Churn Prediction in Financial Data

Faizan Shakeel, Ayshan Yariyeva, Shyam Vandan Mishra

1 Introduction

Churn prediction is one of the most popular Big Data use cases in business. It consists of detecting customers who are likely to cancel a subscription to a service and who will continue.

1.1 Data sets

There are three data sets that we have used in the project.

1.1.1 Transfer History

This data set has the records of the customers who did the transfers of money in HUF currency. The data set contains 307402 rows of data and 5 columns.

- TRANSFER_ID - Unique id for each transfer.
- PARTY_ID - Unique id for each customer.
- CURRENCY - The currency in which the transfers were held.
- AMOUNT - Transfer amount.
- VALUE_DATE - Date of the transfer.

1.1.2 Login History

The data set has the records of the customers with their login time. The data set contains 586182 rows of data and 2 columns.

- PARTY_ID - Unique id of the each customer.
- LAST_LOGIN - Login time and date of the user.

1.1.3 INVOICE_HISTORY

The data set has the records of the invoices of the customers.The data set contains 636915 rows of data and 8 columns.

- INVOICE_ID - Unique id of the purchase.
- PARTY_ID - Unique id of the customer.
- CURRENCY - Type of currency in which the customer purchased.
- TAX_INCLUSIVE_AMOUNT - Tax amount on the purchase.
- PAYABLE_AMOUNT - The total amount to be paid by the customer including tax.
- ISSUE_DATE - The date of issue of the invoice
- TAX_POINT_DATE - VAT due on a particular transaction.
- DUE_DATE - Last date of the payment to deposit

2 Overall system architecture

2.1 System architecture

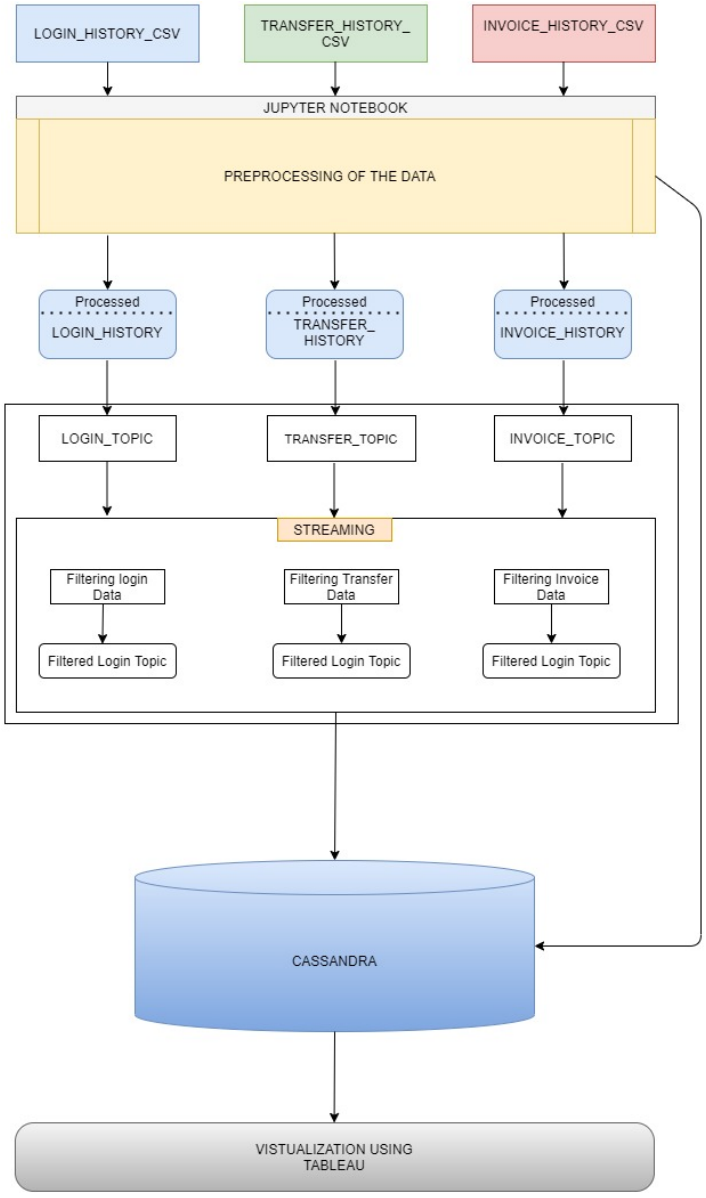

Figure 1: This figure shows the overall architecture of the system that has been implemented in this project

2.2 Jupyter Notebook Procedure

2.2.1 Preprocessing of data

There are three data sets that were provided and to clean and preprocess the data, jupyter has been used. Initially for Login History data, time and date has to be separated for further process and observation. For Transfer and Invoice data set, same procedure has been followed to verify and check for the missing and unrelated values.

2.2.2 Counting the $PARTY\_ID$ from all the Data set

There are repetitive entries of logins,transfers and invoices have been mentioned in the data set. Login shows at which date and on what time login has been made from a particular Id. Transfer data shows on which date transfers have been done. Invoice data set shows the date of the generation of the invoice for a particular id and what is total payable amount, etc. All these data is from 2017 to 2020. Therefore, to know how many times a particular Id is mentioned in the data, counting has been done.

2.2.3 Removing the Duplicates

All the data set combined had more than 15,90,000 data (rows) but there were only few thousand unique $PARTY\_IDs$. Therefore, we have to remove the duplicate data from the data sets.

2.3 Kafka

2.3.1 Why Apache Kafka?

We have used Apache Kafka for our experiment, because it is most popular for processing stream data, it is fast, scalable and distributed. It has characteristics of High throughput, real time etc. as mentioned in the book [1].

2.3.2 CSV data to Kafka Producer

To get this step right we tried many things. Initially we tried sending the data directly from python (after preprocessing) to producer but we could not succeed because of some library issue at the receiving end.

Next we tried using Kafka connect (Spooldir) but this also failed because all the jar files in library could not sink. Then we wrote a code in Java for sending CSV data row by row to Kafka producer.

2.4 Kafka Stream

Kafka stream is a library which is used for performing transformation on data before it is sent to data store. The stream processing takes an input from

producer and performs computing and generates output [2].

2.5 Apache Cassandra and Tableau

The reason for using Cassandra as a data store is that it can handle massive data sets and highly fault tolerant. It is also a decentralized structured storage and a NoSQL database [3]. Creating tables, sending and storing is easy process. Tableau is a sophisticated data visualization and rapid analytics software[4] . It allows to use range to graphs and filters to visualize data efficiently. Simba an ODBC (Open Database Connectivity) driver is used to send data from Cassandra to Tableau. As it is a third party driver, some of the function in Tableau was restricted to use.

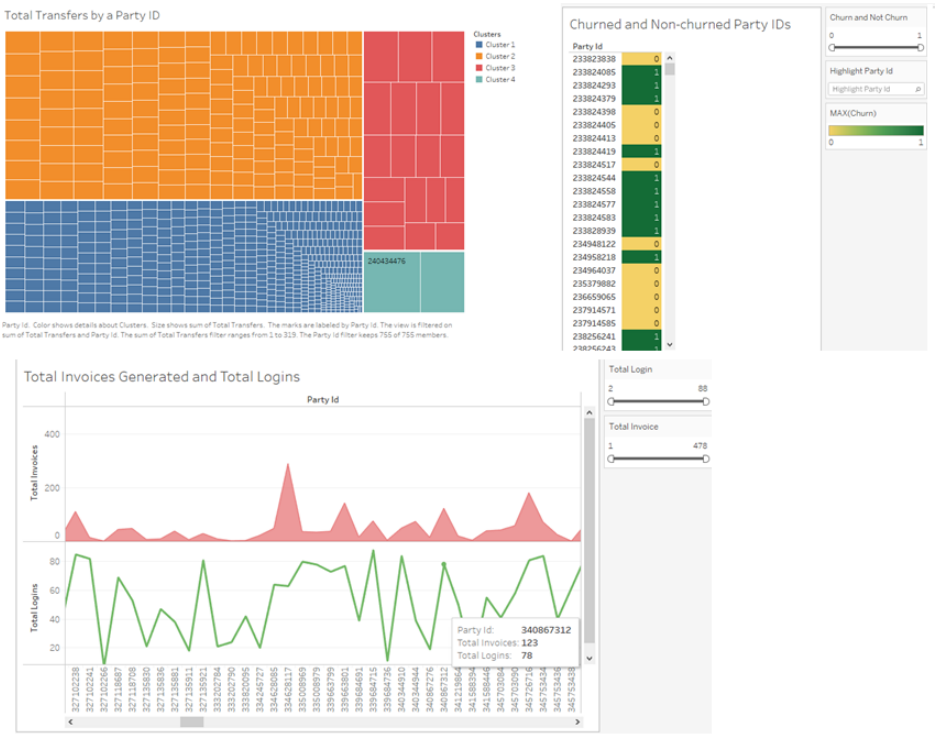

Figure 2: Snapshot of graphs generated in Tableau

3 Experiments

3.1 Individual Streaming of data

As we have been provided with three data sets, without merging we did individual transformations on the streaming data. After taking the data as row from the producer we streamed it and proceeded with transformation.

3.2 Transformation on the Data sets

Individual transformations are being computed over different data sets.
For *Login_history* data sets counting of unique Id has been computed to find out how many times a particular Id logged into the system in four years. This shows the interest of a Party Id in their accounts.Result for this transformation has been obtained by using GroupBy and count.
The other two data sets have used almost same approach to find out churn and not churn Party IDs. For invoice history data set, a filter is used to take out important data which could be used for predicting churn.

3.3 Steps involved in Predicting Churn

The data sets were analysed clearly and useful information were extracted based on certain factors like the number of transactions done by a particular customer, number of times the customer logged into the system, the number of invoices generated for each customer. Based on the Z-score we detected the outliers and removed them(we detected the data points how far they are from the mean) Z-score for a data-point in a distribution with mean and standard deviation is given by the formula:

$$Z = \frac{x - \mu}{\sigma} \tag{1}$$

where:

- μ is the standard deviation
- σ is the mean

3.4 Merging the 3 streams

Merging all the streams after transformation have been tried before sending to data store. We created one consumer to consume data from all the streams and merge but during the procedure we have encountered lots of problems like data set were not compatible with each other and sometime merging only two, not all the streams.

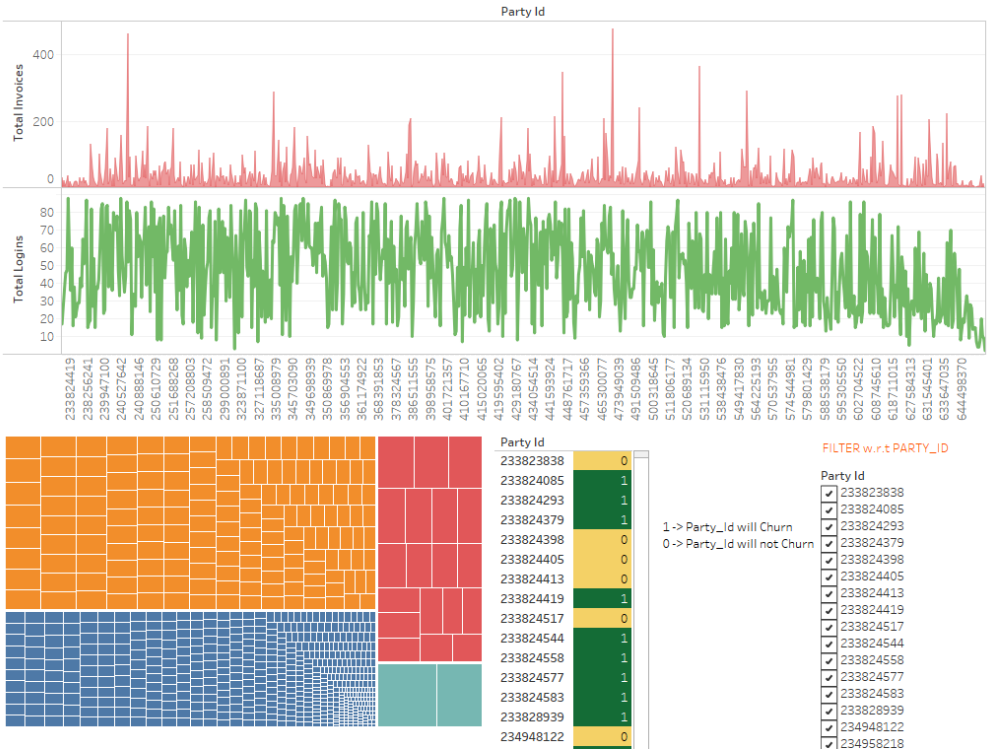

Figure 3: Dashboard Generated in Tableau

4 GitHub Link:

All the files and code can be found in the following link:
https://github.com/erfury/OST-Streamining

References

[1] Nishant Garg : Learning Apache Kafka - Second Edition 2nd Revised ed. Edition (2015)

[2] Study of Apache Kafka in Big Data Stream Processing

[3] Avinash Lakshman, Prashant Malik: Cassandra - A Decentralized Structured Storage System In: ACM SIGOPS Operating Systems Review(2010)

[4] Murphy, Sarah Anne, "Data Visualization and Rapid Analytics: Applying Tableau Desktop to Support Library Decision-Making." Journal of Web Librarianship 7, no. 4 (2013) Decision-Making

[5] Ullah, H. Hussain, I. Ali and A. Liaquat, "Churn Prediction in Banking System using K-Means, LOF, and CBLOF," 2019 International Conference on Electrical, Communication, and Computer Engineering (ICECCE), Swat, Pakistan, (2019)

[6] Bejeck, B. (2018). Kafka Streams in Action: Real-time apps and microservices with the Kafka Streams API (1st ed.). Manning Publications.