Topic	Movie Recommendation (Getting Started)	
Class Description	**Students will load the data and spend some time understanding it, pre-processing and merging it.**	
Class	**C138**	
Class time	**45 mins**	
Goal	• Loading the data into google colab directly from kaggle • Understanding the data • Merging, preprocessing and cleaning it	

Resources Required	• Teacher Resources
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	  ○ Google Colab
	• Student Resources
	  ○ Laptop with internet connectivity
	  ○ Earphones with mic
	  ○ Notebook and pen
	  ○ Google Colab	

Class structure	**Warm Up** **Teacher-led Activity** **Student-led Activity** **Wrap up**	**5 mins** **25 min** **10 min** **5 min**

CONTEXT

- **Review the concepts learned in the earlier classes**

Class Steps	Teacher Action	Student Action
Step 1: Warm Up (5 mins)	Hi <Student Name>! We completed our Exo-Planet catalog last class! We learned a number of fun facts about planets in general too! Now, it's time for us to do something	**ESR:** “Yes!”

	new! In this module, we will be building a movie recommendation system! Are you excited?	
	Before we proceed I have an exciting quiz question for you! Are you ready to answer this question? Teacher click on the  button on the bottom right corner of your screen to start the In-Class Quiz. A quiz will be visible to both you and the student. Encourage the student to answer the quiz question. The student may choose the wrong option, help the student to think correctly about the question and then answer again. After the student selects the correct option, the  button will start appearing on your screen. Click the End quiz to close the quiz pop-up and continue the class.	**ESR:Yes**
	Alright! let's get started now.	
Teacher Initiates Screen Share		

CHALLENGE • **Loading data to Google Colab from Kaggle**		
Step 2: Teacher-led Activity (25 min)	The rapid growth of data collection has led to a new era of information. Data is being used to create more efficient systems and this is where Recommendation Systems come into play. Recommendation Systems are a type of **information filtering system** as they improve the quality of search results and provide items that are more relevant to the search item or are related to the search history of the user.	
	They are used to predict the **rating** or **preference** that a user would give to an item. Almost every major tech company has applied them in some form or the other: Amazon uses it to suggest products to customers, YouTube uses it to decide which video to play next on autoplay, and Facebook uses it to recommend pages to like and people to follow. Moreover, companies like Netflix and Spotify depend highly on the effectiveness of their recommendation engines for their business and success.	
	You must have seen a few recommendation algorithms yourself! Can you guess how the filtering happens?	**ESR:** Varied

	Great! There are basically three types of recommender systems: ● **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.	
	Let's jump into work now! First thing is that we are going to deal with big datasets. It would be a pain to download them first and then re-upload them! How could we directly add the dataset in a google colab from Kaggle?	**ESR:** Varied
	In this project, we will be using the following dataset from Kaggle! https://www.kaggle.com/tmdb/tmdb-movie-metadata This link contains TMDB's data for 5000 movies. The question still comes: How do we upload this directly into google colab? The first thing that we want to do is install the dependencies! For this, let's first create a new notebook in Google Colab. In the new notebook, the first thing that we want to do is to **pip install kaggle**.	

Since we want to run a terminal command in Google Colab, we need to add an “!” in the beginning of the command.

Now, the command becomes

!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)
```

Great! Now, we want to upload Kaggle’s credentials into the google colab.

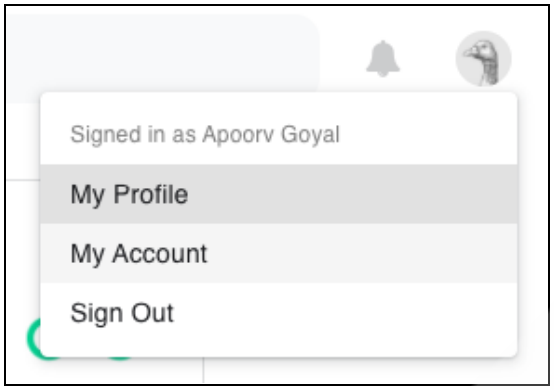
```
from google.colab import files
files.upload()
```

But, where are the credentials?

You will have to download them!

First thing first, login into Kaggle.

After logging in, go to the “My Account” page. (from top right of the page)



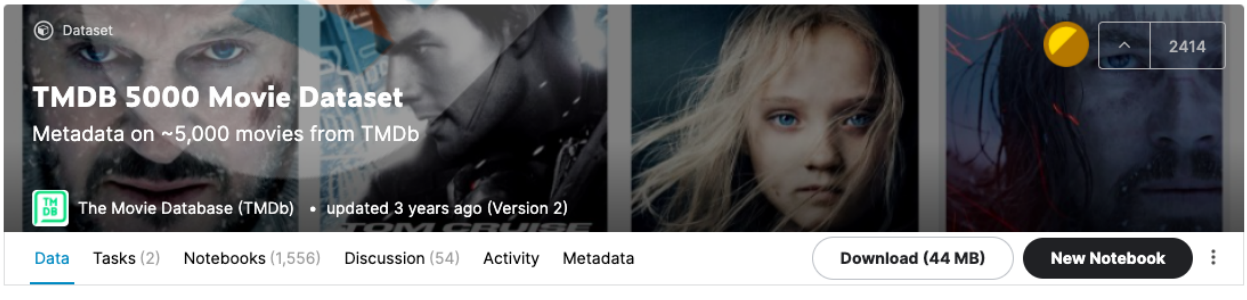
Here, scroll down a bit until you reach the API section and from there, click on **Create New API Token. (or see under the Account tab)**

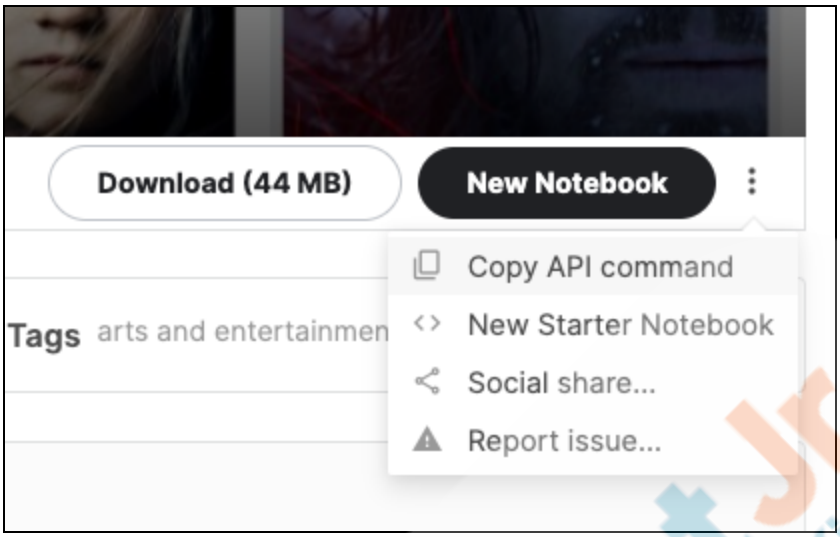
Upload this file (**kaggle.json**) into your Google Colab!

So far so good! Now, lets run a set of commands in our colab as follows

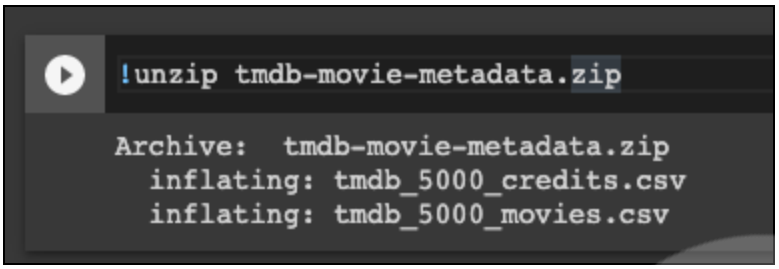
```
!mkdir -p ~/.kaggle
!cp kaggle.json ~/.kaggle/

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

Here, we are first creating a directory **.kaggle** (we have the dot so that it is a hidden directory.)

	We are then copying the kaggle.json file into this directory with the **cp** command. Finally, with the **chmod** command, we are making changes to the permission of the file so that it does not fail. Remember how many times we have had permission errors in our programs? We are now all set to proceed further and add the datasets!	
	For this, we will go to the dataset and click on the 3 dots we have next to the New Notebook button. Then select the Copy API command option. The dataset is available here https://www.kaggle.com/tmdb/tmdb-movie-metadata	
		



	Paste this command into the Google Colab. Make sure to add an exclamation "!" in the beginning of the command to make it work. **!kaggle datasets download -d tmdb/tmdb-movie-metadata**	
	Let's do the **!ls** command to see how stuff looks like in here.	
	Here, we have the data in zip files. We need to extract the data! For this, we will use the following command! `!unzip tmdb-movie-metadata.zip`	



	Great! Now let's wait for the data to download and we are all set and ready to go!	
Teacher Stops Screen Share		
	Now it's your turn. Please share your screen with me.	

- **Ask Student to press ESC key to come back to panel**
- **Guide Student to start Screen Share**
- **Teacher gets into Fullscreen**

ACTIVITY

- **Student will understand the data**
- **Student will pre-process and merge the data**

Step 3: Student-Led Activity (10 min)	Let's first load the tmdb data in pandas dataframe and study it!
	
	`import pandas as pd`
	`import numpy as np`
	
	`df1=pd.read_csv('tmdb_5000_credits.csv')`
	`df2=pd.read_csv('tmdb_5000_movies.csv')`
	
	Now, let's print the head of these Dataframes!
	
	`df1.head()`
		Student writes code to read CSV as Dataframe and print the head file, then spends time with the teacher to understand the data.

```
df2.head()
```

```
[12] import pandas as pd
     import numpy as np

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

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

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe480092514l6c750aca23", "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...

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

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popu
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
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

Let's understand 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 student writes code to create a red square sprite and control it using right and left arrow keys.

	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.	
	We now need to join these two datasets! Can you say how we can join them?	**ESR:** We can join them with an ID.

	Great! Let's quickly do that!
	
	`df1.columns = ['id','tittle','cast','crew']`
	`df2= df2.merge(df1,on='id')`
	
	`df2.head(5)`
	
	Here, we are using the **merge()** function and telling it to merge df1 with df2 on id.
		The student joins the two DataFrames.

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

df2.head(5)
```

	budget	genres	homepage	id	keywords	original_language	original_title	overview
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...
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...

	Great! Now, it looks like our data is ready!	

Teacher Guides Student to Stop Screen Share

FEEDBACK

- **Appreciate the student for their efforts**
- **Identify 2 strengths and 1 area of progress for the student**

Step 4: Wrap-Up (5 min)	So, in this project class we saw how we can load big datasets directly from Kaggle into Google Colab. Although our dataset wasn't big enough, many times, you might be dealing with larger datasets that might have GBs of data. By following this method, all that data can be imported into Google Colab in Minutes (or even seconds) since it will not use your internet but instead would use the internet of the Colab machine.	**ESR:** varied

	We then merged our 2 CSVs into one. How was your experience for this class?	
	Amazing! In the next class, we will do the demographic filtering of our data and learn about the Weighted Rating Method.	-
Teacher Clicks	✖ End Class	

Activity	Activity Name	Links
Teacher Activity 1	Solution	https://colab.research.google.com/gist/shubhamwhj/5b8a76eb35963c2294a0ad5f9051bc16/copy-of-pro-c138-reference-code.ipynb