Analysis of Iowa Voting Trends to Predict Iowa's 2020 Election Results

Team members: Cory Dahn Peng Jiang Gabriel Pagat

1. Problem formulation:

The US election in 2020 is characterized by marked division between the Democrat and Republican parties. With each passing week, people's political convictions are strengthened. Therefore, for our project, we hope to develop an accurate model to predict this election's results. Specifically, we hope to predict the outcome of the 2020 election in Iowa as it pertains to the Presidency, Congressional seats, and Senatorial seats. We had planned to predict the outcome of the Gubernatorial election, but given a lack of data, our training and validation sets only spanned two years each. We focused on the Democrat and Republican parties, since they are the two most prominent parties in American politics. Furthermore, we segregated our results by county, since the entire state of Iowa does not vote in a uniform manner. However, given that the data we found for the Congressional elections in 2020 were segregated by district, we did the same with our past data.

2. Why predict election results?

This endeavor is important because both the Republicans and Democrats need to know where to focus their attention. Time and money are both limited resources for each party, so they cannot dedicate an equal amount of time to each county throughout the state. To make matters worse for them, there is too much data in the dataset to analyze manually. Therefore, we expect that both the Democrat and Republican parties would be interested in our findings. By identifying the counties in which their respective opponents are more popular, they will know where they should focus their campaign efforts in future elections.

3. Methodologies to solve the problem

3.1 Neural Network class regression

Our goal is to predict the 2020 results in Iowa as a state and separated by counties. In order to do that, we need some methodologies to do the regression and prediction. The Neural Network class works very well even though we don't really have testing data. So, for our training data to predict final presidential results in Iowa, we use the results from 1980 to 2016 presidential voting results. That is to say, we only have 10 training data and we are predicting 2020 results. We would say that the results are probably not gonna be very ideal because: first, we don't have much data to use, second, the prediction we acquire doesn't have a strong connection to our training data, third, the actual results have so many variations in the real world that could affect it. If we have a very good model, the regression curve is likely to go through all the points, which is the training data we have, to formulate a regression and extend from 2016 all the way to 2020. Thus, our thoughts would be that this prediction probably won't be able to judge the real world. But we will see how this is going because there are still some sort of connections.

3.2 Random Forest

The Random Forest regressor was another machine learning algorithm we used to predict the election results. Like the neural network, we would need to gather training and testing data from the datasets. Is voting data from elections past enough to accurately predict future elections? The group hypothesized no, the data is not enough for prediction. Elections have complex factors that determine the outcome, and while voting data may be a factor, we would need many more correlated inputs (which we don't have) to have a better prediction.

We decided to use Scikit-Learn's implementation of the Random Forest (RF) algorithm (Buitinck et al). We trained models to predict the outcomes of the 2020 Presidential, Senatorial, and Congressional elections.

Our models' hyperparameters were equivalent to the defaults assigned in Scikit-Learn's implementation, with two exceptions. We used a forest of 2000 trees each with a maximum depth of 10.

4. Dataset

1980 - 2016: https://www.kaggle.com/paultimothymooney/open-elections-data-usa

2020: https://www.kaggle.com/unanimad/us-election-2020

4.1 Data preparation methods

The election data was gathered from two different datasets spanning the years 1980-2020. The open-elections dataset (spanning the years 1980 to 2016) was very organized. The data was split by state, year of election, and type of election (presidential, sentate, house of representatives). So, if we wanted to look at data for a specific state, there was a distinct folder that contained all the election information gathered for that state. Inside the folders were .csv files that varied based on the type of election. The Presidential election .csv contained: office, candidate, party, county, and number of votes, the Senate and House election .csv files had similar columns. This dataset would be used to train the neural network/random forest.

The other dataset contained voting data for this years' election (2020). The main purpose of this dataset was to test how accurate the results from the neural network/random forest would be at predicting the election. The format of this dataset was a little different. Similar to the open-elections dataset, this dataset had .csv files corresponding to the different types of elections. However, unlike the open-elections dataset, each file contained data for all 50 states. This made each file around ~30,000 lines of raw data (no problem for the framework we utilized to parse the data). Election information for every county, in every state was included in the files. The data was easy to read and had similar columns to the open-elections dataset (state, county, party, candidate, number of votes).

After discussion, instead of using MapReduce or Spark, we decided to use pandas dataframes in python because this is the most efficient way to deal with our data. While we could have used MapReduce or Spark, we decided that doing so would have been unnecessary. Not only is pandas more intuitive, but we dealt with a lot of different files in different folders. Using pandas dataframe was much easier to edit, test, and see results.

Using pandas on the dataset was a breeze. Pandas natively supports the parsing of .csv files, so getting the data into a modifiable dataframe was a one-line operation. We ended up discarding many of the features included in the datasets, since they had nothing to do with total vote counts. The only features that we could use were party affiliation, county, and total vote count. While the office to which an election corresponded was relevant to our problem, we dropped that feature because our final curated data was split into several files based on office. We had to add a categorical variable corresponding to the year of the election when we consolidated the datasets into separate files based on office.

To use the collected data as inputs into the random forest, it had to be formatted in a special way. First we had to one-hot encode the data (turning all categorical data into numerical data, think 1s and 0s). Below are images of the before and after one-hot encoding.

President	Democratic	O'Brien	2170	2000
President	Republican	O'Brien	4674	2000
President	Democratic	Osceola	913	2000
President	Republican	Osceola	2064	2000
President	Democratic	Page	2293	2000
President	Republican	Page	4588	2000
President	Democratic	Palo Alto	2326	2000
President	Republican	Palo Alto	2341	2000
President	Democratic	Plymouth	3499	2000
President	Republican	Plymouth	6189	2000
President	Democratic	Pocahontas	1736	2000
President	Republican	Pocahontas	2242	2000
President	Democratic	Polk	89715	2000
President	Republican	Polk	79927	2000
President	Democratic	Pottawattamie	14726	2000
President	Republican	Pottawattamie	18783	2000
President	Democratic	Poweshiek	4222	2000
President	Republican	Poweshiek	4396	2000
President	Democratic	Ringgold	1246	2000
President	Republican	Ringgold	1369	2000
President	Democratic	Sac	2099	2000
President	Republican	Sac	2776	2000
President	Democratic	Scott	35857	2000
President	Republican	Scott	32801	2000
President	Democratic	Shelby	2179	2000
President	Republican	Shelby	3655	2000
President	Democratic	Sioux	2148	2000
President	Republican	Sioux	12241	2000
President	Democratic	Story	17478	2000
President	Republican	Story	16228	2000
President	Democratic	Tama	4045	2000
President	Republican	Tama	4034	2000
President	Democratic	Taylor	1247	2000
President	Republican	Taylor	1770	2000
President	Democratic	Union	2540	2000
President	Republican	Union	3003	2000
President	Democratic	Van Buren	1440	2000
President	Republican	Van Buren	2016	2000
President	Democratic	Wapello	8355	2000
President	Republican	Wapello	6313	2000
President	Democratic	Warren	9521	2000
President	Republican	Warren	9621	2000
President	Democratic	Washington	3932	2000
President	Republican	Washington	4827	2000

Figure 4.1.1 (before one-hot encoding)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	party	adair	adams	allamakee	appanoos	audubon	benton	black haw	boone	bremer	buchanan	buena vist	butler	calhoun	carroll	cass	cedar	cerro gorc	cherokee	chick
2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
4	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
5	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
6	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
7	1	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
8	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
9	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
10	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
11	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	
12	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
13	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	
14	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
15	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	
16	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
17	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	
18	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
19	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	
20	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	
21	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	
22	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
23	1	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	
24	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
25	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	
26	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
27	1	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	
28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
29	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	
30	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	
31	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	
32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	
33	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	
34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
35	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	
36	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
37	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
39	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
40	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
41	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	

Figure 4.1.2 (after one-hot encoding)

Once we did that, we just had to split that data into training and testing sets. The training data was all the data except for the 2020 election results, and the testing was the 2020 results. The training and testing data could now be used as inputs into the random forest.

5. Discussion and analysis

5.1 The dataset we are dealing with

Although this data set is more than 30 Gbs big, when we take a closer look into it, there is a lot of data that is incomplete. For example, it doesn't even have all the data for 50 states. Inside each state, we see a couple of folders containing different years for election. And that is incomplete too. For some states, it has about 20 folders corresponding to 20 different years. Some others only have one digit folders, or no folders at all. Also, inside the folders for each year, they have different csv files. Thus, it is very hard to utilize all of the data since everything seems broken and incomplete. Luckily, we found out that in Iowa, the data for the presidential

election is complete from 1980 to 2016. We retrieved the results from the 2020 election from another dataset.

5.2 Presidential Results

Now, we have all the data we need starting with the total number of votes for democrat and republican from Iowa from 1980 to 2016 as training data. The output regression for the votes to democrat and republican is kinda weird so that we realized this output doesn't mean anything. Because we are just plotting the total number of votes for each party. Even though we will be able to get a prediction for 2020, it is still meaningless because it has a strong connection to total votes with votes to democrat and republican sums up. And this number changes every year. Simply because in different years, there are different numbers of people voting and it has something to do with population too. But what we need is finding the relationship between the votes to democrat and replublican so that we can see who is winning. In order to do that, we can simply use the number of votes to republican minus the number of votes to democrat. If the number is positive, that means there are more votes toward republican and we know republican is winning this county. On the other hand, if the number is negative, similarly, the democrat is winning.

5.2.1 Neural Network

For the regression, we want the model to neither overfit nor underfit. The model we chose produces a fairly good looking curve, albeit it does not precisely pass through all the points. This was intentional, as an overfit regression would not look good on testing data. The same applies to an underfit model. Since we don't really have testing data, we can only make it visually neither overfitting nor underfitting. Based on these theories, we have the following graph produced:

```
run(total_ia_result_years_np, total_ia_result_np, 'sgd', 2000, 0.1)
```

```
NeuralNetwork(1, [50, 20, 20], 1)
```

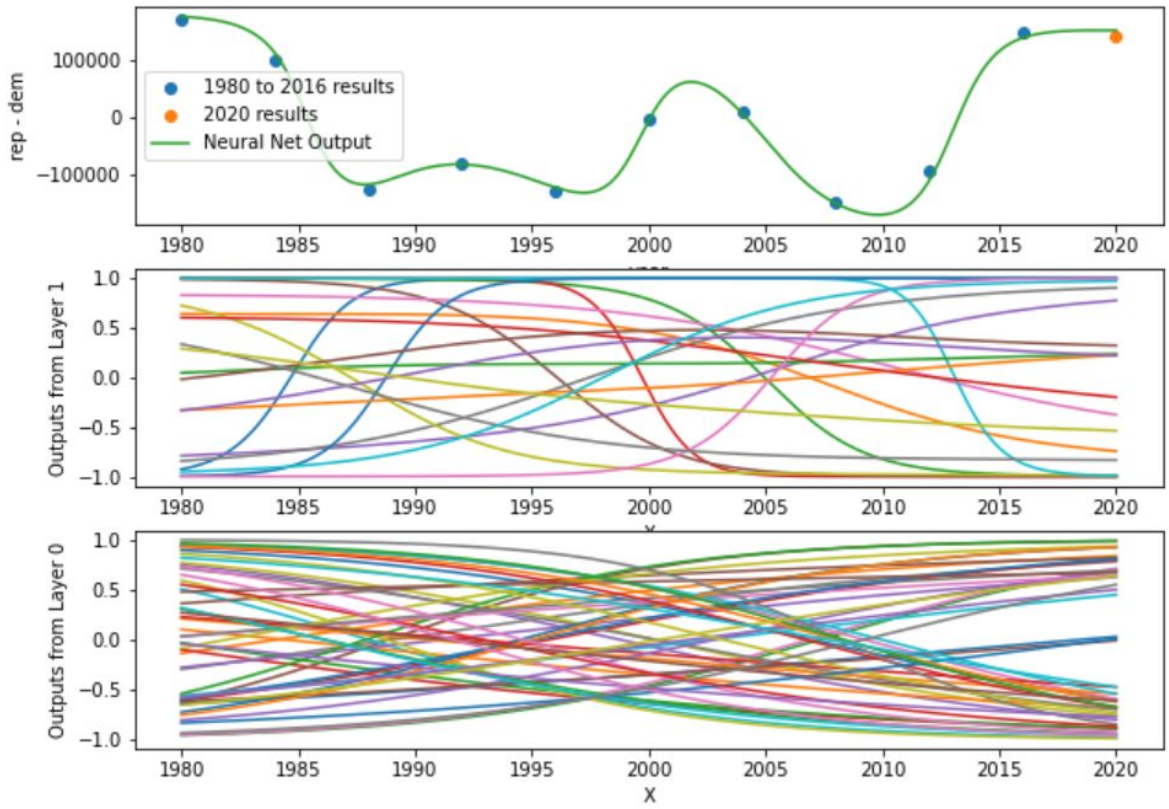


Figure 1

We have three graphs here in Figure 1. In the first graph, the blue dots represent the number of votes to republican minus the number of votes to democrat from 1980 to 2016. The green curve is the regression and we extend it from 2016 to 2020 so that the green curve appeared in 2020 would be our prediction. In 2020, our prediction says republican is winning by about a little bit more than 100000 votes. The orange dot represents the actual results which is very very close to our prediction. As we analysed before, our prediction shouldn't look this good, or be this accurate. So why does this happen? Clearly this is just lucky from my guess. Because if we only take data from 1980 to 2012 as training data and see if our prediction matches 2016, we will have:

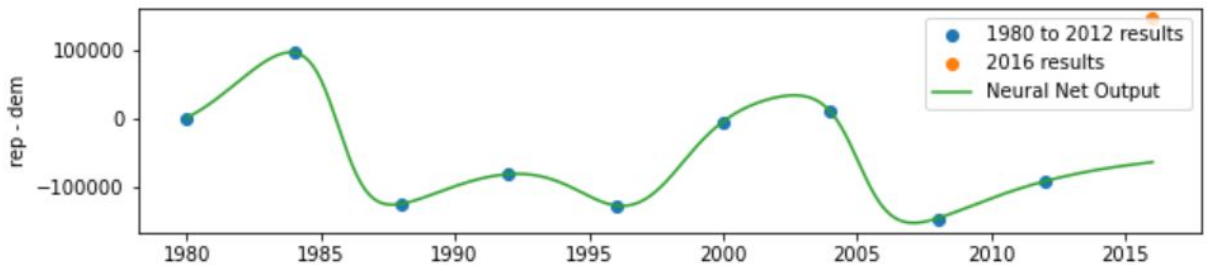
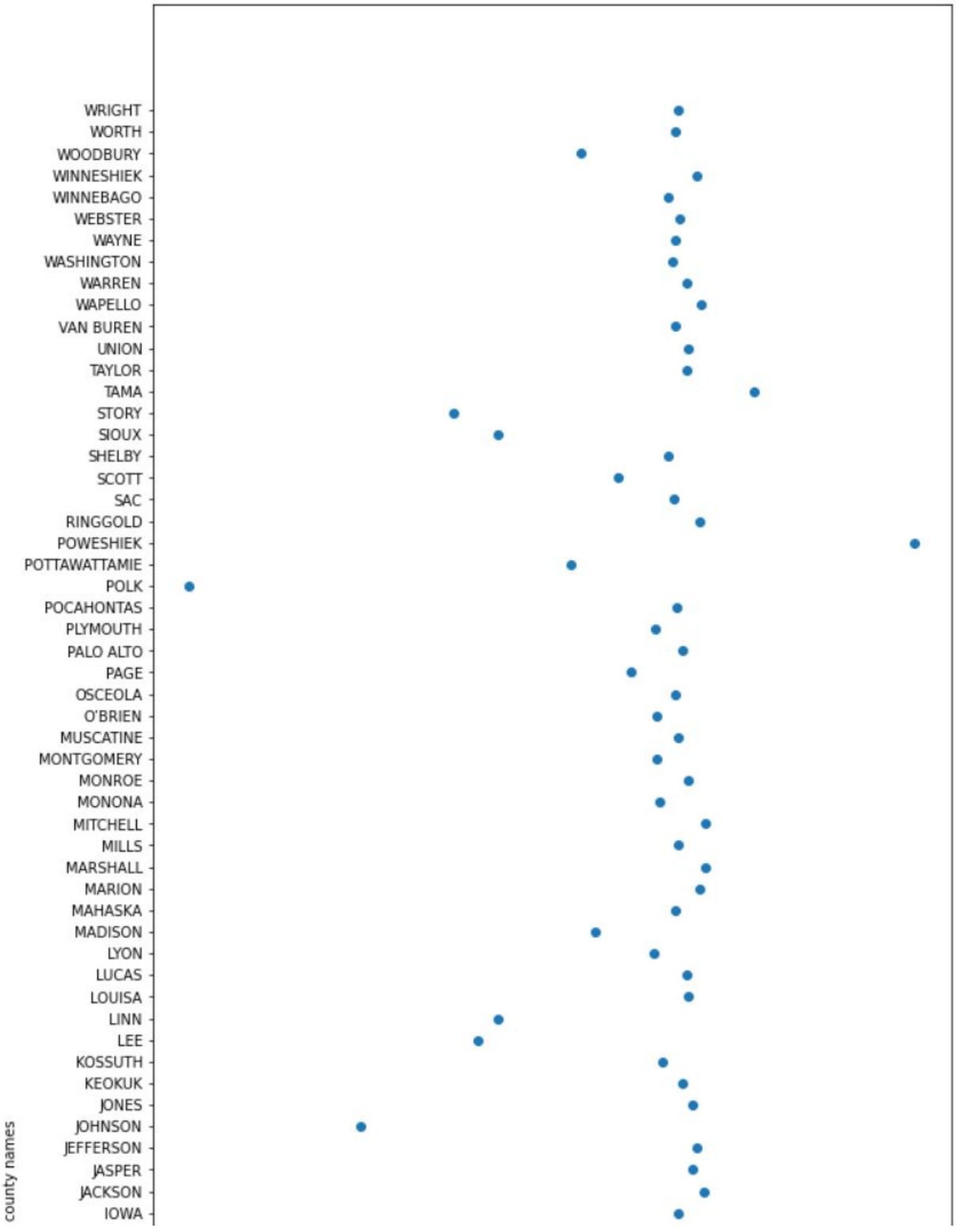


Figure 2

Clearly in Figure 2, our prediction thinks the democrat is winning but the truth is that republican is winning. And the gap between our prediction and the actual results is huge. As we discussed before, there are a lot of factors to affect election results. Simply by connecting all the results in the history isn't enough to predict. On the bright side, we can see the curve is going towards the actual results. It's just not as steep as we would like it to be. So we have a new thought now. Although the curve doesn't judge the actual prediction that much, it still could be somehow a reference. We can test the accuracy in general if we run it repeatedly. The only problem is that we don't have much data for different states. The data for Iowa is pretty much the only state that has a complete record in our data. So we are thinking about doing it by counties inside Iowa. Instead of plotting every county's regression like this, we will just get the results of our predictions and compare that to the actual results.


WRIGHT
WORTH
WOODBURY
WINNESHIEK
WINNEBAGO
WEBSTER
WAYNE
WASHINGTON
WARREN
WAPELLO
VAN BUREN
UNION
TAYLOR
TAMA
STORY
SIOUX
SHELBY
SCOTT
SAC
RINGGOLD
POWESHIEK
POTTAWATTAMIE
POLK
POCAHONTAS
PLYMOUTH
PALO ALTO
PAGE
OSCEOLA
O'BRIEN
MUSCATINE
MONTGOMERY
MONROE
MONONA
MITCHELL
MILLS
MARSHALL
MARION
MAHASKA
MADISON
LYON
LUCAS
LOUISA
LINN
LEE
KOSSUTH
KEOKUK
JONES
JOHNSON
JEFFERSON
JASPER
JACKSON
IOWA
county names

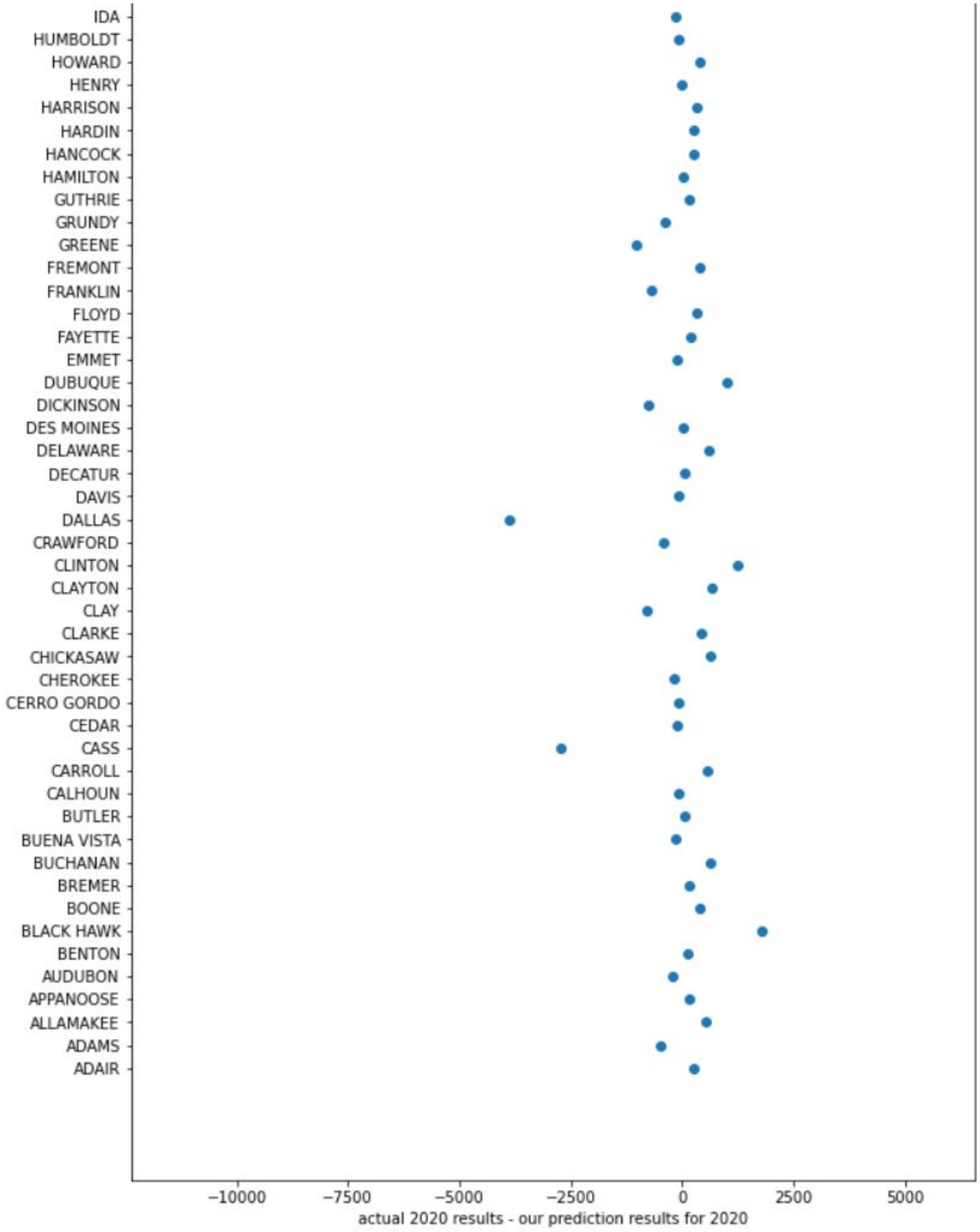


Figure 3

There are 99 counties in Iowa, that's why the graph looks long. The y axis is just the names of each county. The x axis is what we really care about. It is the actual results of each county

minus the results from our predictions. In other words, if the value is close to 0, that means the prediction is very close. If it is far away from zero, there is an obvious gap between the prediction to the actual result. The mean square error is 3694953.53595529. If we take the square root of that, we have about 1922, which is understandable given how voting habits vary widely from county to county.

Most counties' predictions closely resembled the actual results, as is the case with Greene, Ringgold, and Lee. However, given these counties' smaller population sizes with respect to larger counties like Polk according to the US Census Bureau, it becomes more likely for the predicted results to accidentally resemble the actual results more closely.

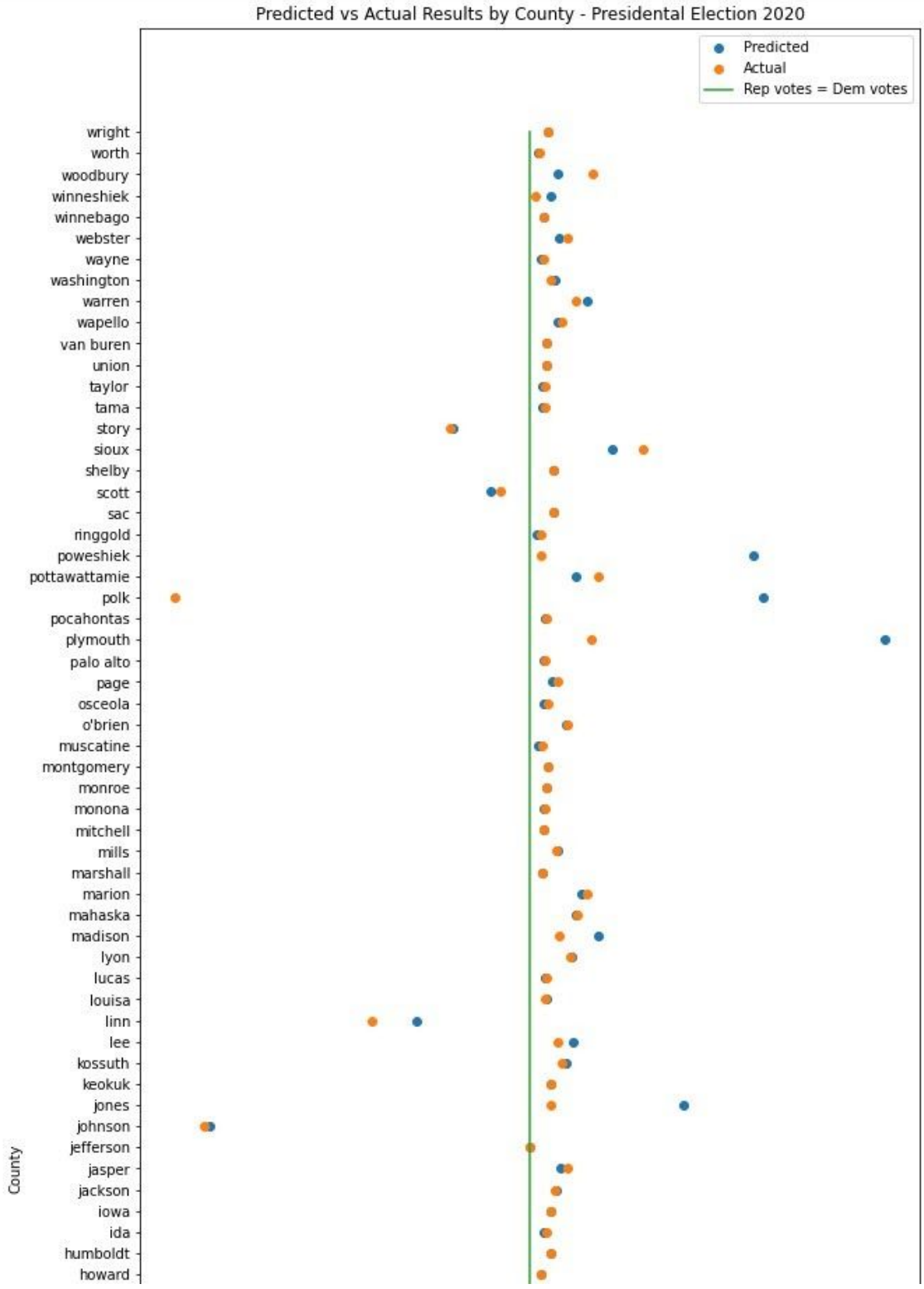
However, a few counties did differ in a way that discredited the model. The most notable example is Polk (population: 490,161), which was predicted to be strongly Republican but actually was even more strongly Democrat. A more subtle example is Dallas county (population: 93,453), in which the model predicted slightly more Democrat votes than Republicans. In reality, the opposite was true - there were slightly more Republican votes.

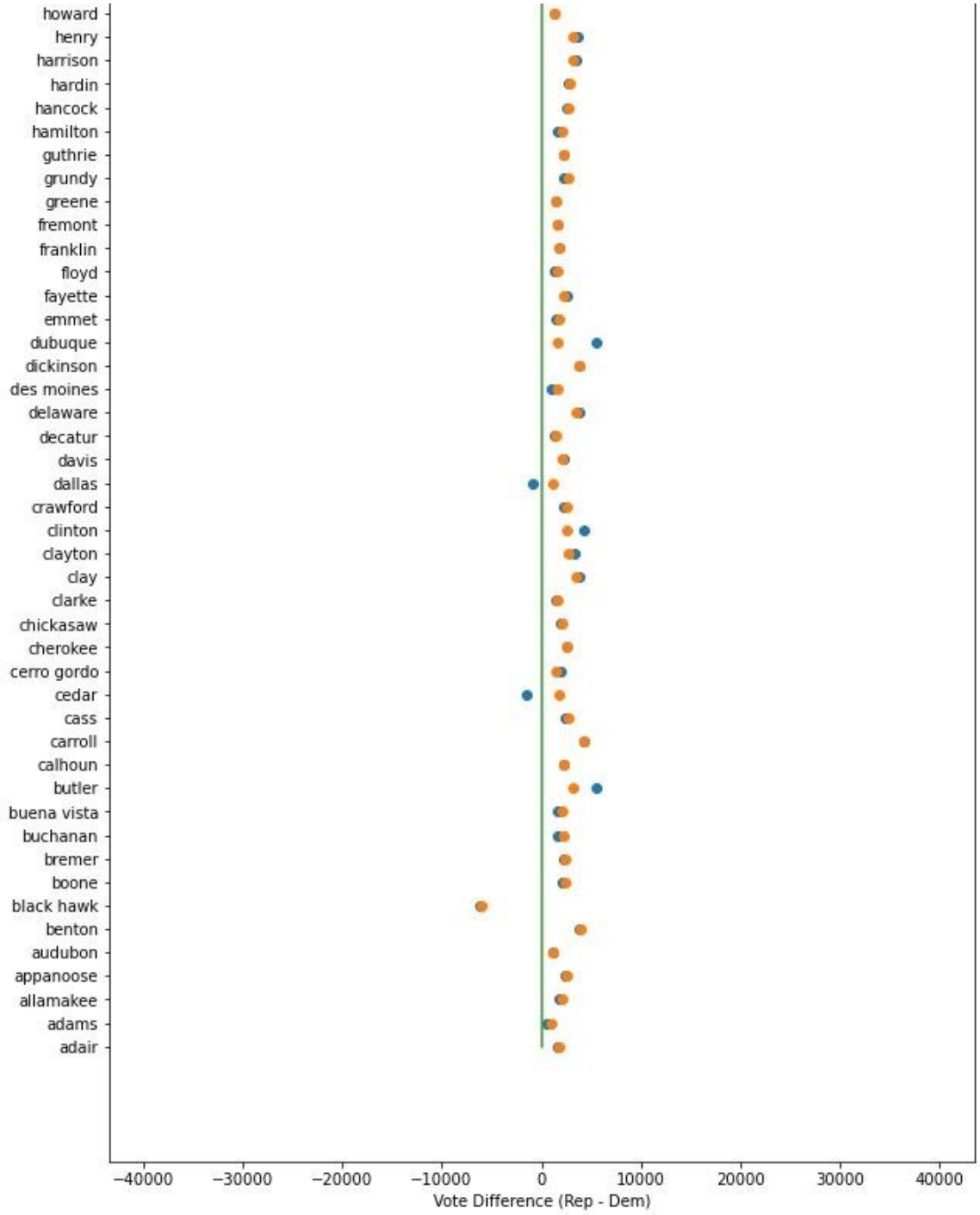
Our Neural Network is doing some kind of jobs but obviously not what we look for. The prediction for Iowa as a whole looked very good, but that was just luck. In the real world, predicting presidential elections doesn't work this way. This experiment is more like a proof that our initial thoughts were wrong.

5.2.2 Random Forest

Our RF model produced results similar to those produced by our Neural Network. The dots on the following graph correspond to the value calculated by subtracting Democrat votes from

Republican votes for that particular county, so the dots to the right of the line indicate a Republican victory while those to the left indicate a Democrat victory.





Our results' RMSE was 1389, which was better than that produced by our Neural Network. However, since the difference in votes for Democrats and Republicans are in many cases less than that, we cannot say that our model reliably predicts the outcome of the election.

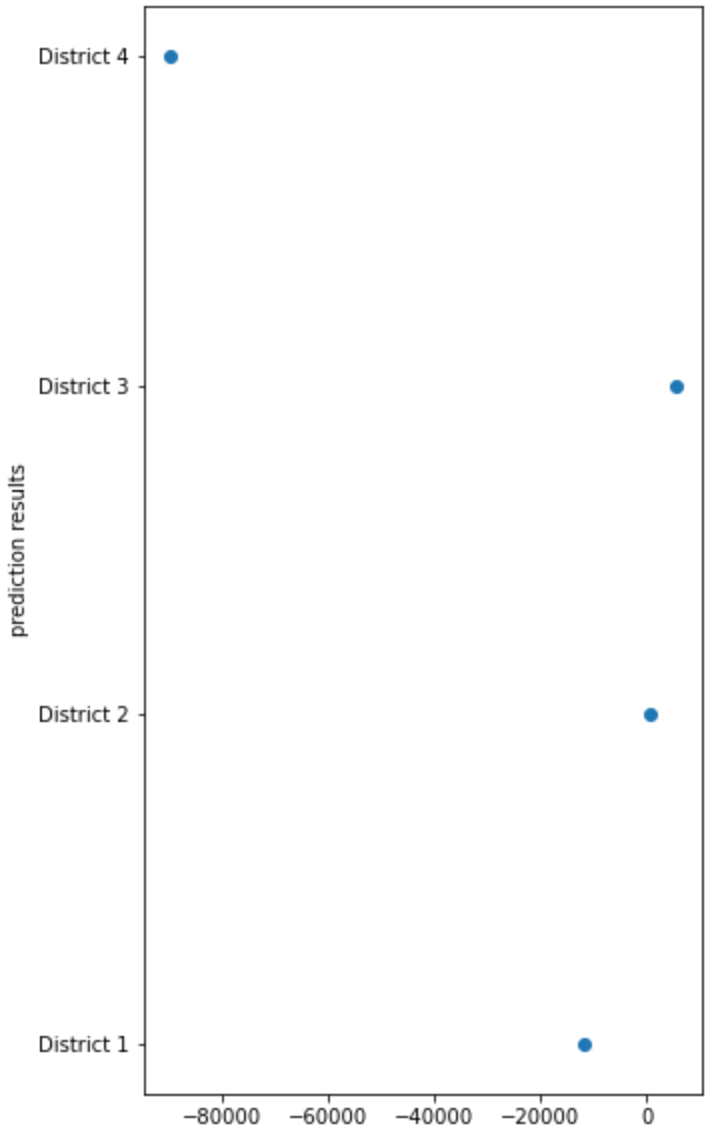
5.3 Congressional Results

Our models for this data did not predict the outcome well. We believe this can be explained by the lack of data. Since Iowa's Congressional districts changed in 2010, we had to drop the years before then. We could not use 2010's data because party affiliation wasn't recorded in our dataset. To make matters worse, the data we found for 2020 only recorded votes per candidate by Congressional district rather than by county, so we had no way to test whether a model trained on our data split up by county is accurate.

We expect that if our 2020 Congressional election results were split up by county, these models would have predicted results that more closely resemble reality. However, as discussed above, this data would have been misleading due to population differences between counties.

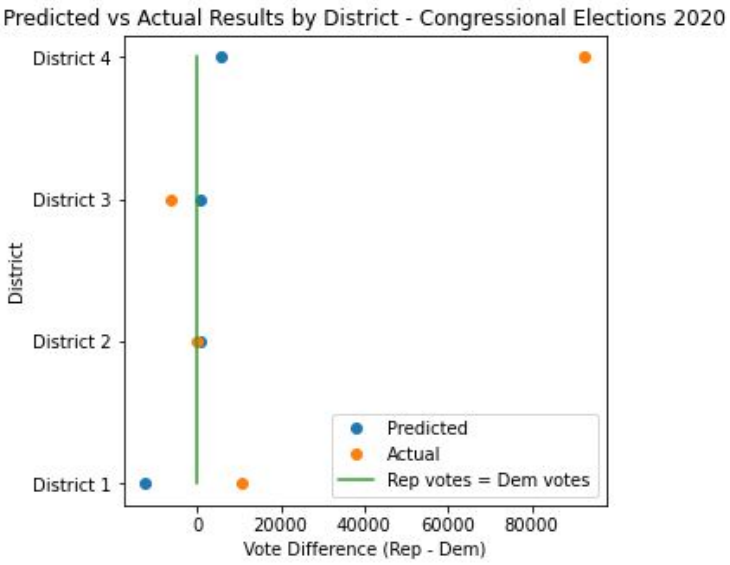
5.3.1 Neural Network

The RMSE produced is ~54,000, which reflects the unreliability of our model stemming from a lack of data. While the predictions of districts 2 and 3 may resemble the actual results, those of districts 1 and 4 are wrong. In the former's case, the predicted result was wrong, while the latter shows Democrats had a greater chance of winning than they actually did.



5.3.2 Random Forest

The RMSE produced is 46,233, which does not inspire confidence in the reliability of this model. While the predictions of districts 2 and 3 may resemble the actual results, those of districts 1 and 4 are wrong. In the former's case, the predicted result was wrong, while the latter shows Democrats had a greater chance of winning than they actually did.



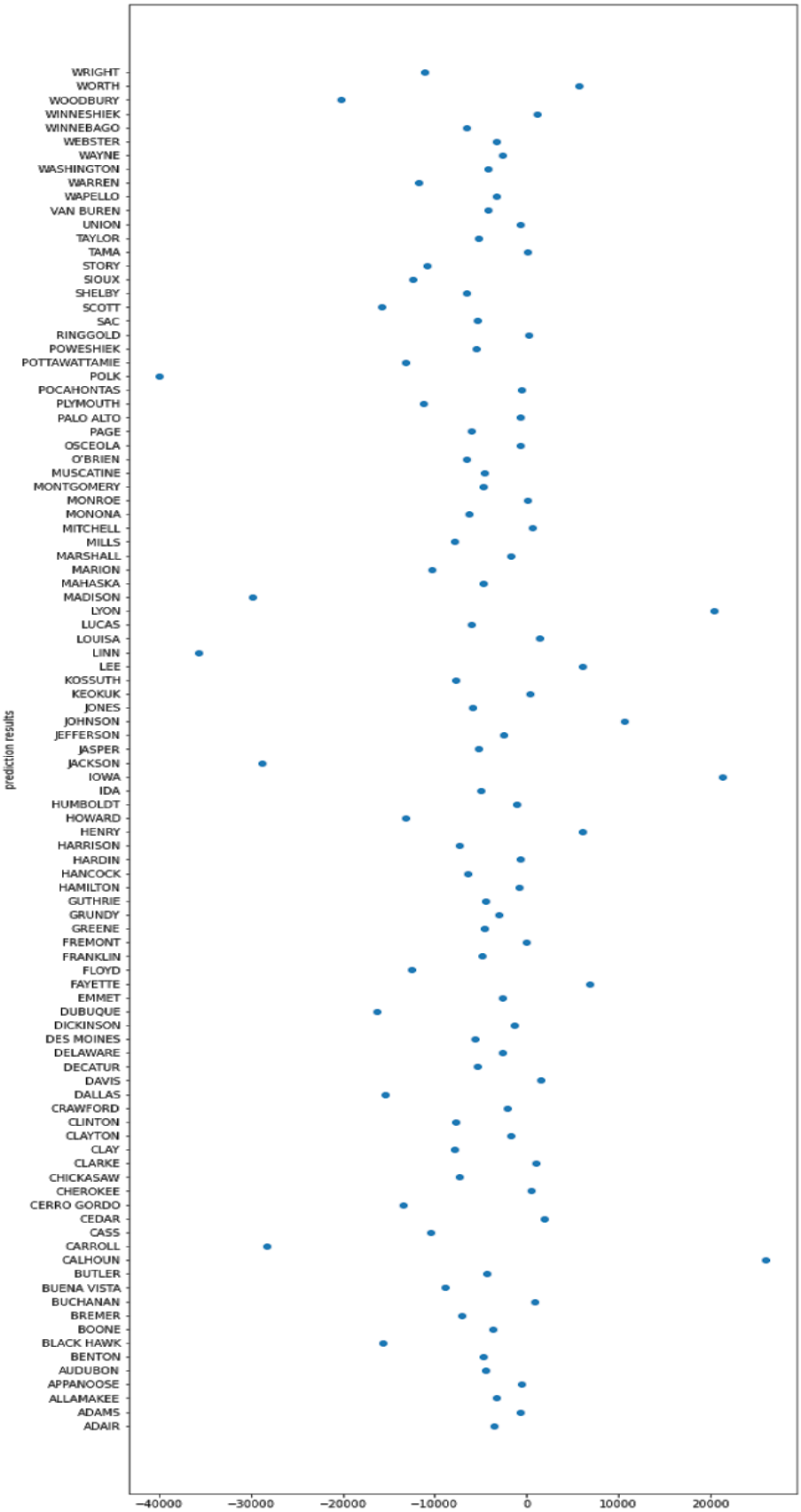
5.4 Senatorial Models

For this particular dataset, election data for some of the counties was missing. Therefore, we chose to ignore 2008. We only have data for the years 2004, 2014, and 2016, which makes these models unsuccessful. It predicted that the majority of counties would vote Democrat, when in reality they voted Republican according to NPR. This discrepancy is explained by a lack of data for certain parties in many counties.

One observation we noticed was that when the difference between Republican and Democrat votes was near zero in the real testing data, the model seems to predict pretty accurately. When that number was further away from zero, the model failed (pretty miserably in the cases of Linn, Polk, and Scott counties due to population differences).

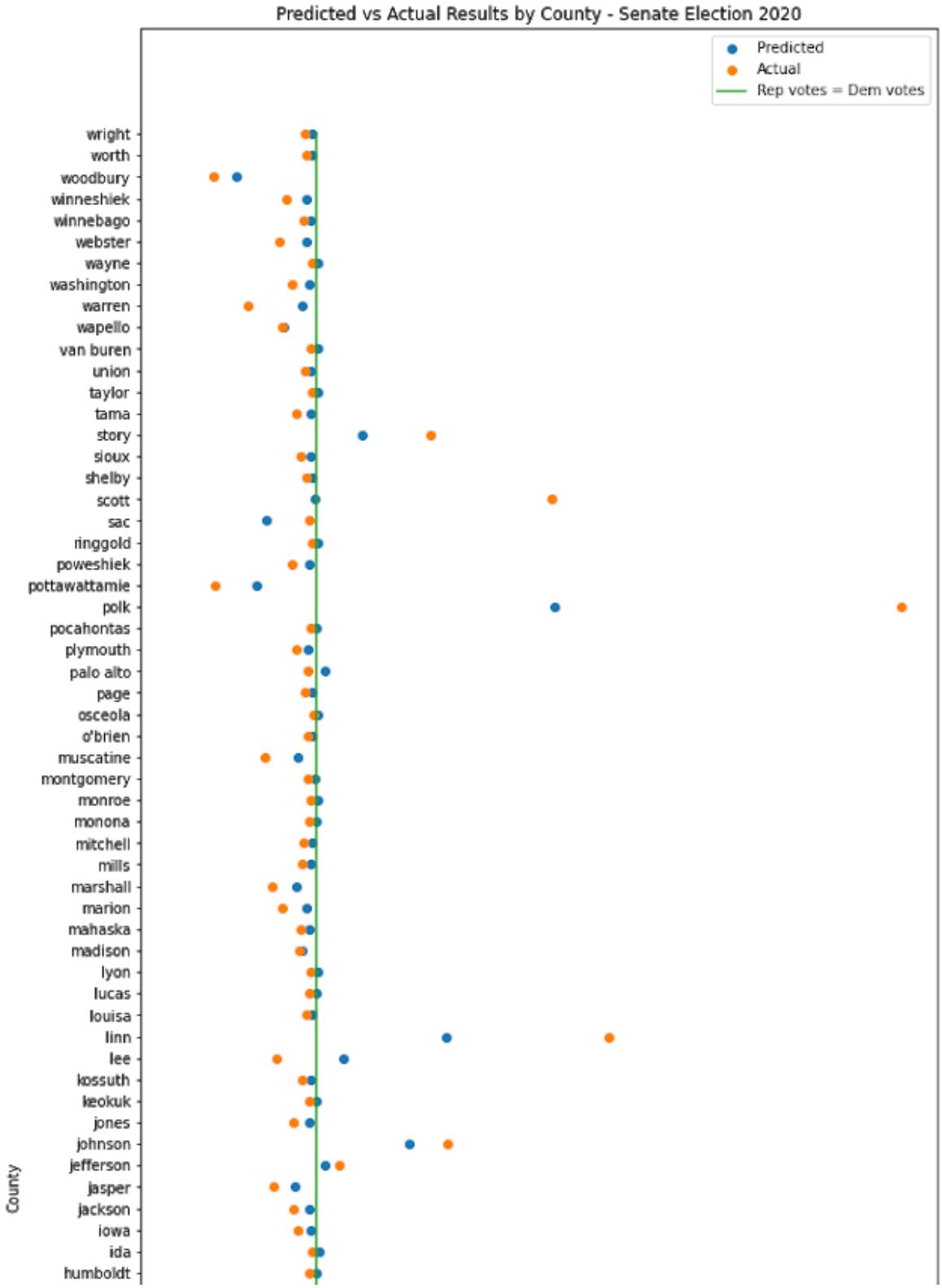
5.4.1 Neural Network

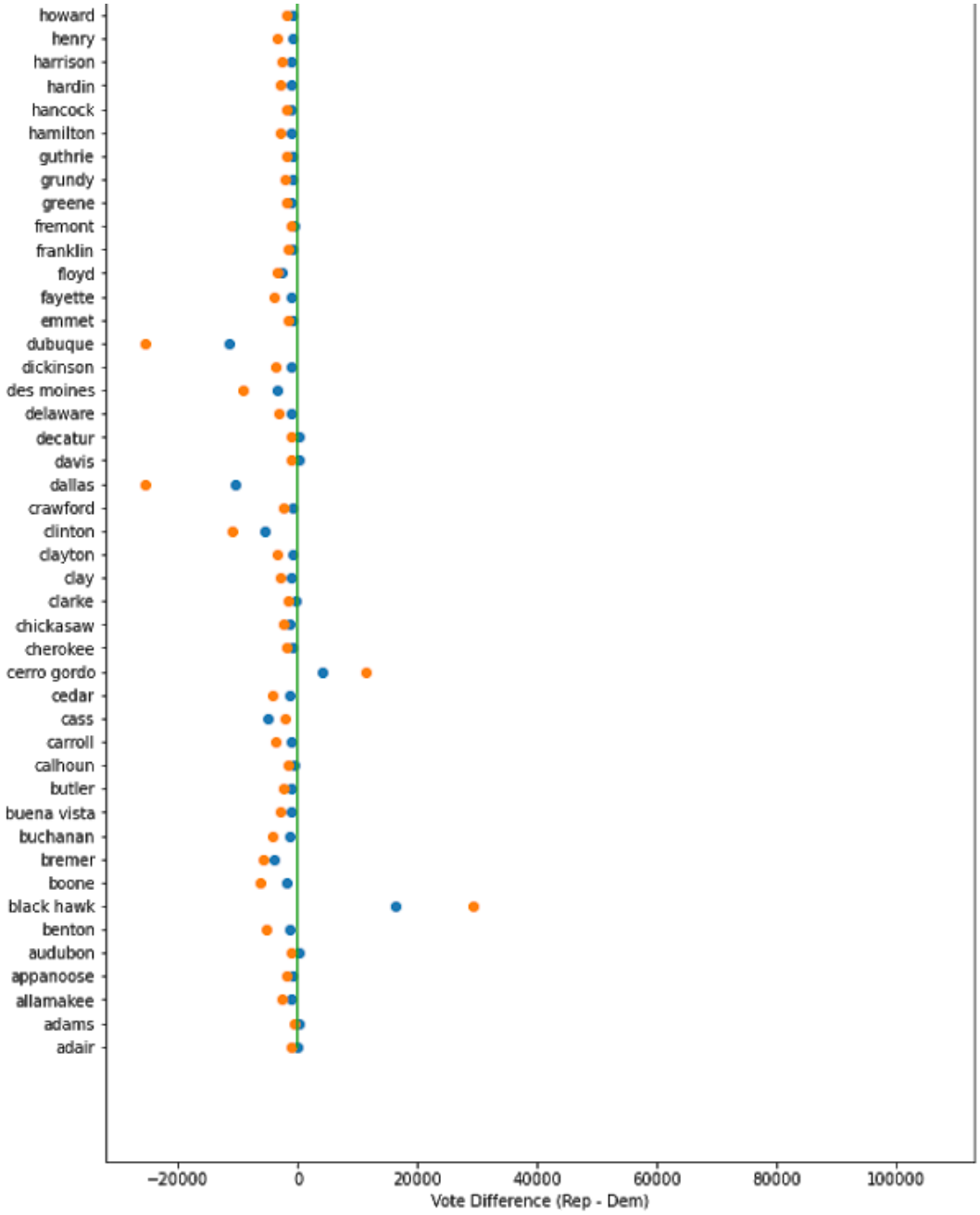
This model's RMSE was 10,298, but since our 2020 dataset is incomplete, this is meaningless.


prediction results
WRIGHT
WORTH
WOODBURY
WINNESHIEK
WINNEBAGO
WEBSTER
WAYNE
WASHINGTON
WARREN
WAPELLO
VAN BUREN
UNION
TAYLOR
TAMA
STORY
SIOUX
SHELBY
SCOTT
SAC
RINGGOLD
POWESHIEK
POTTAWATTAMIE
POLK
POCAHONTAS
PLYMOUTH
PALO ALTO
PAGE
OSCEOLA
O'BRIEN
MUSCATINE
MONTGOMERY
MONROE
MONONA
MITCHELL
MILLS
MARSHALL
MARION
MAHASKA
MADISON
LYON
LUCAS
LOUISA
LINN
LEE
KOSSUTH
KEOKUK
JONES
JOHNSON
JEFFERSON
JASPER
JACKSON
IOWA
IDA
HUMBOLDT
HOWARD
HENRY
HARRISON
HARDIN
HANCOCK
HAMILTON
GUTHRIE
GRUNDY
GREENE
FREMONT
FRANKLIN
FLOYD
FAYETTE
EMMET
DUBUQUE
DICKINSON
DES MOINES
DELAWARE
DECATUR
DAVIS
DALLAS
CRAWFORD
CLINTON
CLAYTON
CLAY
CLARKE
CHICKASAW
CHEROKEE
CERRO GORDO
CEDAR
CASS
CARROLL
CALHOUN
BUTLER
BUENA VISTA
BUCHANAN
BREMER
BOONE
BLACK HAWK
BENTON
AUDUBON
APPANOOSE
ALLAMAKEE
ADAMS
ADAIR
−40000
−30000
−20000
−10000
0
10000
20000

5.4.2 Random Forest

This model's RMSE was 3312. While this is better than the Neural Network's RMSE.





5.4 Conclusions

Both Neural Networks and Random Forests produce roughly equivalent predictions. The graphs for each respective election produced predictions that differed from the actual results by roughly the same amount. However, for these datasets, Random Forest's predictions are slightly better due to smaller RMSE values.

However, regardless of model, these predictions are not very meaningful. There are a lot of factors when it comes to elections. Simply by connecting history results together just won't work really well. The prediction for Iowa state looked very good but that was just luck.
When we consider population differences by county, it becomes apparent that our models do not adequately predict election results. For example, the population of Polk county is about 136 times greater than the population of Adams. Therefore, the seemingly miniscule difference between the predicted and actual election results for Adams county is actually quite significant.

In the real world, more goes into predicting elections than historical results alone. Our models' failure is evidence of this. Voters are likely influenced more by current events than past events. It is also likely that if we had month-by-month polling data that was complete, we could have trained more accurate models.

6. Project contributions

Cory Dahn:

- The majority of the data preparation
- Converted datasets with categorical variables to purely quantitative variables for use in Random Forest
- Wrote code for Presidential and Congressional Random Forest models and plotted corresponding graphs

Peng Jiang:

- Part of the data preparation to correctly feed Neural Network Class
- Implemented Neural Network Class
- Trained the dataset for the presidential election, plotted corresponding graphs, and analyzed the results.

Gabriel Pagat:

- Part of the data preparation for Neural Network
- Implemented the neural network with the Senate and Congressional elections to predict respective elections, and plotted the outcomes
- Wrote code for the Senate Random forest models (training/testing data), and plotted the outcomes

Bibliography

API design for machine learning software: experiences from the scikit-learn project, Buitinck et al., 2013.

NPR (2020). Election Results Iowa. Retrieved from https://apps.npr.org/elections20-interactive/?#/states/IA/S.

US Census Bureau (2019). QuickFacts Iowa. Retrieved from https://www.census.gov/quickfacts/fact/map/US/PST045219.