There are some metrics in Spotify that describe different songs, like acousticness, danceability, energy and so forth. What we are going to do now is analyze some metrics that can describe episodes in a similar way. Spotify supports an API for users to get episode data. To scrape data, we set up an app in Spotify, then use Client ID and Client secret to get an access token. After doing that, we can get data we need by different URLs with a certain Spotify ID.

We used 35 different podcasts and got over 60 thousand episodes. What we use is the description column in the dataset. The “description” column are sentences that make a summary of each episode, so we can extract information from it. We plan to get the top 100 frequency of each word in this column, so that we can use PCA to reduce dimension. However, the top 100 words mainly consisted of prepositions, pronouns and articles, like “to”, “the”, “of”, etc., which is meaningless and not good for extracting information. So, we add these words as noise into a list to avoid them appearing in our final output. We did this work five times to make the top 100 words not contain them. Finally, our words only contain nouns, verbs and some other words that are meaningful.

Now we have one hundred columns corresponding to the frequency of top 100 words that appear in the description. PCA is sensitive to the scale of the data, so the step of standardization is very significant. PCA computes a covariance matrix to understand how features vary together, followed by eigenvalue and eigenvector calculation to determine the eigenvectors and eigenvalues of variance in the data. These eigenvectors define the principal components, with the first capturing the largest variance and subsequent components capturing progressively less while remaining orthogonal to the others. The original data is transformed by projecting it onto these principal components, reducing dimensionality while retaining the most significant variance. Finally, the PCA components are normalized, often scaled to a range like [0, 1], for easier interpretation and comparability across components or datasets. This process simplifies complex datasets while preserving critical patterns. Finally, we get five new metrics with no correlation. By going over the top 100 frequencies of words and the value of metrics, we named them as Historical Themes, Biographical Content, Geographical Contexts, Cultural Narratives and Military Strategies.

After PCA working, we cleaned the data again. There are some observations that did not apply with PCA working and the output is strange. This kind of data is less than one percent of all, so we directly delete them. Then we replaced the original 100 columns of words' frequency data with these five new metrics to apply the cluster. In the Shiny app, we use Euclidean Distance to define whether two spots are close or not.

In our model, the logic is not hard to understand and the metrics sound interpretable. And the metrics are independent with each other so there is no question of overlay or substitution between them. Also, our model has some disadvantages. The first is the distance we choose to define the cluster may not be the best choice. We only try the Euclidean Distance. If possible, it would be better to use some other distances and make comparisons, which will make our cluster more reliable. The other disadvantage is we go over the metrics manually. Although it can make the names of metrics more meaningful, there will exist some bias unavoidable.

By working with this, it gives us a thinking that applies categorical data or string to PCA, while PCA is only available for numerical data. Although there are some disadvantages with this model, the clear and interpretable logic makes it easy to make mistakes.

Contribution

|  |  |  |
| --- | --- | --- |
| Contribution | Chenghao Lou | Yi Ma |
| Summary | PCA and Data Cleaning | Data description, Discussion and Summary |
| Code | Code for applying PCA | Code for scraping data and cluster |
| Shiny App | Reviewed/edited and provided feedback on Shiny app | Responsible for Shiny app |