**PROYECTO INDIVIDUAL Nº1: Machine Learning Operations (MLOps)**

|  |  |
| --- | --- |
| ¡Bienvenidos al primer proyecto individual de la etapa de labs! En esta ocasión, deberán hacer un trabajo situándose en el rol de un ***MLOps Engineer***. |  |

**Descripción del problema (Contexto y rol a desarrollar)**

**Contexto**

Tienes tu modelo de recomendación dando unas buenas métricas 😏, y ahora, cómo lo llevas al mundo real? 👀

El ciclo de vida de un proyecto de Machine Learning debe contemplar desde el tratamiento y recolección de los datos (Data Engineer stuff) hasta el entrenamiento y mantenimiento del modelo de ML según llegan nuevos datos.

**Rol a desarrollar**

Empezaste a trabajar como **Data Scientist** en una start-up que provee servicios de agregación de plataformas de streaming. El mundo es bello y vas a crear tu primer modelo de ML que soluciona un problema de negocio: un sistema de recomendación que aún no ha sido puesto en marcha!

Vas a sus datos y te das cuenta que la madurez de los mismos es poca (ok, es nula 😭): Datos anidados, sin transformar, no hay procesos automatizados para la actualización de nuevas películas o series, entre otras cosas…. haciendo tu trabajo imposible 😩.

Debes empezar desde 0, haciendo un trabajo rápido de **Data Engineer** y tener un **MVP** (*Minimum Viable Product*) para las próximas semanas! Tu cabeza va a explotar 🤯, pero al menos sabes cual es, conceptualmente, el camino que debes de seguir ❗. Así que te espantas los miedos y te pones manos a la obra 💪

|  |
| --- |
| Nota que aqui se reflejan procesos no herramientas tecnologicas. Has el ejercicio de entender cual herramienta del stack corresponde a cual parte del proceso |

**Propuesta de trabajo (requerimientos de aprobación)**

**Transformaciones**: Para este MVP no necesitas perfección, ¡necesitas rapidez! ⏩ Vas a hacer estas, ***y solo estas***, transformaciones a los datos:

* Algunos campos, como **belongs\_to\_collection**, **production\_companies** y otros (ver diccionario de datos) están anidados, esto es o bien tienen un diccionario o una lista como valores en cada fila, ¡deberán desanidarlos para poder y unirlos al dataset de nuevo hacer alguna de las consultas de la API! O bien buscar la manera de acceder a esos datos sin desanidarlos.
* Los valores nulos de los campos **revenue**, **budget** deben ser rellenados por el número **0**.
* Los valores nulos del campo **release date** deben eliminarse.
* De haber fechas, deberán tener el formato **AAAA-mm-dd**, además deberán crear la columna **release\_year** donde extraerán el año de la fecha de estreno.
* Crear la columna con el retorno de inversión, llamada **return** con los campos **revenue** y **budget**, dividiendo estas dos últimas **revenue / budget**, cuando no hay datos disponibles para calcularlo, deberá tomar el valor **0**.
* Eliminar las columnas que no serán utilizadas, **video**,**imdb\_id**,**adult**,**original\_title**,**poster\_path** y **homepage**.

**Desarrollo API**: Propones disponibilizar los datos de la empresa usando el framework ***FastAPI***. Las consultas que propones son las siguientes:

Deben crear 6 funciones para los endpoints que se consumirán en la API, recuerden que deben tener un decorador por cada una (@app.get(‘/’)).

* def **cantidad\_filmaciones\_mes( *Mes* )**: Se ingresa un mes en idioma Español. Debe devolver la cantidad de películas que fueron estrenadas en el mes consultado en la totalidad del dataset.

Ejemplo de retorno: *X cantidad de películas fueron estrenadas en el mes de X*

* def **cantidad\_filmaciones\_dia( *Dia* )**: Se ingresa un día en idioma Español. Debe devolver la cantidad de películas que fueron estrenadas en día consultado en la totalidad del dataset.

Ejemplo de retorno: X cantidad de películas fueron estrenadas en los días X

* def **score\_titulo( *titulo\_de\_la\_filmación* )**: Se ingresa el título de una filmación esperando como respuesta el título, el año de estreno y el score.

Ejemplo de retorno: *La película X fue estrenada en el año X con un score/popularidad de X*

* def **votos\_titulo( *titulo\_de\_la\_filmación* )**: Se ingresa el título de una filmación esperando como respuesta el título, la cantidad de votos y el valor promedio de las votaciones. La misma variable deberá de contar con al menos 2000 valoraciones, caso contrario, debemos contar con un mensaje avisando que no cumple esta condición y que por ende, no se devuelve ningun valor.

Ejemplo de retorno: *La película X fue estrenada en el año X. La misma cuenta con un total de X valoraciones, con un promedio de X*

* def **get\_actor( *nombre\_actor* )**: Se ingresa el nombre de un actor que se encuentre dentro de un dataset debiendo devolver el éxito del mismo medido a través del retorno. Además, la cantidad de películas que en las que ha participado y el promedio de retorno. **La definición no deberá considerar directores.**

Ejemplo de retorno: *El actor X ha participado de X cantidad de filmaciones, el mismo ha conseguido un retorno de X con un promedio de X por filmación*

* def **get\_director( *nombre\_director* )**: Se ingresa el nombre de un director que se encuentre dentro de un dataset debiendo devolver el éxito del mismo medido a través del retorno. Además, deberá devolver el nombre de cada película con la fecha de lanzamiento, retorno individual, costo y ganancia de la misma.

**Deployment**: Conoces sobre [Render](https://render.com/docs/free#free-web-services) y tienes un [tutorial de Render](https://github.com/HX-FNegrete/render-fastapi-tutorial) que te hace la vida mas facil 😄 . Tambien podrias usar [Railway](https://railway.app/), o cualquier otro servicio que permita que la API pueda ser consumida desde la web.

**Análisis exploratorio de los datos**: *(Exploratory Data Analysis-EDA)*

Ya los datos están limpios, ahora es tiempo de investigar las relaciones que hay entre las variables de los datasets, ver si hay outliers o anomalías (que no tienen que ser errores necesariamente 👀 ), y ver si hay algún patrón interesante que valga la pena explorar en un análisis posterior. Las nubes de palabras dan una buena idea de cuáles palabras son más frecuentes en los títulos, ¡podría ayudar al sistema de recomendación!😉

**Sistema de recomendación**:

Una vez que toda la data es consumible por la API, está lista para consumir por los departamentos de Analytics y Machine Learning, y nuestro EDA nos permite entender bien los datos a los que tenemos acceso, es hora de entrenar nuestro modelo de machine learning para armar un sistema de recomendación de películas. El EDA debería incluir gráficas interesantes para extraer datos, como por ejemplo una nube de palabras con las palabras más frecuentes en los títulos de las películas. Éste consiste en recomendar películas a los usuarios basándose en películas similares, por lo que se debe encontrar la similitud de puntuación entre esa película y el resto de películas, se ordenarán según el score de similaridad y devolverá una lista de Python con 5 valores, cada uno siendo el string del nombre de las películas con mayor puntaje, en orden descendente. Debe ser deployado como una función adicional de la API anterior y debe llamarse:

* def **recomendacion( *titulo* )**: Se ingresa el nombre de una película y te recomienda las similares en una lista de 5 valores.

**Video**: Necesitas que al equipo le quede claro que tus herramientas funcionan realmente! Haces un video mostrando el resultado de las consultas propuestas y de tu modelo de ML entrenado!

**Spoiler**: El video NO DEBE durar mas de ***7 minutos*** y DEBE mostrar las consultas requeridas en funcionamiento desde la API y una breve explicacion del modelo utilizado para el sistema de recomendacion. En caso de que te sobre tiempo luego de grabarlo, puedes mostrar explicar tu EDA, ETL e incluso cómo desarrollaste la API.

**Criterios de evaluación**

**Código**: Prolijidad de código, uso de clases y/o funciones, en caso de ser necesario, código comentado.

**Repositorio**: Nombres de archivo adecuados, uso de carpetas para ordenar los archivos, README.md presentando el proyecto y el trabajo realizado. Recuerda que este último corresponde a la guía de tu proyecto, no importa que tan corto/largo sea siempre y cuando tu 'yo' + 1.5 AÑOS pueda entenderlo con facilidad.

**Cumplimiento** de los requerimientos de aprobación indicados en el apartado Propuesta de trabajo

NOTA: Recuerde entregar el link de acceso al video. Puede alojarse en YouTube, Drive o cualquier plataforma de almacenamiento. **Verificar que sea de acceso público, recomendamos usar modo incógnito en tu navegador para confirmarlo**.

Aqui te sintetizamos que es lo que consideramos un MVP aprobatorio, y la diferencia con un producto completo.

|  |
| --- |
|  |

**Fuente de datos**

* + [Dataset](https://drive.google.com/drive/folders/1X_LdCoGTHJDbD28_dJTxaD4fVuQC9Wt5?usp=drive_link): Carpeta con los 2 archivos con datos que requieren ser procesados (movies\_dataset.csv y credits.csv), tengan en cuenta que hay datos que estan anidados (un diccionario o una lista como valores en la fila).
* [Diccionario de datos](https://docs.google.com/spreadsheets/d/1QkHH5er-74Bpk122tJxy_0D49pJMIwKLurByOfmxzho/edit#gid=0): Diccionario con algunas descripciones de las columnas disponibles en el dataset.

**Material de apoyo**

En este mismo repositorio podras encontrar algunos links de ayuda. Recuerda que no son los unicos recursos que puedes utilizar!