Alerts Automatic Triage for CMS web logs monitoring system

F. Legger[1], V. Kuznetsov[1], C.Ariza[1], Y. Sunthornyotin[2]

On behalf of the CMS Monit team

Introduction

The CMS experiment depends on several services that should be monitored on near real time. These systems generate logs from which alerts should be triggered, however, trigger alerts for single events will give an overwhelming number of false positives. The objective of this project is to develop a tool that detects patterns on streams of data, process complex events in order to trigger timely and relevant alerts, and automatically triage issues sending notifications to the proper channel.

Project Overview

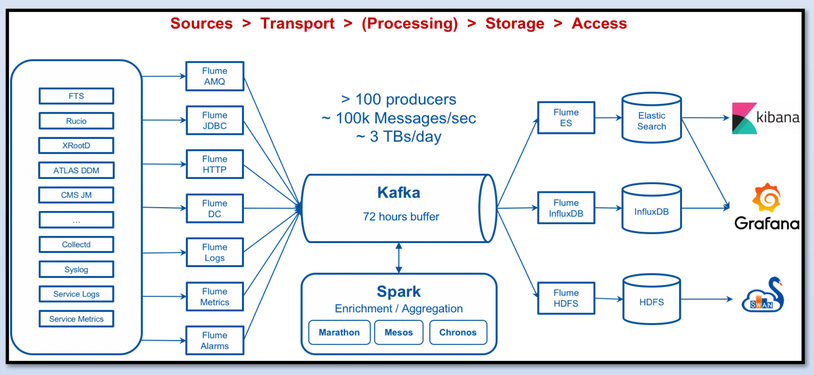

Figure 1:

Proposed System Methodology

- Firstly, all the logs is consumed via Kafka topic. The parameters will be unwrapped and grouped together in specific time window frame. After resolving the number of request per system, the DataFrame is collected in CERN's HDFS.
- Secondly, the collected DataFrame will be consumed again to do complex aggregation in order to find a rolling average of each factor consists of number of user, request, system call, and API call.
- Once the result of rolling average or percentage difference has been retrieved, combine it as machine learning features along with date and the categorical data, for example, the system, user, and API name.
- In the next step, the KMeans model, one of unsupervised machine learning classification method, is created according to the input features and k value or number of cluster determined by Elbow method. Now, the predicted result will be labeled in number 1 to k.
- Finally, by finding the distance from each data point in that particular cluster to the center of its cluster based on Euclidean distance, the anomaly can be found and published to end user via email.

CMS Weblog Parameters

The Kafka topic: 'cmsweb_logs' consumed parameter :

- System name
- User name
- API name
- Kafka Timestamp

'Kafka Timestamp' is used for grouping window timeframe in real-time. The time interval for data aggregation can be changed depends on user preferences.
By consuming message via Kafka and collecting it in HDFS, the collected data is used for observing the trend. Aggregated results from multiple parameter in the CMS Weblog Kafka topic are obtained.
The main aggregated value is %Difference of the amount of system calls, API calls, and user comparing to average amount in the same window frame. The trend can be observed by data visualization which indicates that there're both pattern and non-pattern trend mixed together.

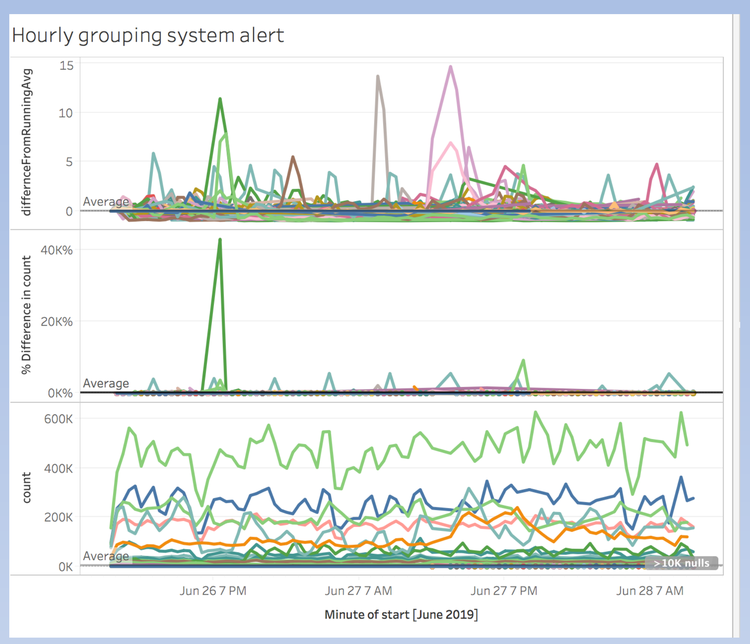

Figure 2

Tools for developed system

- Spark Structured Streaming
- HDFS (Hadoop Distributed File System)
- Spark MLLib: Kmeans / One hot encoder
- Apache Kafka
- Email Notifier
- SWAN

System Architecture

Sequential Diagram

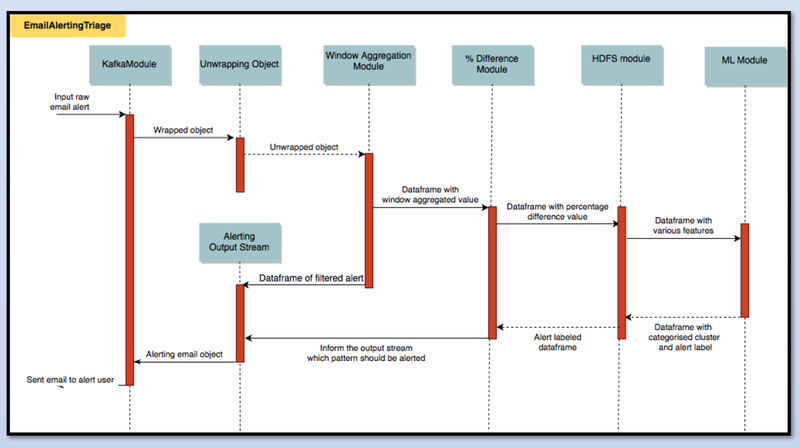

Figure 3

Component Diagram

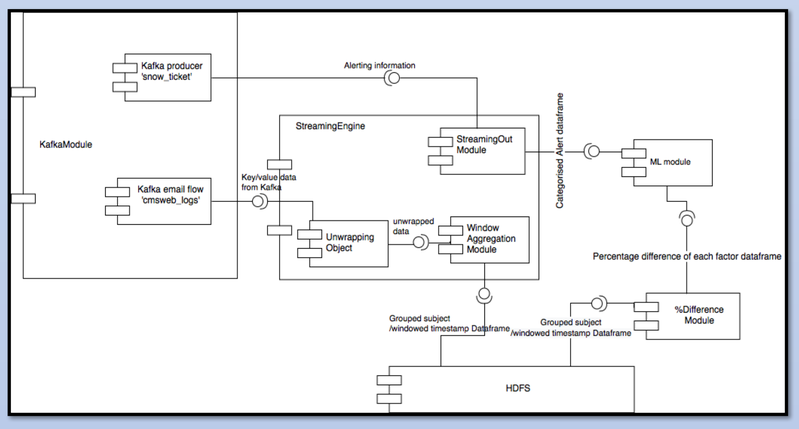

Figure 4

Data Preparation

Since user, system and API name are considered as a categorical data which should be include inside the model, the process of transforming **categorical data** into **numerical data** must be done. In this scenario, **One hot encoder** methodology is selected for implementation.

The features are including:

- System name
- User name
- API name
- Total amount of request per user per system in window time interval
- %Difference of request compare to window average
- %Difference of system call compare to window average
- %Difference of API call compare to window average
- %Difference of amount of user compare to window average
- Weekday/Weekend/Month begin/Month end
- Hour/Minute/Day/Month/Year

Unsupervised Learning: KMeans

By spliting the data into training and testing set, the model is created based on the K value or number of cluster derived from Elbow method. The spark ML model is compatible with spark Dataframe but not yet support in streaming Dataframe.

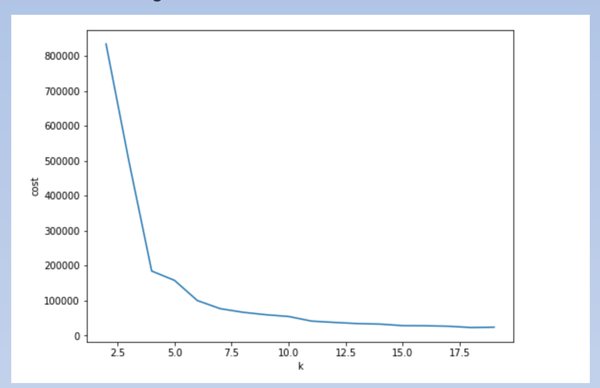

Figure 5

Outlier benchmark

After obtain prediction results given by Kmeans model, each row will be labeled with number of cluster it belongs to. Since center coordinate of each of cluster is indicated inside the model, the distance calculation from the particular data point to the center is possible.
The average distance of every data in the cluster and its standard deviation becomes useful for determine the outlier. By setting the outlier benchmark to be the row which has the distance greater than one standard deviation of mean value, the alert can be labeled.

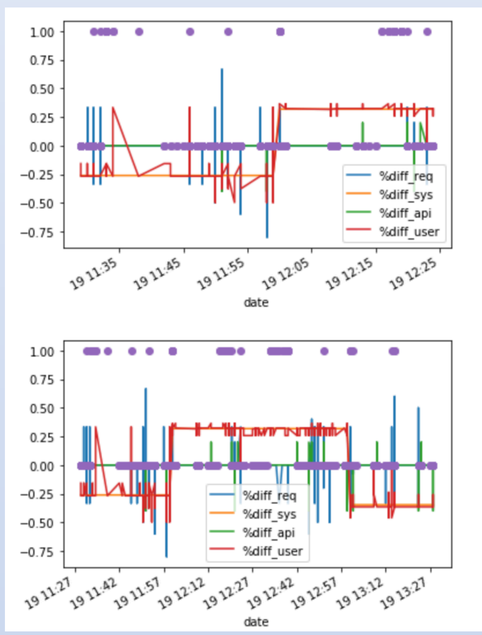

Figure 6

Model evaluation

1. Elbow method: Depends on the input data. Normally, for CMS web logs it's around k=10.
2. Silhouette analysis: After determine proper k value, silhouette with squared Euclidean distance can be calculated from the predicted dataframe. For the model now, it ranging from 0.5894087161 to 0.6417416612

Conclusion

After the system is developed and tested, most of the unnecessary logs has been filtered out and left only the abnormal peak that may be occurred due to one of the factor in the features that we put into KMeans model. Along with the important factors, the email is published to CMS monitoring email and can also be able to trace back to the root cause of the alert. Now, we can reduce overwhelming number of false positives of the CMS experiment log by using Spark Streaming service which will filter alert in real-time and publish the alert notification to proper email channel.

Acknowledgements

To begin, I would like to express my special thanks of gratitude to my supervisors: Federica Legger, Valentin Kuznetsov, and Christian Ariza who gave me the opportunity to do this project, and contribution in stimulating suggestions and encouragement, helped me to coordinate my project. Also, I would like to thank CMS monit team member, who helped finding the workplace and other working materials for me. In addition, thank you for your heart warming welcoming party for summer student, which provide the golden opportunity for us to know people from all around the world and make new friends. Furthermore I would also like to acknowledge with much appreciation the crucial role of the staff: Jennifer Dembski, Eszter Badinova and Despoina Driva for the work they did to organize the whole Summer Students program.

Contact Information

Yanisa Sunthornyotin

Address: Chulalongkorn University
Thailand
Email: yanisa.sunt@gmail.com
Web:
https://github.com/mingyanisa

1.CERN monit team 2.CERN summer student