Improving Supreme Court Forecasting Using Boosted Decision Trees

Aaron Russell Kaufman
PhD Candidate
Department of Government, Harvard University
1737 Cambridge Street, Cambridge, MA 02138, USA
(818) 263–5583
aaronkaufman@fas.harvard.edu

Peter Kraft
AB Candidate
Department of Computer Science, Harvard University
33 Oxford Street, Cambridge, MA 02138, USA
pkraft@college.harvard.edu

Maya Sen
Assistant Professor
John F. Kennedy School of Government, Harvard University
79 John F. Kennedy Street, Cambridge, MA 02138, USA
maya_sen@hks.harvard.edu

Many thanks to Matthew Blackwell, Finale Doshi-Velez, Gary King, Brian Libgober, Chris Lucas, Luke Miratrix, and Kevin Quinn for helpful conversations and valuable feedback. We also thank Josh Blackman, Michael Bommarito, and Dan Katz for comments during early stages of this project.

Abstract

Though used frequently in machine learning, AdaBoosted decision trees (ADTs) are rarely used in political science, despite having many properties that are useful for social science inquiries. In this paper, we explain how to use ADTs for social science predictions. We illustrate their use by examining a well-known political prediction problem, predicting U.S. Supreme Court rulings. We find that our AdaBoosted approach outperforms existing predictive models. We also provide two additional examples of the approach, one predicting the onset of civil wars and the other predicting county-level vote shares in U.S Presidential elections.

1 Introduction

What predicts U.S. Supreme Court rulings? How do we predict whether a country will suffer a civil war? How can we forecast U.S. Presidential election outcomes at the local level? In this paper, we introduce one tool that, though underused in political science, offers attractive properties for studying social science prediction problems: AdaBoosted decision trees (ADTs). Critically for political science data, ADTs capture gains in prediction when there are many variables, most of which add only limited predictive value.

We illustrate their usefulness by examining a topic of intense political speculation—predicting U.S. Supreme Court rulings. Specifically, we use ADTs on a novel dataset that includes case-level information alongside textual data from oral arguments. This approach enables us to predict up to 75% of all case outcomes accurately, with an even higher accuracy among politically important cases. Substantively, this means we are able to predict approximately up to seven more cases (out of around 80) accurately per year year compared to the baseline of guessing that the petitioner will always win, which yields 68% accuracy. As further evidence of the utility of ADTs, we provide two additional examples: (1) predicting whether civil war occurs in a country in a given year (which we predict with 99.0% accuracy) and (2) predicting county-level U.S. Presidential Election outcomes (which we predict with 96.7% accuracy, using 2016 as our example).

2 AdaBoosted Decision Trees and their Applicability to Social Science Questions

With notable exceptions (e.g., Montgomery and Olivella, 2016; Muchlinski et al., 2016; Green and Kern, 2012), tree-based models are rarely used in political science, which tends to focus on substantive interpretation of covariates, ideally with causal implications.[1] Tree-based models—which are designed to incorporate flexible functional forms, avoid parametric assumptions, perform vigorous variable selection, and avoid overfitting—are common, however, in machine learning and statistics.

The simplest kinds of tree-based models partition the data into "leaves" and predict the value of each leaf. For example, a decision tree predicting Supreme Court rulings might start by splitting a set of cases by whether the government is the respondent. If so, the algorithm may predict that the government wins. If not, the algorithm may examine the provenance of the case, and, if it is the result of a circuit split, predict that the petitioner wins. If it is not a circuit split, then it may examine whether Anthony Kennedy spoke frequently at oral argument. If he did, the algorithm may predict that the respondent wins and, if otherwise, that the respondent loses.

We supplement decision trees with boosting. Boosting creates trees *sequentially*, and, as Montgomery and Olivella (2016) explain, each new tree then "improves upon the predictive power of the existing ensemble." The base classifier relies on "weak learners," decision rubrics that perform only slightly better than chance. AdaBoosting initializes by giving each observation equal weight. In the second iteration, AdaBoost will assign more weight to those units that were incorrectly classified previously. Focusing on those units that are hard to classify makes this approach well-suited to social science problems, many of which involve heterogeneity and outliers.[2]

AdaBoost has several properties that make it attractive for social science research. First, it has desirable asymptotic properties in improving predictive accuracy, especially when there are many features that each only contribute a small predictive gain. Examining the Supreme Court, a quirk of predicting its rulings is that although baseline accuracy is high,

[1] See Appendix E for additional discussion of why machine learning may be underused in political science.

[2] For a more technical walk-through of the AdaBoosting algorithm, see Appendix G.

the predictive capacity of any one variable is small, leaving little room for improvement. This situation is common in the social sciences. For example, predicting the advent of civil wars has high baseline accuracy since there are very few wars, but each additional predictor adds relatively little information (Ward et al., 2010). Further, 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 (Hibbs Jr, 2000). This produces a remarkably high baseline accuracy, on top of which other variables (such as campaign effects) add little (Gelman and King, 1993). Second, unlike many machine learning tools, AdaBoost has few parameters, allowing researchers to apply this method out-of-the-box. Third, AdaBoost provides a critical theoretical guarantee: for any given iteration, as long as that model's predictions are finitely better than random chance, the overall model's training error is guaranteed to decrease (Chen et al., 2012).[3] Finally, AdaBoost is agnostic to predictor or outcome data types, be they binary, continuous, or categorical (Elith et al., 2008), simplifying its implementation in dealing with mixed data sets of many predictors.

We also note some drawbacks. Like all non-parametric models, ADTs sacrifice some interpretability of estimates for flexibility of functional form. By avoiding assumptions about the relationship between Supreme Court rulings and covariates, our model provides more robust predictive capacity. However, it obfuscates discussions of statistical significance or effect sizes; rather than interpreting coefficients on covariates, ADTs rely on "feature importance" (See Section 11). Second, it is computationally expensive. ADTs scale linearly in the number of boosting iterations, but polynomially in the number of covariates and exponentially in the interaction depth of features; unfortunately, due to the sequential nature of AdaBoost, it is not amenable to parallelization. Lastly, there exist important problems for which AdaBoost fails. With small sample sizes, unpredictive covariates, or unsuitable base

[3] Train error refers to in-sample model fit, while test error refers to out-of-sample predictive accuracy. In this case, we measure predictive accuracy using exponential loss.

models, AdaBoost will show no improvement over more naive methods and may actually perform worse. Despite this, AdaBoost has been shown to work well in a wide variety of experimental settings among benchmark problems in computer science (Freund and Schapire, 1996).

3 Application of AdaBoosting to the Supreme Court

We illustrate the use of ADTs by predicting rulings by the U.S. Supreme Court. Because the Court decides cases of such magnitude—including cases on Presidential power, states' rights, and international law—even small predictive gains translate into significant policy importance. The simplest predictive algorithm for Supreme Court rulings is that the petitioner (the party appealing the case), wins roughly two thirds of the time (Epstein et al., 2010; Epstein and Jacobi, 2010).[4] In practice, guessing that the petitioner wins every time predicts 67.98% of cases since 2000 accurately (Appendix A1), though several studies have surpassed this baseline (Martin et al., 2004; Katz et al., 2014; Roeder, 2015; Katz et al., 2017). In this paper, we compare our approach to two prominent Supreme Court forecasting models, {Marshall}+ and CourtCast.[5]

To build on these approaches, we implement ADTs using the Python library `scikit-learn` for 10,000 iterations. We train our model (and comparison models) using two Supreme Court data sources from 2005-2015. First, we use case-level covariates from the Supreme Court Database (Spaeth et al., 2015). These include the procedural posture of the case, the issues involved, the identities of the parties, and other case-level factors, detailed in Appendix C.[6]

[4] A favorable ruling is at least a 5–4 majority. We note that our approach is to examine Court *outcomes* as opposed to the *votes* of individual Justices, in line with most papers in the literature.

[5] Source code for CourtCast is at `https://github.com/nasrallah/CourtCast`. Appendix H discusses both in more detail.

[6] Some of these variables are subjectively coded after the ruling is issued (for example, issue area). However, we see no way in which the coding would change pre- and post-decision. Appendix C discusses this in further detail.

Model	Data	Accuracy	Accuracy - Baseline (percentage points)
Baseline	None	67.98%	0
Katz 2017	SCDB	70.20%	2.22
{Marshall}+	SCDB	70.20%	2.22
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) Katz et al. 2017, (3) {Marshall}+, (4) CourtCast, and (5) KKS. "Data" indicates the case-level covariates from the Supreme Court Database ("SCDB"), transcript data from the oral arguments ("oral argument"), or both. The KKS model using the full covariate set triples the added accuracy of the next best model. The least predictive KKS model enjoys a 50% increase in added accuracy over the next best model.

Second, we draw on statements made by the Justices during oral arguments. Scholarship suggests that oral arguments represent an important opportunity for the Justices to gather information and stake out potential positions (Johnson et al., 2006). We draw on textual data from the Court's oral argument transcripts provided by the Oyez Project (Goldman, 2002), which we operationalize into 55 variables, detailed in Appendix C.

4 Results and Comparisons to Other Approaches

We compare predictions based on our model, {Marshall}+, CourtCast, a naive random forest, and the "petitioner always wins" rule. We evaluate all models using ten-fold cross-validation (Efron and Tibshirani, 1997) (see Appendix D), which captures a model's ability to predict withheld samples of the observed data.

In Table 1 and Figure 1, we compare results from our model ("KKS") to the others. For each, we indicate the dataset used, the cross-validation accuracy, and the improvement above the baseline. Models using only oral argument data slightly outperform models using only case-level covariates, but the KKS model incorporating both oral arguments data and

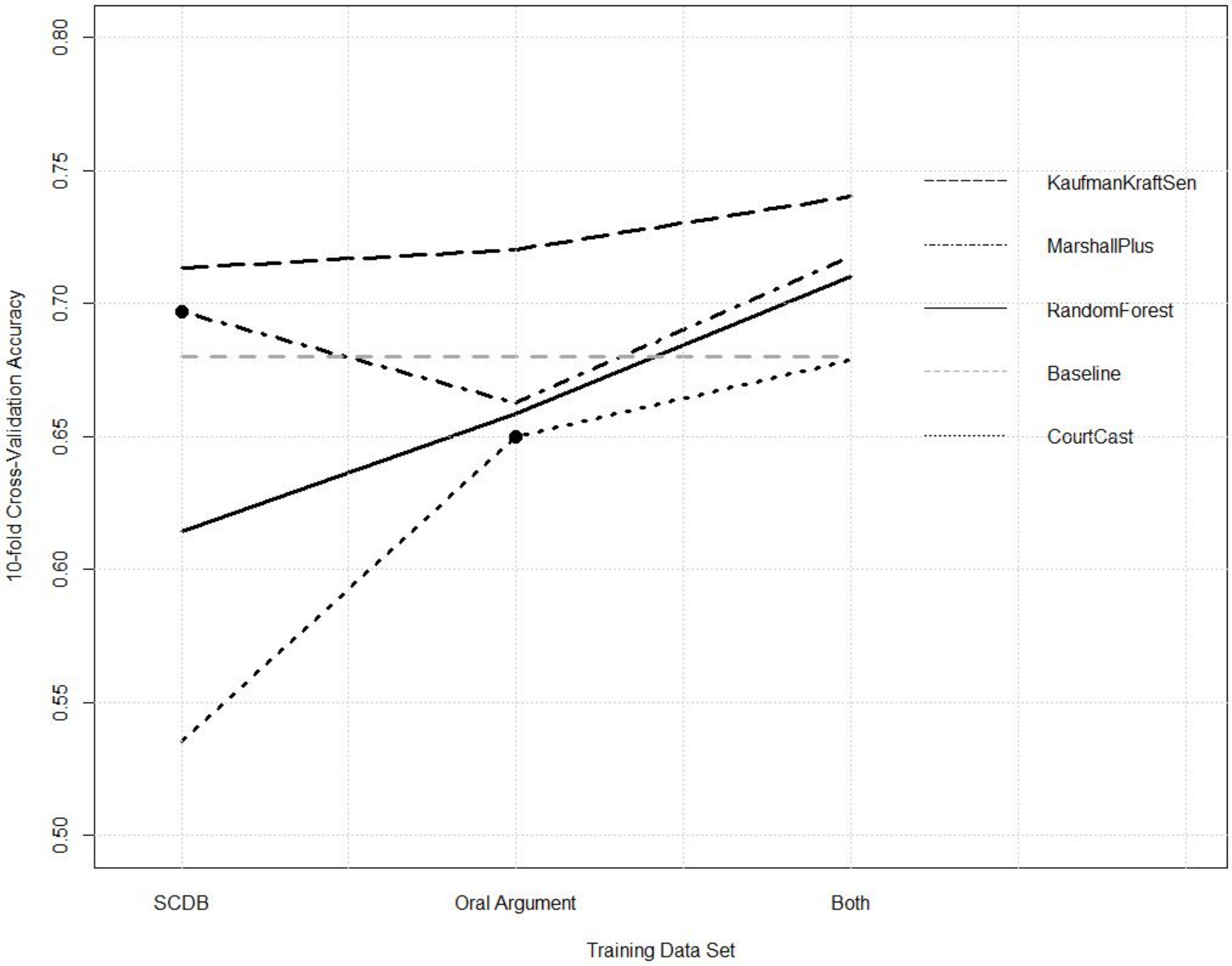

Figure 1: Cross-Validation Accuracy for (1) KKS, (2) {Marshall}+, (3) a naive random forest, (4) the "petitioner always wins" baseline, and (5) CourtCast, across three different training sets. For {Marshall}+ and CourtCast, black dots indicate the original dataset on which those models were trained.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

case-level covariates substantially outperforms the rest. Both {Marshall}+ and CourtCast perform best using the joint dataset; both perform second-best on the dataset for which they were designed.

Among models using only case covariates, the KKS model achieves predictive accuracy of 71.34%, compared to the accuracy of {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%. While CourtCast beats the baseline by 2.02 percentage points, the comparable KKS model surpasses baseline accuracy by 4.04 percentage points. When the KKS model is trained on both datasets, its accuracy increases to 74.04%, 6.06 percentage points above the baseline, a three-fold increase over the best current model. Substantively, this means we are able to predict about seven more cases (out of 80) per term than baseline—a meaningful improvement. Since we calculate these accuracy statistics using 10-fold cross-validation, they include all cases from 2005 to 2015.

5 Predictive Accuracy Conditional on Covariates

Our model enjoys a six percentage point gain over baseline on average, but this increases when we examine subsets of cases. Close 5-4 decisions go to the petitioner 61% of the time on average, and our accuracy for 5-4 cases is 66%, five percentage points above that baseline. We correctly predict 74% of 6-3 cases, 75% of 7-2 cases, 82% of 8-1 cases, and 77% of 9-0 cases; our model provides the biggest accuracy boost, 14 percentage points, for 6-3 decisions.

Our model strongly outperforms baseline in cases related to judicial power (9 points) and federalism (16 points) and in cases where either a state or the federal government is a party (9 points). We see weaker gains in criminal procedure, civil rights, and First Amendment cases (Table 2).

Margin	Baseline	Accuracy	Accuracy - Baseline (Percentage Points)
Margin: 5-4	61%	66%	5
Margin: 6-3	60%	74%	14
Margin: 7-2	69%	75%	6
Margin: 8-1	72%	82%	10
Margin: 9-0	69%	77%	8
Issue: Criminal Procedure	72%	75%	3
Issue: Civil Rights	75%	79%	4
Issue: First Amendment	71%	74%	3
Issue: Economic Activity	65%	73%	8
Issue: Judicial Power	66%	75%	9
Issue: Federalism	53%	69%	16
Government is Party	71%	80%	9
Government is not Party	68%	74%	6

Table 2: KKS model accuracy by decision margin.

5.1 Additional Applications: County-Level U.S. Presidential Vote Share & Civil Wars

ADTs are promising for other political science applications and may outperform even other tree-based methods. To demonstrate this, we compare the accuracy of predictions generated by ADTs to those other methods across two applications. The first is predicting 2016 presidential Democratic vote shares at the county-level and the second is predicting whether a country is suffering from civil war using data from Ward et al. (2010).

For U.S. presidential elections, we perform 10-fold cross-validation on a dataset from the 2010 U.S. Census that includes county-level age, income, education, and gender distributions. The county-level outcome variable indicates whether the Democratic Party's two-party vote share in the 2016 presidential election is greater than 50%. The baseline is calculated by guessing that all counties voted for the Republican candidate. To assess accuracy, we calculate using 10-fold cross-validation the proportion of counties correctly predicted.

In predicting civil wars, we examine a data set indicating which country-years were engaged in civil wars as a function of country-level covariates derived from Collier & Hoeffler

Method	Elections Accuracy	Civil Wars Accuracy
ADTs	0.967	0.990
Random Forest	0.957	0.989
Support Vector Machines	0.954	0.983
Extremely Random Trees	0.948	0.990
LASSO	0.948	0.862
Logistic Regression	0.944	0.987
Baseline	0.876	0.861

Table 3: ADTs outperform other methods,including tree-based methods, in predicting county-level vote share in the 2016 US Presidential Election, as well as civil war incidence, as measured by 10-fold cross-validation.

2002 and Fearon & Laitin 2003, including population, GDP, Polity score, ethnolinguistic fractionalization, and oil reserves. The baseline accuracy is 86.1%, achieved by predicting "no civil war" in all cases. To assess accuracy, we calculate using 10-fold cross-validation the proportion of country-years correctly predicted as either having a civil war or not.

Table 3 presents the results of these analyses. As it makes clear, ADTs perform well, outperforming other linear, nonlinear, and tree-based methods. These improvements are substantively meaningful. Civil wars are devastating; being able to predict them accurately holds great promise for the allocation of scarce peacekeeping resources. In our data set of 6,610 country-years since 1945, our model predicts 853 more cases than the baseline, corresponding to 11.8 additional countries accurately predicted each year. Similarly, as the example of 2016 has shown, presidential elections are consequential and hard to predict. In our data set of 3,082 counties, being able to predict how 31 more counties will likely vote may have a large impact on how campaigns choose to distribute their resources.

6 Discussion and Conclusion

Our contributions are twofold. Fist, we have provided an overview of ADTs, a technique frequently used in machine learning but that is more novel within the social sciences. The approach is promising for many social science questions, owing to its robustness to small sample sizes and its treatment of weakly predictive covariates. As our examples show, this approach performs favorably compared to other commonly used methods across several substantive political science applications.

Second, we have contributed to a growing literature on Supreme Court prediction. The Court is the most reclusive branch of the U.S. government, yet it rules on some of the most important and contentious policy issues of the day. 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 assess more credibly which way these influential rulings may go.

7 References

Archer, K. J. and Kimes, R. V. (2008). Empirical Characterization of Random Forest Variable Importance Measures. *Computational Statistics & Data Analysis*, 52(4):2249–2260.

Chen, S.-T., Lin, H.-T., and Lu, C.-J. (2012). An Online Boosting Algorithm with Theoretical Justifications. *arXiv preprint arXiv:1206.6422*.

Collier, P. and Hoeffler, A. (2002). On the incidence of civil war in africa. *Journal of conflict resolution*, 46(1):13–28.

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

Elith, J., Leathwick, J. R., and Hastie, T. (2008). A working guide to boosted regression trees. *Journal of Animal Ecology*, 77(4):802–813.

Epstein, L. and Jacobi, T. (2010). The strategic analysis of judicial decisions. *Annual Review of Law and Social Science*, 6:341–358.

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.

Fearon, J. D. and Laitin, D. D. (2003). Ethnicity, insurgency, and civil war. *American political science review*, 97(01):75–90.

Freund, Y. and Schapire, R. E. (1996). Experiments with a New Boosting Algorithm. In *Proceedings of the Thirteenth International Conference on Machine Learning*, volume 96, pages 148–156.

Gelman, A. and King, G. (1993). Why Are American Presidential Election Campaign Polls so Variable When Votes Are so Predictable? *British Journal of Political Science*, 23(4):pp. 409–451.

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.

Hibbs Jr, D. A. (2000). Bread and Peace Voting in US Presidential Elections. *Public Choice*, 104(1-2):149–180.

Ho, T. K. (2002). A Data Complexity Analysis of Comparative Advantages of Decision Forest Constructors. *Pattern Analysis & Applications*, 5(2):102–112.

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.

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*.

Katz, D. M., Bommarito II, M. J., and Blackman, J. (2017). A general approach for predicting the behavior of the supreme court of the united states. *PloS one*, 12(4):e0174698.

Liaw, A. and Wiener, M. (2002). Classification and Regression by randomForest. *R News*, 2(3):18–22.

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.

Roeder, O. (2015). How to read the mind of a supreme court justice. https://fivethirtyeight.com/features/how-to-read-the-mind-of-a-supreme-court-justice/.

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.

Ward, M. D., Greenhill, B. D., and Bakke, K. M. (2010). The perils of policy by p-value: Predicting civil conflicts. *Journal of Peace Research*, 47(4):363–375.

8 Appendix A1: Petitioner-Wins Baseline Accuracy

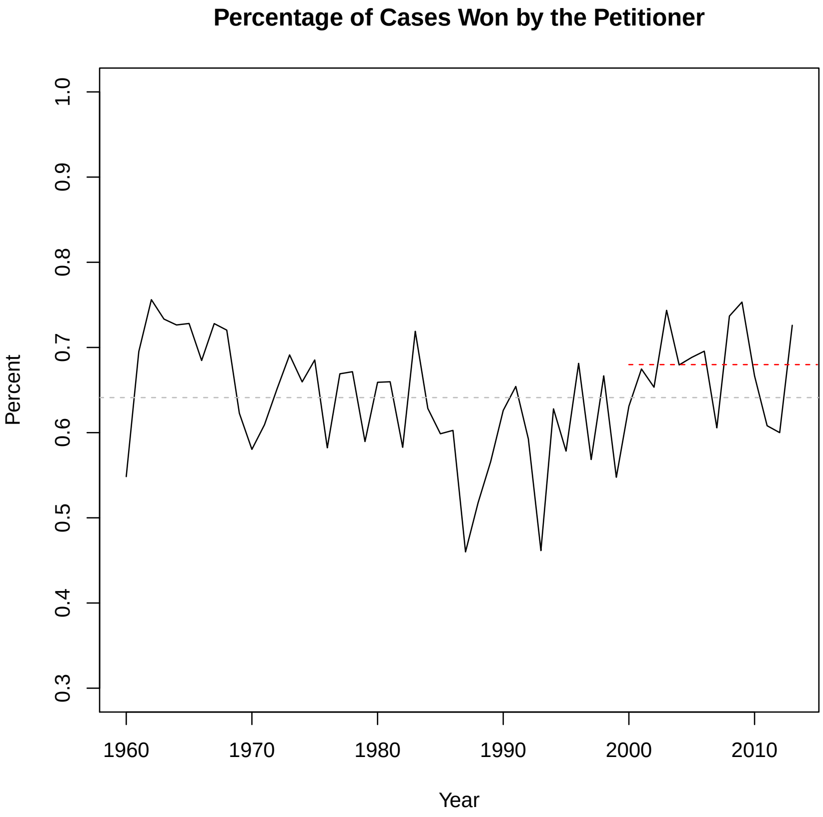

Figure 2: Percentage of Supreme Court cases won by the petitioner. This has averaged 64% since 1960 (gray dashed line) and 68% since 2000 (red dashed line).

9 Appendix A2: Accuracy over time

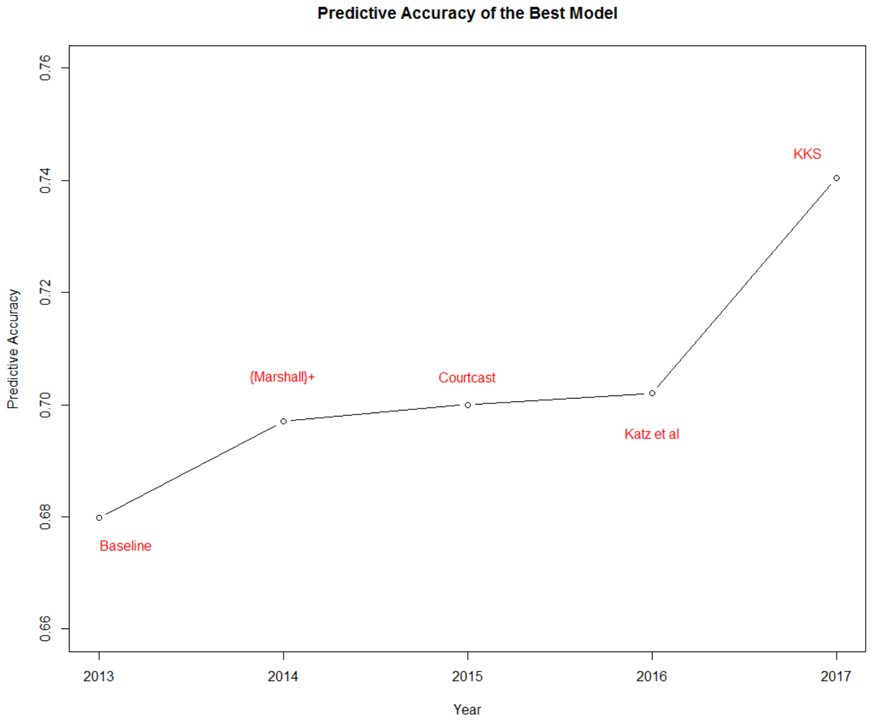

Figure 3: The percentage accuracy of the most accurate Supreme Court prediction model for each year from 2013 to the present.

10 Appendix B: Additional Decision Trees

Below is one decision tree drawn from our KKS AdaBoosted Decision Trees model. Each box represents a feature split, indicated by the first text row in each box. This tree begins with the "lawyers" feature, indicating the relative number of lawyers arguing for the plaintiff and the respondent, split on the value -0.5. The box is very blue, indicating that when that condition holds, it is highly probable that the respondent wins the case. The second row of text, Gini impurity, indicates the probability of incorrect classification based on that node. For the "lawyers" box, this means that classifying court decisions solely on whether "lawyers" ≥ -0.5 would incorrectly classify cases 45.4% of the time. The second row of boxes is the next layer of feature splits. If the condition in the "lawyers" box holds, the tree moves to the left node; otherwise, it moves to the right node. These trees are all three layers deep, though it is possible to construct decision trees with more or fewer layers. The end points of each decision tree are indicated in the bottom rows. For example, in the first tree, if for a certain case the conditions in the "lawyers" box, "KENNEDY_pet_questions" box, and "SCALIA_cc_ratio_res" box are all true, then the prediction for that case is that the Respondent will win, and empirically, the Respondent wins more than 65% of those cases, as indicated by the Gini impurity value.

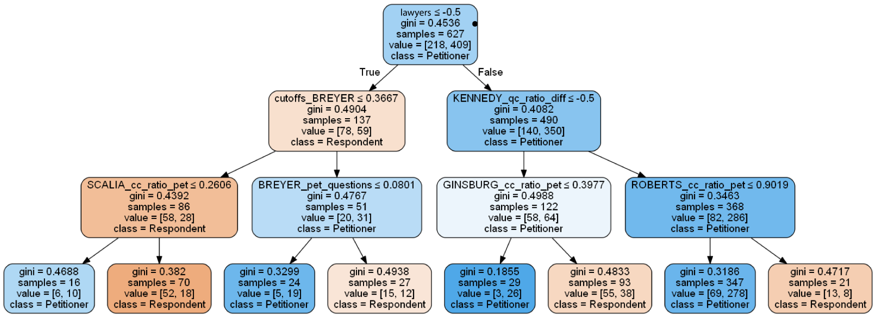

Figure 4: An example decision tree.

11 Appendix C: Variable Selection & Feature List

For each Justice, we compute the following features: questions asked to the petitioner/respondent, words spoken to the petitioner/respondent, interruptions of the petitioner/respondent.[7] We transform these 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.[8] We find that, generally, the most predictive oral argument-derived features are ratios.

Since ADTs are largely black boxes where features enter and predictions are returned, determining which covariates contribute most to the model's success can be difficult. One commonly used method to extract feature importances[9] from random forests involves "feature depth" (Archer and Kimes, 2008). Since random forests consist of decision trees that are ordered variable splits, features that systematically appear earlier in the decision tree are more important to the model. Using this metric, we find that the most important features derive in equal parts from case-level covariates from the Supreme Court Database and the oral argument transcripts. In order of importance, the features 2, 5, and 9 come from the former, and features 1, 3, 4, 6, 7, 8, and 10 come from the latter. In Table 4, we indicate the name of the top 20 features by importance. The value of the "Importance" column is proportional to the average number of times that feature appears in the decision tree, weighted by

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

[8] 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 $(6/8)/(2/8) = 3$.

[9] We use this method as implemented in Python's `scikit-learn` module. In the `R` package `randomForest`, two alternate methods of variable importance are implemented. The first calculates the "total decrease in node impurities from splitting on that variable, averaged across all trees". The second involves a permutation test: permute a variable's values in the test set, and calculate the change in out-of-sample predictive accuracy. A large decrease indicates a highly important variable.

	Feature	Importance
1	Relative Number of Lawyers[10]	0.060
2	Issue Area[11]	0.031
3	Kennedy-Petitioner Questions[12]	0.030
4	Scalia-Respondent Questions	0.029
5	Case Origin: Circuit	0.028
6	Ginsburg-Petitioner Questions	0.027
7	Kennedy QC Ratio[13]	0.027
8	Roberts-Petitioner Questions	0.027
9	Reason for Cert	0.026
10	Interruptions	0.024
11	Ginsburg WC Ratio[14]	0.024
12	Scalia WC Ratio	0.024
13	Ginsburg Question Difference	0.023
14	Ginsburg Questions to the Respondent	0.020
15	Kennedy Cutoff Ratio[15]	0.020
16	Kennedy Question Target[16]	0.020
17	Lower Court Disposition Direction[17]	0.019
18	Scalia-Petitioner Questions	0.019
19	Breyer Cutoff Ratio	0.019
20	Lower Court Disposition	0.019

Table 4: The 20 features which contribute most to the model's accuracy. Across all features, importances sum to 1.

how early in the tree it appears; more simply, higher values indicate more strongly predictive features. These ten variables together account for more than 30% of the value of the model, strongly suggesting the mutual beneficiality of both data sets.

The first table below lists the top 20 features, ranked by importance. The second table lists all 55 features in alphabetical order.

	Feature	Importance
1	Administrative Action	0.011
2	Breyer: Comments to the petitioner divided by total comments	0.015
3	Breyer: Comments to the respondent divided by total comments	0.015
4	Breyer: Respondent comment ratio minus petitioner comment ratio	0.012
5	Case originated in a Circuit Court	0.028
6	Ginsburg: Comments to the petitioner divided by total comments	0.013
7	Ginsburg: Comments to the respondent divided by total comments	0.013
8	Ginsburg: Respondent comment ratio minus petitioner comment ratio	0.016
9	How many questions did Breyer ask the petitioner	0.015
10	How many questions did Breyer ask the respondent	0.017
11	How many questions did Ginsburg ask the petitioner	0.027
12	How many questions did Ginsburg ask the respondent	0.020
13	How many questions did Kennedy ask the petitioner	0.030
14	How many questions did Kennedy ask the respondent	0.015
15	How many questions did Roberts ask the petitioner	0.027
16	How many questions did Roberts ask the respondent	0.014
17	How many questions did Scalia ask the petitioner	0.019
18	How many questions did Scalia ask the respondent	0.029
19	Issue Area	0.031
20	Kenned: Respondent comment ratio minus petitioner comment ratio	0.011

[10] This measure indicates which side had more lawyers present during oral argument proceedings.

[11] This is the issue area of the case, as coded by the Supreme Court Database. Note that while this variable is literally coded after the fact of the case being argued, and thus may be considered post-hoc, we argue that the coding is sufficiently objective to be robust to the outcome of the case.

[12] This is a count of questions asked of the petitioner by Justice Kennedy.

[13] This measure is the difference of ratios of petitioner and respondent questions to total questions. If the petitioner was asked 3 questions and the respondent was asked seven, this ratio is $7/10 - 3/10 = 4/10$.

[14] This measure is the difference of ratios of words spoken to the petitioner and respondent, to total words spoken: If Ginsburg spoke 100 words to the petitioner and 50 to the respondent, this ratio is $100/150 - 50/150 = 50/150$.

[15] This measure is the difference of ratios of the times Kennedy interrupted the petitioner and the times Kennedy interrupted the respondent.

[16] This measure indicates whether Kennedy asked more questions to the petitioner or the respondent.

[17] This measure is whether the lower court ruled in favor of the liberal or conservative side, as determined by the Supreme Court Database.

21	Kennedy: Comments to the petitioner divided by total comments	0.014
22	Kennedy: Comments to the respondent divided by total comments	0.014
23	Lower Court Disposition	0.019
24	Lower Court Disposition Directon	0.019
25	Manner in which the court takes jurisdiction	0.002
26	Number of Lawyers: Ratio	0.060
27	Reason for Cert	0.026
28	Roberts: Comments to the petitioner divided by total comments	0.018
29	Roberts: Comments to the respondent divided by total comments	0.018
30	Roberts: Respondent comment ratio minus petitioner comment ratio	0.011
31	Scalia: Comments to the petitioner divided by total comments	0.016
32	Scalia: Comments to the respondent divided by total comments	0.016
33	Scalia: Respondent comment ratio minus petitioner comment ratio	0.011
34	State of Administrative Action	0.006
35	To which litigator did Breyer ask a higher ratio of questions to comments	0.015
36	To which litigator did Ginsburg ask a higher ratio of questions to comments	0.023
37	To which litigator did Kennedy ask a higher ratio of questions to comments	0.020
38	To which litigator did Roberts ask a higher ratio of questions to comments	0.015
39	To which litigator did Scalia ask a higher ratio of questions to comments	0.015
40	Which litigator did Breyer question more	0.012
41	Which litigator did Breyer speak more to	0.017
42	Which litigator did Ginsburg question more	0.015
43	Which litigator did Ginsburg speak more to	0.024
44	Which litigator did Kennedy question more	0.027
45	Which litigator did Kennedy speak more to	0.016
46	Which litigator did Roberts question more	0.008
47	Which litigator did Roberts speak more to	0.013
48	Which litigator did Scalia question more	0.012
49	Which litigator did Scalia speak more to	0.024
50	Which litigator was interrupted more	0.024
51	Which litigator was interrupted more by Breyer	0.019

52	Which litigator was interrupted more by Ginsburg	0.019
53	Which litigator was interrupted more by Kennedy	0.020
54	Which litigator was interrupted more by Roberts	0.011
55	Which litigator was interrupted more by Scalia	0.017

A concern with single-tree models is that they tend to overfit: outliers and dropped or missing values can have an outsized effect on their predictions. Larger trees with many nodes may reduce outlier sensitivity, but are more prone to overfitting. For this reason, ensemble learning methods, which combine many trees in different ways, are popular in practice. In a "random forest," (1) many trees are constructed simultaneously using bootstrapped samples of the data, (2) each tree's decision rules are generated using random subsets of the covariates, and then (3) the trees' predictions are averaged together (Liaw and Wiener, 2002). The bootstrapping procedure serves to reduce overfitting, while the random covariate selection eliminates systematic correlations between the trees, thereby boosting predictive power[18] (Ho, 2002). Random forests are also, by comparison to other ensemble methods, easy to use, with efficient implementations in `R` (Liaw and Wiener, 2002) and STATA.

Note that there are many potential covariates which we exclude from this model. Time-based covariates—for example, the year or month in which the case was heard, or the Court's median ideal point during the case—we found to *harm* our model's predictive accuracy. We also experimented with including a variable indicating whether the Solicitor General was a litigator in the case, but found it to be similarly unpredictive.

As well, there are covariates which we would like to include in our model but cannot. The number and text of amicus curiae briefs filed, for example, contain a wealth of information about the case related to public and elite opinion. While several data sets of amicus curiae

[18] In most machine learning ensemble methods, many weakly predictive models are aggregated together. If the "weak leaners" are weakly correlated at most, then each model picks up a different piece of the model variance, and the overall model will have more predictive power. If, however, the models are all highly correlated, then the ensembling procedure will add very little, and it is sufficient to take any single model by itself.

exist, none include cases during the period in which we conduct our analysis.

12 Appendix D: Additional Discussion of K-Fold Cross-Validation

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.

K-fold cross-validation is a commonly-accepted metric for model accuracy in computer science and statistics. When performing a single train set and test set split, a data set is randomly partitioned in two, a model is trained on one of the partitions, then used to predict the outcomes for the second partition. Note that a single train set and test set partition is equivalent to a 2-fold cross-validation procedure. This induces a trade-off between model power and accuracy precision: the more observations are reserved for the test set, the fewer may be used to train the model; the fewer observations reserved for the test set, the noisier the measure of out-of-sample predictive accuracy. Ten-fold cross-validation circumvents this trade-off altogether, using 90% of the available data to train the model each iteration, and averaging predictive accuracy across ten folds to increase precision. As K increases to equal N, both model accuracy and accuracy-measurement precision improve. However, computation time increases as well, so in practice, $K = 10$ is common.

13 Appendix E: Thoughts on the Use of Machine Learning in Political Science

We have two primary answers to the question of why political science has been slow to adopt machine learning methods: one is a practical reason, and one is a path-dependent reason. The practical reason is that machine learning works best when there is no measurement error in the outcome variable. Questions like "Is there a cat in this photograph?" are excellent for machine learning for the same reason that "Who will win this Supreme Court case?" is: either there is or is not a cat in a photograph, and either the respondent or the petitioner will win a Supreme Court case. On the other hand, questions like "What is the ideology of this document?" are much harder, because measuring ideology is a nuanced and error-prone endeavor. In short, most of the important dependent variables we care about in political science are noisy and difficult to measure with precision. The path-dependent reason is that political science is often focused on substantive interpretation of covariates, and often with causal implications. Machine learning is ill-suited to this approach, as the functional forms it induces around the data are not amenable to easy linear interpretation. For this reason, we believe that decision trees hold much promise: it is relatively straightforward to examine a decision tree and interpret it.

14 Appendix G: Technical Overview of AdaBoosted Decision Trees

AdaBoosted decision trees combine two powerful machine learning concepts: decision trees and gradient boosting. We will discuss each in turn.

Decision trees are a flexible non-parametric machine learning method for classification (categorical outcomes) or regression (continuous outcomes). The decision tree grows by optimizing “Gini impurity,” measuring how mixed are classes separated by that a given split. A Gini impurity index of 0 indicates that a split perfectly separates classes, while 1 indicates that each branch of a split is evenly divided among classes.

A simple decision tree consisting of one node finds the optimal split as measured by Gini impurity, producing two branches. A decision tree with two layers then performs the same optimal splitting procedure with each branch, resulting in four categories. A decision tree may have arbitrarily many layers, but additional layers increase the risk of overfitting.

```
Result: Decision Tree with n layers
initialization;
for $i \in n$ do
    for $l \in 2^{i-1}$ do
        Find optimal split in leaf $l$ in layer $i$ by minimizing Gini impurity;
    end
end
```

AdaBoosting is an ensembling method for combining multiple models *in sequence*. It is initialized by training a base model, often called a “weak learner,” on the full data set. This weak learner may be any model, often a linear or logistic regression, but in this case we use decision trees as the base learner. After the model is trained, residuals are calculated as the difference between predictions and the truth. In the case of a classification problem, these residuals are binary, whereas in regression problems they may be continuous.

In the second iteration, all observations in the data set are reweighted proportional to the size of their residuals, a new model is run, new predictions, residuals, and weights are calculated, and then the third iteration begins. The number of iterations is at the researcher's discretion, and more is better than fewer. After T iterations, the result is a series of $M_t \forall t \in T$ models, each of which has a prediction $P_{i,t}$ for each observation in the data set. The final prediction for observation i is the average of all predictions for that observation: $\frac{1}{T}\sum_t P_{i,t}$.

```
**Data**: Covariates $x_i$, and outcome $y_i$ for $i \in 1, N$; weights $w_{i,t}$
**Result**: AdaBoosted Predictions after T iterations
initialization;
Initialize $w_{i,t} = 1 \forall i$;
**for** $t \in T$ **do**
    Create a model $M_t$ to predict $y_i$ from $x_i w_{i,t}$ ;
    Generate predictions $\boldsymbol{P}(M_t)$;
    Calculate residuals $\boldsymbol{r} = \boldsymbol{P}(M_t) - \boldsymbol{y}$;
    Calculate $w_{t+1} \propto \boldsymbol{r}$;
**end**
Calculate final predictions: $\frac{1}{T}\sum_{t=1}^{T} \boldsymbol{P}(M_t)$;
```

15 Appendix H: Previous Supreme Court Prediction Models

Statistical models occasionally surpass the "petitioner wins" baseline. For example, Martin et al. compared expert predictions to a classification tree using six case-level covariates.[19] That model correctly predicted 75% out of 68 cases. 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 (Katz et al., 2014).

Following in the steps of Martin et al., recent attempts have shown reliable improvements over the "petitioner wins" baseline. {Marshall}+, which incorporates 95 case-level covariates into a predictive model (Katz et al., 2014), reports a predictive accuracy of 69.7% using

[19] These were circuit of origin, the issue area, the type of petitioner, the type of respondent, the ideological 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).

a random forest variant called Extremely Random Trees. These split candidate features randomly instead of along optimal thresholds, enjoying a decreased variance in estimates at the cost of increased bias. The second attempt is CourtCast (Roeder, 2015), 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 synthesize the results from multiple uncorrelated classifiers into one prediction, mitigate the costs of their constituent methods but often reduce the benefits. Finally, a model by Katz et al. 2017 allows for dynamic, time-varying predictions and reports an accuracy of 70.2%. Despite relatively modest gains in predictive accuracy, they boast the flexibility to predict any case for which covariates exist regardless of court composition or year.[20]

[20] As of writing, we have not had access to the data or replication code.