Teaching Demonstration: K-means Clustering

Ari Smith

Preliminary assumptions

- In context of a course on Machine Learning methods; students have seen supervised learning methods but not unsupervised learning
- Understand key ML terms such as *target, feature, observation*, *hyperparameter*
- Familiarity with Jupyter notebook, has sklearn and pandas properly installed
- Adapted from *Machine Learning in Action* by Prof. Justin Boutilier, itself partially adapted from *The Analytics Edge* by Dimitris Bertsimas

Overview

- Unsupervised learning
- The basics of clustering
- K-means clustering
  - Optimal approach
  - Heuristic approach: Lloyd’s algorithm
- How to use in practice
  - Example: Clustering news articles

Setup and Motivations

Supervised vs. Unsupervised learning

- **Supervised learning**: predict/explain a *target* variable given *feature* variables

- **Unsupervised learning**: learn patterns in observations with no target, only *feature* variables

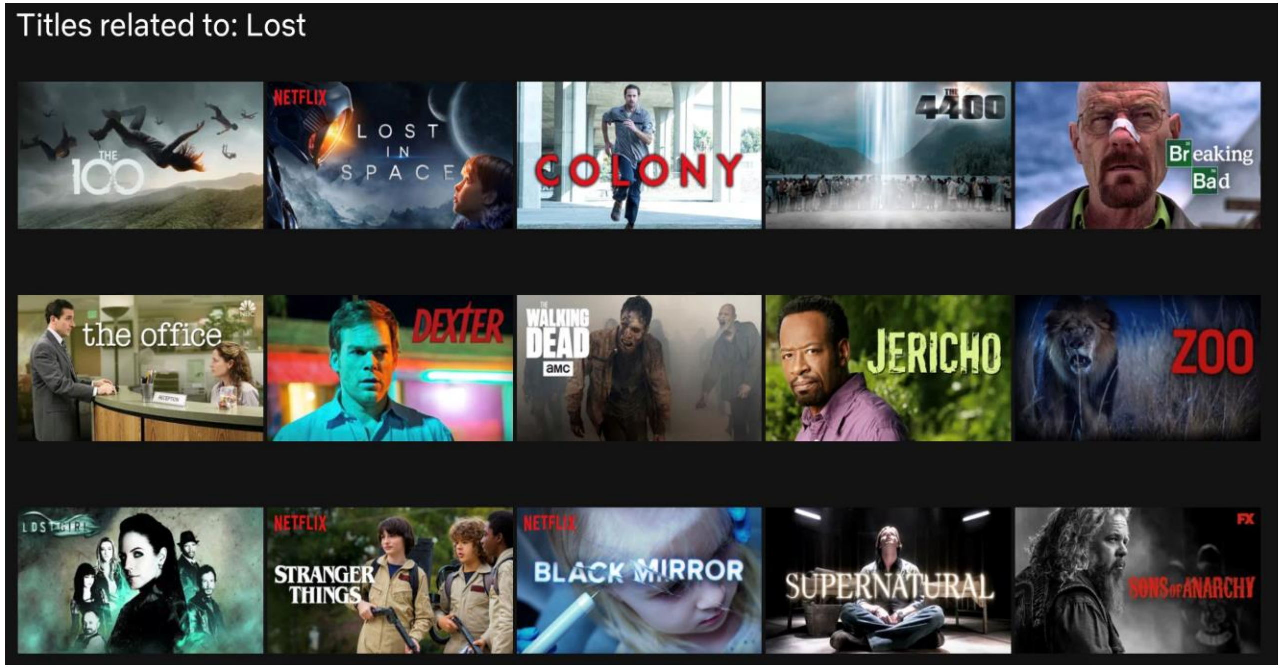
Unsupervised application: taste-communities

- Users of online platforms can often informally develop their own communities based on the content they interact with
- Ex. BookTok, PetTok, PerfumeTok
- Being able to recognize communities, and identify users as members of certain communities can be useful for recommending content

Unsupervised application: taste-communities

- In 2018, Netflix grouped users into ~1300 “taste-communities” to recommend content
  - Prior to 2016, recommendations were given uniformly by country of user

- Cluster 290:
  - Movies like: *Black Mirror, Lost,* and *Groundhog Day*



Clustering Basics

The goal of clustering

- Clustering is an unsupervised learning algorithm

- Partition data into clusters such that the observations within a cluster are similar to each other

How do we define “similar”?

We use *distance metrics* to measure similarity of two observations

Define an observation with *F* features: $\boldsymbol{x_i} = (x_{i1}, x_{i2}, \dots, x_{iF})^T$

A function $d(\boldsymbol{x_1}, \boldsymbol{x_2})$ is a distance metric if it satisfies the following criteria

- **Non-negativity:** $d(\boldsymbol{x_1}, \boldsymbol{x_2}) \geq 0$ *and* $d(\boldsymbol{x_1}, \boldsymbol{x_2}) = 0$ *iff* $\boldsymbol{x_1} = \boldsymbol{x_2}$
- **Symmetry:** $d(\boldsymbol{x_1}, \boldsymbol{x_2}) = d(\boldsymbol{x_2}, \boldsymbol{x_1})$
- **Triangle inequality:** $d(\boldsymbol{x_1}, \boldsymbol{x_2}) + d(\boldsymbol{x_2}, \boldsymbol{x_3}) \geq d(\boldsymbol{x_1}, \boldsymbol{x_3})$

Distance metrics

Euclidean:

$$d(\boldsymbol{x_1}, \boldsymbol{x_2}) = \|\boldsymbol{x_1} - \boldsymbol{x_2}\|_2 = \left(\sum_{f=1}^{F} \left|x_{1f} - x_{2f}\right|^2\right)^{\frac{1}{2}}$$

Manhattan:

$$d(\boldsymbol{x_1}, \boldsymbol{x_2}) = \|\boldsymbol{x_1} - \boldsymbol{x_2}\|_1 = \sum_{f=1}^{F} \left|x_{1f} - x_{2f}\right|$$

Chebyshev:

$$d(\boldsymbol{x_1}, \boldsymbol{x_2}) = \|\boldsymbol{x_1} - \boldsymbol{x_2}\|_\infty = \max_{f=1,\ldots,F} \left|x_{1f} - x_{2f}\right|$$

Defining a cluster

Index set: includes the IDs of all observations in a cluster

$$S_k = \{1,3,7,21,44\}$$

Centroid: the "center" or "representative point" of each cluster

$$\boldsymbol{s}_{\boldsymbol{k}} = \frac{1}{|S_k|} \sum_{i \in S_k} \boldsymbol{x}_{\boldsymbol{i}}$$

Cluster distances

Intra-cluster distance: distance between two points in the same cluster

Inter-cluster distance: distance between points in different clusters

K-means Clustering

K-means clustering

Partition observations into *k* clusters in a way that minimizes the total squared Euclidean distance between each observation and its cluster's centroid

- i.e., minimizes variance of observations within clusters

Hyperparameters

- k – number of clusters

K-means: optimization problem

Decision variables:

- a_{ij}: 1 if observation *i* assigned to cluster *j*, 0 otherwise
- $\boldsymbol{s}_j$: centroid of cluster *j*

Data:

- $\boldsymbol{x}_i$: observation *i*

$$\begin{aligned} \text{minimize} \quad & \textstyle\sum_i \sum_j a_{ij} \, ||\boldsymbol{x}_i - \boldsymbol{s}_j||_2^2 \\ \text{subject to} \quad & \textstyle\sum_j a_{ij} = 1, \qquad \forall i \\ & \textstyle\sum_i a_{ij} \geq 1, \qquad \forall j \\ & a_{ij} \in \{0, 1\}, \quad \forall i, j \\ & s_j \in \mathbb{R}^F, \qquad \forall j \end{aligned}$$

Heuristic Approach

- Optimization problem is *NP-Hard*, time to solve grows exponentially as dataset size increases (not practical for big data)

- Instead, a *heuristic* approach is most often used: Lloyd’s Algorithm

- Understanding how algorithm works has practical implications for using k-means clustering

Lloyd’s Algorithm

1. Randomly initialize k centroids

2. Assign each observation to its closest centroid using the distance metric

3. Recompute the cluster centroids as mean of assigned observations

4. If centroids do not change, stop. Otherwise, return to step 2.

Finds local optimum, but not always global optimum each time

Lloyd's Algorithm Pseudocode

$\boldsymbol{s_j} :=$ random_sample($\boldsymbol{x_1} \dots \boldsymbol{x_n}$)   for $j \in 1 \dots k$

While True:

  $S_j := \{\}$ for $j \in 1 \dots k$

  for $i \in 1 \dots n$:

    $c := \mathrm{argmin}_{j \in 1 \dots k}\, d(\boldsymbol{x_i}, \boldsymbol{s_j})$

    $S_c := S_c \cup \boldsymbol{x_i}$

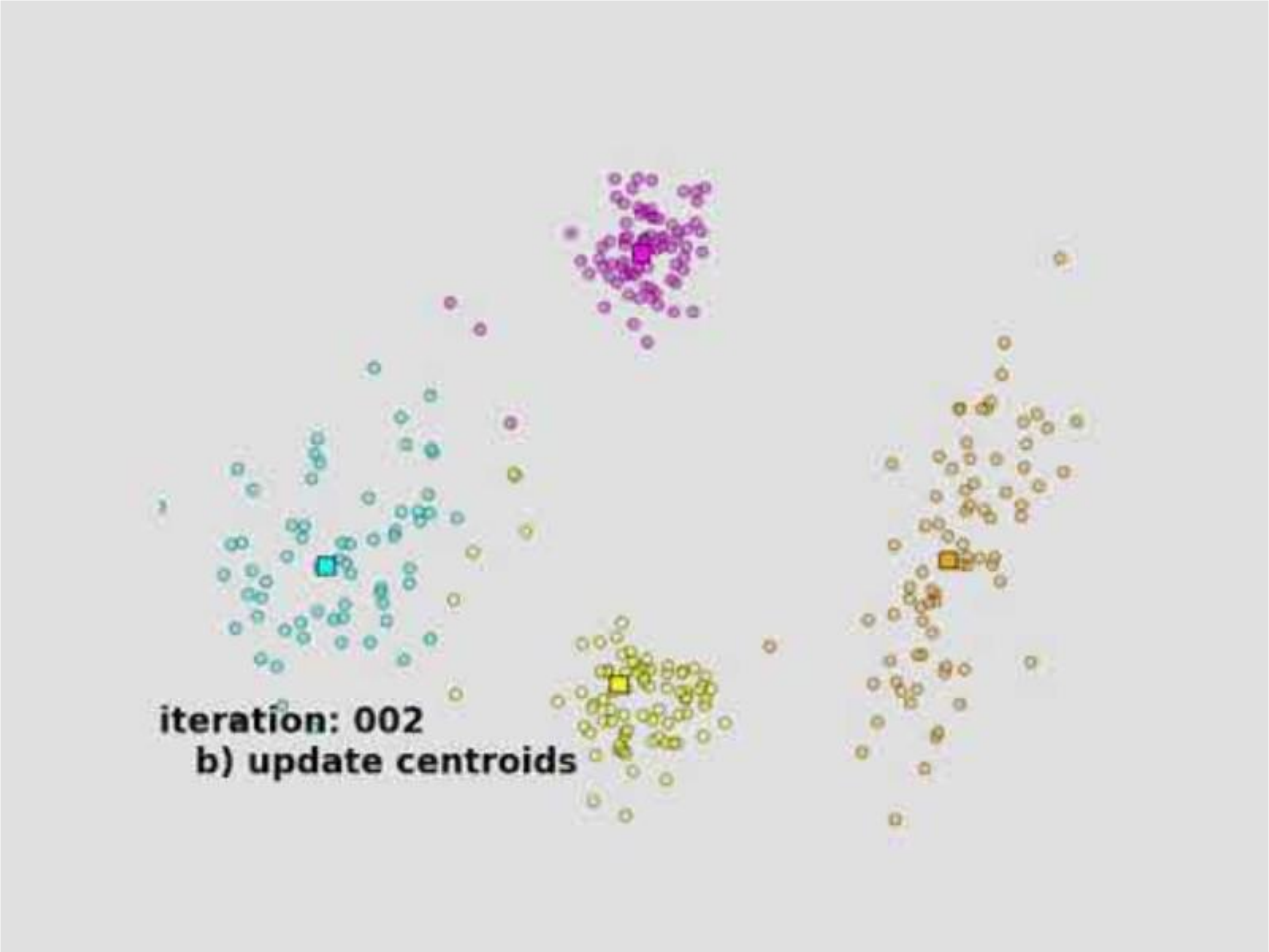
  $\boldsymbol{s_j^*} := \frac{1}{|S_j|} \sum_{i \in S_j} \boldsymbol{x_i}$ for $j \in 1 \dots k$

  if $\boldsymbol{s_j} == \boldsymbol{s_j^*}$ for all $j \in 1 \dots k$:

    return $S_j, \boldsymbol{s_j}$ for $j \in 1 \dots k$

  $\boldsymbol{s_j} := \boldsymbol{s_j^*}$

Lloyd’s Algorithm Example



Lloyd’s Algorithm

- In practice, terminates much faster than optimal formulation
- However, clusters are *local optima*, not *global optima*
- Clusters found depend on randomly chosen starting centroids
- **Repeat algorithm with multiple random starts** and keep best performing clusters, to maximize chance of finding global optima

Using k-means in practice

Hyperparameters

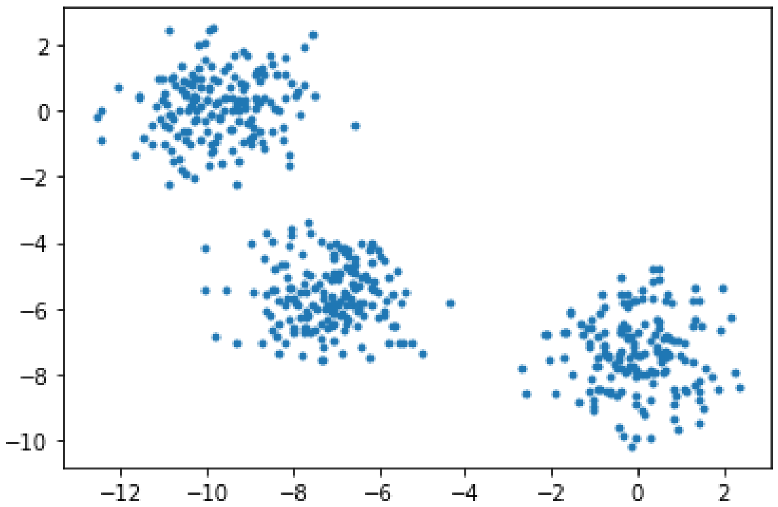
- k
- Number of repetitions with random start

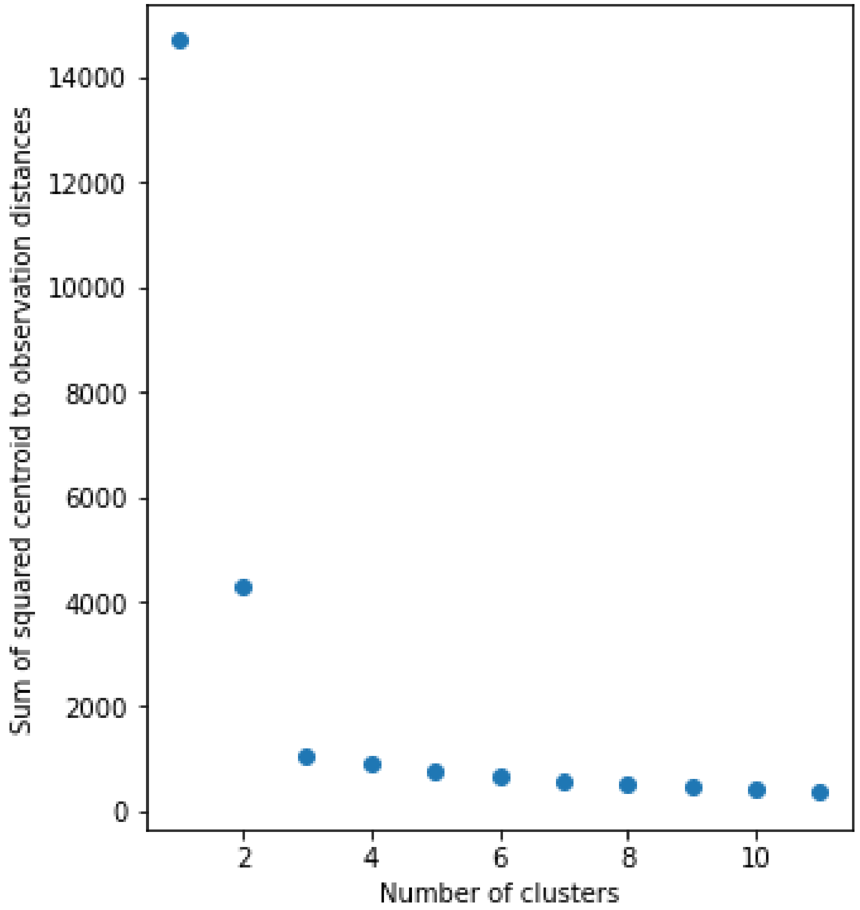
Features should be normalized/on a similar scale so some features don't have outsized impact of distance metric compared to others

Python package sklearn.cluster.Kmeans easy to use (only supports Euclidean)

How do we select *k*?

- Elbow plot method
- Plot total squared intra-cluster distances for many values of *k*
- Find “elbow” where decreases in total distance become marginal

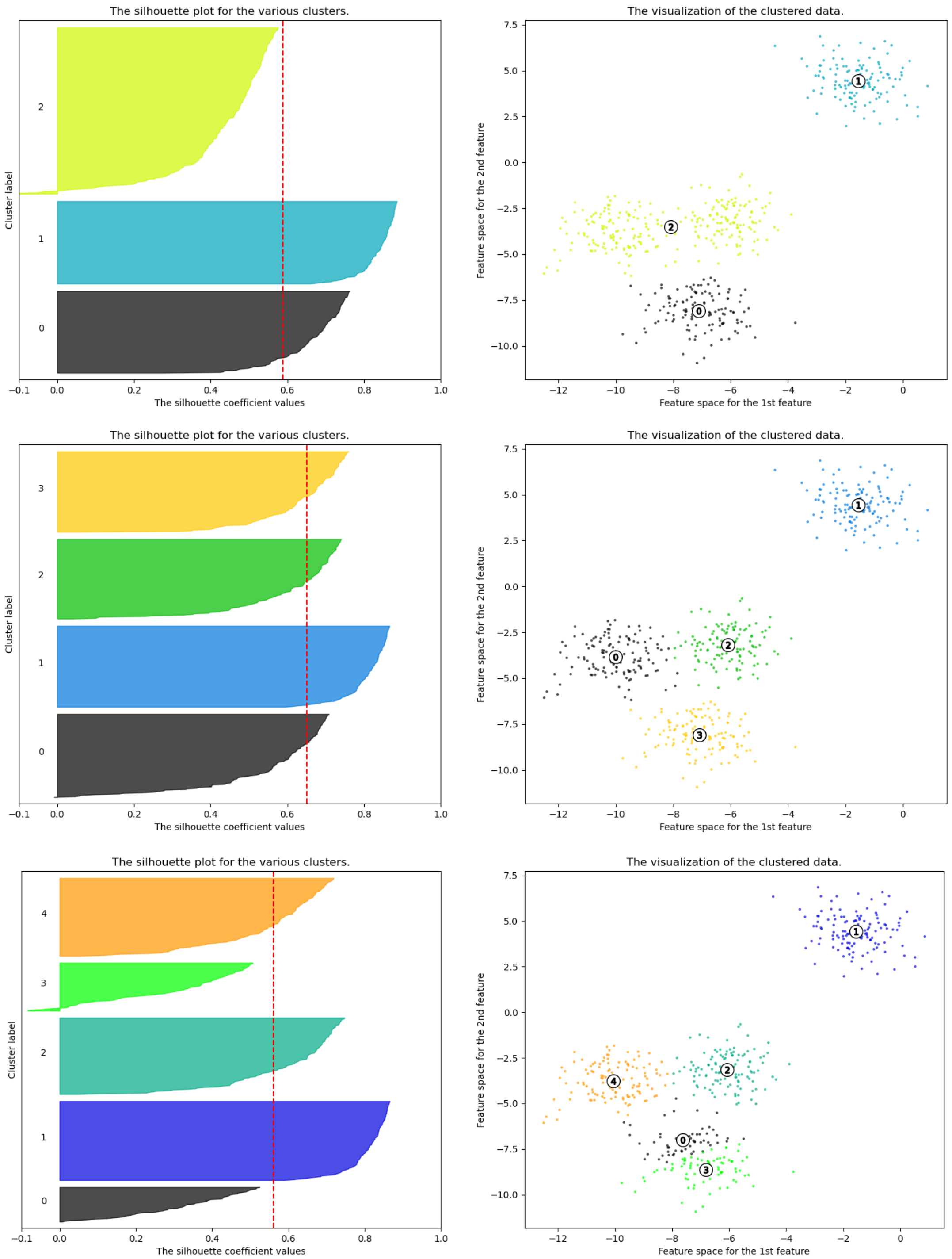




Silhouette Score Method

- Scores observations by how close they are to other members of cluster against closeness to next nearest cluster

- Clusters with low mean scores either:
- can be broken into more clusters (k too low)
- are not meaningfully distinct from another cluster (k too high)



Practical Example: DailyKos

Overview

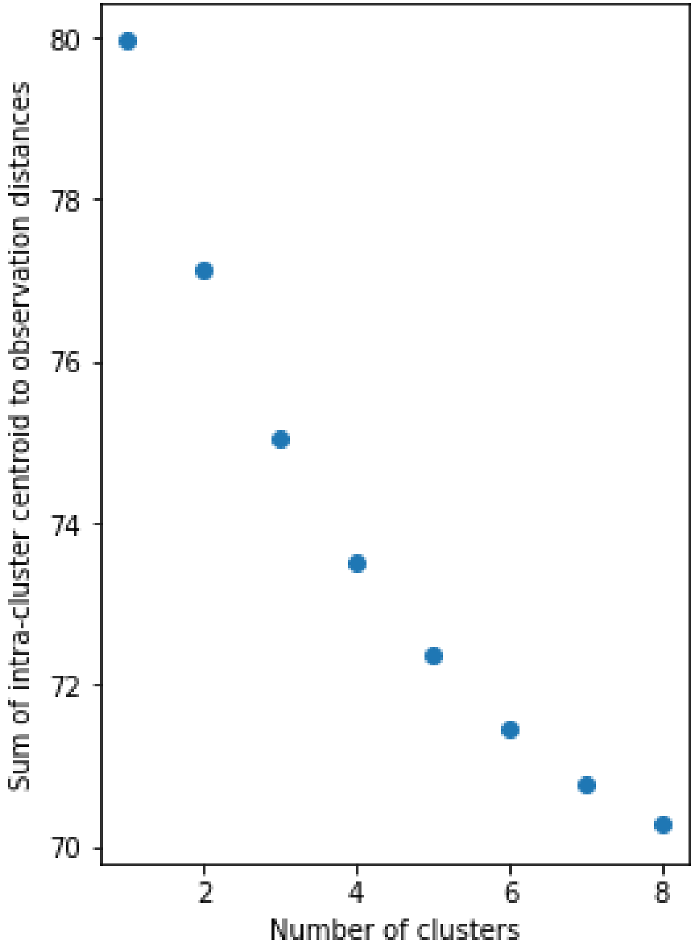
Internet blog, forum, and news site devoted to the Democratic Party and liberal politics

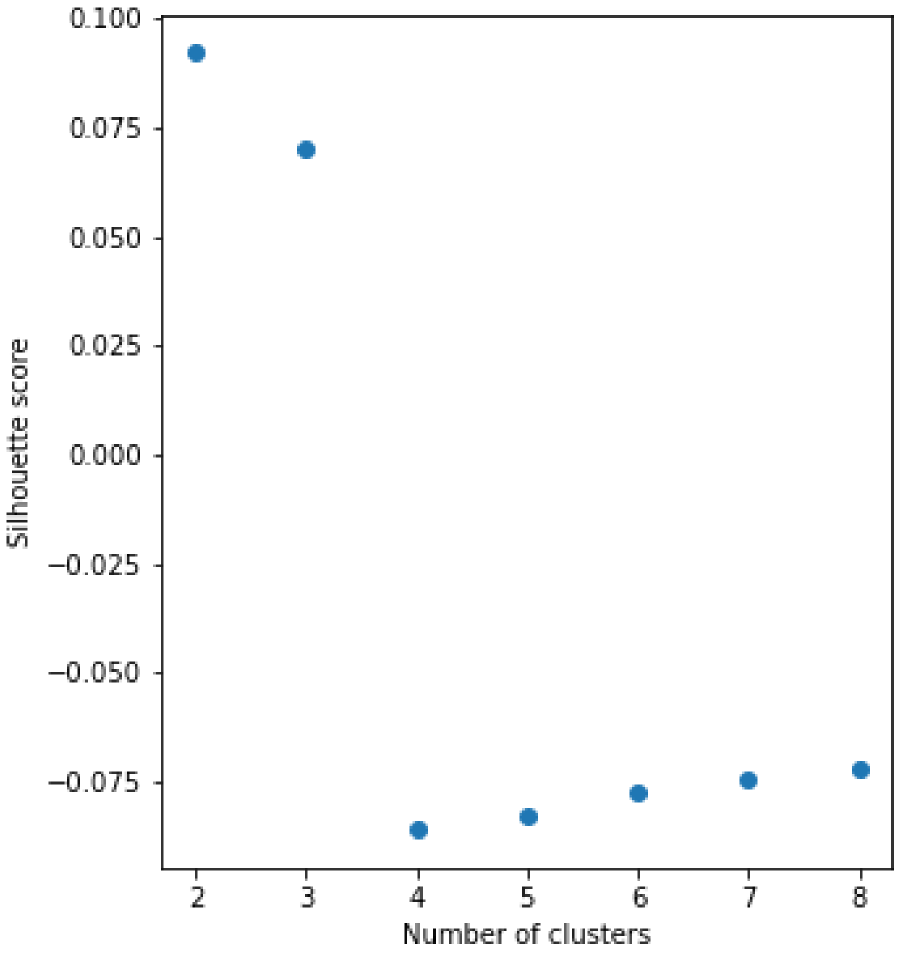
Obtained 3430 articles with 1545 features from Fall 2004

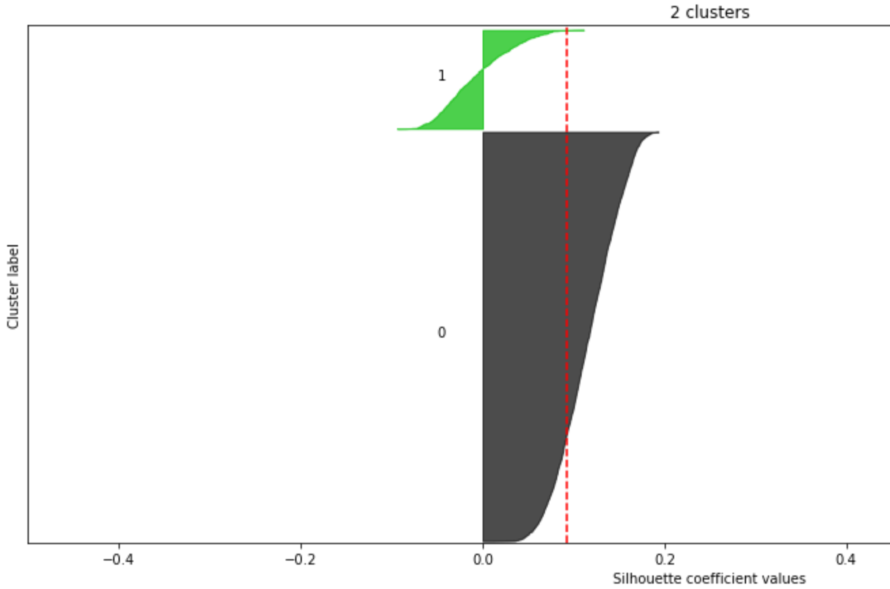
- Each feature is a variable corresponding to the number of times a word appears

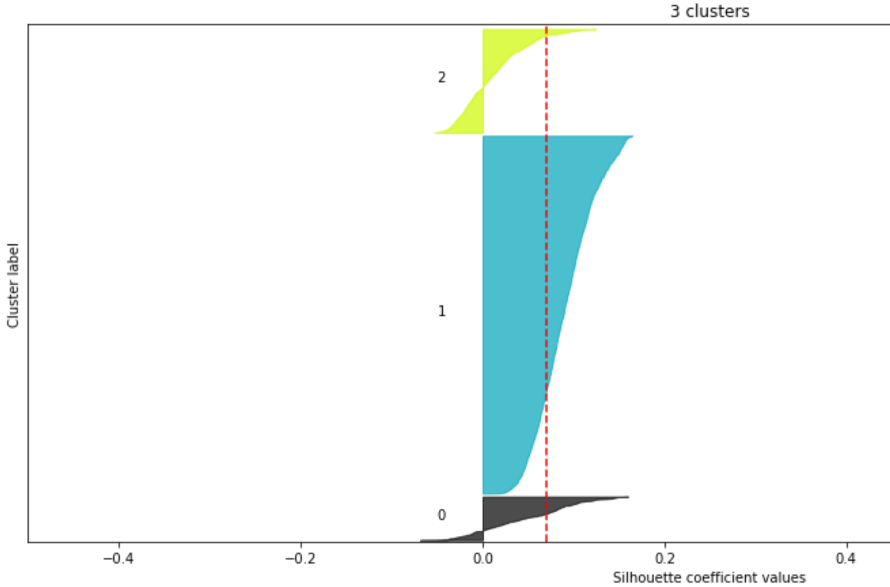
What were the hot topics on DailyKos at the time?

Elbow Plot and Silhouette Plots









Centroid Analysis

- Too many features to look at entire centroids
- Examine 10 largest features of each centroid
  - Most popular words in articles within cluster

cluster 0

dean	0.049536
poll	0.042395
kerry	0.042108
edward	0.029633
primaries	0.028711
clark	0.027232
democrat	0.026924
lieberman	0.015307
gephardt	0.015182
result	0.011227

cluster 1

democrat	0.012004
bush	0.011225
republican	0.010322
elect	0.009296
poll	0.009140
senate	0.007929
house	0.007766
november	0.007584
state	0.007276
vote	0.007257

cluster 2

bush	0.067433
kerry	0.038338
poll	0.017919
general	0.013632
presided	0.012335
administration	0.008843
democrat	0.007560
iraq	0.007437
campaign	0.007400
state	0.007386

Next Lesson

- Hierarchical/Agglomerative Clustering
- Linkage Criteria
- Interpreting Dendrograms
- Application to DailyKos Dataset, comparison with k-means clusters

WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON

Extra Slides

Breakdown of Optimal Formulation

Decision variables:

- a_{ij}: 1 if observation *i* assigned to cluster *j,* 0 otherwise
- $\boldsymbol{s}_j$: centroid of cluster *j*

Data:

- $\boldsymbol{x}_i$: observation *i*

$$\text{minimize} \quad \boxed{\sum_i \sum_j a_{ij} \, ||\boldsymbol{x}_i - \boldsymbol{s}_j||_2^2}$$

$$\begin{aligned} \text{subject to} \quad & \boxed{\sum_j a_{ij} = 1, \qquad \forall i} \\ & \boxed{\sum_i a_{ij} \geq 1, \qquad \forall j} \\ & \boxed{\begin{aligned} a_{ij} &\in \{0, 1\}, && \forall i, j \\ s_j &\in \mathbb{R}^F, && \forall j \end{aligned}} \end{aligned}$$

Total distance between observations and assigned centroids is minimized
All observations are assigned to exactly one cluster
Each cluster is non-empty
Variables are properly defined

Informal proof that Lloyd's alg finds local optimum

- Total distances (aka loss function) is a function of index sets and centroid coordinates
- For any fixed centroid coordinates, assigning observations to nearest centroids minimizes total distances
- For any fixed cluster index sets, setting centroids as mean feature values minimize total distances (for Euclidean metric)
- Loss function cannot increase at any iteration
- If index sets and centroids do not change in an iteration, partial derivative of loss function w.r.t. all input variables = 0, so a local extrema is found

Informal proof that Lloyd's alg terminates

- Index sets must change at every iteration, and cannot repeat (or else loss function would increase)

- There is a finite number of index set combinations (k^n)

- Therefore, Lloyd's algorithm must terminate in a finite number of iterations i (in practice usually significantly smaller than k^n, and does not usually increase rapidly with n)

- Each iteration is $O(knm)$, so full algorithm is $O(knmi)$