Applying Machine Learning to Election Forensics Research: A Case of Russia

Kirill Kalinin

Hoover Institution, Stanford University

Contact Information:

Email: kirill.kalinin@gmail.com

Abstract

Election fraud detection is paramount for upholding the integrity of democratic processes. Traditional approaches to election forensics research involve thorough examination of electoral procedures and extensive statistical analysis of voting data. Statistical analyses utilize various techniques, such as digit tests, trend analyses, and statistical modeling, to detect anomalies in voting patterns and uncover suspicious trends in voter turnout or vote shares. This study aims to apply supervised machine learning algorithms to predict election fraud reported by observers using various election forensics measures based on data from the 2018 Russian presidential elections.

Data

The dataset contains a small sample of election fraud data from Russian presidential election 2018 collected by the activists of the "For Fair Elections" movement(2). The independent observers were able to download footage from 8,000 polling stations out of 46,000 polling stations with installed webcams. While watching a video recording from selected precinct, volunteers recorded their answers to 11 questions describing the execution of procedures required by the law. A total of 271 reports/observations are available to the author.

Target variable: Precinct-level magnitude of election fraud related to turnout, as measured by election observers.

Precinct-level features:

- **electoral**: turnout, incumbent's vote share;
- **geographic**: republics/territories, urban/rural;
- **election forensics**:
  - *precinct-level digit tests*: last digits and second digits in vote counts and turnout, 0s and 5s in turnout and incumbent's vote percentages (`Valid_last`, `Votes_last`, `Valid_last05`, `Votes_last05`, `Valid_second`, `Votes_second`);
  - *precinct-level nonparametric measures*: `Nonp_Shpilkin_raw`, `Nonp_EM_pre_raw`, `Nonp_EM_hist_raw`, `clean.votes.M2`, `fraud.votes.M2`, `clean.votes.M5`, `fraud.votes.M5` based on Kalinin's computations (3);
  - *precinct-level parametric measures, the Bayesian finite mixture model*: `tfraud`, `Ntfraud`, `pfraud`, `Nfraud` based on Mebane's computations (5).

Supervised Learning Algorithms

In this study I will focus on five supervised machine learning algorithms: *Decision Trees*, *Neural Networks*, *Boosted Trees*, *Support Vector Machines*, and *k-Nearest Neighbors*.

ML Challenges

Conducting machine learning experiments in election forensics research poses significant challenges.

- The complexity of factors involved in election fraud and the limited size of the dataset present obstacles to model development.
- Imbalanced classification poses a challenge, as the minority class (election fraud) has too few examples for the model to accurately learn the decision boundary.
- Overfitting issues arise due to the limited number of election observation cases.

Pre-Processing and Performance Evaluation

All the features are normalized using the *preprocessing.normalize* function from the *scikit-learn library*. The normalized features are then converted to a NumPy array. Subsequently, the datasets were divided into training and testing samples, with 80% allocated for training and 20% for testing.

Each classifier utilized in this study undergoes evaluation using the Stratified K-Fold cross-validation that allows to systematically and reliably assess the performance across varying training sample sizes. The function iterates over a range of training sample sizes. For each iteration, it conducts *Stratified K-Fold* cross-validation with 5 folds. This approach ensures that each fold maintains the same class distribution as the original dataset. However, the imbalanced nature of the dataset can lead to overfitting issues. To mitigate this, *Random Over-Sampling* of the training set can be also employed to address class imbalance.

The learning curve uses *F1 Macro Score* calculated by taking the average of the *F1 scores* for each class in a multi-class classification problem. The macro average gives equal weight to each class, regardless of class imbalance. The *F1 Macro Score* is computed based on the model's predictions on a separate test set or validation set, therefore it demonstrates how well the model generalizes to unseen data.

In addition, each classifier is analyzed with a validation curve, a standard tool in machine learning model assessment. The plotting function leverages parameters denoting training scores, validation scores, and corresponding indices to graphically represent the performance of machine learning models across varying parameter values. This visualization facilitates the detection of potential overfitting or underfitting issues in the classifiers.

After conducting manual performance evaluation, I apply the *GridSearchCV* algorithm, which conducts cross-validation on the training data using various combinations of hyperparameters specified in the grid. It evaluates the performance of each combination using the specified scoring metric (*F1_weighted* in my case). After evaluating all combinations during cross-validation, *GridSearchCV* selects the model with the highest average score as the best estimator. This proposed model with the optimal hyperparameters is then run separately. Based on it, I test and evaluate the performance of each algorithm for each class and assess how quickly they perform in terms of wall clock time.

Analysis

Decision Trees

Decision Trees are a versatile supervised learning algorithm that create a model based on a series of binary decisions, representing data features as nodes and outcomes as leaves. The Figure suggests that effective learning requires a minimum of 200 samples, though improvements in the validation set may decrease training set performance. *GridSearch* recommended cost complexity parameter ($ccp\_alpha = 0.064$) and the *entropy* criterion for optimal decision tree performance.

Neural Networks

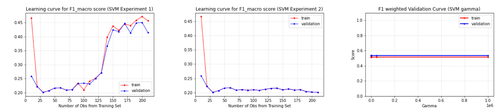
Neural Networks consist of interconnected nodes or neurons arranged in layers that process input data through weighted connections, enabling them to learn complex patterns and relationships. For Neural Networks, *GridSearch* identified the optimal parameters as $alpha = 0.001$, $hidden\_layer\_sizes = (10, 20)$, and $learning\_rate\_init = 0.01$, achieving a best cross-validation score of 0.70.

Boosting

Boosted Trees improve the performance of decision trees by sequentially training models, each correcting the errors of its predecessor. For the Boosting classifier, I utilized the *GradientBoostingClassifier* from the *sklearn* library. The *GradientBoostingClassifier* generally demonstrates improved performance in mitigating overfitting compared to the *AdaBoostClassifier*. It constructs trees sequentially, with each new tree rectifying errors from previously trained ones, whereas *AdaBoost* assigns weights to data points and concentrates on misclassified points in subsequent iterations. *GridSearch* revealed that the best model, with a validation score of 0.818, was obtained with $\alpha = 0.01$ and $n\_estimators = 140$.

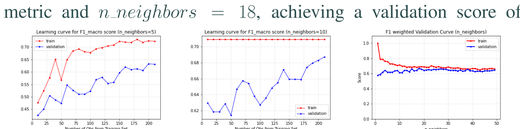
Support Vector Machines

Support Vector Machines are powerful classifiers that find the optimal hyperplane that separates data into distinct classes with the maximum margin, making them effective in high-dimensional spaces and for handling non-linearly separable data using kernel functions. Using *GridSearch* to optimize the Support Vector Machine model, the best performance was achieved with a *linear* kernel, $C = 50$, $degree = 1$, and $gamma = 0.005$, resulting in a validation score of 0.68. This indicates a moderate performance on the validation set, suggesting room for further optimization or alternative approaches.



k-Nearest Neighbors

k-Nearest Neighbors is a simple, instance-based learning algorithm that classifies data points based on the majority class among their k closest neighbors in the feature space. *GridSearch* determined that the optimal parameters for the k-Nearest Neighbors algorithm were the *euclidean* metric and $n\_neighbors = 18$, achieving a validation score of 0.65.



Comparison of Learners

F1-score

The F1-score is a metric commonly used in classification tasks that provides a balance between precision and recall. Intuitively, it can be understood as a measure of a model's accuracy in correctly identifying both positive and negative instances in a dataset.

Table 1 presents F1-scores for different machine learning classifiers across three fraud categories: High, Low, and Medium.

Table 1. F1-scores

	DT	NN	BG	KNN	SVM
High	0.711	0.850	0.789	0.765	0.811
Low	0.723	0.711	0.723	0.744	0.756
Medium	0.333	0.400	0.480	0.485	0.500
acc.	0.655	0.691	0.691	0.673	0.709
mac.avg	0.589	0.654	0.664	0.665	0.689
w.avg	0.627	0.683	0.687	0.690	0.713

It shows accuracy, macro-average, and weighted-average scores for Decision Tree, Neural Network, Boosting Gradient, SVM, and KNN classifiers. The highest F1-score across all metrics and classifiers is achieved by the SVM model, indicating its superior performance on the dataset.

Wall Clock Time

This measure is crucial for evaluating algorithms as it directly reflects the computational resources required for training and inference, impacting the efficiency and scalability of the model.

Table 2. Wall Clock Time (in *seconds*)

Model	Time
Decision Tree	0.01
Neural Network	0.34
Boosting Gradient	0.13
SVM	0.01
KNN	0.01

Table presents the wall clock time (in seconds) for various models. The Decision Tree, SVM and KNN models exhibit consistently low training times. However, the Neural Network and Boosting Gradient models require significantly more time, with the Boosting Gradient model being particularly time-consuming.

Conclusion

The development of ML models capable of accurately predicting election fraud can enhance the efficiency and effectiveness of election monitoring efforts. This study provided a comparison of five supervised machine learning algorithms: *Decision Trees*, *Neural Networks*, *Boosted Trees*, *Support Vector Machines*, and *k-Nearest Neighbors*, with *Support Vector Machines* providing the best performance on our data.

Further Research

I plan to further this study by including more precinct-level data and focusing on the data from *Karta Narushenii* website for all monitored elections since 2011 (4). Since imbalanced classification is a significant issue for this analysis, I intend to utilize class-balanced loss, which assigns sample weight inversely proportional to the class frequency (1).

The paper version of this poster can be accessed online.

References

[1] Y. Cui, M. Jia, T. Lin, Y. Song, and S. J. Belongie. Class-balanced loss based on effective number of samples. *CoRR*, abs/1901.05555, 2019.

[2] A. Gabdulvaleev. Voter turnout was inflated by almost 3 million people in 10 regions during the 2018 presidential election (pochti na 3 mln chelovek zavyisili yavku v 10 regionakh na vyborakh prezidenta 2018). 2018.

[3] K. Kalinin. An Empirical Comparison of Parametric and Nonparametric Methods Applied to the Measurement of Election Fraud. 2022.

[4] K. Narushenii. Karta narushenii. 2024.

[5] J. Walter R. Mebane, D. Ferrari, K. McAlister, and P. Y. Wu. Measuring election frauds. 2022.