Political Analysis of Social Media Data

Supervisd Learning 1

Instructor: Gregory Eady
Office: 18.2.10
Office hours: Fridays 13-15

Today

- Supervised learning
- Video lectures & exercises

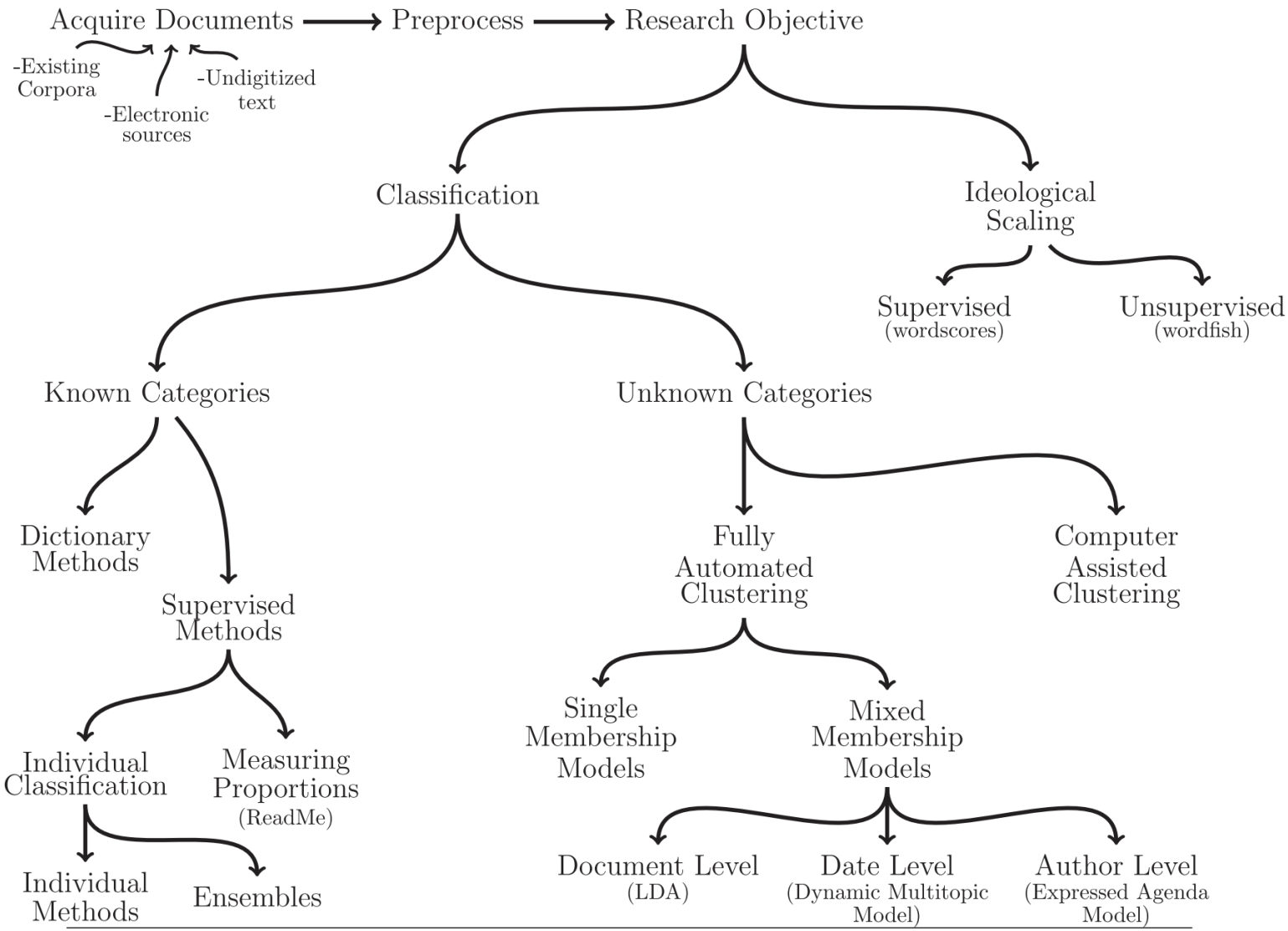


Fig. 1 An overview of text as data methods.

Unsupervised versus supervised learning

- **Unsupervised learning**
  - Learn from the structure of the data themselves
  - Data are unlabeled (e.g. we don't know any user's ideology; we don't any social media post's topic)
  - Examples: LDA, IRT, cluster analysis techniques
- **Supervised learning**
  - Labels on some of the data (can be any type of label, not necessarily a discrete class)
  - Learn the relationships between input variables ("features") and the the labeled values
  - Goal is typically to predict the labels on cases in the data that do not have labels

Benefits and drawbacks of supervised learning

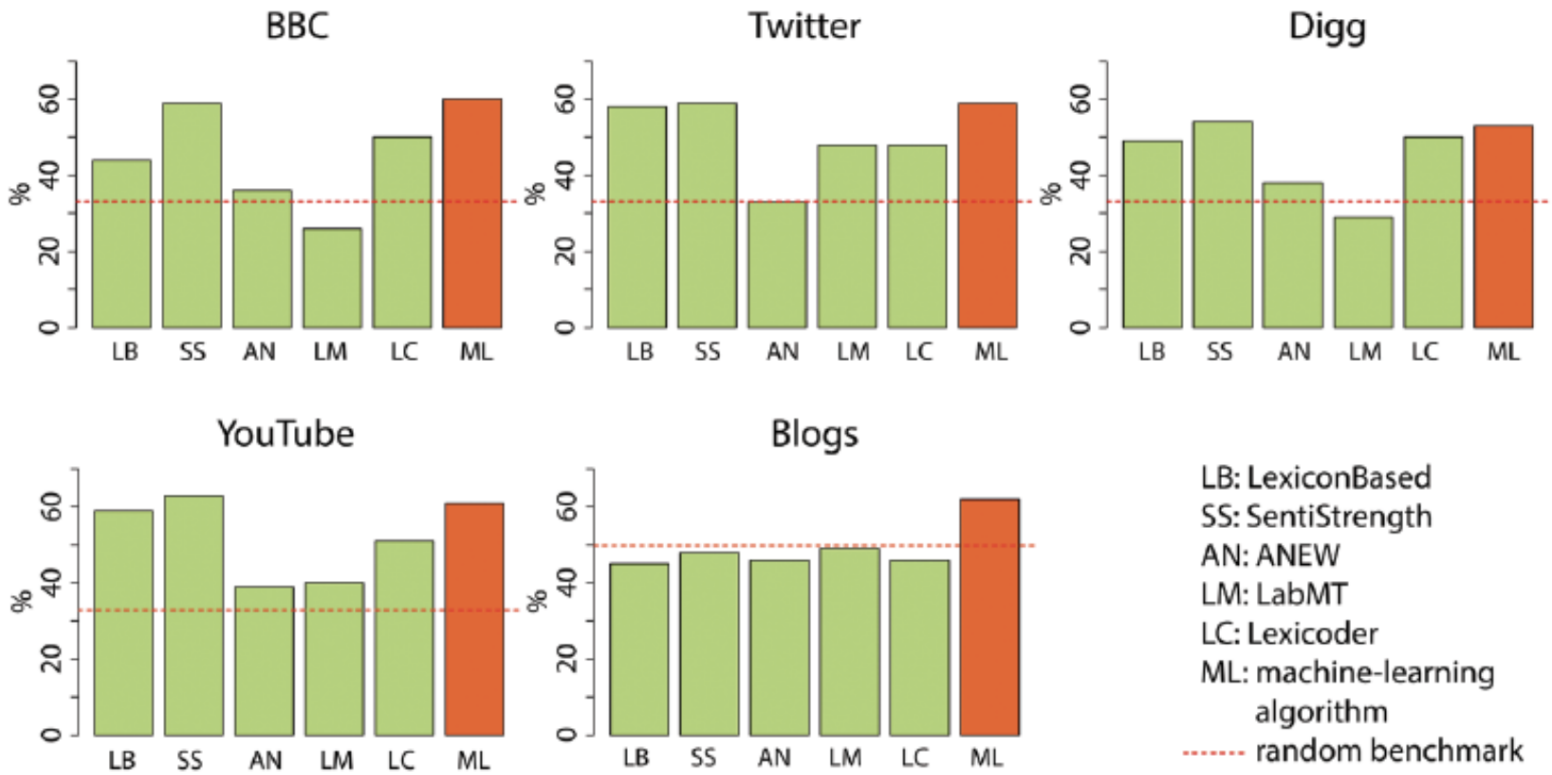
- **Benefits**
  - The researcher specifically decides the concept that is being measured
  - The resulting measure is easily interpretable
- **Drawbacks**
  - Because the researcher decides the concept being measured, he or she needs to measure it (e.g. manual annotation)
  - The resulting measure thus is costly in time and money
  - The availability of LLMs can make this much cheaper in time for a number of tasks

Example: Dictionary vs. supervised learning approaches to sentiment analysis

- **Dictionary-based approaches**
  - Cheap to apply
  - Applicable across many corpora (not specific to a certain corpus)
  - A dictionary's generality means its performance will vary widely across corpora
- **Supervised learning approaches**
  - Expensive to apply
  - Typically is designed to be specific to a given corpus
  - Being specific to a corpus means its performance will almost always be better than that of dictionary-based approaches

González-Bailón and Paltoglou (2015)

FIGURE 3
Lexicons' Accuracy in Document Classification
Compared to Machine-Learning Approach



Jaidka et al. (2020)

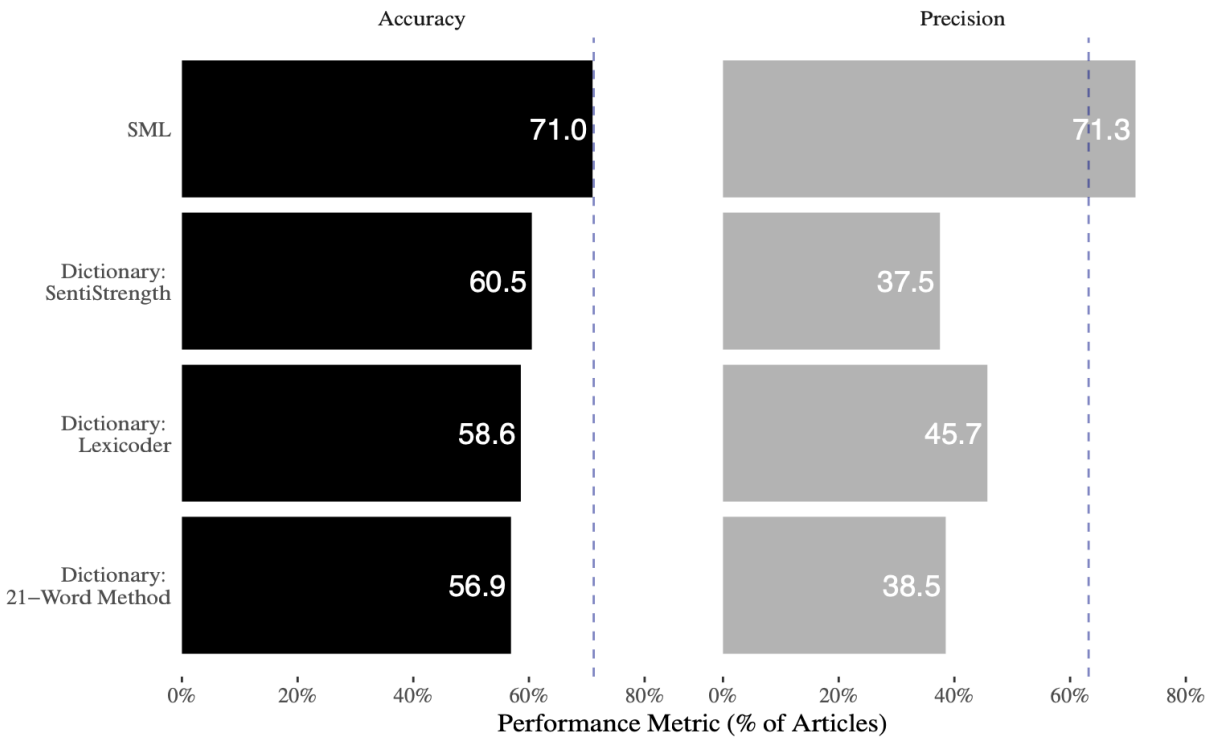
Table 2. Pearson correlations (*r*) between Twitter-based emotions and Gallup-Sharecare Well-Being Index estimates across 1,208 US counties

N = 1,208 U.S. counties	Word-level									Data-driven				
	LIWC 2015			PERMA		ANEW		LabMT		Sentence-level (WWBP)	Sentence-level (Swiss Chocolate)		Person-level	
	Positive	Positive (modified)	Negative	Positive	Negative	Valence	Valence (modified)	Valence	Valence (modified)	Affect	Positive	Negative	WWBP Life Sat.	Direct prediction
Life Satisfaction	-.21	-.06	-.32	.22	-.37	-.03	.15	-.27	.01	.29	.24	-.29	.39	.62
Happiness	-.13	.13	-.27	.27	-.17	.04	.18	-.07	.16	.23	.24	-.30	.23	.51
Worry	.11	.01	.03	-.01	.02	.03	-.05	.02	-.04	.00	-.02	.11	-.03	.52
Sadness	.25	-.01	.22	-.19	.18	.09	-.10	.19	-.09	-.18	-.20	.33	-.23	.64

The gray column headers identify the modified LIWC (removed 3 words), LabMT (removed 15 words), and ANEW (removed 2 words) dictionaries (in the text). The color indicates the direction and magnitude of correlation; white cells are nonsignificant, and all others are $P < 0.05$ corrected for multiple comparisons.

Barberá et al. (2021)

Figure 3: **Performance of SML and Dictionary Classifiers—Accuracy and Precision**



Note: Accuracy (percent correctly classified) and precision (percent of positive articles predicted to be positive) for the ground truth dataset coded by 10 CrowdFlower coders. The dashed vertical lines indicate the baseline level of accuracy if the modal category is always predicted. The corpus used in the analysis is based on the keyword search of *The New York Times* 1980-2011 (see the text for details).

González-Bailón and Paltoglou (2015) on supervised learning for sentiment analysis

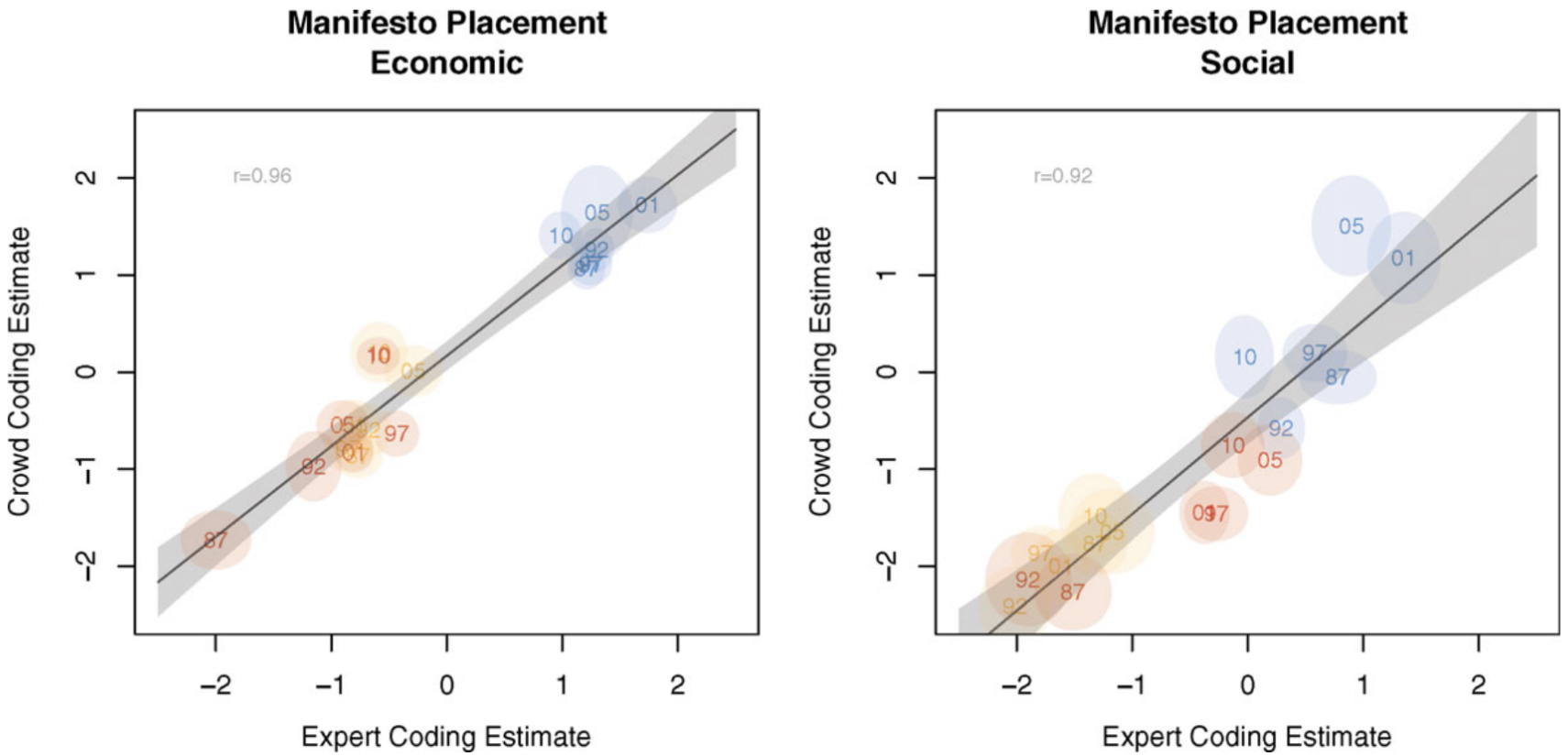
"**Automated content analysis tools offer a trade-off between feasibility and accuracy**: lexicon-based approaches are ready to use and fast to implement, and for this reason they offer an efficient choice when human annotations to train classifiers are unavailable. The validity of their measurement is likely to suffer, however, if the content analyzed relates to specific domains that are not well represented by general purpose dictionaries. The size of the lexicon is less relevant than the correct annotations of the appropriate words, and, when human coding is available, a machine-learning approach can help to scale up the analyses and improve the predictions. For this reason, and on the basis of our findings, **we suggest that future research efforts concentrate on supervised approaches and on building training datasets that can be used to improve learning algorithms and increase their accuracy performance.**"

Creating or finding labeled data

- **Existing labels**
  - Hashtags (can treat these as topics)
  - Twitter profile text (e.g. "I am a Republican")
  - NOMINATE scores for politicians on social media
  - Known party labels of politicians
- **Expert annotation**
  - Manual annotation by researcher
  - Manual annotation by undergraduates / grad students (experts by way of training on a codebook)
- **Crowdsourcing**
  - "Wisdom of the crowds": judgment by ordinary internet users who rate or apply a label to a social media post (Benoit et al., 2016)
  - mTurk or Prolific

Crowdsourcing is surprisingly good (Benoit et al., 2016)

FIGURE 3. Expert and Crowd-sourced Estimates of Economic and Social Policy Positions



Example: Incivility (Theocharis et al. 2016)

In this job, you will be presented with tweets about the 2014 European elections. You will need to classify each tweet into the following series of categories:

1. Polite Vs. Impolite

- **Polite** (a tweet that adheres to politeness standards, i.e. it is written in a well-mannered and non-offensive way) – *@paulmasonews why doesnt #EU take a longer term view? Doesnt #Germany remember their 1940s bailout allowing recovery & growth? #Greece*

- **Impolite** (an ill-mannered, disrespectful tweet that may contains offensive language. This includes: threatening one's rights (freedom to speak, life preferences), assigning stereotypes or hate speech ("nigger", "faggot"), name-calling ("weirdo", "traitor", "idiot"), aspersion ("liar", "traitor"), pejorative speak or vulgarity, sarcasm, ALL CAPS, incendiary, obscene, humiliating.

  – *@Nigel_Farage How's your dirty European non British dirty bitch of a wife? Is she ok? Can't imagine what it's like for you.*

  – *@SLATUKIP – "@DavidCoburnUKip Oh shut up David. You're a bore. @marley68xx"*

Example: Incivility (Theocharis et al. 2016)

The coding process started with a training session in which the coders were introduced to the coding scheme, the software used for coding (i.e. CrowdFlower) and went through a number of short exercises (coding around 40 English language tweets). After the training session all coders were assigned the same 160 English language tweets as a follow-up exercise. This was used to evaluate the overall reliability across all six coders, offer feedback to the coders, and for minor adjustment of the codebook. Given that for the coding of the respective tweets the average reliability was satisfactory across all categories, we went further with assigning the country-specific tweets. As a first step the coders were asked to analyse 1000 tweets. After this stage was finalized, the reliability across all countries was re-assessed and in the cases where the reliability indicators were not satisfactory the coders received detailed feedback. At this point we also introduced the language sub-category to the filter

Measures of coder reliability

- **Average agreement**
- **Correlation**
  - Pearson's correlation coefficient
  - Spearman's ρ (a rank-based statistic)
  - Kendall's τ_B (for ordered data)
- **Inter-coder reliability**
  - These take into account agreement by chance
  - Cohen's κ
  - Krippendorf's α
    - Accounts for more than 2 coders, missing data, and whether continuous, ordered, or discrete data

Example: Incivility (Theocharis et al. 2016)

Table C1: Inter-coder reliability statistics

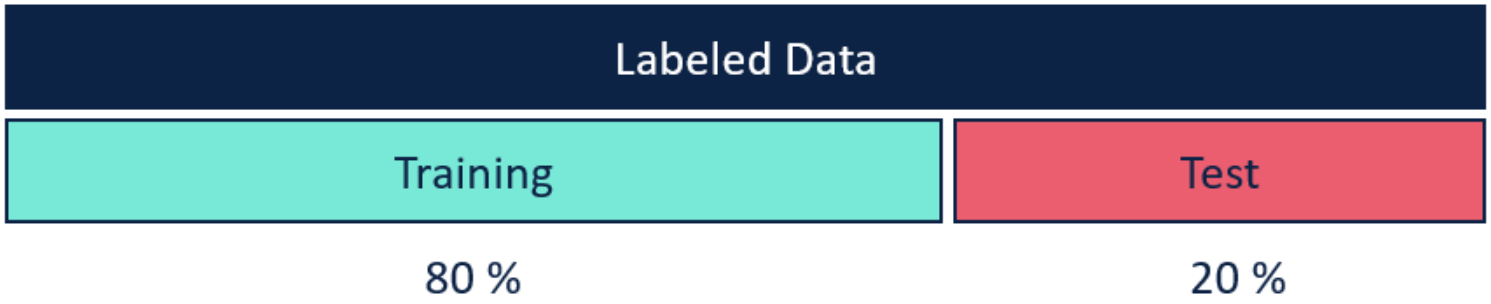
		Germany	Greece	Spain	UK
Summary	Coded by 1/ by 2	2947/2819	2787/2955	3490/1952	3189/3296
	Total coded	5766	5742	5442	6485
Communication	Broadcasting	2755	2883	1771	1557
	Engaging	3011	2859	3671	4928
% Agreement/Krippendorf/Maxwell		79/0.58/0.59	85/0.70/0.70	84/0.66/0.69	85/0.62/0.70
Tone	Impolite	399	1050	121	328
	Polite	5367	4692	5321	6157
% Agreement/Krippendorf/Maxwell		92/0.30/0.85	80/0.26/0.60	93/0.17/0.87	95/0.54/0.90
Morality	Moral	265	204	437	531
	Other	5501	5538	5005	5954
% Agreement/Krippendorf/Maxwell		95/0.50/0.91	97/0.53/0.93	96/0.41/0.92	90/0.39/0.81

Notes: the total number of valid tweets is less than 7,000 because here we exclude tweets we classified as "spam" or in other languages. As measures of inter-coder reliability, we report the percent agreement between the coders for those tweets coded by two coders, Krippendorff's alpha, and also Maxwell score as we consider it most appropriate measure of ICR because it is specifically designed for dichotomous variables.

Basic supervised learning steps

1. Develop a labeled data, often through manual annotation of a small subset of the data
2. Split the labeled data into a "training set" and "test set" at random (e.g. 80% training / 20% test)
3. Train a supervised learning model on the "training set" using cross-validation
4. Use the finalized supervised learning model to test its out-of-sample performance on the "test set"
5. Finally, apply the model to unlabeled data to predict labels for all of the data

Why create a training and test set?

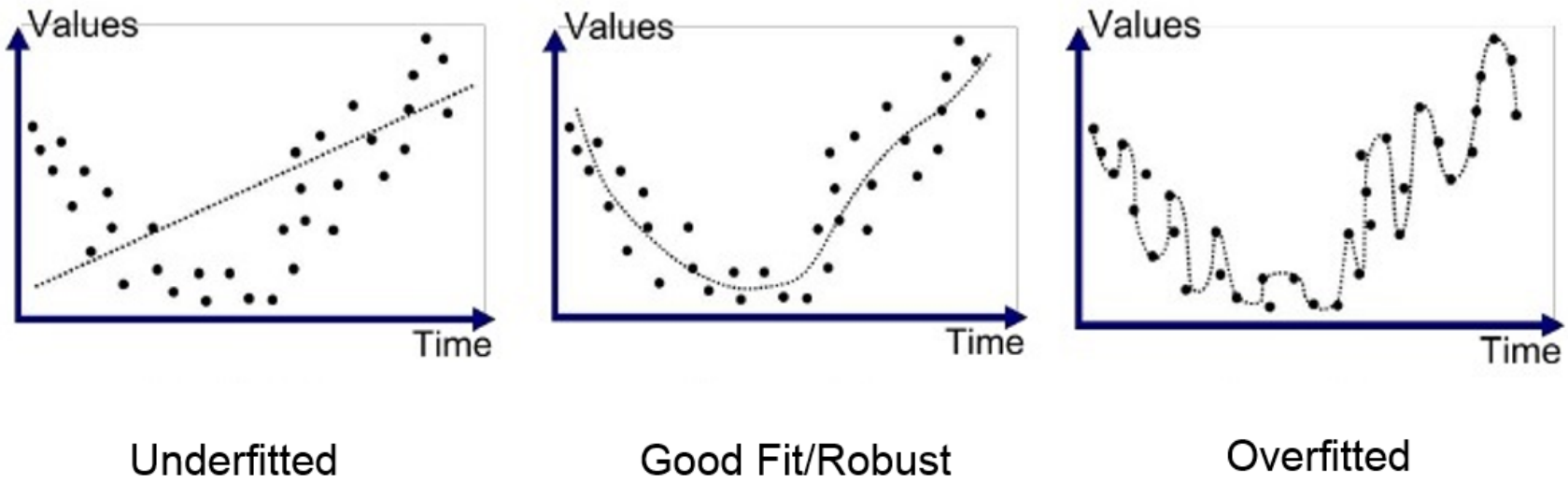


- We want an unbiased estimate of a model's out-of-sample performance

Goals of fitting a model/algorithm to the training set

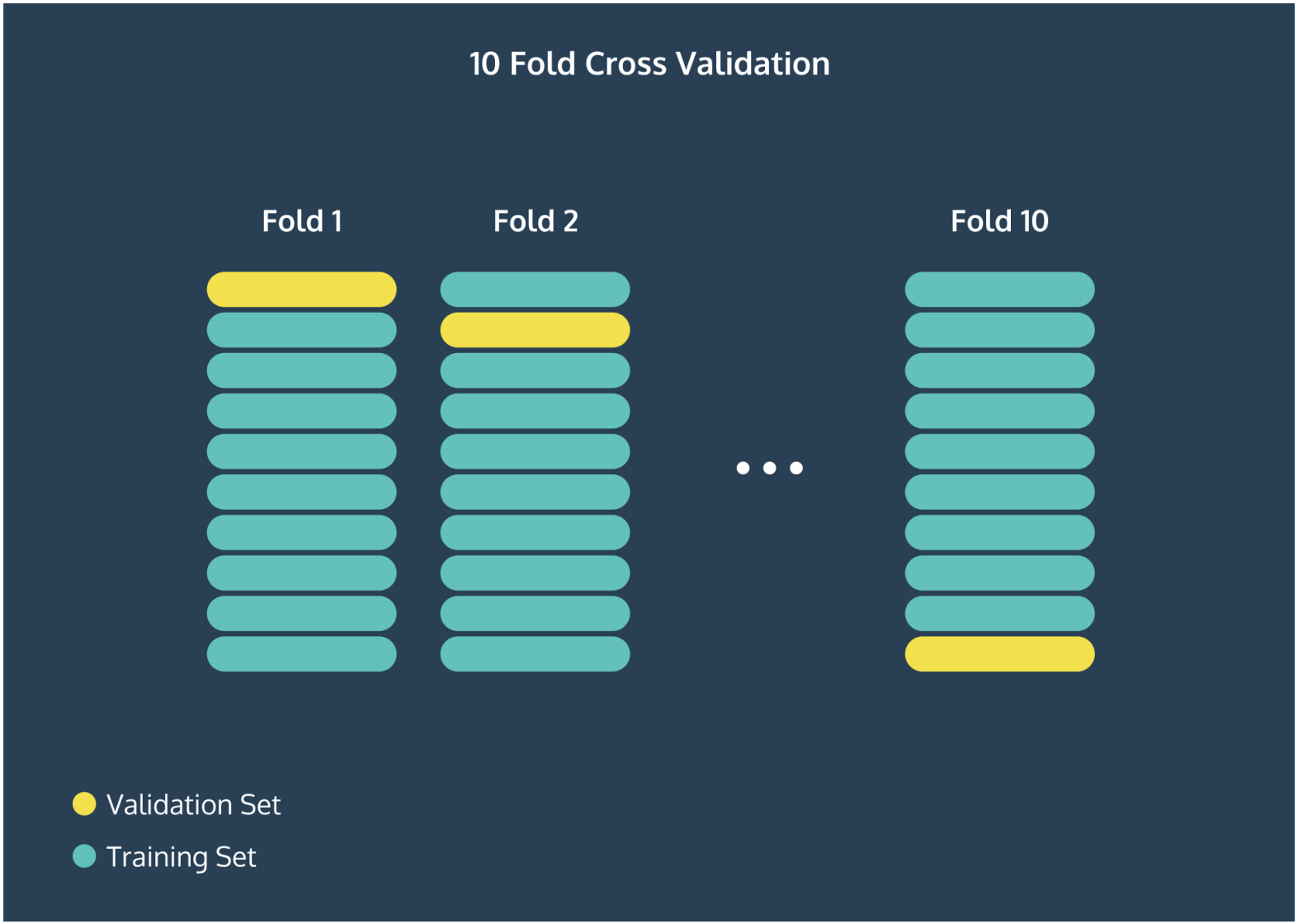
- Out-of-sample prediction: Our goal is to develop a classifier that performs well on data that the model has *never* seen.
- Why? Because we want to predict the labels of other data that we don't yet have labels for (e.g. data that we collect in the future; millions of social media posts that we aren't manually coding)
- We thus want to avoid *overfitting*
  - Overfitting is when a model fits the data it was trained on much better than it does similar data that the model hasn't seen...

A stylized picture of overfitting



But how can we prevent overfitting when training a model? k-fold cross-validation.

1. Split the training data into k sets (typically $k = 10$ or 5)
   - Typically equally sized and randomly assigned
2. For each of the k sets:
   - Hold that set out as a validation set
   - Use the other $k - 1$ sets as the training set
   - Train the model on these aggregated $k - 1$ training data
   - Evaluate the model by predicting the observations in the validation set
3. Fit the model k times, resulting in k evaluations of performance
4. Overall model accuracy can be assessed as the average of these evaluations


10 Fold Cross Validation
Fold 1
Fold 2
Fold 10
Validation Set
Training Set

Model tuning through cross-validation

1. Cross-validation helps us select the **tuning parameters** (also called **hyperparameters**)
2. E.g., in an elastic net model, you need to set the parameter λ prior to fitting the model
3. Tuning parameters are often selected by **grid search**
   - Is a very simple idea: train a model for every combination of hyperparameters that you want to test and see what works best
   - E.g. in elastic net, cross-validate a model first by trying out models with λ set to, say, 0.1, 0.2, 0.5, 1, 1.5, 2

Measuring performance

- If we test multiple machine-learning models, or try multiple hyperparameters for a single model, how do we know which one is "best"?
  - Accuracy
  - Precision
  - Recall
  - F-score

A confusion matrix for binary outcomes

		Actual value	
		Class A	**Class B**
Predicted	**Class A**	True negatives	False negatives
	Class B	False positives	True positives

- Accuracy: correct predictions / total predictions
- Precision: true positives / (true positives + false positives)
- Recall: true positives / (true positives + false negatives)

Confusion matrix example (civil vs. uncivil posts)

		Actual value	
		Civil	**Uncivil**
Predicted	**Civil**	4000	100
	Uncivil	300	200

- Accuracy: 4200 / 4600 = 0.91
  - i.e. Proportion of correct predictions
- Precision: 200 / 500 = 0.40
  - i.e. Proportion of correct positive predictions
- Recall: 200 / 300 = 0.67
  - i.e. Proportion of positives predicted as positive

The precision versus recall trade-off

High recall, but **low precision**

		Actual value	
		Civil	**Uncivil**
Predicted	**Civil**	0	0
	Uncivil	4000	100

Accuracy: 100 / (4000 + 100) = 0.02
Precision: 100 / (4000 + 100) = 0.02
Recall: 100 / (100 + 0) = 1

High precision, but **low recall**

		Actual value	
		Civil	**Uncivil**
Predicted	**Civil**	4000	80
	Uncivil	0	20

Accuracy: 4020 / (4000 + 100) = 0.98
Precision: 20 / (0 + 20) = 1
Recall: 20 / (20 + 80) = 0.20

F-score

- A balance between recall and precision
- Punishes an algorithm as the difference between recall and precision increases

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Example of how F score punishes differences in precision & recall

Recall	Precision	Average	F-score
0.7	0.7	0.7	0.700
0.65	0.75	0.7	0.696
0.6	0.8	0.7	0.686
0.5	0.9	0.7	0.643

Machine learning models/algorithms

- There are *tons*
- Much of machine learning—as a branch of computer science—is dedicated to developing and improving classifiers, and finding fast ways to fit those models to data
- Naive Bayes, lasso/ridge/elastic net, kNN, SVM, random forests, XGBoost
- Ensemble methods (a weighted combination of the best of each method)

A simple machine-learning model: lasso, ridge, and elastic net

OLS:

$$y_i = \beta_0 + \beta_1 X_i^{(1)} + \beta_2 X_i^{(2)} + \ldots + \beta_n X^{(n)} + \epsilon_i \tag{1}$$

Recall that the standard OLS setup minimizes the sum of squared errors:

$$\hat{\beta} = \arg\min_{\beta} = \sum_{i=1}^{N} (Y_i - X_i^{\mathsf{T}} \beta)^2 \tag{2}$$

Regularized regression penalizes the size of regression coefficients:

Lasso regression:

$$\hat{\beta} = \arg\min_{\beta} = \sum_{i=1}^{N} (Y_i - X_i^T \beta)^2 + \lambda ||\beta||$$

Ridge regression:

$$\hat{\beta} = \arg\min_{\beta} = \sum_{i=1}^{N} (Y_i - X_i^T \beta)^2 + \lambda ||\beta||^2$$

Elastic net:

$$\hat{\beta} = \arg\min_{\beta} = \sum_{i=1}^{N} (Y_i - X_i^T \beta)^2 + \lambda_1 ||\beta|| + \lambda_2 ||\beta||^2$$

The parameters λ, λ_1, and λ_2 are the **tuning parameters** / **hyperparameters** (i.e. defined by the researcher *before* the model parameters are estimated)

"One might consider why the penalty term is needed at all outside the case where there are more covariates than observations. ... Ordinary least squares is unbiased; it also minimizes the sum of squared residuals for a given sample of data. That is, it focuses on *in-sample* goodness-of-fit. One can think of the term involving the penalty as taking into account the 'over-fitting' error, which corresponds to the expected difference between in-sample goodness of fit and out-of-sample goodness of fit.

–Athey & Imbens (2017)