Brainstorm

Diseño del DAaaS

### Definición la estrategia del DAaaS

La estrategia consiste en ofrecer un servicio de detección (API) de fraude en las tarjetas de crédito basado en un modelo de predicción. Se puede incorporar un sistema de alerta para el cliente y el banco, para que les avise en el momento para poder bloquear la tarjeta. Se puede monetizar vendiéndolo a bancos o a empresas que ofrecen un servicio de seguro contra el fraude (como identity theft protection). Cogeremos los datos de las bases de datos de los bancos y los llevaremos (los que nos interesan) a una bbdd para que lleguen las transacciones en real-time. A la hora de analizarlos, tendremos en cuenta tanto las transacciones de las tarjetas como la demografía de los clientes. Los datos y factores que hay que tener en cuenta según las susodichas categorías son:

* Transacciones de las tarjetas:

El lugar de compra, cantidad de la compra, la frecuencia del uso, etc.

* Demografía de los clientes:

Dónde viven, dónde suelen hacer compras, qué tipo de compras hacen con la tarjeta, etc.

Tenemos que tener en cuenta la protección de datos porque los datos que vamos a coger de la base de datos de los bancos contienen información sensible. A la hora de coger los datos del dataset podemos usar algoritmos de Python para cifrar los datos.

### Arquitectura DAaaS

*Definir la selección de componentes, la definición de procesos de ingeniería y el diseño de interfaces de usuario. Diseño y ejecución de Proofs-of-Concept (PoC) para demostrar la viabilidad del enfoque DAaaS.*

Los componentes que hacen falta son:

* BBDD existentes de los bancos (lo ideal sería Kafka)
* Componente para llevar los datos de la bbdds del banco a nuestra bbdds (Google Cloud Storage)
* BBDD nuestra para procesar los datos (Cassandra)
* Componente para analizar los datos con Python y crear algoritmos para el modelo de predicción (Tensorflow?)
* Componente para montar el API (Google Cloud Functions)
* Componente para recoger los datos de los resultados del análisis (Google Cloud Storage)
* BBDDs para los resultados (Mongo)
* Componente para avisar a los clientes si ha detectado el API un caso sospechoso de fraude con un push notification
* Componente para visualizar los datos (Tableau)

### DAaaS Operating Model Design and Rollout

1. Coger los datos que nos interesan de las bbdds del banco y llevarlos a Kafka. Preparamos Kafka para que cuando reciba una notificación de una nueva transacción lleve los datos a Cassandra. Esta primera función la hará una persona física al montar la bbdds de Kafka. Automatizamos el proceso de llevar los datos a Cassandra con Python.
2. Usar una trigger para arrancar cada x tiempo una query en Cassandra que detecte datos nuevos. Al detectar datos nuevos el script seguirá con los pasos de limpieza y preparación de los datos.
3. Llevar los datos al modelo de predicción y pasarlos por algunos algoritmos que decidirán si la transacción es sospechosa de fraude o no. Este paso se automatizará también usando Python.
4. Al recibir los datos de la compra, se les pasa a la API y después al modelo de predicción. Una vez tenga una respuesta el modelo de predicción, basado en los algoritmos con los que está entrenado, la respuesta del modelo de predicción se pasa a la API. Si la respuesta es sí, saltará una alerta que se le manda al cliente. Este paso se automatiza con Python para que se ejecute al recibir la respuesta del modelo de predicción. Preparamos un script que se ejecuta cada x tiempo y que busca nuevas respuestas.
5. Notificar a los clientes y el banco en el caso de que la respuesta del modelo sea “sí”. En este caso se bloquea la tarjeta. Esto se puede hacer con un servicio de push notifications. Este paso se automatiza con Python, indicando que al recibir una “sí” como respuesta se mande una notificación al cliente y a través de dichas notificaciones se ejecuta un script para el bloqueo de la tarjeta. Montamos un script que se ejecuta cada x tiempo buscando nueva información de la API.
6. Un script que lleva todos los resultados del análisis a Google Cloud Storage y luego a Mongo.
7. Llevar todos los datos de las transacciones en un CSV a Tableau para visualizar los datos. Este paso lo hará una persona cada semana.

### Detalles que tener en cuenta y puntos de mejora:

* La parte de entrenamiento de los algoritmos lo he dejado muy básico (tal y como lo explicaste con el diagrama) porque aun desconozco el asunto.
* Mi idea es que los script se ejecuten “en bucle” en cada paso, cada x tiempo (no sé a qué intervalo). Así, no haría falta ningún trigger.

Link a Diagrama:

El Diagrama lo he subido a github como archivo separado.