Unsupervised Models

- An unusual place to begin

Unsupervised Models

- An unusual place to begin
- Most data science classes focus on supervised models

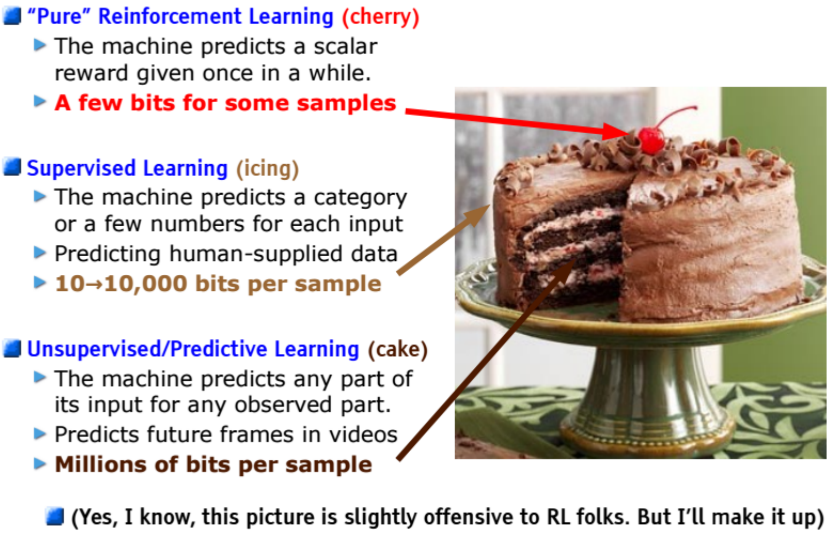
Unsupervised Models

- An unusual place to begin
- Most data science classes focus on supervised models
- Partially practicality

Unsupervised Models

- An unusual place to begin
- Most data science classes focus on supervised models
- Partially practicality
- Unsupervised models: the past and future of data

Unsupervised Models – Yann LeCunn



- **"Pure" Reinforcement Learning** (cherry)
  - The machine predicts a scalar reward given once in a while.
  - **A few bits for some samples**
- **Supervised Learning** (icing)
  - The machine predicts a category or a few numbers for each input
  - Predicting human-supplied data
  - **10→10,000 bits per sample**
- **Unsupervised/Predictive Learning** (cake)
  - The machine predicts any part of its input for any observed part.
  - Predicts future frames in videos
  - **Millions of bits per sample**
- (Yes, I know, this picture is slightly offensive to RL folks. But I'll make it up)

Quantities for comparing texts

Length in characters, words, lines, sentences, paragraphs, pages, sections, chapters, etc.

Readability statistics Use a combination of syllables and sentence length to indicate “readability” in terms of complexity

Vocabulary diversity (At its simplest) involves measuring a *type-to-token ratio* (TTR) where unique words are types and the total words are tokens

Word (relative) frequency counts or proportions of words

Lexical Diversity

- Basic measure is the TTR: Type-to-Token ratio
- Problem: This is very sensitive to overall document length, as shorter texts may exhibit fewer word repetitions
- Special problem: length may relate to the introdution of additional subjects, which will also increase richness

Complexity and Readability

- Use a combination of syllables and sentence length to indicate "readability" in terms of complexity
- Common in educational research, but could also be used to describe textual complexity
- Most use some sort of sample
- No natural scale, so most are calibrated in terms of some interpretable metric
- Implemented in **quanteda** as `textstat_readability()`

Flesch-Kincaid readability index

- F-K is a modification of the original Flesch Reading Ease Index:

  $$206.835 - 1.015\left(\frac{\text{total words}}{\text{total sentences}}\right) - 84.6\left(\frac{\text{total syllables}}{\text{total words}}\right)$$

  Interpretation: 0-30: university level; 60-70: understandable by 13-15 year olds; and 90-100 easily understood by an 11-year old student.

Flesch-Kincaid readability index

- F-K is a modification of the original Flesch Reading Ease Index:

$$206.835 - 1.015\left(\frac{\text{total words}}{\text{total sentences}}\right) - 84.6\left(\frac{\text{total syllables}}{\text{total words}}\right)$$

  Interpretation: 0-30: university level; 60-70: understandable by 13-15 year olds; and 90-100 easily understood by an 11-year old student.

- Flesch-Kincaid rescales to the US educational grade levels (1–12):

$$0.39\left(\frac{\text{total words}}{\text{total sentences}}\right) + 11.8\left(\frac{\text{total syllables}}{\text{total words}}\right) - 15.59$$

Documents as vectors

- The idea is that (weighted) features form a vector for each document, and that these vectors can be judged using metrics of similarity
- A document's vector for us is simply (for us) the row of the document-feature matrix

Characteristics of similarity measures

Let A and B be any two documents in a set and $d(A, B)$ be the distance between A and B.

1. $d(x, y) \geq 0$ (the distance between any two points must be non-negative)
2. $d(A, B) = 0$ iff $A = B$ (the distance between two documents must be zero if and only if the two objects are identical)
3. $d(A, B) = d(B, A)$ (distance must be symmetric: A to B is the same distance as from B to A)
4. $d(A, C) \leq d(A, B) + d(B, C)$ (the measure must satisfy the triangle inequality)

Euclidean distance

Between document A and B where j indexes their features, where y_{ij} is the value for feature j of document i

- Euclidean distance is based on the Pythagorean theorem
- Formula

$$\sqrt{\sum_{j=1}^{j}(y_{Aj} - y_{Bj})^2} \tag{1}$$

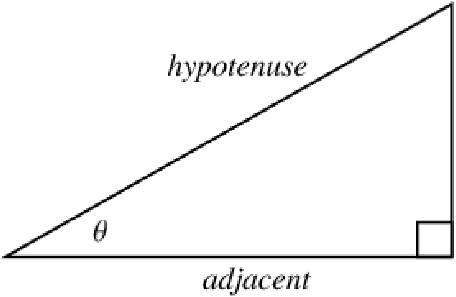
- In vector notation:

$$\|\mathbf{y}_A - \mathbf{y}_B\| \tag{2}$$

- Can be performed for any number of features J (or V as the vocabulary size is sometimes called – the number of columns in of the dfm, same as the number of feature types in the corpus)

A geometric interpretation of “distance”

In a right angled triangle, the cosine of an angle θ or $\cos(\theta)$ is the length of the adjacent side divided by the length of the hypotenuse



We can use the vectors to represent the text location in a V-dimensional vector space and compute the angles between them

Cosine similarity

- Cosine distance is based on the size of the angle between the vectors
- Formula
$$\frac{\mathbf{y}_A \cdot \mathbf{y}_B}{\|\mathbf{y}_A\| \|\mathbf{y}_B\|} \tag{3}$$
- The $\cdot$ operator is the dot product, or $\sum_j y_{Aj} y_{Bj}$
- The $\|\mathbf{y}_A\|$ is the vector norm of the (vector of) features vector $\mathbf{y}$ for document A, such that $\|\mathbf{y}_A\| = \sqrt{\sum_j y_{Aj}^2}$

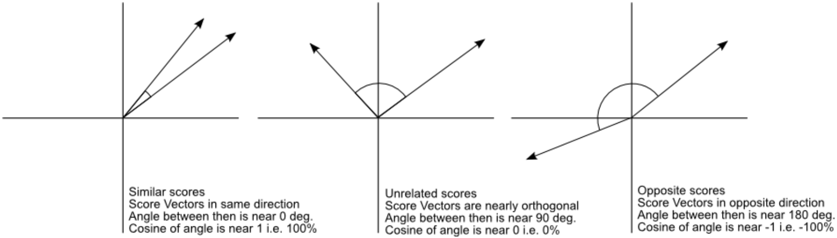
Cosine similarity

- Cosine distance is based on the size of the angle between the vectors
- Formula

$$\frac{\mathbf{y}_A \cdot \mathbf{y}_B}{\|\mathbf{y}_A\| \|\mathbf{y}_B\|} \tag{3}$$

- The $\cdot$ operator is the dot product, or $\sum_j y_{Aj} y_{Bj}$
- The $\|\mathbf{y}_A\|$ is the vector norm of the (vector of) features vector $\mathbf{y}$ for document A, such that $\|\mathbf{y}_A\| = \sqrt{\sum_j y_{Aj}^2}$
- Nice property: independent of document length, because it deals only with the angle of the vectors
- Ranges from -1.0 to 1.0 for term frequencies, or 0 to 1.0 for normalized term frequencies (or tf-idf)

Cosine similarity illustrated



Similar scores
Score Vectors in same direction
Angle between then is near 0 deg.
Cosine of angle is near 1 i.e. 100%

Unrelated scores
Score Vectors are nearly orthogonal
Angle between then is near 90 deg.
Cosine of angle is near 0 i.e. 0%

Opposite scores
Score Vectors in opposite direction
Angle between then is near 180 deg.
Cosine of angle is near -1 i.e. -100%

Example text

Hurricane Gilbert swept toward the Dominican Republic Sunday , and the Civil Defense alerted its heavily populated south coast to prepare for high **winds**, heavy **rains** and high seas.

The **storm** was approaching from the southeast with sustained **winds** of 75 mph gusting to 92 mph .

“There is no need for alarm," Civil Defense Director Eugenio Cabral said in a television alert shortly before midnight Saturday .

Cabral said residents of the province of Barahona should closely follow **Gilbert** 's movement .

An estimated 100,000 people live in the province, including 70,000 in the city of Barahona , about 125 miles west of Santo Domingo .

Tropical **Storm Gilbert** formed in the eastern Caribbean and strengthened into a **hurricane** Saturday night

The National **Hurricane** Center in Miami reported its position at 2a.m. Sunday at latitude 16.1 north , longitude 67.5 west, about 140 miles south of Ponce, Puerto Rico, and 200 miles southeast of Santo Domingo.

The National Weather Service in San Juan , Puerto Rico , said **Gilbert** was moving westward at 15 mph with a "broad area of cloudiness and heavy weather" rotating around the center of the **storm**.

The weather service issued a flash flood watch for Puerto Rico and the Virgin Islands until at least 6p.m. Sunday.

Strong **winds** associated with the **Gilbert** brought coastal flooding , strong southeast **winds** and up to 12 feet to Puerto Rico 's south coast.

Example text: selected terms

- Document 1

  Gilbert: 3, hurricane: 2, rains: 1, storm: 2, winds: 2

- Document 2

  Gilbert: 2, hurricane: 1, rains: 0, storm: 1, winds: 2

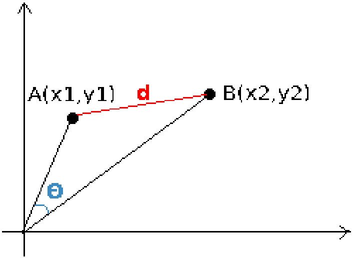
Example text: cosine similarity in R

```
toyDfm <- as.dfm(matrix(c(3,2,1,2,2, 2,1,0,1,2),
nrow = 2, byrow = TRUE))
colnames(toyDfm) <- c("Gilbert", "hurricane", "rain", "storm", "winds")
toyDfm
## Document-feature matrix of: 2 documents, 5 features (10% sparse).
## 2 x 5 sparse Matrix of class "dfm"
##        features
## docs    Gilbert hurricane rain storm winds
##   text1       3         2    1     2     2
##   text2       2         1    0     1     2

textstat_simil(toyDfm, method = "cosine")
##           text1
## text2 0.9438798
```

Relationship to Euclidean distance

- Cosine similarity measures the similarity of vectors with respect to the origin
- Euclidean distance measures the distance between particular points of interest along the vector



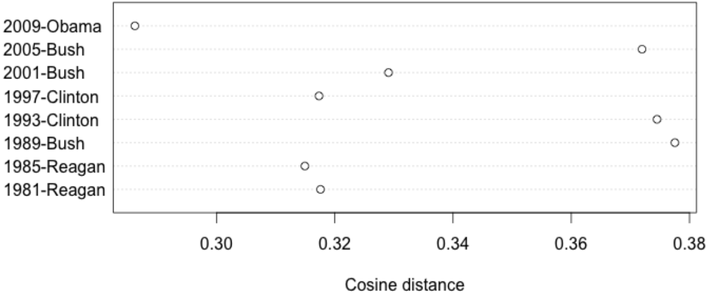
Jacquard coefficient

- Similar to the Cosine similarity
- Formula

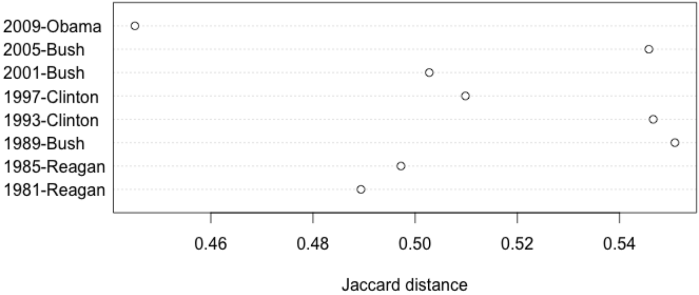
$$\frac{\mathbf{y}_A \cdot \mathbf{y}_B}{\|\mathbf{y}_A\| + \|\mathbf{y}_B\| - \mathbf{y}_A \cdot \mathbf{y}_B} \tag{4}$$

- Ranges from 0 to 1.0

Example: Inaugural speeches



Example: Inaugural speeches



Typical features

- Normalized term frequency (almost certainly)
- Very common to use tf-idf – if not, similarity is boosted by common words (stop words)
- Not as common to use binary features

Weighting strategies for feature counting

term frequency Some approaches trim very low-frequency words. Rationale: get rid of rare words that expand the feature matrix but matter little to substantive analysis

document frequency Could eliminate words appearing in few documents

inverse document frequency Conversely, could weight words more that appear in the most documents

tf-idf a combination of term frequency and inverse document frequency, common method for feature weighting

Strategies for feature *weighting*: tf-idf

- $tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}}$ where $n_{i,j}$ is number of occurences of term t_i in document d_j, k is total number of terms in document d_j

- $idf_i = \ln \frac{|D|}{|\{d_j : t_i \in d_j\}|}$
  where
  - $|D|$ is the total number of documents in the set
  - $|\{d_j : t_i \in d_j\}|$ is the number of documents where the term t_i appears (i.e. $n_{i,j} \neq 0$)

- $tf\text{-}idf_i = tf_{i,j} \cdot idf_i$

Computation of tf-idf: Example

Example: We have 100 political party manifestos, each with 1000 words. The first document contains 16 instances of the word "environment"; 40 of the manifestos contain the word "environment".

- The *term frequency* is $16/1000 = 0.016$
- The *document frequency* is $100/40 = 2.5$, or $\ln(2.5) = 0.916$
- The *tf-idf* will then be $0.016 * 0.916 = 0.0147$

Computation of tf-idf: Example

Example: We have 100 political party manifestos, each with 1000 words. The first document contains 16 instances of the word "environment"; 40 of the manifestos contain the word "environment".

- The *term frequency* is $16/1000 = 0.016$
- The *document frequency* is $100/40 = 2.5$, or $\ln(2.5) = 0.916$
- The *tf-idf* will then be $0.016 * 0.916 = 0.0147$
- If the word had only appeared in 15 of the 100 manifestos, then the *tf-idf* would be 0.0304 (three times higher).

Computation of tf-idf: Example

Example: We have 100 political party manifestos, each with 1000 words. The first document contains 16 instances of the word "environment"; 40 of the manifestos contain the word "environment".

- The *term frequency* is $16/1000 = 0.016$
- The *document frequency* is $100/40 = 2.5$, or $\ln(2.5) = 0.916$
- The *tf-idf* will then be $0.016 * 0.916 = 0.0147$
- If the word had only appeared in 15 of the 100 manifestos, then the *tf-idf* would be 0.0304 (three times higher).
- A high weight in tf-idf is reached by a high term frequency (in the given document) and a low document frequency of the term in the whole collection of documents; hence the weights hence tend to filter out common terms

The idea of ”clusters”

- Essentially: groups of items such that inside a cluster they are very similar to each other, but very different from those outside the cluster
- “unsupervised classification”: cluster is not to relate features to classes or latent traits, but rather to estimate membership of distinct groups

The idea of ”clusters”

- Essentially: groups of items such that inside a cluster they are very similar to each other, but very different from those outside the cluster
- “unsupervised classification”: cluster is not to relate features to classes or latent traits, but rather to estimate membership of distinct groups
- groups are given labels through post-estimation interpretation of their elements
- typically used when we do not and never will know the “true” class labels
- issues: how to weight distance is arbitrary

The idea of ”clusters”

- Essentially: groups of items such that inside a cluster they are very similar to each other, but very different from those outside the cluster
- “unsupervised classification”: cluster is not to relate features to classes or latent traits, but rather to estimate membership of distinct groups
- groups are given labels through post-estimation interpretation of their elements
- typically used when we do not and never will know the “true” class labels
- issues: how to weight distance is arbitrary
    - which dimensionality? (determined by which features are selected)

The idea of ”clusters”

- Essentially: groups of items such that inside a cluster they are very similar to each other, but very different from those outside the cluster
- “unsupervised classification”: cluster is not to relate features to classes or latent traits, but rather to estimate membership of distinct groups
- groups are given labels through post-estimation interpretation of their elements
- typically used when we do not and never will know the “true” class labels
- issues: how to weight distance is arbitrary
  - which dimensionality? (determined by which features are selected)
  - how to weight distance is arbitrary

The idea of ”clusters”

- Essentially: groups of items such that inside a cluster they are very similar to each other, but very different from those outside the cluster
- “unsupervised classification”: cluster is not to relate features to classes or latent traits, but rather to estimate membership of distinct groups
- groups are given labels through post-estimation interpretation of their elements
- typically used when we do not and never will know the “true” class labels
- issues: how to weight distance is arbitrary
  - which dimensionality? (determined by which features are selected)
  - how to weight distance is arbitrary
  - different metrics for distance

k-means clustering

- Essence: assign each item to one of *k* clusters, where the goal is to minimised within-cluster difference and maximize between-cluster differences
- Uses random starting positions and iterates until stable
- as with *kNN*, *k*-means clustering treats feature values as coordinates in a multi-dimensional space

k-means clustering

- Essence: assign each item to one of *k* clusters, where the goal is to minimised within-cluster difference and maximize between-cluster differences
- Uses random starting positions and iterates until stable
- as with *kNN*, *k*-means clustering treats feature values as coordinates in a multi-dimensional space
- Advantages
  - simplicity
  - highly flexible
  - efficient

k-means clustering

- Essence: assign each item to one of *k* clusters, where the goal is to minimised within-cluster difference and maximize between-cluster differences
- Uses random starting positions and iterates until stable
- as with *kNN*, *k*-means clustering treats feature values as coordinates in a multi-dimensional space
- Advantages
    - simplicity
    - highly flexible
    - efficient
- Disadvantages
    - no fixed rules for determining *k*
    - uses an element of randomness for starting values

algorithm details

algorithm details

1. Choose starting values

algorithm details

1. Choose starting values
    - assign random positions to k starting values that will serve as the "cluster centres", known as "centroids" ; or,
    - assign each feature randomly to one of k classes

algorithm details

1. Choose starting values
    - assign random positions to k starting values that will serve as the "cluster centres", known as "centroids" ; or,
    - assign each feature randomly to one of k classes
2. assign each item to the class of the centroid that is "closest"
    - Euclidean distance is most common
    - any others may also be used (Manhattan, Minkowski, Mahalanobis, etc.)
    - (assumes feature vectors have been normalised within item)

algorithm details

1. Choose starting values
    - assign random positions to k starting values that will serve as the "cluster centres", known as "centroids" ; or,
    - assign each feature randomly to one of k classes
2. assign each item to the class of the centroid that is "closest"
    - Euclidean distance is most common
    - any others may also be used (Manhattan, Minkowski, Mahalanobis, etc.)
    - (assumes feature vectors have been normalised within item)
3. update: recompute the cluster centroids as the mean value of the points assigned to that cluster

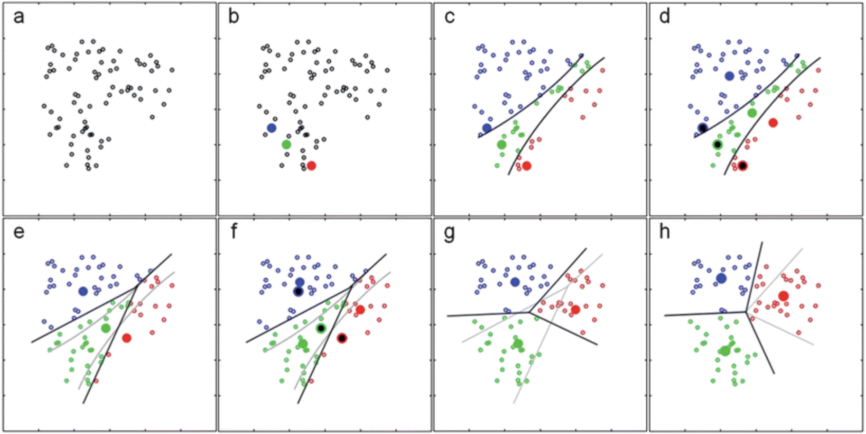
algorithm details

1. Choose starting values
    - assign random positions to k starting values that will serve as the "cluster centres", known as "centroids" ; or,
    - assign each feature randomly to one of k classes
2. assign each item to the class of the centroid that is "closest"
    - Euclidean distance is most common
    - any others may also be used (Manhattan, Minkowski, Mahalanobis, etc.)
    - (assumes feature vectors have been normalised within item)
3. update: recompute the cluster centroids as the mean value of the points assigned to that cluster
4. repeat reassignment of points and updating centroids

algorithm details

1. Choose starting values
   - assign random positions to k starting values that will serve as the "cluster centres", known as "centroids" ; or,
   - assign each feature randomly to one of k classes
2. assign each item to the class of the centroid that is "closest"
   - Euclidean distance is most common
   - any others may also be used (Manhattan, Minkowski, Mahalanobis, etc.)
   - (assumes feature vectors have been normalised within item)
3. update: recompute the cluster centroids as the mean value of the points assigned to that cluster
4. repeat reassignment of points and updating centroids
5. repeat 2–4 until some stopping condition is satisfied
   - e.g. when no items are reclassified following update of centroids

k-means clustering illustrated

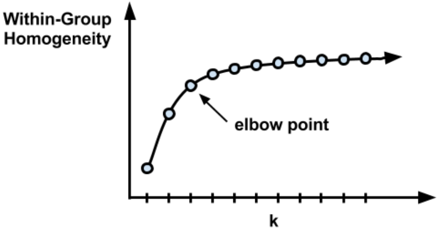


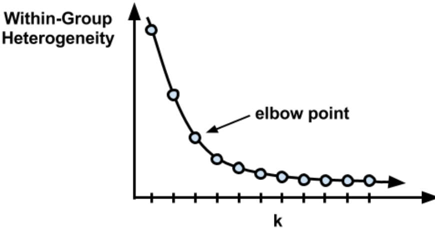
choosing the appropriate number of clusters

- very often based on prior information about the number of categories sought
  - for example, you need to cluster people in a class into a fixed number of (like-minded) tutorial groups
- a (rough!) guideline: set $k = \sqrt{N/2}$ where N is the number of items to be classified
  - usually too big: setting k to large values will improve within-cluster similarity, but risks *overfitting*

choosing the appropriate number of clusters

- "elbow plots": fit multiple clusters with different k values, and choose k beyond which are diminishing gains





choosing the appropriate number of clusters

- “fit” statistics to measure homogeneity within clusters and heterogeneity in between
- Entropy
- Perplexity

Other clustering methods: hierarchical clustering

- *agglomerative*: works from the bottom up to create clusters

Other clustering methods: hierarchical clustering

- *agglomerative*: works from the bottom up to create clusters
- like *k*-means, usually involves *projection*: reducing the features through either selection or projection to a lower-dimensional representation
  1. local projection: reducing features within document
  2. global projection: reducing features across all documents (Schütze and Silverstein, 1997)

Other clustering methods: hierarchical clustering

- *agglomerative*: works from the bottom up to create clusters
- like *k*-means, usually involves *projection*: reducing the features through either selection or projection to a lower-dimensional representation
    1. local projection: reducing features within document
    2. global projection: reducing features across all documents (Schütze and Silverstein, 1997)
    3. SVD methods, such PCA on a normalised feature matrix
    4. usually simple threshold-based truncation is used
       (keep all but 100 highest frequency or tf-idf terms)

Other clustering methods: hierarchical clustering

- *agglomerative*: works from the bottom up to create clusters
- like *k*-means, usually involves *projection*: reducing the features through either selection or projection to a lower-dimensional representation
  1. local projection: reducing features within document
  2. global projection: reducing features across all documents (Schütze and Silverstein, 1997)
  3. SVD methods, such PCA on a normalised feature matrix
  4. usually simple threshold-based truncation is used (keep all but 100 highest frequency or tf-idf terms)
- frequently/always involves weighting (normalising term frequency, tf-idf)

hierarchical clustering algorithm

1. start by considering each item as its own cluster, for n clusters

hierarchical clustering algorithm

1. start by considering each item as its own cluster, for n clusters
2. calculate the $N(N-1)/2$ pairwise distances between each of the n clusters, store in a matrix D_0

hierarchical clustering algorithm

1. start by considering each item as its own cluster, for n clusters
2. calculate the $N(N-1)/2$ pairwise distances between each of the n clusters, store in a matrix D_0
3. find smallest (off-diagonal) distance in D_0, and merge the items corresponding to the i, j indexes in D_0 into a new “cluster”

hierarchical clustering algorithm

1. start by considering each item as its own cluster, for n clusters
2. calculate the $N(N-1)/2$ pairwise distances between each of the n clusters, store in a matrix D_0
3. find smallest (off-diagonal) distance in D_0, and merge the items corresponding to the i, j indexes in D_0 into a new “cluster”
4. recalculate distance matrix D_1 with new cluster(s).

hierarchical clustering algorithm

1. start by considering each item as its own cluster, for n clusters
2. calculate the $N(N-1)/2$ pairwise distances between each of the n clusters, store in a matrix D_0
3. find smallest (off-diagonal) distance in D_0, and merge the items corresponding to the i, j indexes in D_0 into a new "cluster"
4. recalculate distance matrix D_1 with new cluster(s). options for determining the location of a cluster include:
    - centroids (mean)
    - most dissimilar objects
    - Ward's measure(s) based on minimising variance

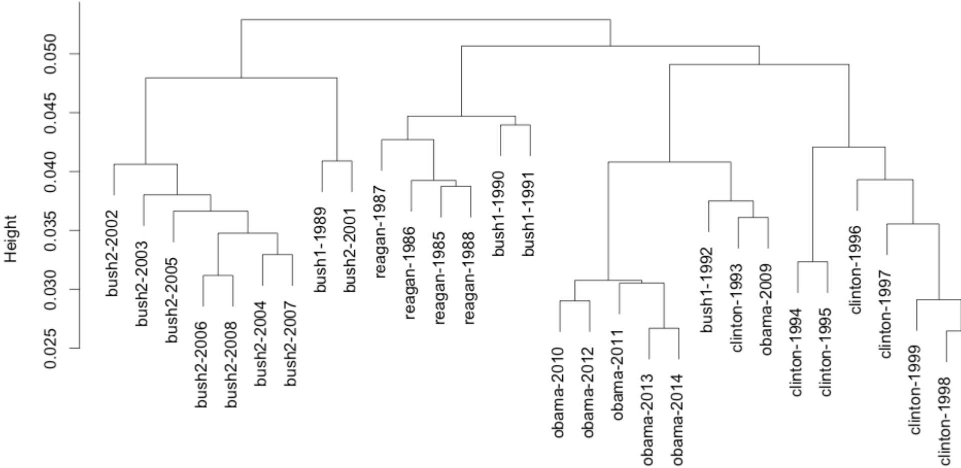
hierarchical clustering algorithm

1. start by considering each item as its own cluster, for n clusters
2. calculate the $N(N-1)/2$ pairwise distances between each of the n clusters, store in a matrix D_0
3. find smallest (off-diagonal) distance in D_0, and merge the items corresponding to the i, j indexes in D_0 into a new "cluster"
4. recalculate distance matrix D_1 with new cluster(s). options for determining the location of a cluster include:
    - centroids (mean)
    - most dissimilar objects
    - Ward's measure(s) based on minimising variance
5. repeat 3–4 until a stopping condition is reached
    - e.g. all items have been merged into a single cluster

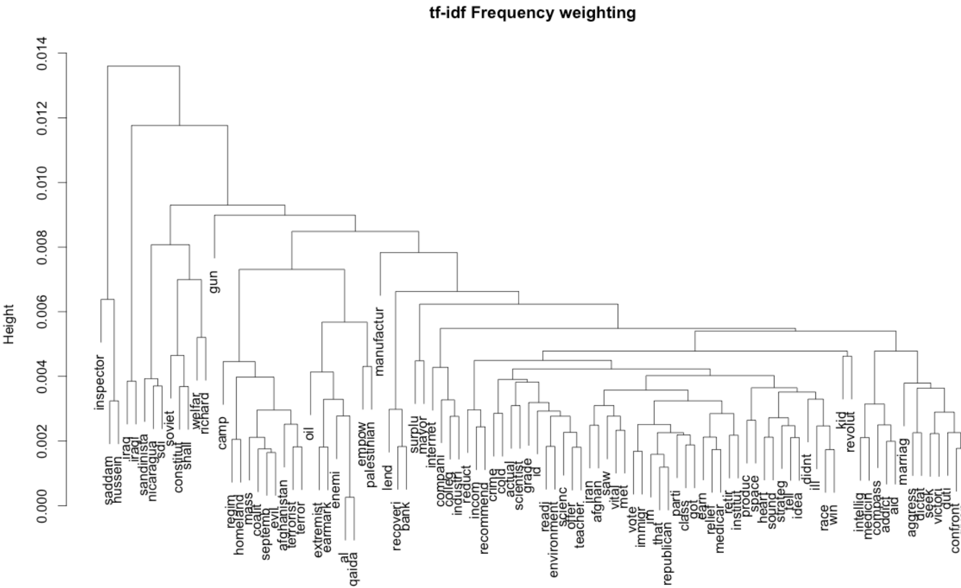
hierarchical clustering algorithm

1. start by considering each item as its own cluster, for n clusters
2. calculate the $N(N-1)/2$ pairwise distances between each of the n clusters, store in a matrix D_0
3. find smallest (off-diagonal) distance in D_0, and merge the items corresponding to the i, j indexes in D_0 into a new "cluster"
4. recalculate distance matrix D_1 with new cluster(s). options for determining the location of a cluster include:
    - centroids (mean)
    - most dissimilar objects
    - Ward's measure(s) based on minimising variance
5. repeat 3–4 until a stopping condition is reached
    - e.g. all items have been merged into a single cluster
6. to plot the *dendrograms*, need decisions on ordering, since there are $2^{(N-1)}$ possible orderings

Dendrogram: Presidential State of the Union addresses



Dendrogram: Presidential State of the Union addresses



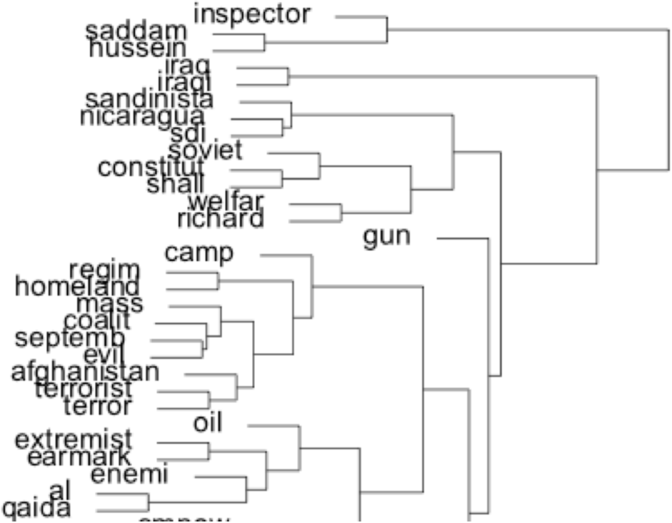
pros and cons of hierarchical clustering

- advantages
  - deterministic, unlike *k*-means
  - no need to decide on *k* in advance (although can specify as a stopping condition)
  - allows hierarchical relations to be examined (usually through *dendrograms*)

pros and cons of hierarchical clustering

- advantages
  - deterministic, unlike *k*-means
  - no need to decide on *k* in advance (although can specify as a stopping condition)
  - allows hierarchical relations to be examined
    (usually through *dendrograms*)
- disadvantages
  - more complex to compute: quadratic in complexity: $O(n^2)$
    – whereas *k*-means has complexity that is $O(n)$
  - the decision about where to create branches and in what order can be somewhat arbitrary, determined by method of declaring the “distance” to already formed clusters
  - for words, tends to identify collocations as base-level clusters (e.g. “saddam” and “hussein”)

Dendrogram: Presidential State of the Union addresses



Topic Models

- Topic models are algorithms for discovering the main "themes" in an unstructured corpus
- Requires no prior information, training set, or special annotation of the texts
  – only a decision on K (number of topics)
- A probabilistic, generative advance on several earlier methods, "Latent Semantic Analysis" (LSA) and "probabilistic latent semantic indexing" (pLSI)

Unsupervised methods scale distance

- Text gets converted into a quantitative matrix of features

Unsupervised methods scale distance

- Text gets converted into a quantitative matrix of features
  - words, typically

Unsupervised methods scale distance

- Text gets converted into a quantitative matrix of features
  - words, typically
  - could be dictionary entries, or parts of speech

Unsupervised methods scale distance

- Text gets converted into a quantitative matrix of features
  - words, typically
  - could be dictionary entries, or parts of speech
- Documents are scaled based on similarity/distance in feature use

Unsupervised methods scale distance

- Text gets converted into a quantitative matrix of features
  - words, typically
  - could be dictionary entries, or parts of speech
- Documents are scaled based on similarity/distance in feature use
- Fundamental problem: distance on which scale?

Unsupervised methods scale distance

- Text gets converted into a quantitative matrix of features
  - words, typically
  - could be dictionary entries, or parts of speech
- Documents are scaled based on similarity/distance in feature use
- Fundamental problem: distance on which scale?
  - Ideally, something we care about, e.g. policy positions, ideology, preferences, sentiment

Unsupervised methods scale distance

- Text gets converted into a quantitative matrix of features
  - words, typically
  - could be dictionary entries, or parts of speech
- Documents are scaled based on similarity/distance in feature use
- Fundamental problem: distance on which scale?
  - Ideally, something we care about, e.g. policy positions, ideology, preferences, sentiment
  - But often other dimensions (language, rhetoric style, authorship) are more predictive

Unsupervised methods scale distance

- Text gets converted into a quantitative matrix of features
  - words, typically
  - could be dictionary entries, or parts of speech
- Documents are scaled based on similarity/distance in feature use
- Fundamental problem: distance on which scale?
  - Ideally, something we care about, e.g. policy positions, ideology, preferences, sentiment
  - But often other dimensions (language, rhetoric style, authorship) are more predictive
- First dimension in unsupervised scaling will capture main source of variation, whatever that is

Unsupervised methods scale distance

- Text gets converted into a quantitative matrix of features
    - words, typically
    - could be dictionary entries, or parts of speech
- Documents are scaled based on similarity/distance in feature use
- Fundamental problem: distance on which scale?
    - Ideally, something we care about, e.g. policy positions, ideology, preferences, sentiment
    - But often other dimensions (language, rhetoric style, authorship) are more predictive
- First dimension in unsupervised scaling will capture main source of variation, whatever that is
- Unlike supervised models, validation comes after estimating the model

Correspondence Analysis

- CA is like factor analysis for categorical data
- Following normalization of the marginals, it uses Singular Value Decomposition to reduce the dimensionality of the document-feature matrix
- This allows projection of the positioning of the words as well as the texts into multi-dimensional space
- The number of dimensions – as in factor analysis – can be decided based on the eigenvalues from the SVD

Singular Value Decomposition

- A matrix $\underset{n \times k}{\mathbf{X}}$ can be represented in a dimensionality equal to its rank d as:

$$\underset{n \times k}{\mathbf{X}} = \underset{n \times d}{\mathbf{U}} \underset{d \times d}{\mathbf{\Sigma}} \underset{d \times k}{\mathbf{V}'} \tag{5}$$

- The $\mathbf{U}$, $\mathbf{\Sigma}$, and $\mathbf{V}$ matrixes "relocate" the elements of $\mathbf{X}$ onto new coordinate vectors in d-dimensional Euclidean space
- Row variables of $\mathbf{X}$ become points on the $\mathbf{U}$ column coordinates, and the column variables of $\mathbf{X}$ become points on the $\mathbf{V}$ column coordinates
- The coordinate vectors are perpendicular (*orthogonal*) to each other and are normalized to unit length

Correspondence analysis

1. Compute matrix of standardized residuals, $\mathbf{S}$:

$$\mathbf{S} = \mathbf{D}_r^{1/2}(\mathbf{P} - \mathbf{r}\mathbf{c}^T)\mathbf{D}_c^{1/2}$$

where $\mathbf{P} = \mathbf{Y} / \sum_{ij} y_{ij}$

$\mathbf{r}$, $\mathbf{c}$ are row/column masses: e.g. $r_i = \sum_j p_{ij}$

$\mathbf{D}_r = \mathrm{diag}(\mathbf{r})$, $\mathbf{D}_c = \mathrm{diag}(\mathbf{c})$

Correspondence analysis

1. Compute matrix of standardized residuals, **S**:
$$\mathbf{S} = \mathbf{D}_r^{1/2}(\mathbf{P} - \mathbf{r}\mathbf{c}^T)\mathbf{D}_c^{1/2}$$
   where $\mathbf{P} = \mathbf{Y} / \sum_{ij} y_{ij}$
   r, **c** are row/column masses: e.g. $r_i = \sum_j p_{ij}$
   $\mathbf{D}_r = \mathrm{diag}(\mathbf{r})$, $\mathbf{D}_c = \mathrm{diag}(\mathbf{c})$
2. Calculate SVD of **S**

Correspondence analysis

1. Compute matrix of standardized residuals, **S**:

$$\mathbf{S} = \mathbf{D}_r^{1/2}(\mathbf{P} - \mathbf{r}\mathbf{c}^T)\mathbf{D}_c^{1/2}$$

   where $\mathbf{P} = \mathbf{Y} / \sum_{ij} y_{ij}$

   $\mathbf{r}$, $\mathbf{c}$ are row/column masses: e.g. $r_i = \sum_j p_{ij}$

   $\mathbf{D}_r = \mathrm{diag}(\mathbf{r})$, $\mathbf{D}_c = \mathrm{diag}(\mathbf{c})$

2. Calculate SVD of **S**

3. Project rows and columns onto low-dimensional space:

$$\theta = \mathbf{D}_r^{1/2}\mathbf{U} \text{ for rows (documents)}$$

$$\phi = \mathbf{D}_c^{1/2}\mathbf{V} \text{ for columns (words)}$$

Correspondence analysis

1. Compute matrix of standardized residuals, **S**:

$$\mathbf{S} = \mathbf{D}_r^{1/2}(\mathbf{P} - \mathbf{r}\mathbf{c}^T)\mathbf{D}_c^{1/2}$$

where $\mathbf{P} = \mathbf{Y} / \sum_{ij} y_{ij}$

$\mathbf{r}$, $\mathbf{c}$ are row/column masses: e.g. $r_i = \sum_j p_{ij}$

$\mathbf{D}_r = \mathrm{diag}(\mathbf{r})$, $\mathbf{D}_c = \mathrm{diag}(\mathbf{c})$

2. Calculate SVD of **S**

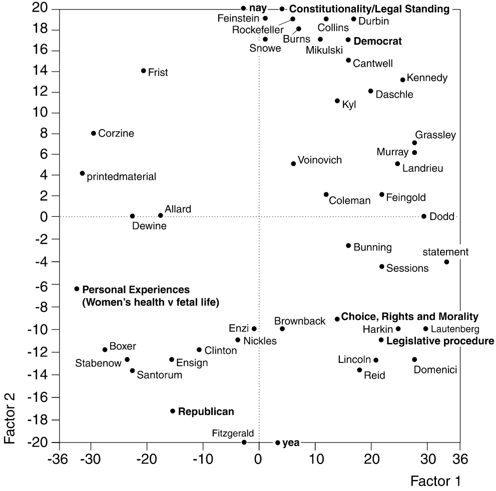
3. Project rows and columns onto low-dimensional space:

$$\theta = \mathbf{D}_r^{1/2}\mathbf{U} \text{ for rows (documents)}$$

$$\phi = \mathbf{D}_c^{1/2}\mathbf{V} \text{ for columns (words)}$$

Mathematically close to log-linear poisson regression model (Lowe, 2008)

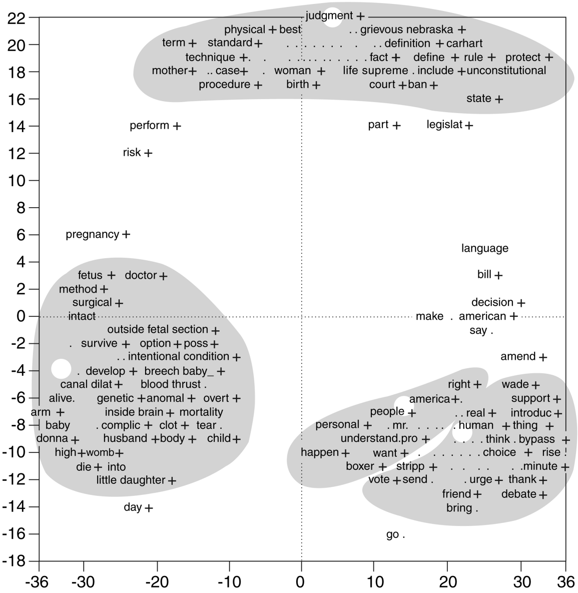
Example: Schonhardt-Bailey (2008) - speakers



	Eigenvalue	% Association	% Cumulative
Factor 1	0.30	44.4	44.4
Factor 2	0.22	32.9	77.3

Fig. 3. Correspondence analysis of classes and tags from Senate debates on Partial-Birth Abortion Ban Act

Example: Schonhardt-Bailey (2008) - words



Interpreting scaled dimensions

How can we validate that we are measuring a construct of interest?

1. Semantic validity

Interpreting scaled dimensions

How can we validate that we are measuring a construct of interest?

1. Semantic validity
   - Most discriminant words correspond to extremes of dimension of interest

Interpreting scaled dimensions

How can we validate that we are measuring a construct of interest?

1. Semantic validity
   - Most discriminant words correspond to extremes of dimension of interest
2. Convergent/discriminant construct validity

Interpreting scaled dimensions

How can we validate that we are measuring a construct of interest?

1. Semantic validity
    - Most discriminant words correspond to extremes of dimension of interest
2. Convergent/discriminant construct validity
    - Estimated positions match other existing measures where they should match, and depart where they should depart

Interpreting scaled dimensions

How can we validate that we are measuring a construct of interest?

1. Semantic validity
   - Most discriminant words correspond to extremes of dimension of interest
2. Convergent/discriminant construct validity
   - Estimated positions match other existing measures where they should match, and depart where they should depart
3. Predictive validity

Interpreting scaled dimensions

How can we validate that we are measuring a construct of interest?

1. Semantic validity
   - Most discriminant words correspond to extremes of dimension of interest
2. Convergent/discriminant construct validity
   - Estimated positions match other existing measures where they should match, and depart where they should depart
3. Predictive validity
   - Variation in positions or word usage corresponds with expected events

Interpreting scaled dimensions

How can we validate that we are measuring a construct of interest?

1. Semantic validity
   - Most discriminant words correspond to extremes of dimension of interest
2. Convergent/discriminant construct validity
   - Estimated positions match other existing measures where they should match, and depart where they should depart
3. Predictive validity
   - Variation in positions or word usage corresponds with expected events
4. Hypothesis validity

Interpreting scaled dimensions

How can we validate that we are measuring a construct of interest?

1. Semantic validity
   - Most discriminant words correspond to extremes of dimension of interest
2. Convergent/discriminant construct validity
   - Estimated positions match other existing measures where they should match, and depart where they should depart
3. Predictive validity
   - Variation in positions or word usage corresponds with expected events
4. Hypothesis validity
   - Variation in positions or word usage can be used effectively to test substantive hypotheses