UNIVERSITÄT
LEIPZIG

Forschungsseminar CSS – Supervised & Unsupervised ML

GWZ H2.1.15, 27.11.2025

Felix Lennert, M.Sc.

OUTLINE

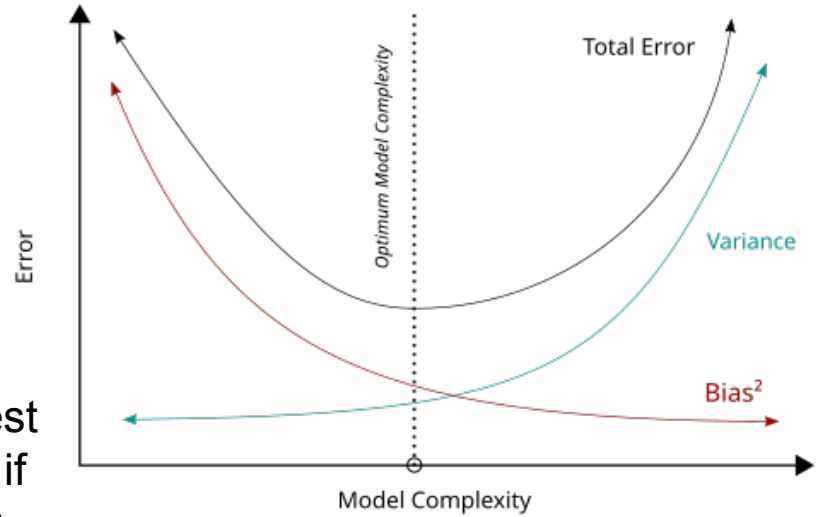
- Appendix to last week
  - Lasso/Ridge
  - Naive Bayes
  - SVM
  - Random Forest
  - XGBoost
- Unsupervised ML; Topic Modeling
  - Value for Social Sciences
  - LDA
  - In Practice
  - Evaluation Strategy

OUTLINE

- Appendix to last week
  - Lasso/Ridge
  - Naive Bayes
  - SVM
  - Random Forest
  - XGBoost
- Unsupervised ML; Topic Modeling
  - Value for Social Sciences
  - LDA
  - In Practice
  - Evaluation Strategy

REGULARIZATION – LASSO & RIDGE

- The Overfitting Challenge:
  - Complex models memorize training data
  - Poor performance on new data
  - Too many features relative to samples
- "Bias-Variance Trade-off"
  - Bias: model makes strong assumptions about data, underfits (e.g., using straight line for curve)
  - Variance: how much does prediction on test set vary from predictions on training set – if model works perfectly on training data but not so well on test data, variance is high



REGULARIZATION – LASSO & RIDGE

- The Solution: Penalize Complexity
- Minimize: Prediction Error (RSS) + Penalty for Complexity
- This forces models to be simpler and more generalizable

- Ridge: shrink everything gradually – minimize
  - punishes large coefficients heavily
  - every coefficient shrinks to 0 – but never becomes 0
  - Good for: multicollinearity (spreads weights across correlated features)

$$RSS + \lambda \sum \beta_j^2$$

- Lasso: minimize
  - Penalty grows linearly
  - Coefficients can become exactly 0
  - Good for: feature selection – weakly predictive features are thrown out

$$RSS + \lambda \sum |\beta_j|$$

NAIVE BAYES

- Calculate probability of each class given the features, pick the most likely
- Using Bayes Theorem:
  P(Class|Features)=P(Features|Class) * P(Class)/P(Features)
- "How likely is positive class, given we see words 'loved' and ‘amazing'?"
  P(Positive|love, amazing)

- Learn probability of each word appearing in positive vs. negative reviews
- New review: multiply probabilities
- Pick class with higher probability

- Naive: Features are independent given the class

NAIVE BAYES

Example:

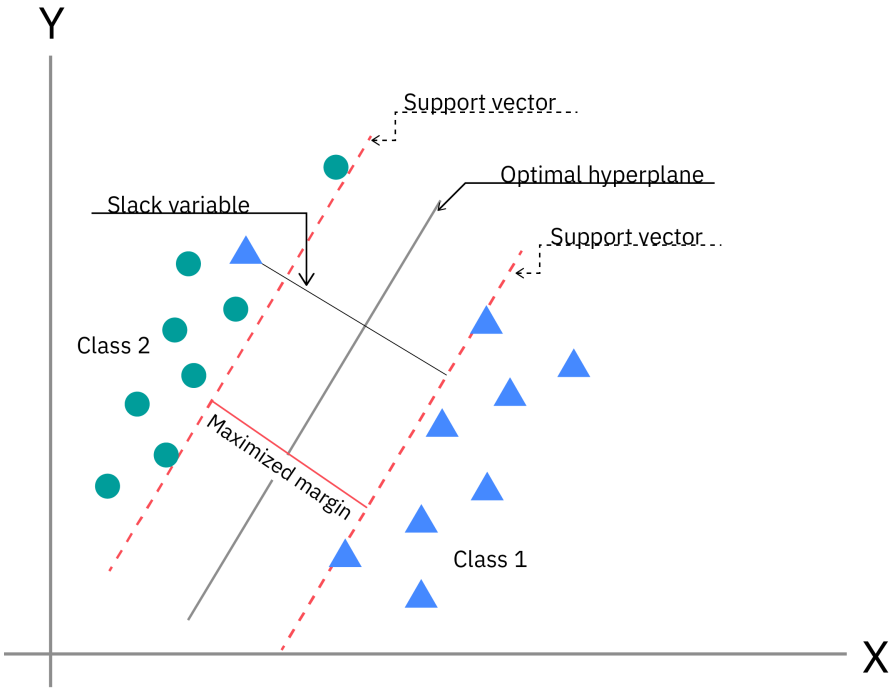
- positive reviews:
  - loved appears in 60 (=> P(loved|positive)=0.6
  - P(amazing|positive)=0.5; P(terrible|positive)=0.05
- negative reviews:
  - loved appears in 10 (=> P(loved|positive)=0.10
  - P(amazing|positive)=0.15; P(terrible|positive)=0.70

NAIVE BAYES

- New review: “love amazing”;
  baseline probabilities:
  P(pos)=0.5, P(neg)=0.5
- Compare:
  P(pos|love, amazing)=P(loved|pos)*P(amazing|pos)*P(pos)/P(loved, amazing) = 0.6*0.5*0.5/P(love, amazing)
  P(neg|love, amazing)=P(loved|neg)*P(amazing|neg)*P(neg)/P(loved, amazing) = 0.1*0.15*0.5/P(love, amazing)
- We can ignore P(love, amazing) in our comparison, because we don’t want absolute probabilities but only need to see which one is more likely – P(love, amazing) is the same for both equations
- P(pos|love,amazing)=0.15 > P(neg|love,amazing)=0.0075 => review is positive

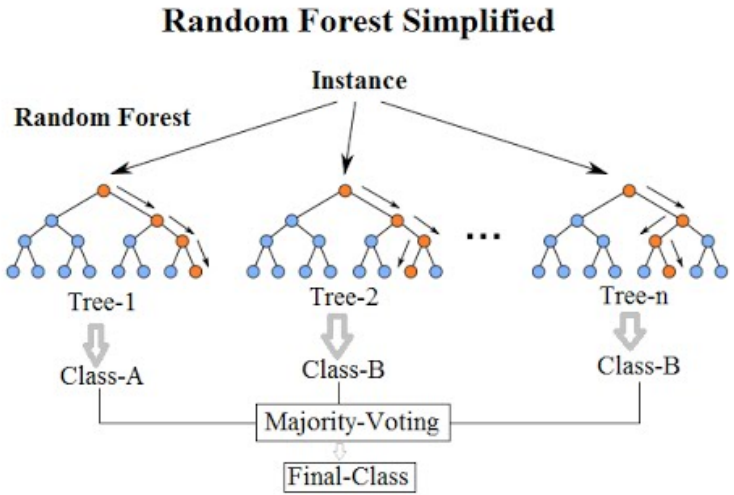
SVM

- Idea: find the boundary between classes with the widest safety margin
- Support vectors: critical points defining the boundary



RANDOM FOREST – ENSEMBLE LEARNING

- Idea: build many decision trees, combine their votes
- Approach:
  - Create as many training sets as trees (sample with replacement)
  - At each split, consider subset of features
  - Treat trees independently, let each decide
  - Vote: Average predictions/majority

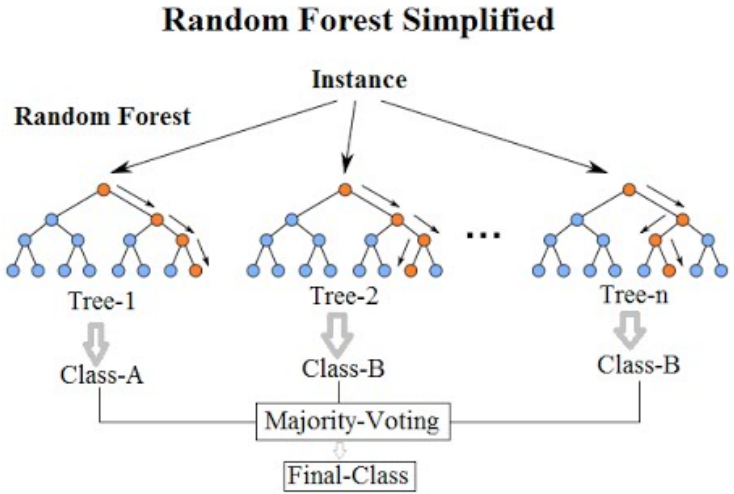


RANDOM FOREST – “BAGGING”

“Ask 100 independent experts and let them vote”

Key properties:

- Independence:
  - Each tree is trained separately
  - contributes equally
- Diversity:
  - Each tree sees slightly different data
  - uses different features at each split => different trees use different features
- Hence, Individual trees overfit differently, overfitting cancels out

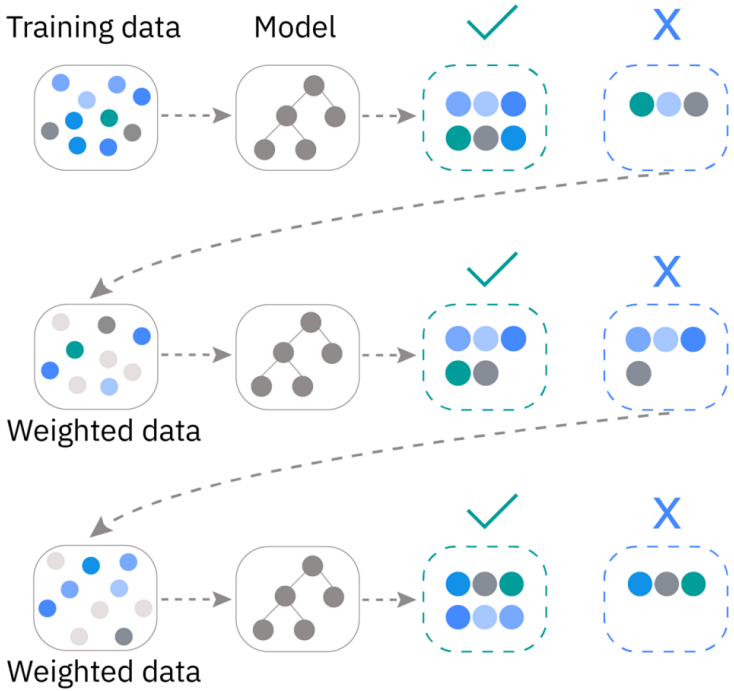


XGBOOST – "GRADIENT BOOSTING"

Build trees sequentially, the next one corrects the former one's mistakes
=> One person learns, improves, learns more, improves…

- Start simple with baseline prediction
- Find errors
- Train a new tree to predict these errors
- Add correction with learning rate (each new tree carries less weight – tunable parameter) – default: 0.3
- Find errors, train new tree on errors, repeat repeat repeat (until "n_estimators" is reached)

XGBOOST – “GRADIENT BOOSTING”



RULES OF THUMB

- Always start simple (Naive Bayes or Lasso)
- Benchmark everything (compare multiple models)
- Consider your constraints (time, interpretability, resources)
- Results from 10-fold cross validation with basic preprocessing:

Approach	Accuracy
Naive Bayes	0.64
Logistic Regression	0.74
Logistic Regression (Lasso, lambda=0.01)	0.74
Logistic Regression (Ridge, lambda=0.01)	0.74
SVM	0.82
Random Forest	0.80
XGBoost	0.79

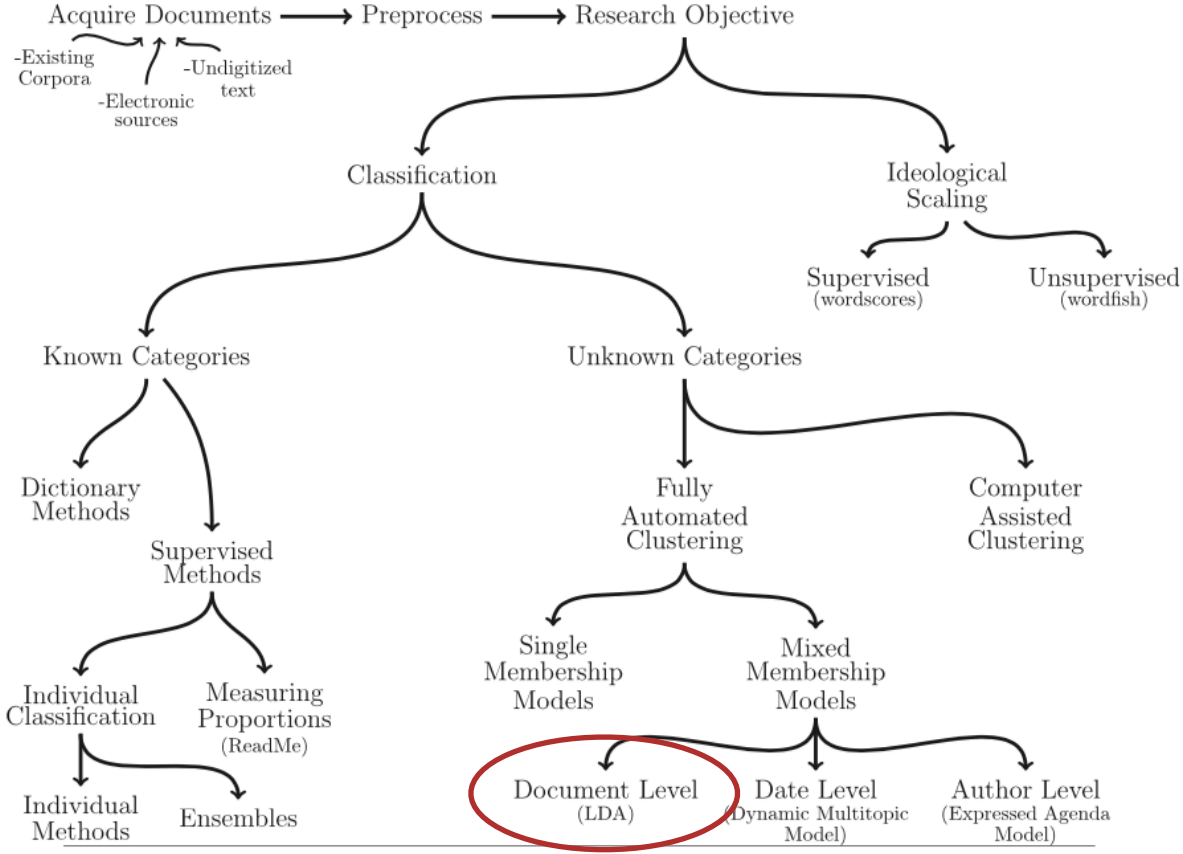
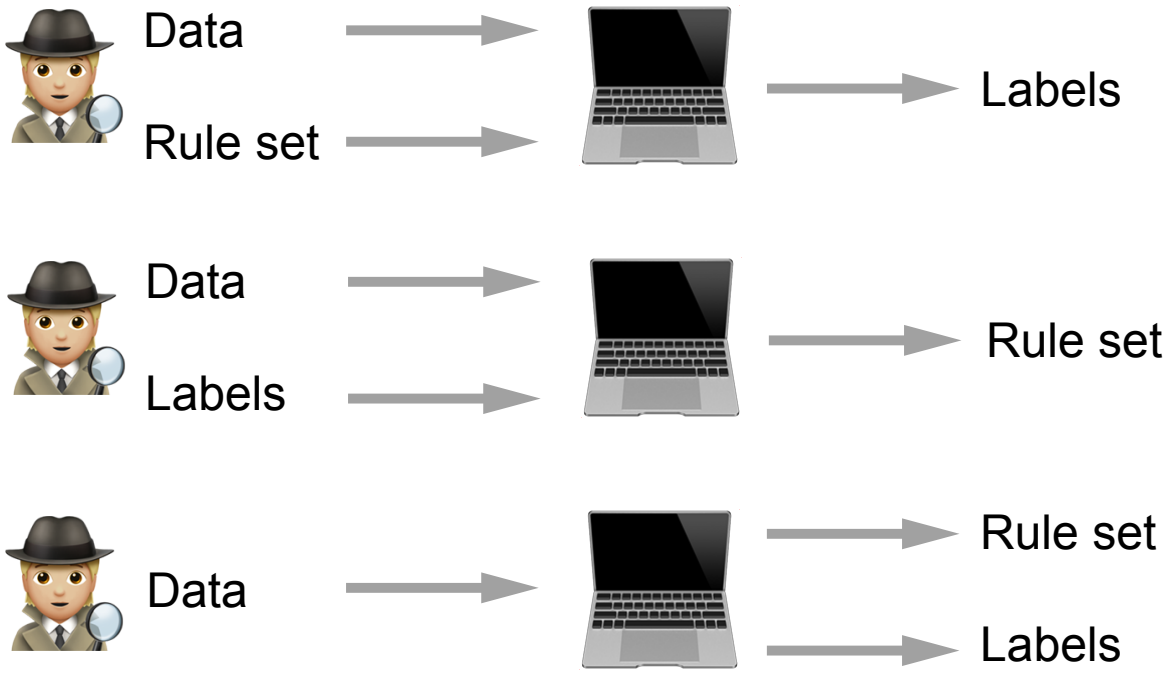


Fig. 1 An overview of text as data methods.

inspired by Ash (2018)

HOW TO PRODUCE THESE DATA?



Dictionary-based analysis
Computer applies rules

Supervised ML
Computer learns relationship (“rules”) between data and answers

Unsupervised ML
Computer suggests rules and answers based on patterns in data

TOPIC MODELING'S VALUE FOR SOCIAL SCIENTISTS (DIMAGGIO ET AL. 2013)

A good approach for distance-reading should fulfill four requirements

- *explicitness* – others should be able to replicate it
- *automation* – as data sets become larger
- *inductive* – shall not rely on researcher's priors too much
- *take into account context* – terms can mean different things in different contexts (*relationality* of meaning)

TOPIC MODELING'S VALUE FOR SOCIAL SCIENTISTS

Topic models

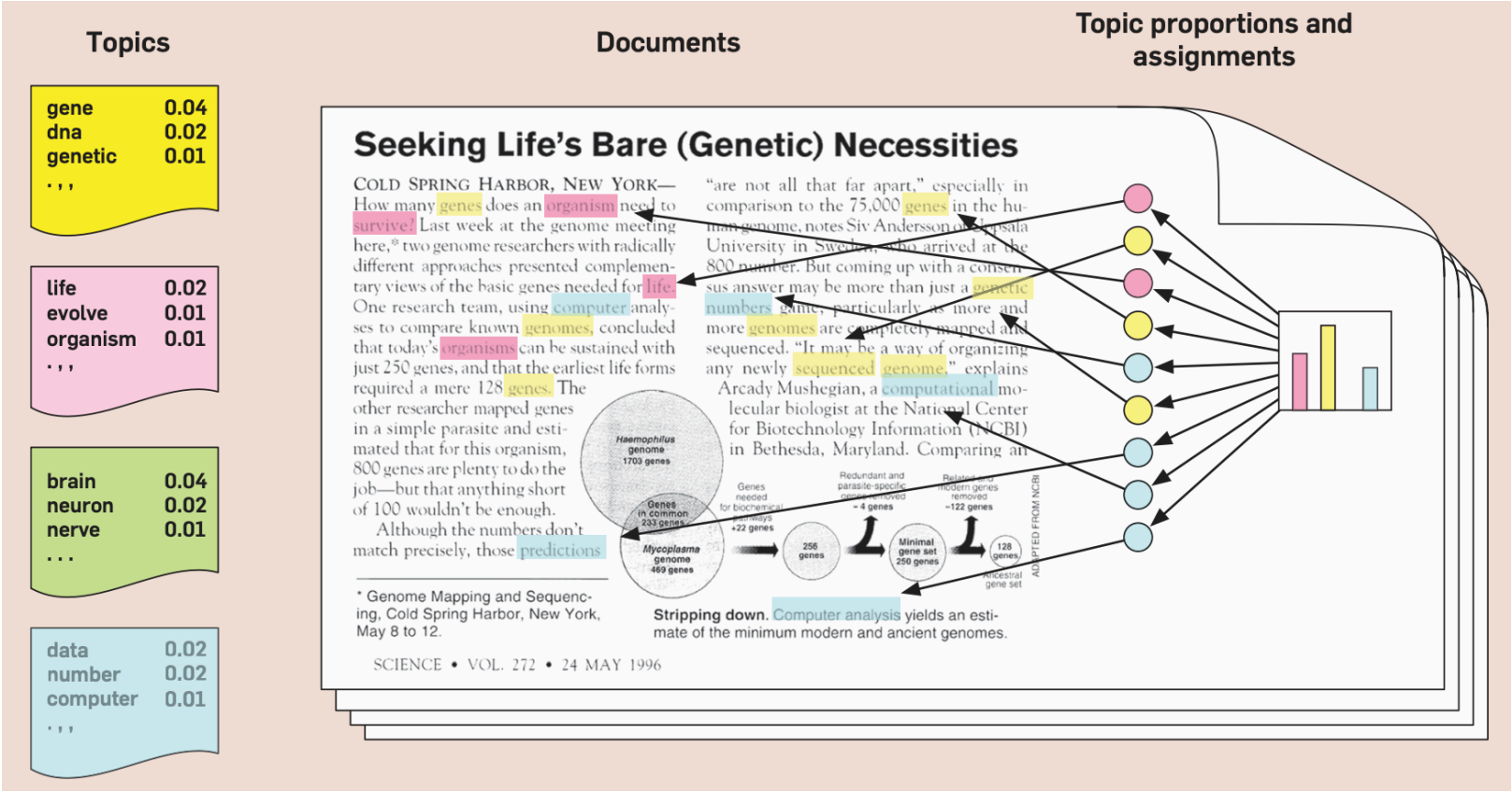
- *organize* documents into topics based on their content, i.e., the words they contain
- *organize* terms into topics based on their co-appearance

- documents are a mixture of topics
- topics are a mixture of words
- words can appear in multiple topics

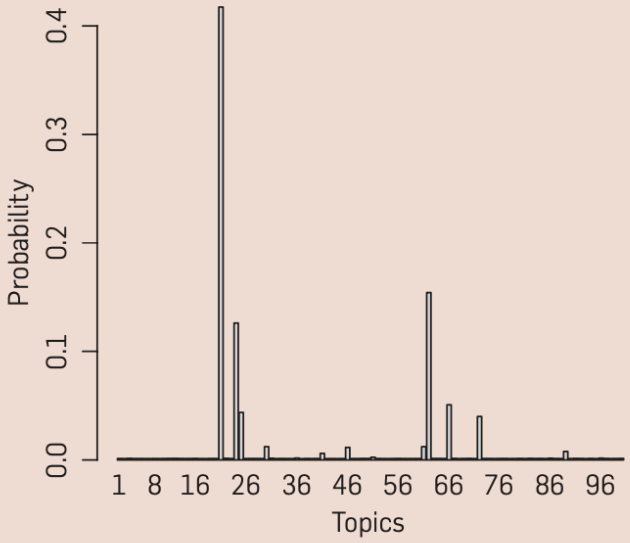
TOPIC MODELING'S VALUE FOR SOCIAL SCIENTISTS (DIMAGGIO ET AL. 2013)

A good approach for distance-reading should fulfill four requirements

- *explicitness* – others should be able to replicate it ⇨ parameters are explicit
- *automation* – as data sets become larger ⇨ computer does the work
- *inductive* – shall not rely on researcher's priors too much ⇨ unsupervised
- *take into account context* – terms can mean different things in different contexts (*relationality* of meaning) ⇨ words can belong to different topic



Blei 2012, p. 78



"Genetics"	"Evolution"	"Disease"	"Computers"
human	evolution	disease	computer
genome	evolutionary	host	models
dna	species	bacteria	information
genetic	organisms	diseases	data
genes	life	resistance	computers
sequence	origin	bacterial	system
gene	biology	new	network
molecular	groups	strains	systems
sequencing	phylogenetic	control	model
map	living	infectious	parallel
information	diversity	malaria	methods
genetics	group	parasite	networks
mapping	new	parasites	software
project	two	united	new
sequences	common	tuberculosis	simulations

Blei 2012, p. 79

STYLIZED APPROACH (TAKEN FROM CHEN 2013)

Topic models assume the following data generation process

- author decides on length of text
- author decides on topics
- author draws words from vocabulary of topics

STYLIZED APPROACH (TAKEN FROM CHEN 2013)

Example: 5 sentences, 2 topics

- I like to eat broccoli and bananas.
- I ate a banana and spinach smoothie for breakfast.
- Chinchillas and kittens are cute.
- My sister adopted a kitten yesterday.
- Look at this cute hamster munching on a piece of broccoli.

STYLIZED APPROACH (TAKEN FROM CHEN 2013)

Example: 5 sentences, 2 topics

- I like to eat broccoli and bananas. ⟹ 100% food
- I ate a banana and spinach smoothie for breakfast. ⟹ 100% food
- Hamsters and kittens are cute. ⟹ 100% adorable animals
- My sister adopted a kitten yesterday. ⟹ 100% adorable animals
- Look at this cute hamster munching on a piece of broccoli. ⟹ 50% adorable animals, 50% food

⟹ IDEA OF LDA: topics are mixture of words, documents mixture of topics (and of words)

STYLIZED APPROACH (TAKEN FROM CHEN 2013)

Example: 5 sentences, 2 topics

- I like to **eat broccoli** and **bananas**. ⇒ 100% food
- I **ate** a **banana** and **spinach smoothie** for **breakfast**. ⇒ 100% food
- *Hamsters* and *kittens* are *cute*. ⇒ 100% adorable animals
- My sister *adopted* a *kitten* yesterday. ⇒ 100% adorable animals
- Look at this *cute hamster* **munching** on a piece of **broccoli**. ⇒ 50% adorable animals, 50% food

Problem: For the computer, all the words look the same

STYLIZED APPROACH (TAKEN FROM CHEN 2013)

- assign a **topic t** at random to each **word w** in each **document d**

⟹ number of topics (*k*) is chosen **<u>before</u>**

- go through each **word w** in each **document d**
- assume that all the other assigned topics (to the words) are correct
- compute $p(t \mid d)$= the proportion of **words w** in **document d** that are currently assigned to **topic t**
- compute $p(w \mid t)$= the **proportion of w** being **assigned to t** (over all documents)
- new **topic distribution for w**: $p(t \mid d) \times p(w \mid t)$
- …repeat until a steady state is achieved

STYLIZED APPROACH (TAKEN FROM CHEN 2013)

- assign a **topic t** at random to each **word w** in each **document d**

assign a **topic t** at random to each **word w** in each **document d**

here: k=2

	broccoli	banana(s)	munching	hamster	kitten	spinach	smoothie	cute
S 1	1	2						
S 2		2				1	1	
S 3			2		1			2
S 4					2			
S 5	2			2				2
S …								

STYLIZED APPROACH (TAKEN FROM CHEN 2013)

LDA takes as input the documents and the assumed number of topics

It aims to learn the proportion α of each topic t in a document

- Learning process:
  - go through each **word w** in each **document d**
  - assume that all the other assigned topics (to the words) are correct
  - compute **p(topic t | document d) =** the proportion of **words in document d** that are currently assigned to **topic t** (⟹ if a word appears in a document, it is likely to be of the same topic)
  - compute **p(word w | topic t) =** the **proportion of w** being **assigned to t** (over all documents)
  - new **topic distribution for w**: **p(topic t | document d) * p(word w | topic t)**
  - …repeat until a steady state is achieved

- go through each **word w** in each **document d**
- assume that all the other assigned topics (to the words) are correct
- compute $p(t \mid d)$= the proportion of **words w** in **document d** that are currently assigned to **topic t**

	broccoli	banana(s)	munching	hamster	kitten	spinach	smoothie	cute
S 1	p(T=2|S 1)=1	2						
S 2		2				1	1	
S 3			2		1			2
S 4					2			
S 5	2			2				2
S …								

STYLIZED APPROACH (TAKEN FROM CHEN 2013)

- compute $p(w \mid t)$ = the **proportion of w** being **assigned to t** (over all documents)

$\Rightarrow p(w = broccoli \mid t = 1) = 0$

$\Rightarrow p(w = broccoli \mid t = 2) = 1$

STYLIZED APPROACH (TAKEN FROM CHEN 2013)

- go through each **word w** in each **document d**
- assume that all the other assigned topics (to the words) are correct
- compute $p(t \mid d)$= the proportion of **words w** in **document d** that are currently assigned to **topic t**
- compute $p(w \mid t)$= the **proportion of w** being **assigned to t** (over all documents)
- new **topic distribution for w**: $p(t \mid d) \times p(w \mid t)$

$\Rightarrow p(broccoli, t = 1) = 0 \times 0 = 0$

$\Rightarrow p(broccoli, t = 2) = p(t = 2 \mid d = s_1) \times p(w = broccoli \mid t = 2) = 1$

STYLIZED APPROACH (TAKEN FROM CHEN 2013)

- go through each **word w** in each **document d**
- assume that all the other assigned topics (to the words) are correct
- compute $p(t \mid d)$= the proportion of **words w** in **document d** that are currently assigned to **topic t**
- compute $p(w \mid t)$= the **proportion of w** being **assigned to t** (over all documents)
- new **topic distribution for w**: $p(t \mid d) \times p(w \mid t)$

- **…repeat until a steady state is achieved**

STYLIZED APPROACH

In the end, the topic model will give us two coefficients:

- γ (gamma), document-topic probability: the proportion of words in a document coming from a topic
- β (beta), term-topic probability: the probability of a term coming from a topic

UNSUPERVISED LEARNING WITH TEXT – THE PROCESS

- Choose a set of documents (corpus) and a number of topics *k*
  ⟹ usually *k* is not known a priori – estimation by training multiple models and comparing different measures
- Preprocess the documents
  ⟹ e.g., tokenization (also: bi- and trigrams), stemming/lemmatization, remove frequent words, etc. – for ramifications, see Denny and Spirling (2018)
- Learn topic model
- Make sense of topics

CHOICE OF CORPUS

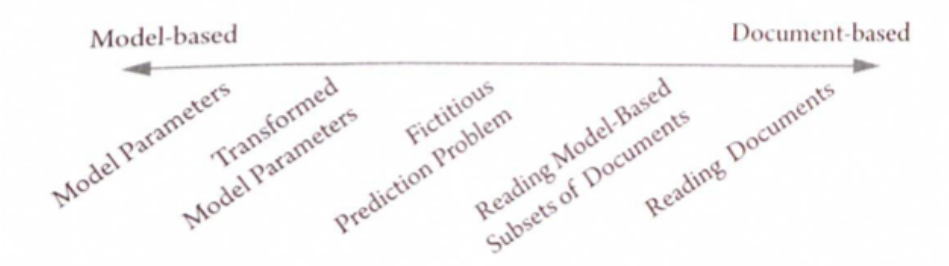
- Not as important here, model searches for structure
- Documents should have a certain length (since model assumes documents to be a *mixture of topics*)
  ⟹ for short texts, e.g., Tweets, specific “single-membership” models exist

CHOOSING K

- “One of the most difficult questions in Unsupervised Learning” (Grimmer and Stewart 2013: 19)
- No straightforward thing to do
- Solution: train many models and calculate evaluation scores for them (using R package “ldatuning”, or “stm::searchK()”)

MAKING SENSE OF TOPICS

- LDA gives you two values:
  - the probability that a word belongs to a topic, β
  - the probability that a document belongs to a topic, γ
- Goal: to give topics labels



Grimmer, Roberts, and Stewart 2022: 160

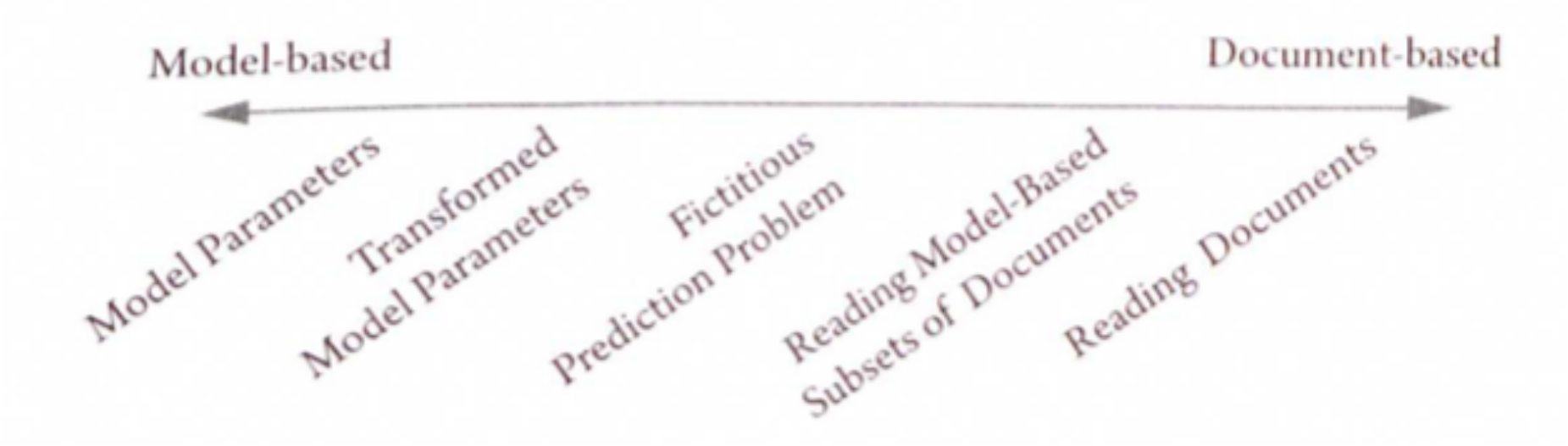
MAKING SENSE OF TOPICS

- Goal: to give topics labels
- Look at most prevalent terms contained in topics
  - *apophenia* (seeing patterns in random sets)
  - confirmation bias (seeing what you want to see)
- Read documents that consist mainly of words drawn from topics
  - tedious

⟹ but, remember the rules of text mining: VALIDATE VALIDATE VALIDATE

⟹ in this case: ensure that your topics constitute what you think they do

MAKING SENSE OF TOPICS



Grimmer, Roberts, and Stewart 2022: 160

EXTENSION: STRUCTURAL TOPIC MODELS

LDA comes with a bunch of limitations:

- Only takes text into account (no document covariates) – **topics are learned taking covariates into account**
- Topic-word distribution is stationary, cannot vary between documents (Republicans and Democrats may talk about the same topics but use different terms) – **different documents may contain the same topic but use different lingo**
- Topics are treated as independent from each other – **topics are allowed to be correlated**

⟹ Structural Topic Models mitigate these shortcomings

EXTENSION: SEEDED TOPIC MODELS

LDA comes with a bunch of limitations:

- Topics may actually be known in the beginning
- However: if LDA doesn't find the topic, this doesn't work
- Solution: **define ("seed") topics before – assign certain terms to topics**

⟹ Seeded topic model

RESULT

- Finally, you have added a new label to your document, namely its topic distribution
- You can use this label as a dependent as well as an independent variable for further inference

UNIVERSITÄT
LEIPZIG

MERCI

Felix Lennert
Institut für Soziologie

felix.lennert@uni-leipzig.de
www.uni-leipzig.de

REFERENCES

- Barberá, Pablo, Amber E. Boydstun, Suzanna Linn, Ryan McMahon, and Jonathan Nagler. 2021. "Automated Text Classification of News Articles: A Practical Guide." *Political Analysis* 29(1):19–42.
- Denny, Matthew J. and Arthur Spirling. 2018. "Text Preprocessing For Unsupervised Learning: Why It Matters, When It Misleads, And What To Do About It." Political Analysis 26(2):168–89.
- DiMaggio, Paul, Manish Nag, and David Blei. 2013. "Exploiting Affinities Between Topic Modeling and the Sociological Perspective on Culture: Application to Newspaper Coverage of U.S. Government Arts Funding." *Poetics* 41(6):570–606.
- Grimmer, Justin, Margaret Roberts, and Brandon Stewart. 2022. *Text as Data: A New Framework for Machine Learning and the Social Sciences*. Princeton: Princeton University Press.
- Hurtado Bodell, Miriam, Måns Magnusson, and Marc Keuschnigg. n.d. "Seeded Topic Models in Digital Archives: Analyzing the Swedish Understanding of Immigration, 1945–2019." OSF Preprint.
- Roberts, Margaret E., Brandon M. Stewart, Dustin Tingley, and Edoardo M. Airoldi. 2013. "The Structural Topic Model and Applied Social Science." in *NIPS 2013 Workshop on Topic Models*.