Real-time fraudulent transactions detection

BIG DATA ANALYTICS. MILESTONE 2

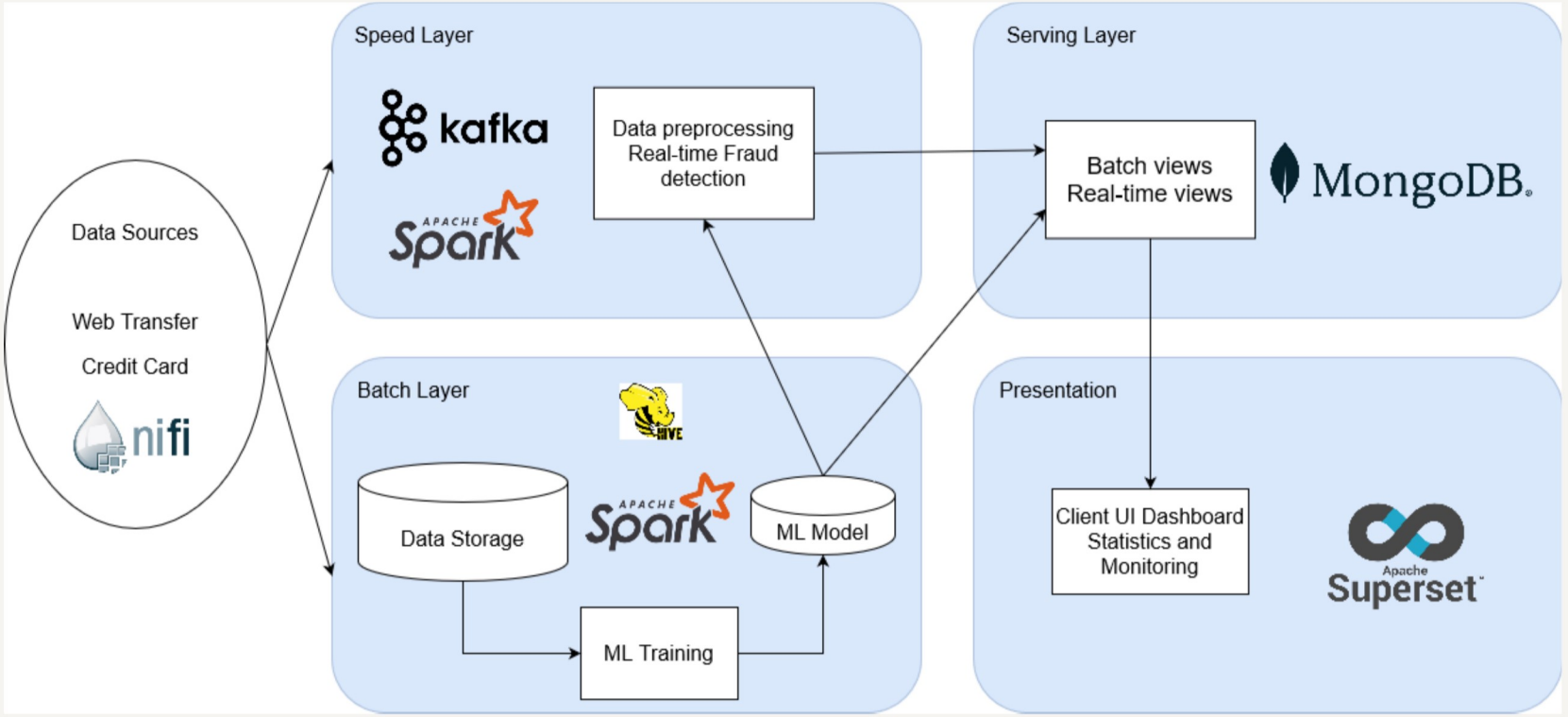
TEAM: SKAMP

SUPERVISOR: MGR INZ. JAKUB ABELSKI

Updates from Milestone 1

- Project architecture layout was updated to better follow the Lambda architecture:
  - included NoSQL database as a storage for data querying in the Serving layer;
  - master data storage was moved to the Batch layer;
  - The architecture schema was updated to better represent the data flow.

Project Architecture



Datasets

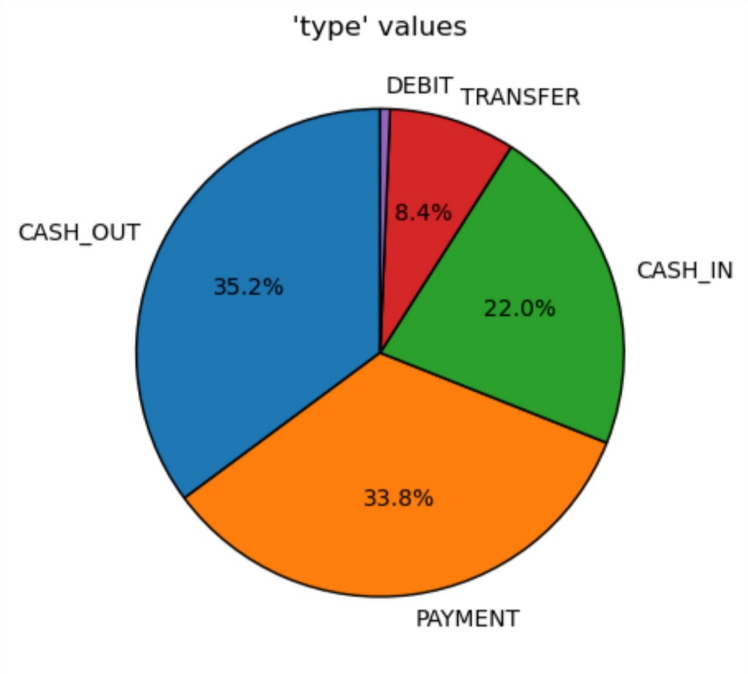
Data Source	Content	Volume	Fraud, %	Link
1. Fraudulent Transactions Data	Dataset for predicting fraudulent transactions for a financial company.	6,362,620 rows and 10 columns (493.53 MB)	0.13%	Kaggle
2. Credit Card Fraud	Contains features with transactional context.	1,000,000 transactions (58.9 MB)	8.7%	OpenML
3. Credit Card Transactions Synthetic Data Generation	A collection of synthetic credit card transaction data.	1,785,308 transactions; 5,000 customers; (153.66 MB)	3%	Kaggle
4. Credit Card Fraud Detection	Transactions made by credit cards in September 2013 by European cardholders.	284,807 transactions (150.83 MB)	0.17%	Kaggle

1. 'Fraudulent Transactions Data' from Kaggle

33.81% of all transactions have Merchant destination

Column	Content	Type
step	maps a unit of time in the real world. In this case 1 step is 1 hour of time. Total steps 744 (30 days simulation)	int
type	type of the transaction. Available values: CASH-IN, CASH-OUT, DEBIT, PAYMENT and TRANSFER	str
amount	amount of the transaction in local currency	float
nameOrig	customer who started the transaction	str
oldbalanceOrg	initial balance before the transaction	float
newbalanceOrig	new balance after the transaction	float
nameDest	customer who is the recipient of the transaction	str
oldbalanceDest	initial balance recipient before the transaction. Note that there is not information for customers that start with M (Merchants)	float
newbalanceDest	new balance recipient after the transaction. Note that there is not information for customers that start with M (Merchants)	float
isFraud	this is the transactions made by the fraudulent agents inside the simulation. In this specific dataset the fraudulent behavior of the agents aims to profit by taking control or customers accounts and try to empty the funds by transferring to another account and then cashing out of the system	int
isFlaggedFraud	the business model aims to control massive transfers from one account to another and flags illegal attempts. An illegal attempt in this dataset is an attempt to transfer more than 200.000 in a single transaction	int

1. 'Fraudulent Transactions Data' from Kaggle (cont.)



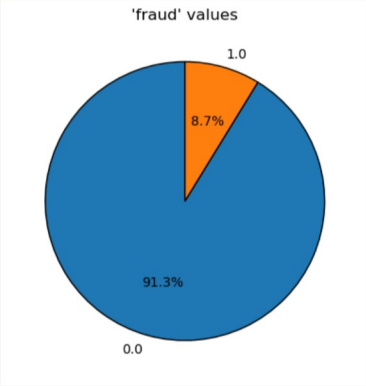
16 transactions contain amount 0.0 with type CASH_OUT. All of them are fraudulent

Only 8213 (0.13%) out of over 6 million transactions are fraudulent.

2.'Credit_Card_Fraud_' from OpenML

No 'amount' feature

8.7% of the fraud transactions



Column	Content	Type
distance_from_home	This is a numerical feature representing the geographical distance in kilometers between the transaction location and the cardholder's home address.	float
distance_from_last_transaction	This numerical attribute measures the distance in kilometers from the location of the last transaction to the current transaction location.	float
ratio_to_median_purchase_price	A numeric ratio that compares the transaction's price to the median purchase price of the user's transaction history.	float
repeat_retailer	A binary attribute where '1' signifies that the transaction was conducted at a retailer previously used by the cardholder, and '0' indicates a new retailer.	[0, 1]
used_chip	This binary feature indicates whether the transaction was made using a chip (1) or not (0).	[0, 1]
used_pin_number	Another binary feature, where '1' signifies the use of a PIN number for the transaction, and '0' shows no PIN number was used.	[0, 1]
online_order	This attribute identifies whether the purchase was made online ('1') or offline ('0').	[0, 1]
fraud	A binary target variable indicating whether the transaction was fraudulent ('1') or not ('0').	[0, 1]

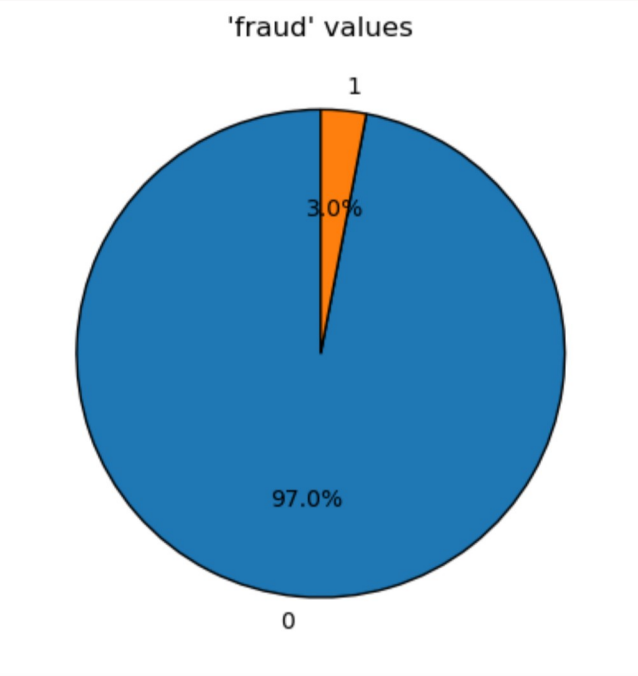
3. 'Credit Card Transactions Synthetic Data Generation' from Kaggle

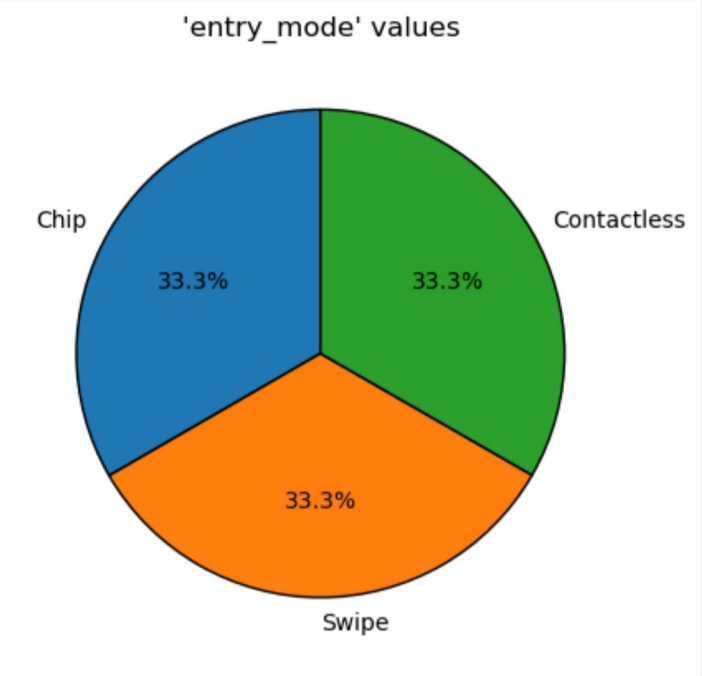
Customer data must be added during preprocessing

3% of fraudulent transactions

Column	Content	Type
transaction_id	Random string containing specific transactions id	str
post_ts	Date and time of the transaction	str
customer_id	Specific customer id	str
bin	Bank Identification Number	int
terminal_id	Specific terminal id	int
amt	Transaction amount	float
entry_mode	Mode of the transaction. Possible values are Contactless, Chip and Swipe.	str
fraud	Target variable containing 1 for fraudulent transaction and 0 otherwise	int
fraud_scenario	Additional label for the transaction. 97% of the dataset has value 0. No specific description for each scenario is provided.	int, [0, 1, 2]
mean_amount	Average transaction amout for a specific customer	float
std_amount	Standard deviation of the transaction amout for a specific customer	float
mean_nb_tx_per_day	Mean number of transactions per day for a specific customer	float
customer_bin	Bank Identification Number of a customer	int

3. 'Credit Card Transactions Synthetic Data Generation' from Kaggle (cont.)





4. 'Credit Card Fraud Detection' from Kaggle

Column	Content	Type
Time	The seconds elapsed between the transaction and the first transaction in the dataset	int
V1 ... V28	The principal components obtained with PCA. The original features and more background information about the data are not provided.	float
Amount	Transaction amount	float
Class	Target variable; 1 for fraudulent transaction and 0 otherwise	int, [0, 1]

Only 492 (0.17%) out of 280k transactions are fraudulent.

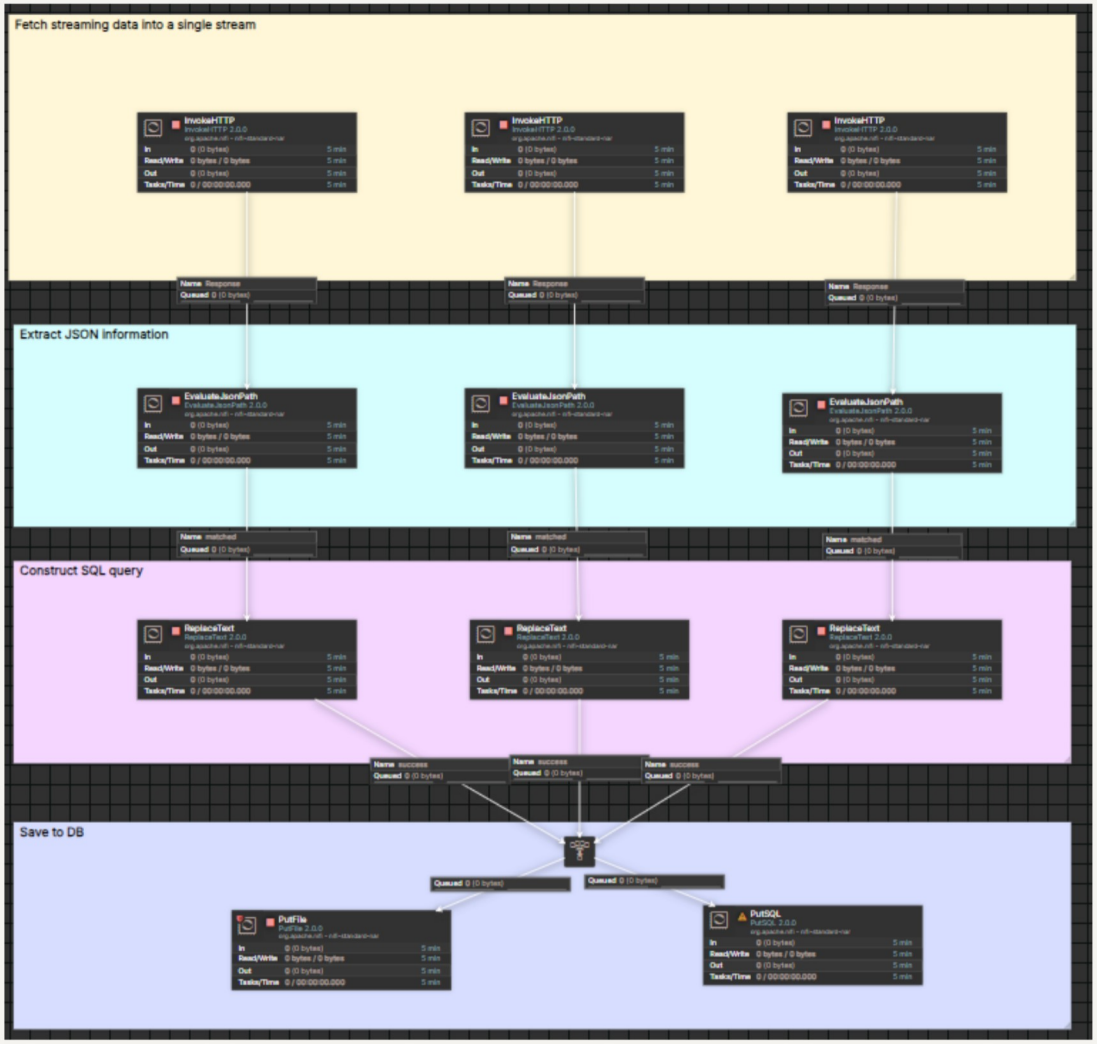
No business information about the features; only principal components obtained with PCA.

Data pre-processing

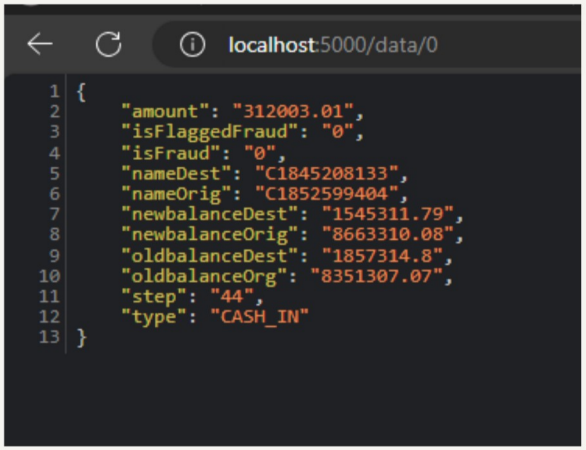
- All datasets are either already in numeric form, or can be easily transformed to numeric values;
- The data format and business information of the features are extremely different. It makes it almost impossible to combine them into a single huge dataset. Therefore, each should be treated as a different data source and processed separately.

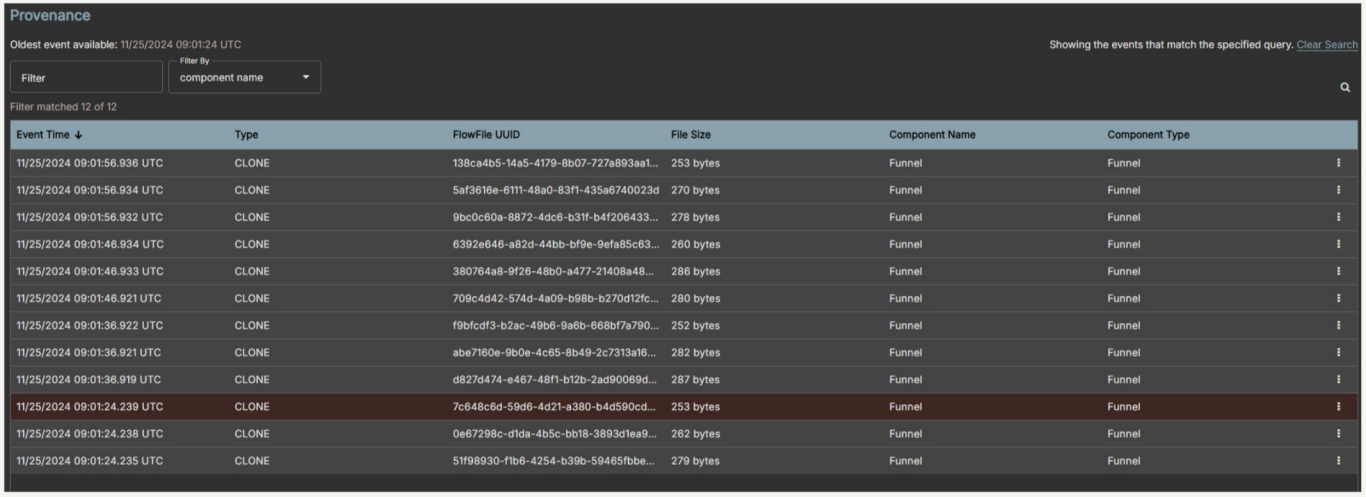
Data acquisition

1. Ingest data from HTTP endpoints every x seconds
2. Extract json variables
3. Prepare a SQL statement
4. Combine streams and save to the database



Data acquisition (cont.)





Data storage strategy

Main storage for master data and pre-processed data: Apache Hive.

Data tables will be created for each dataset separately for futher processing:

- transactions_d1 – raw data from the dataset 1;
- transactions_processed_d1 – processed data from the dataset 1;
- transactions_d2 – raw data from the dataest 2;
- transactions_processed_d2 – processed data from the dataset 2;
- … etc.

Unit testing and CI/CD

```
============================================= test session starts =============================================
platform win32 -- Python 3.13.0, pytest-8.3.3, pluggy-1.5.0 -- C:\ProgramFiles\Anaconda3\envs\bigdata13\python.exe
cachedir: .pytest_cache
rootdir: C:\home\WUT\Semester_3\BigData\Big-Data-Analytics
collected 12 items

services/streaming_simulation/test_streaming_simulation.py::StreamingSimulationTestCase::test_data_stream PASSED [  8%]
tests/data_utils/test_utils.py::test_preprocess_1_payment PASSED                                            [ 16%]
tests/data_utils/test_utils.py::test_preprocess_1_cash_in PASSED                                            [ 25%]
tests/data_utils/test_utils.py::test_preprocess_1_cash_out PASSED                                           [ 33%]
tests/data_utils/test_utils.py::test_preprocess_1_debit PASSED                                              [ 41%]
tests/data_utils/test_utils.py::test_preprocess_1_unknown PASSED                                            [ 50%]
tests/data_utils/test_utils.py::test_preprocess_row_2 PASSED                                                [ 58%]
tests/data_utils/test_utils.py::test_preprocess_3_contactless PASSED                                        [ 66%]
tests/data_utils/test_utils.py::test_preprocess_3_chip PASSED                                               [ 75%]
tests/data_utils/test_utils.py::test_preprocess_3_swipe PASSED                                              [ 83%]
tests/data_utils/test_utils.py::test_preprocess_3_unknown PASSED                                            [ 91%]
tests/data_utils/test_utils.py::test_preprocess_row_4 PASSED                                                [100%]

============================================= 12 passed in 0.57s ==============================================
```

Plan for future work

- ML tasks
  - Binary classification models: Logistic Regression, Random Trees and ensembles;
  - Anomaly detection: clustering, NN-based autoencoders;
- Batch processing
  - Historical data analysis;
  - ML models training;
  - Batch views preparation;
- Stream processing
  - Real-time data preprocessing;
  - New data fraud prediction;
  - Real-time views preparation;
- Presentation
  - Apache Superset interactive dashboards;
  - Visualization of batch and stream processing.

Thank you for attention!

TEAM: SKAMP

SALVEEN_SINGH.DUTT.STUD@PW.EDU.PL

KARINA.TIURINA.STUD@PW.EDU.PL

MIKOLAJ.MALEC.STUD@PW.EDU.PL

PATRYK.PRUSAK.STUD@PW.EDU.PL