SNLP assignment 4: Clustering languages

Deadline: Jun 27, 2018 @ 10:00 CEST

In this set of exercises, we will try to discover the relatedness of languages based on a multi-lingual word list. We will use a simplified setting and a few unusual twists for the sake of exercise. However, the problem is of a real research interest, as well as being potentially useful in some NLP applications.

The data set we will use for this exercise comes from NorthEuraLex.[1] The database contains pronunciations of words expressing over 1000 concepts in more than 100 languages of (northern) Eurasia. For the present exercises, we will only use the phonetic inventories of some of the languages present in the data, without paying attention to the parallel nature of the database.

The main data file, `northeuralex-words.gz`, contains two columns separated by tab. The first column is the 3-letter ISO language code, the second one is tokenized IPA representation of a word. In some of the exercises you will also need the information form `language-family.csv`, which includes the name and language family information of the languages in the `northeuralex-words.gz`. The family information comes from GlottoLog.[2]

The exercises below should be implemented as a single python script, `cluster.py`. Recommended variable names for some of the data structures are printed on the left margin, and referred to with this name in instructions that follow. Some of the exercises ask you to plot figures, for which you will most likely use `mathplotlib.pyplot` package. You are required to produce these figures as PDF files, and save as the names indicated in the exercises for your final submission.[3] For exercises where you are asked to print out some information, you should just print them to the console (e.g., use the `print()` function). You do not need to write down the answers to the questions on the right margin, but trying to find the answers to these questions will help understanding the methods better.

Exercise 1. *Feature extraction*

Read in `northeuralex-words.gz` and create two variables

`features` A matrix where rows correspond to languages, columns correspond to IPA tokens, and values indicate the number of times the IPA token observed in the words of the language.[4]

`languages` A list with the language codes with the same order as the rows of the data matrix.

Exercise 2. *K-means, Silhouette score*

Using the `features` matrix you created in Exercise 1, repeat k-means clustering with k in range $[2, 70]$, and plot the value of the *error function* (sum of squared distances from the centroids), and *silhouette score* for each k-value.[5]

Why am I doing this?

- Experiment with unsupervised learning (clustering, dimensionality reduction)
- Interpreting / evaluating clustering results
- A glance at computational historical linguistics

[1] `http://northeuralex.org/`, (Dellert and Jäger, 2017).

[2] http://glottolog.org/ (Hammarström et al. 2018).

All source code and data files must be pushed to your assignment repository before the deadline.

[3] You may, of course, plot the graphs on screen during development.

[4] Using a dense matrix may simplify some of the later tasks.

[5] What is the reasonable number of clusters for this data?

Save the figures to files named `kmeans-error.pdf` and `kmeans-silhouette.pdf`, respectively.

Exercise 3. *Principle component analysis*

The rows of the `features` matrix you have created are rather long 'language vectors' with some redundancy. That is, occurrences of certain phones will correlate with others. As a result, it is reasonable to expect a reasonable amount of compression to be achieved using PCA without losing a lot of information.

`features2d` First, fit a PCA model, reducing the dimensionality to 2. Store the reduced data set in variable `features2d`. Plot the data points on a two-dimensional graph, labeling them with the 3-letter language codes from the `languages` list. Save the graph as `pca.pdf`.[6]

- Print out the ratio of variance (≈information) *lost* due to dimensionality reduction.[7]

`featuresPCA` Find out the smallest number `d` that preserves at least 90 % of the variation in the original data (`features`). Use the variable name `featuresPCA` for the transformed `d`-dimensional data.[8]

[6] Inspect the graph carefully. Do related languages placed closer on the graph? Can you assign any interpretations to the two PCA dimensions?

[7] What does that mean for the interpretation of the graph created in the previous step?

[8] How much compression would you achieve by losing only about 10 % of the variation?

Exercise 4. *Evaluation with gold-standard labels*

The file `language-family.csv` contains 'language family' information for the languages in our data set, as a CSV file.[9]

`family` read the 4th column (labeled `family_l1`) of the file `language-family.csv` and store it as a list named `family` such that the family of the language in the list `languages` is placed at the corresponding index of the new list.

- Fit k-means model with the same number of unique families in the `family` list, using both original data (`features`) and with dimensionality reduced data (`featuresPCA`).
- Print out and inspect the gold-standard family labels and k-means clusters assigned to each language by both models.[10]
- Calculate and print out the *homogeneity*, *completeness* and *V-measure* for both models.[11]

[9] The file contains a few additional fields, including a second-level family classification, (e.g., Indo-European is broken down to Balto-Slavic, Germanic, etc.). Although use of this field is not required in this assignment, you are encouraged to do further evaluations/explorations based on second-level families (also in Exercise 6).

[10] Can you manually inspect how well the model is doing?

[11] Do the scores agree with your judgment? Which model performs better? Why? (Due to random initialization, your results may slightly differ if you run the clustering multiple times.)

Exercise 5. *Calculating distances*

Create an upper triangle matrix such that the i^{th} row and j^{th} column of the upper triangle of the matrix is the Euclidean distances between i^{th} and j^{th} languages in the `languages` list. Other values in the matrix should be set to 0. Store your matrix with variable name `dist`.

For each language in `languages`, print out the closest language according to the your distance matrix.[12]

[12] Are the closest languages the ones you may expect to be close? Are there any surprises? You are encouraged to repeat this exercise with *cosine similarity* for determine the closest languages.

Exercise 6. *Hierarchical clustering*

Perform agglomerative clustering using *single*, *complete* and *average* linking based on the distances calculated in Exercise 5.

Plot a dendrogram for each linkage method, and save the resulting graphics as `dendrogram-single.pdf`, `dendrogram-complete.pdf` and `dendrogram-average.pdf`, respectively.[13]

[13] Are the differences between the linkage methods as you expected? Which method (k-means or agglomerative clustering) makes more sense for this problem? If you wanted the same number of clusters as gold-standard families, at what height would you need to 'cut' the dendrogram? Does the clusters agree with k-means? Are they better, or worse?