**Analyzing and Visualizing Cricket Streaming Data Using Kafka**

**1. Introduction: (Nikitha Kethireddy)**

In recent years, advances in hardware technology have facilitated the ability to collect the data continuously and data streaming is one such process. This process is used to perform real-time analysis on streaming data and it allows users to access the content immediately over the internet rather than having them wait for the data to be downloaded or storing the data first. Data streaming is becoming more useful and necessary in today's world and is being applied in a broad range of industries such as medical, transportation, finance, real estate, gaming, and so on.

In our project, we are fetching the data from an API in the JSON format and we are serializing the data into byte format, we will be streaming that serialized data into Kafka and later on, we are again deserializing the byte data into JSON format. Once the unstructured data is obtained in the JSON format, we are storing that unstructured data into MongoDB to retain its structure and then we will be using ODBC connector to fetch the data from MongoDB to PowerBI for performing the data visualization. Apart from that, we are also performing commentary sentiment analysis where we are using the NLTK package and generating emoji for the scoring commentary.

**2. Requirements: (Nikitha Kethireddy)**

There are some requirements which our project is implementing.

* Our project will be able to fetch the data from an API in the JSON format
* Our project will be able to stream the live data using Kafka
* Our project will be able to store the streaming data into MongoDB
* Our project will be able to visualize the live data in PowerBI
* Our project will be able to analyze the positive, negative, and neutral score commentary by providing certain emoji depending upon the comments.

**3. Tools and technologies used: (Nikitha Kethireddy)**

The following are the main tools and technologies which we have used in our project.

* JAVA and Python programming are used for streaming the data into Kafka and also for analyzing the scoring commentary
* Apache Kafka is used for streaming the live data in our project
* Jupyter Notebook, IntelliJ is the integrated development environment which we are using for developing our application since they provide us with many advantages and flexibility.
* MongoDB is the database that we have chosen for storing the JSON data and it is helping us to retain the structure of the database.
* ODBC data connector is used to access the data which is present in MongoDB and helps us to connect to PowerBI
* We are using PowerBI as a visualization tool for visualizing our data in the form of graphs, tables.

**4. Data source (Deepthi Chokka)**

For our project, our main source of data is an Indian cricket news website called Cricbuzz. This is a Sports website exclusively for Cricket. The link to this website is as follows: [www.cricbuzz.com](http://www.cricbuzz.com)

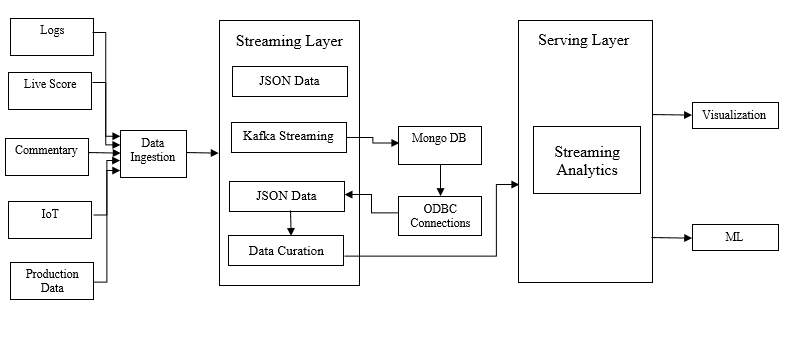
**5. Data (Deepthi Chokka)**

This data consists of various attributes which are commentary, name of team1, name of team2, for every team and wickets, overs bowled, score, maiden, wide, no balls for the bowling team, and team score, wickets, overs, ID for the batting team. The data will be extracted in the JSON file format. For our project, will be using the following attributes:

* **ID:** Uniquely generated number to distinguish cricket teams
* **Bat\_team\_innings\_score:** Score of the bat team innings
* **Bat\_team\_innings\_overs:** Number of overs played by the batting team
* **Bow\_team\_innings\_score:** Score when the bowling team is bowling
* **Bow\_team\_innings\_overs:** Number of overs played by the bowling team
* **Name of Team1:** Name of one team
* **Name of Team2:** Name of the other team
* **Commentary:** Commentary for every overplayed

**6. Data extraction**

**(Block Diagram) (Deepthi Chokka)**



1. **Data Extraction (Raghunandan Naishadam)**

For extracting the data from Cricbuzz we will make use of the CricBuzzParser package using which we will create an instance for CricBuzzParser and retrieve the data we required from that instance.

* **Fetching Scorecard from Cricbuzz**

We have a package called CricBuzzParser in Java which allows us to retrieve the scorecard for an ongoing (live) cricket match.

In the CricBuzzParser package, we have a method called Parse () that returns the scorecard of an ongoing match. The response is an array of scorecard objects that has batsmen and bowler scores.

Please use the below URL to download the CricBuzzParser package for java.

* <https://github.com/yashrs/Cricbuzz-API-Java>

Below are the objects that will be returned by Parse () method which will have a live scorecard.

* **Venue:** It contains venue details such as stadium time, match hours
* **Team\_1:** Team 1 name
* **Team\_2:** Team 2 name
* **Bowler:** number of runs given, wide balls, name of the bowler
  + **batsman:** number of runs and balls, name of the batsmen
* **Pseudocode for fetching the scorecard**

While always true {

* Create CricBuzzParser object with Cricbuzz API link as input
* Initialize CricBuzzParser object with RetreiveURL() method without any input arguments.
* For that particular CricBuzzParser object retrieve contents of URL using Parse() method and store it as a String in one variable
* Initialize the custom object class with the above retrieve data as arguments.
* Initialize the Kafka producer with the topic name and custom object reference }

1. **Data streaming using Kafka (Raghunandan Naishadam)**

Kafka is a distributing streaming platform, in which we have Topics and Partitions. To understand this, we need to know about Topic and Partition”. The intuition behind Topic is a place where data gets stored and partition is something like dividing each topic into one or more partitions. Below steps that need to be followed to stream the data through Kafka into Mongo DB which will be used for visualization.

* **Kafka Cluster Setup:**

1. Download Kafka from their official website and place it on the Computer hard drive
2. Open command prompt in the path where Kafka is downloaded i.e., in computer hard drive
3. **Start zookeeper:** Open command prompt and run below command

.\bin\windows\zookeeper-server-start.bat .\config\zookeeper.properties

1. **Start Kafka:** Open another command prompt, run the below command and zookeeper should be running

.\bin\windows\kafka-server-start.bat .\config\server.properties

* **MongoDB Setup**

1. Download MongoDB from their official website and place it on the Computer hard drive
2. Run Mongod.exe file that is present in the bin folder of MongoDB file

3. Start zookeeper: Open command prompt and run below command .\bin\windows\zookeeper-server-start.bat .\config\zookeeper.properties

4. Start Kafka: Open another command prompt, run the below command and zookeeper should be running

.\bin\windows\kafka-server-start.bat .\config\server.properties

* **Creating a new topic in Kafka:**

Create a new topic named *livescore* using the below command using command prompt.

.\bin\windows\kafka-topic.bat –zookeeper localhost: 2181 --topic livescore –-create –-partition 1 –replication-factor 1

* **Creating a new Collection in MongoDB:**

1. Create a new Document named *MyDb.*
2. Create a new Collection inside the MyDb document name *sampleCollection*

* **Run the Producer in IntelliJ**

To create a Kafka producer client application, we need to create a configuration property.

1. Bootstrap servers. – Provides the Kafka servers details
2. Key-Value Serializers – data has to be binary encoded, so we are using ByteArraySerializer

Below is a Pseudocode for Kafka producer.

Main () {

* Initialize FileInputStream object;
* Initialize Properties object;
* Create a CustomObject with required parameters;
* Create Serializer and implement Serializer<CustomObject>;
* Configure these properties Bootstrap.Servers, key & value serializers to Properties;
* Initialize KafkaProducer with Properties;
* Specify topic Name;

While always true {

* Create an Instance of the CricbuzzParser object with API as input;
* Retrieve live score from that object;
* Create an instance of Custom Object with the live score as input;
* Initialize ProducerRecord with key as topic name and value as CustomObject;
* Send the above record;
* Send messages to from ProducerRecord object;
* Use the thread to delay time;

}}

* **Run the Consumer in IntelliJ**

Here we are using the consumer to send the data into MongoDB. To create a Kafka consumer client application, we need to create a configuration property.

* + 1. Bootstrap servers. – Provides the Kafka servers details
    2. Key-Value Serializers – data has to be binary encoded, so we are using ByteArraySerializer

Below is a Pseudocode for Kafka Consumer.

Main (){

* Initialize FileInputStream object;
* Initialize Properties object;
* Initialize MongoDB client object and fetch the collection
* Create Deserializer and implement Deserializer<CustomObject>;
* Configure these properties Bootstrap.Servers, key & value serializers to Properties;
* Initialize KafkaConsumer with Properties;
* Specify topic Name;

While always true {

* Retrieve live score from that Deserializer object;
* Push the live score into MongoDB
* Use the thread to delay time;}}

**8. ODBC Connection (MongoDB to PowerBI) (Nikitha Kethireddy)**

After getting the unstructured data in JSON format and storing it in the database, the following steps need to be performed to fetch the data into POWER BI

* We need to select the ODBC Administrator option (32 or 64 bit depending upon the system)
* After selecting ODBC data source administrator, a pop up will be opened where we need to select System DSN and choose the option called ADD
* When we click the ADD button, a pop will be opened where we need to select the option called Data Direct 8.0 MongoDB and select the option called FINISH
* Once we click finish, a pop will be opened and we need to fill all the details depending upon our requirement and we need to choose the option called TEST CONNECT and a popup will be opened again.
* Once the pop up is opened we need to enter the user name and password related to our database
* Once we provide our credentials, we need to press the OK option, if the credentials and database details are correct, we will get an alert that a connection is established, we need to click OK button.
* Once the connection is established, we need to make sure to click on APPLY and OK buttons in order add the DSN to our System DSN
* Once everything is done, the DSN which we added is displayed in the System DSN of the ODBC Data Source Administrator.
* When we are done, we can now open PowerBI desktop and choose the option called GETDATA in order to get the data from MongoDB.
* When we choose the option GETDATA, different options are displayed and we need to select the option OTHER and ODBC and we need to press CONNECT button
* Once we click on CONNECT button, an alert dialog is displayed where we need to enter our username and password of MongoDB and click on CONNECT option
* Once we connect, another pop up will be displayed where we need to select the Data source name which we created in the early steps and we need to click OK
* Once we are connected, the tables present in the MongoDB are displayed and we need to press the option called LOAD in order to LOAD the data.

**9. Data Curation (Jeevan Reddy Mure)**

We used power bi for cleaning and transforming data. Imported data from MongoDB using ODBC connectors is loaded into power bi and data is in the format of JSON. We used power query which has incredible features help us clean and analyze data. There is lots of data of different scorecard details fetched and we only need the data of particular columns so we performed group by query in retrieving the particular data using unique id and the output will be scorecard of individual teams and we only need overs, wickets and team score which can be achieved by performing group by query in power query and used same technique in retrieving the current score of individual team.

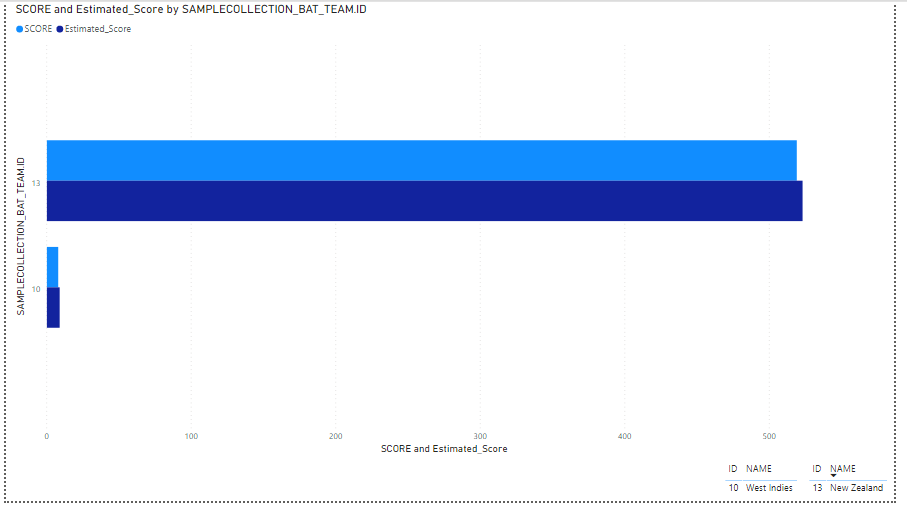
Imported data from mongo dB is in the form of JSON format which contains records and nested data. In ordered to use the achieved data for visualization we converted the records into tables and expanded the nested the data. We used power query in simplifying the data and changing data types. The data which is in the form of decimal is changed to whole numbers. We profiled the columns into valuable data for deeper analysis. The transformed data will be in the form of tables which can be used for further process.

**10. Data visualization (Jeevan Reddy Mure)**

Data visualization is the final step of our project. We used power bi for visualizing the live data which is transformed and stored in tables. The transformed data consists of team id, score and overs.

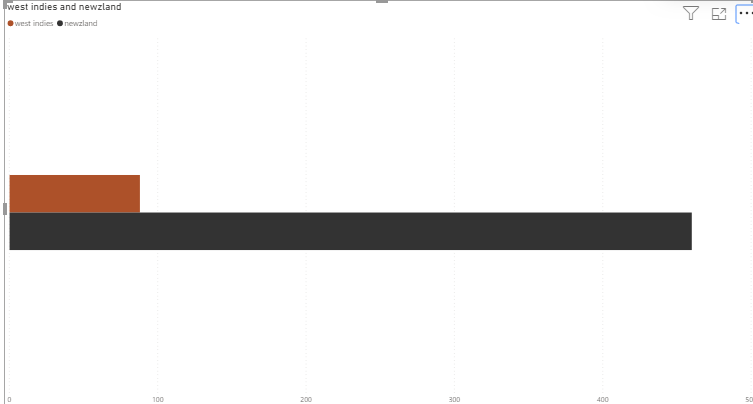
Compare the current score and estimate the score for the next over?

We created a custom column to calculate the estimated score for the next based on current run rate. The custom column created will use current score and overs to calculate the required estimated score. Estimated score is the estimation of score for the next over. Estimated score = [Score] ([current score]/[No of overs bowled]).The estimated will change every time once the current score is updated.



Above is the bar chart compares the current score and estimated score. Light blue color bar indicates the current score and dark blue indicates the estimated score. As you can see West Indies score three runs in their first over of innings and estimated score is six runs by next over. When it comes to score rate the current run rate will be the run rate for the estimated score. The teams are indicated by their id and legend indicates the team names for the id.

Compare the scores of both Teams



Above bar chart compares the team score of both West Indies and New Zealand the brown bar indicates the current score of West Indies and black bar is indicating the team score of New Zealand. The bar will get to change once score updates.

# **11. Prediction Analysis using NLTK (Dheeraj Edupaganti)**

NLTK stands for Natural Language Toolkit, which is used to perform sentiment analysis in python.

Sentiment analysis is normally applied on a writing or a sentence to determine whether it is positive, negative or neutral. It is performed to know the feeling of the people towards the particular product.

VADER-lexicon is a tool to perform sentiment analysis for all the sentiments expressed in social media. It labels the sentiment as positive or negative based on their respective scores.

Detection of polarity lies with the text, paragraph, clause or sentence.

At times, it is difficult to perform sentiment analysis on a sentence or clause which holds multiple sentiments i.e., the sentence consists of both polarities (positive and negative)

VADER lexicon sentiment analysis is performed by summing up the intensity of each word in a text or sentence. So as per the summing information the polarity scores will be generated and hence positive or negative scores can be determined. Polarity is mainly classified based on expression of text or sentence i.e., positive, negative or neutral. The overall opinion of a sentence or text can be determined by aggregating all the sentiments

To use VADER lexicon, first it need to be imported form nltk and download

VADER Sentiment Intensity Analyzer takes a string as input and returns scores based on different categories like negative, positive, neutral and compound.

As a part of Sentiment Analysis, Json data is fetched from Pycricbuzz package using json.dumps and then load the data into data frame. Using NLTK package positive or negative sentiment analysis is performed on the commentary generated in the cricbuzz. Based on the polarities respective emoji is assigned to the commentary comment.

Any data related to cricket can be fetched from Pycricbuzz package, like matches, commentary, live score, scorecard, id etc. Here commentary is fetched based on id of the respective match i.e., commentary (‘id’)

**Pseudo code: (Dheeraj Edupaganti)**

The VADER lexicon sentiment analysis is performed based on the lexicon dictionary. The dictionary consists of around 7500 sentiment features. The polarity varies from -4 to +4, which are extremely negative and positive scores respectively

Define a function comment ()

Initialize cricbuzz method to variable c

Assign commentary with id ‘30549’ to variable ‘comm’

Dump and load the json data into a variable ‘jdata’

Using pandas convert the ‘jdata’ to data frame

Print the data frame

For every commentary in a data frame of commentary

Clean the data using regular expression

If polarity score of ‘pos’ is greater than polarity score of ‘neg’

Then print grinning face with big eyes emoji and thumbs up emoji with commentary and over information

Else if polarity score of ‘pos’ is less than polarity score of ‘neg’

Then print thumbs down emoji with commentary and over information

Else if polarity score of ‘pos’ is equal to polarity score of ‘neg’

Then print normal face emoji with commentary and over information

Schedule to run the comment () function for every 5 minutes

While the condition is true

Schedule to run the pending jobs

Sleep for 1 second