STS532 Final Paper: An Application of Dynamic Forecasting to the 2016 U.S. Presidential Election

Josiah Rottari

Spring 2024

1 Abstract

This paper builds on the ideas presented in "Dynamic Bayesian Forecasting of Presidential Elections in the States" by Drew A. Linzer through an application to the 2016 presidential election in the United States using the software PyStan with code and data included.

2 Introduction

Every citizen and corporation in the United States has a stake in the outcome of presidential elections and these stakes also extend to foreign citizens and corporations. Having accurate forecasts of elections allows stakeholders to create better plans for the future. Hence, accurate forecasts of elections are valuable.

There are a variety of approaches one may take to forecast an election such as a regression based on past elections, a weighted average of recent state-level polls, or a combination of these two methods. generally speaking, the model I focus on in this paper falls into the latter-most approach just described.

The 2016 election presents an interesting case study since the outcome was a surprise to many with various major news outlets, and a popular prediction market, seemingly getting the election outcome wrong [1], [2]. In this paper, I will apply the dynamic Bayesian forecasting model developed by Linzer to the 2016 U.S. presidential election to investigate how well it generalizes and, in the case it does not seem to generalize or perform well, I outline the approaches I take to improve performance, try to identify the possible issues given the particular software and data at my disposal and provide an overview of my future plans to troubleshoot the model. [3]. I provide commentary on the issues I run into with PyStan when building and fitting the model, aimed at anyone interested in using PyStan to build their own models. Details beyond those found in my paper on how I prepare the polling data to be used in the model, and how various plots are produced, are found in the accompanying Jupyter notebook.

3 The Model

Here, I briefly outline Linzer's model. Further details can be found in Linzer's paper and I note where my approach differs from those of Linzer throughout the paper [3]. Let $j = 1, \ldots, J$ be the day of the campaign where J is election day. Let the fifty states be indexed by $i = 1, \ldots, 50$. Let h_i be a long-term forecast of the share of votes won by a candidate in state i. Here, h_i is estimated using the time-for-change model of election results, as is done in Linzer's work, and the regression coefficients used for this paper are found in Alan I Abramowitz's paper cited here [4]. Now,

$$\pi_{ij} = \frac{\exp(\beta_{ij} + \delta_{ij})}{1 + \exp(\beta_{ij} + \delta_{ij})}$$

be the estimated proportion of votes a candidate would receive in state i on day j where

$$\begin{aligned}
\beta_{ij} &\sim N(\beta_{ij+1}, \sigma_\beta^2) \\
\delta_j &\sim N(\delta_{j+1}, \sigma_\delta^2) \\
\sigma_\beta^2, \sigma_\delta^2 &\sim \text{Uniform}(0, \infty) \\
\beta_{iJ} &\sim N(\text{logit}(h_i), s_i^2) \\
\delta_J &= 0.
\end{aligned}$$

Note that Linzer does not specify the interval over which $\sigma_\beta^2, \sigma_\delta^2$ are taken to be uniform, so I assign them an improper uniform prior on $(0, \infty)$. The variance s_i^2 is set by the analyst and affects how much weight is given to the informative prior, h_i, in each state. Linzer recommends setting $\tau_i = \frac{1}{s_i^2}$ no higher than 20 and this is what I do throughout the paper. Finally, let $y_{ij} \sim \text{binomial}(n_{ij}, \pi_{ij})$ where y_{ij} is the number of poll respondents in state i saying that they will vote for a particular candidate on day j. The βs are meant to track the trend of the $\pi_{ij} s$ at a state level and the δs are meant to track nationwide trends in the $\pi_{ij} s$. This allows estimation of π_{ij} on days when polling data is not available for a particular state. Figure 1 provides a visual illustration of how these various parameters connect. The k index in the figure denotes an arbitrary parameter with day index $1 < k < J$ and the dashed edges denote the various parameters between 1 and k and between k and J. Note how this figure is for one state and in reality there are fifty similar graphs all connected through the δ parameters to form a single graph representing the whole model.

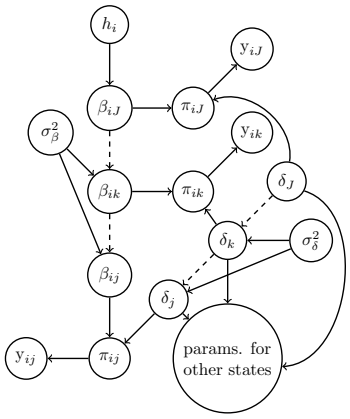

Figure 1: Graphical Structure of Model for State i With Its Connections to Other States

4 Fitting the Model

4.1 Overview

To apply this model, one must estimate thousands of parameters using the posterior distribution which may or may have a neat analytic form, and it likely it does not, so we estimate the posterior using MCMC in PyStan using the "sample" function once the model is fit. Linzer uses three MCMC chains with $200,000$ samples each with $100,000$ of the samples being burn-in samples. I do not have the time/computational resources to take this many samples, so one of the questions I investigate is: what's the smallest number of samples I can get away with for the model to make reasonable predictions?

4.2 The Data

The data I use to fit this model is published by FiveThirtyEight.com and is downloaded directly from the URL in the citation [5]. The head of the data is displayed in Figure 2. There are $12,624$ rows corresponding to different polls, some of which are duplicate entries. Note that we have raw poll data,

adjusted poll data, and ratings on the quality of different polls. Later, I discuss how using raw versus adjusted poll data and how filtering polls based on their rating impacts the model's performance.

cycle	branch	type	matchup	forecastdate	state	startdate	enddate	pollster	grade	samplesi	populatio	poll_wt	rawpoll_c	rawpoll_tr	rawpoll_jo	rawpoll_n	adjpoll_cl	adjpoll_tr	adjpoll_jo	adjpoll_m	multiversi	url	poll_id	question_	createddate	timestamp
2016	President	polls-plus	Clinton vs	11/8/2016	U.S.	11/3/2016	11/6/2016	ABC News	A+	2220	lv	8.720654	47	43	4		45.20163	41.7243	4.626221			https://ww	48630	76192	11/7/2016	11/8/2016 9:35
2016	President	polls-plus	Clinton vs	11/8/2016	U.S.	11/1/2016	11/7/2016	Google Co	B	26574	lv	7.628472	38.03	35.69	5.46		43.34557	41.21439	5.175792			https://da	48847	76443	11/7/2016	11/8/2016 9:35
2016	President	polls-plus	Clinton vs	11/8/2016	U.S.	11/2/2016	11/6/2016	Ipsos	A-	2195	lv	6.424334	42	39	6		42.02638	38.8162	6.844734			http://proj	48922	76636	11/8/2016	11/8/2016 9:35
2016	President	polls-plus	Clinton vs	11/8/2016	U.S.	11/4/2016	11/7/2016	YouGov	B	3677	lv	6.087135	45	41	5		45.65676	40.92004	6.069454			https://d2	48687	76262	11/7/2016	11/8/2016 9:35
2016	President	polls-plus	Clinton vs	11/8/2016	U.S.	11/3/2016	11/6/2016	Gravis Ma	B-	16639	rv	5.316449	47	43	3		46.84089	42.33184	3.726098			http://ww	48848	76444	11/7/2016	11/8/2016 9:35
2016	President	polls-plus	Clinton vs	11/8/2016	U.S.	11/3/2016	11/6/2016	Fox News/	A	1295	lv	5.218141	48	44	3		49.02208	43.96631	3.057876			http://ww	48619	76163	11/7/2016	11/8/2016 9:35
2016	President	polls-plus	Clinton vs	11/8/2016	U.S.	11/2/2016	11/6/2016	CBS News	A-	1426	lv	4.881873	45	41	5		45.11649	40.92722	4.341786			http://ww	48521	76058	11/7/2016	11/8/2016 9:35
2016	President	polls-plus	Clinton vs	11/8/2016	U.S.	11/3/2016	11/5/2016	NBC News	A-	1282	lv	4.836171	44	40	6		43.58576	40.77325	5.365788			http://ww	48480	75974	11/6/2016	11/8/2016 9:35
2016	President	polls-plus	Clinton vs	11/8/2016	New Mexi	11/6/2016	11/6/2016	Zia Poll		8439	lv	4.609492	46	44	6		44.82594	41.59978	7.870127			http://proj	48614	76158	11/7/2016	11/8/2016 9:35

Figure 2: Head of Polling Data

4.3 Data Challenges

As I mentioned above there are duplicates in the data. With many of my model fits, I made a significant mistake when dealing with the duplicates. Initially, I simply used the "drop.duplicates" function in Pandas. This function will drop a row if the entire row is the same as another row. I thought this was sufficient, but found that some of the decimal places in presumably duplicate rows were different, so the function was not dropping as many duplicates as it should have. Due to this, I fit the model countless times on data that had duplicates, which made it seem like there was much more data than there was. This lead to the model giving the data much more weight than it should have received. I fixed this error by filtering just on the "poll_id" column.

4.4 Modeling Challenges: Election Differences

Linzer's focus on the 2008 election allowed for more convenient estimation of Obama's probability of winning since there was no significant third-party candidate that year. They estimated Obama's chances of winning by taking one sample from the posterior, for each state, on election day and awarding Obama a state's electoral votes if the sample of π_{iJ} was greater than 0.50. Then they summed these electoral votes and repeated this process to estimate the distribution of electoral votes. The 2016 presents a challenge because, in advance of election day, there was reason to believe third-party candidates would receive enough votes for the 0.50 threshold to not work. To work around this, I fit the model for both Clinton, and for Trump, then follow a similar sampling procedure where I sample from both posterior distributions on election day and award electoral votes to the candidate whose sample of π_{iJ} is larger.

This approach also leads to an additional challenge of setting a prior for the non-incumbent party. The time-for-change model used to estimate h_i is based on election results for the incumbent party, and in 2016 Clinton was a member of the incumbent party [4]. Hence, setting a prior for the model in the non-incumbent party takes some additional care. I take a lazy approach, for convenience, and set the prior in the model for Trump as the complement of Clinton's prior and acknowledge that there is likely a better approach.

4.5 Modeling Challenges: PyStan

The main challenges I faced with PyStan were getting the syntax correct, determining how to format the polling data for the model to use, fixing $\delta_J = 0$ as the paper does and, generally, not having a lot of available examples to work from on the internet and deciphering how the pretty thorough documentation for Pystan 2 translates to PyStan 3 (the version I use) since the PyStan 3 documentation is not as thorough.

For syntax, I believe earlier versions of PyStan have you declare what type of data an object contains eg. int, real, etc., and then you declare the type of the object eg. int, array, matrix. In may case, I found that it was the opposite; you declare the type of object, what kind of entries it has real, int, etc. then give it a name. Please see the accompanying Jupyter notebook to observe the latest syntax for Pystan.

For getting the data formatted correctly, I had issues conceptualizing what shape my polling data should be and how to translate it to PyStan. Since I was using data for up to 112 days of the campaign, for 50 different states, and this data had both the size of the sample and the number of respondents saying that they would vote for a particular candidate, I knew that I wanted the data to be some sort of three-dimensional object. Initially, I thought I could use an array of length fifty where each entry of

the array was a 112×2 matrix with the proper data. Conceptually, this makes sense, but the problem with this approach is that matrix objects in the PyStan language only have real or complex entries. This is a problem since the data given to the PyStan binomial distribution must be an integer. So, when I was using this approach, I would get errors from the binomial distribution note knowing how to deal with real type entries even though these entries were integers. My solution was to make a $50 \times 112 \times 2$ array of data since arrays can have any data type inside them.

My third biggest challenge was with deterministically setting $\delta_J = 0$. I stored my delta parameters in an array called deltas and attempted to set this entry equal to zero using deltas

1

$= 0$ (the index in the model as coded in PyStan is days until election day) in various parts of the code. I tried doing this in the "parameters" section, in the "model section", and in the "transformed data" section. I believe there's a way to do this in the "transformed variables" section, but it seemed overly complex and inefficient because I believe it would force the code to sort of jump between the "model" and "transformed data". The former two approaches did not work. Here, the solution was to set the initial conditions for the deltas equal to zero when I sampled from the posterior.

I also made the mistake of setting $\delta_J \sim N(0, \sigma_\delta^2)$ in the PyStan model, which I believed stemmed out of frustration with my problems above, but this is simply not the model described by Linzer and led to plots as those shown in Figure 3.

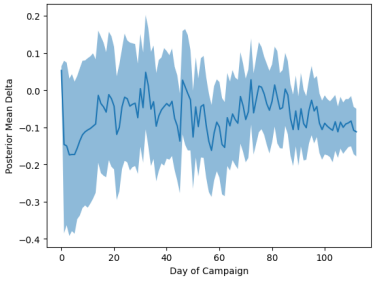

Figure 3: Plot of the Mean Delta Estimates

Linzer's model implies that the deltas need to be anchored to 0 on the day of the campaign, and that's what his plots show, and this figure does not demonstrate this.

5 Preliminary Results and Model Troubleshooting

Initially, I fit the model with 4 chains, and $1,000$ samples with 500 serving as burn-in.

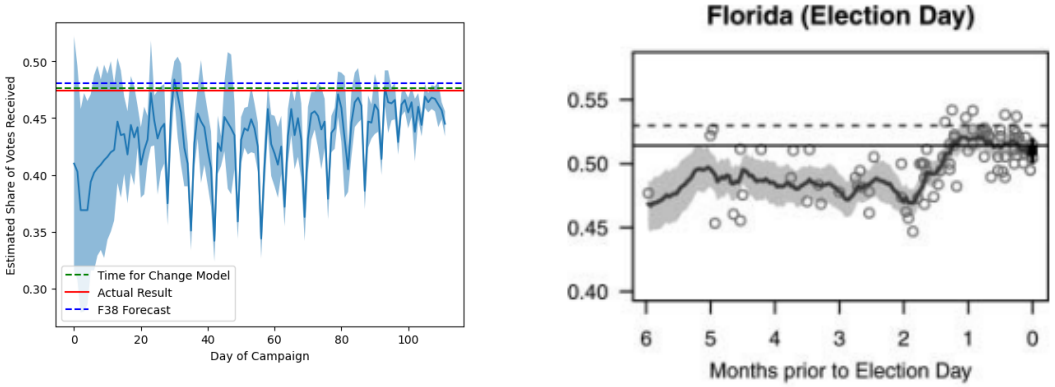

Figure 4: 2016 Model Fit With Inproperly Filtered Data and Improperly Set δ_J Versus 2008 Model Fit for Florida [3]

Figure 4 juxtaposes the initial, incorrect, fit of the model using data with duplicates, and without δ_J anchored, on the 2016 election in Florida, with the model fit by Linzer in Florida for the 2008 election. I present this figure since it provides some motivation for how I went about catching my mistakes. Solid, horizontal lines represent the actual outcome, the green dashed line on the left and the dashed line on the right represent the prior forecast using the time for change model. The jagged solid lines represent the mean of the $\pi_{ij}s$ drawn from the posterior, call it $\hat{\pi}_{ij}$. Linzer uses the proportion of posterior draws of π_{iJ} higher than 50% as their estimate of π_{iJ}, but does not say explicitly that this is what they do for π_{ij} in general, but I believe this is implied. I use the mean for convenience rather than drawing samples from the posterior and determining what proportion of them lie above 50%. The shaded region on the left is a 95% HPD credible interval for π_{ij} and the shaded region on the right is a 90% HPD credible interval. The dots on the figure on the right represent actual polling data. Another difference in how this plot was fit is that here I estimate β_{ij} for every day whereas Linzer estimates β_{ij} using $\beta_{it[j]}$ where t is the time-period containing day j and $t = 0, \ldots, J/3$. I use a similar process later since it is what Linzer does and it saves computation time when doing MCMC.

Clearly, the figure I produced for the 2016 exhibits higher day-to-day variance, and the 95% HPD interval does not cover the true election result and is narrower than that of Linzer's. The observations about the narrower intervals and higher variance held for all 50 states, hence why I only show one figure. My initial hypothesis for why this was the case was that my simulation had not converged to the posterior. Figure 5 shows the distribution of $\hat{R}$ for my initial, flawed in various ways, simulation with the spike at 1 being an artifact of the $\pi_{ij}s$ being computed inside one chain since they are computed in the "generated quantities" block of the PyStan model. Linzer used $\hat{R} \approx 1$ to judge convergence as well as plots to ensure the parameters were mixing, and we see that the samples do not meet the $\hat{R} \approx 1$ criterion.

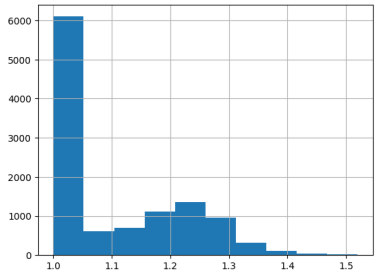

Figure 5: Distribution of the Gelman-Rubin Statistic $\hat{R}$ Using 4 Chains and 500 Samples

From here, still not realizing I coded the model wrong and was using bad data, I thought taking more posterior draws and estimating the βs using $\beta_{it[j]}$ where $t = 0, \ldots, \frac{112}{4}$, similar to Linzer, would fix the variance and the low coverage issues and they did not. When I took these steps, the posterior seemed to converge, but the plot produced looked almost identical to Figure 4 with all the same issues.

My initial fits, still with the duplicated data and incorrect model, were done using the "raw_poll" data, so I thought using the adjusted poll data might fix things. It did not. I then thought the issue might be with using low-quality polls, so I filtered the poll data to have a grade of at least a B. This also did not fix my issues. Finally, and after too long, I went back to check if the duplicates were being dropped correctly and simultaneously noticed my error in specifying the δ_J value. I now turn to my actual results and keep with using the adjusted poll data from polls with a grade of at least B.

6 Results

Since I caught many of my major mistakes late in the process, in the interest of time, all the results presented going forward are based on 1000 posterior samples with 500 burn-in and 3 chains – far short of Linzer's 200,000 samples with 100,000 burn-in with 3 chains. I seemingly got reasonable enough results and convergence based on these smaller sample numbers, and I note that a larger sample size is likely more appropriate. I somewhat justify this claim of convergence in Figure 6 with the distributions of $\hat{R}$ for the 16-week Clinton and Trump models fit as I described above, but $\hat{R}$ does vary for the 4, 8, and 12 week forecasts. Sometimes it's generally higher and sometimes it's generally lower. Going

forward, when I say "16-week" model, I mean the model fit with all the election data from day 0 of the campaign through election day, ie. the forecast that would be available on election day. For example, a 4- week model is a model made earlier in the campaign using the first four weeks of data.

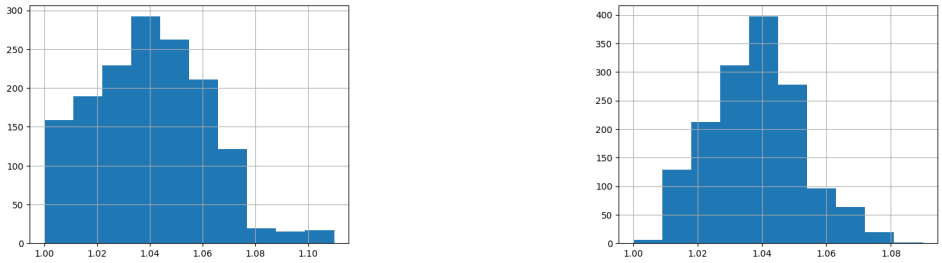

Figure 6: Clinton $\hat{R}$ Distribution (Left), Trump $\hat{R}$ Distribution (Right)

For the sake of completeness, Figure 7 provides the same comparison as Figure 4 and shows how I was able to remedy some of the variance issues, but the plot is still not as smooth as Linzer's and the coverage is much lower.

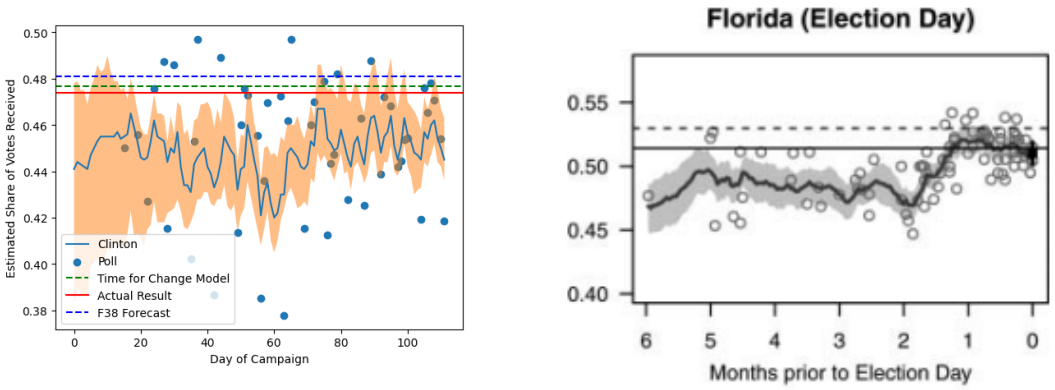

Figure 7: 2016 Model Fit Versus 2008 Model Fit for Florida [3]

In Figure 8, I present the coverage probabilities of the 97% HPD intervals for the π_{ij} across all 50 states for the Clinton and Trump models fit with all 16 weeks of data and juxtapose them with the coverage probabilities present in Linzer's paper. In my case, the coverage is quite poor.

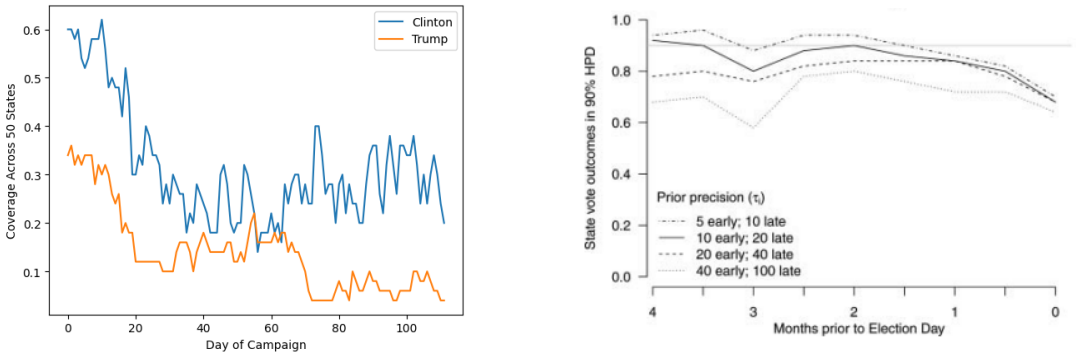

Figure 8: 16 Week Model Nominal Coverage Rates for Clinton and Trump in 2016 Versus Linzer's, 16 Coverage Rates for Obama in 2008[3]

Figure 9 shows the estimated distribution of Clinton's electoral votes created using 10,000 samples with replacement from both posterior distributions with the process described in Section 4.4. This model gives Clinton a 79% chance of winning on election day with 281 electoral votes expected.

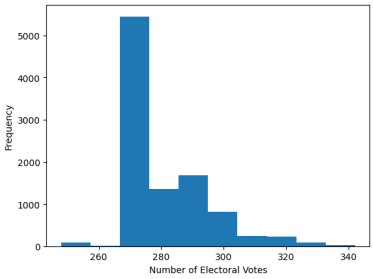

Figure 9: Distribution of Clinton Electoral College Votes: 16 Week Model

On election day, Clinton was given a 85% [6], a 90% [7], 82% with 4% bookie edge [2], 91% [8], and 71.8% [9] chance of winning by various news outlets and prediction markets on election day. The 71.8% is from FiveThirtyEight and was intentionally chosen. The other figures were arbitrarily chosen from the first page of Google with the search "Clinton probability of winning 2016". At a minimum, the forecast produced from this model falls in line with other widely circulated forecasts from 2016.

I also fit the model with $4, 8$, and 12 weeks of data respectively, repeat the computation of coverage probabilities by day and the estimation of clinton's probability of winning on election day. I compare the forecasts I produce for Clinton's chances in 2016 with New York Times' forecasts and with FiveThirtyEight's forecasts (keeping in mind that Clinton lost).

Figures 10 and 11 display the nominal coverage rates for these different forecasts, where I display the model fit with 16 weeks worth of data again for easier comparison with the other fits. One thing to note is that the coverage rates differ significantly in the earlier periods which I don't believe should happen since all these models were fit with at least four weeks of data. This is probably a result of the posteriors not converging. Moreover, the coverage rates in all of them are not really anywhere near the 95% nominal coverage rate that we should expect. I do not have an explanation for this.

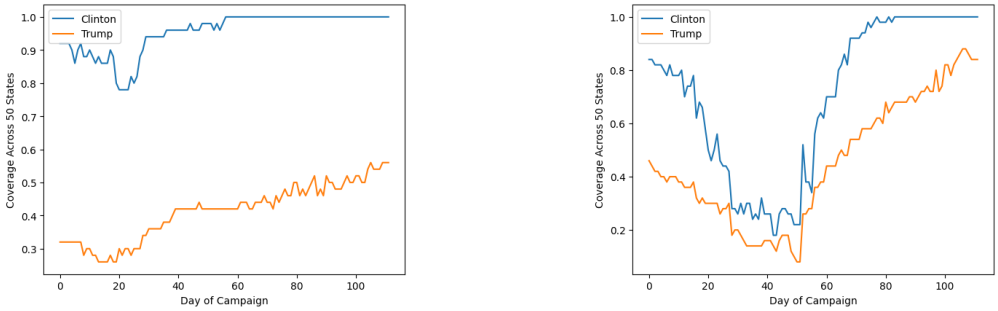

Figure 10: 4 Week Nominal Coverage Rates (Left) and 8 Week Nominal Coverage Rates (Right) for Clinton and Trump in 2016

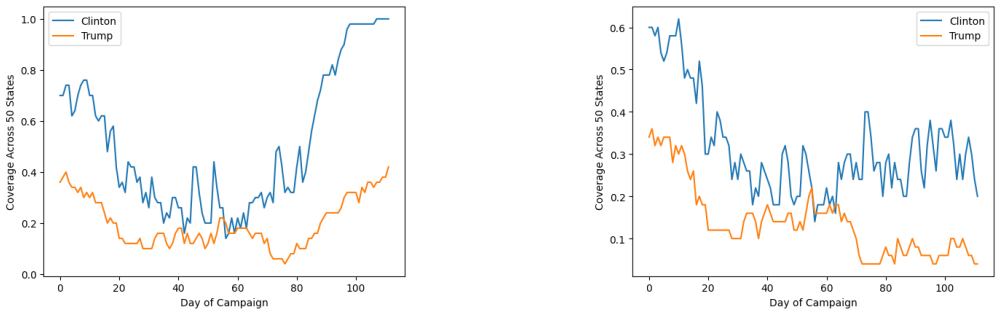

Figure 11: 12 Week Nominal Coverage Rates (Left) and 16 Week Nominal Coverage Rates (Right) for Clinton and Trump in 2016

I don't believe there is much to deduce from Table 1 given the previous analysis that I have presented on the model in 2016. If one was blindly gambling using Linzer's model based on my fit, one would have done better than using NYT's forecast and perhaps better than FiveThirtyEight, but I do not condone Russian roulette.

Table 1: 2016 Clinton Forecast Comparisons

Weeks Until Election	NYT	FiveThirtyEight	Linzer's Model
12	83%	83.4%	50%
8	83%	67.3%	72%
4	82%	81.8%	57%
0	85%	71.4%	79%

7 Conclusion

Although I have failed to replicate Linzer's clean and seemingly excellent results for the 2016 election I remain optimistic about this model's general applicability and I have not exhausted all means of troubleshooting my results. At this point, believing that there are no glaring mistakes in my code I think it's reasonable to try more samples when estimating the posterior and then estimate each π_{ij} as the proportion of π_{ij}s from Clinton's posterior that are higher than those in Trump's posterior. I acknowledged earlier that Linzer did something similar for π_{iJ}, but does not explicitly say how he estimates π_{ij} for $j < J$. I did this once and the variance was actually higher, but I did not use that many samples and the posteriors likely had not converged. Another reasonable next step is to use data from 2008 and from the same source as my 2016 data. If the variance in the model remains, then there is still the chance that I have made a mistake in my code or that the data from the source is of low quality. But if the model works well, I could reasonably deduce that there's either an issue with the 2016 data or something fundamentally different about the 2016 election compared to the 2008. Replicating my analysis using 2012 data also seems reasonable. Another thing I might try, and I am apprehensive about it, is simply not including polls with unreasonable results based on past elections. Finally, if none of this works, I intend to reach out to Linzer to see if I can get his data. Assuming a response, this would be the quickest way to troubleshoot my work, but might rob me of the fun of trying to reproduce the results on my own.

References

[1] "2016 presidential election forecast maps - 270towin," May 2016. [Online]. Available: https://www.270towin.com/2016-election-forecast-predictions/

[2] "Predictit." [Online]. Available: https://www.predictit.org/markets/detail/1234/Who-will-win-the-2016-US-presidential-election

[3] D. A. Linzer, "Dynamic bayesian forecasting of presidential elections in the states," *Journal of the American Statistical Association*, vol. 108, no. 501, pp. 124–134, 2013. [Online]. Available: https://doi.org/10.1080/01621459.2012.737735

[4] A. I. Abramowitz, "Forecasting the 2008 presidential election with the time-for-change model," *PS: Political Science and Politics*, vol. 41, no. 4, pp. 691–695, 2008. [Online]. Available: http://www.jstor.org/stable/20452296

[5] May 2024. [Online]. Available: https://projects.fivethirtyeight.com/general-model/president_general_polls_2016.csv

[6] "2016 election forecast: Who will be president?" May 2016. [Online]. Available: https://www.nytimes.com/interactive/2016/upshot/presidential-polls-forecast.html

[7] "Clinton has 90 percent chance of winning: Reuters/ipsos states of the nation," Nov 2016. [Online]. Available: https://www.reuters.com/article/idUSKBN1322J0/

[8] "Political prediction market: Clinton's odds rise again — cnn politics," Nov 2016. [Online]. Available: https://www.cnn.com/2016/11/07/politics/political-prediction-market-hillary-clinton-donald-trump/index.html

[9] "2016 election forecast," Nov 2016. [Online]. Available: https://projects.fivethirtyeight.com/2016-election-forecast/