Voter Prediction Model with Machine Learning (2016)

Seth Marceno(8934838), Derek Yan(4377347)

12/10/2019

1.

Voter behavior prediction is a hard problem because modeling human behavior is nearly an impossible task. The models that we use to predict elections are not always accurate. For many instances there are predictors that can be measured such as the economic growth/policies or changing opinions due to powerful political advertisements. Similarly, predictors that are measureable are subject to survivorship bias. For instance, online polls for candidates are voluntary or can be found on certain websites, which can skew results, or some people may lie about who they vote for; thus, these results do not encompass the population's feelings and the corrections statisticians try to make may not be accurate.

2.

Nate Silver's approach was unique because he had to add time series to his model and when accounting for the random variation of pollings from each state, Silver didn't look at the maximum variation, he took into account the full range of probabilities. Polls are prior to actual voting and therefore don't accurately model the population's feelings at the time of the actual election. This can be accounted for by generating a time series that helps to model changing intentions, by using the full range of probabilites instead of the maximum variation, Silver is able to better account for the change in support for a given candidate.

3.

In 2016, aggregated polls missed the correct results in many important swing states. This mistake led to a miscalculation of final results for the election. However, this miscalculation for these individual swing states tended to overstate the margin in which Clinton was ahead; furthermore, the national polls were off in the same direction. The bigger the lead Trump was predicted in a given state, the more he outperformed his polls. In order to make predictions better in the future, we must come up with ways to model a degree of uncertainty. For instance, voter turnout was lower than expected, as well as the amount of people who were unwilling to vocally admit who they supported. Therefore if models in the future can account for these degrees of uncertainty, we will see more accurate results.

county	fips	candidate	state	votes
Los Angeles County	6037	Hillary Clinton	CA	2464364
Los Angeles County	6037	Donald Trump	CA	769743
Los Angeles County	6037	Gary Johnson	CA	88968
Los Angeles County	6037	Jill Stein	CA	76465
Los Angeles County	6037	Gloria La Riva	CA	21993

4.

Looking at election.raw dataset, we find that it is a data frame with 18345 rows of observations and 5 columns of variables.

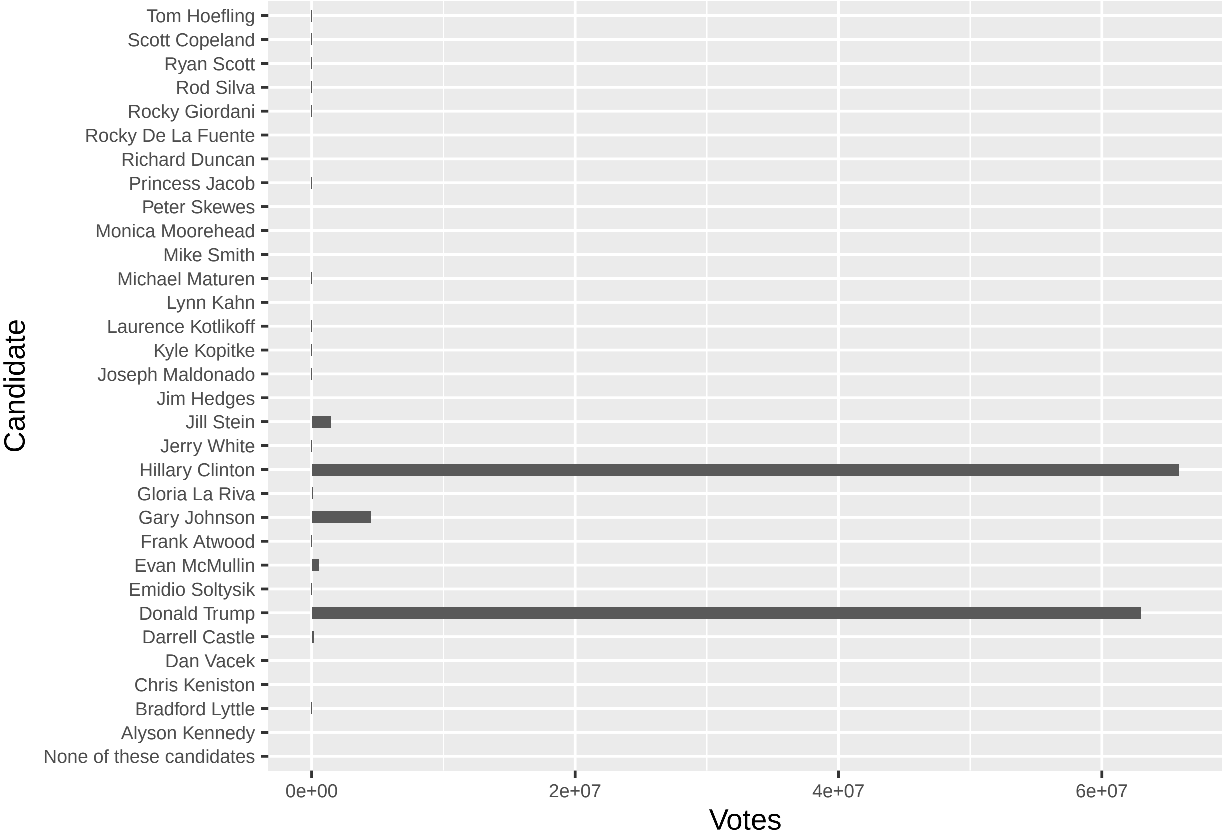
Because Alaska is a very small state (in terms of population), we remove those entries (fips = 2000) from the election data. This is because there is only one voting county in Alaska, and this data is already represented by the state total in the dataset. Therefore, by removing fips = 2000 in this data set, we are removing redundant data.

5.

Going forward we will have three subsets of the data election.raw, federal results for all of the candidates, state results for each candidate, and county results for each candidate.

6.

In the 2016 presidential race, there were 32 named presidential candidates.



7.

Now we are going to create two new variables county_winner and state_winner which are data frames containing the winner in each county for county_winner, and the winner in each state for state_winner respectively.

Here we have the first 6 rows of county_winner:

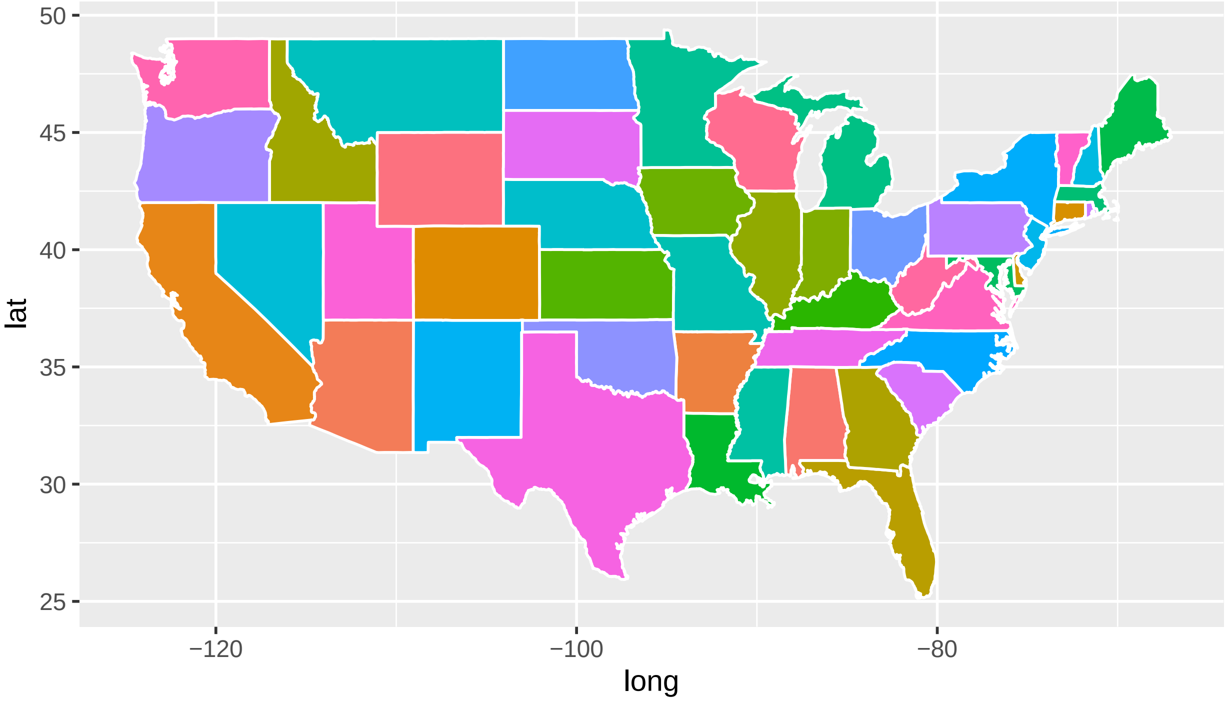
county	fips	candidate	state	votes	total	pct
Los Angeles County	6037	Hillary Clinton	CA	2464364	3421533	0.7202514
Cook County	17031	Hillary Clinton	IL	1611946	2156395	0.7475189
Maricopa County	4013	Donald Trump	AZ	747361	1536743	0.4863279
Harris County	48201	Hillary Clinton	TX	707914	1305434	0.5422825
San Diego County	6073	Hillary Clinton	CA	735476	1291078	0.5696604
Orange County	6059	Hillary Clinton	CA	609961	1186203	0.5142130

Here we see the first 6 rows of state_winner:

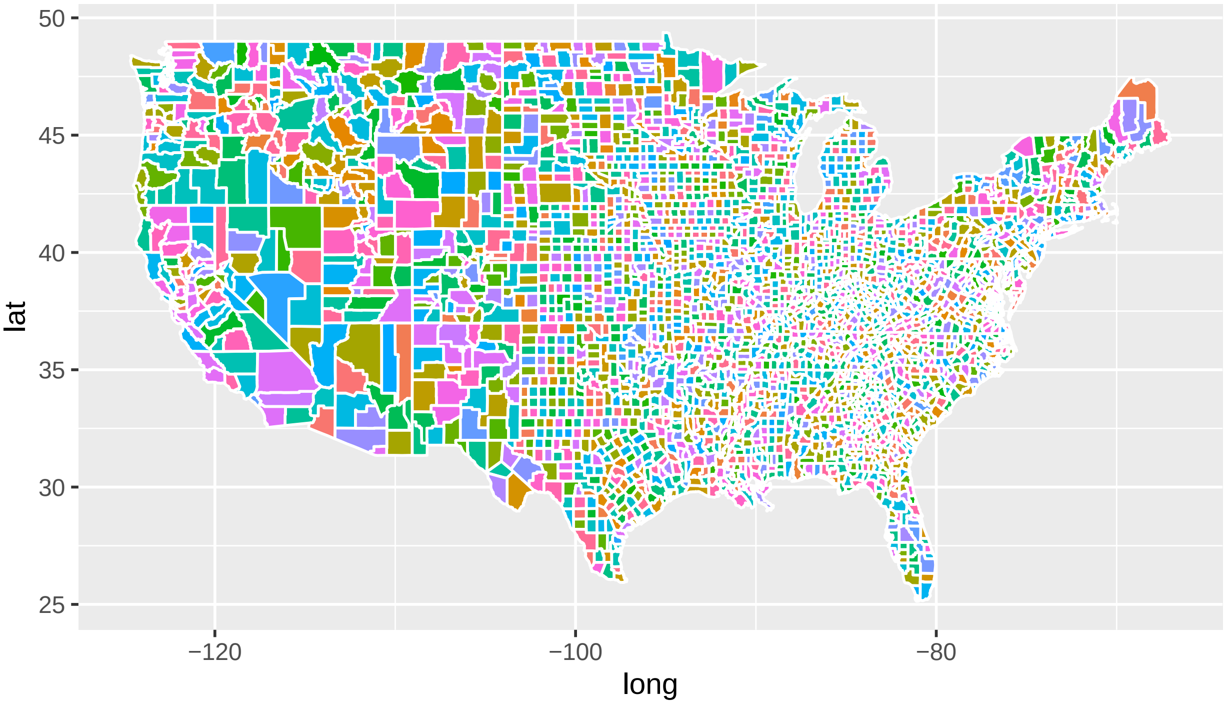
county	fips	candidate	state	votes
NA	CA	Hillary Clinton	CA	8753788
NA	FL	Donald Trump	FL	4617886
NA	TX	Donald Trump	TX	4685047
NA	NY	Hillary Clinton	NY	4556124
NA	PA	Donald Trump	PA	2970733
NA	IL	Hillary Clinton	IL	3090729

8.

Here we will create a map of the United States in order to help our visualizations.

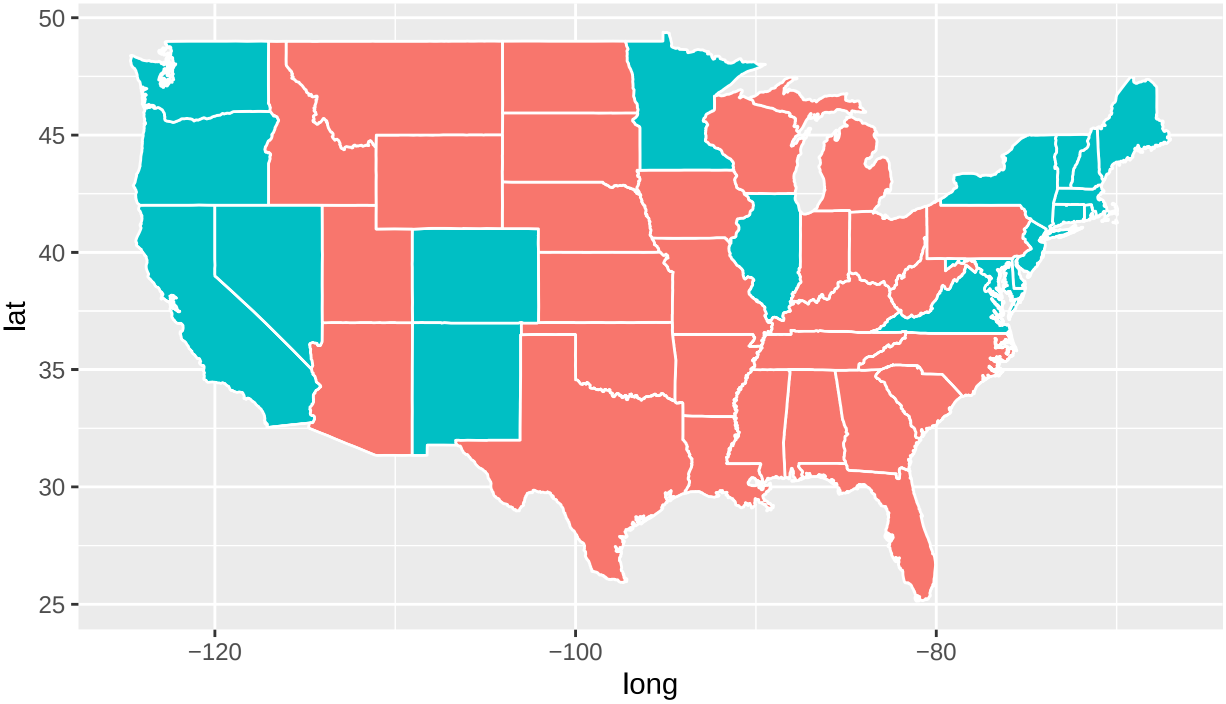


Below we see the United States divided and colored by counties:



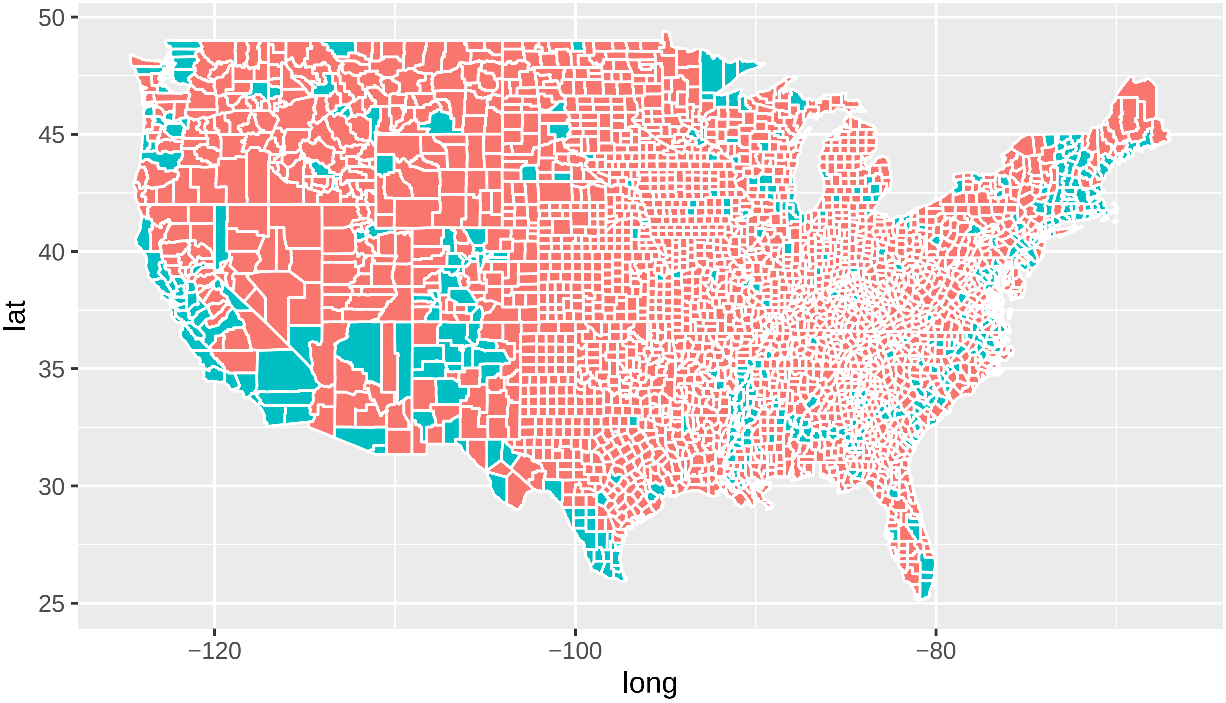
9.

On the map below, blue states correspond to Clinton and red states correspond to Trump winning:



10.

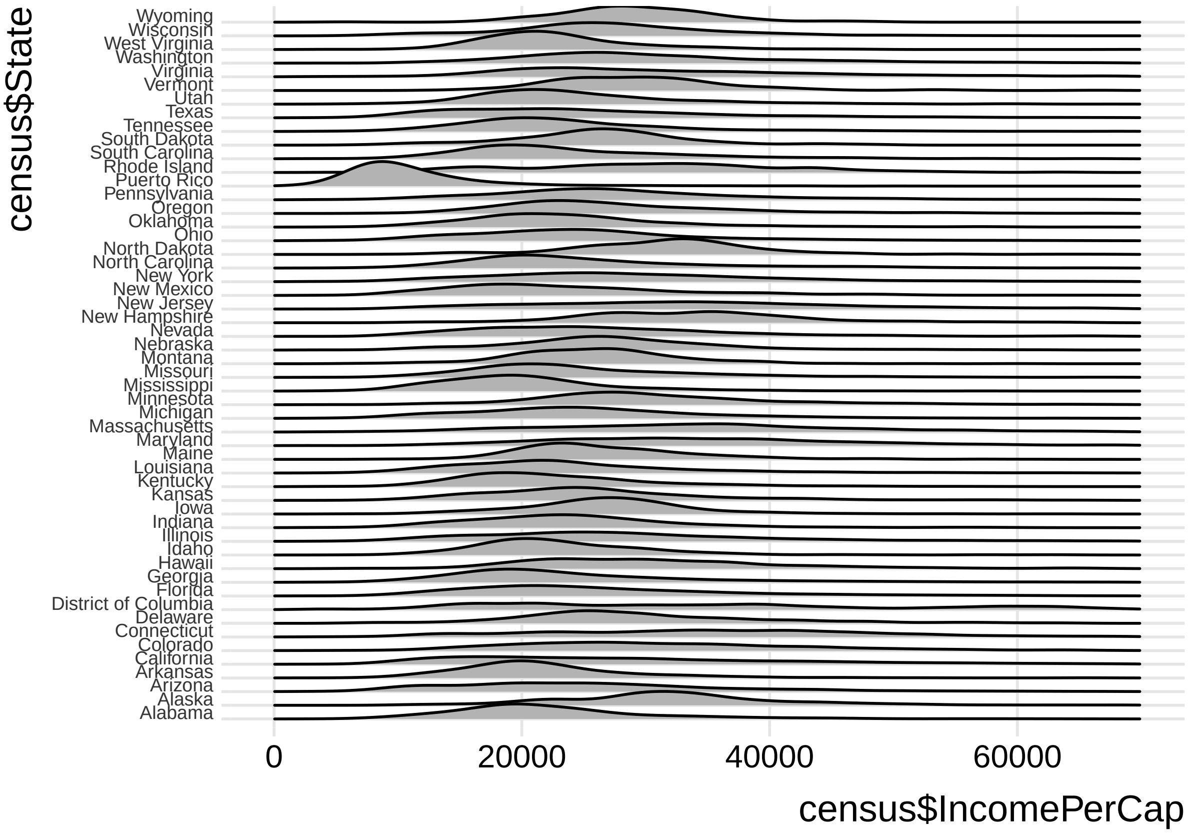
Here we have the same map separated into regions:



11.

The following graphic is made from the census data package looking at the average income per capita for each state:

```
## Picking joint bandwidth of 2000
```



From our ridgeline graph above, found using the census data, we find that states that tended to have a higher income per capita, such as North Carolina, Iowa, and North Dakota tended to vote for Trump.

12.

Since the census data set contains a lot of information, we will be aggregating much of it for simplicity of use. We will be cleaning to filter out any rows with missing data and combining/deleting rows. We will call this new set census.del.

State	County	Men	White	Citizen	Income	IncomeErr
Alabama	Autauga	48.43266	75.78823	73.74912	51696.29	7771.009
Alabama	Baldwin	48.84866	83.10262	75.69406	51074.36	8745.050
Alabama	Barbour	53.82816	46.23159	76.91222	32959.30	6031.065
Alabama	Bibb	53.41090	74.49989	77.39781	38886.63	5662.358
Alabama	Blount	49.40565	87.85385	73.37550	46237.97	8695.786
Alabama	Bullock	53.00618	22.19918	75.45420	33292.69	9000.345

13.

Now we will run some principal component analysis on both county and sub county levels of data.

Here we have a glimpse at ct.pc:

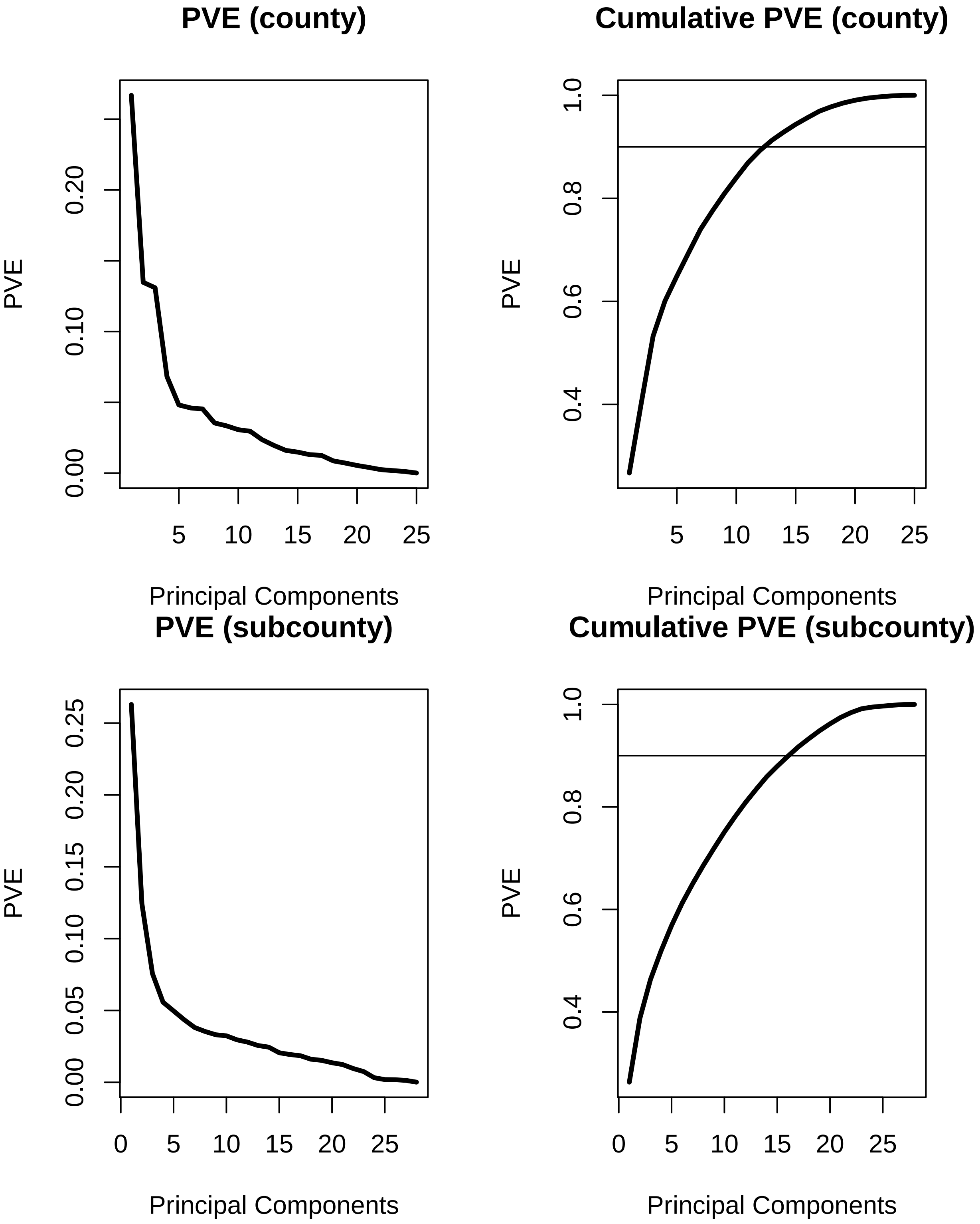
	PC1	PC2
Men	0.0068204	-0.1688521
White	0.2230248	0.0490840
Citizen	0.0046844	-0.0414236
Income	0.3200357	0.1351496
IncomeErr	0.1698710	0.1422103
IncomePerCap	0.3515555	0.0675449

Here we have a glimpse at subct.pc:

	PC1	PC2
TotalPop	-0.0324018	0.0116376
Men	-0.0173008	-0.0494517
White	-0.2404249	-0.3085397
Citizen	-0.1608439	-0.2307240
Income	-0.3025102	0.1556430
IncomeErr	-0.1989442	0.2224288

When doing principal component analysis on our county census data (ct.pc) and sub county census data (subct.pc), we chose to center because the mean of each column in our data is not zero, so we don't want the first principal component to point towards the mean. We chose to scale because we want our predictors to have variance one. Doing some exploratory analysis we find that the features: IncomePerCap, ChildPoverty, and Poverty have the highest absolute value for the first principal component for ct.pc. For subct.pc we find the highest absolute values of the first principal component correspond to IncomePerCap, Professional, and Poverty. On top of this, for both ct.pc and sbct.pc, we see the features such as men and women have the same value for PC1 and PC2, but have opposite signs. This is because these values are directly correlated with each other, specifically, the percentage of women in a given county is just 1 - the percentage of men.

14.



Looking at our PVE graphs we find that we must include 13 principal components for ct.pc and 17 principal components for subct.pc in order to have 90% of variance explained.

15.

Now we will perform hierarchical clustering on the dataset census.ct. Below we see a summary of the groupings for our hierarchical clustering with 10 clusters of census.ct.

tree.census	Freq
1	2871
2	155
3	2
4	7
5	46
6	4
7	105
8	20
9	4
10	4

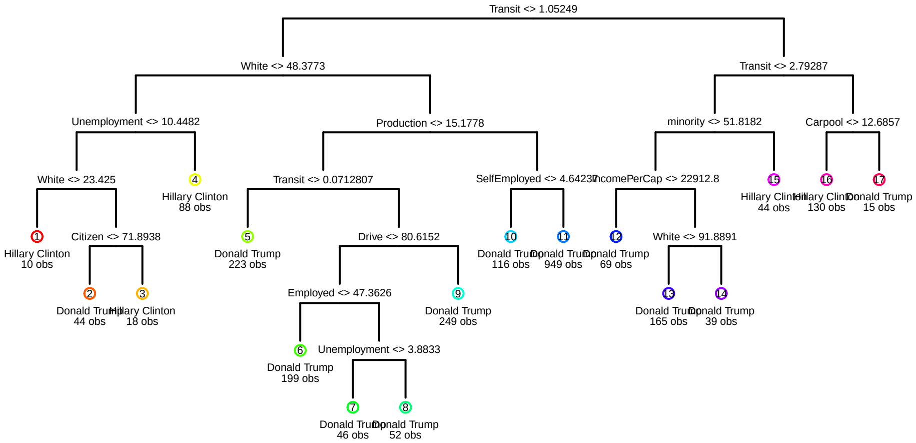
Here we re-run hierarchical clustering with only the first two principal components:

tree.pc	Freq
1	1335
2	829
3	358
4	69
5	78
6	482
7	41
8	8
9	9
10	9

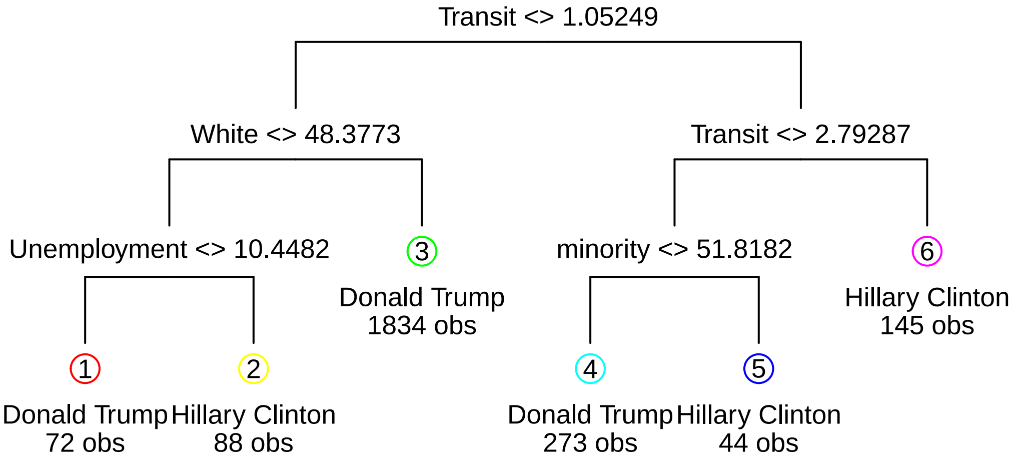
For our hierarchical cluster from census.ct, we have San Mateo clustered into group 1. For our hierarchical clustering from the our data frame of the first two principal components, we have San Mateo clustered into group 7. It makes more sense for San Mateo to be clustered into group 1 because it is clustered with counties that are near it. The hierarchical clustering with the first two principal components is off because the first two principal components dont encapsulate all of the variability in our data.

16.

Now we will train a decision tree on our trn.cl dataset.



From cross validation we find that we want to prune our tree to classify into 6 categories.



From the tree above we can see some good trends in the countries voting behavior. In general, Donald Trump seems to do better in counties that have a greater proportion of white people, smaller proportion of minorities, and a smaller proportion of unemployed citizens. On the other hand Hillary Clinton outperforms Trump in counties that have a higher proportion of minorities, with a higher unemployment rate.

Lastly, we will add the training and testing error of our pruned tree to our records matrix:

	train.error	test.error
tree	0.0789902	0.0715447
logistic	NA	NA
lasso	NA	NA

17.

Now we will run logistic regression to predict the winning candidate in each county. From the summary of our logistic regression, we see that the predictors Men, White, Citizen, Income, IncomePerCap, IncomePerCapErr, Professional, Service, Office, Production, Drive, Carpool, WorkAtHome, MeanCommute, Employed, PrivateWork, FamilyWork, and Unemployment are all significant. Comparing these predictors to our decision tree, we see similarities in all but minority. Minority is included in our tree model but not significant in our logistic regression. Looking at some of the more significant predictors in our logistic regression model we see that every percent increase in Unemployment corresponds to a 0.2097 increase in odd that the candidate

Donald Trump wins. Similarly, every percent increase in proportion of citizens in the county corresponds to a 0.1274 increase in odds that Donald Trump wins the county.

Lastly, we will add the training and test errors of our logistic fit to our records matrix:

	train.error	test.error
tree	0.0789902	0.0715447
logistic	0.0712541	0.0699187
lasso	NA	NA

18.

When creating our logistic fit we found that we have complete separation which is a sign that we may be overfitting our model.

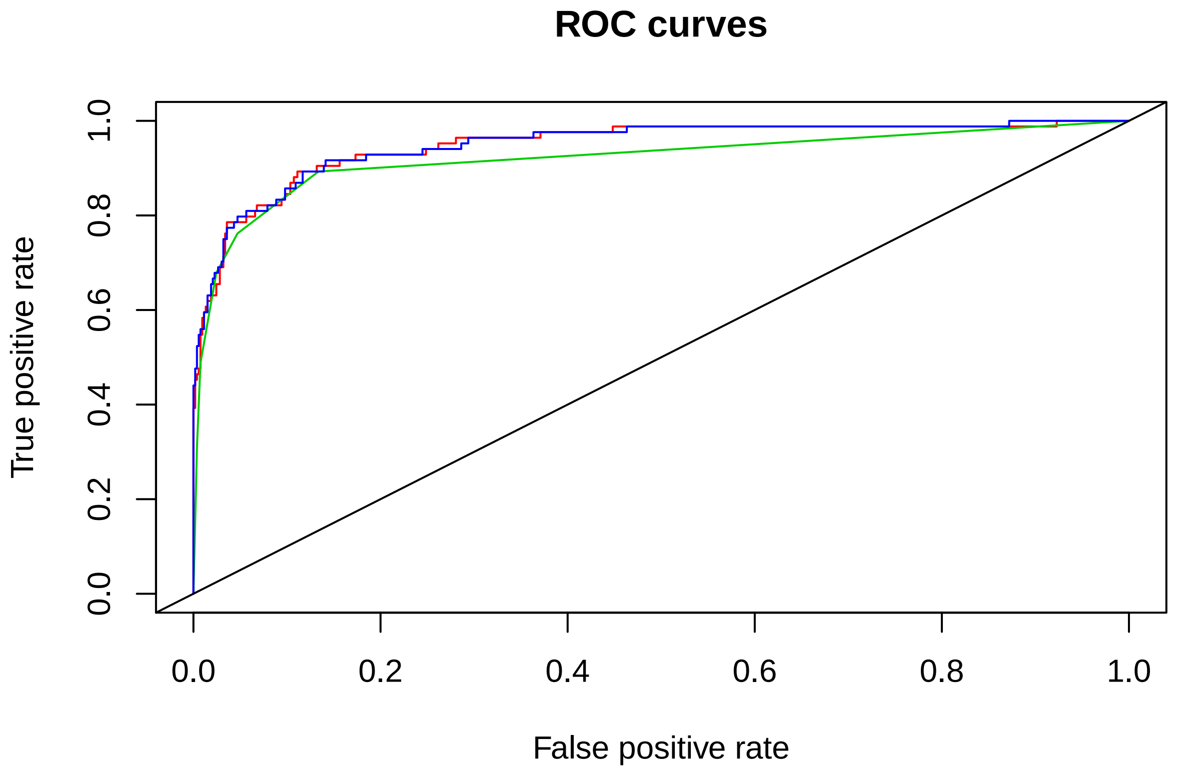
Using LASSO regression, we find that the best lambda value to use is $\lambda = 0.00076031136$. We see that the non-zero coefficients using this lambda for a LASSO regression are the all the predictors except for ChildPoverty, OtherTransp, SelfEmployed, and minority, which are all zero. Comparing this to a non-penalized model we see that almost all of the zero coefficients for our penalized model were unsignificant predictors for our unpenalized model.

Saving the training error and test errors to our records matrix:

	train.error	test.error
tree	0.0789902	0.0715447
logistic	0.0712541	0.0699187
lasso	NA	NA

19.

Here we are looking at the ROC curve of our decision tree, logistic regression, and LASSO regression:



Looking at the results of our ROC curves we see that the red curve corresponds to logistic regression, the green curve corresponds to our decision tree, and the blue curve corresponds to our LASSO regression. We see that our logistic regression model has the best misclassification error; however, due to the fact that we got 0 or 1 for some of our predicted probabilites, this model may be overfit to the training data. To account for this, we ran a LASSO regression which out performed our decision tree model. This may be because our final pruned decision tree only took into account 4 predictors out of a total of 26 total predictors. Thus, due to the high amount of possible predictors, the decision tree may have left out significant predictors that were taken into account for our logistic and LASSO regression models.

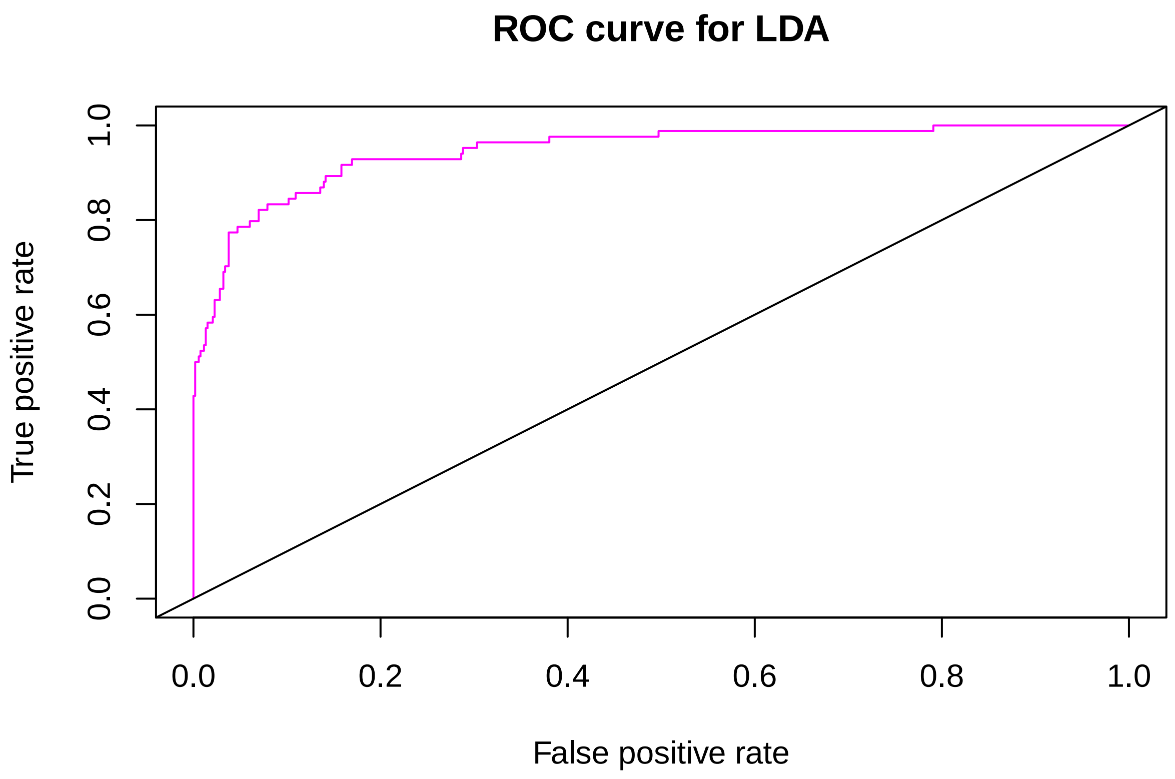
20.

When we ran our logistic regression, we found that we may have complete separation. This led us to question if LDA could perform better under these circumstances, if in fact, the distribution of our predictors is normal with the same variance.

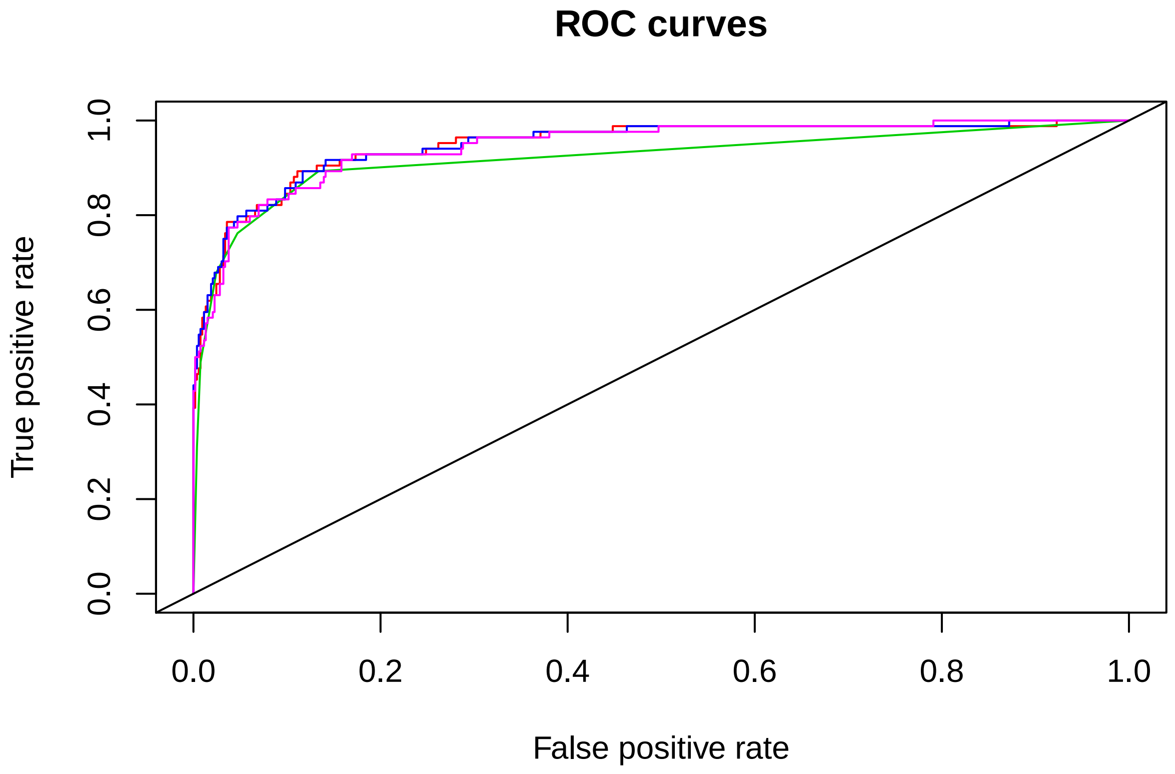
Here we can see the training and test error for out LDA fit:

	train.error	test.error
tree	0.0789902	0.0715447
logistic	0.0712541	0.0699187
lasso	NA	NA

Here we will take a look at the ROC curve for LDA:



Now adding this curve to our ROC curves graph above:



Now looking at all of the AUC's for our various models:

	AUC
Tree	0.919805398618958
Logistic	0.948121244731421
LASSO	0.948838669177658
LDA	0.945520581113806

We find that LDA is slightly out-performed by our LASSO model. This may be because the distribution of our predictors we assumed to be normal were not actually normally distributed. We believe that our decision tree model did the worst because we do not have rectangular decision bountaries between the two classes.