user_filter.py:
output->allUser_filtered.csv
select useful data, i.e. items released after the user join date.
output format: <uid, like_date, release_date, page_url>

user_avg.py:
input -> allUser_filtered.csv
output-> user_avg_td.csv
calculate the average time_diff for each user
output format: <uid, time_diff>

allUser_like.py:
input -> allUser_filtered.csv
output-> allUser_like.csv
select the tuple from user data
output format: <uid, page_url, like_date>

fri_pair.py:
output -> friend_pairs.csv
find all pairs of friends
output format: <uid1, uid2>

fri_pair_like.py/fri_pair_like_unfilter.py:
input -> friend_pairs.csv , allUser_like.csv/allUser_like_unfilter.csv
output -> fri_pair_td.csv
find the common items of each pair, and then select those meeting our requirements, say time_diff within 3 days. Calculate the average time_diff for each pair, sort them and draw figure.
output format: <uid1, uid2, time_diff>

draw_user_td.py:
input -> user_avg_td.csv
read in the data, sorted by the time_diff descending, and hash the user id, draw the figure.

pair_requirement_ratio.py/pair_requirement_ratio_unfilter.py:
input -> friend_pairs.csv , allUser_like.csv/allUser_like_unfilter.csv
for each pair, the items in common:

    required common items/ total common items

then draw the figure.

new

neighbor_prob.py:

input -> fri_pair_td.csv
read in all the pair time_diff, for each user, get a list like [uid] : {<f1,1/t1>,...,<fn,1/tn>}, reverse the time difference since the larger the time difference, the smaller probability.

two_proportion.py/two_portion_unfilter.py:
input -> friend_pairs.csv , allUser_like.csv/allUser_unfilter.csv
output->user_proportion.csv
calculate :
each pair: influenced/total common
Jaccard coefficient

each user: influenced/ user's list of items

draw_user_proportion.py:
input->user_proportion.csv
draw the figure

friend_pair_score.py
input->fri_pair_td.csv, friend_pair_Jaccard.csv
draw figure:
1/t represent the score, and for friend pairs, normalized as ti/sqrt(t1^2+t2^2+...)
x-axis represents friend pairs
red is normalized score and blue is Jaccard Coefficient

movie_avg.py:
input->allUser+movie_filtered.csv
for certain time period , say half a year, show relationship between the item released time and like_date

movie_filter.py:
input->django_facebook_facebookfriendlike.csv, django_facebook_facebooklike.csv
output->allUser+movie_filtered.csv
choose only items that are released after the user's join date

user_cluster.py:
input->django_facebook_facebookfriendlike.csv
output->user.arff
preprocess the data file and put into the format of <user id: time_diff>, which can be a standard input of WEKA

item_cluster.py:

input->django_facebook_facebookfriendlike.csv
output->item.arff
preprocess the data file and put into the format of <item id: time_diff>, which can be a standard input of WEKA

show_distribution.py:
input->django_facebook_facebooklike.csv
show the bar figure of the distribution of items likes where x-axis represents the time difference periods (say 5, 10,...) , and y-axis represents the number of items that belong to this period

show_release_relationship.py:
input->django_facebook_facebookfriendlike.csv
show the relationship between the time difference and release date where x-axis represents the release_date for the item, and y-axis represents the time_diff for this item