com*
[5]*Professor, Ashoka Women's Engineering College Kurnool. Email Id: vadde.sujatha49@gmail.com*
[*6]*Professor, Department of Data Science, School of Computing, MOHAN BABU UNIVERSITY (Erstwhile Sree Vidyanikethan Engineering College (Autonomous))Tirupati – 517102, Andhra Pradesh. Email Id: dr.kkbaseer@gmail.com (Corresponding Author)*

2. Literature Survey

In [1], the authors developed a weather monitoring system by using the message queuing telemetry transport (MQTT) technique. Direct database connections were replaced with (MQTT), which makes it simple to create distributed information systems and may remove system

migration complexity from various relational database management systems (RDBMS). In conclusion, the system is described as an execution of the IoT-based climate supervising system.

In [2], numerous platforms and types of equipment are used to analyse weather monitoring systems. One of these is Klara, an Android app loaded on a smartphone that generates a multi-day prediction that includes temperature and atmospheric pressure. The Meteotest API, which offers climatic and satellite data for certain geographic coordinates as well as air pollution measures and backcasting, is another method used for weather forecasting by the authors.

In [3], the authors describe about a cost-effective IoT infrastructure for monitoring and storing meteorological data, such as temperature, humidity, air pressure, and dust particles in a residential area is designed and deployed using open-source technology. The virtual private server (VPS) at a distance receives the data from the Internet of Things (IoT) device. A server programme runs continually to gather the data and log it into a database. Additionally, there are instructions on how to install an IoT server application that makes use of the message queuing telemetry transport protocol, safeguard a VPS server, and set it up.

In [4], In order to enhance service reliability and network performance, the SMS vision relies on the effective administration and best use of network services and infrastructures through the use of ICT, The projected system consists of networking cables, facts parsing tools, text processing, offline and real-time data modeling, large data store and decision levels, and other facts resources. To develop each Big Data layer, comparative study of the available Big Data solutions might be used. Additionally, as a proof of concept, the roving users identification model was created using Apache Spark application. The model filters information from streaming procedures before it is deserialized and transmitted to the Kafka application. The model tests verified and validated Apache Spark's capability to establish the basis for a Data centre as a key function for processing data from online mobile networks.

In [5], The authors created a Wireless Sensor Network(WSN) system for CO2 monitoring by distributing a massive volume of data using Kafka and Impala. Data from sensor nodes is collected, temporarily stored, streamed via the Kafka platform, and finally stored in the Impala database. The system performed well when tested with data collected from our own constructed sensor nodes.

In [6], artificial neural networks (ANN) are used, and we choose a few important factors as well as network metrics as the features. They created a Kafka testbed using Docker containers to get enough training data for our model. Given varied network conditions, Kafka's dependability for various application situations can be estimated using the neural network model. A crude dynamic configuration strategy is used in the trials that greatly increases the dependability while ensuring message timeliness. They suggested a crude dynamic configuration strategy in the trials that greatly increases the dependability while ensuring message timeliness.

In [7], The authors suggested a design that makes use of open source tools like Apache Kafka for online and offline message ingestion and Apache Spark is used for processing, simplifying, and structuring current and real-time data. The structure is used to portray the processed data in a more captivating and readable manner.

In [8], The telemetry service is used to present an unique Kafka-based monitoring framework. The authors suggested a framework that goes above conventional monitoring solutions by utilising Kafka's inherent scalability and dependability. The framework enables the continuous monitoring of optical system data and the delivery of such data to numerous clients via straightforward compressed text messages. Additionally, the suggested structure maintains activities like root cause failure research are made easier by a compressed history of the observed data. The created monitoring platform's practical assessment, scalability, resilience, and end-to-end message latency are experimentally proven, while taking the disaggregated paradigm into consideration.

In [9], The authors provided a method for adapting an existing brownfield installation for usage in an industrial IoT context by building a modular and adaptable gateway. A streaming platform-based, highly decoupled architecture is suggested to meet time restrictions and flexibility needs. The data is annotated using Industry 4.0 modelling standards in order to make it useful for downstream operations. By utilising the loosely coupled gateway we suggest, they sought to assist in making It is possible to make existing installations Industry 4.0 ready without explicitly re-engineering them.

In[10], The authors provided an event-driven real-time air quality inspection (EAQI) framework that can be reused to help create distributed sensor systems for gathering and evaluating pollution sensor measurements. The architecture is based on event streams, which store immutable raw data as events and provide both the most recent values and an entity history. This framework is made to be flexible, scalable, and simple to integrate with data analysis techniques.

3. Proposed System

We described the entire procedure for creating the architecture in current chapter.

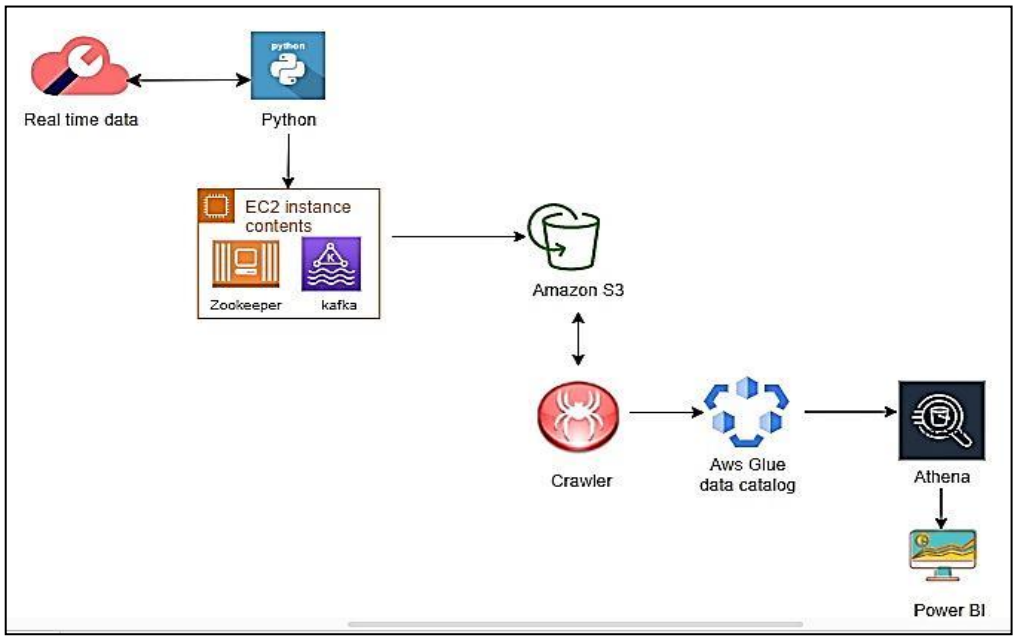

Fig 1: Data flow diagram

- the Kafka framework is setup on AWS EC2 occurrence, to set up the Zookeeper server and Kafka broker.
- Setup Python API and transmit the Kafka server's (Producer) real-time stock data.
- Using the Python API (Consumer), the data moved from the Kafka server is accumulated in an AWS S3 bucket.
- To set up AWS crawler to scan stock data from S3 bucket, which creates R schema automatically and stores data in AWS Glue data catalogue.
- To create views on the AWS Glue data catalogue for AWS Athena stock market data analysis.
- Acquire the information from the AWS Athena outputs and moved it as R data so that Power BI can display it [16].

Let PR stand for producers, CO for consumers, and KT stand for the set of Kafka topics.

The following expression can be used to explain Kafka's fundamental principles:

KT=PR+CO (1)

where KT is the cluster of every topic, PR is the cluster of every producer who transmits messages to subjects, and CO is the cluster of every consumer who receives messages from topics.

The basic connection among producers, topics, and consumers in Kafka is depicted by this equation. Topics that producers have written are conveyed to consumers via messages. The collection of messages created and absorbed by message producers and message consumers is indeed the collection of topics

The expression can also be expanded to illustrate Kafka's use of partitioning:

KT=PR+CO + PR(x) CO(y) (2)

where PR(x) denotes the grouping of x producers' partitions and CO(y) denotes the grouping of y consumers' partitions. This equation illustrates the idea of partitioning in Kafka, where each topic is split up into a number of partitions, all of which are consumed by a certain audience.Ultimately, expression (1) depicts Kafka's fundamental functionality, whereas expression (2) illustrates the idea of partitioning in Kafka. These equations collectively give a mathematical illustration of the main components of the Kafka's architecture. Assume DS1 is the collection of data resources (S3 Bucket), D(c) is the collection of data that a crawler has extracted (Data Catalog), and CR is the collection of AWS crawlers.

An AWS Crawler's fundamental functioning can be described by the expression:

D(c)= DS1 x CR (3)

Where DS is the cluster of all crawled data sources, D(c) is the cluster of all data that has been retrieved from Data sources, and CR is the cluster of all AWS Crawlers that have been used to obtain the information. The basic connectivity among data sources and AWS Crawlers is represented by this expression.

The expression can also be expanded to illustrate the idea of categorising and sorting data:

$$DS2= D(c) ((F(X)) \quad (4)$$

Where X is the count of searches and F(X) is a collection of queries applied to the data (Athena). D(c) is the AWS glue Data Catalog, which offers data for use in queries. The query results are kept in DS2 and have the DS2 id of an AWS S3 bucket (2).

Thus, expression (3) shows the fundamental functionality of AWS Crawler, and expression (4) illustrates the idea of data filtering and categorization. Combined, these expressions offer a mathematical illustration of the structure and salient characteristics of the AWS Crawler.

The following mathematical equation can be used to visualise AWS Athena results:

$$VZ (A)=f(DA,DT) \quad (5)$$

Where:

VZ= Visualization

A = AWS Athena Results

DA = PowerBI tool (Data Analysis)

DT = Data Transformation tool (e.g. SQL, Python)

The function f(DA, DT) covers the steps involved in transforming and analysing the AWS Athena results A and producing a visual representation (VZ) of the data utilising data analysis technique DA and the data transformation method DT.This expression emphasises the significance of using data analysis and data transformation tools to efficiently visualise the findings returned by AWS Athena requests

4. Implementation

4.1 Setup Kafka framework through AWS service

Kafka:

Real-time stream processing is the focus of the distributed data storage system Kafka. The majority of the sources that continously produce broadcasting data send registrations of information to each other. A streaming site that can process this constant intake of data in a systematic and progressive manner is necessary. To give consumers the key benefits of each messaging technology, Kafka combines publish-subscribe and queuing. Because it allows for the spread of data handling among several consumer instances, queuing is particularly flexible. Reusability in Kafka's model enables several distinct implementations to receive from data streams and operate independently at their respective speed.

Amazon Elastic Compute Cloud (EC2):

The AWS cloud’s Amazon EC2 provides scalable processing capability. Because there is no initial hardware expenditure required when using Amazon EC2, programs can be developed and released more quickly. Utilizing Amazon EC2, we deploy quite so many or as few virtualized resources as we need, setup of security and networking settings, and manage RAM. With Amazon EC2, we can scale up or down to handle changes in demand or popularity spikes, which eliminates the need to forecast traffic.

This module creates instances using the AWS EC2 service, connects Kafka via SSH, configures the JDK, and runs Zookeeper and Kafka servers.The process of setting up an AWS EC2 instance to load and retrieve Kafka is depicted in Fig. 2's flow diagram. Launch the servers for Kafka and Zookeeper in this module.

Steps involved in this module are:

1. Creating an EC2 instance in AWS Cloud
2. Connect to EC2 Instance
3. Install Java
4. Installing Kafka on an AWS EC2 Instance
5. File Directory of Kafka
6. Start running Zookeeper
7. Running Kafka Broker

Your EC2 instance of Kafka has now been successfully launched. By making the necessary adjustments to the security groups, you can use the Kafka-Python API or Kafka-Java API to browse your Kafka-server

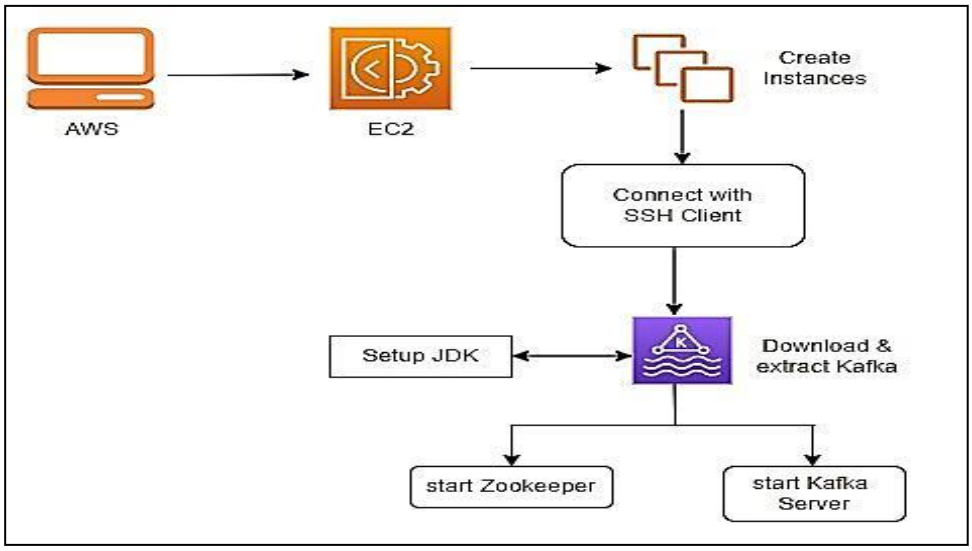

Fig. 2: Flow diagram to setup kafka framework

A. Data produced to Kafka node Python API (Producer)

Kafka node receives real-time data source and does monitoring.

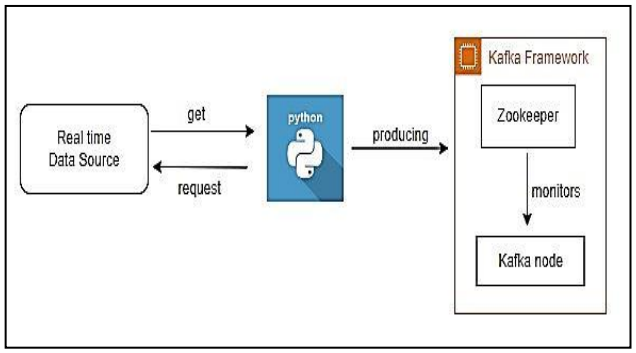

Fig.3: Flow diagram for producer

Figure 3 shows the data that how this module's Python API is used to produce and send to the Kafka node. The kafka node receives the real-time stream data production.

The following steps are involved in this module:

1. Python libraries
2. Data produced to kafka node

Algorithm 1: Producer

Function: p(ev, Tp) -> K(Tp, A)

begin

//p - Producer

//ev - event

//Tp – Topic

//K – Kafka Broker

//A – Acknowledgement

if Tp.append(ev) then

A(ad, ap)

else:

A(ad, rj)

A(ad, ap, rj)->Zk(0|1)

//Z: Zookeeper

// ad: acknowledgement data

// ap/rj: approve/reject

//Ack returns 0 if record is successfully written on

//Topic otherwise returns 1 to the Zookeeper node.

end

B. Data consumed to AWS S3 from Kafka node through Python API (Consumer)

The data is consumed in AWS S3 buckets that are being generated from Kafka node through Python API

Figure 4 shows the data that this module's Python APIs from the Kafka node ingested and sent to AWS S3. The information available from the Zookeeper server.

Various steps involved in performing this module involves:

1. Create bucket in S3
2. Data consumed to S3

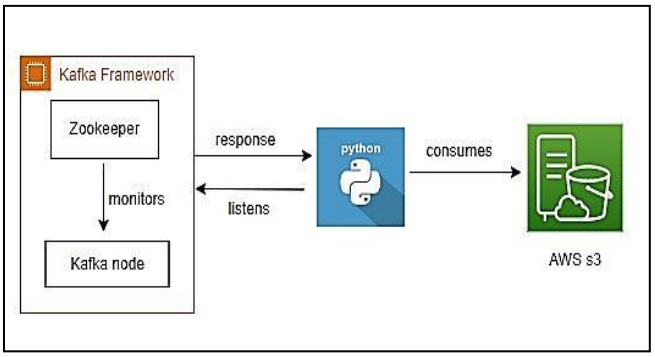

Fig.5: Flow diagram for consumer

Algorithm 2: Consumer

Function: K(Tp, A) -> c(Tp, rd)

begin

//K – Kafka Broker

//Tp - Topic

//A – Acknowledgement

//rd – record

//c - Consumer

if c(Tp, rd) then

A(ad, ap)

else:

A(ad, rj)

Ack(ad, ap, rj)->ZK(0|1)

// ZK: Zookeeper

// ad: acknowledgement data

// ap/rj: approve/reject

//Ack returns 0 if record is successfully written on

//Topic otherwise returns 1 to the Zookeeper node.

end

.

C. Data analysis through AWS Crawler, Glue data catalog and Athena

The AWS crawler examines the data in the Amazon S3 bucket and automatically generates a schema. AWS Athena is then used to run queries on the data and generate results

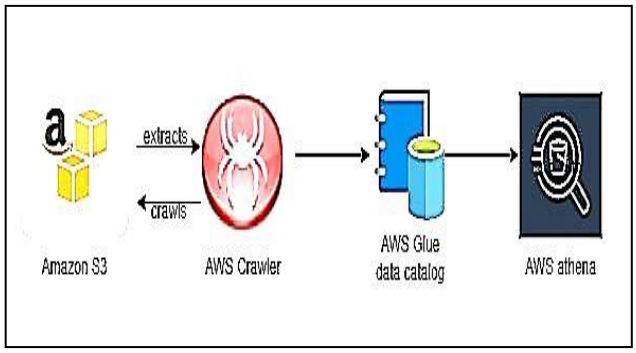

Fig.5: Flow diagram to analyze the data

The following steps are included in this module:

1. Create bucket in S3

E. Comparing results and analyse the visualizations in Power BI dashboard

After being acquired in power BI, the AWS Athena findings are again shown.

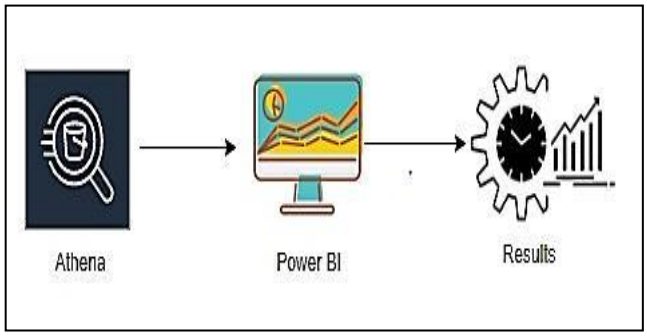

Fig.6: Flow diagram to compare results

The flow diagram shown in Fig.5 is related to analyze the straming data through AWS Crawler, Glue statistics record and AWS Athena.The flow diagram in Fig. 6 relates to case studies on stock market data in Power BI desktop and compares the outcomes and evaluates the various displays. To view the output of Athena, the data will be first converted in a.CSV file and integrated into PowerBI.

Power BI Desktop:

Power BI desktop is the primary authoring and publishing tool for Power BI. Advanced users and programmers can use it to create brand-new models and reports from scratch. Costs: Null.

Power BI service

Hosting for Power Bl's data models, reports, and dashboards is provided through an online (SaaS). In the cloud, management, exchange, and interaction take place.Steps involved in this module are:

1. Exporting results from Athena result to Power BI

2. Analyze visualizations

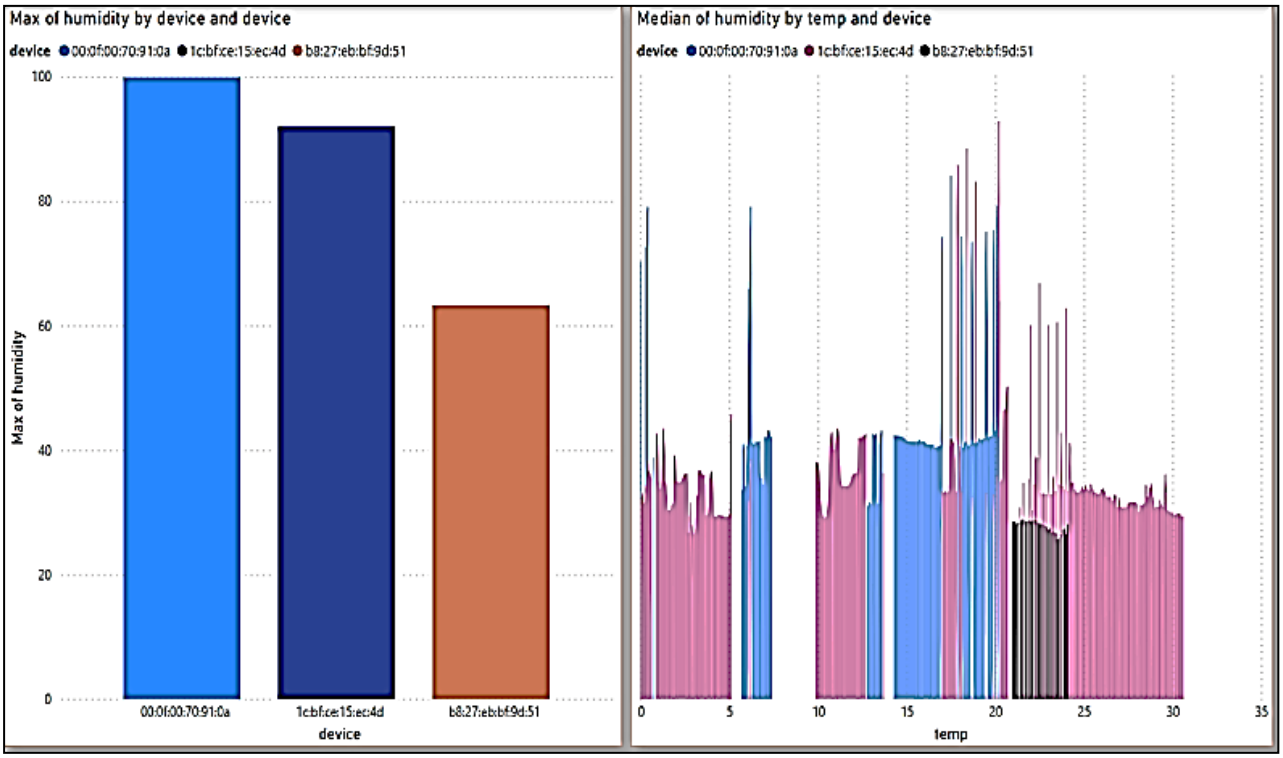

Fig.7: Stacked & Ribbon charts to obtain maximum humidity

Figure 7 displays a "stacked column chart" and a "ribbon chart," respectively. The Stacked Column Chart displays the highest degree of humidity measured by several Sensor Devices. The temperature and humidity fluctuations of several sensor devices are shown on the ribbon chart

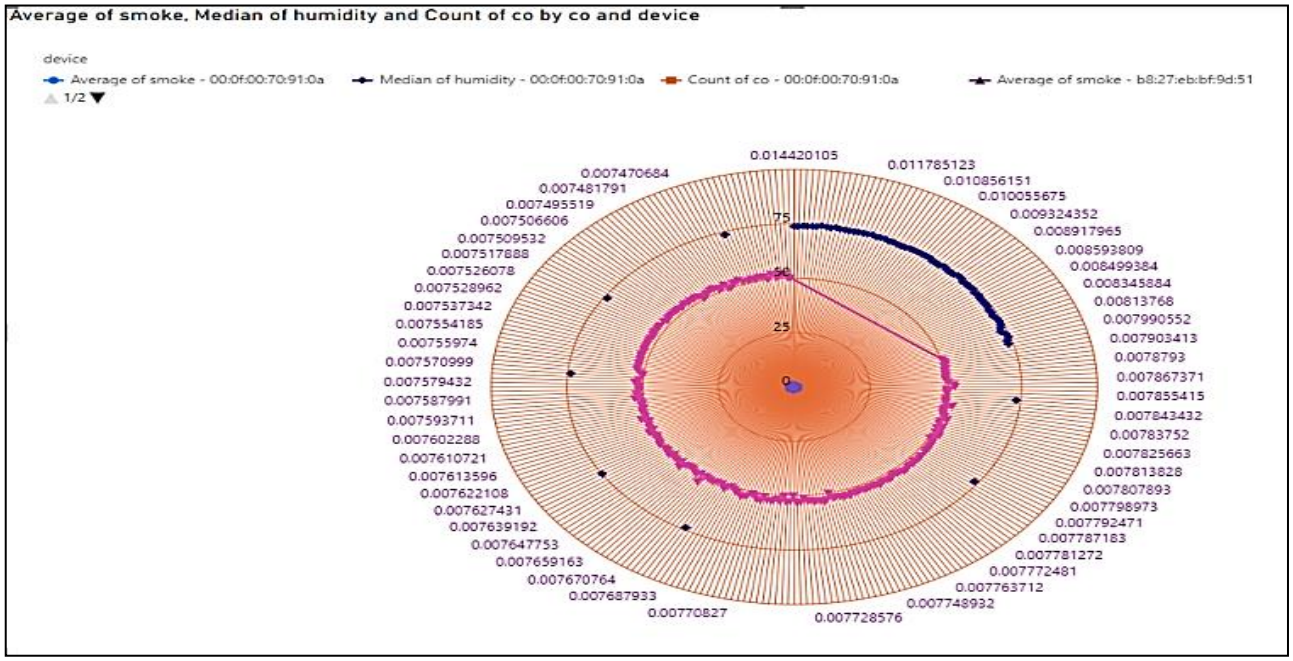

Fig.8: Radar chart to compare multiple variables

Figure 8 displays a radar graphic that contrasts many parameters, including CO, humidity, and smoke, depending on various sensor devices

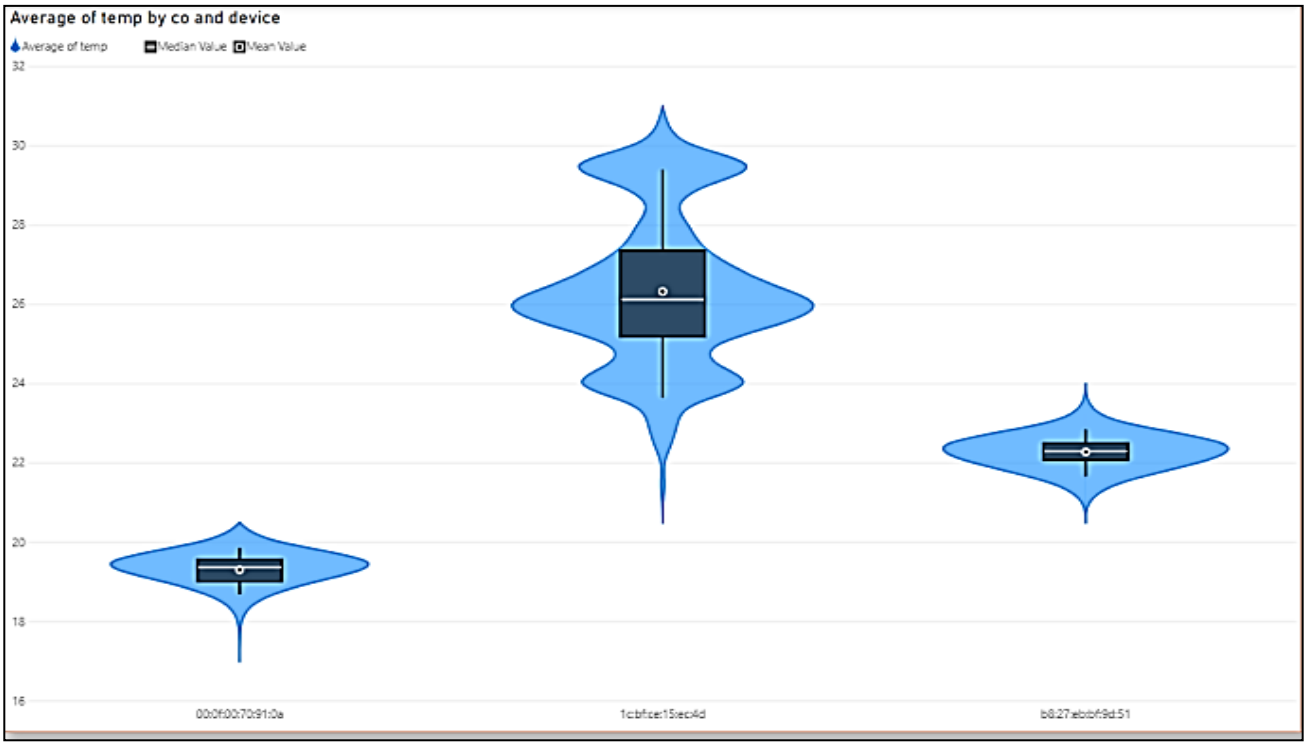

Fig.9: Violin Plot related to statistical analysis

Figure 9 illustrates the "Violin Plot," which illustrates how the temperature of various Sensor devices is statistically analysed by filtering the CO data

5. Conclusions

Big data analytics will improve weather prediction outcomes and assist experts in making more precise weather predictions. In general, firms that wish to stay competitive in today's fast-paced, data-driven world must implement real-time data analytics.In this project, we showed how the Kafka framework can be used for real-time analysis, but in the years ahead, we'll concentrate on using Kafka for machine learning projects' training and prediction

References

[1] Y. -C. Tsao, Y. T. Tsai, Y. -W. Kuo and C. Hwang, "An Implementation of IoT-Based Weather Monitoring System," 2019 IEEE International Conferences on Ubiquitous Computing & Communications (IUCC) and Data Science and Computational Intelligence (DSCI) and Smart Computing, Networking and Services (SmartCNS), 2019, pp. 648-652, doi: 10.1109/IUCC/DSCI/SmartCNS.2019.00135.J.Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] Suciu, C. Beceanu, M. Anwar, C. M. Balaceanu and M. Dobrea, "Weather Monitoring for Predicting Thermal Comfort and Energy Efficiency," 2018 IEEE 24th International Symposium for Design and Technology in Electronic Packaging (SIITME), 2018, pp. 156-160, doi: 10.1109/SIITME.2018.8599260.K. Elissa, “Title of paper if known,” unpublished.

[3] Mohapatra and B. Subudhi, "Development of a Cost-Effective IoT-Based Weather Monitoring System," in IEEE Consumer Electronics Magazine, vol. 11, no. 5, pp. 81-86, 1 Sept. 2022, doi: 10.1109/MCE.2021.3136833.

A. Suleykin and P. Panfilov, "Distributed Big Data Driven Framework for Cellular Network Monitoring Data," 2019 24th Conference of Open Innovations Association (FRUCT), 2019, pp. 430-436, doi: 10.23919/FRUCT.2019.8711912.

[4] R. Wiska, N. Habibie, A. Wibisono, W. S. Nugroho and P. Mursanto, "Big sensor-generated data streaming using Kafka and Impala for data storage in Wireless Sensor Network for CO2 monitoring," 2016 International Workshop on Big Data and Information Security (IWBIS), 2016, pp. 97-102, doi: 10.1109/IWBIS.2016.7872896.

[5] N. MAHESH, & A. VYSHNAVI. (2014). A Novel Control Strategy of PV making System with LPC for Loading Balance of Distribution Feeders. *International Journal of Computer Engineering in Research Trends*, *1*(6), 570–574.

[6] M. D'silva, A. Khan, Gaurav and S. Bari, "Real-time processing of IoT events with historic data using Apache Kafka and Apache Spark with dashing framework," 2017 2nd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology

(RTEICT), 2017, pp. 1804-1809, doi: 10.1109/RTEICT.2017.8256910.

[7] A. Sgambelluri, A. Pacini, F. Paolucci, P. Castoldi and L. Valcarenghi, "Reliable and scalable Kafka-based framework for optical network telemetry," in Journal of Optical Communications and Networking, vol. 13, no. 10, pp. E42-E52, October 2021, doi: 10.1364/JOCN.424639.

[8] N. Braunisch, S. Schlesinger and R. Lehmann, "Adaptive Industrial IoT gateway using kafka streaming platform," 2022 IEEE 20th International Conference on Industrial Informatics (INDIN), 2022, pp. 600-605, doi: 10.1109/INDIN51773.2022.9976153.

[9] S. Winberg and S. Singh, "Real-Time Event-driven Air Quality Inspection Framework for City-wide Pollution Level Monitoring," 2021 International Conference on Electrical, Communication, and Computer Engineering (ICECCE), 2021, pp. 1-6, doi: 10.1109/ICECCE52056.2021.9514133.

[10] Shweta P. Patil, Jyothi Digge, & Mahesh Kadam. (2020). Performance Analysis of Free Space Optical Communication Network with Different Modulation Techniques. *International Journal of Computer Engineering in Research Trends*, *7*(10), 1–5.

[11] Neha Narkhede, Gwen Shapira, and Todd Palino. 2017. Kafka: The Definitive Guide Real-Time Data and Stream Processing at Scale (1st.ed.).O'ReillyMedia, Inc.

[12] Y. Drohobytskiy, V. Brevus and Y. Skorenkyy, "Spark Structured Streaming: Customizing Kafka Stream Processing," 2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP), 2020, pp. 296-299, doi: 10.1109/DSMP47368.2020.9204304.

[13] K. Kato, A. Takefusa, H. Nakada and M. Oguchi, "A Study of a Scalable Distributed Stream Processing Infrastructure Using Ray and Apache Kafka," 2018 IEEE International Conference on Big Data (Big Data), 2018, pp. 5351-5353, doi: 10.1109/BigData.2018.8622415.

[14] Suryakant Acharekar, Prashant Dawnade, Binay Kumar Dubey, & Prof. Prabhakar Mhadse. (2020). IoT Based Weather Monitoring System. *International Journal of Computer Engineering in Research Trends*, *7*(4), 20–22.

[15] B. D. D. Nayomi, D. W. Albert, V. Sujatha, K. K. Baseer and M. J. Pasha, "A Framework for Processing and Analysing Real-Time data in e-Commerce Applications," 2023 8th International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2023, pp. 1495-1500, doi: 10.1109/ICCES57224.2023.10192771.

[16] Raj, R., & Sahoo, D. S. S. . (2021). Detection of Botnet Using Deep Learning Architecture Using Chrome 23 Pattern with IOT. Research Journal of Computer Systems and Engineering, 2(2), 38:44. Retrieved from https://technicaljournals.org/RJCSE/index.php/journal/article/view/31

[17] Prasanthi, T. S. ., Rayavarapu, S. M. ., Lavanya, Y. L. ., Kumar, G. S. ., Rao, G. S. ., Goswami, R. K. ., & Yegireddy, N. K. . (2023). Performance Analysis of Different Applications of Image Inpainting Based on Exemplar Technique. International Journal on Recent and Innovation Trends in Computing and Communication, 11(4), 113–117. https://doi.org/10.17762/ijritcc.v11i4.6393

[18] Rohokale, M. S., Dhabliya, D., Sathish, T., Vijayan, V., & Senthilkumar, N. (2021). A novel two-step co-precipitation approach of CuS/NiMn2O4 heterostructured nanocatalyst for enhanced visible light driven photocatalytic activity via efficient photo-induced charge separation properties. Physica B: Condensed Matter, 610 doi:10.1016/j.physb.2021.412902