Report for Challenge #1: P>N

**Introduction**

The original data set contains 300 variables, which have been classified into two groups, group 0 and group 1. However, there are only 250 data points within the training set, that must be used to interpret the classification of the remaining 19,750 points. Therefore, there are two key parts to this challenge: dimension reduction and cross validation. In this report, I will describe the method used to create a classifier that should be at least 99% accurate.

**Step 1: Initial data exploration shows differences in the Poisson distribution of the two groups**

In the challenge description it was stated that all of the 300 variables contained values between 0 and 1, and that the data was classified on a “secret” algorithm. However, there were several salient points that were still unknown. Firstly, how was the data distributed and would there be a way to see if there are any differences between the two groups in the training set. Finally, it is also worth checking to see if the information provided is true. By plotting the cumulative distribution of the full data set, as well as the two groups known in the training set we can answer these questions (**Figure 1**).

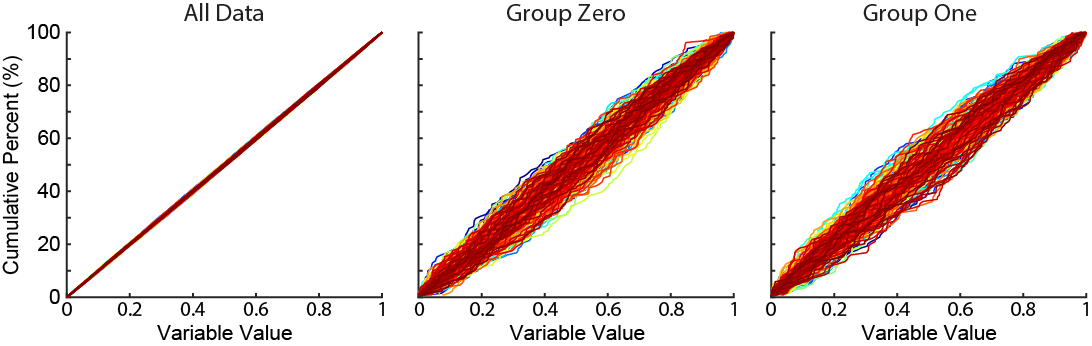


Figure - Cumulative distribution plots of all data (left), group 0 (middle) and group 1 (right). Each coloured line represents the cumulative distribution of a variable.

As we can see the data is distributed between 0 and 1 for all variables and that the distribution is not normally distributed, but more likely to be a single Poisson distribution. Thankfully, we can see that the distribution of values does appear to be different between the two groups (middle and right panels) indicating that a solution is possible.

**Step 2: Unbiased Dimension reduction fails**

The first step would be to reduce the number of dimensions analysed. If the two groups are differentially distributed, then could this be used to reduce the distributions? Initially, three main ways were assessed: UMAP, tSNE, and PCA. Both UMAP and tSNE, completely failed to reduce the dimensions which was most likely due to the fact that they have too much of a bias towards local variation. However, would PCA work? **Figure 2** shows the cumulative amount of variance explained by each component (dimension). As the shape looks linear, particularly for the first ~50 variables PCA does not seem to be a valid approach.

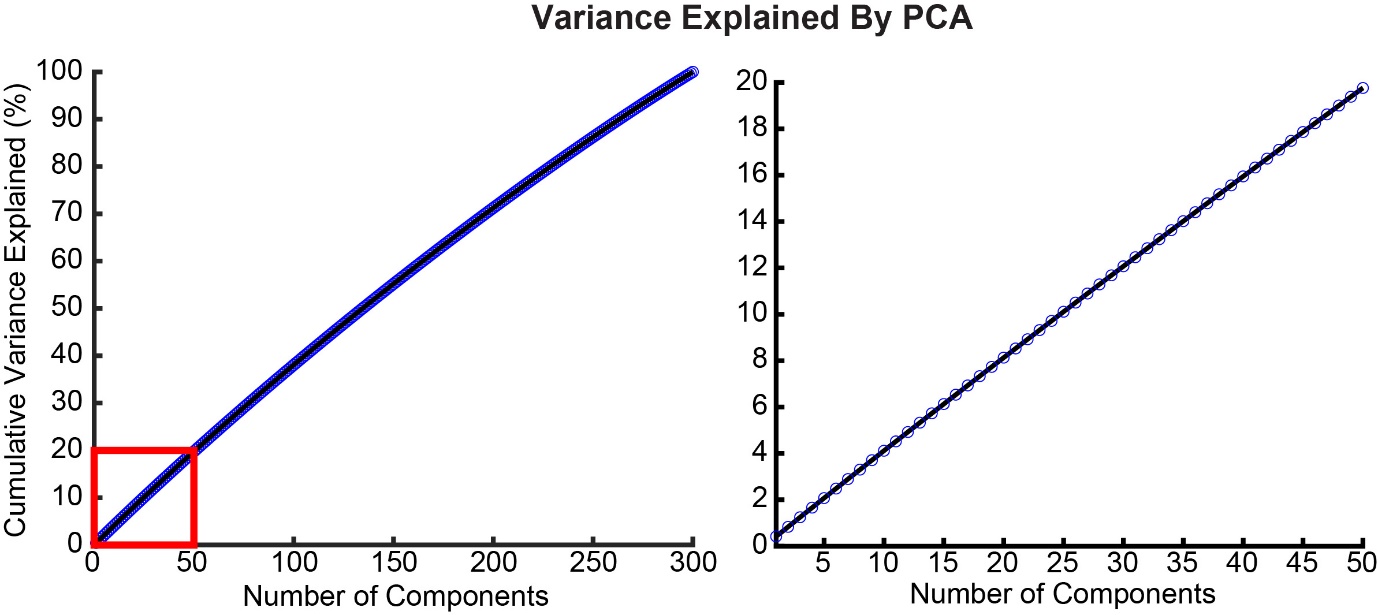


Figure - Principal Component Analysis is unable to reduce the number of variables, as each component represents a small amount of the variance. The left panel displays the variance explained by each component of the PCA. The right panel is a zoomed in section (red box, left panel).

**Step 3: Partial Least Squares (PLS) can effectively reduce the dimensionality**

As unguided/unbiased dimension reduction techniques failed, could regression models be a suitable option? Partial Least Squares (PLS) is a well-used technique to classify gene expression, in these cases often the number of dimensions (genes) are much higher than the number of cells evaluated. Furthermore, PLS can be combined with discriminant analysis (PLS-da) to classify groups after regression. To initially investigate this, we performed PLS on the whole training set (**Figure 3**). 98.7% of the whole variance can be explained by the first 10 latents (left panel) while the mean square error (MSE, right panel) shows a highly accurate regression after 5 or so latents are considered.

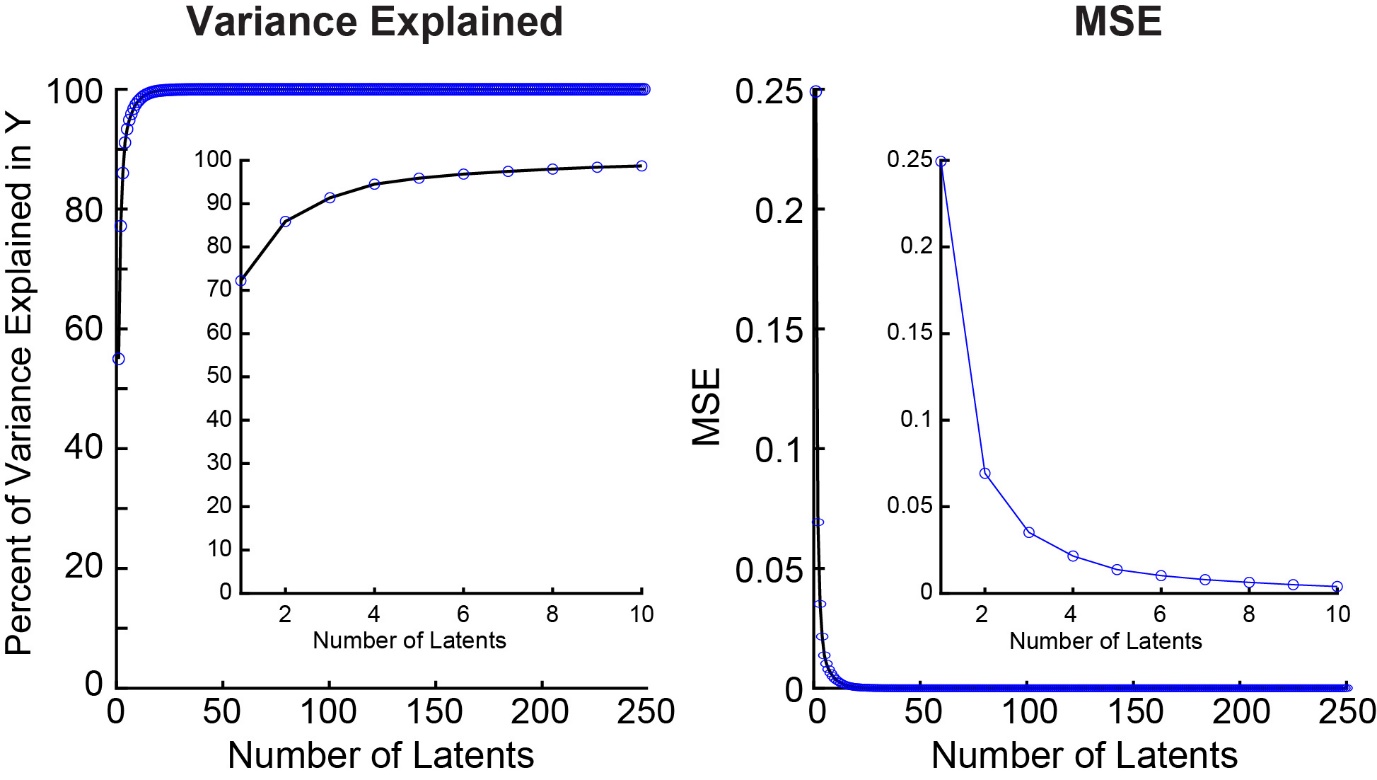
****

Figure - Initial PLS investigation suggests that it can effectively reduce the variables down to <10 latents. The left panel displays the cumulative variance explained by each latent. Inset displays the first 10 variables. Right panel displays the mean square error (MSE) of the Y variable after PLS regression has been performed. Inset shows the first 10 latents

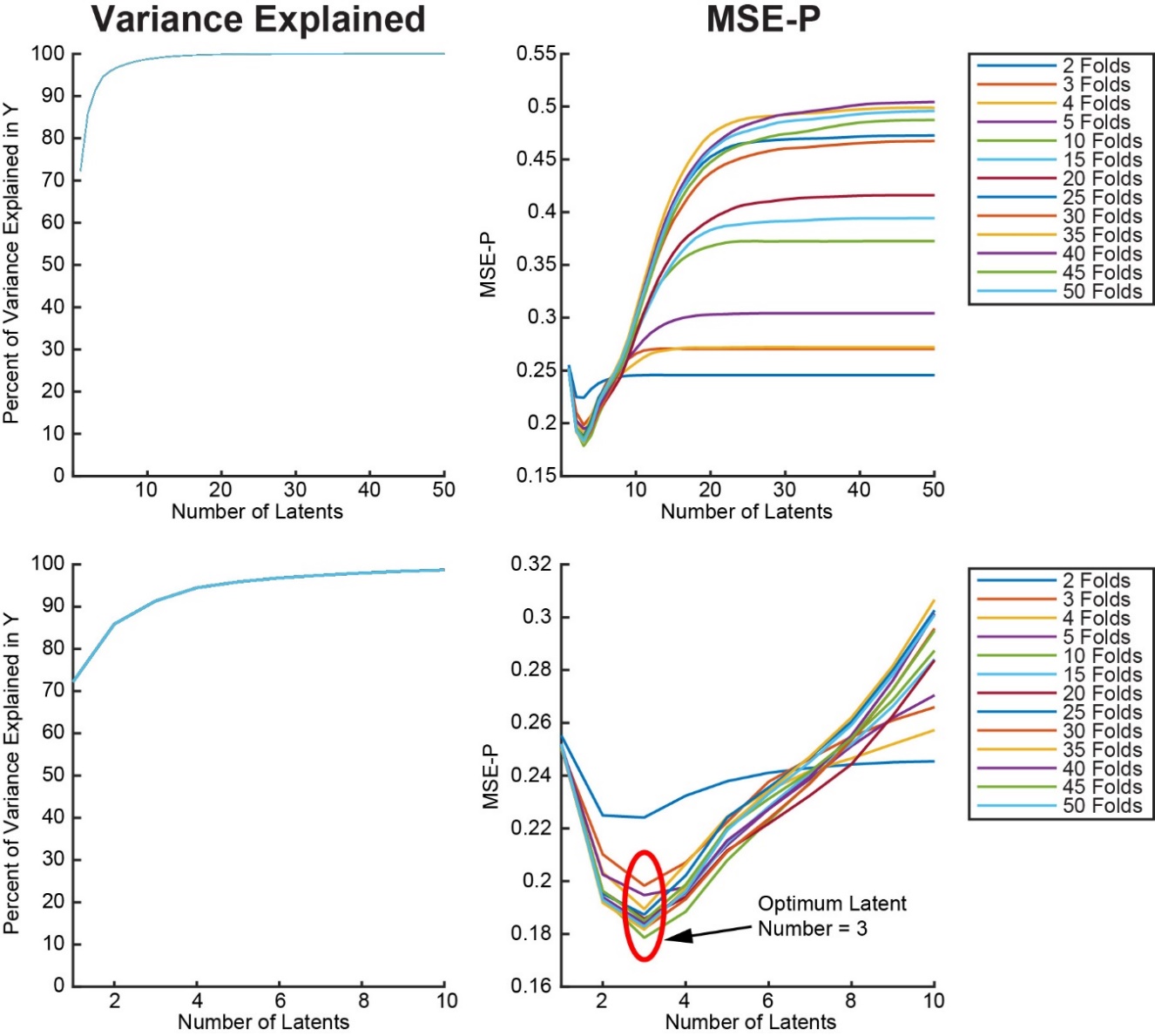
**Step 4: Determining the number of latents to use with cross-validation**

Figure - PLS with K-fold cross validation reveals 3 latents to be the most accurate regression. The left panels show the variance explained by the PLS of each fold for the first 50 (top) or first 10 (bottom) latents. The right panels show the mean square error of the prediction (MSE –P) for the test fold. The top right panel, shows the first 50 latents, the bottom right displays the first 10.

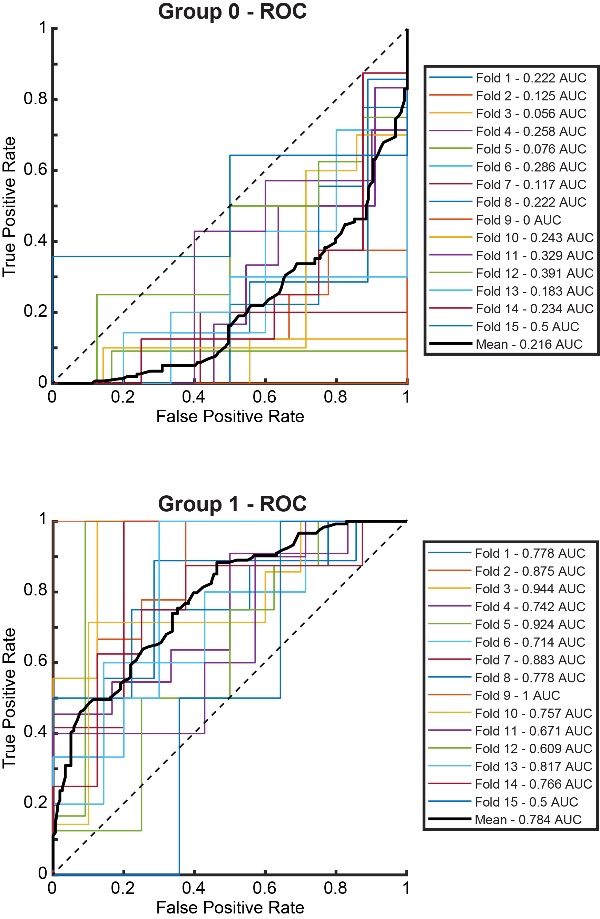
While one could theoretically select a number of latents from the MSE produced by the full training set, there is the possibility of over-fitting the data by adding too many latents. To assess this, we can perform cross-validation, by regressing the data on a subset of the training set then evaluate the performance on the remaining points (leave one out). However, given our small n this does not seem to be effective. There are two alternatives which are Monte Carlo simulations and k-fold cross validation. Both Monte Carlo simulations (data not shown) and k-fold (**Figure 4**) were investigated. They both produced similar results suggesting that for each fold, 3 latents was the most accurate (bottom right panel) which explains 91.4% of the variance. Over fitting is visible beyond 3 latents (top right panel, F**igure 4)**.

Figure 5 - PLS-da is a poor model for the data

**Step 5: Discriminant analysis fails to accurately discern the classification**

The next step is to use the predictors from the PLS regression (to 3 latents) to determine a threshold value that will spilt the points into the two classification groups. These can be evaluated by the ROC (receiver operating characteristic) curves (**Figure 5**). As before cross validation was performed with a 15 k-fold validation. Sadly, discriminant analysis performed rather poorly with an AUC of ~0.784. Furthermore, there was a high degree of variability in the performance in each fold, meaning that the n is not sufficient to make a prediction.

**Step 6: Utilising an SVM after PLS regression produces the most accurate model**

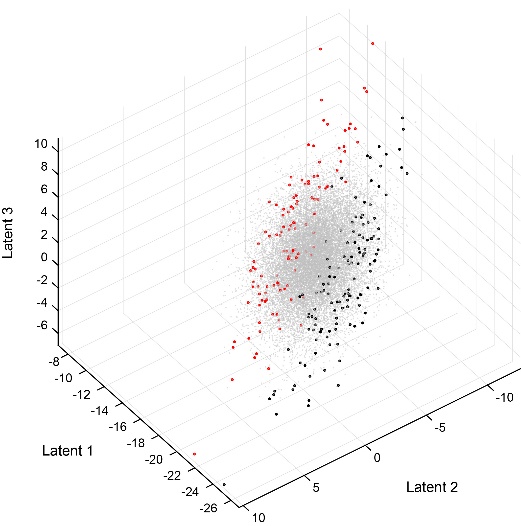
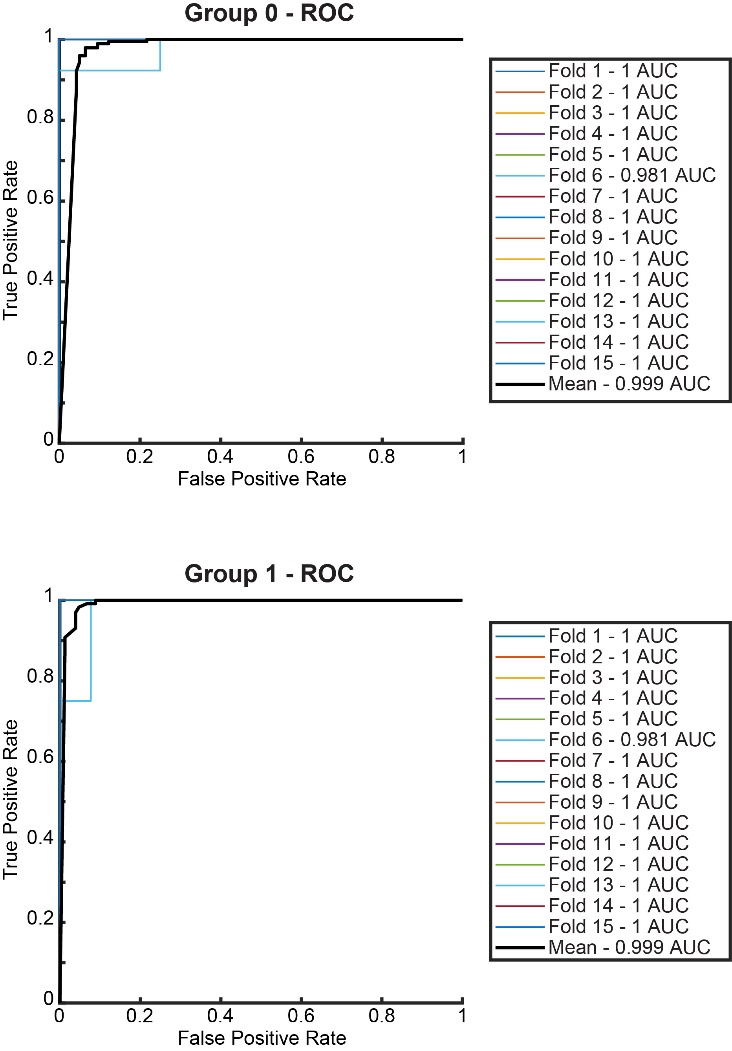
However, the regression aspect is still effective, therefore I investigated a whole range of machine learning models on the training set after it was reduced down to 3 dimensions. Using the classification learner app in MATLAB I was able to assess 30 different models and discovered that a linear support vector machine (SVM) produced the most accurate results (data not shown). Why would this be the case? As the data has now been reduced to 3 dimensions we can now visualise the data and assess how it is split (**Figure 6**). By eye, we can clearly see a difference between group 0 (black points) and group 1 (red points). In fact, there appears to be a line which divides the two groups. Additionally, by plotting the points of an unknown class (grey points). We can see that it looks like the data belongs to a singular cluster split arbitrarily down the middle.

Figure 6 - Visualisation of the 3 latents. Group 0 points (black), Group 1 points (red), and unclassified points (grey).

But how accurate is the SVM? Once again we performed a K fold cross validation before using the full training data as the model. For a fair comparison we used the same number of folds as with the PLS-da (15). The accuracy is now much improved with only one fold producing errors (Fold 6, **Figure 7**). Meaning that the model is highly effective for the training set

**Conclusion**

I’ve produced a model that according to the data here should be highly accurate (~99%). To reduce the dimensionality, I performed a partial least squares regression, and to counter the low sample size a K-fold cross validation was performed. Data was then split using a SVM approach.

However, there is a small caveat. By visualising the latents that I used to train the model it appears that the training set is more broadly distributed than the unknown data points. This is either inherent in the data, or more likely due to the fact that the known data points will influence the regression used to reduce the dimensions.

Figure 7 - SVM ROC curves for both group 0 and group 1. Classification appears to be highly effective