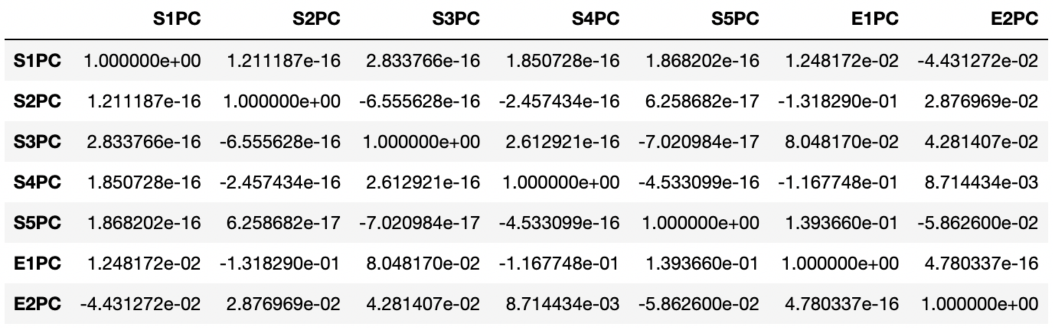
Capstone Project

Kelly Deng rd2755
Spring 2022

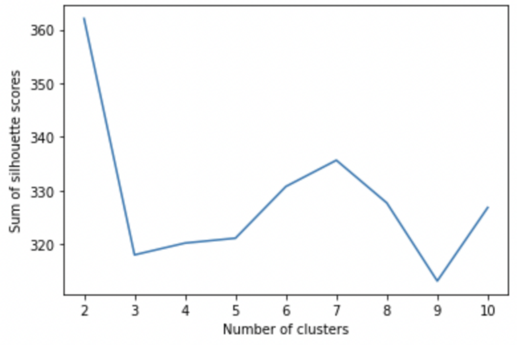
In this project, **dimensional reduction** is utilized in Question 1 and 2. I did **PCA** respectively for Sensation seeking, Personality, and Movie experience, and extracted needed principal components for correlation and clustering. Missing datas in Question 1 to 7 are solved by *dropna()*, whether through element-wise or row-wise depending on specific conditions. For Question 8 to 10, I used **IterativeImputer** to replace Nan with imputations. For different questions, I extracted particular columns through *df.iloc*, combined them together through *pd.concat()* if needed, and stored them in **DataFrames**. For hypothesis tests, I take **alpha = 0.05**.

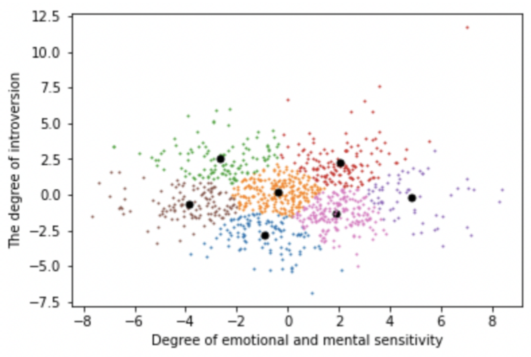
Question1: To show the relationship between sensation seeking and movie experience, I concatenated the two parts of the dataset and removed nan row-wise, by which I only kept users who rated all the characteristics in both sections. I did PCA for both, and Kaiser criterion suggests 6 factors for Sensation seeking while 2 factors for Movie experience. The two factors account for 48.257% of the variance in Movie experience, so I decided to find the factors that account for a similar percentage of variance in Sensation seeking. 5 factors covering 46.852% were chosen. Then, I did correlation between the first 5 principal components of Sensation seeking(noted as “S_PC”) and the first 2 principal components of Movie experience(noted as “E_PC”), and a correlation coefficient matrix is shown. The relationship between sensation seeking and movie experience is not strong.



	S1PC	S2PC	S3PC	S4PC	S5PC	E1PC	E2PC
S1PC	1.000000e+00	1.211187e-16	2.833766e-16	1.850728e-16	1.868202e-16	1.248172e-02	-4.431272e-02
S2PC	1.211187e-16	1.000000e+00	-6.555628e-16	-2.457434e-16	6.258682e-17	-1.318290e-01	2.876969e-02
S3PC	2.833766e-16	-6.555628e-16	1.000000e+00	2.612921e-16	-7.020984e-17	8.048170e-02	4.281407e-02
S4PC	1.850728e-16	-2.457434e-16	2.612921e-16	1.000000e+00	-4.533099e-16	-1.167748e-01	8.714434e-03
S5PC	1.868202e-16	6.258682e-17	-7.020984e-17	-4.533099e-16	1.000000e+00	1.393660e-01	-5.862600e-02
E1PC	1.248172e-02	-1.318290e-01	8.048170e-02	-1.167748e-01	1.393660e-01	1.000000e+00	4.780337e-16
E2PC	-4.431272e-02	2.876969e-02	4.281407e-02	8.714434e-03	-5.862600e-02	4.780337e-16	1.000000e+00

Question2: First, I did PCA for the row-wise removed nan Personality dataset, finding 8 factors suggested by Kaiser criterion. I chose the first two since they account for the most variance of the dataset. By showing the loading matrix of each of them, I listed the original factors that can explain the meaning of the new factor. “Degree of emotional and mental sensitivity” is what I concluded for the first factor, and “Degree of introversion” is for the second. Then, using these two new factors and columns, I computed K-means through the silhouette method, finding that the optimal K is 7. Finally, I did clustering and plotted the graph. The seven types of personalities decided on “Degree of emotional and mental sensitivity” and “Degree of introversion” are shown.





Question3: I did a hypothesis test with Null: ratings of popular movies are not higher than unpopular movies, and Alternative: popular movies are rated higher than unpopular movies. First, I removed nan element-wise for the 400 rating columns. Then, I counted the number of each movie's ratings and computed the median number of ratings. Movies are splitted into two groups by the median: popular(number of ratings > median) and not_popular(number of ratings < median). Finally, I did a one-tailed Mann-Whitney U test for the two groups since we are comparing ordinal movie ratings. The result shows that p<0.05, so we can reject null and conclude that movies that are popular are rated higher than movies that are less popular.

```
u1,p1 = stats.mannwhitneyu(popular ,not_popular, alternative='greater')
u1,p1
```

```
(35404.0, 8.485716560078964e-41)
```

Question4: I defined the Null hypothesis as male and female viewers rate "Shrek" in the same way, and the Alternative hypothesis as male and female viewers rate "Shrek" differently. First, I extracted the "Shrek (2001)" and the gender identification columns and joined them together to form a new dataframe. Then, I used *groupby()* and *get_group()* to separate the ratings by gender and got two groups: male and female. Nan was removed element-wise respectively in the two groups. Finally, I did a two-sided Mann-Whitney U test, finding that p>0.05. So we cannot reject null and cannot say male and female viewers rate "Shrek" differently.

```
#two-sided Mann-Whitney U test
u,p = stats.mannwhitneyu(female,male)
u,p
```

```
(96830.5, 0.050536625925559006)
```

Question5: I defined the Null hypothesis as only child do not enjoy "The lion king" more than people with siblings, and Alternative hypothesis as only child enjoy the movie more than people with siblings. Data was extracted and formatted the same as the last question. And by the same method, I got two element-wise removed nan data groups: onlyChild and withSiblings. Then, I did a one-sided Mann-Whitney test and found that p-value is very large. So, we cannot reject null. Then, I did another one-sided test but on the other side. What I got is p<0.05. In this way, I concluded that actually only child enjoy "The Lion King" less than people with siblings.

```
#one-sided
u,p = stats.mannwhitneyu(onlyChild,withSiblings, alternative = 'greater')
u,p
```

```
(52929.0, 0.978419092554931)
```

```
#one-sided
u,p = stats.mannwhitneyu(onlyChild,withSiblings, alternative = 'less')
u,p
```

```
(52929.0, 0.021599364978414245)
```

Question6: I defined the Null hypothesis as people who like to watch movies socially do not enjoy “The Wolf of Wall Street” more than those who prefer to watch alone, and Alternative hypothesis as people who like to watch movies socially enjoy “The Wolf of Wall Street” more than those who prefer to watch alone. Data was extracted and formatted the same as the last question. And by the same method, I got two element-wise removed nan data groups: socially and alone. Then, I did a one-sided Mann-Whitney test and found that p-value is very large. So, we cannot reject null. People who like to watch movies socially do not enjoy “The Wolf of Wall Street” more than those who prefer to watch alone.

```
#one-sided Mann-Whitney U test
u,p = stats.mannwhitneyu(socially,alone, alternative = 'greater')
u,p
```

```
(49303.5, 0.9436657996253056)
```

Question7: In this question, I showed the equality consistency of each franchise by comparing each movie with its next movie(in time sequence). For each franchise, my Null is the qualities of movies in this franchise are consistent, and my Alternative is movies of this franchise have different qualities. First, I printed out all the movies of a franchise(using a *for loop*) and extracted their corresponding rating columns. Nan was removed element-wise, and an array called “series_rating” was created to store the rating columns in movies’ time order. Then, I did two-sided Mann-Whitney tests for each movie and the one next to it through a *for loop*. Here is the result for “Star Wars”. “Star Wars” has 6 movies, and the quality of the third and forth, fifth and sixth movies is inconsistent, since their $p<0.05$. So, Null is rejected for “Star Wars”, and its movie quality is inconsistent. I did this process for all the franchises. My conclusion is that only Harry Potter has consistent quality movies(all $p>0.05$), and all the others don’t.

```
Star Wars: Episode IV - A New Hope (1977) (array([21]),)
Star Wars: Episode II - Attack of the Clones (2002) (array([93]),)
Star Wars: Episode V - The Empire Strikes Back (1980) (array([174]),)
Star Wars: Episode 1 - The Phantom Menace (1999) (array([273]),)
Star Wars: Episode VII - The Force Awakens (2015) (array([336]),)
Star Wars: Episode VI - The Return of the Jedi (1983) (array([342]),)
```

```
#Use loop to do several hypothesis tests simultaneously
for i in range(len(series_rating)-1):
    u,p = stats.mannwhitneyu(series_rating[i],series_rating[i+1])
    print(u,p)
```

```
125332.0 0.08189846600030656
114952.0 0.3034256316184082
133033.5 4.730789191200012e-15
124054.5 0.4503093559063731
78970.0 1.4828150905774305e-27
```

Question8: For the last three prediction questions, I chose to do Linear Regression for all of them. Since we will predict the whole 400 movie ratings, if we remove missing datas, no matter through row-wise or element-wise, it will be either too less data to predict or too disordered to fit in the model. So, I decided to fill in the missing datas. I imported *IterativeImputer* from *sklearn* to do this job. It took about 10 minutes to run the code, and here's part of the dataset after imputation.

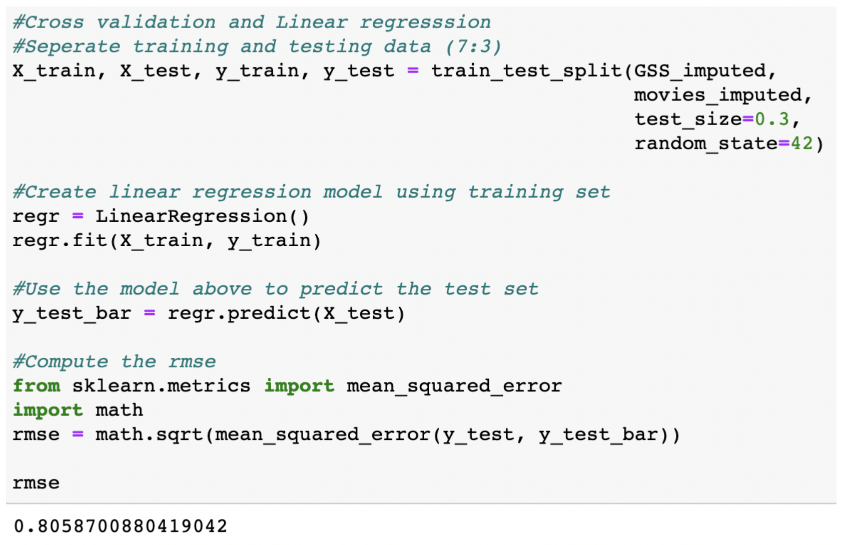
	0	1	2	3	4	5	6	7	8	9	...	467	468	469	470
0	2.060089	1.219609	4.000000	3.452489	3.000000	3.475948	2.856775	1.759794	2.873154	2.469153	...	1.0	6.0	2.0	5.0
1	2.585111	2.098402	1.500000	1.852190	2.439677	2.953439	2.677461	3.343155	1.715700	2.437288	...	3.0	1.0	1.0	6.0
2	3.458714	2.792202	3.005196	1.895448	3.108396	2.516230	2.455243	3.078429	2.685970	2.822577	...	5.0	4.0	3.0	5.0
3	2.404148	1.971983	2.000000	2.023382	3.000000	2.542075	1.446218	3.257445	2.976212	4.000000	...	3.0	1.0	1.0	4.0
4	1.561322	2.241729	3.500000	1.197154	0.500000	2.262482	0.500000	1.000000	2.089817	0.000000	...	2.0	3.0	2.0	5.0

Then, I assigned the 400 movie rating columns as "y", personality columns as "X", both from the imputed dataset. Cross validation is necessary to avoid overfitting, so I used the *X_train, X_test, y_train, y_test = train_test_split()* function to generate training and testing sets, in which I set the ratio as 7:3(3 for testing). Then, the *LinearRegression()* function was called, and "X_train" and "y_train" were fitted in. I used the model to do prediction in "X_test", and the result was labeled "y_test_bar". Finally, I computed RMSE for the test set and the outcome is shown below.

```
#Compute the rmse
from sklearn.metrics import mean_squared_error
import math
rmse = math.sqrt(mean_squared_error(y_test, y_test_bar))

rmse
```

```
0.7776553777395455
```

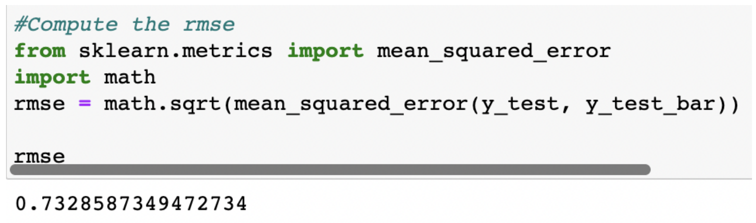
Question9: For this question, I extracted movie rating columns and the GSS(gender identity, sibship status and social viewing preferences) columns from the imputed dataset. Then I did cross validation, multiple linear regression, and computed the RMSE all in the same way as the last question. Here is the code display and the outcome.



```
#Cross validation and Linear regresssion
#Seperate training and testing data (7:3)
X_train, X_test, y_train, y_test = train_test_split(GSS_imputed,
                                                    movies_imputed,
                                                    test_size=0.3,
                                                    random_state=42)

#Create linear regression model using training set
regr = LinearRegression()
regr.fit(X_train, y_train)

#Use the model above to predict the test set
y_test_bar = regr.predict(X_test)

#Compute the rmse
from sklearn.metrics import mean_squared_error
import math
rmse = math.sqrt(mean_squared_error(y_test, y_test_bar))

rmse
```

```
0.8058700880419042
```

Question10: For this question, except for changing "X" into all available factors, I did exactly the same way as the two questions before. Here's the outcome.



```
#Compute the rmse
from sklearn.metrics import mean_squared_error
import math
rmse = math.sqrt(mean_squared_error(y_test, y_test_bar))

rmse
```

```
0.7328587349472734
```