Predicting Legislative Bill Passage in Wisconsin: A Machine Learning Framework for Policy Prioritization

Ryan Wang Karen Gonzalez Luke Nalewajk
Carnegie Mellon University
Heinz College of Information Systems and Public Policy
Machine Learning for Public Policy (MLPP) – Fall 2025

Executive Summary

The American Civic Protection Association (ACPA) is a national civil rights organization that monitors legislation across the United States and intervenes when proposed bills threaten individual freedoms. Each year, tens of thousands of bills are introduced at federal and state levels, many of which have the potential to reshape access to healthcare, education, voting rights, and other fundamental protections. Because ACPA operates with limited legal and analytical resources, it cannot review every bill in detail. As a result, the organization faces a persistent risk: devoting effort to bills that are unlikely to advance while failing to identify those that pose an immediate and credible threat to civil liberties.

This project addresses that challenge by developing a predictive system designed to help ACPA focus its attention on the legislation that is most likely to pass. We use Wisconsin as a proof-of-concept environment for model development. Wisconsin is an especially informative test case because its legislature is competitive, its margins are narrow, and its procedural rules offer clean, well structured data. In such a setting, even modest shifts in advocacy or public engagement can meaningfully alter policy outcomes. A well calibrated prediction model can therefore provide significant practical value.

Our goal is to identify, each month, the top 15 percent of active bills that are most likely to pass during a legislative session. This threshold reflects ACPA's operational need for a manageable and actionable list of high priority bills. To achieve this, we construct a modeling pipeline using features derived from bill metadata, sponsor information, procedural events, vote histories, and temporal activity patterns. Multiple model families are evaluated, including decision trees, random forests, boosted models, and scaled logistic regression. Among them, scaled logistic regression demonstrated the strongest performance and interpretability, making it suitable for integration into ACPA's decision workflows.

The results reveal both promise and complexity. The model is able to distinguish high risk bills with strong precision in the top 15 percent of predictions. At the same time, the analysis highlights structural limitations in the underlying legislative data. Missingness is pervasive, particularly for amendments and certain vote features, and procedural activity is unevenly distributed across sponsor groups. These factors affect not only predictive accuracy but also fairness. Our sponsor party fairness audit shows that the model performs differently across political groups, largely due to disparities in how legislative momentum is recorded.

Taken together, these findings suggest that predictive modeling can substantially improve ACPA's ability to allocate resources effectively, but model outputs must be interpreted with care. A field trial is recommended to evaluate how the system performs when embedded into real organizational workflows. The trial should measure timeliness, accuracy, and equity in staff decision making, ensuring that predictions enhance rather than distort advocacy priorities.

Looking ahead, several directions offer opportunities for refinement. These include expanding fairness analyses to additional attributes, incorporating text-based features from full bill language, improving coverage of procedural events, and designing chamber specific models. Strengthening the data pipeline and addressing fairness concerns will be essential for deploying the system at scale.

In sum, this project demonstrates the feasibility and value of a predictive approach to legislative monitoring. When coupled with careful evaluation and responsible use, such a system can help ACPA intervene earlier, allocate its resources more effectively, and better protect the civil liberties of communities throughout the United States.

I. Background and Introduction

A. Problem Motivation

The American Civic Protection Association (ACPA) is a national organization dedicated to protecting the rights of all citizens through advocacy efforts against legislation that infringes on civil liberties. Their work includes monitoring newly introduced bills, challenging harmful laws in court when necessary, and ensuring that protective statutes are put in place when deemed necessary. Because ACPA operates nationally, it must track tens of thousands bills introduced across state and federal legislatures. Many of these bills have the potential to negatively affect vulnerable or marginalized populations, yet only a fraction can realistically receive focused attention due to the organization's limited legal, analytical, and advocacy resources.

B. Why This Problem Matters

The scale of legislative activity makes it unrealistic for ACPA to review every bill that may threaten civil liberties. As a result, the organization risks allocating effort to bills that ultimately fail while overlooking bills that pose a serious and credible threat of passing. The consequences of such oversight can be significant, particularly for historically disadvantaged populations. Laws that restricted education, reproductive autonomy, voting rights, or equal treatment have repeatedly harmed people and communities. These past examples illustrate how failing to intervene early in the legislative process can allow harmful policies to advance with lasting impact.

To address this challenge, ACPA seeks a systematic way to identify which bills are most likely to pass so it can concentrate its resources where they matter most. A reliable prediction model would allow the organization to prioritize high risk legislation, improve the timing and coordination of advocacy responses, and better protect communities whose civil liberties are most at risk. More broadly, such a system would enhance both the efficiency and effectiveness of ACPA's efforts, enabling the organization to use its finite resources in a more targeted and equitable manner. As a proof of concept, this project applies the proposed approach to the Wisconsin Legislature. We will create and evaluate a machine learning model capable of identifying which bills are most likely to pass, allowing us to assess the feasibility and policy relevance of deploying such a system in practice as well as make a recommendation on the best approach going forward for the ACPA.

C. Policy Context and Impact

The main policy goal of this project is to prioritize effectiveness as the focus will be to accurately predict the number of true "likely-to-pass" bills in the top 15% and flag them for ACPA. However, equity is also a goal since we want to ensure that the model does not ignore bills that disproportionately affect minority or vulnerable groups. Additionally, efficiency is another factor that is considered since we think it's important to provide timely predictions so staff have actionable insights, and not just retrospective analysis.

Once a bill is virtually guaranteed to pass, advocacy is no longer effective, which creates the same challenge as working on a bill with very low chances. We must therefore consider this balance when interpreting the top 15 percent. Bills that fall in the 45 to 65 percent range are genuine tossups and warrant additional attention. They offer more opportunities for effective intervention than bills that are nearly certain to pass or fail.

Some trade-offs are expected to occur, as a highly complex model may be more accurate but also less interpretable. For ACPA's needs, explainability and trust are important, so we aim to balance performance with transparency. We want to ensure that the manner and factors that the model uses to predict whether the bill will pass or not impacts our equity and efficiency goals. However, the main priority of the project will be the effectiveness of the model in aiding ACPA in their mission.

D. Justification Behind Wisconsin

The primary reason for picking Wisconsin is its unique position as one of the few battleground states in the union. While it leans Republican, the Senate stands at 18–15 and the House at 54–45. This opens the state up to having a wide variety of Bills being able to pass quite easily from both sides with just a few swing votes to reach a majority. We theorize that interventions using the ACPA's resources would be very effective, as advocacy efforts or legal challenges could easily have the effectiveness to swing the few votes required to prevent laws that threaten civil liberties from passing. This would make predictions from our model more immediately impactful than in states with lopsided majorities.

Wisconsin's legislative process is also straightforward: bills move from the House to the Senate, and if approved, to the governor for signature or veto. Wisconsin also features a helpful reintroduction rule, which says bills that fail or expire must be refiled under a new bill_id, giving us clean, trackable data across sessions.

II. Related Work

A. ML Approaches for Legislative Forecasting

Prior machine learning research includes bill level prediction models that rely on textual features or procedural histories. Some studies predict bill passage by analyzing language patterns in bill titles or text, identifying keywords that tend to correlate with legislative success [1]. Others model individual legislator votes using ideology, party or affiliation, then aggregate these predictions into a passage probability for each bill [2]. GovTrack is a widely used example of this approach, publishing probabilistic forecasts for congressional bills using logistic regression [3].

B. Traditional / Non-ML Approaches

Beyond machine learning, policy analysts typically rely on heuristics rooted in procedural signals. For example, early committee endorsements, the identity or influence of sponsors, the presence of bipartisan support, or parallels to previously passed legislation. These approaches offer interpretability and familiarity, though they struggle to scale with increasing legislative volume. Manual review is costly and slow, and often fails to identify systematic patterns buried in large or complex datasets.

C. How Our Work Differs

Our project differs in both formulation and purpose. Instead of producing a single probability for each bill at the end of a session, we designed a system that updates monthly during the active voting period. Our goal is operational, not merely predictive. The model is designed to support real time triage by identifying the top 15 percent of bills most likely to pass, allowing ACPA to concentrate on those that may pose the greatest risk. We also integrate event level procedural data, sponsor information, and temporal features, creating a richer and more dynamic representation of legislative activity than many text-only or vote-only approaches.

III. Problem Formulation and Solution Overview

A. Analytical Formulation

We define the problem as follows:

> *Every month for the open voting duration of the session (1.5 years), for all the active (any other status than passed) bills that are introduced in the current legislative session in Wisconsin that are related to Human Rights, can we identify the top 15% of bills that are most likely to pass to better allocate ACPA resources to assist in protecting civil liberties?*

This formulation reflects several structural considerations. Wisconsin operates on a two-year session cycle beginning in odd years. Most legislative activity occurs in the first year, with additional voting through early spring of the second year. A 1.5 year prediction window captures this cycle while allowing ACPA to intervene before bills reach irreversible procedural stages. Monthly updates balance responsiveness with data stability.

B. Operational Goals (Top 15%)

The ACPA requested that the system focus on a manageable subset of high risk bills in order to match organizational capacity and staffing constraints. The top 15 percent threshold was therefore chosen to produce a list that is both operationally feasible and analytically meaningful. This threshold creates a clear priority target while preserving enough variation between bills to allow effective ranking.

Predictions must balance accuracy with timeliness. Monthly predictions enable staff to review bills while they are still moving through the legislative process rather than after they have already advanced too far. In practice, identifying highly likely bills is necessary but not sufficient. Some bills will have such high predicted probabilities that intervention may no longer be effective, and others will fall into very low ranges where resources are unlikely to change the outcome. For this reason, ACPA is especially interested in bills that fall into the intermediate probability range, generally between 45 percent and 65 percent. These cases are important because targeted advocacy could realistically alter their legislative trajectory.

C. High-Level Description of Our Approach

Our approach uses a temporal modeling framework that mirrors how legislation progresses through a Wisconsin session. We construct monthly as-of dates that allow the model to incorporate newly recorded procedural activity and generate predictions in a manner consistent with how ACPA would monitor bills in real time. These time based splits ensure that training and evaluation follow the natural flow of legislative information rather than mixing signals across periods.

The feature set draws from several sources of legislative information. Procedural event histories capture how bills move through committees, chambers, and floor processes. Sponsor metadata reflects patterns of political support, while additional features incorporate chamber of origin, bill type, text size, and available voting records.

Together, these components describe both momentum and structural characteristics of each bill.

We explore a range of model families that are suitable for ranking tasks within a policy environment. These include logistic regression, random forests, decision trees, and gradient-boosted models. Each model is evaluated within the temporal framework to understand how it handles legislative data, how it responds to procedural updates, and how easily it can be integrated into an ongoing monitoring workflow.

The system is designed for recurring deployment. Each month, the pipeline updates its feature matrix with new events, votes, and sponsor actions, then produces an updated ranking of active bills. This structure allows ACPA to continuously track changes in bill activity and maintain an up-to-date view of which proposals may require closer attention. The workflow emphasizes clarity and consistency so that predictions can be incorporated into regular advocacy planning without substantial overhead.

IV. Data Description and Exploration

A. Data Sources and Time Span

Our primary data source is LegiScan, a platform that aggregates legislative records from all U.S. states and territories. The dataset spans roughly ten years and includes bill metadata, full text, sponsors, events, votes, amendments, and session timelines. Although the data is continuously updated, historical completeness varies, especially for procedural features.

B. Cohort Definition

The cohort includes all active Wisconsin bills within a legislative session. A bill is considered active if it holds any status other than passed. Bills that fail, die in committee, or expire at the end of a session must be reintroduced under a new bill identifier in future sessions. This reintroduction rule creates clean temporal boundaries and reduces ambiguity when assigning labels, since each identifier corresponds to a single, session specific legislative trajectory.

C. Label Definition

The prediction target is whether a bill passes within the session. Labels are defined as binary indicators based on the final status recorded by the end of the session. Although labels reflect an outcome measured at the session level, predictions are generated monthly so that the model operates within the evolving legislative process.

D. Exploratory Data Analysis

Exploratory analysis of the LegiScan dataset highlights several structural patterns in how legislation moves through the Wisconsin Legislature. First, legislative activity is highly uneven across bills. Some bills accumulate long procedural histories with numerous committee actions, hearings, and floor events, while others remain relatively inactive and receive only a small number of recorded events before the session ends. This variation suggests that legislative momentum is not uniform and that bills follow distinct procedural pathways.

Second, bill activity follows clear temporal cycles. Early months in each session typically show a surge in bill introductions, followed by increases in committee referrals and early actions. Activity rises again during key legislative periods when voting deadlines approach. These temporal patterns informed the choice to use monthly as-of-dates and motivated the need for time aware prediction windows.

Third, descriptive statistics reveal substantial differences across bill categories. Certain bill types, such as appropriations or administrative measures, tend to progress further in the legislative process, while others stall early. Similarly, sponsor characteristics show identifiable patterns, including differences in introduction volume across parties and variation in typical bill trajectories. These patterns suggest that both bill type and sponsor metadata may contain predictive signal.

Finally, exploratory visualization shows that passage remains a relatively rare outcome. Most bills do not become law, and the base rate varies across legislative sessions. This imbalance affects how precision at top k should be interpreted and reinforces the importance of a ranking based evaluation framework.

Overall, the EDA indicates that legislative data is shaped by temporal cycles, heterogeneous bill trajectories, and structural differences across bill types and sponsors. These observations guided our modeling choices by highlighting which features reflect meaningful legislative behavior and which aspects of the process vary systematically over time.

E. Important Observations Informing Model Choices

Exploratory analysis of the LegiScan dataset highlights several structural patterns that shaped our modeling decisions. First, legislative activity follows clear temporal cycles. Bills tend to accumulate actions in bursts that correspond to committee periods, floor calendars, and session deadlines. This pattern motivated the use of monthly as-of dates and time aware validation.

Second, bills display substantial variation in their procedural trajectories. Some accumulate long sequences of actions, while others remain relatively inactive. These differences suggested that event histories and sponsor characteristics would be meaningful components of the feature set.

Third, descriptive statistics show consistent variation across bill types and sponsor groups. Certain categories of bills move through the legislature at different rates or follow distinct pathways. This motivated the inclusion of metadata features such as chamber of introduction, sponsor party, and bill type.

Overall, these observations indicated that a time structured modeling framework and a feature set grounded in legislative behavior would be appropriate for predicting bill passage within a session.

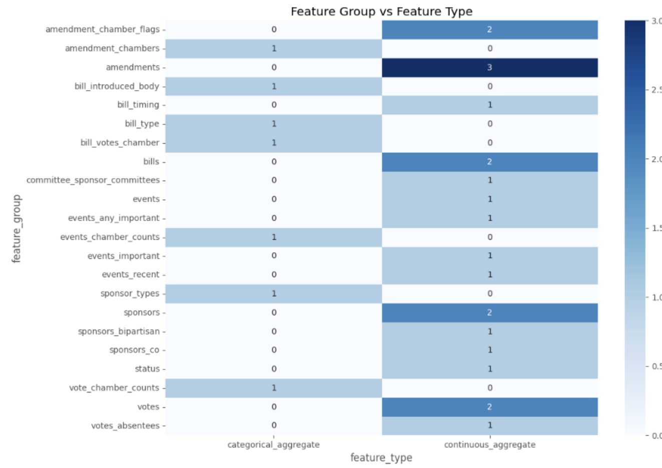

Fig. 1: Feature Group vs Feature Type

V. Modeling Approach

The code for the features used, models ran, and evaluations metrics can be found in the GitHub repository that was used throughtout the course of this project: Bills 3 GitHub Repo

A. Features Used

The feature set combines information from procedural events, sponsor characteristics, bill metadata, vote summaries, and amendment records. The pipeline generates 163 features across 22 groups, with each group capturing activity over several temporal windows, including two weeks, one month, six months, one year, and the full session.

Procedural event features summarize how bills move through the legislative process, including committee actions, chamber activity, and important events such as hearings or votes. Sponsor based features capture political support through counts of sponsors, party composition, and co-sponsorship patterns. Bill level metadata adds contextual information such as bill type, chamber of introduction, text size, and age since introduction.

Vote related features provide aggregate indicators of support, and amendment features record both the number of amendments and their chamber of origin. Together, these components create a structured representation of legislative behavior.

B. Model Types and Hyperparameters

We evaluated a diverse set of model families to determine which modeling approaches are compatible with a ranking based prediction task in a legislative context. The model classes included:

- **Scaled Logistic Regression**, using the `saga` solver while tuning penalty and C to find the balance between under and overfitting.
- **Decision Tree**, configured to tune maximum depth, minimum samples, and minimum samples split since they largely influence tree complexity. Focus was put on maximum depth since shallow trees earlier in the process performed better.
- **Random Forest**, tuned the parameters number of estimators, maximum depth, maximum features, and minium samples split to evaluate which complexity and randomness the tree benefited the most from.
- **XGBoost**, configured with the DART booster and tuned values for `gamma` and `max_depth`. Parameter tuning focused on maximum depth since earlier in the process, shallow trees also performed better for this model type.
- **Baselines**, including a DummyClassifier and rankers such as BaselineRankMultiFeature used for bill text size, number of sponsors, number of parties sponsoring, and number of events.

These model's can be found in the triage configuration file (Appendix E), as well as here below:

The best performing models we finalized and compared were:

- **Scaled Logistic Regression**, using the `saga` solver with L2 penalty and C = 0.2.
- **Decision Tree**, configured with a criterion of gini, maximum depth of 3, minimum samples per leaf of 10, and a minimum samples split of 10.
- **Random Forest**, using 1000 trees, maximum depth of 10, 0.01 as the maximum feature setting, and minimum number of samples required to split of 0.001.
- **XGBoost**, configured with the DART booster and

```
# model tuning parameters
  'sklearn.ensemble.RandomForestClassifier':
      n_estimators: [5000]
      max_depth: [40, 50, 60]
      max_features: ['log2', 0.01, 0.1, 0.5]
      n_jobs: [-2]

  'triage.component.catwalk.estimators.classifiers.ScaledLogisticRegression':
      C: [0.05, 0.1, 0.2]
      penalty: ['l1', 'l2']
      solver: ['saga']

  'xgboost.XGBClassifier':
      booster: ['dart']
      tree_method: ["hist"]
      max_depth: [1, 2, 3]
      subsample: [0.25, 0.75, 1]
      gamma: [0, 0.1, 1.0]
      learning_rate: [0.05, 0.1, 0.2]
      n_estimators: [500]

  'sklearn.tree.DecisionTreeClassifier':
      max_depth: [2, 3, 5]
      min_samples_split: [10, 20]
      min_samples_leaf: [5, 10, 15]
      criterion: ['gini']
```

Fig. 2: Models Ran for Parameter Tuning

tuned values for `gamma` of 1, maximum depth of 3, subsample of 1, using a histogram tree method, number of estimators of 100, learning rate of 0.1, and a subsample ratio of columns of 0.3.

- **Baselines**, configured using the events recent feature where it performed best when it ranked bills higher based on the number of events a it had. The best performing baseline multirank feature was when it used the total number of events a bill has ever had up to the 'as of date'.

These model groups were evaluated using four temporal train–validate splits aligned with the training end dates of 2011, 2013, 2015, and 2017. In total, the modeling grid produced 40 trained models across 10 model types. All models were trained using Triage's standard supervised learning pipeline, and evaluation metadata was recorded for each temporal slice.

C. Temporal Cross-Validation Setup

The temporal structure of Wisconsin legislative sessions guided our validation strategy. We applied rolling origin temporal cross-validation, where each model trains on all data available up to a specified cutoff and predicts on the subsequent period. The four training end times (2011, 2013, 2015, 2017) align with natural session boundaries and reflect major transitions in legislative activity.

This setup simulates the operational scenario ACPA faces: making predictions using only historical information and assessing model behavior on future bills. It also allows us to examine whether model performance remains consistent across legislative cycles.

D. Training / Validation Splits

For each split:

- Training data contained all bills introduced before the train end date.
- Validation data covered the next 1.5 years of active bills.
- Monthly as-of dates produced 97 total prediction points.

Fig. 3: Train–Validation Splits Across Legislative Sessions

Cohort sizes varied substantially over time, with some months containing only a few active bills and others exceeding 1,400. The passage baserate also shifted across months, which motivated the need for a ranking evaluation metric.

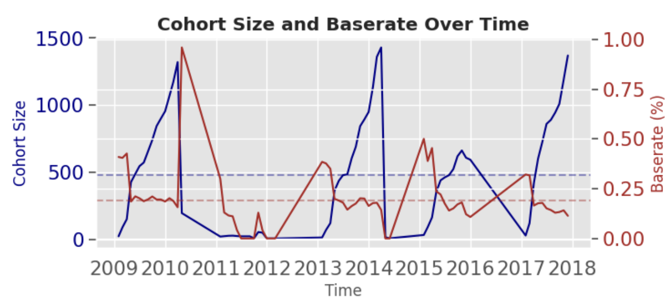

Fig. 4: Cohort size and baserate across monthly prediction points.

VI. Evaluation

A. Overall Performance Metrics

Using precision@15% as the performance priority metric, the best model group achieved an average precision of 0.653 across temporal splits. This corresponds to the Scaled Logistic Regression model group (ID 2589), confirming that relatively simple models can outperform more complex ensembles when data is sparse and highly structured.

Baseline models based on bill text or number of sponsors/political parties performed substantially worse. Dummy classifiers hovered around the base rate, while simple rankers achieved precision values between 0.09 and 0.23. The Baseline feature regarding events did perform a bit better and the highest had a precision value of 0.66.

The ranking stability across splits was consistent with the ExperimentReport summary, which showed that Scaled Logistic Regression had the highest mean performance and smallest cross-split variance.

B. Results Over Time

Temporal performance trends reveal a clear trajectory. Precision@15% improved markedly across the early splits, peaking around 2013–2015, and then declining slightly by 2017. This pattern suggests that mid period legislative cycles were more predictable, possibly due to better event coverage or more stable legislative behaviors.

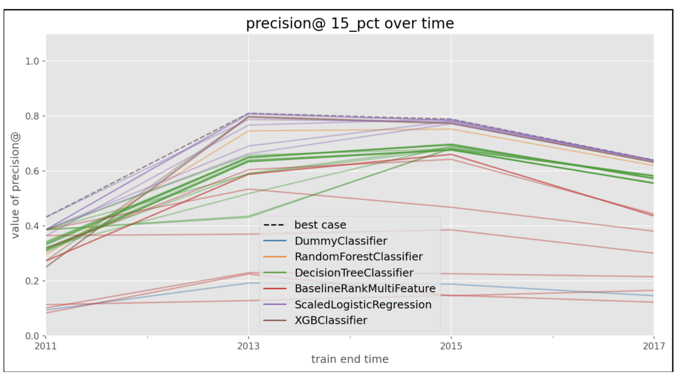

Fig. 5: Precision at 15 Percent Over Time Across Model Families

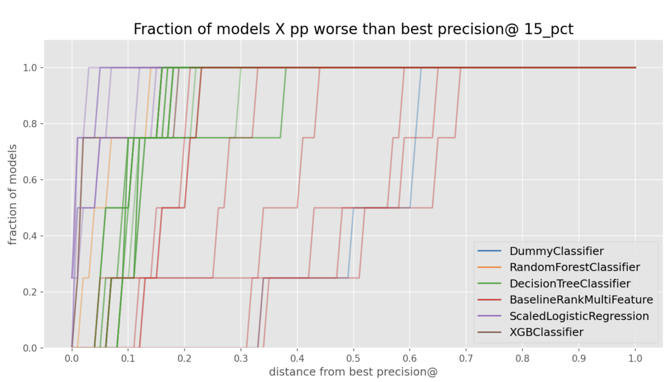

Fig. 6: Distribution of Model Performance Relative to the Best Precision@15%

An alternative view examines how far each model falls from the best case performance in each split. Scaled Logistic Regression frequently approaches the ideal frontier, while tree based methods show larger and more variable gaps.

C. Model Selection

Model selection followed the Triage Audition framework. Using a maximum allowed distance of 1.0 from the best model's precision@15% score, the Auditioner retained 14 candidate model groups that demonstrated competitive and reliable performance across the temporal splits. These retained models represented a diverse set of approaches, but all showed consistent behavior rather than one time peak performance.

The model that was selected was 6038. It is the one that had performed the best across the different train/validation sets. This is a Logistic Regression model with the parameters of ”C”: 0.2, ”solver”: ”saga”, ”penalty”: ”l2”. When compared to the baseline that used the heuristic that ranked bills with more events higher, our model performed much better at 0.828 compared to 0.66.

The audition process emphasized temporal stability in addition to raw accuracy. This criterion is essential for operational use, since ACPA requires a model that performs reliably across legislative cycles rather than excelling only in isolated periods.

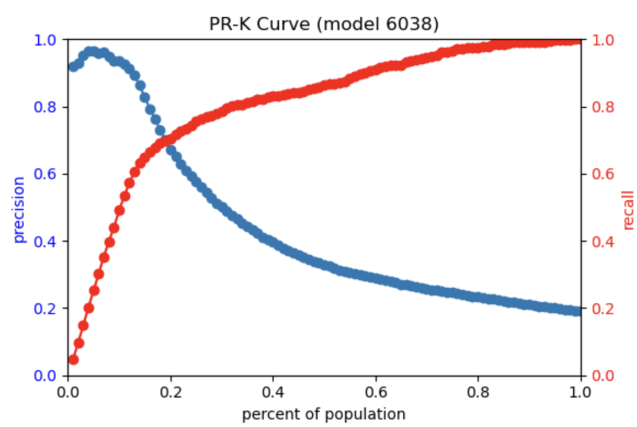

Fig. 7: Precision–Recall–K Curve for Selected Model

D. Feature Importance and Interpretability

Feature inspection revealed a strong dependence on event based features, particularly those capturing:

- recent committee or chamber activity,
- counts of important events,
- sponsor related indicators, and
- bill age or text size.

Feature importance analyses and cross-tab summaries, combined with missingness distributions, suggest that the model captures legislative momentum rather than policy content. This aligns with the insights from our fairness audit groups with lower procedural visibility exhibit weaker prediction performance. Figure 8 below is for the model that was chosen and shows the strong reliance on event based features, specifically on the number of events marked as important a bill has. This aligns with the real world system as a bill that has a lot of important events means that is is actively being discussed and has momentum. Reliance on event based features can also be seen in Figure 9, another top performing model, whose top features were relating to the recency in events a bill has.

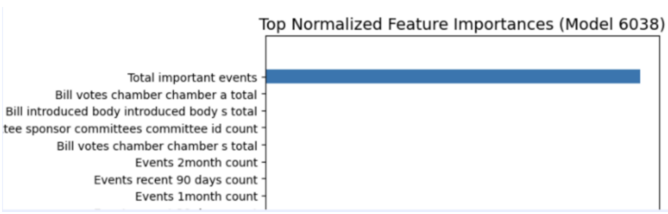

Fig. 8: Top Normalized Feature Importances for Model 6038

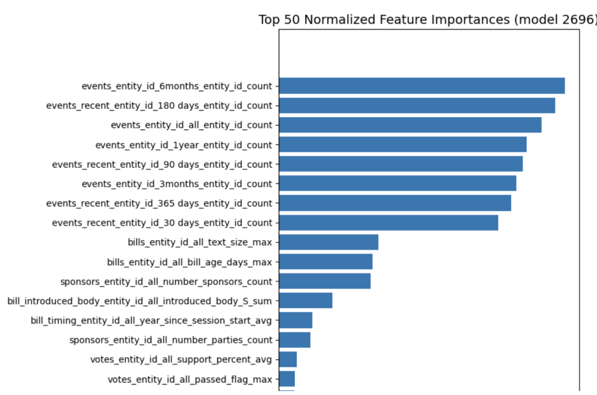

Fig. 9: Top Normalized Feature Importances for Model 2696

E. Bias and Fairness Audit

The fairness audit focuses on whether the model treats bills sponsored by different political parties equitably. This question matters because sponsor party influences visibility, legislative momentum, and the likelihood that a bill gathers procedural events. As shown in our final presentation, the models rely heavily on event driven features, and this reliance has direct implications for how different sponsor groups are represented in the predictions.

1) Attribute of Interest (Sponsor Party): We examine `sponsor_party` because it captures a salient political dimension that shapes legislative outcomes. Party sponsorship influences how quickly a bill gains traction, how much attention it receives, and which committees act on it. These structural forces mean that the sponsor party is not simply a demographic variable but a proxy for access, institutional power, and agenda control. The fairness audit therefore helps determine whether the model amplifies or obscures these deeper political imbalances.

Fig. 10: Disparity metrics across sponsor party groups.

2) Reference Group Definition: Democratic sponsored bills serve as the reference group. This choice reflects both their representational weight in the dataset and their relative stability across sessions. Using Democrats as the baseline allows disparities in True Positive Rate (TPR) and False Discovery Rate (FDR) to be interpreted as deviations from a well populated benchmark. A ratio above one indicates that the comparison group receives more favorable model treatment, while a value below one indicates the opposite.

The model displays measurable differences across sponsor groups. Republican sponsored bills exhibit higher TPR values relative to Democrats, suggesting that the model identifies their successful bills more reliably. Meanwhile, bills sponsored by legislators categorized as Other receive lower TPR values, which means their successful bills are more likely to go undetected.

FDR results move in the other direction. Both Republicans and Other sponsors benefit from lower false discovery rates. When the model predicts these bills will pass, it is less likely to be incorrect. This asymmetry reflects the broader pattern identified in our presentation, namely that the model is learning legislative momentum rather than policy substance. Bills with more procedural activity, which tend to be concentrated among major party sponsors, are easier for the model to classify.

3) Interpretation of Fairness Metrics: Taken together, these results suggest that the model treats sponsor groups unequally. Republican-sponsored bills are both easier for the model to detect and more likely to receive accurate positive predictions. Bills sponsored by the Other category face the opposite challenge. Their predictions are noisier, and successful bills are missed more often.

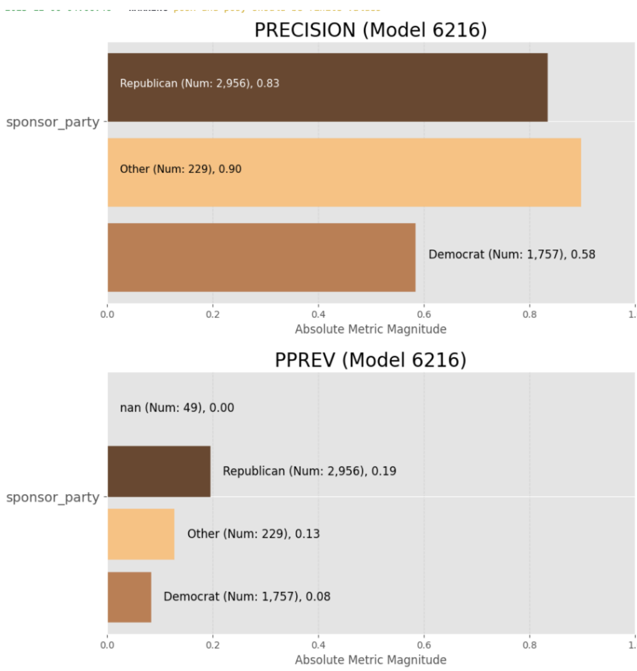

Fig. 11: Precision + PPREV bar charts for Model 621

This pattern is consistent with broader data issues highlighted in the final presentation. Lower amounts in key procedural features, especially amendments and votes, affects groups differently. When a model depends heavily on procedural events, groups with less complete event histories will exhibit lower model performance. The fairness audit therefore reveals not only a model artifact but also a structural property of the Wisconsin legislative process.

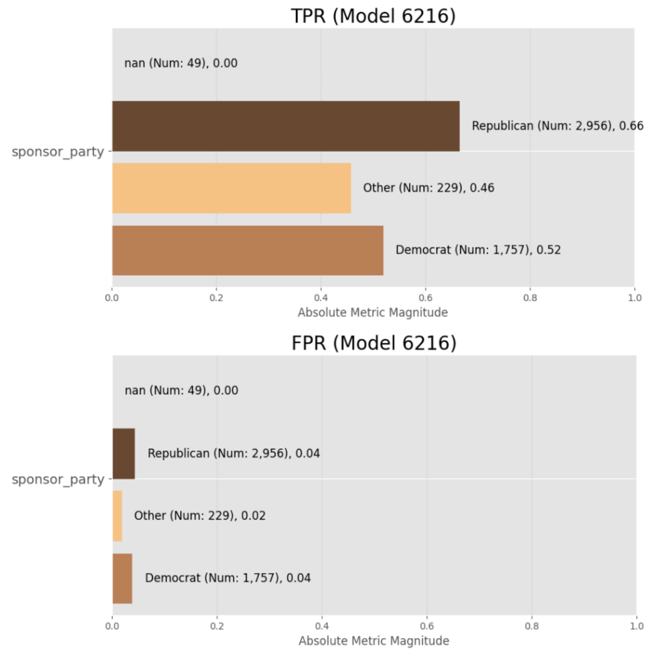

Fig. 12: TPR + FPR bar charts for Model 6216

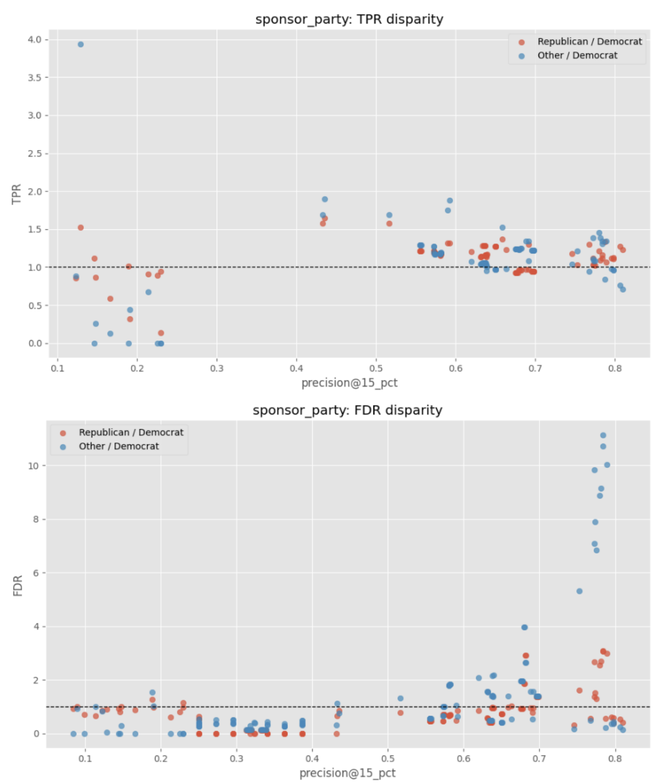

Fig. 13: Sponsor_party: TPR and FDR Disparity

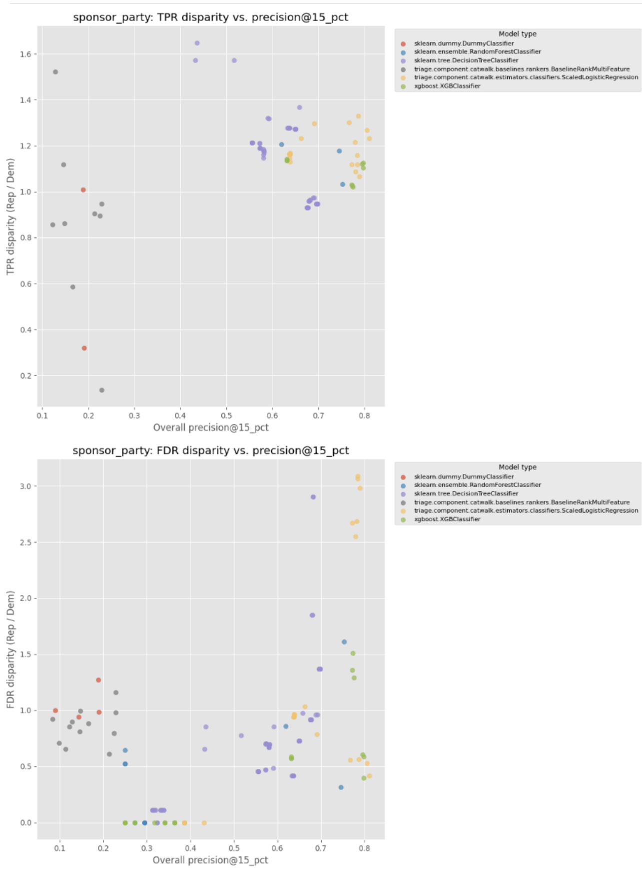

Fig. 14: Sponsor_party: Relationship Between TPR/FDR Disparity and Precision@15%

VII. Discussion of Results

A. What the Results Reveal About the Data

The analysis highlights a core insight: Wisconsin legislative data is deeply uneven. Some bills accumulate extensive procedural histories, while others remain sparse or undocumented. Our final presentation emphasized that many amendment and vote related features exhibit 90–100% missingness. The model learns from what is recorded, not from what is absent. This means that predictions reflect institutional patterns rather than policy substance. In particular, event heavy bills tend to dominate the signal space, while early stage or low activity bills remain difficult for any model to classify.

B. What We Learned About the Predictive Task

Predicting bill passage is feasible but subject to structural constraints. The model performs well when procedural activity is available, and our best method, Scaled Logistic Regression, achieves strong precision in the top 15% of predictions. However, the task becomes significantly more challenging when features are missing or sparse. We also learned that many models excel in similar regions of the feature space, suggesting that the underlying difficulty arises from the legislative process itself rather than model choice.

C. Implications for Policy Decisions

The model has the potential to improve organizational efficiency by identifying high risk bills early. However, fairness disparities introduce real policy concerns. If its outputs are used directly in triage, the system may unintentionally prioritize Republican sponsored bills while deprioritizing those sponsored by smaller or less active parties. This pattern mirrors the caveats raised in the presentation regarding the overdependence on legislative momentum. Decision makers should therefore interpret predictions in light of both accuracy and equity.

VIII. Field Trial Design

A. Purpose of the Field Deployment

A field deployment allows us to evaluate not only the model's predictive performance but also its influence on organizational behavior. It reveals how staff interact with the system, how early they respond to alerts, and whether predictions change intervention timing. This step is essential to understand whether the model produces meaningful improvements in the workflow.

B. Proposed Evaluation Protocol

A randomized assignment procedure will split the incoming bills into a control group following existing practices and a treatment group receiving model generated rankings. Staff decisions, timing and coverage will be monitored. The evaluation must also consider ethical safeguards. Analysts should be reminded that model predictions complement rather than replace expert judgment. Monitoring should track whether the model shifts attention toward bills with already rich procedural histories, which was a documented concern in the presentation.

C. Measuring Organizational Impact

The impact will be evaluated on efficiency, effectiveness, and equity. Efficiency gains can be measured by reductions in time-to-action. Effectiveness will focus on whether the organization flags more high-risk bills before major events occur. Equity must also be analyzed, with attention to whether field deployment amplifies or reduces disparities in sponsor-party representation.

IX. Policy Recommendations

A. How Stakeholders Should Use the Model

The model should be used as a screening tool that highlights bills requiring closer human review. Because its strongest signals come from procedural momentum, predictions made early in the legislative process or for bills with sparse event data should be interpreted cautiously. Human analysts must remain part of the decision process to avoid overlooking important bills that have not yet accumulated events.

B. Recommended Actions Based on Model Insights

Based on performance patterns and fairness results, we recommend four primary operational practices:

1) **Prioritize mid-probability "toss-up" bills.** Precision at 15 percent results show that the model is most informative in the 45 to 65 percent likelihood range. These are the bills where ACPA's intervention can have the greatest impact. Extremely high-probability and very low-probability bills should receive less emphasis.
2) **Use human-in-the-loop review for groups with known disparities.** Sponsor-party fairness results reveal meaningful differences. Republicans show higher TPR values (0.66) but also higher FOR and PPR disparities (for example, FOR at 2.10 and PPR at 3.99). This means bills sponsored by some groups may be over-flagged or under-flagged. Human reviewers should examine predictions for these groups carefully to avoid systematic bias.
3) **Recalibrate and reassess the model each legislative cycle.** Temporal performance drift, including strong improvement through 2013 to 2015 followed by a decline in 2017, indicates that the model's accuracy depends on session specific patterns. Regular updates are necessary to maintain reliability and avoid outdated predictions.
4) **Track and investigate large monthly fluctuations in cohort size and baserate.** Because the number of active bills and the base passage probability vary dramatically across months, model outputs will be more stable in some periods than others. Monitoring these fluctuations helps ACPA detect when predictions may be less reliable due to unusual legislative activity or sparse cohorts.

These four steps represent the most impactful operational changes ACPA can implement based on our model's strengths and limitations.

C. Equity, Effectiveness, and Ethical Considerations

Fairness audits should be incorporated into each session's workflow to prevent unintended disparities. Uneven procedural visibility creates differences in TPR, FDR, and PPR across sponsor groups, which can translate into inequitable prioritization. Ethical deployment requires transparency about where the model performs well, where it does not, and consistent human oversight to ensure that predictions do not reinforce existing structural biases.

X. Limitations and Future Work

A. Data Limitations

Missingness in event and amendment features remains a central limitation of this analysis. Amendment related variables show almost complete missingness across all time windows, and vote-level features often exceed 90 percent missingness. These gaps reduce the amount of usable signal and create uneven visibility across bills. Sparse procedural histories also contribute to instability in fairness metrics, since groups with fewer recorded events may appear less active even when they are substantively meaningful. Temporal variation in cohort size and baserate further complicates evaluation and model consistency across months.

B. Modeling Limitations

The models rely heavily on procedural momentum and do not incorporate bill content. Because event-based features dominate the feature importance results, predictions risk becoming reflections of how often a bill moves through the legislative process rather than a substantive assessment of the bill itself. Temporal drift in performance, including the decline observed in the 2017 split, indicates that procedural signals are session dependent. Without incorporating textual features or relational networks among sponsors, committees, and bill topics, the models cannot capture deeper legislative structure or policy meaning.

C. Bias and Fairness Limitations

Our fairness analysis focused solely on sponsor party. The disparity results show that TPR, FDR, FOR, and PPR vary by sponsor group. For example, Republican sponsored bills receive higher TPR values but also significantly higher FOR and PPR disparities compared to the Democrat reference group. This indicates uneven prediction reliability across political groups. Additional attributes such as chamber, bill type, subject area, or committee assignment were not audited, and intersectional fairness was not examined. Uneven procedural visibility and high missingness likely contribute to these disparities, but the exact mechanisms remain unclear.

D. Recommendations for Future Improvements

Several areas of future work can strengthen both model performance and equitable deployment:

- **Expand fairness analysis across more attributes.** Audit disparities across chamber, bill type, subject area, and sponsor characteristics to identify broader fairness risks. Extend the analysis beyond single attribute comparisons to include intersectional groups.
- **Include NLP features from full bill text.** Incorporate substantive policy content rather than relying only on procedural signals. Text-based embeddings or transformer models can capture the thematic and normative dimensions of legislation that procedural histories cannot represent.
- **Build chamber-specific models.** Train separate models for the House and Senate to reflect their different legislative processes, voting patterns, and procedural rhythms. This could reduce noise introduced by combining heterogeneous legislative dynamics.
- **Improve data completeness.** Strengthen the coverage of events, amendments, vote tallies, and committee activity. Additional data collection or imputation strategies may reduce missingness, which was a major caveat identified throughout the final presentation.
- **Reevaluate temporal structure and update frequency.** Because performance varied substantially across time, future work should experiment with alternative temporal splits and more frequent recalibration to mitigate drift.

Future improvements in these areas would reduce model bias, increase interpretability, and enhance the reliability of predictions in real operational settings.

Appendix

A. *Train/Validation Splits Table*

	Train Set				Validation Set			
Train-Validation Pair ID	Earliest row as of date	Latest row as of date	Start date for labels	End date for labels	Earliest row as of date	Latest row as of date	Start date for labels	End date for labels
1	2013-01-01	2014-04-03	2013-01-01	2014-01-01	2015-01-01	2016-06-27	2015-01-01	2016-01-01
2	2013-01-01	2016-06-27	2013-01-01	2016-01-01	2017-01-01	2018-03-16	2017-01-01	2018-01-01
3 (most recent)	2013-01-01	2018-03-16	2013-01-01	2018-01-01	2019-01-01	2020-04-13	2019-01-01	2020-01-01

Fig. 15: Train Validation Pairs

B. *Temporal Metric Graphs*

Here are our temporal grafts for our most recent runs:

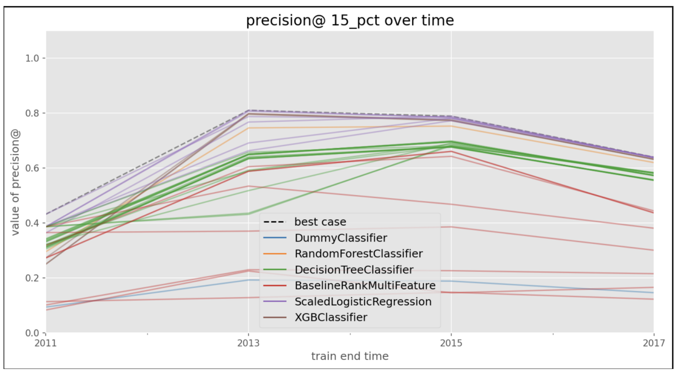

Fig. 16: Precision at 15 percent for our most recent Run

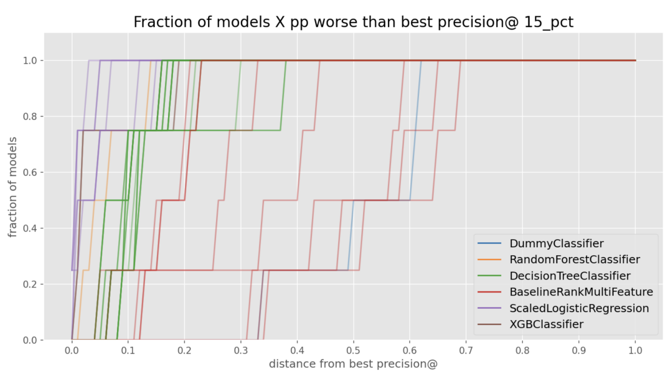

Fig. 17: Model Performance Relative to the Best Precision@15%

C. *Top Model Details*

1) Top Models: The top models that are reasonably different for all of our runs are as follows:

The criteria to select the top models was mean precision @15 percent. Because our top 5 were all very similar, I widened the scope to the top 30 models to get models that performed well but had meaningful differences between them.

Top Selected Models Overview

model_id	model_group_id	model_type	hyperparameters	mean_stochastic_value
2696	689	sklearn.ensemble.RandomForestClassifier	{"max_depth": 10, "max_features": 0.01, "n_estimators": 1000, "min_samples_split": 0.001}	0.8396
6038	2453	triage.component.catwalk.estimators.classifiers.ScaledLogisticRegression	{"C": 0.2, "solver": "saga", "penalty": "l2"}	0.8275
5455	1958	triage.component.catwalk.estimators.classifiers.ScaledLogisticRegression	{"C": 0.1, "solver": "lbfgs", "penalty": "l2"}	0.8235
2691	684	sklearn.ensemble.RandomForestClassifier	{"max_depth": 10, "max_features": "log2", "n_estimators": 100, "min_samples_split": 0.001}	0.8168
3402	914	xgboost.XGBClassifier	{"gamma": 1.0, "booster": "dart", "nthread": 44, "max_depth": 3, "subsample": 1, "tree_method": "hist", "n_estimators": 100, "learning_rate": 0.1, "colsample_bytree": 0.3}	0.8100
6316	2709	triage.component.catwalk.baselines.rankers.BaselineRankMultiFeature	{"rules": [{"feature": "events_recent_entity_id_all_entity_id_count", "low_value_high_score": false}]}	0.6606

Fig. 18: Top Models

2) PR-K graphs: The PR-K graphs for our models in order of best performing to worst are as follows:

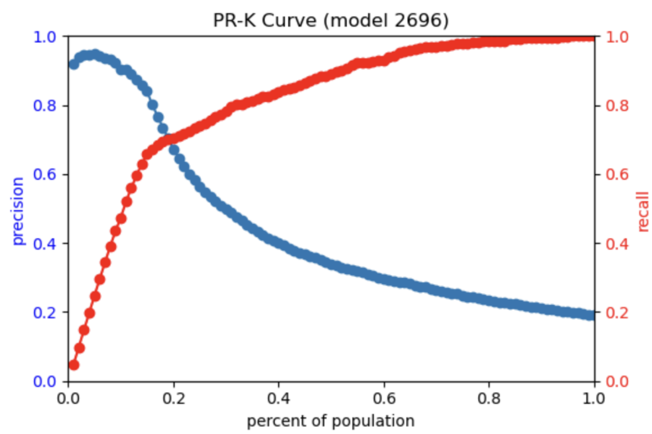

Fig. 19: PR-K for Random Forrest Model 2696

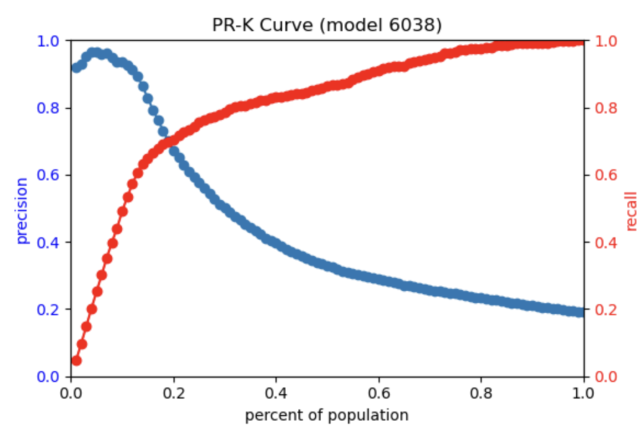

Fig. 20: PR-K for Logistic Regression Model 6038

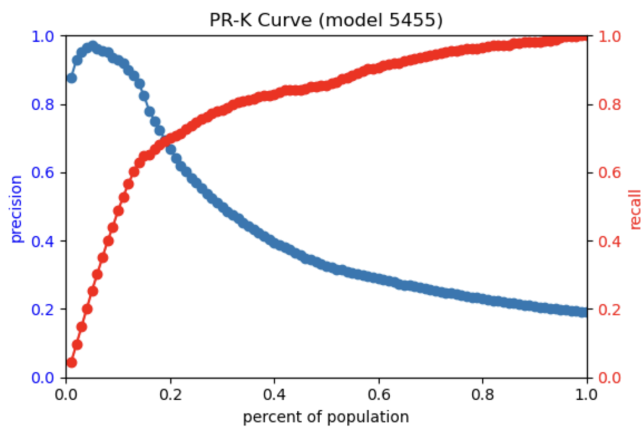

Fig. 21: PR-K for Logistic Regression Model 5455

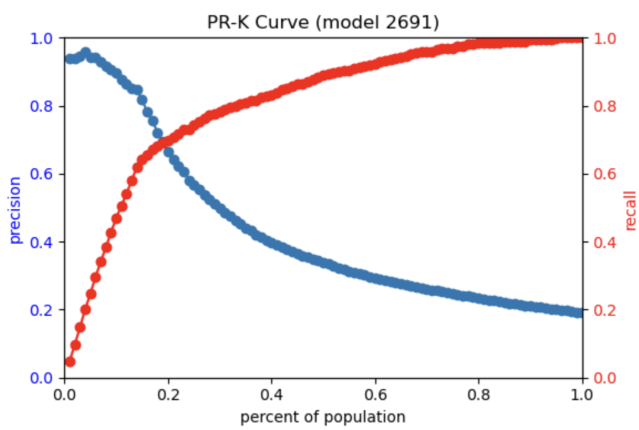

Fig. 22: PR-K for Random Forrest Model 2691

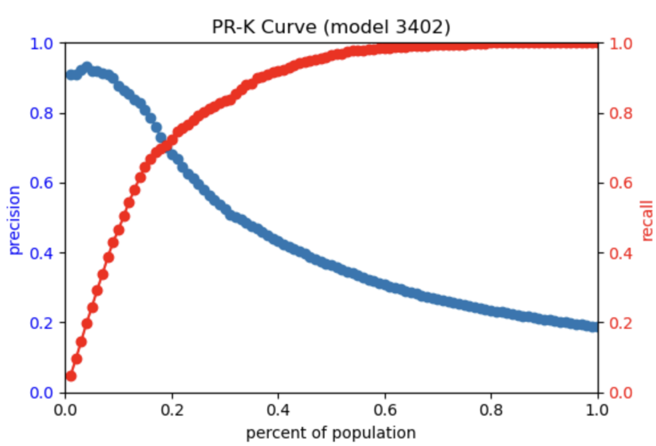

Fig. 23: PR-K for XGBoost Model 3402

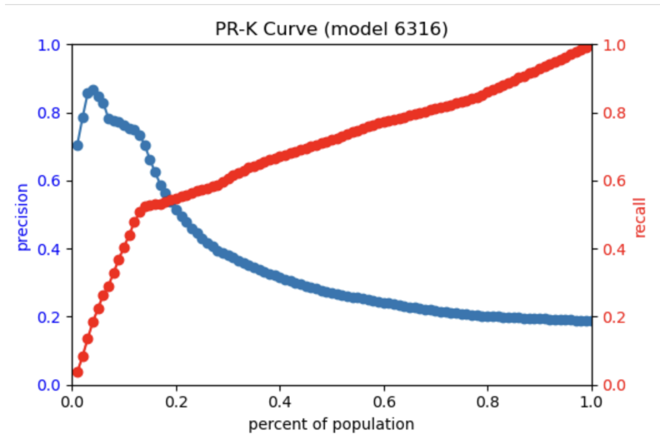

Fig. 24: PR-K for Baseline Model 6316

3) Feature Importance: The feature importance for our models in order of best performing to worst are as follows:

```
==============================
Top 25 Features – Model 2696
==============================
```

	feature	norm_importance
55	events_entity_id_6months_entity_id_count	0.1148
56	events_recent_entity_id_180 days_entity_id_count	0.1110
57	events_entity_id_all_entity_id_count	0.1055
58	events_entity_id_1year_entity_id_count	0.0995
59	events_recent_entity_id_90 days_entity_id_count	0.0979
60	events_entity_id_3months_entity_id_count	0.0952
61	events_recent_entity_id_365 days_entity_id_count	0.0932
62	events_recent_entity_id_30 days_entity_id_count	0.0882
63	bills_entity_id_all_text_size_max	0.0399
64	bills_entity_id_all_bill_age_days_max	0.0375
65	sponsors_entity_id_all_number_sponsors_count	0.0367
66	bill_introduced_body_entity_id_all_introduced_body_S_sum	0.0215
67	bill_timing_entity_id_all_year_since_session_start_avg	0.0135
68	sponsors_entity_id_all_number_parties_count	0.0127
69	votes_entity_id_all_support_percent_avg	0.0073
70	votes_entity_id_all_passed_flag_max	0.0065
71	committee_sponsor_committees_entity_id_all_committee_id_count	0.0060
72	bill_type_entity_id_all_bill_type_JR_sum	0.0022
73	bill_votes_chamber_entity_id_all_chamber_A_sum	0.0022
74	status_entity_id_all_passed_any_chamber_max	0.0021
75	bill_type_entity_id_all_bill_type_B_sum	0.0020
76	bill_votes_chamber_entity_id_all_chamber_S_sum	0.0019
77	bill_votes_chamber_entity_id_all_chamber__NULL_sum	0.0013
78	votes_entity_id_all_support_percent_imp	0.0007
79	votes_entity_id_all_passed_flag_imp	0.0006

Fig. 25: Feature Importance for Random Forrest Model 2696

```
==============================
Top 25 Features – Model 6038
==============================
```

	feature	norm_importance
335	events_important_entity_id_all_important_flag_sum	0.9748
336	bill_votes_chamber_entity_id_all_chamber_A_sum	0.0005
337	bill_introduced_body_entity_id_all_introduced_body_S_sum	0.0004
338	committee_sponsor_committees_entity_id_all_committee_id_count	0.0003
339	bill_votes_chamber_entity_id_all_chamber_S_sum	0.0003
340	events_entity_id_2month_entity_id_count	0.0002
341	events_recent_entity_id_90 days_entity_id_count	0.0002
342	events_entity_id_1month_entity_id_count	0.0002
343	events_recent_entity_id_30 days_entity_id_count	0.0002
344	events_entity_id_3months_entity_id_count	0.0002
345	bills_entity_id_6months_bill_age_days_max	0.0002
346	bills_entity_id_3months_bill_age_days_max	0.0002
347	bills_entity_id_10month_bill_age_days_max	0.0002
348	bills_entity_id_3months_text_size_max	0.0002
349	bills_entity_id_2month_bill_age_days_max	0.0002
350	bills_entity_id_8month_bill_age_days_max	0.0002
351	bills_entity_id_1year_bill_age_days_max	0.0002
352	events_entity_id_10month_entity_id_imp	0.0002
353	events_entity_id_6months_entity_id_imp	0.0002
354	bill_timing_entity_id_all_year_since_session_start_imp	0.0002
355	bill_introduced_body_entity_id_all_introduced_body__NULL_sum	0.0002
356	events_entity_id_3months_entity_id_imp	0.0002
357	events_entity_id_2month_entity_id_imp	0.0002
358	bill_type_entity_id_all_bill_type__NULL_sum	0.0002
359	events_entity_id_1year_entity_id_imp	0.0002

Fig. 26: Feature Importance for Logistic Regression Model 6038

```
==============================
Top 25 Features — Model 5455
==============================
```

	feature	norm_importance
165	events_important_entity_id_all_important_flag_sum	0.8873
166	bill_introduced_body_entity_id_all_introduced_body_S_sum	0.0015
167	bill_votes_chamber_entity_id_all_chamber_A_sum	0.0014
168	committee_sponsor_committees_entity_id_all_committee_id_count	0.0013
169	bill_votes_chamber_entity_id_all_chamber_S_sum	0.0010
170	events_recent_entity_id_90 days_entity_id_count	0.0010
171	events_entity_id_2month_entity_id_count	0.0010
172	events_entity_id_3months_entity_id_count	0.0010
173	bills_entity_id_6months_bill_age_days_max	0.0010
174	bills_entity_id_3months_bill_age_days_max	0.0010
175	events_entity_id_1month_entity_id_count	0.0009
176	events_recent_entity_id_30 days_entity_id_count	0.0009
177	bills_entity_id_10month_bill_age_days_max	0.0009
178	bills_entity_id_8month_bill_age_days_max	0.0009
179	events_recent_entity_id_180 days_entity_id_count	0.0009
180	events_entity_id_6months_entity_id_count	0.0009
182	sponsors_entity_id_all_number_parties_count	0.0009
181	sponsors_bipartisan_entity_id_all_party_id_count	0.0009
183	bills_entity_id_2month_bill_age_days_max	0.0008
184	bills_entity_id_1year_bill_age_days_max	0.0008
185	bill_type_entity_id_all_bill_type_B_sum	0.0008
186	bills_entity_id_1month_bill_age_days_max	0.0008
187	bills_entity_id_3months_text_size_max	0.0008
194	events_entity_id_1year_entity_id_imp	0.0008
198	events_entity_id_8month_entity_id_imp	0.0008

Fig. 27: Feature Importance for Logistic Regression Model 5455

```
==============================
Top 25 Features — Model 2691
==============================
```

	feature	norm_importance
0	events_entity_id_6months_entity_id_count	0.1413
1	events_recent_entity_id_90 days_entity_id_count	0.1008
2	events_recent_entity_id_180 days_entity_id_count	0.0970
3	events_recent_entity_id_30 days_entity_id_count	0.0916
4	events_entity_id_all_entity_id_count	0.0878
5	events_recent_entity_id_365 days_entity_id_count	0.0824
6	events_entity_id_3months_entity_id_count	0.0812
7	events_entity_id_1year_entity_id_count	0.0806
8	bills_entity_id_all_text_size_max	0.0552
9	bills_entity_id_all_bill_age_days_max	0.0484
10	sponsors_entity_id_all_number_sponsors_count	0.0474
11	bill_introduced_body_entity_id_all_introduced_body_S_sum	0.0280
12	bill_timing_entity_id_all_year_since_session_start_avg	0.0140
13	sponsors_entity_id_all_number_parties_count	0.0131
14	committee_sponsor_committees_entity_id_all_committee_id_count	0.0080
15	votes_entity_id_all_passed_flag_max	0.0067
16	votes_entity_id_all_support_percent_avg	0.0053
17	bill_type_entity_id_all_bill_type_JR_sum	0.0020
18	bill_votes_chamber_entity_id_all_chamber_A_sum	0.0019
19	bill_votes_chamber_entity_id_all_chamber_S_sum	0.0019
20	bill_type_entity_id_all_bill_type_B_sum	0.0016
21	status_entity_id_all_passed_any_chamber_max	0.0016
22	bill_votes_chamber_entity_id_all_chamber__NULL_sum	0.0011
23	votes_entity_id_all_passed_flag_imp	0.0006
24	votes_entity_id_all_support_percent_imp	0.0005

Fig. 28: Feature Importance for Random Forrest Model 2691

```
==============================
Top 25 Features — Model 3402
==============================
```

	feature	norm_importance
110	events_entity_id_6months_entity_id_count	0.2510
111	events_entity_id_1year_entity_id_count	0.0776
112	events_recent_entity_id_90 days_entity_id_count	0.0774
113	events_recent_entity_id_180 days_entity_id_count	0.0713
114	events_recent_entity_id_30 days_entity_id_count	0.0690
115	events_entity_id_all_entity_id_count	0.0574
116	bill_votes_chamber_entity_id_all_chamber_S_sum	0.0520
117	bill_votes_chamber_entity_id_all_chamber_A_sum	0.0345
118	events_entity_id_3months_entity_id_count	0.0327
119	bill_votes_chamber_entity_id_all_chamber__NULL_sum	0.0293
120	sponsors_entity_id_all_number_parties_count	0.0227
121	votes_entity_id_all_passed_flag_imp	0.0222
122	bill_introduced_body_entity_id_all_introduced_body_S_sum	0.0203
123	committee_sponsor_committees_entity_id_all_committee_id_count	0.0173
124	events_recent_entity_id_365 days_entity_id_count	0.0156
125	bills_entity_id_all_text_size_max	0.0155
126	bill_type_entity_id_all_bill_type_JR_sum	0.0152
127	bill_type_entity_id_all_bill_type_B_sum	0.0147
128	bills_entity_id_all_bill_age_days_max	0.0144
129	votes_entity_id_all_support_percent_avg	0.0124
130	sponsors_entity_id_all_number_sponsors_count	0.0122
131	bill_timing_entity_id_all_year_since_session_start_avg	0.0097
132	bill_introduced_body_entity_id_all_introduced_body__NULL_sum	0.0096
133	votes_entity_id_all_passed_flag_max	0.0088
134	status_entity_id_all_passed_any_chamber_max	0.0087

Fig. 29: Feature Importance for XGBoost Model 3402

```
==============================
Top 25 Features — Model 6316
==============================
```

	feature	norm_importance
505	events_recent_entity_id_all_entity_id_count	1.0000
627	sponsors_entity_id_1month_number_parties_count	0.0000
609	events_entity_id_1year_entity_id_imp	0.0000
610	events_entity_id_2week_entity_id_count	0.0000
611	events_entity_id_2week_entity_id_imp	0.0000
612	events_entity_id_6month_entity_id_count	0.0000
613	events_entity_id_6month_entity_id_imp	0.0000
614	events_entity_id_all_entity_id_count	0.0000
615	events_entity_id_all_entity_id_imp	0.0000
616	events_important_entity_id_1month_important_flag_sum	0.0000
617	events_important_entity_id_1year_important_flag_sum	0.0000
618	events_important_entity_id_2week_important_flag_sum	0.0000
619	events_important_entity_id_6month_important_flag_sum	0.0000
620	events_important_entity_id_all_important_flag_sum	0.0000
621	events_recent_entity_id_1month_entity_id_count	0.0000
622	events_recent_entity_id_1year_entity_id_count	0.0000
623	events_recent_entity_id_2week_entity_id_count	0.0000
624	events_recent_entity_id_6month_entity_id_count	0.0000
625	sponsors_bipartisan_entity_id_all_party_id_count	0.0000
608	events_entity_id_1year_entity_id_count	0.0000
607	events_entity_id_1month_entity_id_imp	0.0000
606	events_entity_id_1month_entity_id_count	0.0000
596	bill_votes_chamber_entity_id_all_chamber_S_sum	0.0000
589	bill_votes_chamber_entity_id_2week_chamber__NULL_sum	0.0000
590	bill_votes_chamber_entity_id_2week_chamber_S_sum	0.0000

Fig. 30: Feature Importance for Baseline Model 6136

4) Cross-Tabs: The top 10 feature cross-tabs for our models in order of best performing to worst are as follows:

Top Crosstab Features – Model 2696

feature	Mean on Top-15%	Mean on Bottom (100-15)%	Mean Ratio (Top / Bottom)
bill_votes_chamber_entity_id_all_chamber_S_sum	0.565	0.008	68.448
bill_votes_chamber_entity_id_all_chamber_A_sum	0.502	0.050	9.998
events_recent_entity_id_30 days_entity_id_count	6.415	0.869	7.385
events_recent_entity_id_90 days_entity_id_count	12.696	2.075	6.117
events_entity_id_3months_entity_id_count	12.766	2.097	6.087
events_recent_entity_id_180 days_entity_id_count	16.036	3.286	4.879
events_entity_id_6months_entity_id_count	16.121	3.334	4.835
events_entity_id_all_entity_id_count	16.987	4.119	4.124
events_recent_entity_id_365 days_entity_id_count	16.987	4.119	4.124
events_entity_id_1year_entity_id_count	16.987	4.119	4.124

Fig. 31: Cross-Tabs of Top 10 Features for Model 2696

Top Crosstab Features – Model 6038

feature	Mean on Top-15%	Mean on Bottom (100-15)%	Mean Ratio (Top / Bottom)
vote_chamber_counts_entity_id_all_chamber_S_sum	0.247	0.001	349.252
bill_votes_chamber_entity_id_all_chamber_S_sum	0.541	0.012	43.278
bill_votes_chamber_entity_id_all_chamber_A_sum	0.534	0.045	11.986
events_recent_entity_id_30 days_entity_id_count	6.386	0.874	7.307
events_entity_id_1month_entity_id_count	6.450	0.888	7.267
events_entity_id_2month_entity_id_count	10.406	1.598	6.513
events_recent_entity_id_90 days_entity_id_count	12.270	2.151	5.705
events_entity_id_3months_entity_id_count	12.340	2.173	5.680
events_recent_entity_id_180 days_entity_id_count	15.231	3.429	4.442
events_entity_id_6months_entity_id_count	15.299	3.479	4.397

Fig. 32: Cross-Tabs of Top 10 Features for Model 6038

Top Crosstab Features – Model 5455

feature	Mean on Top-15%	Mean on Bottom (100-15)%	Mean Ratio (Top / Bottom)
vote_chamber_counts_entity_id_all_chamber_S_sum	0.250	0.000	1059.084
bill_votes_chamber_entity_id_all_chamber_S_sum	0.523	0.016	33.638
bill_votes_chamber_entity_id_all_chamber_A_sum	0.477	0.055	8.715
events_recent_entity_id_30 days_entity_id_count	6.547	0.845	7.745
events_entity_id_1month_entity_id_count	6.609	0.860	7.689
events_entity_id_2month_entity_id_count	10.725	1.541	6.959
events_recent_entity_id_90 days_entity_id_count	12.607	2.091	6.029
events_entity_id_3months_entity_id_count	12.680	2.113	6.002
events_chamber_counts_entity_id_all_chamber_S_sum	8.266	1.751	4.721
events_recent_entity_id_180 days_entity_id_count	15.566	3.369	4.620

Fig. 33: Cross-Tabs of Top 10 Features for Model 5455

Top 10 Crosstab Features for Model 2691

Feature	Mean on Top-15%	Mean on Bottom (100-15)%	Mean Ratio (Top / Bottom)
bill_votes_chamber_entity_id_all_chamber_S_sum	0.547000	0.011000	48.376000
events_recent_entity_id_30 days_entity_id_count	6.681000	0.822000	8.130000
bill_votes_chamber_entity_id_all_chamber_A_sum	0.447000	0.060000	7.470000
events_recent_entity_id_90 days_entity_id_count	13.099000	2.004000	6.536000
events_entity_id_3months_entity_id_count	13.167000	2.027000	6.497000
events_recent_entity_id_180 days_entity_id_count	16.151000	3.266000	4.945000
events_entity_id_6months_entity_id_count	16.224000	3.316000	4.893000
events_entity_id_all_entity_id_count	17.088000	4.101000	4.167000
events_recent_entity_id_365 days_entity_id_count	17.088000	4.101000	4.167000
events_entity_id_1year_entity_id_count	17.088000	4.101000	4.167000

Fig. 34: Cross-Tabs of Top 10 Features for Model 2691

Top 10 Crosstab Features for Model 3402

Feature	Mean on Top-15%	Mean on Bottom (100-15)%	Mean Ratio (Top / Bottom)
bill_votes_chamber_entity_id_all_chamber_S_sum	0.395000	0.005000	82.445000
status_entity_id_all_passed_any_chamber_max	0.005000	0.001000	8.496000
events_recent_entity_id_30 days_entity_id_count	6.039000	0.918000	6.576000
events_recent_entity_id_90 days_entity_id_count	12.916000	2.373000	5.444000
events_entity_id_3months_entity_id_count	12.971000	2.401000	5.403000
events_recent_entity_id_180 days_entity_id_count	17.127000	3.947000	4.339000
events_entity_id_6months_entity_id_count	17.220000	3.995000	4.310000
events_entity_id_1year_entity_id_count	18.627000	4.912000	3.792000
events_entity_id_all_entity_id_count	18.627000	4.912000	3.792000
events_recent_entity_id_365 days_entity_id_count	18.627000	4.912000	3.792000

Fig. 35: Cross-Tabs of Top 10 Features for Model 3402

Top 10 Crosstab Features for Model 6316

Feature	Mean on Top-15%	Mean on Bottom (100-15)%	Mean Ratio (Top / Bottom)
events_entity_id_1month_entity_id_count	5.883000	0.958000	6.140000
events_recent_entity_id_1month_entity_id_count	5.883000	0.958000	6.140000
events_chamber_counts_entity_id_all_chamber_S_sum	10.404000	1.972000	5.275000
bill_votes_chamber_entity_id_1year_chamber_S_sum	0.202000	0.039000	5.199000
bill_votes_chamber_entity_id_all_chamber_S_sum	0.202000	0.039000	5.199000
events_recent_entity_id_6month_entity_id_count	19.036000	3.674000	5.181000
events_entity_id_6month_entity_id_count	19.036000	3.674000	5.181000
events_entity_id_all_entity_id_count	21.441000	4.415000	4.857000
events_recent_entity_id_all_entity_id_count	21.441000	4.415000	4.857000
events_entity_id_1year_entity_id_count	21.441000	4.415000	4.857000

Fig. 36: Cross-Tabs of Top 10 Features for Model 6316

5) Bias Audit Outputs: This appendix includes the full set of fairness metrics used in the audit, including TPR, FDR, PPR, and other disparity measures across sponsor_party groups. These correspond to the analyses discussed in Section 6.E of the main report.

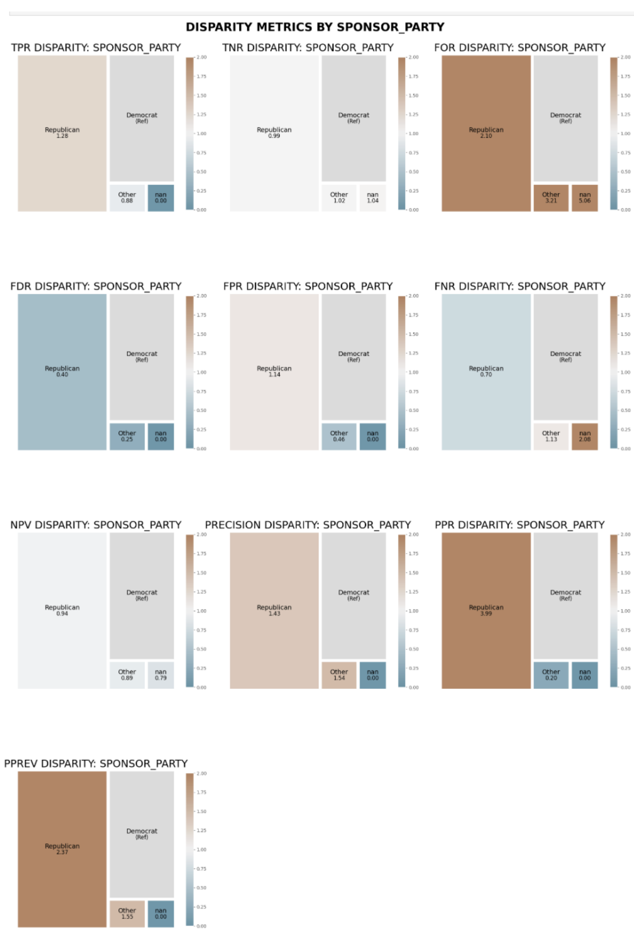

Fig. 37

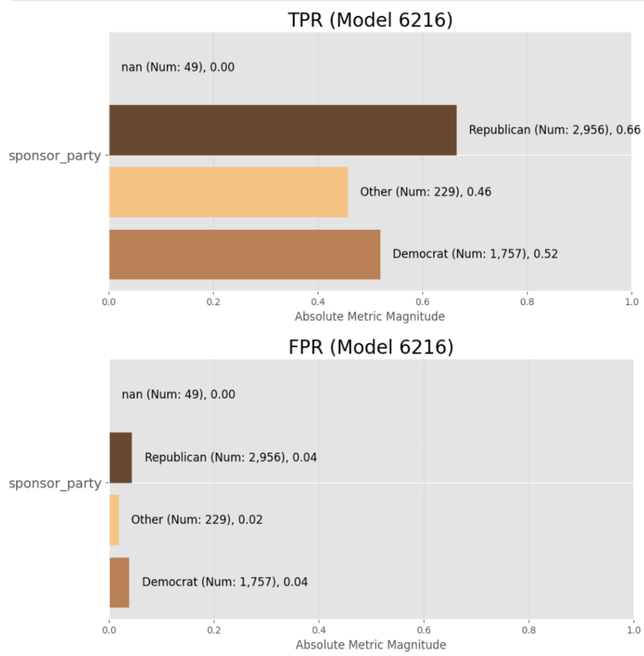

Fig. 38

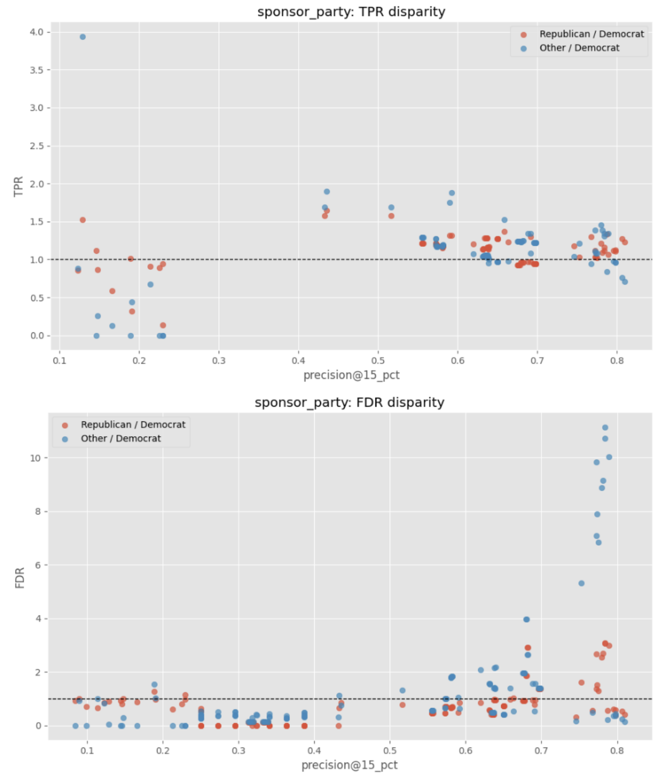

Fig. 40

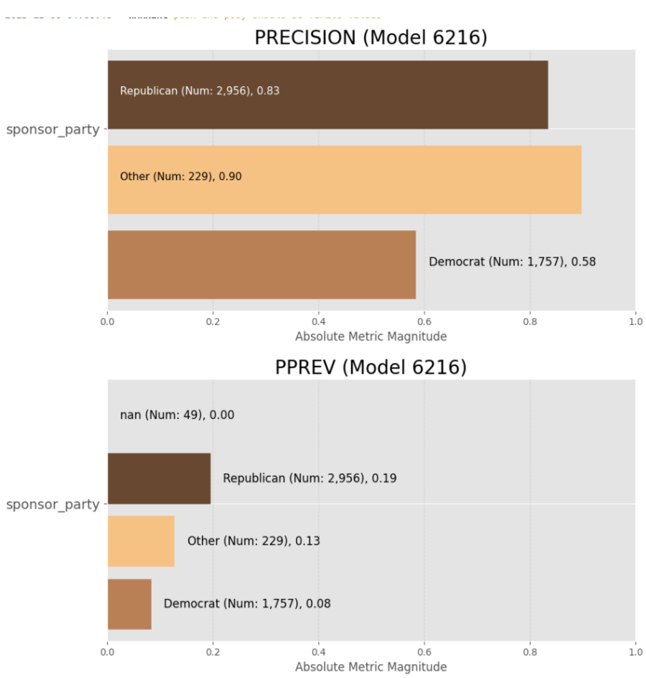

Fig. 39

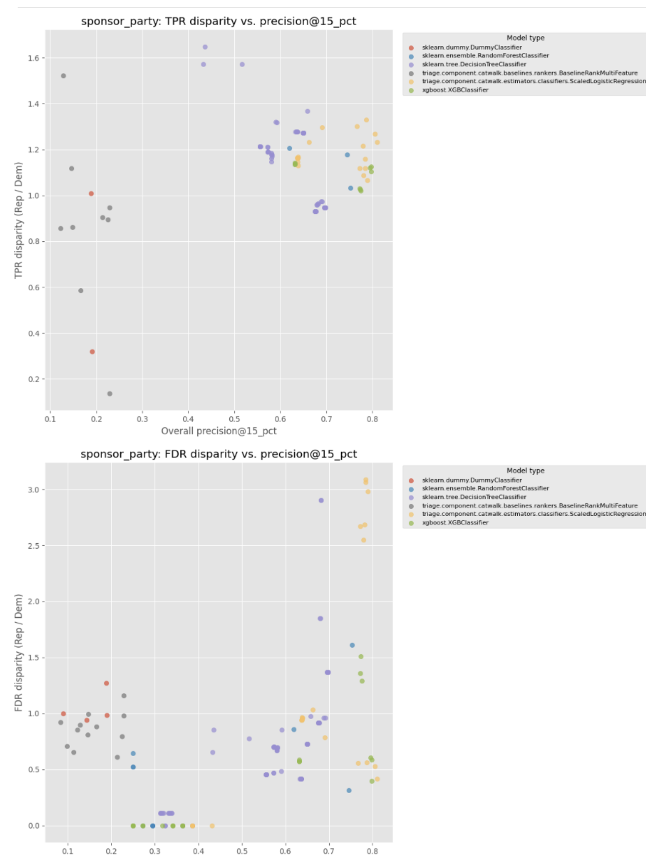

Fig. 41

D. Github Repository

1) Repo Structure and Instructions: The repo includes the following files:

- **bills_triage_config.yaml**: Has been updated with work conducted over the course of the semester. You can find the cohort, label generation, feature generation, bias audit, and model grid that were used. It currently holds our larger model grid where parameter tuning was conducted, and has all of our features.
- **small_grid.yaml**: Where smaller model grids were used with the same features.
- **run.py**: Used to run the code in the yaml files where line 18 defines where the triage.log output should go, and on line 50 the yaml file used is defined.
- **1_checking_results.ipynb, 2_bias_&_fairness.ipynb**: Used for bias auditing based on political parties of sponsors. To run, you would need to update necessary paths based on user
- **models_over_time.ipynb**: Analysis of models conducted using auditioner and crosstabs.

2) Documented Code Files Link: Links to the files can be found here:

- **bills_triage_config.yaml**
- **small_grid.yaml**
- **run.py**
- **1_checking_results.ipynb**
- **models_over_time.ipynb**
- **2 bias and fairness**

E. Triage Configuration File: Including Features and Model Grid

A pdf version of the triage config file (bills_triage_config.yaml) can be seen in the pages below. It can also be viewed on our GitHub.

```
config_version: 'v8'

model_comment: 'dev-config'
random_seed: 23895478

# TIME SPLITTING
# The time window to look at, and how to divide the window into
# train/test splits
temporal_config:
  feature_start_time: '2009-01-01' # earliest date included in
features
  feature_end_time: '2019-01-01'  # latest date included in features
  label_start_time: '2009-01-01' # earliest date for which labels are
avialable
  label_end_time: '2019-01-01' # day AFTER last label date (all dates
in any model are < this date)
  model_update_frequency: '2years' # how frequently to retrain models
(using 100year here to just get one split)
  training_as_of_date_frequencies: ['1month'] # time between as of
dates for same entity in train matrix
  test_as_of_date_frequencies: ['1month'] # time between as of dates
for same entity in test matrix
  max_training_histories: ['100years'] # length of time included in a
train matrix
  test_durations: ['1year'] # length of time included in a test matrix
(0 days will give a single prediction immediately after training end)
  # like our project timeout
  training_label_timespans: ['1year']
  test_label_timespans: ['1year']



# COHORT & LABEL GENERATION
# Labels are configured with a query with placeholders for the
'as_of_date' and 'label_timespan'. You can include a local path to a
sql file containing the label query to the 'filepath' key (preferred)
or include the query in the 'query' key
#
# The query must return two columns: entity_id and outcome, based on a
given as_of_date and label_timespan.
# The as_of_date and label_timespan must be represented by
placeholders marked by curly brackets.
#
# In addition to these configuration options, you can pass a name to
apply to the label configuration
# that will be present in matrix metadata for each matrix created by
this experiment,
# under the 'label_name' key. The default label_name is 'outcome'.
# Finds Wisconsin bills (id = 49), whose progress date is within the
label_timespan, and are introduced after as_of_date
```

```
# DEFINITION OF LABEL
# Definition of a positive label (outcome = 1):
#   For a bill, if it is marked with a status = 4 (passed) in
bill_progress at any point between:
#   as_of_date  < progress_date <= as_of_date + label_timespan
#   then this bill is labeled as 1
#
# Definition of a negative label (outcome = 0):
#   - Bills that have NO progress record with status = 4 in that
window
#
# ASSUMPTIONS:
#   Bills must have at least one progress event in the timespan
(bp.bill_id IS NOT NULL)
#   Bills must have been introduced on or before the as_of_date
#   Only Wisconsin bills are included (state_id = 49)
#
# Label Timespan:
#   Defines how far into the future after as_of_date we consider a
bill eligible to pass
label_config:
  name: 'wi_bill_passed_within_label_window'
  query: |
    SELECT
      b.bill_id AS entity_id,
      CASE
        WHEN MAX(CASE WHEN bp.bill_status = 4 THEN 1 ELSE 0 END) = 1
THEN 1
        ELSE 0
      END AS outcome
    FROM ml_policy_class.bills b
    LEFT JOIN ml_policy_class.sessions s
      ON b.session_id = s.session_id
    LEFT JOIN ml_policy_class.bill_progress bp
      ON b.bill_id = bp.bill_id
      AND bp.progress_date > '{as_of_date}'::date
      AND bp.progress_date <= ('{as_of_date}'::date + INTERVAL
'{label_timespan}')
    WHERE s.state_id = 49
          AND bp.bill_id IS NOT NULL   -- this is to exclude bills
that do not have any progress in timespan
          AND b.introduced_date <= '{as_of_date}'::date
    GROUP BY b.bill_id

  # name: 'quickstart_label' # optionally, give your label a name to
help track results (uncomment if using)


# FEATURE GENERATION
# The aggregate features to generate for each train/test split
```

```
feature_aggregations:
# initial events based feature
  - prefix: 'events'
    from_obj: |
      (SELECT bill_id::INT AS entity_id,
              event_date AS knowledge_date,
              chamber
       FROM ml_policy_class.bill_events) AS bill_events
    knowledge_date_column: 'knowledge_date'
    aggregates_imputation:
      all:
        type: 'constant'
        value: 'zero_noflag'
    aggregates:
      - quantity: 'entity_id'
        metrics: ['count']
    intervals: ['2week', '1month', '6month', '1year', 'all']
# finds number of sponsors of a bill, and number of political parties
sponsoring a bill
  - prefix: 'sponsors'
    from_obj: |
      (SELECT bill_id::INT AS entity_id,
              introduced_date AS knowledge_date,
              sponsor_id,
              party_id,
              role_name,
              sponsor_type
       FROM ml_policy_class.bill_sponsors
       LEFT JOIN ml_policy_class.bills USING (bill_id)) AS sponsors
    knowledge_date_column: 'knowledge_date'
    aggregates_imputation:
      all:
        type: 'zero_noflag'
    categoricals_imputation:
      all:
        type: 'null_category'
    aggregates:
      - # total number of unique sponsors
        quantity:
          number_sponsors: "distinct sponsor_id"
        metrics:
          - 'count'
      - # total number of unique parties sponsoring this bill
        quantity:
          number_parties: "distinct party_id"
        metrics:
           - 'count'
    intervals: ['2week', '1month', '6month', '1year', 'all']
# finds age of bill
  - prefix: "bills"
```

```
    from_obj: |
      (SELECT
          b.bill_id::INT AS entity_id,
          b.introduced_date AS knowledge_date,
          b.bill_type,
          b.subjects,
          b.introduced_body,
          bt.text_size,
          (b.introduced_date::date) AS bill_age_days
       FROM ml_policy_class.bills b
       LEFT JOIN ml_policy_class.bill_texts bt USING (bill_id)) AS
bills
    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
      all:
        type: "zero_noflag"
    aggregates:
      -
        quantity:
          bill_age_days: "'{collate_date}'::DATE -
knowledge_date::DATE"
        metrics:
          - 'max'
      - quantity: "text_size"
        metrics: ["max"]
    intervals: ["all"]


    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
      all:
        type: "zero_noflag"

    intervals: ['2week', '1month', '6month', '1year', 'all']
# support_percent finds proportion of YEA votes, passed_flag finds
whether the bill passed in that vote event
  - prefix: "votes"
    from_obj: |
      (SELECT
          bv.bill_id::INT AS entity_id,
          bv.vote_date AS knowledge_date,
          (bv.yea::FLOAT / NULLIF(bv.total, 0)) AS support_percent,
          bv.passed::INT AS passed_flag
       FROM ml_policy_class.bill_votes bv) AS votes
    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
      all: {type: "mean"}
    aggregates:
      - quantity: "support_percent"
        metrics: ["avg"] # impute with zero_noflag?
```

```
      - quantity: "passed_flag"
        metrics: ["max"]
    intervals: ['2week', '1month', '6month', '1year', 'all']
# amendment_count finds the total number of amendments, adopted_flag
represents whether an amendment was adopted, not_adopted_flag
represents whether an amendment failed
  - prefix: "amendments"
    from_obj: |
      (SELECT
          ba.bill_id::INT AS entity_id,
          ba.amendment_date AS knowledge_date,
          ba.adopted::INT AS adopted_flag,
          1 AS amendment_count,
          CASE WHEN ba.adopted = 1 THEN 0 ELSE 1 END AS
not_adopted_flag
       FROM ml_policy_class.bill_amendments ba) AS amendments
    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
      all: {type: "zero_noflag"}
    aggregates:
    - quantity: "amendment_count" # counts number of amendments
      metrics: ["sum"] # zero_noflag?
    - quantity: "adopted_flag" # counts number of adopted
      metrics: ["sum", "avg"] # zero_noflag?
    # - quantity: "not_adopted_flag" #counts number of not adopted
    #   metrics: ["sum"]

    intervals: ['2week', '1month', '6month', '1year', 'all']

# finds whether the bill has ever passed any chamber (house or senate)
  - prefix: "status"
    from_obj: |
      (SELECT bill_id::INT AS entity_id,
              progress_date AS knowledge_date,
              MAX(CASE WHEN bill_status = 4 THEN 1 ELSE 0 END) AS
passed_any_chamber
       FROM ml_policy_class.bill_progress
       GROUP BY bill_id, progress_date) AS status
    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
      all: {type: "zero_noflag"}
    aggregates:
      - quantity: "passed_any_chamber"
        metrics: ["max"]
    intervals: ["all"]
# counts all bill events to represent recent activity
  - prefix: "events_recent"
    from_obj: |
      (SELECT bill_id::INT AS entity_id,
              event_date AS knowledge_date
```

```
        FROM ml_policy_class.bill_events) AS events_recent
      knowledge_date_column: "knowledge_date"
      aggregates_imputation:
        all: {type: "zero_noflag"}
      aggregates:
        - quantity: "entity_id"
          metrics: ["count"]
      intervals: ['2week', '1month', '6month', '1year', 'all']
  # finds which distinct committees are sponsoring a bill
    - prefix: 'committee_sponsor_committees'
      from_obj: |
        (SELECT
          b.bill_id::INT AS entity_id,
          b.introduced_date AS knowledge_date,
          sp.committee_id
        FROM ml_policy_class.bills b
        LEFT JOIN ml_policy_class.sessions_people sp
          ON b.session_id = sp.session_id
        WHERE sp.committee_sponsor = TRUE
        ) AS committee_sponsor_committees
      knowledge_date_column: 'knowledge_date'
      aggregates_imputation:
        all:
          type: 'zero_noflag'
      aggregates:
        - quantity:
            committee_id: "distinct committee_id"
          metrics:
            - 'count'
      intervals: ['all']
  # which chambers the bill has been voted in (house or senate)
    - prefix: 'bill_votes_chamber'
      from_obj: |
        (SELECT
          b.bill_id::INT AS entity_id,
          b.introduced_date AS knowledge_date,
          bv.chamber
        FROM ml_policy_class.bills b
        LEFT JOIN ml_policy_class.bill_votes bv
          ON b.bill_id = bv.bill_id
        GROUP BY b.bill_id, b.introduced_date, bv.chamber) AS
  bill_votes_chamber
      knowledge_date_column: 'knowledge_date'
      categoricals_imputation:
        all:
          type: 'null_category'
      categoricals:
        - column: 'chamber'
          choice_query: |
            SELECT DISTINCT chamber
```

```
          FROM ml_policy_class.bills b
          LEFT JOIN ml_policy_class.sessions s
          ON b.session_id = s.session_id
          LEFT JOIN ml_policy_class.bill_votes bv
          ON b.bill_id = bv.bill_id
          WHERE s.state_id = 49
        metrics:
          - 'sum'
    intervals: ['2week', '1month', '6month', '1year', 'all']

  # finds how many years have passed between the legislative session
start and the date the bill was introduced (trying to see if odd or
even session years make a difference)
  - prefix: 'bill_timing'
    from_obj: |
      (SELECT
          b.bill_id::INT AS entity_id,
          b.introduced_date AS knowledge_date,
          EXTRACT(YEAR FROM (b.introduced_date - s.year_start)) AS
year_since_session_start
      FROM ml_policy_class.bills b
      LEFT JOIN ml_policy_class.sessions s
        ON b.session_id = s.session_id) AS bill_timing
    knowledge_date_column: 'knowledge_date'
    aggregates_imputation:
      all:
        type: 'zero'
    aggregates:
      - quantity: 'year_since_session_start'
        metrics: ['avg']
    intervals: ['all']


# finds the chamber body where the bill was introduced (House/Senate).
  - prefix: 'bill_introduced_body'
    from_obj: |
      (SELECT
          b.bill_id::INT AS entity_id,
          b.introduced_date AS knowledge_date,
          b.introduced_body
      FROM ml_policy_class.bills b) AS bill_introduced_body
    knowledge_date_column: 'knowledge_date'
    categoricals_imputation:
      all:
        type: 'null_category'
    categoricals:
      - column: 'introduced_body'
        choices: ['S', 'H']
        metrics: ['sum']
    intervals: ['all']
```

```
# finds counts for each bill type
  - prefix: 'bill_type'
    from_obj: |
      (SELECT
          b.bill_id::INT AS entity_id,
          b.introduced_date AS knowledge_date,
          b.bill_type
      FROM ml_policy_class.bills b) AS bill_type
    knowledge_date_column: 'knowledge_date'
    categoricals_imputation:
      all:
        type: 'null_category'
    categoricals:
      - column: 'bill_type'
        choice_query: |
          SELECT DISTINCT bill_type
          FROM ml_policy_class.bills
          WHERE bill_type IS NOT NULL
        metrics: ['sum']
    intervals: ['all']
# finds number of events marked as important
  - prefix: "events_important"
    from_obj: |
      (SELECT
          bill_id::INT AS entity_id,
          event_date AS knowledge_date,
          important::INT AS important_flag
      FROM ml_policy_class.bill_events) AS events_important
    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
      all: {type: "zero_noflag"}
    aggregates:
      - quantity: "important_flag"
        metrics: ["sum"]
    intervals: ['2week', '1month', '6month', '1year', 'all']
# finds whether at least one important event has occurred for a bill
  - prefix: "events_any_important"
    from_obj: |
      (SELECT
          bill_id::INT AS entity_id,
          event_date AS knowledge_date,
          CASE WHEN important = 1 THEN 1 ELSE 0 END AS any_important
      FROM ml_policy_class.bill_events) AS events_any_important
    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
      all: {type: "zero_noflag"}
    aggregates:
      - quantity: "any_important"
        metrics: ["max"]
```

```
    intervals: ['2week', '1month', '6month', '1year', 'all']
# calculates the counts of events occurring in House vs Senate
  - prefix: "events_chamber_counts"
    from_obj: |
      (SELECT
          bill_id::INT AS entity_id,
          event_date AS knowledge_date,
          chamber
      FROM ml_policy_class.bill_events) AS events_chamber_counts
    knowledge_date_column: "knowledge_date"
    categoricals_imputation:
      all: {type: "null_category"}
    categoricals:
      - column: "chamber"
        choices: ["H", "S"]
        metrics: ["sum"]
    intervals: ["all"]
# calculates the total number of co-sponsors (not primary sponsors)
  - prefix: "sponsors_co"
    from_obj: |
      (SELECT
          bill_id::INT AS entity_id,
          introduced_date AS knowledge_date,
          CASE WHEN role_name != 'Primary' THEN 1 ELSE 0 END AS
co_flag
      FROM ml_policy_class.bill_sponsors
      LEFT JOIN ml_policy_class.bills USING (bill_id)) AS sponsors_co
    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
      all: {type: "zero_noflag"}
    aggregates:
      - quantity: "co_flag"
        metrics: ["sum"]
    intervals: ["all"]
# finds count for each sponsor type in wisconsin (primary, joint,
sponsor(generic/unspecified))
  - prefix: "sponsor_types"
    from_obj: |
      (SELECT
          bill_id::INT AS entity_id,
          introduced_date AS knowledge_date,
          sponsor_type
      FROM ml_policy_class.bill_sponsors
      LEFT JOIN ml_policy_class.bills USING (bill_id)) AS
sponsor_types
    knowledge_date_column: "knowledge_date"
    categoricals_imputation:
      all: {type: "null_category"}
    categoricals:
      - column: "sponsor_type"
```

```
        choice_query: "SELECT DISTINCT sponsor_type FROM
ml_policy_class.bill_sponsors"
        metrics: ["count"]
    intervals: ["all"]
# calculates the count of distinct parties represented among a bill’s
  - prefix: "sponsors_bipartisan"
    from_obj: |
      (SELECT
          bill_id::INT AS entity_id,
          introduced_date AS knowledge_date,
          party_id
      FROM ml_policy_class.bill_sponsors
      LEFT JOIN ml_policy_class.bills USING (bill_id)) AS
sponsors_bipartisan
    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
      all: {type: "zero_noflag"}
    aggregates:
      - quantity:
          party_id: "distinct party_id"
        metrics: ["count"]
    intervals: ["all"]
# counts amendments by which chamber they originated in
  - prefix: "amendment_chambers"
    from_obj: |
      (SELECT
          bill_id::INT AS entity_id,
          amendment_date AS knowledge_date,
          chamber
      FROM ml_policy_class.bill_amendments) AS amendment_chambers
    knowledge_date_column: "knowledge_date"
    categoricals_imputation:
      all: {type: "null_category"}
    categoricals:
      - column: "chamber"
        choices: ["H", "S"]
        metrics: ["count"]
    intervals: ["all"]
# binary indicator features for whether an amendment occurred in each
chamber
  - prefix: "amendment_chamber_flags"
    from_obj: |
      (SELECT
          bill_id::INT AS entity_id,
          amendment_date AS knowledge_date,
          CASE WHEN chamber='H' THEN 1 ELSE 0 END AS amend_H,
          CASE WHEN chamber='S' THEN 1 ELSE 0 END AS amend_S
      FROM ml_policy_class.bill_amendments) AS amendment_chamber_flags
    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
```

```
      all: {type: "zero_noflag"}
    aggregates:
      - quantity: "amend_H"
        metrics: ["max"]
      - quantity: "amend_S"
        metrics: ["max"]
    intervals: ["all"]
# counts voting events by chamber
  - prefix: "vote_chamber_counts"
    from_obj: |
      (SELECT
          bill_id::INT AS entity_id,
          vote_date AS knowledge_date,
          chamber
      FROM ml_policy_class.bill_votes) AS vote_chamber_counts
    knowledge_date_column: "knowledge_date"
    categoricals_imputation:
      all: {type: "null_category"}
    categoricals:
      - column: "chamber"
        choices: ["H", "S"]
        metrics: ["sum"]
    intervals: ["all"]
# flags whether any absentee votes occurred
  - prefix: "votes_absentees"
    from_obj: |
      (SELECT
          bill_id::INT AS entity_id,
          vote_date AS knowledge_date,
          CASE WHEN absentees > 0 THEN 1 ELSE 0 END AS absentee_flag
      FROM ml_policy_class.bill_votes) AS votes_absentees
    knowledge_date_column: "knowledge_date"
    aggregates_imputation:
      all: {type: "zero_noflag"}
    aggregates:
      - quantity: "absentee_flag"
        metrics: ["sum"]
    intervals: ["all"]


#
# MODEL SCORING
# How each trained model is scored
#
# Each entry in 'testing_metric_groups' needs a list of one of the
metrics defined in
# catwalk.evaluation.ModelEvaluator.available_metrics (contributions
welcome!)
# Depending on the metric, either thresholds or parameters
#
```

```
# Parameters specify any hyperparameters needed. For most metrics,
# which are simply wrappers of sklearn functions, these
# are passed directly to sklearn.
#
# Thresholds are more specific: The list is dichotomized and only the
# top percentile or top n entities are scored as positive labels
# Want to evaluate metrics for top 15 predicted bills (top_n: [15])
#   Because we want to identify the top 15 of bills that are most
likely to pass
#   we use precision@k because it indicates how many of the top-ranked
bills that we identified actually pass
scoring:
  testing_metric_groups:
    # TODO: FILL IN YOUR TESTING PERFORMANCE METRICS HERE
    -
      metrics: [precision@, recall@]
      thresholds:
          top_n: [15]
          percentiles: [1, 2, 3, 4, 5, 6, 7, 8, 9,
              10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
              20, 21, 22, 23, 24, 25, 26, 27, 28, 29,
              30, 31, 32, 33, 34, 35, 36, 37, 38, 39,
              40, 41, 42, 43, 44, 45, 46, 47, 48, 49,
              50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
              60, 61, 62, 63, 64, 65, 66, 67, 68, 69,
              70, 71, 72, 73, 74, 75, 76, 77, 78, 79,
              80, 81, 82, 83, 84, 85, 86, 87, 88, 89,
              90, 91, 92, 93, 94, 95, 96, 97, 98, 99,
              100]
  training_metric_groups:
    # TODO: FILL IN YOUR TRAINING PERFORMANCE METRICS HERE
    -
      metrics: [precision@, recall@]
      thresholds:
          top_n: [15]
          percentiles: [1, 2, 3, 4, 5, 6, 7, 8, 9,
              10, 11, 12, 13, 14, 15, 16, 17, 18, 19,
              20, 21, 22, 23, 24, 25, 26, 27, 28, 29,
              30, 31, 32, 33, 34, 35, 36, 37, 38, 39,
              40, 41, 42, 43, 44, 45, 46, 47, 48, 49,
              50, 51, 52, 53, 54, 55, 56, 57, 58, 59,
              60, 61, 62, 63, 64, 65, 66, 67, 68, 69,
              70, 71, 72, 73, 74, 75, 76, 77, 78, 79,
              80, 81, 82, 83, 84, 85, 86, 87, 88, 89,
              90, 91, 92, 93, 94, 95, 96, 97, 98, 99,
              100]

# INDIVIDUAL IMPORTANCES
individual_importance:
  methods: [] # empty list means don't calculate individual
```

```
importances
  # methods: ['uniform']
  n_ranks: 5

# BIAS AUDIT
# Compares model behavior across groups based on the political parties
of the sponsors of the bills
# Democrat is used as the reference group for sponsor_party
# specifically to see if there is a difference between bills that are
sponsored by Democrat sponsors vs Republican sponsors
bias_audit_config:
    from_obj_table: |
        (
            SELECT
                bs.bill_id::INT AS entity_id,
                b.introduced_date AS knowledge_date,
                CASE
                    WHEN bs.party_id = 1 THEN 'Democrat'
                    WHEN bs.party_id = 2 THEN 'Republican'
                END AS sponsor_party
            FROM ml_policy_class.bill_sponsors bs
            LEFT JOIN ml_policy_class.bills b
                ON bs.bill_id = b.bill_id
            LEFT JOIN ml_policy_class.sessions s
                ON b.session_id = s.session_id
            WHERE s.state_id = 49
        ) AS demo
    knowledge_date_column: "knowledge_date"
    entity_id_column: "entity_id"
    attribute_columns:
        - "sponsor_party"
    ref_groups_method: "predefined"
    ref_groups:
        sponsor_party: "Democrat"
    thresholds:
        percentiles: [10, 15]


# MODEL GRID PRESETS
# Triage now comes with a set of predefined *recommended* grids
# named: quickstart, small, medium, large
# See the documentation for recommended uses cases for those.
#
# model_grid_preset: 'quickstart'

# Add your baselines
grid_config:
  # base rate
  'sklearn.dummy.DummyClassifier':
    strategy: ['most_frequent']
```

```
  # baselines which represent what people at the ACPA would do without
a ML model

'triage.component.catwalk.baselines.rankers.BaselineRankMultiFeature':
    rules:
      # bill text size
      - [{feature: 'bills_entity_id_all_text_size_max',
low_value_high_score: True}]

      # sponsors and parties
      - [{feature: 'sponsors_entity_id_all_number_sponsors_count',
low_value_high_score: False}]
      - [{feature: 'sponsors_entity_id_all_number_parties_count',
low_value_high_score: False}]

      # events
      - [{feature: 'events_recent_entity_id_2week_entity_id_count',
low_value_high_score: False}]
      - [{feature: 'events_recent_entity_id_1month_entity_id_count',
low_value_high_score: False}]
      - [{feature: 'events_recent_entity_id_6month_entity_id_count',
low_value_high_score: False}]
      - [{feature: 'events_recent_entity_id_1year_entity_id_count',
low_value_high_score: False}]
      - [{feature: 'events_recent_entity_id_all_entity_id_count',
low_value_high_score: False}]

# model tuning parameters
  'sklearn.ensemble.RandomForestClassifier':
      n_estimators: [5000]
      max_depth: [40, 50, 60]
      max_features: ['log2', 0.01, 0.1, 0.5]
      n_jobs: [-2]


'triage.component.catwalk.estimators.classifiers.ScaledLogisticRegress
ion':
      C: [0.05, 0.1, 0.2]
      penalty: ['l1', 'l2']
      solver: ['saga']

  'xgboost.XGBClassifier':
      booster: ['dart']
      tree_method: ["hist"]
      max_depth: [1, 2, 3]
      subsample: [0.25, 0.75, 1]
      gamma: [0, 0.1, 1.0]
      learning_rate: [0.05, 0.1, 0.2]
      n_estimators: [500]
```

```
'sklearn.tree.DecisionTreeClassifier':
    max_depth: [2, 3, 5]
    min_samples_split: [10, 20]
    min_samples_leaf: [5, 10, 15]
    criterion: ['gini']
```

REFERENCES

[1] Yano, T. (2011). *Text as Actuator: Text-Driven Response Modeling and Prediction in Politics*. Retrieved from https://www.cs.cmu.edu/ taey/pub/tae-yano-actuating-text.pdf

[2] Gulshen, K., Makow, N., & Hernandez, P. (2015). *Predicting bill passage*. CS 229 Final Project, Stanford University. Retrieved from https://cs229.stanford.edu/proj2015/242_report.pdf

[3] GOV TRACK. (n.d.). *Analysis Methodology*. Retrieved from https://www.govtrack.us/about/analysis