Training a Logistic Regressor to Predict Political Party Affiliation

Vahan Bznuni

Study.com

Abstract

Political identity prediction is arguably a textbook example of the application of Machine Learning (ML) to achieve real-time inference. This report describes an educational proof-of-concept command-line agent that predicts a respondent's U.S. political party affiliation based on a 12-question survey. Survey questions were carefully crafted based on statistical data on divisions among U.S. voters on political issues. Feature engineering was achieved via domain-aware ordinal encoding of the responses and Z-scaling, while experimental linear weighting to was explored to boost signal. A multinomial logistic regression model was fitted on a realistic synthetic LRM-generated dataset (n=1,000), following a stratified 80-20 split. Hyperparameter tuning via grid search was performed to achieve optimal model performance. A 5-fold stratified cross-validation was performed on the 80% training portion, followed by a final evaluation using the 20% held-out set to obtain performance metrics. The ~79% accuracy achieved demonstrates how lightweight, interpretable models can be utilized to achieve statistically significant classification performance from limited implementation complexity, across a variety of domains and applications.

Methods

Survey Design and Data Collection

A 12-question survey was designed to capture ideological leanings across U.S. voters anchored on the political landscape circa 2024. Questions were crafted to maximize information signal from responses, based on statistical data on polarization across political issues. Recent polling was used to identify partisan gaps on core policy concerns. Table 1 summarizes nine issues that were found where the difference between Democrats and Republicans exceeds ~ 30 percentage points (Brennan, 2024; Pew Research Center, 2024). These topics formed the backbone of the 12-question instrument (see Appendix A).

Table 1

Percentage of respondents who rated issues as very problematic or extremely important

Issue	Democrats / lean Democrats (%)	Republicans / lean Republicans (%)	Δ (gap, %)
Inflation	46%	80%	44%
The federal budget deficit	35%	71%	36%
Illegal immigration	27%	78%	51%
Gun violence	68%	27%	41%
The state of moral values	32%	61%	29%
Climate change	58%	12%	46%
The economy*	36%	66%	30%
Terrorism / National Security*	31%	60%	29%
Crime*	20%	52%	32%

Note. Percentages represent the share of respondents, by party affiliation, who rated the issue as either a “very big problem” or “extremely important” (marked by *), as reported by Brennan (2024) and the Pew Research Center (2024). Δ = absolute percentage-point gap between partisan groups.

Data Management

Plain-text CSV files were chosen as the data storage format due to their simplicity, scalability, human readability, and broad compatibility across programming environments. Each row in the dataset corresponds to one survey respondent, with encoded answers in fixed column positions and the final political affiliation label as the terminal column. The seed file contains 1 000 synthetically generated rows produced by an OpenAI o3 LRM (2025) instructed to approximate current US party proportions. Additional rows are appended as new surveys are completed, and a Java *DataStore* class handles read/write operations and rolling backups.

Model Choice & Agent Architecture

Three classifiers were considered for this project: Naïve Bayes, Logistic Regression, and Decision Trees. Logistic Regression was chosen due to its good balance of flexibility and complexity, ease of implementation, abundance of documentation, known good performance, and interpretability. Naïve Bayes was rejected in favor of Logistic Regression due to Naïve Bayes’ strong assumption of feature independence (which could not be reliably assumed in this context), while Decision Trees were rejected due to increased risk of overfitting and added implementation complexity without a well-defined justification.

A multinomial logistic regression model was built from scratch using pure Java, utilizing the batch Gradient Descent algorithm on Cross-Entropy Loss function, in a scalable, object-oriented program architecture. The resulting classifier was wrapped in a simple CLI “agent” loop (collect → preprocess → predict → retrain) that is initially trained on the seed data and then continuously ingests user answers,

makes a prediction, receives the true label, retrains online, and reports updated accuracy, recall, precision, and F_1 back to the user until they choose to exit.

Feature Engineering: Encoding, Scaling, and Weighting

To improve model accuracy, three successive transforms are applied to the raw survey response values before being passed to training.

An ordinal encoding scheme maps each input value to an integer that rises with the *intensity* of the stated sentiment (see Appendix B). Question #1 through #3 measure 3rd party signal, while questions #4 through #12 measure lean between the major parties.

After encoding, the twelve encoded columns are rescaled into a standard normal distribution (also known as standardization/z-scoring) to improve stability.

Exploratory manual search was explored for weighting options on the major party lean questions (#4-#12), as documented in the results section, while the live program is set to operate without weighting to achieve the best measured performance.

Training, Hyper-parameter Tuning, and Evaluation

On initial startup, the model trains on the seeded artificial data to obtain baseline performance (and report metrics). Initially, data is split into a train set (80%) and a test set (20%) using stratification. Then, the 80% train set is further split into validation-train (64%) and validation-test (16%) sets. A range of possible hyperparameter values was selected around default values specified by GeeksforGeeks (2024) and scikit-learn developers (2025). On each initial training run, the program performs an automated grid search across the hyperparameter space and selects the best ones based on 5-fold Cross-Validation. A new model is re-trained online after each live user input, using the previously discovered hyperparameters.

Results and Limitations

Performance of the model trained and tuned on the encoded/scaled/weighted bootstrap set is summarized in the following table.

Table 2

Performance metrics of LR model trained on LLM-generated dataset (n=1,000) using scaling/weighting

Accuracy	71.29%
Recall	71.29%
Precision	69.89%
F1 Score	67.58%

Note. For weighting, 3 was raised to the power of the class value, for features 4 through 12.

The next table shows performance metrics of a model trained on data that was only encoded but not scaled/weighted.

Table 3

Performance metrics of LR model trained on LLM-generated dataset (n=1,000) without scaling/weighting

Accuracy	74.75%
Recall	74.75%
Precision	Undefined / None
F1 Score	Undefined / None

Finally, the best performance was obtained by a model trained on data that was encoded and scaled, but not weighted, as shown in the following table.

Table 4

Performance metrics of LR model trained on LLM-generated dataset (n=1,000) with scaling only

Accuracy	75.74%
Recall	75.74%

Precision	74.49%
F1 Score	72.33%

A manual search revelated that weighing either did not affect the metrics positively, or worsened performance to varying degree. Notably, scaling and weighting with the same pipeline but using a 3rd party regressor from The Statistical Machine Intelligence and Learning Engine (Smile) library (Li, 2024) did produce a positive difference with large linear weights for the same features (~k=1,000). However, given the significant variance, it is more likely that these differences are due to statistical noise and/or distortions, rather than any statistically significant signal obtained by weighting.

The major limitations of this process were (a) the synthetic nature of the data, (b) the unverified nature of the LLM-produced synthetic data, (c) the relatively small scale of the data as compared to modern ML and data science standards, (d) the linear nature of Logistic Regression and the resulting inherent limitation to learn non-linear patterns and very complex feature interactions, (e) the limited scope of model selection (i.e. a single model), (f) the performance limitations of manual implementation as compared with production library code for the ML core, and (g) the limited scope and complexity of hyper-parameter tuning, weighting parameter search, as well as other stages in the ML pipeline, as compared with modern and sophisticated approaches.

Conclusion

This project served as a valuable opportunity for educational exploration of core components of the ML process, as applied to a textbook classification task. The obtained results show both the strong potential of limited-complexity ML to yield meaningful results, as well as the limitations of basic techniques, motivating the justification of more sophisticated approaches and paving the way for their future exploration.

References

Brenan, M. (2024, October 9). Economy most important issue to 2024 presidential vote. Gallup. https://news.gallup.com/poll/651719/economy-important-issue-2024-presidential-vote.aspx

GeeksforGeeks. (2024, May 22). How to optimize logistic regression performance. https://www.geeksforgeeks.org/machine-learning/how-to-optimize-logistic-regression-performance/

Li, H. (2024). Smile (Version 4.4.0) [Computer software]. https://haifengl.github.io

OpenAI. (2025). OpenAI o3 [Large language model]. OpenAI. https://www.openai.com

Pew Research Center. (2024, May 23). Public's positive economic ratings slip; inflation still widely viewed as major problem (Report). Pew Research Center. https://www.pewresearch.org/wp-content/uploads/sites/20/2024/05/PP_2024.5.23_economy-national-problems_REPORT.pdf

scikit-learn developers. (2025). LogisticRegression (Version 1.7.0) [Computer software documentation]. In scikit-learn. Retrieved July 1, 2025, from https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

Appendix A

Survey Questionnaire

For the purpose of this survey, if your response would have been different in the year 2024 than it is now due to a change in policy or events other than any change in your personal views, please provide the response as it would have been in 2024.

1. In recent elections, how often have you voted for candidates from both major parties on the same ballot (for different offices)?
    a. Often.
    b. Sometimes.
    c. Rarely.
    d. Never.
2. How well do either of the two major U.S. political parties represent your views overall?
    a. Very well.
    b. Somewhat.
    c. Not very well.
    d. Not at all.
3. Should more than two major parties have seats in Congress?
    a. No - two major parties are enough.
    b. Yes - minor/third parties should also have seats.
4. How big of a problem is Inflation?
    a. Major problem.
    b. Somewhat of a problem.
    c. Not much of a problem.
5. How big of a problem is the federal budget deficit?
    a. Major problem.
    b. Somewhat of a problem.
    c. Not much of a problem.
6. How big of a problem is Illegal immigration?
    a. Major problem.
    b. Somewhat of a problem.
    c. Not much of a problem.
7. How big of a problem is gun violence?
    a. Major problem.
    b. Somewhat of a problem.
    c. Not much of a problem.
8. How big of a problem is the current state of moral values?
    a. Major problem.
    b. Somewhat of a problem.
    c. Not much of a problem.
9. How big of a problem is climate change?
    a. Major problem.
    b. Somewhat of a problem.
    c. Not much of a problem.
10. How important of an issue is terrorism / national security?
    a. Extremely important.

   b. Somewhat important.
   c. Not very important.
11. How important of an issue is the economy?
   a. Extremely important.
   b. Somewhat important.
   c. Not very important.
12. How important of an issue is crime?
   a. Extremely important.
   b. Somewhat important.
   c. Not very important.
13. Please select party affiliation that best describes you:
   a. Democratic / lean Democratic.
   b. Republican / lean Republican.
   c. Independent.
   d. Other / 3rd Party (ex. Green, Libertarian, etc.)

Appendix B

Encoding

Table B1

0-Indexed Ordinal Encodings

Question #	Question (short)	Response range (left -> right)	Encoding	Order
1	Ticket split	(Often, Sometimes, Rarely, Never)	[3, 2, 1, 0]	Reversed
2	Party fit	(Well, Somewhat, Not well, Not at all)	[0, 1, 2, 3]	Forward
3	More parties	(No, Yes)	[0, 1]	Forward
4	How much of an issue	(Major, Somewhat, Not much)	[2, 1, 0]	Reversed
5	How much of an issue	(Major, Somewhat, Not much)	[2, 1, 0]	Reversed
6	How much of an issue	(Major, Somewhat, Not much)	[2, 1, 0]	Reversed
7	How much of an issue	(Major, Somewhat, Not much)	[2, 1, 0]	Reversed
8	How much of an issue	(Major, Somewhat, Not much)	[2, 1, 0]	Reversed
9	How much of an issue	(Major, Somewhat, Not much)	[2, 1, 0]	Reversed
10	How important	(Extremely, Somewhat, Not very)	[2, 1, 0]	Reversed
11	How important	(Extremely, Somewhat, Not very)	[2, 1, 0]	Reversed
12	How important	(Extremely, Somewhat, Not very)	[2, 1, 0]	Reversed
13	Affiliation	(Dem, Rep, Independent, Other)	[0, 1, 2, 3]	Forward

Note. “Forward” = integer 0 assigned to the first (left-most) verbal option.

“Reversed” = integer 0 assigned to the last (right-most) verbal option.