STAT 602 - Final Project

Alex Soupir

April 25, 2020

Libraries used: MASS, foreach, doParallel, knitr, kableExtra, MNLpred, nnet, neuralnet, class, featuretoolsR (some backend python code with reticulate needs to be fixed), magrittr, matrixStats, and e1071

The data being explored in this document is from handwriting of 40 different people (subjects) writing 6 different phrases (conditions) 3 times each and in both cursive and print (group) measured with the *MovAlyzeR*. This document outlines some exploration to find out if there is reason to believe further work is reasonable or time-worthy (*skunk work*) and not a full on analysis itself. *Accuracies were measured by dividing the number of correct predictions by the total number of predictions. Here, predictions are important and will be used as the metric for model performance.*

Initially, the data frame included all of the different "*Segments*" broken down by the direction of the writing, but the mean of all fo the different trials (each combination of subject + condition + group + replication) was taken as a baseline for predictions. With the new summary data where each row is a unique replication. These values were than checked for values that are only zero's and removed. These variables are `RelativeDurationofPrimary`, and `RelativeSizeofPrimary`.

Since from the in class 'Playing with Data' we got some benchmark accuracies from linear discriminate analysis and quadratic discriminate analysis using the principal components of the remaining variables for the different response variables `group`, `subject`, `condition`, and `joint` which is the combination of all 3. **It's important to note that the model for the joint variables doesn't seem to work for the quadratic discriminate analysis**

So these are the accuracies to beat through other methods or further exploration. The next thing that I tried to do was remove other varaibles that didn't *appear* to show separation in the box plots that were used in the 'Playing with Data' code. This was only for `group` separation, however. The columns that visually appeared to have no separation were `PeakVerticalVelocity`, `HorizontalSize`, `AverageNormalizedyJerkPerTrial`, `AbsoluteJerk`, `AverageNormalizedJerkPerTrial`, `NumberOfPeakAccelerationPoints`, and `AveragePenPressure`. This resulted in a higher increase in accuracies for `group` and `condition` but not for the `subject` or `joint` response variable.

With slight improvements in `group` and `condition` but not `subject` or `joint`, I thought another place to look would be using the means themselves in the LDA instead of the principal components, as well as scaling the means. Using the mean values from the variables with visible `group` separation increased the prediction accuracy of `group`, `subject`, `condition`, and `joint` over the use of principal components for the LDA, but the QDA decreased `group` and `subject` accuracies, while slightly boosting `condition`. Adding back in the variables that appeared to have some separation in the boxplots split by `group` increased all 4 response variables when using the LDA and `subject` and `condition` when using QDA. Quadratic discriminate analysis again had shown a decrease in the prediction accuracy for `group` when using the means from the data frame used in class like seen with the means of those variables with some visible separation between the 2 `group` factors - `CUR` and `PRI`. Finally, for LDA and QDA I scaled the means of the data used in class which had minor accuracy increases over the raw means for the LDA's `group` and `condition` (`subject` and `joint` that was the same) where as the QDA didn't change from the unscaled data.

After working with the LDA and QDA above with the column means of the samples, I went with the suggestion given in lecture about logistic regressions. this didn't work for the joint prediction because there were 'too many (12000) weights'. I also did a single non-cross validated approach just to get some kind of an idea and then followed it with a cross validation for better accuracy judgement. The prediction accuracy of `group` and `subject` did increase (the logistic regression without CV was just to see if it was worth looking further into but not itself very trustworthy), and the `condition` accuracy did show an increase but it wasn't as high as the others (`condition` accuracy ended up at only 0.425 while `group` and `subject` were 0.928 and 0.725, respectively). All values were greater than the baseline set in lecture so the logistic regression shows promise.

Single regular `glm` was giving errors, I found the `multinom` logistic regression function which can be used for predicting probabilities of classes for the larger response sizes rather than the response like `glm`. https://www.r-bloggers.com/how-to-multinomial-regression-models-in-r/

Another avenue that I have used for classifications in the past was using neural networks. The `nerualnet` package makes this relatively easy without having to code perceptrons from scratch (actually taught how to do that on codecademy under the data science path for python) along with a bunch of customizations. Looking at some of the recommendations for how big to make the hidden layer, a stack overflow poster had said that good guidelines are 2/3 the input + the output, somewhere between the input shape and output shape, or no more than twice the input. I chose to go with 15 as a round number that was close to the 2/3 input + output for `group` and maintained it for the others. The input data was also normalized (have read some other places this is good practice so it doesn't put inappropriate weights in certain variables and they all start on a "level" playing field). The accuracy of the neural net is surprisingly well for the `group` and `subject` (since the condition failed to converge) with 0.931 and 0.621, respectively. This is better than the baselines, however with `condition` and `joint` not converging it doesn't warrant further investigation at this time. With different parameteters this may be possible. https://stats.stackexchange.com/questions/181/how-to-choose-the-number-of-hidden-layers-and-nodes-in-a-feedforward-neural-netw

Following the logistic regression, I thought maybe looking at a KNN set of models would provide a good way to classify. For KNN, I used the means data frame that was constructed from the 'Playing with data' along with the scaled verson of the means using `scale`. The unscaled accuracy with $K = 1$ was lower for all response combinations than was the scaled data. The best joint prediction came from $K = 1$ for the scaled data where the model achieved an accuracy of 0.342. Other combinations included `group:subject` where the highest accuracy was 0.609 for $K = 1$, `group:condition` with a best accuracy of 0.465 with $K = 1$, and `subject:condition` with a best accuracy of 0.346 on the scaled data with $K = 1$. The scaled data also performed better with just the single response variables, and with $K = 1$. Response `group` was 0.935, `subject` was 0.624, and `condition` was 0.481.

Table 1: KNN accuracies of K=1 and K=2 for both scaled and unscaled means data.

	K=1_Unscaled	K=2_Unscaled	K=1_Scaled	K=2_Scaled
knn.g.s.c.acc	0.1104167	0.0895833	0.3423611	0.2916667
knn.g.s.acc	0.3034722	0.2423611	0.6090278	0.5451389
knn.g.c.acc	0.2000000	0.1694444	0.4645833	0.4020833
knn.s.c.acc	0.1159722	0.0979167	0.3458333	0.2888889
knn.g.acc	0.7326389	0.7159722	0.9347222	0.9236111
knn.s.acc	0.3388889	0.3034722	0.6236111	0.5513889
knn.c.acc	0.2458333	0.2284722	0.4805556	0.4006944

When all of the previous methods weren't able to get the `condition` accuracy up higher than 0.425 (LOOCV logistic regression), I thought it would be good to go back to the beginning and push further with descriptive statistcs than what was done in lecture. At first I tried to use a package called `featuretoolsR`, but there is something that doesn't work right with the backend and `reticulate` to port into python (https://github.com/magnusfurugard/featuretoolsR). Here, I used the `matrixStats` package to compute MovAlyzR variable summaries by trial for mean, median, variance, standard deviation, interquartile range,

range, min, and max. After computing that for each of the output variables, the data frame is now 200 predictors, but there are some that have no real use and can be removed by checking if the variance is zero and if so, remove it. The remaining table has 178 predictors (22 columns had a variance of 0 and therefore no difference between all of the samples [rows]) were used for pca and the different components were looped through to determine the best number for the particular response. Using these metrics as predictors for `group`, `subject`, `condition`, and `joint` proved fairly effective with accuracies of 0.9597, 0.86944, 0.7201, and 0.7076, respectively. So it looks as though an LDA taking more information about the original data is more powerful than just means.

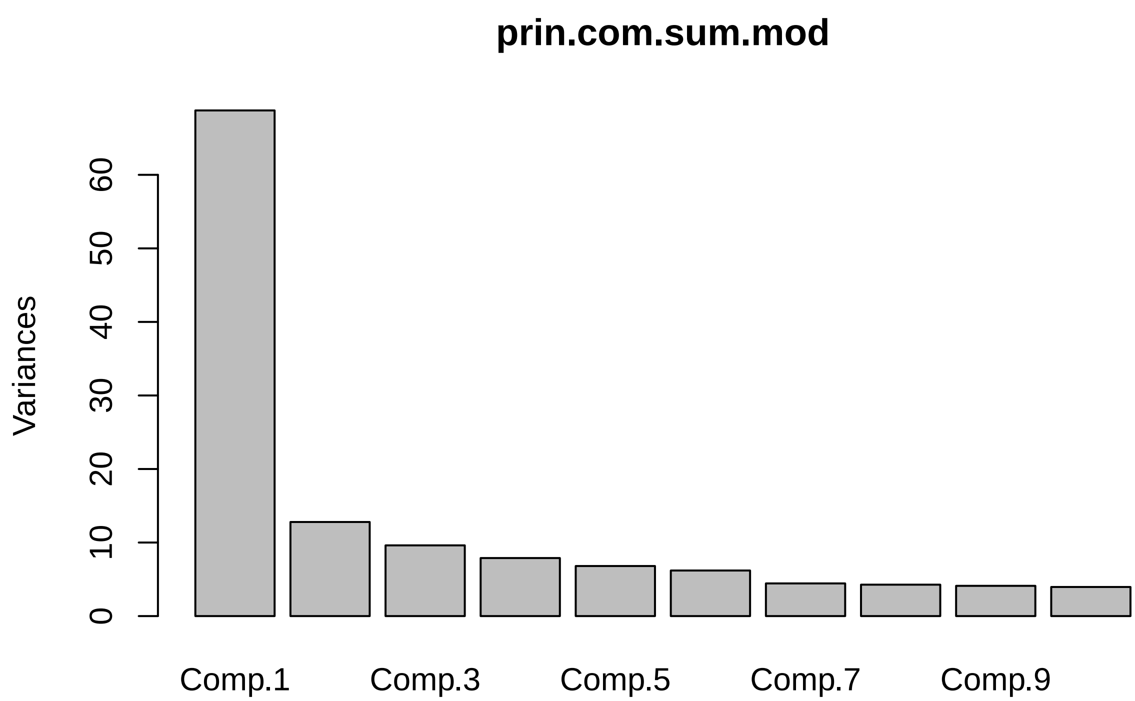

Figure 1: Principal components of the descriptive statistics values for the 23 columns used in lecture (mean, median, variance, standard deviation, interquartile range, range, min, and max).

On top of just using the values mentioned above, I used the `e1071` package to calculate the column skewness, and kurtosis, while adding median absolute deviation from the `matrixStats` package. After computing these for each column, there were now 34 columns that were removed for having a variance of 0. Adding these extra descriptors increased the accuracy of the three response variables independently, but not the `joint` response which decreased from 0.7076 to 0.6813, which is higher than the baseline from lecture. *With these descriptive data, I attempted XGBoost with high number of K-fold (attempt to get close to LOOCV) but is computationally expensive for only about 30% at best prediction accuracy.*

Predictions for the unlabeled data - Using the first descriptive statistics LDA model, the prediction accuracy I would expect to be similar to the test since that was from LOOCV. To be conservative, I would expect predictions for `group` to be around 90%, `subject` around 80%, while `condition` and `joint` accuracy around 70%. To make the predictions, the testing data will be used in two steps with `predict()`; first with the prinipal component models, then with the LDA models to arrive at the final predicted classes.

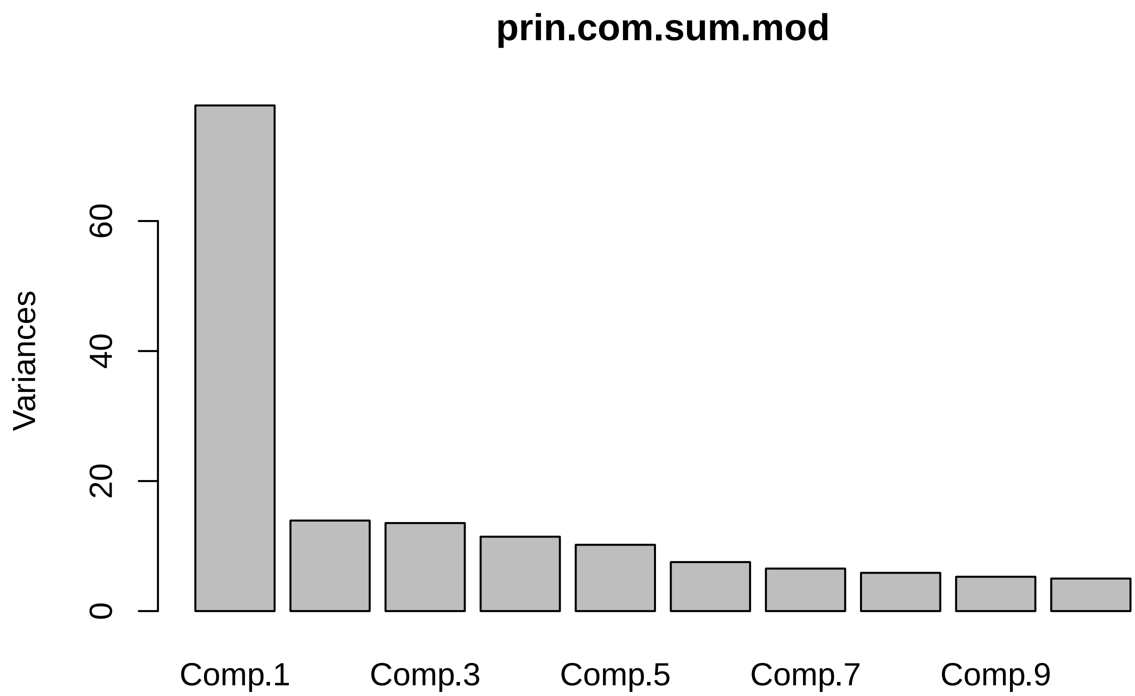

Figure 2: Principal components of the descriptive statistics values for the 23 columns used in lecture (mean, median, variance, standard deviation, interquartile range, range, min, max, skewness, kurtosis and median absolute deviation).

Table 2:

	Group	Subject	Condition	Joint
LDA PCA (cps baseline)	0.8798611	0.3854167	0.3034722	0.2861111
QDA PCA (cps baseline)	0.8256944	0.5444444	0.2638889	NA
LDA PCA vars with separation	0.8951389	0.2895833	0.3263889	0.2347222
QDA PCA vars with separation	0.8687500	0.3986111	0.2743056	NA
LDA on means with separation	0.9145833	0.3979167	0.3722222	0.3756944
QDA on means with separation	0.6979167	0.4736111	0.2847222	NA
LDA on class means	0.9222222	0.7000000	0.4048611	0.5701389
QDA on class means	0.6881944	0.6583333	0.3819444	NA
LDA on scaled class means	0.9229167	0.7000000	0.4055556	0.5701389
QDA on scaled class means	0.6881944	0.6583333	0.3819444	NA
One shot Logistic Regression (no CV) scaled class	0.9326389	0.9430556	0.4555556	NA
Logistic Regression LOOCV scaled class	0.9277778	0.7527778	0.4250000	NA
Neural net LOOCV (1/25)	0.9310345	0.6206897	NA	NA
LDA Descriptive Stats - 1	0.9597222	0.8694444	0.7201389	0.7076389
LDA Descriptive Stats - 2	0.9604167	0.8770833	0.7500000	0.6812500