AMATH 482/582: HOMEWORK 4

JONATHAN BEAUBIEN

Applied Mathematics Department, University of Washington, Seattle, WA
beaubj@uw.edu

Abstract. We have been tasked with classifying Republican and Democratic politicians from the 1984 House of Representatives based on their votes on 16 bills. By using unsupervised spectral clustering, semi-supervised learning techniques, and relatively simple decision boundaries we were able to accurately predict a members party status with an accuracy of $88-89\%$.

1. Introduction and Overview

We have been tasked with classifying politicians from the 1984 House of Representatives based on their votes on 16 bills. There are 435 members of the House where 267 are from the Democratic Party and 168 are from the Republican party.

Our first task will be to use an unsupervised machine learning technique called spectral clustering. We will not know the labels for each member and their voting record but will try to cluster the politicians by political party. We will report our accuracy for varied hyper-parameters.

Next, we will see what accuracy we can get using a semi-supervised learning technique which will combine spectral clustering and linear regression. By knowing the labels for a small subsection of our data, we can see if that insight will help improve the accuracy of our classification.

2. Theoretical Background

The first technique we will use is unsupervised and is called spectral clustering. This technique uses the eigenvalues of a similarity matrix of our data which can help us cluster using fewer dimensions. Once we have this dimensionality reduction, we can perform a more classical clustering technique such as k-means or others. The first step of spectral clustering is creating a similarity matrix with a entries that correspond to weights of a similarity graph. Our data set will be $X = \{\underline{x}_0, \ldots, \underline{x}_{N-1}\} \in \mathbb{R}^d$ and a corresponding symmetric matrix $W \in \mathbb{R}^{N\times N}$ with non-negative entries $w_{ij} \geq 0$. In our case, we have $N = 435$ (members) and $d = 16$ (votes). We then define an undirected, weighted graph $G = \{X, W\}$ where the $\underline{x}_j \in \mathbb{R}^d$ are the vertices of G and the entries w_{ij} of W denote weights that are associated to edges that connect $\underline{x}_i$ to $\underline{x}_j$ [2]. Our weight function will be defined as:

$$w_{ij} = \exp\left(-\frac{\|\underline{x}_i - \underline{x}_j\|_2^2}{2\sigma^2}\right) \tag{1}$$

W is now our similarity matrix but to transform it into a graph Laplacian matrix (which has better properties) we will define the diagonal degree matrix ($d \in \mathbb{R}^N$) as

Date: March 11, 2022.

$$D = \begin{bmatrix} d_0 & & & \\ & d_1 & & \\ & & \ddots & \\ & & & d_{N-1} \end{bmatrix} \text{ where } d_j = \sum_{i=0}^{N-1} W_{ji}$$

Our unnormalized graph Laplacian will then be defined as

$$\tilde{L} = D - W \tag{2}$$

We can compute the eigendecomposition of our graph Laplacian $\tilde{L} = \tilde{Q}\tilde{\Lambda}\tilde{Q}^T$. The most important property of our graph Laplacian is this [2]: if the graph G has k-disconnected components (i.e. subgraphs that are not connected by any edges) then the eigenvalues of $\tilde{L}$ follow this structure

$$0 = \lambda_0 = \lambda_1 = \ldots = \lambda_{k-1} < \lambda_k \leq \lambda_{k+1} \leq \ldots$$

where λ_i corresponds to the eigenvector $\underline{\tilde{q}}_i$ of $\tilde{Q}$. Since our problem will have 2 clusters, an important vector is the Fielder vector defined as $\underline{\tilde{q}}_1$. We will use the sign of the Fielder vector as a simple clustering strategy for our unsupervised method where $\underline{\hat{y}} = \text{sign}(\underline{\tilde{q}}_1)$. However, we can construct a feature map and use M eigenvectors of our graph Laplacian matrix (called Laplacian embedding) as follows [4]

$$F(\underline{x}_j) = \begin{bmatrix} \underline{q}_{0j} \\ \underline{q}_{1j} \\ \vdots \\ \underline{q}_{(M-1)j} \end{bmatrix}$$

For our semi-supervised learning model we will use the first J observations as labelled data and the rest $(435 - J)$ as unlabelled. We can then solve a linear regression problem (with $\underline{y}$ as our labels) where $A_{ij} = F(X)_{ij}$, $\underline{b}_i = \underline{y}_i$, and $i = 0, \ldots, J-1$ and $j = 0, \ldots, M-1$ using least squares as follows

$$\hat{\beta} = \underset{\beta \in \mathbb{R}^M}{\text{argmin}} \, \|A\beta - \underline{b}\|_2^2 \tag{3}$$

Thus, we have a predictor for X where $\underline{\hat{y}} = \text{sign}(F(X)\hat{\beta})$. We will encode our labels as $+1$ for Republican and -1 for Democrat (so $\underline{y} \in \{-1, 1\}^{435}$) and for the features we will use the encoding of a 'no' vote as -1, a 'yes' vote as $+1$ and a '?' (missing or unavailable data) as 0.

For both models we will use an accuracy score as follows

$$\text{Accuracy} = 1 - \frac{1}{435} \times \text{number of misclassified members} \tag{4}$$

3. Algorithm Implementation and Development

The unsupervised and semi-supervised spectral clustering algorithms were modified versions of Prof. Bamdad's lectures [3] and [4]. The scikit-learn [5] implementation of linear regression was used in a Python environment. NumPy [1] was also used for matrix and array manipulation.

4. Computational Results

Let us first take a look at the Laplacian embedding of our data for the first 3 eigenvalues excluding the first one (which is ≈ 0 as there are two clusters). We can see in Figure 1 most of the variance of our data is along the Fielder vector, $\underline{\tilde{q}}_1$. This is why we are going to use the sign of the Fielder vector as our classification boundary.

Our first results in Figure 2 will correspond to our unsupervised spectral clustering method. We will run our clustering algorithm with many values of $\sigma \in (0, 4]$ to find σ^* where accuracy is maximized. With a step size for σ of 0.04, here is a graph of the findings.

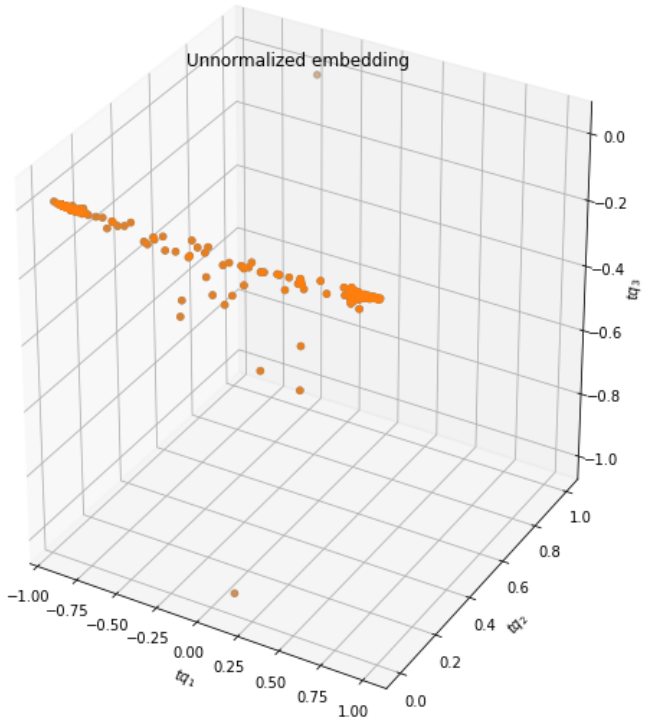

FIGURE 1. The data is not colored by political party, each member is simply an orange dot. Again, most of the variance is along the Fielder vector.

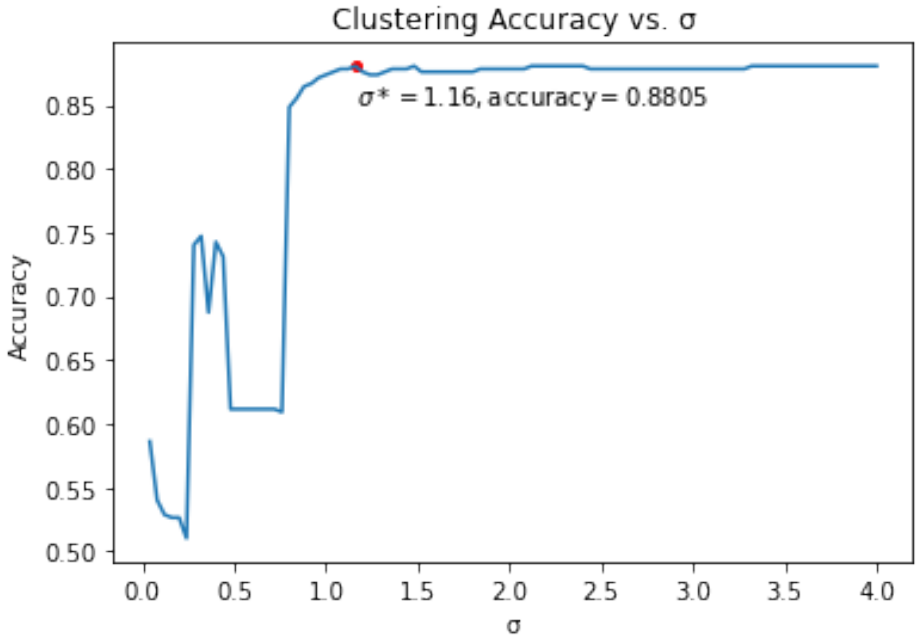

FIGURE 2. Accuracy is the proportion of correct classifications out of 435. The optimal value σ^* is shown to be 1.16 with an accuracy $\approx 88\%$.

	M=2	M=3	M=4	M=5	M=6
J=5	0.8896	0.8873	0.8390	0.8689	0.8689
J=10	0.8873	0.8160	0.8505	0.7103	0.6850
J=20	0.8827	0.8206	0.8643	0.8390	0.8735
J=40	0.8804	0.8367	0.8758	0.8804	0.8643

TABLE 1. Accuracy is the proportion of correct classifications out of 435. The results are surprising because the most accurate configuration is $M = 2$ and $J = 5$

Sticking with our $\sigma^* = 1.16$, we can now use semi-supervised learning with various values of M and J where M is the number of eigenvectors to use in our Laplacian embedding and J is the number of labelled data points we will use. After using our Laplacian embedding and linear regression we can get an accuracy value for pairs of $M = 2, 3, 4, 5, 6$ and $J = 5, 10, 20, 40$ in Table 1.

The most accurate combination is $M = 2$ and $J = 5$. This is incredibly surprising because theoretically more eigenvectors in the Laplacian embedding and more labelled data points would increase the accuracy of the classification. While $\approx 89\%$ accuracy is good, it is only 1% better than the unsupervised clustering accuracy.

5. Summary and Conclusions

We were able to use spectral clustering in unsupervised learning to cluster 435 Democratic and Republican politicians in the United States House of Representatives using their votes on 16 bills during 1984. Our accuracy was 88% after tuning hyper-parameters and using the labels for validation. We only had to use the sign of the Fielder vector to get this accuracy in our clustering. Then, after moving to a semi-supervised approach using more eigenvectors and including small subsections of labelled data, we only marginally improved the accuracy of our classifications.

Acknowledgements

The author is thankful to Professor Bamdad for his helpful Python notebooks on spectral clustering and semi-supervised learning. The author is also thankful to Ava Mistry for her help regarding citations and relevant equations.

References

[1] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020.

[2] B. Hosseini. Introduction to graph laplacians. University of Washington (LOW 216), Feb 2022.

[3] B. Hosseini. Semi-supervised learning. University of Washington-Seattle (LOW 216), Feb 2022. AMATH 482/582.

[4] B. Hosseini. Spectral clustering. University of Washington-Seattle (LOW 216), Feb 2022. AMATH 482/582.

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