Unsupervised Learning

Sagnik Bhadra

Georgia Institute of Technology
sbhadra8@gatech.edu

November 7, 2021

I. Abstract

The two clustering algorithms which are used on the wine and ufc dataset are K-Means and Expectation Maximization. Then there were four dimensionality reduction algorithms used on the datasets: PCA, ICA, Randomized Projections and SelectFromModel. After which the K-Means and Expectation Maximization clustering algorithms were used used on the newly produced datasets. Lastly, the neural network classifier was used to analyze the dataset.

II. Datasets

i. The Wine Dataset

The wine dataset is a very basic classification dataset which has already been pre-processed and is well-behaved. The dataset contains 13 features which are a chemical analysis of wines grown in a region in Italy. All of the features are continuous. There are three different cultivars which are the classes or target.

The dataset used has 5000 instances and there are no missing values since it is pre-processed. It is also divided almost equally among the classes which are represented as $1, 2, 3$ in the data. However, in order to achieve an accurate weighting on the features, the data was normalized. The *MinMaxScaler* function was used to normalize the data. Once the data is normalized, it is split up into training and test data. 80% of the data is used for training and 20% of the data is for the testing set. The $StratifiedShuffleSplit$ is used for this split as it preserves the percentage of samples for each class.

The weighted average of the precision and the recall, $f1\_score$, was used as the metric while trying to optimize hyperparameters. Since the classes are evenly represented, the average of the predictions are weighted evenly.

ii. UFC Dataset

The other dataset with was used is the UFC dataset. This dataset has about 4000 instances and 156 features. The target is the winner of for every instance or match. There is also a feature for the number of rounds up to which the match lasts, however that has been withdrawn so the learners do not have any pre-conceptions as to the length of the fight, which could sway the learning and the predictions.

Since in the UFC, the red fighter is the favorite, the data is skewed towards red winning. Hence, there is a larger weight if the learner is able to predict a blue win correctly.

Unlike the wine dataset, the features of the ufc dataset are both continuous and binary. The target class is also binary, a red victory or a blue victory. Like the wine dataset, the weighted average of the precision and the recall,$f1\_score$, was used as the metric while trying to optimize hyperparameters. However, unlike the wine dataset, since the ufc dataset does not have equal instances for each class, the average of the $f1\_score$ had to be unweighted.

Like with the wine dataset, the data was normalized. The *MinMaxScaler* function was used to normalize the data for the UFC dataset as well. The $StratifiedShuffleSplit$ was used for spliting the dataset into the training set and testing set.

III. Clustering Algorithms

First there were two clustering algorithm, K-Means and Expectation Maximization, which were used to cluster the instances of the wine and ufc datasets.

i. UFC Dataset

The first dataset that was used was the UFC dataset. As seen when using the supervised learning algorithms, it is hard to achieve good accuracy on this dataset as there are 156 features and not a lot of instances, hence the curse of dimensionality plays a part.

i.1 K-Means

There are two different metrics which were used to determine how many clusters should be used. The first is the silhouette score which is a measure of how compact a cluster its proximity to it's closest cluster. The second is the homogeneity which is a measure of how many of the instances in a cluster to the same class.

As can be seen in the below figure, the homogeneity score is slowly increasing as the number of cluster increases. This makes sense as when there are a higher number of clusters, it is more likely that a higher percentage of each cluster belongs to the same class. The silhouette score has multiple peaks hence there are certain number of clusters which work well. The best combination of the score is $k = 23$. When using $k = 2$ for clustering, the accuracy score was 0.43 which is slightly better than the supervised learning algorithms.

i.2 Expectation Maximization

To run the Expectation Maximization algorithm, the GaussianMixture function was used. Since the initial center of the cluster are random, the silhouette and homogeneity score curves will be slightly different. As for K-Means, the homogeneity score rises steadily as expected and the silhouette score has several spikes. The best number of clusters was $k = 22$. When using $k = 2$ for clustering, the accuracy score was 0.40 which is slightly worse than the supervised learning algorithms.

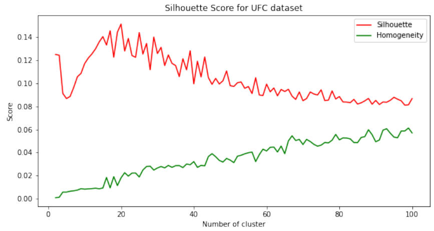

Figure 1: *Silhouette and Homogeneity Score for UFC dataset for K-Means*

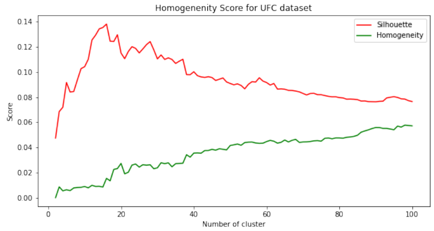

Figure 2: *Silhouette and Homogeneity Score for UFC dataset for Expectation Maximization*

ii. Wine Dataset

The second dataset is the wine dataset which is comparatively simpler as was seen when using the supervised learning algorithms which were consistently able to achieve an accuracy higher than 90%.

ii.1 K-Means

For the K-Means algorithm, first the sum of squared error metric was used to find the optimal number of clusters. As can be seen from the below graph, the elbow bend is at $k = 3$.

To confirm, the silhouette score and homogeneity score metrics were also used. This outputted the below graph. It can be seen that the homogeneity score increased rapidly from for low values of k and then gradually increases

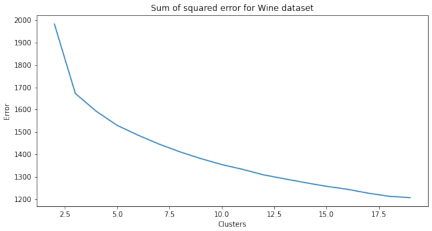

Figure 3: *Sum of squared error for Wine dataset*

whereas the silhouette score increases slightly early on and then starts decreasing. Hence, it is confirmed that the optimal number of clusters is $k = 3$.When using $k = 2$ for clustering, the accuracy score was 0.37 which is less than half of the supervised learning algorithms.

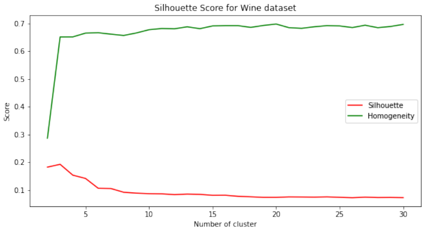

Figure 4: *Silhouette and Homogeneity Score for Wine dataset for K-Means*

ii.2 Expectation Maximization

Using the Expectation Maximization algorithm, the silhouette score and homogeneity score show a similar curve as the K-Means algorithm. Hence, both the algorithms confirm that $k = 3$ is the ideal number of clusters. However, the accuracy score using Expectation Maximization algorithm was only 0.30. which is extremely low for the wine dataset. This is due to the algorithm only being able to capture one of the classes accurately and not the other two. This is probably due to the fact that the cluster center start at random points and thus can skew the clusters that are formed.

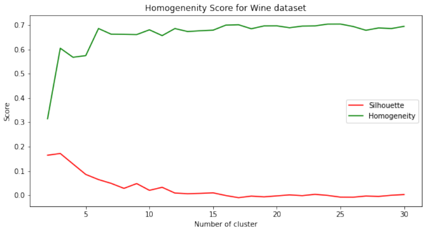

Figure 5: *Silhouette and Homogeneity Score for Wine dataset for Expectation Maximization*

IV. Dimensionality Reduction Algorithms

The two datasets have very different amounts of features. The UFC dataset has 156 features which makes it a likely candidate for dimensionality reduction. However, the wine dataset only has 13 features which doesn't allow for much dimensionality reduction.

i. PCA

i.1 UFC Dataset

Since there are so many features in the UFC dataset, feature reduction from 2 to 150 dimensions where tried using the PCA algorithm. The explained variance and the reconstruction error were then analyzed. From the fitting, using 30 dimensions the explained variance is 0.9 and using 72 dimensions the explained variance is 0.98. 72 was used as the number of dimensions and the following graph was produced for the first three components. It can be seen that the instances were not clustered effectively.

i.2 Wine Dataset

Since there were only 13 features in the wine dataset, feature reduction from 2 to 13 dimensions where tried using the PCA algorithm. The explained variance and the reconstruction error were then analyzed. From the fitting, using 8 dimensions the explained variance is 0.93. 8 was used as the number of dimensions and the following graph was produced for the

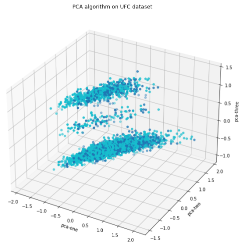

Figure 6: *Clustering after dimensionality reduction by PCA algorithm on UFC dataset*

first two components. It can be seen that the instances were clustered effectively.

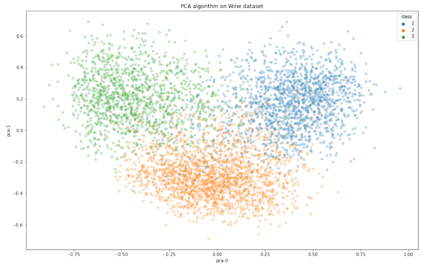

Figure 7: *Clustering after dimensionality reduction by PCA algorithm on Wine dataset*

ii. ICA

ii.1 UFC Dataset

Unlike for the PCA algorithm the dimensionality reduction for ICA was informed by the reconstruction error than the explained variance. Below is the reconstruction error for using dimensions 1 to 150.

The amount of dimension that was optimal was 80 as that reduces the number of dimensions by almost half while keeping a low error. It can be seen that the instances were not clustered effectively, however the clustering for the ICA algorithm is very different than the PCA algorithm. This is due to the fact that PCA maximizes variance when reducing dimensions whereas ICA maximizes independence.

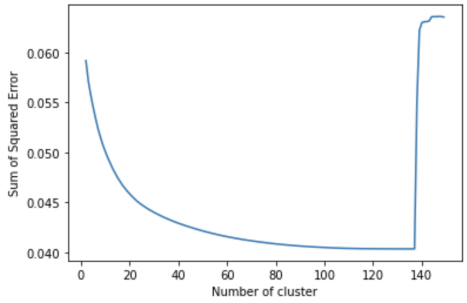

Figure 8: *Reconstruction Error for ICA algorithm on UFC dataset*

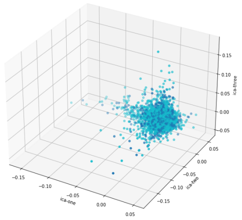

Figure 9: *Clustering after dimensionality reduction by ICA algorithm on UFC dataset*

ii.2 Wine Dataset

Like for the UFC dataset the dimensionality reduction was informed by the reconstruction error for the wine dataset for the ICA algorithm. Below is the reconstruction error for using dimensions 1 to 13. It seems as though the error decreases in a linear fashion. The amount of dimension that was chosen was 3 as it seems like the error decreases at a ever so slightly less rate for higher dimensionality.

The amount of dimension that was chosen was 3 as it seems like the error decreases at a ever so slightly less rate for higher dimensionality. It can be seen that the instances were clustered effectively even though the clustering is very different than that of PCA.

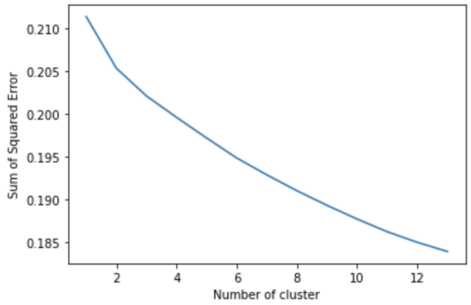

Figure 10: *Reconstruction Error for ICA algorithm on Wine dataset*

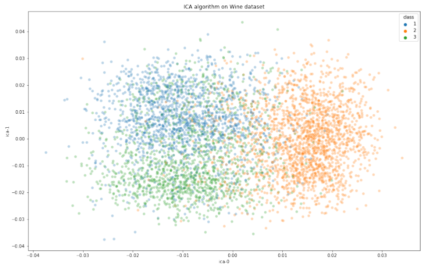

Figure 11: *Clustering after dimensionality reduction by ICA algorithm on Wine dataset*

iii. Randomized Projection

iii.1 UFC Dataset

Like for the ICA algorithm the dimensionality reduction was informed by the reconstruction error. Below is the reconstruction error for using dimensions 2 to 150. It seems as though the error decreases in a linear fashion.

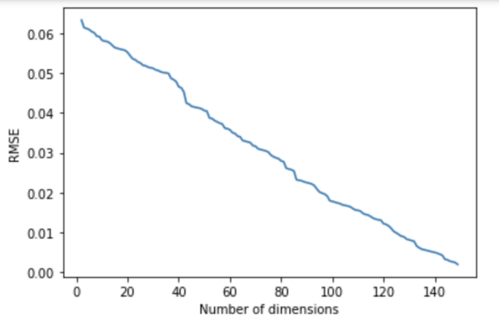

Figure 12: *Reconstruction Error for RP algorithm on UFC dataset*

The amount of dimension that was chosen was 40 as that is about a quarter of the dimension and the decrease in error is linear. It can be seen that the instances were not clustered effectively. It looks different than both PCA and ICA where all the datapoints are still clustered in the middle of the three dimensions.

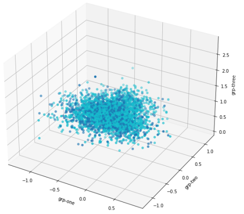

Figure 13: *Clustering after dimensionality reduction by RP algorithm on UFC dataset*

iii.2 Wine Dataset

The reconstruction error for using dimensions 1 to 13 was graphed in the below figure. It can be seen that the error decreases consistently until about 9 dimensions where it stabilizes until all 13 features are used. Hence, the amount of dimension that was chosen was 9.

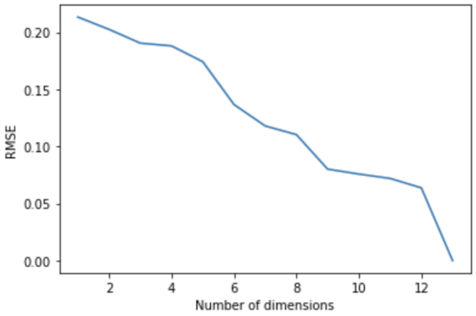

Figure 14: *Reconstruction Error for RP algorithm on UFC dataset*

It can be seen that the clustering using the randomized projections is not as clean as the PCA and ICA algorithms.

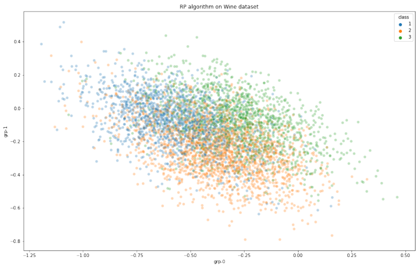

Figure 15: *Clustering after dimensionality reduction by RP algorithm on Wine dataset*

iv. SelectKBest

iv.1 UFC Dataset

From the reconstruction error curve it was seen that a dimension of 40 was the optimal in much of the same manner as for the randomized projections algorithm.

iv.2 Wine Dataset

From the reconstruction error curve it was seen that a dimension of 5 was the optimal in much of the same manner as for the randomized projections algorithm.

V. Clustering after Dimensionality Reduction

Since there are two datasets, two scoring metrics and four different dimensionality reduction algorithms, there are 16 different clustering algorithms which had to be run. Hence, instead of providing the graphs for each of the clustering algorithms, only the ones which have noteworthy information.

When using the PCA dimensionality reduction algorithm and the K-Means clustering algorithm on the UFC dataset resulted in the silhouette score to peak when using 16 clusters. This is due to the fact that PCA reduced the dimensions to 72 and it was easier to learn on.

However, the highest silhouette score was reached when clustering on the UFC dataset reduced by the SelectKBest algorithm. It got a silhouette score of 0.21. This is due to case that the SelectKBest algorithm reduced the dimension the most of any of the dimensionality reduction algorithms and hence there was more data compared to the number of dimensions and the learner was able to learn more effectively.

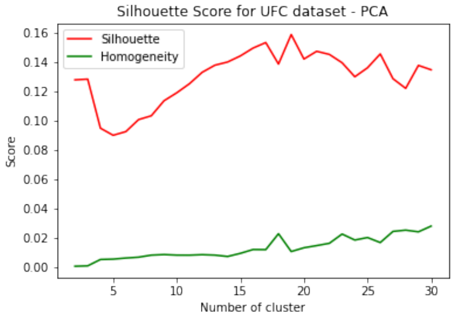

Figure 16: *Silhouette and Homogeneity Score for UFC dataset for K-Means after PCA reduction*

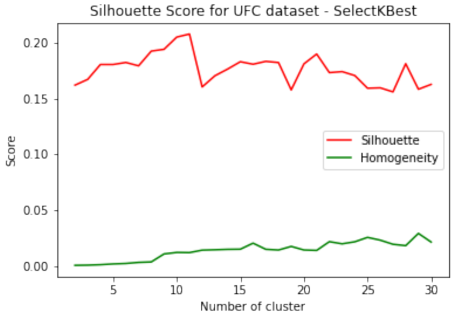

Figure 17: *Silhouette and Homogeneity Score for UFC dataset for K-Means after SelectKBest reduction*

VI. Neural Network Learner on Wine Dataset after Dimensionality Reduction

The wine dataset was chosen over the UFC dataset since the neural network learner performed much better on the aforementioned dataset.

i. PCA

For the PCA algorithm, a dimensionality of 8 was used which had a explained variance ratio of 0.93. As a neural network learner with 2 hidden layers with 60 nodes in each layer was shown to be the best hyperparameters in supervised learning, they were used on the dimension reduced dataset. This produced the following training and test learning curve with an accuracy score of 0.93 on the training set.

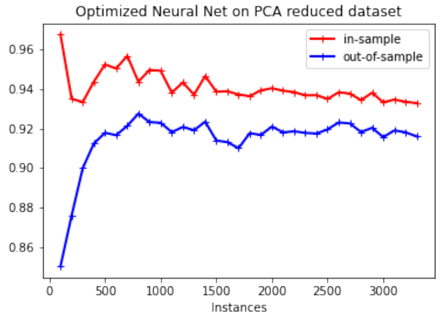

Figure 18: *Initial Neural Net on PCA reduced dataset*

However, there was some hyperparameter tuning for the hidden layer size, both for 1 hidden layer and 2 hidden layers. It seemed as though having 2 hidden layers with 10 nodes in each layer performed the best on the test set. The learning algorithm was able to get an accuracy score of 0.925 on the test set with 8 dimensions on the test set which is pretty close to what was able to be achieved by the neural network learner when learning on all dimensions.

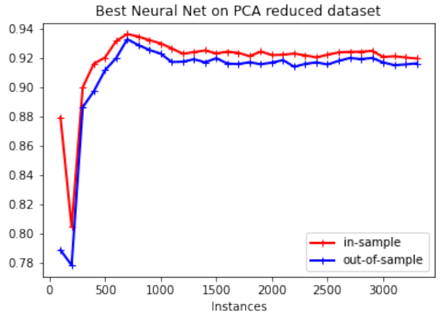

Figure 19: *Best Neural Net on PCA reduced dataset*

ii. ICA

For the ICA algorithm, a dimensionality of 3 was used which had a explained variance ratio of 0.93. As a neural network learner with 2 hidden layers with 60 nodes in each layer was shown to be the best hyperparameters in supervised learning, they were used on the dimension reduced dataset. This produced the following training and test learning curve with an accuracy score of 0.91 on the training set.

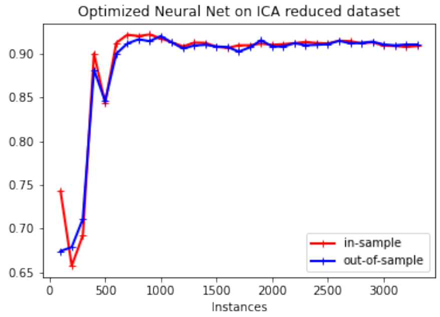

Figure 20: *Initial Neural Net on ICA reduced dataset*

However, there was some hyperparameter tuning for the hidden layer size, both for 1 hidden layer and 2 hidden layers. It seemed as though having 2 hidden layers with 80 nodes in each layer performed the best on the test set. The learning algorithm was able to get an accuracy score of 0.9167 on the test set with 3 dimensions which is pretty close that was able to be achieved by the neural network learner after the PCA algorithm reduced the dataset to 8 dimensions. This means that even with underfitting the data with just 3 dimensions, the ICA algorithm kept a lot of the relevant information.

iii. Randomized Projections

For randomized projections, the Gaussian Random Projection function was used since the data in the dataset is already processed and is clean data. A dimensionality of 9 was used since the randomized projection is not smart about dimensionality reduction. As a neural network learner with 2 hidden layers with 60

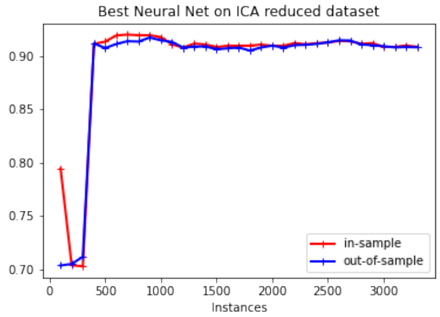

Figure 21: *Best Neural Net on ICA reduced dataset*

nodes in each layer was shown to be the best hyperparameters in supervised learning, they were used on the dimension reduced dataset. This produced the following training and test learning curve with an accuracy score of 0.88 on the training set. This turned out to be the best value for the hidden layer size hyperparameter after hyperparameter tuning with an accuracy score of 0.8708 on the test set. The performance of the learning algorithm was relatively notably worse. This is due to the fact the randomized projections algorithm was not able to capture the necessary information when reducing the dimensions. We saw in the reconstruction error where even reducing the dimension by one had a large jump in the error.

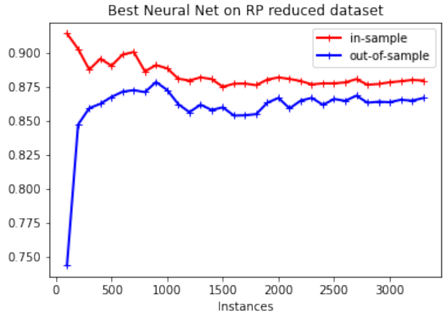

Figure 22: *Best Neural Net on RP reduced dataset*

iv. Select K Best

For the Select K Best algorithm, the Lasso function was used. A dimensionality of 5 was used which is about a third of the original number of features. As a neural network learner with 2 hidden layers with 60 nodes in each layer was shown to be the best hyperparameters in supervised learning, they were used on the dimension reduced dataset. This produced the following training and test learning curve with an accuracy score of 0.91 on the training set.

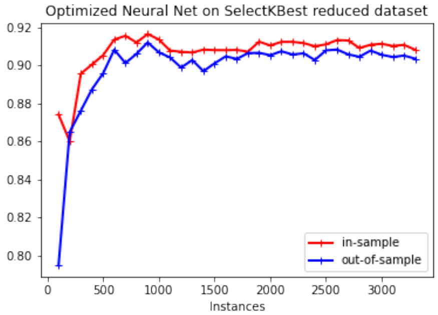

Figure 23: *Initial Neural Net on SelectKBest reduced dataset*

However, after hyperparameter tuning on the hidden layer size, it was observed that using 2 hidden layers with 100 nodes in each layer performed better with an accuracy score of 0.8942 on the test set. The Select K Best algorithm performed worse than the ICA algorithm which used fewer features and worse than the PCA algorithm which used more features.

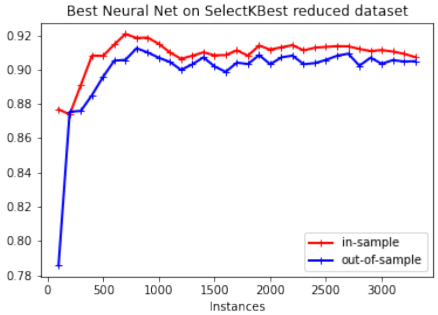

Figure 24: *Best Neural Net on SelectKBest reduced dataset*

VII. Neural Network Learner on Wine Dataset after Clustering

The wine dataset was chosen over the UFC dataset since the neural network learner performed much better on the aforementioned dataset. For fitting using the neural network learner, the clusters produced by the K-Means and Expectation Maximization clustering algorithms were used instead of using the clusters in addition to the features.

i. K-Means

Instead of using 3 clusters as was shown to be optimal when performing K-Means clustering, 15 clusters were used since this provides a higher homogeneity score. It also leads to a better labelling the data when there are a higher number of clusters. A neural network with default values performed extremely poorly and was only able to get a *f1_score* of 0.58 on the training set.

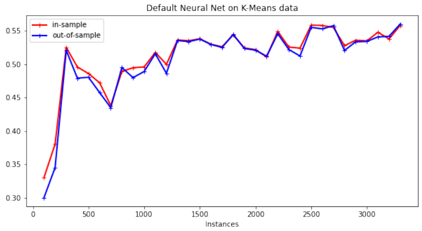

Figure 25: *Default Neural Net after K-Means clustering*

Hence, it needed some hyperparameter tuning. Neural networks with 2 hidden layers were tested with a range of nodes for each layer from 60 to 100. It seemed as though an higher amount of nodes in the hidden layers performed better with the neural network with 2 hidden layers and 100 nodes in each layer getting an *f1_score* of 0.89 on the test set. While this is significantly better than the default neural network, it still performed worse than just using the neural network on the labelled data itself. This signifies that using multiple learners together does not always yield better results.

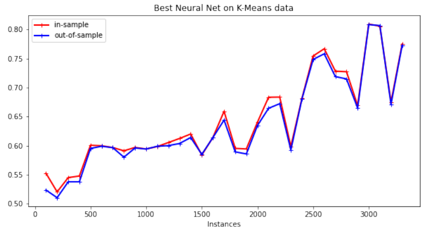

Figure 26: *Best Neural Net after K-Means clustering*

ii. Expectation Maximization

Like for K-Means, 15 clusters were used in the Expectation Maximization algorithm since this provides a higher homogeneity score. It also leads to a better labelling the data when there are a higher number of clusters. A neural network with default values performed extremely poorly and was only able to get a *f1_score* of 0.65 on the training set. This is significantly better than the default neural network learner using K-Means clustering.

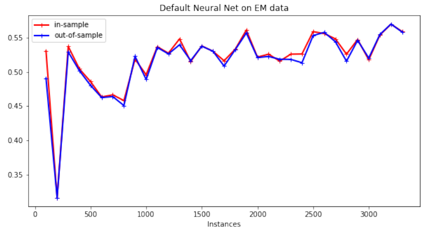

Figure 27: *Default Neural Net after Expectation Maximization clustering*

However, it still needed some hyperparameter tuning. Neural networks with 2 hidden layers were tested with a range of nodes for each layer from 60 to 100. It seemed as though an higher amount of nodes in the hidden layers performed better with the neural network with 2 hidden layers and 100 nodes in each layer getting an *f1_score* of 0.8306 on the test set. While this is significantly better than the default neural network, it still performed worse than the best neural network with the K-Means clustering and worse than just using the neural network on the labelled data itself. This signifies that using multiple learners together does not always yield better results.

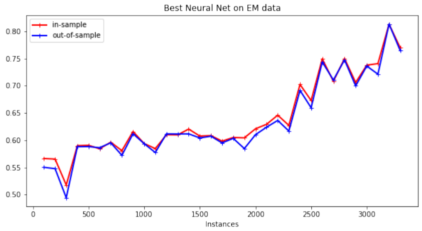

Figure 28: *Best Neural Net after Expectation Maximization clustering*

VIII. Run Time

Running the neural network learner on datasets reduced by the clustering and dimensionality reduction algorithms were quite a lot faster than just running the supervised learning learner on the original dataset itself. This is because there are significantly fewer computations for forward pass backpropagation. The K-Means and Expectation Maximization datasets took about twice as long as PCA, ICA, RP and SelectKBest. This is due to the fact that the clustering labels are much more complex than the features themselves, hence the weights need to be adjusted more. Hence it is a trade off between time and accuracy. Below are the score and times from the neural networking learning on the standard wine dataset as well as learning on the datasets reduced by the clustering and dimensionality reduction algorithm.

Algorithm	f1_score	Time
NN alone	0.9405	1540
K-Means	0.89	287
EM	0.8306	295
PCA	0.925	89
ICA	0.9167	135
RP	0.8708	147
SelectKBest	0.8942	180

IX. Conclusion

The fact that the wine and UFC datasets were so different in it's composition made them ideal candidates for different tests. Since the UFC dataset had 156 features, it was an ideal candidate for the dimensionality reduction algorithms. Using this dataset, it was found that the PCA and ICA algorithms were better at retaining information and having accurate predictions than the randomized projections and the Select K Best algorithms.

The wine dataset was ideal for testing the performance of the clustering algorithms as they were able to get good accuracy. The wine dataset was also ideal to test using two different learners, one clustering algorithm and a neural network on the labels produced by the labels of the clustering algorithm. This showed that using two learning algorithms does not lead to a better accuracy. The wine dataset was also used to test the neural network on the dimensionality reduced datasets. It was seen that especially the PCA and ICA algorithm performed really well as they were able to get a $f1\_score$ of 0.925 and 0.9167 with only using 8 and 3 dimensions respectively. Since the ICA algorithm uses fewer dimensions, it is even more impressive, however, it indicated that there isn't a lot of difference in the performance between local and global search. Lastly, it was seen that after hyperparameter tuning, the neural network had very similar accuracy on both in and out of sample data after clustering and dimensionality reduction.