Applications of SVD: Principal Component Analysis

CLA365 Inc Technical Report

Courtney Brown, Tenzin Dayoe, Bethany Wang

Submitted 5/4/25

Abstract

In this report, we will demonstrate the utility of using SVD for principal component analysis to evaluate similarities among observations in a dataset. We will use data of votes from the US Senate for our analysis.

Introduction

Principal Component Analysis (PCA) is a technique that takes a dataset with possibly correlated features and extracts new set of uncorrelated features called principal components. The principal components are linear combinations of the original feature vectors and are ordered by the amount of variance they explain. The first principal component explains the greatest variance in the data, the second the next greatest, and so on. PCA seeks to preserve as much variability as possible in fewer dimensions. Given an $n \times p$ data matrix X, where n is the number of observations and p is the number of features, we subtract the mean of each column to form a centered matrix for our analysis. Then we compute the correlation matrix as $\frac{1}{(n-1)} X^\top X$. The diagonalization of the correlation matrix is $V \Lambda V^\top$, where $V^\top$ is the loadings matrix representing how much of each original feature is present in the principal components, and XV is the matrix of scores which stores the principal components. The data can be projected onto the first k principal components to obtain a reduced representation.

Singular Value Decomposition (SVD) is a general matrix factorization method that decomposes real-valued matrix X into the product of three matrices: $X = U \Sigma V^\top$. Here, U contains the left singular vectors, which represent the output space. Σ is a diagonal matrix of singular values, and V contains the right singular vectors, which represent the input space. When SVD is applied to centered data, the vectors in $V^\top$ are equivalent to the principal components obtained through PCA. Moreover, the squared singular values from Σ divided by $n-1$ correspond to the eigenvalues of the correlation matrix from Λ. This connection implies that SVD can be used to perform PCA without computing the correlation matrix directly. SVD provides a more computationally efficient way to perform PCA, especially for large datasets.

Case Study: Senate Votes

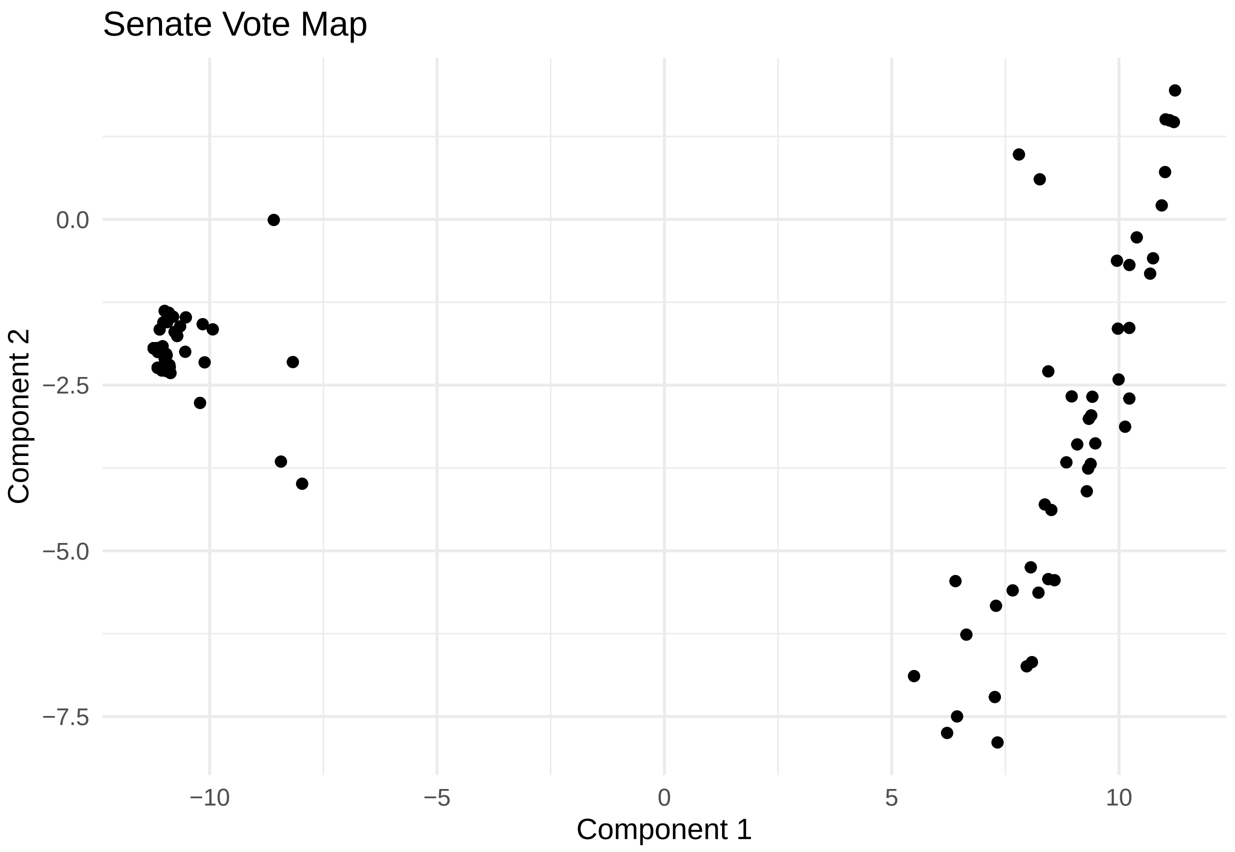
The matrix used for this SVD analysis is constructed from US Senate voting data containing senator votes for 131 bills from the 2025 session of the 119th Congress, where each row represents a senator and each column corresponds to a specific bill vote (for example $vote_1, vote_2, \dots$) . The entries are encoded numerically where 1 is for “yea”, 0 is for no vote, -1 is for “nay”.

Converting this dataset into a matrix gives us a large rectangular 100×131 matrix. Applying SVD to the votes matrix factorizes it as $X = U \Sigma V^\top$.

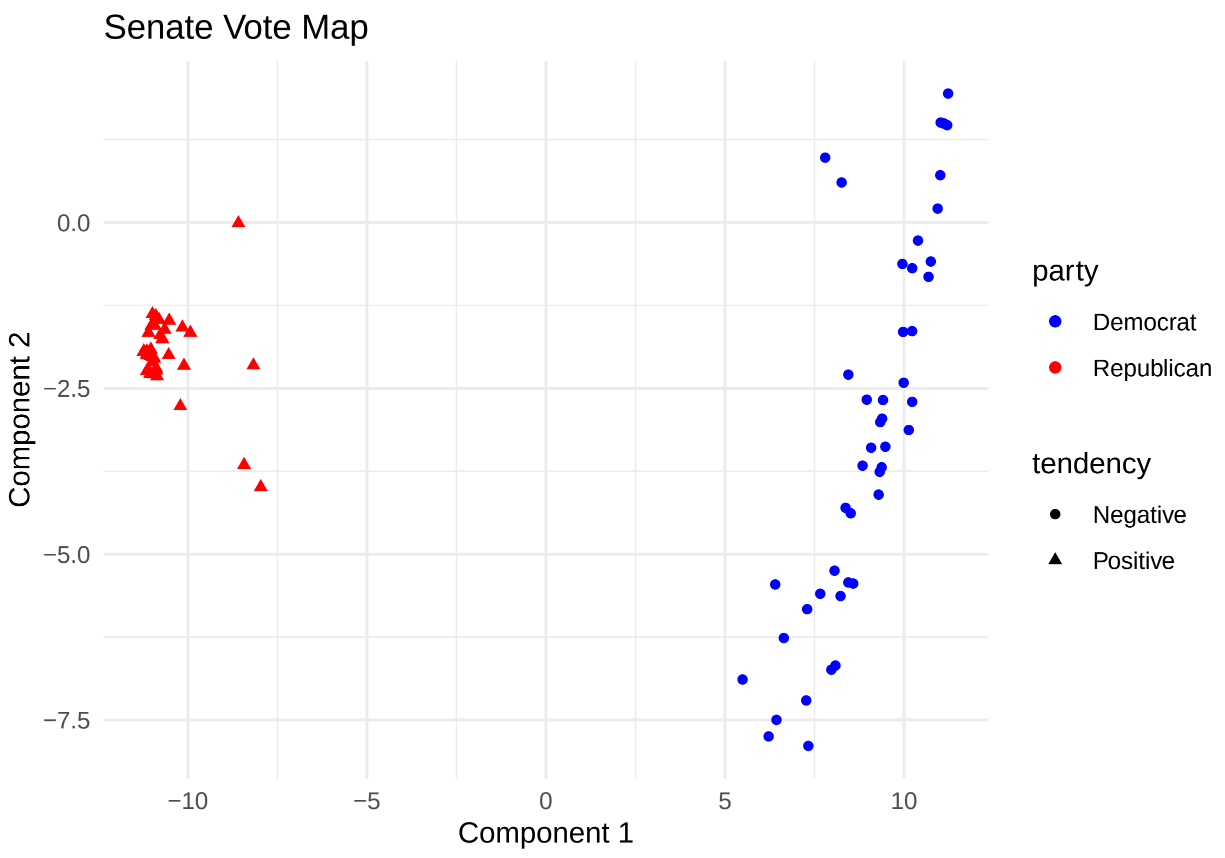
In the context of PCA, $U\Sigma$ give us the scores matrix, the columns of which are the principal components ordered by importance, and $V^\top$ gives us the loadings matrix.

Using just the first 2 principal components and their associated loadings, we can construct a rank 2 approximation of the original voting matrix. Using this approximation, we can take the sum of each row to determine the senators' voting tendencies. If the sum of their votes is negative, then we say they have a "tendency" to vote negative, and if the sum of their votes is positive then we say they have a "tendency" to vote positive.

By taking the first principal components, we can project each senator into a 2D space that captures the most significant variation in voting behavior. We used Senate Vote #228 to obtain the senators' names and party affiliations, though names are not included in the below plot in order to preserve legibility.



This plot shows a clear divide between two groups of senators. Having obtained the senators' party affiliations and voting tendencies, we then modified the plot to investigate which, if either, of these facets was identified by the PCA.



This plot shows that the first principal component carries information about the party affiliation and voting tendency of the senators.

Conclusion

Using the example of US Senate voting data, we showed that Principal Component Analysis performed with SVD can extract meaningful patterns from high dimensional data. The 2D projection using the first two principal components revealed a clear separation among senators.

The first component captures the most significant variation in voting behavior – evidently partisan alignment and voting tendencies, which both correspond with ideological differences between the groups.

The second principal component, which is less dominant, appeared to reflect more subtle distinctions. While we couldn't identify what, exactly, this component captured, we observed that the Democratic senators have much more variation along this axis than the Republican senators.

Further use of this method with this data could include obtaining voting data from previous Senate sessions and performing the same analysis. Comparisons of the different PCAs could reveal shifts in which factors influence voting behavior in the US Senate.

R Code Appendix

Scraper Code

This is the code written by Tenzin to form a single CSV file of the Senate votes from GovTrack's congressional votes database, run from Google Colab.

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
warnings.filterwarnings('ignore')

def getURL(sNumber):
    return f"https://www.govtrack.us/congress/votes/119-2025/s{sNumber}/export/csv"

urls = [getURL(i) for i in range(100, 231)]

combinedDf = None
for i, url in enumerate(urls):
    df = pd.read_csv(url, skiprows=1)

    df = df[['person', 'vote']]


    vote_col = f'vote_{i}'
    df[vote_col] = df['vote'].map({
        'Yea': 1,
        'Nay': -1,
        'Not Voting': 0,
        'NA': 0,
        '': 0
    }).fillna(0)

    df = df[['person', vote_col]]

    if combinedDf is None:
        combinedDf = df
    else:
        combinedDf = pd.merge(combinedDf, df, on='person', how='outer')

print(combinedDf)

combinedDf = combinedDf.fillna(0)
combinedDf.to_csv('combined_votes.csv', index=False)

from google.colab import files
files.download('combined_votes.csv')
```

Analysis Code

Votes matrix

```
# Excludes the first column of Senator IDs
votes <- as.matrix(read.csv("combined_votes.csv"))[,2:132]
```

SVD for PCA

```
out <- svd(votes)
sc_loadings <- t(out$v)
sc_sdev <- out$d
```

```
sc_scores <- out$u %*% diag(out$d)
```

Names and Parties

```
singleCase <- read.csv("congress_votes_119-2025_s228.csv")
senator_names <- as.matrix(singleCase)[,5]
senator_parties <- as.matrix(singleCase)[,6]
```

Tendencies

```
votes_approx <- sc_scores[,1:2]%*%sc_loadings[1:2,]
tendencies <- rowSums(votes_approx)
senator_tendencies <- tendencies
for (i in 1:length(tendencies)) {
  if (tendencies[i] < 0) {
    senator_tendencies[i] <- "Negative"
  } else {
    senator_tendencies[i] <- "Positive"
  }
}
```

Dataframe for Plot

```
votesFrame <- data.frame(x = sc_scores[,1], y = sc_scores[,2], name = senator_names,
  party = senator_parties, tendency = senator_tendencies)
```