Machine Learning, Text Data, and Supreme Court Forecasting*

Aaron Kaufman† Peter Kraft‡ Maya Sen§

January 5, 2017

Abstract

What predicts the behavior of Justices on the U.S. Supreme Court? Previous attempts to develop predictive models of Supreme Court behavior have found success using either (1) text data taken from oral argument proceedings, or (2) quantitative legal data. In this article, we incorporate both data sets using an AdaBoost decision tree regressor, a popular approach in machine learning that is relatively underused in political science. As we show, our AdaBoosted approach substantially outperforms existing predictive models of Supreme Court outcomes which use exclusively one data source or rely on simpler modeling strategies. Substantively, this improved predictive success indicates that combining both legal information and the information revealed by the Justices themselves in the months leading to the decision provide the most information as to Justices' decision-making. We conclude the article by discussing possible applications of the AdaBoost approach within the social sciences.

Word Count: 3710

*Many thanks to Matthew Blackwell, Gary King, Brian Libgober, Chris Lucas, Luke Miratrix, and Finale Doshi-Velez for helpful conversations and valuable feedback. We also thank Oyez Project for providing Supreme Court oral argument data, and Josh Blackman, Michael Bommarito, and Dan Katz for comments during early stages of this project.

†Department of Government, Harvard University, aaronkaufman@fas.harvard.edu

‡Department of Computer Science, Harvard University, pkraft@college.harvard.edu

§John F. Kennedy School of Government, Harvard University, maya_sen@hks.harvard.edu

1 Introduction

What predicts the behavior of Justices on the U.S. Supreme Court? The Supreme Court reaches its decisions behind closed doors, but nonetheless rules on some of the most important issues in American politics today—including LGBT rights, access to health care, and religious liberties. This question also speaks to fundamental questions of what the law is and is thus of significant scholarly and philosophical interest (Martin et al., 2004; Schauer, 1998). For example, as Oliver Wendell Homes noted, "[t]he prophecies of what the courts will do in fact, and nothing more pretentious, are what I mean by the law." (Jr., 1897, pp. 460-61).

In this paper, we contribute to a growing literature predicting Supreme Court decision-making by combining advances in machine learning with theoretical contributions from the literature on judicial decisionmaking. We present an analysis using an AdaBoosted random forest, a machine learning ensemble model that is well-suited to predicting political phenomena. We combine this approach with a novel data set that includes information on the cases heard by the Court alongside information revealed by the Justices themselves during oral argument. This enables us to predict up to 75% of all Supreme Court case outcomes accurately, an improvement over existing approaches[1] (Martin and Quinn, 2002; Katz et al., 2014; Nasrallah, 2014). Our contributions are therefore (1) to expand the predictive toolkit of political scientists by incorporating techniques from machine learning, and (2) to expand the range of information used by predictive models on Supreme Court decision making. In so doing, we improve the accuracy of Supreme Court forecasting, an important exercise not just for Court watchers and members of the public trying to gain certainty over an opaque decision making process as well as for scholars of judicial decisionmaking.

[1]Predicting that the petitioner wins each case yields a 68% accuracy rate. We predict accurately as much as seven percentage points above this baseline, with other established models predicting slightly more than two percentage points above this baseline.

2 Existing Predictive Models of Supreme Court Outcomes

The most straightforward predictive algorithm for Supreme Court outcomes is well known among Court watchers: the petitioner, or the party that appealed the case to the Supreme Court, enjoys a favorable ruling approximately two-thirds of the time (Epstein et al., 2010).[2] This may be because the Supreme Court is unwilling to agree to hear a case unless some number of Justices are interested overturning the lower court ruling—in which case the logical conclusion is that the Court is more likely than not to rule in favor of the petitioner. In practice, this very simple predictive rule—one in which the petitioner wins every time—has a surprisingly high predictive accuracy of 67.98% across Supreme Court cases since 2000 (see Appendix A). This "petitioner wins" rule is the baseline to which we compare our model as well others.

Surprisingly, many well-regarded attempts at prediction are unable to significantly outperform this standard, including predictions made by human experts (Ruger et al., 2004; Martin et al., 2004). In a small comparison of 68 cases, Martin et al. recruited 83 law professors and other Court experts to predict case outcomes in their areas of expertise prior to oral argument proceedings. Among these 68 cases, the experts correctly predicted case outcomes 59.1% of the time and correctly predicted Justices' individual votes 67.9% of the time.

Statistical models can, however, occasionally surpass the "petitioner wins" baseline. The same Martin et al. study compared these expert predictions to a simple statistical model, a classification tree using only six case-level covariates.[3] That model correctly predicted 75%

[2]For our purposes, we operationalize a favorable ruling as at least a 5—4 majority in favor of one party, usually the petitioner. We note that our approach is to examine Supreme Court *outcomes* as opposed to the *votes* of individual Justices, in line with most papers in the literature and with the substantive interests of many Court watchers, who tend to focus on individual Justices only insofar as their votes are predictive of the eventual overall ruling.

[3]These were: circuit of origin, the issue area, the type of petitioner, the type of respondent, the ideological

out of 68 cases, but only 66% of the individual votes. Although the statistical model does beat the "petitioner wins" baseline, its findings are limited by the the small sample size of the study (Martin et al., 2004, p. 765) and that it examined only one natural Court with highly Justice-specific covariates, raising concerns of over-fitting (Katz et al., 2014).

Following in the steps of Martin et al., recent attempts have shown more reliable improvements over the "petitioner wins" baseline. Among these is {Marshall}+, which incorporates 95 case-level covariates into a predictive model (Katz et al., 2014) and reports a predictive accuracy of 69.7%. The algorithm operates using a variant of random forests called extremely randomized trees. These split candidate features randomly instead of along optimal thresholds, thus enjoying a decreased variance in estimates at the cost of increased bias. The second attempt is CourtCast, which uses three features derived from oral arguments transcripts: (1) the number of words uttered by each Justice when talking to the parties, (2) the sentiment of the words used, and (3) the number of times each Justice interrupts. CourtCast reports a predictive accuracy of 70%. The CourtCast model is an unweighted ensemble classifier consisting of random forests, support vector machines, and logistic regression. Ensemble methods, which generally consist of synthesizing the results from multiple orthogonal classifiers into one prediction, mitigate the costs of their constituent methods but can often reduce the benefits. Notably, they have a propensity to overfit small data sets.

3 AdaBoosted random forests and their Applicability to Social Science Questions

To further improve on these approaches, we turn to an AdaBoosted random forest (Zhu et al., 2009), which performs the best of several methods we test. Throughout this paper,

direction of the lower-court ruling, and whether the case raised a constitutional issue. Experts were free to consider any information they wished (Martin et al., 2004, p. 762).

we measure model performance through 10-fold cross-validation, which captures the model's ability to predict withheld samples of the observed data (Arlot et al., 2010).

Decision Trees in Social Science. With notable exceptions (e.g., Muchlinski et al., 2016; Green and Kern, 2012; Kastellec, 2010), tree-based models are rarely used in political science; they are standard, however, in machine learning and statistics. Tree-based models are a flexible, nonparametric (or semiparametric) class of methods designed to incorporate flexible functional forms, to avoid parametric assumptions of linear models, and to perform vigorous variable selection while avoiding potential problems of overfitting. The simplest kinds of tree-based models are single trees, which partition the sample space and then generate a predicted value to each region.

Here, we supplement random forests with boosting. Boosting involves creating trees *sequentially*, with each subsequent tree grown on re-weighted training data. As Montgomery and Olivella (2016) explain, each new tree then "improves upon the predictive power of the existing ensemble." For our boosting algorithm, we use AdaBoost (Freund and Schapire, 1997). The base classifier relies on "weak learners," or decision rubrics that perform only slightly better than chance: leveraging many weak learner classifiers is often better than averaging only a few stronger classifiers. AdaBoosting operates by applying this algorithm to a training set, giving each observation equal weight. In the next iteration, AdaBoost will assign more weight to those units which were incorrectly classified in the previous iteration; that is, those units that are misclassified in one round will have a higher probability of being selected as part of the training set in the next round. The algorithm specifically focuses on those units that are hard to classify—an approach that is particularly well suited for social science problems, which may frequently involve outliers.

AdaBoosting has good asymptotic properties in improving predictive accuracy, especially when there are many features that each only contribute a small predictive advantage. Indeed,

looking at the Supreme Court, a quirk of predicting its rulings is that, although baseline accuracy tends to be high, the predictive capacity of any one variable is quite small, leaving relatively little room for improvements. This is a situation with many parallels in the social sciences. In comparative politics, for example, predicting the advent of civil wars has a very high baseline accuracy since there are very few wars, but each additional predictor adds relatively little information (**?**). In American politics, to use another example, changes in which party controls the U.S. Presidency are often summarized by the "bread and peace" model: the incumbent party wins when the economy is growing, except during unpopular wars (**?**). This produces a remarkably high baseline accuracy, on top of which other variables (such as campaign effects, candidate effects, or demographic changes) add seemingly little (**?**). AdaBoosting is well suited for these problems. For additional description of the costs and benefits of Adaboosted random forests, see Appendix A.

Application of AdaBoosting to the Supreme Court. We first find the maximally predictive single decision tree using methods as implemented in the Python library `scikit-learn` (Pedregosa et al., 2011). (An example of such a tree is provided in Figure 1.) We then use that single tree to generate predictions for every Supreme Court case in our sample, indicating the probability that the petitioner or the respondent will win. Next, we re-weight our data set proportionally to the size of each observation's fitted residual according to the predictions of that single tree. In a case where that decision tree predicts that the petitioner will win with 84% probability and the petitioner did win, that observation's weight is proportional to 16%. If instead that case's outcome was that the respondent won, the weight would be proportional to 84%. Then we use that weighted data set to generate the next decision tree, calculate that single tree's fitted residuals, and re-weight. We repeat this procedure for 10,000 iterations, storing each intermediate model. To predict the results of new cases outside of our training set, we generate predicted values using each of the 10,000

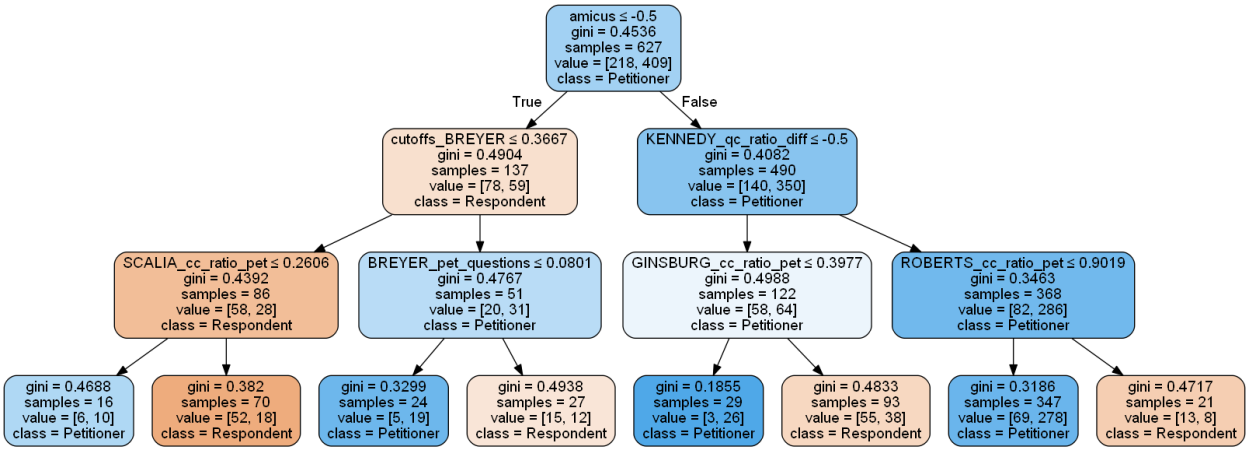

Figure 1: Example of a decision tree trained to predict U.S. Supreme Court outcomes. Each box represents a feature split, indicated by the first text row in each box. (These feature splits are described in Appendix B.) Bluer nodes are ones in which the petitioner is predicted to win; the more orange the box, the higher probability the respondent wins. Gini, short for Gini impurity, indicates the probability that a randomly chosen observation would be incorrectly classified at that node.

intermediate models and average all 10,000 predictions together.

4 Supreme Court Data

Both {Marshall}+ and CourtCast enjoy predictive gains over the "petitioner wins" baseline, though both draw on different data sources. {Marshall}+ draws on case covariates, while CourtCast relies on oral argument transcripts. This suggests that both data sets contain variable predictive information.

We train our AdaBoosted model using both types of data. The first data source is case-level covariates, which come from the Supreme Court Database (Spaeth et al., 2015). These data include a case-specific variables capturing the procedural posture of the case, the issues involved, the identities of the parties, etc. The second data source is statements made by the Justices during oral arguments. Existing scholarship suggests that oral arguments represent an important opportunity for the Justices to gather information from attorneys

and stake out potential positions (Johnson et al., 2006). We draw on textual data from the Supreme Court's oral argument transcripts, which we collect from the Oyez Project (Goldman, 2002). For each Justice, we compute the following features: (1) questions asked to the petitioner, (2) questions asked to the respondent, (3) words spoken to the petitioner, (4) words spoken to the respondent, (5) interruptions of the petitioner, and (6) interruptions of the respondent.[4] We transform the raw oral arguments data in two ways. First, we create dichotomous indicators for each Justice indicating if that Justice asked more questions, spoke more words, or interrupted more frequently the petitioner or the respondent attorney (27 total variables). Second, we calculate for each Justice the appropriate ratios of speech targeted toward each attorney for words spoken, questions asked, and interruptions.[5] We find that, generally, the most predictive oral argument-derived features are ratios.

5 Results and Comparisons to Other Approaches

Below, we present predictions based on our model, {Marshall}+, CourtCast, a naive random forest, and the "petitioner always wins" rule, which we take as our baseline. We evaluate all models using an identical ten-fold cross-validation protocol (Efron and Tibshirani, 1997). For a data set with n observations, we first partition the data into 10 subsets of size $\frac{n}{10}$. This algorithm first trains a model on partitions 2 through 10, then predicts the outcome measure for the first subset and records the number of correct predictions. Next, a model is trained on subsets 3 through 10 and 1, and then a prediction is generated for subset 2, recording its accuracy. This is repeated for all 10 subsets. The total percentage of correct predictions is treated as the model's out-of-sample predictive accuracy. For a longer discussion of K-fold cross-validation, see Appendix E.

[4]For consistency in comparisons, we compute these measures identically to the CourtCast model.

[5]For example, for interruptions, we calculate for each Justice the ratio of times the Justice interrupted the liberal litigator versus the conservative litigator. If Scalia interrupted the liberal litigator six times but only interrupted the conservative litigator two times, this value would be $(6/8)/(2/8) = 3$

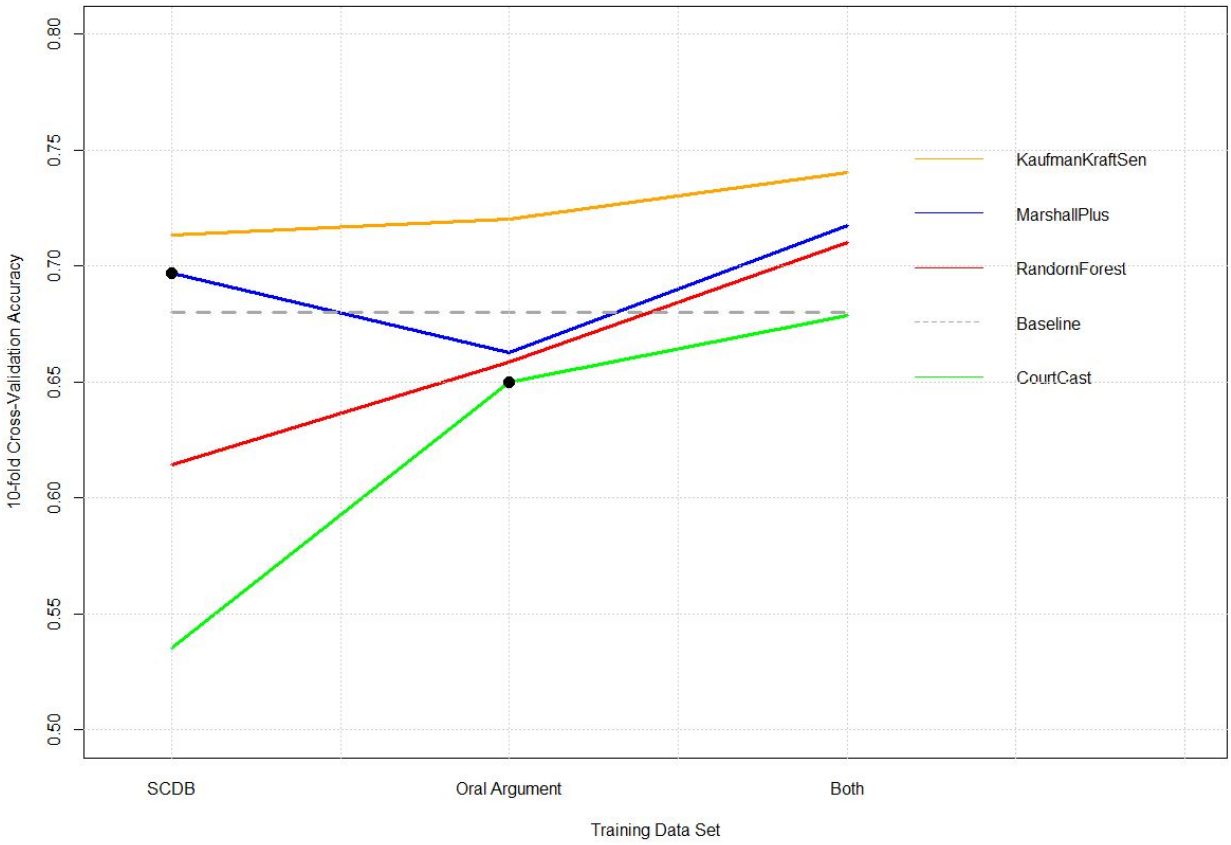

Figure 2: Cross-Validation Accuracy for (1) KKS, (2) {Marshall}+, (3) a naive random forest, (4) the "petitioner always wins" baseline, and (5) CourtCast, across three different training sets. For {Marshall}+ and CourtCast, black dots indicate the original data set on which those models were trained. Regardless of training data set, KKS outperforms all previous models.

In Table 1 and Figure 2, we compare results from our model ("KKS") to the others. For each model, we indicate the data set used, the cross-validation accuracy, and the improvement above the baseline accuracy. Models using only oral argument data slightly outperform models using only case-level covariates from the Supreme Court Database, but the KKS model incorporating *both* oral arguments transcripts data and case-level covariates substantially outperforms the rest. Note that CourtCast's cross-validation accuracy using its original data set is 64.99%, substantially lower than their self-reported accuracy

Model	Data	Accuracy	Accuracy - Baseline
Baseline	None	67.98%	0%
{Marshall}+	SCDB	69.70%	1.72%
CourtCast	oral argument	70.00%	2.02%
KKS	SCDB	71.34%	3.36%
KKS	oral argument	72.02%	4.04%
KKS	Both	74.04%	6.06%

Table 1: Accuracy for (1) the "petitioner always wins" baseline, (2) {Marshall}+, (3) CourtCast, and (4) KKS. Data refers to the case-level covariates from the Supreme Court Database ("SCDB"), transcript data from the oral arguments ("oral argument"), or both. KKS model using the full covariate set triples the added accuracy of the next best model. The least predictive KKS model enjoys a 50 percentage point increase in added accuracy over the next best model.

as calculated using a single train-test split. While this does not suggest over-reporting by CourtCast, we prefer ten-fold cross-validation over a single train-test split as a more robust measure of model accuracy (Arlot et al., 2010). Both {Marshall}+ and CourtCast perform best using the joint data set; both perform second-best on the single data set on which they were originally designed.

Among models using only case-level covariates from the Supreme Court Database, the KKS model reaches predictive accuracy of 71.34%, compared to the accuracy {Marshall}+, which is 69.7%. While {Marshall}+ beats the baseline by 1.72 percentage points, the KKS model using the same data surpasses baseline accuracy by 3.34 percentage points, almost double the added predictive value. Similarly, among the models using only oral argument data, the KKS model reaches predictive accuracy of 72.02%, compared to the CourtCast accuracy of 70.0%. CourtCast beats the baseline by 2.02 percentage points, the comparable KKS model surpasses baseline accuracy by 4.04 percentage points, exactly double the added predictive value. When the KKS model is trained on both data sets, its accuracy increases to 74.04%, or 6.06 percentage points above the baseline, a three-fold increase over the best current model. Since we calculate these accuracy statistics using 10-fold cross-validation, they include all Supreme Court cases from 2005 to 2015.

Margin	Accuracy	Accuracy - Baseline
5-4	66%	0%
6-3	74%	8%
7-2	75%	9%
8-1	82%	16%
9-0	77%	11%

Table 2: KKS model accuracy by decision margin.

There is notable heterogeneity in our prediction accuracy (Table 2). Narrow 5-4 decisions are more difficult to predict, and wider margins are easier to predict. Our accuracy for 5-4 cases is at the baseline, at 66%. We predict 74% of 6-3 cases correctly, 75% of 7-2 cases correctly, 82% of 8-1 cases correctly, and 77% of 9-0 cases correctly.

6 Discussion and Conclusion

In this paper, we have made two specific contributions. First, we have contributed to, and improved on, the literature on Supreme Court prediction. The Supreme Court is the most reclusive of the three branches of the federal government; at the same time, the Court adjudicates some of the most important and contentious policy issues of the day, including important rulings on health care, campaign finance reform, and affirmative action. Increasing the predictive accuracy of forecasting models not only allows scholars to understand how this important branch of government operates, but also, we believe, allows researchers to more credibly assess which way these significant policy rulings will go.

Second, we have provided an overview of the AdaBoost regression forest, a technique that, although frequently used in machine learning, is novel within the social sciences. This approach is particularly appropriate for many social science questions, not just Supreme Court forecasting, owing to its robustness to small sample sizes and its careful treatment of very weakly predictive covariates. Such problems may include predicting civil wars, predicting when strong incumbents may be successfully challenged, and predicting public opinion

in the face of floor or ceiling effects. An additional uses may be in calculating propensity scores for matching in the presence of many covariates.

7 References

Arlot, S., Celisse, A., et al. (2010). A Survey of Cross-Validation Procedures for Model Selection. *Statistics Surveys*, 4:40–79.

Efron, B. and Tibshirani, R. (1997). Improvements on Cross-Validation: The 632+ Bootstrap Method. *Journal of the American Statistical Association*, 92(438):548–560.

Epstein, L., Landes, W. M., and Posner, R. A. (2010). Inferring the Winning Party in the Supreme Court from the Pattern of Questioning at Oral Argument. *The Journal of Legal Studies*, 39(2):433–467.

Freund, Y. and Schapire, R. E. (1997). A Decision-Theoretic Generalization of On-line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, 55(1):119–139.

Goldman, J. (2002). The OYEZ Project [On-line].

Green, D. P. and Kern, H. L. (2012). Modeling Heterogeneous Treatment Effects in Survey Experiments with Bayesian Additive Regression Trees. *Public Opinion Quarterly*, 76(3):491–511.

Johnson, T. R., Wahlbeck, P. J., and Spriggs, J. F. (2006). The Influence of Oral Arguments on the US Supreme Court. *American Political Science Review*, 100(01):99–113.

Jr., O. W. H. (1897). The Path of the Law. *Harvard Law Review*, 10:457.

Kastellec, J. P. (2010). The Statistical Analysis of Judicial Decisions and Legal Rules with Classification Trees. *Journal of Empirical Legal Studies*, 7(2):202–230.

Katz, D. M., Bommarito, M. J., and Blackman, J. (2014). Predicting the Behavior of the Supreme Court of the United States: A General Approach. *Available at SSRN 2463244*.

Martin, A. D. and Quinn, K. M. (2002). Dynamic Ideal Point Estimation Via Markov Chain Monte Carlo for the US Supreme Court, 1953—1999. *Political Analysis*, 10(2):134–153.

Martin, A. D., Quinn, K. M., Ruger, T. W., and Kim, P. T. (2004). Competing Approaches to Predicting Supreme Court Decision Making. *Perspectives on Politics*, 2(04):761–767.

Montgomery, J. M. and Olivella, S. (2016). Tree-based Models for Political Science Data. *American Journal of Political Science*.

Muchlinski, D., Siroky, D., He, J., and Kocher, M. (2016). Comparing Random Forest with Logistic Regression for Predicting Class-Imbalanced Civil War Onset Data. *Political Analysis*, 24(1):87–103.

Nasrallah, C. (2014). Courtcast.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Ruger, T. W., Kim, P. T., Martin, A. D., and Quinn, K. M. (2004). The Supreme Court Forecasting Project: Legal and Political Science Approaches to Predicting Supreme Court Decisionmaking. *Columbia Law Review*, pages 1150–1210.

Schauer, F. (1998). Prediction and Particularity. *Boston University Law Review*, 78:773.

Spaeth, H. J., Epstein, L., Martin, A. D., Segal, J. A., Ruger, T. J., and Benesh, S. C. (2015). *The Supreme Court Database*. Center for Empirical Research in the Law at Washington University.

Zhu, J., Zou, H., Rosset, S., and Hastie, T. (2009). Multi-Class Adaboost. *Statistics and its Interface*, 2(3):349–360.