Isabella Silva
CS4641
Machine Learning

Assignment 3: Unsupervised Learning

For this assignment I will be analyzing and understanding the following unsupervised learning algorithms:

K-Means Clustering
Expectation Maximization
PCA
ICA
Randomized Projections
Linear Discriminant Analysis

After running each of the clustering algorithms, K-Means and Expectation Maximization, I will run the Dimensionality Reduction algorithms on my datasets. I will then run the clustering algorithms *again* to see the effects of the dimensionality reduction on the clustering algorithms. Finally, I will rerun my original neural network on the processed data, and examine the final results.

I will be using the same two datasets as before: the first of which explores exoplanet stars and the second which determines country happiness.
https://www.kaggle.com/keplersmachines/kepler-labelled-time-series-data
https://www.kaggle.com/unsdsn/world-happiness
For further information on my datasets, please see the "Dataset and Classification" portion of my Assignment 1.

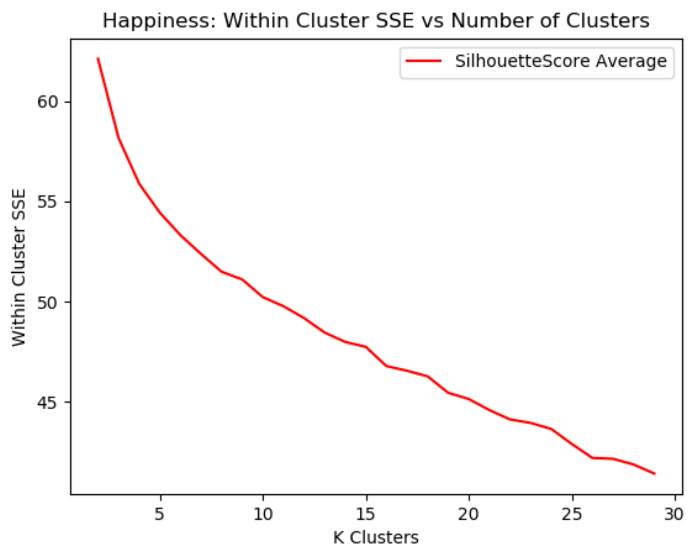
I will once again be using Python to construct these unsupervised learning algorithms with the Scikit learn library and using WEKA to perform my dimensionality reductions.

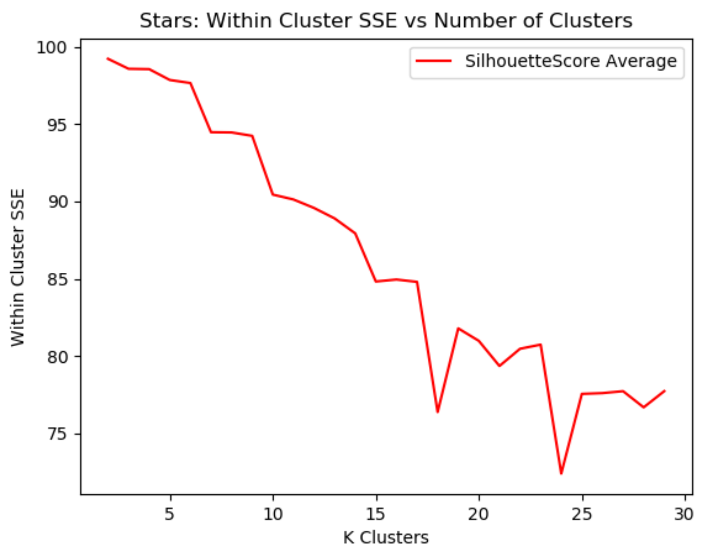
K-MEANS CLUSTERING

Hypothesis: The accuracy of this algorithm will be dependent on the value of K. I believe that the higher the value of K the Sum of Squared Error will be lower.

K-Means clustering is a learning algorithm that takes a given set of data and works to partition or 'cluster' points into k sets. The sets are determined iteratively. The first step of the algorithm is to choose k points randomly. These randomly chosen points are known as 'centroids' because they are meant to be the centers of the clusters and they generate clusters based on the nearest points to those centroids. In this case 'nearest' mean cluster is defined as the cluster with the mean least squared Euclidean Distance, although some other distance metrics can be used they do not guarantee the optimum. The new means are then treated as the centroids for the next iteration of the algorithm to further determine an accurate cluster. The algorithm terminates when the cluster centroid assignments remain the same, also known as convergence. This algorithm is *not* guaranteed to find the optimum.

I expect that a larger value for K will decrease the Sum of Squared Error (SSE) because, similar to KNN, a larger number of clusters will allow for more specific feature selection. However, I K-Means is not susceptible to overfitting in the way that KNN is. This is because K





Means creates clusters based on categories of features, thus even when K is equal to the number of data points, we will only have specific categorized boxes to place new data points into.

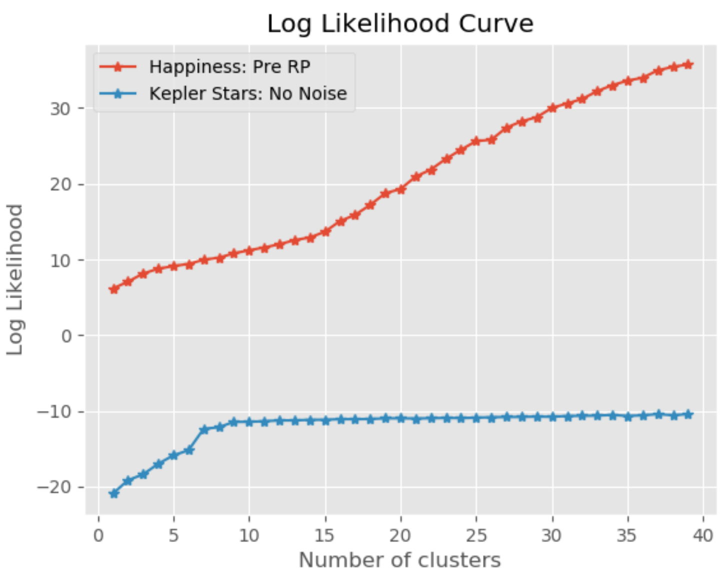
The data displayed on these graphs supports my hypothesis as it shows that an increase in the number of k clusters does indeed result in a lower error. The World Happiness dataset follows an elbow curve where we can see a somewhat distinct point at which the error stops decreasing dramatically and begins to even out. There is a slight similarity to this curve for the Kepler Stars data, however it is more of a trend with large spikes due to noise. Over the past few assignments, this dataset has shown to be far more susceptible to spikes and dips. I suspect that this is a result of the enormity of this particular dataset given that the size is the primary difference between the World Happiness Dataset and the Kepler Stars Dataset.

EXPECTATION MAXIMIZATION

Hypothesis: I think that Expectation Maximization will have a similar output to K-Means because of the similarity between the two algorithms.

Expectation Maximization is another clustering algorithm that works similarly to K-Means in the sense that it also takes data points and clusters them around a chosen centroid. It uses this method to maximize the overall probability of the data points given the clusters. However, EM differs from K-Means in that it does not assign clusters to maximize difference between data based on features. Instead EM computes the probability that a data point will belong to a cluster based on one or more probability distributions. EM is a 'soft' clustering algorithm, meaning that each data point has the potential to be a part of *n* different clusters where *n* is at most the total number of clusters.

For this implementation of Expectation Maximization I am going to look at the effect of number of clusters on the log likelihood function. This function is how I plan on representing the effect of the EM algorithm on the probability of a given data point being assigned to the correct cluster. The log likelihood function curve is used to portray the maximum likelihood of a given point belonging to a feature cluster. This portraying is derived via solving the log likelihood equation for a given value that maximizes the log likelihood of a point.



As we can see from the graph above, the Kepler Stars Dataset has a clear point where it maximizes around 9 clusters and we see the log likelihood plateau at -10. On the other hand, the World Happiness Dataset has a progressive optimization where it continually improves. This data is somewhat consistent with the data we received through K-Means. However, K-Means was *less* optimal on the larger dataset. This is visible as both algorithms display an optimal number of clusters before reaching a plateau. The EM algorithm reaches an optimal point more quickly and with fewer clusters whereas the K-Means algorithm does not begin an obvious maximization until more than 10 clusters.

Over all, this data supports my hypothesis as it shows that the EM algorithm treats data similarly to the K-Means algorithm. EM also yields similar results to K-Means with regards to the optimal value of *k* for the number of clusters needed to maximize the the log likelihood.

DIMENSIONALITY REDUCTION ALGORITHMS

Hypothesis: I believe that the dimensionality reduction algorithms will affect the Kepler Stars dataset the most because much of the noise is due to the large number of features.

Dimensionality Reduction algorithms are a type of support algorithm that allow machine learning programs to, ideally, work more effectively and optimally. The idea behind dimensionality reduction is that a set of principles can be used to reduce the number of variables or features under consideration. This bears a striking similarity to the principal of pruning in decision trees, however dimensionality reduction uses feature selection, or extraction, to discriminate between variables.

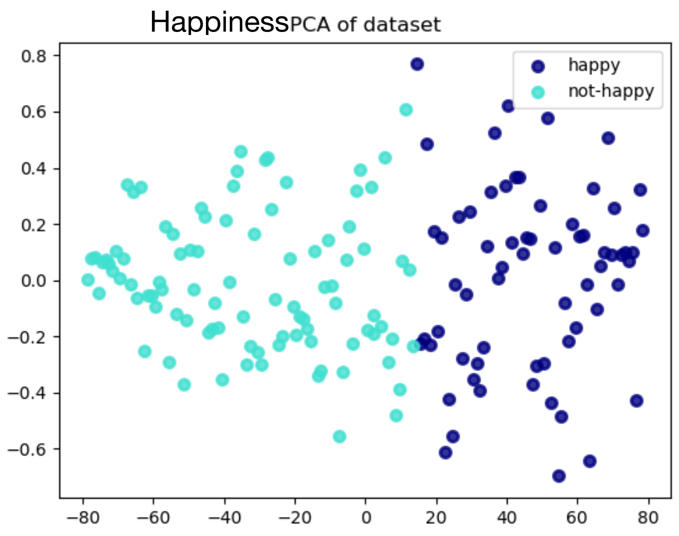
PRINCIPAL COMPONENTS ANALYSIS

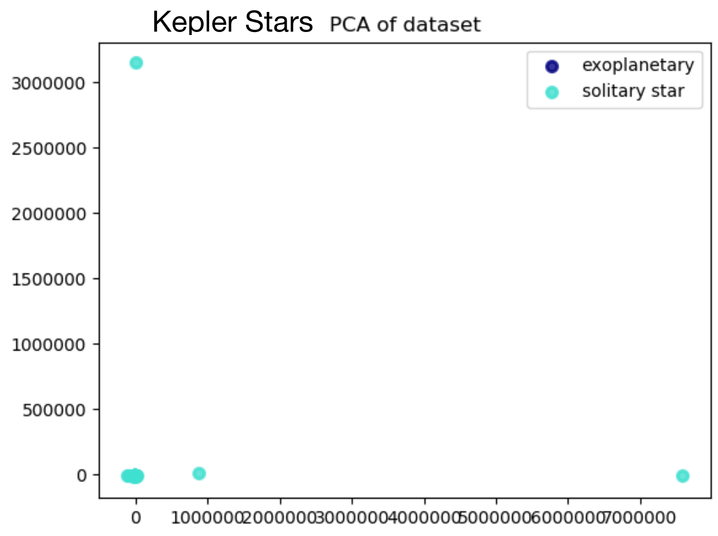
Hypothesis: I expect this dimensionality reduction algorithm to improve the classification outcome of the clustering algorithms above.

The Principal Component Analysis, or PCA, algorithm is a dimensionality reduction algorithm that uses the variance of the features of the data to maximize the data separability. This allows it to cluster data more accurately and acutely. PCA linearly maps data features onto a lower dimension where the feature variance is then maximized. The principal components are then reconstructed to create a large fraction of the variance of the original data. This algorithm will ideally remove redundant variable features from the dataset that do not contribute much to cluster definition, once again functioning similarly to pruning.

Using WEKA, the Principle Components Attribute Evaluator showed that there were 4 of the 12 attributes in the World Happiness Dataset that reflected the output the most strongly. The clustering returned by PCA on the happiness dataset was fairly accurate.

For the Kepler Stars Dataset there was a very surprising effect showing that there were only 3 attributes out of the thousands. The clustering on the stars dataset was far less effective.





	Dataset	Reduced Number of Attributes	Original Number of Attributes	Standard Deviation
K-Means	World Happiness	4	12	< 0.5
	Kepler Stars	3	3197	4
Expectation Maximization	World Happiness	12 (Full Data)	12	< 1
	Kepler Stars	4	3197	10

The data shown above was fascinating to me for a variety of reasons. The foremost is that the dimensionality reduction for the Kepler Stars Dataset was so immense. One of the primary problems that I have faced with the Kepler Dataset is the enormous size resulting in overfitting. Being able to reduce the number of attributes from thousands to only 3 made me immediately skeptical of the accuracy of the results as well as hopeful with how they would react to previous algorithms. Assuming that the dataset remains accurate, I am excited to see this change in data reflect in noise reduction.

For the World Happiness Dataset, I find these results even more interesting in some ways. Two of the primary attributes that were chosen by WEKA as particularly indicative of a country's happiness were:

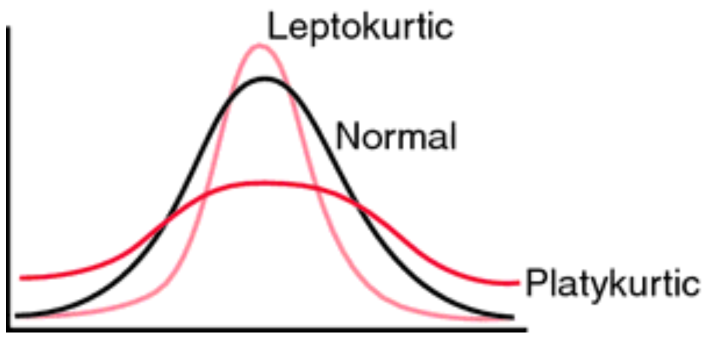
Happiness Rank
Happiness Score

This was particularly interesting to me because it is data that is clearly positively correlated to a country's binary happiness score. Given a high ranking and a high score it can be assumed that the country's output will be 'happy' and vice versa. I find this interesting because it is so clear, that it is virtually useless to a human analyzing this data. Assuming a person had the data regarding a country's happiness score and rank, it is unlikely that they would need to predict further if the country is considered happy or not. However, from the perspective of the algorithm, these two attributes were the most highly correlated and the most useful in determining an output. The algorithm recognized that GDP, Honesty, Health, and the other attributes were not nearly as indicative. This is something I had not even considered before running this algorithm because in my mind it was clear that looking at these two attributes (Rank and Score) were a result of the other attributes, a form of output in their own ways. Given a set of data regarding basic country statistics, I think this algorithm would perform differently and I would like to test it further on different datasets to see the extent of its reliability.

INDEPENDENT COMPONENTS ANALYSIS

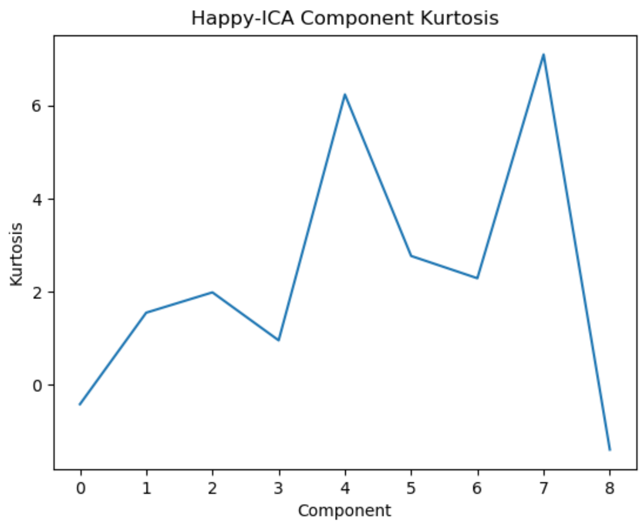
Hypothesis: I think that Independent Component Analysis will have the same results as Principal Component Analysis.

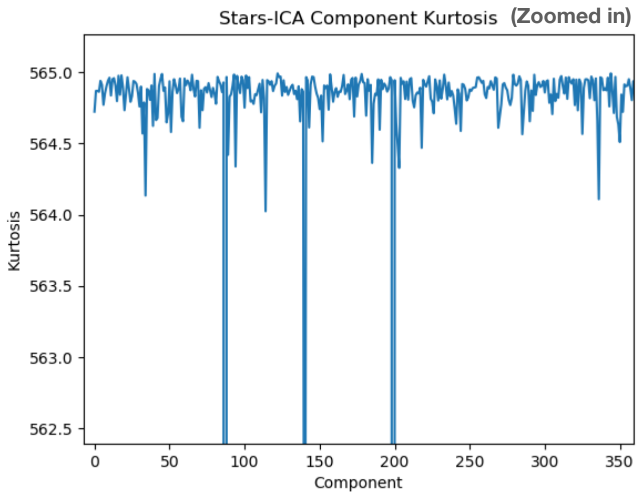
The Independent Components Analysis, or ICA, algorithm is also a linear dimensionality reduction algorithm that creates columns based on the independent components of the dataset. These independent components are determined under the assumption that each data sample is made up of other independent components. Two components are said to be independent if there is an absence of correlation between the two features. Another way of explaining this is by saying that new information regarding one data point does not influence knowledge of a secondary point. This is a sufficient definition of independent variables in the case of ICA because it is a *linear* dimensionality reduction algorithm, and correlation measure linear dependence.



Kurtosis is a measure of a graphs similarity to the Gaussian normal distribution. The kurtosis of a normal distribution is 3 whereas less than 3 is known as platykurtic and more than 3 is known as leptokurtic (as seen below taken from medical-dictionary.thefreedictionary.com). Figure 1: *https://medical-dictionary.thefreedictionary.com/Excess+kurtosis*

Independent Component Analysis shows a similar number of components as PCA for the World Happiness Dataset. As can be seen from the graph above, the number of components that results in the normal distribution is 3, just slightly under the components returned by PCA using WEKA. This reaffirms the reliability of the PCA results for the World Happiness Dataset, however the same cannot be said for the Kepler Stars Dataset.



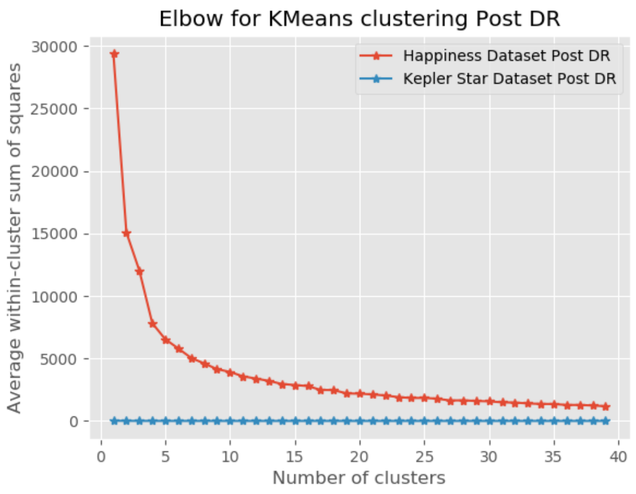


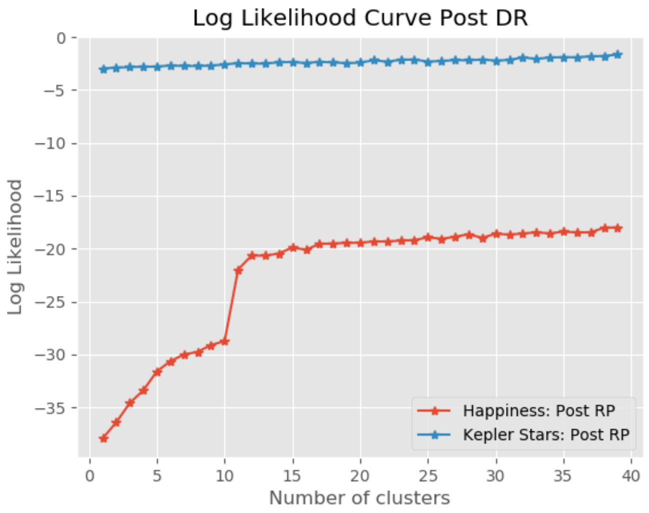
The minimum kurtosis value for this dataset is 281, leagues above 3 and showing a steep curve in the distribution of the data. This kurtosis value occurs at 200 components and does not show the same accuracy that was given by PCA. For this much larger dataset only the outlier values were correctly affected by the dimension reduction which shows that ICA was not an effective algorithm for the larger Kepler Stars dataset.

RANDOMIZED PROJECTIONS

Hypothesis: I think that Randomized Projections will have less accuracy than PCA and ICA, because it is an algorithm that generally sacrifices accuracy for speed and less memory usage.

Randomized Projections are comparatively simpler and produces fewer errors on average than the previous two algorithms and are commonly used in natural language processing (Randomized Indexing). Randomized projection works in accordance to the Johnson-Lindenstrauss Lemma. This lemma states that any vectors in a given high dimension, *d,* can be projected onto some lower dimensional space, *k,* while preserving the inter-point distances,





assuming that the points lie on a Euclidean Space and $k < d$. The hyper parameter that I worked with in this case was the number of dimensions.

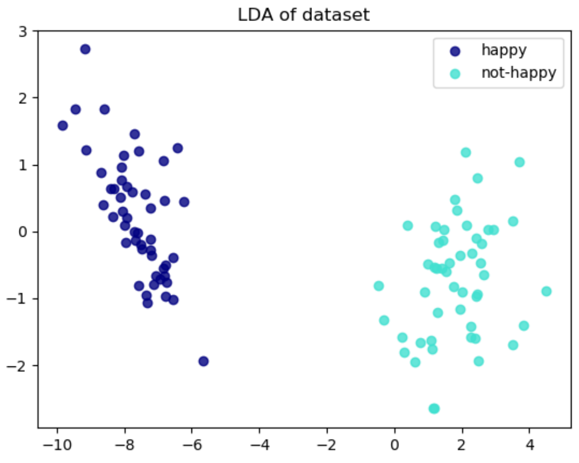
The K-Means Algorithm performed considerably more accurately on the dataset with the feature selection performed by the Randomized Projections algorithm. As the number of dimensions increased, the error for each dataset decreased drastically. The graphs themselves resemble the graphs returned by K-Means, *except* that the cluster error is considerably lower for both datasets. After running Expectation Maximization there were very similar results showing that once again RP was an effective algorithm.

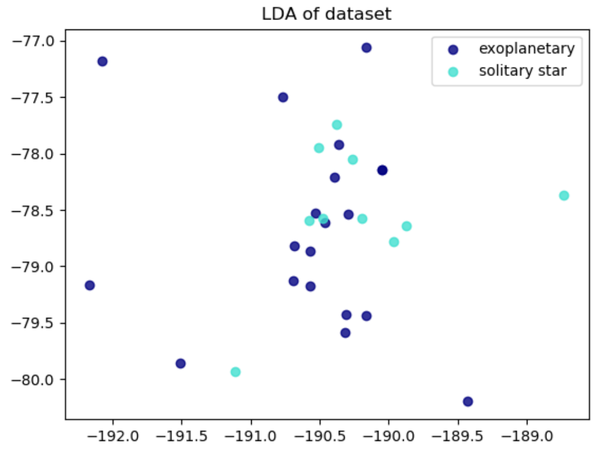
Over all, Randomized Projections proved to be a relatively accurate dimensionality reduction algorithm for both datasets, although it was far more effective on the World Happiness Dataset.

LINEAR DISCRIMINANT ANALYSIS

Hypothesis: I believe that LDA will be more accurate than PCA, but less accurate that RP because RP uses a similar concept but maintains linear distance across dimensions whereas LDA does not.

Linear Discriminant Analysis is a supervised dimension reduction algorithm that expresses one feature as a linear combination of other features. LDA maps the difference between feature classes to reduce dimensions. LDA is similar to PCA in the way that it discriminates between features using feature classes, *however* LDA distinguishes between independent variables and dependent variables whereas PCA, or factor analysis algorithms, do not. LDA is able to choose between independent and dependent variables to appropriately reduce the dimensions that are dependent. The dimension reduction performed by LDA was far more effective when determining accuracy than PCA as it had a similar effect but to a higher degree.

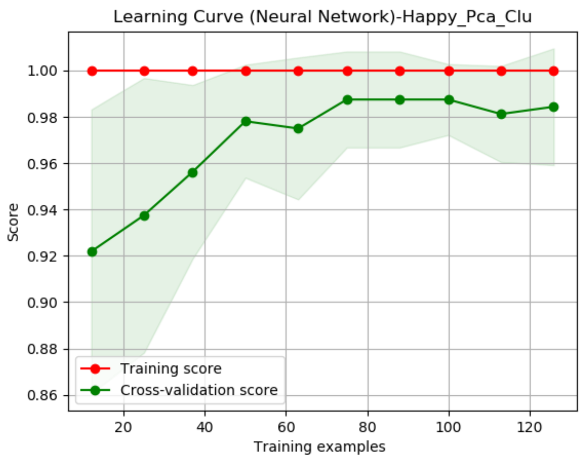


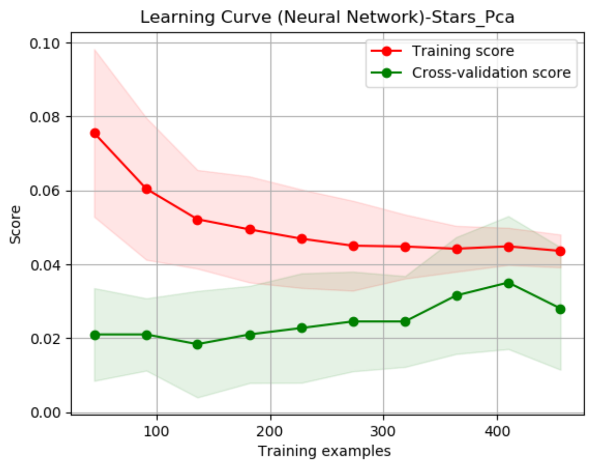


As we can see from the datasets above, they cluster similarly to PCA, but with a clearer distance between clusters. This supports my hypothesis that LDA is more accurate than PCA, however it is also more accurate than RP. The graph above (to the left) shows the clear distinguishing cluster between the binary outputs of the data which are labelled by ‘happy’ and ‘not-happy’. LDA accurately determined the difference between clusters where the axes are the prominent feature classes in the World Happiness Dataset. The LDA algorithm was less effective on the Kepler Stars dataset when determining the if a star was ‘exoplanetary’ or ‘solitary’. However, it is still more effective than the other two algorithms. In this case the clusters are still difficult to distinguish, but there is a vague line between the two clusters. Meanwhile the other algorithms gave virtually no information for the larger dataset. This supports the fact that the LDA algorithm is one that is meant to be more accurate when applied to particularly large datasets.

NEURAL NETWORK WITH CLUSTERING

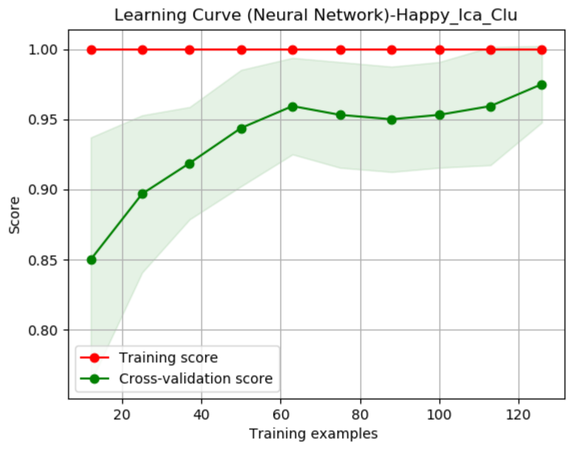
After running the neural network on the various different clustered datasets, I found many different results regarding which clustering algorithm I had run on the data previously.

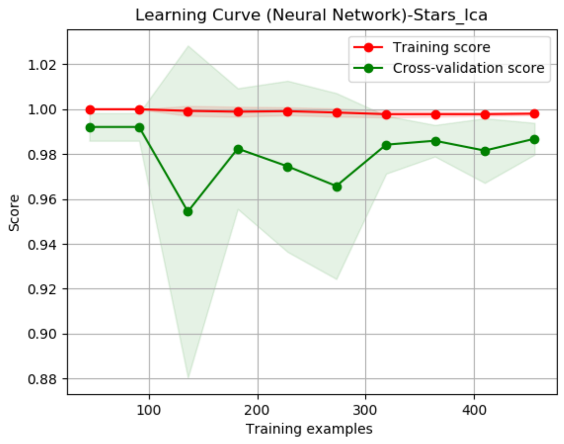




When my Neural Network was run on the dataset that was clustered using Principal Components Analysis, the cross-validation score was positively trending. On the World Happiness Dataset, the cross-validation score shows a clear positive effect of the NN showing that around 80 training iterations returned the maximum accuracy. On the other hand, the PCA reduced dataset was very inaccurate when the NN was run on it. The accuracy of the NN was below 1%. The Kepler Stars dataset also had an odd effect on my training score where the training score actually *decreased* as the iterations continued. This shows some overfitting to the data which could be due to the fact that the Kepler Stars Dataset became too small after dimensionality reduction. This makes sense as the dimensionality reduction from PCA (as can be seen above) was far too drastic and resulted in very sparse clusters.

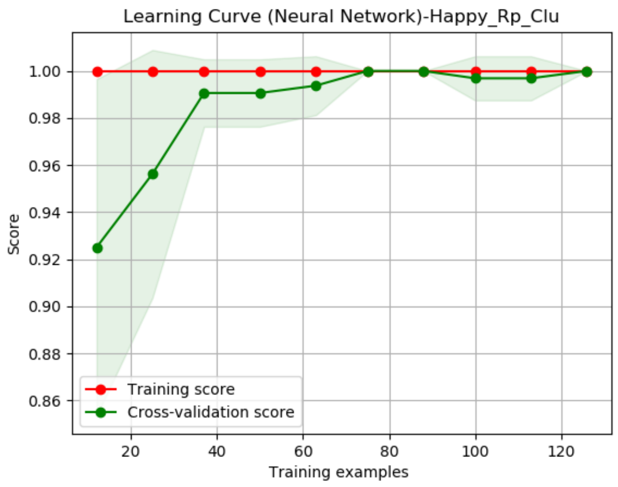
When my Neural Network was run on the dataset that was clustered using Independent Components Analysis, the cross-validation score was very similar for the two datasets. Further, the accuracy was far higher on average for the ICA reduced dataset as opposed to the PCA. For the World Happiness Dataset, the accuracy still reached almost 98% accuracy at 120 iterations, however it was lower up until that point. In this way ICA underperformed the PCA iteration per iteration, but matched it over all. For the Kepler Stars Dataset the ICA reduced data was *far* more

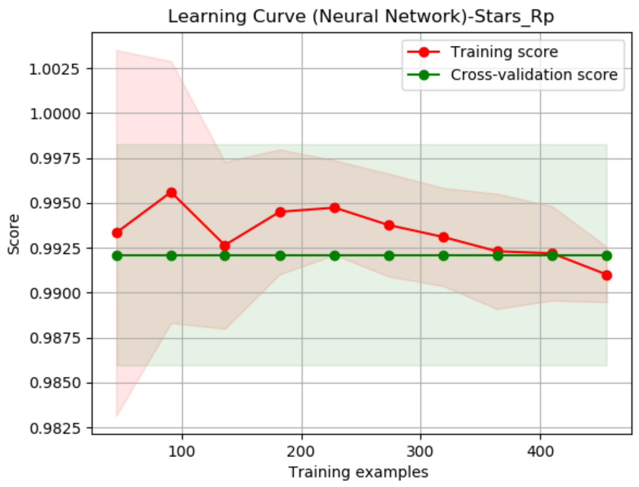




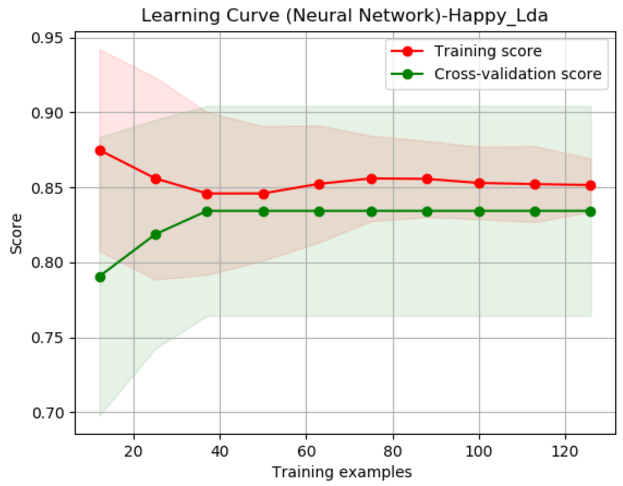
accurate than the data produced by the PCA reduction. Once again, this outperformance is intuitive due to the poor reduction performed by PCA. ICA provided very clearly featured data that worked well with my Neural Net, resulting in over 98% accuracy despite the spikes in the beginning due to noise.

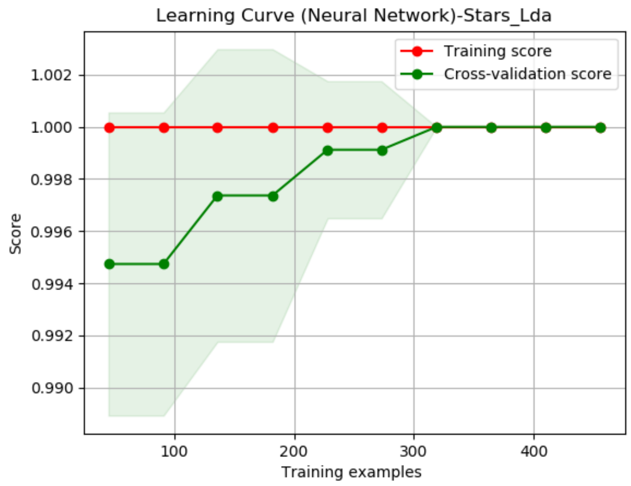
The World Happiness Data that was reduced using Randomized Projections once again optimized at 80 training examples. This maximum accuracy was higher than both PCA and ICA when my Neural Network was run on it. This further shows that RP was an effective algorithm in reducing the World Happiness Dataset so that it could be used by the Neural Net to produce accurate results. The effect of RP was also positive on the Kepler Stars Dataset as it had a very





high accuracy. Unfortunately the perfectly straight trend in the cross-validation score paired with the decreasing training score shows us that the Neural Network strongly overfit the RP reduced data.





Finally, Linear Discriminant Analysis was highly effective on the much larger Kepler Stars dataset, creating a reduced dataset that resulted in a very high accuracy from my Neural Network. This is supported by the fact that LDA is known for being effective on larger datasets. However, LDA had an odd effect on the World Happiness Dataset by overfitting *and* having the lowest accuracy of the four datasets processed by the Neural Network. Between 70 and 90 iterations, where the other datasets showed optimal accuracy, the Neural Network run on the LDA reduced World Happiness Dataset remained constant with no improvement. This shows overfitting that I believe is due to the dataset having too few features for the Neural Network to effectively learn on.

Overall, there were once again different results for the two different datasets. The Linear Discriminant Analysis is the most effective Data Reduction algorithm to preprocess the large Kepler Stars Dataset when the learning algorithm being used is a Neural Network. On the other hand, Randomized Projections is the optimal Data Reduction algorithm to use for the smaller World Happiness Dataset.

CONCLUSION

Of the Dimensionality Reduction Algorithms that I explored, the one that I found the most interesting was Linear Discriminant Analysis. I think that this algorithm is what I would consider the 'best' algorithm to use because of the way that it handled datasets of different sizes. A problem that I have faced repeatedly when using different algorithms in conjunction with the World Happiness and Kepler Stars Datasets is lack of robustness. Many algorithms that I have looked at have either been accurate for the smaller dataset and performed poorly on the larger dataset, or overfit the smaller dataset and were accurate on the larger dataset. Linear Discriminant Analysis was able to reduce both datasets so that they could be clustered with high accuracy. The Neural Network was then able to learn from these reduced datasets with peak accuracy for the larger dataset and above 80% accuracy for the smaller dataset.