Benjamin Inbar

Final paper: DATA 71200 Summer 2021

**Dataset and project aim**

The dataset for this project consists of various magnetic resonance imaging (MRI) metrics recorded from multiple scans of 150 patients with and without Alzheimer’s disease. It is a longitudinal data set, with each patient completing a minimum of 2 scans at least 1 year apart. Prior to their scans, each patient was given a clinical dementia rating (“CDR”), and a threshold was set on this score such that the patient was classified as either “demented” (with Alzheimer’s) or “nondemented” (normal, without Alzheimer’s). The goal of this project was to predict this clinical classification of a patient given their MRI metrics. I chose this set given my own background and current work in the neuroscience field, the ever-growing interest and utilization of machine learning in medical imaging for assisting imaging technicians and doctors with neurological diagnoses, and the usability of the dataset itself, which contains only 13 dimensions and is fairly clean. I found the data via the open source data science host site Kaggle (<https://www.kaggle.com/jboysen/mri-and-alzheimers>), which had in turn sourced it from the Open Access Series Imaging Studies (<https://www.oasis-brains.org/#access>), a public project by the Washington University Alzheimer’s Disease Research Center and its partners in an effort to make neuroscience data freely available “to facilitate future discoveries in basic and clinical neuroscience.”

**Cleaning the data**

Although these data were already fairly clean, for the purposes of my project I further streamlined it. First, I dropped rows that contained “NaN” null values in the “SES” and “MMSE” features, bringing the total number of instances from 373 to 354, a drop of ~5% of the data. This was not ideal given the relatively few number of instances in this data set to begin with, but it would have been inherently difficult to impute a value particularly for “SES” (Socioeconomic Score). For the “MMSE,” or Mini-Mental State Exam, a clinical diagnostic for cognitive functioning, it would have theoretically been possible to impute the mean of similar genders, ages, and sexes, and may be worth considering for future analyses to reduce the number of dropped instances.

Next, I dropped some features which held no predictive value, such as “Subject ID” and “MRI ID.” Another, “Hand” (left vs. right handedness), showed every instance in the set classified as right-handed, so this was not really a variable but a constant.[[1]](#footnote-1) I also dropped 2 features related to the longitudinal nature of the study: MR delay, defined as was the time between MRI scans for a given patient in days, and visit number (i.e. 1, 2, 3, etc). Notably, in doing so I essentially eliminated the longitudinal paradigm, by treating each imaging session as a unique instance and erasing the granularity of within-patient analysis. Since some patients had more imaging sessions or instances than others (e.g. 4 instead of 2), this meant that some of the 150 patients were automatically weighted more heavily than others in the 373 instances in the set, leading to some biasing of the effects of sex, brain volume, education, etc. One possible solution for this in future analyses could be to normalize all patient visits such that each was represented by a similar number of imaging sessions by 1) calculating the mean of each feature for patients with only 2 sessions and imputing new sessions or instances for those patients (although as mentioned this would be challenging for features such as “SES”) or 2) eliminating the excess sessions for patients that had more than 2 sessions. Aside from this, the other transformations were straightforward: 1) replacing “Converted” as a diagnosis for patients that had initially clinically been identified as “Nondemented” but became “Demented” after the 2nd or 3rd imaging session, 2) label encoding the “M” and “F” sex attributes to “1” and “0” and the outcome variable “Demented” and “Nondemented” to “0” and “1,” respectively (in retrospect I could have reversed these encodings for better intuitive clarity).

**Visualizing the data**

Visualizing the cleaned training data using the .hist() function to show frequency distribution histograms for each feature illustrated that the continuous variables were mostly normally distributed. Initially, I interpreted this to mean that scaling the data might not be necessary (though later it became as much). The exceptions to the normal appearing distributions were the “MMSE” and “CDR” features, which skewed toward higher and lower values/scores, respectively. This was interesting as it suggested that patients could have a relatively high cognitive functioning score and a low dementia rating, and still be classified as demented.

Running a scatter matrix of all the features revealed some of their correlations. Most notably, the “eTIV,” the estimated total intracranial volume, and “ASF,” or atlas scaling factor, which appeared in an almost straight line across the chart and thus highly correlated with one another. The ASF is derived from and calculated based on body anatomy and proportional to eTIV, so this makes sense. Also correlated were “nWBV,” or normalized whole brain volume, and age, as well as eTIV and nWBV. It is well studied that whole brain volume (and by extension the estimated total intracranial volume) decrease with age, so this also made sense.

These visual observations were borne out in the Pearson correlation matrix (displayed as a seaborn heatmap). Other notable correlations that were not obvious in the histograms since they were categorical variables, but were salient in the correlation matrix were: MMSE with CDR, which showed a relatively strong (-0.72) negative correlation, and SES and EDUC, or years of education, also showing a strong (-0.73) negative correlation. From clinical and socioeconomic perspectives these intuitively made sense.

Finally, running transformations on two features (squaring, cubing, log, exponentials), showed changes in magnitude and scale (e.g. reduction of axis ranges, or reduction of variance, or both) that were possible for downstream application to each algorithm.

**Supervised learning: logistic regression and decision trees**

For the supervised learning tasks, since the goal was to predict, or indeed classify, an instance of dementia in a binary fashion (i.e. 0 or 1, demented vs. nondemented), I first chose a logistic regression algorithm. Logistic regression was particularly suited to the binary nature of this classification problem, since it assigns a binary class in absolute terms, given the logit of the probability of an instance, and sets a clear threshold between 0 and 1 at 50%. When first instantiating and running the logistic regression algorithm on my training data (using 5-fold cross validation), I got a “convergence error/warning (for lbfgs solver),” though it still ran and produced a result, if lackluster one. Some investigating revealed that this had to do with the number of iterations, or attempts, that the algorithm tried in converging to the best possible result, and that it could not do so under the default number of iterations (100). This was due to trouble optimizing the cost/loss function, the confidence scoring, and some confusing mathematics (i.e. the Hessian) that were not optimal because of the high variability of my data. Initially, I resolved the issue by increasing the number of iterations to 1000. This worked, as all but one of the 5 cross validation scores were >90%. But there are caveats: 1) it worked in part because my dataset is small. For larger datasets this would increase running time/cost significantly. 2) It increases the chances of overfitting. Therefore, ultimately, what was needed was to scale the data so that the algorithm would have less difficulty reaching an optimal output with greater confidence. Though there are several scaling options in Scikit-Learn (MinMaxScaler, MaxAbsScaler, etc) I used StandardScaler, such that for each feature, the mean was 0 and standard deviation remained constant i.e. created Gaussian data. Doing so resolved the convergence warning (even at 100 iterations) and significantly increased the accuracy scoring on both the training and testing data. The results of the confusion matrix were likewise encouraging, showing 100% precision for the demented class, and 93% for nondemented. This meant that in the test data, 100% of all predicted demented cases were in fact demented, but only 93% of all predicted nondemented cases were nondemented. The recall for demented was 91% and 100% for nondemented, meaning that it only detected 91% of the demented cases (even though it correctly predicted them 100% of the time), and detected 100% of the nondemented cases (but only correctly predicted them 93% of the time). Since both the precision and recall were relatively high, the F1 score(s) was as well, averaging 95.5% between the two classes. From a medical perspective, this is very good, especially with the oversight of a doctor. Still, I ran a grid search with various regularization coefficients ‘C.’ The best result was a C of 1, which is the default for logistic regression in Scikit-Learn, so this hyperparameter was not modified.

I next tried implementing a decision tree classifier, which is suited to categorical and unscaled data such as that in this set. Decision trees operate based on node-leaf splits based on “purity” of number of class instances, so there was no need for preprocessing or feature scaling to optimize or fit an equation. The instantiation and execution of this algorithm was much more straightforward than the logistic regression. Although the initial 5-fold cross validation scores were slightly lower on average (~88%) than those for the logistic regression, the training and test accuracies, the precision, recall, and F1 scores were in the ~95% range, indicating good performance. In fact, the F1 scores for each class average ~94.5%, just 1% below that of the logistic regression algorithm. However, after running a simple grid search and re-instantiating the classifier with the best parameters (‘criterion’: entropy, ‘max\_depth’: 5, ‘max\_features’: 8), the scores increased to exactly match those of the logistic regression with scaled data. This illustrates the power of the decision tree classifier on unscaled datasets that have high variance or categorical predictors. Moreover, by running a grid search and discovering that the optimal number of max\_features is 8, it is possible to understand more about the features which explain most of the variance in the data. Given the success of both these algorithms, it does also beg the question of whether it would be possible to improve upon the predictive power using another algorithm at all.

**PCA**

Since principal component analysis works best if features are centered around the origin, I ran PCA on my scaled (using StandardScaler) training data. Using Scikit-Learn’s built-in function “n\_components = x” where x is a float between 0 and 1 corresponding to the variance explained by those components, I was able to input 0.95 and perform a PCA returning the number of components explaining 95% of the variance in my data, which was 7 components. Since both of my algorithms from project 2 performed identically well after preprocessing and tuning, I ran each again with the 95% PCA transformed data. The scoring metrics for the Logistic Regression remained unchanged, suggesting that reducing the number of components to 7 had no effect and that the original scaling of the data was sufficient to obtain an optimal result for this algorithm. This makes sense, since scaling essentially reduces the variance of each feature already.

For the Decision Tree Classifier, the scoring metrics were actually lower with the PCA transformed data than on the original scaled data itself. Note that here I excluded the parameter “max\_features=8,” since this threw up an error because 8 was greater than the number of dimensions of the PCA transformed dataset. The reason for the reduced performance in the decision tree classifier using the PCA transformed data is not entirely clear to me. Barring anything related to the scaling of the data, or in the reduction of features that assisted the classifier, my guess has to do with the way PCA transforms the data and how decision trees work. Since decision trees are very good at classifying data in non-linear spaces, and PCA transforms the data into a linear space or “hyperplane,” this could be inconsistent with the way the features are utilized by the classifier, thus lowering its performance.

**Clustering**

When running KMeans on my scaled (but non PCA transformed) training data, the first interesting observation was the elbow plot. The downward slope of the plot appeared to be smoother across various cluster number iteration than that for my unscaled data (unscaled plot not shown in code). This might reflect the normalizing effect of the scaling of the training data: more normalized inertia across various numbers of KMeans clusters suggests similar variances of all features i.e. with data points that are (relatively) evenly spread within the feature space in which KMeans operates. A cluster number of 4 appeared to be approximately optimal, so I ran this, but ultimately found that adjusting the cluster number to 3 gave the best adjusted Rand and Silhouette scores (0.14 and 0.24, respectively). Unfortunately, those scores were still quite low, suggesting poor performance of the algorithm, and reinforcing what was first glimpsed in the elbow plot: that the clusters generated did not produce useful partitions of the data. One explanation for this could be the high number of categorical features in my dataset. Since these are discrete features, they do not lend themselves to Euclidean distance measurements utilized not just by KMeans, but the Agglomerative and DBSCAN clustering algorithms as well, which returned similarly low scoring metrics.

Using PCA as a preprocessing step only negligibly improved the KMeans scores, from 0.14 to 0.15 for the Rand and from 0.24 to 0.25 for the Silhouette score. For the Agglomerative clustering, I utilized the same number of clusters since that gave the best results. Still, the initial Rand score, at 0.21, actually decreased after PCA processing, to 0.09 (Silhouette increased from 0.22 to 0.23). For DBSCAN, I adjusted the “eps” and “min\_samples” parameters until I found the best possible scores, 0.14 for Rand and Silhouette. It is worth noting that the “min\_samples” selected was 2, which is too low, but higher “min\_samples” values returned 0 or negative Rand and Silhouette coefficients. PCA processing of the data using DBSCAN increased the Rand from 0.14 to 0.15, but the Silhouette score decreased, from 0.14 to 0.11. All of these low metrics were reflected in the visualizations, which for each feature pair that I plotted, showed little meaningful clustering (except for two of the PCA components in the KMeans). Overall, the PCA preprocessing had negligible or even slightly negative effects on what were already very weak performances of each clustering algorithm.

**Summary**

The pipeline of data selection, cleaning, visualization, and processing for learning algorithms is a learning process for the analyst as much as it is for the machine. Both the power and potential pitfalls of various algorithms and tests become clear quickly when applying them to what was in this case a clean but low-dimensional dataset. Both the supervised learning algorithms, after processing the data (scaling for logistic regression, and best grid search parameters for decision tree classifier) performed extremely well. So well, in fact, that further processing using PCA had no effect on one and actually reduced the scoring metrics of the other. Unsupervised techniques such as clustering performed very poorly, owing perhaps to the categorical nature of several of the features as well as the dataset’s small size. In retrospect, selecting a larger dataset with more dimensions and “messier” data would have been beneficial in this regard, providing more insight into unsupervised techniques.

1. Moreover, as far as I am aware there is little scientific consensus on whether handedness contributes to disease in general (in part due to problems with adequate statistical representation of each group). [↑](#footnote-ref-1)