Prediction for the 2020 United States Presidential Election using Machine Learning Algorithm: Lasso Regression

Sinha, Pankaj and Verma, Aniket and Shah, Purav and Singh, Jahnavi and Panwar, Utkarsh

Faculty of Management Studies, University of Delhi

13 October 2020

PREDICTION FOR THE 2020 UNITED STATES PRESIDENTIAL ELECTION USING MACHINE LEARNING ALGORITHM: LASSO REGRESSION

Pankaj Sinha *Aniket Verma* *Purav Shah* *Jahnavi Singh* *Utkarsh Panwar*
Faculty of Management Studies
University of Delhi

ABSTRACT

This paper aims at determining the various economic and non-economic factors that can influence the voting behaviour in the forthcoming United States Presidential Election using Lasso regression, a Machine learning algorithm. Even though contemporary discussions on the subject of the United States Presidential Election suggest that the level of unemployment in the economy will be a significant factor in determining the result of the election, in our study, it has been found that the rate of unemployment will not be the only significant factor in forecasting the election. However, various other economic factors such as the inflation rate, rate of economic growth, and exchange rates will not have a significant influence on the election result. The June Gallup Rating, is not the only significant factor for determining the result of the forthcoming presidential election. In addition to the June Gallup Rating, various other non-economic factors such as the performance of the contesting political parties in the midterm elections, Campaign spending by the contesting parties and scandals of the Incumbent President will also play a significant role in determining the result of the forthcoming United States Presidential Election. The paper explores the influence of all the aforementioned economic and non-economic factors on the voting behaviour of the voters in the forthcoming United States Presidential Election.

The proposed Lasso Regression model, forecasts that the vote share for the incumbent Republican Party to be 41.63% in 2020 US presidential election. This means that the incumbent party is most likely to lose the upcoming election.

INTRODUCTION

The result of the forthcoming United States Presidential Election holds importance for the major developing and developed countries all across the world. In order to forecast the result of the forthcoming election, various studies have been conducted by economists and political scientists all around the world. Even in the past, various researchers have attempted forecasting the results of the United States Presidential Elections. While the emphasis is put on the influence of economic variables in the voting behaviour for the forthcoming Presidential Election in some of these studies, other studies put emphasis on the role of non-economic factors in influencing the voting behaviour. In this paper, we investigate how a combination of economic and non-economic variables influence the election result.

Some of the previous studies conducted by Fair (1978, 2016), Silver (2011), Jérôme and Jérôme (2011), Cuzán, Heggen, and Bundrick (2016), Abramowitz (1988), among various others investigate the influence of various economic and non-economic factors on the results of United States Presidential Elections. In the forecasting model proposed by Fair (1978, 2016), the focus is on the economic factors such as the unemployment rate, rate of inflation, growth rate of real per capita GDP, etc. Although the economic factors have been considered to play a significant role in determining the results of the Presidential Election in various other studies, the research conducted by Silver (2011) shows that there exists only a small correlation between the vote share percentage of the incumbent President and the rate of employment in the economy during his tenure. However, the rate of unemployment in the economy during the tenure of the Incumbent President is considered to be the most important economic factor in forecasting the election result, as per the model proposed by Jérôme and Jérôme (2011). Furthermore, the emphasis is on the significance of the economy's rate of growth in the first six months of the year in which the election is to be held, as per the model proposed by Abramowitz (1988). The study conducted by Lichtman (2005, 2008) also used the rate of economic growth as a significant factor in forecasting the

election result. In the study conducted by Erikson and Wlezien (1996), a comprehensive view of the economic indicators was adopted as an index of major economic factors is considered to forecast the result of the Presidential Election. However, as per the Bread and Peace model of Hibbs (2000, 2012) growth in the real disposable per capita income is considered to be a significant factor. In addition to the aforementioned studies, Sinha and Bansal (2008) derived the predictive density function under the hierarchical priors in order to determine the election result, with the help of the Fair's model.

Another important economic factor, in addition to the growth rate of the economy, considered to be significant in forecasting the election result is the inflation rate in the economy. In the model proposed by Fair (1978, 2016), the absolute value of the growth rate of the GDP deflator is used to determine the election results. Furthermore, using a similar definition of inflation, the study conducted by Cuzan, et al (2000) aims to forecast the presidential election result by way of running simulation on fiscal models.

The non-economic factors which can influence the election result include military interventions, scandals and international crises, as emphasised in the model proposed by the study conducted by Mueller (1970). In our research, it has also been found that the amount of funds spent on election campaigns by both the contesting political parties will also play a significant role in determining the result of the forthcoming presidential election.

In reference to the proposed models mentioned above, this paper seeks to forecast the result of the forthcoming Presidential election with the help of a Lasso Regression Model and using a combination of non-economic and economic variables.

SIGNIFICANCE OF VARIABLES CONSIDERED

It could be concluded on the basis of the aforementioned studies; various economic and non-economic variables play a crucial role in forecasting the results of Presidential Elections in the United States. The various economic variables and non-economic variables considered in the paper for forecasting the election result are mentioned in this section.

Economic Variables

In this section, the various economic variables considered for forecasting the result of the forthcoming US presidential election are listed out. The perception of the voters is influenced by factors such as growth rate of the economy, unemployment rate, and rate of inflation. The state of the global economies may be indicated by global indicators such as exchange rates, gold rates and oil prices. The state of the global economies impacts the state of the United States economy and thus can impact the result of the forthcoming Presidential Election. The economic factors considered in this paper to determine the result of the forthcoming United States Presidential Election include the following: -

1. **Inflation:** Average percentage inflation rates for the calendar year prior to the election year have been considered. The year prior to the election year was considered because this year was exceptional due to the Covid-19 pandemic. Average percentage inflation rates are calculated by using the Consumer Price Index published monthly by the usinflationcalculator.com.

2. **Unemployment Rate:** The average of the civilian unemployment rate (percent) for the January to March period of the election year has been considered, which is published by the U.S. Bureau of Labour Statistics.

3. **Economic Growth:** The annual percentage rate of growth of the real GDP per capita of the election year is considered. The data has been taken from the Federal Bank of St. Louis.

4. **Gold Prices:** The inflation-adjusted yearly average gold prices in dollars per ounce are considered with data from the National Mining Organization (U.S.).

5. **Gold Price Index:**
    a. If the price of gold in dollars per ounce in the previous election year is greater than the price of gold in dollars per ounce in the current election year, then the index's value is 0.
    b. If the price of gold in dollars per ounce in the previous election year is lesser than the price of gold in dollars per ounce in the current election year, then the index's value is 1.

6. **Oil Prices**: Average annual domestic crude oil prices in dollars per barrel, after being adjusted for inflation, have been considered for the respective election years. Prices are adjusted for inflation to January 2020 prices using CPI-U from the Bureau of Labor Statistics.

7. **Exchange Rate:** The exchange rate has been considered as the U.S. Dollars to One British Pound (not seasonally adjusted) for June in the election year.

    The data for all the economic variables from 1952 to 2020 considered for forecasting the election result is summarized in the appendix.

Non-economic Variables

As understood from the review of previous studies done on forecasting the result of Presidential Elections, various non-economic and social factors influence voting behaviour. The voters' perception of the incumbent party and the opposition, the non-incumbent party, is influenced by various non-economic factors. The Gallup Rating, for example, is a measure of the approval rating for the work done by the Incumbent President during his tenure. The non-economic variables considered in this paper to forecast the result of the forthcoming United States Presidential Election include the following: -

1. **Gallup Job Approval Rating:** The Gallup Job Approval Rating or the Presidential Work Approval rating is a measure of the percentage of the United States population that approves or disapproves of the work done by the Incumbent President during his tenure as the President of the United States. The Gallup Job Approval Rating considered in this paper is for June of the election year. The major reason why the rating for June of the election year is considered instead of the rating for the months closer to the election month is that the Gallup Job Approval Rating for June of the election year is relatively freer from the electoral mood swings.

2. **Average Gallup Rating:** It represents the Gallup approval rating for the incumbent President throughout the tenure. Data for both Gallup Job Approval Rating and Average Gallup Rating has been taken from the Gallup Rating website.

3. **Crime Rate:** The Average annual total crime rate per 100,000 people in the United States during the incumbent President's tenure is considered. Total crime rate includes violence, property crimes, murder, rape, robbery, assault, burglary, larceny-theft & vehicle theft.

4. **Power of Period:** It is an indicator of the amount of time that the incumbent President's party has been in power. It has been defined as a binary variable with two values 0 and 1
    a. 1, if the incumbent party was in the White House for two or more term
    b. 0 otherwise.

5. **Mid-Term Performance:** This variable is the same as defined in Sinha et al. (2012) for forecasting the results of the 2012 elections. It is defined as:

$$M = (\text{House}_{\text{Seats}} * \text{House}_{\text{Results}} + \text{Senate}_{\text{Seats}} * \text{Senate}_{\text{Results}}) / (\text{House}_{\text{Seats}} + \text{Senate}_{\text{Seats}})$$

6. **Campaign Spending Index:** Campaign spending data for both the incumbent and challenger party have been taken from the Federal Election Commission (U.S.) Website. The campaign spending index is calculated by taking the ratio of the incumbent to non-incumbent campaign spending.
    a. If the ratio is less than 1, the value of index is 0
    b. If the ratio is less than 2, the value of index is 1
    c. If the ratio is greater than or equal to 2, the value of index is 2

7. **Scandal Rating:** Scandals are perceived negatively by the voting population. This affects the incumbent party's popularity during Presidential elections. Scandal rating attempts to take into account the effect of scandals on the election outcome. The ratings to this variable are as follows:

    a. No major scandal during Presidential tenure; rating = 0
    b. At least one major scandal during Presidential term; rating = 1
    c. The scandals that lead to termination of the president during his term, rating = 2

DATA SOURCES

All the values for economic and non-economic variables are considered from 1952 till 2016. The data for growth of the economy has been taken from the Federal Bank of St. Louis. The data for inflation is considered average percentage inflation rates for the calendar year before the election year source is usinflationcalculator.com. Unemployment rate and oil price data is taken from the U.S. Bureau of Labour Statistics. Historical data for gold prices is taken from the National mining organization.

Non-economic factor like scandal rating have been arrived by secondary research on past U.S. Presidential tenure. Historical data previous to the tenure of Donald Trump have been gathered from Sinha et al. (2012) for forecasting the results of 2012 elections. The data has been collected from the articles and essays on the history of U.S. president, which include dedicated white house resource and other reliable resources like Miller Centre. The different Gallup ratings were taken from the Gallup Presidential Poll (2012). The crime rate data is collected form the The disaster center website which provides uniform crime rate data from 1960 to 2019. The Campaign spending data for both the incumbent and challenger party have been taken from the Federal Election Commission (U.S.) Website.

The dependent variable in our model is the vote percentage of the incumbent party Presidential election, which is obtained from uselectionatlas.org.

METHODOLOGY

There are a few key aspects of the method that we would like to follow for this paper. First, we wish to make use of machine learning algorithms to improve our predictions and second is to not lose the interpretability of the model while doing so (or make it simpler if possible). Thus, the first step was to scout a regression-based machine learning algorithm that suits our criteria. Here, we identified lasso regression helps us achieve both our goals and the end results speak volumes about the accuracy of the algorithms

Prior to training the model on the said model, we need to identify the significant features and for that we use stepwise regression analysis. In order to remove any biases from our end, we performed both:

1. Forward Stepwise regression:
2. Backward Stepwise regression:

Now, we decided to use the results of the above two techniques to carry out the hyperparameter (L1 regularization parameter - λ) tuning to identify the optimized value of regularization parameter based on the root means square errors of the predicted values of 2012 and 2016 elections.

Post that we trained our model on the best fit amongst the two stepwise regression techniques and calculate the incumbent vote percentage for the year 2020.

Lasso Regression

The algorithm we have used here is the Lasso Regression Algorithm. Lasso regression performs L1 regularization, which adds a penalty which is nothing but sum of the absolute values of coefficients multiplied by a penalty factor. In regularization, we introduce additional information to improve our model or to prevent issues like overfitting. Usually, regularization like L1 can result in sparse matrices with lesser variables. Coefficients of some variables can become zero and those are eliminated from the model. If we choose the value of this penalty factor to be very high, a lot of variables can end up having coefficients equal to zero.
As our aim is to get a model which uses the power of machine learning but avoids entering the black box, Lasso regression technique seems suitable for our cause.

It tries to calculate the **β-lasso** vector, which is nothing but an array of weights of various independent variables

$$\hat{\beta}^{\text{lasso}} = \arg\min_{\beta} \sum_{n=1}^{N} \frac{1}{2}(y_n - \beta x_n)^2 + \lambda \sum_{i=1}^{p} |\beta_i|$$

Fig1. Optimization algorithm for Lasso Regression

It is the same as minimizing the sum of squares with constraint Σ |**βi**|≤ s. Some of the βs are shrunk to exactly zero, resulting in a regression model that's easier to interpret. A tuning parameter, λ controls the strength of the L1 penalty. λ is basically the amount of shrinkage.

When λ = 0, the algorithm works like a simple multiple linear regression model and no varaibles are eliminated. As λ increases, we will find more variables getting their weights shrunk down to zero and thereby being eliminated from the model. We realize that as λ increases, bias increases and as λ decreases, the variance increases. Intercept(constant) feature is usually not impacted unless it turns out to be severely insignificant.

Stepwise Regression

1. Forward (Step-up) Regression:

This method is useful when we have a large pool of variables available and we need to systematically choose the variables that contribute the most towards the variability of our dependent variable. Initially there are no variables in our set of independent variables. On each iteration, we include the variable that boosts our R-Squared value the most. We stop the iterations when none of the remaining variables are significant. The only issue with this technique is that you cannot remove a variable once it is part of the set.

The output of this model with a significance value of 0.05 are:

- June_gallup
- scandal_rating_2

Table 1: Output of Forward (Step -up) Regression Analysis

Dep. Variable	vote	**R-squared**	0.756
Model	OLS	**Adj. R-squared**	0.721
Method	Least Squares	**F-statistic**	21.68
Date	Sat, 31 Oct 2020	**Prob (F-statistic)**	5.16E-05
Time	04:02:44	**Log-Likelihood**	-44.159
No. Observations	17	**AIC**	94.32
Df Residuals	8	**BIC**	96.82
Df Model	8		
Covariance Type	nonrobust		

	Coefficient	**Std. err**	**t**	**P>|t|**	**[0.025**	**0.975]**
June_gallup	0.5332	0.082	6.537	0	0.358	0.708
Scandal_rating_2	-5.7234	2.121	-2.698	0.017	-10.273	-1.173
intercept	-0.1153	6.233	-0.019	0.986	-14.489	14.258

Omnibus	1.514	**Durbin-Watson**	2.001
Prob(Omnibus)	0.469	**Jarque-Bera(JB)**	0.886
Skew	0.099	**Prob(JB)**	0.642
Kurtosis	1.9	**Cond. No.**	228

As we can see from the results of the regression analysis carried out on the factors obtained from the forward regression, the p-values are less than 0.05 for both June-Gallup and Scandal_rating_2. The R squared value and adjusted R Squared values are .756 and .721 respectively which can be considered as a moderate result from the model. The F-value is significant and shows that the model is a good fit.

2. Backward (Step-down) Regression

This method starts with a model where all the variables are included initially. We get to set the significance level at which we would like to eliminate the variables. On each iteration, the variable which is the least significant is eliminated. The process is terminated when there is no insignificant variable in the system. The issue with this process is that it may retain variables that might not be actually significant.

The output of this model gave us the following variables:

- June_gallup
- Unemployment
- Exchange rate (GBP/USD) - June -New Data
- Midterm_values
- Incumbent_president_running_0,
- Scandal_rating_0
- Scandal_rating_2
- Campaign_spending_2.0

Table 2: Output of Backward (Step- down) Regression Analysis

Dep. Variable	vote	**R-squared**	0.965
Model	OLS	**Adj. R-squared**	0.93
Method	Least Squares	**F-statistic**	27.58
Date	Sat, 31 Oct 2020	**Prob (F-statistic)**	4.82E-05
Time	04:02:44	**Log-Likelihood**	-27.656
No. Observations	17	**AIC**	73.31
Df Residuals	8	**BIC**	80.81
Df Model	8		
Covariance Type	nonrobust		

	Coefficient	**Std. error**	**t**	**P>|t|**	**[0.025**	**0.975]**
June_gallup	0.7095	0.058	12.71	0	0.575	0.844
Unemployment	3.6417	0.705	5.167	0.001	2.016	5.267
Exchange Rate (GBP/USD) - June - New Data	-4.2702	1.437	-2.971	0.018	-7.584	-0.956
Midterm_values	4.2489	0.922	4.609	0.002	2.123	6.375
Incumbent_president_running_0	7.3631	2.032	3.624	0.007	2.678	12.049
Scandal_rating_0	11.3266	2.055	5.511	0.001	6.587	16.066
Scandal_rating_2	-3.012	1.203	-2.503	0.037	-5.787	-0.237
Campaign_spending_2.0	12.6021	2.743	4.594	0.002	6.276	18.928
intercept	-0.1153	6.233	-0.019	0.986	-14.489	14.258

Omnibus	0.015	**Durbin-Watson**	2.255
Prob(Omnibus)	0.992	**Jarque-Bera(JB)**	0.169
Skew	-0.057	**Prob(JB)**	0.919
Kurtosis	2.525	**Cond. No.**	771

As we can see from the regression analysis, all the factors have a p-value less than 0.05. Also, the R-squared and explained R-squared values are .96 and .93, which can be considered good under the circumstances. The significance of F-value is also a reflection of the model being a good fit.

Hyperparameter Tuning

Hyperparameters are basically variables that impact the working (learning rate/penalty/depth etc.) of an algorithm (in our case: λ). Here we will do hyperparameter tuning (or optimization) to identify the value of λ. Hyperparameter optimization is the technique of identifying the value of this variable which optimizes the performance of the algorithm.

To optimize the value of λ, we use RMSE (root mean squared error) as the metric. Here we loop the value of the penalty factor between 0.01 and 0.5 in steps of 0.01 and try to minimize the RMSE for 2012 and 2016 predictions.

We do it for both the models available and try to see which combination of variables and at what rate does the algorithm perform better.

For Forward Regression method, we get the optimum learning rate to be .49 and RMSE 1.98

For Backward Regression method, we get the optimum learning rate to be .15 and RMSE 0.643.

As we can clearly see the variables received from the backward regression method outperform the forward regression set of variables in terms of R squared value as well as RMSE. Now we see the results obtained from back-testing our set of variables that we propose for the model.

Back-testing

Here we run the Lasso Regression model by training the model with data from years 1952-2008 for predicting values for 2012 election and subsequently train the model with data from years 1952-2012 to predict the incumbent vote share for the year 2016. We will use the variables that were shortlisted by the backward regression method. The L1 regularization parameter taken is 0.15.
The results of the same are summarized in the table below:

Table 3: Performance of Proposed model for previous election years

Election Year	Actual Vote Percentage	Predicted Vote Percentage
2012	51	51.72
2016	48.02	48.57

The results obtaining from running this model are highly accurate and it seems to capture the factors impacting the incumbent vote share in a more holistic manner.

PROPOSED MODEL

The proposed model for training the lasso regression is

INCUMBENT_VOTE_SHARE = β1+ β2 JUNE_GALLUP + β3 UNEMPLOYMENT + β4 EXCHANGE RATE + β5 MIDTERM_VALUES + β6INCUMBENT_PRESIDENT_RUNNING_0 + β7 SCANDAL_RATING_0 + β8 SCANDAL_RATING_2 + β9 CAMPAIGN_SPENDING_2.0

RESULTS

At the end of training the Lasso regression model, we receive the following value for **βlasso.** As we can see here, the lasso regressor, as expected, shrunk a few parameters down to zero and makes the entire model even simpler. The values of parameters given below are significant at 5% level of significance.

Table 4: Estimates of parameters of proposed Model using Lasso regression

Independent variable	**β**
June_gallup	0.590
Unemployment rate	1.093
Exchange rate	0
Midterm_value	0.665
Incumbent_president_running_0	0
Scandal_rating_0	5.914
Scandal_rating_2	-4.3188
Campaign_spending_2.0	1.582
L1 regularization parameter(λ)	0.15

*All estimated parameters are significant at 5% level of significance

When we use the above-mentioned significant values of parameters of the model as weight and combine them with below mentioned values of Independent variables taken for the year 2020, we will get the incumbent vote share. June Gallup as expected plays a huge role in reflecting the sentiment of the citizens of the United States. Contrary to popular belief, the Unemployment data has a very slight positive correlation with the vote share and the model weight thus turns out to be positive. The higher value can also be an attempt to balance for some extremely negative variable in the system. Other factors are self-explanatory and make logical sense in terms of their weights.

Table 5: Values of Independent variables of the proposed model for the year 2020

Variable	Value (2020)
June_gallup	38
Unemployment	3.83
Exchange rate	1.25
Midterm_values	-0.63
Incumbent_president_running_0	0

Scandal_rating_0	0
Scandal_rating_2	0
Campaign_spending_2	0

While we put the values in the proposed model, we receive the vote share for the incumbent party to be 41.63%. That means that the incumbent party is most likely to lose the upcoming elections.

CONCLUSION

The proposed model predicts that the **vote share for the incumbent party to be 41.63%** in the 2020 US Presidential Election. The Republican Party would fail to get the required number of votes to win the 2020 US presidential election. The model, backed by a machine learning algorithm (Lasso Regression) has shown promising results while back testing, for the years 2012 and 2016, with an RMSE of just 0.643.

Going back to the our proposed model it seems like the campaign spending and midterms results will play a significant role in 2020 US Presidential Election.

REFERENCES

1. Lewis-Beck, M. S. & Rice, T. W. (1982).Presidential Popularity and Presidential Vote. The Public Opinion Quarterly, 46 4, 534-537.

2. Fair, R. C. (1978). The effect of economic events on votes for president. Review of Economics and Statistics, 60, 159-173

Fair, R.C. (2016). Vote-Share Equations: November 2014 update, retrieved from http://fairmodel.econ.yale.edu/vote2016/index2.htm

3. Silver, N. (2011). On the Maddeningly Inexact Relationship between Unemployment and Re-Election, retrieved from http://fivethirtyeight.blogs.nytimes.com/2011/06/02/on-themaddeningly- inexact-relationship-between-unemployment-and-re-election/.

4. Jérôme, Bruno & Jérôme -Speziari, Veronique. (2011). Forecasting the 2012 U.S. Presidential Election: What Can We Learn from a State Level Political Economy Model. In Proceedings of the APSA Annual meeting Seattle, September 1-4 2011

5. Cuzán, A. G., Heggen R.J., & Bundrick C.M. (2000). Fiscal policy, economic conditions, and terms in office: simulating presidential election outcomes. In Proceedings of the World Congress of the Systems Sciences and ISSS International Society for the Systems Sciences, 44th Annual Meeting, July 16–20, Toronto, Canada.

6. Abramowitz A. I. (1988). An Improved Model for Predicting the Outcomes of Presidential Elections. PS: Political Science and Politics, 21 4, 843-847

7. Lichtman, A. J. (2005). The Keys to the White House. Lanham, MD: Lexington Books.

Lichtman, A. J. (2008). The keys to the white house: An index forecast for 2008. International Journal of Forecasting, 24, 301–309.

8. Erikson, R. S., and Wlezien, C. (1996). Of time and presidential election forecasts. PS: Political Science and politics, 31, 37-39

9. Hibbs D. A. (2000). Bread and Peace voting in U.S. presidential elections. Public Choice, 104, 149–180.

Hibbs, Douglas A. (2012). Obama's Re-election Prospects Under 'Bread and Peace' Voting in the 2012 US Presidential Election. Retrieved from: http://www.douglas-hibbs.com/HibbsArticles/HIBBS_OBAMA-REELECT-31July2012r1.pdf.

10. Sinha, P. and Bansal, A.K. (2008). Hierarchical Bayes Prediction for the 2008 US Presidential Election. The Journal of Prediction Markets, 2, 47-60.

11. Mueller J.E. (1970), Presidential Popularity from Truman to Johnson. The American Political science review, 64, 18-34. 22.

12. Usinflationcalculator.com (2020). Current US Inflation Rates: 2009-2020. Retrieved from https://www.usinflationcalculator.com/inflation/current-inflation-

rates/#:~:text=The%20annual%20inflation%20rate%20for,published%20on%20October%2013%2C%202020

13. Bureau of Labor Statistics (2020). Civilian Unemployment Rate. Retrieved from https://www.bls.gov/charts/employment-situation/civilian-unemployment-rate.htm

14. Federal Reserve Bank of St. Louis (2020). Real GDP Per Capita. Retrieved from https://fred.stlouisfed.org/series/A939RX0Q048SBEA#0

15. National Mining Association. Historical Gold Prices, 1833 to Present. Retrieved from https://nma.org/wp-content/uploads/2019/02/his_gold_prices_1833_pres_2019.pdf

16. Inflationdata.com (2020). Historical Crude Oil Price (Table). Retrieved from https://inflationdata.com/articles/inflation-adjusted-prices/historical-crude-oil-prices-table/

17. Federal Reserve Bank of St. Louis, retrieved from https://fred.stlouisfed.org/series/EXUSUK

18. Gallup Presidential Poll. (2016). Presidential Job Approval Centre. Retrieved from https://news.gallup.com/poll/203198/presidential-approval-ratings-donald-trump.aspx

19. DisasterCenter.com (2020). US Crime Rates 1960-2019. Retrieved from http://www.disastercenter.com/crime/uscrime.htm

20. History, Art and Archives, US House of Representatives (2020). Election Statistics, 1920 to Present. Retrieved from https://history.house.gov/Institution/Election-Statistics/Election-Statistics/

21. Federal Election Commission (2020). Campaign Finance Data (2020) Biden for President. Retrieved from https://www.fec.gov/data/committee/C00703975/?tab=spending

22. Federal Election Commission (2020). Campaign Finance Data Donald J. Trump. Retrieved from https://www.fec.gov/data/committee/C00580100/?tab=spending&cycle=2020

23. BBC.com (2020). Trump impeachment: The short, medium and long story. Retrieved from https://www.bbc.com/news/world-us-canada-49800181

APPENDIX

Table 6: Popular and Electoral Votes received by Incumbent party candidates

Source: uselectionatlas.org

Year	Popular vote	Electoral vote
1952	44.33%	16.80%
1956	57.37%	86.10%
1960	49.55%	40.80%
1964	61.05%	90.30%
1968	42.72%	35.50%
1972	60.67%	96.70%
1976	48.01%	44.60%
1980	41.01%	9.10%
1984	58.77%	97.60%
1988	53.37%	79.20%
1992	37.45%	31.20%
1996	49.23%	70.40%
2000	48.38%	49.40%
2004	50.73%	53.20%
2008	45.60%	32.20%
2012	51.01%	61.70%
2016	48.02%	42.20%

Table 7: Scandals during Presidential Terms and the Corresponding Ratings

Year	Incumbent President	Scandals	Rating
1952	Harry S. Truman	• Continuous accusations of spies in the US Govt. • Foreign policies: Korean war, Indo China war White house renovations • Steel and coal strikes • Corruption charges	1
1956	Dwight D. Eisenhower	• None	0
1960	Dwight D. Eisenhower	• U-2 Spy Plane Incident • Senator Joseph R. McCarthy Controversy • Little Rock School Racial Issues	1
1964	John F. Kennedy	• Extra-marital relationship	0
	Lyndon B. Johnson	• None	
1968	Lyndon B. Johnson	• Vietnam war • Urban riots • Phone Tapping	1
1972	Richard Nixon	• Nixon Shock	0
1976	Richard Nixon	• Watergate	2
	Gerald Ford	• Nixon Pardon	
1980	Jimmy Carter	• Iran hostage crisis • 1979 energy crisis • Boycott of the Moscow Olympics	1
1984	Ronald Reagan	• Tax cuts and budget proposals to expand military spending	0
1988	Ronald Reagan	• Iran-Contra affair • Multiple corruption charges against high ranking officials	1
1992	George H W Bush	• Relegation on election promise of no new taxes • "Vomiting Incident"	1
1996	Bill Clinton	• Firing of White House staff • "Don't ask, don't tell" policy	1
2000	Bill Clinton	• Lewinsky Scandal	2
2004	George W Bush	• None	0
2008	George W Bush	• Midterm dismissal of 7 US attorneys • Guantanamo Bay Controversy and torture	1
2012	Barack Obama	• None	0

2016	Barack Obama	• None	0
2020	Donald Trump	• Ukraine Impeachment Scandal • Tax Evasion	1

Table 8: Gallup Ratings
Source: Gallup Presidential Poll (2020)

Year	Incumbent President	June Gallup Rating	Average Gallup Rating
1952	Harry S. Truman	31.5	36.5
1956	Dwight D. Eisenhower	72	69.6
1960	Dwight D. Eisenhower	59	60.5
1964	Lyndon B. Johnson	74	74.2
1968	Lyndon B. Johnson	41	50.3
1972	Richard Nixon	57.5	55.8
1976	Gerald Ford	45	47.2
1980	Jimmy Carter	33.6	45.5
1984	Ronald Reagan	54	50.3
1988	Ronald Reagan	50	55.3
1992	George H W Bush	37.3	60.9
1996	Bill Clinton	55	49.6
2000	Bill Clinton	57.5	60.6
2004	George W Bush	48.5	62.2
2008	George W Bush	29	36.5
2012	Barack Obama	46.4	49.0
2016	Barack Obama	51.6	48.0
2020	Donald Trump	38	41

Table 9: Mid-Term Election Results (1948-2018); Source: Office of the Clerk (US)

Year	Incumbent Party	Mid Term Election Year	House Seats		House Result	Senate Seats		Senate Result	Midterm Values
			D	R		D	R		
		1948	263	171		54	42		
1952	Democratic	1950	234	199	1	48	47	1	1
		1952	213	221		46	48		
1956	Republican	1954	232	203	-1	48	47	-1	-1
		1956	234	201		49	47		
1960	Republican	1958	283	153	-1	64	34	-1	-1
		1960	262	175		64	36		
1964	Democratic	1962	258	176	1	67	33	1	1
		1964	295	140		68	32		
1968	Democratic	1966	248	187	1	64	36	1	1
		1968	243	192		58	42		
1972	Republican	1970	255	180	-1	54	44	-1	-1
		1972	242	192		56	42		
1976	Republican	1974	291	144	-1	61	37	-1	-1
		1976	292	143		61	38		
1980	Democratic	1978	277	158	1	58	41	1	1
		1980	242	192		46	53		
1984	Republican	1982	269	166	-1	46	54	1	-0.63
		1984	253	182		47	53		
1988	Republican	1986	258	177	-1	55	45	-1	-0.63
		1988	260	175		55	45		
1992	Republican	1990	267	167	-1	56	44	-1	-1
		1992	258	176		57	43		
1996	Democratic	1994	204	230	-1	48	52	-1	-1
		1996	207	226		45	55		
2000	Democratic	1998	211	223	-1	45	55	-1	-1
		2000	212	221		50	50		
2004	Republican	2002	204	229	1	48	51	1	1
		2004	202	232		44	55		
2008	Republican	2006	233	202	-1	49	49	0	-0.82
		2008	256	178		55	41		
2012	Democratic	2010	193	242	-1	51	47	1	-0.63
		2012	200	234		53	45		
2016	Democratic	2014	188	247	-1	44	54	1	-0.63
		2016	194	241		46	52		
2020	Republican	2018	235	199	-1	45	53	1	-0.63

Table 10: Economic Data

Source: a: Bureau of Labour Statistics; b: usinflationcalculator.com; c: National Mining Organization; d: inflationdata.com; e: Federal Bank of St. Louis

Year	Unemployment[a]	Inflation[b]	Gold_price_index[c]	Gold Price ($/ounce)[c]	Oil Prices[d]	Ex. rate (USD/GBP)[e]
1952	3.07	7.9		34.6	26.92	2.79
1956	4.03	-0.4	1	34.99	27.92	2.80
1960	5.13	0.7	1	35.27	25.41	2.80
1964	5.47	1.3	0	35.1	24.95	2.79
1968	3.73	3.1	1	39.31	23.55	2.39
1972	5.77	4.4	1	58.42	22.21	2.57
1976	7.73	9.1	1	124.74	59.4	1.76
1980	6.3	11.3	1	615	117.3	2.34
1984	7.87	3.2	0	361	71.41	1.38
1988	5.7	3.6	1	437	32.48	1.78
1992	7.37	4.2	0	343.82	35.39	1.86
1996	5.53	2.8	1	387.81	33.63	1.54
2000	4.03	2.2	0	279.11	41.02	1.51
2004	5.7	2.3	1	409.72	51.39	1.83
2008	5	2.8	1	871.96	109.25	1.97
2012	8.27	3.2	1	1668.98	97.17	1.56
2016	4.93	0.1	0	1250.74	39.02	1.42
2020	3.83	1.8	1	1392.6	39.42	1.25

Table 11: Non-Economic Data
Source: a: http://www.disastercenter.com/crime/uscrime.html ; b: Wikipedia; c: Wikipedia; d: Federal Election Commission (www.fec.gov)

Year	Crime rate[a]	Incumbent President Running[b]	Period of power[c]	Campaign spending Index[d]
1952		0	1	0
1956		1	0	2
1960		0	1	1
1964	1998.35	1	0	0
1968	2624.4	0	1	0
1972	3549.85	1	0	2
1976	4566.18	1	1	1
1980	5267.7	1	0	0
1984	5646.73	1	0	1
1988	5317.2	0	1	1
1992	5780.83	1	1	0
1996	5448.25	1	0	1
2000	4724.23	0	1	0
2004	4119.85	1	0	1
2008	3854.08	0	1	0
2012	3444.35	1	0	1
2016	3049.85	0	1	1
2020	2672.35	1	0	0

US ELECTION FORECASTING USING MACHINE LEARNING

In [1]:

```
# Import libraries that will be used for mathematical, data transformations and statistical purpos
es
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import statsmodels.formula.api as sm
import statsmodels.tools.tools as st
from sklearn.linear_model import Lasso
from sklearn.metrics import mean_squared_error as mse
import math
```

In [2]:

```
#Read the Data file
df1 = pd.read_excel(r'C:\Users\Purav Shah\Desktop\FMS\US Election\Data_2020.xlsx',index_col =
'Year')
```

In [3]:

```
#View the imported file
df1.head()
```

Out[3]:

	vote	Growth	June_gallup	Avg_gallup	Unemployment	Inflation_prev_year	Gold_price	Oil_prices	Exchange rate (GBP/USD) - June - New Data	Crim
Year										
1948	49.6	NaN	39.5	55.6	3.73	14.4	NaN	29.73	4.03	NaN
1952	44.3	1.82	31.5	36.5	3.07	7.9	NaN	26.92	2.79	NaN
1956	57.4	0.62	72.0	69.6	4.03	-0.4	1.0	27.92	2.80	NaN
1960	49.6	-0.02	59.0	60.5	5.13	0.7	1.0	25.41	2.80	NaN
1964	61.1	4.71	74.0	74.2	5.47	1.3	0.0	24.95	2.79	1998

In [4]:

```
#Add a column of 1 that mimics the intercept
df1['intercept'] = np.ones(len(df1['vote']))
```

In [5]:

```
#Taking the data from 1952-2020 onwards
df2 = df1[1:]
```

In [6]:

```
cols = df1.columns
cols
```

Out[6]:

```
Index(['vote', 'Growth', 'June_gallup', 'Avg_gallup', 'Unemployment',
       'Inflation_prev_year', 'Gold_price', 'Oil_prices',
       'Exchange rate (GBP/USD) - June -New Data', 'Crime_rate_full_tenure',
       'Midterm_values', 'Incumbent_president_running', 'Period_of_power',
```

```
       'Midterm_values', 'Incumbent_president_running', 'Period_of_power',
       'Scandal_rating', 'Campaign_spending', 'Growth_annual_change',
       'intercept'],
      dtype='object')
```

In [7]:

```
#Taking dependent variable vote outside the matrix from 1952-2016
y = df1[cols[0]]
y = y[1:-1]
```

In [8]:

```
data = df2.dropna(axis = 1)
```

In [9]:

```
data.head()
```

Out[9]:

	Growth	June_gallup	Avg_gallup	Unemployment	Inflation_prev_year	Oil_prices	Exchange rate (GBP/USD) - June - New Data	Midterm_values	Incur
Year									
1952	1.82	31.5	36.5	3.07	7.9	26.92	2.79	1.0	0
1956	0.62	72.0	69.6	4.03	-0.4	27.92	2.80	-1.0	1
1960	-0.02	59.0	60.5	5.13	0.7	25.41	2.80	-1.0	0
1964	4.71	74.0	74.2	5.47	1.3	24.95	2.79	1.0	1
1968	4.46	41.0	50.3	3.73	3.1	23.55	2.39	1.0	0

In [10]:

```
cols1 = data.columns
```

In [11]:

```
x = data.drop(columns=[cols[1]])
```

In [12]:

```
#Taking dummy variables for categorical(ordinal in our case) variables
dfmain = pd.get_dummies(x,columns= ['Incumbent_president_running', 'Period_of_power', 'Scandal_rati
ng', 'Campaign_spending'], drop_first=False)
dfmain = pd.DataFrame(dfmain)
dfmain
```

Out[12]:

	June_gallup	Avg_gallup	Unemployment	Inflation_prev_year	Oil_prices	Exchange rate (GBP/USD) - June - New Data	Midterm_values	Growth_annu
Year								
1952	31.5	36.5	3.07	7.9	26.92	2.79	1.00	2.32
1956	72.0	69.6	4.03	-0.4	27.92	2.80	-1.00	0.35
1960	59.0	60.5	5.13	0.7	25.41	2.80	-1.00	0.52

1964	74.0	74.2	5.47	1.3	24.95	2.79	1.00	4.31
1968	41.0	50.3	3.73	3.1	23.55	2.39	1.00	2.87
1972	57.5	55.8	5.77	4.4	22.21	2.57	-1.00	4.14
1976	45.0	47.2	7.73	9.1	59.40	1.76	-1.00	4.37
1980	33.6	45.5	6.30	11.3	117.30	2.34	1.00	-1.40
1984	54.0	50.3	7.87	3.2	71.41	1.38	-0.63	6.30
1988	50.0	55.3	5.70	3.6	32.48	1.78	-1.00	3.23
1992	37.3	60.9	7.37	4.2	35.39	1.86	-1.00	2.15
1996	55.0	49.6	5.53	2.8	33.63	1.54	-1.00	2.57
2000	57.5	60.6	4.03	2.2	41.02	1.51	-1.00	3.00
2004	48.5	62.2	5.70	2.3	51.39	1.83	1.00	2.87
2008	29.0	36.5	5.00	2.8	109.25	1.97	-0.82	-1.07
2012	46.4	49.0	8.27	3.2	97.17	1.56	-0.63	1.53
2016	48.0	1.0	4.93	0.1	39.02	1.42	-1.00	1.00
2020	38.0	41.0	3.83	1.8	50.01	1.25	-0.63	-4.35

In [13]:

```
#Removing the final row of data for 2020 as our test data
df20 = dfmain[-1:]
df20
```

Out[13]:

	June_gallup	Avg_gallup	Unemployment	Inflation_prev_year	Oil_prices	Exchange rate (GBP/USD) - June - New Data	Midterm_values	Growth_annua
Year								
2020	38.0	41.0	3.83	1.8	50.01	1.25	-0.63	-4.35

In [14]:

```
#Data from 1952-2016 as our train data
dfx = dfmain[:-1]
for col in dfx.columns:
    dfx[col] = dfx[col].apply(lambda x : float(x))
dfx
```

```
C:\Anaconda3\lib\site-packages\ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: http://pandas.pydata.org/pandas-
docs/stable/indexing.html#indexing-view-versus-copy
  after removing the cwd from sys.path.
```

Out[14]:

	June_gallup	Avg_gallup	Unemployment	Inflation_prev_year	Oil_prices	Exchange rate (GBP/USD) - June - New Data	Midterm_values	Growth_annua
Year								

1952	31.5	36.5	3.07	7.9	26.92	2.79	1.00	2.32
1956	72.0	69.6	4.03	-0.4	27.92	2.80	-1.00	0.35
1960	59.0	60.5	5.13	0.7	25.41	2.80	-1.00	0.52
1964	74.0	74.2	5.47	1.3	24.95	2.79	1.00	4.31
1968	41.0	50.3	3.73	3.1	23.55	2.39	1.00	3.87
1972	57.5	55.8	5.77	4.4	22.21	2.57	-1.00	4.14
1976	45.0	47.2	7.73	9.1	59.40	1.76	-1.00	4.37
1980	33.6	45.5	6.30	11.3	117.30	2.34	1.00	-1.40
1984	54.0	50.3	7.87	3.2	71.41	1.38	-0.63	6.30
1988	50.0	55.3	5.70	3.6	32.48	1.78	-1.00	3.23
1992	37.3	60.9	7.37	4.2	35.39	1.86	-1.00	2.15
1996	55.0	49.6	5.53	2.8	33.63	1.54	-1.00	2.57
2000	57.5	60.6	4.03	2.2	41.02	1.51	-1.00	3.00
2004	48.5	62.2	5.70	2.3	51.39	1.83	1.00	2.87
2008	29.0	36.5	5.00	2.8	109.25	1.97	-0.82	-1.07
2012	46.4	49.0	8.27	3.2	97.17	1.56	-0.63	1.53
2016	48.0	1.0	4.93	0.1	39.02	1.42	-1.00	1.00

June_gallup Avg_gallup Unemployment Inflation_prev_year Oil_prices Exchange rate (GBP/USD) - June -New Data Midterm_values Growth_annual_change

In [15]:

```
#Initial regression analysis for all variables
regressor_OLS = sm.OLS(endog = y, exog = dfx).fit()
regressor_OLS.summary()
```

```
C:\Anaconda3\lib\site-packages\scipy\stats\stats.py:1394: UserWarning: kurtosistest only valid for
n>=20 ... continuing anyway, n=17
  "anyway, n=%i" % int(n))
```

Out[15]:

OLS Regression Results

Dep. Variable:	vote	**R-squared:**	0.981
Model:	OLS	**Adj. R-squared:**	0.852
Method:	Least Squares	**F-statistic:**	7.559
Date:	Sat, 31 Oct 2020	**Prob (F-statistic):**	0.123
Time:	04:02:41	**Log-Likelihood:**	-22.260
No. Observations:	17	**AIC:**	74.52
Df Residuals:	2	**BIC:**	87.02
Df Model:	14		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
June_gallup	0.6117	0.193	3.173	0.087	-0.218	1.441
Avg_gallup	0.0085	0.087	0.098	0.931	-0.364	0.381
Unemployment	3.4190	1.932	1.770	0.219	-4.894	11.732
Inflation_prev_year	0.2337	0.445	0.525	0.652	-1.680	2.147
Oil_prices	-0.0109	0.073	-0.149	0.895	-0.327	0.305
Exchange rate (GBP/USD) - June -New Data	-3.5426	4.691	-0.755	0.529	-23.727	16.642
Midterm_values	3.3046	2.774	1.191	0.356	-8.632	15.241
Growth_annual_change	0.3225	1.086	0.297	0.795	-4.350	4.995

intercept	4.9962	3.303	1.513	0.269	-9.213	19.206
Incumbent_president_running_0	7.3171	2.580	2.836	0.105	-3.785	18.419
Incumbent_president_running_1	-2.3210	3.347	-0.693	0.560	-16.722	12.080
Period_of_power_0	4.1538	3.147	1.320	0.318	-9.388	17.696
Period_of_power_1	0.8424	2.595	0.325	0.776	-10.321	12.006
Scandal_rating_0	9.6191	3.303	2.912	0.100	-4.593	23.831
Scandal_rating_1	-1.6967	2.119	-0.801	0.507	-10.812	7.419
Scandal_rating_2	-2.9263	3.515	-0.832	0.493	-18.051	12.199
Campaign_spending_0.0	-3.0083	2.388	-1.260	0.335	-13.283	7.266
Campaign_spending_1.0	-1.5337	2.659	-0.577	0.622	-12.973	9.906
Campaign_spending_2.0	9.5382	3.609	2.643	0.118	-5.992	25.068

Omnibus:	3.060	**Durbin-Watson:**	2.339
Prob(Omnibus):	0.216	**Jarque-Bera (JB):**	1.669
Skew:	-0.764	**Prob(JB):**	0.434
Kurtosis:	3.146	**Cond. No.**	1.14e+18

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The input rank is higher than the number of observations.
[3] The smallest eigenvalue is 1.02e-31. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.

In [16]:

```
#Forward regression with initally all varaibles outside the consideration set
def forward_regression(X, y,
                       threshold_in,
                       verbose=True):
    initial_list = []
    included = list(initial_list)
    while True:
        changed=False
        excluded = list(set(X.columns)-set(included))
        new_pval = pd.Series(index=excluded)
        for new_column in excluded:
            if (len(included) == 0):
                x1 = pd.DataFrame(X[included+[new_column]])
                x1['constant'] = 1
                model = sm.OLS(y, x1).fit()
            else:
                model = sm.OLS(y, pd.DataFrame(X[included + [new_column]])).fit()
            new_pval[new_column] = model.pvalues[new_column]
        best_pval = new_pval.min()
        if best_pval < threshold_in:
            best_feature = new_pval.idxmin()
            included.append(best_feature)
            changed=True
            if verbose:
                print('Add  {:30} with p-value {:.6}'.format(best_feature, best_pval))

        if not changed:
            break

    return included
```

In [17]:

```
forward_regression(dfx,y,.05,verbose = True)
```

```
Add  intercept                      with p-value 1.46728e-15
Add  June gallup                    with p-value 0.000146503
```

```
Add  June_gallup                      with p-value 0.000119303
Add  Scandal_rating_2                 with p-value 0.0179419
```

Out[17]:

```
['intercept', 'June_gallup', 'Scandal_rating_2']
```

In [18]:

```
fcols = ['intercept', 'June_gallup', 'Scandal_rating_2']
dfx1 = dfx[fcols]
test = df20[fcols]
regressor_OLS = sm.OLS(endog = y, exog = dfx1).fit()
regressor_OLS.summary()
```

```
C:\Anaconda3\lib\site-packages\scipy\stats\stats.py:1394: UserWarning: kurtosistest only valid for
n>=20 ... continuing anyway, n=17
  "anyway, n=%i" % int(n))
```

Out[18]:

OLS Regression Results

Dep. Variable:	vote	**R-squared:**	0.755
Model:	OLS	**Adj. R-squared:**	0.720
Method:	Least Squares	**F-statistic:**	21.53
Date:	Sat, 31 Oct 2020	**Prob (F-statistic):**	5.35e-05
Time:	04:02:42	**Log-Likelihood:**	-44.210
No. Observations:	17	**AIC:**	94.42
Df Residuals:	14	**BIC:**	96.92
Df Model:	2		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
intercept	25.5520	3.809	6.709	0.000	17.383	33.721
June_gallup	0.5329	0.082	6.514	0.000	0.357	0.708
Scandal_rating_2	-5.7028	2.128	-2.680	0.018	-10.267	-1.139

Omnibus:	1.613	**Durbin-Watson:**	1.996
Prob(Omnibus):	0.446	**Jarque-Bera (JB):**	0.911
Skew:	0.101	**Prob(JB):**	0.634
Kurtosis:	1.884	**Cond. No.**	228.

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [19]:

```
brmse = 10000
blr = 0
lr = 0.01
while lr < .5:

    #Back-testing the variables for 2016 elections
    btx1 = dfx1[:-1]
    bty1 = y[:-1]
    lassof = Lasso(alpha=lr)
    lassof.fit(btx1, bty1)
    yp1 = lassof.predict(dfx1[-1:])

    mse1 = mse(list(yp1),list(y[-1:]))
```

```
    #Back-testing the variables for 2012 elections
    btx2 = dfx1[:-2]
    bty2 = y[:-2]
    lassof2 = Lasso(alpha=lr)
    lassof2.fit(btx2, bty2)
    yp2 = lassof2.predict(dfx1[-2:-1])
    mse2 = mse(list(yp2), list(y[-2:-1]))

    rmse = math.sqrt((mse1+mse2)/2)
    if rmse<brmse:
        blr = lr
        brmse = rmse
    lr = lr + 0.01

print(blr,brmse)
```

```
0.49000000000000027 1.985239036648162
```

In [20]:

```
#back-testing the variables for 2012 elections
btx2 = dfx1[:-2]
bty2 = y[:-2]
lasso = Lasso(alpha=.49)
lasso.fit(btx2, bty2)
yp2 = lasso.predict(dfx1[-2:-1])
yp2, y[-2:-1]
```

Out[20]:

```
(array([49.60511617]), Year
 2012    51.0
 Name: vote, dtype: float64)
```

In [21]:

```
btx1 = dfx1[:-1]
bty1 = y[:-1]
lassof = Lasso(alpha=blr)
lassof.fit(btx1, bty1)
yp1 = lassof.predict(dfx1[-1:])
yp1, y[-1:]
```

Out[21]:

```
(array([50.45652358]), Year
 2016    48.02
 Name: vote, dtype: float64)
```

In [22]:

```
#predicting for 2020 elections
lasso = Lasso(alpha=blr)
lasso.fit(dfx1, y)
dict1 = [fcols, list(lasso.coef_)]
coefs = pd.DataFrame(dict1)
coefs
```

Out[22]:

	0	1	2
0	intercept	June_gallup	Scandal_rating_2
1	0	0.470607	-2.72179

In [23]:

```
#Calculating the final vote percentage
vote_p = lasso.predict(test)
```

```
print(str(vote_p[0])[:5] +'%')
```

```
45.45%
```

In [24]:

```
#Backward regression with initally all variables inside consdieration set and removing them by mea
ns of lowest significance
def backward_regression(X, y,
                        threshold_out,
                        verbose=False):
    included=list(X.columns)
    while True:
        changed=False
        model = sm.OLS(y, st.add_constant(pd.DataFrame(X[included]))).fit()
        # use all coefs except intercept
        pvalues = model.pvalues.iloc[1:]
        worst_pval = pvalues.max() # null if pvalues is empty
        if worst_pval > threshold_out:
            changed=True
            worst_feature = pvalues.idxmax()
            included.remove(worst_feature)
            if verbose:
                print('Drop {:30} with p-value {:.6}'.format(worst_feature, worst_pval))
        if not changed:
            break
    return included
```

In [25]:

```
backward_regression(dfx,y,.05,verbose = True)
```

```
Drop Avg_gallup                     with p-value 0.930556
Drop Oil_prices                     with p-value 0.890576
Drop Period_of_power_1              with p-value 0.639493
Drop Campaign_spending_1.0          with p-value 0.513509
Drop Inflation_prev_year            with p-value 0.48878
Drop Incumbent_president_running_1  with p-value 0.588726
Drop Scandal_rating_1               with p-value 0.480652
Drop intercept                      with p-value 0.663837
Drop Period_of_power_0              with p-value 0.34016
Drop Growth_annual_change           with p-value 0.43309
Drop Campaign_spending_0.0          with p-value 0.2848
```

Out[25]:

```
['June_gallup',
 'Unemployment',
 'Exchange rate (GBP/USD) - June -New Data',
 'Midterm_values',
 'Incumbent_president_running_0',
 'Scandal_rating_0',
 'Scandal_rating_2',
 'Campaign_spending_2.0']
```

In [26]:

```
#Taking the remaining variables to build our model
bcols = ['June_gallup',
 'Unemployment',
 'Exchange rate (GBP/USD) - June -New Data',
 'Midterm_values',
 'Incumbent_president_running_0',
 'Scandal_rating_0',
 'Scandal_rating_2',
 'Campaign_spending_2.0','intercept']
dfx1 = dfx[bcols]
test = df20[bcols]
regressor_OLS = sm.OLS(endog = y, exog = dfx1).fit()
regressor_OLS.summary()
```

```
C:\Anaconda3\lib\site-packages\scipy\stats\stats.py:1394: UserWarning: kurtosistest only valid for
n>=20 ... continuing anyway, n=17
```

```
n>=20 ... continuing anyway, n=17
  "anyway, n=%i" % int(n))
```

Out[26]:

OLS Regression Results

Dep. Variable:	vote	**R-squared:**	0.965
Model:	OLS	**Adj. R-squared:**	0.930
Method:	Least Squares	**F-statistic:**	27.58
Date:	Sat, 31 Oct 2020	**Prob (F-statistic):**	4.82e-05
Time:	04:02:44	**Log-Likelihood:**	-27.656
No. Observations:	17	**AIC:**	73.31
Df Residuals:	8	**BIC:**	80.81
Df Model:	8		
Covariance Type:	nonrobust		

	coef	**std err**	**t**	**P>|t|**	**[0.025**	**0.975]**
June_gallup	0.7095	0.058	12.171	0.000	0.575	0.844
Unemployment	3.6417	0.705	5.167	0.001	2.016	5.267
Exchange rate (GBP/USD) - June -New Data	-4.2702	1.437	-2.971	0.018	-7.584	-0.956
Midterm_values	4.2489	0.922	4.609	0.002	2.123	6.375
Incumbent_president_running_0	7.3631	2.032	3.624	0.007	2.678	12.049
Scandal_rating_0	11.3266	2.055	5.511	0.001	6.587	16.066
Scandal_rating_2	-3.0120	1.203	-2.503	0.037	-5.787	-0.237
Campaign_spending_2.0	12.6021	2.743	4.594	0.002	6.276	18.928
intercept	-0.1153	6.233	-0.019	0.986	-14.489	14.258

Omnibus:	0.015	**Durbin-Watson:**	2.255
Prob(Omnibus):	0.992	**Jarque-Bera (JB):**	0.169
Skew:	-0.057	**Prob(JB):**	0.919
Kurtosis:	2.525	**Cond. No.**	771.

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

In [27]:

```
brmse = 10000
blr = 0
lr = 0.01
while lr < .5:

    #Back-testing the variables for 2016 elections
    btx1 = dfx1[:-1]
    bty1 = y[:-1]
    lassob = Lasso(alpha=lr)
    lassob.fit(btx1, bty1)
    yp1 = lassob.predict(dfx1[-1:])

    mse1 = mse(list(yp1),list(y[-1:]))

    #Back-testing the variables for 2012 elections
    btx2 = dfx1[:-2]
    bty2 = y[:-2]
    lassob2 = Lasso(alpha=lr)
    lassob2.fit(btx2, bty2)
    yp2 = lassob2.predict(dfx1[-2:-1])
    mse2 = mse(list(yp2),list(y[-2:-1]))
```

```
    mse2 = mse(list(yp2), list(y[-2:-1]))

    rmse = math.sqrt((mse1+mse2)/2)
    if rmse<brmse:
        blr = lr
        brmse = rmse
    lr = lr + 0.01

print(blr,brmse)
```

```
0.15 0.6433248622554568
```

In [28]:

```
btx1 = dfx1[:-1]
bty1 = y[:-1]
lasso = Lasso(alpha=blr)
lasso.fit(btx1, bty1)
yp1 = lasso.predict(dfx1[-1:])
yp1, y[-1:]
```

Out[28]:

```
(array([48.5724892]), Year
 2016    48.02
 Name: vote, dtype: float64)
```

In [29]:

```
btx2 = dfx1[:-2]
bty2 = y[:-2]
lasso = Lasso(alpha=blr)
lasso.fit(btx2, bty2)
yp2 = lasso.predict(dfx1[-2:-1])
yp2, y[-2:-1]
```

Out[29]:

```
(array([51.72283431]), Year
 2012    51.0
 Name: vote, dtype: float64)
```

In [30]:

```
#Running the optimized model to predict value for the 2020 elections
lasso = Lasso(alpha=blr)
lasso.fit(dfx1, y)
dict1 = [bcols, list(lasso.coef_)]
coefs = pd.DataFrame(dict1)
coefs
```

Out[30]:

	0	1	2	3	4	5	6
0	June_gallup	Unemployment	Exchange rate (GBP/USD) - June - New Data	Midterm_values	Incumbent_president_running_0	Scandal_rating_0	Scandal_rating_2
1	0.590488	1.093	-0	0.665358	0	5.91549	-4.31881

In [31]:

```
#Calculating the final vote percentage
vote_p = lasso.predict(test)
print(str(vote_p[0])[:5] +'%')
```

```
41.63%
```