Congress Clustering Final Report

Matthias Denu | CS6220 Data Mining Techniques

Motivation

The United States Congress is comprised of 535 voting members. There are 435 in the House of Representatives, one for each congressional district in the country, and 100 in the senate. The passage of legislation in either of these chambers depends on the amount of support a bill gets. Representatives and senators vote according to differences in political party, ideology, and region. For congresspeople, passage of a bill often depends on how well they are able to persuade and drum up support among colleagues, this can lead to groups of members voting together in “blocks”.

This project aims to be foundational work in segmenting members of Congress into different clusters, based on publicly available information. Done successfully, this could be used to better predict which members are likely to vote together, predict which are likelier to vote against their party, and identify which members most-likely to persuade other voting members.

This work could be built on later to help identify regional differences in voting patterns or to identify emerging political groups gaining power in Congress (e.g. newer members vs. “the old guard”).

Data Sources

ProPublica is a nonprofit newsroom whose mission is "To expose abuses of power and betrayals of the public trust by government, business, and other institutions, using the moral force of investigative journalism to spur reform through the sustained spotlighting of wrongdoing." (ProPublica, 2020, About Us).

ProPublica makes an API available for users to gather information about Congress (ProPublica Congress API). This was the primary source of data used to construct the dataset to carry out this project.

Census data (Census.gov) and Google Maps were also used to get central GPS coordinates for each state and each congressional district, however, although collected, the coordinates were not used in this version of the clustering project. For those interested, an Excel workbook of these coordinates has been included with the submitted project files.

Data Munging

The data returned from the ProPublica API was text formatted as JSON. Figure 1. shows a snippet of an example response from the *members* endpoint.

From the available fields on each member object, 13 were selected as the most relevant to a meaningful clustering. The were *id, in_office, date_of_birth, gender, party, dw_nominate,*

seniority, total_votes, missed_votes, total_present, missed_votes_pct, votes_with_party_pct, and *votes_against_party_pct.*

Many of the selected fields are self-explanatory, however the *dw_nominate* field is interesting because it is a rating given by the Voteview project that "allows users to view every congressional roll call vote in American history on a map of the United States and on a liberal-conservative ideological map including information about the ideological positions of voting Senators and Representatives." (Voteview, 2020, About Us).


```
{
   "status":"OK",
   "copyright":" Copyright (c) 2019 Pro Publica Inc. All Rights Reserved.",
   "results":[
      {
         "congress": "116",
         "chamber": "Senate",


         "num_results": 100,
         "offset": 0,
         "members": [
              {
                 "id": "A000360",
                 "title": "Senator, 2nd Class",
                 "short_title": "Sen.",
                 "api_uri":"https://api.propublica.org/congress/v1/members/A000360.json",
                 "first_name": "Lamar",
                 "middle_name": null,
                 "last_name": "Alexander",
                 "suffix": null,
                 "date_of_birth": "1940-07-03",
                 "gender": "M",
                 "party": "R",
                 "leadership_role": null,
                 "twitter_account": "SenAlexander",
```


Figure 1. Snippet of a ProPublica API Response

Once the JSON results for each chamber had been converted to pandas DataFrames and concatenated, the *id* field was used as an index (e.g. primary key) and only used for identification, not clustering. The *dw_nominate_field* was the only field with null values. The 109 null values were replaced with the average value of the other 428 samples. The *gender* field was turned into a binary value of either 0. or 1. for female and male. A one-hot-encoder was used to create Boolean fields (0. or 1.) for each of Democrat, Independent, and Republican. The *date_of_birth* field was truncated to only the year and converted to an integer. A Boolean field for *chamber* was added to represent the House of Representatives and the Senate.

This resulted in a dataset having 14 fields. The samples were filtered to only keep those where the *in_office* field was true. This resulted in 537 members in total, which is greater than 535 because there are 6 non-voting delegates who are members of Congress. It was decided to keep these datapoints as they were not expected to adversely impact the clustering.

Finally, the data in each column was scaled using a *sklearn.preprocessing.MinMaxScaler* so that the range of each feature was on the range [0, 1].

Besides the Python programming language, the technology stack used for this project consisted of the following tools:

- Jupyter Notebooks
- Requests: http for Humans
- scikit-learn
- pandas
- matplotlib
- numpy

Clustering Methods

The dataset consisting of a manageable number of datapoints allowed different clustering methods to be tried quickly, with different parameters. Manually looking through the data and reasoning about it suggested that there wasn't a huge need to be concerned about outliers or nonlinear relationships between features.

After looking the distribution of each feature in the dataset (Appendix A.), and after looking at a plot of the explained variance for each component of PCA (Figure 2.), there was no reason to believe that there would be non-convex clusters hidden in the data.

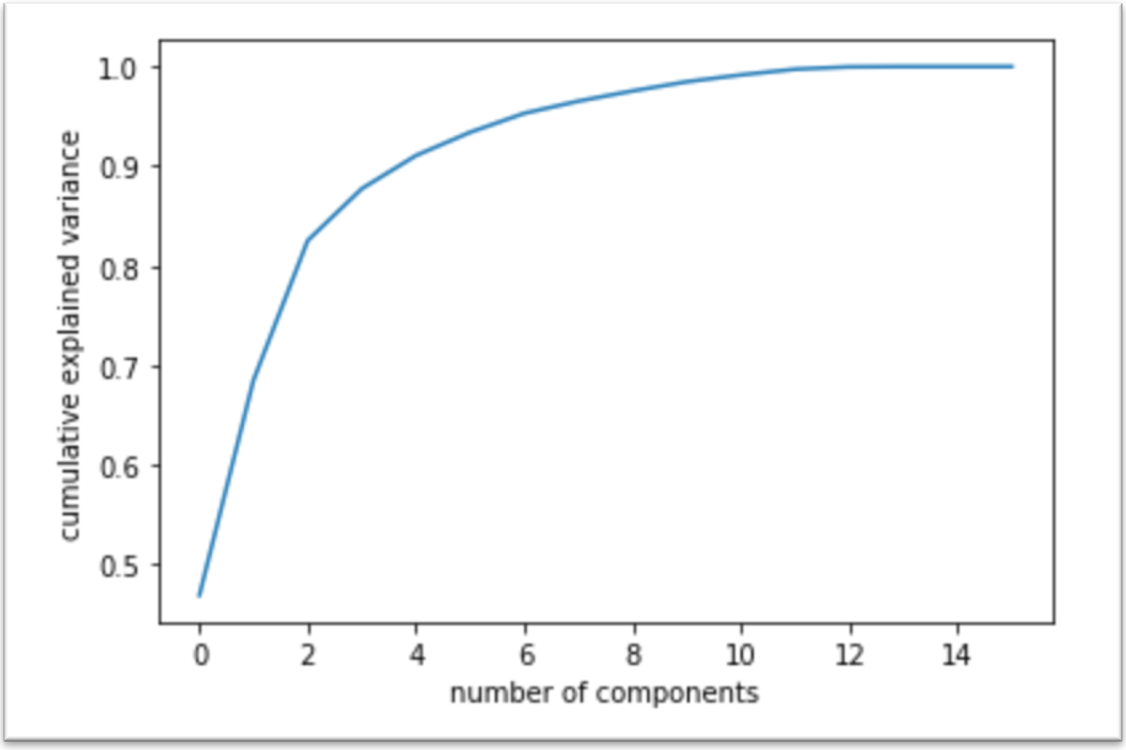

Figure 2. Plot of PCA Component Explained Variance

This knowledge led to the assumption that it should be possible to find a good k-means clustering of the data. Spectral clustering and DBSCAN clustering were also used to add variety when evaluating the results.

The first step taken was to plot the data along two dimensions using the PCA done earlier. Because PCA had shown there to be 2 components that explained ~85% of the data, this seemed like a reliable method. The result of that plotting is shown in Figure 3.

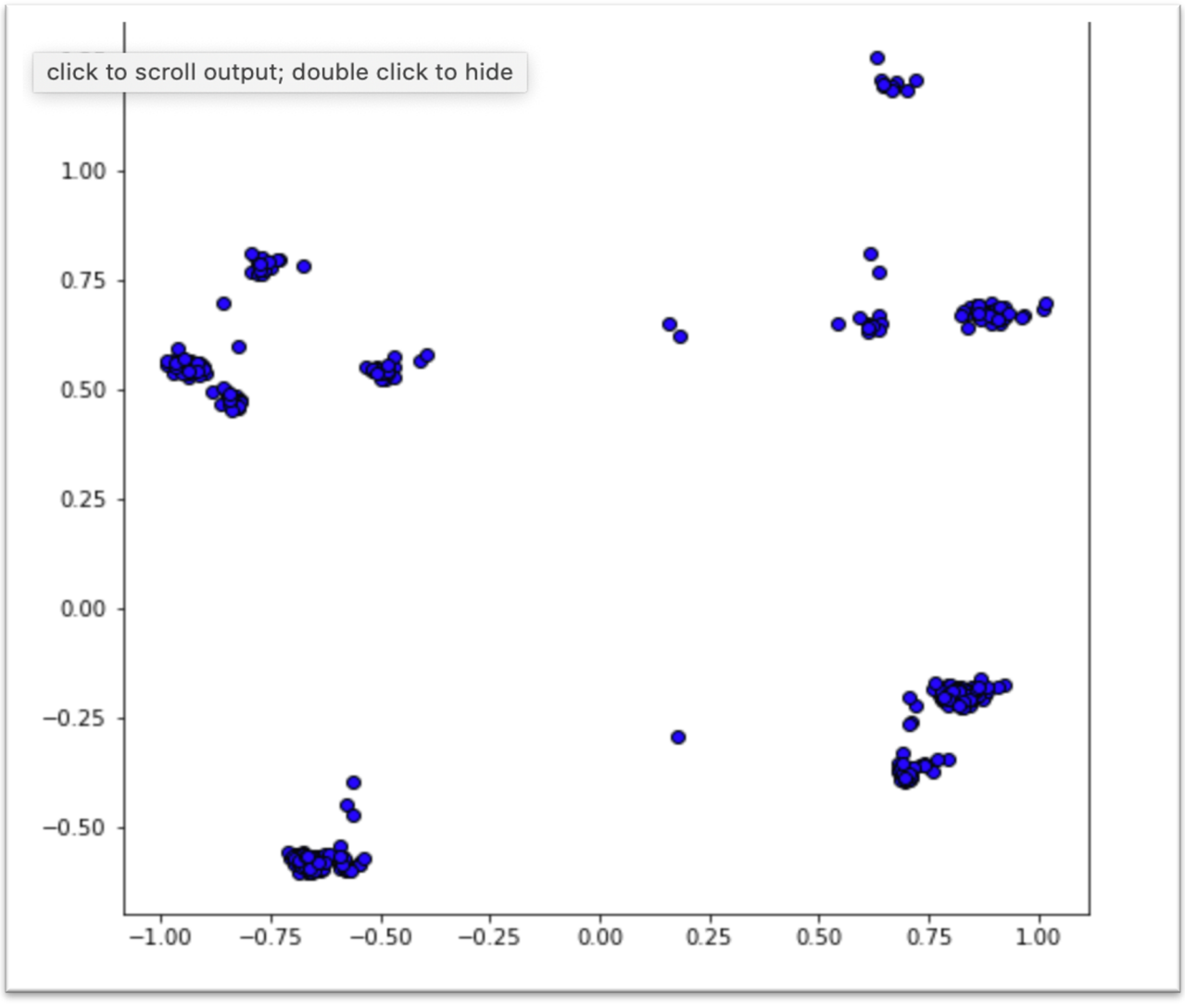

Figure 3. Plot of Datapoints on 2 Dimensions

This visualization shows that a good number of clusters is expected to be between 4 and 11, but that numbers outside of that range should be viewed with skepticism.

Three metrics were used during evaluation: silhouette coefficient, Calinski-Harabasz Index, and Davies-Bouldin Index (scikit-learn, 2020).

The metrics are listed below. Not surprisingly, all indicate that the clusters were dense and well-separated.

Silhouette Coefficient

- DBSCAN – 0.609
- Spectral – 0.581
- K-Means – 0.463

Davies-Bouldin Index (lower is better)

- DBSCAN – 0.579
- Spectral – 0.705
- K-Means – 0.993

Calinski-Harabasz

- DBSCAN – 554.282
- Spectral – 434.499
- K-Means – 430.724

DBSCAN consistently outperformed the other two algorithms, with spectral clustering doing second-best and k-means not performing as well as the other two. Looking at their respective visualizations (Figure 4., 5. 6.) , it's clear that DBSCAN's better performance is due to the

eliminating outliers from clusters and labeling them as noise. And it can be seen that spectral clustering outperforms k-means because it does not have overlapping clusters like k-means.

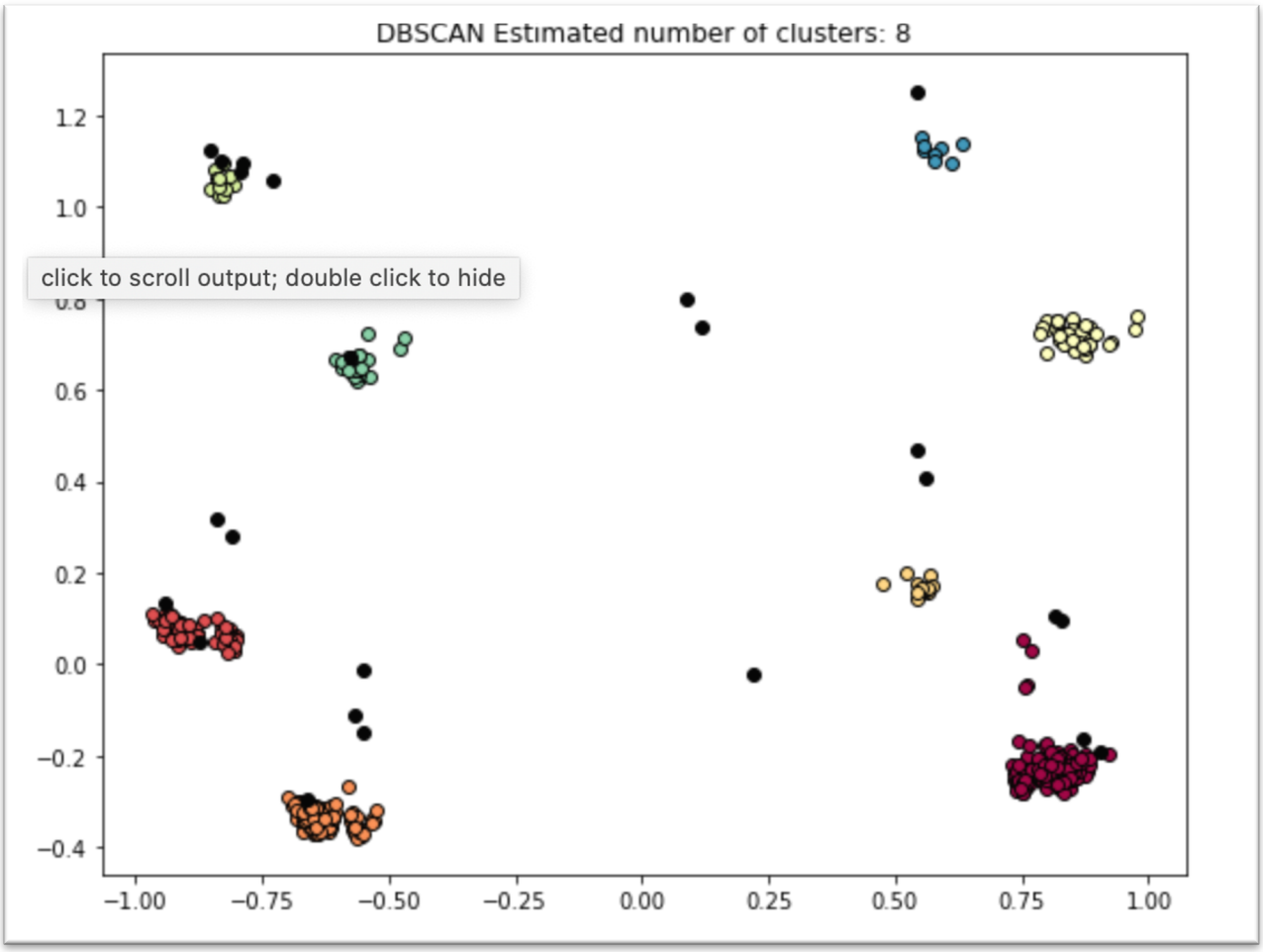

Figure 4. DBSCAN Clustering Plot.

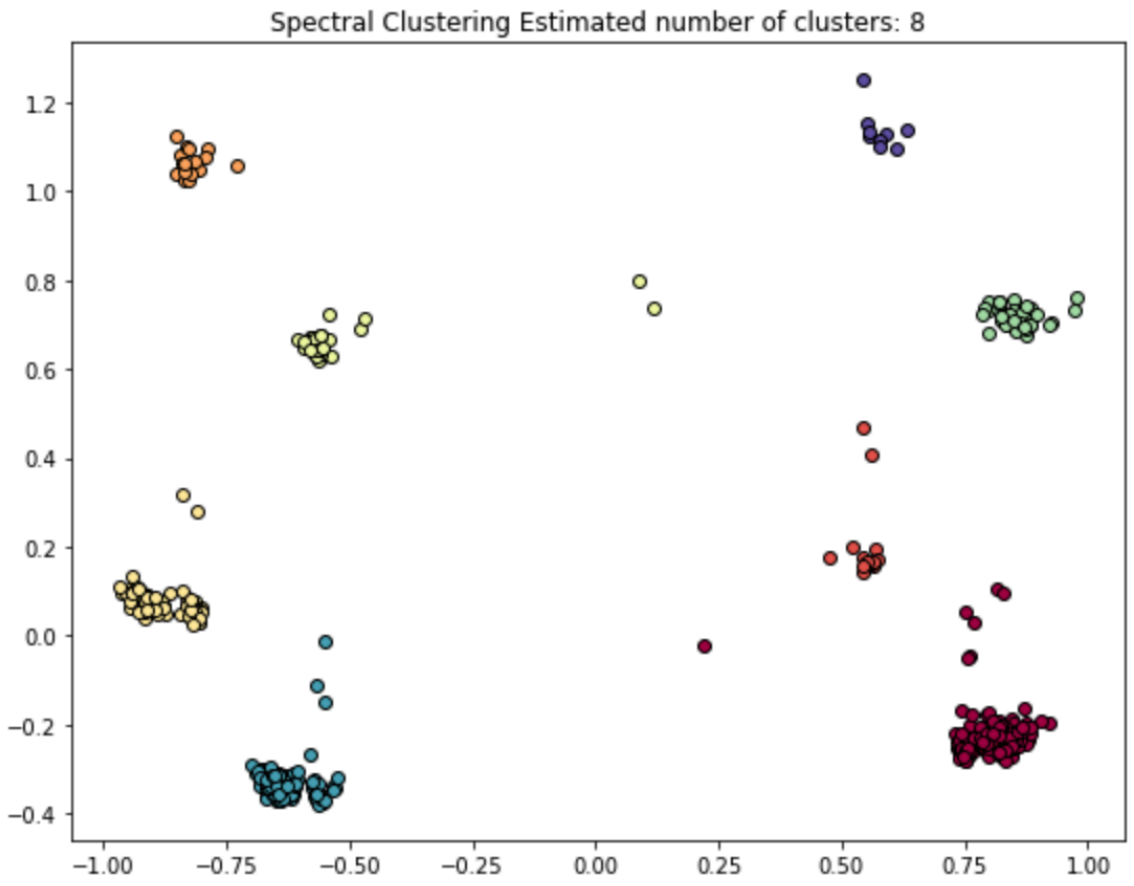

Figure 5. Spectral Clustering Plot

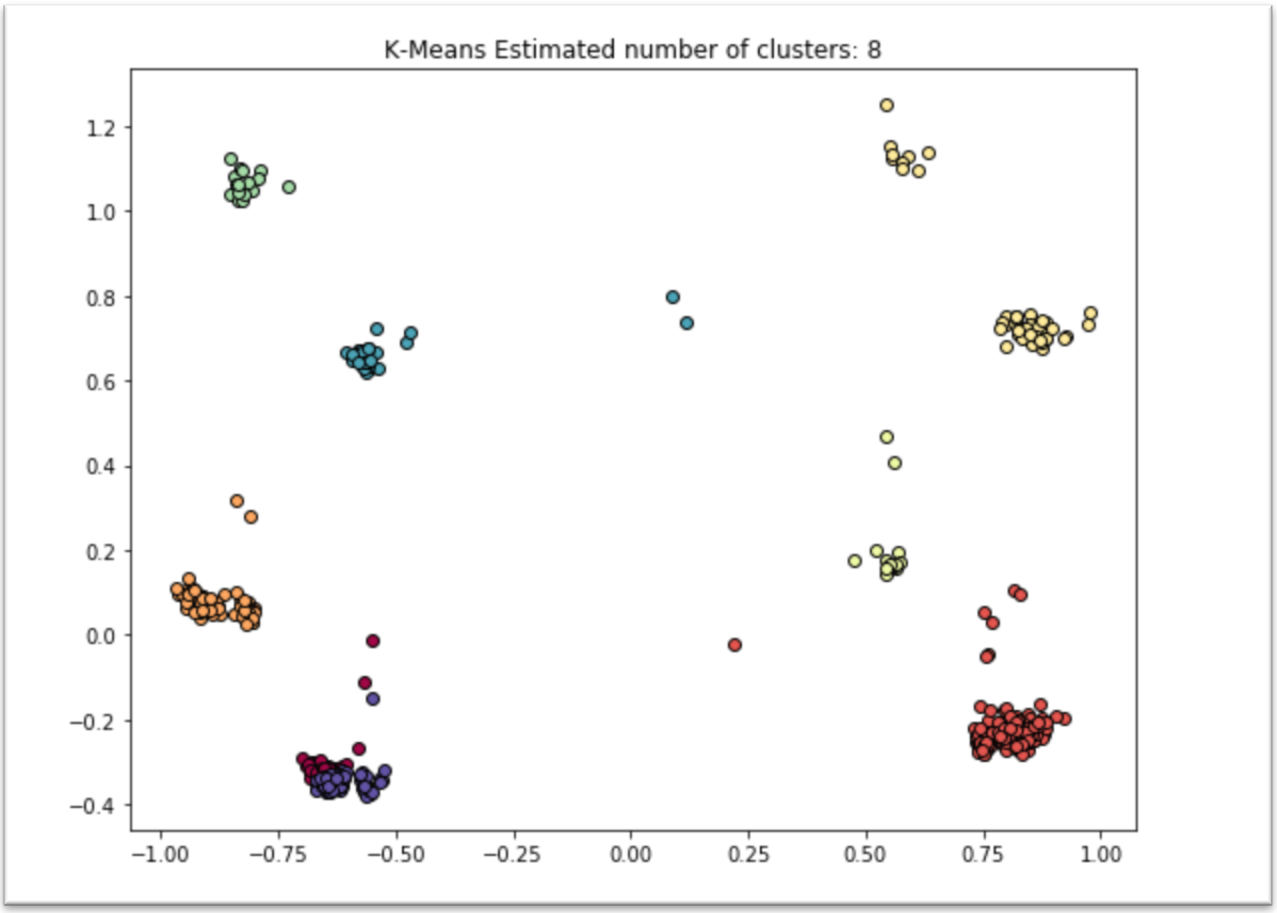

Figure 6. K-Means Clustering Plot

Findings

This project has shown that it is indeed possible to make a good clustering of members of Congress.

More work will need to be done to determine defining characteristics of each cluster. The outliers identified by DBSCAN may be of interest as well.

Future Work

There are two tasks that would add value to this project:

- Determine why there might be 8 nice clusters and what their discriminating characteristics might be
- Incorporate geography datapoints into the dataset to see if regional differences affect members
- Incorporate the number of terms served

References

Lastname, F. M. (Year, Month Date). *Title of page.* Site name. URL

Census.gov. (2020). *TIGER/Line Shapefiles.* https://www.census.gov/geographies/mapping-files/time-series/geo/tiger-line-file.html

ProPublica. (2020). *About Us.* https://www.propublica.org/about

ProPublica. (2020). *ProPublica Congress API*. projects.propublica.org/api-docs/congress-api/

scikit-learn. (2020). *2.3.1 Clustering performance evaluation*. https://scikit-learn.org/stable/modules/clustering.html#clustering-evaluation

Appendix A. Distribution of Each Feature int the Dataset

	dw_nominate	seniority	total_votes	missed_votes	total_present	missed_votes_pct	votes_with_party_pct	votes_against_party_pct	chamber	democrat	independent	republican	gender	date_of_birth
count	537.000000	537.000000	537.000000	537.000000	537.000000	537.000000	537.000000	537.000000	537.000000	537.000000	537.000000	537.000000	537.000000	537.000000
mean	0.044668	10.476723	805.625698	30.828678	0.206704	4.130540	94.421899	5.483073	0.186220	0.523277	0.005587	0.471136	0.756052	1960.096834
std	0.408043	8.959065	151.849160	57.125553	0.826402	8.398267	5.436472	5.446501	0.389647	0.499924	0.074604	0.499632	0.429862	11.788003
min	-0.769000	1.000000	35.000000	0.000000	0.000000	0.000000	58.650000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1933.000000
25%	-0.351000	4.000000	879.000000	6.000000	0.000000	0.690000	92.730000	1.550000	0.000000	0.000000	0.000000	0.000000	1.000000	1951.000000
50%	0.044668	8.000000	879.000000	15.000000	0.000000	1.820000	95.900000	4.030000	0.000000	1.000000	0.000000	0.000000	1.000000	1960.000000
75%	0.429000	14.000000	879.000000	31.000000	0.000000	3.870000	98.350000	7.160000	0.000000	1.000000	0.000000	1.000000	1.000000	1969.000000
max	0.916000	48.000000	882.000000	803.000000	14.000000	91.350000	99.570000	41.350000	1.000000	1.000000	1.000000	1.000000	1.000000	1989.000000