Predicting Trump Voters Using Machine Learning Models

submitted by

Boyan Petkov & Christoph Drechsel

Student ID: 23039871 & 22060503

Degree Programme: Master of Science in Economics

Advisor: Prof. Dr. Rincke

16 January 2023

Table of Content

1. Introduction

The 2016 presidential election in the U.S. was perceived by many as unusual. Some aspects of the election such as the campaign rhetoric and the use of social media were unprecedented in American history (Hendricks and Schill, 2017). Furthermore, the election's outcome with the victory of Donald Trump came to the surprise of many experts and civilians. But could it have been predicted, and if so, which factors would have been the strongest predictors? In our analysis, we use data from the General Social Survey (GSS) to perform such a prediction. Doing so, we use machine learning algorithms to predict whether participants in the 2018 GSS voted for Donald Trump in the 2016 presidential election based on several factors.

Our analysis is motivated by two insights from previous research. First, it is helpful to examine which topics were dominant in the 2016 election. Reny et al. (2019) find that vote switching to either Trump or Clinton occurred among a significant number of White voters and "that this vote switching was more associated with racial and immigration attitudes than economic factors". Second, as immigration attitudes seemingly played an important role in voting behavior, we follow Chandler and Tsai (2001) who find some associations between social factors and immigration attitudes using data from the General Social Survey.

However, our analysis differs in that we mainly focus on the predictive power of our models regarding voting behavior in the 2016 presidential election. Therefore, we use different methods and variables, i.e. we use machine learning algorithms and additionally include variables that capture some political attitudes, such as respondents' attitudes towards government spending or jurisprudence. These are presented in more detail in section 2. Finally, the results of each model are discussed in section 3.

2. Data and Methods

As stated above, we use data from the 2018 GSS. The GSS was conducted 32 times between 1972 and 2018 for the National Data Program for the Social Sciences at the National Opinion Research Center, University of Chicago. It collects information on a variety of characteristics, opinions and attitudes of a selected sample of English-speaking and noninstitutionalized individuals in the United States. In 2018, there were 2348 participants in the core survey. However, the extended part of the GSS is structured in several categories of questions. All respondents participated in the core survey, but only subgroups of the participants responded to the different subcategories of the extended GSS. Therefore, and mainly due to missing values in our selected variables, the sample we use is reduced to 1009 individuals. The variables used in our models are derived from participants' responses to 23 questions in the 2018 GSS. We start with the dependent variable.

The dependent variable is based on a sequence of questions in the GSS regarding the 2016 election. Respondents were first asked if they remember for sure whether they voted in that election. Then, those who responded with "voted" were asked, whether they voted for Clinton or Trump. Possible responses were: "Clinton", "Trump", "Other candidate", "Didn't vote for president", "Don't know", and "No answer". We focus our prediction problem on the subsample that voted in the 2016 election. Therefore, as we aim to predict who voted for Trump, we create a binary variable that takes the value 1 for participants that responded with "Trump", and 0 for participants that responded with "Clinton" or "Other candidate". The remaining respondents were dropped from the sample. It is worth noting that those who did not vote were asked who they would have voted for president, if they had voted. However, as this does not have any impact on the outcome of an election, we do not take these responses into account.

The independent variables were derived from participants' responses to several questions regarding different characteristics and attitudes. They consist of basic social characteristics, such as age, sex, education, religion, marital status, and ethnicity (for ethnicity we focused on whether someone is Hispanic or not, as large parts of Trumps immigration rhetoric was targeted at this racial group), but also on work related characteristics (labor force status, number of hours worked, income, self-employment, industry, government work, years in armed forces, and occupational prestige score), attitudes towards government spending (for childcare, scientific research, alternative energy sources, mass transportation, social security, and parks and recreation), political participation (whether they voted in the 2012 election), and attitude towards the death penalty in the case of murder. Many of these variables are categorical in the original data set. Therefore, we create dummy variables that capture each category, while leaving one category out to avoid collinearity. To categorize respondents' industry, we use industry codes reducing the number of industries from 232 to 12. All variables and summary statistics are presented in **Figures 1 – 3** in the appendix.

We use several machine learning models in order to find the best predictive outcome. We use 70% of our sample for training and 30% for testing the models. As we face a classification problem, each model is defined accordingly. We start by training a logistic regression and then proceed to run Lasso- and Ridge-regressions followed by an elastic net. Then, a Ridge Classifier, a decision tree, a random forest, bagged trees, boosted trees, a Neural Network, and an Ensemble are performed. The predictive power of each model is then evaluated using the testing set. For each model, we evaluate the accuracy score (overall fraction of correct predictions), precision score (fraction of correct Trump predictions among all Trump predictions), and recall score (fraction of correct Trump predictions among all actual Trump voters).

Prior to training each model, we perform hyperparameter tuning. In machine learning, hyperparameters refer to the parameters that are set before the model is trained on a dataset and not the parameters that are directly learnt by the model. The goal is to set the hyperparameters in a way, that their combination maximizes the model performance. Two examples of such hyperparameters are the parameters that controls the maximum depth and the minimum number of samples required for a split in a decision tree model. There are different techniques to choose these hyperparameters, in this particular case a grid search algorithm is used to find the combination of parameters that result in the highest accuracy score. This algorithm uses exhaustive search over all possible combinations of hyperparameters and is therefore slower than other techniques but is very effective in finding the best model configuration. Taking the example previously mentioned, if 4 possible values for each of the two hyperparameters are fed to the grid search algorithm, all 16 possible combinations are calculated and the combination with the highest accuracy is chosen. To make the results more robust, a 10-fold cross-validation approach is used on top of this. Using this procedure, each model is tuned to achieve maximum accuracy. Additionally, we use Lasso-regression to perform feature selection and compare each model's predictive power using only the selected features and using all independent variables. Finally, we also evaluate how each model performs in- and out-of-sample. All models are implemented in Python.

3. Results

Feature Selection with Lasso

As stated above, prior to our prediction models, we first perform feature selection by using a Lasso regression. Lasso (least absolute shrinkage and selection operator) shrinks the coefficient estimates towards or sets them to zero by introducing a penalty term to the cost function that penalizes large coefficients. In our case, as we face a classification problem, the penalty term is introduced to a Logistic Regression (see below). While Lasso reduces the variance of the coefficients, it also increases the bias. When parameters are chosen optimally, it provides an optimal combination of variance and bias, thus minimizing the classification error. Lasso prevents overfitting by shrinking the coefficients of unimportant features to zero and removing them from the model. Therefore, it cannot only be used for predictions, but also for feature selection.

Prior to performing Lasso, there are 56 total features available for our prediction problem. Lasso effectively removes 16 of them and therefore selects 40 features for our predictions with feature selection. Notably, the two predictors for voting Trump with the highest coefficients following this approach are whether someone is unemployed/laid off and whether respondents favor the death penalty in the case of murder. The coefficient of the latter is positive, indicating that those who favor the death penalty were more likely to have voted for Trump than those who oppose it. A visualization of all coefficients is presented in **Figure 4** in the appendix.

Logistic Regression

Logistic Regression is a linear model used for binary classification. It is used to predict a probability of a given input belonging to a certain class, i.e. in our case, the probability of someone with certain characteristics and attitudes voting for Trump. All model coefficients (without feature selection) of the Logistic Regression are visualized in **Figure 5** in the appendix.

Recall that prior to training all the models (with and without feature selection) we perform hyperparameter tuning. Without feature selection, the Logistic Regression approach scores an accuracy of 74.92%. Although the difference is rather small, feature selection improves the overall accuracy of the model's predictions, predicting 75.58% of all items in the testing set correctly. Without feature selection, 74 out of 107 "Trump predictions" are correct, while 74 out of 117 actual Trump voters were classified correctly, achieving a precision and a recall of 69.16% and 63.25%, respectively. While feature selection decreases the model's precision score (85 out of 127, leading to 66.93%), it improves the recall score to 72.65%, with 85 out of 117 Trump voters identified correctly. So while Logistic Regression with feature selection better identifies actual Trump supporters, it also classifies individuals who did not vote for Trump less accurately. A comparison of predictive accuracy with and without feature selection for all models is presented in **Figure 6** in the appendix.

Finally, our comparisons of in- and out-of-sample performances refer to each model without feature selection. In the case of the logistic model, the predictive performance is better out-of-sample than in-sample (accuracy of 74.92% compared to 72.23%, respectively). A comparison of in- and out-of-sample performance for all models is presented in **Figure 7** in the appendix.

Lasso

We already performed Lasso for feature selection. However, using Lasso for our prediction problem, we still implement one model using all independent variables (which eliminates the same features) and one with the features selected above, allowing it to set additional coefficients to zero. However, the former is referred to as Lasso without feature selection (noting that feature selection is still implicitly performed), while the latter is referred to as Lasso with feature selection (noting that feature selection is implicitly performed a second time)

Lasso without feature selection scores a better accuracy of 75.91% compared to 75.58% with feature selection. Without feature selection, 71 out of 98 "Trump predictions" are correct, while 71 out of 117 actual Trump voters were classified correctly, obtaining a precision and a recall of 72.45% and 60.68%, respectively. While Lasso with feature selection decreases the model's precision score (85 out of 127, leading to 66.93%), it substantially improves the recall to 72.65%, with 85 out of 117 Trump voters identified correctly (producing the exact same confusion matrix as the Logistic Regression).

Like the Logistic Regression, Lasso (without feature selection) performs better when performed out-of-sample than in-sample (accuracy of 75.91% compared to 73.79%, respectively).

Ridge

Similar to Lasso, Ridge also introduces a penalty term to the cost function of a (Logistic) Regression model that penalizes large coefficients. However, contrary to Lasso, Ridge penalizes large squared coefficients and therefore prevents corner solutions, i.e. Ridge shrinks less important coefficients, but never sets them to zero. Therefore, shrinkage without implicit feature selection is performed. However, we still use one specification with and without feature selection by the Lasso Regression described above.

Ridge without feature selection predicts with a poorer accuracy than both previous models without feature selection (74.59%). However, the accuracy slightly improves to 74.92% when performed with feature selection. Both specification produce almost the same predictions. The only difference is that the specification with feature selection classifies one more Trump voter correctly (62 out of 117) compared to Ridge without feature selection (61 out of 117). Nevertheless, both specifications perform poorly when classifying Trump voters correctly (recall scores of 52.99% and 52.14%, respectively. However, compared to both previous models, both Ridge specifications achieve high precision scores (74.70% with and 74.39% without feature selection). Ridge seemingly improves the classification of those who did not vote for Trump, while worsening the identification of those who did.

Contrary to the previous models, Ridge (without feature selection) performs slightly better in-sample than out-of-sample (accuracy of 74.78% compared to 75.59%, respectively).

Elastic Net

Like Lasso and Ridge, an Elastic Net builds on a linear (Logistic) Regression by introducing a penalty term to its cost function. However, an Elastic Net can be seen as a compromise between Lasso and Ridge, as the term is a linear combination of the penalties of Lasso and Ridge. In our case, hyperparameter tuning assigns a weight of 0.1 to the Lasso penalty term (and therefore 0.9 to the Ridge penalty term).

Without feature selection, the Elastic Net produces the exact same predictions as Ridge without feature selection, achieving an accuracy of 74.59%, a relatively high precision of 74.39%, and a poor recall of 52.14%. This seems natural, as the Ridge penalty term is assigned a higher weight than the Lasso penalty term. However, Elastic Net with feature selection has an overall poor performance compared to all previous models, achieving an accuracy of only 70.63%. Compared to the specification with feature selection it produces worse predictions for individuals who did not vote for Trump (precision of 67.95%) and also for individuals who did vote for him (recall of 45.30%).

As the Elastic Net without feature selection produces the same predictions as Ridge without feature selection, its in- and out-of-sample performance are also the same (accuracy of 74.78% and 74.59%, respectively).

Ridge Classifier

We perform one last linear model by using scikit-learn's Ridge Classifier function. This function creates a model that utilizes the same penalty term as Ridge, but differs in that it first converts the binary target values into {-1, 1} and then treats the prediction problem as a linear regression task. Therefore, the predicted class correspond to the sign of the predicted value. While this approach often performs similar in cross-validation scores, it often can be substantially faster than Logistic Regressions.

Indeed, without feature selection, this approach produces the best overall predictive performance with an accuracy of 76.90%. With feature selection, the achieved accuracy of 74.59% does not stand out compared to the other models, predicting the same number of items correctly as Ridge without feature selection, only with better recall (60.68%) and poorer precision (69.61%). For comparison, the Ridge Classifier function without feature selection identifies Trump voters with a recall of 64.10% and those who did not vote for him with a precision of 72.82%.

The Ridge Classifier function performs substantially better out-of-sample (accuracy of 76.90%) than in-sample (accuracy of 73.08%).

Decision Tree

Leaving linear (logistic) Regressions behind, we perform a first non-parametric model. A Decision Tree (in our case Classification Tree) uses recursive binary splitting to grow. This means that it partitions the data based on the features (nodes) that provide the highest information gain about the target variable. For classification trees with qualitative outcomes, the most commonly occurring outcome is of interest in a subregion. The Classification Tree uses the classification error rate as a criterion on the basis of which it performs its binary splits. This partitioning is performed until a stopping rule is reached, e.g. maximum number of nodes or minimum number of partitions in an observation. Based on the resulting partitions the classification tree uses an "if-then" rule operating through a tree-structured decision process in order to reach a classification decision. While complex trees reduce the bias, they are also prone to overfitting. The decision tree we obtain has 10 nodes. It is notable and seems natural that the set of nodes is similar to the set of features with high coefficients in the Lasso Regression. The top ten features of the tree are presented in **Figure 8** in the appendix.

The decision tree with and without feature selection achieves an overall poor performance. While it only achieves an accuracy of 67.99% without feature selection, it performs even worse with feature selection, scoring an accuracy of only 65.02% (the worst out of all models). Without feature selection, it achieves a precision of 64.29% and a recall of only 38.46%. However, the tree with feature selection performs even poorer in predicting both Trump voters and individuals who did not vote for Trump, scoring a precision of 57.97% and a recall of 34.19%.

The in- and out-of-sample comparison supports the statement that decision trees are prone to overfitting. While the Decision Tree (without feature selection) achieves an out-of-sample accuracy of 67.99%, it performs substantially better in-sample with an accuracy of 71.80%.

Random Forest

Unlike the single Decision Tree, a Random Forest is made up of multiple decision trees. Each tree is trained on a different subset of the data and combines the final prediction is decided by averaging over the predictions of all the trees. This in turn can reduce the variance and prevent the model from overfitting. The top ten features of our random forest are visualized in **Figure 9** in the appendix.

The Random Forest without feature selection overall performs relatively well in predicting voting behavior with an accuracy of 75.58%. The Random Forest without feature selection performs poorer, achieving an accuracy of 73.60%. Without feature selection, the performance difference in classifying Trump voters and non-Trump-voters is not as large as in the models presented so far (recall of 66.67% and precision of 69.03%). This changes for the specification with feature selection, which is significantly better at classifying those who did not for Trump (precision of 72.30%) than it is at identifying Trump voters (recall of 51.28%).

Like the single Decision Tree, the Random Forest performs better in-sample (accuracy of 76.63%) than out-of-sample (accuracy of 75.58%). However, the difference in performance is not as large.

Bagged Trees

Bagged Trees are similar to a Random Forest in that the model also builds multiple decision trees. However, contrary to the Random Forest model, the single trees are identical and are trained on different subsets of the data. Like a Random Forest, Bagged Trees can also reduce overfitting.

Although they follow a similar approach as the Random Forest, Bagged Trees produced poorer predictions, both with and without feature selection (accuracy of 73.60% and 70.63%, respectively). Like the previous tree-based models, Bagged Trees were substantially better at classifying individuals who did not vote for Trump than individuals who did. Without feature selection, Bagged Trees achieved a precision of 71.76% and a recall of 52.14%, while with feature selection, the precision score is 76.50% and the recall is only 46.15%.

Again, Bagged Trees performed better in-sample (accuracy of 75.36%) than out-of-sample (accuracy of 73.60%).

Boosted Trees

Another tree-based approach are Boosted Trees. Unlike Bagged Trees, Boosted Trees grow single trees sequentially while using information from previously grown trees. The main concept behind Boosted Trees is that they apply single trees on a modified version of the data using the residuals of the previous trees weak trees, i.e. each tree effectively tries to correct the previous tree's mistakes. Therefore, Boosted Trees can predict more accurately than Random Forest, but are also more likely to overfit on the training data.

However, Boosted Trees without feature selection performed poorer than the Random Forest and the Bagged Trees without feature selection (accuracy of 72.28%). Unlike the other tree-based models, they performed better with feature selection, achieving an accuracy of 73.93%. Again, they had significantly higher precision scores than recall scores with a precision scores of 70.51% without feature selection and 71.43% with feature selection, and recall scores of only 47.01% with and without feature selection.

Again, Boosted Trees performed better in-sample (accuracy of 75.64%) than out of sample (accuracy of 72.28%).

Neural Network

Neural Networks are a nested non-linear regression class inspired by the structure and function of the human brain. They consist of layers of interconnected nodes (neurons) that each receive information from the input nodes (predictors) and process and transmit it to the output nodes. Neural Networks work based on the concept of forward propagation. The layers between input and output nodes are called hidden layers. Hidden nodes are neither predictors nor output, but instead receive information in the form of a weighted sum of the input features, which they transform according to a non-linear transfer function (often sigmoid) and transmit it to the next highest layer. Finally, the output value that is supported by the highest evidence is chosen. Neural Networks generally face a trade-off between bias and variance, as many layers provide a more flexible structure, but make the model more prone to overfitting. In our case, two hidden layers with 75 neurons each, produced the best result through hyperparameter tuning.

In our case, the Neural Network produced poor prediction with and without feature selection, scoring an accuracy of 69.97% and 67.66%, respectively. It had a similar predictive quality in classifying Trump voters and individuals who did not vote for Trump. The Neural Network without feature selection achieved a precision and recall score of 58.12%, while the Neural Network with feature selection classified with a precision of 60.66% and a recall of 63.25%.

Similar to the tree-based models, the Neural Network performed better in-sample than out-of-sample (accuracy of 70.83% compared to 67.66%, respectively).

Ensemble

Finally, an Ensemble combines multiple models (of the same or different type) as a weighted average of several machine learning algorithms. One example for an Ensemble would be a Random Forest. However, the last prediction model we use is a combination of different model types. We combine Ridge Classifier (OLS) with a Random Forest (tree based) and a Logistic Regression.

However, compared to other models, this combination did not produce outstanding predictions. The Ensemble without feature selection achieved an accuracy of 75.25% and an accuracy of 74.26% with feature selection. However, it performed well classifying Trump voters with a recall of 74.36% with and without feature selection. At the same time, the Ensemble produced poorer classifications of individuals who did not vote for Trump, with a precision score of 65.91% without feature selection and 64.44% with feature selection.

Finally, the Ensemble performed slightly better out-of-sample than in-sample (accuracy of 75.25% compared to 73.52%, respectively). This seems natural, as the Ridge Classifier and the Logistic Regression had a better out-of-sample performance, while the Random Forest performed only slightly better in-sample.

4. Conclusion

We predicted whether participants in the 2018 GSS voted for Donald Trump in the 2016 presidential election based on several factors, including basic background characteristics, work characteristics, and political attitudes. We did so by using different types of machine learning algorithms, ranging from linear OLS and logistic regressions to non-linear tree-based algorithms, a Neural Network, and a combination of them. On average, the linear models performed better than the non-linear ones. Furthermore, the Ridge Classifier (OLS) without feature selection achieved the highest predictive accuracy, while the lowest accuracy was achieved by the Decision Tree with feature selection. Some linear models were better at identifying Trump voters, some were better at identifying individuals who did not vote for Trump. However, non-linear models were mostly better at classifying the latter. Also, all non-linear models performed better in-sample than out-of-sample. Finally, while most models performed predictions with similar accuracy, the differences between the best and the poorest models were quite substantial, with the maximum difference in accuracy being 9.24 percentage points.

However, the highest achieved accuracy being 76.90% shows that none of these models would have been sufficiently accurate to predict an election outcome as close as the 2016 presidential election. Additionally, many of the strongest predictors we used for, such as attitude towards the death penalty and government spending on childcare, are usually private information and can only be collected through surveys. This limits data availability and prevents machine learning algorithms from being trained with large trainings sets which is usually beneficial. However, social media collects massive amounts of data and the field of machine learning progresses with a high speed, so who knows what is possible? (Chandler and Tsai, 2001) (Hendricks and Schill, 2017) (Reny et al., 2019)

5. References

Chandler, C.R., Tsai, Y., 2001. Social factors influencing immigration attitudes: an analysis of data from the General Social Survey. Soc. Sci. J. 38, 177–188. https://doi.org/10.1016/S0362-3319(01)00106-9

Hendricks, J.A., Schill, D., 2017. The Social Media Election of 2016, in: Denton Jr, R.E. (Ed.), The 2016 US Presidential Campaign: Political Communication and Practice, Political Campaigning and Communication. Springer International Publishing, Cham, pp. 121–150. https://doi.org/10.1007/978-3-319-52599-0_5

Reny, T.T., Collingwood, L., Valenzuela, A.A., 2019. Vote Switching in the 2016 Election: How Racial and Immigration Attitudes, Not Economics, Explain Shifts in White Voting. Public Opin. Q. 83, 91–113. https://doi.org/10.1093/poq/nfz011

6. Appendix

Figure 1 – Binary Items

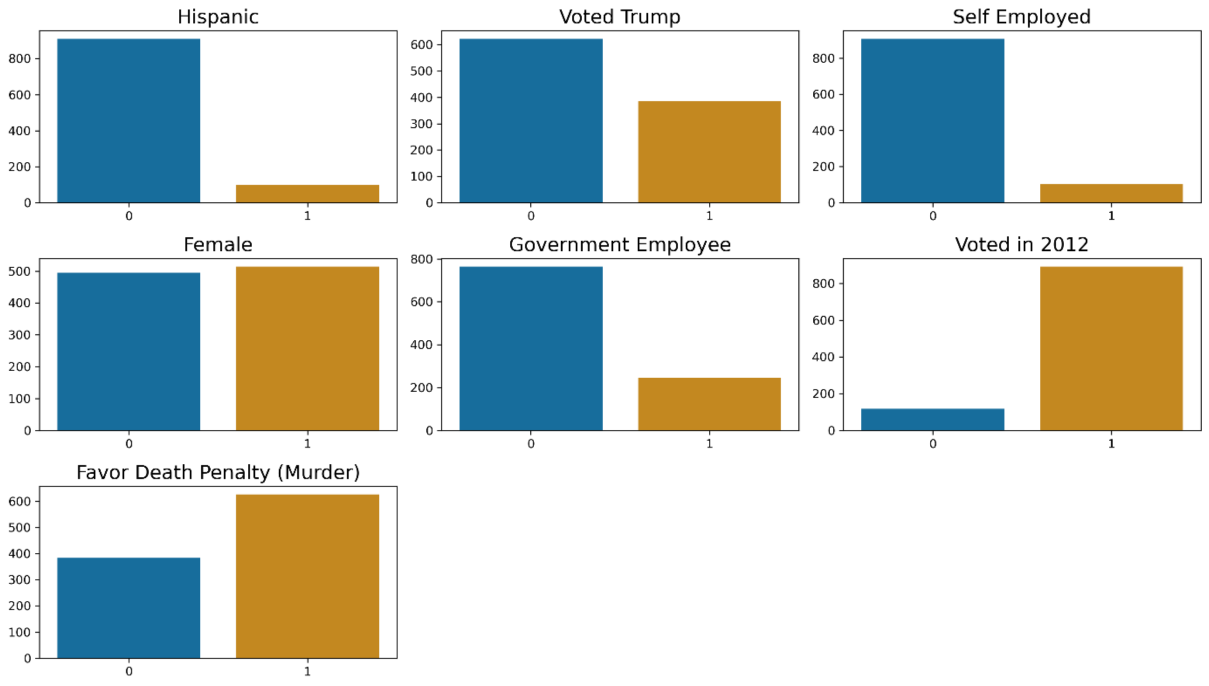


Figure 2 – Numerical Items

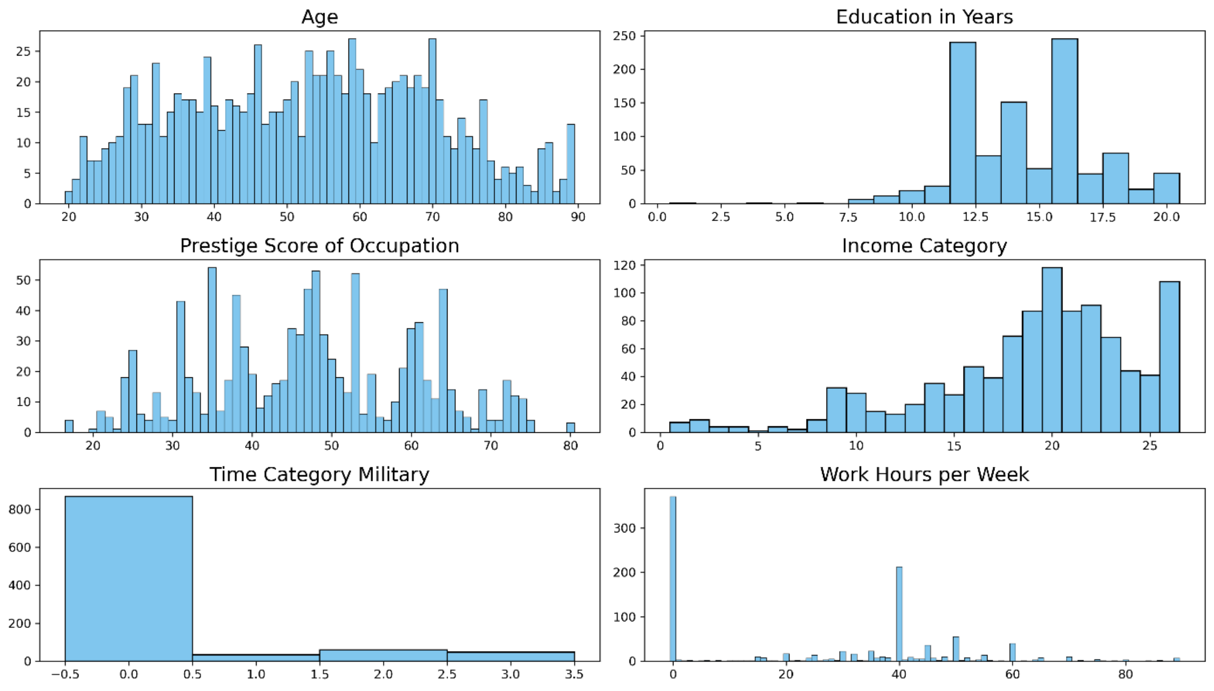


Figure 3 – Categorical Items

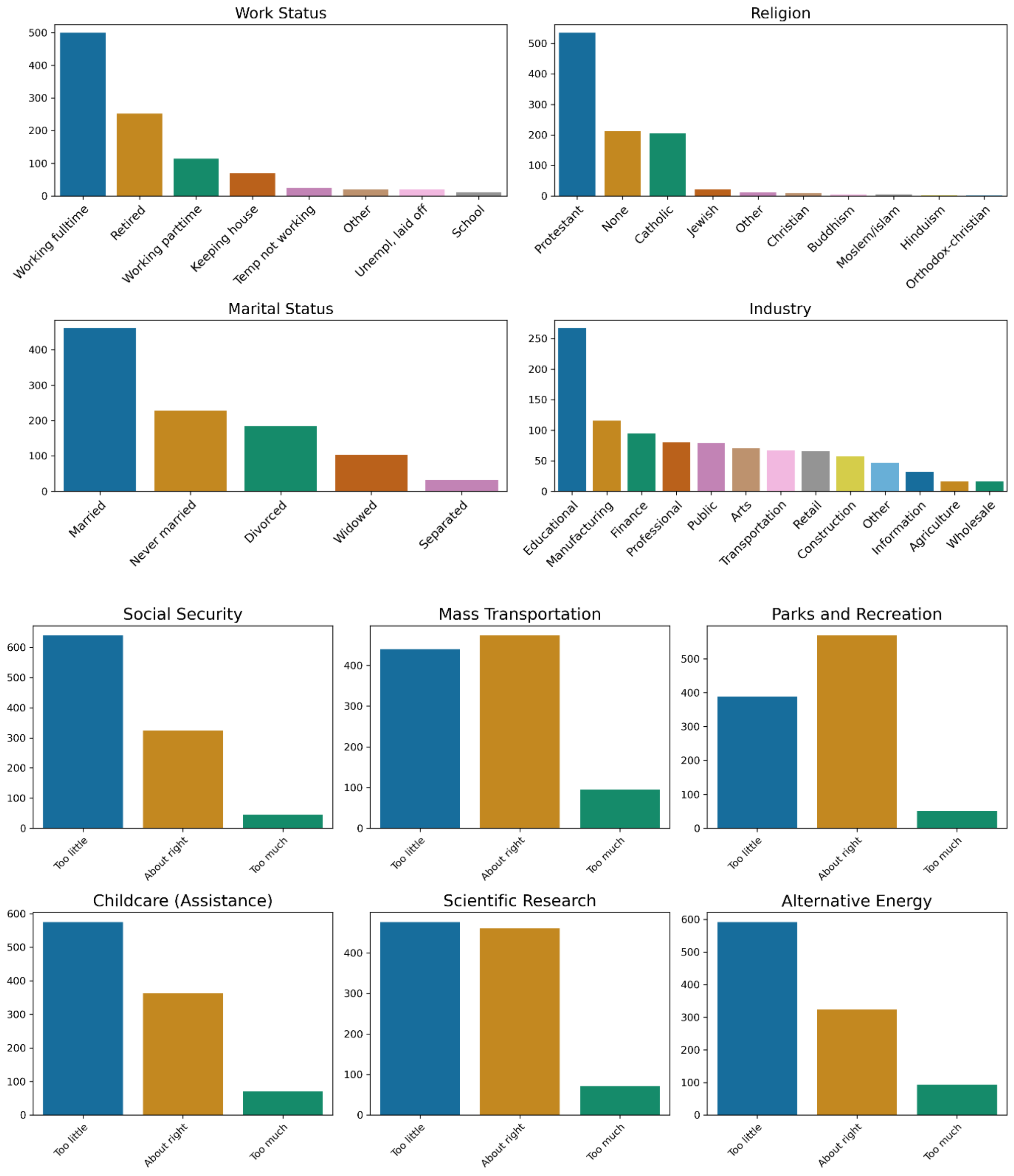


Figure 4 – Feature Selection with Lasso

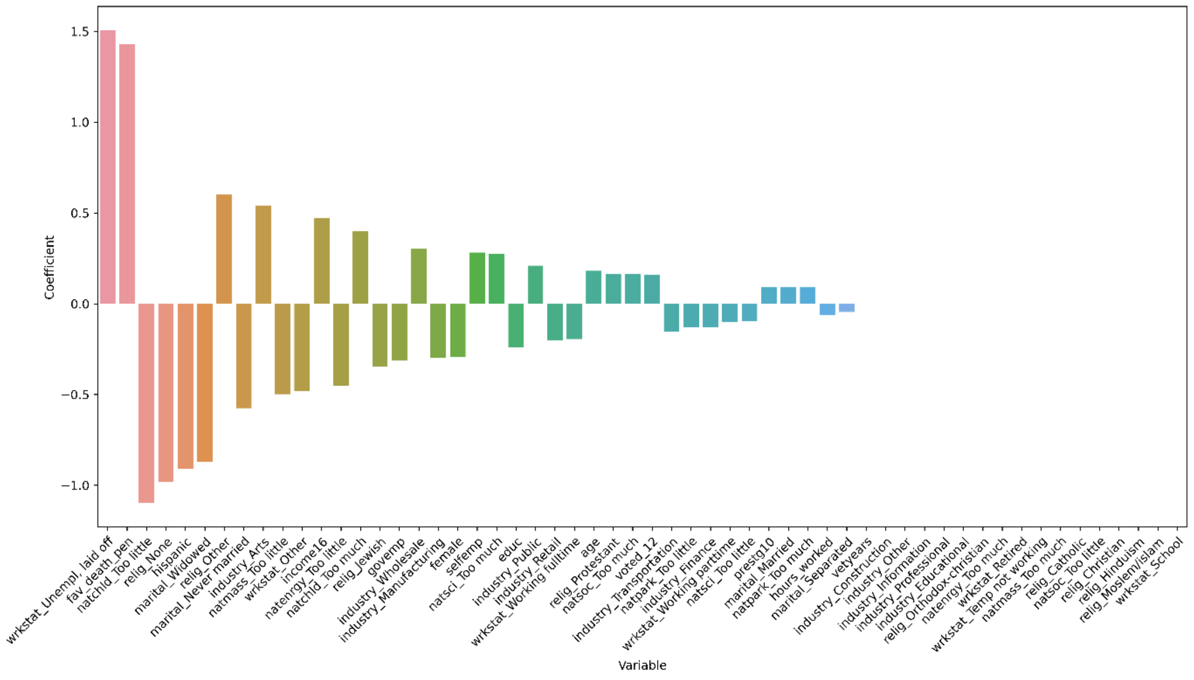


Figure 5 – Logistic Regression Coefficients

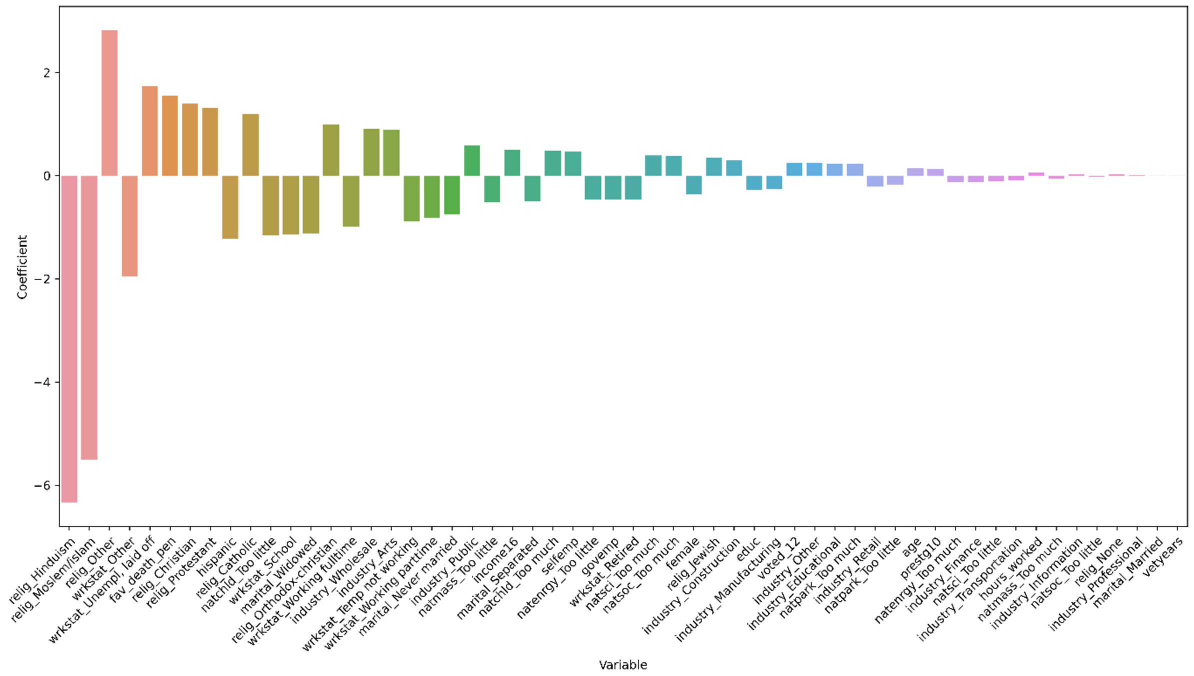


Figure 6 – Accuracy Scores

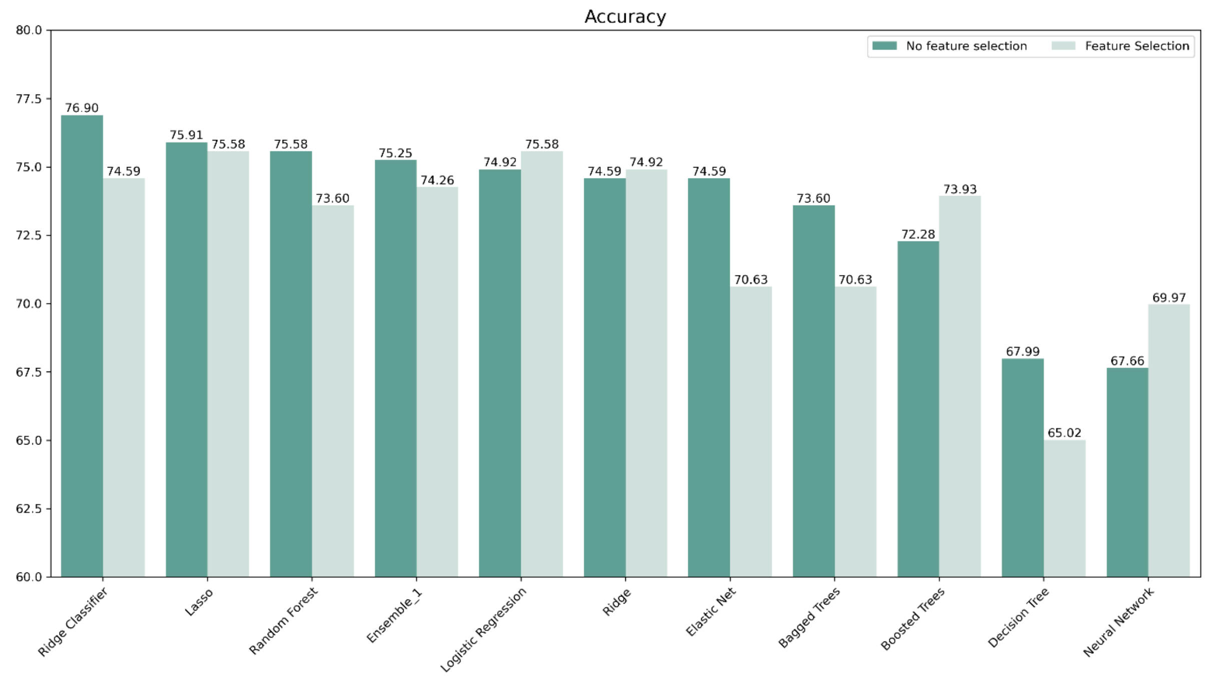


Figure 7 – In-Sample and Out-Of-Sample Comparison

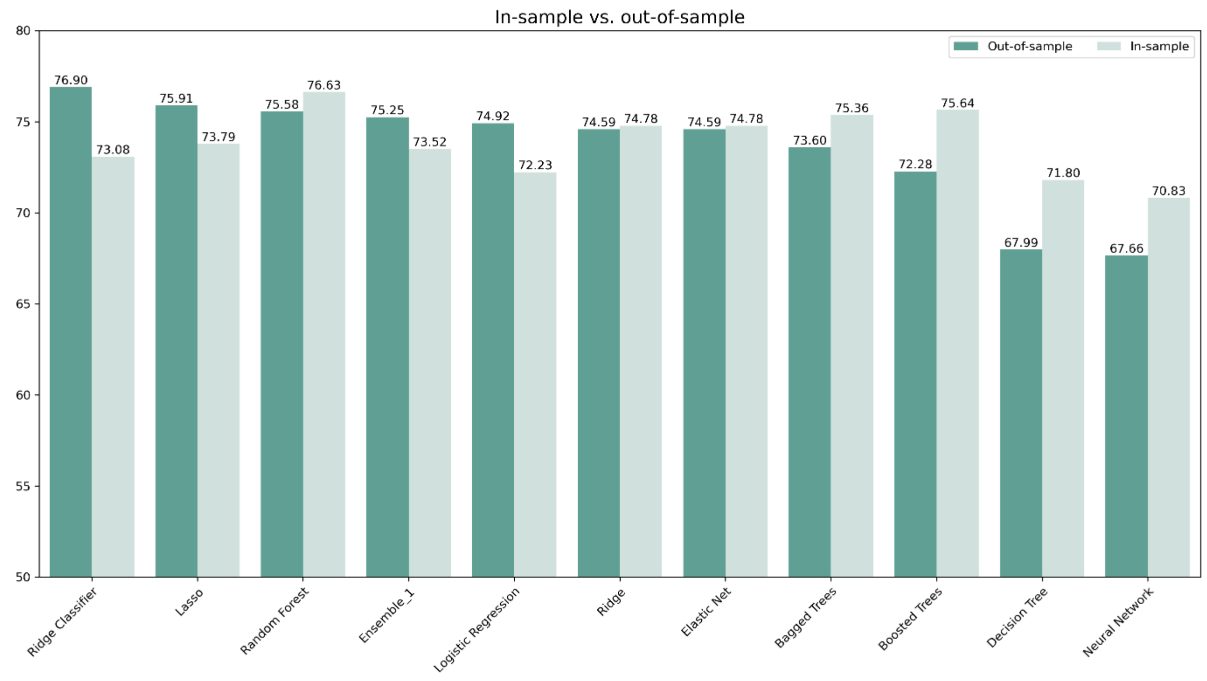


Figure 8 – Decision Tree Features

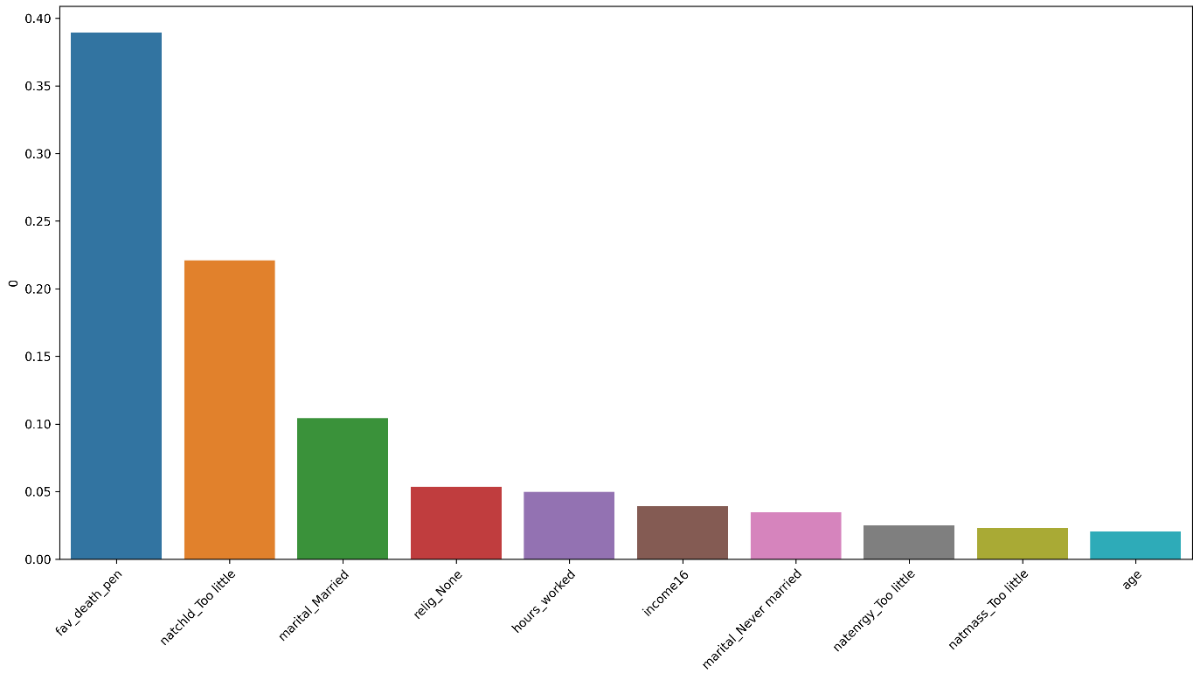
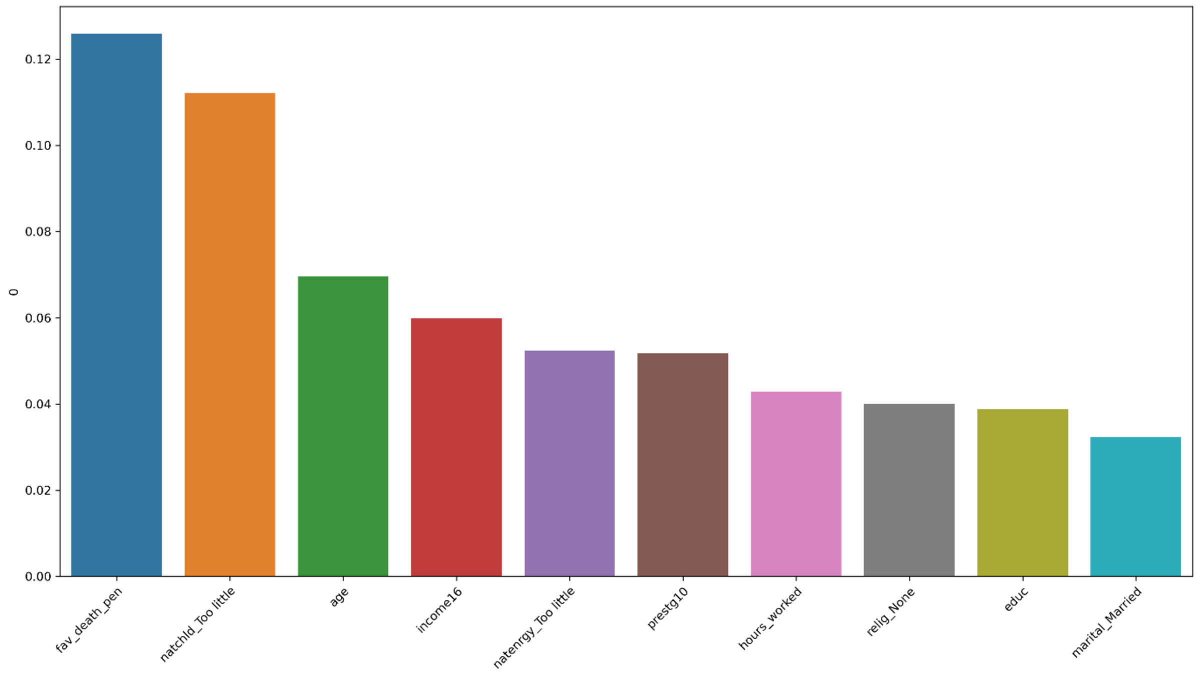


Figure 9 – Random Forest Features


fav_death_pen
natchld_Too little
age
income16
natenrgy_Too little
prestg10
hours_worked
relig_None
educ
marital_Married