Using Principal Components Analysis to Examine Factors Predicting the Transition from Pre-diabetes to Diabetes

Chris Haddad

Jeff Coady

Sanjay Roberts

CSCI 4441

Winter, 2019

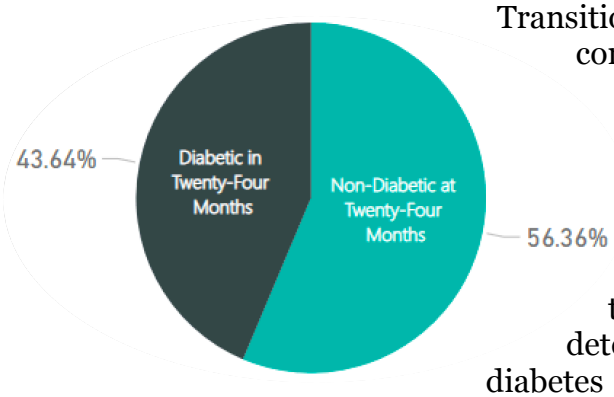
An essential aspect of prediction model training is parameter tuning. This step can be time consuming when performed on the full dataset. One strategy to reduce parameter tuning execution time is to employ principal component analysis (PCA). In the following pages we will explore PCA as applied to dimensionality reduction for the purpose of reducing model training execution time using the Allscripts' Pre-Diabetic to Diabetic Transition dataset. We will first explore our hypothesis, then the dataset, how we can use PCA on our dataset to test our hypothesis, PCA theory, and the results of our experiment.

Hypothesis

Employing Principal Component Analysis will reduce model training time while maintaining prediction accuracy.

DataSet

To test our hypothesis we will be using Allscripts' Pre-Diabetic to Diabetic Transition dataset. This dataset is comprised of around 150,000 pre-diabetic patients of which around 65,000 of those patients transition to diabetes mellitus type II within twenty-four months. These patients were categorized as pre-diabetic by their Hemoglobin A1c score and determined to have transitioned to diabetes by having an International Classification of Disease (ICD) code indicating diabetes.

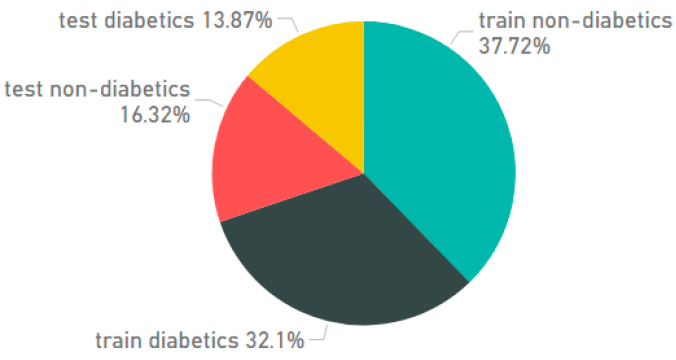


Feature Columns
Abnormal BP
Age
BMI
Diastolic
Ethnicity
Family History
Gender
Hemoglobin A1c
Race
Systolic

The features of the dataset can be seen in the table to the right, 'Feature Columns.' A custom discretization approach was used on continuous columns while discrete columns were one-hot encoded.

Experiment

In order to test the efficacy of PCA on execution time we created an experiment in RSudio using R language with Random Forest Ensemble Learning method as a classifier predictor. The classification in the aforementioned dataset is a binary class of progressed to diabetes (1) or did not progress to diabetes (0) by ICD.



We split the dataset into a training-set and a test-set with a seventy to thirty percent split respectively as shown in the graph to the right. We first trained the model without using PCA, recording the execution time and prediction test set accuracy. We then retrained the model, this time using the principal components (PC) that account for 95% of the variance, recording the execution time and prediction accuracy. We then ran the experiment again with fourteen PCs, then thirteen, and so on, each time recording the accuracy and execution time.

If our hypothesis is correct, we would expect to see a reduction in execution time as we iterate from the full dataset through the PCs in descending order, while accuracy is maintained.

Principal Component Analysis Theory

PCA is a dimensionality reduction method that takes a large number of correlated predictor variables and from them creates a new set of orthogonal and uncorrelated components to be used in place of the original variables. Ideally, these extracted components will be a much smaller set than the original set of variables but will account for almost as much variance, thereby minimizing information loss. This is useful in reducing redundancy among correlated variables, leaving more degrees of freedom to increase the likelihood of detecting true effects. Also, dimensionality reduction helps when results of PCA are passed to machine learning algorithms in that they reduce training times for models.

The first step in PCA involves preparing the data. PCA builds on the fact that different predictor variables are likely to be correlated with one another. When predictor variables are correlated, they account for overlapping portions of variance in an outcome measure, and so are ideal candidates for dimension reduction. For PCA, variables are centered and scaled, meaning converted to z-scores so that all variables have a mean of zero and a standard deviation of one.

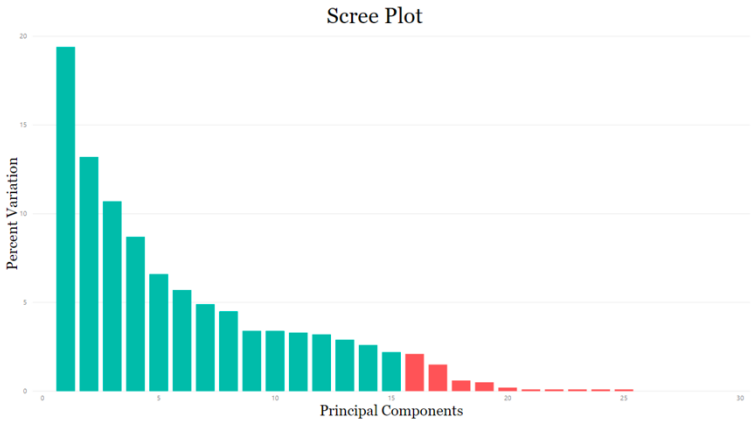
The second step is to calculate eigenvectors and their corresponding eigenvalues from the covariance matrix of the centered and scaled variables. The first eigenvector is that vector through the n-dimensional space (where n = number of predictor variables) that accounts for the most variance. Subsequent eigenvectors are created by the same process, but with a restriction that any new eigenvector must be orthogonal to all previous eigenvectors. The eigenvectors themselves are lists of correlations between that principal component and all predictor variables. The corresponding eigenvalues represent the amount of variance each individual eigenvector accounts for, scaled so that an eigenvalue of one corresponds to 1/n of the total variance. Assuming a large enough sample of observations, the analysis will extract the same number of eigenvectors as there are input variables.

Eigenvectors are lists of correlations between a principal component and all input variables. Therefore, eigenvectors can be examined to determine which of the predictor variables are contributing to that particular eigenvector. In that case, the correlations are described as factor loadings, and those with the strongest correlations (greatest absolute values) contribute most to the principal component.

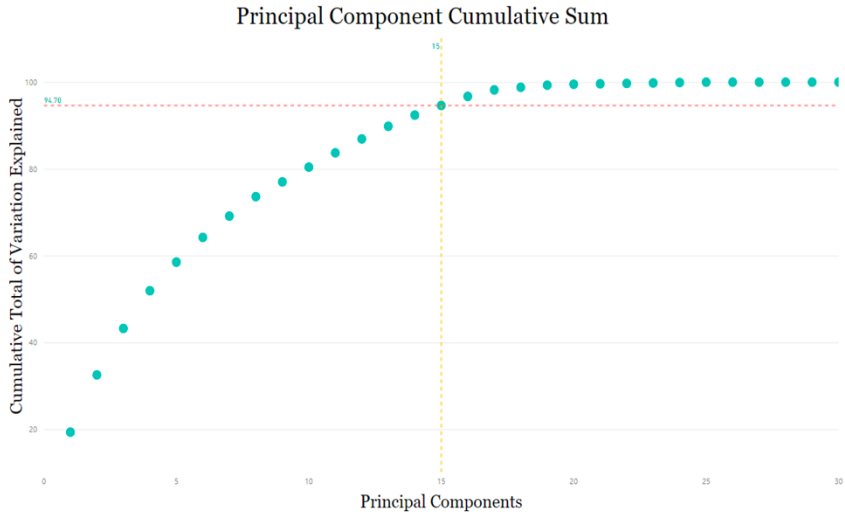
The final step involves determining which components extracted from the analysis should be used in place of the full set of predictor variables, and there are a number of ways to accomplish this. One option is to include all components with eigenvalues greater than 1, meaning those account for more variance than any single variable. A second option is to look at a scree plot, a plot of principal components and their eigenvalues. This is a downward trending plot, and the curve's "elbow" indicates where the amount of variance levels off. Those components before the elbow would be included in this case. A final option is to set up the PCA program to include as many components as necessary to account for some percentage of the original variance, usually ninety-five percent.

Results

The results of calling 'prcomp' on our training set performs a principal component analysis on our data and returns, among other things, the standard deviation of all newly created principal components. We squared the standard deviation to get the variance and then calculated the percent of total variance in each principal component. This allowed us to explore the total amount of variation accounted for by a given number of principal components. Our goal was to use the range of principal components that accounted for ninety-five percent of all variation (shown in green in the scree plot). We used two plots to help us visualize this relationship.

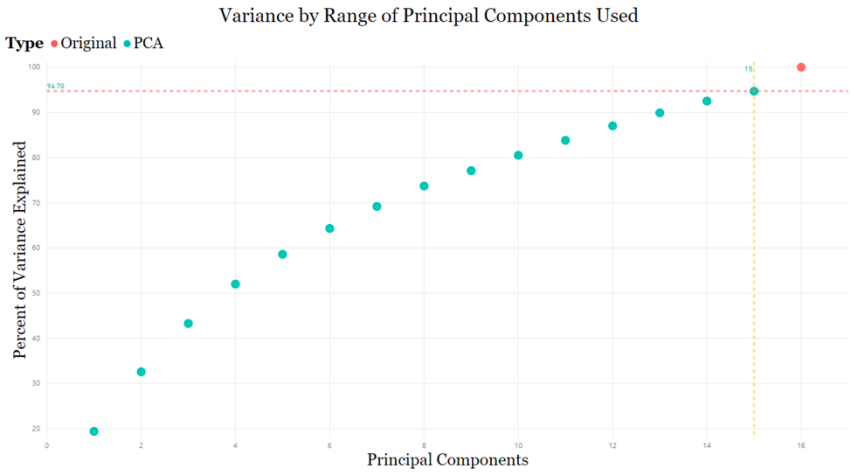


The first was the scree plot show to the right. The x-axis of the scree plot represents principal components one through thirty. The y-axis shows the percent of total variation accounted for by each of the principal components. We can see from the scree plot that somewhere between nine and eighteen principal components we should expect to find our ninety-five percent of total variation. Next, we'll look at an alternative way to visualize this percent of total variation, the cumulative sum plot.



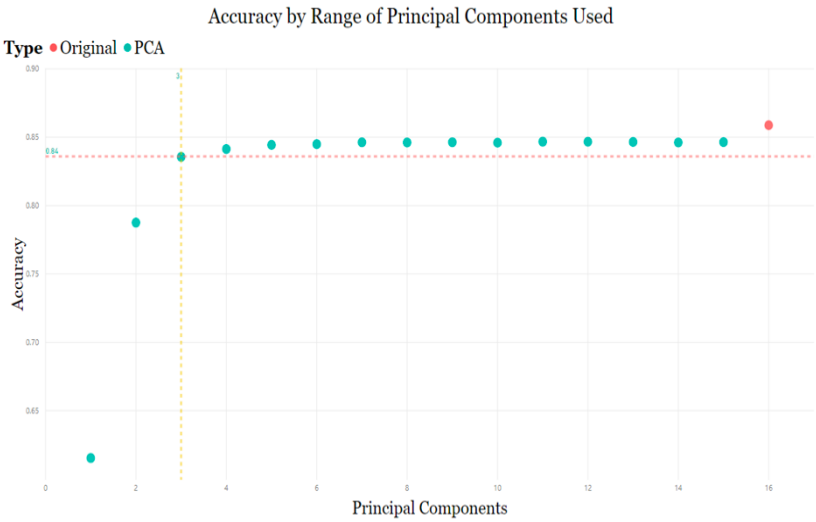
The plot to the left is the cumulative sum of the variance explained by the principal components. Here we see that around one through fifteen components (dotted yellow line) account for nearly ninety-five percent (dotted red line) of the total variance. Therefore, we will begin our experiment with one through fifteen components.

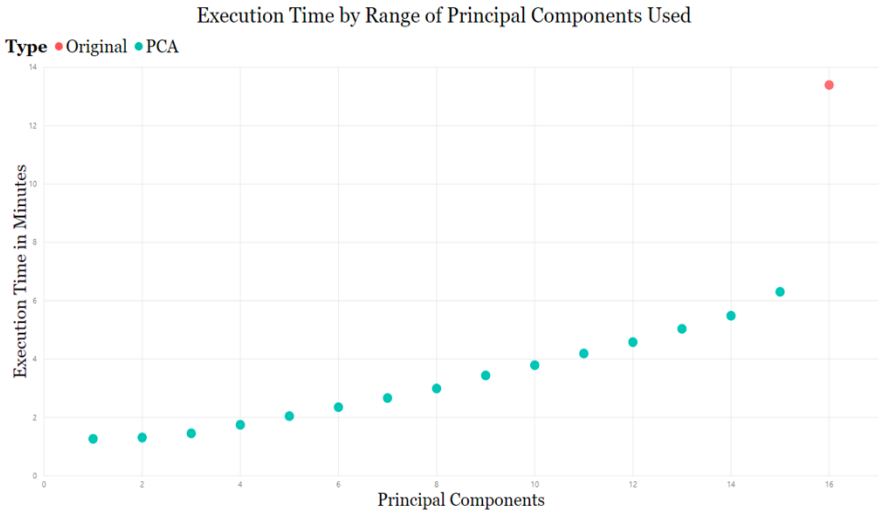
Starting with one through fifteen components we will test the execution time of model training, the prediction accuracy of the model on the test set, and the total variance explained by that range of principal components. We will then reproduce the experiment with one through fourteen components, then one through thirteen, and so on until we reach just one principal component. Each of the results will be compared with the original dataset that did not use principal component analysis. The range of the x-axis in the following visualizations has been forced to account for the original dataset.



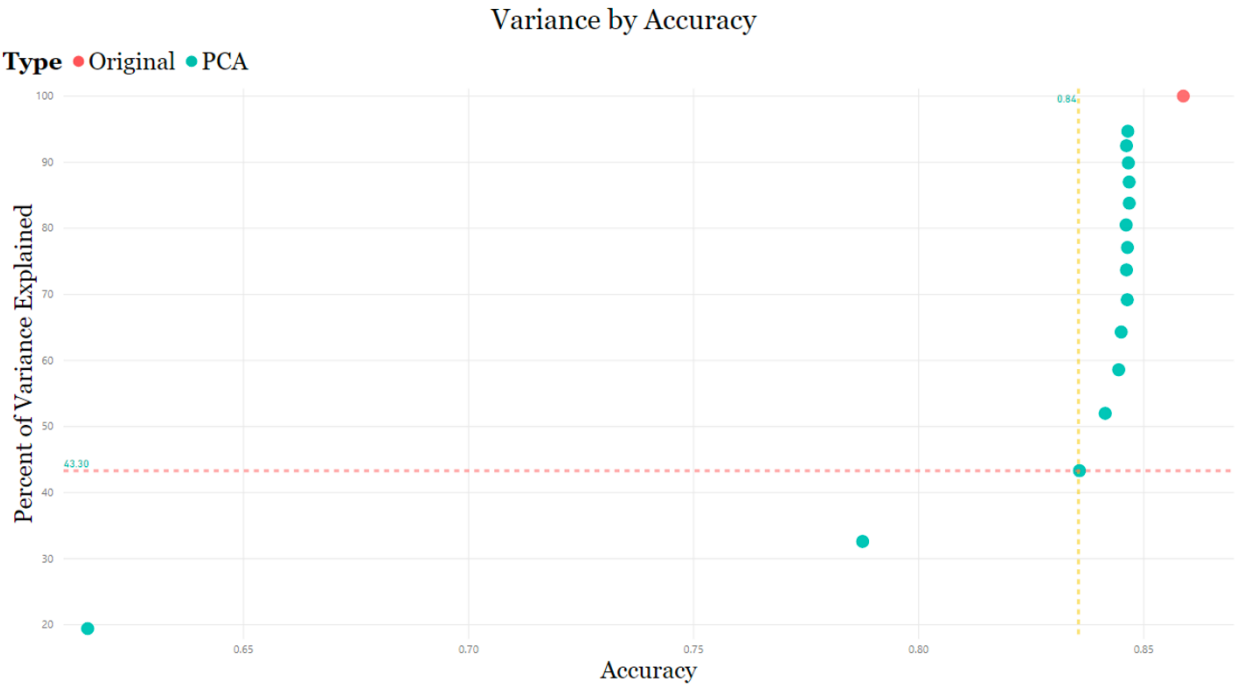
The graph to the left shows that as we decreased the range of our principal components from one through fifteen to just one we decreased the total variance we are accounting for. Highlighted with yellow and red dotted lines are the one through fifteen principal component range and its variance, 94.7%, respectively. This relationship is juxtaposed with the original dataset, shown here as the red data point, which of course accounts for all of the variance.

The graph to the right and the graph below prove our hypothesis. The graph below shows a consistent decline in execution time as we move through the iterations of ranges of principal components while the graph to the right shows that accuracy was maintained, to a point. We've highlighted in the Accuracy versus Principal Components graph the point at which we start to see a steep decline in accuracy.





The red and yellow dotted lines of the graph above show that at one through three principal components the accuracy is eighty-four percent, down from eighty-six percent accuracy of the original dataset. After this iteration, ranges of the principal components one through two and just one by itself have a drastically lower accuracy, seventy-nine and sixty-two percent accuracy respectively.



Final Thoughts

One very exciting aspect drawn out by this analysis is that of the relationship between variance and accuracy, shown in the graph above. In this visualization we see that as we increase the amount of variance accounted for in the training set, we increase our accuracy. We've highlighted the inflection point eighty-four percent accuracy and 43.30% total variance accounted for. This was especially surprising to the team. None of us thought we could obtain that high of an accuracy with that little variance accounted for. Incidentally, it took less than ten percent of the execution time to reach this accuracy as compared to training the model on the original dataset.

From the previous graphs titled Execution Time and Accuracy by Range of Principal Components Used we show that we can successfully use principal component analysis to decrease execution time while maintaining accuracy.

This exploration into the use of PCA to reduce execution time plays a vital role in parameter tuning in model training. Parameter tuning, and even hyper-parameter tuning, requires iterating through all permutations of parameter options for a given model to obtain the best sequence. This can be incredibly time consuming. What we've shown here is the leveraging PCA can help to decrease the time it takes to reach this ideal parameter sequence substantially.

Hotelling, H. (1933). *Analysis of a Complex of Statistical Variables into Principal Components.* Baltimore, MD: Warwick & York.

Hotelling, H. (1936). Relations between two sets of variates. *Biometrika, 28*, 321-377.

Jolliffe, I.T. & Cadima, J. (2016). Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical, and Engineering Sciences, 374*: 20150202.

Pearson, K. (1901). On lines and planes of closest fit to systems of points in space. *Philosophical Magazine, 2*, 559-572.