Proyecto Detección de Fraude - equipo The Balloon

Introducción

Este documento intenta mostrar de una forma no exhaustiva como fue evolucionando el proyecto de “Detección de Fraude” realizado por el equipo **“The Balloon”** para el **“Bootcamp IA I (NOV 23)”** dictado por **KeepCoding**.

Semana 0 (24/06 al 30/06)

El lunes 24 se recibió toda la información relacionada a la práctica final y la formación de equipos. En ese momento nadie tenía una idea completa sobre qué realizar en la práctica, con lo cual no llevamos como tarea investigar que podríamos armar.

Durante el transcurso de la semana surgió la idea de armar algo relacionado al fraude, pero surgieron muchas dudas sobre qué datos íbamos a trabajar, ya que las empresas NO hacen públicos este tipo de información.

Por otro lado también surgió la inquietud de poder armar un solo equipo, ya que si se armaban dos equipos, uno debería tener tan solo dos integrantes. Por suerte nos dieron el OK y formamos un equipo con cinco integrantes:

- Angel Luis Ortega Amador
- Carlos Molano Gómez
- Juan Carlos Avalos
- Otger Peidro Cid
- Pedro Turiel Miranda

El viernes 28 inscribimos al equipo y nos pusimos a buscar datasets que nos puedan llegar a servir para el proyecto y nos fuimos comunicando utilizando un grupo en Discord.
También ese mismo viernes se armó un workspace en Trello donde se creó un tablero para dejar las tareas.

Semana 1 (01/07 al 07/07)

El lunes 01 nos fuimos comunicando para ver que datasets podíamos utilizar.
Encontramos algunos en Kaggle y otros en Huggin Face.
Ese mismo día tuvimos nuestra primera meet usando discord y nos llevamos como tarea revisar los datasets sugeridos para dejar solo con los que se va a trabajar.
También fuimos resolviendo los accesos a Trello para que todos puedan trabajar.

El martes 02 terminamos de seleccionar las datasets.
Conversamos sobre el alcance que podría llegar a tener el proyecto, la idea de máxima fue tener un RAG que se nutra de documentación sobre fraude y puede realizar consultas a un modelo de Machine Learning sobre fraude en transacciones de tarjeta de crédito y consultar usando Computer Vision si la imágen de documento es real o no.

Para esto deberíamos armar:

- Modelo de predicción de Fraude con Machine Learning
- Modelo de predicción de imágen Falsa con Computer Vision
- Entrenar los modelos en la Nube
- Montar en la nube una API y un frontend
- Montar en la nube un RAG

En ese momento surgieron miles de dudas sobre el alcance real y los costos, con lo cual apuntamos a llegar a algo un poco menos ambicioso, pero a la vez exigente.

Nos llevamos como tarea comenzar a revisar si necesitamos unir dos datasets para Computer Vision y ver como hacer para trabajar en google colab gratis.

También decidimos cambiar a Microsoft Teams para las reuniones, ya que en discord si nos complicaba compartir pantalla y configurar audio y video correctamente.

El jueves 04 confirmamos los datasets a utilizar, ya que tuvimos que consultar varias bases de datos para buscar datasets con datos que pudieran servirnos.
Este tema no cuenta con muchos datos ya que puede incumplir con normas de privacidad, por lo que encontrar un dataset aceptable fue una tarea complicada.

Una vez elegidos los datasets y definido el alcance del proyecto, pasamos a completar el listado de tareas en Trello definiéndolo en secciones ToDo, Doing y Done.
En un primer momento queríamos utilizar grandes volúmenes de datos, pero posteriormente chocamos con restricciones en cuanto al hardware.

Podemos resumir lo que vamos a necesitar realizar:

- Modelo de predicción de Fraude con Machine Learning
- Modelo de predicción de imágen Falsa con Computer Vision
- Almacenar los resultados en la nube para poder consultarlos
- Montar en la nube una API que realice consultas

Luego comenzamos a avanzar según la disponibilidad de cada integrante.

Machine Learning:

Se comenzó a trabajar en el EDA utilizando colab compartido, que no persiste los resultados, con lo cual en cada ejecución se debe descargar nuevamente el dataset.
Se armó el diccionario de datos y se notó la necesidad de balancerlo, debido a que sólo contiene el 0,5% de datos marcados como fraude.
Se identificó que se va a trabajar con Clasificación Binaria, ya que el TARGET “is_fraud” es una columna con valores 0 y 1.

Check List:

- Modelo de predicción de Fraude con Machine Learning

- Modelo de predicción de imágen Falsa con Computer Vision
- Almacenar los resultados en la nube para poder consultarlos
- Montar en la nube una API que realice consultas

Semana 2 (08/07 al 14/07)

Una de las intenciones que teníamos era utilizar 2 datasets de imágenes para tener un buen conjunto de datos, pero a la hora de hacer algunas pruebas con modelos, incluso con Colab de pago, encontramos que no teníamos hardware suficiente para entrenar Redes Neuronales con tal cantidad de datos, por lo que terminamos utilizando solo parte de uno.

Machine Learning:

Se discutió con el equipo sobre la necesidad de balancear el dataset para evitar un poco el overfitting. Entre las posibilidades estaba agregar datos o reducir el dataset de forma proporcional para utilizar todos los datos de fraude y no generar más datos falsos.

Se decidió tomar los datos de fraude y agregarles una cantidad proporcional de 7 a 1 de datos que NO SON FRAUDE.

Luego de obtener el dataset balanceado se realizó la división del mismo en dos datasets uno de TRAIN y otro de TEST.

Se realizó todo el EDA sobre el dataset de TRAIN.
En el mismo se analiza las features de texto, las numéricas, se ve que relación tiene nuestro TARGET contra las diferentes features.
Se va documentado cuales son las features que se podrían eliminar y porqué.
Se generan diversos gráficos, como por ejemplo histogramas y count plots.
Por último se arma un diagrama de correlación para confirmar las features numéricas que se desean eliminar.

Al finalizar el EDA se creó otro notebook para el pipeline, en el cual tanto para el dataset de TRAIN como el de TEST se eliminan features, se realiza LabelEncoder sobre las features de texto y luego un StandardScaler sobre todas las features que se van a utilizar para entrenar.

Luego se procede a entrenar varios modelos con su configuración por default, para poder comprobar cual brinda un mejor resultado.

Los modelos utilizados fueron:
Random Forest, LASSO, KNN, Decision Tree, SVM y Gradient Boosting

Obteniendo un mejor “Accuracy” mediante **Random Forest**.

Luego procedemos a obtener los mejores hiper-parámetros para entrenar el modelo.

Al modelo lo utilizamos para entrenar tanto el dataset de TRAIN como el de TEST y mostramos las diferentes métricas.

Despliegue:

Se comenzó a trabajar en colab utilizando pyngrok y mlflow y surgió la necesidad de tener todo los datasets y modelos compartidos, para que se puedan usar en el despliegue.
Ya que de lo contrario se debería entrenar el modelo en el backend, y en esta etapa no estaba eso contemplado por temas de costos.
Debido a esto se tuvo que crear una carpeta compartida, proceder a guardar los modelos generados en Machine Learning y acceder a los mismos desde google drive.
Se especifican las rutas de los archivos de entrenamiento y prueba, así como los modelos previamente entrenados y los resultados de los pipelines de modelos guardados.

Computer Vision:

Comenzamos el proyecto de Computer Vision investigando datasets de HuggingFace para poder realizar el proyecto. Tras varios análisis decidimos decantarnos por el dataset especificado anteriormente.
Una vez elegido el dataset comprobamos que ya estaba dividido en train, validación y test y programamos una función para disminuir el tamaño de los datasets en un octavo. Esto lo hicimos porque sino el coste computacional de entrenar el modelo era muy elevado.
Finalmente, programamos una función para crear tensores y arrays de numpy para las imágenes y etiquetas del dataset.

Check List:

- ~~Modelo de predicción de Fraude con Machine Learning~~
- Modelo de predicción de imágen Falsa con Computer Vision
- Almacenar los resultados en la nube para poder consultarlos
- Montar en la nube una API que realice consultas

Semana 3 (15/07 al 21/07)

Despliegue:

Ya que el equipo tiene algo de experiencia en AWS, se decide desplegar los modelos en AWS.
Tras hacer búsqueda de información se encuentra una guía de como desplegar un servidor de MLflow en una instancia EC2 usando S3 para guardar los artefactos de MLflow.

Los pasos que se han realizado son:

- Crear S3 Bucket en AWS.
- Crear EC2 Instance siendo el sistema operativo Ubuntu en AWS.
- Configurar la base de datos PostgreSQL.
- Configurar los Security Groups.
- Acceso remoto al servidor de MLflow Server.
- Usar SSL para la conexión segura y usar https.

- Comprar el dominio the-balloon-project.com para poder dirigir las peticiones del servidor MLflow.

También se tuvieron que crear sub-carpetas específicas para guardar los datasets, resultados de entrenamientos y modelos, correspondientes a ML, CV y despliegue.
Ya que de lo contrario, cuando se ejecutaban los colab, se obtenía más datos de los necesarios.

Computer Vision:

Creados los tensores y arrays con las imágenes y etiquetas, programamos una función para crear los embeddings de los tensores de imágenes. Esto lo hicimos para que fuera más sencillo entrenar el modelo.
Seguidamente, creamos un modelo sencillo y utilizamos la función hyper opt para tener una primera idea acerca de los hiperparámetos que eran más convenientes a utilizar. Con ello, fuimos ajustando poco a poco el modelo para conseguir el modelo con la mejor precisión posible.

Check List:

- ~~Modelo de predicción de Fraude con Machine Learning~~
- Modelo de predicción de imágen Falsa con Computer Vision
- ~~Almacenar los resultados en la nube para poder consultarlos~~
- Montar en la nube una API que realice consultas

Semana 4 (22/07 al 24/07)

Despliegue:

Se realizan algunos cambios para que se registren bien los datos y nombres de los modelos en MLFlow.

Se realizan pruebas generales para ver que todo funcione y se visualicen los artifacts.

Conversamos para ver los pro y los contra de intentar armar una API y subirla a la nube, pero se decidió no avanzar y enfocarnos en probar todo lo que se va a presentar.

Computer Vision:

Tras consultarlo con nuestro profesor de DeepLearning y contrastar la información con estudios de distintas universidades, decidimos dar por válido los resultados obtenidos y generar un primer modelo válido.
Aunque este primer modelo era bueno, decidimos intentar obtener un mejor modelo y por ello generamos un nuevo modelo con otra arquitectura diferente. Los resultados obtenidos fueron mejores y decidimos decantarnos por este modelo como el modelo final.
Por último, subimos los resultados a MLFlow y guardamos el modelo en un fichero para que se pudiera ejecutar siempre que fuera necesario sin necesidad de ejecutar el notebook.

Check List:

- ~~Modelo de predicción de Fraude con Machine Learning~~
- ~~Modelo de predicción de imágen Falsa con Computer Vision~~
- ~~Almacenar los resultados en la nube para poder consultarlos~~
- Montar en la nube una API que realice consultas

Decidimos dejar pendiente el montar una API en la nube y focalizarnos en terminar de pulir algunos detalles para que funcione todo lo que armamos.