**Capstone 1: Consolidated Report**

***Predicting Psychological Distress***

Jared Hammernik

**Proposal**

With increasing awareness of mental health issues, I thought it would be interesting to study what factors may contribute to it, and see if it can be predicted.

This may have relevance to school administrators and workplaces who may have an interest in keeping their students or employees productive and stress free, but it also may have relevance to each of us individually who wish for ourselves and our families to live a stress free life. Perhaps this kind of analysis can help us understand what are the main contributors, and perhaps by being able to predict it we can understand who may be at a higher risk, allowing for appropriate intervention to take place.

I propose using data from the adult 2017 California Health Interview Study (CHIS), which is a phone survey conducted annually by the UCLA Center for Health Policy Research. The survey was voted one of the top 50 data sources in the country by the Health Data Consortium. The survey is very large scale and uses various statistical techniques in an attempt to obtain an accurate sample, and they also make all their techniques publically available on their website. All in all the survey collected information on over 21,000 adults, and had over 400 questions involved. The questions asked everything from the participants race, eating habits, and income to their type of insurance, household status, and whether they voted in the 2016 election.

The questionnaire asks a series of questions about the participants recent emotional state, including nervousness, hopelessness, restlessness, and depression, and combines them to form a score based on Kesslers K6 scale from 0-24. A Kessler score of over 13 in the survey is correlated with likely experiencing psychological distress. The questionnaire asks the same series of questions in two forms: the first is for the last month, and the second for the worst month of the past year. A K6 score above 13 in the last month is made as a yes/no column “Likely has had psychological distress in the past month” and a score above 13 for their worst month is made as a yes/no column “Likely has had psychological distress in the past year”.

I propose using a variety of statistical and machine learning approaches to learn what variables are the most prominent predictors of psychological distress, and see how well it can be predicted using binary classification. While less than 4% of respondents were likely to have had psychological distress in the past month, roughly 8.5% were likely to have distress in the past year. Because the smaller class imbalance, I propose predicting on those who have most likely experienced distress in the last year. I propose using a variety of modeling techniques such as random forest, gradient boosting, logistic regression, and K-Nearest Neighbors. I also hope to test the effectiveness of an ensemble of all these techniques.

For more information about the survey and it’s methodologies, one can visit the website here: <https://healthpolicy.ucla.edu/chis/Pages/default.aspx>

**Data Wrangling**

Data wrangling for this project involved many steps. First I had to download the data from <http://healthpolicy.ucla.edu/chis/Pages/default.aspx> after registering and agreeing to the terms and conditions. The data came in multiple files. Of interest to me were ADULT\_LABEL.SAS, which included a list of column abbreviations and the corresponding long form name of the column, ADULT\_FORMAT.SAS, which included a list of column abbreviations and the corresponding column type, and adult.sas7bdat, which included the data.

After exploring the label file, I saw that it was a text file with “=” separating labels and values, so I used pd.read\_csv to read the file with a “=” separator. To read the data file, I used pd.read\_sas. After exploring the data, I saw that there were 21,153 rows and 483 columns, so I knew that I had to do some form of dimension reduction. The first thing I did is I found any columns (survey questions) that were not answered by all participants and I removed them. These were questions that were only asked to a participant if they responded a certain way to another question. However, this only brought column number down to 227.

I then decided it was best to go through all of these columns and manually select columns of interest. While doing this, I realized that many columns were more or less duplicates and contained the exact same information in a different format, so I went through and did my best to remove these columns. In the end I manually selected 78 columns, and put a list of the columns in a separate file called select\_cols.csv. I also noticed that the formats file was not very informative since every unique categorical question had a unique format label and did not mention anything about whether the data was continuous, categorical or ordinal. Therefore I also added the select\_cols.csv file my own categorization of column type.

All continuous and categorical variables I left as is. For all columns that were ordinal and categorical, I reviewed each one individually and determined if I needed to implement any feature engineering. For the most part I saw that they were correctly ordered and numbered, so I could leave them and treat them as continuous variables. I considered imputing some values and replacing each number category (i.e. if one income category were $50,000 - $70,000, I could replace the number assigned to this category with the min, max, or middle of the range). However since many of the models I planned on using initially were tree based classification models, and therefore wouldn’t make a difference, I decided the slight information loss was not worth the effort, as determining the replacement values would have to be done for many columns and column categories, and replaced one by one. Therefore, I ended up leaving most as continuous variables. However there were a couple columns that I determined I needed to make special column changes to. For example, in the question ‘HOW OFTEN FIND FRESH FRUIT/VEG IN NEIGHB (PUF RECODE)' I noticed it had one answer that was ‘doesn’t shop for fruits and vegetables’ which was answered less than 1% of the time. Therefore for this question I decided to impute all of these responses with the mode, i.e. the most commonly answered category.

From observing the data I also noticed that there were 3 proxy interviews, which I decided just to delete. After reading the file into a pandas dataframe, and deleting the proxies, I read a separate data frame from the select\_cols.csv file. From this dataframe, I selected the column names I wanted from the larger data file. Because these select columns were that were answered by everyone, there was no missing data. I then read a third dataframe from the labels file, which I used to rename all the columns into the longer format for better understanding of what each column meant.

For all my self-labeled categorical values, I knew that I had to one-hot encode them. I did this by selecting all of the categorical columns from my selected column dataframe, placing them into a new dataframe, one-hot-encoding them, dropping the previous categorical columns from the selected column dataframe, then joining the one-hot-encoded dataframe back to the original. With the one-hot-encoded variables added, I had a dataframe with 121 columns. I also decided to rename the one-hot-encoded variables into the long form that made more readable sense before joining. One problem I ran into is that I noticed I was working with a previous version of sci-kit learn whose OneHotEncoder did not have the option to drop one column from each question as not to duplicate data. I ran into many issues trying to update sci-kit learn in my Anaconda environment, and therefore I decided to leave the columns as is, although I certainly would have done this if that option was available.

Now that I had a nice clean dataframe, I thought it would be best to try recursive feature elimination using RFECV to reduce the number of dimensions further. However after trying this for the random forest, it took an extremely long time to run, only reduced the number of columns down to 71, and did not improve model performance. Rather than run RFECV for all of the additional models I planned to try, I decided not to use RFECV.

Ultimately since I realized that I would also be running linear models like Logistic Regression, I knew I had to remove collinear variables anyway, which I figured could also serve as a good way to reduce the number of columns. Therefore, I wrote a function to remove collinear variables from the dataframe. This function identified pairs of columns that had a high pearson correlation coefficient, and then removes the one from the pair that has the lowest correlation with the target variable. I ran this function on my data and removed all the one of all pairs of correlated variables that had an absolute value of a Pearson correlation score of over 0.5. This brought the number of columns down to 92. At this point I was ready to begin my exploratory data analysis.

**Statistical Data Analysis**

One of my main goals with this project is to understand which variables contribute the most to psychological distress and how. As part of my exploratory data analysis, I used several statistical techniques to identify these relationships. Because I had a large number of variables, I only specifically looked at a few variables of interest.

For continuous variables, I developed a function to plot the histogram density of the distribution of those who were likely to have psychological distress, alongside the distribution of those who were not. Since the classes were very imbalanced, it would be difficult to spot differences in the histograms, so to make it easier to compare the two I made sure both plots were density plots. In addition, I also performed an independent T-test between the two groups, to see how likely it could be said the mean of the two distributions come from the sampling distribution. In other words, to see if there is a statistically significant difference between the two. After running this test on all continuous variables, and assuming a 95% confidence, it can be said that for weight weight, with a p value of 0.16, we cannot reject the null hypothesis that that they may come from the same distribution (as a side note, I found it interesting that weight was however one of the most important variables in the Random Forest model, which I suspect has to do with the fact that it appears that that distress was more common at the extreme ends of the weight distribution, which would not necessarily affect the mean, but could be accounted for tree based models). For other variables such as age and poverty threshold level (analogous to income), with p values of 5.95e-159 and 3.05e-79 respectively, we can certainly reject the null hypothesis that each group comes from the same distribution. Therefore we can be fairly certain that these factors have a strong effect on psychological distress and that the variations in their distributions are not due to random chance.

For categorical variables of interest, I first wrote a function to plot the proportions, in percentage, of those in each category that were likely to be psychologically distressed. In doing so, we get a decent idea of how the categorical variable effects psychological distress. In addition, I wrote a function to perform a chi-squared test and return the resulting p-value to determine if there is a statistically significant difference between the observed and expected frequencies in each category. For gender, 9.45% of females were likely to be psychologically distressed, compared to 7.37% of males. The chi-squared p-value of this was 8.87e-08, indicating a significant relationship. For race, for simplicity’s sake I only looked at self-reported white vs self-reported non-white. 8.92% of non-white individuals were likely to be psychologically distressed, compared to 8.43% of whites. The chi-squared p-value of this was 0.298, which at a 95% confidence level, does not indicate a statistically significant relationship.

**In-Depth Analysis**

Here we will explore the use of various machine learning models to predict whether someone was likely to have experienced psychological distress in the past year. I chose a number of different models including Random Forest, Logistic Regression, k-Nearest Neighbors, and Gradient Boosting. In addition, I decided to do an ensemble voting classifier among these 4. Because I knew I was going to use a linear model such as Logistic Regression, I needed to remove collinear variables. I did this in the data wrangling step by writing a function to determine collinear variables, and then remove the one with the lowest correlation to the target variable. I performed this function on my data which removed a column for all pairs of columns with greater than 0.5 for the absolute value of the pearson correlation coefficient between them. This step also served as a dimension reduction technique.

With this new dataset, I then used sklearn’s train\_test\_split to divide my dataset into training set (80%) and test set (20%). I set the target variable to be 'LIKELY HAS HAD PSYCHOLOGICAL DISTRESS IN THE LAST YEAR' which was a binary classification (1s for yes, 2s for no). Since I was going to be using non tree-based algorithms such as Logistic Regression and KNN, I knew I also needed to standardize the variables making the mean 0 and standard deviation 1. To do this I used sklearns StandardScaler. I used StandardScaler’s fit\_transform method on the variables of the training set and used this scaler to transform the variables of the test set as well.

Next I knew I needed to determine a metric in which to optimize my models and determine success. Because my data was imbalanced, with roughly 8.5% likely having experienced distress, I decided to look at precision and recall, specifically the area under the curve of the precision recall curve. Since sklearn’s GridSearchCV does not have this exact metric as a scorer, I had to write my own precision recall AUC scoring function to pass in.

*Random Forest*

With this this scoring function in place, I decided to start my first model: Random Forest. I first set out to optimize the parameters of this using GridSearchCV. I decided to test various numbers of standard random forest parameters including the numbers of trees and the number of variables to choose from at each split, however I also decided to test both gini and entropy as splitting criteria, and also various methods built in to the function for balancing the data considering my data was very unbalanced. Optimizing for Precision-Recall AUC, the best parameters were to have no balancing techniques applied, entropy as splitting criteria, 500 trees, and 10 max features. Using these optimized parameters, I made a random forest model and tested it on the test set. As expected when classifying an imbalanced dataset with a default threshold of 0.5, the accuracy was very high (as it predicts most as majority class). In order to get a better look at the quality of the model, I looked at both the ROC curve and the Precision-Recall curve, including the ROC AUC and the statistic I am optimizing for: PR AUC. Observing the ROC curve, the model appears to do quite well with an ROC AUC of **0.83**, however after looking at the Precision Recall curve we see the effect of the imbalanced data, as the PR AUC was only **0.34**

*Logistic Regression*

Next I decided to try Logistic Regression. I used GridSearchCV to optimize for the value of C, between L1 and L2 regularization, as well as whether or not to use the built in balancing feature since I am using imbalanced data. Optimizing for PR AUC, the optimal parameters were a C of 0.01, L1 (“Lasso”) regularization, and using the built in balancing feature. After building a Logistic Regression model using these parameters and running it on the scaled test set, I once again plotted the ROC and precision recall curve. From these curves we observe an ROC AUC of **0.84**, which is slightly better than the random forest, and a PR AUC of **0.34**, which was the same.

*K-Nearest Neighbors*

The third model I ran was KNN. Once again using GridSearchCV, I optimized only for the number of neighbors, K. Optimizing for PR AUC, the best value of K was 500. Building a KNN model using this parameter and running it on the scaled test set, I once again plotted the ROC and precision recall curves. The ROC AUC of this model was **0.80**, and the PR AUC was **0.32**, both of which were lower than both random forest and logistic regression. I expected this to be worse than the others, but I was surprised that it was still somewhat close to the other models.

*Gradient Boosting*

I then created a Gradient Boosting Model. I optimized for a variety of parameters including the learning rate, the number of trees, the number of variables to consider at each split, as well as the loss function. Optimizing for PR AUC, the best number of trees was 100, the max features was 10, the learning rate was 0.1, and the loss function was deviance. Building a model with these optimized parameters and running on the test set, I again plotted the ROC and precision recall curve. The ROC AUC of this model was **0.84** and the PR AUC was **0.34**. These were the exact same values achieved using Logistic Regression.

*Voting Classifier*

Finally, I wanted to take full advantage of the wisdom of these models and build an ensemble voting classifier that looked at all 4. Using VotingClassifier and using ‘soft’ voting based on the average of the probabilities, I built a model and ran it on the scaled test set. After plotting the ROC and precision recall curve, I found the ROC AUC was **0.84** and the PR AUC was **0.35**. While an ROC AUC of 0.84 was observed in the Logistic Regression and Gradient Boosting models, our metric of interest PR AUC reaches its highest point yet at 0.35.