729G86/TDP030 Language Technology

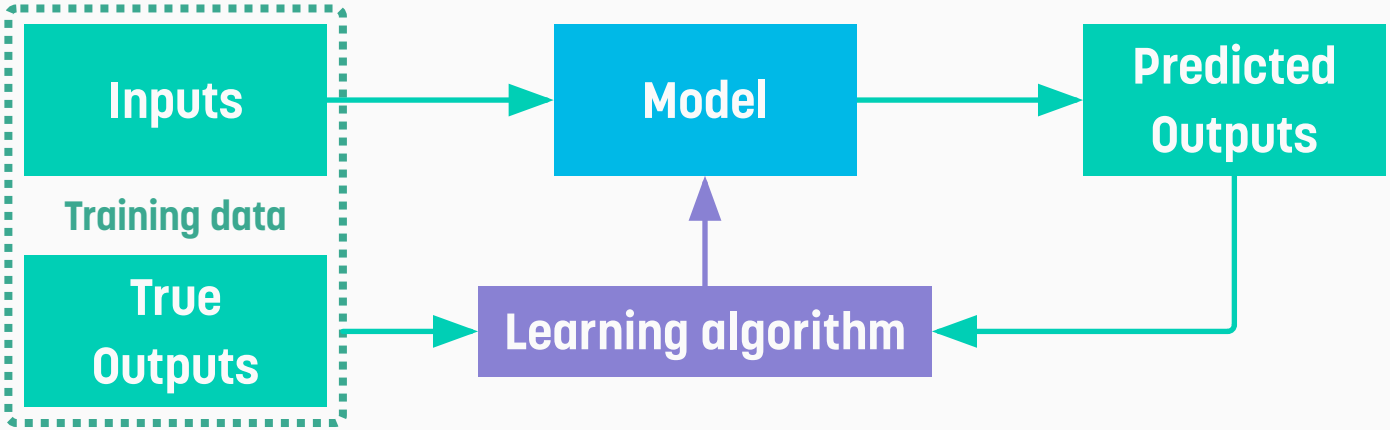
Machine Learning & Text Classification

Marcel Bollmann

Department of Computer Science (IDA)

LINKÖPING UNIVERSITY

- **Machine learning** is the foundation of modern language technology.



- We define a **task** as going from an **input** to an **output**.
- We use a **learning algorithm** to "train" the model to become good at this task.

Recurring questions

1. What **task** do we want to solve, and what **data** do we need to solve it?
   - *e.g.* What are the inputs and outputs that define the task?
2. How do we **prepare our data** before we feed it into the classifiers?
   - *e.g.* How do we turn our text into numerical vector representations?
3. What **models** or **algorithms** can we use and how do they work?
   - *e.g.* Which model can we use for a classification problem? How do we train it?
4. How do we **evaluate** how good our model is?
   - *e.g.* Which evaluation measures are appropriate for the task?

What we will (and won't) cover

- Previous **knowledge of machine learning** is helpful, but not required.
  - We will briefly review basic machine learning concepts.
- This course cannot replace a machine learning class.
  - If you have zero previous knowledge in this area, supplementary reading on the fundamentals of machine learning may be helpful.
- We will focus on aspects **specific to language technology**.
  - *e.g.* We will not go into the mathematical foundations of deep neural networks.

Outline

Text Classification

- Examples
- What we won't look at

Supervised Machine Learning

- Training & Testing
- Statistical vs. Neural Models

Vectorization

- Bag of Words
- Tokenization
- Preprocessing

Statistical Classifiers

- Naive Bayes
- Logistic Regression

Evaluation

- Accuracy
- Confusion Matrix
- Precision, Recall, F1
- Baselines

Text Classification

What is text classification?

Definition

Text classification comprises all classification tasks that categorize entire text documents into predefined, discrete classes.

- The **input** is a **text document** written in natural language.
  - "document" can mean any granularity: sentence, paragraph, social media post, book, ...
- The **output** is a label for the entire document.

Example: Sentiment analysis

I love it so much! The mic works great!!!! I use it for online live classes, cosplay, and to look cute!! The lightup feature really works great! The app also works great too! The sound sounds amazing too! I just wish it had a case for when I travel.

positive

Not durable. The cord came apart from the audio adjuster. The saddest part is that happens only two months after it was purchased, and no force was applied. Definitely, I will not purchase and I do not recommend the item.

negative

Adapted from Amazon

Example: Spam detection

Hello, we are pleased to confirm your order 412116 has been received and will be processed shortly. We will email you an update when we have shipped it. Please see below for the order details and the total price.

not spam

Greetings, we have checked your website and it looks like your website has serious SEO issues. We can get you into the TOP3 search results for a very affordable price. Please contact us on Whatsapp

spam

Adapted from my own inbox

Example: Categorizing issues in hotel reviews

The carpet in our room was stained and looked like it hadn't been cleaned in months.

- Staff
- Cleanliness
- Facilities
- Value for money
- Location

Example: Natural language inference (NLI)

Premise: *A man inspects the uniform of a figure in some East Asian country.*
Hypothesis: *The man is sleeping.*

contradiction

Premise: *Soccer game with multiple males playing.*
Hypothesis: *Some men are playing a sport.*

entailment

Premise: *An older and younger man smiling.*
Hypothesis: *Two men are smiling and laughing at the cats playing on the floor.*

neutral

Source: NLP-progress

Example: Grammatical correctness

I want to try and buy some whiskey.

acceptable

She goes and buying some whiskey.

unacceptable

Who did you get a description of?

acceptable

What did you ask who saw?

unacceptable

Source: Corpus of Linguistic Acceptability

Multi-label text classification

- **Multi-label** classification predicts **multiple labels** per document.

> *The carpet in our room was stained and looked like it hadn't been cleaned in months. Also, the receptionist was very unfriendly when we complained about it.*

Cleanliness Staff

- This requires slightly different techniques and evaluation strategies
  ⟶ *we won't look at this here!*

Classification vs. regression

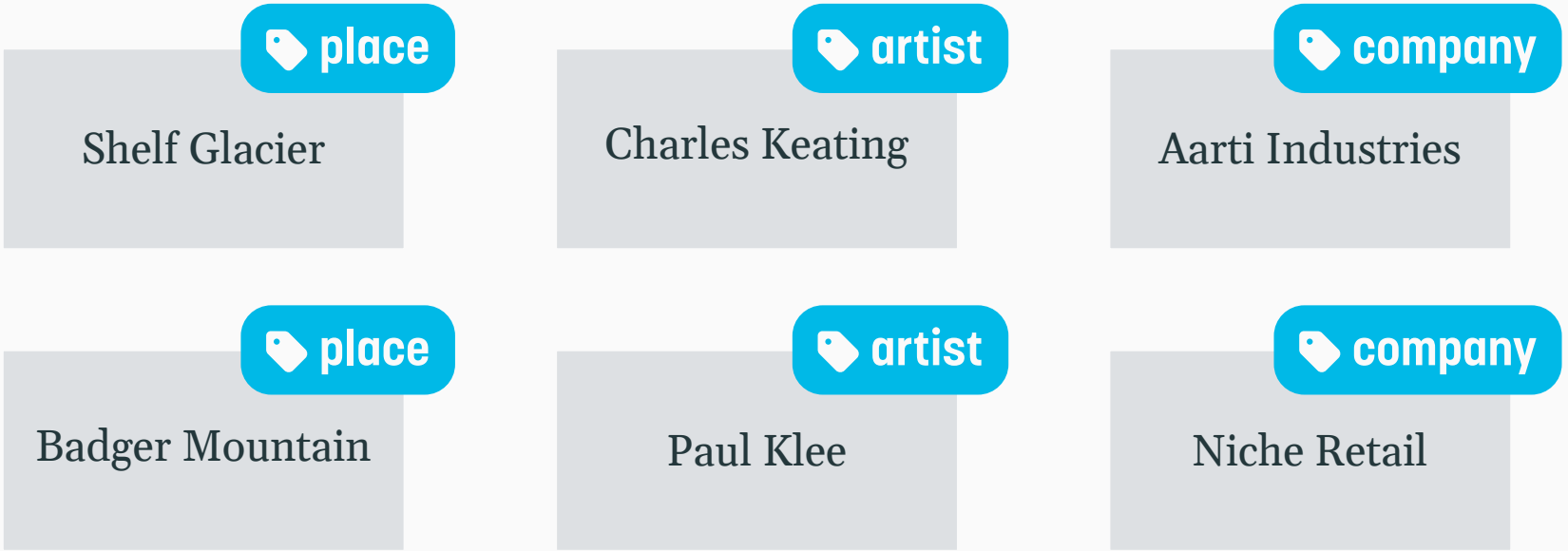
- Language technology tasks are usually **classification** tasks.
- This contrasts with **regression**, where we try to predict continuous values.

> *I love it so much! The mic works great!!!! I use it for online live classes, cosplay, and to look cute!!*

- Regression tasks require different kinds of models and evaluation strategies
  ⟶ *we also won't look at this here!*

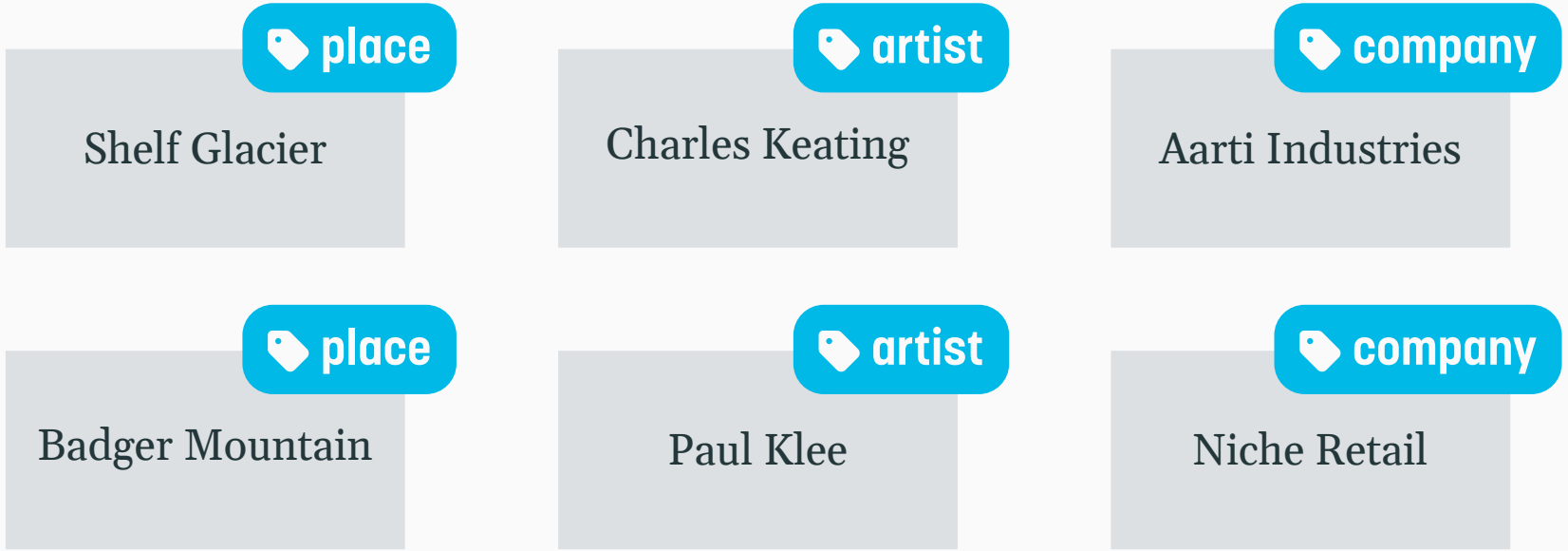
Supervised Machine Learning: Important Concepts

- In supervised machine learning, we need **training data** where each **input** x is annotated with the gold-standard output **label** y.
  - **gold-standard**: the "best available" label that we assume to be correct



Inspired by the DBpedia14 dataset

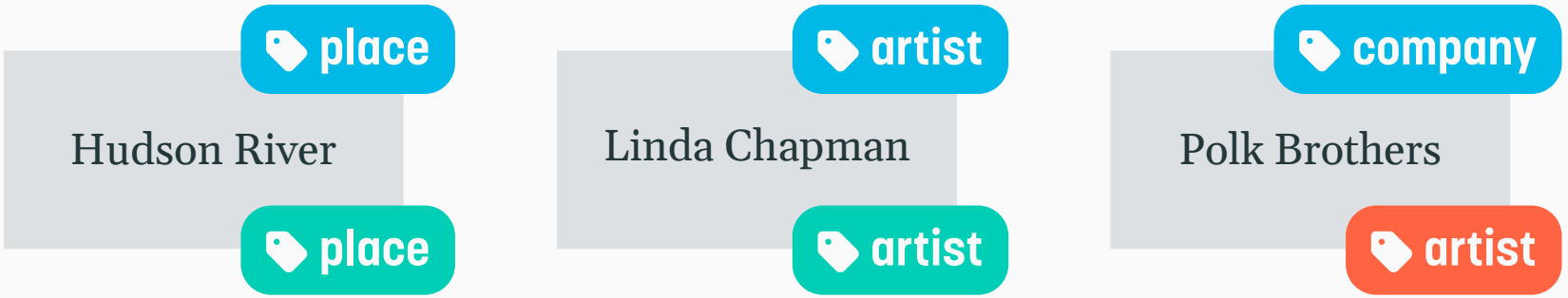
Training a model



- We use this data to **train a classification model**.
  - This means running a **learning algorithm** to set the **parameters** of the model so that it becomes "good" at predicting the correct output y from an input x.

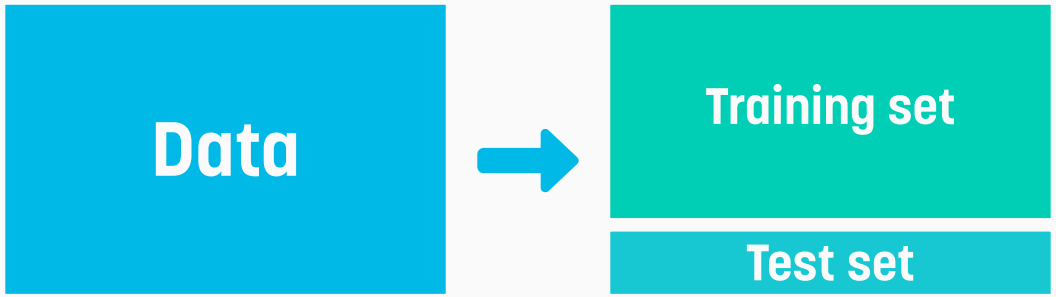
Testing a model

- Our goal is to make the classifier **perform well** on **data it hasn't seen before**!
  - We want the model to **generalize** to new data.
- We estimate this performance on **separate test data**.
  - It's important that the classifier hasn't seen this data before!



Train and test sets

- If we just have one dataset, we can **randomly split** it into **train** and **test** sets.
  - We want to keep the largest part for training.



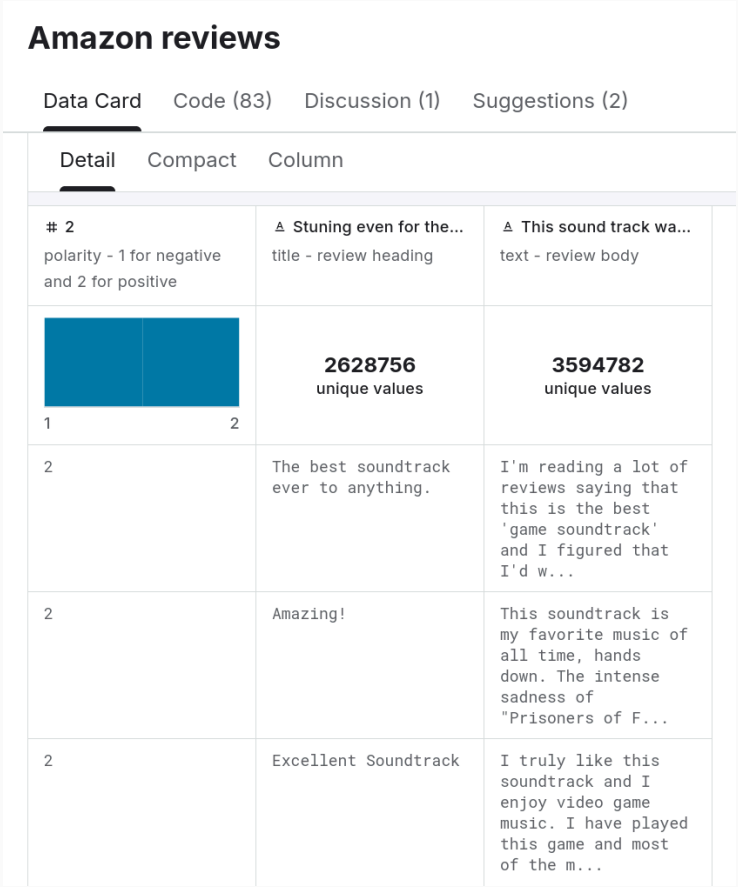
Cardinal rule of machine learning experiments

The test data cannot be used for any purpose prior to the final test!

Serrano et al. (2023)

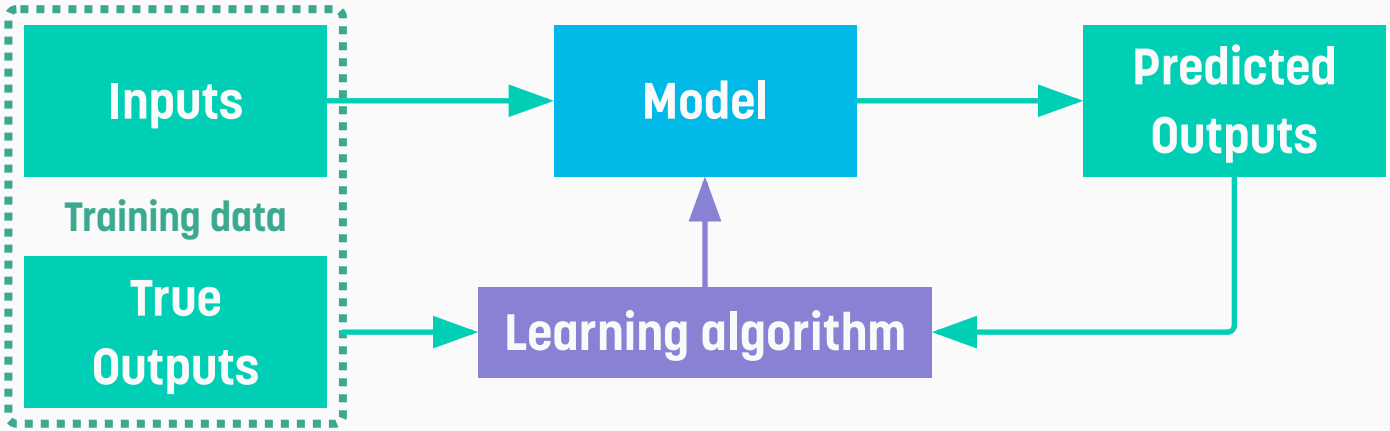
Getting data for training a model

- Sometimes, we can **re-use annotated datasets** that other people have made.
  - But: double-check the quality…
- Sometimes, we can **scrape data** from the web.
  - *e.g.* use the "star rating" from Amazon as the "ground truth" if a review is positive or negative
- If neither option is possible, we might have to **manually annotate** data ourselves.
  - Often requires a lot of time & effort!



Source: Kaggle

- Assuming we have data, what **classification model** can we use?



Statistical machine learning vs. deep learning

- **"Traditional"** classification models
  - Logistic Regression
  - Naive Bayes
  - Decision Trees
  - Support Vector Machines (SVMs)

⬇

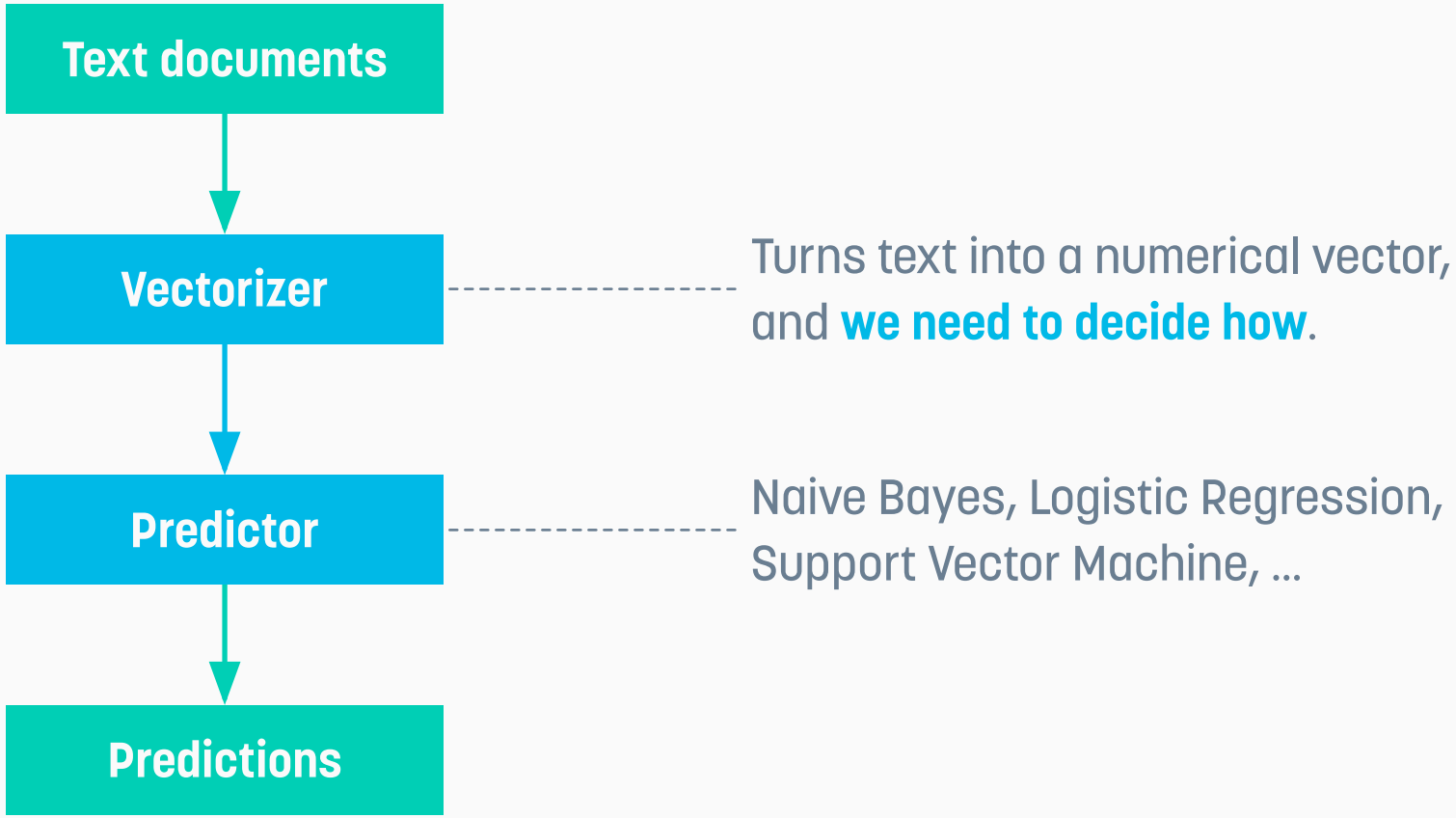
- **Fast** to train & run
- Depending on the task, they can achieve competitive results

- Artificial **neural networks**
  - Recurrent neural networks (RNNs)
  - LSTM networks
  - Transformer architecture
  - Encoder/decoder models

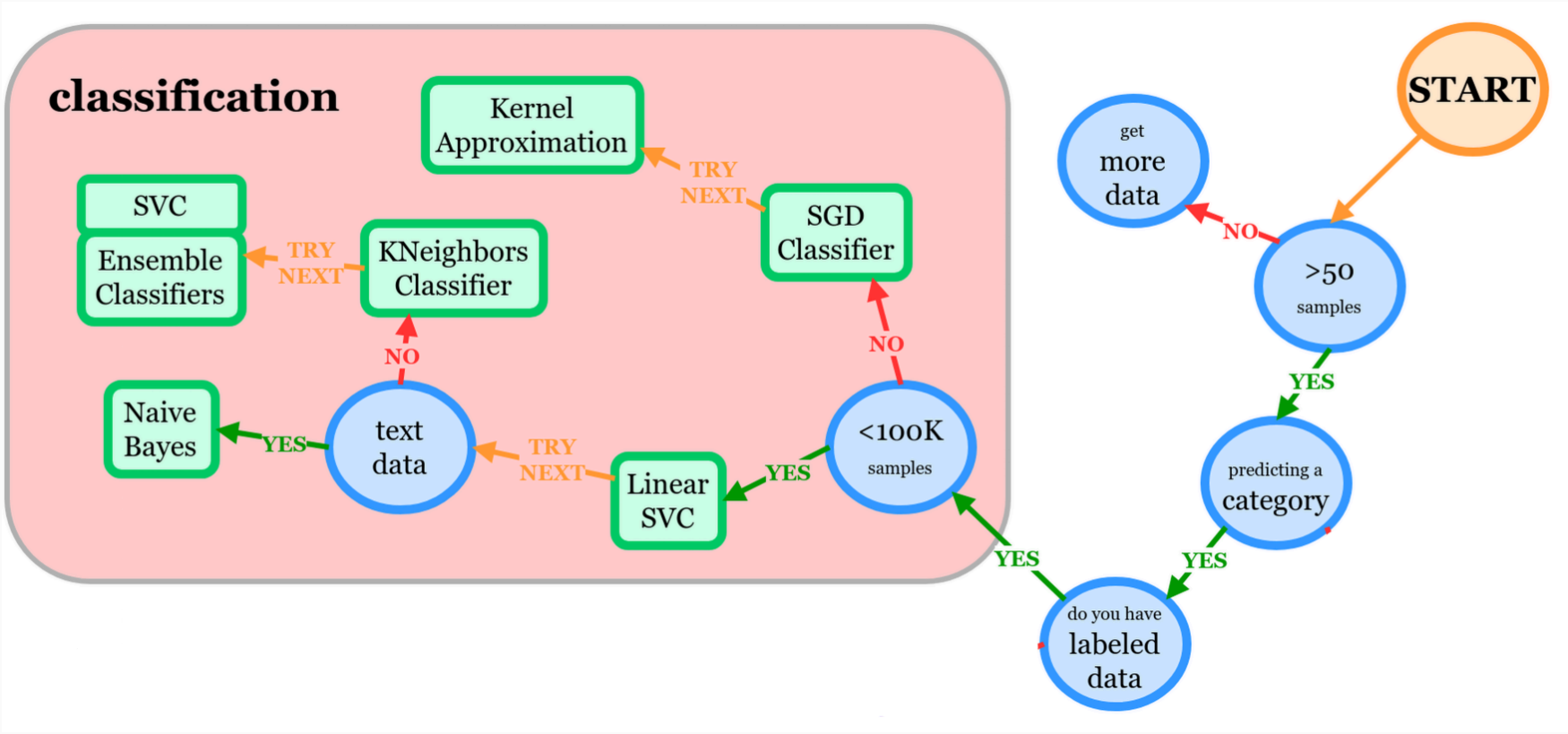
⬇

- **State-of-the-art** on many tasks
- Require powerful hardware to run
- Require a lot more data to train

Statistical machine learning pipeline

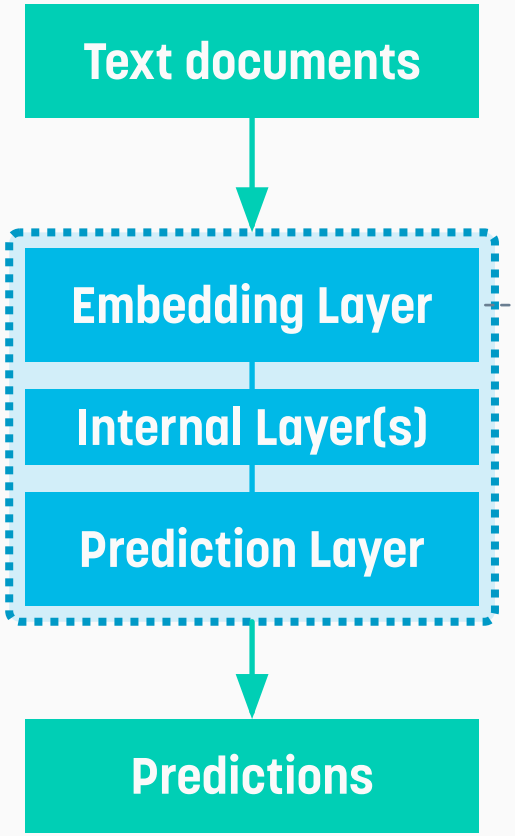


Statistical classifiers in scikit-learn



From the sci-kit learn algorithm cheat sheet

Neural machine learning pipeline



Turns text into a numerical vector, but done "inside" the model.

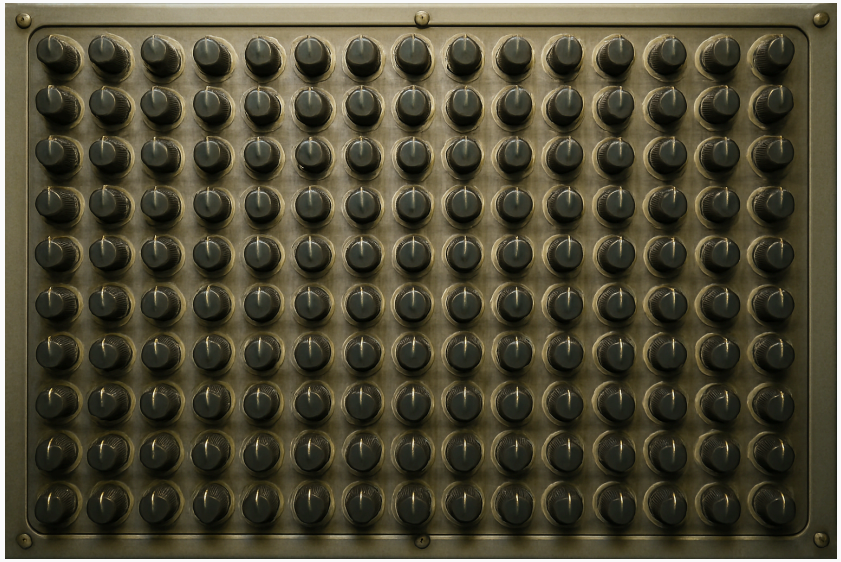
→ Input should be **"raw" text**.

What does “training a model” do?

- **Training a model** means running an algorithm that **tweaks its parameters** so that it best fits the training data.
  - **parameter**: a numerical value that is stored inside the model

- Traditional machine learning typically uses between **a few dozen** and **thousands** of parameters.

- Large language models (based on neural networks) can often have **millions** or **billions** of parameters.



Important Concepts

- training set & test set
- statistical models vs. neural models
- vectorizer & predictor

Turning Text into Vectors

From text to vectors

- Before we can use any statistical classifier, we need to decide **how to represent our text documents** as a numerical vector.

```
"I love it so much! The mic works great!!!! I use it for online live
classes, cosplay, and to look cute!! The lightup feature really
works great! The app also works great too! The sound sounds amazing
too! I just wish it had a case for when I travel."
```

- This involves **preprocessing**, **tokenization**, and **vectorization**.

Bag-of-words approach

- The simplest way to vectorize a text document is to **count how often each words appears**.

- We can turn this into a vector

$$\boldsymbol{v} \in \mathbb{N}^k$$

where

  - k is the **vocabulary size**, *i.e.* the total number of words we can represent
  - $\boldsymbol{v}_n$ is the **count** of the n-th word in our vocabulary (*i.e.* how often that word appears in our text)

a	1
adjuster	0
amazing	1
⋮	⋮
it	3
item	0
just	1
⋮	⋮
will	0
wish	1
works	3

Bag-of-words approach

- The counting approach is also called the **bag-of-words** method.
- Any information about **word order is lost** in a bag-of-words representation.
  - It's like "throwing the words in a bag"...

a also amazing and app case classes cosplay cute feature for for great great great had I I I I it it it just lightup live look love mic much online really so sound sounds The The The The to too too travel use when wish works works works ! ! ! ! ! ! ! ! ! ! ! , , .

adjuster after and and apart applied audio came cord Definitely do durable force from happens I I is it item months no not not Not only part purchase purchased recommend saddest that the the The The two was was will , , ...

How do we split up our text into “words”?

And what happens to punctuation, capitalization, etc.?

Tokenization

Definition

Tokenization is the task of segmenting a text into *words* or *subword* units.

- We want to segment the input text into smaller units, called **tokens**.
- They should **help the model learn** the relationship between input and output.
  - *e.g.* seeing the word "*amazing*" corresponds to the "positive" class

⚡ This is a **much harder problem** than it first seems!

- *e.g.* What is a word? What is a "meaningful" unit?
- Active area of research!

What is a good tokenization?

- What happens if we just **split on whitespace**?

Input

```
'They thought, “Is 9.5 or 525,600 my favorite number?”,
before seeing Dr. Bob’s dog talk.'.split()
```

Output

```
['They', 'thought,', '“Is', '9.5', 'or', '525,600', 'my',
'favorite', 'number?”,', 'before', 'seeing', 'Dr.',
'Bob’s', 'dog', 'talk.']
```

- Do **punctuation marks** really “belong” to the word they are attached to?
  - Key point: ML models won’t be able to “look inside” the tokens we create!

What is a good tokenization?

A better tokenization

```
['They', 'thought', ',', '"', 'Is', '9.5', 'or',
'525,600', 'my', 'favorite', 'number', '?', '"', ',',
'before', 'seeing', 'Dr.', 'Bob', "'", 's', 'dog',
'talk', '.']
```

- Here, punctuation marks become their own tokens, with some exceptions.
  - "Dr." is still a single token
  - Decimal and thousand separators are not split up ("9.5" or "525,600")
- This is traditionally achieved via **regular expressions**.

Example from the Tok-tok tokenizer

Preprocessing steps

```
['Not', 'durable', '.', 'The', 'cord', 'came', 'apart', 'from',
'the', 'audio', 'adjuster', '.', 'The', 'saddest', 'part', 'is',
'that', 'happens', 'only', 'two', 'months', 'after', 'it','was',
'purchased', ',', 'and', 'no', 'force', 'was', 'applied', '.',
'Definitely', ',', 'I', 'will', 'not', 'purchase', 'and', 'I', 'do',
'not', 'recommend', 'the', 'item', '.']
```

- Do we care about **capitalization**? `'Not'` vs. `'not'`
- Do we care about **inflection**? `'purchase'` vs. `'purchased'`
- Do we care about **"non-content" words**? `'the'`, `'and'`, `'I'`

Preprocessing steps

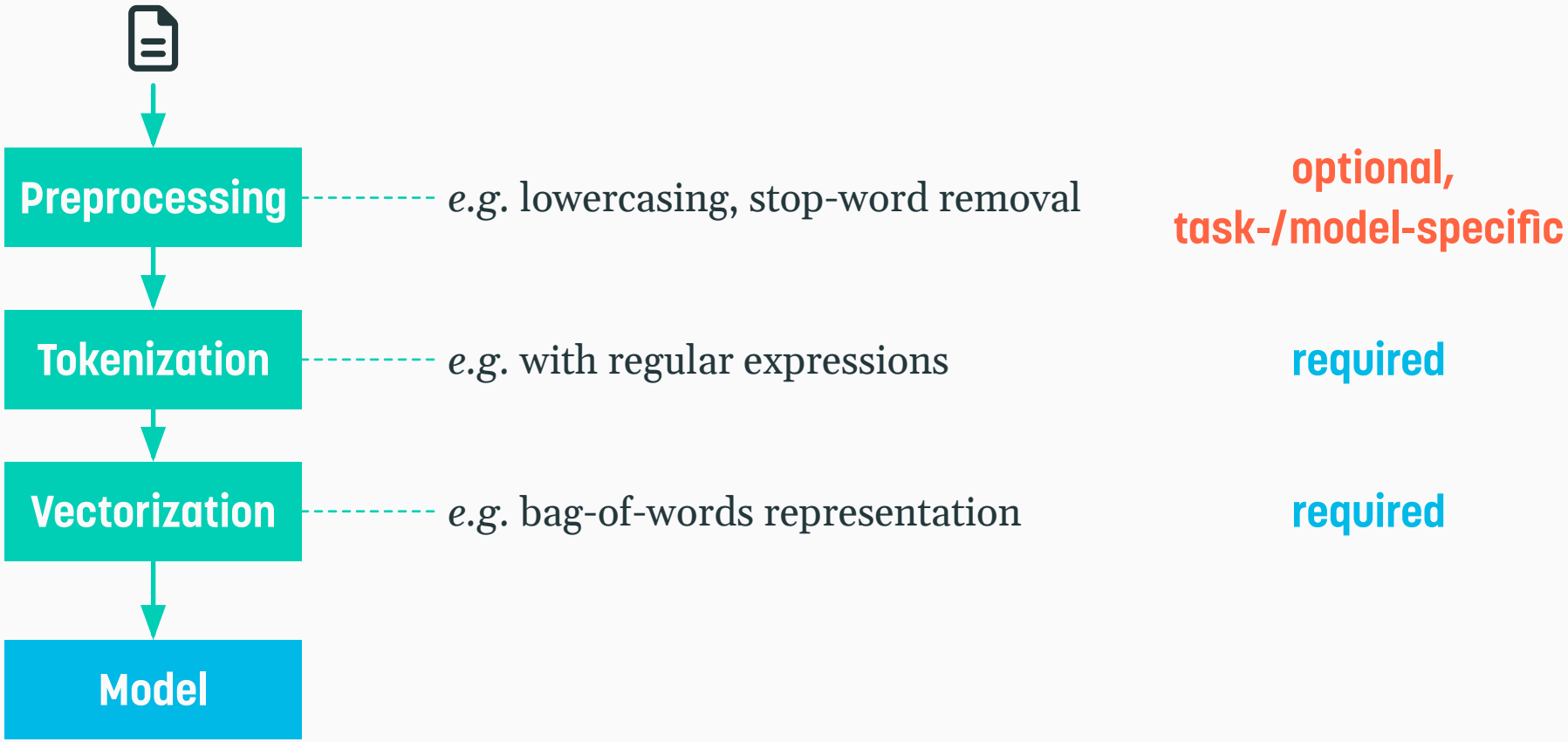
- If we don't care about capitalization, we can **lowercase the input**.
  - **But:** sometimes capitalization might be informative — *Apple* vs. *apple*
- If we don't care about inflection, we can use **stemming** or **lemmatization**.
  - *purchase, purchasing, purchased* ⟶ *purchase*
- If we don't care about words like "*the*" or "*I*", we can use **stop word removal**.
  - "Stop word" = a word that doesn't contribute to the task we are solving
  - Depends very much on the task; there is no universal stop word list!

Examples of possible stop words in English

a about above across after afterwards again against all almost alone along already also although always am among amongst amount an and another any anyhow anyone anything anyway anywhere are around as at back be became because become becomes becoming been before beforehand behind being below beside besides between beyond both bottom but by ca call can cannot could did do does doing done down due during each eight either eleven else elsewhere empty enough even ever every everyone everything everywhere except few fifteen fifty first five for former formerly forty four from front full further get give go had has have he hence her here hereafter hereby herein hereupon hers herself him himself his how however hundred i if in indeed into is it its itself just keep last latter latterly least less made make many may me meanwhile might mine more moreover most mostly move much must ...

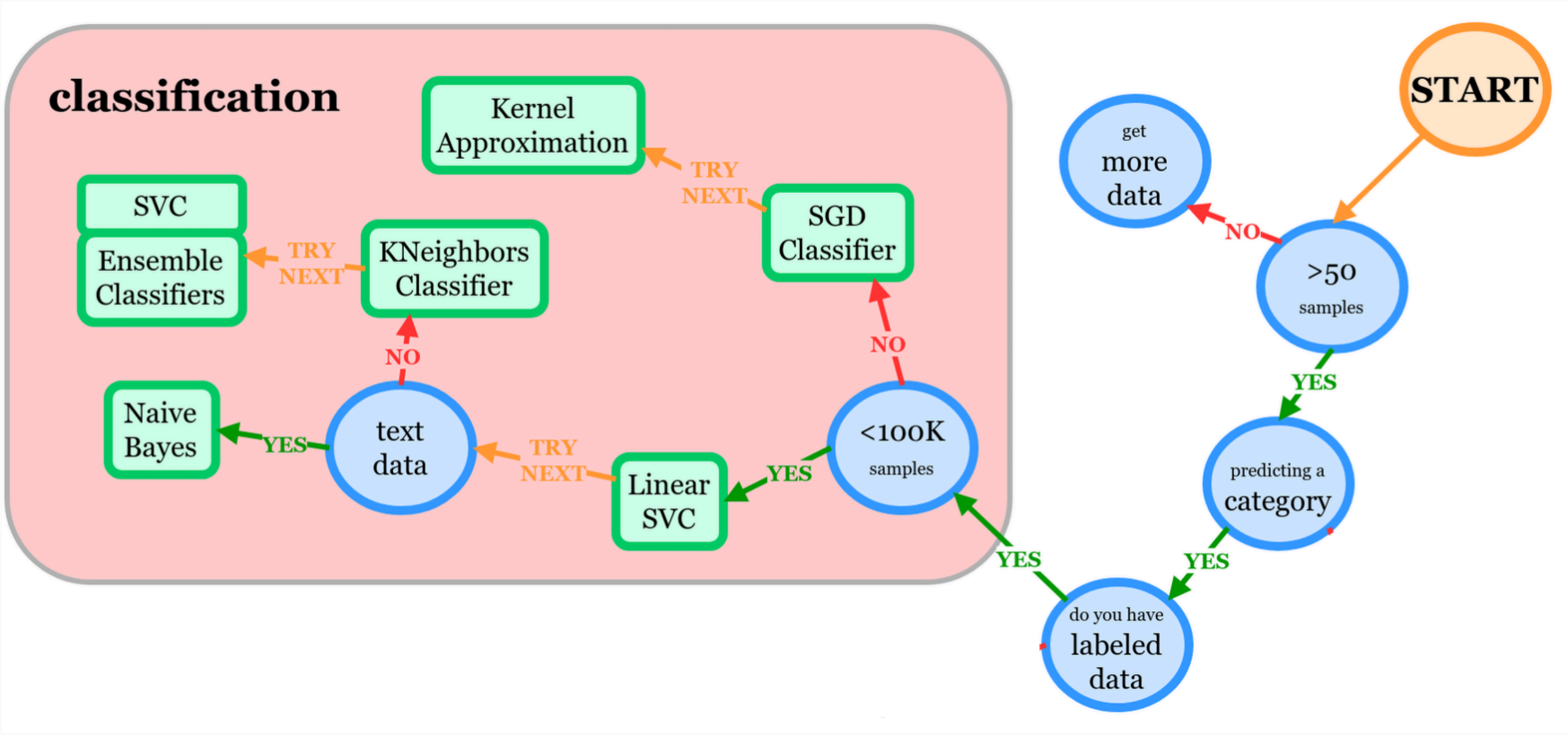
Taken from spaCy

Where are we now?



Statistical Classifiers

Reminder: Classifiers in scikit-learn



From the sci-kit learn algorithm cheat sheet

Many statistical classifiers to choose from…

- **Naive Bayes**
  - Can be trained on very little data, works well with text
  - "Naive" because it assumes all features are independent from each other
- **Logistic regression**
  - Widely-used algorithm, can also work well with text
  - Is the foundation of neural networks
- **Support vector machines (SVM)**, *e.g.* `LinearSVC`
- **Decision trees**
- ⋮

Naive Bayes

- For classification, we would like to know the **conditional probability**:

$$P(\text{class} \mid \text{document})$$

- Naive Bayes learns $P(\text{class})$ and $P(\text{document} \mid \text{class})$.
  - These are easy to estimate, but we won't go into the details here.

- It then uses **Bayes' rule** to convert between the two.

$$P(\text{class} \mid \text{document}) = \frac{P(\text{document} \mid \text{class}) \cdot P(\text{class})}{P(\text{document})}$$

we can ignore this for the prediction

Naive Bayes, informally

Which label has the highest probability, given that the input is “Linda Chapman”?

- $P(\text{place}) \cdot P(\text{“Linda”} \mid \text{place}) \cdot P(\text{“Chapman”} \mid \text{place}) \longrightarrow 10\%$
- $P(\text{artist}) \cdot P(\text{“Linda”} \mid \text{artist}) \cdot P(\text{“Chapman”} \mid \text{artist}) \longrightarrow 54\%$ ← highest, so we predict “artist”
- $P(\text{company}) \cdot P(\text{“Linda”} \mid \text{company}) \cdot P(\text{“Chapman”} \mid \text{company}) \longrightarrow 36\%$

Logistic regression

- Logistic regression learns weights and biases instead of probabilities.

- **Weights** express how "important" each input feature is for a given class.
  - With a bag-of-words vector, the 'features' are the words from the document.

$$w(\text{“River”}, \text{place}) = +10$$
$$w(\text{“Linda”}, \text{artist}) = +8$$
$$w(\text{”Inc.”}, \text{artist}) = -12$$

- **Biases** express that certain classes might be more likely than others.

Logistic regression, informally

score

Which label has the highest ~~probability~~, given that the input is “Linda Chapman”?

- $b(\text{place}) + w(\text{“Linda”}, \text{place}) + w(\text{“Chapman”}, \text{place}) \approx -0.4$
- $b(\text{artist}) + w(\text{“Linda”}, \text{artist}) + w(\text{“Chapman”}, \text{artist}) \approx 2.5$ ← highest score
- $b(\text{company}) + w(\text{“Linda”}, \text{company}) + w(\text{“Chapman”}, \text{company}) \approx 1.7$

From scores to probabilities

- For each possible class j, logistic regression computes a score $z_j \in [-\infty, +\infty]$:

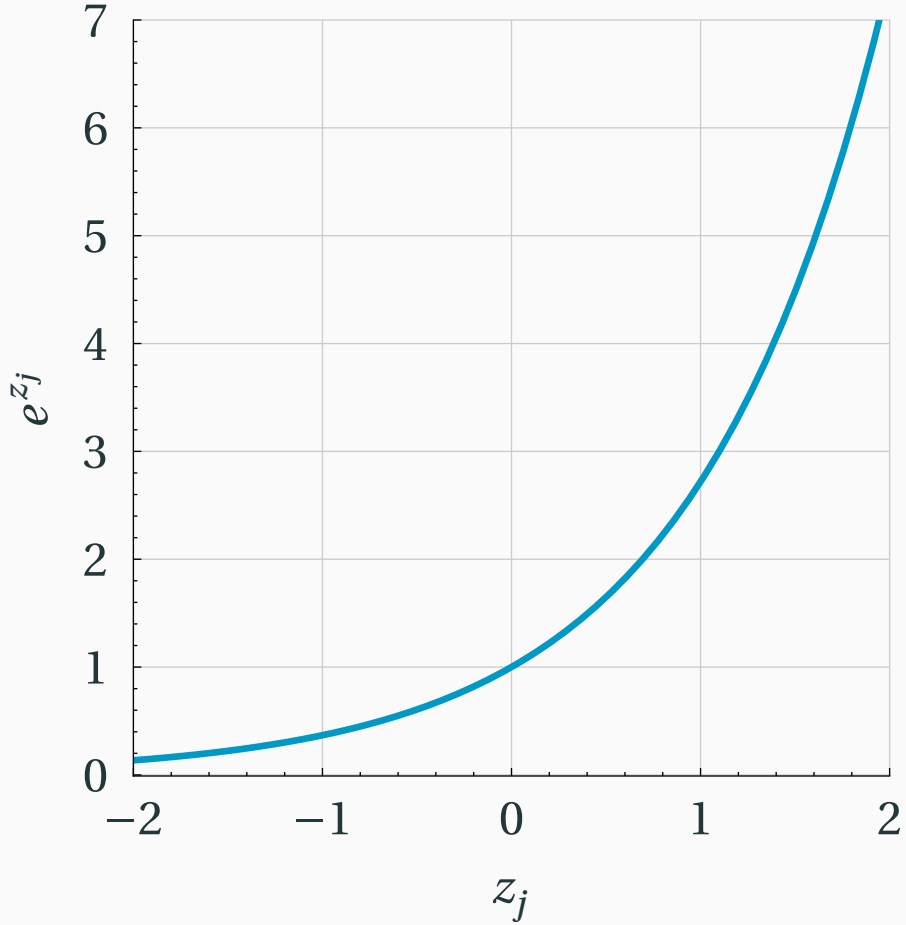
$$z_j = w_j \cdot v + b_j$$

our input vector

- For prediction, we want to **interpret** these scores **as probabilities**.
- The **softmax** function helps us do exactly that:

$$\text{softmax}\left(z_j\right) = \frac{e^{z_j}}{\sum_{i=1}^{K} e^{z_i}}$$

Exponential function



From scores to probabilities

- Before softmax:

place	−0.4
artist	2.5
company	1.7

- Scores are in $[-\infty, +\infty]$

- After softmax:

place	0.04
artist	0.66
company	0.30

- Scores are in $[0, 1]$
- Scores sum up to 1

  → can be interpreted as probabilities!

Statistical classifiers and feature vectors

- Importantly, all statistical classifiers work with **feature vectors** as inputs.
  - **Bag-of-words** is just one way to turn a text document into a feature vector.

- How we **represent our document** as a feature vector determines what the classifier can learn!
  - For example, if "*Apple*" and "*apple*" are represented as different features, the classifier could learn that "*Apple*" is more likely to refer to a company than "*apple*".

Important Concepts

- Naive Bayes
- logistic regression

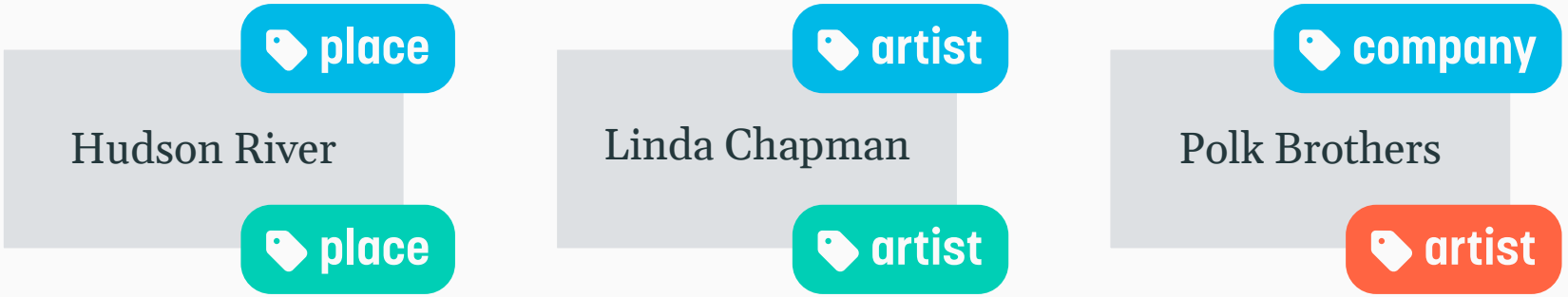
You do not need any of the math from this section for the exam.

- For more details about the algorithms, see Jurafsky & Martin (2025).
  - Chapter 4: Logistic Regression
  - Appendix B: Naive Bayes

Evaluation Metrics

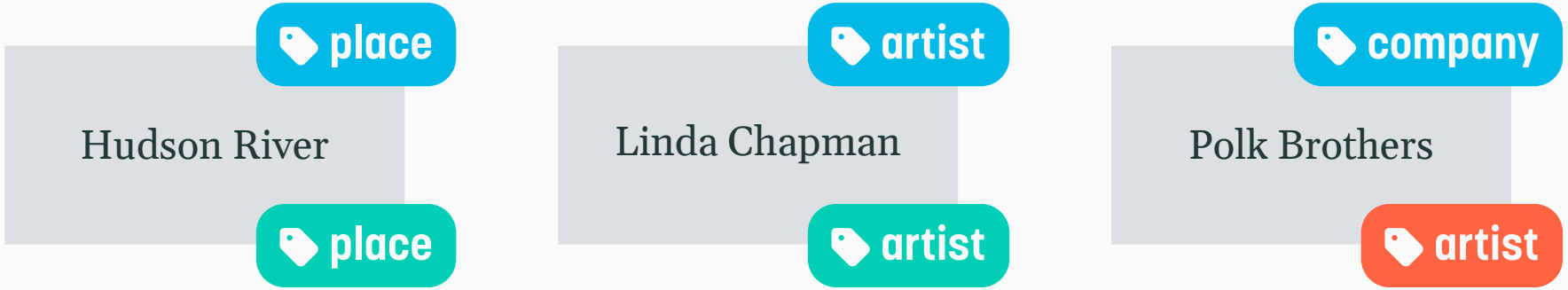
Reminder: Evaluation procedure

- We need a test set with documents and **gold-standard labels**.
  - "gold-standard" = assumed to be correct; *e.g.* produced or verified manually



- We evaluate our classifier by comparing them against the **predicted labels**.

Accuracy



- **Accuracy** is the proportion of documents for which the classifier was "correct."
  - gold-standard label = predicted label

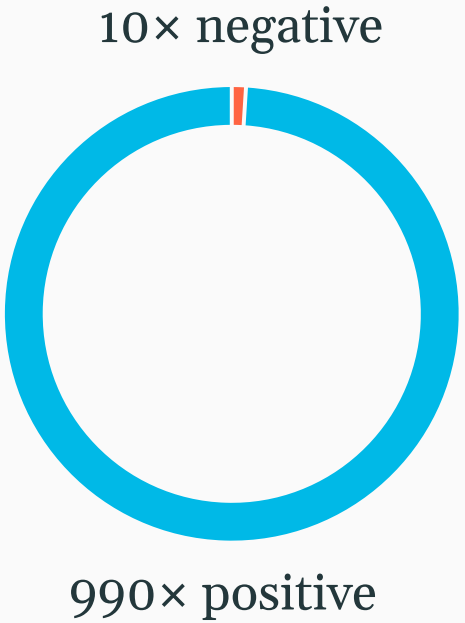
$$\text{accuracy} = \frac{\text{\# of correct predictions}}{\text{\# of all predictions}}$$

Accuracy can be misleading

- If the distribution of class labels is **unbalanced**, accuracy can often be misleading.

Consider…

- Assume a classifier that **always** predicts "positive".
  - Clearly, this is not a very useful classifier…
- If the test data looks like on the right, this "classifier" will achieve an **accuracy of 99%**.



Confusion matrix

gold-standard \ predicted	positive	negative
positive	990	0
negative	10	0

	positive	negative
positive	true positive	false negative
negative	false positive	true negative

Confusion matrix with three classes

		predicted: place	predicted: artist	predicted: company
gold-standard	place	58	6	1
	artist	5	11	2
	company	0	7	43

Number of documents
with the gold-standard label “artist”
where the model predicted “company”

Accuracy from a confusion matrix

	place	artist	company
place	58	6	1
artist	5	11	2
company	0	7	43

$$\text{accuracy} = \frac{\text{\# of correct predictions}}{\text{\# of all predictions}}$$

Precision and recall

Precision and **recall** focus on
how well the model performs
with respect to a specific class.

Precision

When the model predicts x,
how often is it correct?

- The proportion of correctly classified documents (out of all documents) for which **the model predicts** x.

Recall

When the gold-standard is x,
how often is it predicted?

- The proportion of correctly classified documents (out of all documents) for which **the gold-standard class** is x.

Precision with respect to a class

	place	artist	company
place	58	6	1
artist	5	11	2
company	0	7	43

$$\text{precision}(\text{`artist'}) = \frac{\text{\# of correct predictions of `artist'}}{\text{\# of all documents predicted to be `artist'}} = \frac{11}{6 + 11 + 7}$$

Recall with respect to a class

	place	artist	company
place	58	6	1
artist	5	11	2
company	0	7	43

$$\text{recall}(\text{`artist'}) = \frac{\text{\# of correct predictions of `artist'}}{\text{\# of all documents with gold-standard class `artist'}} = \frac{11}{5 + 11 + 2}$$

F1-score

- We often want to achieve a balance between precision and recall.
- The **F1 score** is the **harmonic mean** of the two values:

$$F_1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

- For example, if precision $= 0.3$ and recall $= 0.6$, then:

$$F_1 = \frac{2 \times 0.3 \times 0.6}{0.3 + 0.6} = 0.4$$

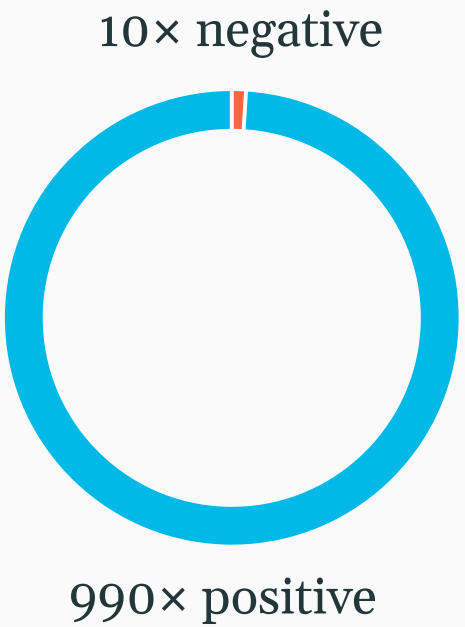
The importance of baselines

"My classifier got an F1-score of 0.74 – is that good?"

- Evaluation metrics are **no absolute measures** of performance.
  - What is a "good" classifier performance is usually *relative*.
- We typically judge a classifier by **comparing it** against another one.
  - Then we can say e.g. "*classifier X has better recall than classifier Y*"
- The point of comparison is also called the **baseline**.
  - Baseline: a classifier that we compare against & want to beat, often a very simple one

Most-frequent-class baseline

- A simple baseline for classification problems is the **most-frequent-class baseline**.
  - A "dummy" classifier that always predicts the most frequent class, no matter what the document is
- If the training data looks as on the right, this baseline would **always predict "positive"**.
  - Any "real" classifier should hopefully be better than that!



Important Concepts

- accuracy
- precision, recall, F1-score
- confusion matrix
- baselines, most-frequent-class baseline