Introduction to Sentiment Analysis
-- *Session 2: Recap* --

Winter Semester 2019/2020
Instructor: Michael Wiegand
Institute for Computational Linguistics
Heidelberg University, Germany

Goals of this Session

- Recapitulation on principles and terminology which will occur in many of the papers presented in this seminar.
- *You should already learned this in some previous lecture(s).*
- I will **not** talk about specific learning algorithms.

Outline

- Machine Learning
- Evaluation

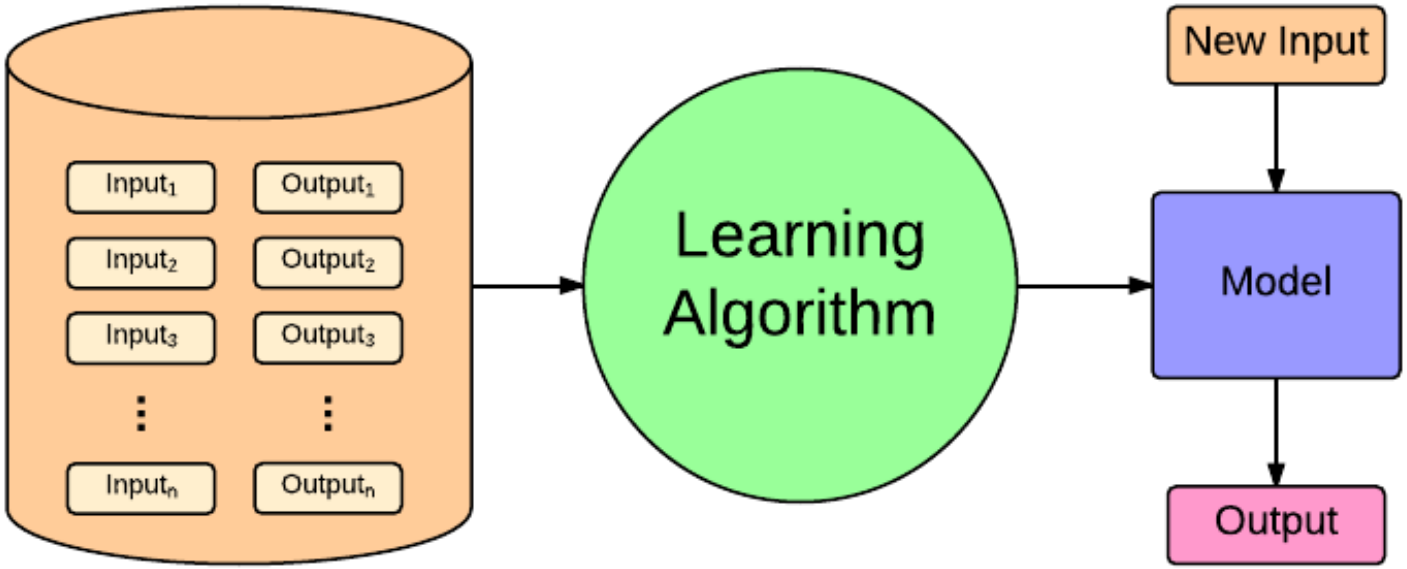
Outline

- Machine Learning
- Evaluation

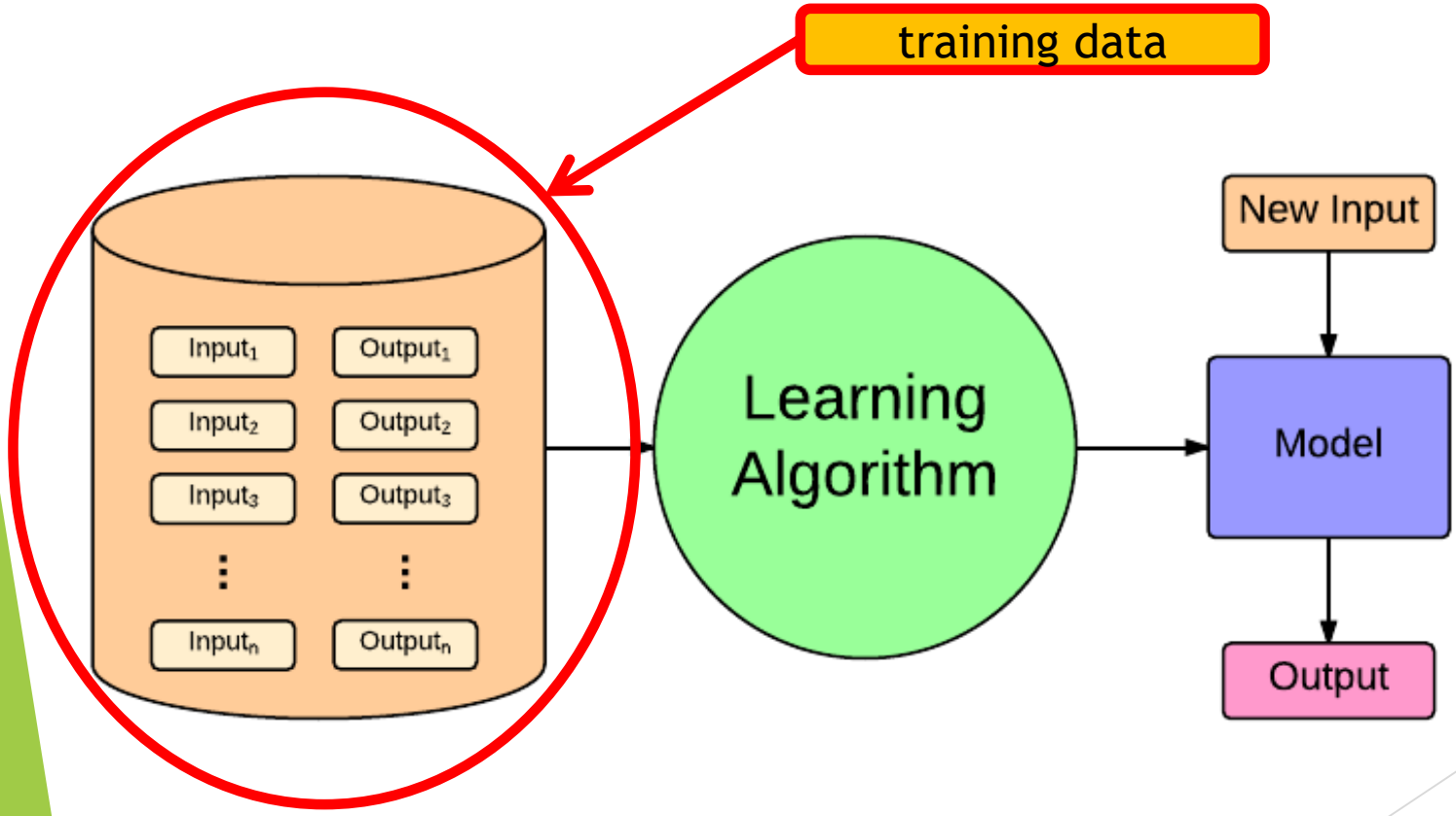
Machine Learning

- Some way to build a classifier.
- Differences to rule-based approach:
  - Do **not** write rules.
  - Define a set of **features** and let a **learning algorithm** find out automatically which set of features (typically translated to feature weights) are most suitable.
- This is a data-driven approach.
- Nowadays, most research problems are coped with in a data-driven manner.

Machine Learning – Pipeline

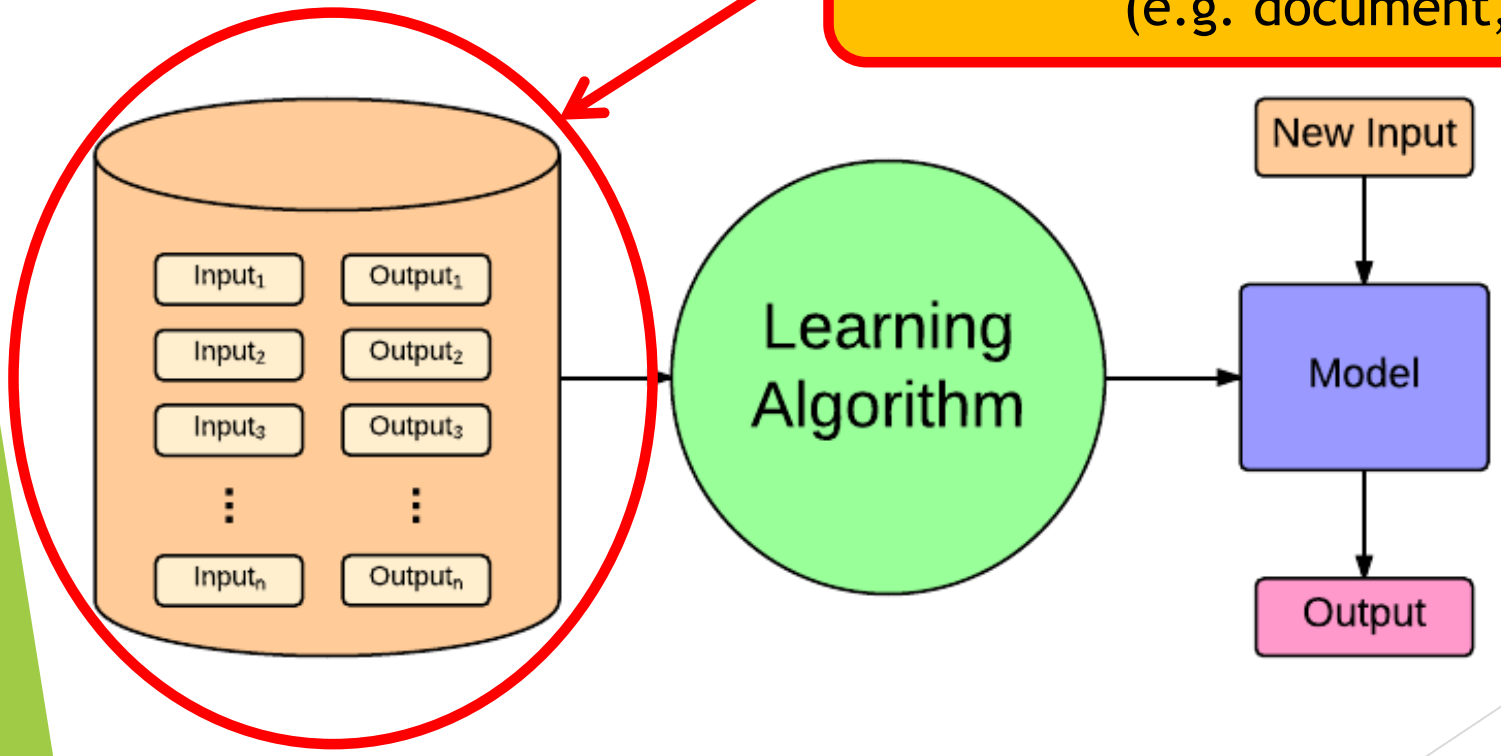


Machine Learning – Pipeline

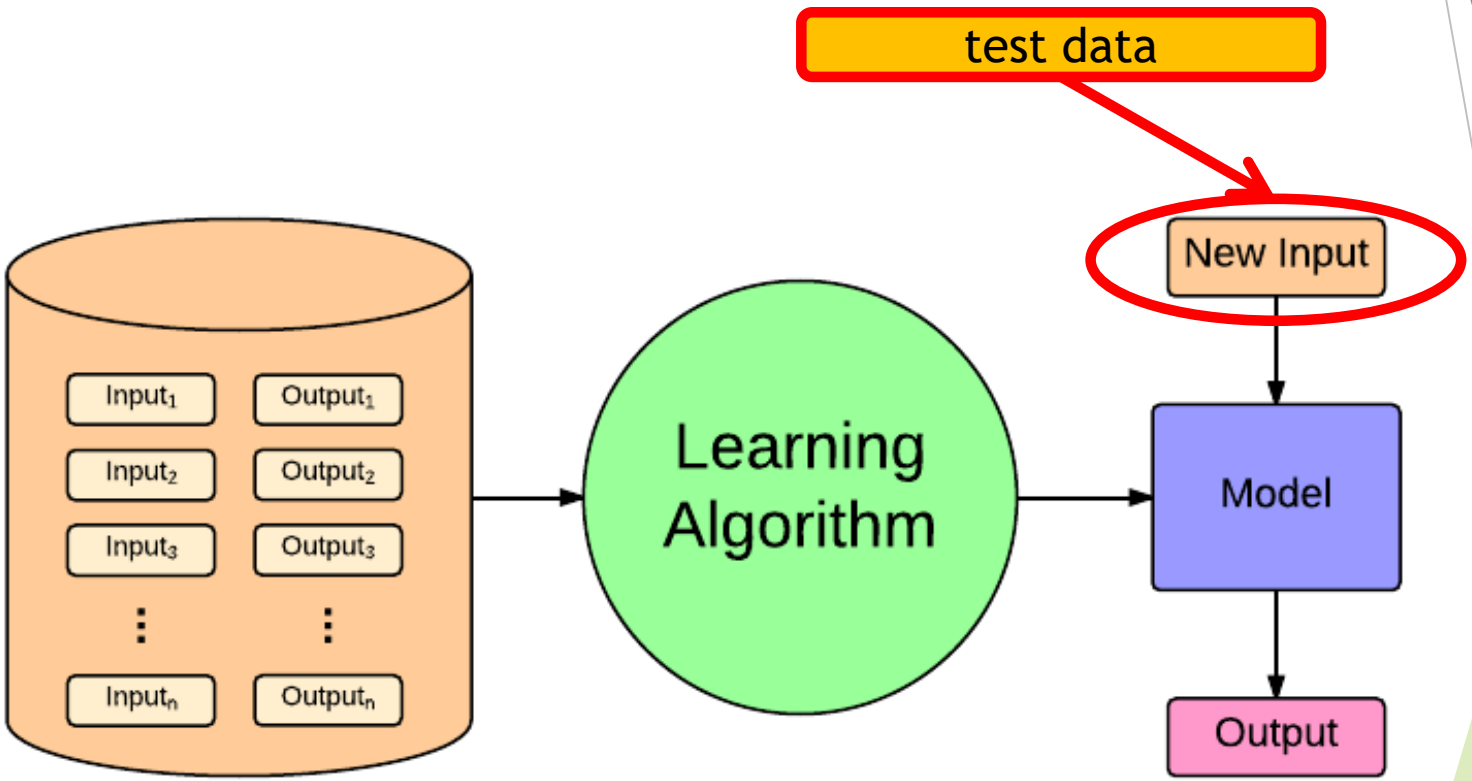


Machine Learning – Pipeline

Comprises labeled **instances**
Instance: unit that is to be classified
(e.g. document, sentence etc.)

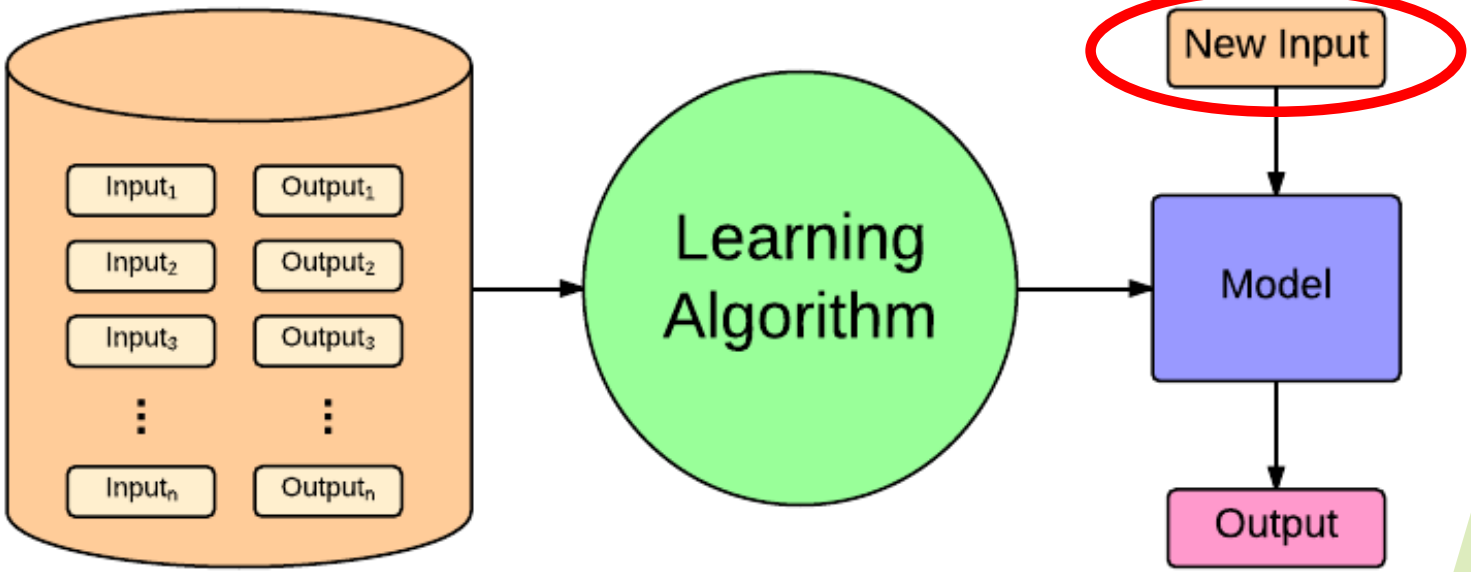


Machine Learning – Pipeline

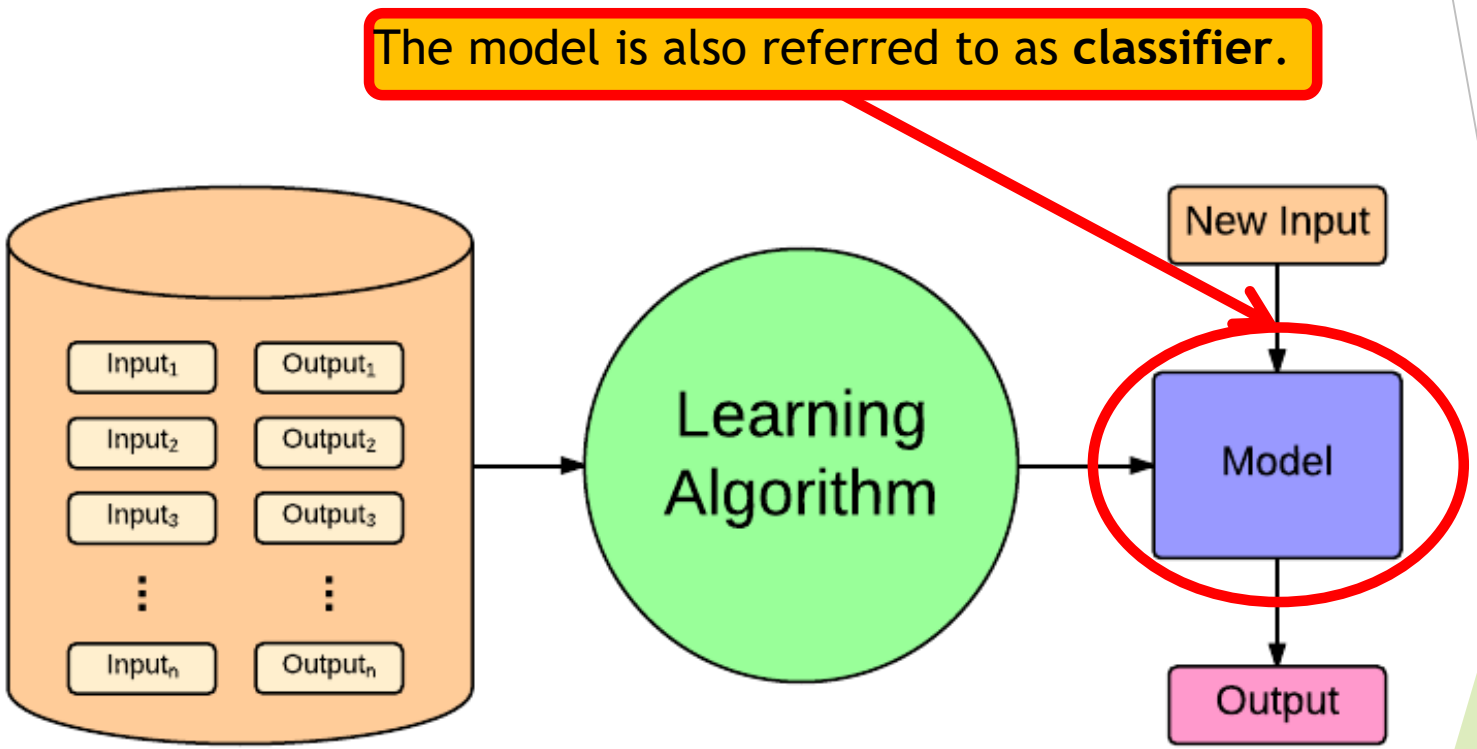


Machine Learning – Pipeline

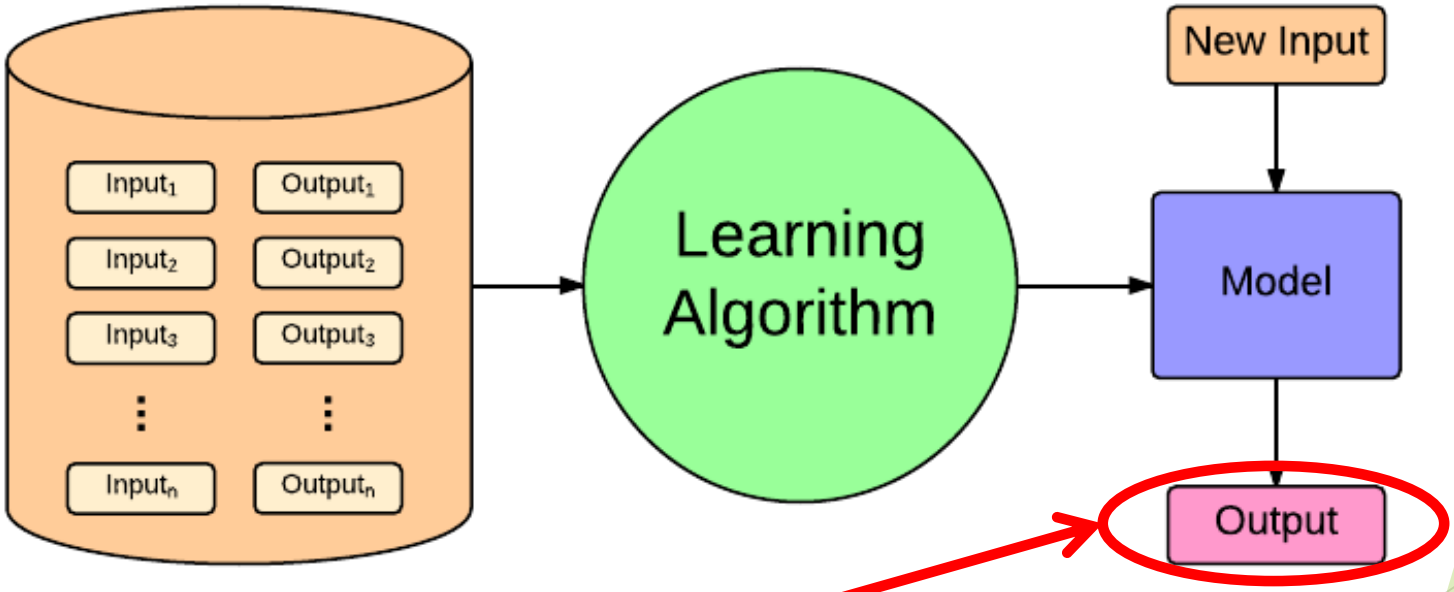
For testing the model, the test data are treated as unlabeled/unknown data.



Machine Learning – Pipeline



Machine Learning – Pipeline

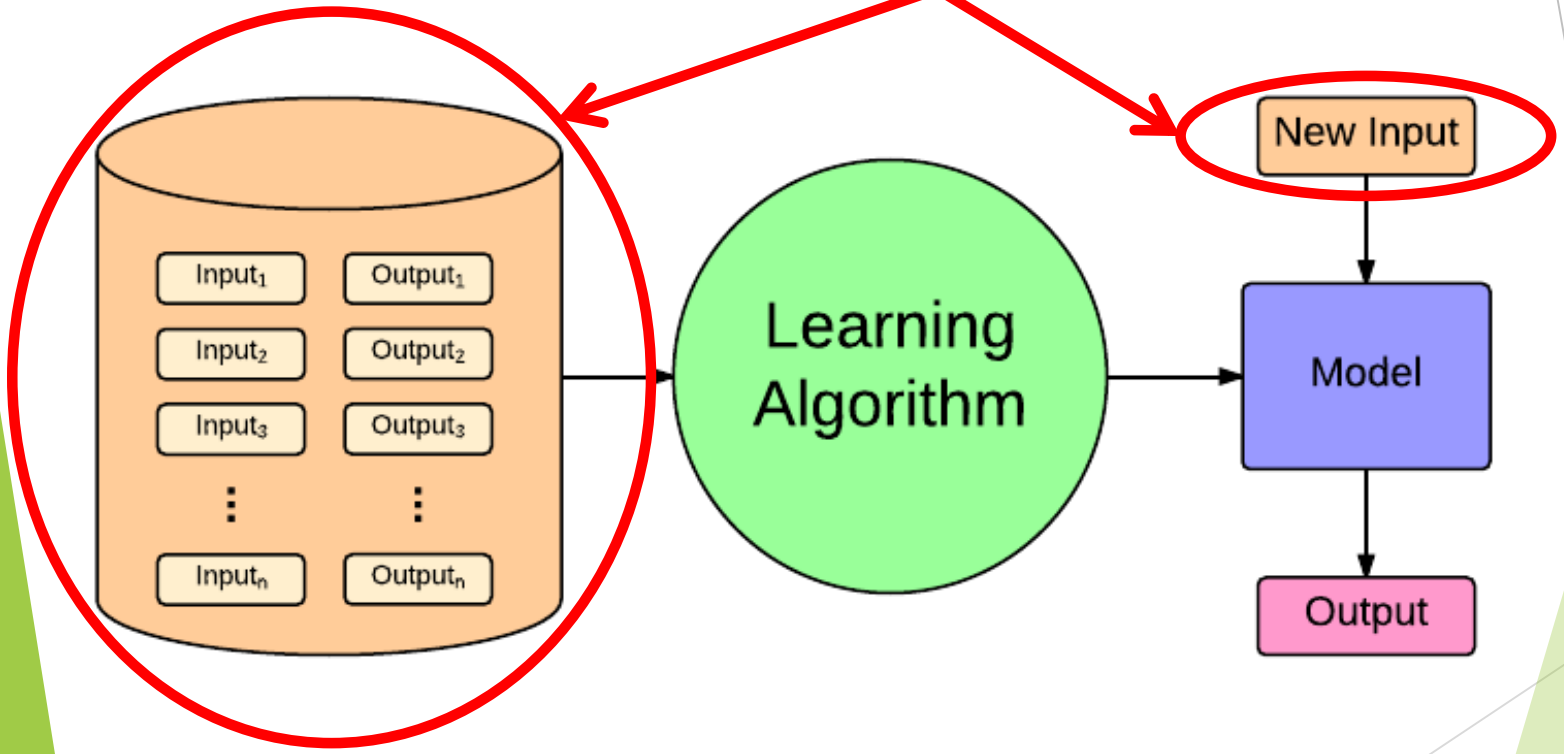


Once the model has produced labels, we can compare the predicted labels against the actual labels → *evaluation*

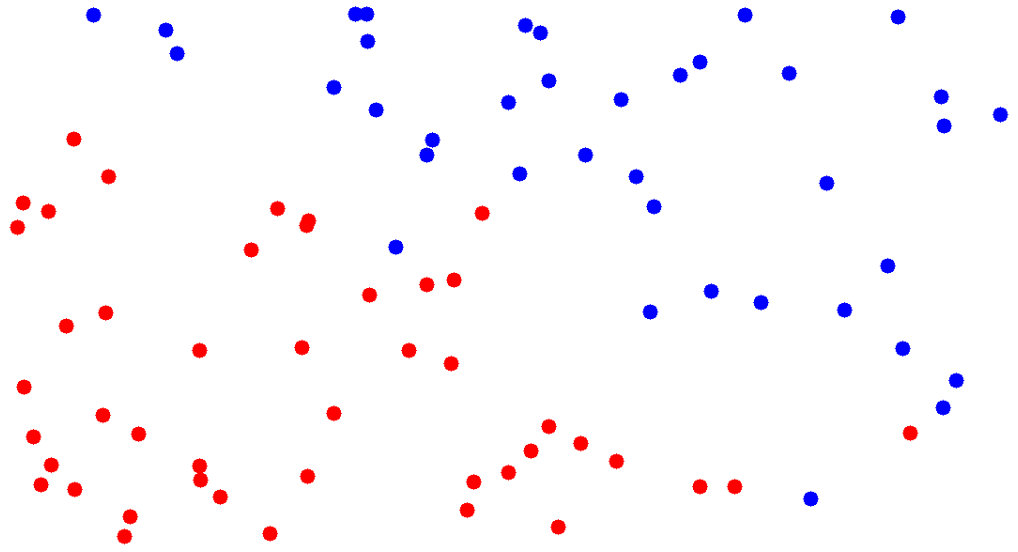
Machine Learning – Pipeline

Training and test data must **not** be identical.

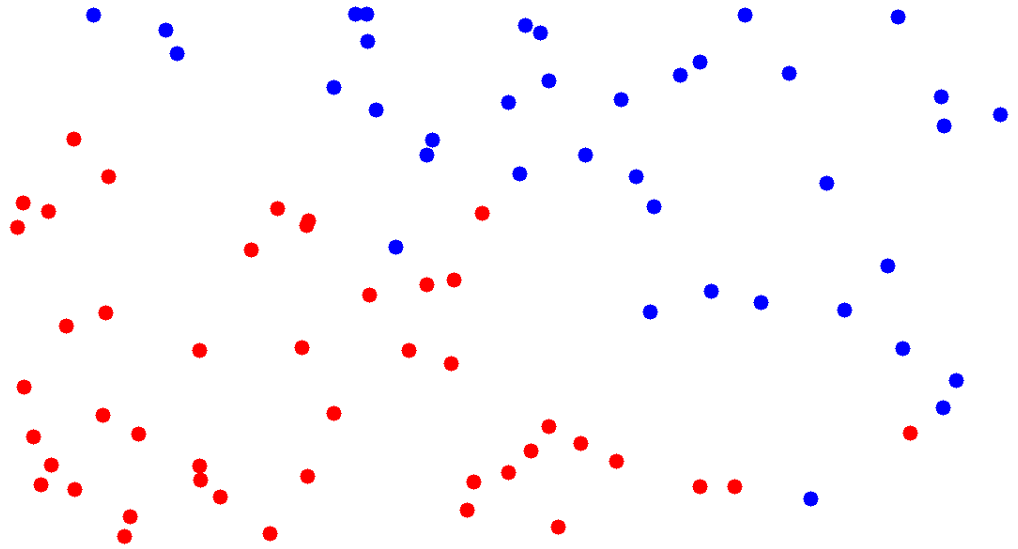
Otherwise, the learned model **overfits**.



How a classifier „sees“ the world

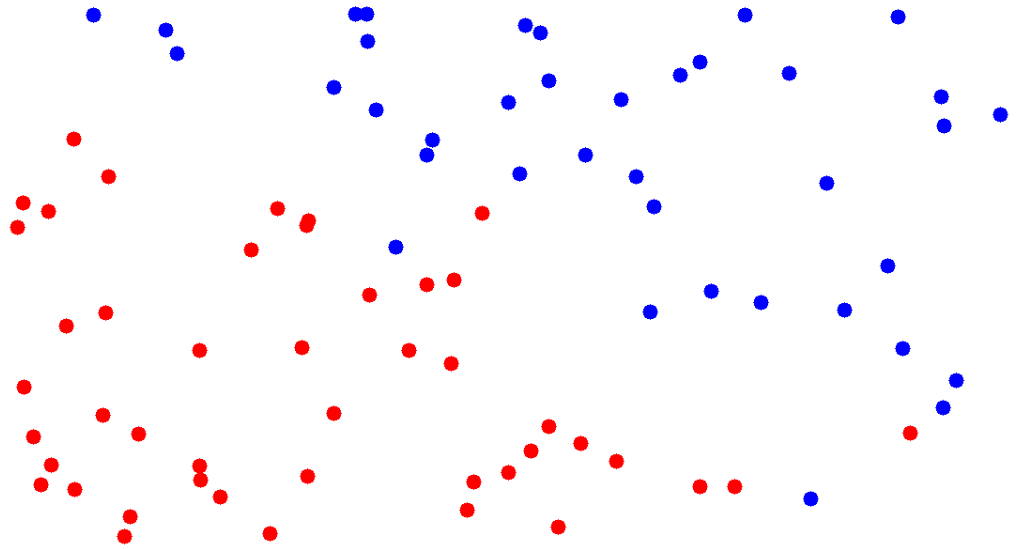


How a classifier „sees“ the world



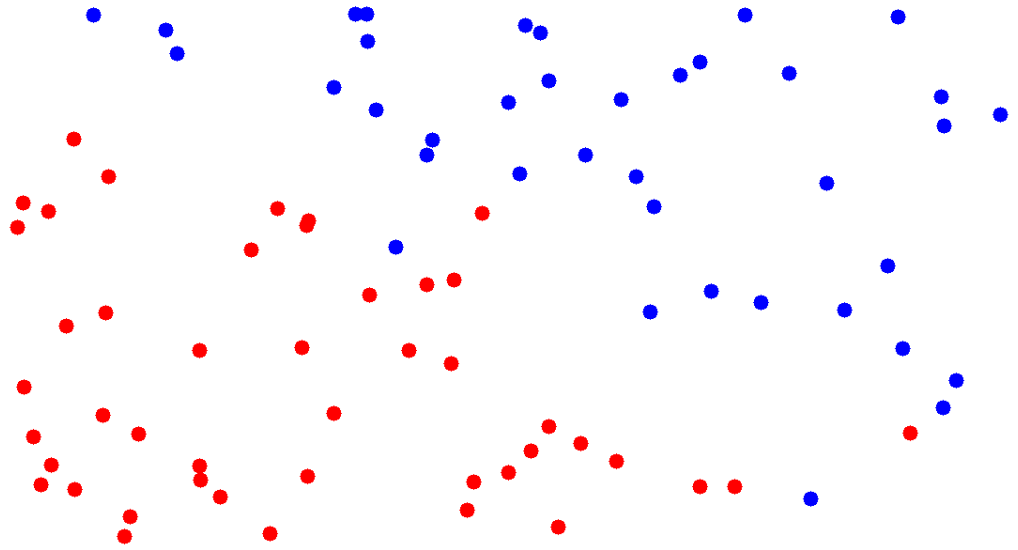
Dots represent data instances in some feature space.

How a classifier „sees“ the world

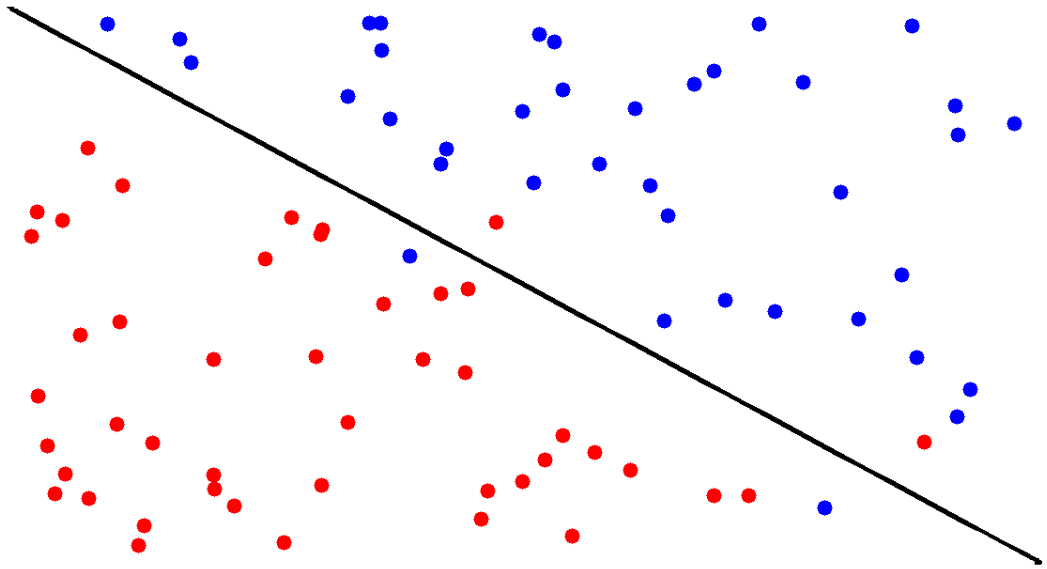


The colours blue and red represent two different classes to be distinguished.

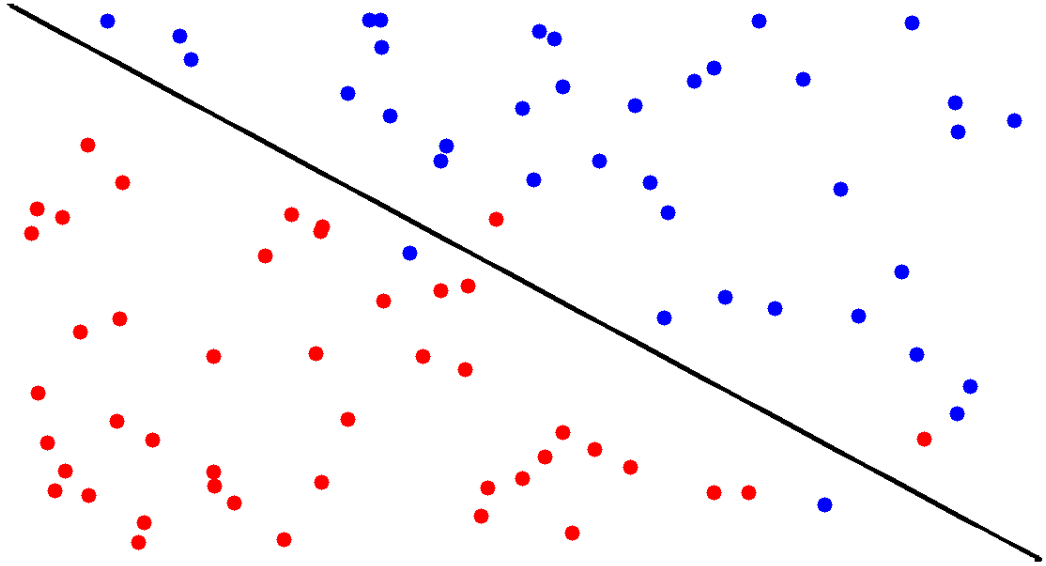
How a classifier „sees“ the world



How a classifier „sees“ the world

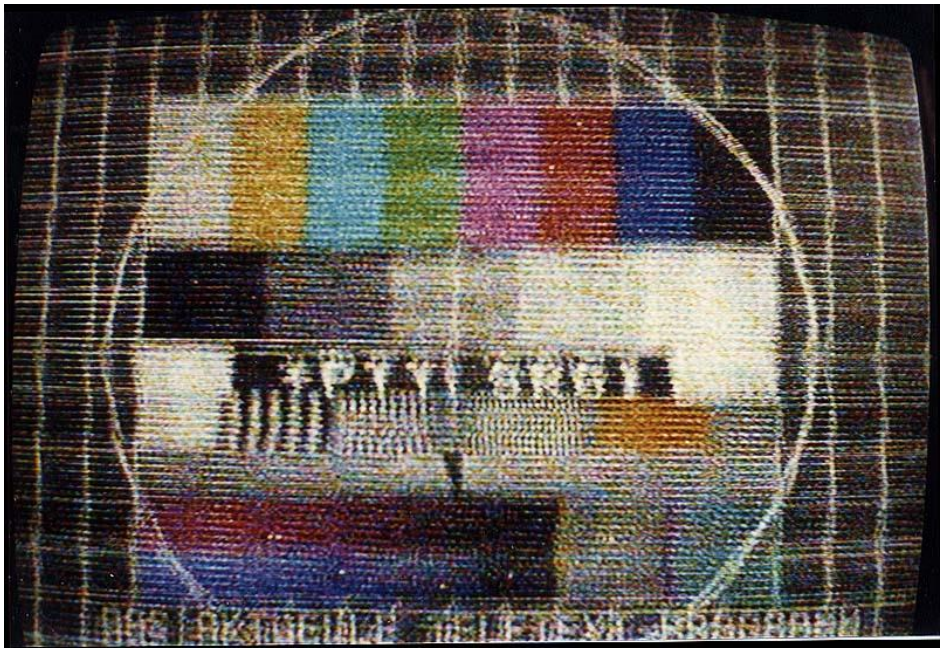


How a classifier „sees“ the world



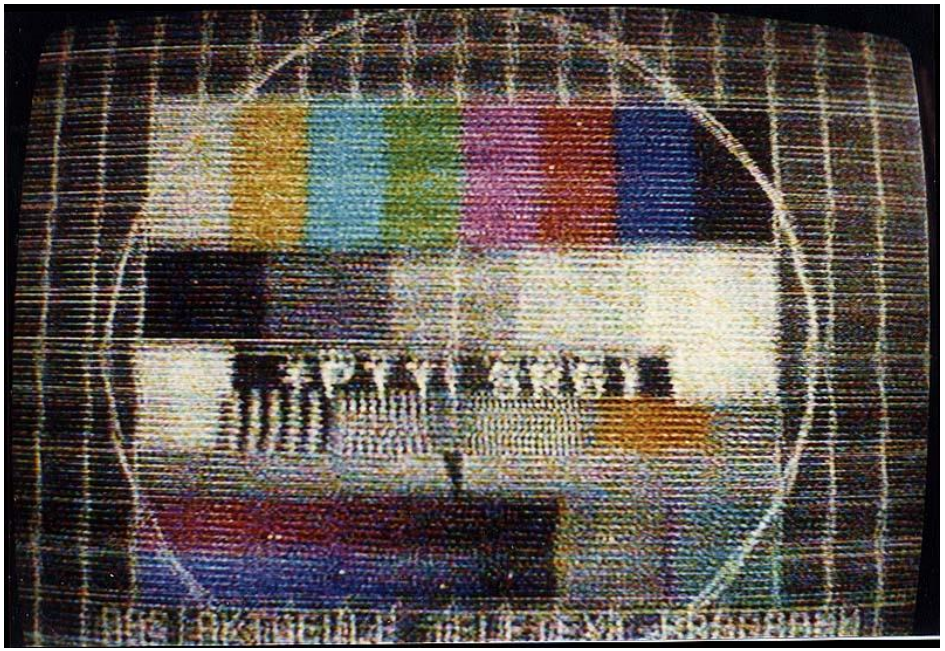
Most learning algorithms try to (linearly) separate the data instances.

Noise in Training Data



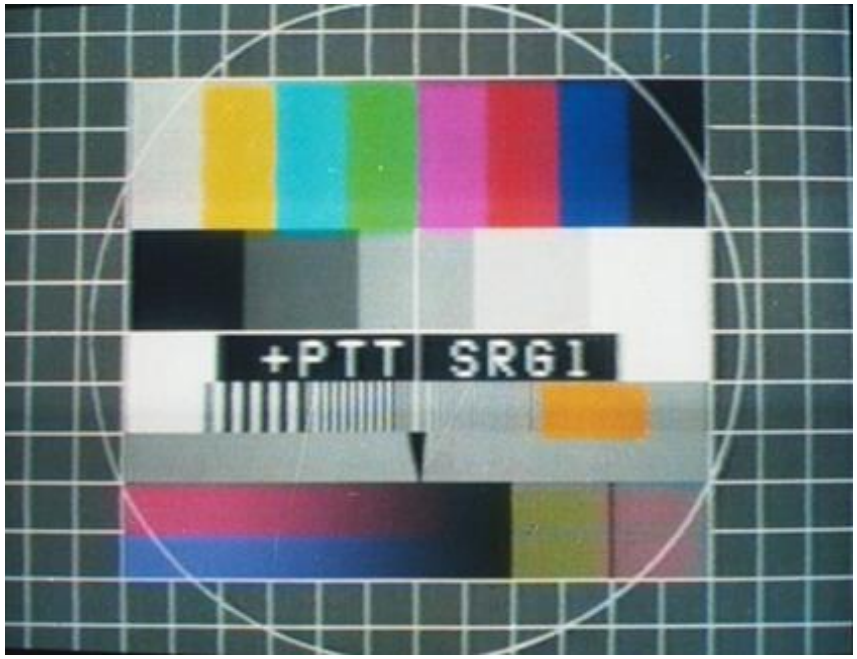
All training data in NLP contain noise.

Noise in Training Data



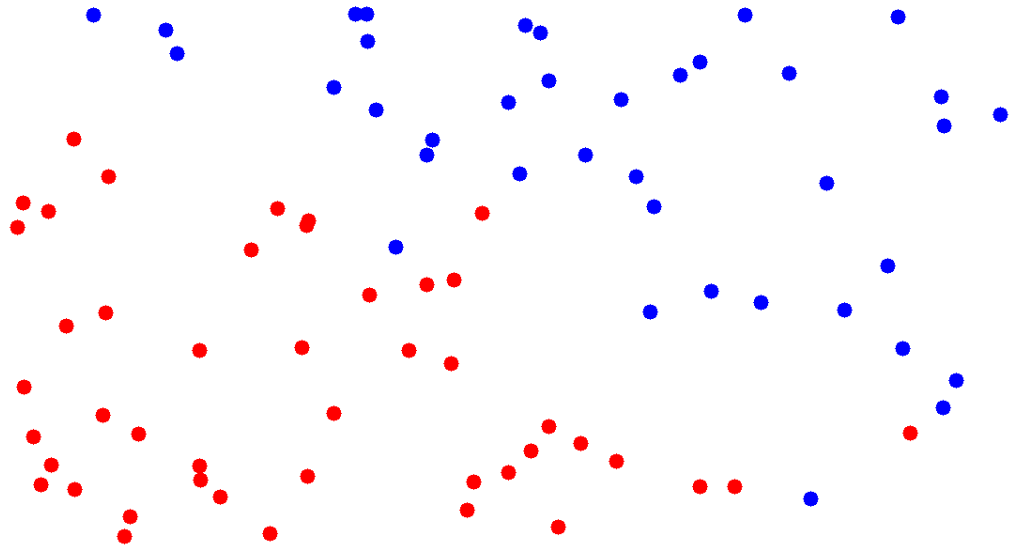
Classifier should learn the actual classes and not the noise!

Noise in Training Data

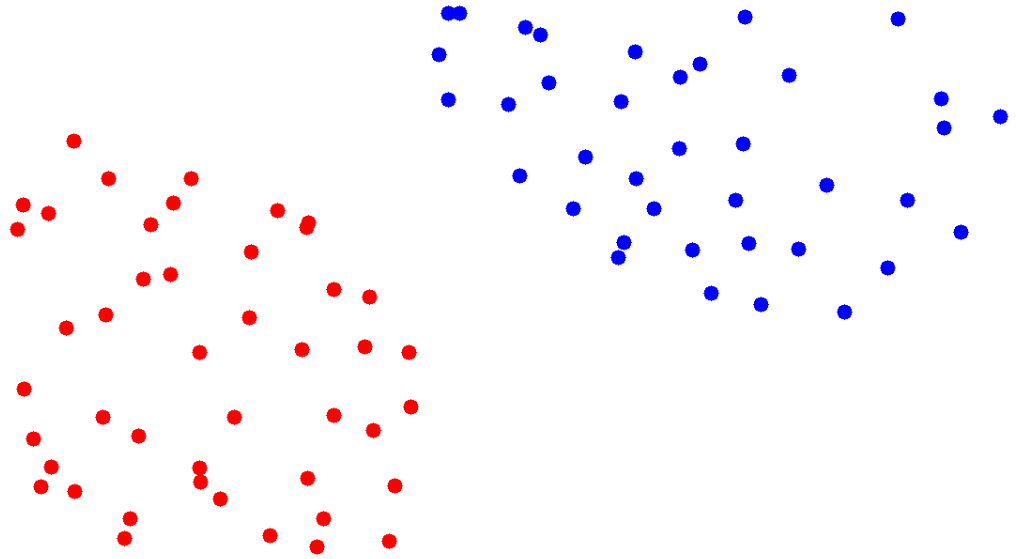


Noiseless data are not feasible.

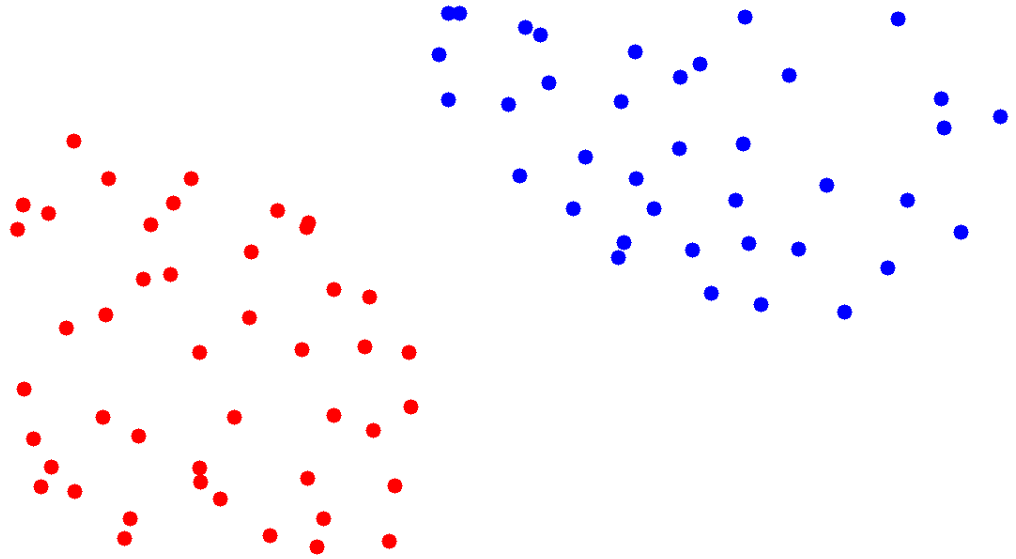
Impact of Feature Engineering



Impact of Feature Engineering



Impact of Feature Engineering



Good feature engineering may bring about a better separability of the data instances.

Document Vectors vs. Word Vectors

- For many traditional classifiers in text classification, a document is represented by a document vector:
  - Vector components represent words within document (e.g. word presence or word count).
- More recent classifiers (particularly deep learning algorithms) operate on word vectors:
  - A vector represents a word.
  - In order to represent a document: some operation on word vectors representing words in documents needs to be applied (e.g. averaging).

Document Vectors vs. Word Vectors

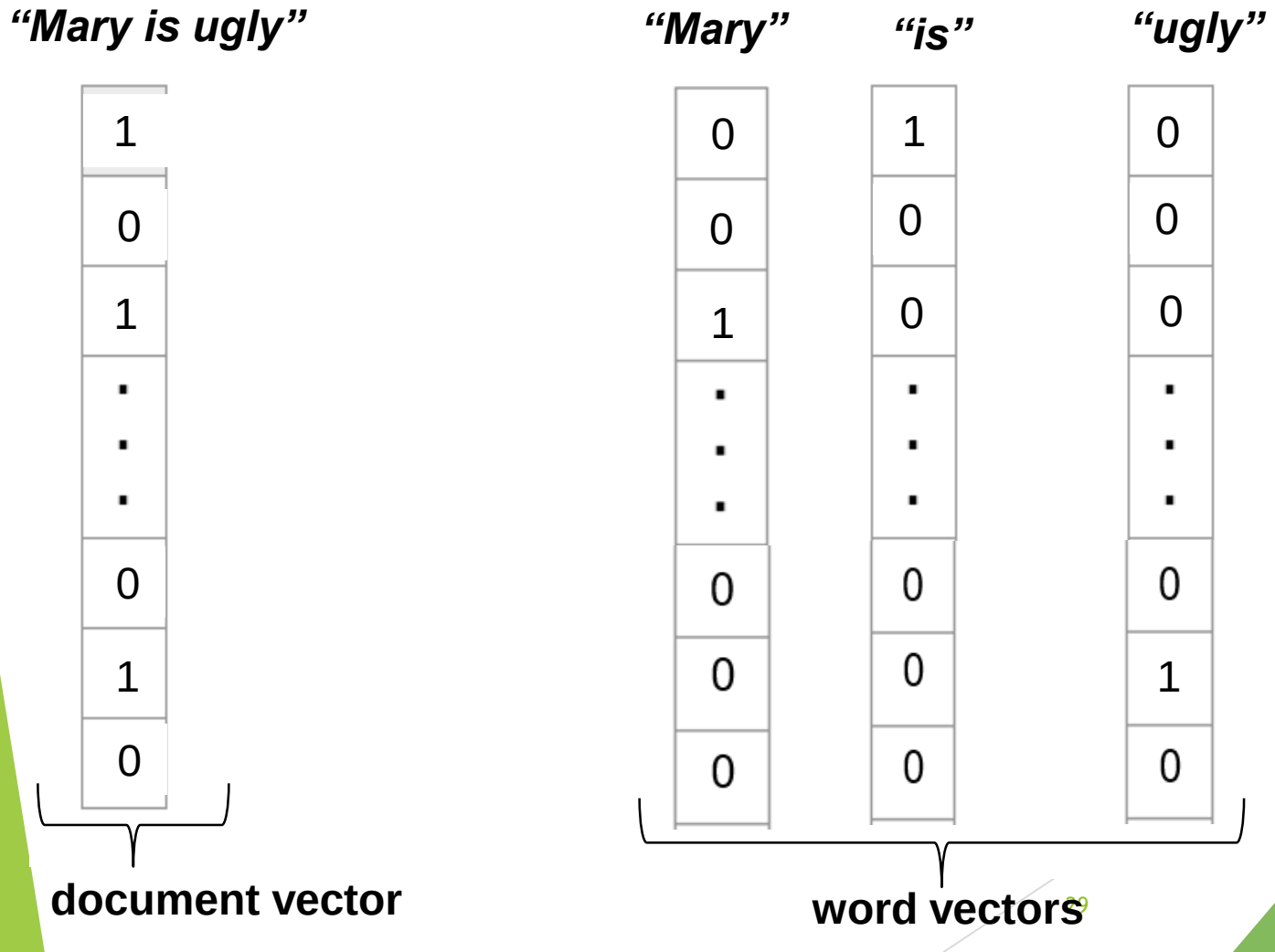
We take a very broad definition of *documents*. A tweet/review/sentence can also be considered a document.

- For many classificati
  document
  - Vector c
    docume
- More recent classifiers (particularly deep learning algorithms) operate on word vectors:
  - A vector represents a word.
  - In order to represent a document: some operation on word vectors representing words in documents needs to be applied (e.g. averaging).

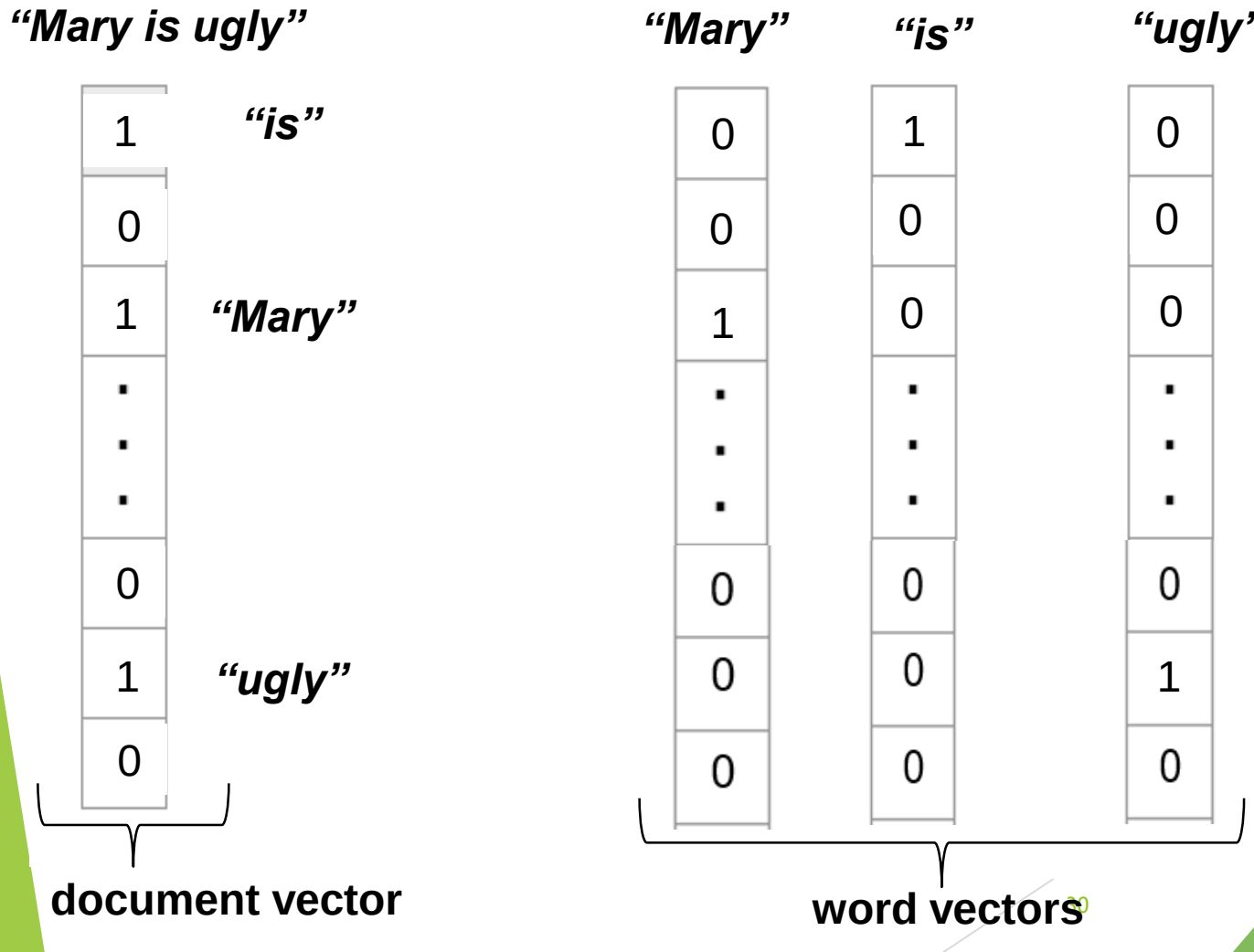
Document Vectors vs. Word Vectors

- For many traditional classifiers in text classification, a document is represented by a document vector:
  - Vector components represents words within document (e.g. word presence or word count).
- More recent classifiers (particularly deep learning algorithms) operate on word vectors:
  - A vector represents a word.
  - In order to represent a document: some operation on word vectors representing words in documents needs to be applied (e.g. averaging).

Document Vector and Word Vectors for *Mary is ugly*



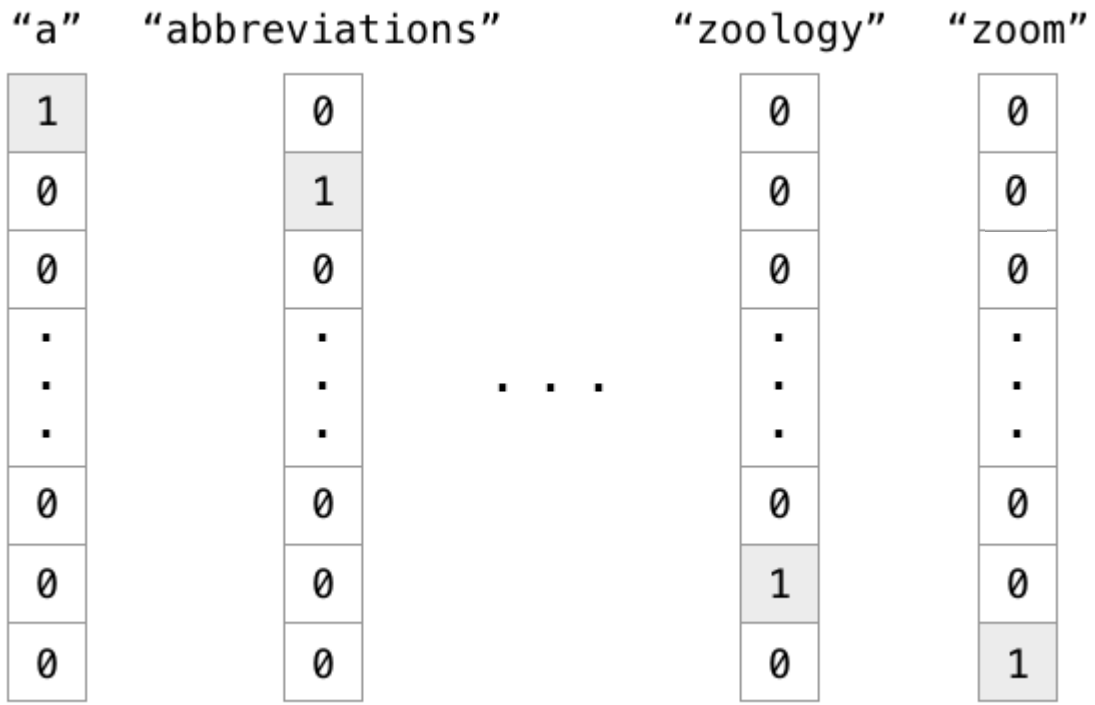
Document Vector and Word Vectors for *Mary is ugly*



Word Vector Representations

- A standard vector representation of words is a **one-hot** representation:
  - Binary vector.
  - Vector dimensionality represents an entire word vocabulary.
  - Each vector component represents one unique word.
  - For each word vector, only one component has value 1, all other components are 0.
- Such word representation can be very effective.

Illustration of One-Hot Representation



Shortcomings of One-Hot Representation

- Produces high-dimensional vectors.
- For small training sets, such vectors may be too sparse.
- Produces too coarse-grained similarities:
  - cosine(vec(*apple*), vec(*apple*)) = 1
  - cosine(vec(*apple*), vec(*pear*)) = 0
  - cosine(vec(*apple*), vec(*dog*)) = 0
- No means to generalize beyond the words observed in the training data.

Shortcomings of One-Hot Representation

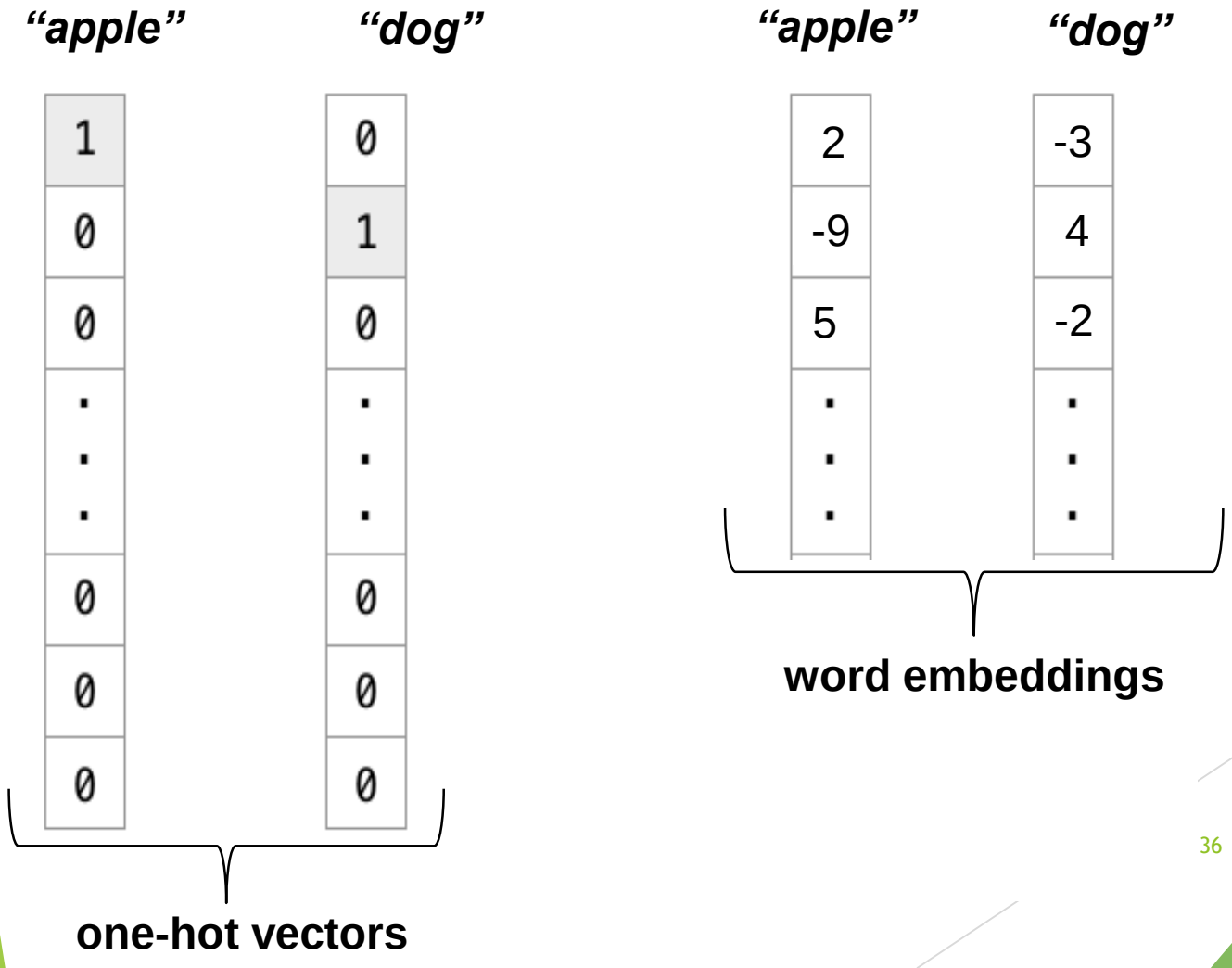
- Produces high-dimensional vectors.
- For small training sets, such vectors may be too sparse.
- Produces too coars
  - cosine(vec(*app*
  - cosine(vec(*apple*), vec
  - cosine(vec(*apple*) *(dog*)) = 0
- No means to generalize beyond the words observed in the training data.

Imagine *apple* was in the training data and *pear* was only in the test data!

Word Embeddings

- Induced from large unlabeled corpora (Mikolov et al., 2013).
- Word vector represents contexts with which word has been observed in corpus.
- Dense vectors (*typically 100-500 dimensions*).
- Vectors are non-binary, more than one component can be non-zero.
- Produce more linguistically adequate similarities:
  - cosine(vec(*apple*), vec(*apple*)) = 1.00
  - **cosine(vec(*apple*), vec(*pear*)) = 0.89**
  - cosine(vec(*apple*), vec(*dog*)) = 0.14

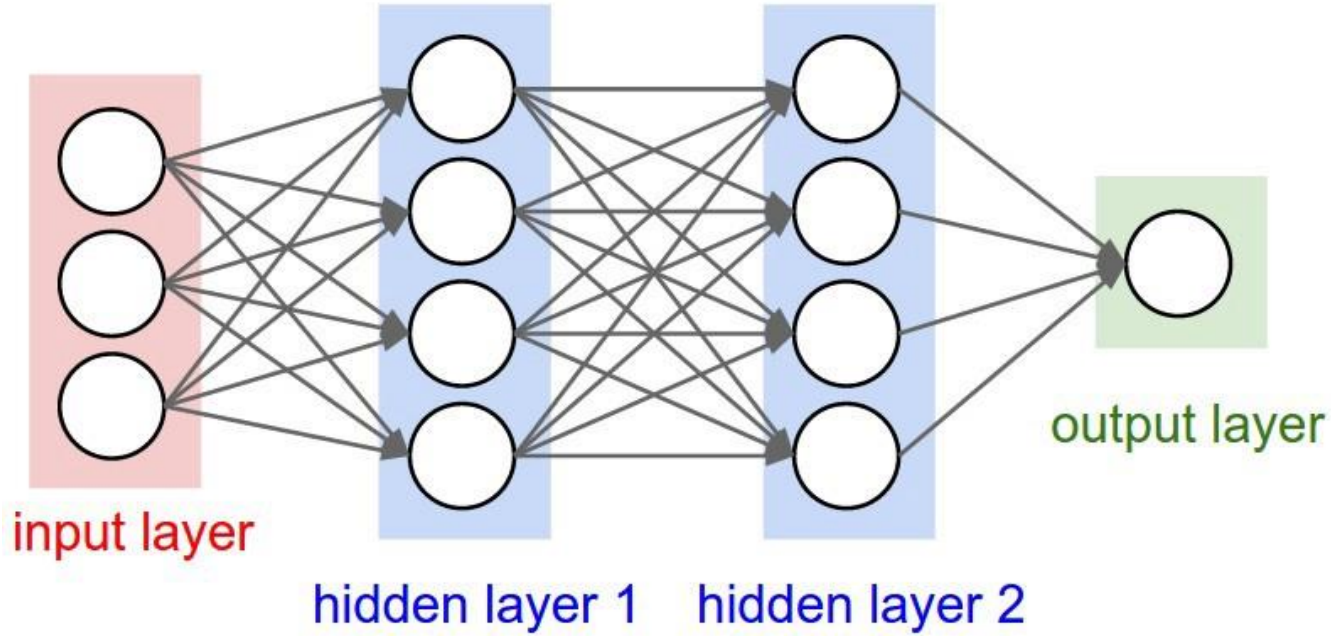
One-Hot vs. Word Embeddings



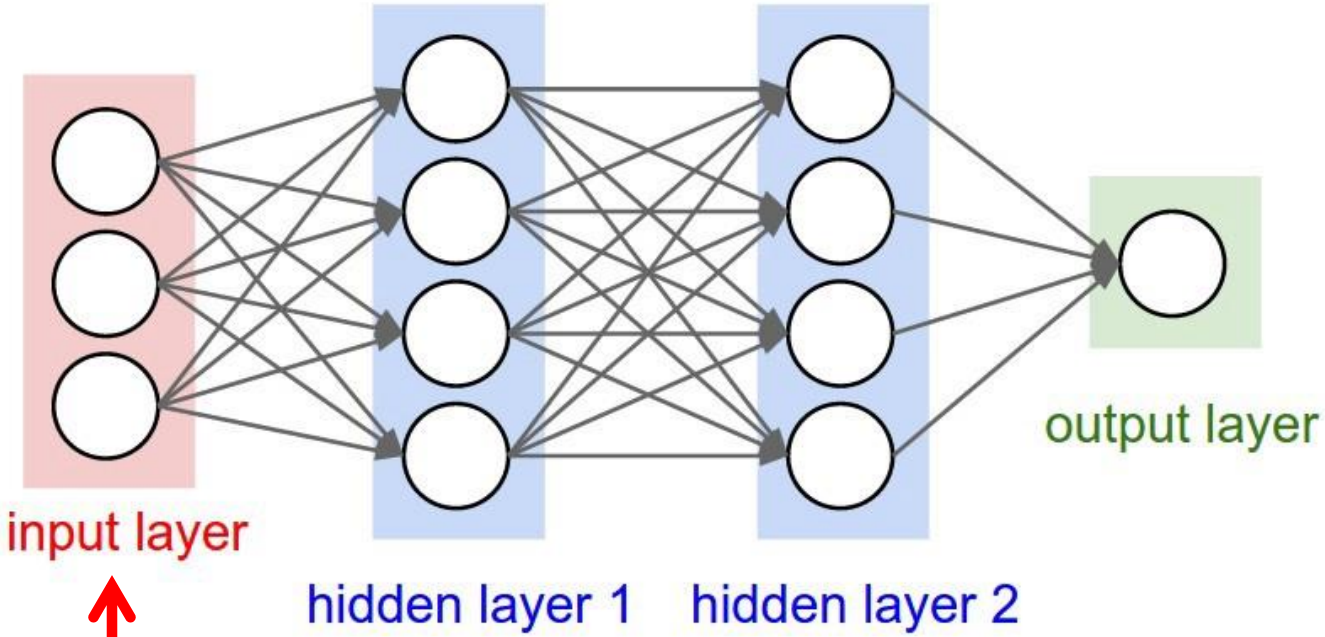
Document Vectors vs. Word Vectors

- Much of very recent research just employs as features word vectors encoding word embeddings → deep learning.
- With respect to word vectors/word embeddings, we cannot really encode much further explicit linguistic knowledge.
- Document vectors, on the contrary, allow us to incorporate much more linguistic knwoledge.
- The focus of this course is on linguistic modeling, so we will consider feature engineering using document vectors.

Deep Learning Illustrated

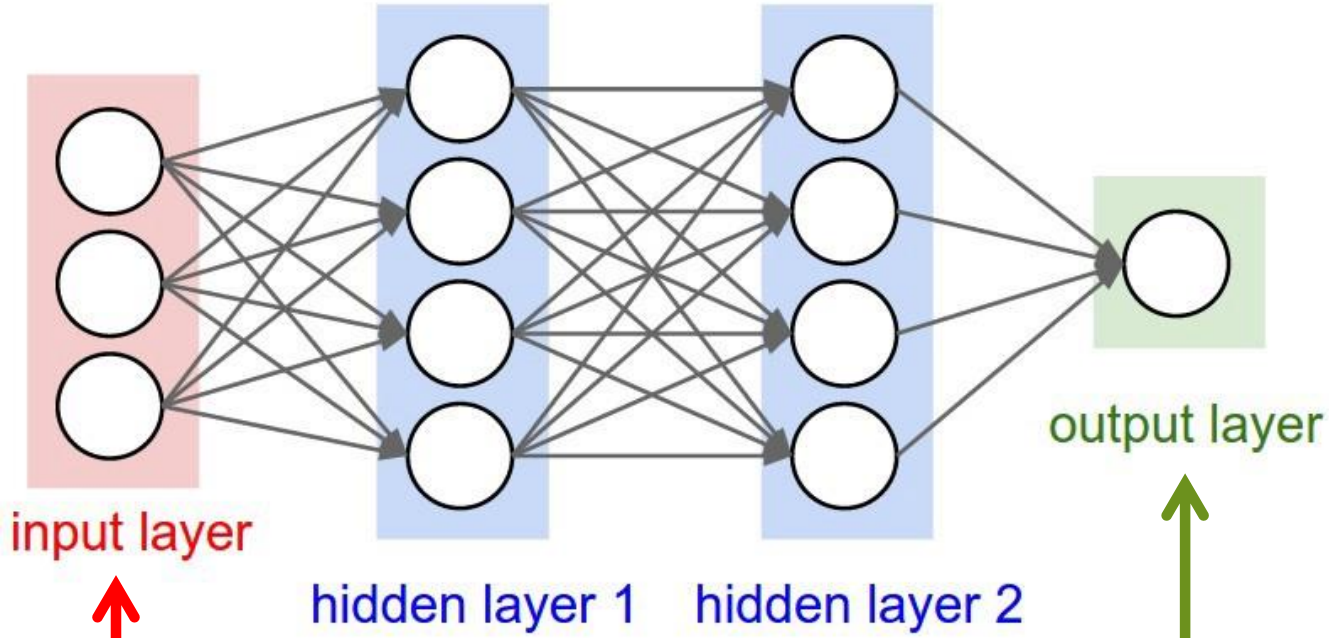


Deep Learning Illustrated

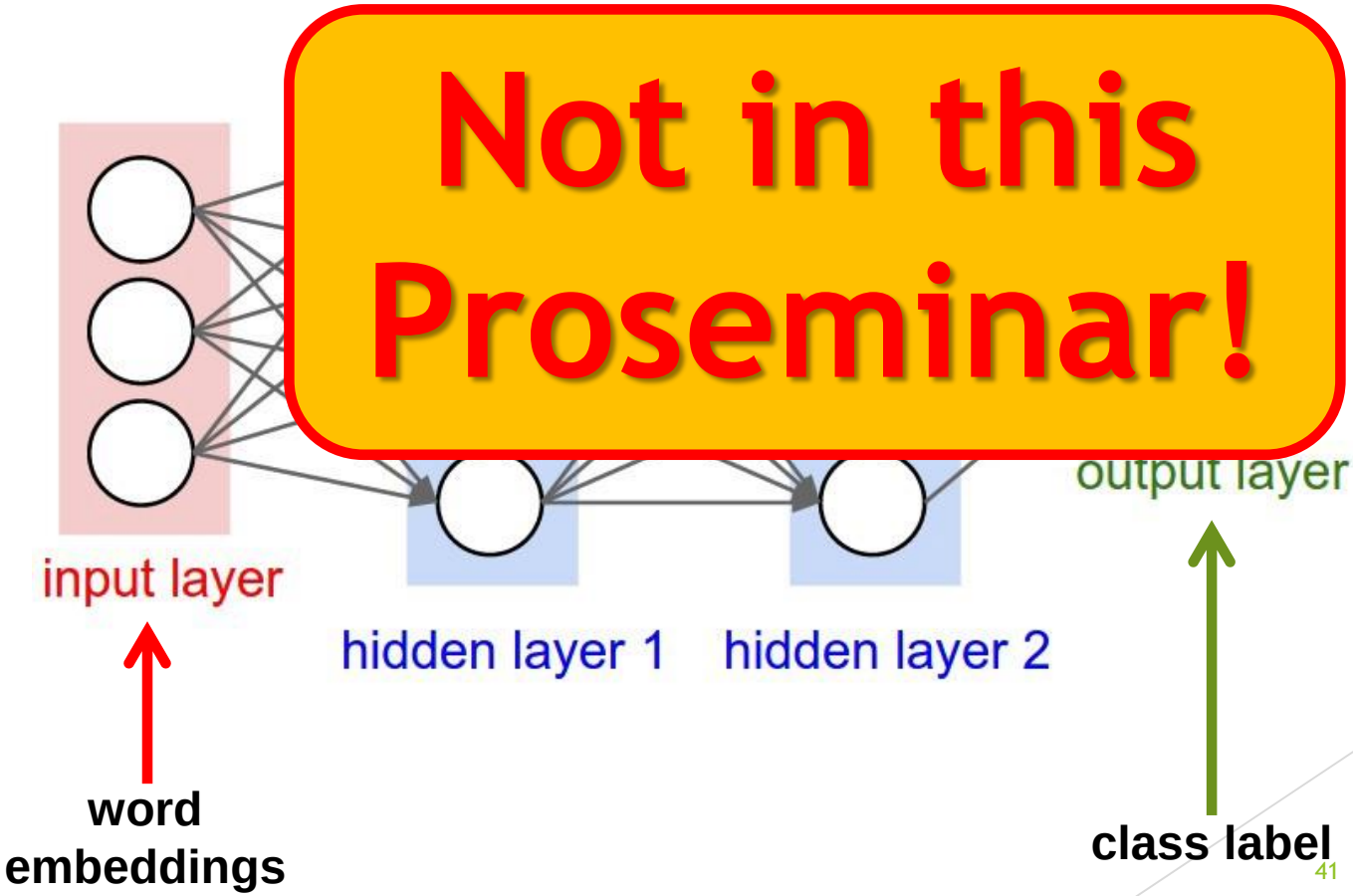


word embeddings

Deep Learning Illustrated



Deep Learning Illustrated



Linguistic Feature Engineering

- Not just encoding which words have been observed!
- Further features:
  - What types of POS do we observe in a sentence?
    - Count adjectives → subjective language.
  - Being within the scope of a negation is an important feature for classifying the polarity:
    - [**No** student likes our new instructor].
  - What is the relation syntactic relation of between words.
    - $[\text{Peter}]_{\text{subj}}$ loves Mary. → Being the subject is a predictive cue for opinion holders.

Machine Learning – Some Additional Remarks

- The previous illustration depicted a case where all training data are labeled → ***supervised learning***.
- There are also scenarios in which only some parts of the training data are labeled → ***semi-supervised learning***.
- Another possible setting is where all training data are unlabeled → ***unsupervised learning***.

Examples of Supervised Classifiers

- Naive Bayes
- Decision Trees
- Maximum Entropy Classifier
- Support Vector Machines
- Logistic Regression
- Conditional Random Fields
- Neural Networks (~Deep Learning)

Relevance of Machine Learning in this Seminar

- Many approaches to solve some specific task are based on learning algorithms.
- The actual algorithms **are** not the focus of this seminar.
- Our focus is on:
  - The actual task setting → how can the problem be **formalized**?
  - The information needed to solve the task → **feature design**.

Outline

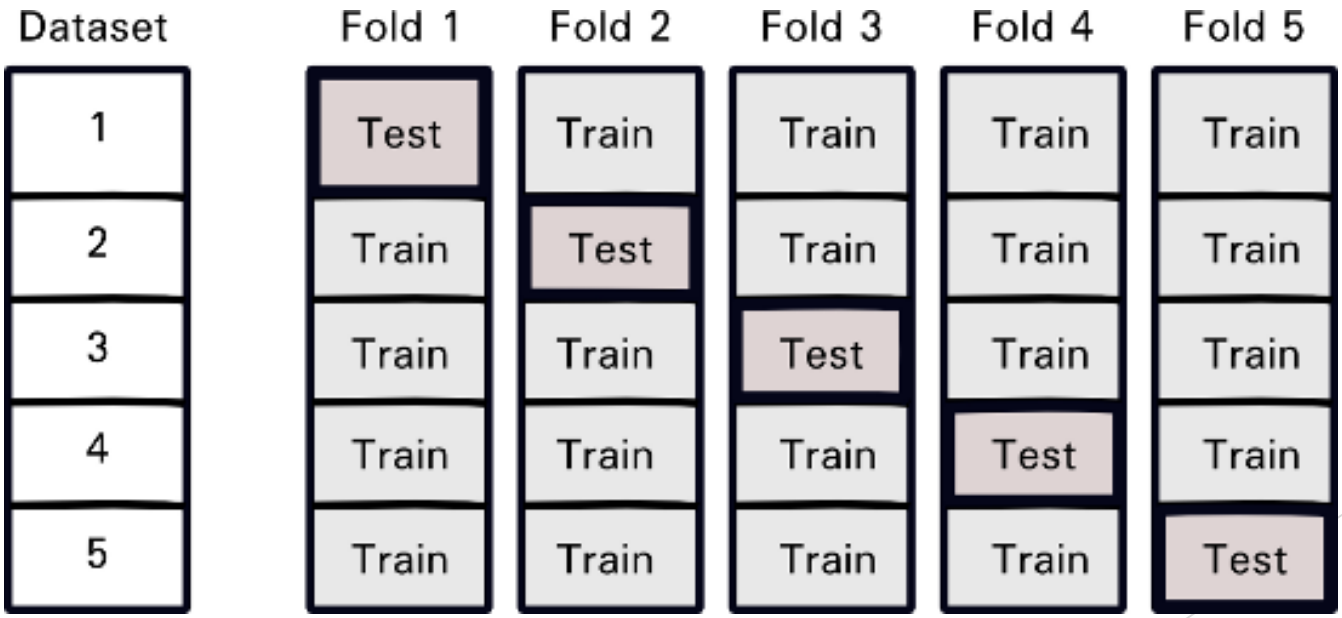
- Machine Learning
- Evaluation

The Most Common Setting of Evaluation

Dataset consists of:

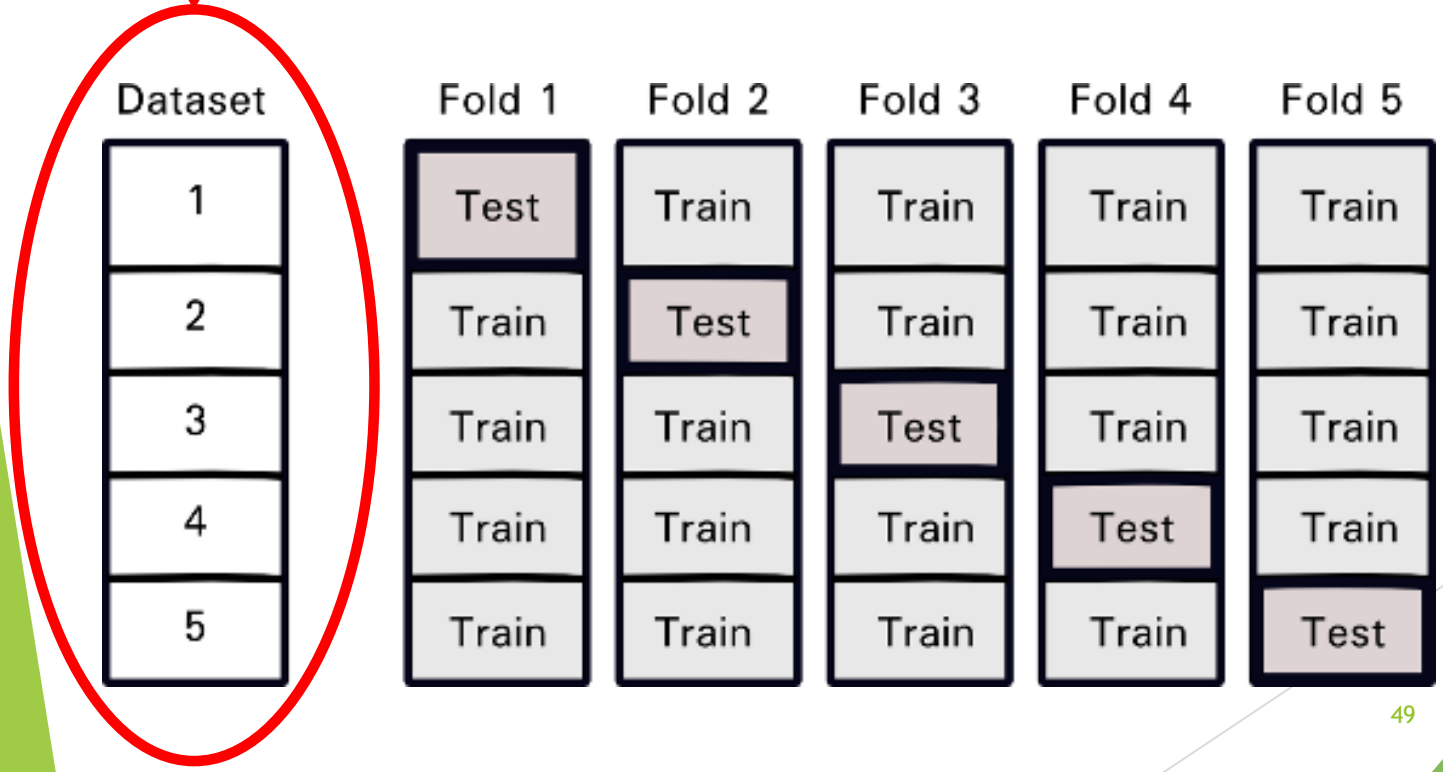
- training data
- test data
- development data for feature exploration, parameter tuning *(not always used!)*

N-fold Crossvalidation



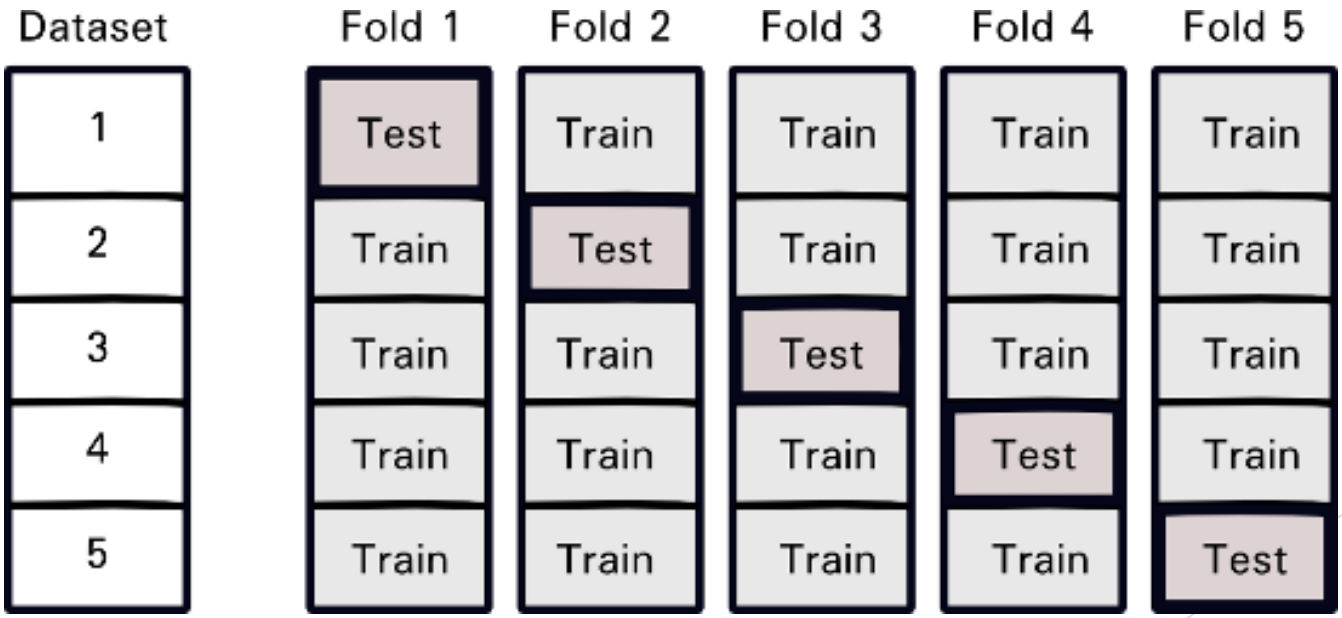
N-fold Crossvalidation

An alternative setting to a fixed training and test set.
There is only one labeled data set.



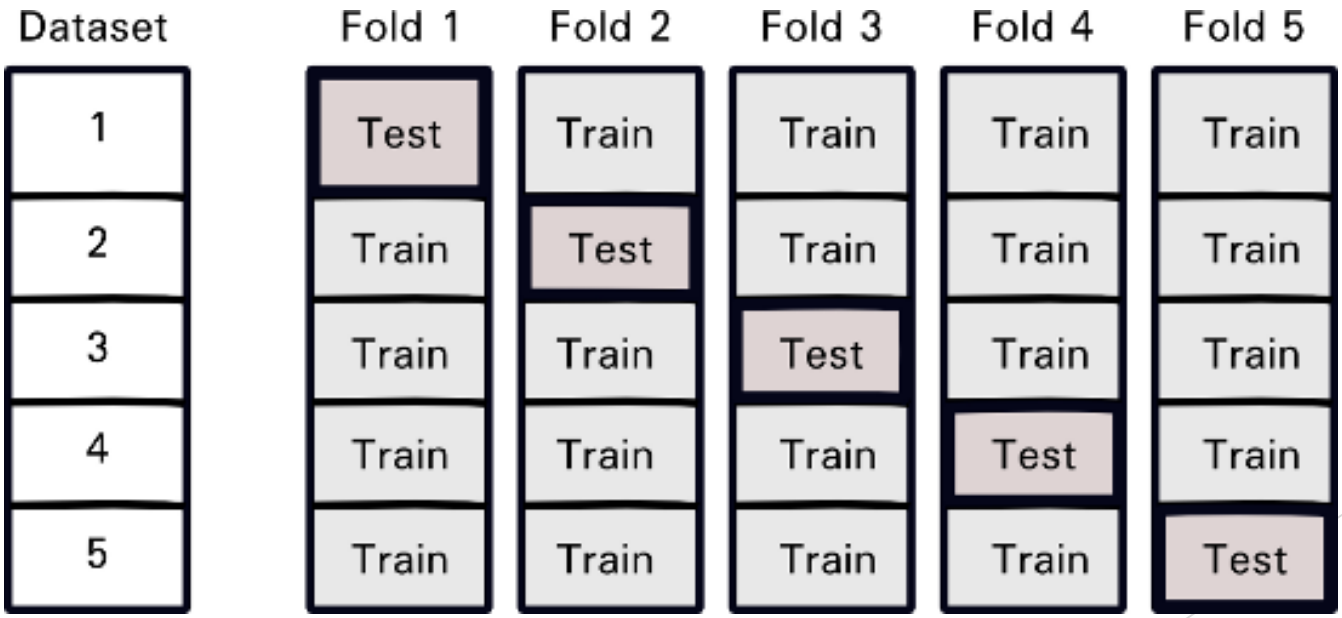
N-fold Crossvalidation

- Dataset is divided into *n* folds.
- We have *n* different experiments; each time a different fold is the test fold; the remaining folds are training data.

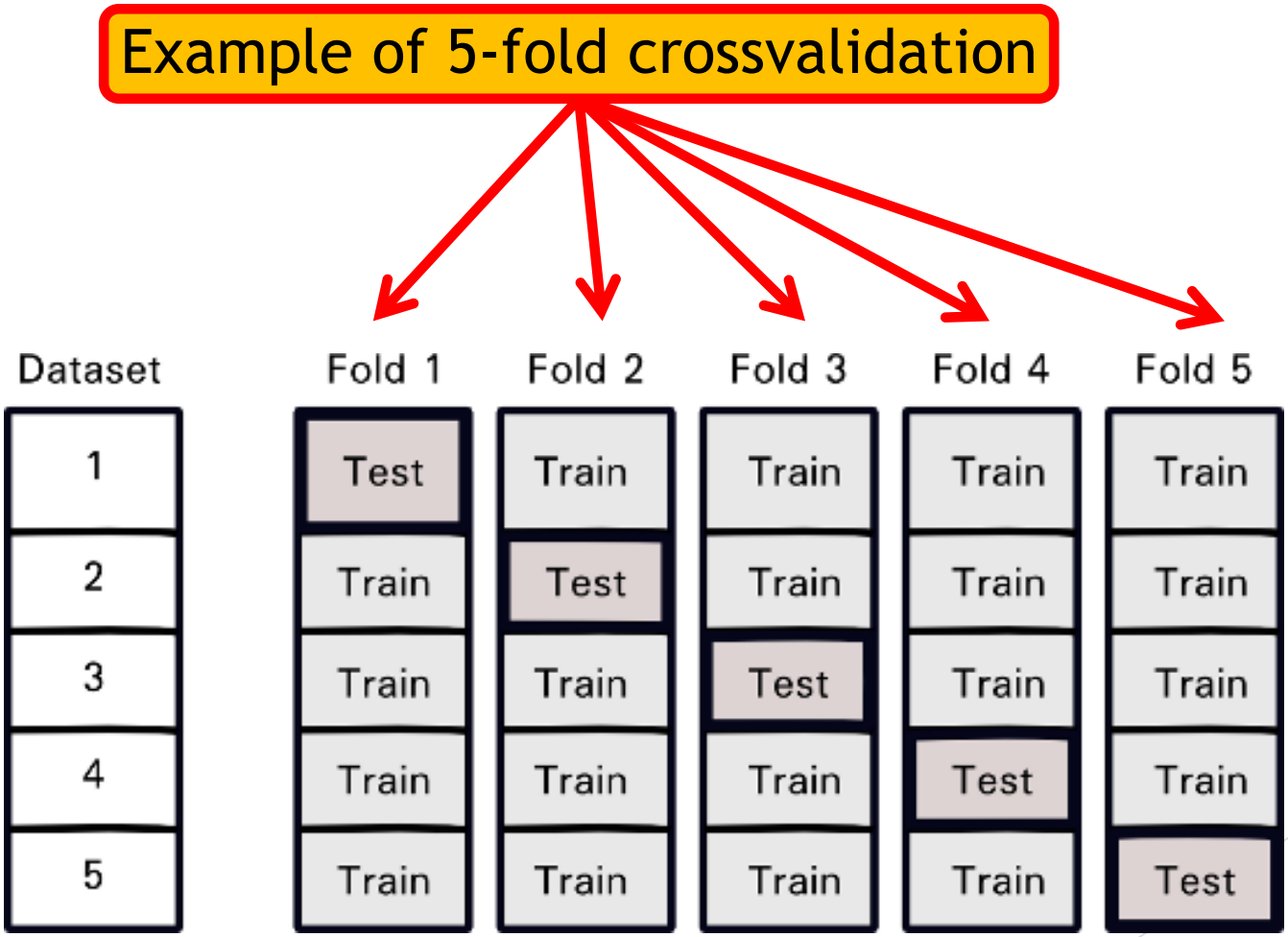


N-fold Crossvalidation

- In each experiment we carry out supervised learning/classification.
- As a final result, we average the scores obtained from the different folds.



N-fold Crossvalidation



Evaluation Measure: Accuracy

- Accuracy = $\frac{number\ of\ correct\ classifications}{number\ of\ all\ instances}$
- Just **one** score telling how many instances are correctly classified.

Evaluation Measures – Precision, Recall, F-score

- For these measures, we focus on **one** class!
- True positives (TP): prediction and actual label are positive
- False positives (FP): prediction label is positive but actual label negative
- False negatives (FN): prediction label is negative but actual label positive

Evaluation Measures – Precision, Recall, F-score

- Precision = $\frac{TP}{FP+TP}$
- Recall = $\frac{TP}{FN+TP}$
- F-score = $\frac{2*Precision*Recall}{Precision+Recall}$

Evaluation Measures – Precision, Recall, F-score

- Precision: how good are the positive predictions that are made.
- Recall: how good is the general coverage.
- F-score: combined score for Precision and Recall

Accuracy vs. F-score

- Accuracy: one score summarizing all predictions for all classes.
- F-score: one score for the prediction of one class
- Can also average F-scores for the different classes.
- Accuracy may not be very telling in case of very imbalanced class distributions:

  Given a two-class problem with class A occurring 95% of the time, is an accuracy of 95 really good?

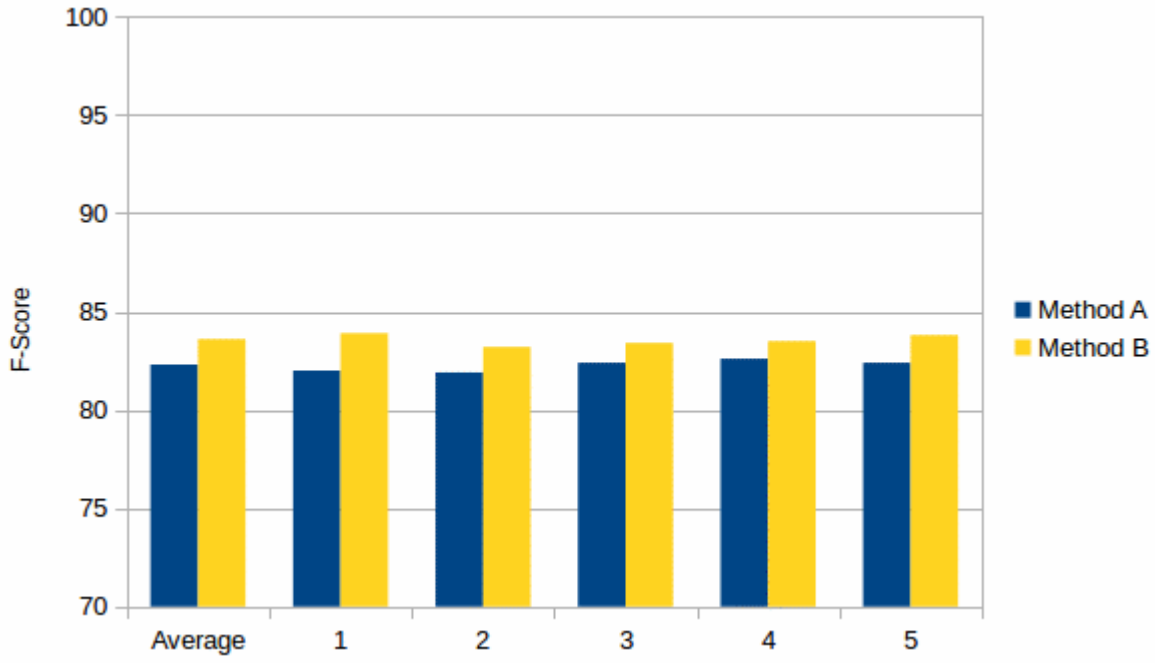
Evaluation – How to decide whether some score is good?

- Just producing one isolated score is not really meaningful:
  - *My method produces on my (new) gold standard an F-score of 0.81.*
- Need to compare against ***other*** methods (*baselines*):
  - other previously published methods for the same task
  - some trivial methods: majority-class classifier, randomly guessing

Statistical Significance

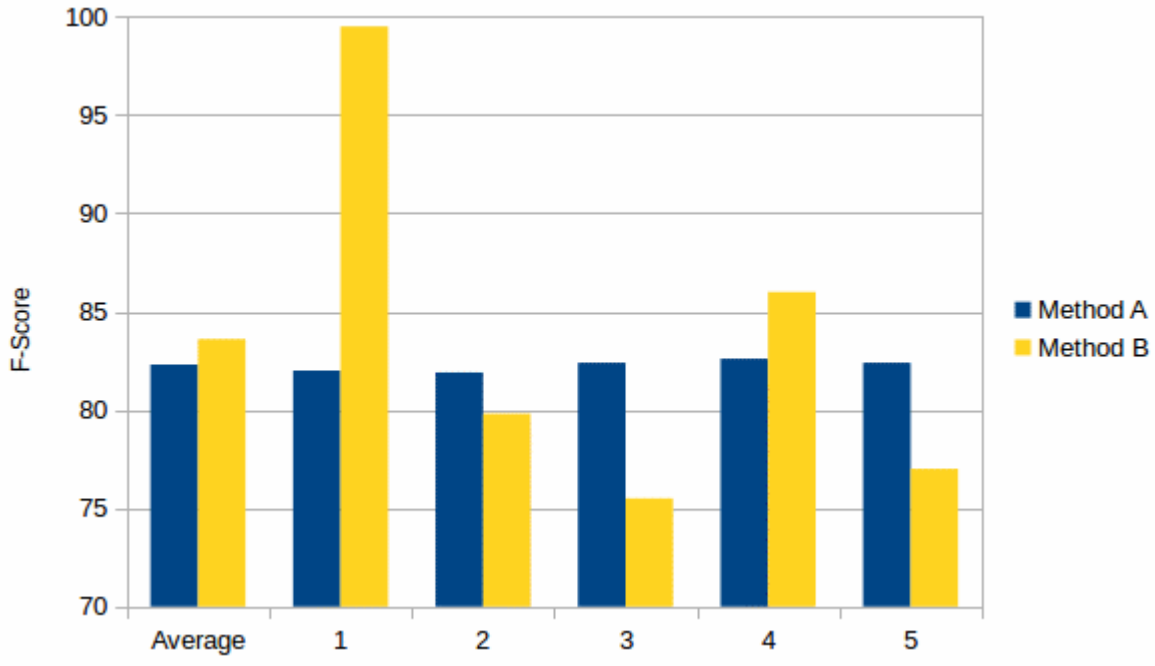
- In NLP, performance differences produced by different methods are often small.
- Example:
  - method A produces an F-score of 82.3
  - method B produces an F-score of 83.6
- One needs to establish whether the difference is *meaningful* or just happened by chance.

Statistical Significance



- Imagine these results being the individual results from a 5-fold crossvalidation
- The differences between method B and method A are very systematic → the improvement suggested by average scores are likely to be statistically significant.

Statistical Significance



In this other case, there is a high fluctuation between the different results → the improvement suggested by average scores are unlikely to be statistically significant

Statistical Significance

- Solution: statistical significance testing
- Idea: estimate the probability *p* that the differences of scores have been caused by chance.
- We typically regard results as statistically significant, if $p < 0.05$
- *Remember*: if you encounter the word *significant* in a paper, it typically means *statistically significant*.

Evaluation of a Gold Standard

- In order to be able to do a quantitative evaluation, a dataset with ***manual*** annotation has to be created *(commonly referred to as gold standard)*.
- Need some form of proof that those *human labels* are meaningful.
- This is typically achieved by measuring *interannotation agreement*.

Interannotation Agreement (IAA)

- At least some subset of the gold standard needs to be annotated by *2 different annotators*.
- Interannotation agreement: *checks in how far two (or more) manual annotations of the same data agree*.
- Only if IAA is sufficiently high, the gold standard is useful.

Cohen's Kappa κ

- A common measure for IAA (with two annotators).
- P(A) = proportion of times judges agree
- P(E) = what agreement would we get by chance
- K = $\frac{P(A)-P(E)}{1-P(E)}$
- What κ values are acceptable?
  - poor agreement = Less than 0.20
  - fair agreement = 0.20 to 0.40
  - moderate agreement = 0.40 to 0.60
  - good agreement = 0.60 to 0.80
  - very good agreement = 0.80 to 1.00
- There are different interpretations!

Further Reading

- Christopher Manning and Hinrich Schütze: **Foundations of Statistical Natural Language Processing**, *MIT Press*. 1999.
- Christopher Manning, Prabhakar Raghavan and Hinrich Schütze: **Introduction to Information Retrieval**, *Cambridge University Press*. 2008.
- Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean: **Distributed Representations of Words and Phrases and their Compositionality**, *NIPS*, 2013.