Predicting U.S. 2016 Election Results

PSTAT 131/231

June 13th, 2018

Group Members:

Jacobo Pereira-Pacheco
Lash Tan

Advisor:

Alexander Franks

Background

Predicting voter behavior is a difficult problem for many reasons. First, each voter's sentiment on every candidate is impacted by a seemingly infinite amount of factors. Some of these factors are more important than others – party affiliation (which usually correlates with many smaller factors), current political climates, high-profile legislation, and even the individual's economic status are just a drop in the bucket in the amount of components that will ultimately influence the decision made on election day. Second, collecting data on every eligible voter in America is virtually impossible. Polls must estimate the overall distribution of voter preferences by sampling just a portion of the population, and systematic polling errors are to be expected. Finally, these political sentiments are constantly fluctuating. A poll that was taken a week prior may easily be outdated, and keeping up-to-date predictions is extremely cost intensive.

Nate Silver's fivethirtyeight predictions took a cautious look at polling errors which gave Trump a much higher likelihood of winning the election compared to other predictions. Silver was conscious of the differences between errors in the national polls and errors in the state polls. Due to the Electoral College, errors in the state polls would most likely have much more of an impact than errors in the national polls. Silver points out that one of the biggest reasons why his model gave Trump a relatively higher chance of winning the election was because polling errors in states could be correlated, and if the reason behind a particular polling error is enough to flip one state, it may be enough to flip other states with similar demographics as well. This was highlighted in the 2016 election as Clinton ended up losing several Midwestern states that she was originally projected to win. In this sense, Silver's predictions were more robust than others.

Several analysts believe that these complexities involving polling errors played a major role in less accurate predictions in the 2016 election. As already mentioned, many of these election forecasts did not take polling errors into account. This was reflected by a limited amount of forecasting outcomes where Trump would come out as the victor when, in reality, Trump had many more paths to win the presidency. Future predictions may be improved in this sense by gathering more data on the historical accuracy of polls (possibly by region and/or demographic), which will reveal the reliability of certain polls compared to others and can allow analysts to compensate appropriately. Challenges in predicting future elections consist of incorporating better methods to take into account the inaccuracy of polls. Journalists often communicate results of election forecasts models by stating the candidate that is most likely to win the election and how he or she will win it, all without attempting to convey the more intricate details of their methods.

One attempt to explain polling errors is that voters were more apprehensive in sharing their preferred candidate. Trump has been a very polarizing and controversial figure, and it is possible (with some evidence) that some voters who preferred Trump were less inclined to share this sentiment in a live conversation. One popular solution to polling with sensitive questions is the *random-response model*. This method implements some anonymity to the interviewees' responses, even face-to-face with the interviewer. This way, the voter has less incentive to give false information, and polls can eliminate some of the error given by less truthful responses. By taking a case-by-case approach to solve particular reasons behind polling error, we can certainly expect to take great strides in future election forecasting.

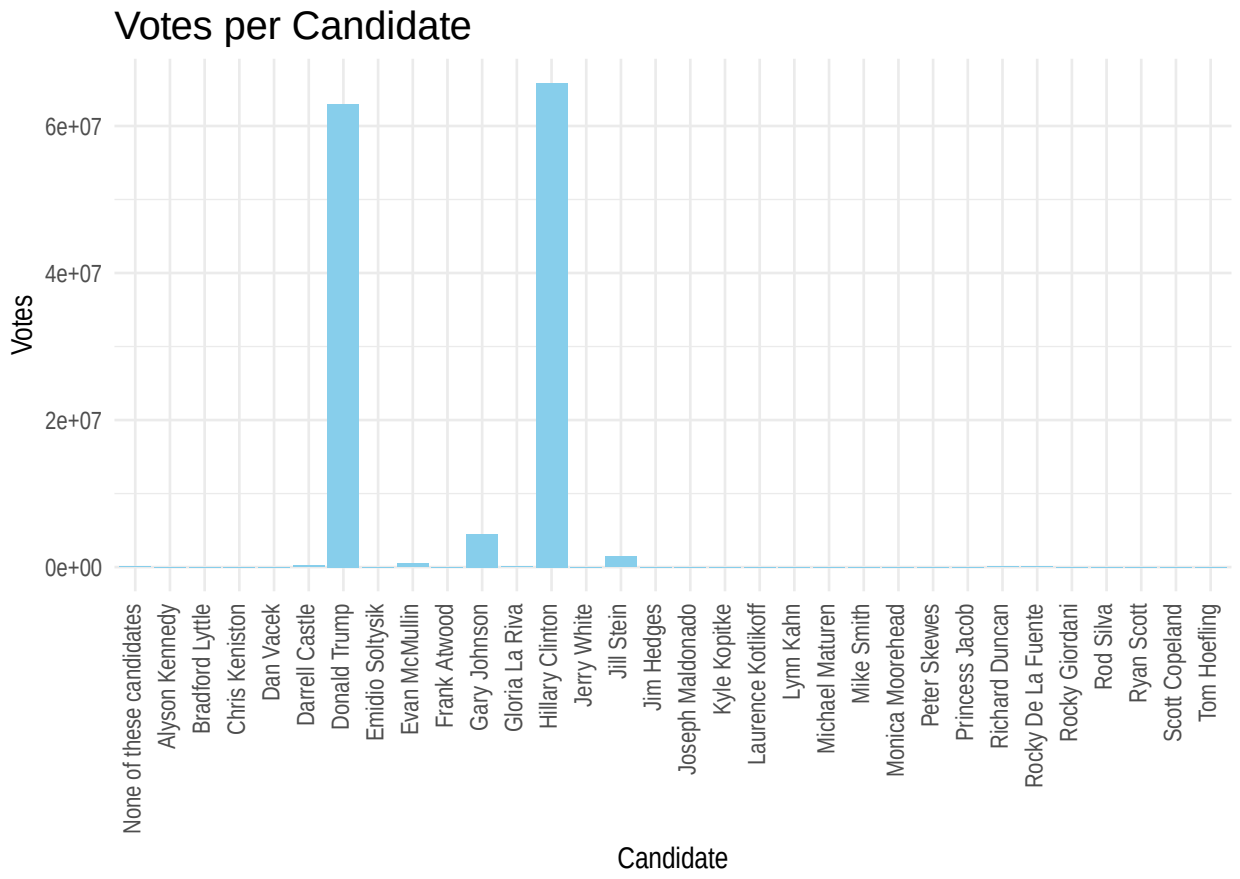
Data

The first data set that we will examine is the `election.raw` data set containing voting tallies per county, state, and federal levels.

Data Wrangling

We begin our analysis by separating our data into three groups: federal-level summary, state-level summary, and county-level data. Each group provides a breakdown of total votes for each candidate at the federal, state, and county levels, respectively. A glimpse into this data reveals 31 *named* candidates (with *'None of*

these candidates' representing everyone else), and we may explore the distributions of the votes received by candidate in a simple bar chart.



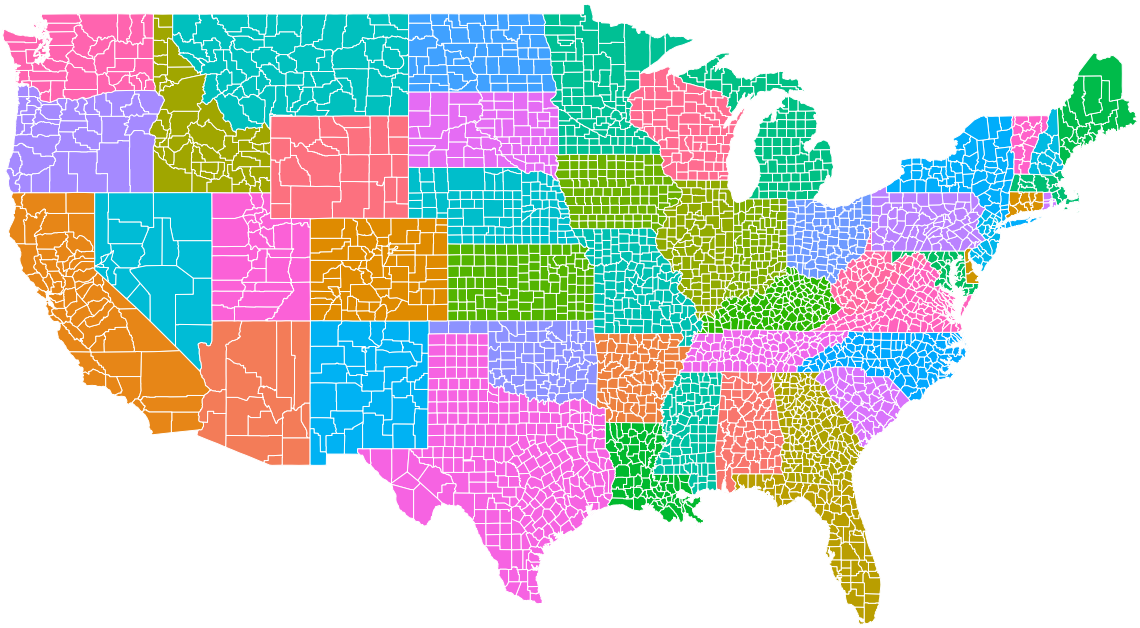
We can see that Donald Trump and Hillary Clinton accounted for a vast majority of the popular vote, which is indicative of the two-party system in the United States. Our focus will be primarily on these two candidates.

To get a more narrow perspective on election outcomes, we will examine states and counties by the candidate who received the majority of votes in each region.

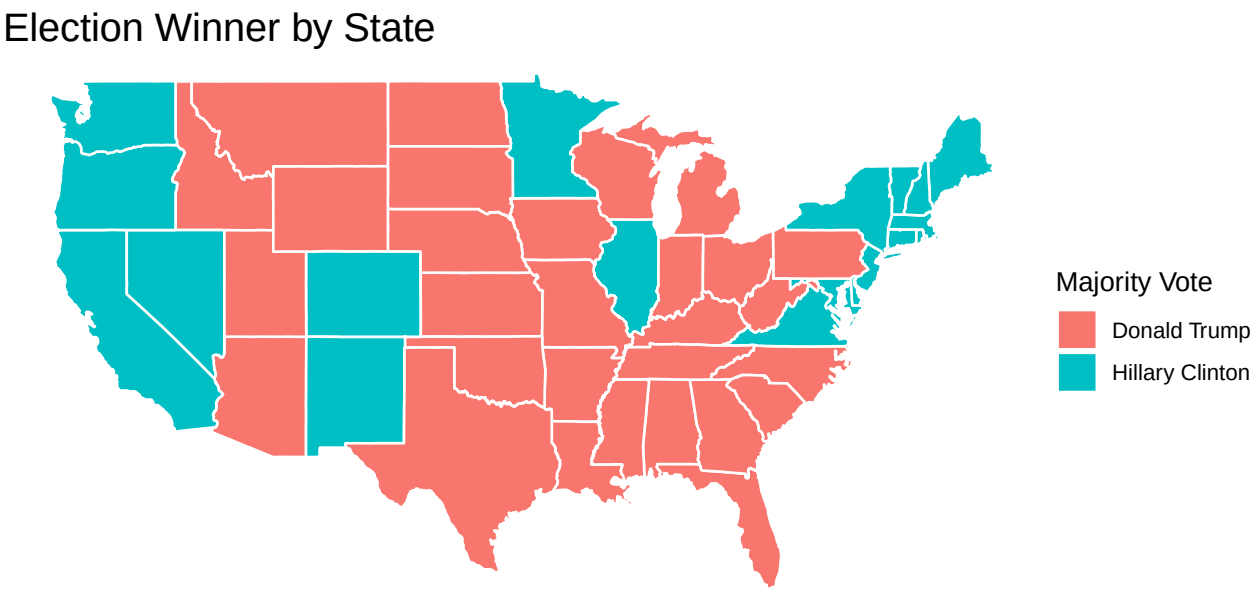
Visualization

To illustrate the election winners by region, we visualize the winners with the `ggplot2` package in R. First, we demonstrate a state-level/county-level map simply for reference.

Map of the United States by County

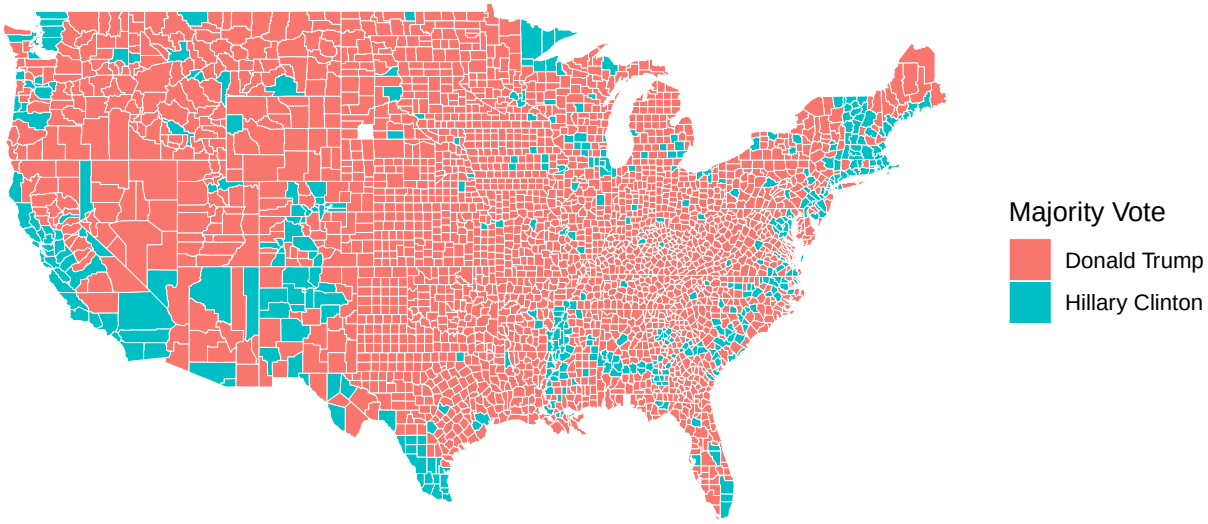


Now we may look at the majority vote for each state. Relative to party colors, blue represents a Hillary Clinton majority vote while red represents a Donald Trump majority vote. Many states historically vote for the same political party year after year, but "swing states" have a tremendous influence over who wins the candidacy. We can see that many Midwestern states such as Iowa, Ohio, Pennsylvania, Michigan, and Wisconsin went in favor of Trump.

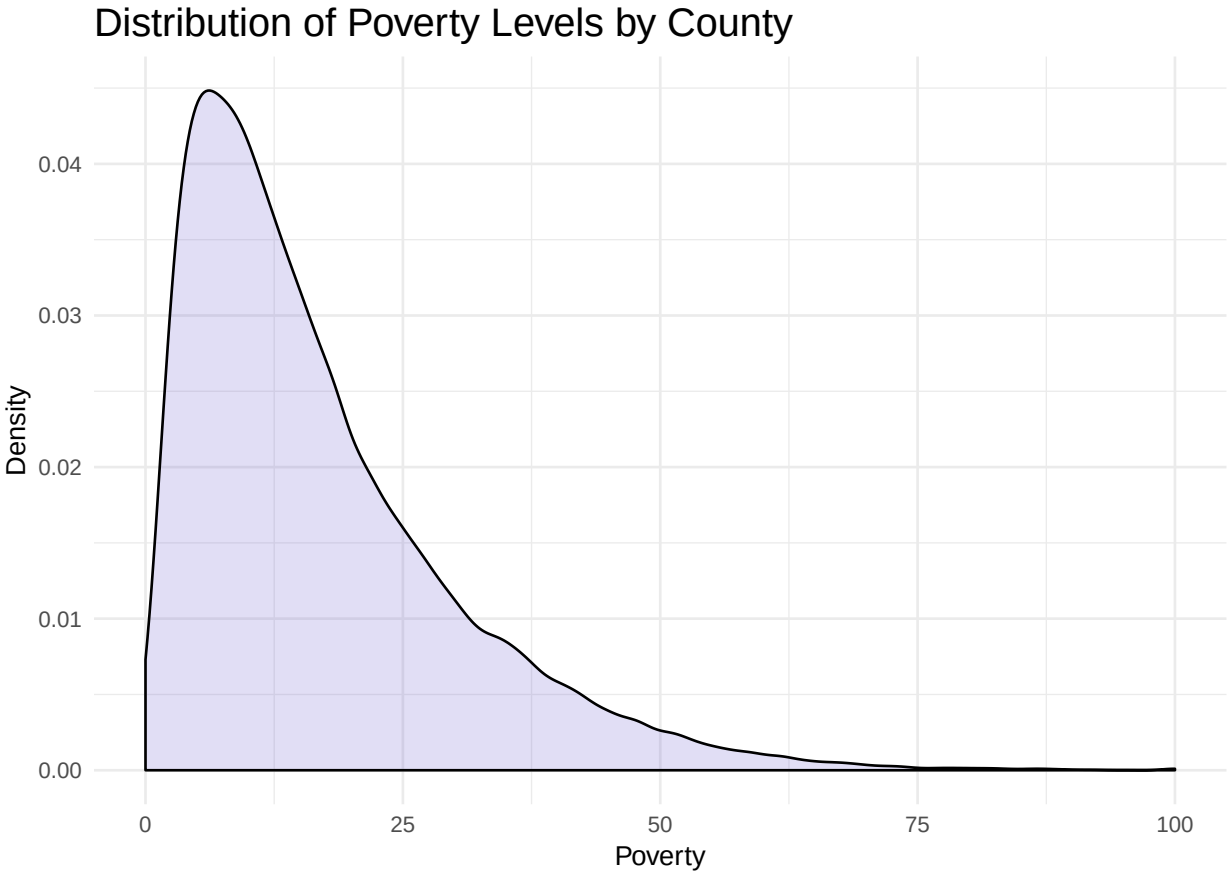


Similarly, we may look at the majority vote by county. While most of the geographical region may have voted for Trump, the more densely populated metropolitan regions went more in favor of Clinton. Note that some counties had missing data for candidate winners.

Election Winner by County

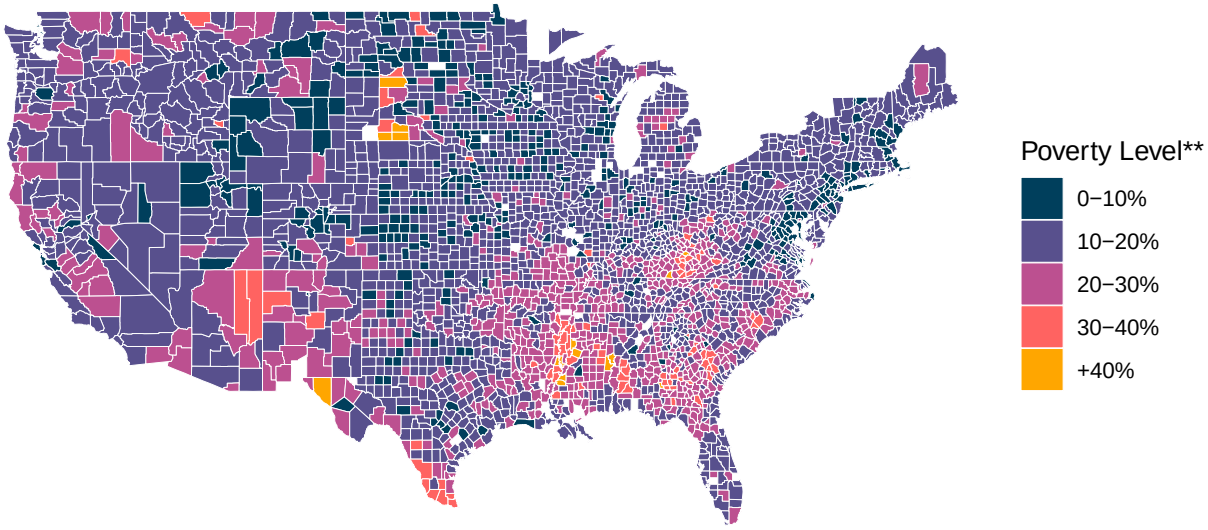


If we were to examine poverty levels by county, we may make comparisons with how it may correlate with election outcomes. First, let us look at the distribution of poverty by county. The following graph displays the distribution of poverty in the United States, defined by the percentage of people who are living in poverty in each county.



It seems that a majority of the counties have between 0-25% of their residents living above the poverty level. We may plot different levels of poverty by county in the same manner as before in hopes to notice any trends.

Poverty Level by County



While this map may be by no means conclusive, we may see that some regions with higher poverty levels may favor Clinton, particularly in the South. This may be a bit of a surprise, as Trump seemed to appeal to the lower class in typically red-voting areas.

From the `census` data, we are able to extract some useful information of each county's demographics. We begin by doing some standard cleaning of the data and selecting appropriate attributes.

First, we group `Hispanic`, `Black`, `Native`, `Asian`, and `Pacific` into a `Minority` attribute while dropping the `White` attribute. While these first 5 groups do not completely compose the entirety of the minority population in the United States, they make up a vast majority. And, we decided that `White` is complementary to `Minority` (even though it does not quite add up to the total population), and therefore one attribute (as a percentage) begets the other. Next, we drop `Walk`, `PublicWork`, `Construction`, and `Women` for the same reason as dropping `White` – these attributes can be computed by subtracting the totals by complementary attributes that are already in the data set.

As the `census` data set contains data for subdivisions of each county, we must also aggregate our data appropriately into county level data by using populations as weights. The following is the first several rows of the final data set, `census.ct`:

State	County	Men	Citizen	Income	IncomeErr	IncomePerCap	IncomePerCapErr	Poverty	Ch
Alabama	Autauga	48.43266	73.74912	51696.29	7771.009	24974.50	3433.674	12.91231	1
Alabama	Baldwin	48.84866	75.69406	51074.36	8745.050	27316.84	3803.718	13.42423	1
Alabama	Barbour	53.82816	76.91222	32959.30	6031.065	16824.22	2430.189	26.50563	4
Alabama	Bibb	53.41090	77.39781	38886.63	5662.358	18430.99	3073.599	16.60375	2
Alabama	Blount	49.40565	73.37550	46237.97	8695.786	20532.27	2052.055	16.72152	2
Alabama	Bullock	53.00618	75.45420	33292.69	9000.345	17579.57	3110.645	24.50260	3

Dimensionality Reduction

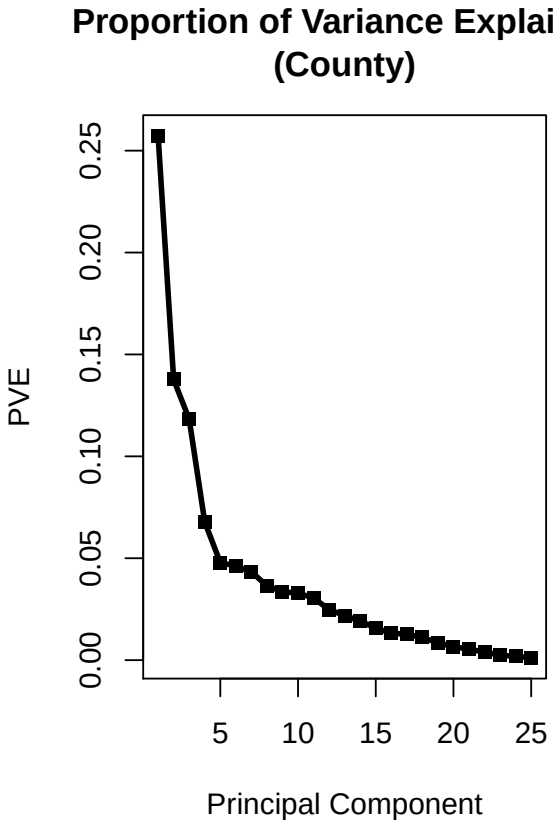
Our next goal is to implement dimension reduction in order to selectively choose our features and to reduce measurement noise. We utilize Principal Component Analysis (PCA) in order to capture only the main features of the `census` data. But first, we must scale and center the features in order to force the variance of our predictors to 1. Our motivation for doing so is that the predictors are on different scales – many predictors are percentages, but we also have variables representing income and population. These predictors have a much higher magnitude and could therefore create disproportionate effects on our PCA output.

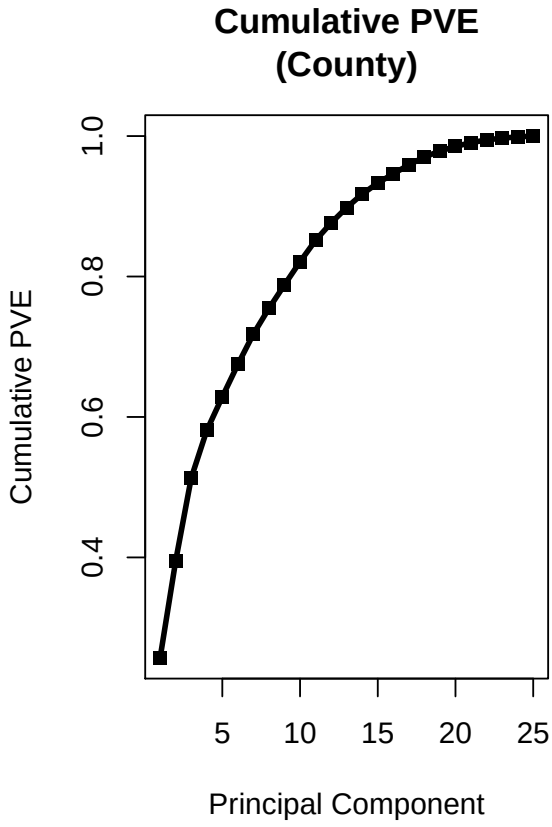
Once scaled, we can see the features that have the most significance for each principal component by examining the features with the highest absolute values in the loadings matrix. The first two of the following features have the highest absolute loadings in PC1 and PC2 for county data, and the second two features have the highest absolute loadings in PC1 and PC2 for subcounty data:

County		Subcounty	
PC 1	PC 2	PC 1	PC 2
IncomePerCap	SelfEmployed	IncomePerCap	Drive
0.3678114	-0.4164585	0.335491	-0.4990207

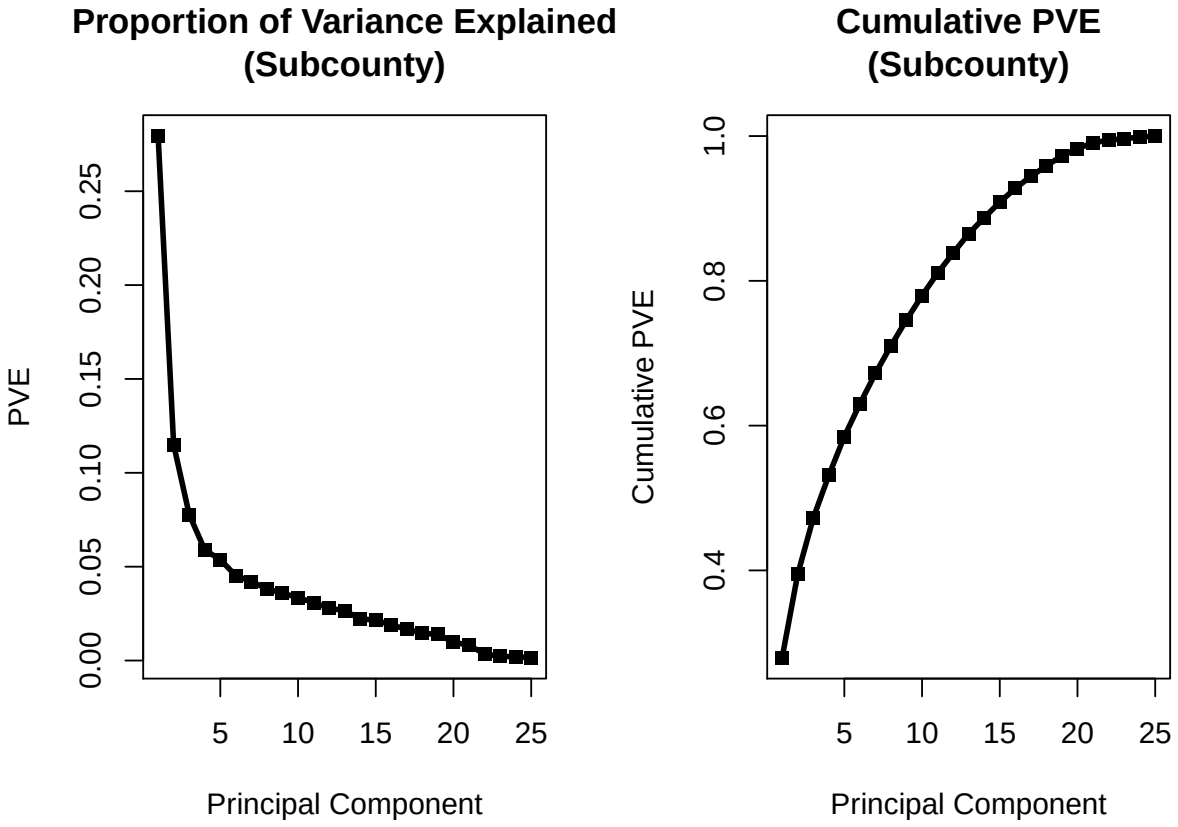
As PCA is a method to balance dimension reduction with a loss of information, we can set a threshold of the amount of variance explained that we want to keep when we decide upon a certain number of components to preserve. Each principal component contributes a portion of the variance explained in the data set, and we can selectively choose those which have the highest proportion of variance explained (PVE).

Upon examination, we discover that we need 14 principal components from the county data and 15 from the subcounty data in order to explain 90% of the total variance in each data set. We may plot the PVE and the cumulative PVE by principal component for the county data:





Likewise, we may also create the same plots for subcounty data:



We may also perform hierarchical clustering in order to group similar counties according to these same features. We may compare two methods using this tool – one with PCA and one without PCA.

When applying hierarchical clustering, we must again scale our features. In this case we use Euclidean distances (along with complete linkage), so our predictors must be on a similar scale in order to obtain desirable results. However, we have already scaled these features using PCA, so we will not scale a second time on this data.

We have chosen 10 as the number of clusters to group each county into for both the original `census.ct` data and the `ct.pc` data on which we have performed PCA. Once grouped, we may see the distribution of counties placed in each cluster.

Original Data Set									
1	2	3	4	5	6	7	8	9	10
2398	739	6	7	5	45	1	10	3	4

PCA Data Set									
1	2	3	4	5	6	7	8	9	10
1449	1388	133	121	4	8	42	14	56	3

The original data set and the PCA data both placed a heavy amount of observations in the first two groups, but we can see some discrepancies in the remaining 7 groups as PCA had a relatively higher amount in these

groups.

In addition to the two tables above, we may also compute the number of observations that are classified into the same groups across both methods and observations that are classified into different groups alike. In the table below, the diagonal elements are classified into the same groups whereas the off-diagonal elements are classified into different groups.

Original	PCA									
	1	2	3	4	5	6	7	8	9	10
1	1401	806	132	1	0	0	0	2	56	0
2	47	567	0	87	0	0	38	0	0	0
3	0	1	0	0	4	1	0	0	0	0
4	0	0	0	0	0	7	0	0	0	0
5	0	2	0	0	0	0	0	3	0	0
6	0	7	0	31	0	0	0	5	0	2
7	0	0	0	0	0	0	0	1	0	0
8	0	5	1	0	0	0	4	0	0	0
9	1	0	0	2	0	0	0	0	0	0
10	0	0	0	0	0	0	0	3	0	1

There are a significant amount of counties that have been placed into different groups. To get a sense of which of these two methods is more accurate, we may simply look at one specific county in comparison with all the other counties in that grouping for each method. Let us look at San Mateo, a strongly Democratic-leaning county in the San Francisco region. Here are the other counties in group 6 that San Mateo is grouped with under the original `census.ct` groupings:

State	County	Men	Citizen	Income	IncomeErr	IncomePerCap	IncomePerCapErr	Pover
California	Alameda	49.00514	64.73888	83129.49	12634.54	37299.07	4705.082	12.714
California	Contra Costa	48.83284	65.64452	89623.17	13784.88	39265.13	4964.571	10.884
California	Marin	48.27963	70.02243	98924.65	17537.96	60992.69	9349.889	8.3301
California	San Francisco	50.89265	73.58313	85425.17	14863.15	52230.87	7837.429	13.351
California	San Mateo	49.19773	64.20050	100369.92	16123.02	47881.29	6115.552	8.0111
California	Santa Clara	50.26387	60.56144	100743.85	15214.63	43879.60	5480.027	9.7476

And here is the grouping for San Mateo under the `ct.pc` PCA groupings, this time placed in group 4:

State	County	Men	Citizen	Income	IncomeErr	IncomePerCap	IncomePerCapErr	Poverty
California	Marin	48.27963	70.02243	98924.65	17537.96	60992.69	9349.889	8.330185
California	Mono	51.68245	66.97300	56967.01	11982.98	29437.19	4402.312	4.894804
California	Napa	49.55701	64.80383	73856.62	14485.37	36674.12	5938.609	10.01198
California	San Mateo	49.19773	64.20050	100369.92	16123.02	47881.29	6115.552	8.011122
California	Santa Clara	50.26387	60.56144	100743.85	15214.63	43879.60	5480.027	9.747668

With a little bit of knowledge of the geographical locations of these counties along with historical trends[1] for these regions, it would be safe to assume that the closest counties to San Mateo - those displayed in the original data set - provide for a more appropriate clustering. One possible explanation as to why PCA did not perform as well is that the first few principal components may have captured most of the variation

[1] https://www.nytimes.com/elections/2012/results/states/california.html https://www.nytimes.com/elections/2008/results/states/california.html

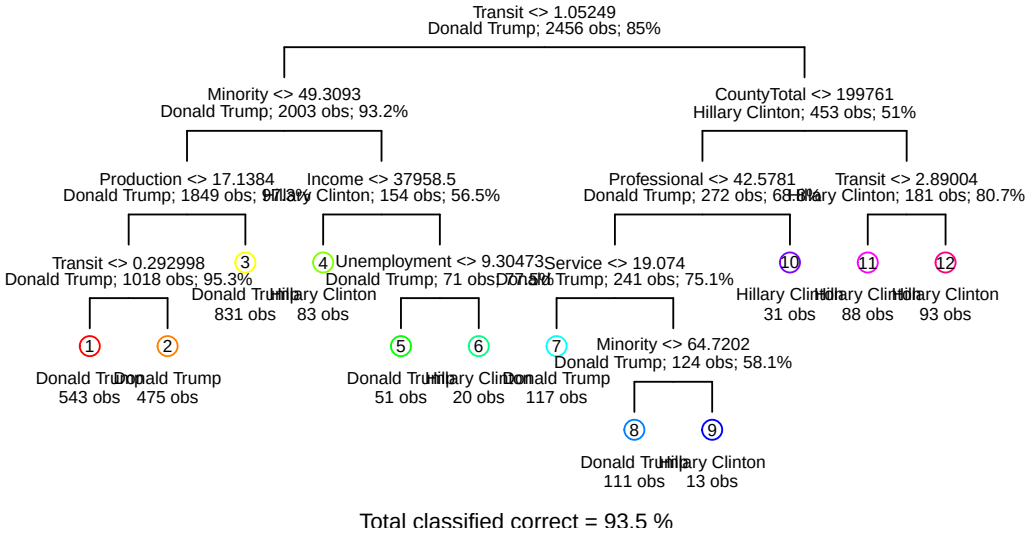
between counties according to our selected predictors, but this variation was not necessarily relevant to our objective of determining how these predictors are associated with election results. Another possibility could be that San Mateo is unique in the sense that it may more closely resemble counties that supported Trump than counties that supported Clinton based on the predictors that PCA deemed of higher importance. For instance, income per capita was shown to contribute significantly for PC1, and San Mateo's income per capita may be more aligned with Trump-supporting counties.

Classification

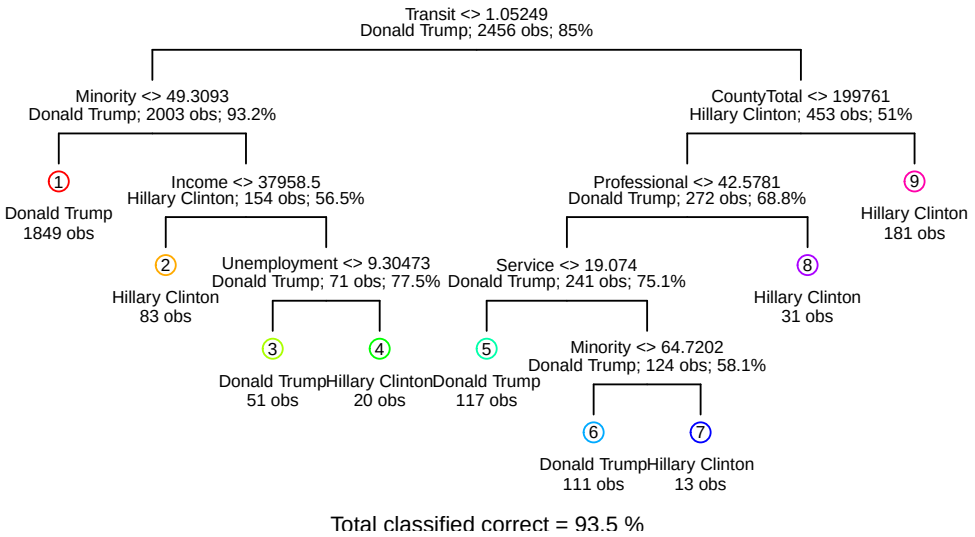
We turn to machine learning classification methods to compare and contrast the effectiveness of decision trees, logistic regression, and lasso regression on our data set.

Decision Tree

We begin by constructing a simple decision tree on a training set using the default parameters. Upon drawing this tree, we see that there are 12 terminal nodes with 93.5% correct classifications. Here is the tree visualized:



We now use cross validation in order to choose the optimal number of terminal nodes in order to prune the tree. In the figure below, we can see that we maintain the same classification rate if we prune the tree to 9 terminal nodes:



We may immediately notice the 85%/15% class imbalance of counties won by Trump compared to counties won by Clinton which can explained by the dominance of geographical area won by Trump that was represented in the map shown earlier. From the tree, we can trace the important features that led to each candidate's success. For instance, Trump won a vast majority of counties where voters used public transit less (Transit $<$ 1.05249) and where the same counties had relatively fewer minorities (Minority $<$ 49.3093). These counties account for 1849 of the 2456 observations in our training set, and, after examining this subset, Trump won 97.3% of these counties. In the other direction, if we look at counties where more of the voters used public transit (Transit $>$ 1.05249) and the same counties had higher population (CountyTotal $>$ 199761), this accounts for 181 out of the 2456 observations, and Clinton won 80.7% of these counties. From these two cases alone, it is clear that demographics played a major role in the direction of voting outcomes.

From the pruned decision tree, we obtain a training error of 0.06514658 and a test error of 0.08306189.

Logistic Regression

We turn to logistic regression to classify counties by candidate preference. To do so, we simply run logistic regression on all predictors in the `trn.cl` data set. The training error for this model is 0.06596091 and the test error is 0.07817590, which is a slight improvement over the decision tree model.

Another point of interest is to examine the predictors that each model deems of significant importance. The pruned decision tree used `Transit`, `Minority`, `Unemployment`, `Carpool`, `CountyTotal`, and `Employed` in its model. By comparison, the significant variables for the logistic model are `Citizen`, `IncomePerCap`, `Professional`, `Service`, `Production`, `Drive`, `Carpool`, `Employed`, `PrivateWork`, `Unemployment`, and `Minority`.

The decision tree was more selective in deciding the significant variables. It also used `Transit` on the first split,

whereas logistic regression did not consider this variable to be of significant importance. `FamilyWork` and `Service` had the highest absolute coefficients of the significant variables in the logistic model, meaning unpaid family workers and service workers had a very high influence over the results of the election. Computationally, `FamilyWork` has a coefficient of -1.149, and we may interpret this as follows: an increase in 1 unit of `FamilyWork` changes the odds of our prediction *multiplicatively* by $e^{-1.149}$. Likewise for `Service`, which has a coefficient of 0.3724, we may interpret this with an increase in 1 unit in `Service` leads to a change in the odds of our prediction multiplicatively by $e^{0.3724}$.

LASSO

One issue with our logistic model is that we have fit the model on our entire data set without adjusting for overfitting. If we implement a LASSO penalty on logistic regression, we may perform variable selection in hopes that our model will have increased performance on test data. We may use the `cv.glmnet` function to use cross-validation in selecting the best lambda, our penalty parameter. After implementing this value of lambda, we expect to have some predictors left out of our final model.

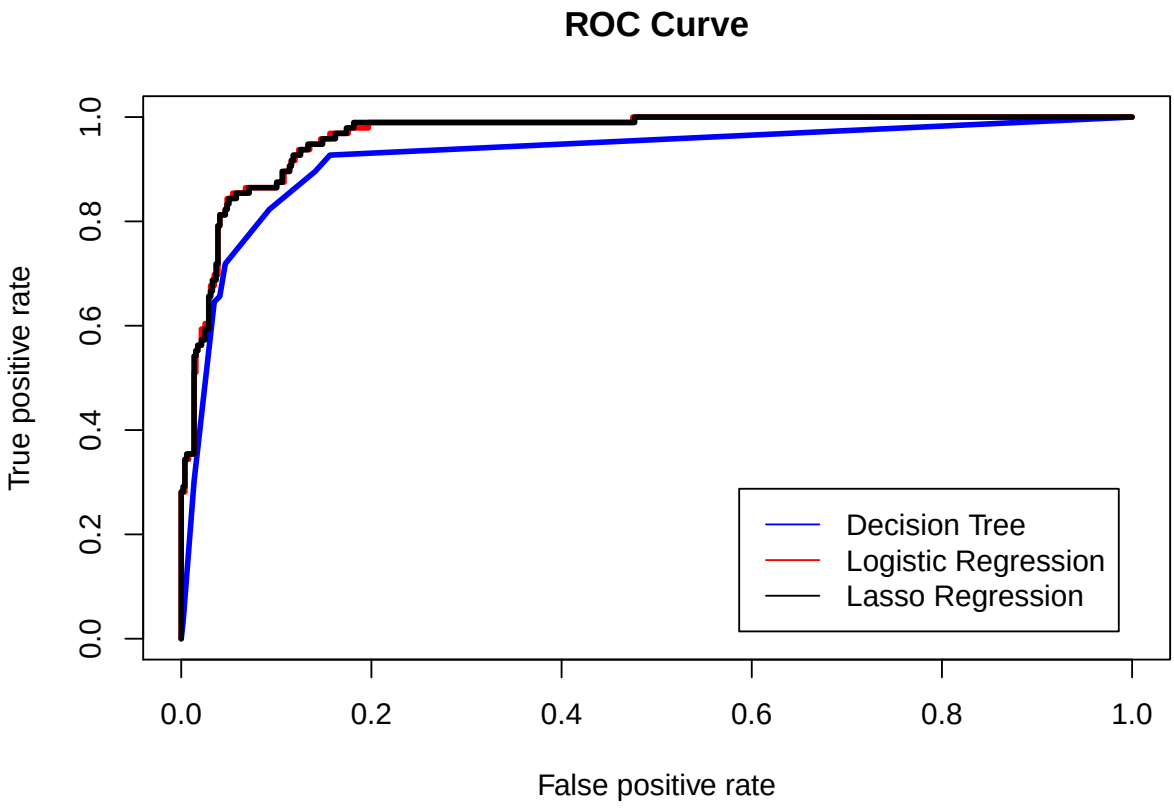
Upon building our model, we see that `ChildPoverty` and `SelfEmployed` are the two variables that have been removed. As LASSO shrinks coefficients, every other coefficient should be smaller (approaching zero) compared to the logistic model. As mentioned earlier, `FamilyWork` and `Service` have coefficients of -1.149 and 0.3724, respectively, and in the LASSO model, these coefficients have new values of -1.109139 and 0.3616098, respectively.

Finally, we calculate the training error to be 0.06596091 and the test error to be 0.07654723. This aligns with our previous prediction that LASSO would perform better than logistic regression in terms of misclassification errors.

Now we may make comparisons between each classification method. First, let us remind ourselves of the error rates for each model.

	train.error	test.error
tree	0.0651466	0.0830619
logistic	0.0659609	0.0781759
lasso	0.0659609	0.0765472

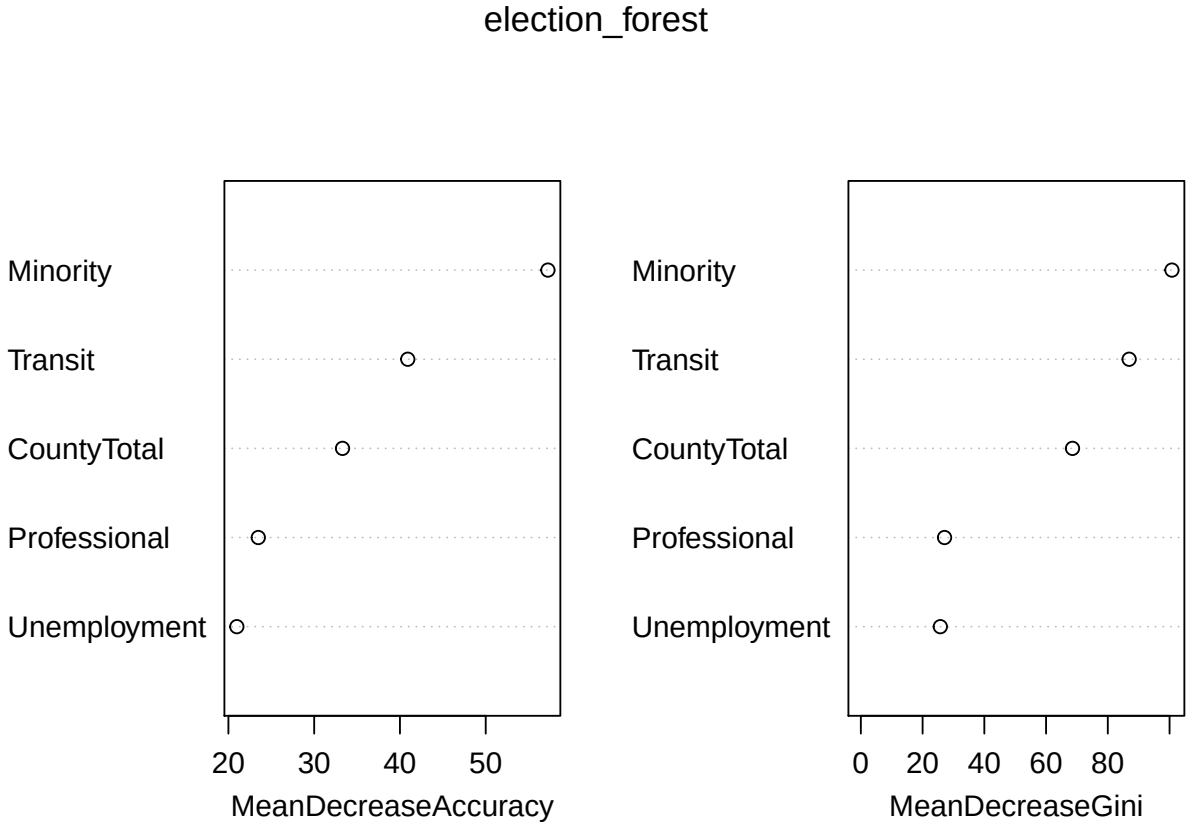
We can see that the training errors for each model performed similarly, but we are more interested in the test errors for each model. Logistic regression and LASSO outperformed the decision tree, with LASSO slightly outperforming logistic regression. However, we may be more interested in the true positive rates and false positive rates as opposed to misclassification rates, so we may be interested in ROC curves for each model.



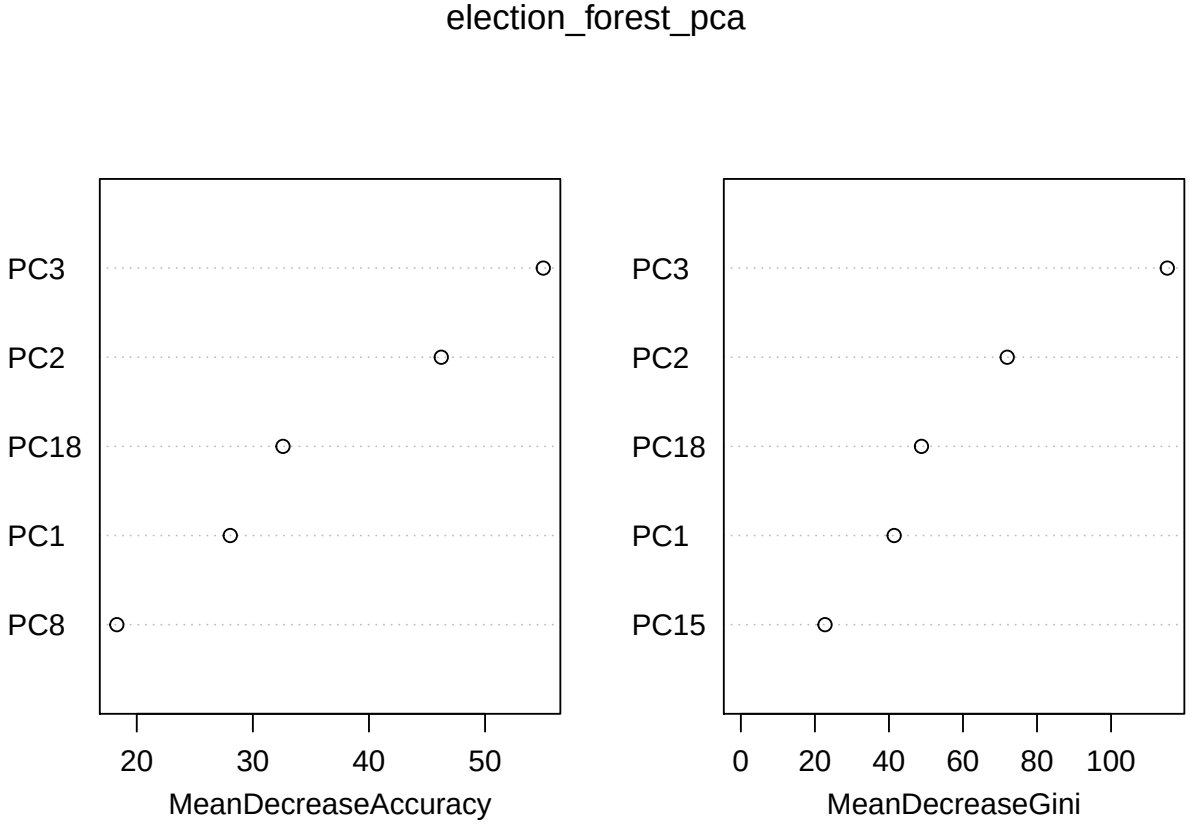
The ROC curves show that logistic regression and LASSO will virtually give the same performance on this data, and both methods outperform the decision tree. This may be indicative that the assumptions of logistic regression (i.e. homoscedasticity, linearity in log-odds, etc.) hold to be true. This would result in a higher performance compared to a non-parametric model such as the decision tree. As such, perhaps we may obtain even better results if we were to employ another method.

Random forests can be considered as the "swiss army knife" of classification methods in the modern machine learning era. They are relatively simple to fit and often give more desirable results than other methods. We will test this sentiment by training a model on the same `election.cl` data in 4 ways: one on the original data set, one on data that has been transformed with PCA, one with only the variables that account for 90% of the variance, and one with the five principal components that are deemed to be the most important in predicting county winners.

After fitting a random forest model on the training set, we may extract the most important variables considered by the model. The five most important variables `Minority`, `Transit`, `CountyTotal`, `Professional`, and `Unemployment`, shown in the figure below.



Likewise, we may extract the most important features of the PCA-transformed data set after fitting a random forest model. We will save these five principal components to employ a random forest model on these components alone. These components are `PC3`, `PC2`, `PC18`, `PC1`, and `PC8`. What stands out is that `PC8` and more so `PC18` were considered the most important features by the random forest, even though they likely do not explain much of the variation of the predictors in our data set. As mentioned earlier, this can be explained by how variance explained associates with the variance of the predictors in the data set, and may not be particularly relevant to the objective at hand.

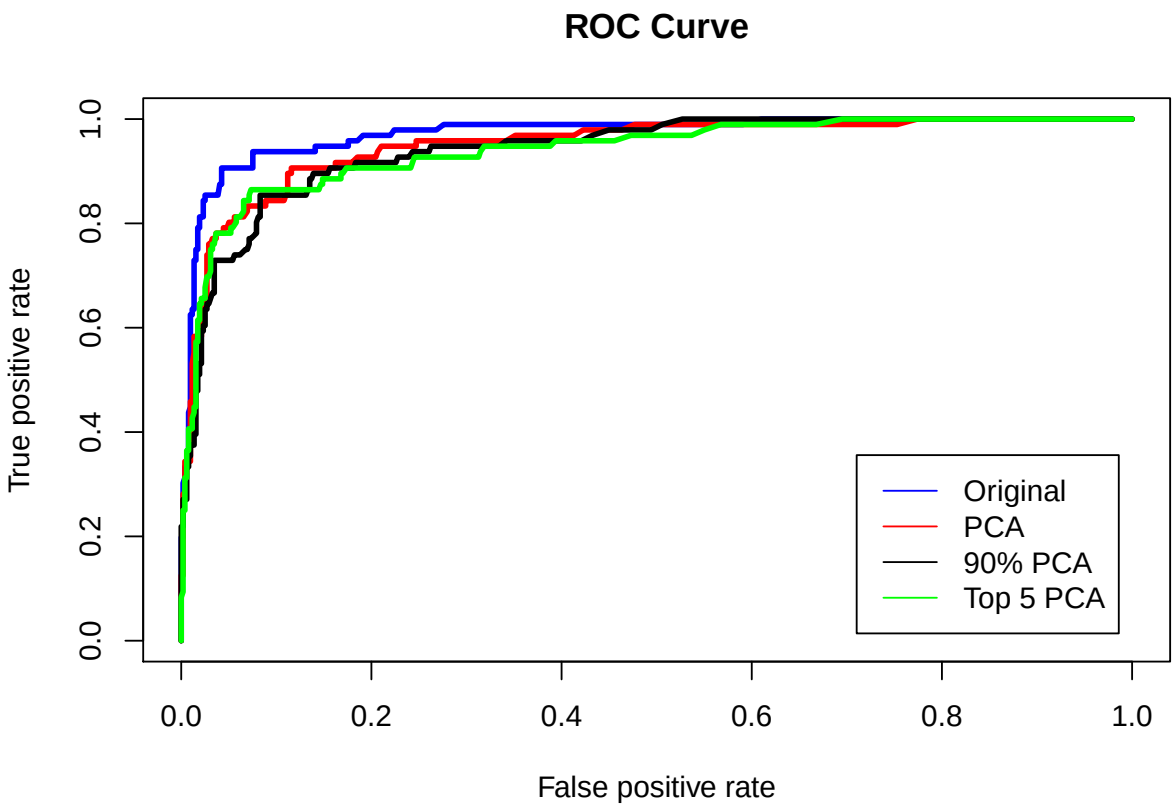


In the PCA data set, we find that 14 principal components are required to maintain 90% of the variance of the data, so we fit a random forest model on the first 14 principal components in this case.

After building all four models, we may examine the classification rate as we did on the earlier three models.

	train.error	test.error
original	0.0012215	0.0488599
PCA	0.0012215	0.0765472
90% PCA	0.0012215	0.0895765
5 PCA	0.0012215	0.0765472

In this case, the original data set obtained the best classification rate out of all four models. This may be expected, as principal components is a dimension reduction method and can still be useful to make data more compact, which would be more desirable with large data sets. The most intriguing result is how well the 5 PCA model performed in comparison with the 90% PCA model. The 5 PCA model had very similar results compared to the entire PCA model, showing that perhaps this is a better method in dimension reduction in comparison to choosing the first several principal components when fitting a random forest model. Let us visualize the ROC curves and compute the AUC for each model.



	Original	PCA	90% PCA	Top 5 PCA
AUC	0.9721284	0.9494148	0.9418537	0.9400941

Perhaps we too hastily endorsed the top five variable method after comparing AUC values. This method performed the worst out of these four classification techniques, which may lead us to believe that this method took advantage of a class imbalance to obtain a higher classification rate. At any rate, all four models performed tremendously, showing that random forests are indeed a viable tool.

Concluding Remarks

It is clear that supervised learning on 2016 election data provides very accurate results. A handful of demographic data for every county is sufficient in predicting these results accurately, leading us to believe that demographics played a highly significant role in election outcomes. It would be unfair to assume that these predictions would be comparable for every presidential election, so it may be of interest to implement a model built on 2016 data to see how accurate we can predict the outcomes of the 2012 election. A change in candidates would not necessarily be a major issue as we should be able to replace candidates based on their respective political parties, and while several counties may flip based on party, we expect a majority of counties to remain consistent. Based on how well the models built on 2016 data perform on 2012 data, we may attempt to implement these models on 2020 data to predict the next election's winner.

As for the predictions of 2016 being surprisingly less successful than other years, we can see that predictions *by county* was most likely not the culprit. If we were to attempt to improve the ultimate winner of the

election, we would want to improve predictions based on outcomes beyond whether or not individual counties vote one way or another. That is, we would want to build a model that accounts for polling errors, correlation between polling errors, and scenarios similar to the 2016 election when Clinton won the popular vote but still lost the election. In all, we have a classic example of how machine learning algorithms are limited when we do not build them upon sound reasoning.