Applications of Artificial Intelligence

– Winter Term 21/22 –

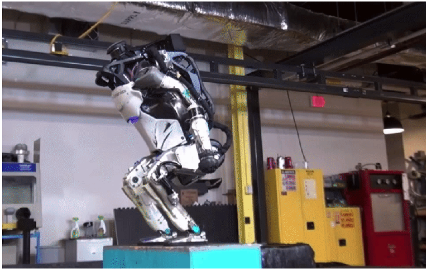
Chapter 05

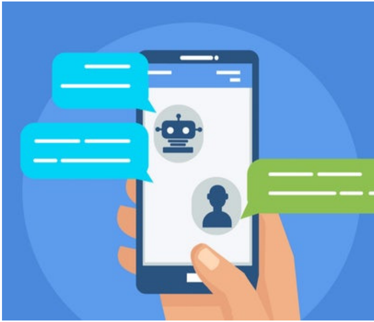
Machine Learning: Basics

Prof. Dr. Adrian Ulges

RheinMain University of Applied Sciences

Machine Learning: Recent Successes images: [6] [7] [1] [5] [8]





Outline

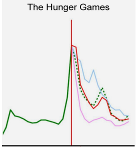
Machine Learning: Definition

> *"Machine learning is a scientific discipline that explores the construction and study of* **algorithms that can learn from data**. *Such algorithms operate by building a* **model** *from* **example inputs** *and using that to make* **predictions or decisions**, *rather than following strictly static program instructions."*
>
> (en.wikipedia.org)

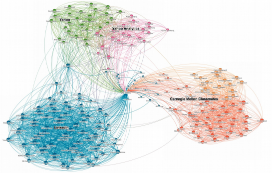
> *"A computer program is said to* **learn** *from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with Experience E."*
>
> (Tom Mitchell (1998))

Machine Learning: Tasks

Regression



Clustering

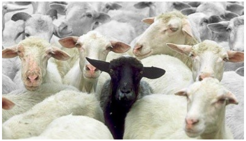


Recommendation

Classification

Data Reduction

Anomaly Detection



Supervised vs. unsupervised Learning

Supervised Learning

- Our goal is a **prediction** about an object.
- We call this prediction the **label** or **target**.
- In NLP, objects are *text passages, words, questions, …* , and labels are *question types, relevance, sentiment, …*
- We learn from samples $\mathbf{x}_1, ..., \mathbf{x}_n$ and labels $y_1, ..., y_n$.

Unsupervised Learning

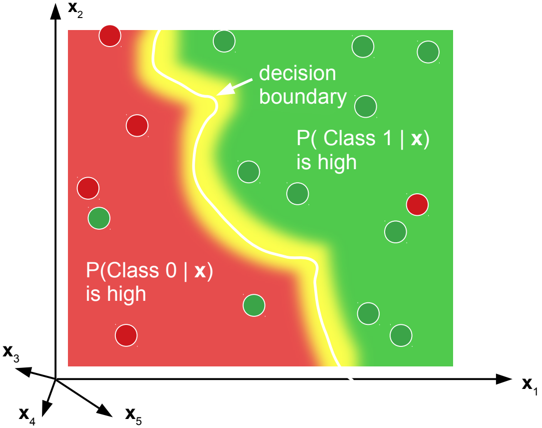
- unsupervised learning = **no labels**, only features $\mathbf{x}_1, ..., \mathbf{x}_n$
- learning the data's **structure**: subgroups, patterns, outliers …
- **important in NLP! Learning a language's structure from text corpora**.

Classification: Features+Labels image: [2]

- Machine learning makes predictions about real-world objects.
- We describe an object by a **feature vector** $\mathbf{x}$ *(bold=vectors!)*.
- Our goal is to predict a **label** y for the object.

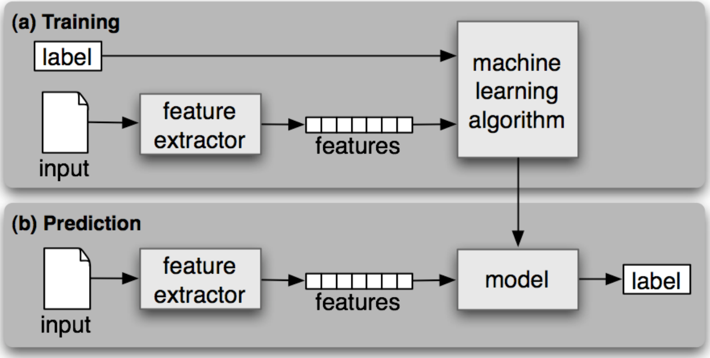
	A	B	C	D	E	F	G	H	I	J	K	L
1	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
2	1	0	3	Braund, Mr. Owen Harris	male	22	1	0	A/5 21171	7.25		S
3	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Thayer)	female	38	1	0	PC 17599	71.2833	C85	C
4	3	1	3	Heikkinen, Miss. Laina	female	26	0	0	STON/O2. 3101282	7925		S
5	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35	1	0	113803	53.1	C123	S
6	5	0	3	Allen, Mr. William Henry	male	35	0	0	373450	8.05		S
7	6	0	3	Moran, Mr. James	male		0	0	330877	8.4583		Q
8	7	0	1	McCarthy, Mr. Timothy J	male	54	0	0	17463	51.8625	E46	S
9	8	0	3	Palsson, Master. Gosta Leonard	male	2	3	1	349909	21075		S
10	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27	0	2	347742	11.1333		S
11	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)								
12	11	1	3	Sandstrom, Miss. Marguerite Rut								
13	12	1	1	Bonnell, Miss. Elizabeth								
14	13	0	3	Saundercock, Mr. William Henry								
15	14	0	3	Andersson, Mr. Anders Johan								
16	15	0	3	Vestrom, Miss. Hulda Amanda Adolfina								
17	16	1	2	Hewlett, Mrs. (Mary D Kingcome)								
18	17	0	3	Rice, Master. Eugene								
19	18	1	2	Williams, Mr. Charles Eugene								
20	19	0	3	Vander Planke, Mrs. Julius (Emelia Maria Van								
21	20	1	3	Masselmani, Mrs. Fatima								
22	21	0	2	Fynney, Mr. Joseph J								
23	22	1	2	Beesley, Mr. Lawrence								
24	23	1	3	McGowan, Miss. Anna "Annie"								
25	24	1	1	Sloper, Mr. William Thompson								
26	25	0	3	Palsson, Miss. Torborg Danira								
27	26	1	3	Asplund, Mrs. Carl Oscar (Selma Augusta Emi								
28	27	0	3	Emir, Mr. Farred Chehab								
29	28	0	1	Fortune, Mr. Charles Alexander								
30	29	1	3	O'Dwyer, Miss. Ellen "Nellie"								
31	30	0	3	Todoroff, Mr. Lalio								
32	31	0	1	Uruchurtu, Don. Manuel E								
33	32	1	1	Spencer, Mrs. William Augustus (Marie Eugen								
34	33	1	3	Glynn, Miss. Mary Agatha								
35	34	0	2	Wheadon, Mr. Edward H								
36	35	0	1	Meyer, Mr. Edgar Joseph								
37	36	0	1	Holverson, Mr. Alexander Oskar								
38	37	1	3	Mamee, Mr. Hanna								
39	38	0	3	Cann, Mr. Ernest Charles								
40	39	0	3	Vander Planke, Miss. Augusta Maria								
41	40	1	3	Nicola-Yarred, Miss. Jamila								
42	41	0	3	Ahlin, Mrs. Johan (Johanna Persdotter Larsso								
43	42	0	2	Turpin, Mrs. William John Robert (Dorothy Ann								
44	43	0	3	Kraeff, Mr. Theodor								

ML: Geometric View



- Feature vectors **x** are points in feature space.
- Labels correspond to points' colors.
- The ML model estimates the class probability $P(c|\mathbf{x})$.
- From this, we can derive decision boundaries between classes.

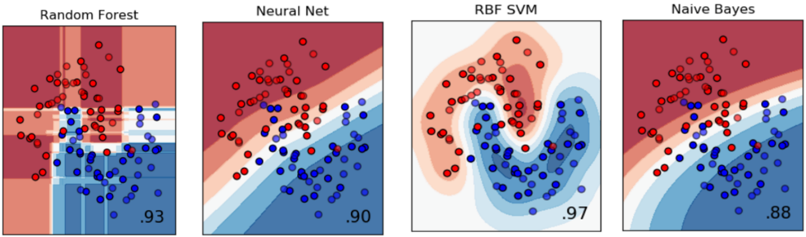
ML: System Pipeline image: [3]



Here: Batch Learning / Offline Learning

1. We train the system on training data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n$ with labels $y_1, y_2, \ldots, y_n$, obtaining a model ϕ.
2. Given a new object $\mathbf{x}$, the model predicts the label $\phi(\mathbf{x})$.
3. Training happens offline, the application of the model online.

ML: Sample Approaches



Different Approaches/Models

- **Random Forests**: Recursive Splitting of feature space.
- **Neural Networks**: Stacking of linear decision units.
- **SVMs**: Similarity comparison of objects.
- **Probabilistic Models (e.g., Naive Bayes)**: Use probabilities.

There is no **universally superior** approach
(no-free-lunch theorem).

Generalization in Feature Space

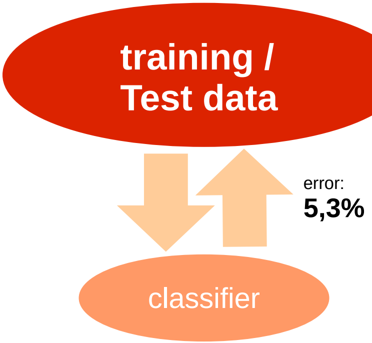
Generalization and Overfitting

> *"The real value of a scientific explanation lies not in its ability to explain (what one has already seen), but in predicting events that have yet to (be seen)."*
>
> (Blumer et al. 1987)

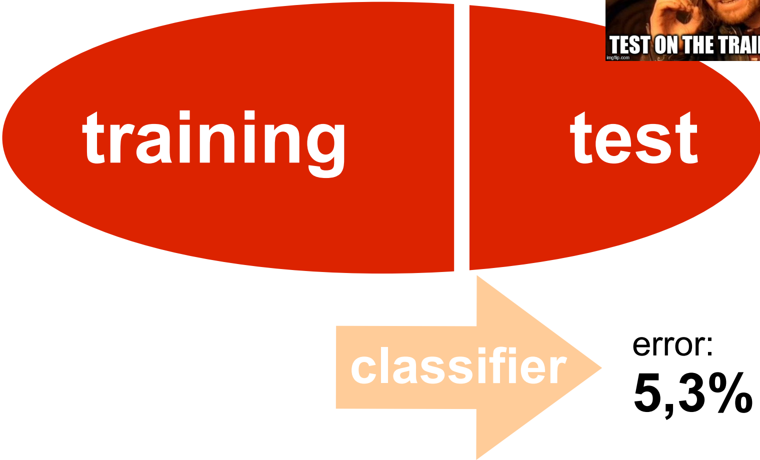
- **Generalization** is the model's capability wo work well on data **unseen in training**.
- **Overfitting** means that the model works well on the training data but poorly on new data.
- Overfitting tends to happen when...
  - ... the training set is very small
  - ... the model has too many parameters
  - ... the model does not fit the data well.

Evaluating ML Systems

- **machine learning cycle**: iteratively, work over ...
  - data
  - features
  - models
  - parameters
- **key driver**: **benchmarking**

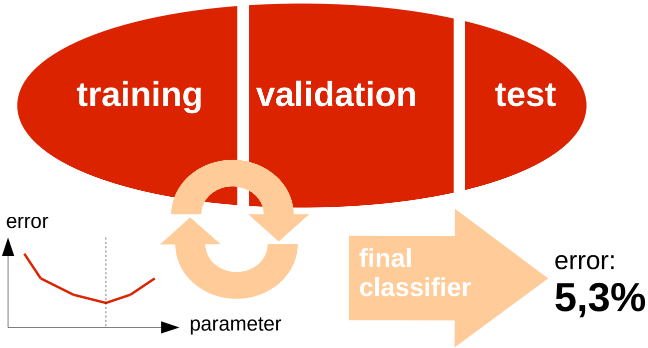


Evaluating ML Systems



- We **split** our dataset into **training and test data**.
- We **benchmark** (only) on the test data.
- **Never**: "test on the training data"!

Validation sets



- Some parameters of ML models are **trained**, others **(so-called, free)** parameters are set **manually**.
- **Example**: logistic regression → regularization parameter C.
- **Approach**: "trial and error" = **manual search** / **grid search**.
- $\Big($ train → validate → train → validate → ... $\Big)$ → test.

Outline

Logistic Regression: Approach

- **Logistic Regression** is a simple, widely used model for **classification** (!).
- Given: training samples $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ with labels $y_1, \ldots, y_n \in \{0, 1\}$ *(two classes)* .
- Goal: Learn a **classifier** function $\mathbb{R}^d \to \{0, 1\}$, which assigns **classes** to samples.

Approach

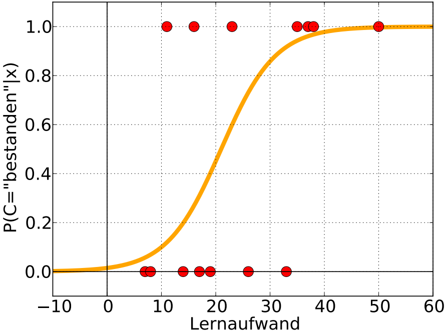
- Our model estimates a **probability** $P(C = 1|\mathbf{x})$.
- The classifier picks the class **with highest probability**.

Logistic Regression: Approach

We estimate the probability with the so-called **Sigmoid function**:

Example: Passing the math exam

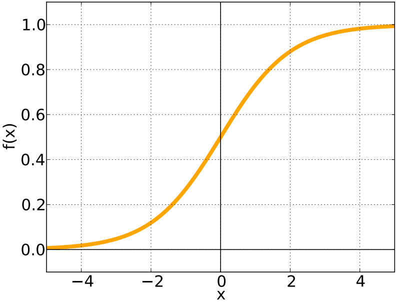
- x = learning effort, C = passed (1) / not passed (0)
- **Given**: training set $x_1, ..., x_n \in \mathbb{R}$ with labels $y_1, ..., y_n \in \{0, 1\}$.
- **Goal**: estimate $P(C = 1|x)$



Logistic Regression: Base Modell

As a regression function, we use the **sigmoid**:

$$P(C = 1|x) := f(x) = \frac{1}{1 + e^{-x}}$$



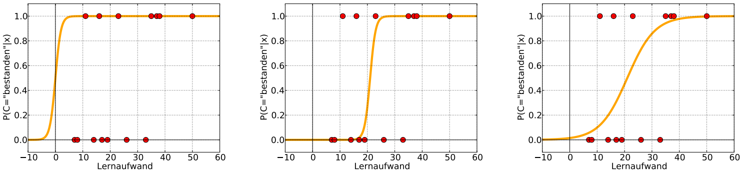
- $lim_{x \to -\infty} f(x) = 0$ and $lim_{x \to \infty} f(x) = 1$.
- $P(C = 1|x = 0) = f(0) = 0.5$ *(i.e., we pick class* 1 *if* $x \geq 0$*).*

Logistic Regression: Base Model

- We allow the function **shift** and **squeeze/stretch/mirror**:

$$f(x; w_0, w) = \frac{1}{1 + e^{-(w_0 + w \cdot x)}}$$

- The parameters w_0, w are estimated in **training** *(soon)*.



Multi-variate Logistic Regression

So far, we have only **one input feature** *(learning effort)*.

- How do we use **multiple features** $\mathbf{x} \in \mathbb{R}^d$?
- We extend the sigmoid function:

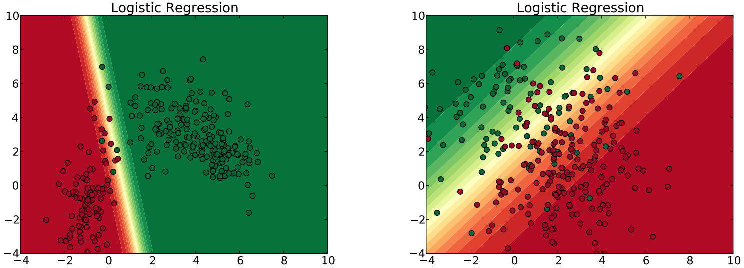
$$f(\mathbf{x}; w_0, w_1, w_2 \ldots, w_d) = \frac{1}{1 + e^{-(w_0 + w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots + w_d \cdot x_d)}}$$

or short (with vector $\boldsymbol{w} := (w_1, \ldots, w_d)$):

$$f(\mathbf{x}; w_0, \boldsymbol{w}) = \frac{1}{1 + e^{-(w_0 + \mathbf{x} \cdot \boldsymbol{w})}}$$

- This model's decision boundary is $\boldsymbol{x} \cdot \boldsymbol{w} + w_0 = 0$.
  This is a **hyperplane** *(in normal form)*!

Logistic Regression: Illustration



- The decision boundary is **linear**. This is why we call logistic regression a **linear classifier**.
- *More complex ("non-linear") models will follow later*.
- The parameter $\boldsymbol{w}$ determines the decision boundaries' orientation, w_0 shifts the boundary.
- $\boldsymbol{w}$ also determines the **smoothness** of the decision function.

Logistic Regression: Training

Key Question: Training

- **Given**: a training set $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$ with labels $y_1, \dots, y_n \in \{0, 1\}$.
- **Goal**: estimate w_0 and $\mathbf{w}$.

Approach: Maximum-Likelihood Estimation

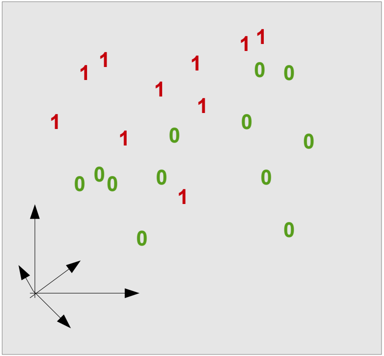
- **Idea**: choose the parameters such that the observed training set becomes "maximally likely".
- For **positive** samples ($y_i = 1$), $f(\mathbf{x}_i)$ should be **high**:

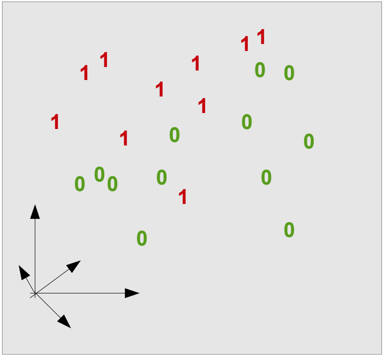
$$P(C = 1|\mathbf{x}_i) \approx f(\mathbf{x}_i) \approx 1$$

- For **negative** samples ($y_i = 0$), $f(\mathbf{x}_i)$ should be **low**:

$$P(C = 1|\mathbf{x}_i) \approx f(\mathbf{x}_i) \approx 0$$

Logistic Regression: Example





Logistic Regression: Optimization

ML Estimation

We define a likelihood function and maximize it:

$$w_0^*, \mathbf{w}^* = \arg\max_{w_0,\mathbf{w}} \underbrace{\prod_{i:y_i=1} f(\mathbf{x}_i) \cdot \prod_{i:y_i=0} (1 - f(\mathbf{x}_i))}_{\text{"likelihood function"}\, L(w_0,\mathbf{w})}$$

We rewrite the formula *(sums are better than products)*:

$$\begin{aligned}
w_0^*, \mathbf{w}^* &= \arg\max_{w_0,\mathbf{w}} \prod_{i:y_i=1} f(\mathbf{x}_i) \cdot \prod_{i:y_i=0} (1 - f(\mathbf{x}_i)) \\
&= \arg\max_{w_0,\mathbf{w}} \prod_{i} f(\mathbf{x}_i)^{y_i} \cdot (1 - f(\mathbf{x}_i))^{1-y_i} \qquad // \text{ log} \\
&= \arg\max_{w_0,\mathbf{w}} \sum_{i} y_i \cdot log(f(\mathbf{x}_i)) + (1 - y_i) \cdot log(1 - f(\mathbf{x}_i)) \\
&= \arg\min_{w_0,\mathbf{w}} -\sum_{i} y_i \cdot log(f(\mathbf{x}_i)) + (1 - y_i) \cdot log(1 - f(\mathbf{x}_i))
\end{aligned}$$

Logistic Regression: Optimization

$$\underset{w_0,\mathbf{w}}{\arg\min} \; \underbrace{-\sum_i y_i \cdot log(f(\mathbf{x}_i)) + (1 - y_i) \cdot log(1 - f(\mathbf{x}_i))}_{\text{cross entropy } E(w_0,\mathbf{w})}$$

Remarks

- This function E is also known as the **cross-entropy**.
- It cannot be minimized analytically. But there are **numerical solutions**, such as gradient descent or Newton's method.
- The weights in $\mathbf{w}$ indicate how **important** each feature is for the classifier.

Logistic Regression: Regularization

- **Observation**: The model tends to **overfit** if ...
  - ... single features get too much weight.
  - ... many unimportant features obtain a small weight $\neq 0$ each.
- **Approach**: We **regularize** the learning problem to avoid outlier weights and clip weights to zero.
- We compute the weight vector **w**'s **norm**, i.e.:

$$||\mathbf{w}||_1 := |w_1| + |w_2| + ... + |w_d| \qquad \text{L1-Norm}$$

$$||\mathbf{w}||_2 := \sqrt{w_1^2 + w_2^2 + ... + w_d^2} \qquad \text{L2-Norm}$$

- We adapt the optimization problem such that "wrong" weights are **"punished"**:

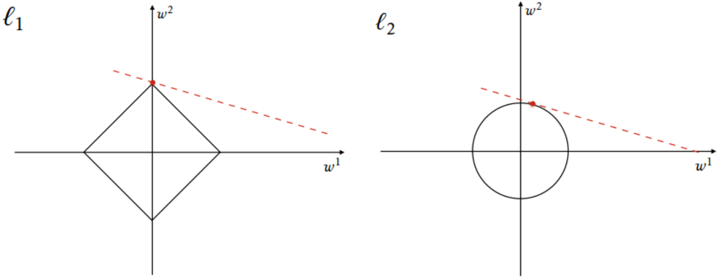
$$\arg\min_{w_0, \mathbf{w}} \; E(w_0, \mathbf{w}) \; + \; C \cdot ||\mathbf{w}||_1 \qquad // \text{ L1 regularization}$$

$$\arg\min_{w_0, \mathbf{w}} \; E(w_0, \mathbf{w}) \; + \; C \cdot ||\mathbf{w}||_2 \qquad // \text{ L2 regularization}$$

- $C > 0$ ist a free parameter.

Logistic Regression: Regularization

Effects of L1 vs. L2 regularization



- We maximize a linear target function with constraint $||\mathbf{w}||_1 = 1$ (left) and $||\mathbf{w}||_2 = 1$ (right).
- L1 regularization tends to set uninformative features' weights to zero. The classifier **selects features**, the solution is **sparse**.
- L2 regularization avoids **outliers** *(= extreme weights)*.

Logistic Regression: Multi-Class Problems

- So far, we have only looked at two classes:

$$P(C{=}1|\mathbf{x}) = \frac{1}{1 + e^{-(w_0 + \mathbf{x}\cdot\boldsymbol{w})}} \quad ; \quad P(C{=}0|\mathbf{x}) = \frac{e^{-(w_0 + \mathbf{x}\cdot\boldsymbol{w})}}{1 + e^{-(w_0 + \mathbf{x}\cdot\boldsymbol{w})}}$$

- For **multi-class problems**, each class gets a weight vector: $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_K$.
- We adapt the formula slightly → **multinomial** logistic regression:

$$P(C{=}1|x) = \frac{exp\left(\mathbf{x}\cdot\mathbf{w_1}\right)}{\sum_{k=1}^{K} exp\left(\mathbf{x}\cdot\mathbf{w_k}\right)}$$

$$P(C{=}2|x) = \frac{exp\left(\mathbf{x}\cdot\mathbf{w_2}\right)}{\sum_{k=1}^{K} exp\left(\mathbf{x}\cdot\mathbf{w_k}\right)}$$

$$\ldots$$

$$P(C{=}K|x) = \frac{exp\left(\mathbf{x}\cdot\mathbf{w_K}\right)}{\sum_{k=1}^{K} exp\left(\mathbf{x}\cdot\mathbf{w_k}\right)}$$

- **Idea**: Compute scores $\mathbf{x}\cdot\mathbf{w_K}$ per class, **normalize** them to probabilities.

Logistic Regression: Discussion

Aspects of Logistic Regression

- **simplicity** ☺, only linear decision boundaries ☹.
- **interpretability**: assign a weight for each feature ☺
- **tradition**.
- **correctness** for normally distributed classes of equal variance.
- **Few parameters** to fit → good results even for few training samples *(rules of thumb: 10 samples per class per feature are sufficient).*

Would **linear** regression work? → no, because ...

... the output values would not be probabilities *(in [0,1])*.

Outline

1. Basics
2. Logistic Regression
3. ML in NLP
4. Application Example: Named Entity Recognition

Why use ML in NLP?

A sample text
Robin hood was successful at U.S. box offices. It is not even remotely a cool movie, though.

... encoded with a Caesar chiffre
Spcjo ippe xbt tvddfttgvm bu VzTz cpy pggjdftz ju jt opu fwfo sfnpufmz b dppm npwjf uipvhiuif rvjdl cspxo gpy kvnqt pwfs uif mbaz ephz

... as a token sequence
17234 5689 437 47381 259 3784 93847 78437 17 24 15 75 1746 429875 28 9827 552341 33 426 17

NLP is Hard!

- ... combine information on character level (U.S.), word level (robin hood), sentence level (sentiment?).
- **Rules** are **not enough** *(not ... cool?)*!
- **Learning** needs to generalize across various ways to express the same semantics!

Machine Learning and NLP

We use machine learning to learn from **sample texts** how to interpret words/sentences/documents in new contexts.

Examples

1. **Part-of-Speech (POS) Tagging**

   "The ***bear*** *survives in summer on fish and fruit."* → *NN*
   "Your efforts will ***bear*** *fruit."* → *VB*

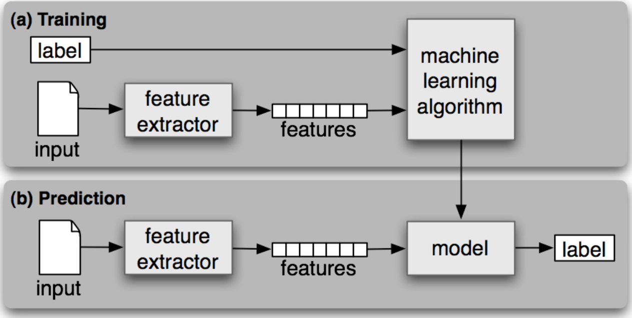
2. **Sentiment Analysis**

   "I can not believe it – What a cool video!" → ☺
   "This video is not cool – What a..." → ☹

3. **Answer Type Detection**

   "Who founded Virgin Airlines" → *PERSON/INDIVIDUAL*
   "What state capital is the largest?" → *GEO/CITY*

ML Setup for Text



1. **Feature Extraction**: transform text (as a string) into a feature vector $\mathbf{x}$.
2. **Classification**: map $\mathbf{x}$ to a class *(e.g., by logistic regression)*.

Features in NLP: Bag-of-Words

Feature Extraction (here, 'Bag-of-Words' features)

- ... transforms text documents into feature vectors $\mathbf{x}$.
- **Step 1**: Preprocessing *(lowercasing, stemming)*.
- **Step 2**: Collect all tokens in a vocabulary $\{t_1, \ldots, t_m\}$.

Documents

"The two most important days in your life are the day you are born and the day you find out why."

"Every second of your life is important so learn to enjoy"

Tokens (preprocessed)

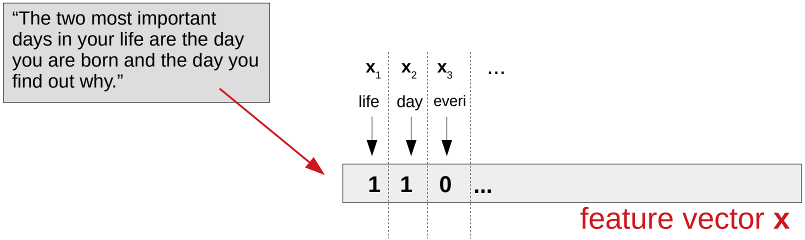
{ '**day**', 'most', 'two', 'born', 'in', 'out', 'why', 'are', 'you', 'life', 'import', 'the', 'find', 'and' }

{ '**everi**', 'of', 'is', 'import', 'enjoy', 'learn', 'to', 'you', 'life', 'so', 'second' }

Vocabulary

{ 'enjoy', 'to', 'you', 'so', 'out', 'import', 'the', 'why', 'day', 'most', 'and', 'everi', 'of', 'is', 'two', 'born', 'in', 'learn', 'are', 'life', 'second', 'find' }

Features in NLP: Bag-of-Words



Step 3: Compute the feature vector $\mathbf{x}$

- Every document is transformed to a boolean vector $\mathbf{x} = (x_1, \ldots, x_m) \in \{0, 1\}^m$.
- Each entry x_i is 1 if term t_i appears in the document *(and $x_i = 0$ otherwise)*.
- Note that the *order of terms* is neglected!

Features in NLP: Advanced Features

- Usually, we use **more than single words** as features.
- **"Good" features** depend on the problem at hand!
- We use various **feature functions** f to derive features from text.

Examples

- **Sentence Segmentation**: capitalization, presence of known abbreviations *("U.S.A.", "e.g.", ...)*
- **spam classification**: identity of sender, ...
- **sentiment classification**: adjectives, bigrams, ... *("very funny" vs. "not funny")*
- ...

We often "help" the system by selecting "good" features

- Why? higher efficiency, less overfitting.

Features in NLP: Feature Functions

- We use so-called feature functions $f_1, \dots, f_m$ to generate our feature vector $\mathbf{x}$.
- A **Feature function** f_i is given an input word/text x and a class c and computes a feature value $f_i(x, c)$.
- The feature vector becomes class-dependent: $\mathbf{x}^c = (f_i(x, c), f_2(x, c), \dots, f_m(x, c))$
- Often, features are binary *(indicator functions)*.

Examples

- Sentiment Classification

$$f_i(x, c) = \begin{cases} 1 & \text{if } (\text{"great"} \in x \\ & \quad \wedge\, c = +) \\ 0 & \text{else} \end{cases}$$

- Sentence segmenter

$$f_i(x, c) = \begin{cases} 1 & \text{if } (\mathit{case}(w_{i+1}) = \mathit{upper} \\ & \quad \wedge\, c = \mathit{EOS}) \\ 0 & \text{else} \end{cases}$$

Features in NLP: Examples [4] (Kapitel 7)

Feature functions

$f_1(x,c) = \mathbf{1}_{'great' \in x \,\wedge\, c=+}$

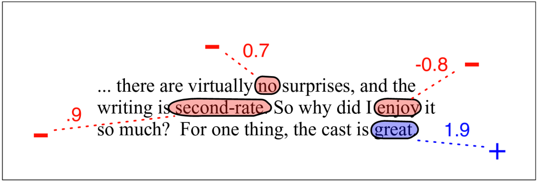
$f_2(x,c) = \mathbf{1}_{'no' \in x \,\wedge\, c=-}$

$f_3(x,c) = \mathbf{1}_{'second-rate' \in x \,\wedge\, c=-}$

$f_4(x,c) = \mathbf{1}_{'enjoy' \in x \,\wedge\, c=-}$

$f_5(x,c) = \mathbf{1}_{'chuck\ norris' \in x \,\wedge\, c=+}$

Learned Weights



Classification Result

$$P(c=+|x) = \frac{e^{1.9+0}}{e^{1.9+0} + e^{0.9+0.7-0.8}} = 82\%$$

$$P(c=-|x) = \frac{e^{0.9+0.7-0.8}}{e^{1.9+0} + e^{0.9+0.7-0.8}} = 18\%$$

Outline

1. Basics

2. Logistic Regression

3. ML in NLP

4. Application Example: Named Entity Recognition

Information Extraction

- **Information Extraction** deals with extracting "meaning" from text.
- Usually, this is limited to **entities** and **relations** between them.

Here: Named Entity Detection

- A "named entity" is a **unique, named object**.

named entities	**no named entities**
D. Trump, McDonalds, Nile, 13.04.2017, 20$	*sand, cat, company*

- Dates and important quantities are often considered named entities *(see above)*.
- Entities are often assigned to **classes** *(z.B. "person", "organization", "money", "location")*.

Named Entity Recognition (NER): Example image: [4]

Citing high fuel prices, [$_{\text{ORG}}$ **United Airlines**] said [$_{\text{TIME}}$ **Friday**] it has increased fares by [$_{\text{MONEY}}$ **$6**] per round trip on flights to some cities also served by lower-cost carriers. [$_{\text{ORG}}$ **American Airlines**], a unit of [$_{\text{ORG}}$ **AMR Corp.**], immediately matched the move, spokesman [$_{\text{PER}}$ **Tim Wagner**] said. [$_{\text{ORG}}$ **United**], a unit of [$_{\text{ORG}}$ **UAL Corp.**], said the increase took effect [$_{\text{TIME}}$ **Thursday**] and applies to most routes where it competes against discount carriers, such as [$_{\text{LOC}}$ **Chicago**] to [$_{\text{LOC}}$ **Dallas**] and [$_{\text{LOC}}$ **Denver**] to [$_{\text{LOC}}$ **San Francisco**].

Example Applications

- **linking** texts with structured information sources *(Wikipedia, databases, knowledge graphs)*
- **sentiment analysis**: *Who/what* is a text about?
- **question answering**: detecting *answer candidates*.

Remarks

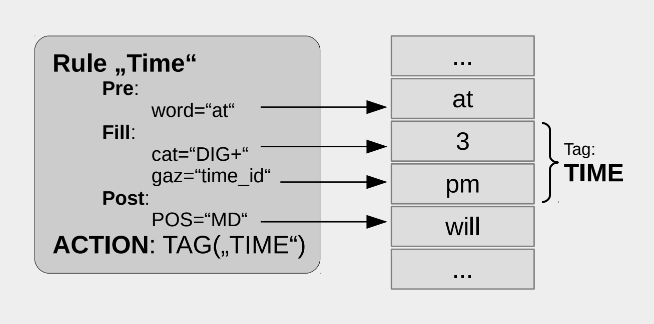
- NER includes the **segmentation** of entities:

Als Angela Merkel Seehofer rügte...

NER: Approaches

Commonly, we combine three approaches

1. rule-based systems



2. matching n-grams against collections of **known** names/places/persons *(so-called gazetteers)*
3. machine learning *(here)*.

NER with Machine Learning

- We formulate NER as a **sequence-to-sequence problem**: Given a sequence of tokens $x_1, ..., x_n$, generate a corresponding sequence of **tags** $t_1, ..., t_n$.
- We choose the **tag vocabulary** such that the labels cover both **entity types** and **boundaries**!

3 Types of Labels ("BIO" labels)

1. **B-T**: an entity with type T **start** here *(B-PERSON, B-LOCATION, ...)*
2. **I-T**: the token is part of an entity but does **not start here** *(I-PERSON, I-LOCATION, ...)*
3. **O**: the token belongs to **no** entity.

Americal	*Airlines*	*,*	*a*	*company*	*in*	*New*	*York*	*...*
B-ORG	*I-ORG*	*O*	*O*	*O*	*O*	*B-LOC*	*I-LOC*	*...*

NER with ML: Features

Which features are interesting to assign tokens to named entities (and entity types)?

- **word form** (X/x = uppercase/lowercase letter, d = digit)

  "We announced the TGX-42 today, which ..."
  → *XXX-dd (or shorter: X-d)*

- **prefixes/suffixes** of various length, e.g. $\mathbf{1}_{suffix(w)='ton'}$

  "He was in distress" vs. "He was in Fartington"

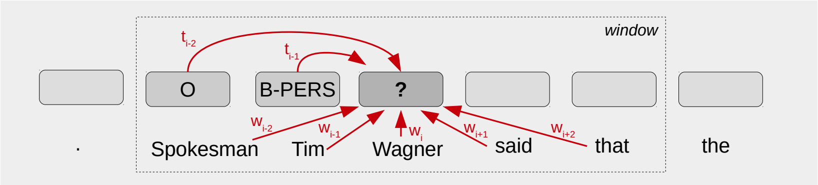
- **POS Tags**, of neighbor words, e.g. $\mathbf{1}_{POS(w(-1))=PREPOSITION}$

  "...the CEO of infor ..."

- **identity** of words, e.g. $\mathbf{1}_{w(+1)='said'}$
- presence in gazeteer
- ...

NER: Classifier

- To label token x_i, we pool features from a **local window** $x_{i-w}, \ldots, x_{i-1}, x_i, x_{i+1}, \ldots, x_{i+w}$.
- We also use the **tags/labels** of **preciding tokens** $t_{i-w}, \ldots, t_{i-1}$ as features.



Inference / Tagging of Sequences

- **Method 1**: Classify each term left-to-right, make a hard local decision per term *(Greedy-Decoding)*.
- **Method 2 (better)**: Global optimization with so-called *Viterbi coding* [4] (Chapter 10).

References I

[1] Google DeepDream robot: 10 weirdest images produced by AI 'inceptionism' and users online (Photo: Reuters).
http://www.straitstimes.com/asia/east-asia/alphago-wins-4th-victory-over-lee-se-dol-in-final-go-match (retrieved: Nov 2016).

[2] 'Untergang der Titanic' Illustration von Willy Stöwer für die Zeitschrift Die Gartenlaube.
https://de.wikipedia.org/wiki/RMS_Titanic (retrieved: Oct 2016).

[3] S. Bird, E. Klein, and E. Loper.
Natural Language Processing with Python.
O'Reilly Media, Inc., 2009.

[4] Daniel Jurafsky and James H. Martin.
Speech and Language Processing: An Introduction to Natural Language Processing (3rd Edition Draft Chapters).
2017.

[5] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean.
Distributed Representations of Words and Phrases and their Compositionality.
In Advances in Neural Information Processing Systems 26, pages 3111–3119. Curran Associates, Inc., 2013.

[6] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis.
Human-level control through deep reinforcement learning.
Nature, 518(7540):529–533, 02 2015.

[7] Mary-Ann Russon.
Google DeepDream robot: 10 weirdest images produced by AI 'inceptionism' and users online.
http://www.ibtimes.co.uk/google-deepdream-robot-10-weirdest-images-produced-by-ai-inceptionism-users-online-1509518 (retrieved: Nov 2016).

References II

[8] C. Szegedy.
Building a deeper understanding of images (Google Research Blog).
https://research.googleblog.com/2014/09/building-deeper-understanding-of-images.html
(retrieved: Nov 2016).