Rob Lindsay **Assignment 3** **Machine Learning 6341**

EXECUTIVE SUMMARY

Data

This report serves to show how different clustering algorithms can produce very different results, how different feature selection techniques can produce very different results, and how both feature selection and clustering can be incorporated in classification processes as another candidate for classification problems

Data

The data that was used was found in the SeoulBikeData.csv. The data contains hourly information on environment variables as well as seasonal and holiday data. The number of bikes that were rented over the course of each hour is what we will derive the classification target off of.

With the exception of the initial challenges faced with special characters in the column titles and the column titles not aligning with the data, both of which were addressed in Assignment 1, the data was exceptionally clean. Using similar data manipulation techniques from the first assignment, target encoding was used over one hot encoding for the ANN model.

For the classification target, I decided to go with the median value of the rented bikes for my train set. Having a balanced classification would allow me to choose Accuracy as my indication for how well each classification model was performing. Using the median as a target value reduced my concern about making sure that there was a representative distribution in my train and test sets as well, although this can easily be accomplished throughout the code.

Classification Model - ANN

As in the previous assignment, I utilized GridSearchCV in conjunction with the Keras library to build out my ANNs but with reduced hyperparameters options to speed up the overall run time. Had the goal been to maximize performance without overfitting, I would have expanded the hyperparameters that I used in the model. However, the primary purpose of this assignment is interpreted as understanding how clustering and feature reduction techniques can be used in the classification process. As such, I choose to limit the spread of my hyperparameters.

I utilized relu for all hidden layer activation functions and sigmoid for the final activation function. For the compiler, I choose to use the following settings: (optimizer='adam', loss='binary_crossentropy', metrics=['accuracy']). EarlyStopping callback functionality was employed as before.

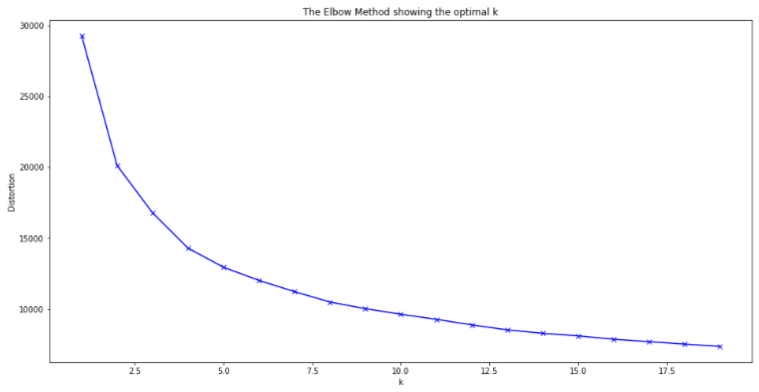
Train & Test Sets

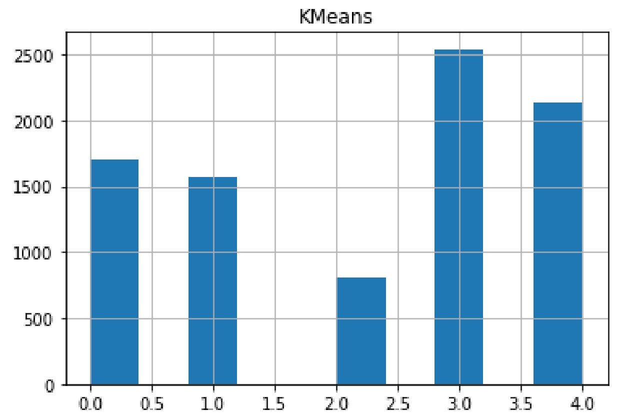
I split my data into a 60/40 with classification values coming from the 'MedianOrHigher' variable that I created. For the ANN models utilizing feature selections, each model pulled training and testing values from their respective feature selection algorithm outputs. For instance, the ANN model utilizing Forward Selection techniques had a different set of data than the ANN model utilizing PCA. For the final task, the two original clusters were used for the data along with their associated 'MedianOrHigher' values. I expanded on this task and incorporated all cluster outcomes for comparative purposes, the results of which will be discussed.

TASK 1: Clustering with K-Means and Expectation Maximization

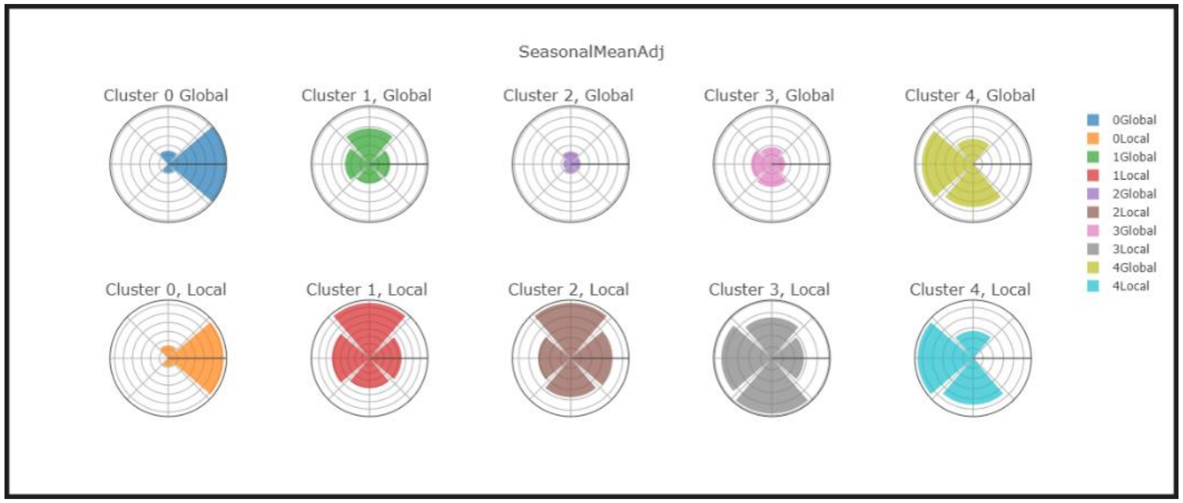
K-Means

For K-means clustering, I attempted to employ the elbow method by plotting Distortion. Unfortunately, there was not a significant moment at which the elbow indicated an ideal clustering. After exploring the histograms of several cluster values, I arbitrarily choose 5 as my number of clusters, mainly because I liked the distribution of clusters and felt like it would be a good starting point for experimentation.



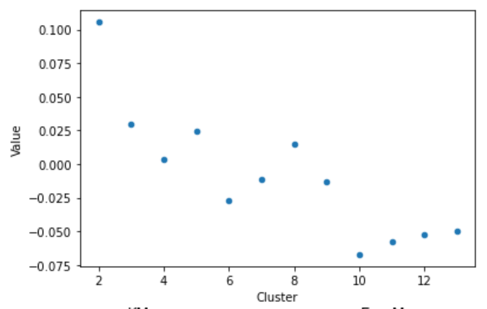


Visualizing the clusters was challenging as I realized that the categorical variables that employed the target encoding presented a challenge. However, I came up with a technique utilizing the barpolar feature of Plotly that allowed me to see how the categories were spread across all the clusters as well as on the individual clusters (this helped as some clusters had very few representatives of a particular category). As an example, here is the categorical variable SeasonalMeanAdj. Here we can see that there are clearly 4 seasons as expected. The upper row shows the distribution across each cluster with consistent sizing and provides insight at the macro level. For instance, from "Cluster 0, Global", we can see that it is primarily formed from one season(spring) and has no contribution at all from one season(fall). As you scan the top row, you see that there is very little representation of spring in any of the other clusters. Cluster 2, which we had already seen in our histogram as having the lowest number of members, is difficult to see in the top row. Here, the bottom row helps to paint a zoomed in picture of the spread of seasons that are represented. Other categorical takeaways that I had (not pictured) within the K-means clustering were that clusters 1 and 4 were heavily influenced by HourRank, Clusters 2 and 3 consisted exclusively of the same two days.

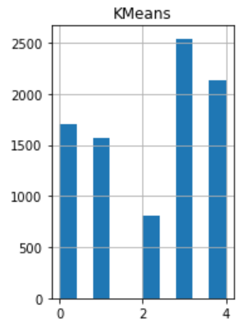


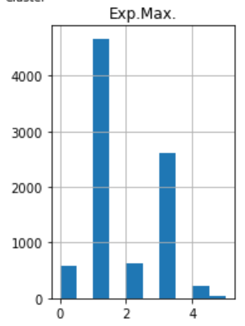
Expectation Maximization

For clustering utilizing Expectation Maximization, I attempted to employ the silhouette value to determine the number of clusters. None of the silhouette values were particularly strong with maybe the exception of a cluster of 2. But a cluster of 2 would not have been very productive for this experiment. So, I decided to limit my scope to a value between 4 and 9 clusters. I settled on a value of 6 as this was the first clustering that wasn't significantly unbalanced. But overall, the distribution to the different cluster designations was not nearly as balanced as what we saw in K-means.

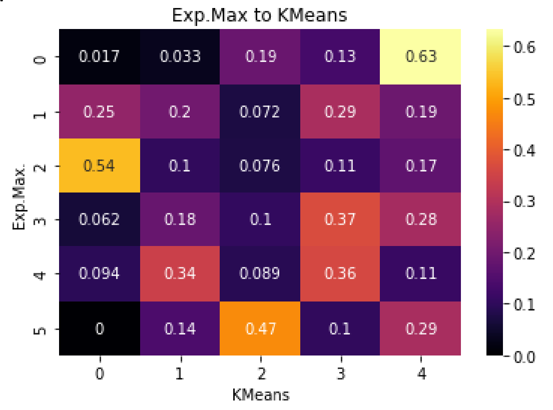


The same visualization techniques that were employed with K-means can be employed here, but I found the more interesting approach to be in comparing the two clustering techniques to each other. Through the use of a heat map, you can see how each cluster from one clustering algorithm is distributed throughout the clusters in another clustering algorithm. To better understand how the mapping works, the numbers represent the percentage of the cluster formed by the algorithm listed on the Y-axis mapped to the clusters formed by the algorithm listed on the X-axis. So all row values will add up to 1 while column addition would merely serve as another representation of the histogram distribution and wouldn't serve much purpose. Two heatmaps can be employed to see if significant relationships are bidirectional, but with unbalanced distribution within the Expectation Maximization clusterings, I was not expecting to find a significant bidirectional relationship.

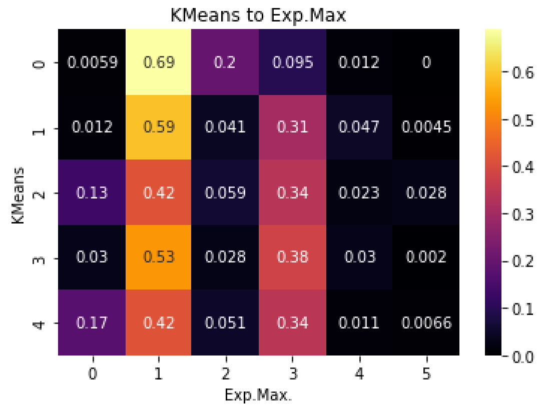




Starting with the Exp.Max. mapping to the K-Means clusters, you can see that there are very strong relationships between 2 to 0, 5 to 2, and 0 to 3. In other words, more than 50% of the observations in cluster 2 of the Expectation Maximization algorithm were also found in cluster 0 of the K-Means algorithm. However, clusters 2, 5, and 0 were three out of the four lowest observation numbers for the Expectation Maximation clusters.



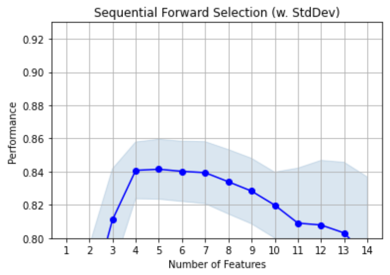
When looking at the K-Means to Exp.Max mapping, we see right away that there is a significant relationship between every K-means cluster and cluster 1 in the Expectation Maximization algorithm. This is not surprising given that cluster 1 contained nearly half of the overall observations for the Expectation Maximization algorithm. In fact, we see a similar relationship formed with cluster 3 for the Expectation Maximization algorithm, the other major cluster within this algorithm.



Unsurprisingly, there were not any clusters that jumped out as being almost identical. The closest bidirectional relationship is found in the 3 to 3 mappings with both clusters sharing a little more than 1/3rd of their observations. Interestingly, these both contained about the same number of total observations.

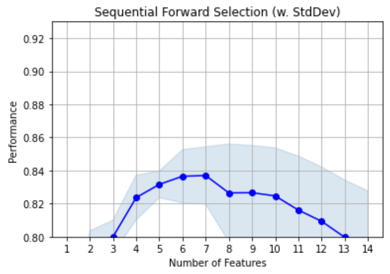
TASK 2: Feature Reduction through Forward Selection, PCA, ICA, and Randomized Projections

Forward Selection



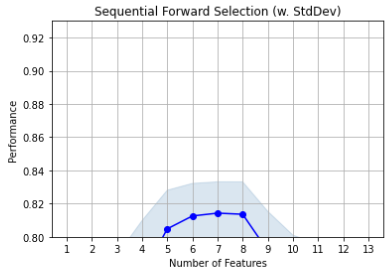
To establish a Forward Selection Model, I utilized a SequentialFeatureSelector from mlxtend.feature_selection and employed KNN to generate a winning feature selection of HourRank, Temperature, Rainfall, Holiday, and Functioning Day. However, combinations of 4, 6 and 7 features also produced very good results for KNN

Principle Component Analysis



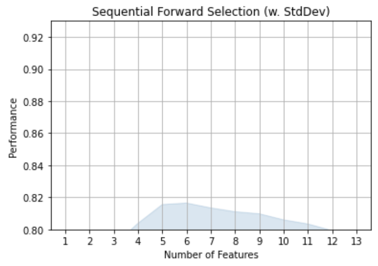
To establish a Principal Component Model, I utilized a PCA from sklearn.decomposition. To reduce the features further, I choose to piggyback on the Forward Selection method and once again employed KNN to generate a winning feature selection of components 3,4,6,7,8, 11, and 12. I found it interesting that several of the components (0,1,2) that form the strongest parts of the PCA were not instrumental in the KNN model implementation.

Independent Component Analysis



Continuing with the piggybacking approach, I utilized a FastICA from sklearn.decomposition and reduced the features to components (0, 1, 3, 8, 9, 10, 11)

Randomized Projection



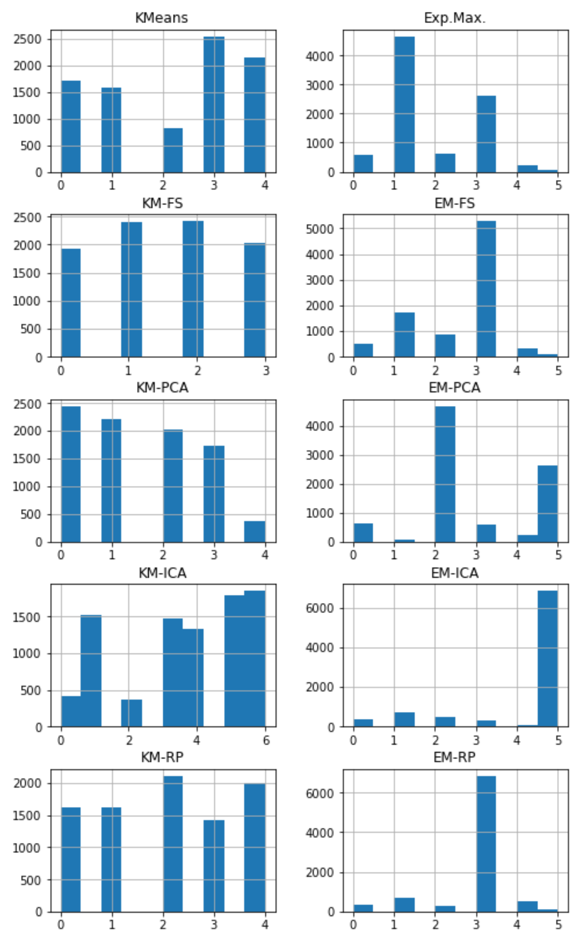
Finally, for the last feature reduction approach, I utilized GaussianRandomProjection from sklearn.decomposition. Keeping the same scope for KNN performance, you can see that Randomized Projection appears to have the poorest performance as you can only see the standard deviation showing up on our window. Features (0,5,9,10,11) were the winning features.

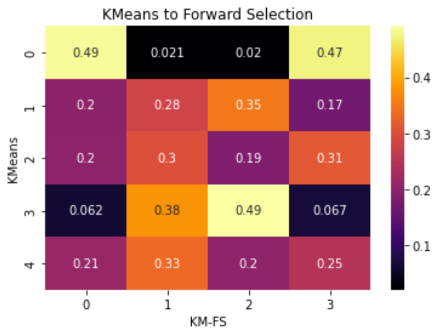
TASK 3: Rerun Clustering Algorithms with 4 Feature Reduction Methods

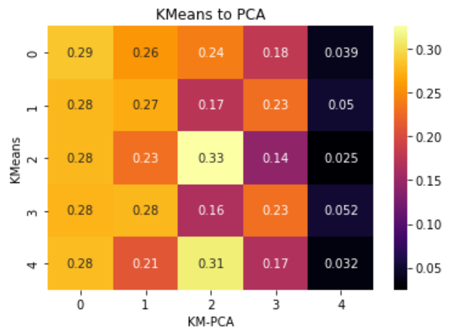
After rerunning the K-Means and Expectation Maximization Algorithms with the 4 Feature Reduction datasets, the first thing that jumps out is how we see a similar pattern at the macro level for balance of distribution of observations. K-means is a far more balanced distribution at each level with most clusters being of similar size. Expectation Maximization on the other hand continues to have one major cluster and sometimes a second mid major cluster. So we see a trend that Expectation Maximization will create several tightly packed clusters and one or two broader clusters while K-means tends to have a more evenly distributed clustering with less compact clusters.

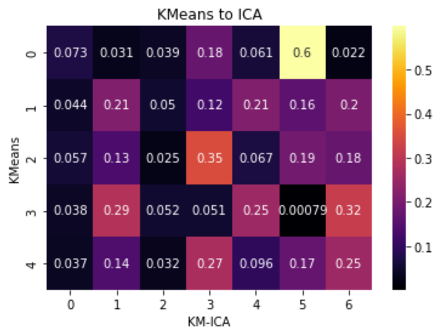
With such unbalanced results, there isn't much use in revisiting the heatmaps for the various Expectation Maximization results. However, we might be able to see similarities by comparing our original K-Means clusters to the Feature Reduction versions of the K-Means clusters.

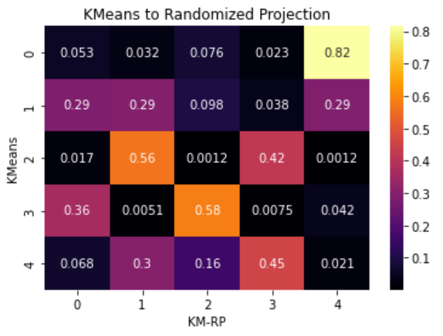
There are a couple of things that jump out to me. When mapping to Forward Selection, the 0 cluster splits almost perfectly down the middle into clusters 0 and 3 with one more near 50% split going from cluster 3 to cluster 2. The PCA model appears to have a pretty even redistribution throughout, with the last cluster having relatively small values, but these are expected with such a small overall representation of observations for cluster 4. For the ICA mapping, there are not very many strong mappings other than 60% of our original 0 cluster going to cluster 5. The Randomized Projection model appears to have several strong mapping partners. 82% of the original 0 cluster is found in the RP 4 cluster. Cluster 1 has been essentially redistributed equally to clusters 0, 1, and 4. Meanwhile, original clusters 2, 3, and 4 have anywhere from 75% to 98% of their observations redistributed to just two clusters in the RP clustering.











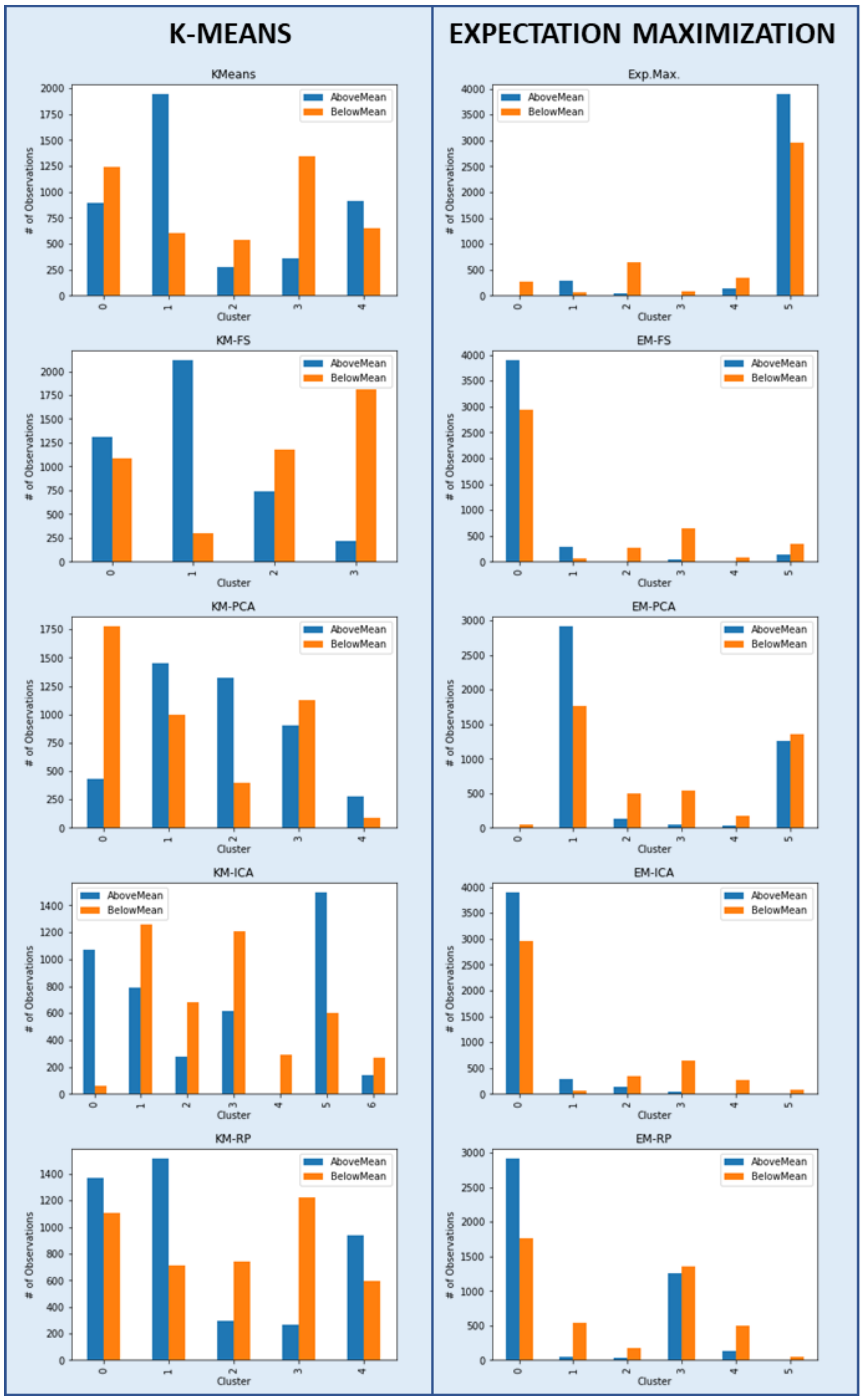
One thing that hasn't been touched on that is significant to this assignment's experimentation is how the final classification lines up with the clusters. Now that we have developed multiple clusters, it is appropriate to visit this topic and explore the relationships between algorithms, their clusters, and the binary classification problem that we have been focused on throughout the semester.

Previously, we saw the relationship between seasons and the created clusters. The polar bar plot worked well for multiclassification variables. With binary variables though, a more traditional bar plot is appropriate and requires a more intuitive manipulation and user understanding of the data and its visualization.

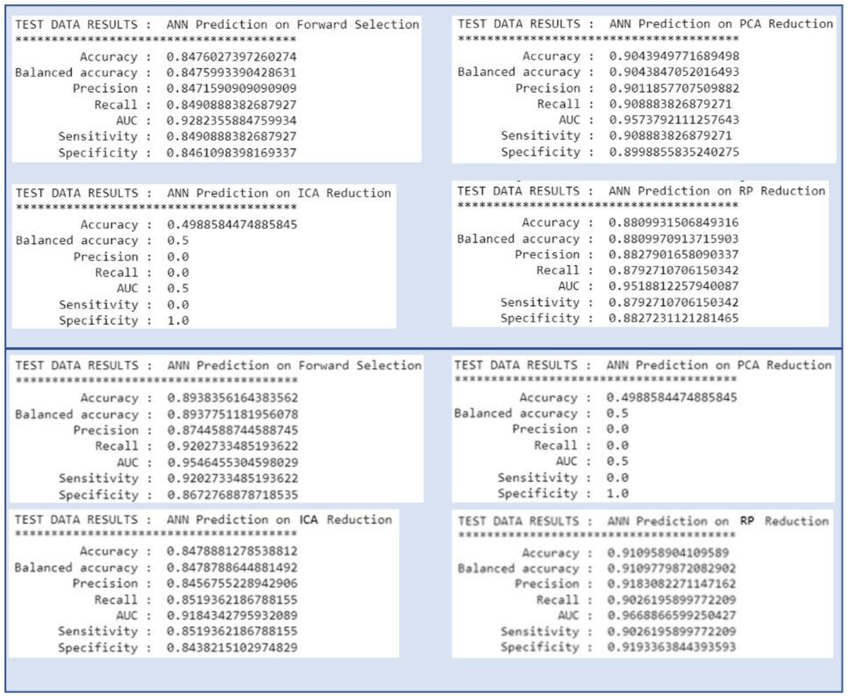
Looking at the bar charts to the right for each clustering algorithm, the blue bars represent observations with rentals over the mean value and orange bars represent observations below the mean value. You can see that there are several clusters that could be strong contributors to correct predictions for a classification algorithm.

Exploring the K-means algorithms, we see several clusters in which have strong representation of either above or below mean. Cluster 4 in the ICA is 100% below mean while Cluster 0 is almost entirely above mean. Cluster 1 for both K-Means and KM-FS are heavily weighted with above mean values while Cluster 3 in both KM-FS and KM-RP are heavily weighted below the mean.

Shifting to Expectation Maximization, we see that the unbalanced nature can often be extremely useful in its potential for identifying binary classification. With our base clustering model, Cluster 5 adds very little value as the above and below are relatively evenly matched. However, even though there are fewer observation in the other clusters, all of them can give a strong indication as to if an observation is below or above the mean, with Clusters 0 and 3 being formed entirely below the mean. As we look at the other Expectation Maximization clustering models, we see a similar pattern where the major cluster(s) don't add much value, but the minor clusters typically paint a very significant picture, generally indicating below mean observations in high likelihood across the board.



TASK 4: Run Neural Network Classification on 4 Feature Reduction Methods



```
TEST DATA RESULTS :  ANN Prediction on Forward Selection
***************************************
         Accuracy :  0.8476027397260274
Balanced accuracy :  0.8475993390428631
        Precision :  0.8471590909090909
           Recall :  0.8490888382687927
              AUC :  0.9282355884759934
      Sensitivity :  0.8490888382687927
      Specificity :  0.8461098398169337
```

```
TEST DATA RESULTS :  ANN Prediction on PCA Reduction
***************************************
         Accuracy :  0.9043949771689498
Balanced accuracy :  0.9043847052016493
        Precision :  0.9011857707509882
           Recall :  0.908883826879271
              AUC :  0.9573792111257643
      Sensitivity :  0.908883826879271
      Specificity :  0.8998855835240275
```

```
TEST DATA RESULTS :  ANN Prediction on ICA Reduction
***************************************
         Accuracy :  0.4988584474885845
Balanced accuracy :  0.5
        Precision :  0.0
           Recall :  0.0
              AUC :  0.5
      Sensitivity :  0.0
      Specificity :  1.0
```

```
TEST DATA RESULTS :  ANN Prediction on RP Reduction
***************************************
         Accuracy :  0.8809931506849316
Balanced accuracy :  0.8809970913715903
        Precision :  0.8827901658090337
           Recall :  0.8792710706150342
              AUC :  0.9518812257940087
      Sensitivity :  0.8792710706150342
      Specificity :  0.8827231121281465
```

```
TEST DATA RESULTS :  ANN Prediction on Forward Selection
***************************************
         Accuracy :  0.8938356164383562
Balanced accuracy :  0.8937751181956078
        Precision :  0.8744588744588745
           Recall :  0.9202733485193622
              AUC :  0.9546455304598029
      Sensitivity :  0.9202733485193622
      Specificity :  0.8672768878718535
```

```
TEST DATA RESULTS :  ANN Prediction on PCA Reduction
***************************************
         Accuracy :  0.4988584474885845
Balanced accuracy :  0.5
        Precision :  0.0
           Recall :  0.0
              AUC :  0.5
      Sensitivity :  0.0
      Specificity :  1.0
```

```
TEST DATA RESULTS :  ANN Prediction on ICA Reduction
***************************************
         Accuracy :  0.8478881278538812
Balanced accuracy :  0.8478788644881492
        Precision :  0.8456755228942906
           Recall :  0.8519362186788155
              AUC :  0.9184342795932089
      Sensitivity :  0.8519362186788155
      Specificity :  0.8438215102974829
```

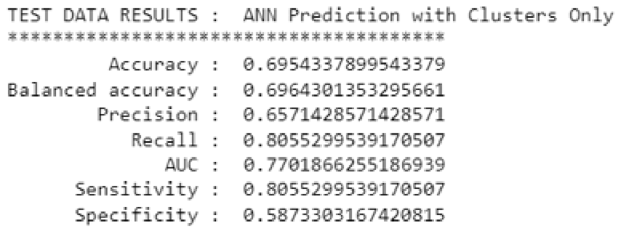
```
TEST DATA RESULTS :  ANN Prediction on RP Reduction
***************************************
         Accuracy :  0.910958904109589
Balanced accuracy :  0.9109779872082902
        Precision :  0.9183082271147162
           Recall :  0.9026195899772209
              AUC :  0.9668866599250427
      Sensitivity :  0.9026195899772209
      Specificity :  0.9193363844393593
```

Using the ANN approach developed in Assignment 3, I fed the reduced features to the model. However, in order to save time, I limited the scope of my hyperparameters. This led to a significant reduction in the overall processing time for each of the ANN model iterations. However, one thing that I touched on during the previous assignment became far more prevalent in this assignment. My implementation needs to be improved upon as at is suspectable to repeatability issues. To the left you can see two different runs through the feature reduction models. In the top set, the ICA results end up being no better than a random guess. In the bottom, we see the same thing but with the PCA results. This is a result of me rerunning the feature selections after finding the results. In essence, I found out what the best values were over my GridSearchCV and then reran the model to try to reproduce the results. However, the results weren't always the same and appear to have found local minimums or maximums that prevent consistent reproduction of results. This is something that certainly would need to be improved upon in a business setting but for educational purposes, it serves as a good lesson for the dangers of trying to reproduce results.

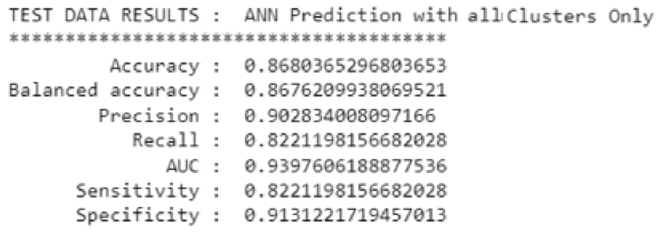
Looking past the errors in replication and taking a macro view of the classification results when compared to the results that were achieved in Assignment 3, you can see that there is potential that using feature reductions could be as effective as our full feature model from Assignment 3. For instance, 91% accuracy on our testing set for the RP model is very impressive.

TASK 5: Run Neural Network Classification With K-Means & Expectation Maximization Outputs

Unsurprisingly, changing the available features to only include the two cluster designations as training input did not yield great results, but it was still a significant improvement over a random guess with almost 70% accuracy on the testing data. With decent results from just two cluster models, it raised the question of what type of results we could get from using all of the clustering models that we developed between task 1 and task 3. The result was much improved over using just the original 2 cluster results. It would be interesting to see the results of just the K-Means clusters or just the Expectation Maximization clusters. As indicated in the discussion of Task 3, the Expectation Maximization minor clusters seem to add strong value but in low quantities and the K-means clusters seem to add a good proportion of generalized value. I think removing either cluster sets would end up lowering the overall accuracy results



```
TEST DATA RESULTS :  ANN Prediction with Clusters Only
***************************************
         Accuracy :  0.6954337899543379
Balanced accuracy :  0.6964301353295661
        Precision :  0.6571428571428571
           Recall :  0.8055299539170507
              AUC :  0.7701866255186939
      Sensitivity :  0.8055299539170507
      Specificity :  0.5873303167420815
```



```
TEST DATA RESULTS :  ANN Prediction with all Clusters Only
***************************************
         Accuracy :  0.8680365296803653
Balanced accuracy :  0.8676209938069521
        Precision :  0.902834008097166
           Recall :  0.8221198156682028
              AUC :  0.9397606188877536
      Sensitivity :  0.8221198156682028
      Specificity :  0.9131221719457013
```