Day 7: Extracting Clusters and Topics from Texts

Kenneth Benoit

Quantitative Analysis of Textual Data

November 18, 2014

Day 7 Outline

- classification v. clustering: *kNN* classifier
- *k*-means clustering
- hierarchical clustering
- topic models: LDA, extensions
- applications

- Next time: focus on social media and data management

k-nearest neighbour classifiers

- A non-parametric method for classifying objects based on the training examples taht are *closest* in the feature space

k-nearest neighbour classifiers

- A non-parametric method for classifying objects based on the training examples taht are *closest* in the feature space
- A type of instance-based learning, or “lazy learning” where the function is only approximated locally and all computation is deferred until classification

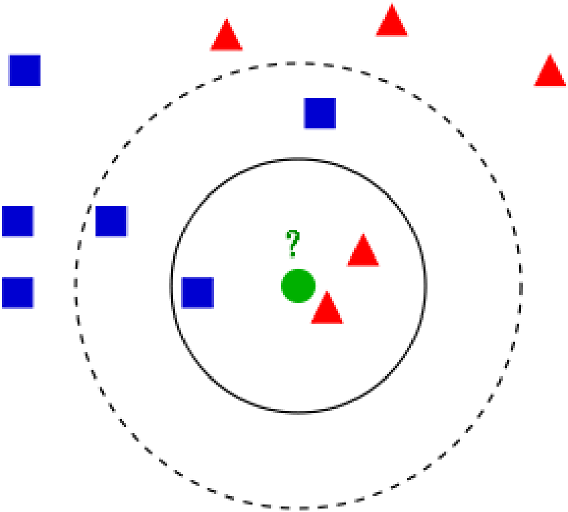
k-nearest neighbour classifiers

- A non-parametric method for classifying objects based on the training examples taht are *closest* in the feature space
- A type of instance-based learning, or "lazy learning" where the function is only approximated locally and all computation is deferred until classification
- An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors (where k is a positive integer, usually small)

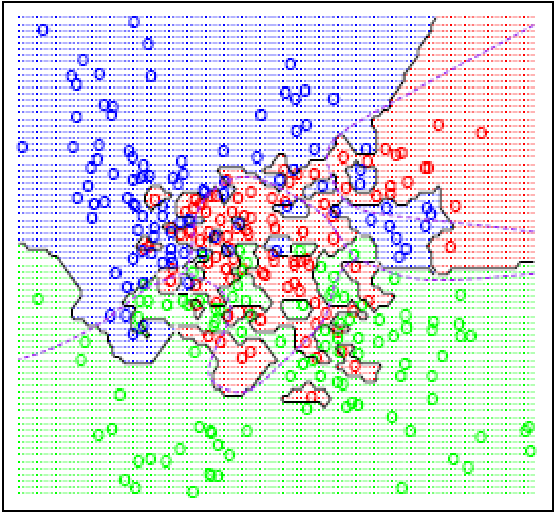
k-nearest neighbour classifiers

- A non-parametric method for classifying objects based on the training examples taht are *closest* in the feature space
- A type of instance-based learning, or “lazy learning” where the function is only approximated locally and all computation is deferred until classification
- An object is classified by a majority vote of its neighbors, with the object being assigned to the class most common amongst its k nearest neighbors (where k is a positive integer, usually small)
- Extremely *simple*: the only parameter that adjusts is k (number of neighbors to be used) - increasing k *smooths* the decision boundary

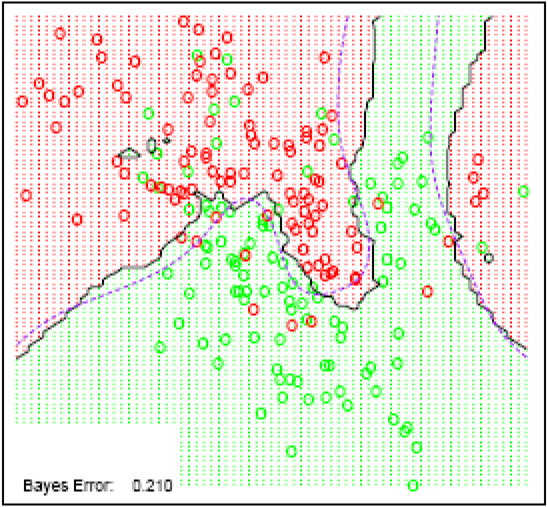
k-NN Example: Red or Blue?



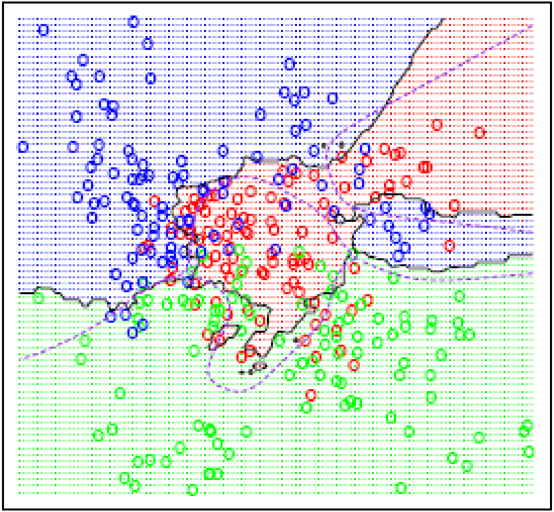
$k = 1$



$k = 7$



$k = 15$



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 minimized 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 minimized 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 minimized 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, Mikowski, Mahalanobis, etc.)
   - (assumes feature vectors have been normalized 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, Mikowski, Mahalanobis, etc.)
   - (assumes feature vectors have been normalized within item)
3. update: recompute the cluster centroids as the mean value of the points assigned to that cluster

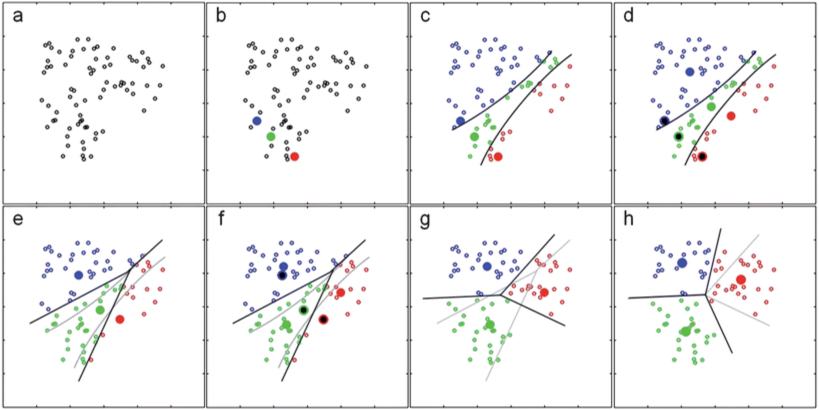
algorithm details

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, Mikowski, Mahalanobis, etc.)
   - (assumes feature vectors have been normalized 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, Mikowski, Mahalanobis, etc.)
   - (assumes feature vectors have been normalized 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

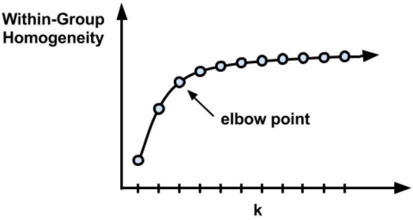


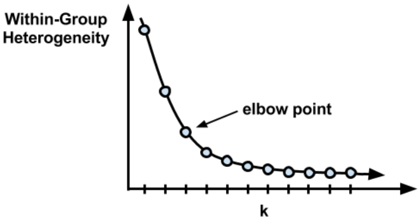
choosing the appropriate number of clusters

- 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
-
- numerous examples exist
- "iterative heuristic fitting"* (IHF) (trying different values and looking at what seems most plausible)

choosing the appropriate number of clusters

- “fit” statistics to measure homogeneity within clusters and heterogeneity in between
-
- numerous examples exist
- “iterative heuristic fitting”* (IHF) (trying different values and looking at what seems most plausible)

* Warning: This is my (slightly facetious) term only!

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: reducting 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: reducting features across all documents (Schütze and Silverstein, 1997)
    3. SVD methods, such PCA on a normalized 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: reducting features across all documents (Schütze and Silverstein, 1997)
  3. SVD methods, such PCA on a normalized feature matrix
  4. usually simple threshold-based truncation is used (keep all but 100 highest frequency or tf-idf terms)
- frequently/always involves weighting (normalizing 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 minimizing variance

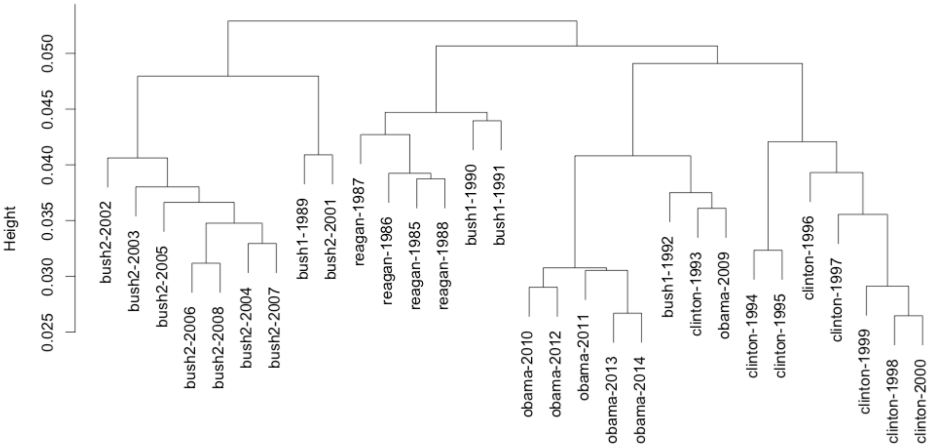
hierarchical clustering algorithm

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 minimizing variance
5. repeat 3–4 until a stopping condition is reached
   - e.g. all items have been merged into a single cluster

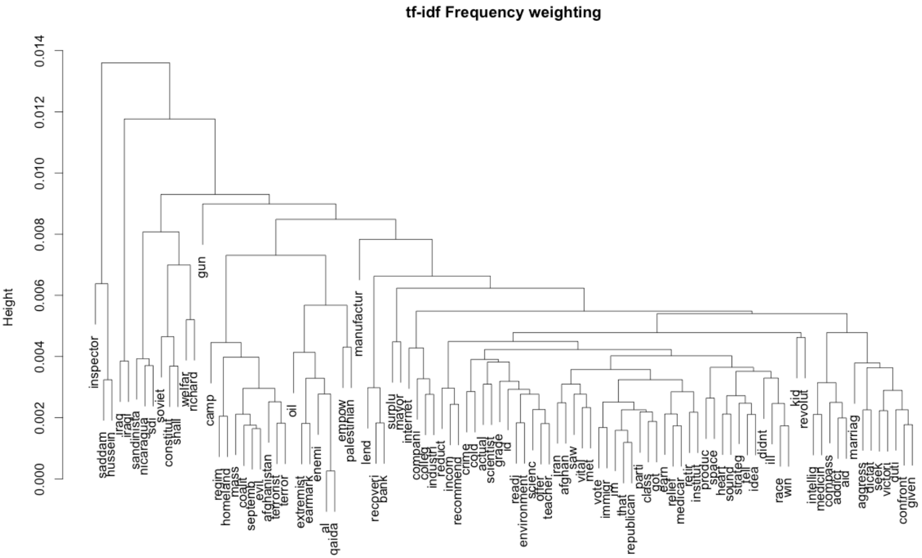
hierarchical clustering algorithm

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 minimizing 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



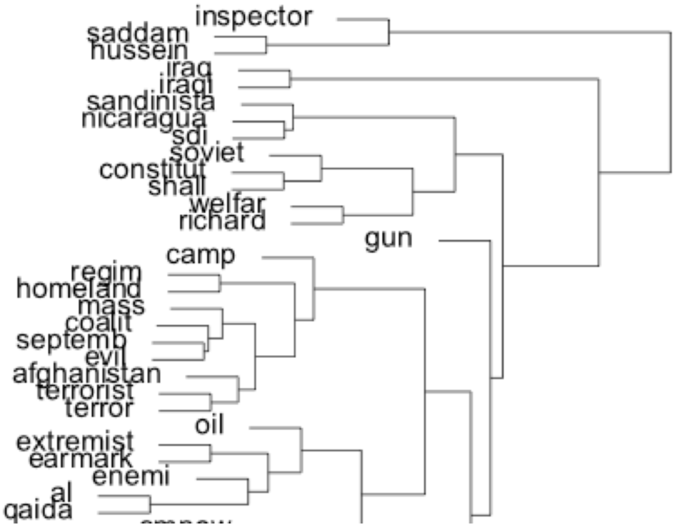
pros and cons of hierarchical clustering

- 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 probabalistic, generative advance on several earlier methods, "Latent Semantic Analysis" (LSA) and "probabalistic latent semantic indexing" (pLSI)

differences from previous models

unigram model
: each word each word is assumed to be drawn from the same term distribution

mixture of unigram models
: a topic is drawn for each document and all words in a document are drawn from the term distribution of the topic

mixed-membership models
: documents are not assumed to belong to single topics, but to simultaneously belong to several topics and the topic distributions vary over documents

Uses and applications

- Topic models are algorithms for discovering the main themes that pervade a large and otherwise unstructured collection of documents
- Can be used to organize the collection according to the discovered themes
- Topic modeling algorithms can be applied to massive collections of documents
- Topic modeling algorithms can be adapted to many kinds of data. among other applications, they have been used to find patterns in genetic data, images, and social networks

Advantages over cruder methods

- parametric, so we get estimates of parameters for topic proportions in each document, and topic weights for each word
- can incorporate additional information hierarchically (e.g. using "structural" topic models)
- but we pay for these benefits in the form of far greater computational complexity

Latent Dirichlet Allocation

- The LDA model is a Bayesian mixture model for discrete data where topics are assumed to be uncorrelated (in "classic" LDA)
- LDA provides a generative model that describes how the documents in a dataset were created
- Each of the K *topics* is a distribution over a fixed vocabulary
- Each document is a collection of words, generated according to a multinomial distribution, one for each of K topics
- Inference consists of estimating a posterior distribution from a joint distribution based on the probability model from a combination of what is observed (words in documents) and what is hidden (topic and word parameters)

Latent Dirichlet Allocation

- So the process is, roughly:
  1. Choose a number of topics
  2. Choose a distribution of topics, and create a document from this distribution
  3. For each topic, generate words according to a distribution specific to that topic
- The goal of inference in LDA is to discover the topics from the collection of documents, and to estimate the relationship of words to these

Latent Dirichlet Allocation: Details

- For each document, the LDA generative process is:
  1. randomly choose a distribution over topics (a multinomial of length K)
  2. for each word in the document
     2.1 Probabilistically draw one of the K topics from the distribution over topics obtained in (a), say topic β_k (each document contains topics in different proportions)
     2.2 Probabilistically draw one of the V words from β_k (each individual word in the document is drawn from one of the K topics in proportion to the document's distribution over topics as determined in previous step)
- The goal of inference in LDA is to discover the topics from the collection of documents, and to estimate the relationship of words to these, *assuming this generative process*

LDA generative model

How to generate

1. Term distribution β for each topic is drawn:

$$\beta \sim \text{Dirichlet}(\delta)$$

   β is the term distribution of topics and contains the probability of a word occurring in a given topic

2. proportions θ of the topic distribution for the document are drawn by

$$\theta \sim \text{Dirichlet}(\alpha)$$

3. For each of the N words in each document
   - choose a topic $x_i \sim \text{Multinomial}(\theta)$
   - choose a word $w_i \sim \text{Multinomial}(p(w_i|z_i, \beta))$

Graphical model for LDA using plate notation

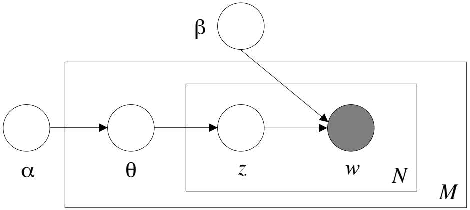


Figure 1: Graphical model representation of LDA. The boxes are "plates" representing replicates. The outer plate represents documents, while the inner plate represents the repeated choice of topics and words within a document.

Estimation and the "Dirichlet" part

- The Dirichlet is the conjugate prior distribution for the multinomial, and is used in the Bayesian inference required to estimate these parameters
- Estimation is performed using (collapsed) Gibbs sampling and/or Variational Expectation-Maximization (VEM)

Estimation and the "Dirichlet" part

- The Dirichlet is the conjugate prior distribution for the multinomial, and is used in the Bayesian inference required to estimate these parameters
- Estimation is performed using (collapsed) Gibbs sampling and/or Variational Expectation-Maximization (VEM)

  posterior:

$$p(\beta_{1:K}, \theta_{1:D}, z_{1:D} \mid w_{1:D}) = \frac{p(\beta_{1:K}, \theta_{1:D}, z_{1:D}, w_{1:D})}{p(w_{1:D})}$$

- (for us) Implemented easily in R for VEM and Gibbs

Illustration of the LDA generative process

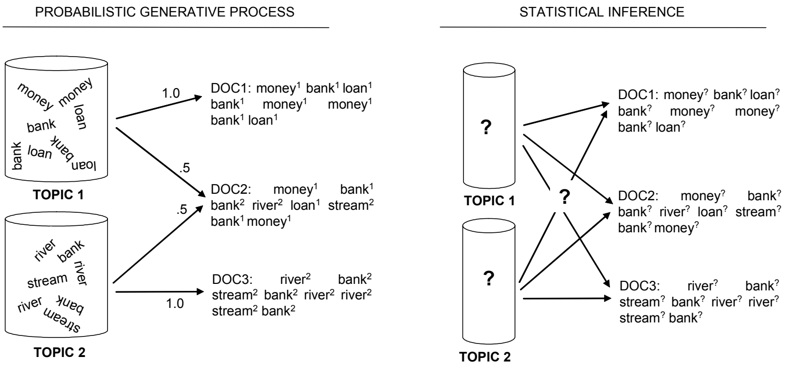


Figure 2. Illustration of the generative process and the problem of statistical inference underlying topic models

(from Steyvers and Griffiths 2007)

Topics example

Topic 247

word	prob.
DRUGS	.069
DRUG	.060
MEDICINE	.027
EFFECTS	.026
BODY	.023
MEDICINES	.019
PAIN	.016
PERSON	.016
MARIJUANA	.014
LABEL	.012
ALCOHOL	.012
DANGEROUS	.011
ABUSE	.009
EFFECT	.009
KNOWN	.008
PILLS	.008

Topic 5

word	prob.
RED	.202
BLUE	.099
GREEN	.096
YELLOW	.073
WHITE	.048
COLOR	.048
BRIGHT	.030
COLORS	.029
ORANGE	.027
BROWN	.027
PINK	.017
LOOK	.017
BLACK	.016
PURPLE	.015
CROSS	.011
COLORED	.009

Topic 43

word	prob.
MIND	.081
THOUGHT	.066
REMEMBER	.064
MEMORY	.037
THINKING	.030
PROFESSOR	.028
FELT	.025
REMEMBERED	.022
THOUGHTS	.020
FORGOTTEN	.020
MOMENT	.020
THINK	.019
THING	.016
WONDER	.014
FORGET	.012
RECALL	.012

Topic 56

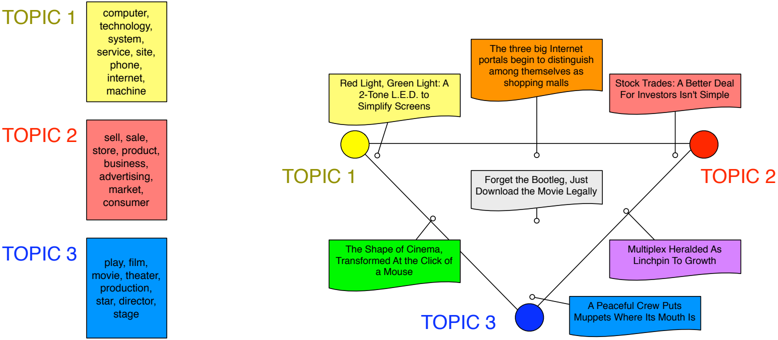
word	prob.
DOCTOR	.074
DR.	.063
PATIENT	.061
HOSPITAL	.049
CARE	.046
MEDICAL	.042
NURSE	.031
PATIENTS	.029
DOCTORS	.028
HEALTH	.025
MEDICINE	.017
NURSING	.017
DENTAL	.015
NURSES	.013
PHYSICIAN	.012
HOSPITALS	.011

Figure 1. An illustration of four (out of 300) topics extracted from the TASA corpus.

(from Steyvers and Griffiths 2007)

Often K is quite large!

Example



(a) Topics (b) Document Assignments to Topics

Model evaluation (K)

- can compute a likelihood for “held-out” data
- perplexity: can be computed as (using VEM):

$$\text{perplexity}(w) = \exp\left\{ -\frac{\sum_{d=1}^{M} \log p(w_d)}{\sum_{d=1}^{M} N_d} \right\}$$

- lower perplexity score indicates better performance

Evaluating model performance: human judgment

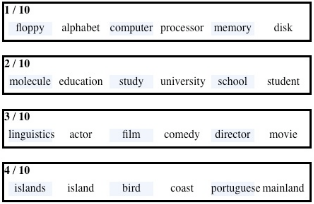
(Chang, Jonathan et al. 2009. "Reading Tea Leaves: How Humans Interpret Topic Models." *Advances in neural information processing systems*.)

Uses human evaluation of:

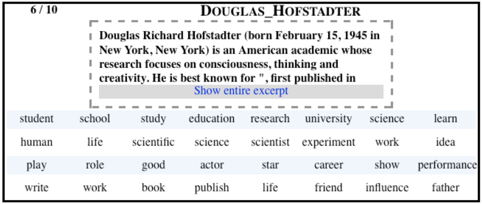
- whether a topic has (human-identifiable) semantic coherence: word intrusion, asking subjects to identify a spurious word inserted into a topic
- whether the association between a document and a topic makes sense: topic intrusion, asking subjects to identify a topic that was not associated with the document by the model

Example

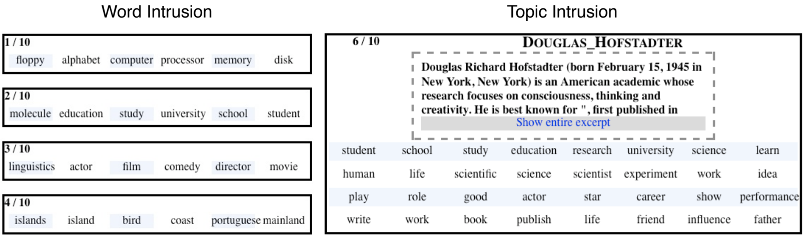
Word Intrusion



Topic Intrusion



Example



- conclusions: the quality measures from human benchmarking were negatively correlated with traditional quantitiative diagnostic measures!

Drawbacks of LDA

- discards word order
- assumes documents are exchangeable
- the setting of the hyperparameters has led to a great deal of confusion, even as we note above, leading to a misconception about the effective- ness of different forms of posterior inference
- unclear how to choose the number of topics K

Extensions to LDA

- relax independence of topics
  - Correlated Topic Model (Blei and Lafferty 2007): Dirichlet prior is replaced with a logistic Normal distribution
  - Dynamic Topic Model (Blei and Lafferty 2006): parameters change using an evolution model
  - Add additional information
- Expressed Agenda Model (Grimmer 2010): allows for differences in topic probabilities across authors
- Add additional information
  - Dirichlet-Multinomial Topic Model (Mimno and McCallum (2008): parameterized the Dirichlet parameter using covariates
  - Structural Topic Model: Airoldi, Roberts, and Stewart (2011)

Which implementation in R?

- `lda`
- `topicmodels`
- `mallet`
- `stm`
- quanteda: `textmodel_lda()`