Ray M. Liu
October 25, 2019
CS 7641: Machine Learning
Dr. Isbell

Unsupervised Learning

Overview: This paper will be a thorough analysis of six different unsupervised learning algorithms: (K-Means clustering, Expectation Maximization, PCA, ICA, Randomized Projections, and Feature Importance using the ExtraTreesClassifier) on two datasets of my choice. K-Means clustering and EM are both clustering algorithms whereas PCA, ICA, Randomized Projection, and Feature Importance are dimensionality reduction algorithms. For each of the six algorithms, we will be tuning the different hyperparameters for analysis and then finally we will be running a neural network learner on the newly projected data to evaluate the difference in performance using dimensionality reduction + neural network vs. solely neural network.

Dataset:

- The first dataset is the “Adult Data Set” from the UCI Machine Learning Repository. The dataset was obtained from the 1994 Census database with 48,842 data points, 14 features, and 2 target categories: More than 50k income or Less than 50k income.
- The second dataset is the “Graduate Admissions Data Set” from Kaggle. The dataset predicts admission rates based on these features: GRE score, TOEFL score, University Rating, SOP, LOR, Undergraduate GPA, and research experience.

Clustering Algorithms:

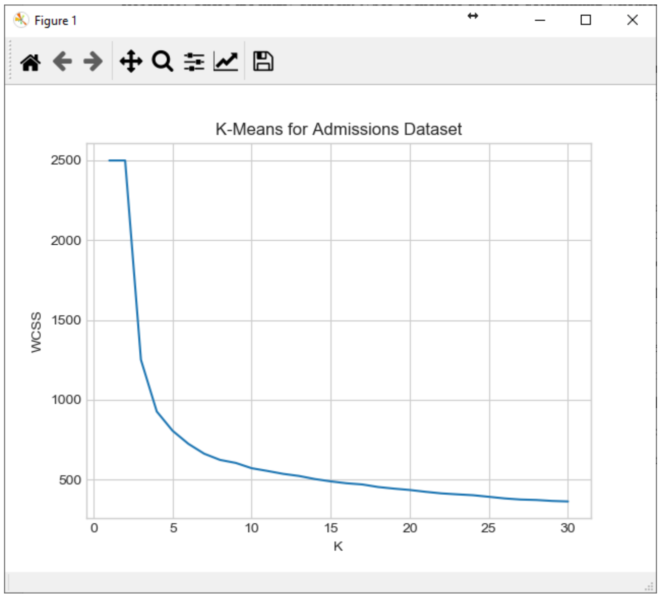
Clustering algorithms come in two flavors, supervised and unsupervised. In this paper, we will be talking about the unsupervised version of clustering algorithms. Clustering is a task where users group similar objects together into different clusters, where similar objects are grouped together and dissimilar objects are not grouped together and placed further apart. What defines closeness? There are many different types of metrics used for determining whether something is “close” to another. To name a few: Euclidean distance and the Manhattan distance.

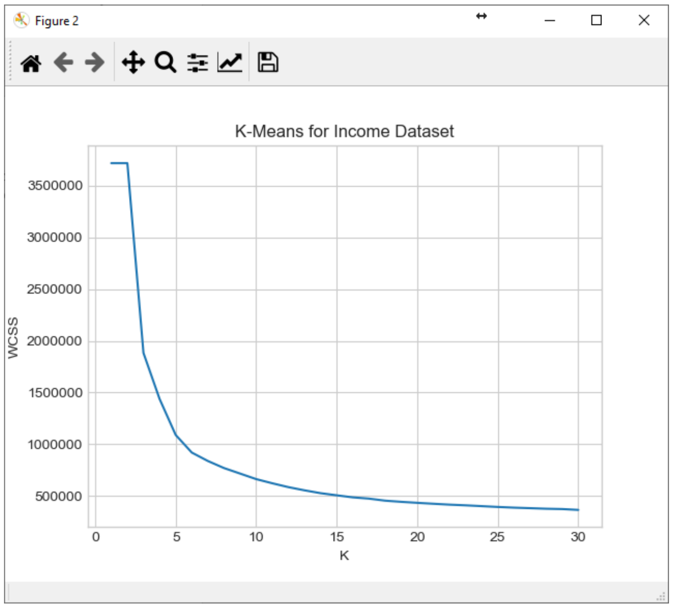
In this paper, we will be using the Euclidean distance metric for clustering algorithms. The reason as to why we chose to use the Euclidean distance metric is because it worked best for our problem.

Ray M. Liu
October 25, 2019
CS 7641: Machine Learning
Dr. Isbell

K-Means Clustering:

The first clustering algorithm we will be looking at is the k means clustering algorithm. The k means clustering algorithm is a simple and intuitive algorithm but also provides powerful results and has a huge amount of expressive power. K-means works as follows: we choose the number of clusters, k, and these k cluster locations are randomly chosen from out feature space so that data which are "closer" to these cluster locations are grouped within these clusters. After grouping all the data points, we compute the new center for those clusters and repeat this process of finding similar data points until we reach a point of convergence, where the centroid cluster has moved less than the threshold specified or it has not moved at all. The term "closer" which is used here is the distance metric used for the specific problem. This distance metric can be Euclidean distance, Manhattan distance, or a user specified distance metric of their choice.





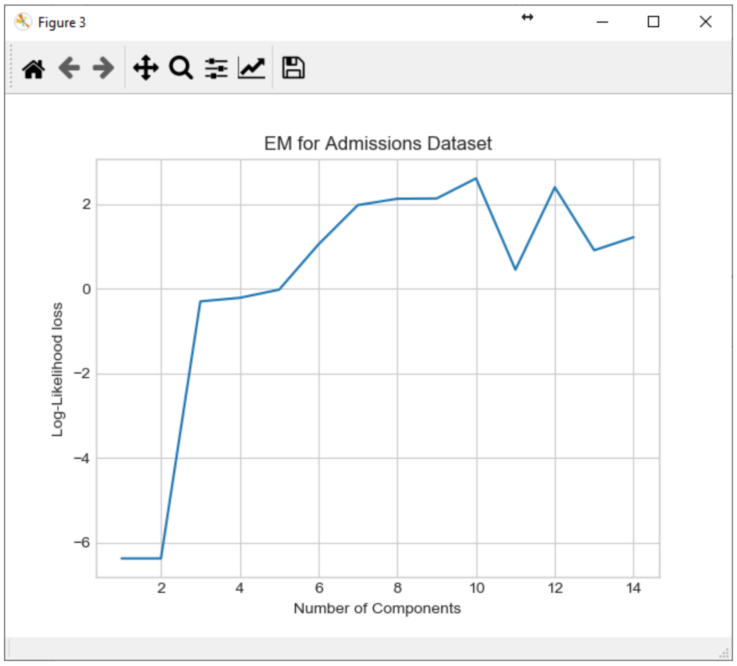
The k means clustering algorithm has only one hyperparameter of interest and that is the value K, which corresponds to the number of clusters for which the dataset will have. The hyperparameter K isn't something we can eyeball and require multiple tries to figure out. Therefore, for this problem we are optimizing the value of K for our specific datasets.

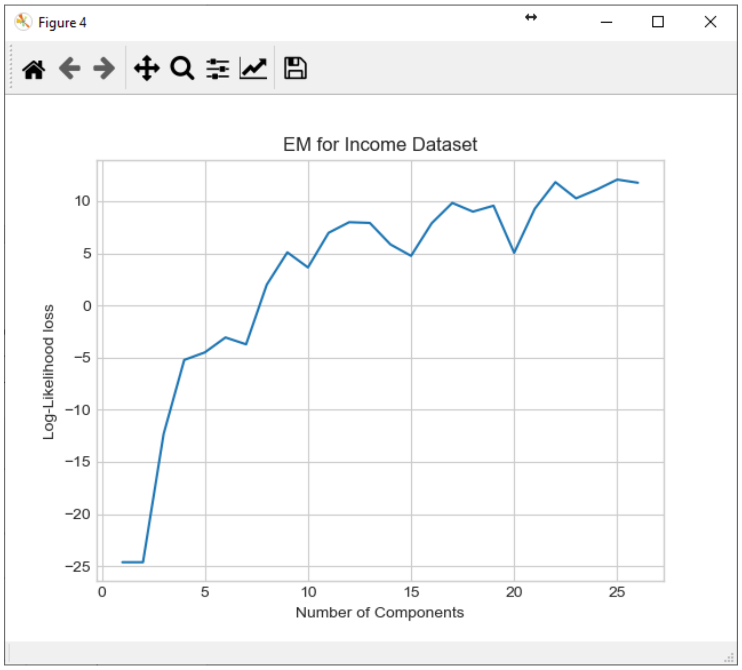
In the graphs displayed above, we are plotting WCSS vs. K. Where WCSS stands for the within cluster sum of squared errors. WCSS is the sum of squared errors for each cluster using the Euclidean distance as the metric and computing the distance from the data point to the cluster's centroid. We then average out all the WCSS for the K clusters that we have to compute the total WCSS for that specific k means.

Using the WCSS as the sole metric for a good value of K for k means isn't the best idea because we may overfit to the data. Therefore, we will be using the elbow method for finding a value of K that works decently well for the problem. The elbow method works as follows: we find the number of clusters such that adding another cluster doesn't give us much of a better model. So for our case, we choose a value K that doesn't decrease by WCSS as much. For the admission dataset, that value looks to be roughly 5 and for the income dataset, that value looks to be roughly 5 as well.

Expectation Maximization (EM):

The next clustering algorithm we will be looking at is Expectation Maximization. EM is very different from k means, where EM is a soft clustering algorithm. The main difference between EM and k means is that EM allows a point to be in multiple clusters vs. k means which says a point can only be in one cluster. For a point to be in multiple clusters, we assign probabilities for each data point to belong in each cluster. The probabilities associated assigned to the data point for each cluster represents the chance that the Gaussian distribution generates that sample data point for that cluster. Therefore, higher probabilities mean it is more likely for the data point to belong in that cluster and low probabilities represent very unlikely, but there still exists a small chance, that the data point will belong in that cluster. EM then proceeds to maximize the likelihood that the data belongs to certain clusters and minimizes the likelihood for data the does not belong to a cluster.





Running the EM algorithm on the dataset, we focus on only one hyperparameter: the number of components used. In the graph above, we show the log-likelihood vs. number of components for the EM clustering algorithm to determine what the best number of components for EM will be. A naïve way of choosing the best number of components for the EM clustering algorithm is to choose the one with maximal log-likelihood. This isn't the best approach because the highest log-likelihood is the best because it overfits to the data and will not generalize well to future unseen data. Therefore, the best approach to finding a decent result for the number of components is to either graph the data and the number of components for the EM algorithm to examine whether overfitting has occurred or to use the elbow method. For our problem, I chose 7 number of components for the admissions dataset and 9 for the income dataset because there isn't much of an increase in the log-likelihood beyond that point.

Conclusion:

Clustering Algorithm	Admission Dataset	Income Dataset
K-Means	5	5
Expectation Maximization (EM)	7	9

The results provided by these algorithms are not optimal and can be further improved since the elbow method was used to determine the optimal K values. The elbow method isn't the best method, but provides decent results that should not overfit to the data.

Dimensionality Reduction:

Dimensionality reduction algorithms are very different from all other algorithms in that they are tasked with either reducing the number of features within the data or increasing the number of features in the data. The goal of either increasing or decreasing the number of features is to increase the expressive power of the data. However, dimensionality reduction algorithms are mostly used for, as the name suggests, reducing the number of dimensions to solve the curse of dimensionality. By reducing the number of dimensions, the model requires less data to fill the feature space with hopes of obtaining shorter runtimes and better accuracy.

Dimensionality reduction algorithms has two flavors: feature selection and feature transformation.

Feature selection is the process of reducing the number of features in the feature space by choosing a select number of features with the most impact on the dataset. A great example is the importance of house size and what the weather will be tomorrow as features for determining the price of my house.

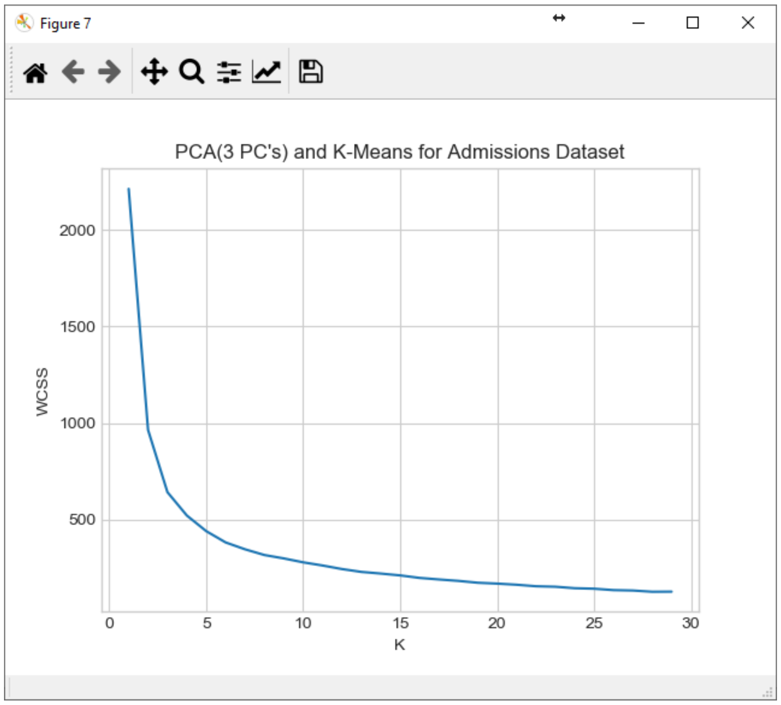
Feature selection will filter out tomorrow's weather as having minimal impact on the house price but will increase the value for house size as a relevant feature because it has an impact on the house price.

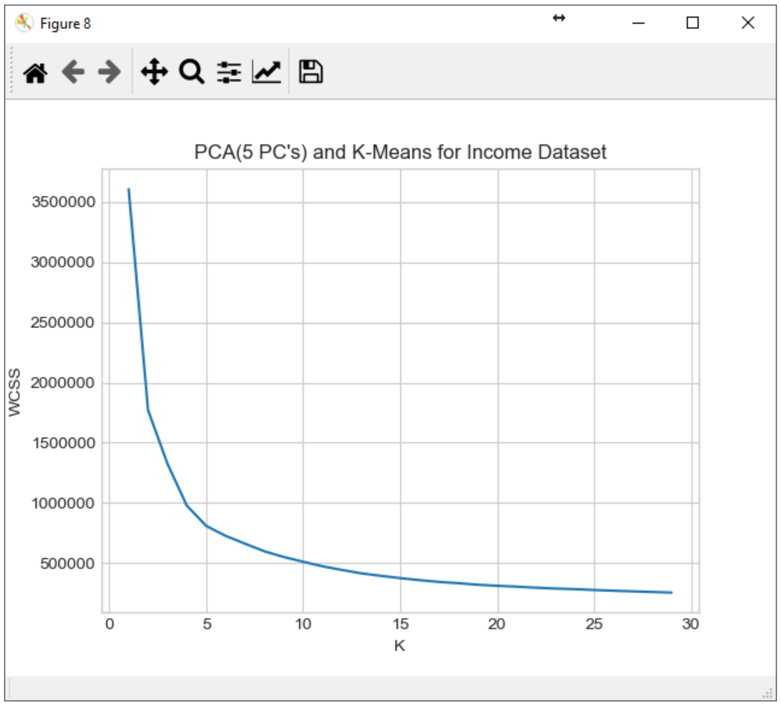
Feature transformation Is similar to feature selection in that they both reduce the number of features, but the process for which feature transformation reduces the number of features is that it combines multiple features together and creates a whole new feature whereas feature selection takes existing features, hence the name feature transformation. This is done by transforming the larger feature space into a smaller feature space while retaining as much information(variance) as possible. This in turn reduces the complexity of our problem and has the potential to make non-linearly separable data to become linearly separable, much like how kernels are used in SVM's. This makes feature transformation algorithms more powerful than feature selection algorithms.

Principal Component Analysis (PCA):

To start off, we will be examining PCA which is a linear dimensionality reduction algorithm that uses eigenvalues and eigenvectors. PCA belongs in the family of feature transformation algorithms and works by maximizing the variance for the transformed data. The newly transformed feature space is composed of principal components and this will be a hyperparameter of choice that the user must decide on. The goal of PCA is to retain as much variance as possible for the data while using only the number of principal components specified by the user. The algorithm works by first finding the best first principal component vector and then repeatedly finding orthogonal principal components vectors until we reach the maximum number of principal components specified.

What is the metric used to determine whether PCA is working well for our problem? There is no best way, but there are different metrics we can use to evaluate whether PCA is working for our data. If PCA can reduce the number of features needed for our dataset, that is a win-win situation because we are solving the curse of dimensionality. Therefore, as long as our data is representable in a smaller feature space, that means it's doing well. But then we have the issue of which exact number of principal components is the best to choose from? To solve this, we revisit log-likelihood, elbow method, and introduce a new method the Kaiser criterion. The Kaiser criterion states that we only keep principal component vectors where their eigenvalues are greater than 1.





The number of principal components, 3 and 5 were chosen using the log-likelihood graph generated from the code(not shown here) and using the elbow method of the log-likelihood to obtain an acceptable number for the principal components. Arguably, we can use the kaiser criterion for determining a better value for the number of principal components, but for the sake of simplicity and consistent results, I chose to use the elbow method.

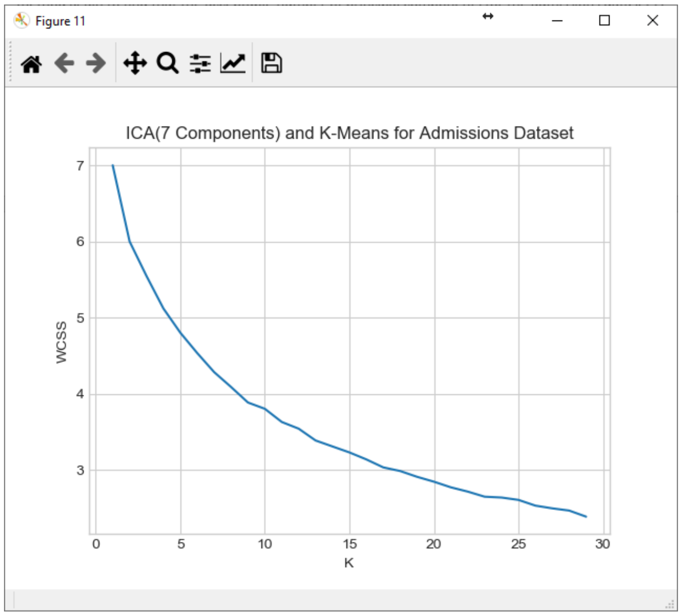
In the graphs above, we show the results of plotting a PCA transformation of the two datasets with 3 and 5 respective principal components with different values of K and its corresponding WCSS score. We can see that the data starts to overfit as K increases and so we use the elbow method again to determine an acceptable value for K. From the graph above, we can see that an acceptable value for K will be 5 for both the admissions dataset and the income dataset. This means that the number of clusters for with PCA and without PCA hasn't changed since we're still using a K value of 5.

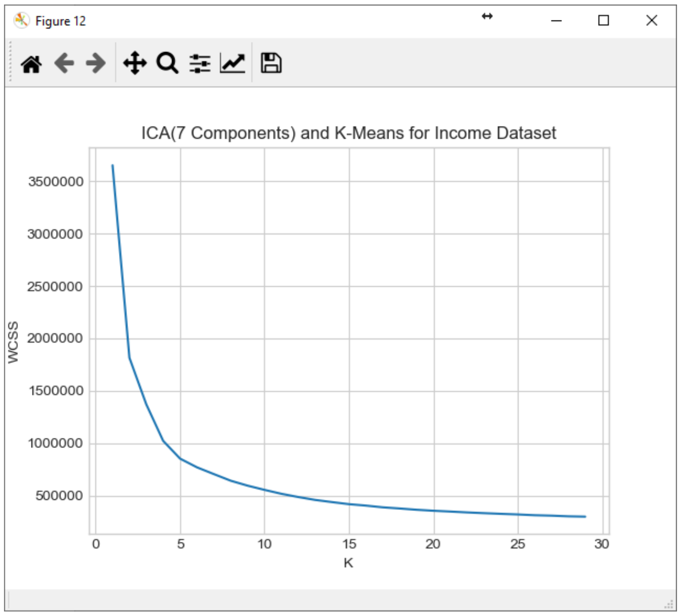
For EM, the graph isn't displayed here but has very interesting results where the trend isn't as spiky as we saw previously in the EM findings. The graph is more smoothed out. For EM, we use the elbow method again to find that the acceptable number of principal components for the admissions dataset is 3 and for the income dataset the acceptable number of principal components is 2.

In the future, another possible metric for analysis is to use the kaiser criterion to filter out principal components with eigenvalues less than 1.

Independent Component Analysis (ICA):

The next dimensionality reduction algorithm we will be examining is Independent Component Analysis (ICA). ICA belongs in the family of feature transformation algorithms and is similar to PCA but attempts to achieve the result of dimensionality reduction in a way that is different from PCA. ICA assumes that there are hidden independent features within the dataset and its goal is to find derived independent components from the features specified within the dataset. The independent components themselves provide no information to each other, but when combined together they maximize the amount of information. This essentially allows us to reconstruct the original dataset.





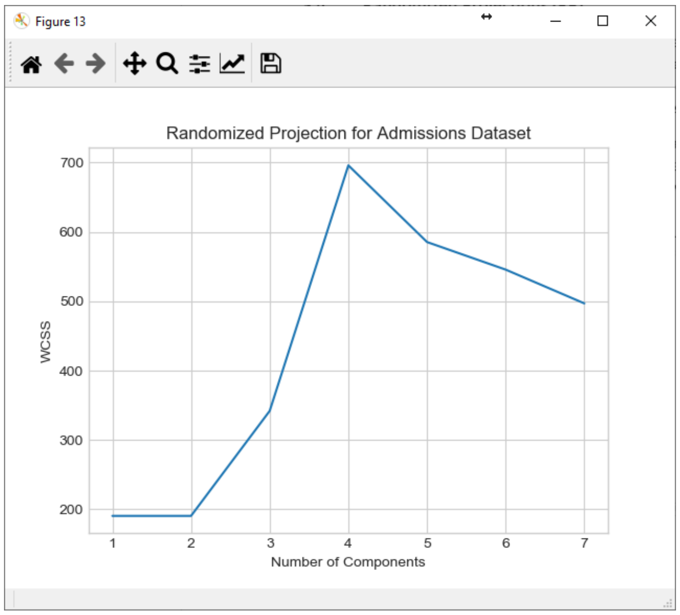
We run ICA on both datasets and graph the results with their corresponding kurtosis score (in the code, but not here). A univariate normal distribution has a kurtosis of 3 which means we should try to aim for a kurtosis of 3 for a normal distribution of our ICA transformed dataset. In the graph not shown here, a kurtosis of 3 for the admissions dataset corresponds to 7 components for ICA and a kurtosis of 3 for the income dataset also corresponds to 7 components.

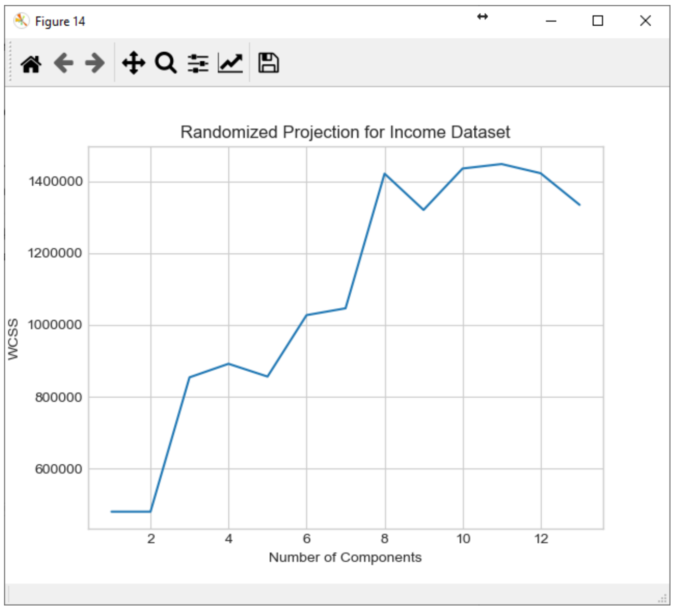
Using 7 components for ICA, we plot the graph for WCSS vs. K and using the elbow method, we can deduce that an acceptable value of K for the admissions dataset will be somewhere around 10-15 and for the income dataset, the value of K will be 5 which matches some of our previous findings. For clustering, we can see that we are still maintaining a value of 5 clusters for the income dataset regardless of whether we use ICA or not for K-means. However, things have changed for the admissions dataset where the number of clusteres is not somewhere in the range of 10-15 and not 5 anymore. So with ICA, we can see changes in the number of clusters required for the admissions dataset.

For EM, the graph isn't displayed here (shown in the code), but the number of principal components for the admissions dataset is 2 and the income dataset is 6. The values were chosen using the log-likelihood and elbow method.

Randomized Projections (RP):

Randomized Projection will be the final feature transformation algorithm for dimensionality reduction that we will be examining. RP runs significantly faster and provides acceptable results. RP works as follows: we choose the number of components and the algorithm takes that information to generate a randomized matrix for transforming our feature space into a smaller space with the specified number of dimensions. RP's use a randomized matrix for transformation which means our results are sensitive to the seed value which is why I decided to specify a seed value for consistent results.

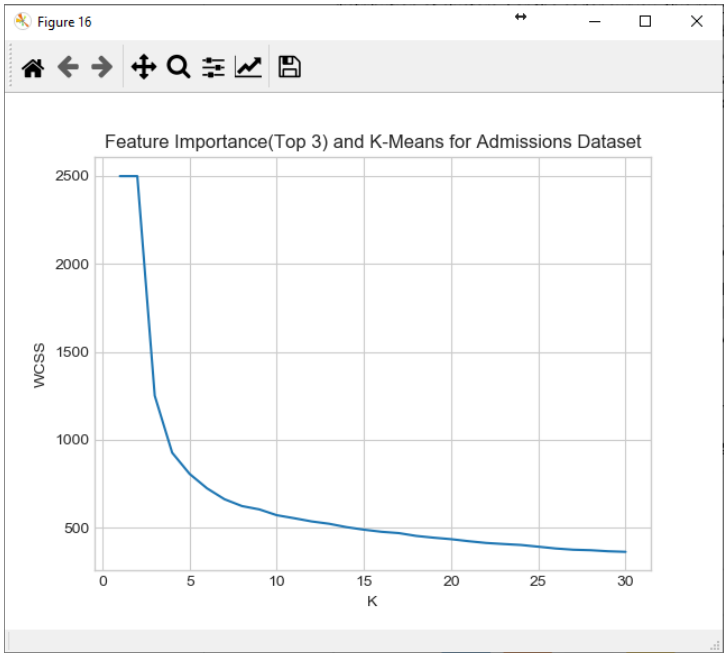


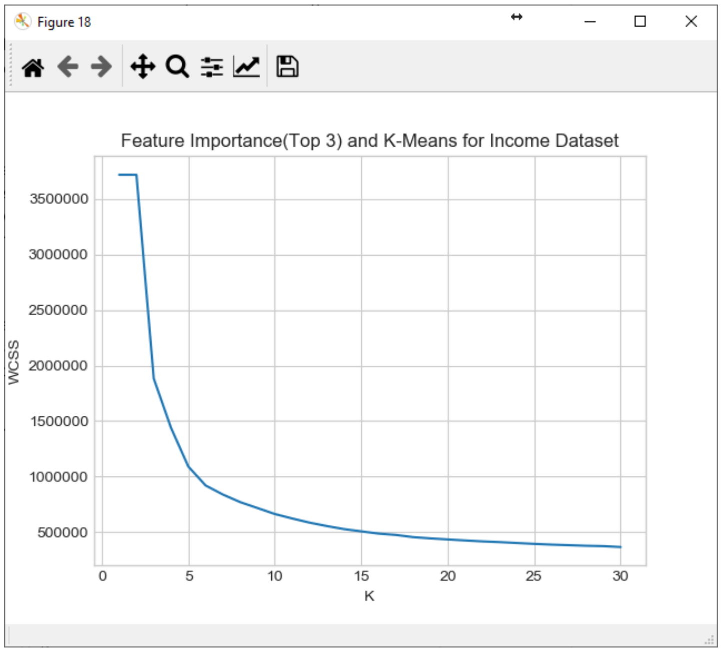


A K value of 5 and multiple iterations was used for RP since that was the value, we consistently found for both PCA and ICA. We can see that RP wants us to use 2 number of components for both datasets. This differs significantly from our PCA and ICA findings where we are recommended 3 and 5 components for PCA and 7 and 7 components for ICA. RP was run multiple times for examining how different the graphs turned out to be upon every iteration. And it was expected that the graphs turned out to be different on every iteration.

Feature Importance (ExtraTreesClassifier):

Feature Importance is a dimensionality reduction technique that belongs in the family of feature selection. It works by creating a decision tree classifier for obtaining the features that provide the most information for our dataset. The tree then provides a value for each feature and how important they are for the dataset. We then rank the features from high to low, for to how impactful the feature is.





According to the graph (not shown here) for feature importance, the most important features for the admissions dataset is CGPA, GRE Score, and TOEFL Score. For the income dataset, the most important features are fnlwgt (the estimated number of people the data entry represents), age, and education level. The findings aren't too surprising for both datasets because GPA, GRE score, and TOEFL score are all important indicators for whether one will be accepted into graduate school or not and for the income dataset, fnlwgt (the estimated number of people the data entry represents) tells us how many people the data entry represents and if it represents more people, the more reliable it is which intuitively makes sense. And as for age and education level, the older and more educated one is, the more income they will make which coincides with what is seen in society. Where smarter, older workers make more money than younger less educated workers.

We can see that the K value after taking the top 3 features stays at 5 for both datasets. The findings here match the number of clusters needed for PCA, ICA, and just k-means on the raw data.

Conclusion:

Dimensionality Reduction	Admissions Dataset	Income Dataset
PCA	KMeans: 5, EM: 3	KMeans: 5, EM: 2
ICA	KMeans: 12, EM: 2	KMeans: 5, EM: 6
Random Projection	KMeans: 5, EM: 5	KMeans: 5, EM: 5
Feature Importance	KMeans: 5, EM: 5	KMeans: 5, EM: 5

Results for Neural Network:

Dimensionality Reduction	Admissions Dataset	Income Dataset
None	0.83	0.83
PCA	0.6175	0.803
ICA	0.6275	0.754
Random Projection	0.6175	0.754
Feature Importance	0.6175	0.754

The results are that our neural network performed worse when dimensionality reduction was applied first on the dataset before being passed it into the neural network.

Final Conclusion:

While experimenting with the clustering and dimensionality reduction algorithms, there was already an indication that our neural network will perform worse with the reduced dimensions. This was because we used the elbow method for estimating the number of dimensions. The elbow method provides us with acceptable results but no the best. Using the elbow method means that there will be a loss in information upon reducing the number of dimensions but we are usually guaranteed to not overfit which is what we see in the results for the neural network. Using a more reliable method for determining the correct number of dimensions for the dimensionality reduction algorithms should provide an accuracy that's close to the baseline accuracy (no dimensionality reduction applied) if not better. Although, I learned a lot from this assignment because I found that ICA is used to solve problems like the "cocktail party problem", PCA is great for compression because it reduces the number of dimensions whilst retaining as much information as possible, and feature importance is great for analyzing which features are more important than others in a business setting.