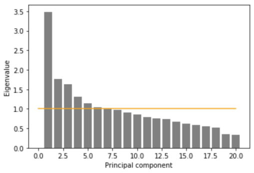
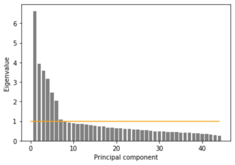
Becky Bian
lb3622@nyu.edu

1. Data preprocessing

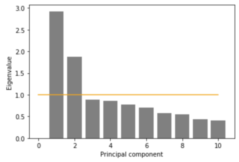
In this project, we can observe that the dataset has two types of features: movie ratings and other subjective survey responses. Because each column under the movie rating group represents a distinct movie, which is independent from other columns under the same group whereas those subjective survey response attributes can be grouped into sensation seeking, movie experiences, we decide to pre-process these attributes using different methods separately. To start with the movie rating columns, because the **missing value** in the movie rating column indicates people who didn't view this movie, we decided to drop rows with missing values. However, since the number of missing values in each movie is slightly different and the operation to drop all of them at the beginning will delete too much data, we decided to drop missing values in the later analysis where the specific movie name is specified. In terms of the columns under factor loadings such as sensation seeking, we observed that the number of missing values in each column is relatively small (approximately 15) relative to the number of responses in total, and hence we decide to use attribute median to replace the null values in each column since it can also help us to prevent deleting too many rows by deleting rows for each column. In addition, since the value -1 in the column of "only child" and "social viewership preference" indicates people who didn't respond to the question, we only treat it as missing values and deleting rows with value of -1 corresponding to these columns. After dealing with the missing values, because the attributes are in different scales, we then manage to **standardize** all data with the use of stats.zscore(). Finally, whereas the movie rating column does not require further processing except the missing value deletion in later analysis, we found that there are too many variables under each factor loadings such as sensation seeking (20), personality types (44). Hence, we decided to use **PCA** to conduct **dimensionality reduction**. After using the PCA function imported from sklearn.decomposition, I managed to decide the number of principal components for the data of sensation seeking, personality type, and movie rating as 6, 7, and 2 respectively according to the Kaiser Criterion.



```
Number of factors selected by Kaiser criterion for sensation seeking: 6
```



```
Number of factors selected by Kaiser criterion for personality: 7
```



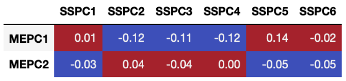
```
Number of factors selected by Kaiser criterion for movie experience: 2
```

After determining the number of principal components and conducting PCA based on the component numbers, we then obtain a new dataset containing the transformed data for the newly-constructed principal components for each factor loadings; merging them with the scaled movie rating dataset, I can generate a relatively clean dataset for later further analysis.

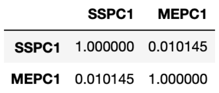
2. What is the relationship between sensation seeking and movie experience?

For this question, because the sensation seeking group still has 6 principal components whereas the movie experience group still has 2 principal components after the **PCA**, I decided to draw a correlation matrix for the table containing all these 8 columns and select part of them (the part which only contains correlation between SS and ME without those comparing factors within the same group) as the heatmap shown below to examine the relationship between two groups. According to the heatmap, it can be observed that the relationship between these two groups is relatively low as the highest correlation between any two factors from each group only reaches 0.136 (in red box) whereas other correlation coefficients are relatively close to 0 (ie. 0.01, -0.05, -0.11). Hence, we can conclude that sensation seeking and movie experience has a low correlation overall.

In addition, to further solidify our finding, we also try to reduce the dimensionality of both sensation seeking and movie experience from original data into 1 principal components respectively, computing their correlation as 0.010145, which indicates a very low correlation and hence supports our finding from the heatmap above that there's no significant relationship between the two loadings.



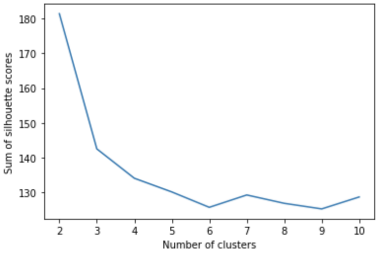
	SSPC1	SSPC2	SSPC3	SSPC4	SSPC5	SSPC6
MEPC1	0.01	-0.12	-0.11	-0.12	0.14	-0.02
MEPC2	-0.03	0.04	-0.04	0.00	-0.05	-0.05

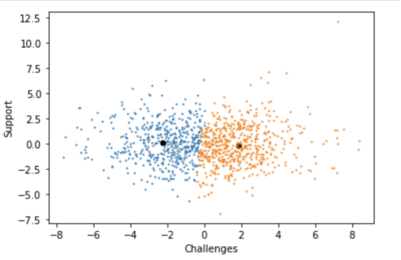


	SSPC1	MEPC1
SSPC1	1.000000	0.010145
MEPC1	0.010145	1.000000

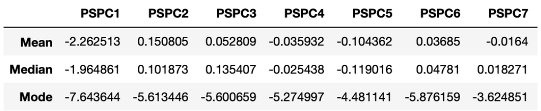
3. Is there evidence of personality types based on the data of these research participants? If so, characterize these types both quantitatively and narratively.

As we need to determine if there's personality types and there's no labels on our data whereas the data also contains many features, I decided to use **k-means clustering** algorithm here to help with the task. Since the personality type data has been standardized and dimensionally reduced by PCA, we can directly conduct clustering algorithms on the data. With the use of both **Elbow and Silhouette methods** (below plot shows Silhouette method), we find that the **optimal number of clusters** should be 2 (set k=2). After conducting the 2-means clustering on the dataset, we manage to conduct another PCA to construct 2 principal components for the personality type data so that we can plot it according to the clusters in a 2-d scatter plot, shown below.

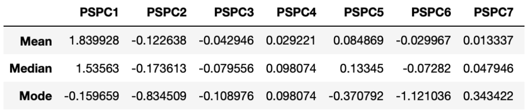




In addition, I also separate the dataset into two according to their clusters, computing the mean, median, and mode for each of the original 7 principal components so that we can have a better illustration about the characteristics of each attribute in different clusters. The table below just shows the brief information about each attribute in the 2 clusters.

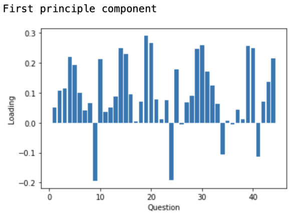


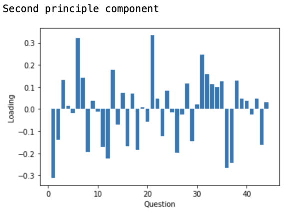
	PSPC1	PSPC2	PSPC3	PSPC4	PSPC5	PSPC6	PSPC7
Mean	-2.262513	0.150805	0.052809	-0.035932	-0.104362	0.03685	-0.0164
Median	-1.964861	0.101873	0.135407	-0.025438	-0.119016	0.04781	0.018271
Mode	-7.643644	-5.613446	-5.600659	-5.274997	-4.481141	-5.876159	-3.624851



	PSPC1	PSPC2	PSPC3	PSPC4	PSPC5	PSPC6	PSPC7
Mean	1.839928	-0.122638	-0.042946	0.029221	0.084869	-0.029967	0.013337
Median	1.53563	-0.173613	-0.079556	0.098074	0.13345	-0.07282	0.047946
Mode	-0.159659	-0.834509	-0.108976	0.098074	-0.370792	-1.121036	0.343422

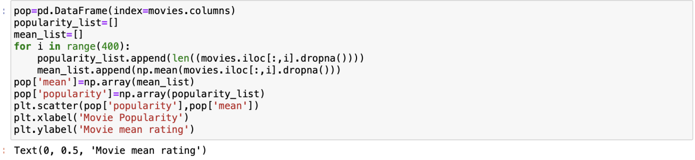
I then characterize the 2 clusters by interpreting the loading graph for the first 2 principal components after PCA. It can be indicated from the below figure that our first cluster can be characterized as relatively optimistic and curious whereas the second cluster can be characterized by a relatively pessimistic personality.





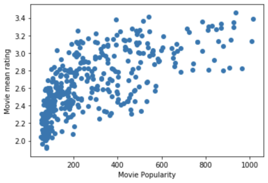
4. Are movies that are more popular rated higher than movies that are less popular?

In this case, to determine if more popular movies are rated higher than movies that are less popular, we plan to figure out and reflect on the correlation between the movie popularity and their mean rating across the 400 movies we have. To quantify the popularity of a movie, we use the number of people who have watched the movies as an indicator (the number of ratings, which is the number of rows without null values in each column). Hence, by looping across the original 400 movies, we managed to generate a new table with the mean rating and popularity of each movie stored. Observing the scatterplot for the popularity and mean rating column in this new table shown below, we found that there is a positive relationship between the two variables whereas their correlation of 0.699161 also tells the same signal that a positive relationship exists between the movie popularity and mean ratings.

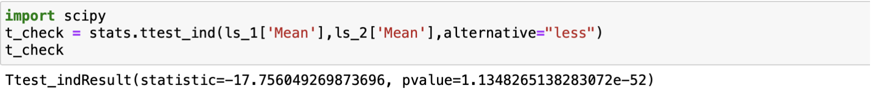


```
pop=pd.DataFrame(index=movies.columns)
popularity_list=[]
mean_list=[]
for i in range(400):
    popularity_list.append(len((movies.iloc[:,i].dropna())))
    mean_list.append(np.mean(movies.iloc[:,i].dropna()))
pop['mean']=np.array(mean_list)
pop['popularity']=np.array(popularity_list)
plt.scatter(pop['popularity'],pop['mean'])
plt.xlabel('Movie Popularity')
plt.ylabel('Movie mean rating')
```

```
Text(0, 0.5, 'Movie mean rating')
```



On the other hand, to further solidify our conclusion, we also conduct an independent **t-test** on the dataset with numeric movie mean ratings and popularity. To prepare for the test, we firstly computed the median of the movie popularity which is 197.5 and separated the original datasets into two separate tables based on the comparison result with the popularity median. This means that we define popularity median as the threshold to tell which movies are more popular on average and which are less popular on average, obtaining two independent tables which contain movies based on their popularity category as well as their mean ratings computed before as our two prepared independent samples of our t-test. In terms of the t-test setup, we set the null hypothesis as there's no difference on the movie ratings between more popular and less popular movies whereas the alternative hypothesis is that more popular movies tend to be rated higher on average than less popular ratings (less popular movies tend to be rated lower than more popular ratings on average). In addition , we also set the significance level alpha as 0.05 as generally used. Hence, we managed to compute the p-value of the 1-sided t-test with the use of stats.ttest_ind(alternative="less") (as we put less popular movie in the first argument), finally obtaining the p-value as 1.1348265138283072e-52, much lower than 0.05; hence, we are able to reject the hypothesis that there's no difference on the movie ratings between more popular and less popular movies and therefore there's sufficient evidence to tell that less popular movies tend to rate lower than more popular movies and hence it can conclude that more popular ratings tend to be rated higher than less popular ratings.

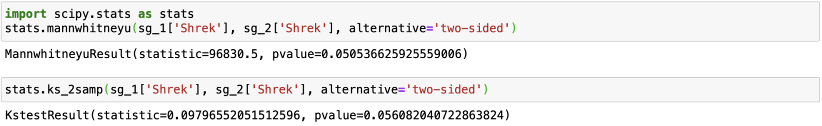


```
import scipy
t_check = stats.ttest_ind(ls_1['Mean'],ls_2['Mean'],alternative="less")
t_check
```

```
Ttest_indResult(statistic=-17.756049269873696, pvalue=1.1348265138283072e-52)
```

5. Is enjoyment of 'Shrek (2001)' gendered, i.e. do male and female viewers rate it differently?

In order to determine if males and females rate the movie "Shrek" differently, I decided to use a two-sided hypothesis testing, and quantify the movie enjoyment with rating median for rating variables, finally selecting the Mann–Whitney U test since the rating variable does not follow a normal distribution. Before conducting the test, I firstly extract the column of "Shrek" rating and gender into a new data set. Then, I manage to address some remaining missing values on the preprocessed data: because the gender column has been preprocessed at the beginning, we only need to focus on the missing values in the movie rating column for "Shrek". Due to the fact that null values in the movie rating column indicate people who didn't view the movie, I then managed to drop all null values in the data set which was standardized before. In terms of the set up of the Mann–Whitney U test, we set the null hypothesis as the two populations of female and male are equal on their rating median on the movie "Shrek" (male and female rating come from population with the same median) whereas the alternative hypothesis is that there's a difference on the movie rating between male and female viewers on "Shrek" (indicating that the movie rating is gendered). In addition, we also set the significance level alpha at 0.05 as generally used. After transforming the original dataset into two separate datasets based on the gender of viewers (sg_1 only contain female viewers while sg_2 only contain male viewers), we managed to compute the p-value of the **Mann-Whitney U tes**t with the use of stats.mannwhitneyu(sg_1['Shrek'], sg_2['Shrek'], alternative='two-sided'), finally obtaining the p-value as 0.050536625925559006, higher than 0.05; hence, we fail to reject the null hypothesis that there's no difference on movie rating median for "Shrek" across the gender of viewers and therefore **there's no sufficient evidence to tell that enjoyment of "Shrek" is gendered**. In addition, the conduction of **Kolmogoriv-Smirnov test** which is generally used to test whether the two samples come from the same population also generate a p-value of 0.056082040722863824, which is higher than 0.05 and further solidify our conclusion that there's no sufficient evidence to tell that enjoyment of "Shrek" is gendered.



```
import scipy.stats as stats
stats.mannwhitneyu(sg_1['Shrek'], sg_2['Shrek'], alternative='two-sided')
```

```
MannwhitneyuResult(statistic=96830.5, pvalue=0.050536625925559006)
```

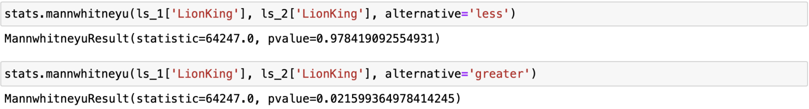
```
stats.ks_2samp(sg_1['Shrek'], sg_2['Shrek'], alternative='two-sided')
```

```
KstestResult(statistic=0.09796552051512596, pvalue=0.056082040722863824)
```

6. Do people who are only children enjoy 'The Lion King (1994)' more than people with siblings?

In order to determine if people who are only children enjoy the movie "The Lion King" more than people with siblings, I decided to use a one-sided hypothesis testing, and quantify the movie enjoyment with rating median for rating variables, finally selecting the Mann–Whitney U test as rating variable does not follow a normal distribution. Before conducting the test, I firstly extract the column of "Lion King" rating and only child into a new data set. Then, I manage to address some remaining missing values on the preprocessed data: because the only child column has been preprocessed at the beginning, we only need to focus on the missing values in the movie rating column for "Lion King". In addition, because of the fact that null values in movie rating column indicate people who didn't view the movie, I managed to drop all null values in the movie rating column which was standardized before. In terms of the set up of the **Mann–Whitney U test**, we set the null hypothesis as the two populations of viewers with siblings and without siblings are equal on their rating median on the movie "Lion King" (only child and non-only child ratings come from population with the same median) whereas the alternative hypothesis is that people who are only child rate the movie "Lion King" higher than people with siblings and hence its population has a higher movie rating median (non-only child rate the movie lower than only child). In addition, we also set the significance level alpha at 0.05 as generally used. After transforming the original dataset into two separate datasets based on the sibling status of viewers (ls_1 only contain viewers with siblings while ls_2 only contain viewers without siblings), we managed to compute the p-value of the Mann-Whitney U test with the use of stats.mannwhitneyu(ls_1['LionKing'], ls_2['LionKing'], alternative='less') (as the first argument I input is non-only child), finally obtaining the p-value as 0.978419092554931, higher than 0.05; hence, we are unable to reject the null hypothesis that there's no difference on movie rating median for "Lion King" across the "only child" status of viewers and therefore **there's no sufficient evidence to tell that enjoyment of "Lion King" is higher for the only child viewers.**

On the other hand, we use the stats.mannwhitneyu(ls_1['LionKing'], ls_2['LionKing'], alternative='greater') (as the first argument I input is non-only child) to test the case when the alternative hypothesis is that people who are not only child rate the movie "Lion King" higher than people with siblings and hence its population has higher rating median. In this case, we obtain a p-value of 0.021599364978414245, much lower than 0.05; hence, we are able to reject the null hypothesis that there's no difference on movie rating median for "Lion King" across the "only child" status of viewers and therefore there's sufficient evidence to tell that enjoyment of "Lion King" is higher for the viewers with siblings. As a result, it can be suggested that although no sufficient evidence can tell that people who are only children enjoy 'The Lion King (1994)' more than people with siblings, we have strong evidence to tell that **viewers with siblings enjoy the movie more**.



```
stats.mannwhitneyu(ls_1['LionKing'], ls_2['LionKing'], alternative='less')
```

```
MannwhitneyuResult(statistic=64247.0, pvalue=0.978419092554931)
```

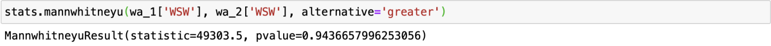
```
stats.mannwhitneyu(ls_1['LionKing'], ls_2['LionKing'], alternative='greater')
```

```
MannwhitneyuResult(statistic=64247.0, pvalue=0.021599364978414245)
```

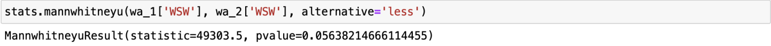
7. Do people who like to watch movies socially enjoy 'The Wolf of Wall Street (2013)' more than those who prefer to watch them alone?

In order to determine if people who like to watch movies socially enjoy the movie "The Wolf of Wall Street" more than people who prefer to watch movies alone, I decided to use a one-sided hypothesis testing, and quantify the movie enjoyment with rating median for rating variables, finally selecting **Mann–Whitney U test** since rating variable does not follow a normal distribution. Before conducting the test, I firstly extracted the columns of "The Wolf of Wall Street" rating and "movie enjoyment alone" into a new data set. Then, I manage to address some remaining missing values on the preprocessed data: because the "social viewership preference" column has been preprocessed at the beginning, we only need to focus on the missing values in the movie rating column for "Wolf of Wall Street". In addition, because of the fact that null values in the movie rating column indicate people who didn't view the movie, I managed to drop all null values in the movie rating column. In terms of the set up of the **Mann–Whitney U test**, we set the null hypothesis as the two populations of viewers who like to watch movies socially and viewers who don't are equal on their rating median on the movie "Wolf of Wall Street" (two rating samples come from population with the same median) whereas the alternative hypothesis is that people who like to watch movies socially rate the movie "Wolf of Wall Street" higher than people who don't and its population has higher rating median. In addition, we also set the significance level alpha at 0.05 as generally used. After transforming the original dataset into two separate datasets based on the movie social enjoyment status of viewers (wa_1 only contain viewers who like to watch movies socially while wa_2 only contain viewers who enjoy movies alone), we managed to compute the p-value of the **Mann-Whitney U test** with the use of stats.mannwhitneyu(wa_1['WSW'], wa_2['WSW'], alternative='greater') (as the first argument I input is people who enjoy socially), finally obtaining the p-value as 0.9436657996253056, higher than 0.05; hence, we fail to reject the null hypothesis that there's no difference on movie rating median of "Wolf of Wall Street" across "movie enjoyment alone" status of viewers and therefore there's no sufficient evidence to tell that enjoyment of "Wolf of Wall Street" is different depending on the "movie enjoyment alone" preferences of viewers. As a result, **there's no sufficient evidence to suggest that people who like to watch movies socially enjoy 'The Wolf of Wall Street (2013)' more than those who prefer to watch them alone**.

On the other hand, we use the stats.mannwhitneyu(wa_1['WSW'], wa_2['WSW'], alternative='less') (as the first argument I input is people who enjoy socially) to test the case when the alternative hypothesis is that people who enjoy watching alone rate the movie "Wolf of Wall Street" higher than people who enjoys watching movie socially and its population has a higher rating median(people who enjoy watching movies socially enjoy this film less than those enjoy alone watching). In this case, we obtain a p-value of 0.05638214666114455, still higher than 0.05; hence, we are unable to reject the null hypothesis that there's no difference on movie rating of "Wolf of Wall Street" across "movie enjoyment alone" status of viewers and therefore there's no sufficient evidence to tell that enjoyment of "Wolf of Wall Street" is different depending on the "movie enjoyment alone" preferences of viewers. As a result, there's no significant evidence to suggest that people who like to watch movies alone enjoy 'The Wolf of Wall Street (2013)' more than those who prefer to watch them socially either.



```
stats.mannwhitneyu(wa_1['WSW'], wa_2['WSW'], alternative='greater')
MannwhitneyuResult(statistic=49303.5, pvalue=0.9436657996253056)
```



```
stats.mannwhitneyu(wa_1['WSW'], wa_2['WSW'], alternative='less')
MannwhitneyuResult(statistic=49303.5, pvalue=0.05638214666114455)
```

8. There are ratings on movies from several franchises (['Star Wars', 'Harry Potter', 'The Matrix', 'Indiana Jones', 'Jurassic Park', 'Pirates of the Caribbean', 'Toy Story', 'Batman']) in this dataset. How many of these are of inconsistent quality, as experienced by viewers?

In this case, we would like to evaluate if the movie rating across each film within the same series (eg. Star Wars) are equal which can reflect their quality from the perspective of the movie viewer. In this case, we still quantify the movie quality by rating median as the rating variables of discrete values and the variable distribution does not follow a normal distribution. Hence, we start with generating 8 small tables which only contain the movies from each different franchise (Star War I, Star War II….) separately from the original large dataset with 400 movies. In order to illustrate our analysis logic here, we use the Star War series as an example. According to the newly generated StarWar table, we observed that there are 6 movies in total, which means that we have 6 samples and we are required to evaluate if there's a rating difference among these 6 samples. Hence, the evaluation and comparison between their medians is hard within the context of traditional hypothesis testing; as a result, we finally selected **Kruskal-Wallis test** which can conduct hypothesis testing on multiple groups to assist with the task. Before conducting the test, we firstly deal with the missing values within the columns as we didn't preprocess them at the beginning due to the special meaning of movie rating null values in this dataset. In this case, since we need to evaluate the consistency of quality, we only keep the ratings from people who viewed all of the 6 movies since others' ratings may not reflect their ideas about the entire franchise quality. After dropping all the missing values which indicate people who didn't watch all of the 6 movies, we set a null hypothesis as the medians of all the movie populations within the franchise are equal whereas the alternative hypothesis as the there will be at least one population median that differs from the rest within the franchise to start our Kruskal-Wallis test. In addition, we also set the significance level alpha at 0.05 as generally used.Then, with the help of stats.kruskal() under the scipy.stats, we are able to compute the F-statistic as 193.51026675400544 and p-value as 6.940162236984522e-40, much lower than the significance level 0.05; hence, we are able to reject the hypothesis that there's no difference on movie rating across the Star War movies within the same franchise and therefore there's sufficient evidence to tell about the quality inconsistency from the perspective of viewers.

Following the similar steps as Star War analysis, we are able to compute the F-statistic and p-value for the other 7 franchises as shown in the table below:

Franchise Name	**Test-stat**	**p-value**	**Hypothesis**
Star War	193.51026675400544	6.940162236984522e-40	Reject
Harry Potter	5.8739552218536755	0.11790622831256074	Fail to reject
Matrix	40.32303905969196	1.7537323830838066e-09	Reject
Indiana Jones	54.19395477406098	1.020118354785894e-11	Reject
Jurassic Park	49.42733030275783	1.8492328391686058e-11	Reject

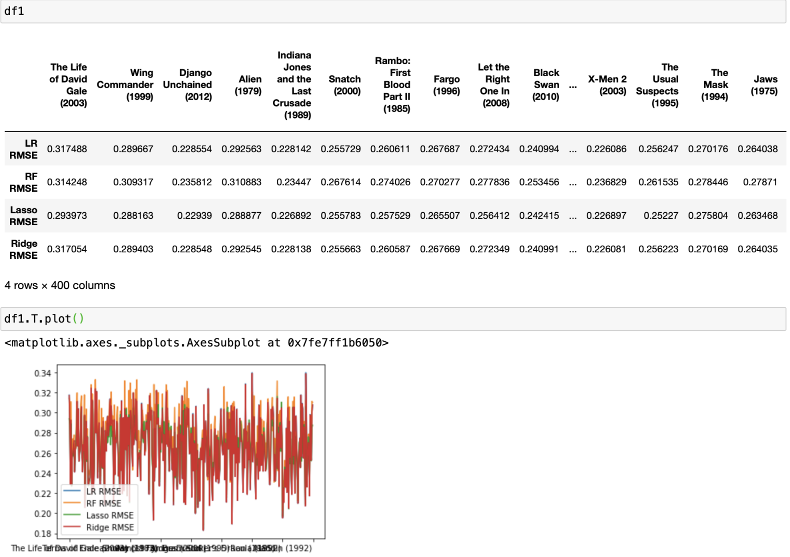
Pirates of Caribbean	6.660021086485515	0.035792727694248905	Reject
Toy Story	23.496729938969775	7.902234665149812e-06	Reject
Batman	84.65778425637279	4.1380499020034183e-19	Reject

According to this table, we can observe that 7 among 8 franchises has the test result which rejects the null hypothesis that the medians of all the movie populations within the franchise are equal, which means that there are 7 franchises that have inconsistent movie quality in total, leaving only Harry Potter as the one with consistent quality.

9. Build a prediction model of your choice (regression or supervised learning) to predict movie ratings (for all 400 movies) from personality factors only. Make sure to use cross-validation methods to avoid overfitting and characterize the accuracy of your model.

In this case, we firstly extract the personality factors (after PCA) as well as 400 movies from the original preprocessed dataset into a new small dataset. Because of the large number of missing values as well as the missing value amount difference across different movie ratings (ie. Life of David Gale has 1021 missing values whereas Django Unchained only has 644 missing values) which indicates the people rating each movie is very different, it's necessary for us to analyze and preprocess and analyze the data for each specific movie respectively. Hence, we randomly pick Alien(1979) as an example to illustrate our analytical process as well as the prediction model that can be employed to predict rating of any other movies since the independent factors are the same whereas the coefficient of final output and fitness may be slightly different due to movie type difference. As we have mentioned, because the missing values here indicate the people who did not rate this movie, here we decide to drop the missing values from the table directly to get rid of those who didn't watch this movie. As we have already standardized the data to address different scales of columns during the pre-processing process at the beginning, we can directly employ the model on this prepared dataset. In order to determine which supervised learning algorithm fits our data better, we managed to fit **linear regression, random forest, lasso regression, as well as ridge regression with 10-fold cross-validation** to avoid overfitting problems, generating relatively low RMSE values of 0.29256347213540407, 0.30756718605868255, 0.2888765942571356, 0.29254486405398117 respectively, indicating that all models are effective and accurate in the prediction of the movie Alien.

In order to test the effectiveness of all of the four models for all the movie ratings, we then repeat the procedures above (model fitting with cross-validation), and generate an RMSE table and plot of RMSE for each movie shown below. According to the plot, we can observe that all of the models perform stably well with a RMSE stable at around 0.2-0.35 level (fluctuates within this range in the plot), which is relatively low. However, the fluctuations of ridge regression and multiple linear regression are still stronger than random forest and lasso regression models according to the plot. Hence, we conclude that whereas all models work well for the selected movie Alien, random forest and lasso regression work much better and stably to accurately predict all the movie rating predictions.



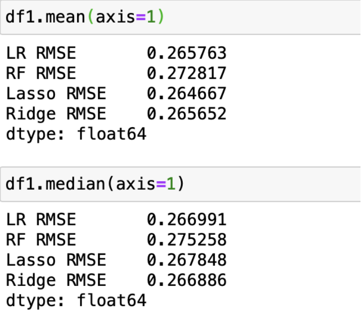
```
df1
```

	The Life of David Gale (2003)	Wing Commander (1999)	Django Unchained (2012)	Alien (1979)	Indiana Jones and the Last Crusade (1989)	Snatch (2000)	Rambo: First Blood Part II (1985)	Fargo (1996)	Let the Right One In (2008)	Black Swan (2010)	...	X-Men 2 (2003)	The Usual Suspects (1995)	The Mask (1994)	Jaws (1975)
LR RMSE	0.317488	0.289667	0.228554	0.292563	0.228142	0.255729	0.260611	0.267687	0.272434	0.240994	...	0.226086	0.256247	0.270176	0.264038
RF RMSE	0.314248	0.309317	0.235812	0.310883	0.23447	0.267614	0.274026	0.270277	0.277836	0.253456	...	0.236829	0.261535	0.278446	0.27871
Lasso RMSE	0.293973	0.288163	0.22939	0.288877	0.226892	0.255783	0.257529	0.265507	0.256412	0.242415	...	0.226897	0.25227	0.275804	0.263468
Ridge RMSE	0.317054	0.289403	0.228548	0.292545	0.228138	0.255663	0.260587	0.267669	0.272349	0.240991	...	0.226081	0.256223	0.270169	0.264035

4 rows × 400 columns

```
df1.T.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fe7ff1b6050>
```



```
df1.mean(axis=1)
```

```
LR RMSE       0.265763
RF RMSE       0.272817
Lasso RMSE    0.264667
Ridge RMSE    0.265652
dtype: float64
```

```
df1.median(axis=1)
```

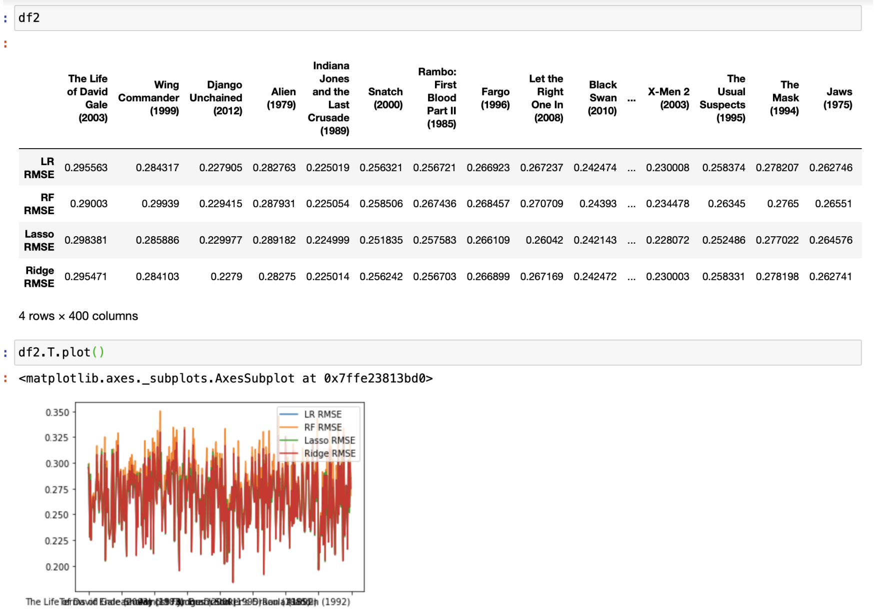
```
LR RMSE       0.266991
RF RMSE       0.275258
Lasso RMSE    0.267848
Ridge RMSE    0.266886
dtype: float64
```

In addition, we also compute the mean and median of the RMSE for 400 movie predictions based on these four models shown above, observing that lasso regression has lowest mean and relatively lower median RMSE than others. Hence, combining with the fluctuation trend observed in the above plot, we can conclude that lasso regression may work most effectively with stable performance whereas others also work well to predict for all the movie ratings.

10. Build a prediction model of your choice (regression or supervised learning) to predict movie ratings (for all 400 movies) from gender identity, sibship status and social viewing preferences (columns 475-477) only. Make sure to use cross-validation methods to avoid overfitting and characterize the accuracy of your model.

In this case, we firstly extract the three factors of gender identity, sibship status, and social viewing preferences as well as 400 movies from the original preprocessed dataset into a new small dataset. Similarly as we did in question 8, we randomly pick Alien(1979) as an example to illustrate our analytical process as well as the prediction model that can be employed to predict rating of any other movies since the independent factors are the same whereas the coefficient of final output and fitness may be slightly different due to movie type difference. As we have mentioned in question 8, we manage to drop the missing values from the table directly to get rid of those who didn't watch this movie based on the preprocessed dataset which has already experienced the normalization. In order to determine which supervised learning algorithm fits our data better, we managed to fit both linear regression, random forest, lasso regression, and ridge regression with 10-fold cross validation to avoid overfitting problems, generating low RMSE values of 0.2827630685480894, 0.28749297893687664, 0.28918202539331234, and 0.28274988228032333 respectively (code similar to question 8), indicating that all models are effective and accurate in the prediction of the movie Alien.

In order to test the effectiveness of all of the four models for all the movie ratings, we then repeat the procedures above (model fitting with cross-validation), and generate an RMSE table and plot of RMSE for each movie shown below. According to the plot, we can observe that all of the models perform stably well with a RMSE stable at around 0.1-0.35 level (fluctuates within this range in the plot), which is relatively low. However, the fluctuations of ridge regression and random forest models are still much stronger than lasso regression and linear regression models as they sometimes reach to over 0.35 level whereas lasso and linear regression models always fluctuate within 0.225-0.325 range. Hence, we conclude that whereas all models work well for the selected movie Alien, linear regression and lasso regression work much better and more stably to accurately predict all the movie rating predictions.

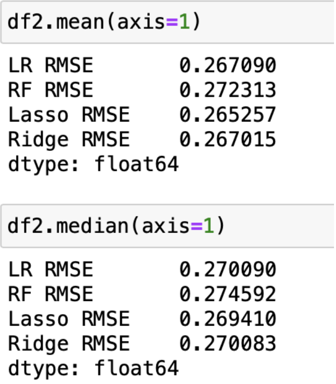


```
: df2
:
```

	The Life of David Gale (2003)	Wing Commander (1999)	Django Unchained (2012)	Alien (1979)	Indiana Jones and the Last Crusade (1989)	Snatch (2000)	Rambo: First Blood Part II (1985)	Fargo (1996)	Let the Right One In (2008)	Black Swan (2010)	...	X-Men 2 (2003)	The Usual Suspects (1995)	The Mask (1994)	Jaws (1975)
LR RMSE	0.295563	0.284317	0.227905	0.282763	0.225019	0.256321	0.256721	0.266923	0.267237	0.242474	...	0.230008	0.258374	0.278207	0.262746
RF RMSE	0.29003	0.29939	0.229415	0.287931	0.225054	0.258506	0.267436	0.268457	0.270709	0.24393	...	0.234478	0.26345	0.2765	0.26551
Lasso RMSE	0.298381	0.285886	0.229977	0.289182	0.224999	0.251835	0.257583	0.266109	0.26042	0.242143	...	0.228072	0.252486	0.277022	0.264576
Ridge RMSE	0.295471	0.284103	0.2279	0.28275	0.225014	0.256242	0.256703	0.266899	0.267169	0.242472	...	0.230003	0.258331	0.278198	0.262741

4 rows × 400 columns

```
: df2.T.plot()
: <matplotlib.axes._subplots.AxesSubplot at 0x7ffe23813bd0>
```



```
df2.mean(axis=1)
```

```
LR RMSE       0.267090
RF RMSE       0.272313
Lasso RMSE    0.265257
Ridge RMSE    0.267015
dtype: float64
```

```
df2.median(axis=1)
```

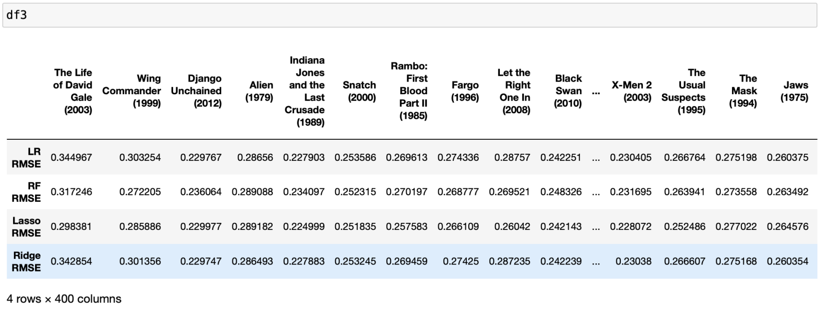
```
LR RMSE       0.270090
RF RMSE       0.274592
Lasso RMSE    0.269410
Ridge RMSE    0.270083
dtype: float64
```

In addition, we also compute the mean and median of the RMSE for 400 movie predictions based on these four models shown above, observing that lasso regression has lowest mean and median RMSE. Hence to conclude, in this case, lasso regression may work most effectively with stable performances whereas other models also work well to predict for all the movie ratings.

11. Build a prediction model of your choice (regression or supervised learning) to predict movie ratings (for all 400 movies) from all available factors that are not movie ratings (columns 401- 477). Make sure to use cross-validation methods to avoid overfitting and characterize the accuracy of your model.

In this case, we directly use the preprocessed dataset obtained from data preprocessing. Similarly as we did in question 8, we randomly pick Alien(1979) as an example to illustrate our analytical process as well as the prediction model that can be employed to predict rating of any other movies since the independent factors are the same whereas the coefficient of final output and fitness may be slightly different due to movie type difference. As we have mentioned in question 8, we manage to drop the missing values from the table directly to get rid of those who didn't watch this movie based on the preprocessed dataset which has already experienced the normalization. In order to determine which supervised learning algorithm fits our data better, we managed to fit all of models of linear regression, random forest, lasso regression, and ridge regression with 10-fold cross validation to avoid overfitting problems, generating low RMSE values of 0.28656006053519806, 0.2890896785807213, 0.28918202539331234, and 0.2864930392037451 respectively (code similar to question 8), indicating that all models are effective and accurate in the prediction of the movie Alien.

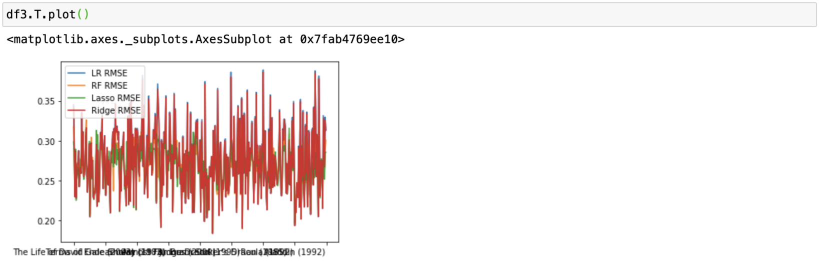
In order to test the effectiveness of all of the four models for all the movie ratings, we then repeat the procedures above (model fitting with cross-validation), and generate an RMSE table and plot of RMSE for each movie shown below. According to the plot, we can observe that all of the models perform stably well with a RMSE stable at around 0.2-0.4 level (fluctuates within this range in the plot), which is relatively low. However, the fluctuations of ridge regression and multiple linear regression are still much stronger than random forest and lasso regression models as they sometimes reach to 0.4 level whereas lasso and random forest models always fluctuate within 0.25-0.325 range. Hence, we conclude that whereas all models work well for the selected movie Alien, random forest and lasso regression work much better and stably to accurately predict all the movie rating predictions.



```
df3
```

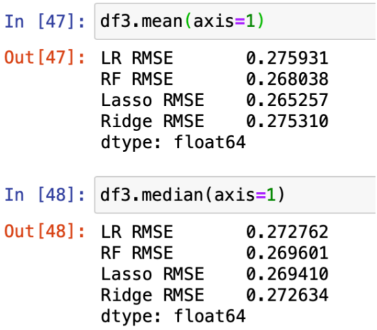
	The Life of David Gale (2003)	Wing Commander (1999)	Django Unchained (2012)	Alien (1979)	Indiana Jones and the Last Crusade (1989)	Snatch (2000)	Rambo: First Blood Part II (1985)	Fargo (1996)	Let the Right One In (2008)	Black Swan (2010)	...	X-Men 2 (2003)	The Usual Suspects (1995)	The Mask (1994)	Jaws (1975)
LR RMSE	0.344967	0.303254	0.229767	0.28656	0.227903	0.253586	0.269613	0.274336	0.28757	0.242251	...	0.230405	0.266764	0.275198	0.260375
RF RMSE	0.317246	0.272205	0.236064	0.289088	0.234097	0.252315	0.270197	0.268777	0.269521	0.248326	...	0.231695	0.263941	0.273558	0.263492
Lasso RMSE	0.298381	0.285886	0.229977	0.289182	0.224999	0.251835	0.257583	0.266109	0.26042	0.242143	...	0.228072	0.252486	0.277022	0.264576
Ridge RMSE	0.342854	0.301356	0.229747	0.286493	0.227883	0.253245	0.269459	0.27425	0.287235	0.242239	...	0.23038	0.266607	0.275168	0.260354

4 rows × 400 columns



```
df3.T.plot()
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fab4769ee10>
```

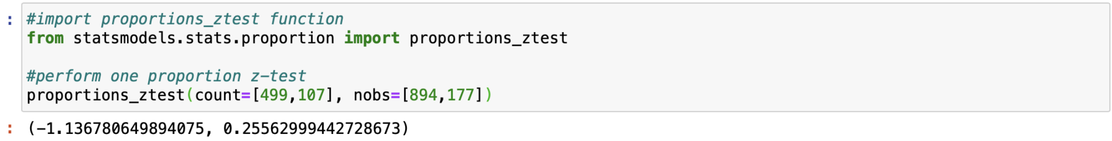


```
In [47]: df3.mean(axis=1)

Out[47]: LR RMSE       0.275931
         RF RMSE       0.268038
         Lasso RMSE    0.265257
         Ridge RMSE    0.275310
         dtype: float64

In [48]: df3.median(axis=1)

Out[48]: LR RMSE       0.272762
         RF RMSE       0.269601
         Lasso RMSE    0.269410
         Ridge RMSE    0.272634
         dtype: float64
```

In addition, we also compute the mean and median of the RMSE for 400 movie predictions based on these four models shown above, observing that lasso regression has lowest mean and median RMSE. Hence to conclude, in this case, lasso regression may work most effectively with stable performance whereas other models also work well to predict for all the movie ratings.

12. Extra credit

As an only child in family who enjoys watching movies together with friends, I'm quite curious about the preferences of other only children, and hence I want to evaluate the relationship between only child status and their preference on watching movies alone or socially. To start with, I firstly extract the only child column and the social movie watching preference column from the large preprocessed dataset into a smaller new dataset. As the data of these two columns have already been preprocessed at the beginning, we do not need to deal with missing values any more in this case. As the data we have for the only child and social watching preference column are both binary data, I decided to run a 2 proportion z-test to evaluate if the proportion of people who enjoy watching movie alone in the group of only child and in the group of non only child is equal, which can offer me some insights about the impact of only child status on the social movie watching preferences. In terms of the test setup, we set the null hypothesis as that the two population proportions are equal whereas the alternative hypothesis is that the two population proportions are not equal. In addition, we also set the significance level alpha is 0.05 as generally used. After splitting the table into two separate ones based on the only child status (one table only contain viewers who are only child while the other only contain viewers who are not only child), we managed to compute the number of observations for those prefer to watch movies alone as well as the sample size in the two groups of only and non-only child. With the use of proportions_ztest(), we are able to compute the p-value of the 2 proportion test, finally obtaining the p-value as 0.25562999442728673, higher than 0.05; hence, we fail to reject the null hypothesis that the two population proportions are equal and therefore there's no sufficient evidence to tell that the proportions of people who enjoy watching movie alone are different depending on the only child status of viewers. As a result, there's no significant evidence to suggest that social watching movie preferences differ across only child status and hence we cannot tell that people who are only children prefer to watch movies alone more than those who are not only children.



```
: #import proportions_ztest function
  from statsmodels.stats.proportion import proportions_ztest

  #perform one proportion z-test
  proportions_ztest(count=[499,107], nobs=[894,177])

: (-1.136780649894075, 0.25562999442728673)
```