Chloe Nam

Professor Wallisch

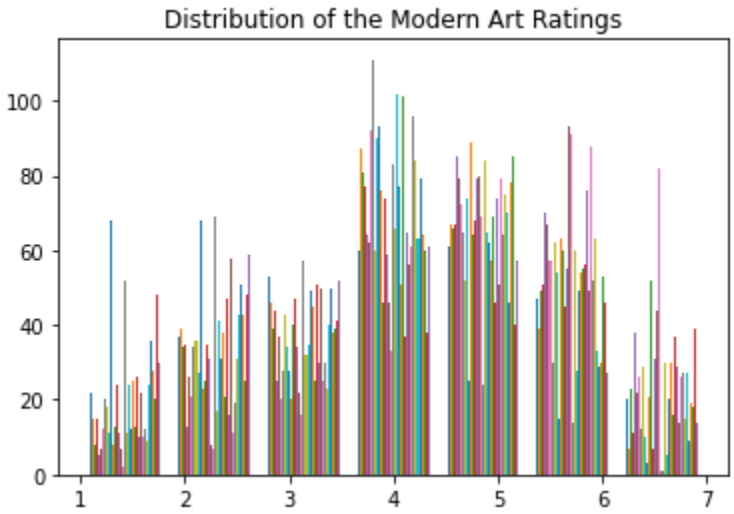
Intro to Data Science (DS-UA-112)

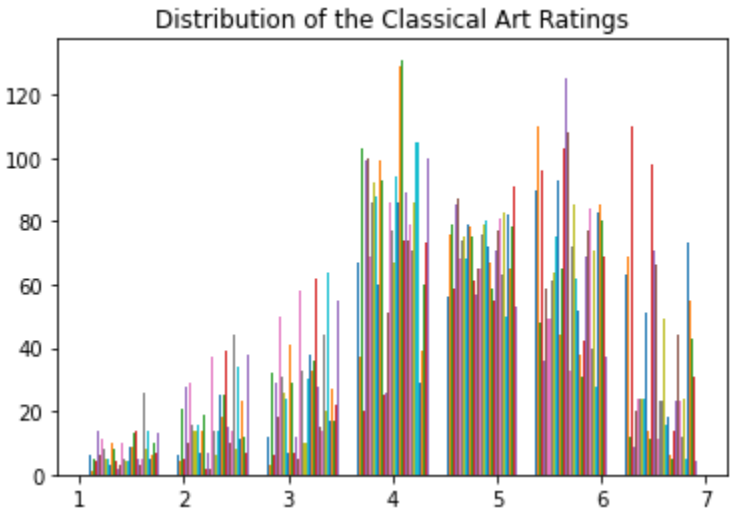
20 December 2022

Final Capstone Project

Introduction: For my capstone project, I first utilized NumPy and pandas libraries to read the two .csv files given to us. I set theArt.csv dataset to a variable titled 'theArt' and set theData.csv to a variable called 'data'. I performed a Principal Component Analysis (PCA) for reducing dimensionality of the dataset. If the question included any columns or rows in the datasets that had NaN values, I cleaned the data by creating a pandas dataframe and utilizing the .dropna() function or utilizing the ~np.isnan() function. In order to minimize the amount of data lost, only the variables necessary to answer the specific question would be cleaned. The data was transformed by being standardized through z-scoring, and then rotated in order to plot it in the new coordinate system when performing principal component analysis.

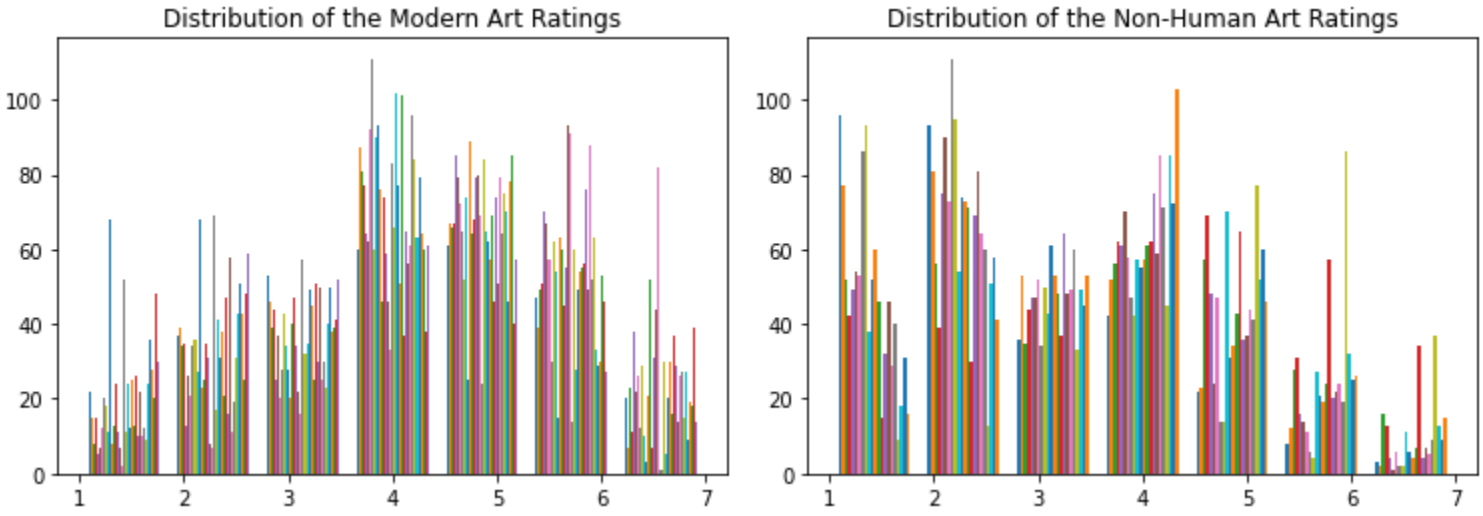
1) **Is classical art more well liked than modern art?**





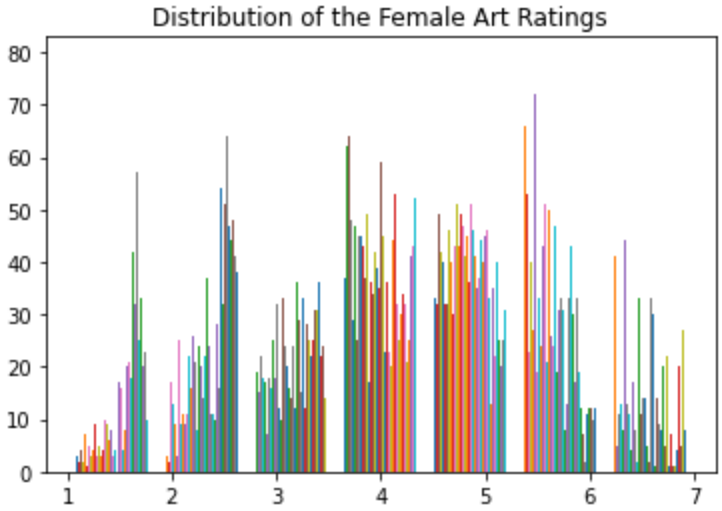
Yes, classical art is more well liked than modern art. By calculating the mean of preference ratings for both classical art (4.74152) and modern art (4.25657), we can see that the mean is higher for classical art. We can also perform the Mann Whitney U statistical test to get the u statistic (52313.76) and p value (0.10917) which shows that there is not a statistically significant difference between the two samples as the p value is higher than 0.05. Furthermore, simply by reading the two plots we can see that the classical art plot is skewed more to the right than the modern art plot, proving that there are higher ratings given for classical art.

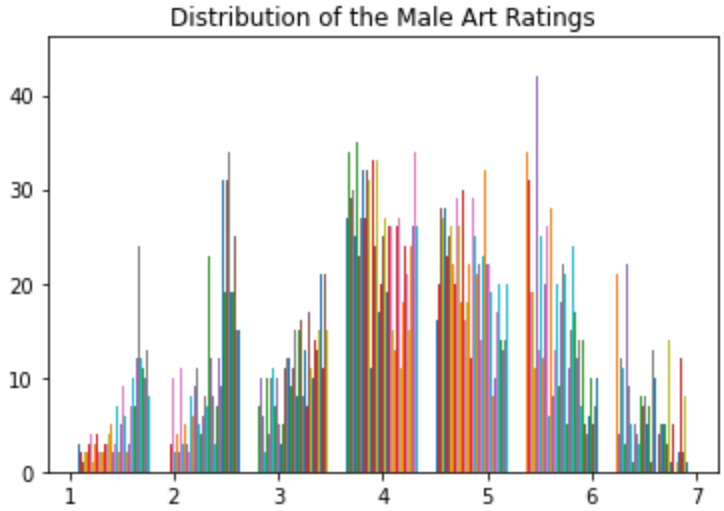
2) **Is there a difference in the preference ratings for modern art vs. non-human (animals and computers) generated art?**



Yes, there is a difference in the preference ratings for modern art vs. non-human (animals and computers) generated art. By calculating the mean of preference ratings for non-human generated art (3.33348) and comparing it to the mean of preference ratings for modern art which we have previously found (4.25657), we see that modern art is more liked than non-human generated art. Furthermore, by running the Mann Whitney U statistical test, we see that the p-value is 2.31582e^-259, which is less than 0.05. Thus, there is a statistically significant difference between the preference ratings for modern art and non-human art.

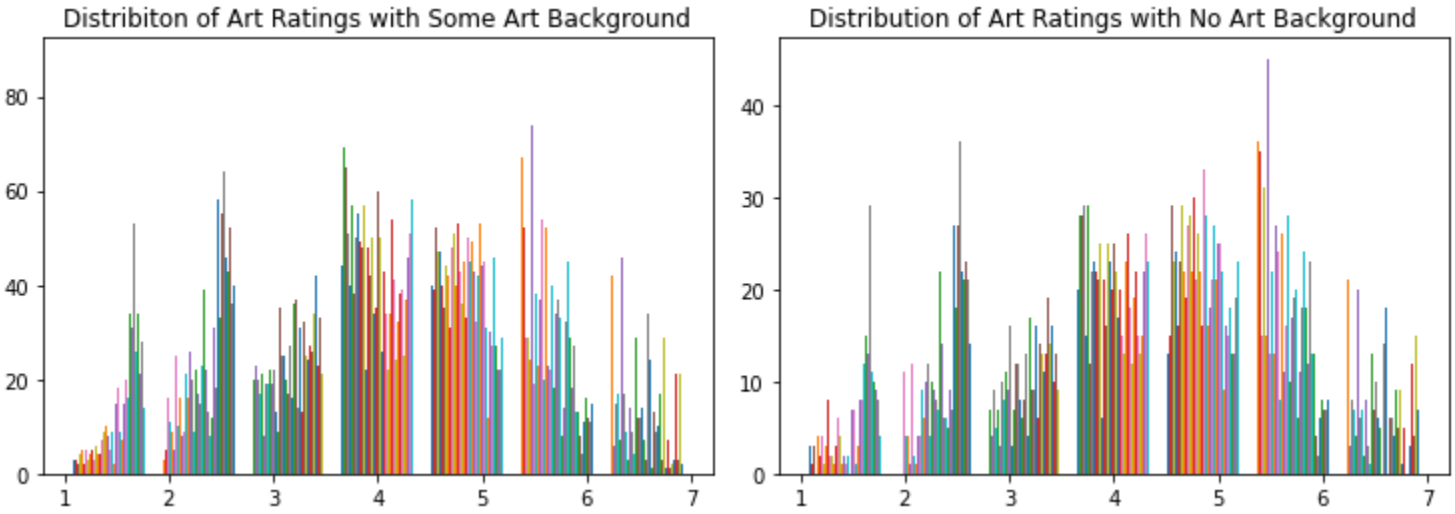
3) Do women give higher art preference ratings than men?





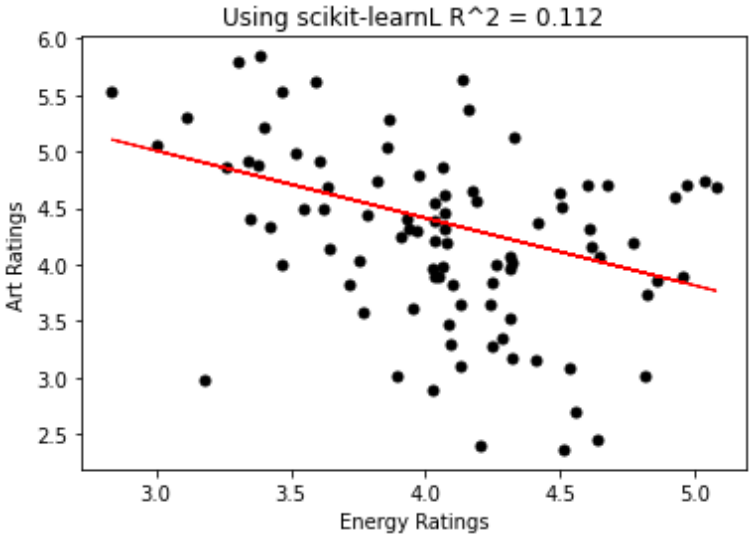
In order to determine whether women give higher art preference ratings than men, I first set a 2D data array which extracts the column of user gender to a variable called “gender” and a 2D data array which extracts all rows of the columns 0 to 91 and set it to a variable called “art_ratings”. Then, I put these variables in a pandas dataframe and dropped NaN values in order to clean up the data. After cleaning, I changed the data frames back into a NumPy array and calculated the median values which both equaled 4.0 which shows that women do not give higher art preference ratings than men. I then ran the Mann Whitney U test which showed us that the U-statistic is 8563.42308 and the p-value is 0.37475. The p-value is greater than 0.05, which proves again that there is not a statistically significant difference between male and female art ratings.

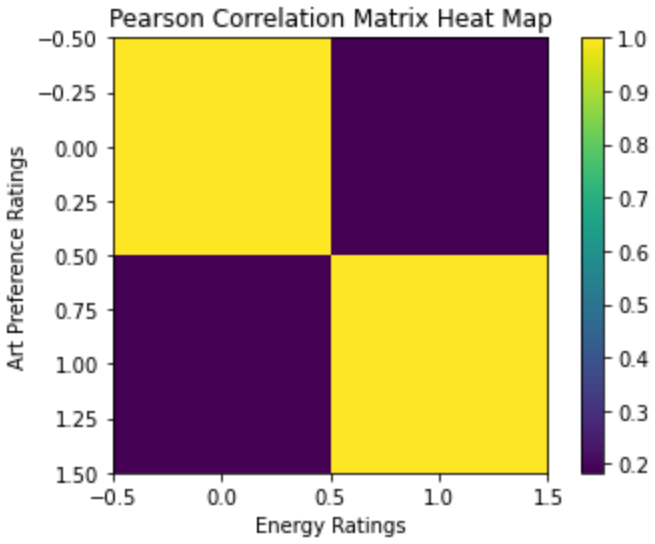
4) **Is there a difference in the preference ratings of users with some art background (some art education) vs. none?**



In order to determine whether there is a difference in the preference ratings of users with some art education vs. no art education, I first created a 2D data array which extracted the column of art education and set it to the variable “art_education”. I then converted it to a pandas dataframe titled “df_art_education” in order to drop NaN values using the .dropna() function. After cleaning the data, I changed it back to a NumPy array and calculated the medians with “np.median()” of some art education and no art education which both equaled 4.0. Since the medians are the same, this shows that there is no difference in the preference ratings of users with some art background vs no art background. Furthermore, I ran a Mann Whitney U statistical test which showed a U-statistic of 9051.43956 and a p-value of 0.45469. Since the p-value is greater than 0.05, this shows that there is not a statistically significant difference between users with some art education and no education.

5) **Build a regression model to predict art preference ratings from energy ratings only. Make sure to use cross-validation methods to avoid overfitting and characterize how well your model predicts art preference ratings.**

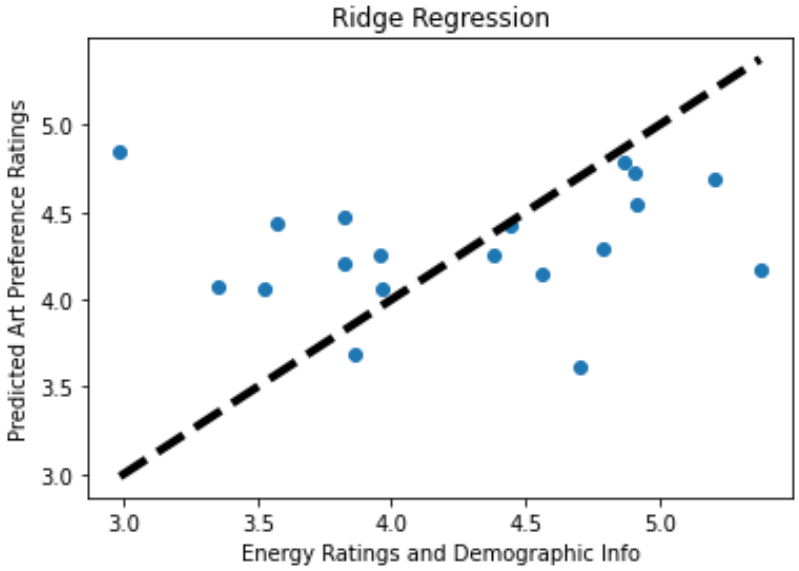




I first created a 2D data array which extracts all rows of the columns from 91 to 182 and set it to a variable titled “energy_ratings”. I created another 2D data array which extracts all rows from the first column to the 91st and set it to a variable titled “art_ratings”. I then used the np.column_stack() function to combine the input arrays by stacking them column-wise, so that the resulting array has one column for each input array. I used the np.corrcoef() function to calculate the Pearson correlation coefficient between the two input arrays which measures the linear relationship between the two variables. Specifically in this code, the ‘corrcoef’ function is being used to compute the correlations between the two input arrays ‘data5[:,0]’ and ‘data5[:,1]’. Then, I calculated the mean, median, and standard deviation of “energy_ratings” and “art_ratings” and also created a Correlation heat map which shows the correlation between energy ratings and art preference ratings. I began building my linear regression model by splitting the mean into train and test sets and plotting the original data (the means). I fit the train and test set and calculated the R-squared score which equaled 0.11165. I also calculated the RMSE which equaled 0.68526. The coefficient (b1) equaled -0.59723 and the intercept (b0) equaled 6.79912. I created a regression line with “yHat = slope * energy_mean + intercept” and did predictions with “y_pred = ourModel.predict(X_test). I calculated mean absolute error

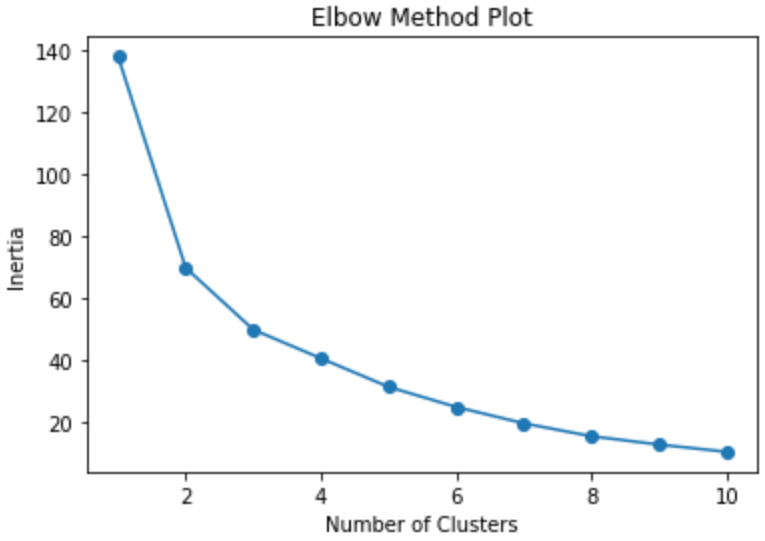
(0.50602), mean squared error (0.68526), root mean squared error (0.68526), and plotted the regression line titled “Using skikit-learnL R^2 = {:.3f} which is shown above.

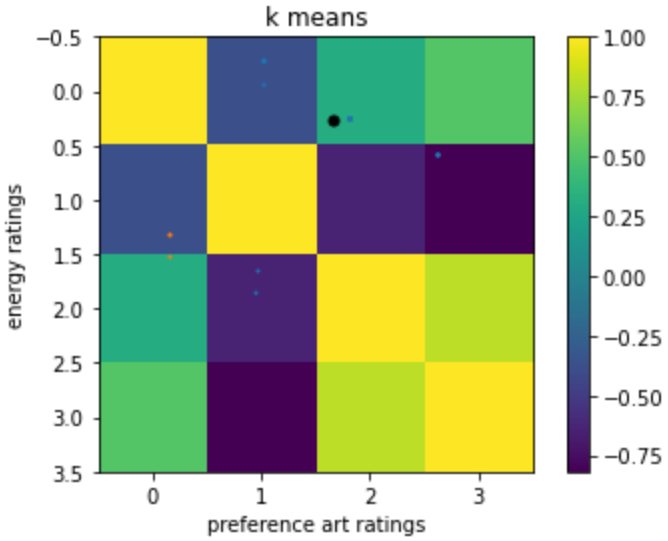
6) **Build a regression model to predict art preference ratings from energy ratings and demographic information. Make sure to use cross-validation methods to avoid overfitting and comment on how well your model predicts relative to the “energy ratings only” model.**

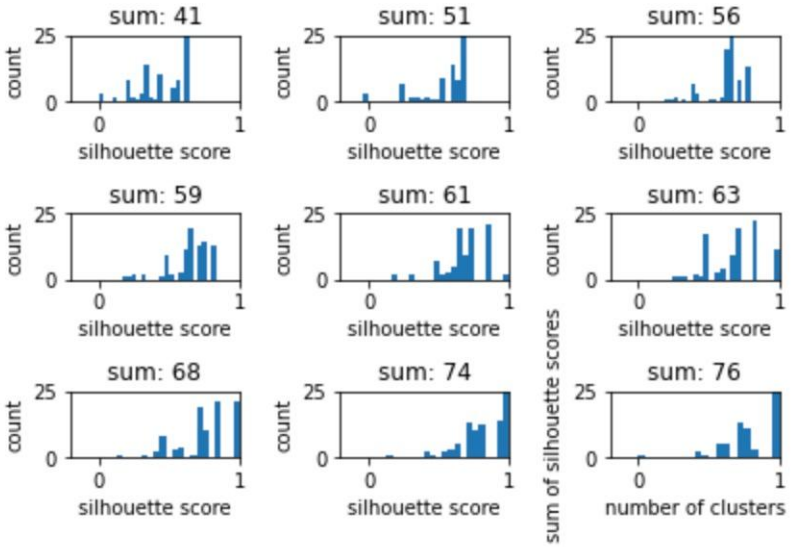


To start, I created a data array for energy ratings that indexed columns 91 to 182 from the data array and a data array for the demographic information that indexed columns 215 to 220. I checked for NaNs in both arrays and then used the np.column_stack() function to combine both arrays. I then found the mean for energy rating, demographic information, art ratings and made sure to drop the NaN values. Using the train_test_split from sklearn.model_selection, I cross validated my data to avoid overfitting. Next, I used a Ridge regression model to fit, predict, and plot the Ridge regression for predicting art preference ratings from energy ratings and demographic information. Lastly, I calculated the mean squared error (0.48693) which shows how well the model predicts relative to the energy ratings only model.

7) **Considering the 2D space of average preference ratings vs. average energy rating (that contains the 91 art pieces as elements), how many clusters can you – algorithmically - identify in this space? Make sure to comment on the identity of the clusters – do they correspond to particular types of art?**



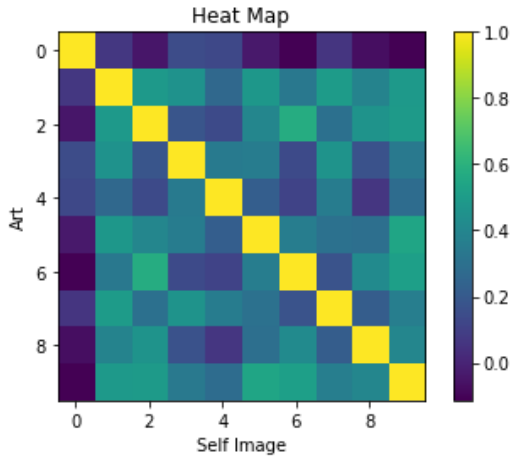


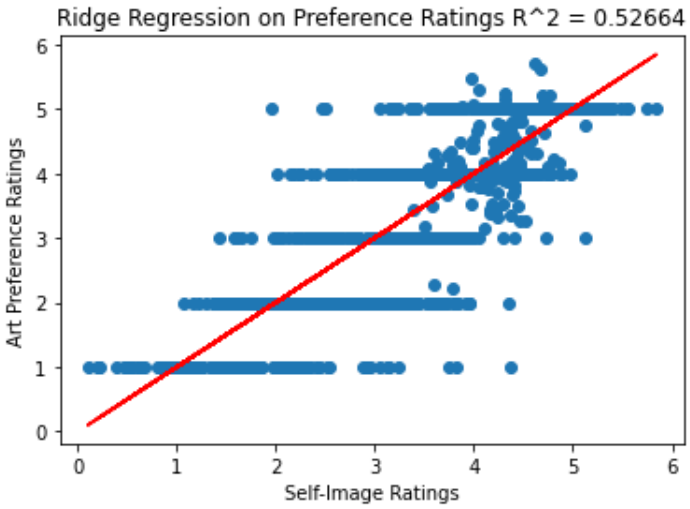


For this question, I used a kNN means model and the elbow method in order to identify the number of clusters in this space. Based on the K-means graph I plotted, we see that there are 3 clusters. First, I created an empty list called inertias and used a for loop to fit a K-Means model

to the data array for a range of values of n_clusters (1 to 10). For each value of n_clusters, the code fits a K-Means model using the KMeans class from scikit-learn's cluster module and the fit method. It then appends the model's inertia, which is a measure of the sum of the squared distances between the points within each cluster and the cluster's centroid, to the inertias list. After the for loop completes, the inertias list will contain the inertias for each of the K-Means models fit to the data for the specified range of values of n_clusters. I then performed PCA to find the number of clusters through silhouette scores. For each value of n_clusters, the code fits a K-Means model using the KMeans class from scikit-learn's cluster module and the fit method, and extracts the labels and cluster centers from the fitted model. It then uses the silhouette_samples function from scikit-learn's metrics module to compute the silhouette score for each data point, reshapes the resulting array into a column vector, and sums the values in the vector. The sum is then assigned to the corresponding element of the sSum array. The last part of my code for this question shows that I created a Pandas DataFrame containing the original data (data) and the predicted cluster labels (predicted_types) using the concat function and assigned it to the kmeans_results variable. I then used NumPy's corrcoef function to compute the correlations between the columns of the kmeans_results DataFrame and plotted the correlations using matplotlib's imshow and colorbar functions.

8) **Considering only the first principal component of the self-image ratings as inputs to a regression model – how well can you predict art preference ratings from that factor alone?**





I created an array called self_image that indexes from columns 205 to 215 and used the array called art which indexes from columns 0 to 91. I performed data cleaning by checking for NaNs in my combined matrix. I then created a heat map and performed a PCA. I used the Kaiser Criterion which told us the number of factors selected by the Kaiser Criterion is 2. Then we make the regression model, fit, predict, and create the scatter plot for preference ratings. Based on our R-squared value of 0.52664 , we can see that the model is decently okay at predicting the art preference ratings well from this factor alone since R-squared ranges from 0 to 1. I also calculated the RMSE which is a value of 0.55321.

9) **Consider the first 3 principal components of the "dark personality" traits – use these as inputs to a regression model to predict art preference ratings. Which of these components significantly predict art preference ratings? Comment on the likely identity of these factors (e.g. narcissism, manipulativeness, callousness, etc.).**

```
524
525   dark_personality = np.array(data[:, 183:195])
526   art = np.array(data[:, 0:91])
527
528   art_mean = art.mean(axis = 1)
529
530   dark_personality_NaNCheck = np.any(np.isnan(dark_personality))
531   art_NaNCheck = np.any(np.isnan(art_mean))
532
533   dark_personality_art = np.column_stack((dark_personality, art_mean))
534   dark_personality_art_NaN = np.isnan(dark_personality_art)
535   dark_personality_art_NaNrows = np.where(dark_personality_art_NaN.any(axis = 1))[0]
536   dark_personality_art_noNaN = np.delete(dark_personality_art, dark_personality_art_NaNrows, axis = 0)
537
538   dark_personality_art_noNaN = (pd.DataFrame(dark_personality_art_noNaN)).iloc[:,0:12]
539   art_mean_noNANs = (pd.DataFrame(dark_personality_art_noNaN)).iloc[:,12]
540
541   #self_image_art_corr = np.corrcoef(self_image_art_noNaN, art_mean_noNANs)
542   #demographic_art_corr = np.corrcoef(demographic, art_mean_noNANs)
543
544   corrMatrix = np.corrcoef(dark_personality_art_noNaN, rowvar = False)
545   plt.imshow(corrMatrix)
546   plt.xlabel("Dark-personality")
547   plt.ylabel("Dark-personality")
548   plt.colorbar()
549   plt.show()
550
551   #PCA
552   zscoredData = stats.zscore(dark_personality_art_noNaN)
553   pca = PCA().fit(zscoredData)
554   eigVals = pca.explained_variance_
555   loadings = pca.components_*-1
556   rotatedData = pca.fit_transform(zscoredData)*-1
557   varExplained = eigVals/sum(eigVals)*100
558
559   kaiserThreshold = 1
560   print('Number of factors selected by Kaiser criterion:', np.count_nonzero(eigVals > kaiserThreshold)) #This outputs 2
561   # 3 Factors selected for the critera
562
563
564   kairserThresholdPassers = eigVals > kaiserThreshold
565   PCA1 = pca.transform(zscoredData)[:, kairserThresholdPassers]
566
567   #PCA Number 2
568   pca2 = PCA()
569   X_pca2 = pca2.fit_transform(PCA1)
570   n_components2 = sum(pca2.explained_variance_ > 1)
571   X_pca2 = X_pca2[:, :n_components2]
572   #PCA Number 3
573   pca3 = PCA()
574   X_pca3 = pca3.fit_transform(X_pca2)
575   n_components3 = sum(pca3.explained_variance_ > 1)
576   X_pca3 = X_pca3[:, :n_components3]
577
578
```

10) Can you determine the political orientation of the users (to simplify things and avoid gross class imbalance issues, you can consider just 2 classes: "left" (progressive & liberal) vs. "nonleft" (everyone else)) from all the other information available, using any classification model of your choice? Make sure to comment on the classification quality of this model.

```
In [30]: runcell('Question #10', '/Users/chloenam/Desktop/finalcapstone.py')
Total Number of Users:  279
Number of users who lean left:  159
Average age of users who lean left:  159
STD of age of users who lean left:  159
Number of users who are non-left leaning:  120
Average age of users who are non-left leaning:  0    19.575
dtype: float64
STD of age of users who are non-left leaning:   0    1.308068
dtype: float64
```

Total Number of Users: 279

Number of users who lean left: 159

Average age of users who lean left: 159

STD of age of users who lean left: 159

Number of users who are non-left leaning: 120

Average age of users who are non-left leaning: 19.575

STD of age of users who are non-left leaning: 1.308068

I created an array for age which indexes column 215 and an array for political orientation titled "poli_orient" which indexes column 217. I cleaned the data by checking for NaN values and creating a combined matrix array of both age and political orientation. I calculated the sum of left leaning users and their average age and standard deviation. I also calculated these same three for non-left leaning users. I then created my scatter plot to show for Age and Political Orientation. I did a Logistic Regression model and then used the Random Forest Classifier to predict the model accuracy. I also attempted to use the Area Under Curve function to comment on the classification quality of this model. AUC is calculated by plotting the TPR against the FPR and then calculating the area under the curve. A model with a high AUC will have a curve that is closer to the top left corner of the plot (representing a TPR of 1 and an FPR of 0), while a model with a low AUC will have a curve that is closer to the diagonal line (representing a TPR and FPR of 0.5).