System wykrywania podejrzanych transakcji.

Analiza danych w czasie rzeczywistym – projekt zaliczeniowy

Michał Truszczyński

Cel projektu

- Cel: Zbudowanie zautomatyzowanego systemu wykrywania podejrzanych transakcji.
- Potrzeba biznesowa
- Problem: Oszustwa kartowe i płatnicze generują znaczące straty dla instytucji finansowych. Problem jest znaczący ponieważ rośnie liczba transakcji elektronicznych a więc i liczba tego typu przestępstw. Jak najszybsze wykrywanie tego typu sytuacji umożliwia zablokowanie transakcji przed ich finalizacją. Instytucją finansowym w procesie zarządzania ryzykiem podejrzanych transakcji zależy zarówno na unikaniu fałszywych alarmów jak i nie przepuszczaniu realnych zagrożeń.
- Korzyści z wdrożenia rozwiązania: redukcja strat finansowych i kosztów związanych z kontrolą, zwiększenie zaufania klientów, dostosowanie się do reguł i wymagań prawnych, automatyzacja procesów, ograniczenie potrzebnych nakładów pracy administracyjnej.

Architektura systemu - przepływ danych 1

1. Dane wejściowe - plik: creditcard.csv
   - Źródłem danych jest zbiór transakcji płatniczych z oznaczeniem, czy są one oszustwem (Class = 1) czy nie (Class = 0).
   - Dane zawierają m.in. czas transakcji, kwotę oraz 28 cech (V1–V28) po transformacji PCA.
   - Źródło danych: Kaggle, https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud
2. Producent – symulator strumienia danych
   - Wczytuje dane z pliku CSV i przesyła każdą transakcję jako wiadomość do Kafki.
   - Dodaje losowe lub zdefiniowane opóźnienia, aby naśladować prawdziwy strumień zdarzeń.
3. Kafka – system kolejkowania zdarzeń
   - Odbiera dane z producenta i buforuje je w tzw. Topicu.
   - Zapewnia niezawodne przekazywanie danych do konsumenta (Spark).
   - Działa jako pośrednik między źródłem danych a analizą w czasie rzeczywistym.

Architektura systemu - przepływ danych 2

4. Spark – konsument i analityk
   - Spark odczytuje dane z Kafka topicu i przetwarza je w mikropartiach (batchach)
   - Ładuje wcześniej wytrenowany model (model.pkl) i dokonuje predykcji na każdej transakcji.
   - Wykrywa, które transakcje są potencjalnym oszustwem (prediction == 1).
5. Model ML – wykrywanie oszustw
   - Model został wytrenowany offline na zbiorze creditcard.csv
   - Po załadowaniu do Sparka działa jako funkcja predykcyjna.
   - Służy do klasyfikacji transakcji na fraud lub non-fraud w czasie rzeczywistym.
6. WebSocket – przesyłanie wyników do użytkownika
   - Jeśli transakcja zostanie uznana za podejrzaną, Spark przesyła ją do serwera WebSocket.
   - WebSocket zapewnia stałe połączenie z przeglądarką użytkownika, umożliwiając natychmiastowe wyświetlanie alertów.

Architektura systemu - przepływ danych 3

7. Dashboard HTML – interfejs użytkownika
   - Użytkownik widzi nowo wykryte przypadki oszustw w formie wiadomości na stronie web.
   - Wizualizacja odbywa się w czasie rzeczywistym, bez potrzeby odświeżania strony.
8. Docker Compose – uruchomienie całości
   - Każdy komponent (Kafka, Spark, Produtent, Model, WebSocket) to osobny kontener.
   - Cały system można uruchomić jednym poleceniem docker compose up kafka producer consumer websocket.
   - Wcześniej należy uruchomić kontener trainer aby stworzyć model.

Propozycje kierunku rozwoju

1. Udoskonalenie modelu wykrywania podejrzanych transakcji.
2. Rozbudowa dashboardu:
   - Dodanie statystyki wykrytych transakcji podejrzanych
   - Dodanie panelu do ręcznej oceny transakcji przez analityków
   - Statystyki wykrytych nieprawidłowości
3. Automatyczna reakcja systemu:
   - Powiadomienia SMS, email do klientów
   - Blokada konta
   - Integracja z systemem bankowym