

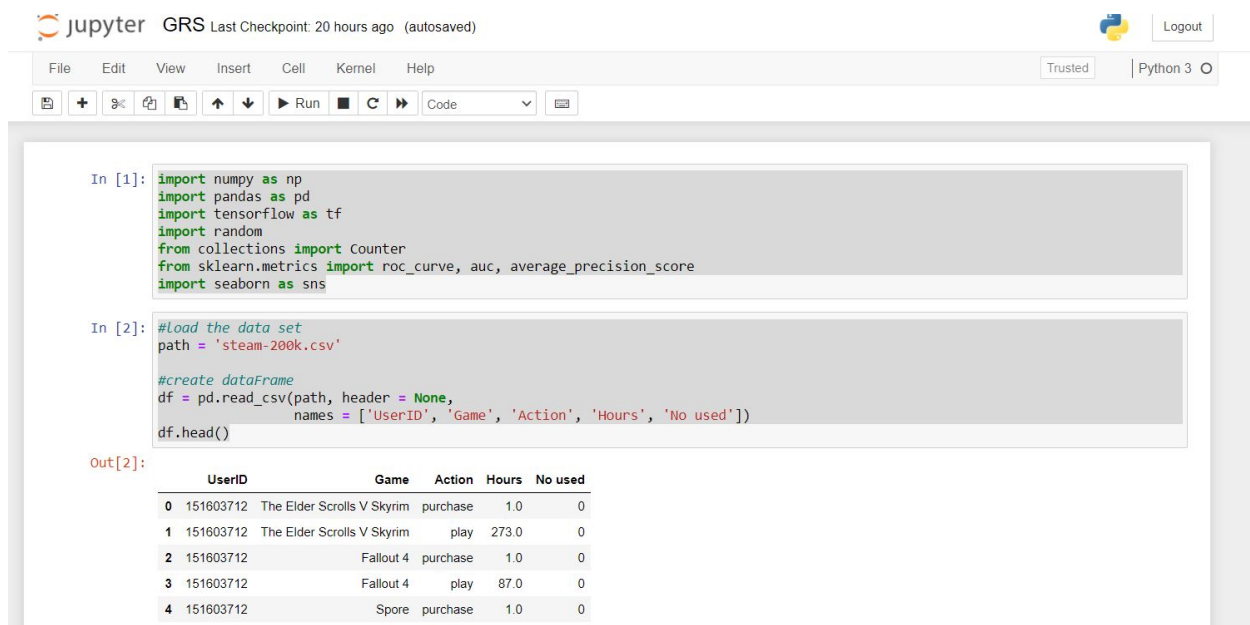
Game Recommendation System by Yun-Ting Chen

In order to run the program and get the result properly, the following items are required:

1. jupyter notebook
2. Steam 200k.csv dataset

Steps:

1. Open the GRS.ipynb file via jupyter notebook



```
In [1]: import numpy as np
import pandas as pd
import tensorflow as tf
import random
from collections import Counter
from sklearn.metrics import roc_curve, auc, average_precision_score
import seaborn as sns

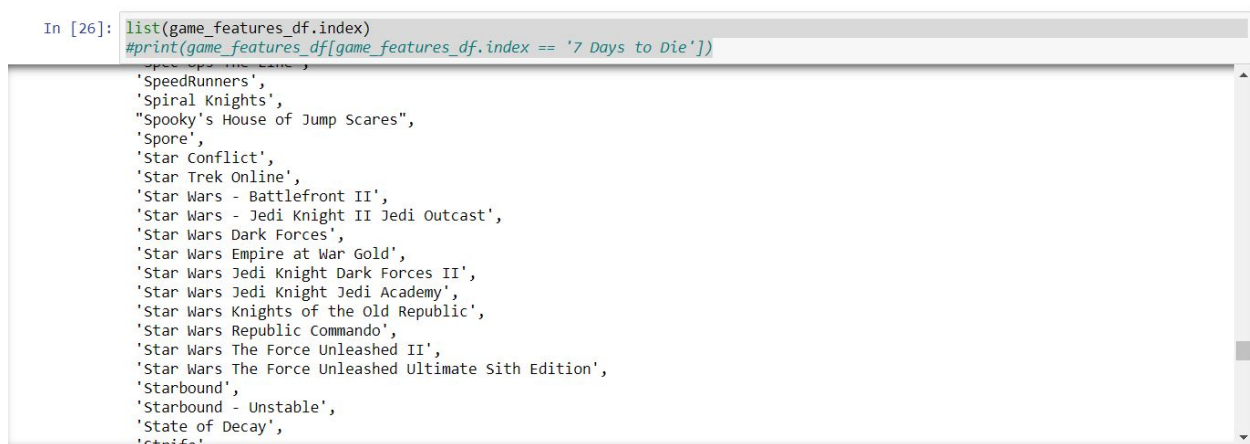
In [2]: #Load the data set
path = 'steam-200k.csv'

#create dataframe
df = pd.read_csv(path, header = None,
                names = ['UserID', 'Game', 'Action', 'Hours', 'No used'])
df.head()
```

Out[2]:

	UserID	Game	Action	Hours	No used
0	151603712	The Elder Scrolls V Skyrim	purchase	1.0	0
1	151603712	The Elder Scrolls V Skyrim	play	273.0	0
2	151603712	Fallout 4	purchase	1.0	0
3	151603712	Fallout 4	play	87.0	0
4	151603712	Spore	purchase	1.0	0

2. Put the steam 200k.csv dataset under the same folder for the project
3. Run through every single code in the file
4. To change the recommended games based on different games as input



```
In [26]: list(game_features_df.index)
#print(game_features_df[game_features_df.index == '7 Days to Die'])
```

'SpeedRunners',
'Spiral Knights',
'Spooky's House of Jump Scares',
'Spore',
'Star Conflict',
'Star Trek Online',
'Star Wars - Battlefront II',
'Star Wars - Jedi Knight II Jedi Outcast',
'Star Wars Dark Forces',
'Star Wars Empire at War Gold',
'Star Wars Jedi Knight Dark Forces II',
'Star Wars Jedi Knight Jedi Academy',
'Star Wars Knights of the Old Republic',
'Star Wars Republic Commando',
'Star Wars The Force Unleashed II',
'Star Wars The Force Unleashed Ultimate Sith Edition',
'Starbound',
'Starbound - Unstable',
'State of Decay',
'Strife'

Look at the list in the game_features_df to find a desired game

5. Copy and paste the game title to the "find" variable to change the game title input

```
In [51]: query_index = 0
         find = "Star Wars - Battlefront II"
         print(query_index)
         #distances, indices = model_knn.kneighbors(game_features_df.iloc[query_index,:].values.reshape(1, -1), n_neighbors = 6)
         distances, indices = model_knn.kneighbors(game_features_df[game_features_df.index == find].values.reshape(1, -1), n_neighbors=6)

0
```

*example, in this case, is find = "Star Wars - Battlefront II"

6. Run the code again

7. The recommended games result will show up at the end

```
for i in range(0, len(distances.flatten())):
    if i == 0:
        print('Recommendations for {0}:\n'.format(find))
    else:
        print('{0}: {1}, with distance of {2}:\n'.format(i, game_features_df.index[indices.flatten()[i]], distances.flatten()[i]))
```

Recommendations for Star Wars - Battlefront II:

```
1: Star Wars Republic Commando, with distance of 0.5542421182357637:
2: Star Wars Empire at War Gold, with distance of 0.7119411732663936:
3: Star Wars Jedi Knight Jedi Academy, with distance of 0.7349072331365503:
4: Star Wars Dark Forces, with distance of 0.7455887892533721:
5: STAR WARS Knights of the Old Republic II The Sith Lords, with distance of 0.7643803949493543:
```