Legislation approval ratings prediction via vote correlation

Yuan Deng, Junyang Gao, Xiaoxue Wang
Department of Computer Science
Duke University
{ericdy, jygao, xxw211}@cs.duke.edu

Abstract

We implement the machine learning techniques, including topic modeling, support vector classification, and spectral partitioning, to predict the legislation approval ratings. Our novelty is to incorporate a new feature called **vote correlation**, which represents the voting similarities / correlations between two voters.

1 Introduction

In United States, there exists a special process to turn a bill into a law, including[1]

- Introducing the Bill and Referral to a Committee
- Committee Action: Hearings and Mark Up
- Committee Report
- Floor Debate and Votes
- Referral to the Other Chamber
- Conference on a bill
- Action by the President
- Overriding a Veto

Intuitively, to turn a bill into a law, there are several rounds of votes to decide whether the clauses or properties of the bills are acceptable or not. We focus on the votes, the "floor debate and votes" sub-process, to predict the voting results. In order to predict results, one straightforward way is to predict each voter's vote according to their voting history and aggregate their votes to form an outcome. However, we attempt another way to formulate the problem and introduce a new concept, called *vote correlation* to make use of the second order information in order to improve our prediction.

2 Outline of our algorithm

Different from the traditional way of prediction, in which based on the voting history of each voter, the system simply predicts the vote for the new legislation, our system works as follows:

1. For each pair of voters, compute vote correlations;
2. Classify voters into two groups by vote correlations;
3. Predict the aggregate vote of the two groups;

3 Problem Formulation

Assume the set of bills is $\mathcal{B}$ while the set of votes are $\mathcal{V}$. Moreover, each vote is corresponding to a bill, and thus, we denote $\mathcal{V}_b \subseteq \mathcal{V}$ as the set of votes corresponding to a specific bill $b \in \mathcal{B}$. Let F_b and F_v represents the feature vector of bill $b \in \mathcal{B}$ and $v \in \mathcal{V}$, respectively. In addition, we concatenate the two feature vector to get the real feature vector of a vote, which is $G_v = F_v \circ F_b$.

As for the voters, suppose the set of voters is $\mathcal{N}$ and each voter $p \in \mathcal{N}$ participates into set of votes $\mathcal{V}_p$. On the contrary, denote $\mathcal{N}_v$ for $v \in \mathcal{V}$ to be the set of voters participating in vote v.

Define function $f_p : \mathcal{V}_p \to \{0, 1\}$ as the function mapping the vote $v \in \mathcal{V}_p$ to his vote, which is either *yes*(1) or *no*(0). We introduce the notion of *vote correlation* for vote $v \in \mathcal{V}$ as a function $g_v : \mathcal{N}_v \times \mathcal{N}_v \to \{0, 1\}$ to represent whether two voters have the same vote for a specific vote v.

Finally, according to the function g_v for $v \in \mathcal{V}$, we divide the voters into two groups by a function $D_v : (g_v : \mathcal{N}_v \times \mathcal{N}_v \to \{0, 1\}) \to \{0, 1\}^{|\mathcal{N}_v|}$ to say which group each voter belongs to. In other words, all voters marked with the same label have the same vote in vote $v \in \mathcal{V}$.

4 Algorithm Analysis

Generally, our algorithm incorporates following ingredients and techniques:

- f_b, f_v, Feature extraction for votes and bills (topic modeling [2, 9]);
- g_v: Vote correlation computation (support vector classification (SVC) [3]);
- D_v: Group partition (spectral partitioning of graphs [4, 5]);
- Final step: Aggregate voting (majority rules [6] and SVC);

4.1 Feature Extraction

We extract the feature of a vote by running topic modeling on the description of bills and votes separately to get feature vectors for bill $b \in B$ and vote $v \in V$ as F_b and F_v.

Take bills as an example. Formally, given a set of descriptive text of bills and votes, according to bag-of-words assumptions, stating that *a text can be represented as the bag of its words, ignoring grammar and even word order but only maintaining multiplicity*, we can obtain a words-to-bill matrix as $M^{\mathcal{B}}$, in which $M^{\mathcal{B}}_{ij}$ represents the number of occurrence of $word[i]$ in $bill[j]$. Our objective is to decompose the matrix $M^{\mathcal{B}}$ as the product of two matrices, words-to-topic matrix $A^{\mathcal{B}}$ and topic-to-bill matrix $W^{\mathcal{B}}$, specifying the probability distribution of words in a topic and the probability distribution of topics in a bill.

That is, compute two rank k matrix (k is the number of topics), $A^{\mathcal{B}}$ and $W^{\mathcal{B}}$ to minimize the Frobenius Norm:

$$\|M^{\mathcal{B}} - A^{\mathcal{B}}W^{\mathcal{B}}\|_F = \sum_{ij} \left((M^{\mathcal{B}} - A^{\mathcal{B}}W^{\mathcal{B}})_{ij}\right)^2$$

As a result, we can use the topic-to-bill matrix $W^{\mathcal{B}}$ as the feature of each bill.

The traditional way to achieve this objective is to run a Latent Dirichlet allocation (LDA) model [7] and estimate the latent variables via famous expectationmaximization (EM) algorithm [8]. Recently, Arora et al. [9] propose a new topic modeling algorithm with provable guarantee under *separability* condition. For practice concern, we run MALLET package [2] as our sub-routine to implement topic modeling algorithm.

Finally, after computing F_b and F_v, the feature vector of a specific vote $v \in \mathcal{V}_b$ for $b \in \mathcal{B}$ is the concatenation of F_b and F_v.

$$G_v = F_v \circ F_b$$

4.2 Compute Vote Correlation

Our objective is to train a model to compute the *vote correlation* prediction function

$$g_v : \mathcal{N}_v \times \mathcal{N}_v \to \{0,1\}$$

for a specific vote $v \in \mathcal{V}$.

For voter a and b, select the votes that both of them participated by $\mathcal{V}_{ab} = \mathcal{V}_a \cap \mathcal{V}_b$. We enumerate each pair $(a,b) \in \mathcal{N}_v \times \mathcal{N}_v$ and formulate a classification problem as follows:

The training examples (x_v, y_v), where x_v is a feature vector of vote $v \in \mathcal{V}_{ab}$ and $y_v \in \{0,1\}$ to denote whether voter a and b have the same vote in vote v. That is, for all $v \in \mathcal{V}_{ab}$

- let $x_v = G_v$;
- $y_v = 1$ if and only if $f_a(v)$ equals to $f_b(v)$;

We run a support vector classification with radial basis function (rbf) kernel to obtain a non-linear classifier. Here, rbf kernel is defined as

$$K(\mathbf{x}, \mathbf{x}') = \exp\big(-\gamma \|\mathbf{x} - \mathbf{x}'\|^2\big)$$

We set $\gamma = 100$ and solve the following programming,

$$\begin{array}{rc} \min & \frac{1}{|\mathcal{V}_{ab}|}\sum \xi_v + \lambda c^T K c \\ \text{subject to:} & y_v \sum_{v'} c_v K(x_v, x_{v'}) \geq 1 - \xi_v \\ & \xi_v \geq 0 \end{array}$$

4.3 Group Partitioning

In the next step, we compute the group division function

$$D_v : (g_v : \mathcal{N}_v \times \mathcal{N}_v \to \{0,1\}) \to \{0,1\}^{|\mathcal{N}_v|}$$

In order to partition the group into two groups, we treat the matrix M obtained function g_v as an instance of a rank-2 random matrix C. The rationale to make this assumption is that we consider any two voters in the same group would have a constant probability p to be predicted to have the same vote while another constant probability q to have different votes. Thus, we can apply *spectral partitioning of random graphs* [4] according to following theorem

Theorem 1 *If $\tilde{M}$ is the best rank-k approximation to M, then for every rank k matrix C:*

$$\|\tilde{M} - C\|_F^2 \leq 5k\|M - C\|_2^2$$

where $\|M - C\|_2$ is the spectral norm of matrix $(M - C)$, which is bounded by $O(\sqrt{np})$ if M is a n-by-n matrix.

Thus, the above theorem shows that the rank-2 approximate $\tilde{M}$ of matrix M has average loss $O(1/\sqrt{n})$. As $n \to \infty$, such loss vanishes.

Moreover, the best rank-2 approximation of a matrix can be easily computed by singular vector decomposition of matrix M and keeping the top 2 eigenvalues s_1, s_2 and their corresponding eigenvectors $\mathbf{u}_1, \mathbf{u}_2$.

- if $s_2 = 0$, all voters are in the same group A;
- else, if $\mathbf{u}_1(p) > \mathbf{u}_2(p)$ then voter p is in group A; otherwise, voter p is in group B;

4.4 Aggregate Voting

After partitioning voters into two groups A and B, we treat each group as a super-voter (aggregate by majority rule) and predict its vote.

Again, we model the problem as a classification problem as follows: for each vote $v \in \mathcal{V}$

- let $x_v = G_v$;
- as for y_v:
  - if more than two thirds of voters vote for *yes*, then $y_v = 1$;
  - if more than two thirds of voters vote for *no*, then $y_v = 0$;
  - otherwise, discard v;

Again, we run a support vector classification with radial basis function kernel to obtain a classifier.

5 Experiments

5.1 Data Description

The data we used are crawled from open-source datasets from the web.

Vote	12348
Voter	955
Total Votes by voters	12592

Table 1: Dataset statistics

5.2 Experiments Setup

Due to computational complexity, it is impossible for us to run an experiment on the entire data sets. Thus, for each input of new vote, we select 50 voters and predict the results among these 50 voters.

5.3 Results and Discussion

We use *leave-one-out cross-validation* to measure the performance of our algorithm and the correctness rate of our algorithm is roughly 72% and the precision / recall matrix is as follows:

	0	1
0	261	83
1	120	262

Table 2: Precision / recall matrix

The correct rate is acceptable but there is a huge space for further improvement. Several possible ways may include

- Use a better topic modeling to extract the features of votes;
- Use other model to train the classifier than SVC;
- In aggregate voting, maybe other rules than majority rule can be implemented;

6 Conclusion

We successfully incorporate several machine learning techniques to develop a new way for legislation approval ratings predictions. Different than the traditional method, we examine the second order information, vote correlation between pairs of voters to improve the performance of algorithms.

References

[1] Steps in making a bill a law: The federal legislative process. `https://www.naeyc.org/policy/federal/bill_law`. [Online; accessed 05-Dec-2015].

[2] Shawn Graham, Scott Weingart, and Ian Milligan. Getting started with topic modeling and mallet. *The Programming Historian*, 2, 2012.

[3] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. A practical guide to support vector classification, 2003.

[4] Frank McSherry. Spectral partitioning of random graphs. In *Foundations of Computer Science, 2001. Proceedings. 42nd IEEE Symposium on*, pages 529–537. IEEE, 2001.

[5] Amit Kumar and Ravindran Kannan. Clustering with spectral norm and the k-means algorithm. In *Foundations of Computer Science (FOCS), 2010 51st Annual IEEE Symposium on*, pages 299–308. IEEE, 2010.

[6] Anthony J McGann. The tyranny of the supermajority how majority rule protects minorities. *Journal of Theoretical Politics*, 16(1):53–77, 2004.

[7] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[8] Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the royal statistical society. Series B (methodological)*, pages 1–38, 1977.

[9] Sanjeev Arora, Rong Ge, Yonatan Halpern, David M. Mimno, Ankur Moitra, David Sontag, Yichen Wu, and Michael Zhu. A practical algorithm for topic modeling with provable guarantees. In *ICML (2)*, volume 28 of *JMLR Proceedings*, pages 280–288. JMLR.org, 2013.