Movie Recommendation (Getting Started)

What is our GOAL for this MODULE?

The goal of this module is to load the data from Kaggle directly into Google Colab, understand the data and perform operations (Merging, etc.).

What did we ACHIEVE in the class TODAY?

- Loaded the dataset directly into Google Colab from Kaggle
- Prepared the data

Which CONCEPTS/CODING BLOCKS did we cover today?

- Command Line
- Kaggle API
- Google Colab
- Pandas DataFrames
- Merging

How did we DO the activities?

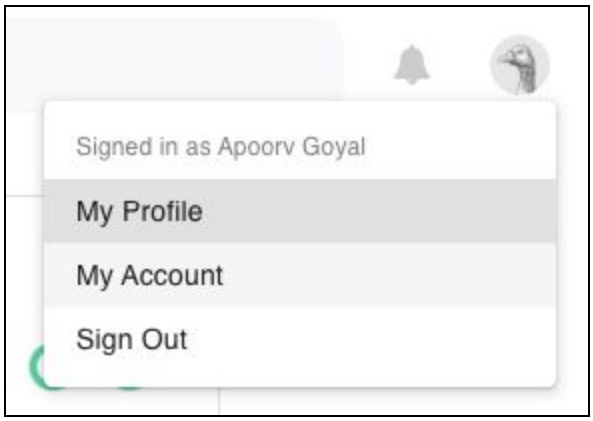
1. Understand the 3 types of filtering:
    - **Demographic Filtering**- They offer generalized recommendations to every user, based on movie popularity and/or genre. The System recommends the same movies to users with similar demographic features. Since each user is different , this approach is considered to be too simple. The basic idea behind this system is that movies that are more popular and critically acclaimed will have a higher probability of being liked by the average audience.
    - **Content Based Filtering**- They suggest similar items based on a particular item. This system uses item metadata, such as genre, director, description, actors, etc. for movies, to make these recommendations. The general idea behind these recommender systems is that if a person likes a particular item, he or she will also like an item that is similar to it.
    - **Collaborative Filtering**- This system matches persons with similar interests and provides recommendations based on this matching. Collaborative filters do not require item metadata like its content-based counterparts.
2. Go to this link. This is the data that we want to use.
3. In Colab, if we want to run terminal commands, we have to add an exclamation mark "!" before all the commands. Install Kaggle using pip.

    !pip install kaggle

```
!pip install kaggle

Requirement already satisfied: kaggle in /usr/local/lib/python3.6/dist-packages (1.5.8)
Requirement already satisfied: slugify in /usr/local/lib/python3.6/dist-packages (from kaggle) (0.0.1)
Requirement already satisfied: python-slugify in /usr/local/lib/python3.6/dist-packages (from kaggle) (4.0.1)
Requirement already satisfied: python-dateutil in /usr/local/lib/python3.6/dist-packages (from kaggle) (2.8.1)
Requirement already satisfied: certifi in /usr/local/lib/python3.6/dist-packages (from kaggle) (2020.6.20)
Requirement already satisfied: tqdm in /usr/local/lib/python3.6/dist-packages (from kaggle) (4.41.1)
Requirement already satisfied: six>=1.10 in /usr/local/lib/python3.6/dist-packages (from kaggle) (1.15.0)
Requirement already satisfied: urllib3<1.25,>=1.21.1 in /usr/local/lib/python3.6/dist-packages (from kaggle) (1.24.3)
Requirement already satisfied: requests in /usr/local/lib/python3.6/dist-packages (from kaggle) (2.23.0)
Requirement already satisfied: text-unidecode>=1.3 in /usr/local/lib/python3.6/dist-packages (from python-slugify->kaggle) (1.3)
Requirement already satisfied: chardet<4,>=3.0.2 in /usr/local/lib/python3.6/dist-packages (from requests->kaggle) (3.0.4)
Requirement already satisfied: idna<3,>=2.5 in /usr/local/lib/python3.6/dist-packages (from requests->kaggle) (2.10)
```

4. Go to kaggle's website and login to Kaggle. We want to generate credentials to be able to use their API. Click on the top right corner of the screen and select "My Account".



5. Scroll down to the API section and click **Create New API Token**.
6. Upload the json file that is automatically downloaded into Google Colab:

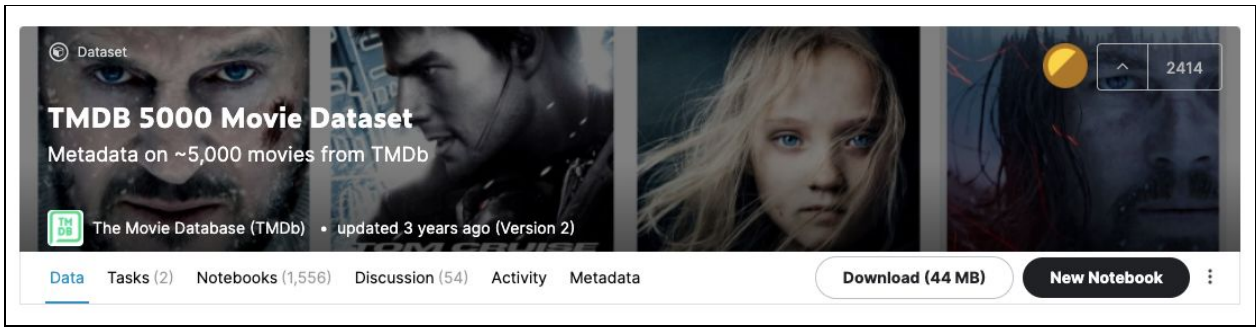
```
from google.colab import files
files.upload()
```

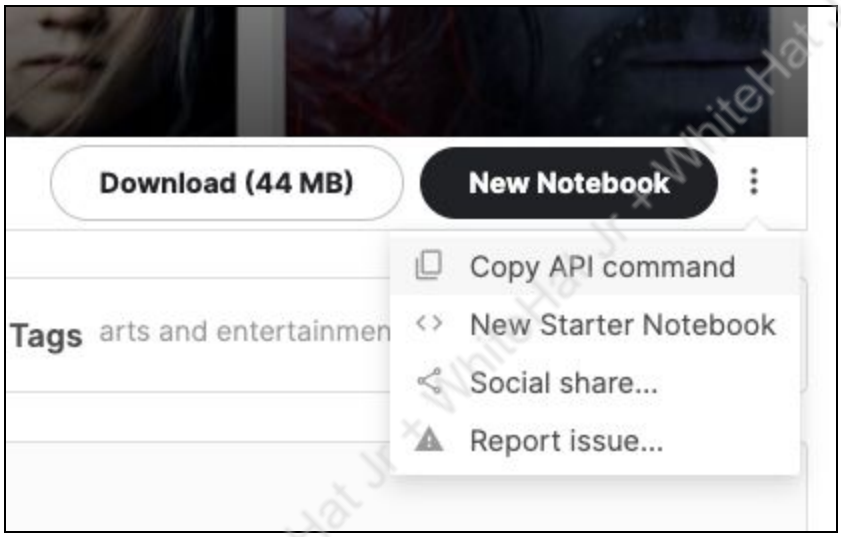
7. Run the following commands in your Colab to make sure there are no permission errors.

```
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/

!chmod 600 ~/.kaggle/kaggle.json
```

8. Now to download the data, click on the 3 dots here on this link, which contains TMDB's data for 5000 movies.

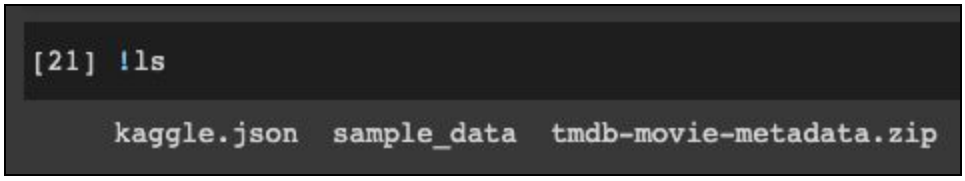




9. Paste the command in google colab. Do not forget the exclamation mark "!" before the command.

```
!kaggle datasets download -d tmdb/tmdb-movie-metadata
```

10. Run the "**!ls**" command. The output should be the following:



11. We have it in .zip format. Unzip it with the following command and cross check with

the !ls command.

```
!unzip tmdb-movie-metadata.zip
```

```
[22] !unzip tmdb-movie-metadata.zip

     Archive:  tmdb-movie-metadata.zip
       inflating: tmdb_5000_credits.csv
       inflating: tmdb_5000_movies.csv

[24] !ls

     kaggle.json  tmdb_5000_credits.csv  tmdb-movie-metadata.zip
     sample_data  tmdb_5000_movies.csv
```

12. Load the CSV files and Pandas DataFrame and print the head of both the files.

```
import pandas as pd
import numpy as np

df1=pd.read_csv('tmdb_5000_credits.csv')
df2=pd.read_csv('tmdb_5000_movies.csv')
```

```
df1.head()
```

```
df2.head()
```

13. The output should be like:

```
[26] df1.head()
```

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	49529	John Carter	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...

```
[27] df2.head()
```

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615
		[{"id": 28,			[{"id": 470,			A cryptic	

14. Study the data.
    - The first dataset contains the following features:
        - **movie_id** - A unique identifier for each movie.
        - **cast** - The name of lead and supporting actors.
        - **crew** - The name of Director, Editor, Composer, Writer etc.
    - The second dataset has the following features:
        - **budget** - The budget in which the movie was made.
        - **genre** - The genre of the movie, Action, Comedy ,Thriller etc.
        - **homepage** - A link to the homepage of the movie.
        - **id** - This is in fact the movie_id as in the first dataset.
        - **keywords** - The keywords or tags related to the movie.
        - **original_language** - The language in which the movie was made.
        - **original_title** - The title of the movie before translation or adaptation.
        - **overview** - A brief description of the movie.
        - **popularity** - A numeric quantity specifying the movie popularity.
        - **production_companies** - The production house of the movie.
        - **production_countries** - The country in which it was produced.
        - **release_date** - The date on which it was released.
        - **revenue** - The worldwide revenue generated by the movie.
        - **runtime** - The running time of the movie in minutes.
        - **status** - "Released" or "Rumored".
        - **tagline** - Movie's tagline.
        - **title** - Title of the movie.
        - **vote_average** - average ratings the movie received.
        - **vote_count** - the count of votes received.
15. Merge the 2 datasets into one dataframe.

```
df1.columns = ['id','tittle','cast','crew']
df2= df2.merge(df1,on='id')
```

```
df2.head(5)
```

16. Our data is good to go now!

What's NEXT?

In the next class, we will work on the Demographic Filtering with this data.

EXTEND YOUR KNOWLEDGE:

You can read the following blog on speed of our planet to understand more:
https://medium.com/towards-artificial-intelligence/recommendation-systems-104bdfe3f93f