Exploring Political Contributions Through the Lens of Machine Learning Algorithms

Tyler Dickson
Georgia Institute of Technology
tdickson30@gatech.edu

***Abstract*—In the rapidly evolving political landscape, few studies have applied targeted predictive analysis to understand political donation patterns in response to societal events. While existing research covers major trends, it rarely focuses on socio-economic factors that influence donations. This research builds on that by using machine learning models to predict donation patterns tied to political affiliations and reactions to school shootings.**

Through the examination of two distinct datasets, this study applies Support Vector Machines (SVM) and Extreme Gradient Boosting (XGBoost) to analyze political contributions with aims of predicting donations to either Democrat or Republican affiliated committees based on a set of socio-economic and donation-specific variables. Additionally, Neural Networks and K-Nearest Neighbors (KNN) models are utilized on a separate dataset to assess the impact of school shootings on donations to pro-gun and anti-gun affiliated committees.

The results show that socio-economic variables can predict donation behavior with high accuracy, addressing the existing research gap. These findings answer key questions about donation behavior predictors and the influence of crisis events on ideological leanings.

I. Introduction

As election season approaches, political donations face heightened scrutiny. These donations are crucial to political campaigns, providing the resources needed for candidates to reach constituents, increase their visibility, and ultimately secure elections. Donations fund essential campaign activities, such as advertising, travel, rallies, and staffing, making them the financial backbone of any political effort. By analyzing contributions to political committees, we can gain valuable insights into the political leanings and priorities of different segments of the U.S. population.

Some of the most contentious of political committees are those that advocate for gun rights. Since 2002, there have been 354 recorded school shootings in the U.S., resulting in 177 fatalities and 369 injuries. [1] These tragic events have placed gun policy at the center of debate, influencing how people people politically donate.

In this project, we leverage machine learning algorithms to analyze trends in two classification problems. Our classification problems involve predicting an output category based on a set of inputs. The goal is to assign each instance to one of several predefined classes, using specific features to guide the prediction. Within our datasets, the target variable takes on discrete values. We construct our model to learn from patterns in the data to assign new instances to the correct class.

In Dataset 1, we apply Support Vector Machines (SVM) and Extreme Gradient Boosting (XGBoost) to analyze political contributions to predict donations to either Democrat or Republican affiliated committees based on a set of socio-economic and donation-specific variables. In Dataset 2, we apply Neural Networks and K-Nearest Neighbors (KNN) to assess the impact of school shootings on donations to pro-gun and anti-gun affiliated committees.

II. Dataset 1: Predicting Political Party Donations

A. *Hypothesis for Dataset 1*

Given a set of defined inputs—city, state, zip code, occupation, day, month, year, and transaction amount—it is possible to predict, with a high degree of certainty, whether an individual will contribute to a Democrat-affiliated committee or a Republican-affiliated committee.

B. *Data Preprocessing*

The Federal Election Commission (FEC) maintains comprehensive records of yearly donations. This analysis focuses on *Contributions by individual* and made in 2023, which are available at Individual Contributions Data Description.

Initially, the dataset consists of 21 columns with varying relevance to our model. We first narrow down the number of inputs by dropping unnecessary columns, keeping only those relevant to our hypothesis. The dataset contains over 29 million rows, making memory optimization essential. We convert appropriate columns to categorical variables and downcast floats and integers.

The dataset does not include separate columns for day, month, and year, which are important inputs for our model. However, the 'image number' variable contains the date information in an irregular format. We extract the relevant date components and create new columns for day, month, and year to use as predictive variables.

To determine party affiliation, we incorporate the FEC's *Committee master* dataset. Description for this dataset can be found at Committee Master Data Description. Both *Committee Master* and *Contributions by individual* datasets contain a common variable, 'committee ID', which allows us to merge the two datasets. We then drop all columns unrelated to our hypothesis.

Next, we create a new column for party affiliation. A value of 1 is assigned to committees affiliated with the Republican Party, and 0 is assigned if the committee is affiliated with the Democratic Party. Finally, we split the data with a ratio of 80/20 percent training test split respectively.

C. *Support Vector Machine: Kernel Selection*

The first model was created using SVM. The goal of the SVM algorithm is to find an optimal decision boundary, or hyperplane, that maximizes the margin between classes in an N-dimensional space [2]. SVM is especially powerful in classification problems because of its ability to maximize this margin, leading to better generalization.

During the development of the SVM model, I initially tested the accuracy using three different kernels: Linear, Polynomial, and Radial Basis Function (RBF). Each kernel captures different relationships in the data, allowing for the exploration of different ways to model feature complexity. [3] Accuracy scores produced by each kernel is displayed in Table I

TABLE I
SVM KERNEL ACCURACY SCORES

Kernel	**SVM Kernel Accuracy Scores**			
	Linear	***RBF***	***Polynomial (Degree 3)***	***Polynomial (Degree 4)***
Accuracy	0.83095	0.831775	0.83095	0.83095

[a] A Degree change in Polynomial Kernel results in identical accuracy scores, indicating nonlinearity is not being by introduction of a quartic decision boundary.

The scores produced by the Linear and Polynomial (degree=3) kernels are identical (0.83095), indicating that the decision boundary produced by these two kernels is very similar. This suggests that the data may either be linearly separable or have simple interactions that don't require a higher-order polynomial.

Running the polynomial kernel with a higher degree of 4 produced the same accuracy score, indicating the introduction a quartic decision boundary did not improve performance. This implies that further increasing the dimensional space did not capture any additional meaningful patterns in the data. While higher-dimensional spaces can potentially make the data linearly separable, this also increases the risk of overfitting, especially when the underlying patterns do not benefit from this added complexity. [4]

The RBF kernel produced the highest accuracy (0.831775), slightly outperforming the linear and polynomial kernels. This suggests that while the data exhibits a some nonlinearity. The RBF kernel is well-suited for capturing complex, nonlinear patterns by mapping the data into an infinite-dimensional space. However, the small improvement in accuracy suggests that the nonlinearity present in the dataset is subtle and doesn't require a highly complex decision boundary.

The RBF kernel's performance demonstrates its flexibility in handling nonlinear data, but the minimal increase in accuracy compared to the linear kernel indicates that the data does not have significant nonlinear relationships.

D. *Support Vector Machine: Learning Curve Analysis*

Learning curve analysis offers valuable insights into how the SVM model's performance learns with increasing data volume and complexity. Utilizing cross-validation, the model's capability to generalize effectively on unseen data is assessed. Cross-validation partitions the dataset into k distinct segments, or folds, to provide a more robust estimate of the model's performance when compared to the initial 80/20 training to test split. In our analysis, k was set to 5, meaning the dataset was divided into five folds. Each fold served sequentially as the validation set, with the remaining four folds used for training. The accuracy score for each fold was recorded and averaged, providing a more accurate approximation of our sample performance. [5]

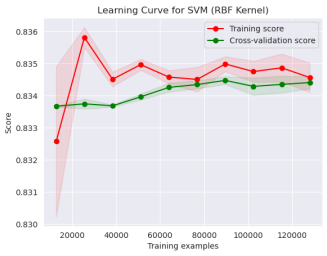

Fig. 1. SVM: Learning curve; 200k

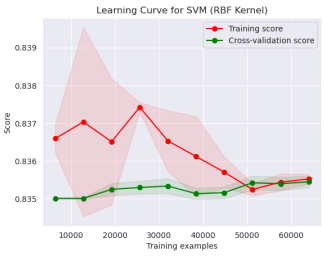

Fig. 2. SVM: Learning Curve; 100k

The learning curves depicted in Figures 1 and 2 provide key insights into the model's fit and scalability. In Figure 1 shows a learning curve for the model trained on 200k samples. The initial lower training score which spikes with the introduction of more data suggests rapid learning but not overfitting. This can be deduced by the stabilization of scores and narrow variance (indicated by the shaded area). Reduced variance signifies consistent performance across different cross-validation folds, highlighting model robustness and reduced sensitivity to specific training splits.

Comparatively, Figure 2 shows a learning curve for the model trained on 100k samples. Although a wider initial variance may indicate a greater sensitivity to the data's partitioning, this curve does converges around 50k samples, suggesting that beyond this point, additional data contributes minimally to performance. This is displayed in both curves' convergence, indicating that larger datasets do not necessarily enhance the model's generalization capabilities beyond a certain threshold.

Differences highlighted in these learning curves emphasizes the importance of choosing an optimal sample size. While the model initially stabilized around 100k observations, the convergence of training and validation scores at lower sample sizes suggests a potential for reducing the training dataset without compromising model accuracy or generalization capability. This finding is important when configuring efficient resource allocation when dealing with SVM, which generally utilizes significant compute resources.

Both learning curves ultimately indicate that the SVM model, regardless of the sample size, generalizes well to new

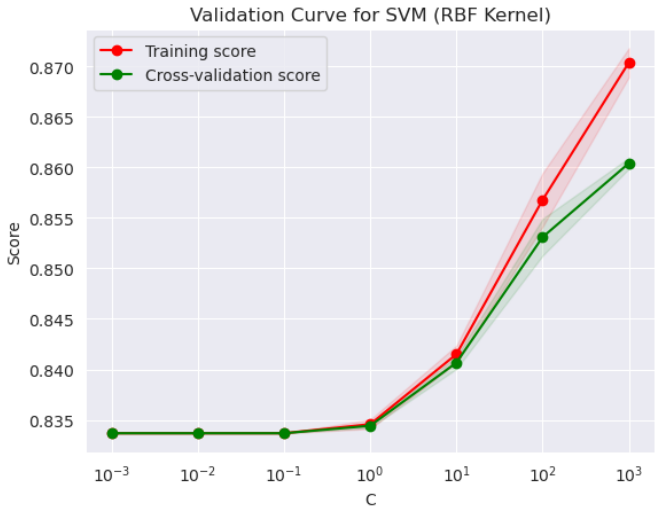

Fig. 3. Validation curve with 200k samples

data as long as the model has an adequate number of examples to learn the underlying patterns effectively.

E. Support Vector Machine: Validation Curve Analysis

Plotting the validation curve provides insights into how model parameters influence its performance. Observing both the training and validation scores over a range of values for hyperparameters helps identify overfitting or underfitting, tune hyperparameters, and balance bias and variance.

Figure 3 displays the validation curve for our SVM model trained on 200k samples. The hyperparameter C, as denoted on the x-axis represents the hyperparameter C on a range from 0.001 to 1000. C is the regularization parameter in SVM that controls the trade-off between achieving a low error on the training data and minimizing the model complexity for better generalization. [6] . This is apparent in Figure3 when C is small (ranges from **0.001** to **0**), both training and validation scores are low, which indicates underfitting in instances where the model is too constrained to effectively capture the pattern of the data. As C increases, both scores improve significantly, suggesting that less regularization allows the model to fit the data better. At C = **100**, the training and cross validation scores reach their optimal level. Beyond this, the scores begin to plateau, indicating that further increases in C might lead to overfitting. [6]

While the validation curve can assist us in selecting our C parameter, we can further refine our choice through the implementation of a grid search, incorporating other hyperparameters such as gamma, which defines how much of an influence of a single training example has. [3]

F. Support Vector Machine: Tuning Hyperparameters

A grid search can identify best hyperparameters for the model. Based on the insights from our validation curve, we know that the target range for C should be close to C=100. Running the search we identify values for C and gamma while evaluating the model's performance across these values to determine optimal settings.

The model's grid search identified the best parameters as C=100, gamma=1, and cross-validation score of 0.8137375, aligning closely with the insights derived from the validation curve. These results validate our earlier analysis and confirms the effectiveness of the selected hyperparameter values in optimizing model performance.

To further visualize the effects of hyperparameter tuning, Table II represents our results prior to tuning hyperparameters, while Table III depicts results post tuning.

TABLE II
CLASSIFICATION REPORT FOR SVM MODEL BEFORE TUNING PARAMETERS

Class	Precision	Recall	F1-score	Support
Democrat Donation	0.83	1.00	0.91	33238
Republican Donation	0.76	0.01	0.02	6762
Accuracy			0.83	40000
Macro Avg	0.80	0.50	0.46	40000
Weighted Avg	0.82	0.83	0.6	40000

TABLE III
CLASSIFICATION REPORT FOR SVM MODEL AFTER TUNING PARAMETERS

Class	Precision	Recall	F1-score	Support
Democrat Donation	0.91	0.87	0.89	16554
Republican Donation	0.47	0.56	0.51	3446
Accuracy			0.82	20000
Macro Avg	0.69	0.72	0.70	20000
Weighted Avg	0.83	0.82	0.82	20000

a) Key Takeaways After Tuning Hyperparameters:

- Improved Recall for Republican Donations: After tuning, there was a significant improvement in the recall for Republican donations, increasing from **0.01** to **0.56**. This improvement indicates the model became much better at identifying positive instances (democrat donations).
- Decrease in Precision for Republican Donations: The precision for Republican donations decreased from **0.76** to **0.47**, indicating a trade-off between recall and precision.
- Improved F1-score for Republican donations: The F1-score for Republican donations improved dramatically from **0.02** to **0.51**, suggesting a better balance between precision and recall.
- Decrease in Overall Accuracy: The overall accuracy slightly decreased from **0.83** to **0.82**. However, the macro average and weighted average scores improved, suggesting a more balanced model that performs better across both classes, not just the majority class. [7]

G. Extreme Gradient Boosting (XGBoost)

This section explores the Extreme Gradient Boosting (XGBoost) algorithm and its application to our dataset, where the dependent variable y represents whether a donation is predicted to be Republican-affiliated. XGBoost provides several

advantages over SVM, notably in efficiency. Unlike SVM, XGBoost does not require computations in high-dimensional space, which significantly reduces computational costs. Additionally, XGBoost includes built-in cross-validation at every iteration, eliminating the need for separate cross-validation procedures.

As with the SVM model, we start by splitting the data into training and test sets using an 80/20 split. Due to the computational efficiency of XGBoost, we proceed to tune hyperparameters and fit the model based on these identified optimal parameters, and increase our sample size to 500k observations. Unlike our approach with SVM, we employ randomized search instead of grid search due to its effectiveness in managing large hyperparameter spaces.

Hyperparameter tuning results in a subsample of **0.8**, scale position weight of **1**, n-estimators of **100**, max depth of **10**, learning rate of **0.1**, gamma of **0**, and colsample by tree of **0.7**. Results of applying these hyperparameters to our model is displayed in Table IV.

TABLE IV
CLASSIFICATION REPORT FOR XGBOOST AFTER TUNING PARAMETERS

Class	Precision	Recall	F1-score	Support
0	0.93	0.97	0.95	82314
1	0.83	0.64	0.73	17686
Accuracy			0.91	100000
Macro Avg	0.88	0.81	0.84	100000
Weighted Avg	0.91	0.91	0.91	100000

*a) **Key Takeaways After Tuning Hyperparameters**:*

- Complex Model Configuration: The use of a max depth of 10 and 100 n-estimators allows XGBoost to adapt to a complex model configuration, with 500k observations.
- Preventing Overfitting: A subsample and colsample bytree results show the model samples **80%** of the training data and **70%** of the features when building each tree, which can prevent overfitting.
- High Precision: An average precision of **0.88** across classes suggests the model is quite accurate in its predictions.
- Recall Disparity: The average recall of **0.81** suggests room for improvement in detecting positive samples more consistently.
- Model Performance: The weighted average score of **0.91** reflects the precision, recall, and F1-score while considering class distribution imbalance. The high weighted average confirms the model's robust performance across the dataset. [7]

H. Analyzing Results of Dataset 1

In classification problems, the objective is to determine how various input features (x variables) can predict a discrete output (y variable). After implementing our model, we evaluated which features are most influential in predicting whether donations are affiliated with Democrats or Republicans. Figure 4 illustrates the importance of various features in predicting

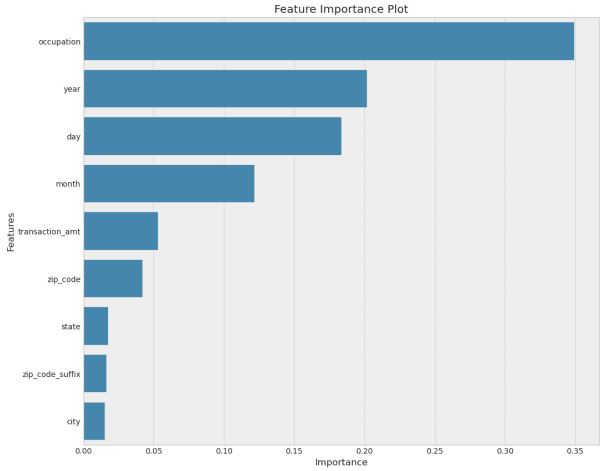

Fig. 4. Feature Importance predicting Republican Affiliated donations

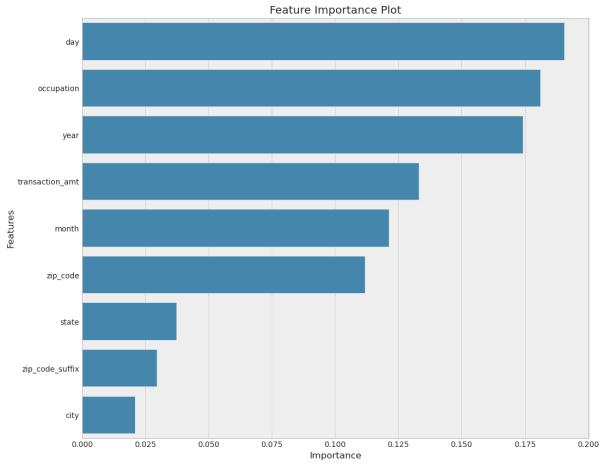

Fig. 5. Feature Importance Predicting Democrat Affiliated donations

Republican-affiliated donations, while Figure 5 represents the the importance of various features in predicting whether a donation is likely to be to a Democrat affiliated group. Results displayed in Figures 4 and 5 reveal significant differences in feature importance based on the political affiliation targeted by the donations.

For Republican-affiliated donations, the donor's occupation emerges as the most significant predictor. This suggests that demographic and economic factors closely linked to one's profession strongly influence the likelihood of supporting Republican causes.

Conversely, for Democrat-affiliated donations, the specific day of the transaction is the most critical predictor. This may be attributable to the timing of major events, election cycles, or critical issues that mobilize donors. Additionally the model predicts less emphasis on geographical factors such as specific zip codes for democrats. This contrasts with Republican donations, where local factors may play a more significant role.

Both models underscore the importance of 'day' and 'year'

in donation predictions, indicating that fundraising efforts should be strategically timed around specific days known for increased activity and during pivotal times such as election years. Additionally, the significance of occupation across both models suggests that customized messaging that resonates with particular demographics could effectively appeal to potential donors for both parties.

This analysis of feature importance provides valuable insights into how various factors influence donor behavior across the political spectrum, highlighting the need for tailored strategies in outreach and fundraising campaigns.

III. Dataset 2: School Shootings and Pro Gun Donations

A. *Hypothesis for Dataset 2*

Given a set of defined inputs—such as donations to pro-gun and anti-gun affiliated committees, the number of casualties (killed, injured), and the type of school shooting—it is possible to predict, with a high degree of certainty, whether a school shooting will lead to a significant spike in donation amounts to pro-gun or anti-gun affiliated committees.

B. *Data Preprocessing*

The Wall Street Journal maintains a comprehensive list of School Shooting Data from 1999 to the present [1]. Using this alongside FEC committee donation data, the analysis focuses on contributions made between 2002 and 2024. As in Dataset 1, we combine the *Individual contributions* and *Committee master* datasets.

The combined dataset contains over 219 million rows, requiring memory-efficient processing. To manage this, we read only columns relevant to our hypothesis, convert categorical variables, downcast floats and integers, and save intermediate data as pickle files.

Next, we aggregate *Individual contributions* by grouping redundant columns and merge the datasets. Pro-gun and anti-gun committee affiliations are defined as new columns. The school shooting dataset is cleaned by removing unnecessary variables, keeping only predictor variables. Finally, we merge the FEC data with the school shooting dataset for analysis.

C. *Neural Network: Initial Challenges*

Neural Networks are generally able to analyze non linear relationships effectively. Given the interaction with events such as school shootings and donations likely does not represent a linear relationship, a neural network is a logical choice given the complexity of the data.. Whereas traditional linear regression models might struggle to produce accurate results.

One of the initial challenges encountered while training the neural network was severe class imbalance in the dataset. When analyzing the effect of school shootings on donation spikes, the majority of cases exhibited a donation spike, which significantly skewed the data. Initially, I defined a spike as a 15% increase in donations within three days of a shooting, but this led to the network overwhelmingly predicting donation spikes in nearly every instance due to the imbalance between spike and non-spike events. Even after increasing the threshold for a spike to a 4600% rise in donations in comparison to the mean, the imbalance persisted. This resulted in the neural network favoring the majority class, and poor generalization. The model simply predicted donation spikes regardless of the input data. It wasn't until I implemented oversampling of the minority class—instances where no donation spike occurred—that the model began to produce more balanced and accurate predictions, improving its ability to distinguish between spike and non-spike events.

D. *Neural Network: Tuning Hyperparameters*

A critical aspect of designing a neural network is tuning hyperparameters such as the number of layers, neurons, learning rate, activation function, batch size, and the number of epochs. In the initial creation of the model, my goal was to establish a working baseline before moving on to hyperparameter tuning. I initially opted for a straightforward architecture with commonly recommended values.

As a baseline, two hidden layers consisting of 128 and 64 neurons were chosen. This setup allowed the model to capture non-linear relationships while managing complexity. Increasing the number of neurons could enable the model to capture more intricate patterns but may risk overfitting the model. Additionally, the ReLU activation function was chosen due to its efficiency in handling non-linear data while maintaining computational simplicity.

The learning rate controls how much the model's weights are updated during training. An optimal learning rate allows for faster convergence without sacrificing accuracy. A baseline learning rate of 0.001 along with the Adam optimizer was chosen. The number of epochs determines how many times the model passes through the entire dataset during training. The model was set to train over 20 epochs, which provided ample iterations to observe the model's performance.

Using Optuna to tune hyperparameters, significant improvements were observed. Tables V and VI display the initial performance metrics for the neural network before and hyperparameters were tuned. Optuna produced hyperparameters with a learning rate of **0.071**, a first hidden layer (number of neurons) of **128**, a second hidden layer of **64**, and a batch size of **64**.

TABLE V
Classification Report for Neural Network Model Before Tuning Parameters

Class	Precision	Recall	F1-score	Support
No Donation Spike	0.89	0.97	0.93	40
Donation Spike	0.97	0.85	0.90	33
Accuracy			0.92	73
Macro Avg	0.93	0.91	0.92	73
Weighted Avg	0.92	0.92	0.92	73

In the first iteration of the model shown in table V, the training accuracy steadily improved over the course of 20 epochs, reaching a final training accuracy of **81.31%**, and test accuracy of **91.78%**, indicating that it generalized well

TABLE VI
CLASSIFICATION REPORT FOR NEURAL NETWORK MODEL AFTER TUNING PARAMETERS

Class	Precision	Recall	F1-score	Support
No Donation Spike	0.98	1.00	0.99	40
Donation Spike	1.00	0.97	0.98	33
Accuracy			0.99	73
Macro Avg	0.99	0.98	0.99	73
Weighted Avg	0.99	0.99	0.99	73

to unseen data. The classification report shows an F1 score of **0.90** for predicting donation spikes and **0.93** for predicting no donation spikes. Notably, the model performed better at predicting no spikes (recall of **0.97**) compared to predicting spikes (recall of **0.85**).

After tuning the hyperparameters using Optuna, the model's performance significantly improved. The final test accuracy increased to **91.63%**, and a training accuracy improved to **98.63%** . The precision and recall for both classes also improved, with the F1 score for donation spikes increasing to **0.98**, reflecting a much better balance between predicting donation spikes and no spikes. The higher test accuracy and improved classification metrics show that the optimized model generalizes better and makes fewer classification errors.

E. Neural Network: Learning Curve Analysis

By examining the learning curve loss and accuracy, we can develop a more complete understanding of our model. The loss curves depicted in the top graph of Figure 6 reveals how effectively the model is minimizing prediction errors over time, while the accuracy curves provide insights into the overall performance and correctness of the model's predictions.

The training loss learning curve (depicted in blue) demonstrates a sharp decline early in the training process and then levels off, suggesting that the model quickly adapts to the training data and effectively minimizes errors. Whereas the validation loss (depicted in orange) starts higher, indicating initial discrepancies when the model encounters new, unseen data. Around epoch **12** the model experiences some fluctuation, which may signal sensitivity to specific data batches. However, it generally follows a downward trend and begins to align more closely with training loss by the end of the training period—displaying reduced error on validation data. Over time, the model's ability to generalize improves.

The accuracy learning curve reveals how well the model predicts correct outcomes. The training accuracy (depicted in blue) starts lower but rapidly climbs before stabilizing at around **90%** by epoch **5**. This indicates the model learns to classify the training data correctly quite early. The validation accuracy (depicted in orange) shows starts higher, then experiences slight fluctuations—likely reflecting the model's response to varying complexities in different data sets. Ultimately, the model displays its ability to fits well to the training data and adapt to new data effectively.

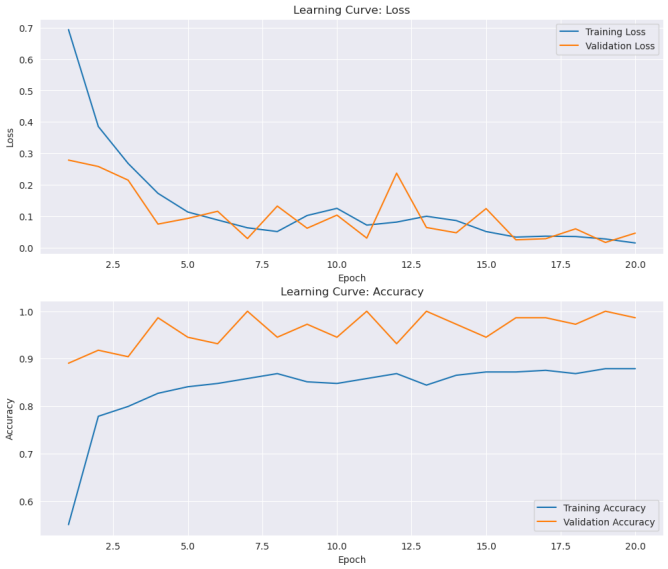

Fig. 6. Enter Caption

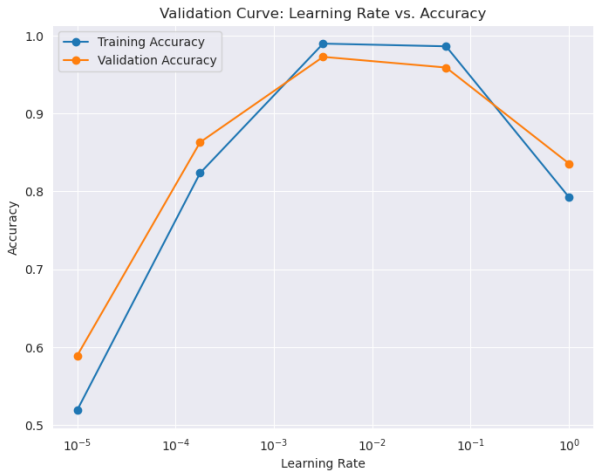

Fig. 7. Neural Network Validation Curve

F. Neural Network: Validation Curve Analysis

Through examination of the validation curve depicted in Figure 7, it's apparent that the curve begins to stagnate when the learning rate reaches **.01**, and decreases sharply in accuracy at **.1**. This validation curve confirms our previously calculated optimal learning rate of **0.071** is correct, and provides balance between learning speed and model convergence.

G. K-Nearest Neighbor (KNN)

KNN is well-suited for classification problems, making it a logical choice for our dataset. KNN operates by assigning an object to the class most common among its k nearest neighbors. The effectiveness of KNN is significantly influenced by the chosen value of k. A small versus large k value affects the bias versus variance tradeoff in the model. A low k value is sensitive to outliers, which can increase variance but reduces

bias. In contrast, a high k value generally reduces variance by establishing broader neighbor classes but can produce a biased model. Therefore, it's crucial to determine an optimal k value. Additionally, a common downfall of KNN occurs with complex data in a high-dimensional space. As more features are added to the dataset, the feature space becomes sparsely populated, which can challenge the classification capability of KNN. [8]

To illustrate the nuances on the effectiveness of our model when values of K are changed, we randomly select a k value of 8, and split the dataset into training-test splits of 80/20. Results of this test are displayed in Table VII.

TABLE VII
CLASSIFICATION REPORT FOR KNN MODEL BEFORE TUNING PARAMETERS

Class	Precision	Recall	F1-score	Support
No Donation Spike	0.84	0.94	0.89	34
Donation Spike	0.94	0.84	0.89	37
Accuracy			0.89	71
Macro Avg	0.89	0.89	0.89	71
Weighted Avg	0.89	0.89	0.89	71

Next, the model is ran again using cross-validation techniques to find an optimal value for k. By utilizing cross-validation, we effectively iterate across the dataset, including values found in both training and test sets. Next, a grid search method is used to search through k values from 1 to 30 to determine which k produces the best performance on unseen data. After running the model, an optimal value for k is found at 5. Test outputs are displayed in Table VIII

TABLE VIII
CLASSIFICATION REPORT FOR KNN MODEL BEFORE TUNING PARAMETERS

Class	Precision	Recall	F1-score	Support
No Donation Spike	0.86	0.88	0.87	34
Donation Spike	0.89	0.86	0.88	37
Accuracy			0.87	71
Macro Avg	0.87	0.87	0.87	71
Weighted Avg	0.87	0.87	0.87	71

Notably, the model with no cross validation and a selected K value of 7 produced a higher test accuracy of **88.73%** when compared to our cross validated model of **87.32%**. There are several reasons this may have occurred. However, data sparsity and distribution likely impacted the model. Given the total number of school shooting occurrences is a relatively small number for KNN (354 instances), the dataset's size itself could be a significant factor. When the dataset is split into training and test sets, especially without cross-validation, there's a risk that unique examples can end up exclusively in the training set or the test set. This may have lead to the model evaluating data unrepresentative to the entirety of our dataset, which can skew accuracy results.

We can analyze KNN's performance by examining its associated confusion matrix values illustrated in Figure 8. Notably,

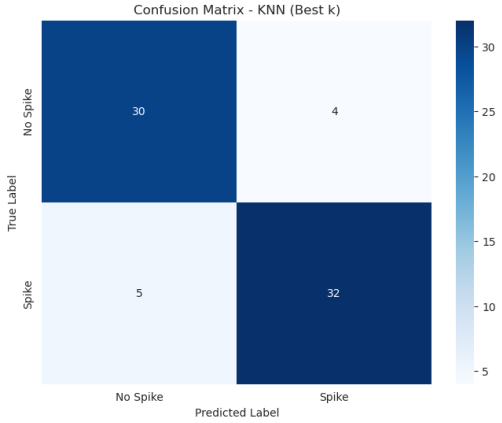

Fig. 8. Cross Validated Confusion Matrix K=5

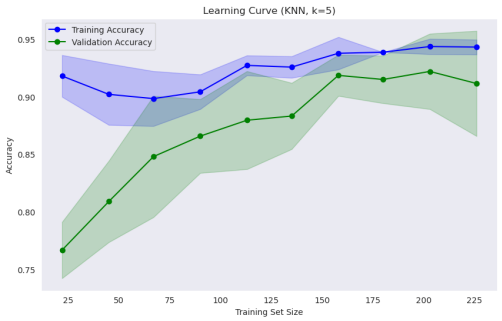

Fig. 9. KNN: Learning Curve

the model demonstrated a strong capability in identifying both true positives and true negatives with correctly identified instances of 32 and 30 respectively. Additionally, the relatively low numbers of false positives and false negatives indicate that the model is balanced; it neither excessively predicts spikes nor fails to predict actual spikes too frequently.

H. K-Nearest Neighbor: Learning Curve Analysis

Figure 9 displays the Learning curve the KNN model. Initially, training accuracy (blue) is slow to increases, but begins to make learning gains from increased sampling. However, as sample size increases above **150**, accuracy plateaus near **87%**, suggesting diminishing returns from adding more training data.

The validation accuracy (green) follows a sharp upward trend but with more variance. As more samples are added to the model, the validation curve narrows the gap, converging around **87%**. The convergence of training and validation accuracy suggests that the model is well-tuned and not significantly overfitting.

I. K-Nearest Neighbor: Validation Curve Analysis

The Validation curve shows a decline in both training and validation accuracy as the number of neighbors increases from **1** to **30**. The sharp drop in training accuracy from **k=1** to **k=4** suggests that using low k values leads to overfitting. This is apparent from the initial high training accuracy and lower

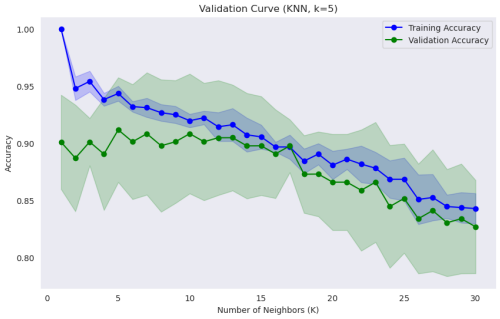

Fig. 10. KNN: Validation Curve

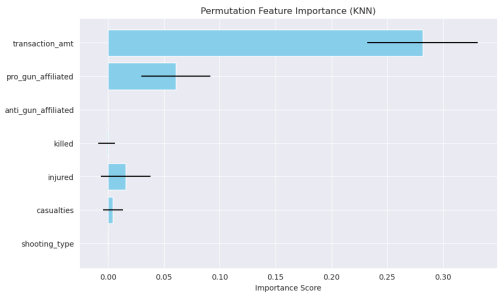

Fig. 11. KNN: Permutation Feature Importance

validation accuracy. As k increases, the model becomes more generalized. However, it is evident that by increasing k values degrade performance, indicating underfitting for larger values. As k increases, the model is unable to capture underlying patterns effectively.

J. Analyzing Results of Dataset 2

The permutation feature for our KNN model depicted in Figure 11 can provide insightful information regarding how various features can predict donation spikes to gun affiliated committees.

The model indicates that the transaction amount has the highest importance score in predicting a donation spike towards gun affiliated committees. Its high score suggests that larger or smaller transactions could be strongly associated with the likelihood of a donation spike following a school shooting, which is likely driven by the intensity of public response.

Whether the committee is pro-gun affiliated is the second most influential feature. This highlights the role of donations associated with pro-gun affiliations in predicting spikes. This likely indicates that donations to pro-gun groups are particularly responsive to school shootings. Interestingly, whether the committee is anti-gun affiliated is not as important when compared to pro-gun affiliations. This suggests that changes or patterns in donations to anti-gun groups may not be as predictive of spikes following school shootings.

CONCLUSION

This paper conducted a detailed analysis of two datasets through the lens of several Machine Learning algorithms. In Dataset 1, we examined the extent to which we could use key donor inputs—including geographic and demographic details along with donation specifics—to predict whether an individual would donate to either Republican or Democrat affiliated committees. We conducted our analysis and built predictive models utilizing Support Vector Machines, and Extreme Gradient Boosting Algorithms. The analysis confirmed our hypothesis by revealing our models are capable of discerning patterns in donation behavior with a high degree of accuracy.

In Dataset 2, we focused on the impact of school shootings on donation behaviors by examining whether school shootings lead to a spike in contributions to gun affiliated committees. Through the deployment of Neural Networks and K-Nearest Neighbors, this hypothesis was tested with various inputs. The results from these models provided significant insights into the society's reactions following school shootings, demonstrating that specific features such as the amount of donations and affiliations play important roles in predicting spikes in contributions.

Across both datasets, the application of multiple analytical approaches allowed for a comprehensive examination of factors influencing donation behaviors. The variance in model and feature importance across different algorithms highlighted the complexity of the predictive tasks and the necessity of choosing the right model based on specific characteristics of the data and the predictive goals.

Implications and Future Work: Findings from this study offers insights for political analysts and policymakers looking to predict public donations to political campaigns based on social factors. Future research can refine the models by adding or removing variables to improve predictive power. Validating the hypotheses through data analysis highlights the value of machine learning in understanding donor behavior across different contexts.

REFERENCES

[1] The Washington Post, "School shootings database," *GitHub Repository*, https://github.com/washingtonpost/data-school-shootings (accessed Sep. 19, 2024).
[2] D. A. Pisner and D. M. Schnyer, "Support vector machine," in *Machine Learning*, A. Mechelli and S. Vieira, Eds. Academic Press, 2020, pp. 101-121. doi: https://doi.org/10.1016/B978-0-12-815739-8.00006-7. [Online].
[3] "Parameter estimation using grid search with a support vector machine," *Scikit-learn Documentation*, https://scikit-learn.org/stable/auto_examples/svm/plot_rbf_parameters.html (accessed Sep. 19, 2024).
[4] IBM, "Support Vector Machine," *IBM Documentation*, https://www.ibm.com/topics/support-vector-machine (accessed Sep. 19, 2024).
[5] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997, p. 112.
[6] "Parameter estimation using grid search with a support vector machine," *Scikit-learn Documentation*, https://scikit-learn.org/stable/auto_examples/svm/plot_svm_scale_c.html (accessed Sep. 19, 2024).
[7] "Precision, recall, F-score support in classification," *Scikit-learn Documentation*, https://scikit-learn.org/stable/modules/generated/sklearn.metrics.precision_recall_fscore_support.html (accessed Sep. 19, 2024).
[8] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997, pp. 231-236.