**Proyecto Individual 1 - Recomendación de Películas**

Este proyecto tiene como objetivo desarrollar un sistema de recomendación de películas utilizando técnicas de Machine Learning. El proyecto abarca el proceso completo de ETL, análisis exploratorio de datos, construcción de un modelo de recomendación y su implementación en una API.

**Tabla de Contenidos**

* [Descripción](#descripci%C3%B3n)
* [Requisitos](#requisitos)
* [Estructura del Proyecto](#estructura-del-proyecto)
* [ETL (Extract, Transform, Load)](#etl-extract-transform-load)
* [EDA (Exploratory Data Analysis)](#eda-exploratory-data-analysis)
* [Modelo de Machine Learning](#modelo-de-machine-learning)
* [Implementación de la API](#implementaci%C3%B3n-de-la-api)
* [Uso](#uso)
* [Contribución](#contribuci%C3%B3n)
* [Licencia](#licencia)

**Descripción**

El archivo Sandagorda - Proyecto Individual 1.ipynb contiene un flujo de trabajo completo para crear un sistema de recomendación de películas. Este flujo de trabajo incluye la extracción, transformación y carga de datos (ETL), análisis exploratorio de datos (EDA), desarrollo de un modelo de Machine Learning para la recomendación de películas y la implementación del modelo en una API.

**Requisitos**

Para ejecutar este notebook, necesitas tener instaladas las siguientes dependencias:

* Python 3.x
* Jupyter Notebook
* Pandas
* NumPy
* Scikit-learn
* Matplotlib
* Seaborn
* Flask (para la API)

Puedes instalar las dependencias utilizando pip:

bash

Copy code

pip install pandas numpy scikit-learn matplotlib seaborn flask

**Estructura del Proyecto**

El proyecto está estructurado en las siguientes secciones:

1. [ETL (Extract, Transform, Load)](#etl-extract-transform-load)
2. [EDA (Exploratory Data Analysis)](#eda-exploratory-data-analysis)
3. [Modelo de Machine Learning](#modelo-de-machine-learning)
4. [Implementación de la API](#implementaci%C3%B3n-de-la-api)

**ETL (Extract, Transform, Load)**

En esta sección se realizan las siguientes actividades:

* **Extracción de Datos**: Carga de datos desde múltiples fuentes como archivos CSV.
* **Transformación de Datos**: Limpieza y preprocesamiento de datos, incluyendo la conversión de tipos de datos, manejo de valores nulos y normalización de datos.
* **Carga de Datos**: Almacenamiento de los datos procesados en un formato adecuado para su análisis posterior.

python

Copy code

# Ejemplo de código para ETL

import pandas as pd

# Cargar datos

df = pd.read\_csv('data/movies.csv')

# Transformar datos

df['release\_date'] = pd.to\_datetime(df['release\_date'], errors='coerce')

df = df.dropna(subset=['release\_date'])

df['budget'] = pd.to\_numeric(df['budget'], errors='coerce').fillna(0).astype(int)

# Guardar datos transformados

df.to\_csv('data/movies\_cleaned.csv', index=False)

**EDA (Exploratory Data Analysis)**

En esta sección se realiza el análisis exploratorio de datos para comprender mejor el conjunto de datos y descubrir patrones interesantes. Incluye:

* Análisis estadístico de las variables.
* Visualización de datos utilizando gráficos como histogramas, diagramas de caja, diagramas de dispersión, etc.
* Identificación de relaciones entre variables.

python

Copy code

# Ejemplo de código para EDA

import matplotlib.pyplot as plt

import seaborn as sns

# Análisis estadístico

print(df.describe())

# Visualización de datos

plt.figure(figsize=(10, 6))

sns.histplot(df['budget'], bins=50, kde=True)

plt.title('Distribución del Presupuesto de Películas')

plt.xlabel('Presupuesto')

plt.ylabel('Frecuencia')

plt.show()

**Modelo de Machine Learning**

En esta sección se construye un modelo de recomendación de películas utilizando técnicas de Machine Learning. Incluye:

* **Selección de Modelo**: Elección del algoritmo de Machine Learning adecuado (por ejemplo, KNN, SVD).
* **Entrenamiento del Modelo**: Entrenamiento del modelo utilizando el conjunto de datos preprocesado.
* **Evaluación del Modelo**: Evaluación del rendimiento del modelo utilizando métricas como precisión, recall y F1-score.

python

Copy code

# Ejemplo de código para Modelo de Machine Learning

from sklearn.model\_selection import train\_test\_split

from sklearn.neighbors import NearestNeighbors

# Preparación de datos

X = df[['budget', 'popularity', 'vote\_average']]

y = df['movie\_id']

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

# Entrenamiento del modelo

model = NearestNeighbors(n\_neighbors=5, algorithm='auto').fit(X\_train)

# Evaluación del modelo

distances, indices = model.kneighbors(X\_test)

print("Distancias: ", distances)

print("Índices: ", indices)

**Implementación de la API**

En esta sección se implementa una API utilizando Flask para permitir la interacción con el modelo de recomendación de películas. Incluye:

* **Configuración de Flask**: Creación de una aplicación Flask.
* **Rutas de la API**: Definición de endpoints para interactuar con el modelo de recomendación.
* **Despliegue de la API**: Implementación de la API en un servidor local o en la nube.

python

Copy code

# Ejemplo de código para la API

from flask import Flask, request, jsonify

app = Flask(\_\_name\_\_)

@app.route('/recommend', methods=['POST'])

def recommend():

data = request.get\_json()

movie\_features = [data['budget'], data['popularity'], data['vote\_average']]

\_, indices = model.kneighbors([movie\_features])

recommendations = df.iloc[indices[0]]['title'].tolist()

return jsonify(recommendations=recommendations)

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

**Uso**

Para ejecutar el notebook y utilizar la API, sigue estos pasos:

1. Clona este repositorio o descarga el archivo Sandagorda - Proyecto Individual 1.ipynb.
2. Abre el archivo en Jupyter Notebook:

bash

Copy code

jupyter notebook Sandagorda - Proyecto Individual 1.ipynb

1. Ejecuta las celdas del notebook en orden para reproducir el análisis y entrenar el modelo.
2. Para ejecutar la API, abre una terminal y navega hasta el directorio donde se encuentra el archivo del proyecto. Luego, ejecuta el siguiente comando:

bash

Copy code

python app.py

Esto iniciará la API en http://127.0.0.1:5000/. Puedes hacer solicitudes a la API utilizando herramientas como Postman o curl.

**Contribución**

Si deseas contribuir a este proyecto, por favor sigue estos pasos:

1. Haz un fork del repositorio.
2. Crea una rama (git checkout -b feature/nueva-funcionalidad).
3. Realiza tus cambios y haz commit (git commit -am 'Añadir nueva funcionalidad').
4. Envía tus cambios a tu rama (git push origin feature/nueva-funcionalidad).
5. Abre un Pull Request.