STATS 202 Final Project

Jeremy P. Binagia

August 15, 2019

1 Introduction

In this report, we will discuss insights obtained from applying the statistical learning methods learned in STATS 202 to a complex dataset that tracks a set of patients that are diagnosed with schizophrenia over time. In short, the dataset contains a set of thirty scores defining the severity of the symptoms of schizophrenia via the Positive and Negative Syndrome Scale (PANSS). Each time a patient is assessed, a number of factors are recorded including the ID of the patient, the ID of the evaluator, the location and day of the assessment, and of course the PANSS scores. The problem background is described in further detail in the project prompt and will not be reiterated here in the interest of keeping the report concise and focused on our team's results and discussion.

The structure of this report follows the organization defined in the project statement. Specifically, we begin our analysis of the data in Section 2, where we utilize statistical tests to ascertain whether or not there is a significant difference in symptoms associated with schizophrenia between patients in a treatment group who are administered an anonymous drug over the course of the study versus those in a separate control group. In Section 3, we sort the patients into the natural "clusters" that arise based on "day 1" factors such as each patient's initial assessment country and visit scores. This is followed by Section 4, where we leverage a variety of statistical learning methods to predict future PANSS scores for a subset of the patients. Finally, in Section 5, we create a set of models that will determine whether or not a given patient assessment will pass an external auditing test. In Section 6, I will provide a brief reiteration of the key takeaways my group acquired from analyzing this dataset.

Note that the code associated with this report may be found at the following repository: `https://github.com/jbinagia/stats202-final-project.git`. My teammate for this project is Sai Gourisankar and our team name for the associated Kaggle competition is "We Use Golf Scoring"; my personal Kaggle username is jbinagia. Both authors contributed equally to the data preparation, analysis, and model training; reports were created independently.

2 Treatment Effect

2.1 Introduction

To begin our analysis of this dataset, we will first check assess the treatment effect. That is, we will examine whether or not patients who received the administered drug demonstrated a significant change in their total PANSS score relative to those who did not. Before delving into our specific hypothesis test, we first sought to visualize the data some intuition for the differences between the treatment and control group. To this end, the total PANSS score for both groups is plotted as a function of the patient's visit day in Fig. 1. At first glance, there does not appear to be a noticeable difference between the treatment and control groups. While it seems unlikely that a statistical test would suggest that the drug does have a significant effect on the patient's, we proceeded with hypothesis testing to formally determine the effect of the drug.

2.2 Primary hypothesis

While there are many possible hypotheses one could consider for this complex dataset, we chose the following hypothesis. Consider a regression of the total PANSS score on the patient's visit day and an interaction

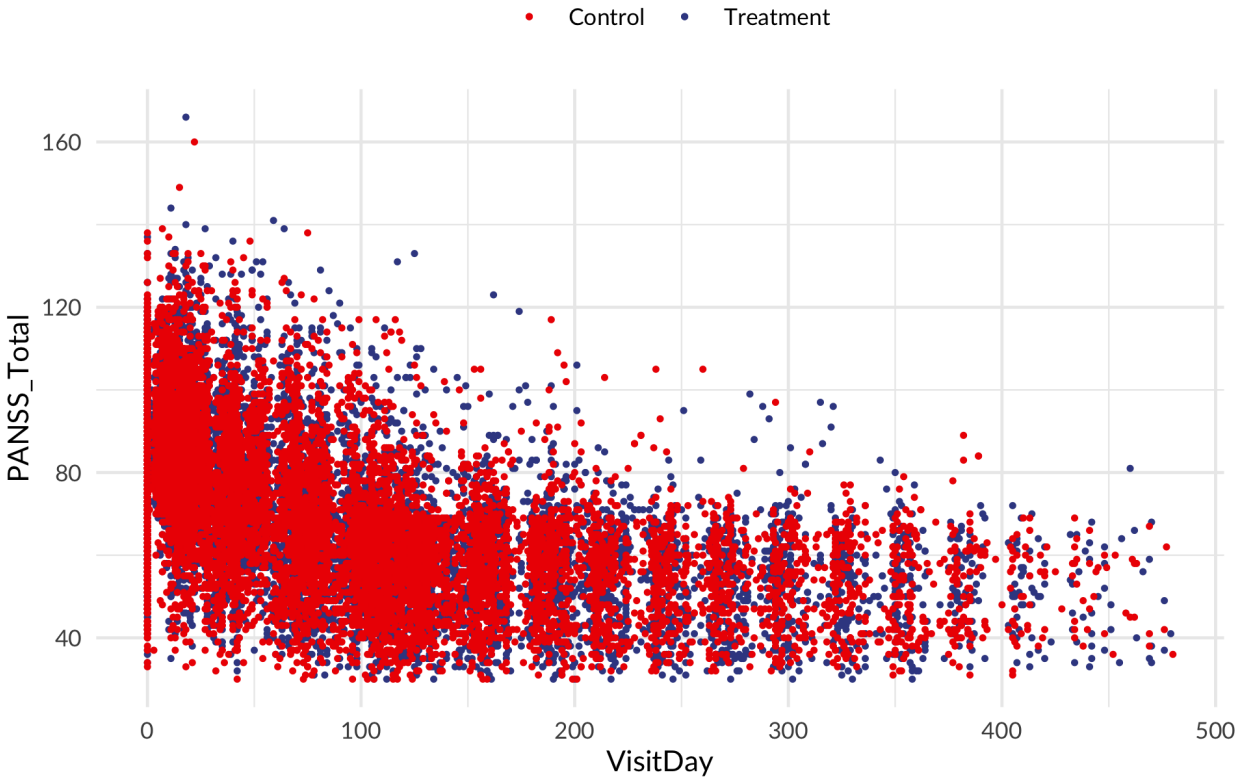


Figure 1: Total PANSS score as a function of each patient's visit day. Visually, we see no appreciable difference between the treatment and control groups.

term between the visit day and a dummy variable for the treatment group, i.e.:

$$\text{PANSS}_{\text{Total}} = \beta_0 + \beta_1 * VisitDay + \beta_2 * VisitDay * Treatment \tag{1}$$

Here we consider a dummy encoding of 1 for the treatment group and 0 for the control group, so that the linear regression for those in the treatment group simplifies to $\text{PANSS}_{\text{Total}} = \beta_0 + (\beta_1 + \beta_2) * VisitDay$. Our null hypothesis is that the treatment has no effect, which amounts to saying that $\beta_2 = 0$. Conceptually, this corresponds to the scenario where the individual regression lines for the two groups have the same slope with respect to the visit day. Intuitively this hypothesis appealed to us since we expected a steeper negative slope with respect to visit day if the drug was to have an effect (if we were to reject the null hypothesis).

Note that the test we are interested in is a partial F-test, whereby we consider the null hypothesis that some subset of our total number of coefficients is zero. That is, we want to understand the effect of adding the interaction term to our model, accounting for the other predictors and their associated coefficients. As we learned in class, the corresponding F-statistic is exactly equal to the square of the t-statistic for each predictor/transformation. Luckily, individual t-statistics (or equivalently p-values) are automatically reported in R once we perform the regression given by Eq. (1). In the case considered here, we found that the p-values for β_0 and β_1 were $< 2\text{x}10^{-}16$ while that for β_2 was equal to 0.893. The corresponding regression coefficients were $\beta_0 = 81.7$, $\beta_1 = -0.117$, and $\beta_2 = 2.13\text{x}10^{-4}$. Hence, there is approximately an 89% chance that we would observe such a value for β_2 by random chance. Thus, we can confidently conclude that there is statistically no significant treatment effect under our given hypothesis, thereby confirming what we qualitatively observed in Fig. 1.

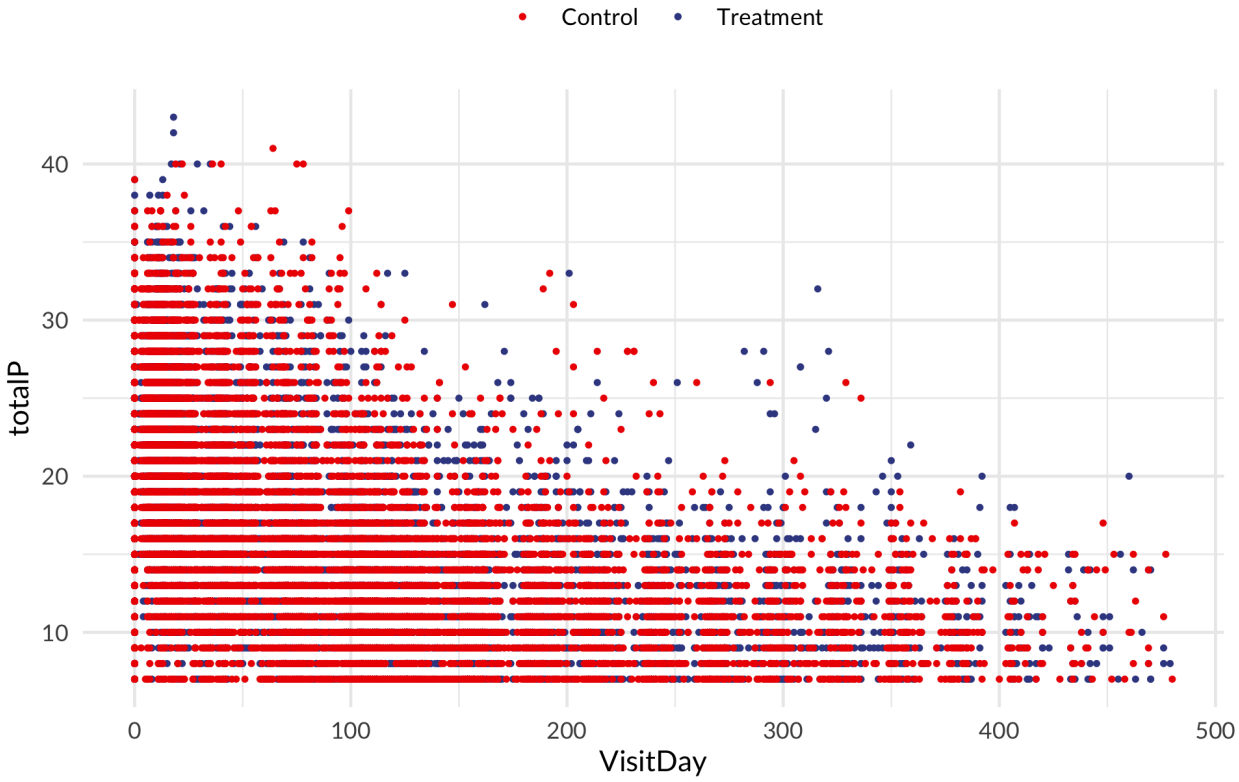


Figure 2: Sum of the PANSS scores for the positive symptoms as a function of each patient's visit day. Visually, we see no appreciable difference between the treatment and control groups.

2.3 Secondary hypotheses

As mention in the previous section, there are many alternatives to the specific hypothesis we considered. For example, one could conduct the same model form but regress total scores for a given category (i.e. the sum of the scores corresponding to just the positive symptoms). Analogous to Fig. 1, we can create a scatter plot of the total score for a category vs. visit day to visually assess the effect of the treatment. This is shown for the case of the positive symptoms in Fig. 2. As before, we do not visually see a significant difference between the treatment and control groups. Conducting regression tests analogous to Eq. (1) but changing the response appropriately, we obtain three new p-values (one for the positive, negative, and general categories). These values are 0.592, 0.663, and 0.705 respectively. Clearly each of these are quite large (seeing as the most generous threshold for significance is having a p-value less than 0.05), so we can safely conclude that the treatment does not have a statistically significant effect on the totality of symptoms in a given category.

To be absolutely sure we were not missing the effect of the treatment in some way, we qualitatively probed two other scenarios. First, we considered the same type of hypothesis as before but now having the response be one of the individual PANSS scores. For brevity these 30 distinct scatter plots are not included in the report but instead may be found in our GitHub repo. The visual conclusion is the same regardless of which score is visualized, however: the qualitative trend for the treatment and control groups are virtually indistinguishable. The second scenario we considered is that perhaps there was an effect on patients in certain studies and not others (for example, as a result of unaccounted differences in how each study was conducted). Before proceeding with a rigorous hypothesis test, we plotted the total PANSS score for each study as a function of visit day. Shown in Fig. 3 is the case of patients who were in study E. Again, we no noticeable difference between the two groups; for this reason, we did not proceed with a partial F-test given the results above.

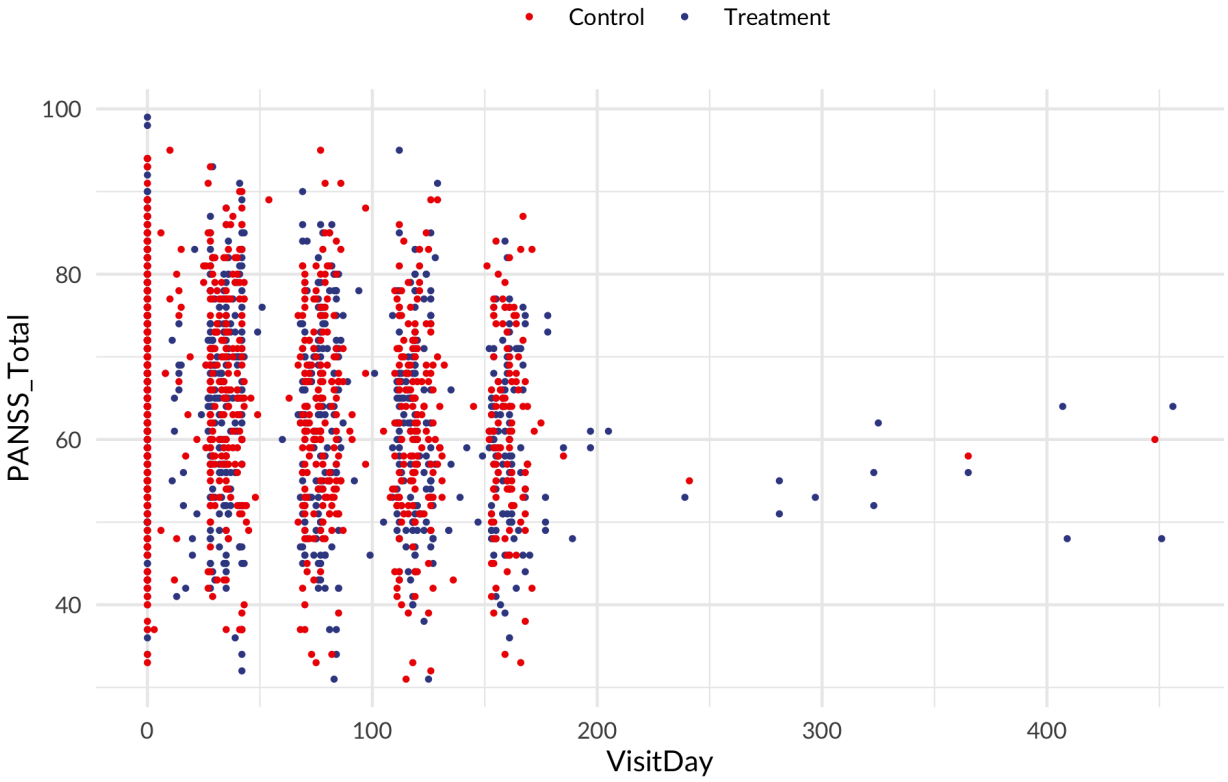


Figure 3: Total PANSS score as a function of each patient's visit day for patient's in study E. Visually, we see no appreciable difference between the treatment and control groups.

2.4 Discussion

In the preceding sections, we concluded that there was no significant difference between the treatment and control groups when it comes to their PANSS scores as a function of time. This is quite interesting seeing as the scores themselves do decrease over time (as desired); it's just that the driver for this decrease does not seem to be the drug itself. What does this suggest about the study? Interestingly enough, this means that either the symptoms naturally decrease over time or that perhaps the mere act of attending the scheduled evaluations leads to a decrease in the symptoms. We believe either hypothesis is intriguing and should be the focus of future studies regarding schizophrenia.

3 Patient Segmentation

3.1 General approach

In this second major part of this project, we attempt to discover the natural groups that emerge given the patients' day 1 information, such as their initial individual PANSS scores. This problem must be tackled using unsupervised learning techniques; two of the most widely used methods that fall into this category are K-means clustering and hierarchical clustering. While we utilized the former and it will be discussed further below, we did not make use of hierarchical methods for the following reason. Hierarchical methods are suited, as the name suggests, when there is a natural hierarchy in the data. This is certainly not the case for our dataset when considering each of the thirty individual PANSS scores; they do not form a hierarchy as would say a dataset concerning different types of fruits (where splits are related to categories such as color or shape).

For this reason, we relied on centroid methods like K-means clustering to determine the clusters that

	Score <dbl>	Method <fctr>	Clusters <fctr>
APN	0.0516	kmeans	2
AD	6.5612	kmeans	6
ADM	0.2116	kmeans	2
FOM	0.8901	kmeans	8
Connectivity	1094.8381	kmeans	2
Dunn	0.1340	pam	6
Silhouette	0.1218	pam	2

Figure 4: `clValid` clustering results. The method `clValid` returns the optimal method and number of clusters given various internal and stability measures (listed on the left).

define the patients in the given studies. Note that categorical data does not lend itself to K-means clustering since a mean of such data cannot be computed (to find the center of each cluster); this means we are limited to clustering solely based on the initial visit PANSS scores. Still, this provides us with thirty independent dimensions, so this is not a great limitation. We should also mention that we do not consider any IDs (e.g. patient or site ID) for the purposes of segmentation since these are arbitrarily assigned and do not measure some inherent feature or property of the patients. Lastly, we do not include the total PANSS score since we already consider all of the individual scores (the former is simply the sum of the latter so including it would yield no new information about the patients).

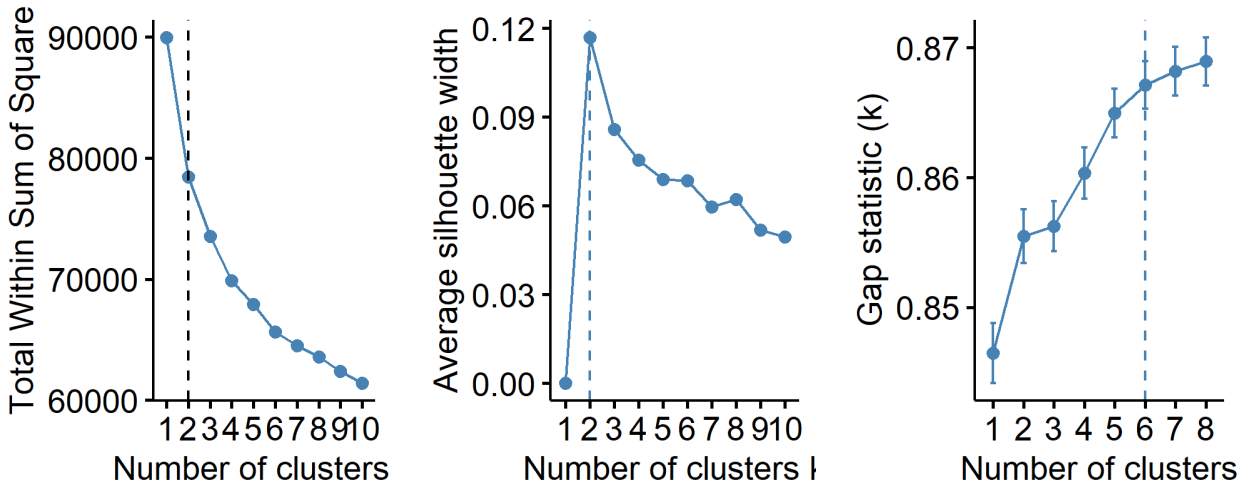
If we are relying on centroid-based methods like K-means clustering, the key question is how do we select number of clusters K? To answer this question, we first used the `clValid` method to give us a starting point for the number of clusters we should be considering. Given a range of numbers of clusters to consider, `clValid` evaluates each one via several metrics based around internal measures such as cluster connectivity as well as cluster stability measures. While I will not describe every metric that is used in the interest of time (although I will describe the key ones we focused on for tuning the specific methods), detailed descriptions of each metric can easily be found in the R documentation for the method. The key takeaway for this subsection is that `clValid` returns the optimal method and cluster size given each metric, as seen in Fig. 4. Notably, a cluster number of 2 is selected 4 out of 7 times, and the next largest size that is selected is having 6 total clusters. Thus, we have reason to suspect that having 2 clusters might be the best choice given how it performs on these various methods, especially if it is only slightly inferior to the larger number of clusters in regards to the other metrics shown here. Furthermore, as we shall soon see, selecting 2 distinct clusters is far superior than having 6 clusters in terms of interpretability.

3.2 K-means

As we mentioned above, we would like to use K-means clustering to group the patients given their initial visit information. While we obtained an initial idea of what cluster sizes we should focus on via `clValid`, we now use the method `fviz_nbclust` from the `factoextra` package to gain detailed information for determining the value of K specifically for the K-means clustering algorithm.

The first plot that `fviz_nbclust` returns is a plot of the total within sum of squares error (WSS) as a function of the number of clusters (Fig. 5a). We refer to this methodology for selecting K as the elbow method since we are searching for the number of clusters that defines the "elbow" of the curve; the reasoning is that this is approximately the point of diminishing returns in regards to WSS. While the elbow is not as clear in this instance as it might be for other problems, we considered our previously favored value of $K = 2$ from `clValid` to be a suitable demarcation in this scenario.

We also examine the average silhouette width (Fig. 5b) when selecting the number of clusters for K-means clustering. In short, the silhouette width of an observation is the difference between its average dissimilarity for its cluster and its average dissimilarity for that of the closest neighboring cluster. The average silhouette width is simply the mean of these individual measures. Hence, we would like to maximize this value so that clusters are compact and neatly separated from one another. Note that the average silhouette width is maximized in this instance when the number of clusters is equal to 2, providing further evidence that this is the value we should ultimately select. Finally, we also consider the gap statistic when deciding on the number of clusters to use for the K-means algorithm. The gap statistic compares the intra-cluster variation to that given a random uniform distribution of the data. Thus, a large gap statistic signifies a good clustering of the data since the intra-cluster variation is significantly different from that under the aforementioned null



(a) Elbow method (b) Silhouette method (c) Gap statistic

Figure 5: Selecting the optimal number of clusters for the K-means algorithm via the elbow method, silhouette method, and the gap statistic.

distribution. `fviz_nbclust` returns the gap statistic as a function of the number of clusters, as shown in Fig. 5c. The largest value of K (while also making use of the one standard error rule) is located at $K = 6$. Interestingly enough, there exists a short elbow at $K = 2$ where there appears to be negligible difference in having 3 clusters versus 2.

As our final consideration for selecting the total number of clusters, we use the `NbClust` method; this method computes 30 different indices with which to select the optimal number of clusters. Detailed information regarding each metric is provided in the associated R documentation. The significant thing for this report is that out of the 30 indices, 10 proposed 2 as the ideal number of clusters. The next most-selected number of clusters was 3, which was selected by 4 different indices. Additional information concerning the output of this function may be found in our segmentation script located in our aforementioned GitHub repository; the key takeaway is that again we see that having 2 clusters tends to be the best option for the majority of these indices.

Given all of the evidence we have obtained for selecting our number of clusters, we proceed to use the K-means clustering algorithm with $K = 2$ after scaling the predictors to have zero mean and unit variance; the results are shown in Fig. 6. Because our data has 30 dimensions at this point, the algorithm automatically visualizes the data along the first two principal components. Interestingly enough, selecting two clusters neatly divides the data along observations who first principal component is positive versus those whose first principal component is negative. This will be discussed in detail at the end of this section of the report.

3.3 K-medoids

There is a method closely related to K-means clustering known as K-medoids, otherwise known as the PAM algorithm (partitioning around medoids). Whereas the center of a cluster in K-means is defined by the mean of the observations, the center of a cluster in PAM is actually one of the data points (i.e. a "medoid"). We became interested in performing some of our clustering analysis with PAM to not only provide a "second opinion" in regards to how the observations might become organized, but also to shore up a potential weakness of K-means. The issue with K-means is that outliers have extreme weight on an average, as opposed to say a median of a distribution. Thus, many view k-medoids as a more robust method since it is not susceptible to this same issue, making it more robust to noise and outliers in the dataset.

As before, we visualize a number of metrics to select the number of clusters for PAM (i.e. Fig. 7). The results are quite similar to that obtained for the K-means algorithm; for this reason, we make the same

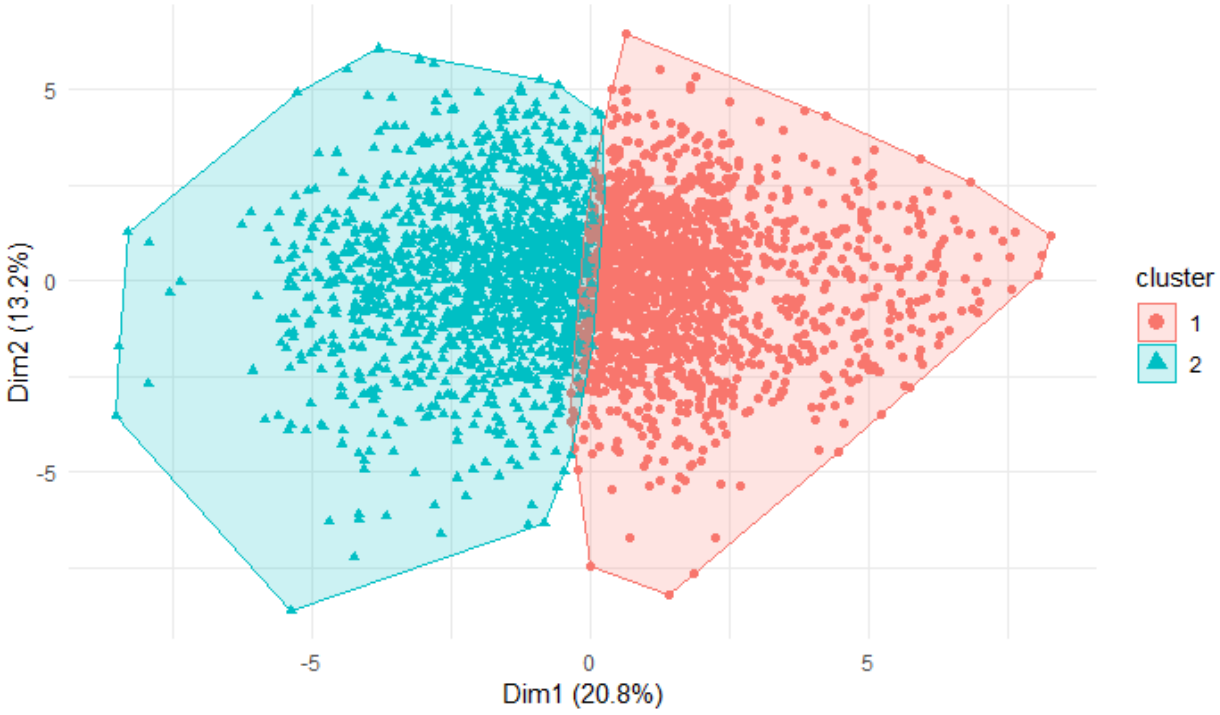


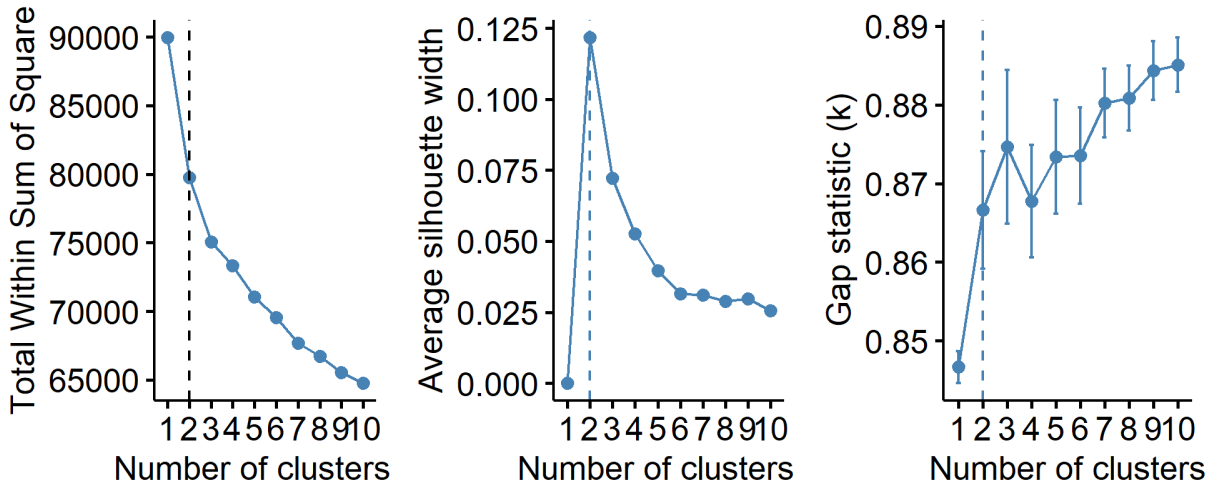
Figure 6: K-means clustering applied with $K = 2$.

conclusions as before (namely leaning towards using $K = 2$ as our selected number of clusters). The fact that Fig. 7 so closely resembles Fig. 5 also gives us confidence in our overall approach (i.e. the clustering is not wildly different once we use a new algorithm).

In regards to the gap statistic for PAM, the optimal value of K selected under the one standard error rule is $K = 2$ (Fig. 7c). This again corroborates what we found for the K-means algorithm, even though the algorithms are of course distinct from one another. Thus, given the above three metrics and the results of K-means subsection, we proceed with the PAM algorithm using two clusters (again scaling the predictors as before), the results of which are shown in Fig. 8. Again, the two clusters are primarily defined by their first principal component value. That is, cluster 2 is largely associated with the positive values of this axis while cluster 1 is associated with the negative values of the first principal component. In contrast to before, however, the clusters have significantly more overlap now. We attribute this to how the two methods select cluster centers, as discussed in the beginning of this section. Overall, though, the results of the PAM algorithm complement those from K-means, giving us faith in the fact that a division along the midpoint of the first principal component is likely the best choice for this dataset.

3.4 Discussion

We found using both K-means and K-medoids (PAM) that the optimal clustering for this dataset is two clusters of roughly the same size that are separated along the first principal component. These principal components were generating automatically by the clustering algorithms; what do they represent in regards to the original dimensions of the data? We can determine this by running a principal component analysis (PCA) independent from clustering via the `prcomp` function, the results of which are shown in Fig. 9. We found that the alternative visualization provided by the `factoextra` package, however, to be easier to interpret (Fig. 10). Here, we not only show how the principal component directions are related to the original dimensions via the loading vectors, but through the colorbar we can better illustrate the contribution of each variable to each principal component. Note that unfortunately these loadings are opposite that seen in Fig. 9; this is of course because the principal components are unique to a sign change (i.e. eigenvectors uniquely



(a) Elbow method (b) Silhouette method (c) Gap statistic

Figure 7: Selecting the optimal number of clusters for the PAM algorithm via the elbow method, silhouette method, and gap statistic.

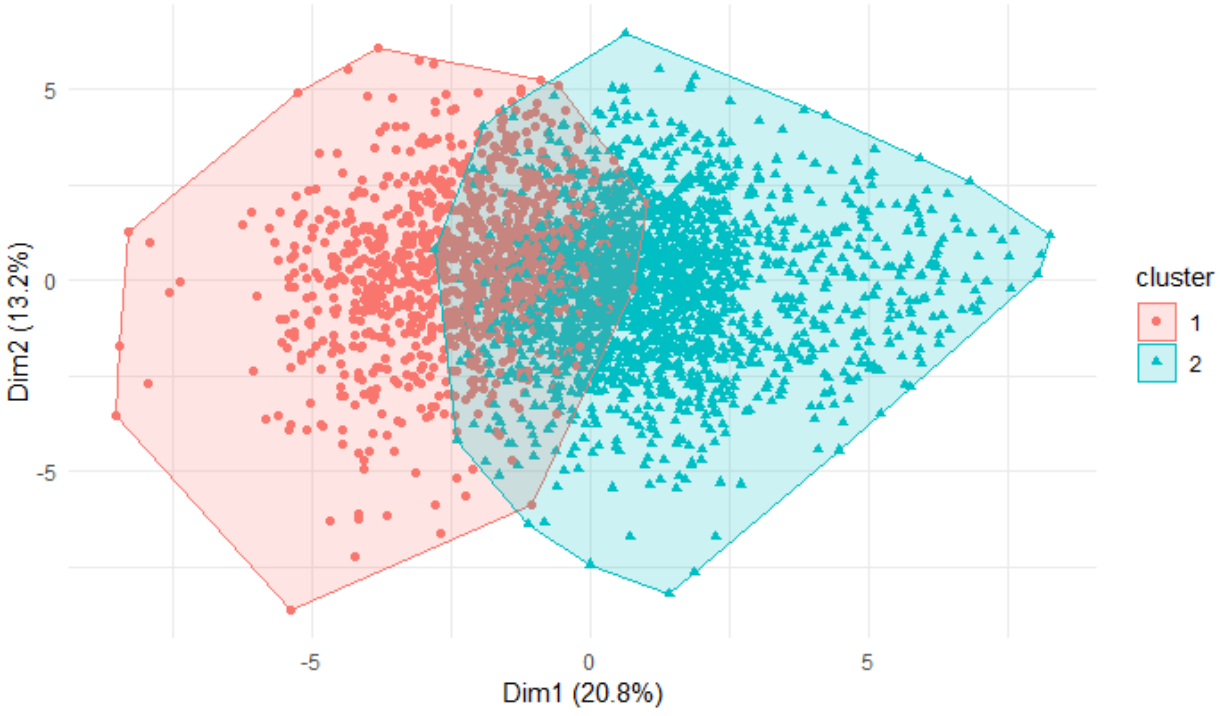


Figure 8

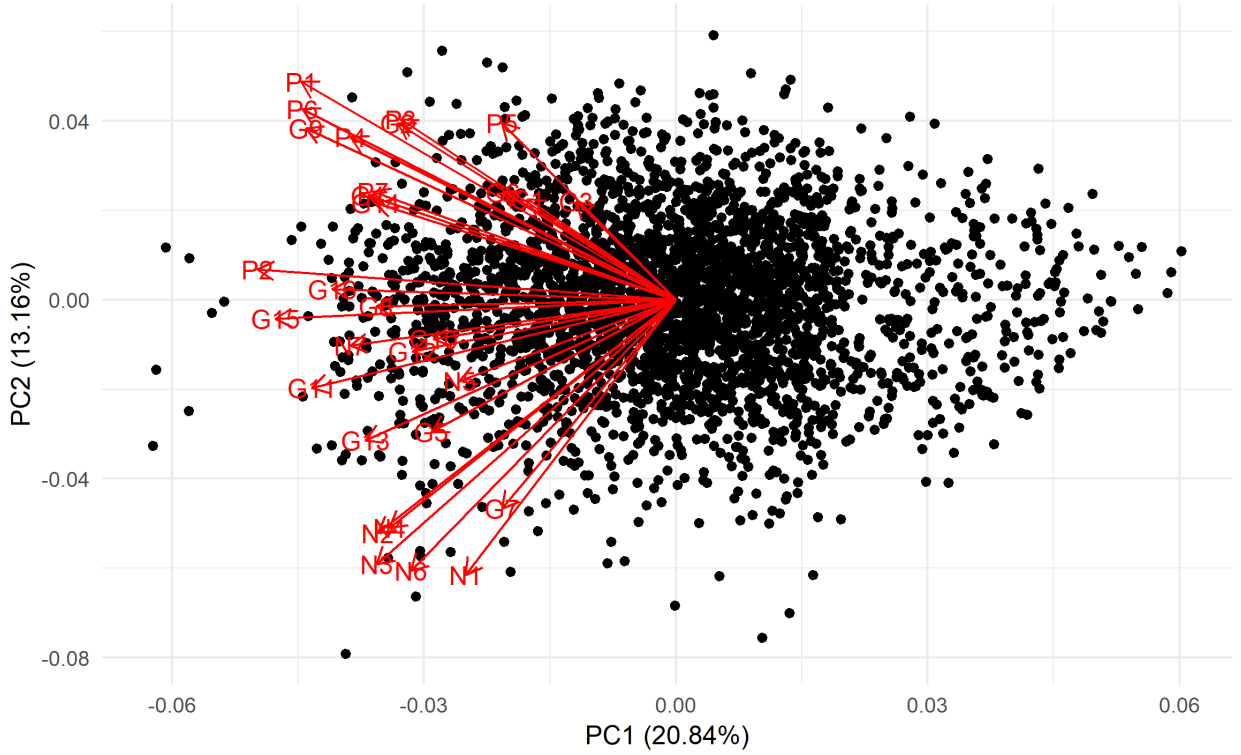


Figure 9: Variable contributions to the first two principal components (using the `prcomp()` method).

define a line rather than a vector direction). This can be seen by plotting the individual observations (colored by their quality of representation by the principal components) in Fig. 11. In other words, Figs. 10 and 11 are 180° rotations of the principal components obtained via the clustering algorithms.

While Figs. 9 and 10 give us an idea of the contributions to the principal components, the percentage contribution can be explicitly calculated and is shown in Fig. 12. We see clearly that the first principal component is mostly explained by P2, G15, P1, P6, G9, and G11 respectively in that order. This means that the clusters are mainly separated by how patients scored on positive symptoms and general psychology symptoms (since the cluster boundary occurs when the first principal component is approximately zero). Indeed, as seen from Fig. 12b, the first five contributors for the second principal components are all scores related to negative symptoms (recall that there are 7 total negative symptom scores). Taking the information from Fig. 12 together with Fig. 9, we see that patients grouped into the cluster located at small principal component 1 (PC1) values (the left cluster) corresponds to patients who scored highly on the positive and general symptoms; the reverse is true for patients found in the other (right) cluster.

The final item we explored in the segmentation section is how the study groups aligned with the clusters derived in this portion of the project. Overlays of the study groups on top of the individual observations in the plane defined by the first two principal components are shown in Figs. 13 and 14. The difference in these two plots is purely visual; it is a matter of illustrating the clusters with ellipses whose centers are located at the cluster centers or using the enclosing polygon that contains all of the observations in a given cluster. The interesting thing to note from these plots is that while study groups B-D are roughly centered in this plane, study group A is clearly biased towards the top left quadrant while study group E is biased towards the right half-plane. What this means is that participants in study group A tend to score higher on the positive and general PNASS scores and lower on the negative symptoms (see for example Fig. 9). In contrast, participants in study group E are characterized by low scores on most of the positive and general symptom scores. This is important to note since it suggests study group E is somewhat distinct from the rest of the studies; in other words, there may be some bias in the succeeding parts of the report where often we use participants in study group E as the test set and the remaining data as our training set.

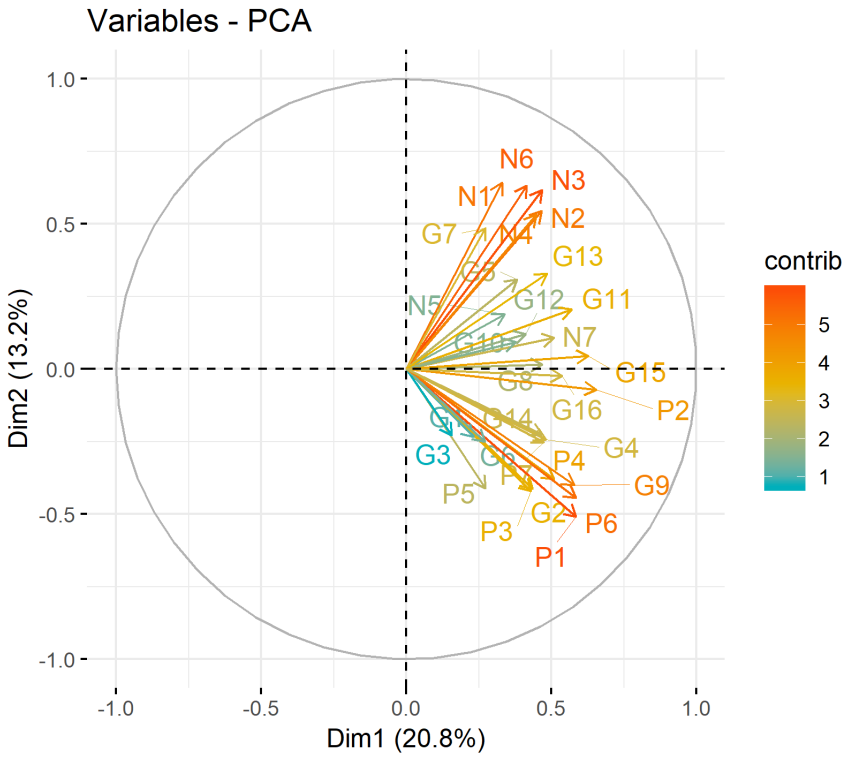


Figure 10: Variable contributions to the first two principal components (using the `fviz_pca_var()` method from the `factoextra` package)

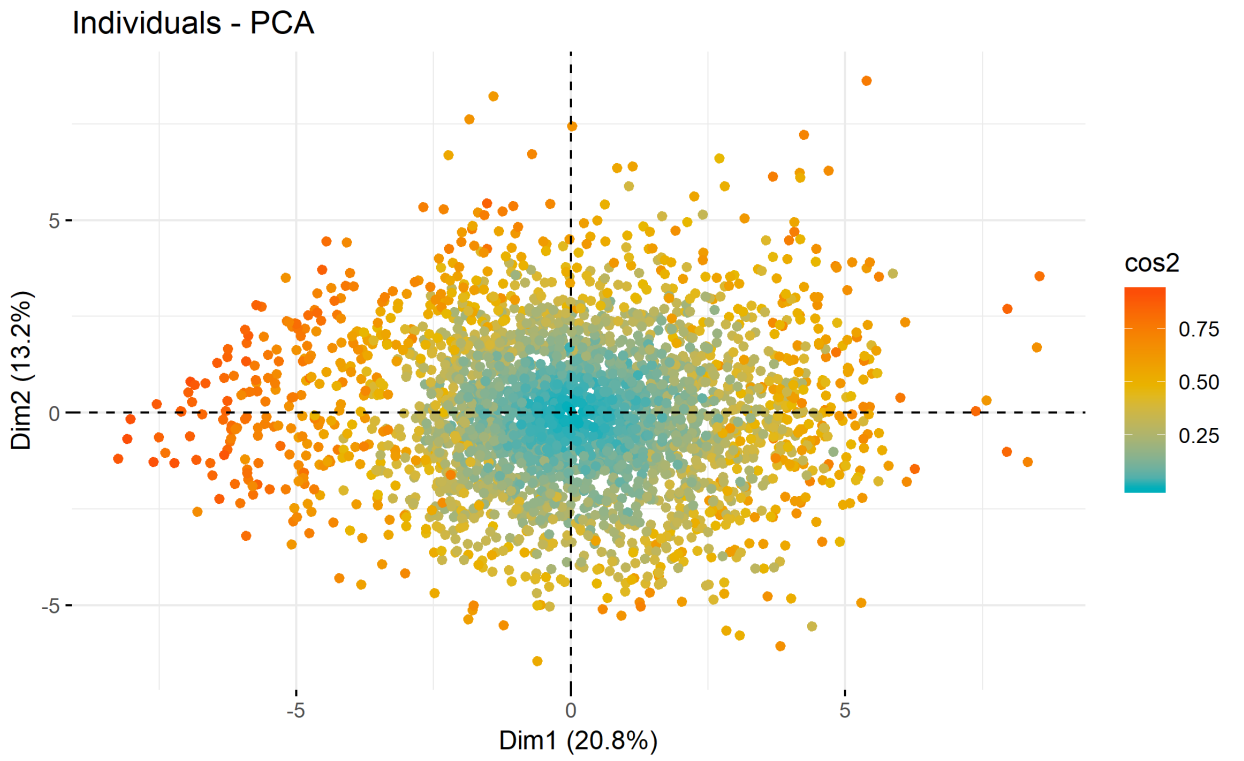
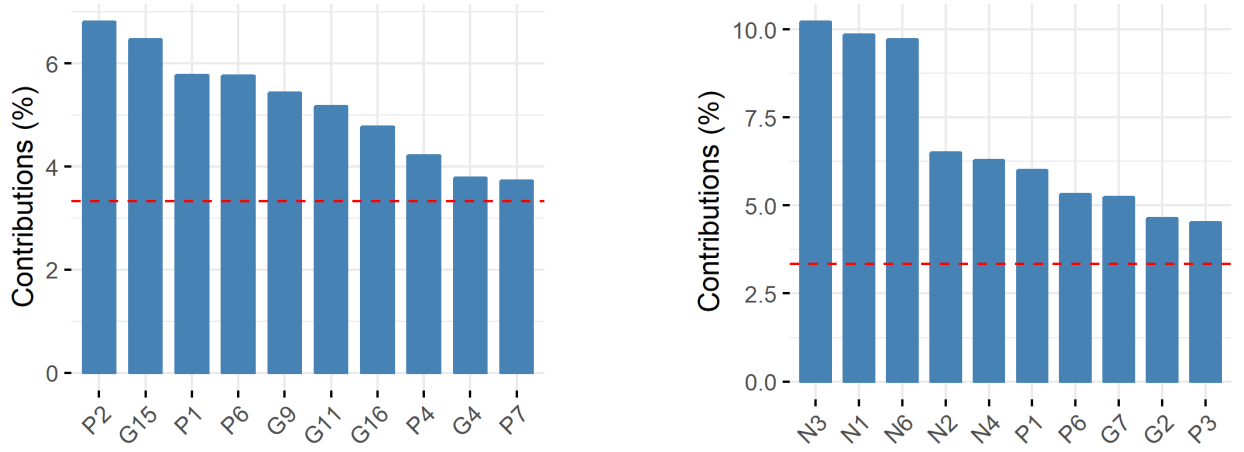


Figure 11: Individual contributions to the first two principal components (using the `fviz_pca_var()` method from the `factoextra` package). The colorbar denotes the quality of representation of each observation in this plane defined by the first two principal components. Hence, `cos2` is small for observations that predominantly lie along the other principal dimensions that are not shown here.



(a) Contributions to PC1

(b) Contributions to PC2

Figure 12: The percentage contributions to the first two principal components (PC1 and PC2).

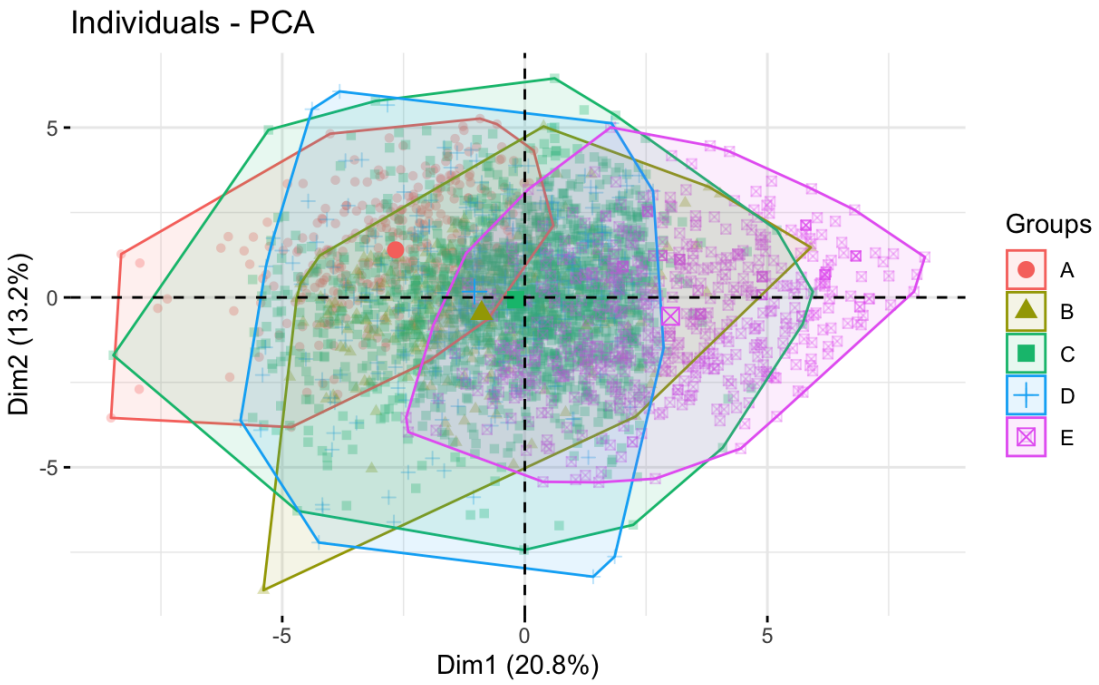


Figure 13: Individuals grouped by their study group in the plane defined by the first two principal components (using ellipses centered at the cluster centers to illustrate the groups).

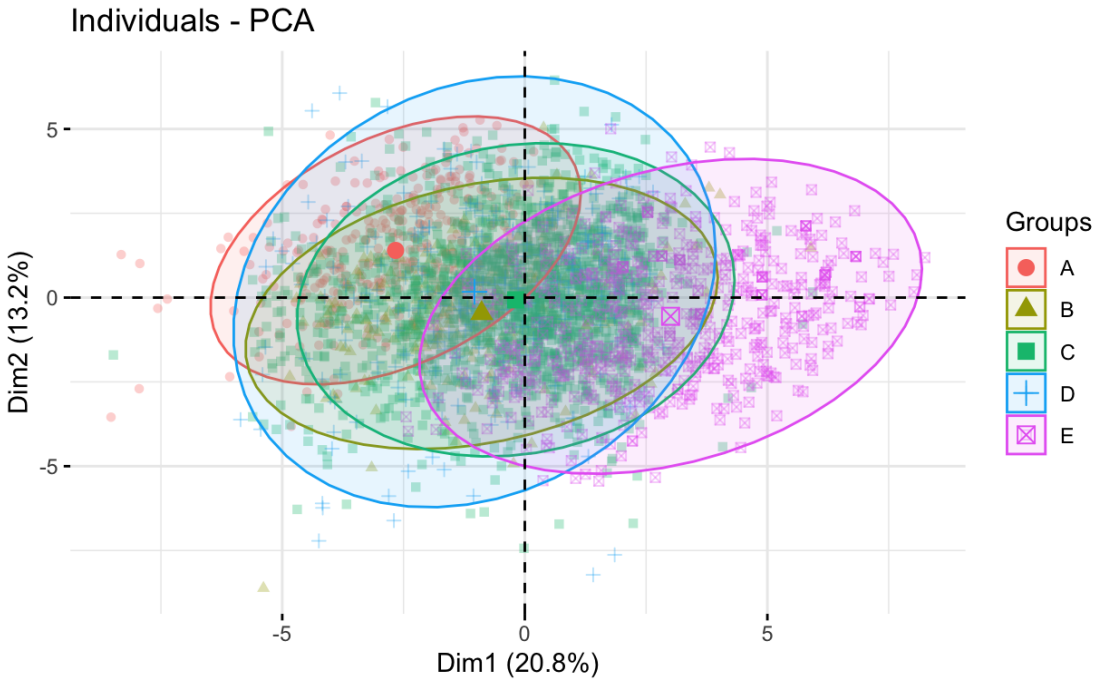


Figure 14: Individuals grouped by their study group in the plane defined by the first two principal components (using the total enclosing surface to illustrate the groups).

4 Forecasting

4.1 General approach

For the third portion of this project, we attempt to predict the 18th week total PANSS scores for the patients in study E. In this subsection, I will describe some of the thoughts and considerations my group thought about before attempting to forecast the 18th week scores. Perhaps the most obvious question is how to define the "18th" week. While at first glance this seems like a straightforward determination (i.e. defining it as a visit where `visitDay` ≈ 126, there are several complicating factors. Some patients drop out of the study, some begin the study much later than others, and others yet have sporadic assessments (so we might expect their scores to not align with someone who visits every week). Because of these complicating factors, we define the 18th week simply as each patient's next visit day, i.e. a visit that occurs exactly one week after their last recorded visit (specifically those subset of patients who are in study E). While this is quite a crude approximation, we thought that it would be an appropriate starting point for analyzing this data-set given the complexities mentioned above. Several alternatives to handling this part of the problem setup are discussed in the final subsection of this part of the report.

In terms of predicting the total PANSS scores for the 18th week, we of course can only base our predictions on what we know must be true of the patient at that point. That is, we cannot utilize individual PANSS scores as predictors since these are unknown. Rather, we must use characteristics of each patient. Specifically, we considered each patient's country, treatment group, visit day, and study group as predictors. Note that we exclude `SiteID` and `RaterID` as predictors since we do not know a priori where or by whom the patient will be evaluated. Finally, `PatientID` is used as a "key" from which we can determine the above mentioned characteristics of each patient when it comes time to make predictions.

Finally, the last item to discuss in terms of problem setup is how we split up our dataset into a training and test set for this portion of the project. Here, we consider the training set to be the data our models learn from. Specifically, it is the set of observations leftover after removing the test set, which consists of the 379 patients we wish to make 18th week predictions for.

4.2 The naive prediction

Just as we will do in the classification portion of this project, we establish a baseline prediction by considering the most naive forecasting method possible. In this case, the most naive approach would be to simply use the most recent total PANSS score for each patient as what we expect them to score on their 18th week visit (i.e. their next visit). This prediction yielded a test MSE of 30.90 (aka the score on the public Kaggle leaderboard). It should be noted that while the idea behind this approach is as crude as one can imagine, at the time of writing this score alone would place our team at the 11th position on the leaderboard out of 38 competitors. Why does this naive approach seem to work so well? We believe part of the reason this approach achieves such a relatively low MSE is that the total PANSS scores cease to vary as a function of `VisitDay` for long times. This is illustrated for example by Fig. 1. Thus, because the total scores do not change significantly by the end of the study, simply using the most recent turns out to be a decent way to forecast future values.

4.3 Exponential smoothing

As will be discussed shortly, several of the other statistical learning methods we considered failed to outperform our naive forecast. Thus, at some point we realized that the most fruitful path forward would be to optimize the naive prediction method. The most obvious way to do so is to include more and more historical data into the prediction for the 18th week. In terms of time series analysis, this idea is formally known as "exponential smoothing". In exponential smoothing, we consider each observation as a linear combination of the two preceding observations:

$$y_i = \alpha y_{i-1} + (1-\alpha) y_{i-2} \tag{2}$$

where y_i denotes the i-th total PANSS score and $0 < \alpha < 1$ is known as the smoothing factor. This equation defines a recursive relationship that goes back to the initial set of observations with respect to time. In this

equation, we see that α determines the relative emphasis that should be placed on historical data (we see that the limit of α approaching 1 represents the naive prediction discussed in the preceding paragraph).

Thus, in an attempt to improve upon the performance exhibited by the naive prediction, we utilize exponential smoothing for a range of values of alpha, truncating the recursion relation at an appropriate number of terms (such that the truncation error is less than 0.5%). Specifically, we consider values of $\alpha = 0.9$, 0.8, and 0.7 and we truncate by only considering the most recent two, three, and four points in time respectively. These predictions yield test MSEs of 29.72, 30.77, and 31.70 respectively; the lowest of these scores places our team at the 6th position on the public leaderboard at the time of writing. It is interesting to note that while the performance of these measures are certainly competitive (and can easily change once the private leaderboard is calculated), the test MSE increases as we include more historical data in our prediction (the exception being that considering the two most recent assessment scores outperforms only considering the very most recent score, aka our naive prediction).

4.4 Discussion

Note that in our GitHub repository, one will find that we actually utilized several of the learning techniques we learned in class including the gradient boosting method (specifically XGBoost) and random forests. These methods were not discussed further here since they yielded test MSEs up to four times larger than found via the naive prediction described above and the predictions provided by exponential smoothing. For example, our hypertuned random forest model yielded a test MSE of 121.52632 (i.e. the value on the public Kaggle leaderboard). We believe this poor performance is attributed to the inherent difficulty tree-based methods have with forecasting. That is, a tree-based method fundamentally cannot predict values outside the range observed in the test set (since the value at a node is calculated as the mean of the contained observations). Indeed, when we examined the 18th week scores predicted by these models, the range of values was much smaller than that observed in the 17th week scores for the patients in study E.

5 Classification

5.1 General approach

In this section, we will attempt to build a set of models that will help us determine if an assessment will pass an external auditing test (i.e. will it be flagged for review or assigned to a clinical specialist for a follow up analysis). As was done in the previous section, we begin by elaborating on the assumptions we made in terms of what predictors we considered, how we split the data up, and what quantitative performance measures we focused on.

In terms of predictors, we primarily focus on the country the assessment was conducted in, whether the patient is part of the treatment or control group, the day of the visit, and the total PANSS score for that visit day. Note that for some methods, we cannot consider country as a predictor since there are assessments in study E that have the United Kingdom as the country of the assessment, a country that is not found in any of the other studies. The reason this is a problem is that certain statistical learning methods cannot handle previously unseen (i.e. not seen in the training set) categorical values (e.g. logistic regression); other methods (e.g. Naive Bayes) do not face this same limitation.

In terms of splitting the data into a training, development, and test set, we take the following approach. We first set aside study E as the test set, i.e. the set of data we wish to make predictions for the `LeadStatus` variable. With the remaining set of observations (comprised of data from studies A-D), we set aside a randomly selected 70% as a training set (to build our models) and 30% as an independent development set (for evaluating the performance of different models). Specifically, we examining how a model performs on the development (dev) set, we will focus on the area under the curve (AUC) of the ROC curve as well as the cross-entropy (or log loss) measures since these do not depend on the probability threshold chosen for an assessment to be considered "Passed" or not. We believe this is the appropriate approach since ultimately we are responsible for predicting the *probability* of an assessment being flagged rather than predicting the binary outcomes ("Pass" or "Fail") themselves.

5.2 Naive Bayes

Analogous to our naive prediction for the forecasting portion of this project, we will now consider the perhaps the most straightforward classification methodology. The first approach we considered is appropriately referred to as the naive Bayes classifier. The naive Bayes classifier essentially calculates the response probabilities via Bayes theorem under the assumption that the predictor variables are conditionally independent of one another (thereby greatly facilitating the computation). The method is referred to as "naive" since this assumption almost assuredly does not hold for some subset of the predictors in the in dataset. Despite this, the naive Bayes classifier serves as a useful "baseline" prediction from which we compare the performance of our other models against. From Fig. 15 we see that the naive Bayes classifier performs decently well on the dev set, with an AUC and log loss of 0.77 and 0.47 respectively. Note that this performance is not reflected in the test set (the public leaderboard), where the log loss is 0.70123. This harkens back to the observation at the end of Section 3, where we noted that study E is distinctly different from patients of the other studies at least in their initial visit scores. We believe we are seeing this bias manifested here since the priors in the naive Bayes classifier are calculated solely based off of patients from studies A-D.

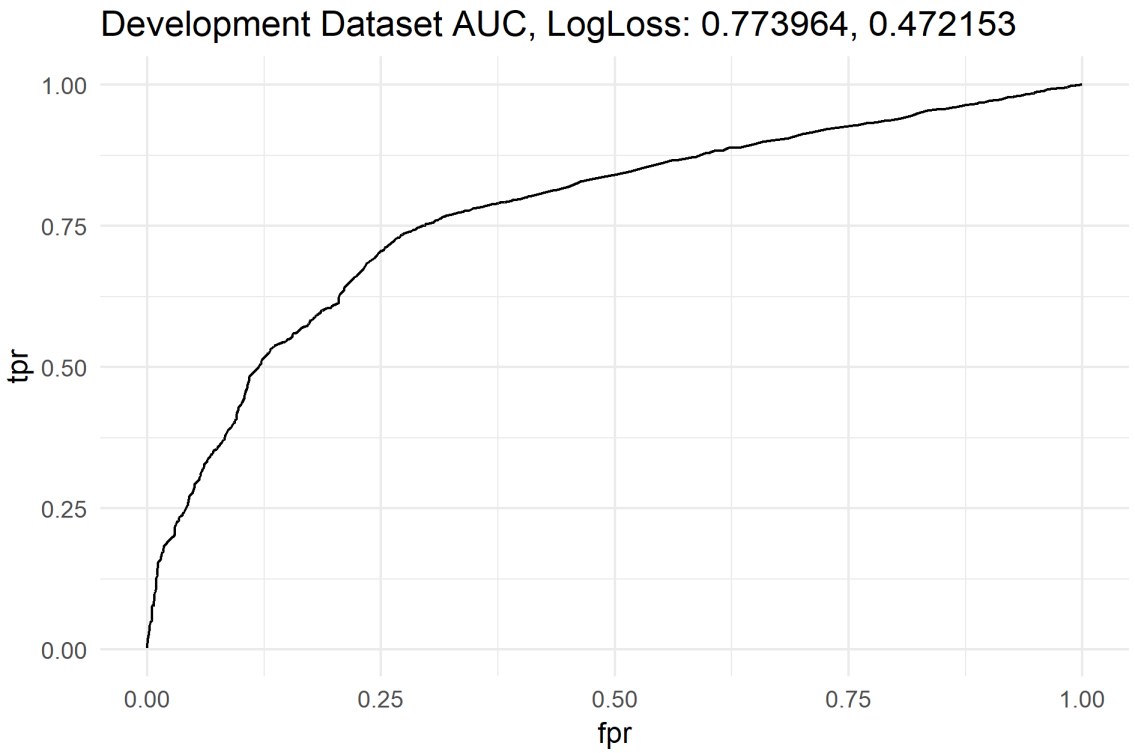


Figure 15: The ROC curve and associated AUC and log loss for the naive Bayes classifier (measured using the development set)

5.3 Logistic regression

Our progression for the classification problem roughly followed that of the overall course; consequently, we began our analysis with one of the first classifiers we learned about: logistic regression. We first begin by considering a logistic regression on simply treatment group, visit day, and total PANSS score (for the reasons described in the previous subsection). The resulting ROC curve is shown in Fig. 16; there, we have listed the development AUC and log loss which are 0.60 and 0.55 respectively. Note that this model yields a log loss of 0.61847 on the test set (the public Kaggle leaderboard) upon training it on the totality of data from studies A-D. While clearly this an improvement of the Naive Bayes test error, we sought to improve upon this error rate further by considering all of the individual PANSS scores in the model.

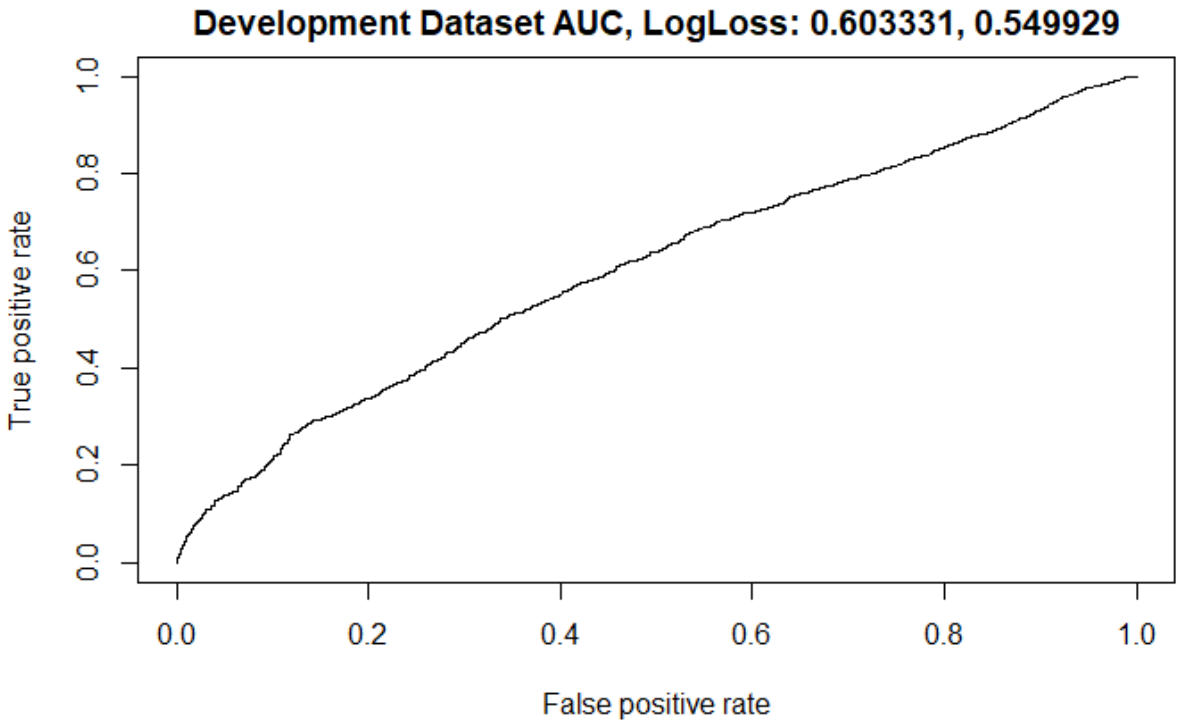


Figure 16: The ROC curve and associated AUC and log loss for logistic regression using only the total PANSS score (measured using the development set)

When considering each of the individual PANSS scores (and thus excluding the total PANSS) score, we obtain a development AUC and log loss of 0.67 and 0.52 respectively (Fig. 17. The corresponding test set log loss (the public Kaggle leaderboard) is 0.67744. This suggests potential overfitting since the seeming improvement on the development set disappears when evaluated on the test set. Although none of the development set was used in training, a small degree of bias exists in our model since the training observations and development set observations only appear from studies A-D whereas the test set considers only patients from study E. Hence there is indeed a source of bias in our model that we cannot directly overcome.

Our next step in our analysis was to consider the logistic regression model that considers all of the individual PANSS scores but to perform feature selection via lasso to ultimately decrease the variance of the model (which should improve the test set score). In Fig. 18, we perform cross-validation (10-folds cross-validation specifically) to determine the optimal value of λ, the shrinkage value in lasso. According to the one standard error rule, the best value of λ is such that only 24 predictors are considered. The predictors we exclude are P1, P2, P6, G1, G6, G7, and G11. The resulting ROC curve for logistic regression with lasso and this particular value of $\lambda = 0.00498$ is shown in Fig. 19 alongside the AUC and log loss on the dev set. While these values of 0.68 and 0.52 for the AUC and log loss are slightly larger than those seen in Fig. 17, the test set log loss was actually 0.66589 which is an improvement relative to the model that includes all of the individual PANSS scores. Hence, in this case, the increase in bias by eliminating the above listed variables from our regression model led to a sufficient decrease in variance such that the total test error rate decreased. It should be emphasized, however, that the original logistic regression model still outperforms the one that utilizes lasso by quite a large margin (test set log loss of 0.61847 vs. 0.66589). This suggests that apparently minimizing variance should be our greatest concern if we wish to improve our test set log loss any further.

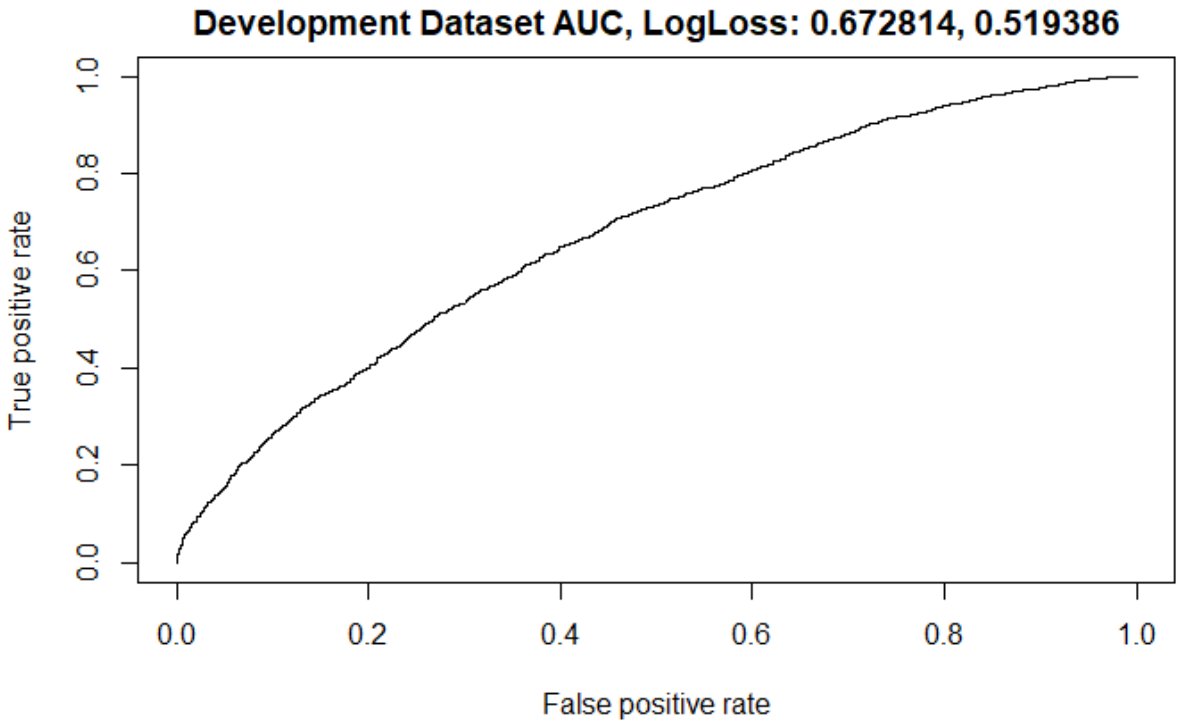


Figure 17: The ROC curve and associated AUC and log loss for logistic regression using all of the individual PANSS scores (measured using the development set)

5.4 Linear discriminant analysis

The next classifier we consider is linear discriminant analysis (LDA). Here, we only consider treatment group, visit day, and total PANSS score as our predictors since this leads to a model with minimal variance (which appeared to be the best approach for minimizing our test error rate given the findings of the last section). LDA is useful not only as a way to improve upon our best test set error, but it also serves to help us understand the nature of the underlying data. The latter comes as a result of comparing the performance of LDA and other methods and inferring what this says about the dataset given the assumptions of the various models. In Fig. 20, we display the ROC curve for LDA; the AUC and log loss on the dev set are 0.60 and 0.55. Training this model on the entire data from studies A-D yields a test set log loss of 0.61866. Hence, LDA performs almost identical to our logistic regression that only utilizes the total PANSS score. Why is this? Of course, logistic regression and LDA both yield a linear decision boundary; if the underlying Bayes decision boundary is indeed linear, then we certainly expect both methods to perform equally (quite) well. Given that LDA is built upon the assumption that the observations from the two classes are drawn from a Gaussian distribution, the equivalent performance of the two methods suggests that this is not a bad assumption for this dataset.

5.5 Quadratic discriminant analysis

The obvious classifier to consider following LDA is of course quadratic discriminant analysis (QDA). In short, we expect QDA to outperform LDA if the decision boundary is slightly nonlinear. Note that we again only consider the three predictions of treatment group, visit day, and total PANSS score. In Fig. 21, we see that the dev set AUC and log loss are 0.65 and 0.56 respectively. The test set log loss (obtained from the public Kaggle leaderboard) is 0.63167. Hence, certainly QDA does not perform poorly per se (as compared to most of the methods considered thus far), but it is slightly inferior to LDA and to our initial logistic regression. Recall that QDA differs from LDA in that each class now has its own covariance matrix - we no longer assume that they are equivalent. However, decrease in bias comes at an increase in variance. Clearly,

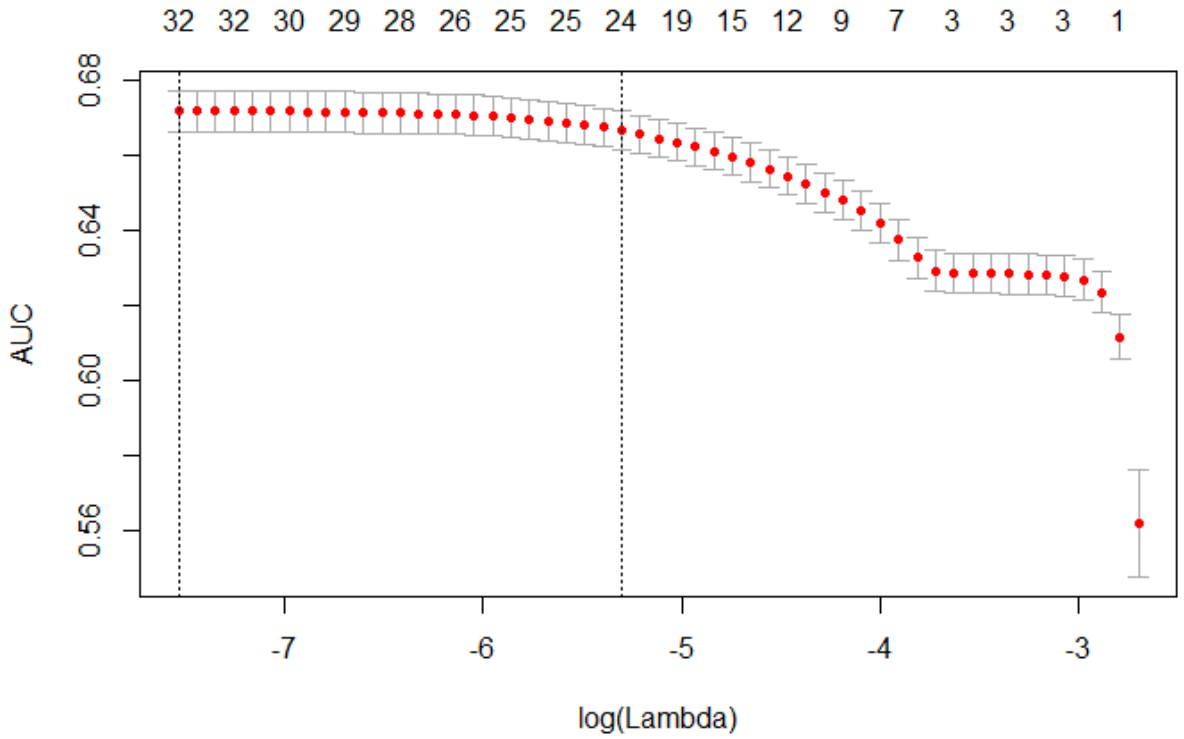


Figure 18: Determining the optimal shrinkage value for λ lasso in the context of our logistic regression on all individual PANSS scores.

this trade-off led to a net decrease in performance of our classifier, which makes sense given that the best classifiers thus far have all been those that are quite inflexible.

5.6 The gradient boosting method and random forests

The final two models we seriously considered were the gradient boosting method (GBM) and random forests. Both of these learning methods are known for being relatively good "out-of-the-box" methods that can be optimized through a variety of hyperparameters to tune. We felt that each of these methods should work well for this dataset given that they both attempt to directly tackle variance in regards to the bias-variance trade-off. GBM does so by considering an ensemble of "weak" learners that are typically short trees (usually stumps) that are only allowed to fit the residuals at a slower rate. Random forests on the other hand minimize model variance by only considering a subset of the total number of predictors at each split in a given tree of the ensemble of learners.

The performance of the gradient boosting method (XGBoost tuned using a random discrete grid search) considering the predictors of treatment group, visit day, and total PANSS score is shown in Fig. 22. Note that the dev set AUC of 0.70 and log loss value of 0.52 are quite competitive with those of the other classifiers considered in this section. Despite this, the test set log loss is found to be 0.67320, which is only superior to the logistic regression that uses all individual PANSS scores and the naive Bayes classifier. It was at this point that our team realized we may not be fully utilizing the strength of GBM by restraining it to using only three predictors. While including the individual PANSS scores previously led to a decrease in test set performance (owing to increased model variance), we thought that GBM would not be as susceptible to this same setback owing to its emphasis on minimizing variance. The ROC curve and associated AUC and log loss for GBM utilizing all individual PANSS scores is shown in Fig. 23. Indeed, we see an impressive increase on the dev set with the model obtaining an AUC of 0.79 and a log loss of 0.45. As before, however, this is not representative of the test set performance. The test set log loss was found to be 0.73258 for this model. We can gain some insight into why this is the case by considering the relative importance of the various predictors

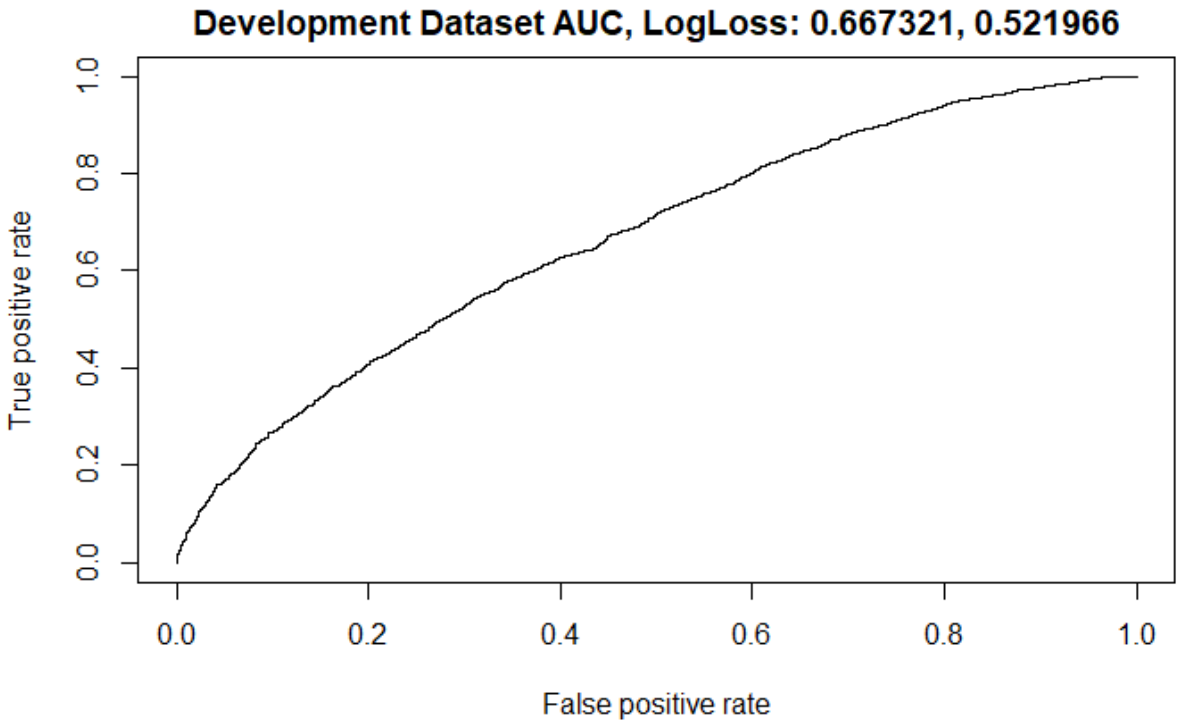


Figure 19: The ROC curve and associated AUC and log loss for logistic regression utilizing lasso (measured using the development set)

in the tuned GBM (Fig. 24). Note that the relative importance, as calculated by the `h2o.varimp_plot()` method, essentially considers how much each predictor decreases the MSE at each step and how often a predictor is used to create a split in the tree. From this plot, we see that the dominant PANSS scores are from the positive and general symptom categories. In fact, only one of the top ten variables is related to a negative symptom (G15). Yet, recall from Section 3 that patients in study E score significantly different in regards to the first principal component (which was shown to be associated largely with positive and general symptoms) from the rest of the patients. This bias has again manifested itself as a deterioration in the perform of GBM on the test set.

The final method we seriously consider is that of random forests. Note that for even a modest hypertuning time (again using a random discrete grid search), the performance on the dev set is impressive (Fig. 25). We observe the best dev set AUC and log loss yet, at values of 0.83 and 0.42 respectively. As before though, the performance on the test set is lackluster at a log loss value of 0.70370. We believe this is related to the same issues discussed above for the gradient boosting method. It appears that for these methods to perform equally well on the test set, one must think of sophisticated ways to include information from study E in the training set (perhaps using a methodology reminiscent of bootstrapping) or by manually tuning the training set to make it more closely resemble the test set (i.e. excluding all of the predictors related to the positive and general symptom categories).

5.7 Discussion

There were many competitive methods we considered for the classification portion of this project; test set log loss values ranged from 0.61847 (our best score on the Kaggle public leaderboard) to 0.73258. Overall, it appears that the least flexible methods including logistic regression and LDA (each considering only three total predictors) perform the best. We attribute this high performance to the fact that patients in study E score significantly different than those in the other studies for the positive and general PANSS categories (as was shown in Section 3). Since our training set for the classification problem does not contain any patients

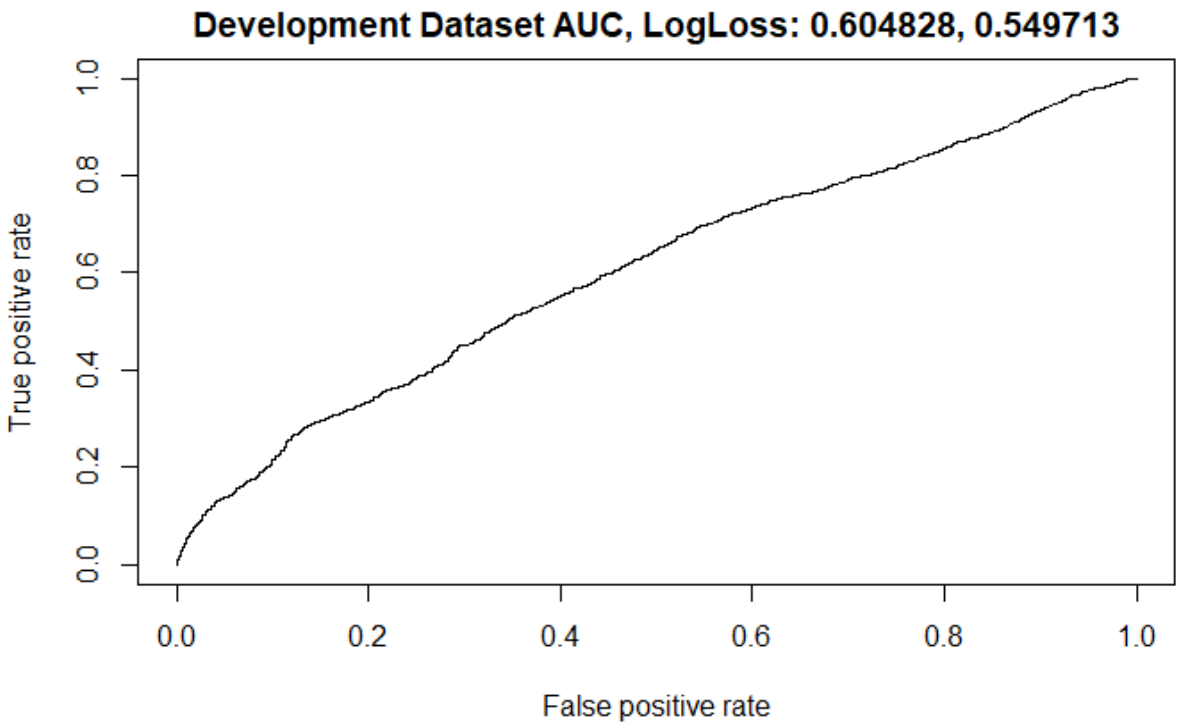


Figure 20: The ROC curve and associated AUC and log loss for linear discriminant analysis (measured using the development set)

from study E, we expect variance to be a problem when it comes time to make predictions on the test set (i.e. the totality of study E).

One other insight for why LDA and logistic regression seem to perform superior to the other methods is actually provided by an attempt to use support vector machines (SVMs) for this problem. In early exploration of using SVMs for this dataset, we visualized the observations for the two classes in the plane spanned by visit day and total PANSS score (Fig. 26). We found that overall SVMs performed quite poorly (hence why they are not further discussed in this report); viewing this plot suggests why that might be. From this plot we immediately see that the data (at least when viewing in this plane) is not at all well separated. Thus, we expect SVMs to struggle to define an appropriate hyperplane; we can somewhat see this from the figure seeing as all points marked with an "X" are support vectors (and there appears to be almost as many support vectors as there are total data points). In contrast, we learned that while logistic regression performs poorly (that is it has unstable parameter values) when classes are well separated, it has no qualms for data that is more contiguous. While the Bayes error rate will be larger in that case, logistic regression (and the closely related LDA) can still perform decently well in such scenarios.

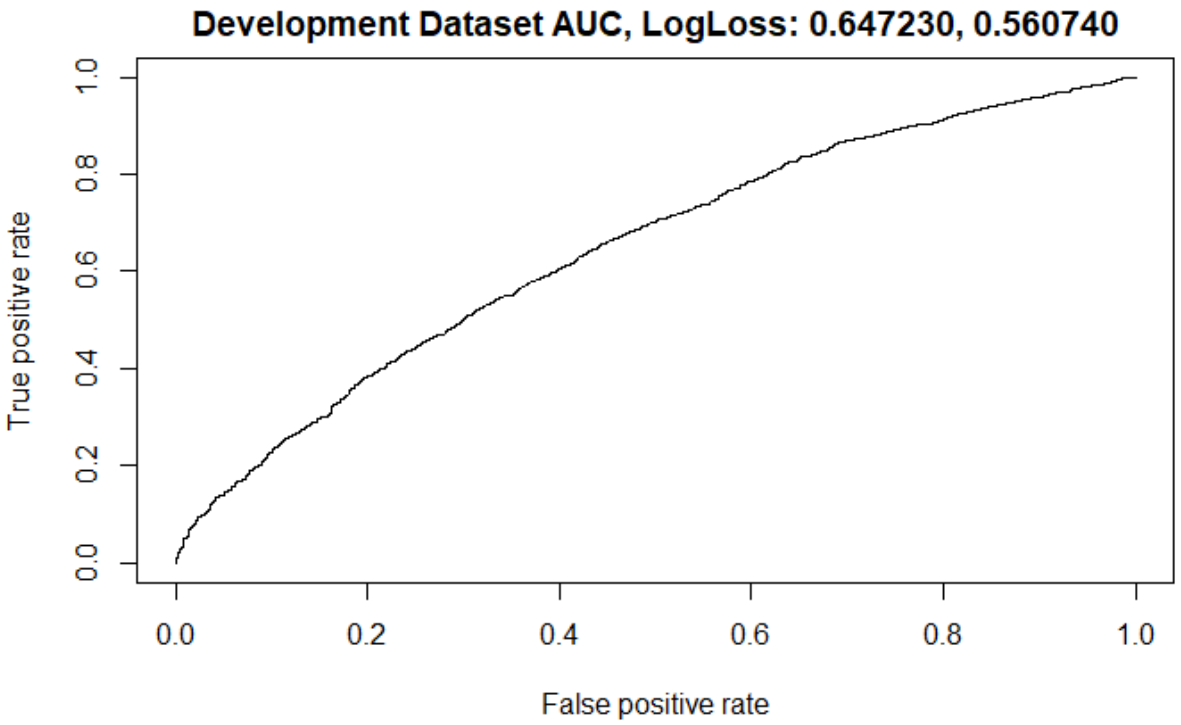


Figure 21: The ROC curve and associated AUC and log loss for quadratic discriminant analysis (measured using the development set)

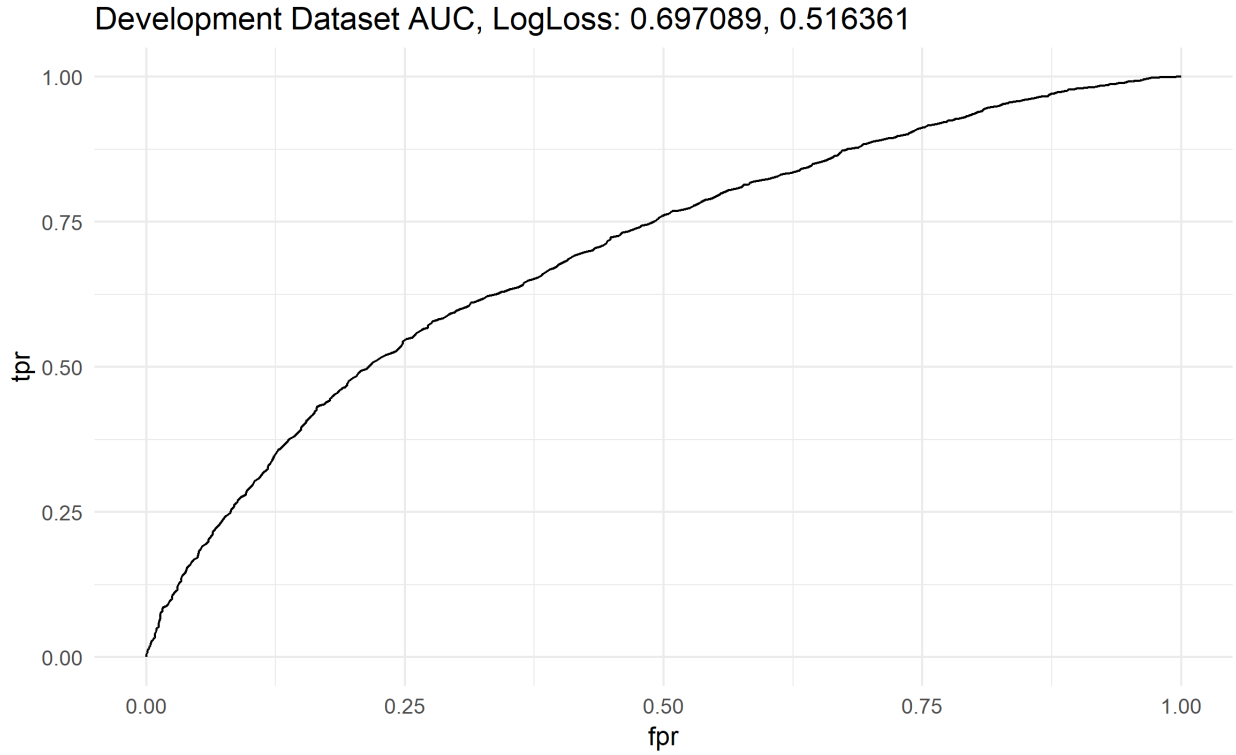


Figure 22: The ROC curve and associated AUC and log loss for the gradient boosting method using the total PANSS score (measured using the development set)

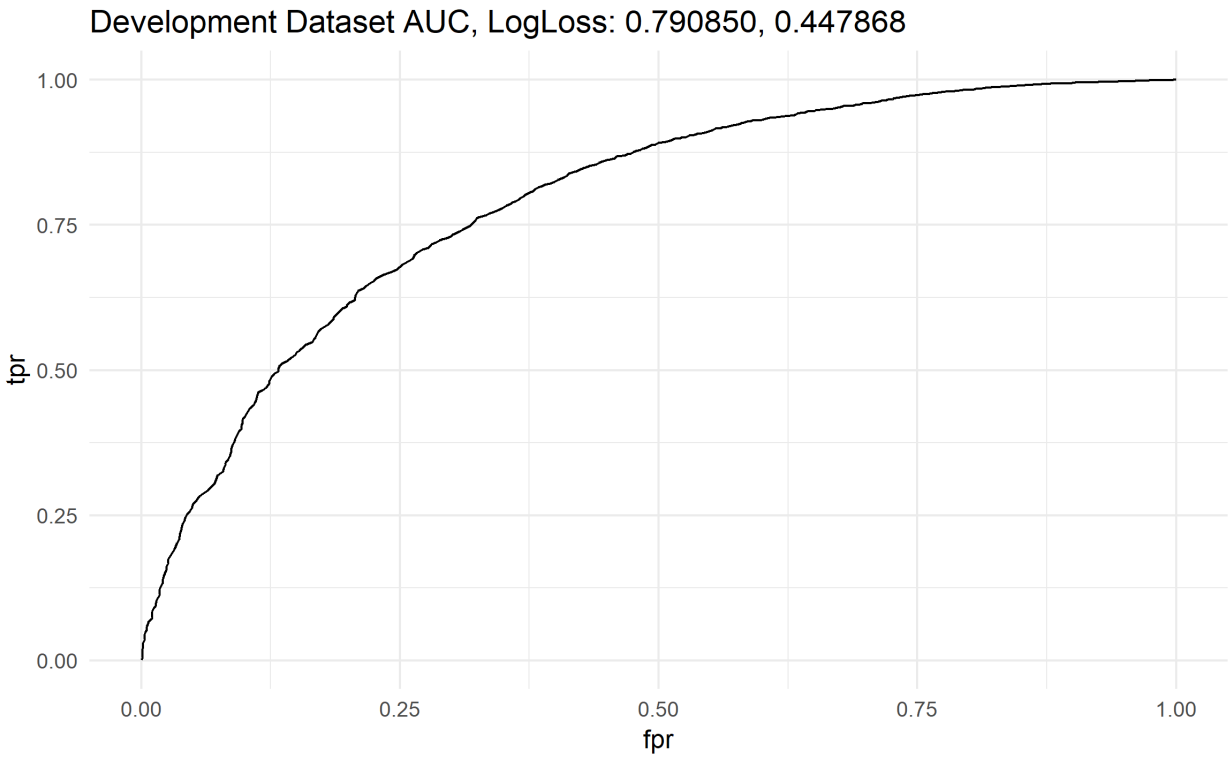


Figure 23: The ROC curve and associated AUC and log loss for the gradient boosting method using all individual PANSS scores (measured using the development set).

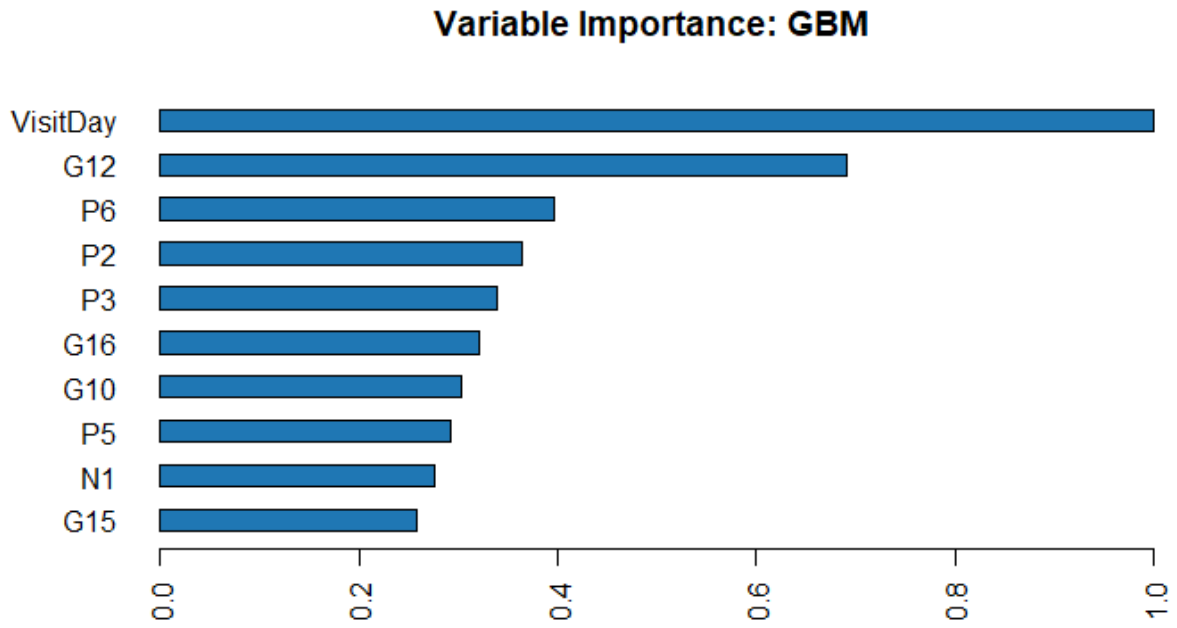


Figure 24: The relative variable importance for the gradient boosting method using the total PANSS score.

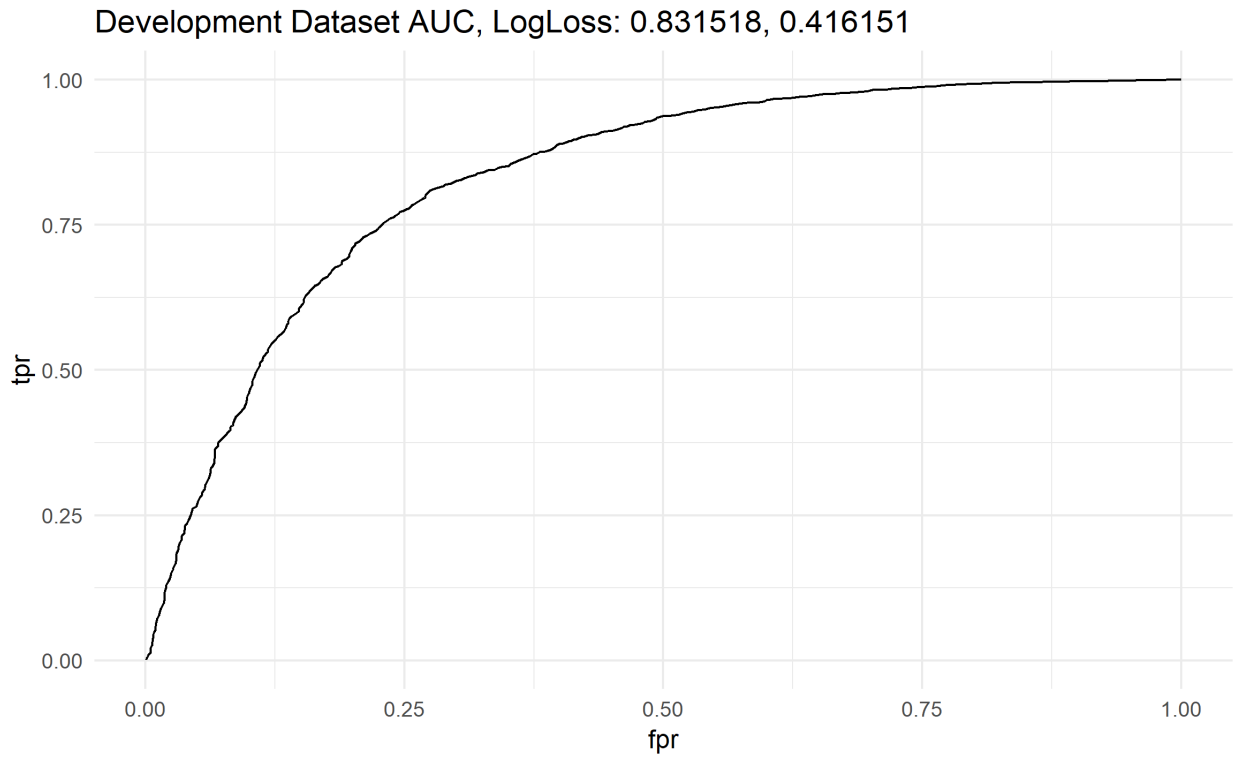


Figure 25: The ROC curve and associated AUC and log loss for the random forests method using all individual PANSS scores (measured using the development set).

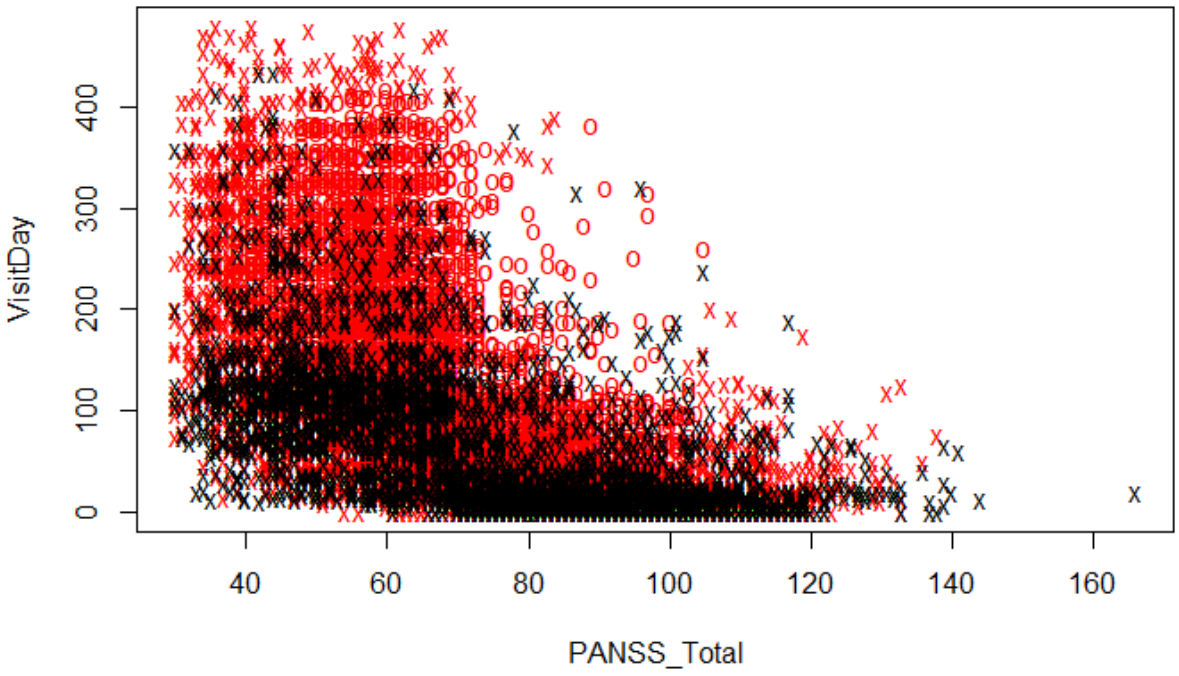


Figure 26: Observations visualized in the `VisitDay` - `PANSS_Total` plane. Color denotes the class identity (whether the audit was passed or not) and "X"s denote support vectors for a linear support vector machine.

6 Summary

To summarize the findings our our analysis, we begin with our conclusions concerning the treatment effect. Interestingly enough, we found that statistical there was not a significant difference between the treatment and control groups. This hypothesis considered the total PANSS score, total PANSS scores for a given symptom category, as well as individual scores; in all cases there was no appreciable difference in the variation of the scores with respect to the patient's visit day. Indeed, one conclusion from this first part of our study is that patient symptoms simply decrease over time irrespective of what treatment group the given patient is in.

To gain further insight into the patients we were studying, we performed unsupervised learning to sort the patients into clusters based on their initial visit scores. Our analysis suggests that the optimal number of clusters is 2. Using either K-means or PAM for our clustering algorithm, we find that the two clusters are of approximately the same size and are neatly separated along the first principal component (the boundary being located at approximately a value of 0 for this dimension). PCA reveals that this dimension is associated with the positive and general symptom scores while the second principal component is predominantly associated with the negative symptom scores. Unfortunately, overlaying the patient study groups onto the individuals plot shows that study E is clearly distinct from the other studies; patients from this study are associated with large values of the first principal component. This suggests a potential source of bias in our analysis for the remaining portions of the project seeing as our training set is void of patients from study E.

In terms of forecasting, we found that "simpler" methods that directly consider the time history of each patient significantly outperformed more advanced holistic learning methods. Specifically, we found that exponential smoothing with various values of the smoothing parameter yielded test MSEs consistently around 30; our best performance was a test MSE of 29.72322 corresponding to exponential smoothing with a smoothing parameter $\alpha = 0.9$. This prediction places our team at the time of writing at the 6th position out of 42 teams on the public Kaggle leaderboard.

Finally, for classification we found that variety of classifiers performed competitively with one another. That is, whereas the percent difference in our best and worst performing forecasting methods was over 300%, the difference for our best and worse classifier was only about 18%. Still, our best classifiers were consistently the least flexible methods, including logistic regression and LDA with only a few predictors (treatment group, visit day, and total PANSS score). We believe these methods perform the best owing to the test set bias mentioned in Section 3; using relatively inflexible models minimizes the model variance that we expect to be an issue since none of our training data includes patients from study E. Our best performing classifier was logistic regression, achieving a test set log loss of 0.61847, placing our team at the 7th position on the Kaggle leaderboard at the time of writing.