

Vorlesungsskript

Maschinelle Lernverfahren für die Sprachtechnologie

Institut für Computerlinguistik
Universität Zürich

<http://www.cl.uzh.ch>

Simon Clematide
simon.clematide@uzh.ch

Frühlingssemester 2016

Inhaltsverzeichnis

1 Organisatorisches	5
1.1 Infos	5
1.1.1 Unterlagen	5
1.1.2 Inhalt	5
1.1.3 Leistungsnachweis	6
1.2 Software	6
2 Einführung in Maschinelles Lernen	7
2.1 Motivation	7
2.2 Maschinelles Lernen	8
2.2.1 Lerntypen	9
2.2.2 Techniken	11
2.2.3 Over-/Underfitting	15
2.3 Literatur	16
2.4 Vertiefung	18
3 Grundlagen	19
3.1 Motivation	19
3.2 Wahrscheinlichkeitstheorie	20
3.2.1 Wahrscheinlichkeitsräume	22
3.2.2 Bedingte Wahrscheinlichkeiten	24
3.3 Informationstheorie	27
4 N-Gramm-Sprachmodelle	32
4.1 Motivation	32
4.2 N-Gramme	32
4.2.1 Intro	33
4.2.2 Verteilung	34
4.2.3 Evaluation	34
4.3 N-Gramm-Schätzmethoden	34
4.3.1 Add-One	34
4.3.2 Interpolation	34
4.3.3 GT-Smoothing	34
4.3.4 KN-Smoothing	35
4.4 Fazit	35

5 Kontinuierliche Sprachmodelle	37
5.1 Rückblick	37
5.2 Distributionalismus	40
5.2.1 Modellarten	40
5.2.2 Embeddings	41
5.3 Kontinuierliche LM	41
5.3.1 Deep Neural Nets	41
5.3.2 word2vec	43
5.3.3 GloVe	44
6 Klassifikation mit NB und Entscheidungsbäumen	46
6.1 Klassifikation	46
6.2 Naive Bayes	50
6.2.1 Dokumentklassifikation	50
6.3 Decision Trees	53
6.3.1 Training	57
6.3.2 Pruning	61
6.3.3 Bagging	62
6.3.4 Random Forest	62
6.3.5 Boosting	63
6.3.6	64
7 Parametrische Modelle, MLE und MAP	66
7.1 Parametrische Modelle	66
7.1.1 Nichtparametrisch	67
7.1.2 Parametrisch	67
7.2 MLE	69
7.2.1 Analytisch	69
7.2.2 Iterativ	71
7.3 MAP	72
8 Logistic Regression und Maximum Entropy	79
8.1 Logistic Regression	80
8.2 Maximum Entropie	82
8.2.1 Motivation	82
8.2.2 Formalisierung	85
8.2.3 Beispiel	87
8.2.4 Fazit	91
9 Support Vector Machines	94
9.1 Support Vector Machines	94
9.2 Entscheidungsgrenzen	94
10 Sequenzklassifikation und CRFs	98
10.1 HMM	98
10.2 CRF	100
10.2.1 Case Study	100
10.2.2 Formal	103

11 Unsupervisierte Verfahren	109
11.1 Clustering	109
11.1.1 Flach	109
11.1.2 Hierarchisch	110
11.1.3 Topic Modeling	112
11.2 LSA	115
11.3 Embeddings	119
11.3.1 multivec	119

Abbildungsverzeichnis

2.1	Worthypothesengraph	8
2.2	Regelbasierte Systementwicklung	16
3.1	Verbundwahrscheinlichkeit	25
3.2	Bedingte Wahrscheinlichkeit	25
11.1	Zutaten für probabilistisches Topic Modeling	114

Kapitel 1

Organisatorisches

1.1 Infos

1.1.1 Unterlagen

Allgemeine Hinweise

- Weiterführende Vorlesung: Nur für Masterstudierende! (Uni-Richtlinie)
- Bei Unterbelegung (weniger als 3 Studierende) wird VL ev. nicht durchgeführt.
- Möglichkeit zur *Nachbuchung* anderer Mastervorlesungen

Olat-Kurs und Lehrmaterialien¹

Campuskurs

“16FS CL WV Maschinelle Lernverfahren für die Sprachtechnologie”

- *Folienskript* im 4-up-Format (farbig und s/w) als PDF-Dokument unter “Material”; 1-up Slides mit funktionierenden Links.
- *Lauftext* des Folienskripts mit Index (PDF) <https://files.ifi.uzh.ch/cl/siclemat/lehre/fs16/ml/script/script.pdf>
- *Literatur*: Pflichtlektüre wird elektronisch zur Verfügung gestellt via OLAT. Sofern nicht frei verfügbar, darf das Material nur innerhalb dieser Vorlesung verwendet und nicht weitergegeben werden.

1.1.2 Inhalt

Lernziele und Konzept der Vorlesung

Lernziele

- Verstehen der grundlegenden Konzepte hinter verschiedenen maschinellen Lernverfahren
- Praktisches und informiertes Anwenden von wichtigen Ansätzen und Werkzeugen

¹<https://www.olat.uzh.ch/olat/url/RepositoryEntry/14837383173>

Konzept

- Kolloquiumsartige “Vorlesung” mit hohem Diskussionsanteil bzw. aktiver Mitarbeit (14-15.45h im BIN 1.D.07);
- Selbständig gelöste Übungen (beinhaltet auch selbständige Lektüre). Individuelle Arbeit im Tutorat montags 16-17.30h im BIN 0.B.04 mit Peter Makarov!
- Besprechung der Übungen im Tutorat.

1.1.3 Leistungsnachweis

Schriftliche Übungen (SU) und schriftliche Prüfung

- 6 *Übungseinheiten/Mitarbeiten* ab Woche 3
- 50% der Schlussnote
- Jede Einheit kann mit 0, 0.5 oder 1 Punkt bestanden werden.
- Benotung: SU-Note = Punktzahl
- Normalerweise *Einzelarbeit* (im 2-Wochenrhythmus abzuliefern)
- Teilweise mit Musterlösungen, aber auch Fragen und Diskussion in der Übungsstunde
- *Schriftliche Schlussprüfung:* 50% der Schlussnote

Schriftliche Schluss-Prüfung

- Zeit: Montag, 6.6.2016 14-15h (60 Minuten)
- Theorieprüfung mit Fokus auf wesentliche Konzepte; handgeschriebener Spickzettel ist erlaubt
- Stoff: Skript, Übungen, Pflichtlektüren

1.2 Software

Software für Übungen

- Programmierübungen werden in Python erfolgen (z.T. mit sklearn, NLTK)
- Weitere Tools werden als Applikationen benutzt:
 - Sprachmodellierungssoftware
 - wapiti für CRF
 - WEKA für Data-Mining
- Arbeit mit eigenem Laptop
- Oder via ssh mit Studentenaccount `m1NN@r2d2.ifi.uzh.ch`

Kapitel 2

Einführung in Maschinelles Lernen

Lernziele

- Verschiedene Typen von maschinellen Lernverfahren (ML) kennen lernen: supervisiert, unsupervisiert und semi-supervisiert
- Verschiedene algorithmische Techniken von maschinellen Lernverfahren kennen lernen: Entscheidungsbäume, Maximum Entropie, SVMs, Clustering, EM-Algorithmen, Graphische Modelle
- Wichtiges Probleme der Modellbildung kennen lernen: Under- und Overfitting
- Übersicht über wichtige Einführungs- und Übersichtsliteratur zum Thema ML

2.1 Motivation

Statistische Modelle menschlicher Sprache

1. Colorless green ideas sleep furiously.
2. Furiously sleep ideas green colorless.

It is fair to assume that neither sentence (1) nor (2) [...] has ever occurred in an English discourse. Hence, in any statistical model for grammaticalness, these sentences will be ruled out on identical grounds as equally "remote" from English. Yet (1), though nonsensical, is grammatical, while (2) is not grammatical. — Chomsky, 1957

Geschätzt mit aggregiertem Bigramm-Sprachmodell [PEREIRA 2000] mit latenten Wortklassen

$$p(w_1 \dots w_n) = p(w_1) \prod_{i=2}^n p(w_i | w_{i-1})$$

$$\frac{p(\text{Colorless green ideas sleep furiously.})}{p(\text{Furiously sleep ideas green colorless.})} \approx 200'000$$

Empirische Widerlegung von Chomskys Argument

Der ungrammatische Satz ist ca. 200'000 Mal unwahrscheinlicher als der semantisch problematische – definitiv keine “identical grounds to be ruled out”.

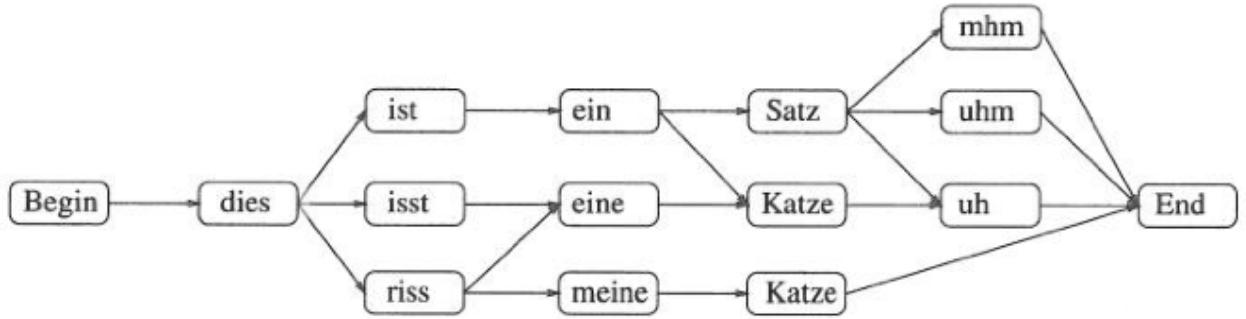


Abbildung 2.1: Worthypothesengraph aus [CARSTENSEN et al. 2004, 580]

Anwendung von Sprachmodellen

Maschinelles Lernen in der Computerlinguistik

[CHURCH 1988]: A stochastic parts program and a noun phrase parser for unrestricted text. It is surprising that a local “bottom-up” approach can perform so well. [...] remarkably few errors are related to the inadequacies of the extremely over-simplified grammar [...]. Apparently, “long distance” dependences are not very important, at least most of the time.

N-Gramm-Modelle

- seit den 50er Jahren bekannt (und verwendet)
- statistisches PoS-Tagging seit Anfang der 80er Jahre
- frühere Anwendung in der akustische Sprachverarbeitung

Maschinelles Lernen in aktueller NLP

- Sprachidentifikation
- Dokumentenklassifizierung
- Maschinelle Übersetzung
- POS-Tagging, Chunking, Parsing
- Sentiment Detection
- Sprachsynthese und Sprachanalyse (Speech-To-Text)
- ...

Dominante Methode in der NLP

Maschinelle Lernverfahren sind zum dominanten Methodenparadigma in der Sprachtechnologie geworden. Siehe z.B. <http://www.mitpressjournals.org/toc/coli/39/4>

2.2 Maschinelles Lernen

Was ist Maschinelles Lernen (*machine learning*)?

http://en.wikipedia.org/wiki/Machine_learning

Ziel

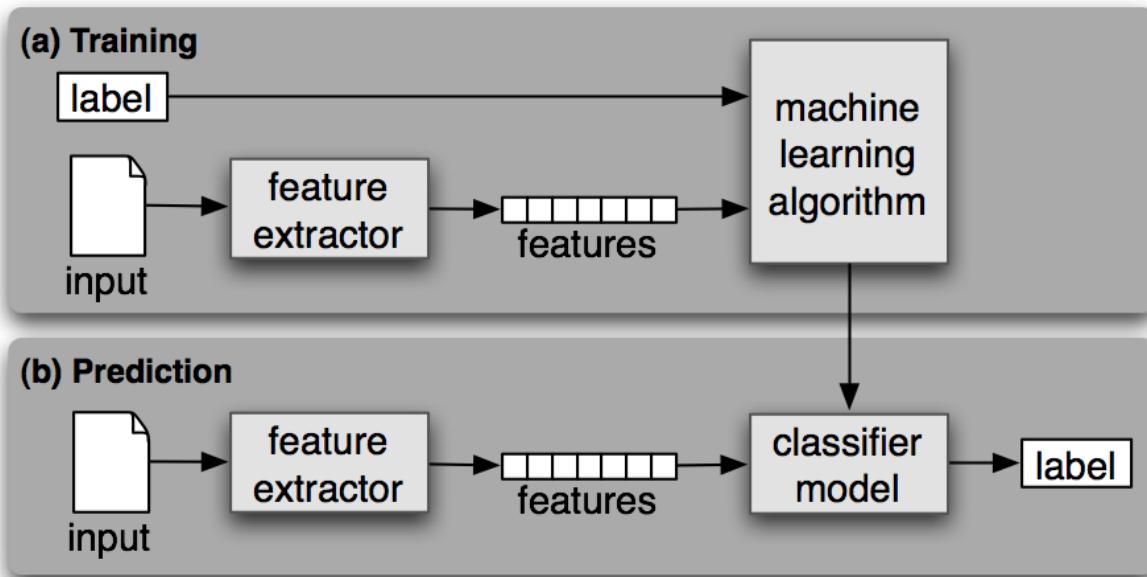
- Erkennung von Struktur sowie Vorhersage von Information aus typischerweise verrauschten und/oder grossen Datenmengen (*pattern recognition*)
- Lernen aus empirischen Daten (Generalisieren)

Grundlagen

- Statistik (Data mining = Statistics + Marketing)
- Wahrscheinlichkeitstheorie
- Informatik

2.2.1 Lerntypen

Wichtiges Paradigma der supervisierten Klassifikation



Quelle: [BIRD et al. 2009, 222]

Ansätze I: Supervisiertes Lernen (*supervised learning*)

Trainingsdaten

... fanden im	Boden (ERDE)	Spuren von Gift.
... schlägt dem Fass den	Boden (ABSCHLUSS)	aus.
... auf dem	Boden (GRUND)	lagen kostbare Teppiche ...
... fand am	Boden (ABSCHLUSS)	der Truhe eine Karte ...
... fiel auf den	Boden (GRUND)	des Ringes ...
... keine Nährstoffe mehr im	Boden (ERDE)	vorhanden, sodass

Vorgehen

1. Merkmals-Extraktion (*feature extraction*)
2. Erstellen eines Klassifikationsmodells aus vorklassifizierten Trainingsdaten
3. Güte eines Modells kann an Testdaten (*held-out data*) gemessen werden

Ansätze II: Unsupervisiertes Lernen

Trainingsdaten

... fanden im	Boden (c1)	Spuren von Gift.
... schlägt dem Fass den	Boden (c2)	aus.
... auf dem	Boden (c3)	lagen kostbare Teppiche ...
... fand am	Boden (c2)	der Truhe eine Karte ...
... fiel auf den	Boden (c3)	des Ringes ...
... keine Nährstoffe mehr im	Boden (c1)	vorhanden, sodass

Vorgehen

1. Merkmals-Extraktion
2. Erstellen eines Clusteringmodells mit n Cluster (Parameter!) über ein Ähnlichkeitsmaß
3. Güte eines Modells lässt sich erst in einer Anwendung messen.

Ansätze III: Semi-supervisiertes Lernen

Eingabe

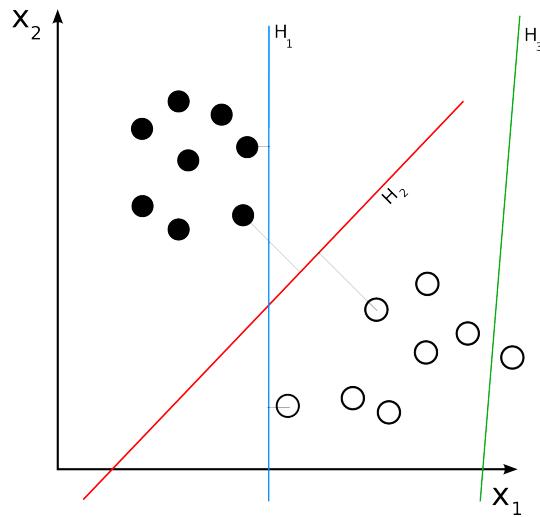
... fanden im	Boden (ERDE)	Spuren von Gift.
... schlägt dem Fass den	Boden (ABSCHLUSS)	aus.
... auf dem	Boden (GRUND)	lagen kostbare Teppiche ...
... fand am	Boden (ABSCHLUSS)	der Truhe eine Karte ...
... fiel auf den	Boden (GRUND)	des Ringes ...
... keine Nährstoffe mehr im	Boden (ERDE)	vorhanden, sodass

Aufgabe

1. Merkmals-Extraktion auf annotiertem Trainingsmaterial (klein)
2. Trainieren von Klassifikationsmodell
3. *Annotation weiterer Daten mit Modell*; allenfalls mit Ähnlichkeits-Evidenz aus Clustering von Daten
4. *Keine Klassifikation möglich: menschliche Intervention*
5. Wiederholung, bis alle Trainingsdaten klassifiziert sind

2.2.2 Techniken

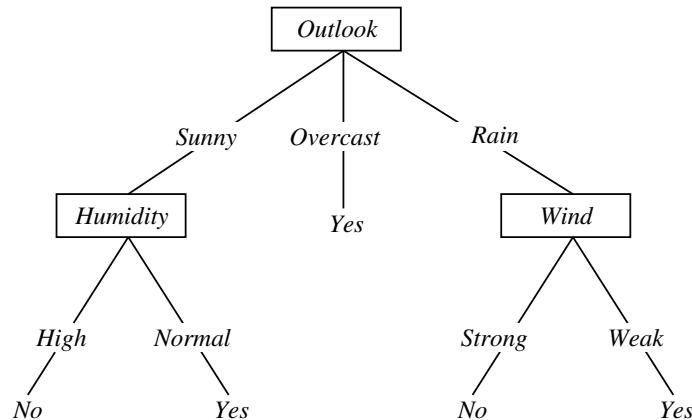
Support Vector Machines



Gibt es eine lineare Grenze? Welche ist die beste?

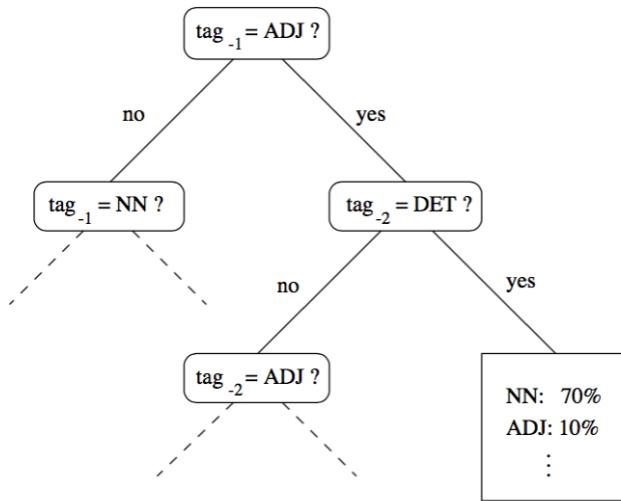
Entscheidungsbäume (*Decision Trees*)

Soll ich Tennis spielen gehen?



Entscheidungsbaum für PoS-Tagging

Welches Tag soll dem aktuellen Wort zugewiesen werden?



Quelle: [SCHMID 1994]

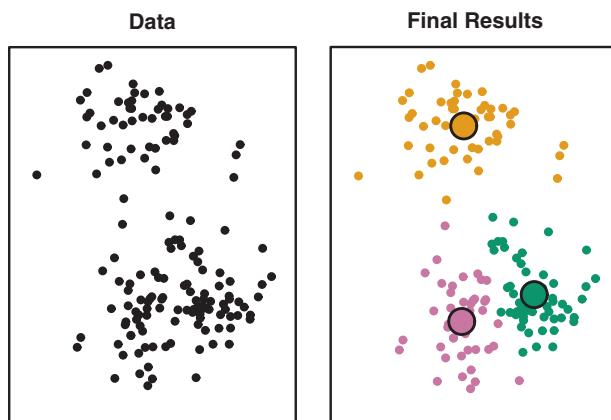
Maximum-Entropy-Modelle (auch *Logistic Regression*) vs. Naive Bayes (Abhängige) Merkmale für Textklassifikation

- Wie oft das Wort w im Text vor?
- Kommt das Wort w im gleichen Satz wie Wort v vor?
- Kommt das Wort w in mehr als 2 Sätzen vor?
- IST DAS WORT w ALL-CAPS geschrieben?

Wie gut kann ein Modell mit abhängigen Merkmalen umgehen?

- Naive Bayes: Im Prinzip nicht besonders gut.
- MaxEnt-Modell: Kein Problem. MaxEnt optimiert Wahrscheinlichkeit von Trainingsdaten ohne unmotivierte Annahmen zu treffen (uniforme Distribution hat höchste Entropie)

Clustering: k -Means



Quelle: [JAMES et al. 2013, 389]

Expectation-Maximization (EM)-Algorithmus

What
is
the
anticipated
cost
of
collecting
fees
under
the
new
proposal
?
?

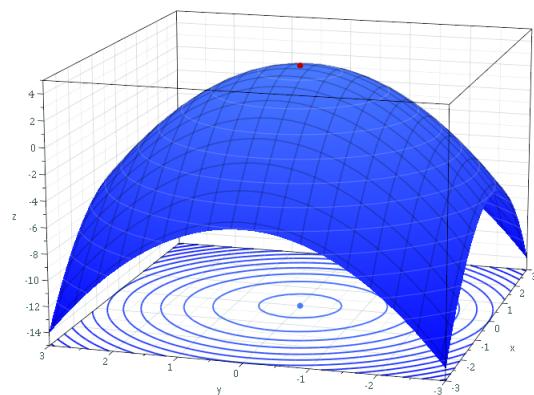
En
vertu
de
les
nouvelles
propositions
,,
quel
est
le
coût
prévu
de
perception
de
les
droits
?
?

Expectation-Maximization (EM)-Algorithmus

What
is
the
anticipated
cost
of
collecting
fees
under
the
new
proposal
?
?

En
vertu
de
les
nouvelles
propositions
,,
quel
est
le
coût
prévu
de
perception
de
les
droits
?
?

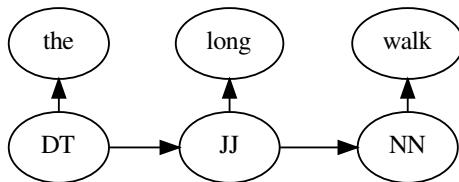
Optimierung



Gibt es ein globales Optimum? Kann es immer erreicht werden?

Graphische Modelle: Hidden-Markov-Modelle (HMM)

The/DT long/JJ walk/NN



Typische Vorhersage-Probleme (*Prediction*)

Kategoriale Vorhersage

- Klassifikation (binär oder n-är) von Einzelereignissen (Klassifikationsprobleme wie Text-kategorisierung): z.B. Logistische Regression (Maximum Entropie), Entscheidungsbäume, SVM
- Klassifikation (binäre oder n-är) von Sequenzen von Ereignissen (Tagging-Probleme): z.B. HMM, (Sequential) CRFs
- Vorhersagen von allgemeinen Strukturen (Relationen, Bäume, Graphen) (Parsing-Probleme): meist spezialisierte Ansätze, aber auch generelle Frameworks für *structured prediction* (SEARN)

Numerische Vorhersage

Schätzen einer numerischen Grösse: z.B. lineare Regression

Herausforderungen

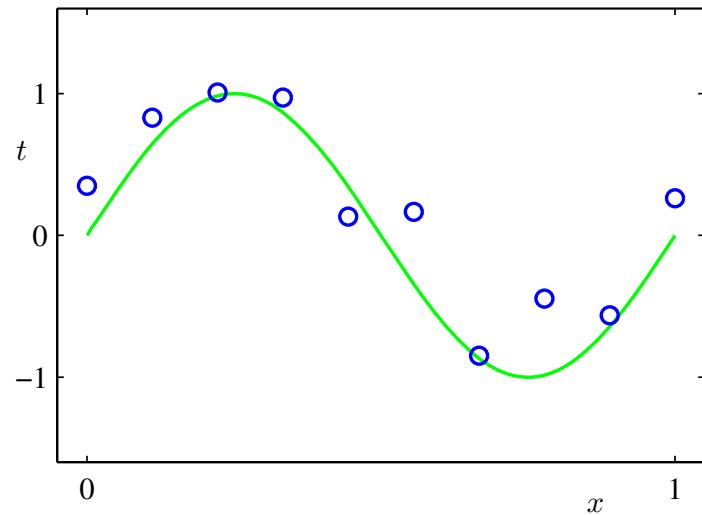
- Feature Engineering: Welche Merkmale (*features*) sind *relevant* ?
 - Verfügbarkeit
 - linguistisches Wissen
 - Informationstheorie
- Method Selection: Welche Methode kann Problem am besten lösen?
 - Statistik
 - Machine Learning
 - Theoretische Informatik

- Performance: Wie macht man Anwendung und Training effizient und optimal? Bei Big Data, hoher Dimensionalität, komplexen Strukturen?
 - Software Engineering
 - Machine Learning

2.2.3 Over-/Underfitting

Distribution und Sample Data

Plot of a training data set of $N = 10$ points, shown as blue circles, each comprising an observation of the input variable x along with the corresponding target variable t . The green curve shows the function $\sin(2\pi x)$ used to generate the data. Our goal is to predict the value of t for some new value of x , without knowledge of the green curve.



Quelle: [BISHOP 2006, 4]

Polynomial Fitting

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \dots + w_M x^M = \sum_{j=0}^M w_j x^j \quad (1.1)$$

where M is the *order* of the polynomial, and x^j denotes x raised to the power of j . The polynomial coefficients w_0, \dots, w_M are collectively denoted by the vector \mathbf{w} . Note that, although the polynomial function $y(x, \mathbf{w})$ is a nonlinear function of x , it is a linear function of the coefficients \mathbf{w} . Functions, such as the polynomial, which are linear in the unknown parameters have important properties and are called *linear models* and will be discussed extensively in Chapters 3 and 4.

Quelle: [BISHOP 2006, 5]

Underfitting und Overfitting

Regelbasiert/handerstellt vs. datenbasiert/ML-mässig

Entitäten

Annotator, Entwickler, Datenset, Theorie/Guidelines, Merkmale, Modell, System, ML-Trainings-Algorithmus

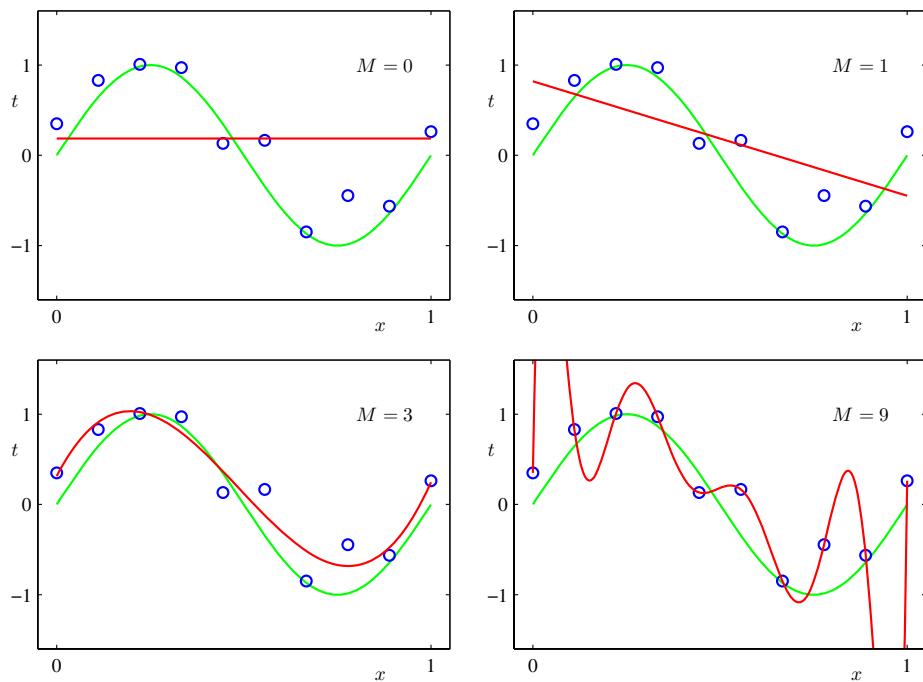


Figure 1.4 Plots of polynomials having various orders M , shown as red curves, fitted to the data set shown in Figure 1.2.

Quelle: [BISHOP 2006, 7]

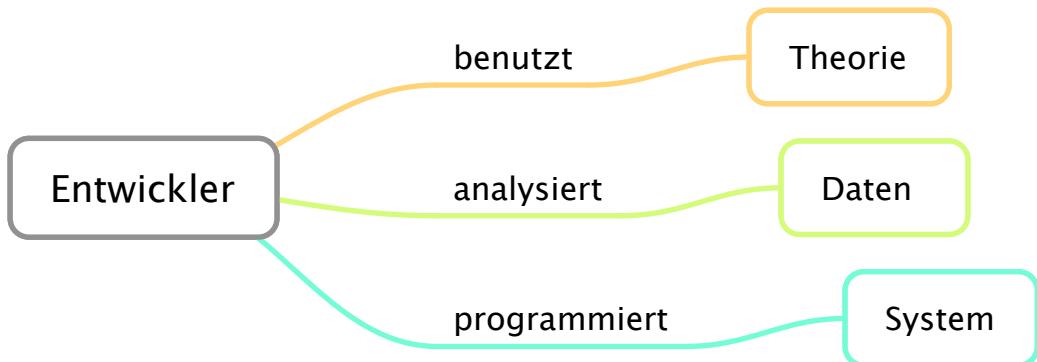


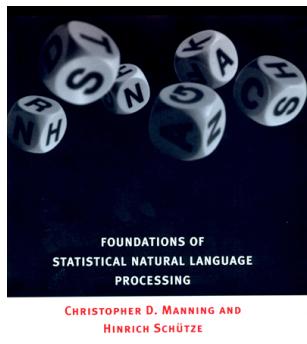
Abbildung 2.2: Regelbasierte Systementwicklung

Relationen

annotiert, benutzt, bestimmt...

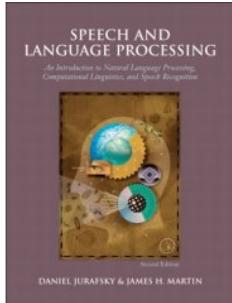
2.3 Literatur

[MANNING und SCHÜTZE 1999]: Foundations of Statistical NLP



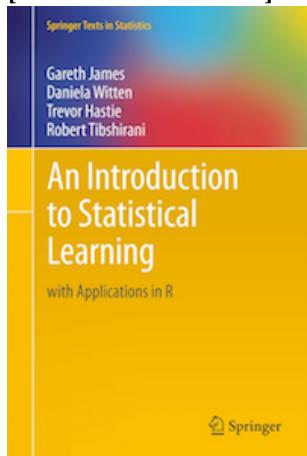
- Das Grundlagenwerk für SNLP, obwohl schon etwas älter
- mathematisch und algorithmisch (nicht spezifisch für eine Programmiersprache)

[JURAFSKY und MARTIN 2008]: Speech and Language Processing



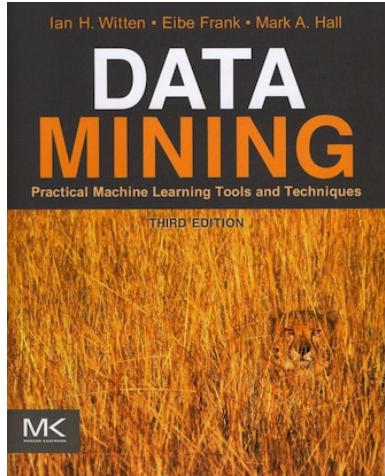
- Standardwerk für NLP
- Kapitel zu HMMs und MEMMs

[JAMES et al. 2013]: An Introduction to Statistical Learning



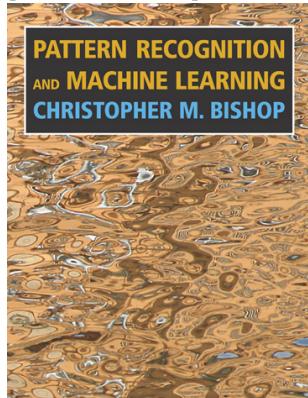
- Einführendes Standardwerk für Maschinelles Lernen (Regression und Klassifikation, aber auch unsupervisierte Verfahren)
- So wenig Mathematik wie nötig, um die Kernkonzepte sauber zu verstehen
- Verschiedene praktische Verfahren (in R) mit breiter Anwendung
- Gibt Videomaterial von MOOCs dazu

[WITTEN et al. 2011]: Data Mining



- Praktisch ausgerichtetes Standardwerk für Data Mining
- Erklärt viele Ansätze, welche im Weka-Toolkit verfügbar sind

[BISHOP 2006]: Pattern Recognition and Machine Learning



- Nicht NLP-spezifisch
- Grundlage für die Vorlesungen zu graphischen Modellen
- Umfangreich und hochwertig

Zusammenfassung

- Einführung zu Maschinellem Lernen
- Übersicht zu verschiedenen Techniken im Maschinellen Lernen
- Literatur

2.4 Vertiefung

- Pflichtlektüre zum Thema "Statistischem Modellieren": Kapitel 2 "Everything you ever wanted to know about statistics (well, sort of)" aus [FIELD und MILES 2010]

Kapitel 3

Grundlagen

Lernziele

- Übergang von Häufigkeitsverteilungen in Daten zu statistischen Modellen: Maximum-Likelihood-Modelle, Simple Naive Bayes Classifier
- Grundbegriffe der Wahrscheinlichkeitstheorie kennen: Wahrscheinlichkeitsräume, Ergebnis, Ereignis, Zufallsvariable, Wahrscheinlichkeitsfunktion einer Zufallsvariable
- Bedingte und Verbundswahrscheinlichkeit und ihr Zusammenhang bei unabhängigen Ereignissen: Bayes' Theorem
- Informationstheoretische Grundbegriffe: Entropie, Kreuzentropie (*Cross Entropy*), Kullback-Leibler-Divergenz, *Mutual Information* (gegenseitige Information)
- Vergleiche von statistischen Modellen

Exkurs: Terminologie in ML und Statistik

3.1 Motivation

Samples von Sprachen

Was sich seit Monaten angedeutet hatte, was nach dem letzten, ernüchternden Vorbereitungstreffen der Unterhändler **in Barcelona** keine Überraschung mehr ist, es reift nun zur sicheren Prognose: Das **in** diversen Konferenzen beschworene Ziel, **in** diesem Jahr noch den soliden Rahmen für ein globales Klimaschutz-Abkommen **in** der Nachfolge des Kyoto-Protokolls zu zimmern,

Machine Learning	Statistics
networks, graphs	model
weights	parameters
learning	fitting
generalization	test set performance
supervised learning	regression/classification
unsupervised learning	density estimation, clustering
large grant = \$1,000,000	large grant = \$50,000
http://statweb.stanford.edu/~tibs/stat315a/glossary.pdf	

ist illusorisch geworden. Daran ist nicht mehr zu zweifeln, wenn nicht einmal der dänische Ministerpräsident Lars Løkke Rasmussen als Repräsentant des traditionell optimistisch auftretenden Gastgebers noch an einen Erfolg glaubt. Kopenhagen wird eine Station von vielen werden auf dem schwierigen Weg der Staatengemeinschaft, die Treibhausgase **in** diesem Jahrhundert gerade noch auf ein halbwegs erträgliches Ausmaß zu begrenzen.

(Triviale) Beobachtung

Einige Wörter kommen häufiger vor als andere.

Frage

Wie induzieren wir aus diesen Beobachtungen ein statistisches Modell?

3.2 Wahrscheinlichkeitstheorie

Text = Sequenz von Wörtern

Begrifflichkeiten

Text t Sequenz $w_0 \dots w_{n-1}$ von Symbolen w

Symbol w Atomare Texteinheiten Wörter oder Buchstaben oder Laute ...

Vokabular V Menge aller möglichen Symbole w

Sprache Alle Texte, die mit V gebildet werden können.

Text der Länge n

$$t = w_0, w_1, w_2, \dots, w_{n-2}, w_{n-1}$$

Worthäufigkeiten abschätzen: $P(w)$

Was sich seit Monaten angedeutet hatte, was nach dem letzten, ernüchternden Vorbereitungstreffen der Unterhändler **in Barcelona** keine Überraschung mehr ist, es reift nun zur sicheren Prognose: Das **in** diversen Konferenzen beschworene Ziel, **in** diesem Jahr noch den soliden Rahmen für ein globales Klimaschutz-Abkommen **in** der Nachfolge des Kyoto-Protokolls zu zimmern, ist illusorisch geworden. Daran ist nicht mehr zu zweifeln, wenn nicht einmal der dänische Ministerpräsident Lars Løkke Rasmussen als Repräsentant des traditionell optimistisch auftretenden Gastgebers noch an einen Erfolg glaubt. Kopenhagen wird eine Station von vielen werden auf dem schwierigen Weg der Staatengemeinschaft, die Treibhausgase **in** diesem Jahrhundert gerade noch auf ein halbwegs erträgliches Ausmaß zu begrenzen.

$$P(\text{in}) = \frac{5}{107} \approx 0.047$$

$$P(\text{Barcelona}) = \frac{1}{107} \approx 0.009$$

Maximum-Likelihood-Schätzung

Wahrscheinlichkeit aus Häufigkeitsverteilung abschätzen

$$P_{\text{MLE}}(w) = \frac{C(w)}{N}$$

$C(w)$ Absolute Häufigkeit (*frequency*) von Wort w

N Gesamtanzahl Wörter im Text

MLE: Maximum Likelihood Estimate

- Das Modell maximiert die Wahrscheinlichkeit der Trainingsdaten.
- Es reserviert keine Wahrscheinlichkeitsmasse für unbekannte Ereignisse (d.h. Wörter, welche nicht im Text vorkommen)
- Optimal für Trainingsdaten, aber nicht optimal für alle andern.

Textwahrscheinlichkeiten abschätzen: $P(w_0, \dots, w_{n-1})$

Ein Text aus zwei Wörtern

$$t = w_0, w_1 = \text{in Barcelona}$$

Bei Unabhängigkeit gilt Multiplikationsregel

$$P(w_0, w_1) = P(w_0) \times P(w_1)$$

$$P(t) = P(w_0, w_1) = P(w_0) \times P(w_1) = \frac{5}{107} \times \frac{1}{107} = \frac{5}{11449}$$

- Gilt nur, wenn w_0 unabhängig von w_1 ist.
- Beispiel: Aus einer Urne mit Wörtern auf Zettelchen wird zufällig mit Zurücklegen gezogen (*bag-of-words model*).
- Beispiele für andere unabhängig Ereignisse?

Mehrteilige Termkandidaten identifizieren

(Pointwise) Mutual Information (PMI)

- Idee: Ein Korpus mit N Wörtern ist eine Folge von N zufälligen unabhängigen Ziehungen aus dem Vokabular.
- Empirische Wahrscheinlichkeit für ein Wort "eisern" ergibt sich aus der Vorkommenshäufigkeit f in einem Korpus: $P(\text{eisern}) = \frac{f(\text{eisern})}{N}$
- Erwartete Wahrscheinlichkeit nacheinander "eisern Vorhang" zu finden: $\hat{P}(\text{eisern Vorhang}) = P(\text{eisern}) \times P(\text{Vorhang})$
- Empirische Wahrscheinlichkeit: $P(\text{eisern Vorhang}) = \frac{f(\text{eisern Vorhang})}{N_2}$
- $$\text{PMI} = \log \frac{P(\text{eisern Vorhang})}{\hat{P}(\text{eisern Vorhang})}$$

Andere Kollokationsmasse (logDice, MI log Freq) mit Sprachdaten unter www.dwds.de

3.2.1 Wahrscheinlichkeitsräume

Grundbegriffe: Ereignis (*event*)

Ereignis (*event*)

“Ein Ereignis (auch Zufallsereignis) ist [...] eine Menge von Ergebnissen eines Zufallsexperiments, dem eine Wahrscheinlichkeit zugeordnet werden kann.”

[http:](http://de.wikipedia.org/wiki/Ereignis_(Wahrscheinlichkeitstheorie))

//de.wikipedia.org/wiki/Ereignis_(Wahrscheinlichkeitstheorie)

- Zufallsexperiment: Wurf eines Würfels; Ziehen eines Wortzettels aus einer Urne
- Ergebnis: Würfelzahl; Wort
- Elementarereignis: Einermenge eines Ergebnisses (ein Wurf); Ziehen eines bestimmten Worts
- Ereignis: Beliebige Menge von Ergebnissen: Z.B. das Würfeln einer geraden Zahl; Ziehen einer Präposition

Wahrscheinlichkeitsräume

Ein formaler Wahrscheinlichkeitsraum ist ein Triple (Ω, Σ, P)

- Nicht-leere Ergebnismenge Ω (*outcomes*)
- σ -Algebra Σ über Ω : Ereignismenge (*events*), welche aus Teilmengen von Ω besteht und unter Komplement und Vereinigung abgeschlossen ist.
- Wahrscheinlichkeitsmass (*probability measure*) $P: \Sigma \rightarrow [0, 1]$, für das gilt
 - $P(\emptyset) = 0$
 - $P(\Omega) = 1$
 - $P(A_1 \cup A_2 \cup \dots) = P(A_1) + P(A_2) + \dots$ (für disjunkte A_i)

Diskrete Wahrscheinlichkeitsräume

Faire Münze: Kopf (*H*), Zahl (*T*)

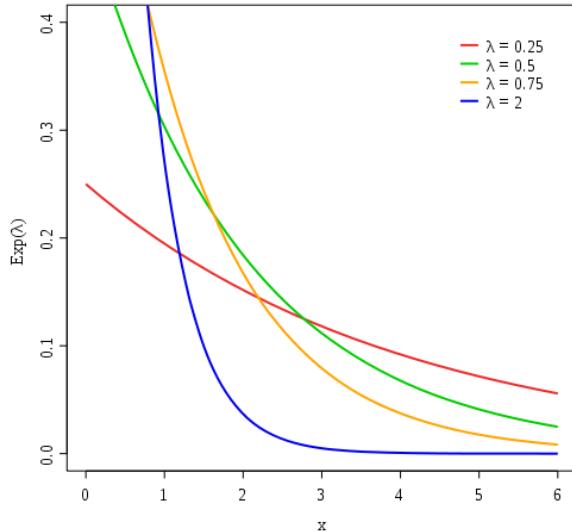
- $\Omega = \{H, T\}$
- $\Sigma = 2^\Omega$ (Potenzmenge, d.h. Menge aller Teilmengen)
- $P = \{(\emptyset, 0), (\{H\}, 0.5), (\{T\}, 0.5), (\{H, T\}, 1)\}$

Für endliche und diskrete Wahrscheinlichkeitsräume ist die Potenzmenge die Standard- σ -Algebra.

Reelle/kontinuierliche Wahrscheinlichkeitsräume

Nicht-negative reelle Zahlen mit Exponentialverteilung ($\lambda = 0.5$)

- $\Omega = \mathbb{R}_{\geq 0} = [0, \infty)$
- $\Sigma = \mathcal{B}(\mathbb{R}) \cap [0, \infty) = \mathcal{B}([0, \infty))$ (Borelsche Algebra)
- $P_{\text{Exp}(0.5)}(A) = \int_A 0.5 \exp(-0.5x)dx$
- Bei überabzählbar grossen Zahlenräumen (reelle Zahlen) hat jede einzelne Zahl die Wahrscheinlichkeit 0.
- Analogie aus Geometrie: Ein Punkt hat keine Länge. Streckenintervalle haben eine Länge.
- Deshalb muss man mit Intervallen von reellen Zahlen arbeiten. Das macht die Borelsche σ -Algebra (Komplement, Vereinigung von Intervallen).



Werte einer Zufallsvariable

Ein Wurf eines fairen 6-seitigen Würfels

Wert	$P(X = x)$
1	$\frac{1}{6}$
2	$\frac{1}{6}$
3	$\frac{1}{6}$
4	$\frac{1}{6}$
5	$\frac{1}{6}$
6	$\frac{1}{6}$

Uniforme Verteilung

Alle Ereignisse haben die gleiche Wahrscheinlichkeit.

Grundbegriffe: Zufallsvariable

Zufallsvariable/-grösse (*random variable*), Zufallszahl/-vektor/-matrix

“Eine Zufallsvariable lässt sich formal als Funktion beschreiben, die den Ergebnissen eines Zufallsexperiments Werte (so genannte Realisierungen) zuordnet.” <http://de.wikipedia.org/wiki/Zufallsvariable>

- Werte einer Zufallszahl: Normalerweise eine reelle Zahl $n \in \mathbb{R}$
- Eine Zufallsvariable X bildet Ergebnisse ($\omega \in \Omega$) auf Werte ab: $X : \Omega \rightarrow \mathbb{R}$
- Mit Zufallsvariablen kann man numerisch rechnen.
- Mit Ereignissen muss man Mengenoperationen machen.
- Werte können auch Folgen von reellen Zahlen sein (Zufallssequenz: $X : \Omega \rightarrow \mathbb{R}^n$)

Werte einer Zufallsvariable und ihre Wahrscheinlichkeit

Ein Wurf eines fairen 6-seitigen Würfels

Ergebnis ω	Ereignis	Zufallsvariablenwert $X(\omega)$	$P(X = x)$
1	{1}	1	$\frac{1}{6}$
2	{2}	2	$\frac{1}{6}$
3	{3}	3	$\frac{1}{6}$
4	{4}	4	$\frac{1}{6}$
5	{5}	5	$\frac{1}{6}$
6	{6}	6	$\frac{1}{6}$

Notationskonventionen Wahrscheinlichkeiten der Werte: $P(X = x)$ oder $p_X(x)$ oder explizit $P(\{\omega \in \Omega \mid X(\omega) = x\})$

Grundbegriffe: Wahrscheinlichkeitsverteilung

Wahrscheinlichkeitsverteilung (*probability (mass) function*)

“Eine Wahrscheinlichkeitsverteilung ist jeweils einer **Zufallsvariablen** zugeordnet. Dabei beschreibt die Wahrscheinlichkeitsverteilung, mit welchen Wahrscheinlichkeiten die Zufallsvariable ihre möglichen Werte annimmt.” <http://de.wikipedia.org/wiki/Wahrscheinlichkeitsverteilung>

$$P(X = x)$$

- “identisch verteilt” (*identically distributed, i.d.*): Mehrere Zufallsvariablen haben dieselbe Verteilung.
- “unabhängig und identisch verteilt” (*independent and identically distributed, i.i.d*)

3.2.2 Bedingte Wahrscheinlichkeiten

Bedingte Wahrscheinlichkeit (*conditional probability*)

Bedingte Wahrscheinlichkeit (definiert falls $P(B) > 0$)

$$P(A \mid B) = \frac{P(A \cap B)}{P(B)}$$

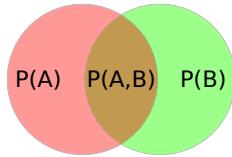


Abbildung 3.1: Verbundwahrscheinlichkeit

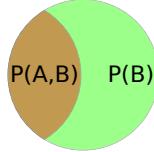


Abbildung 3.2: Bedingte Wahrscheinlichkeit

Bedingte Wahrscheinlichkeit

Normalisierung

Für alle B gilt:

$$\sum_{A \in \Sigma} P(A | B) = 1$$

Bedingte Wahrscheinlichkeit

Geschlechterbestimmung von Vornamen

- Geschlechtsklassen: $C = \{\text{maskulin}, \text{feminin}\}$
- Merkmal letzter Buchstabe: $F = \{a, \neg a\}$

Häufigkeitsverteilung

	$F=a$	$F \neq a$	
$C=\text{maskulin}$	29	2914	2943
$C=\text{feminin}$	1773	3228	5001
Total	1802	6142	7944

Bedingte Wahrscheinlichkeit $P(C | F)$

	$F=a$	$F \neq a$	
$C=\text{maskulin}$	0.0161	0.4744	
$C=\text{feminin}$	0.9839	0.5256	
	1	1	

Randverteilung (marginal distribution)

Verbundswahrscheinlichkeit $P(C, F)$ (*joint probability*)

	F=a	$F \neq a$	
C=maskulin	0.0037	0.3668	0.3705
C=feminin	0.2232	0.4063	0.6295
	0.2268	0.7732	

Randverteilungen

- Wenn wir alle Verbundswahrscheinlichkeiten jeder Reihe/Spalte summieren, erhalten wir $P(C)$ und $P(F)$ jeweils an den Rändern (*margins*) der Tabelle.
- Prozess nennt sich auch ausmarginalisieren von Variablen.
- Im Zusammenhang mit bedingten Wahrscheinlichkeiten sind $P(C)$ und $P(F)$ apriori-Wahrscheinlichkeiten.

Probabilistische Klassifikation: Simpler Naive Bayes

In classification learning problems, the learner is given a set of training examples and the corresponding class labels, and outputs a classifier. The classifier takes an unlabeled example and assigns it to a class. Many classifiers can be viewed as computing a set of *discriminant functions* of the example, one for each class, and assigning the example to the class whose function is maximum (Duda & Hart, 1973). If E is the example, and $f_i(E)$ is the discriminant function corresponding to the i th class, the chosen class C_k is the one for which¹

$$f_k(E) > f_i(E) \quad \forall i \neq k. \quad (1)$$

Suppose an example is a vector of a attributes, as is typically the case in classification applications. Let v_{jk} be the value of attribute A_j in the example, $P(X)$ denote the probability of X , and $P(Y|X)$ denote the conditional probability of Y given X . Then one possible set of discriminant functions is

$$f_i(E) = P(C_i) \prod_{j=1}^a P(A_j=v_{jk}|C_i). \quad (2)$$

[DOMINGOS und PAZZANI 1997, 103]

3 Probleme im Zusammenhang mit Simplem Naive Bayes ([DOMINGOS und PAZZANI 1997, 107])

- Unabhängigkeitsannahme der Merkmale: Wahrscheinlichkeiten werden zwar verfälscht bei abhängigen Merkmalen, allerdings kann die Klassifikation trotzdem korrekt sein.
- Zero Counts: Wahrscheinlichkeit von Merkmalen, welche im Trainingsset nie aufgetaucht sind, aber im Testset auftauchen können. Smoothing
- Kontinuierliche Wertebereiche: Schätzen von Wahrscheinlichkeit von numerischen Werten

3.3 Informationstheorie

Grundfrage

Wie viele Bits braucht man, um eine Zufallsvariable zu speichern?

Informationsgehalt (*self-information*)

Informationsgehalt (auch Überraschung *surprisal*) eines Ereignisses x

$$I(x) = \log_2 \frac{1}{P(x)} = -\log_2 P(x)$$

$\frac{1}{P(x)}$ Invertierung

Logarithmus

x	$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{2}$	0	1	2	4	8
$\log_2(x)$	-3	-2	-1	?	0	1	2	3

Basis 2: bits (log ohne Basis heisst auf Slides immer \log_2)

Basis e : nats

 : In der Informationstheorie wird manchmal $\log(0) = 0$ angenommen.

Erwartungswert einer Zufallsvariable

Erwartungswert \mathbb{E}

$$\mathbb{E}(X) = \sum_x P(x)x$$

Erwartungswert von Wurf

$$\begin{aligned} \mathbb{E}(X) &= \sum_{x=1}^6 \frac{1}{6}x \\ &= \frac{1}{6} \sum_{x=1}^6 x \\ &= 3.5 \end{aligned}$$

Frage: Wozu kann der Erwartungswert nützlich sein?

Entropie als erwarteter Informationsgehalt

Entropie (*entropy*)

$$\begin{aligned} H(X) &= \mathbb{E}(I(X)) \\ &= \mathbb{E}\left(\log_2 \frac{1}{P(X)}\right) \\ &= \sum_x P(x) \log_2 \frac{1}{P(x)} \\ &= -\sum_x P(x) \log_2 P(x) \end{aligned}$$

Entropie einer Zufallsvariable/Verteilung

Die Entropie ist die durchschnittliche Unsicherheit einer einzelnen Zufallsvariable.

$$H(X) = - \sum_x P(x) \log P(x)$$

Buchstaben-Entropie von Zeichenketten

BABBCBBBABCBCBCCACCABABCBCBABC

Frequenzen

x	P(x)
A	0.20
B	0.48
C	0.32

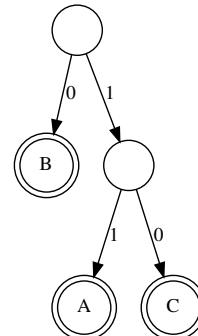
Entropie

$$H(X) \approx 1.49$$

Interpretation: Im Schnitt brauchen wir mindestens ca. 1.49 Bits, um ein Symbol zu speichern.

Code-Bäume

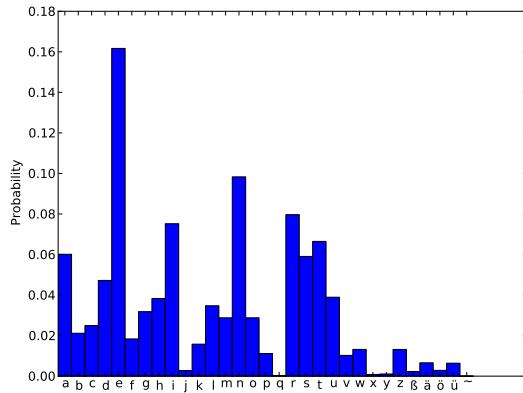
Code-Baum



Kodierter Text

B	0
A	11
B	0
B	0
C	10
B	0

Buchstaben-Entropie von Deutsch



- Entropie: 4.2 bits ($\log(18.38) \approx 4.2$)
- Uniform: $\log(31) \approx 4.95$ bits

Wozu Entropie?

Nutzen

- Entropie macht Aussage wie schwierig das Vorhersagen eines Ereignisses ist. Frage: Wie vorhersehbar ist der Text?
- Kreuzentropie ist Mass für Modellqualität. Frage: Wie gut kann das Modell den Text vorhersehen?
- Kullback-Leibler-Divergenz ist Distanzmass zwischen zwei Modellen. Frage: Wie stark unterscheiden sich zwei Modelle?

Anwendungen in Machine Learning

- Intrinsisches Mass für Schwierigkeit/Qualität.
- Entropieveränderung (*information gain*) als Kriterium im Feature Engineering.
- Cross-Entropy-Minimization als Methode der Modelloptimierung.

Cross-Entropy Entropie des Modells q auf echter Verteilung p

Definition

$$H(p, q) = - \sum_x p(x) \log q(x)$$

Schätzung mit Text als ‘echter’ Verteilung

$$H(T, q) \approx -\frac{1}{n} \sum_{i=1}^n \log q(x_i)$$

q Modellwahrscheinlichkeit (aus Trainingsdaten)

T Text

$x_{1\dots n}$ Evaluationsdaten (aus Text gewonnen)

$H(T, q)$ ist im optimalen Fall $H(T)$. Was bedeutet das?

Modelle vergleichen: KL-Divergenza.k.a Informationsgewinn, relative Entropie
Kullback-Leibler-Divergenz

$$\begin{aligned} D(p \parallel q) &= \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)} \\ &= \sum_{x \in X} p(x)(\log p(x) - \log q(x)) \end{aligned}$$

p **echte** Verteilung; komplettes Modell der Sprache

q Modell von p , z.B. per MLE aus Trainingsdaten geschätzt

Eigenschaften

- $D(p \parallel q) \geq 0$
- $D(p \parallel q) = 0$ nur wenn $p = q$

Erwarteter Informationsgewinn Mutual Information

Fragestellung

Wie stark kann ich die Entropie einer Variable durch weitere Beobachtungen verringern? Der erwartete Informationsgewinn entspricht der Mutual Information.

Definition

$$\begin{aligned} I(X; Y) &= D(p(x, y) \parallel p(x)p(y)) \\ &= \sum_{x \in X, y \in Y} p(x, y) \log \frac{p(x, y)}{p(x)p(y)} \end{aligned}$$

Mutual Information: Beispiel 1

Merkmal: Name *endet* mit a (wahr (1) oder falsch (0))

Verbundswahrscheinlichkeit $P(C, F)$ (joint probability)

	F=1	F=0	
C=maskulin	0.0037	0.3668	0.3705
C=feminin	0.2232	0.4063	0.6295
	0.2268	0.7732	

Mutual Information

$$\begin{aligned} I(C; F) &= p(m, 1) \log \left(\frac{p(m, 1)}{p(m)p(1)} \right) + p(m, 0) \log \left(\frac{p(m, 0)}{p(m)p(0)} \right) \\ &\quad + p(f, 1) \log \left(\frac{p(f, 1)}{p(f)p(1)} \right) + p(f, 0) \log \left(\frac{p(f, 0)}{p(f)p(0)} \right) = 0.1524 \end{aligned}$$

Mutual Information: Anfangsbuchstabe

Merkmal: Name *beginnt* mit A (wahr (1) oder falsch (0))

Verbundswahrscheinlichkeit $P(C,F)$ (joint probability)

	F=1	F=0	
C=maskulin	0.0268	0.3437	0.3705
C=feminin	0.0558	0.5738	0.6295
	0.0826	0.9174	

Mutual Information

$$I(C; F) = p(m, 1) \log\left(\frac{p(m, 1)}{p(m)p(1)}\right) + p(m, 0) \log\left(\frac{p(m, 0)}{p(m)p(0)}\right) \\ + p(f, 1) \log\left(\frac{p(f, 1)}{p(f)p(1)}\right) + p(f, 0) \log\left(\frac{p(f, 0)}{p(f)p(0)}\right) = 0.0006$$

Fazit

Mutual Information

- grosse Mutual Information zeigt an, dass zwei Variablen (hier: Merkmal und Klasse) **nicht** unabhängig sind.
- bedeutet für uns, dass wir (bekanntes) Merkmal für die Vorhersage der (unbekannten) Klasse verwenden können.

Weiterführende Bemerkungen

Verwandte Konzepte in der Informationstheorie

- Bedingte Entropie (*conditional entropy*): Wie viel Entropie bleibt in Zufallsvariable übrig bei Kenntnis einer anderen Zufallsvariable?
- Verbundsentropie (*joint entropy*): Wie viel Bits benötige ich, um beide Zufallsvariablen zu encodieren?
- Perplexität (*perplexity*): $PP(X) = 2^{H(X)}$
- Je niedriger die Perplexität (und mithin die Entropie), desto besser ist das Modell!

Entropie-Algebra

Ähnlich wie bei Wahrscheinlichkeiten können wir zwischen marginaler, bedingter und gemeinsamer Entropie hin- und herrechnen.

Vertiefung

- Pflichtlektüre: Kapitel 2 von [MANNING und SCHÜTZE 2000]: Insbesondere 2.1.1 bis und mit 2.1.8; 2.2.1-3; 2.2.5-2.2.7
- Edx-Kurs mit guten Lehrvideos: MITx: 6.041x Introduction to Probability - The Science of Uncertainty; Unit 4 (discrete random variables) und 5 (continuous random variables)
- Naive Bayes: Der Anfang des Artikels [DOMINGOS und PAZZANI 1997] ist gut lesbar.

Kapitel 4

N-Gramm-Sprachmodelle

Lernziele

- Einfache statistische Modelle von Wortfolgen in natürlicher Sprache verstehen
- Zweck und Wirkung der Markov-Annahme verstehen
- N-Gramm-Wahrscheinlichkeiten schätzen
- Umgang mit unbekanntem Vokabular beim Anwenden
- Umgang mit *Sparse Data*: Smoothing-, Interpolation- und Backoff-Techniken für nicht gesehene oder seltene N-Gramme

4.1 Motivation

Shannon Game [SHANNON 1951]

bahnhof _

Shannon's Frau sieht einen Text aus n Buchstaben und muss solange den nachfolgenden Buchstaben raten, bis sie den korrekten geraten hat.

Entropie von English

Modell	Entropie
Uniform	$4.76 (\log(26 + 1))$
Häufigkeiten	4.03
Mensch	1.30

Entropie misst die Schwierigkeit, eine Zufallsvariable vorherzusagen.
Warum ist die Task für den Menschen leichter?

4.2 N-Gramme

Zipf'sches Gesetz \dagger : Potenzgesetz der Verteilung von Wörtern

<i>r</i>	<i>w</i>	<i>f</i>
1	the	69971
2	,	58334
3	.	49346
4	of	36412
5	and	28853
6	to	26158
7	a	23195
8	in	21337
9	that	10594
10	is	10109
11	was	9815
12	he	9548
13	for	9489
14	"	8837
15	"	8789

Zipf's law states...

the frequency of any word is inversely proportional to its rank in the frequency table.

$$f \propto \frac{1}{r}$$

$$f \cdot r = k$$

Konsequenzen

Wenige Wörter sind sehr häufig! Viele Wörter sind sehr selten!

4.2.1 Intro

Statistische Sprachmodelle auf Wortfolgen: Probabilistische Grammatiken

Video "Introduction to N-Grams" (8 Minuten) von SNLP-Kurs von Dan Jurafsky und Chris Manning¹

 : Wichtige Regel: Melde dich, wenn du etwas nicht verstehst!

Reflexion

- Inwiefern sind Grammatiken und statistische Sprachmodelle äquivalent?
- Im Zeitalter von Big-Data: Können wir nicht die Satzwahrscheinlichkeiten direkt schätzen?

Theoretisch gesehen immer, praktisch gesehen nicht

$$P(w_i | w_1, \dots, w_{i-1}) = \frac{f(w_1, \dots, w_i)}{f(w_1, \dots, w_{i-1})}$$

¹<https://www.youtube.com/watch?v=s3kK1UBa3b0>

(Vokabular =)	1-Gramme	10^5
	2-Gramme	10^{10}
	3-Gramme	10^{15}
	4-Gramme	10^{20}

4.2.2 Verteilung

Wahrscheinlichkeiten von N-Grammen schätzen

Video “Estimating N-Gram Probabilities” (8 Minuten) von SNLP-Kurs <https://www.youtube.com/watch?v=s3kK1UBa3b0>

4.2.3 Evaluation

Normalisierte Perplexität als Evaluationsmass

Video “Evaluation and Perplexity” (11 Minuten) von SNLP-Kurs <https://www.youtube.com/watch?v=0HyVNCvnsTo>

4.3 N-Gramm-Schätzmethoden

4.3.1 Add-One

Laplace-Smoothing: Add-One

Video “Smoothing: Add-one (Laplace) Smoothing” (6 Minuten) von SNLP-Kurs <https://www.youtube.com/watch?v=d8nVJj1M0Yo>

4.3.2 Interpolation

Interpolation und Backoff-Modelle

Video “Interpolation, Backoff, and Web-Scale LMs” (10 Minuten) von SNLP-Kurs <https://www.youtube.com/watch?v=-aMYz1tMfPg>

N-Gramm-PoS-Modell des TnT-Taggers

```
%% TnT statistically tagged file, Mon Mar  7 09:39:20 2016
%% lexicon      : uis.lex
%% ngrams       : uis.123
%% corpus       : /dev/stdin
%% model        : trigrams
%% sparse data : linear interpolation
%% lambda1 = 1.129343e-01 lambda2 = 3.240495e-01 lambda3 = 5.630162e-01
%% unknown mode: statistics of singletons
%% case of characters is significant
%% using suffix trie up to length 10
```

4.3.3 GT-Smoothing

Good-Turing-Smoothing

Video “Good-Turing-Smoothing” (10 Minuten) von SNLP-Kurs <https://www.youtube.com/watch?v=-aMYz1tMfPg>

4.3.4 KN-Smoothing

Kneser-Ney-Smoothing

Video "Kneser-Ney-Smoothing" (10 Minuten) von SNLP-Kurs <https://www.youtube.com/watch?v=wtB00EczoCM>

4.4 Fazit

Zusammenfassung: Smoothing

- Additives Smoothing (*add k*) für Bigramme :

$$\hat{P}(w_1, w_2) = \frac{f(w_1, w_2) + k}{N + k |V^2|}$$

- N = Summe aller Häufigkeiten $f()$ im Zähler; V=Vokabular
- Laplace Smoothing, falls: $k = 1$
- Expected Likelihood Estimation (ELE), falls $k = 1/2$

- Problem: Alle ungewöhnlichen Ereignisse sind gleichwahrscheinlich!
- Problem: Das ist sehr unwahrscheinlich unter Zipf'scher Wortverteilung!

Zusammenfassung: Interpolation

Interpolation von Bigrammen

$$\hat{P}(w_1, w_2) = \lambda \frac{f(w_1, w_2)}{N} + (1 - \lambda) \frac{f(w_2)}{N}$$

N = Summe aller Häufigkeiten $f()$ im Zähler

- Nutzen: Einfache und komplexere Modelle werden miteinander verrechnet!
- Unbekannte Wörter bleiben ein Problem!

Zusammenfassung: Unbekannte Wörter (OOV)

- Benutze Spezialtoken <UNK> für *Out-of-Vocabulary Items*!
- Trainingsphase (Modellbildung):
 - Legt Vokabular V fest auf Grund von Häufigkeitskriterien! Z.B. keine Hapax Legomena!
 - Ersetze alle Wörter $w \notin V$ durch <UNK> im Trainingskorpus!
 - Behandle <UNK> wie ein normales Wort beim Zählen!
- Anwendungsphase:
 - Ersetze alle Wörter $w \notin V$ durch <UNK> im Testkorpus!
 - Behandle <UNK> wie ein normales Wort beim Zählen!

Zusammenfassung: Backoff \uparrow

- Idee: Traue den empirischen Häufigkeiten, wenn sie öfter als t vorkommen!
- Ansonsten benutze einfachere Modelle als Ersatz!

Backoff von Bigrammen

$$\hat{P}(w_1, w_2) = \begin{cases} (1 - \delta) \frac{f(w_1, w_2)}{N} & \text{if } f(w_1, w_2) > t \\ \alpha(w_1) \frac{f(w_2)}{N} & \text{otherwise} \end{cases}$$

- δ : Discounting, damit Backoff-Modell über Wahrscheinlichkeitsmasse verfügt
- $\alpha(w_i)$: Backoff-Gewicht, das die Geschichte von w_2 berücksichtigt

Vertiefung

- Pflichtlektüre: Kapitel 6 von [MANNING und SCHÜTZE 1999]
- Videos von SNLP-Kurs mit Chris Manning zu Language Modelling²
- Klassiker: [CHEN und GOODMAN 1998]

²https://www.youtube.com/playlist?list=PL4LJ1vG_SDpxQAwZYtwfXcQr7kGn19W93

Kapitel 5

Kontinuierliche Sprachmodelle

Lernziele

-

5.1 Rückblick

Klassische wortbasierte N-Gramm Modelle

- Grundidee: Rein sequentielle Wortabfolge ist hilfreich bei vielen NLP-Problemen!
- Grundprinzip: Wort ist atomares Symbol (Wortform, gestemmte Form, Lemma)
- Grundproblem: *Sparse-Data* bei $N > 1$
- Klassische Lösungsansätze: Smoothing und Backoff

Praxis: ARPA LM Dateiformat

ARPA-Format

- Wird von vielen LM-Toolkits unterstützt (z.B. SRILM, IRSTLM, CMU-Cambridge LM Toolkit)
- Back-off-Format: Darstellung interpolierter Modelle, indem man für gesehene n -Gramme direkt interpolierte Wahrscheinlichkeit speichert.
- Wahrscheinlichkeit links (Logarithmus mit Basis 10)
- n -Gramm in der Mitte
- Back-off-Gewicht (unser γ) rechts (Logarithmus mit Basis 10)

Praxis: ARPA Dateiformat für Language-Modeling (LM)

Von vielen LM-Toolkits unterstützt: SRILM[†], KENML[†], IRSTLM, ...

3-gramm-Modell im ARPA-Format

```

\data\
ngram 1=119318
ngram 2=2988549
ngram 3=3373741

\1-grams:
-4.166559 communist -0.4403845
-3.818732 united -0.6698264

\2-grams:
-2.163229 communist states -0.1036584
-1.068001 united states -0.02942845

\3-grams:
-1.148659 communist states of
-2.67347 states of america
-1.61931 states of europe

\end\

```

Dokumentation[†]

- Counts der N-Gramme
- Wahrscheinlichkeit links (Logarithmus mit Basis 10);
- n -Gramm in der Mitte
- Backoff-Gewicht rechts (Logarithmus mit Basis 10)

Klassische Evaluation nach [CHEN und GOODMAN 1998]

Kreuzentropie als Evaluationsmass

- Gemessen als Kreuzentropie über Evaluationsset.
- Gute Korrelation mit Performanz in realen Anwendungen (z.B. Wortfehlerrate in Spracherkennung).

Quantitative Schätzung mit Evaluationstext als ‘echter’ Verteilung

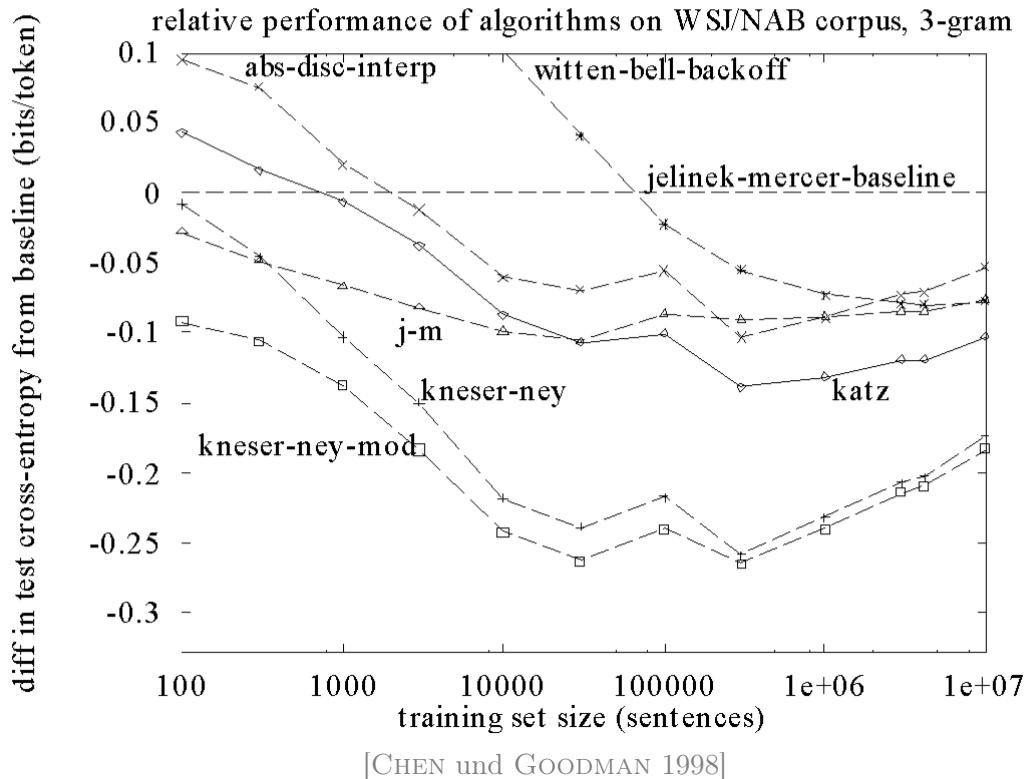
$$H(T, q) \approx -\frac{1}{n} \sum_{i=1}^n \log q(x_i)$$

q Modellwahrscheinlichkeit (aus Trainingsdaten)

T Test-Text mit n Wörtern: $T = (x_1, \dots, x_n)$

Klassische Evaluationsergebnisse nach [CHEN und GOODMAN 1998]

- Menge an Trainingsdaten beeinflusst Qualität unterschiedlich
- Smoothing ist effektives Gegenmittel gegen Spärlichkeit der Daten: höhere n -Gramm-Ordnung liefert i.d.R. auch bessere Resultate. Problem: Modellgrösse.



Je niedriger, umso besser!

Kontinuierliche vs. klassische Sprachmodelle

Table 2: Results on Penn Treebank corpus (evaluation set) after combining all models. The weight of each model is tuned to minimize perplexity of the final combination.

Model	Weight	PPL
3-gram with Good-Turing smoothing (GT3)	0	165.2
5-gram with Kneser-Ney smoothing (KN5)	0	141.2
5-gram with Kneser-Ney smoothing + cache	0.0792	125.7
Maximum entropy model	0	142.1
Random clusterings LM	0	170.1
Random forest LM	0.1057	131.9
Structured LM	0.0196	146.1
Within and across sentence boundary LM	0.0838	116.6
Log-bilinear LM	0	144.5
Feedforward NNLM	0	140.2
Syntactical NNLM	0.0828	131.3
Combination of static RNNLMs	0.3231	102.1
Combination of adaptive RNNLMs	0.3058	101.0
ALL	1	83.5

Quelle: [MIKOLOV et al. 2011]

Minimale Perplexität (PPL) von interpolierten Sprachmodellen

- Standard-LM-Datenset aus Penn-Treebank: 930k Tokens für Training, 74k Development, 82k Test
- Vokabulargrösse: 10k (alles andere gemappt auf unk)

 Eher kleineres Datenset!

5.2 Distributionalismus

Kurzgeschichte des Distributionalismus [SAHLGREN 2008]

- “Die Bedeutung eines Wortes ist sein Gebrauch in der Sprache.” (Ludwig Wittgenstein)
- “Distributional Structure” [HARRIS 1954]
- “You shall know a word by the company it keeps!” (J. R. Firth (1957))
- “words which are similar in meaning occur in similar contexts (Rubenstein & Goodenough, 1965)”
- “words with similar meanings will occur with similar neighbors if enough text material is available” (Schütze & Pedersen, 1995)
- “words that occur in the same contexts tend to have similar meanings” (Pantel, 2005)

5.2.1 Modellarten

Vielfalt distributioneller Modellierung

tokenization

annotation

tagging

parsing

feature selection

: cluster texts by date/author/discourse context/...

↓ ↴

Matrix type	Weighting	Dimensionality reduction	Vector comparison
word × document	probabilities	LSA	Euclidean
word × word	length normalization	PLSA	Cosine
word × search proximity	TF-IDF	LDA	×
adj. × modified noun	PMI	PCA	Dice
word × dependency rel.	Positive PMI	IS	Jaccard
verb × arguments	PPMI with discounting	DCA	KL
:	:	:	KL with skew

Quelle: [POTTS 2013]

5.2.2 Embeddings

Grundidee hinter Buzzword *Embeddings*

Embeddings

- Vektor-Raum-Modelle (*vector space models*) [TURNEY und PANTEL 2010]
- Dimensionsreduktion (Vektorlänge d für Wortrepräsentation: 50-1000)
- Grundidee: Ähnliche Wörter haben ähnliche Vektoren!

Task der kontinuierlichen Wortrepräsentation

Lerne den numerischen Vektor $W_i \in \mathbb{R}^d$ (=embedding), der das i -te Wort eines Vokabulars distributionell adäquat repräsentiert!

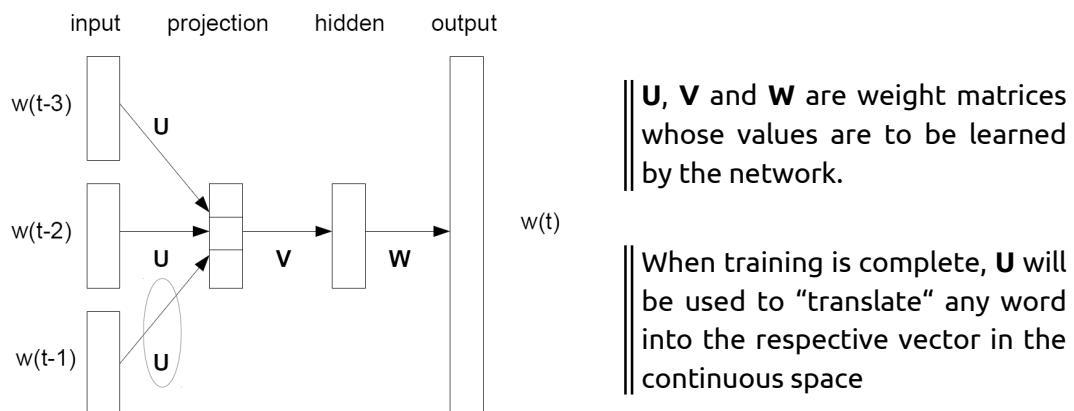
Lernmethoden: Von Deep Learning to Shallow Learning

Ursprünglich oft langsam zu trainierende komplexe Neuronale Netzwerke. Mittlerweile einfachere Modelle (word2vec, GloVe).

5.3 Kontinuierliche LM

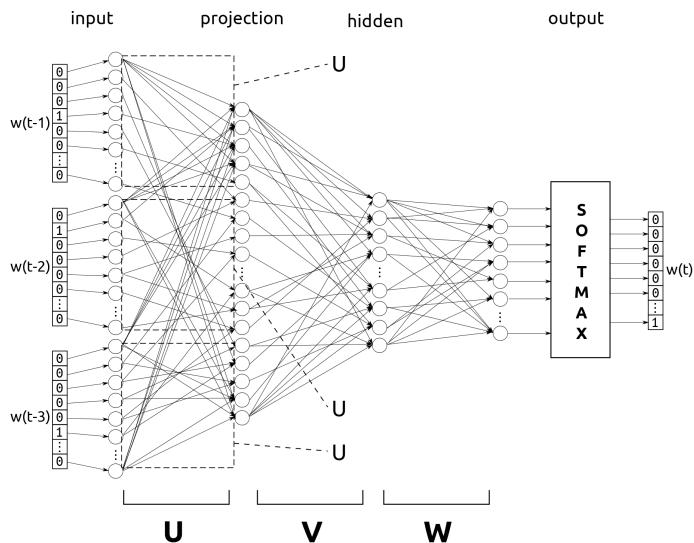
5.3.1 Deep Neural Nets

Feedforward Neural Net Language Model



- Four-gram neural net language model architecture (Bengio 2001)
- The training is done using stochastic gradient descent and backpropagation
- The word vectors are in matrix U

Feedforward Neural Net Language Model



$w(t-3)$	$w(t-2)$	$w(t-1)$	$w(t)$
the	cat	is	black
mary	has	a	lamb
i	know	this	place

The network is trained using $w(t-3)$, $w(t-2)$ and $w(t-1)$ as the input (the “context” of $w(t)$) and $w(t)$ as expected result.

Each word is used as input in its one-hot form in the vocabulary. The output of the network is continuous so a softmax function must be used to assign the output to a word.

6/34

Sehr aufwändig zum Trainieren wegen Hidden Layer und Softmax-Berechnung!

Recurrent Neural Networks

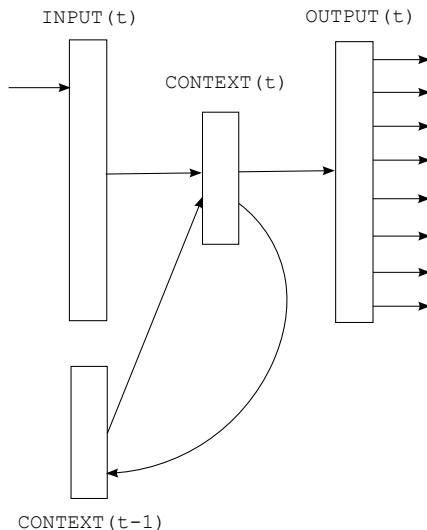


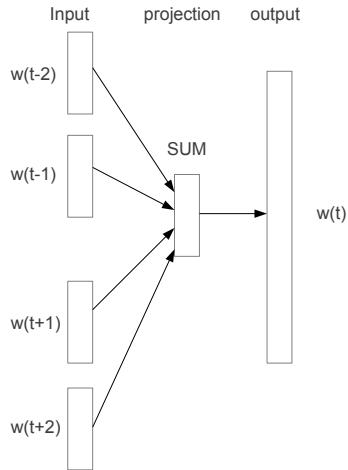
Figure: Recurrent neural network based LM

Quelle: [WESTON und BORDES 2014]

- Recurrent = mit sich selbst verknüpfte Ebene; “Kurzschluss”-Synapsen
- Kontext (vorangehende N-1 Wörter) und aktuelles Wort
- Hidden Layer integriert den Kontext des Worts w_t mit aktuellem Wort
- Besser als FFNNs

5.3.2 word2vec

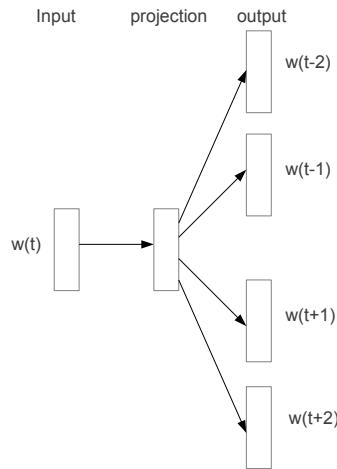
Shallow Neural Nets: Continuous Bag-of-Words (CBOW)



Quelle: [MIKOLOV 2013]

- Gegeben den Kontext, sage das aktuelle Wort voraus!
- Effizient berechenbar mit einfachen neuronalen Netzen (*shallow neural nets*)!
- Besonders gut mit *Negative Sampling* (d.h. konstruierte unwahrscheinliche Kontexte), deren Unwahrscheinlichkeit ebenfalls gelernt werden muss!

Shallow Neural Nets: Skip-Gram

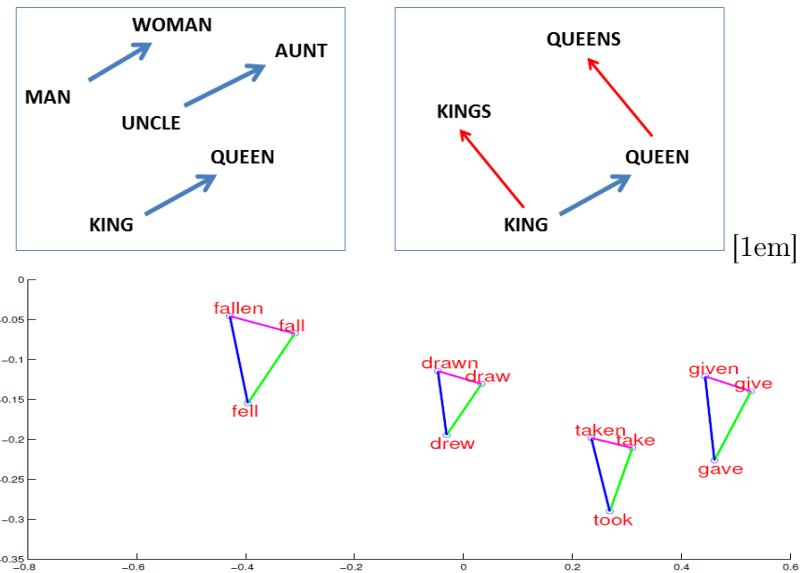


Quelle: [MIKOLOV 2013]

- Gegeben das aktuelle Wort, sage den wahrscheinlichsten Kontext voraus!
- Ebenfalls effizient berechenbar mit einfachen neuronalen Netzen (*shallow neural nets*)!
- Erlaubt die Verarbeitung von grösseren Trainingsdaten als Deep NNs, was bessere Modelle ergibt!
- *There is no data like more data...*

Zusammenhänge im Vektorraummodell

Sowohl semantische als auch syntaktische Zusammenhänge können im Vektorraum implizit gelernt und modelliert werden.



Quelle: [MIKOLOV 2013]

Aufwand und Leistung (semantisch/syntaktische Tests)

Model	Vector Dimensionality	Training Words	Training Time	Accuracy [%]
Collobert NNLM	50	660M	2 months	11
Turian NNLM	200	37M	few weeks	2
Mnih NNLM	100	37M	7 days	9
Mikolov RNNLM	640	320M	weeks	25
Huang NNLM	50	990M	weeks	13
Our NNLM	100	6B	2.5 days	51
Skip-gram (hier.s.)	1000	6B	hours	66
CBOW (negative)	300	1.5B	minutes	72

Quelle: [MIKOLOV 2013]

5.3.3 GloVe

GloVe¹: Global Vectors for Word Representations

- GloVe is an unsupervised learning algorithm for obtaining vector representations for words.
- Training is performed on aggregated global word-word co-occurrence statistics from a corpus,

¹[PENNINGTON et al. 2014] <http://www-nlp.stanford.edu/projects/glove/>

- and the resulting representations showcase interesting linear substructures of the word vector space.

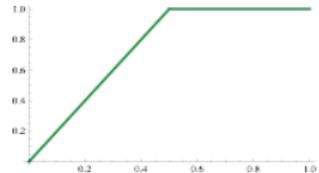
GloVe vs word2vec

- GloVe lernt aus globalen Daten
- word2vec benutzt nur lokale Fenster zum Training

GloVe: Eigenschaften

Multiplikation von Vektoren und Wahrscheinlichkeit
 “Einfaches Optimierungsziel”

$$J = \frac{1}{2} \sum_{ij} f(P_{ij}) (w_i \cdot \tilde{w}_j - \log P_{ij})^2 \quad f \sim$$



GloVe-Vorteile

- Schnell trainierbar
- Gut für kleine und grosse Korpora
- Gut mit Vektoren mit wenigen oder vielen Dimensionen

Vertiefung

- GloVe-Reimplementation in Python <http://www.fold1.me/2014/glove-python/>
- Deep-Learning-Kurs <http://cs224d.stanford.edu/syllabus.html>

Kapitel 6

Klassifikation mit NB und Entscheidungsbäumen

Herzlichen Dank an Torsten Marek und Rico Sennrich für Quelltexte. Some of the figures in this presentation are taken from “An Introduction to Statistical Learning, with applications in R” (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani

Lernziele

- Klassifikationsproblem abstrakt verstehen
- Wichtige Eigenschaften von Klassifikationsmethoden kennen
- Regression von Klassifikation unterscheiden können
- Naive Bayes und Entscheidungsbäume kennen
- Training von Entscheidungsbäumen verstehen
- Fortgeschrittene Techniken von Entscheidungsbäumen kennen (*Bagging, Random Forests*)

6.1 Klassifikation

Was ist Klassifikation?

Abstrakte Sicht auf das Klassifikationsproblem: $y = f(\mathbf{x})$
CLASS = $f(\text{FEATURES})$

Class Kategoriale Grösse (nominale/ordinale Skala)

Features Folge/Vektor von beliebigen Größen (alle Skalen)

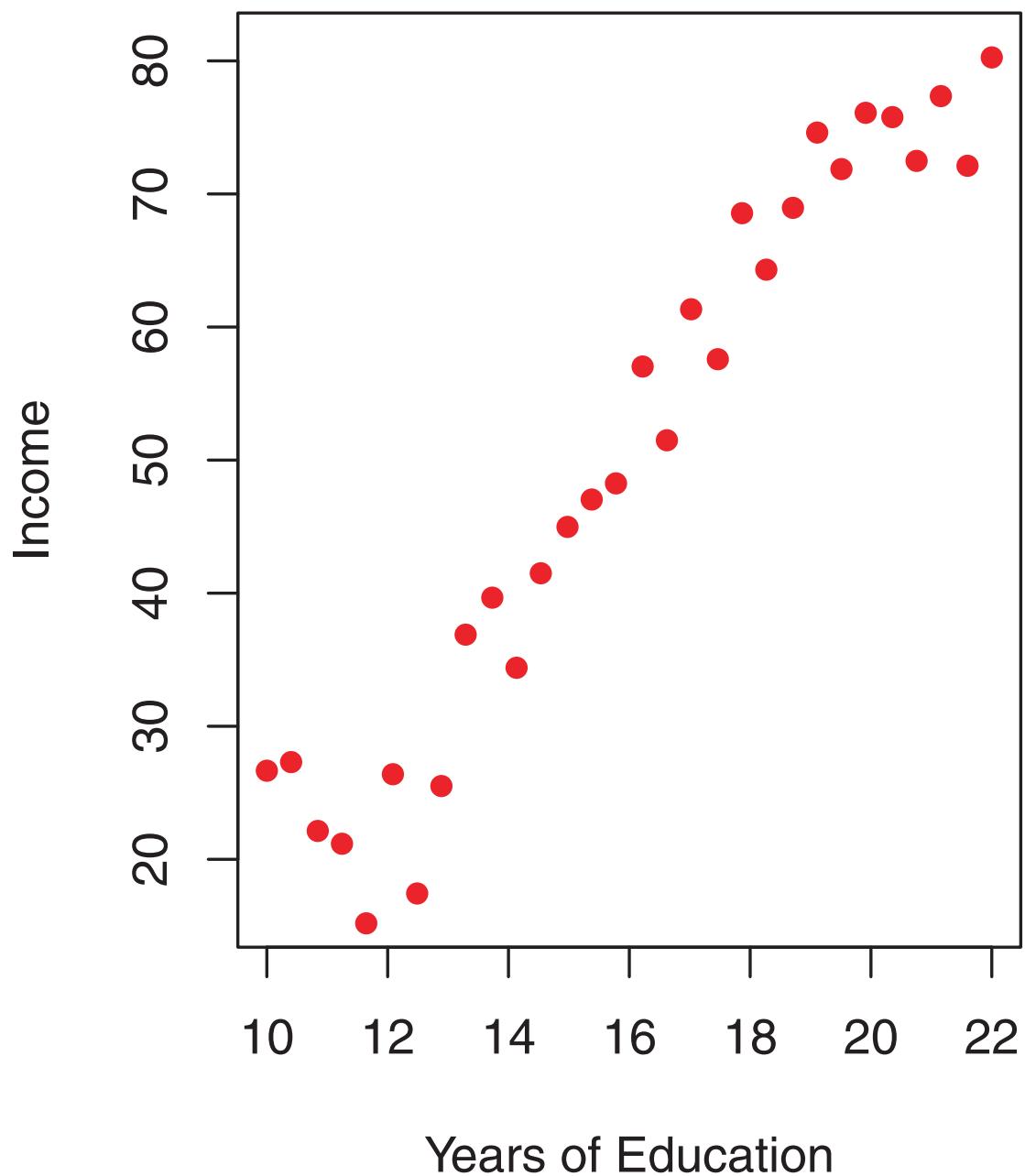
f Klassifikationsfunktion (*predictor function*)

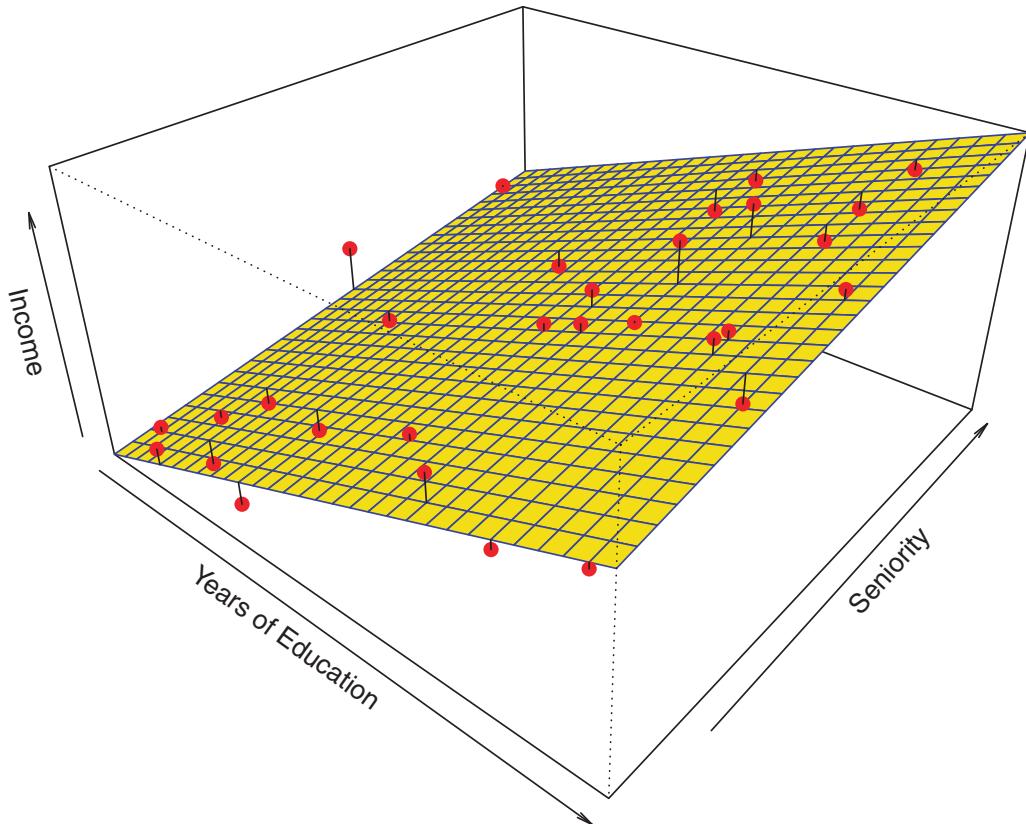
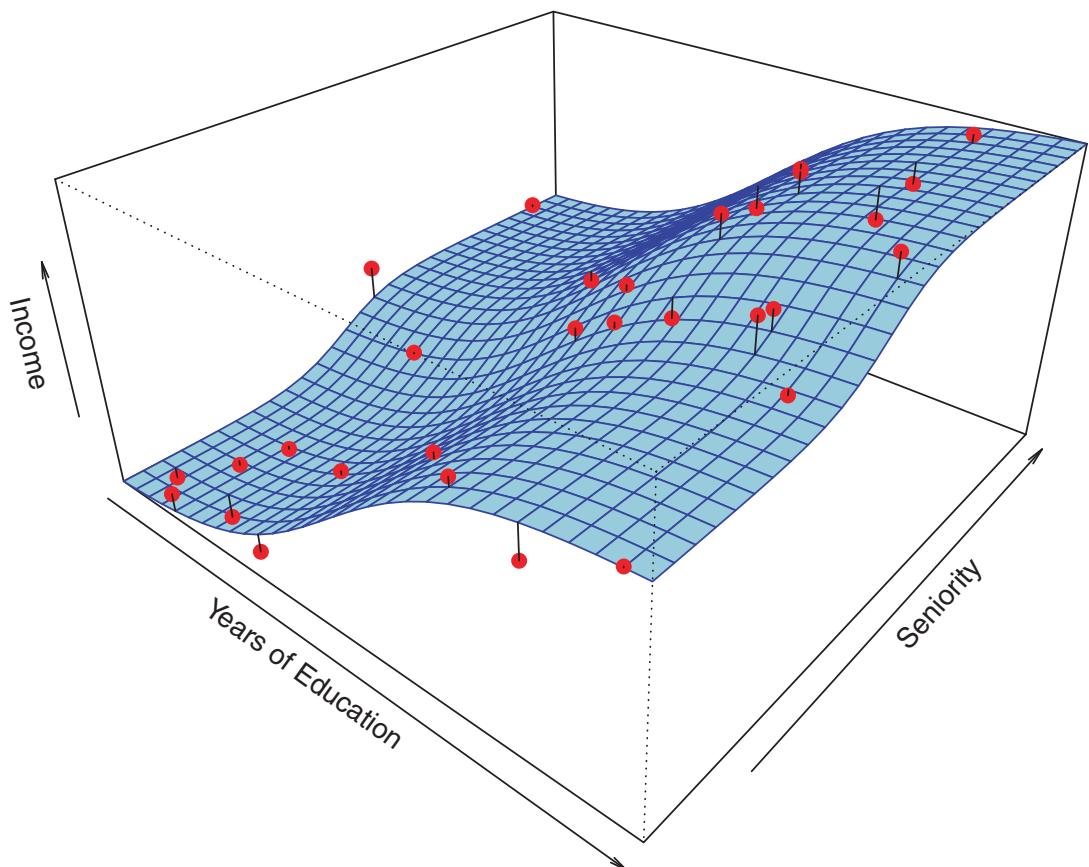
Class	Features
Outcome	Evidence
Response	Predictors
Output Variable	Input Variable
Dependent Variable	Independent Variable

Regression: $f : \mathbb{R}^n \mapsto \mathbb{R}$

Was wären die “Features”? Was wären “Klassen”?

Lineare Regression: Einkommen ist abhängig von ?





[JAMES et al. 2013]

Lineares Regression schätzt Betas (Koeffizienten): β_i
 $\text{income} \approx \beta_0 + \beta_1 \times \text{education} + \beta_2 \times \text{seniority}$

Regressionsmodell mit p Variablen (\approx “Merkmale”)

$$f(X) = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \dots + \beta_p X_p$$

Probabilistische Klassifikation

Probabilistische Methoden berechnen

$$P(\text{CLASS} \mid \text{EVIDENCE})$$

Klassifikation mit Zusatzinfos

- Methode gibt nur die beste Klasse aus
- Methode gibt Konfidenz zur besten Klasse aus
- Methode gibt Wahrscheinlichkeitsdistribution über alle Klassen aus

Für viele Methoden, welche eigentlich nicht probabilistisch sind, können Klassenwahrscheinlichkeiten (*class probabilities*) nachberechnet werden! [WITTEN et al. 2011, 343]

Eigenschaften von Klassifikationsmethoden

- Wie viele Klassen? 2 (*binary*) oder mehr (*multiclass, multinomial classification*)
- Dimensionalität: Umgang mit hoher Anzahl Merkmale (*features*)
- Numerische Merkmale: Möglich, oder ist Diskretisierung (*binning*) erforderlich?
- Kategoriale Merkmale: Möglich, oder ist numerische Kodierung (*coding*) erforderlich?
- Fehlende Merkmale: Können gewisse Merkmale für Items fehlen?
- Merkmalsauswahl: Findet Methode die relevanten Merkmale selbst?
- Abhängige Merkmale: Kann Methode mit untereinander abhängigen Merkmalen umgehen?

Interpretierbarkeit von Modellen

Interpretierbarkeit (interpretability)

Wie gut kann ein Mensch die Entscheidungen eines Klassifikators verstehen/nachvollziehen?

Tradeoff: Leistung (*accuracy*) vs Interpretierbarkeit [JAMES et al. 2013, 24]

- Oft sind die leistungsfähigsten Methoden nur schlecht interpretierbar!
- Gut interpretierbare Methoden sind oft weniger leistungsfähig!
- Entscheidungsbäume/Regellerner vs Neuronale Netzwerke/SVMs
- Aber auch Ensemble-Systeme wie Random Forests!

Output des Regellerners JRIP auf Sentimentklassifikation

JRIP rules:

=====

```
(bad <= 0) and (many >= 1) => class=pos (473.0/139.0)
(bad <= 0) and (life >= 1) and (script <= 0) and (see >= 1)
=> class=pos (97.0/23.0)
(bad <= 0) and (nothing <= 0) and (should <= 0) and (them >= 1) and (plot <= 0)
=> class=pos (80.0/15.0)
(bad <= 0) and (only <= 0) and (performance >= 1) and (her >= 1)
=> class=pos (29.0/4.0)
(worst <= 0) and (best >= 1) and (world >= 1) => class=pos (135.0/48.0)
(worst <= 0) and (have <= 0) and (only <= 0) and (got <= 0)
=> class=pos (90.0/29.0)
(boring <= 0) and (performances >= 1) and (off <= 0) and (bad <= 0)
=> class=pos (37.0/6.0)
(both >= 1) and (plot <= 0) and (between >= 1) => class=pos (56.0/19.0)
(worst <= 0) and (works >= 1) and (together >= 1) => class=pos (29.0/7.0)

=> class=neg (974.0/264.0)
```

Number of Rules : 10

6.2 Naive Bayes

6.2.1 Dokumentklassifikation

Klassifikation von Dokumenten

Bag-of-Word-Modell

- Dokumente als Multimenge (*bag*) Wörtern: Wie oft kommt ein (relevantes) Wort vor?
- Lineare Abfolge wird ignoriert



The bag of words representation

$Y($

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet.

$)=C$



The bag of words representation

$Y($

great	2
love	2
recommend	1
laugh	1
happy	1
...	...

$)=C$



Quelle: Video-Lecture von Dan Jurafsky: <https://www.youtube.com/watch?v=DdYSMwEWbd4>

Naive Bayes Classifier: Probabilistische Formalisierung

Klassifikationsproblem formal

- Bayes'sche Formel: Lerne von gegebenen Klassen und ihrer Evidenz wie man von neuer Evidenz die Klasse abschätzt!
- Umgang mit ungesehenen Wörtern: *Smoothing*

- Video “Formalizing the Naive Bayes Classifier” von SNLP-Kurs <https://www.youtube.com/watch?v=TpjPzKODuXo>

$$\begin{aligned}
 c_{MAP} &= \operatorname{argmax}_{c \in C} P(c | d) && \text{MAP is "maximum a posteriori" = most likely class} \\
 &= \operatorname{argmax}_{c \in C} \frac{P(d | c)P(c)}{P(d)} && \text{Bayes Rule} \\
 &= \operatorname{argmax}_{c \in C} P(d | c)P(c) && \text{Dropping the denominator}
 \end{aligned}$$

Wer ist *argmax*?¹

Funktion: Abbildung von Definitionsbereich (X) auf Wertebereich (Y)
 $f: X \rightarrow Y$

arg max: Argument(e) mit dem maximalen Funktionswert

$$\operatorname{argmax}_{x \in X} f(x) := \{x \in X \mid \forall x_1 \in X : f(x_1) \leq f(x)\}$$

Naive Bayes Classifier III: Lernen von Modellparametern

Bag-Of-Word-Repräsentation Mega-Klassen-Dokumenten

- Pro Klasse ein Mega-Dokument bilden und als Mega-Bag-of-Words repräsentieren
- Wörter mit unterschiedlicher Verteilung pro Klasse sind informativ
- Video “Learning” von SNLP-Kurs <https://www.youtube.com/watch?v=0hxaqDbdIeE>



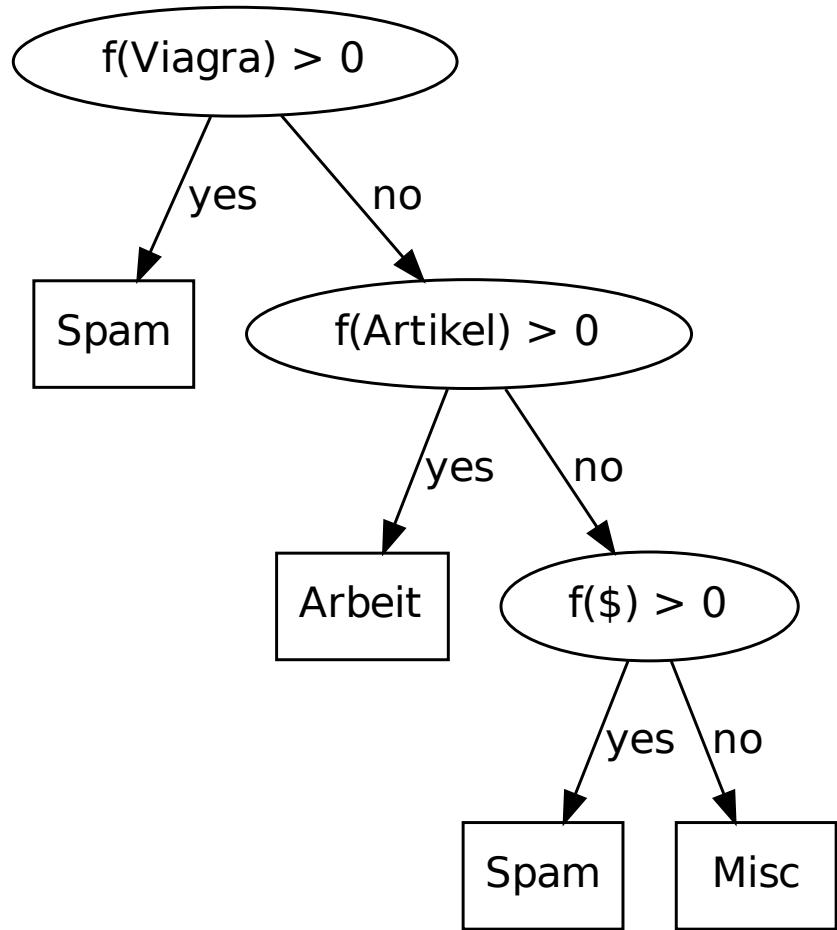
¹https://de.wikipedia.org/wiki/Arg_max



Informativ negativ
Informativ positiv

6.3 Decision Trees

Textklassifizierung mit Entscheidungsbäumen



```

worst <= 0
...
worst > 0
|   fiction <= 0
|   |   today <= 0
|   |   |   whom <= 0
|   |   |   |   similar <= 0
|   |   |   |   |   david <= 0
...
|   |   |   |   |   david > 0
|   |   |   |   |   begin <= 0
|   |   |   |   |   |   actress <= 0: neg (14.0/1.0)
|   |   |   |   |   |   actress > 0: pos (2.0)
|   |   |   |   |   begin > 0: pos (3.0)
|   |   |   |   similar > 0
|   |   |   |   |   although <= 0: neg (8.0/1.0)
|   |   |   |   |   although > 0: pos (4.0)
|   |   |   whom > 0
|   |   |   |   being <= 0
|   |   |   |   |   bunch <= 0: neg (11.0/1.0)
|   |   |   |   |   bunch > 0: pos (2.0)
|   |   |   |   being > 0: pos (4.0)
|   |   |   today > 0
|   |   |   |   3 <= 0
|   |   |   |   |   attention <= 0: pos (7.0)
|   |   |   |   |   attention > 0: neg (2.0)
|   |   |   |   3 > 0: neg (5.0)
|   |   fiction > 0
|   |   |   almost <= 0: pos (11.0/1.0)
|   |   |   almost > 0: neg (4.0)
Number of Leaves : 206

Size of the tree : 411

```

Entscheidungsfragen

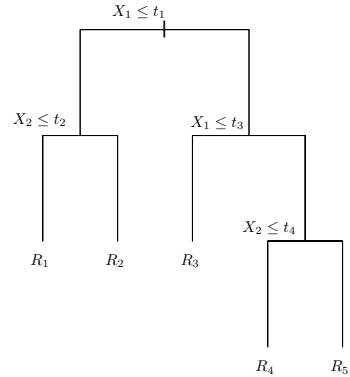
Ziele beim Fragenstellen

- Mit möglichst wenig Fragen
- eine möglichst gute Klassifikation herstellen

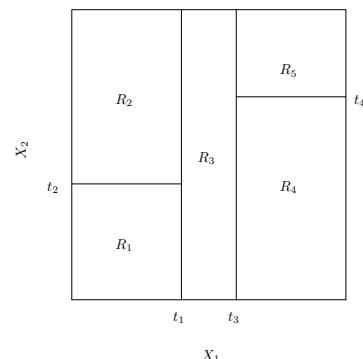
Kriterien

- Arität der Fragen: binäre vs. n-äre Fragen zu kategorialen Merkmalen
- Umgang mit numerischen Attributen: Vergleichsoperatoren, Intervalle
- Umgang mit fehlenden Werten (unbeantwortbare Fragen)

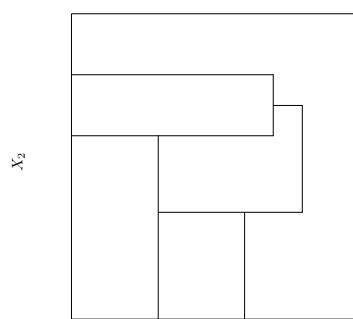
Grundidee: Rekursives Binäres Aufteilen von Merkmalen



Entscheidungsbaum

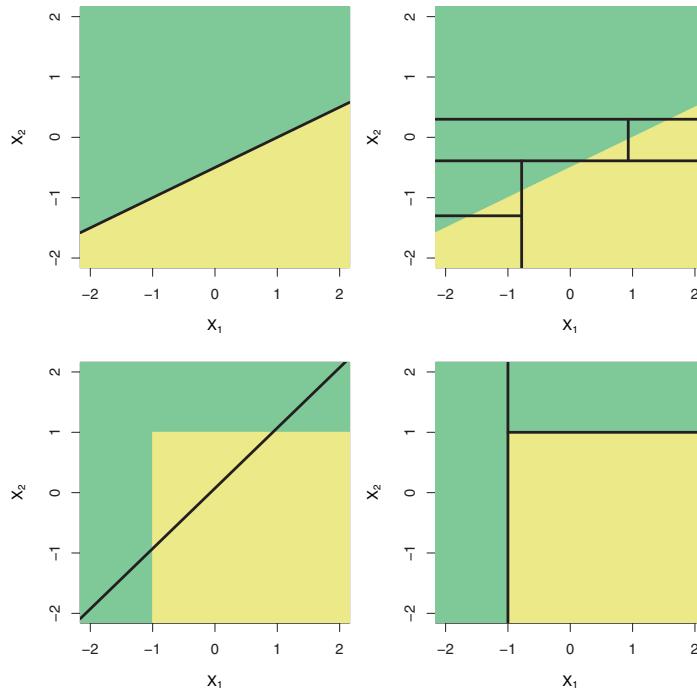


Resultierende Aufteilung des Merkmalsraums



Aufteilung, welche nicht mit rekursivem Split machbar ist

Lineare Klassifikation vs Entscheidungsbäume



- Hängt von der (nicht-)linearen Natur des Problems ab
- Entscheidungsbäume können gewisse nicht-lineare Probleme besser lösen
- Was ist die geometrische Intuition bei Entscheidungsbäumen?

6.3.1 Training

Trainieren von Entscheidungsbäumen

Ansatz

- Initialisierung: Alle Items und Klassen in einen Knoten
- Rekursives Aufteilen der Knoten bis jeder Blattknoten nur noch Items einer einzigen Klasse enthält (oder möglichst extreme Klassenverteilung)

Zutaten

- Menge der möglichen Fragen \mathbb{Q}
- Mass zum Bewerten einer Frage $q \in \mathbb{Q}$ für einen Knoten
- Funktion f zum Wählen der besten Frage $q \in \mathbb{Q}$ für einen Knoten

Beispiel: Knotenaufteilung

Aufteilen

- Gegeben: mehr als eine Klasse pro Blattknoten
- Gesucht: Mass für die Bewertung der Qualität eines Knotensplits

Ein Beispielknoten N

Klasse	<i>freq</i>
Spam	30
News	20
Arbeit	40
Bugreports	10

Welche Frage ist besser?

$q_1 :$

N_L	
Klasse	<i>freq</i>
Spam	30
News	5

N_R	
Klasse	<i>freq</i>
News	15
Arbeit	40
Bugreports	10

$q_2 :$

N_L	
Klasse	<i>freq</i>
Spam	15
News	20
Arbeit	20
Bugreports	5

N_R	
Klasse	<i>freq</i>
Spam	15
Arbeit	20
Bugreports	5

Antwort: Diejenige, welche Knoten von grösserer Reinheit (*purity*) produziert

Impurity: Unreinheitsmass

$i(N) = \text{Impurity des Knotens } N$

- $i(N) = 0$, wenn in N nur eine Klasse zur Verfügung steht
- $i(N) = \text{maximal}$, wenn alle Klassen in N gleich wahrscheinlich sind

Entropy als Impurity

$$i(N) = - \sum_{c \in C} P_N(c) \log P_N(c)$$

$P_N(c)$ Relative Häufigkeit von Klasse c in N

Andere Masse sind möglich (*Gini index*)

Impurity-Verbesserung

Impurity-Verbesserung

$$\Delta i(N) = i(N) - (P_L)i(N_L) - (1 - P_L)i(N_R)$$

P_L Anteil der Elemente im neuen linken Knoten

$N_{L,R}$ Neuer linker/rechter Knoten

	Klasse	<i>freq</i>
$N =$	Spam	30
	News	20
	Arbeit	40
	Bugreports	10

	Klasse	<i>freq</i>
$N_L =$	Spam	30
	News	5

	Klasse	<i>freq</i>
$N_R =$	News	15
	Arbeit	40
	Bugreports	10

And the winner is ...

$q_1 :$

	N_L	
	Klasse	<i>freq</i>
	Spam	30
	News	5

	N_R	
	Klasse	<i>freq</i>
	News	15
	Arbeit	40
	Bugreports	10

	$\Delta i(N)$	
	0.77	

$q_2 :$

	N_L	
	Klasse	<i>freq</i>
	Spam	15
	News	20
	Arbeit	20
	Bugreports	5

N_R	
Klasse	<i>freq</i>
Spam	15
Arbeit	20
Bugreports	5

$\Delta i(N)$
0.17

Der Trainingsalgorithmus

Initialisierung

Starte mit einem einzigen Knoten, der alle Items enthält.

Iteration

1. Suche für jeden unreinen Blatt-Knoten N die beste Frage q
2. Splitte N entsprechend und hänge N_L und N_R als Blätter an N
3. Wende Schritt 1 auf alle neuen Blatt-Knoten an

Stopp-Kriterium

Keine unreinen Knoten mehr!

Stop-Kriterien

Eine Klasse pro Blattknoten

- ✗ overfit
- ✗ sehr grosser Baum

Fehlklassifizierungen unter $x\%$

- ✗ nicht alle Daten stehen zum Training zur Verfügung

Abbruch bei zu wenig Impurity-Verbesserung

- ✓ Äste können unterschiedlich gross sein
- ✗ Parameter ist schwierig zu wählen

6.3.2 Pruning

Pruning: Verkleinern von Bäumen

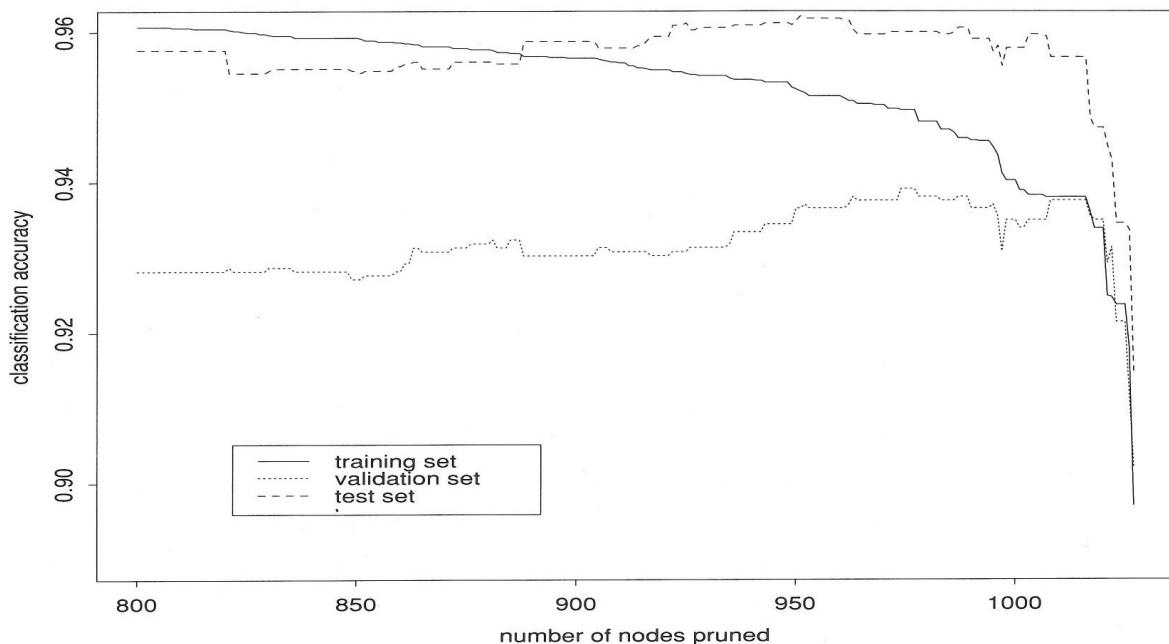
Baumgrössen

- Entscheidungsbäume können sehr gross werden
- *Overfit* extrem wahrscheinlich; Entscheidungsbaum lernt Trainingsdaten auswendig
- Zufällige Vorkommnisse in den Trainingsdaten werden im Classifier modelliert und
- führen zu falschen Ergebnissen auf den Testdaten.
- Hohe Varianz: leichte Änderungen im Trainingsset führen zu unterschiedlichen Modellen

Lösung

- Entfernen/Zusammenlegen von Blattknoten
- folgt auf Erstellen des maximal grossen Baumes
- benutzt z.B. Information gain eines Splits

Effekte von Pruning nach [MANNING und SCHÜTZE 1999, 585]



Fazit

- Modelle sind einfach zu verstehen und interpretieren
- Bäume können graphisch dargestellt werden
- Erlauben alle Skalen
- Weniger gute Klassifikatoren als andere Methoden
- Anfällig für Overfit und Instabilität

6.3.3 Bagging

Bootstrap Aggregation

- Idee: Erstelle viele Entscheidungsbäume aus den Trainingsdaten statt nur 1!
- Aggregierung: Verwende die Mehrheitsentscheidung der Bäume für Klassifikation
- Vorteil: Weniger Instabilität insgesamt, d.h. höhere Varianz!
- Je weniger Pruning, umso höhere Varianz!
- Vorteil: Fast immer bessere Resultate insgesamt ([WITTEN et al. 2011, 353])
- Frage: Wie kann man aus den Trainingsdaten unterschiedliche Bäume herstellen?

Bootstrapping

Bootstrap (Re-)Sampling

- Create n (typically $1000 \geq n \geq 2000$) *resamples* of your original data
- Each *resample* of the original data
 - has the *same number of items* as the original data
 - may contain the same item *several times or not at all* (sampling with replacement)
- The re-samples build a *bootstrap distribution*.

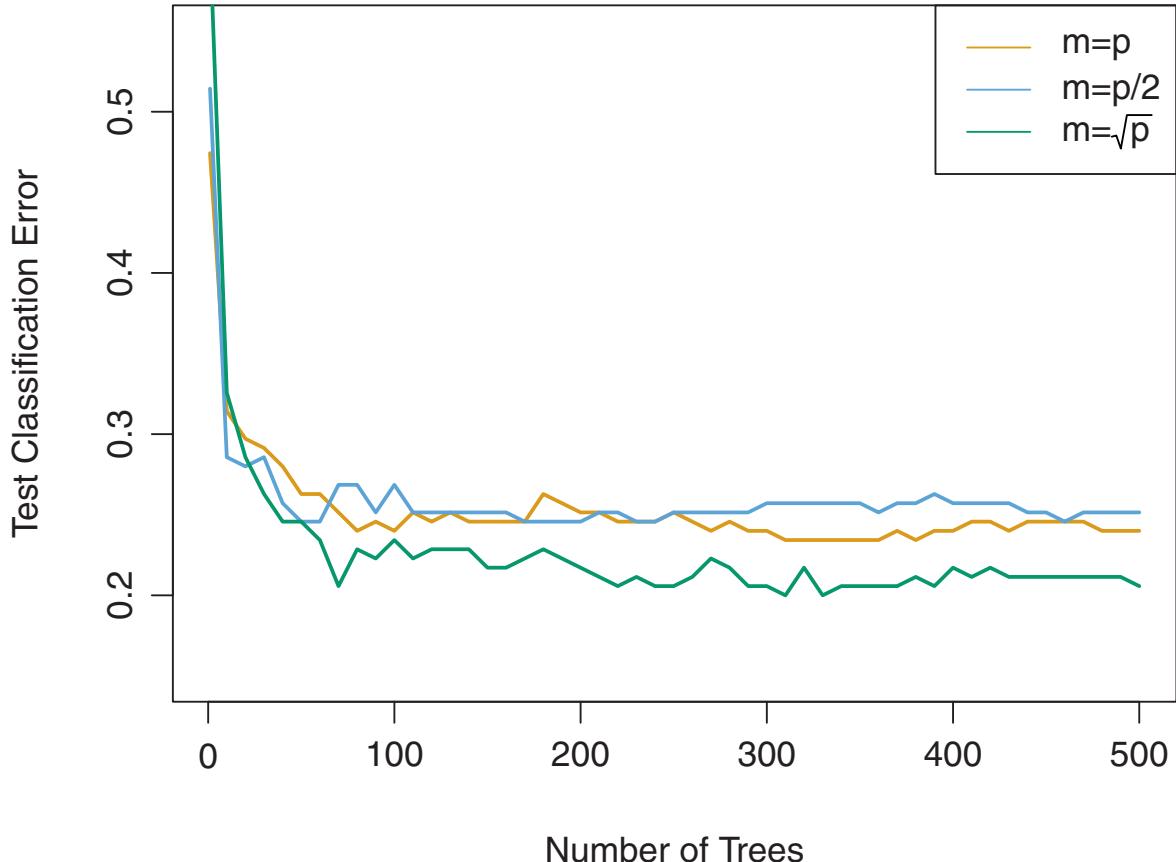
Bootstrapping für Trainingsmaterial = Zufälliges Ziehen mit Zurücklegen aus dem Trainingsmaterial

6.3.4 Random Forest

Random Forests

- Ensemble-Ansatz, welcher auf *Bootstrap Aggregation* aufbaut
- Entscheidungsbäume werden entkorreliert, indem jedem Baum nur ein zufällig ausgewählter eingeschränkter Vorrat an Merkmalen als “Nahrung” gegeben wird.
- Typische Merkmalsmengengröße für einzelne Bäume, wenn insgesamt p Merkmale zur Verfügung stehen: $m = \sqrt{p}$
- Random Forests entspricht Bagging, wenn $m = p$.
- Ergeben gute Leistung auf Kosten der Interpretierbarkeit.
- Aber die Relevanz von Merkmalen (*variable importance*), kann wieder berechnet werden über allen Bäumen.

Decision Tree vs Bagging vs Random Forest



Quelle:[JAMES et al. 2013, 322]

Datenset: 500 Prediktoren ($p=500$) mit 15 Klassen (*Gene Expression Data*)

- Einzelbaum: 47.5% Fehler
- Random Forests mit $m = \sqrt{p}$ können Fehler mehr als halbieren.
- Die Grösse des zufälligen Merkmalssubsets ist relevant für die Leistung.

6.3.5 Boosting

Boosting: Sequentielles Meta-Lernen

Grundidee: Boosting

- Bagging und Random Forests: paralleler Erstellung von t Klassifikationsinstanzen
- Boosting: Instanzen werden sequentiell hintereinandergeschaltet trainiert
- Grundidee: Die $N + 1$ -te Instanz kümmert sich spezifisch um Probleme aus den N vorgängigen Instanzen
- Problemfälle werden in nachfolgenden Instanzen höher gewichtet als korrekte Fälle
- Gewichtung kann erreicht werden, indem falsche Instanzen im Trainingsset dupliziert werden (*resampling*)

AdaBoost.M1-Algorithmus [WITTEN et al. 2011, 359]

Model Generation

```
Assign equal weight to each training instance.  
For each of t iterations:  
    Apply learning algorithm to weighted dataset and store resulting  
    model.  
    Compute error e of model on weighted dataset and store error.  
    If e equal to zero, or e greater or equal to 0.5:  
        Terminate model generation.  
    For each instance in dataset:  
        If instance classified correctly by model:  
            Multiply weight of instance by e / (1 - e).  
    Normalize weight of all instances.
```

Classification

```
Assign weight of zero to all classes.  
For each of the t (or less) models:  
    Add -log(e / (1 - e)) to weight of class predicted by model.  
Return class with highest weight.
```

Klassifizieren unter Boosting

- Grundsätzlich eine gewichtete Abstimmung (*weighted vote*)
- Das Gewicht w hängt von der Fehlerrate e der Klassifikationsinstanz ab.

$$w = -\log \frac{e}{1 - e}$$

- Warum sind Fehlerraten von 0 unerwünscht?
- Warum Fehlerraten $e > 0.5$?

6.3.6 Zusammenfassung

Zusammenfassung

- Entscheidungsbäume sind einfache, interpretierbare und effizient berechenbare nicht-lineare Klassifikatoren
- Einzelne Entscheidungsbäume sind oft schlechter als andere Klassifikationsverfahren und sind tendenziell instabil
- Ensemble- oder Meta-Lernverfahren wie Bagging, Randomisierung oder Boosting können gerade bei einfachen (*weak learner*) Lernverfahren wie Entscheidungsbäumen zu sehr guten Resultaten führen

Vertiefung

- Pflichtlektüre: Kapitel “Tree-Based Methods in [JAMES et al. 2013] ohne Regression Trees
- Eine gute Einführung ins Thema Bagging und Boosting im Kapitel 8 in [WITTEN et al. 2011]

Kapitel 7

Parametrische Modelle, MLE und MAP

Lernziele

- Parametrische vs nicht-parametrische Modelle verstehen
- Distributionen als parametrisierbare Modelle sehen
- Zufallsvariablen zu Distributionen zuordnen
- Verschiedene NB-Varianten mit entsprechenden Zufallsvariablen unterscheiden
- *Maximum Likelihood Estimation* (MLE) für Fixierung von Parametern von Distributionen/Zufallsvariablen
- *Maximum Aposteriori Estimation* (MAP) verstehen
- Die Rolle von Conjugate Priors beim Zusammenhang von MLE und MAP von Bernoulli-Zufallsvariablen verstehen

7.1 Parametrische Modelle

Parametric vs Non-parametric Modelling

Parametrisch

[MURPHY 2012, 16]

- Hat das Modell eine feste Anzahl von Parametern?
- Vorteil: Einfacher zu verwenden (Parameterisierung einer vordefinierten Modellklasse)
- Nachteil: Modellklasse bringt (allenfalls falsche) Annahmen über die Daten ins Modell

Nicht-Parametrisch

- Wächst die Anzahl Parameter des Modells mit wachsender Trainingsmenge?
- Vorteil: Nur die Daten zählen, d.h. flexibles Modell
- Nachteil: Bei hoher Dimensionalität reichen Daten nicht aus für gute Schätzung
- Nachteil: Berechnungsaufwand bei grossen Datensets

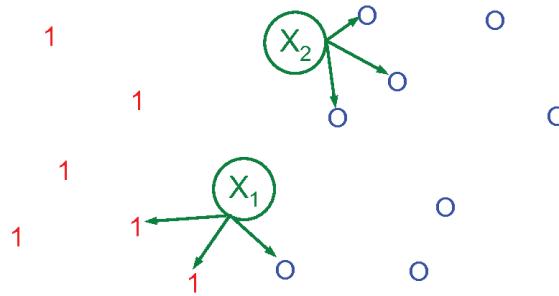
7.1.1 Nichtparametrisch

Beispiel: Nicht-Parametrische Modellierung: KNN

K-nearest Neighbors (KNN) Klassifikation

- Idee: Wähle für die Evidenz \mathbf{x} die Klasse c aus, welche bei den k ähnlichsten Trainingsbeispielen (=nearest neighbors) am häufigsten ist.
- Sogenanntes *memory-based learning* (auch *instance-based learning*)

KNN-Klassifikation mit $K = 3$



[MURPHY 2012, 16]

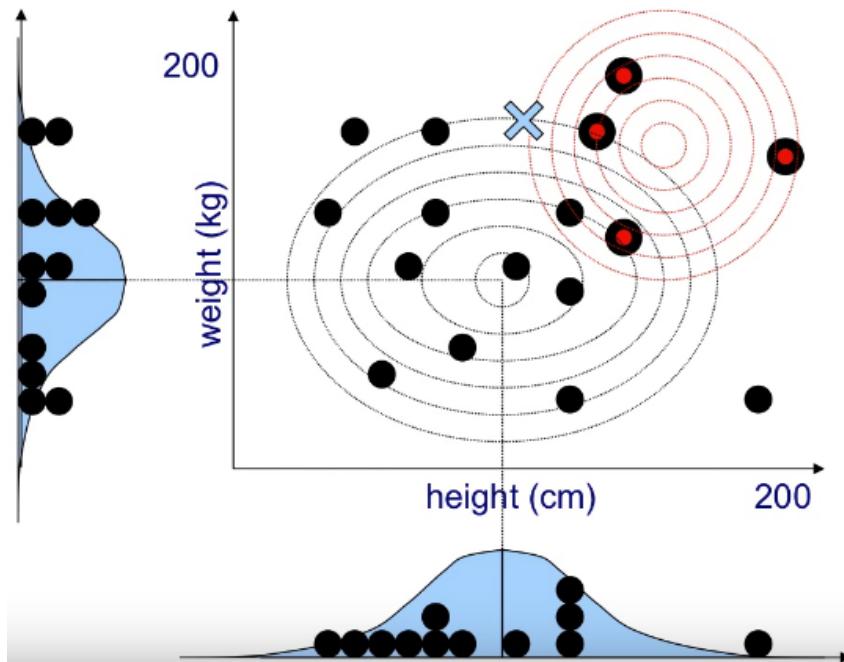
7.1.2 Parametrisch

Beispiel: Parametrische Modellierung: Gaussian NB

Gaussian Naive Bayes

- Problem: Wie kann die Wahrscheinlichkeit von reellen Prädiktoren bei kleinen Trainingsmengen abgeschätzt werden?
- Idee: Annahme einer Normalverteilung (*Gaussian Distribution*) mit 2 Parametern: Mittelwert μ und Varianz σ^2

Klassifikation mit Gaussian NB



Victor Lavrenko

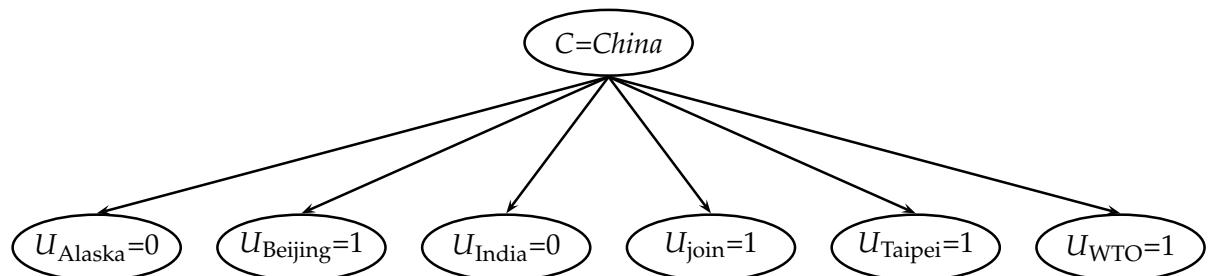
- Details dazu im Video von Victor Lavrenko: <https://www.youtube.com/watch?v=TcAQKPgymLE>
- sklearn-Implementation[†]

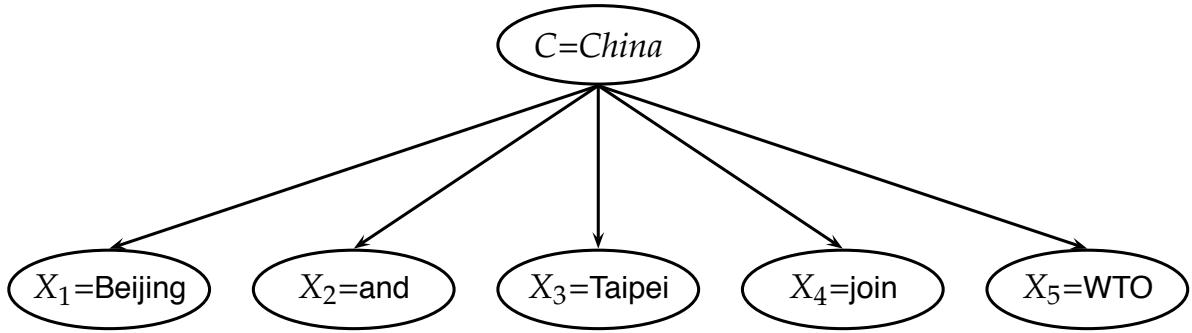
Parameter, Zufallsvariablen und Distributionen: NB

$$p(c | [X_1 = x_1, X_2 = x_2, \dots, X_n = x_n]) = \frac{p(c) \prod_i p(X_i = x_i | c)}{\sum_{c'} p(c') \prod_i p(X_i = x_i | c')}$$

Zusammenhang von Distributionen und Zufallsvariablen [MURPHY 2012, 64]

- Gaussian NB (multivariate Gaussian NB): jede X_i ist normalverteilt: $X_i \sim \mathcal{N}(\mu, \sigma^2)$
- Bernoulli NB (multivariate/multinoulli NB): jede X_i ist bernoulliverteilt: $X_i \sim Ber(\mu)$
- Multinomial NB: jedes X_i ist ein Outcome derselben multinomial verteilten Zufallsvariablen X : $\mathbf{x} = [x_1, x_2, \dots, x_n]$ $X \sim Mu(\boldsymbol{\alpha} | n, \boldsymbol{\theta})$





Welche Modell ist Multinomial NB?

[METSIS et al. 2006, SCHNEIDER 2003]

Namen von Verteilungen

Verteilung von einzelnen Zufallsvariablen und Folgen von Zufallsvariablen

Name	n	K	x
Multinomial	-	-	$\mathbf{x} \in \{0, 1, \dots, n\}^K, \sum_{k=1}^K x_k = n$
Multinoulli	1	-	$\mathbf{x} \in \{0, 1\}^K, \sum_{k=1}^K x_k = 1$ (l-of- K encoding)
Binomial	-	1	$x \in \{0, 1, \dots, n\}$
Bernoulli	1	1	$x \in \{0, 1\}$

Table 2.1 Summary of the multinomial and related distributions.

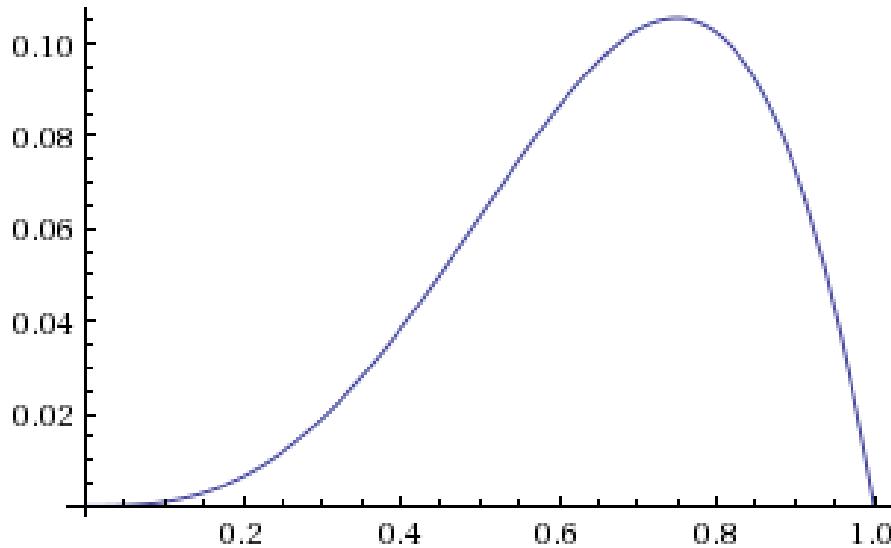
Quelle: [MURPHY 2012, 36]

7.2 MLE revisited

7.2.1 Analytisch

MLE von Bernoulli-Variablen: Parameter μ maximieren

- Gegeben: “Trainingsdaten” einer Münze $C \sim Ber(\mu)$: HHTH
- Gesucht: Parameter μ , welcher die Wahrscheinlichkeit HHTH zu erhalten maximiert (*Maximum Likelihood Estimation*)
- Notation: Kopf: $p(C = H) = \beta$, Zahl: $p(C = T) = 1 - \beta$
- Parametrische Wahrscheinlichkeit der Trainingsdaten: $\beta\beta(1 - \beta)\beta = \beta^3 - \beta^4$
- Likelihood-Funktion für gegebene Daten: $f(\beta) = p(HHTH; \beta) = \beta^3 - \beta^4$



Plot von $f(\beta)$ für $0 \leq \beta \leq 1$

- An welcher Stelle für β ergibt sich die höchste Wahrscheinlichkeit für die Trainingsdaten?
- Beim Maximum von f , d.h. $f' = 0$

Ableiten der MLE von β [DAUMÉ III 2015, 105]

$$\frac{\partial}{\partial \beta} [\beta^3 - \beta^4] = 3\beta^2 - 4\beta^3$$

$$4\beta^3 = 3\beta^2$$

$$\iff 4\beta = 3$$

$$\iff \beta = \frac{3}{4}$$

D.h. Parameter $\mu = \frac{3}{4}$ maximiert Trainingsdaten HHTH für Zufallsvariable C : MLE-Parametrisierung:
 $C \sim Ber\left(\frac{3}{4}\right)$

Verallgemeinerung auf H Köpfe und T Zahlen im Log-Raum

- Trainingsdaten: $\mu^H(1 - \mu)^T$
- im Log-Raum: $H \log \mu + T \log(1 - \mu)$
- Nullstelle der Ableitung: $\frac{H}{\mu} - \frac{T}{(1-\mu)} = 0$
- Nach Umformung:

$$\mu = \frac{H}{(H + T)}$$

MLE von einer normalverteilten Variable

- **Problem:** find most likely Gaussian distribution, given sequence of real-valued observations

3.18, 2.35, .95, 1.175, ...

- **Normal distribution:** $p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$.
- **Likelihood:** $l(p) = -\frac{1}{2}m \log(2\pi\sigma^2) - \sum_{i=1}^m \frac{(x_i - \mu)^2}{2\sigma^2}$.

- **Solution:** l is differentiable and concave;

$$\frac{\partial p(x)}{\partial \mu} = 0 \Leftrightarrow \mu = \frac{1}{m} \sum_{i=1}^m x_i \quad \frac{\partial p(x)}{\partial \sigma^2} = 0 \Leftrightarrow \sigma^2 = \frac{1}{m} \sum_{i=1}^m x_i^2 - \mu^2.$$

Quelle: M. Mohri www.cs.nyu.edu/~mohri/mlu/mlu_lecture_3.pdf

7.2.2 Iterativ

Iteratives Bestimmen des maximalen Arguments einer kontinuierlichen konkaven Funktion: *Gradient Ascent*

Algorithmus $\arg \max_x f(x)$ für einstellige Funktionen

- Parameter
 - Funktion $f : \mathbb{R} \rightarrow \mathbb{R}$
 - Schrittgrösse $a > 0$ (falls a zu gross ist, divergiert der Algorithmus)
- Update-Regel für Argument x:
 - 1: **repeat**
 - 2: $x \leftarrow x + a * f'(x)$
 - 3: **until** “x has converged”
- f' ist die Ableitung von f

Gradient Ascent für $f : \mathbb{R} \rightarrow \mathbb{R}$ in Python

```
def baby_gradient_ascent(f, a, init=random.random(), eps=0.001, maxi=100):
    """ Return arg max of function f: float -> float

    f is a sympy expression with x as symbolic parameter.
    a is the step size. init defaults to a random number [0,1].
    eps is the convergence criterion. maxi is the maximum number of
    iterations."""
    f_deriv = diff(f, x)
    argmax = init
    converged = False
    iteration = 0
    while not converged and iteration < maxi:
        iteration += 1
        oldargmax = argmax
```

```

argmax += a*f_deriv.subs(x, argmax)
print(iteration,'before', oldargmax, 'after',argmax, file=sys.stderr)
if abs(oldargmax - argmax) < eps:
    converged = True
return argmax

```

Analytische (*closed form*) vs iterative Maximierung

- Die analytische Maximierung, d.h. Ableitung der Likelihood-Funktion gleich Null setzen ($f' = 0$), ist nicht für alle Likelihood-Funktionen möglich.
- Verschiedene iterative Verfahren (Gradienten-Verfahren¹) erlauben die Maximierung der Likelihood-Funktion, wenn keine analytische Form der Maximierung angegeben werden kann.
- Die iterativen Verfahren funktionieren natürlich nicht nur für die trivialen einstelligen Likelihood-Funktionen wie $f : \mathbb{R} \rightarrow \mathbb{R}$, sondern für n-stellige Funktionen $f : \mathbb{R}^n \rightarrow \mathbb{R}$.
- Wichtig: Nur wenn genau ein einziges globales Maximum existiert, finden wir es mit diesem Verfahren. Ansonsten landen wir beim nächsten durch steilsten Anstieg erreichbaren lokalen Maximum.

7.3 MAP

Problem bei der MLE

Wie fest sollen wir nur den Trainingsdaten vertrauen?

- HTTT $\rightarrow \mu_{MLE} = 0.25 \rightarrow C_{MLE} \sim Ber(0.25)$
- THTH $\rightarrow C_{MLE} \sim Ber(0.5)$
- TTTT $\rightarrow C_{MLE} \sim Ber(0.0)$

MLE bezieht kein Vorwissen (*prior knowledge*) ein und kann die Unsicherheit der Schätzung nicht einbeziehen.

Maximum a Posteriori Estimation I

¹<https://de.wikipedia.org/wiki/Gradientenverfahren>

Bayesian approaches try to reflect our belief about μ . In this case, we will consider μ to be a random variable.

$$p(\mu|\mathbf{X}) = \frac{p(\mathbf{X}|\mu)p(\mu)}{p(\mathbf{X})} \quad (9)$$

Thus, Bayes' law converts our prior belief about the parameter μ (before seeing data) into a posterior probability, $p(\mu|\mathbf{X})$, by using the likelihood function $p(\mathbf{X}|\mu)$. The maximum a-posteriori (MAP) estimate is defined as

$$\hat{\mu}_{MAP} = \operatorname{argmax}_{\mu} p(\mu|\mathbf{X}) \quad (10)$$

Maximum a Posteriori Estimation II

Note that because $p(\mathbf{X})$ does not depend on μ , we have

$$\begin{aligned}\hat{\mu}_{MAP} &= \operatorname{argmax}_{\mu} p(\mu|\mathbf{X}) \\ &= \operatorname{argmax}_{\mu} \frac{p(\mathbf{X}|\mu)p(\mu)}{p(\mathbf{X})} \\ &= \operatorname{argmax}_{\mu} p(\mathbf{X}|\mu)p(\mu)\end{aligned}$$

Was hilft MAP? I

To take a simple example of a situation in which MAP estimation might produce better results than ML estimation, let us consider a statistician who wants to predict the outcome of the next election in the USA.

- The statistician is able to gather data on party preferences by asking people he meets at the Wall Street Golf Club¹ which party they plan on voting for in the next election
- The statistician asks 100 people, seven of whom answer “Democrats”. This can be modeled as a series of Bernoullis, just like the coin tosses.
- In this case, the maximum likelihood estimate of the proportion of voters in the USA who will vote democratic is $\hat{\mu}_{ML} = 0.07$.

Was hilft MAP? II

Somehow, the estimate of $\hat{\mu}_{ML} = 0.07$ doesn't seem quite right given our previous experience that about half of the electorate votes democratic, and half votes republican. But how should the statistician incorporate this prior knowledge into his prediction for the next election?

The MAP estimation procedure allows us to inject our prior beliefs about parameter values into the new estimate.

Beta-Distribution

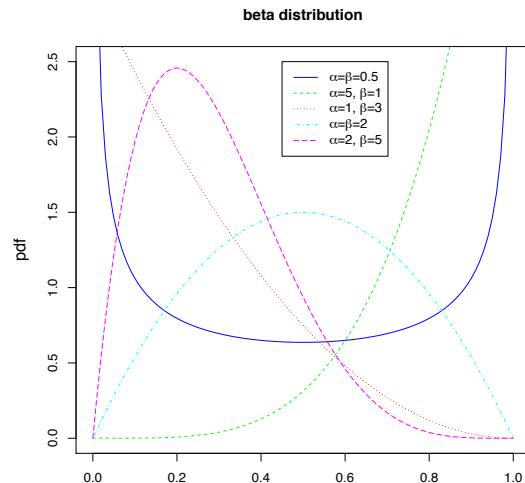
The Beta distribution is appropriate to express prior belief about a Bernoulli distribution. The Beta distribution is a family of continuous distributions defined on $[0, 1]$ and parametrized by two positive shape parameters, α and β

$$p(\mu) = \frac{1}{B(\alpha, \beta)} \cdot \mu^{\alpha-1} (1 - \mu)^{\beta-1}$$

here, $\mu \in [0, 1]$, and

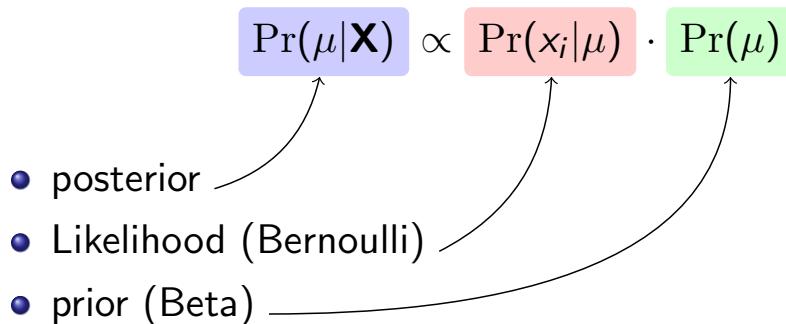
$$B(\alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha) \cdot \Gamma(\beta)}$$

where Γ is the Gamma function (extension of factorial).



Beta-Distribution: Conjugate Prior für Bernoulli

Let's go back now to our problem of predicting the results of the next election. Essentially, we plug in the equations for the distributions of the likelihood (a Bernoulli distribution) and the prior (A Beta distribution).



A-Posteriori-Schätzung formal

We thus have that

- $\Pr(x_i|\mu) = \text{Bernoulli}(x_i|\mu) = \mu^{x_i}(1-\mu)^{1-x_i}$
- $\Pr(\mu) = \text{Beta}(\mu|\alpha, \beta) = \frac{1}{B(\alpha, \beta)} \cdot \mu^{\alpha-1} (1-\mu)^{\beta-1}$

thus

$$\Pr(\mu|\mathbf{X}) \propto \Pr(\mathbf{X}|\mu)\Pr(\mu)$$

is equivalent to

$$\Pr(\mu|\mathbf{X}) \propto \left\{ \prod_i \text{Bernoulli}(x_i|\mu) \right\} \cdot \text{Beta}(\mu|\alpha, \beta) \quad (15)$$

Log-Likelihood-Schätzung formal

$$\begin{aligned} \mathcal{L} &= \log \Pr(\mu|\mathbf{X}) \\ &= \log \left\{ \prod_i \text{Bernoulli}(x_i|\mu) \right\} \cdot \text{Beta}(\mu|\alpha, \beta) \\ &= \sum_i \log \text{Bernoulli}(x_i|\mu) + \log \text{Beta}(\mu|\alpha, \beta) \end{aligned}$$

We solve for $\hat{\mu}_{MAP} = \operatorname{argmax}_\mu \mathcal{L}$ as follows

$$\operatorname{argmax}_\mu \sum_i \log \text{Bernoulli}(x_i|\mu) + \log \text{Beta}(\mu|\alpha, \beta)$$

Note that this is almost the same as the ML estimate except that we now have an additional term resulting from the prior

Analytische Lösung nach viel Ableitung

Finally, if we let our Bernoulli distribution be coded as Republican=1 and Democrat=0, we have that

$$\sum_i x_i = n_r \quad \text{where } n_r \text{ denotes the number of Republican voters}$$

(17)

Then,

$$\begin{aligned} \mu \left[\sum_{i=1}^n 1 + \beta + \alpha - 2 \right] &= \sum_i x_i + \alpha - 1 \\ \mu [n + \beta + \alpha - 2] &= n_R + \alpha - 1 \end{aligned}$$

and finally

$$\hat{\mu}_{MAP} = \frac{n_R + \alpha - 1}{n + \beta + \alpha - 2} \quad (18)$$

MAP vs MLE

It is useful to compare the ML and the MAP predictions. Note again that α and β are essentially the same thing as pseudo-counts, and the higher their value, the more the prior affects the final prediction (i.e., the posterior).

Recall that in our poll of 100 members of the Wall Street Golf club, only seven said they would vote democratic. Thus

- $n = 100$
- $n_r = 93$
- We will assume that the mode of our prior belief is that 50% of the voters will vote democratic, and 50% republican. Thus, $\alpha = \beta$. However, different values for alpha and beta express different strengths of prior belief

MAP vs MLE II: Einfluss von α und β

n	n_R	α	β	$\hat{\mu}_{ML}$	$\hat{\mu}_{MAP}$
100	93	1	1	0.93	0.93
100	93	5	5	0.93	0.90
100	93	100	100	0.93	0.64
100	93	1000	1000	0.93	0.52
100	93	10000	10000	0.93	0.502

Thus, MAP “pulls” the estimate towards the prior to an extent that depends on the strength of the prior

Frequentistische Interpretation von α und β als “fictitious Counts”

- α Wir haben $\alpha - 1$ Mal den Wert 1 gesehen.
- β Wir haben $\beta - 1$ Mal den Wert 0 gesehen.
- Add-one-Smoothing: $\alpha = 2$ und $\beta = 2$

Vertiefung

- Pflichtlektüre: Saubere klassische parametrische Modellierung und empirische Evaluation zu verschiedenen Naive Bayes Varianten zu Text-Klassifikation [MC CALLUM und NIGAM 1998]
- Beispielkode von Peter Makarov zu MLE und MAP <http://kitt.cl.uzh.ch/kitt/polcon/MLE.html> Jupyter Notebook https://gitlab.cl.uzh.ch/makarov/Bernoulli_MLE
- Distributionen: Wikipedia-Seiten oder [MURPHY 2012]
- KNN in Kapitel 16.4 in [MANNING und SCHÜTZE 1999]
- Details der Ableitung von MAP für Bernoulli-Variablen: [ROBINSON 2012]

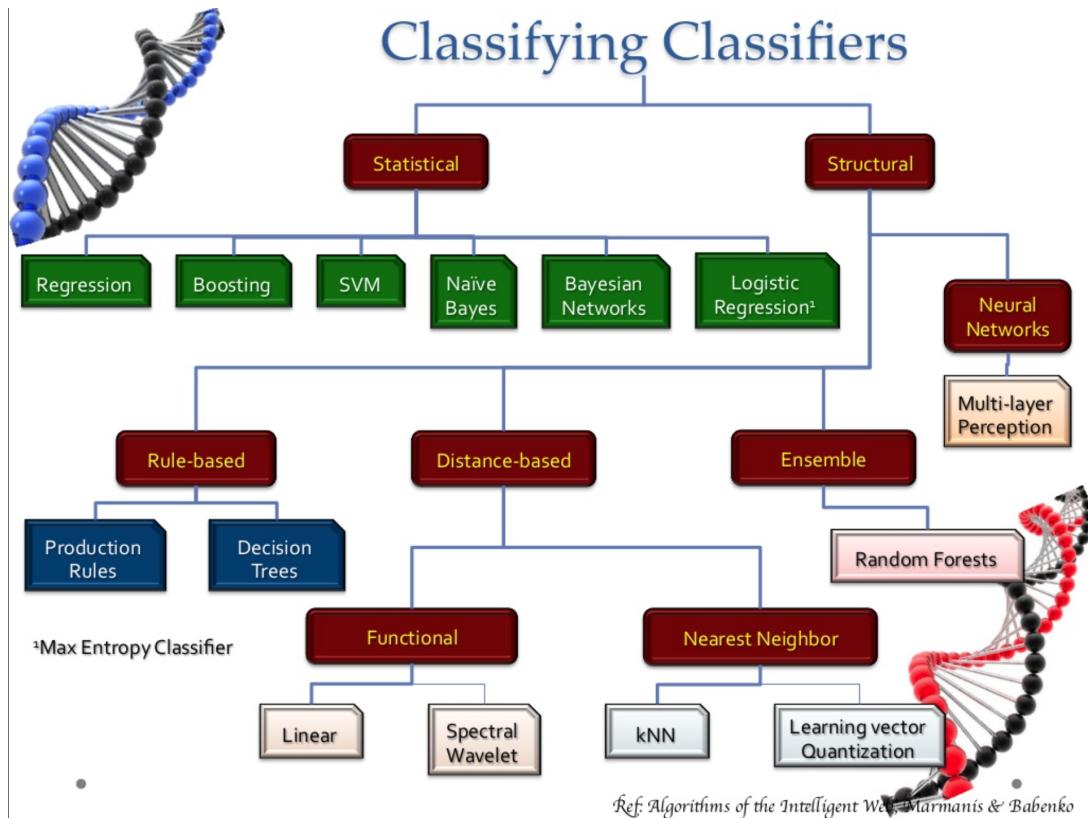
Kapitel 8

Logistic Regression und Maximum Entropy

Herzlichen Dank an Torsten Marek und Rico Sennrich für Quelltexte. Some of the figures in this presentation are taken from “An Introduction to Statistical Learning, with applications in R” (Springer, 2013) with permission from the authors: G. James, D. Witten, T. Hastie and R. Tibshirani

Lernziele

- Logistic Regression verstehen
- Zusammenhang von Logistic Regression und Maximum Entropy verstehen
- Grundlagen von Maximaler Entropie Klassifikation verstehen
- Kodierung von Trainingsdaten in Maximum-Entropie-Merkmale
- Bedingte Exponentielle Modelle mit ME-Merkmalen verstehen
- Unterschied von generativen und diskriminativen Klassifikatoren verstehen



Binäre vs. n-äre Klassifikation [WITTEN et al. 2011, 338]

- Jedes n-äre Klassifikationsproblem, kann auf eine Anzahl binäre Klassifikationsprobleme reduziert werden.
- Verschiedene Strategien sind möglich! Im Prinzip eine Meta-Learn-Methode!
- *One vs Rest*: Jeweils eine Klasse tritt gegeben die Vereinigung aller restlichen Klassen an! Guter Umgang mit imbalanced classes notwendig! Es fliessen immer alle Trainingsbeispiele ins Training ein!
- *Pairwise classification*: Für k Klassen werden $k(k - 1)/2$ binäre gebildet.
- Es fliessen immer nur die Trainingsbeispiele der beiden Klassen ein (effizienteres Training)!
- Die vorhergesagte Klasse eines Testitems ist diejenige, welche am meisten gewonnen hat (Ensemble-Verfahren)!

8.1 Logistic Regression

Linear Regression vs Binäre Logistic Regression

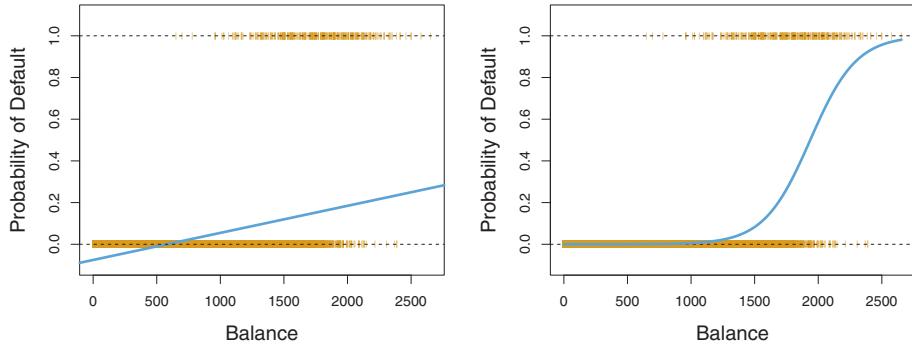


FIGURE 4.2. Classification using the `Default` data. Left: Estimated probability of `default` using linear regression. Some estimated probabilities are negative! The orange ticks indicate the 0/1 values coded for `default`(`No` or `Yes`). Right: Predicted probabilities of `default` using logistic regression. All probabilities lie between 0 and 1.

$$p(x) = \beta_0 + \beta_1 x$$

$$p(x) = \frac{e^{\beta_0 + \beta_1 x}}{1 + e^{\beta_0 + \beta_1 x}}$$

Odds, Log-odds und log-lineare Modelling

Odds

$$\frac{p(X)}{1 - p(X)} = e^{\beta_0 + \beta_1 X}. \quad (4.3)$$

The quantity $p(X)/[1 - p(X)]$ is called the *odds*, and can take on any value between 0 and ∞ . Values of the odds close to 0 and ∞ indicate very low and very high probabilities of default, respectively. For example, on average

Log-Odds (oder *logit*)

By taking the logarithm of both sides of (4.3), we arrive at

$$\log\left(\frac{p(X)}{1 - p(X)}\right) = \beta_0 + \beta_1 X.$$

Log-Linear Modeling

Die rechte Seite entspricht der linearen Regressionsformel, d.h. logistische Regression hat einen logit, der linear in x ist.

Logistic Regression mit mehr als 2 Klassen

- Z.B. mit Meta-Lernverfahren
- Wird in der NLP als Maximum-Entropy-Modellierung bezeichnet.
- Manchmal wird auch Logistic Regression als Oberbegriff für binäre und n-äre log-lineare Modellierung verwendet.

- Aber z.b. nicht in [YU et al. 2010] “Dual coordinate descent methods for logistic regression and maximum entropy models”

8.2 Maximum Entropie

8.2.1 Motivation

Maximum Entropy Modeling? MaxEnt? MEM?

Conditional Maximum Entropy Modelling

In der Sprachtechnologie seit Mitte 90-er-Jahre [BERGER et al. 1996] populäre Verallgemeinerung von *Logistic Regression (conditional exponential models)* auf multinomialen Klassen.

- Log-lineare Modellierung der Wahrscheinlichkeit $p(c|x)$ der Klasse c gegeben die Evidenz x , d.h. lineare Wahrscheinlichkeitsfunktion im Log-Raum
- Abhängigkeit zwischen Merkmalen ist unproblematisch und wird vom Modell “berücksichtigt”
- Erlaubt einfache Integration von heterogener und redundanter Information; Merkmalsauswahl kann parallel zu Optimierung erfolgen
- Einfach zu benutzen ohne aufwändiges Tunen von Hyperparametern (im Vergleich zu SVM)
- mit Garantie für globale Optimierung der Modellparameter (in Bezug auf Trainingsdaten)
- Effizienz: Langsamer als Naive Bayes, da Modell iterativ optimiert wird, aber beste Optimierungstechniken können auf heutiger Hardware Modelle mit Millionen von Merkmalen in wenigen Rechenminuten berechnen
- Gefahr von Overfitting bei spärlichen Merkmalen (Smoothing mit A-Priori-Wahrscheinlichkeiten hilft: *Gaussian Prior*)

Generative vs Discriminative

Diskriminative Modelle

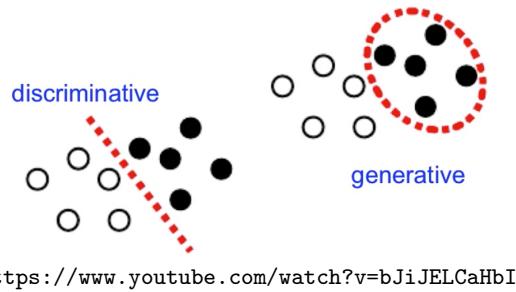
- Maximum-Entropy-Modell schätzt $p(c | x)$.
- Diskriminative Modelle berechnen bedingte Wahrscheinlichkeit direkt.
- Apriori-Wahrscheinlichkeit der Klasse kann als Evidenz in x “eingebaut” werden (Apriori-Merkmal)

Generative Modelle

- Naive Bayes schätzt $p(x | c)$ und $p(c)$, d.h. $p(x | c)p(c)$, d.h. $p(x)$ (unter Unabhängigkeitsannahme).
- Generatives Modell der Evidenz.

Generative vs Discriminative visuell

- Generative Klassifikation modelliert die Klassen.
- Diskriminative Klassifikation modelliert nur die Grenze.



<https://www.youtube.com/watch?v=bJiJELCaHbI>

Generative Stories

- Generative Modelle haben immer eine *generative story*,
- wie die modellierte Daten entstehen.

Generative Story für Multinomial Naive Bayes für binäre Textklassifikation

- Zutat 1: Eine (un)faire Münze (Bernoulli-Variable), welche die beiden Klassen parametrisiert.
- Zutat 2: Für jede Klasse C einen vielseitigen Würfel (Multinomial-Variable), dessen Seiten die Wortverteilung für die Klasse darstellt.
- Story für ein Dokument:
 1. Wirf die Münze, um eine Klasse zu bestimmen.
 2. Nimm den zur Klasse zugehörigen Würfel, und würfle ihn so oft, wie dein Dokument Wörter haben soll (Bag-of-Words-Modell)

Beispiel für Modellselektion mit Maximaler Entropie

Übersetzungen für Englisch in nach Französisch

$$p(\text{dans}) + p(\text{en}) + p(\text{à}) + p(\text{au cours de}) + p(\text{pendant}) = 1$$

⌚ Verkürzte Notation: $p(\text{dans})$ entspricht $p(\text{dans} \mid \text{in})$ usw.

Modell mit maximaler Entropie

Wie sieht die Wahrscheinlichkeitsverteilung mit maximaler Entropie aus?

Word	p
dans	$\frac{1}{5}$
en	$\frac{1}{5}$
à	$\frac{1}{5}$
au cours de	$\frac{1}{5}$
pendant	$\frac{1}{5}$

1. empirische Nebenbedingung (*constraint*)

Übersetzungen für in

$$p(\text{dans}) + p(\text{en}) = \frac{3}{10}$$
$$p(\text{dans}) + p(\text{en}) + p(\grave{a}) + p(\text{au cours de}) + p(\text{pendant}) = 1$$

Modell mit maximaler Entropie

Word	<i>p</i>
dans	$\frac{3}{20}$
en	$\frac{3}{20}$
grave{a}	$\frac{7}{30}$
au cours de	$\frac{7}{30}$
pendant	$\frac{7}{30}$

2. empirische Nebenbedingung

Übersetzungen für in

$$p(\text{dans}) + p(\grave{a}) = \frac{1}{2}$$
$$p(\text{dans}) + p(\text{en}) = \frac{3}{10}$$
$$p(\text{dans}) + p(\text{en}) + p(\grave{a}) + p(\text{au cours de}) + p(\text{pendant}) = 1$$

... mehr Constraints

- Verteilung ist möglich
- Wie findet man eine möglichst uniforme Verteilung?

Maximum-Entropy-Modelle

Modell

- modelliert alle Bedingungen, welche bekannt sind
- keine Annahmen über nicht bekannte Fakten
- Modell mit anderen Wahrscheinlichkeiten ist empirisch nicht belegbar
- *Occam's Razor*

Ein Modell mit vielen Namen ¹

- Kurz: MaxEnt oder ME
- “Maximum entropy (ME) models, variously known as log-linear, Gibbs, exponential, and multinomial logit models” [MALOUF 2002]

¹https://en.wikipedia.org/wiki/Multinomial_logistic_regression

8.2.2 Formalisierung

Probabilistische Klassifikation

Begrifflichkeiten: MAP-Dekodierung

$$c_{MAP} = \arg \max_{c \in C} p(c)p(c | \mathbf{x})$$

\mathbf{x} Evidenz als Merkmalsvektor (Notationsvariante: \vec{x})

c Klasse aus der Menge der Klassen C

c_{MAP} Wahrscheinlichste Klasse (*maximum a posteriori class*)

Merkmalsvektoren

- Eine Menge von Merkmalen $\mathbf{x} = \vec{x} = [x_1, x_2, \dots, x_n]$ bezeichnen wir auch als Merkmalsvektor.
- Nominale Merkmale werden numerisch kodiert.

Numerische Merkmalskodierung (*indicator functions*)

Fragen

- Wird “in” durch “en” übersetzt und kommt im Text danach “April”?
- x_{SuccWord} : Dasjenige Merkmal, dass das nachfolgende Wort ausdrückt.

Indikator-Funktion ME-Merkmal (*joint features: evidence and class*)

$$f_i(\mathbf{x}, c) = \begin{cases} 1 & \text{if } c = \text{en} \wedge x_{\text{SuccWord}} = \text{April} \\ 0 & \text{otherwise} \end{cases}$$

Mehrdeutigkeit des Merkmalsbegriffs in der ML-Community

- Achtung: In der Maximum-Entropy-Literatur kodieren Merkmale das gemeinsame Auftreten von Beobachtungen und einer Klasse.
- In der übrigen Literatur ist mit “Merkmal” nur die Evidenz gemeint, unabhängig von der Klasse.

Die empirische Verteilung von Merkmalen

Empirischer Erwartungswert $\mathbb{E}(f_i)$

$$\begin{aligned} \mathbb{E}(f_i) &= \sum_{\mathbf{x}, c} \tilde{p}(\mathbf{x}, c) f_i(\mathbf{x}, c) \\ &= \frac{1}{N} \sum_{j=1}^N f_i(\mathbf{x}_j, c_j) \end{aligned}$$

$\tilde{p}(\mathbf{x}, c)$ Empirische Wahrscheinlichkeit von \mathbf{x}, c

N Anzahl der Trainingsinstanzen

Interpretation von $\mathbb{E}(f_i)$

- Wird aus den Trainingsdaten gewonnen
- Wie oft “feuert” f_i ?

Bedingte exponentielle Modelle mit gewichteten Merkmalen

Allgemeine Modellformel ohne Prior

$$p(c \mid \mathbf{x}) = \frac{1}{Z_{\mathbf{x}}} \exp \left(\sum_{i=1}^K \lambda_i f_i(\mathbf{x}, c) \right)$$

λ_i Gewicht von Merkmal i

K Anzahl Merkmale

Z_x Normalisierung: $Z_x = \sum_c \exp \left(\sum_i \lambda_i f_i(\mathbf{x}, c) \right)$

Modellauswahl = Bestimmen von λ_i

2 Ziele müssen gleichzeitig erreicht werden

1. Die Erwartungswerte des Modells entsprechen den empirischen Erwartungswerten aus den Trainingsdaten.
2. Die Entropie der resultierenden Wahrscheinlichkeitsverteilung ist maximal.

Problem bei der Modellauswahl

- Wir können das optimale Modell nicht direkt berechnen.
- Lösung: Iterative Annäherung an optimales Modell.
- Verschiedene Verfahren: Generalized Iterative Scaling, Improved Iterative Scaling, Conjugate Gradient, LM-BFGS usw. [MALOUF 2002]

8.2.3 Beispiel

Beispiel: Filmkritik-Klassifikation/1

Featurevektor \vec{x}

x_0 = Dokument enthält das Wort ‘schlecht’

Trainingsinstanzen $(\vec{x}, c) \in D$

$[([1], \ominus), ([1], \ominus), ([1], \ominus), ([1], \ominus), ([0], \ominus), ([0], \ominus), ([0], \ominus), ([0], \ominus)]$

Klassen $c \in C$

c	Bedeutung
\ominus	negative Kritik
\oplus	positive Kritik

Erwartungswerte $E(p(c | \vec{x}))$

- $E(p(\ominus | [1])) = 0.75$
- $E(p(\oplus | [1])) = 0.25$
- $E(p(\ominus | [0])) = 0.5$
- $E(p(\oplus | [0])) = 0.5$

Erwartungswerte $E(p(c))$

- $E(p(\ominus)) = 0.75$
- $E(p(\oplus)) = 0.25$

Beispiel: Filmkritik-Klassifikation/1: Naive Bayes

$$p(c | \vec{x}) = \frac{p(c) \prod_i p(x_i | c)}{\sum_{c'} p(c') \prod_i p(x_i | c')} = \frac{p(c) \prod_i p(x_i | c)}{Z_{\vec{x}}}$$

$Z_{\vec{x}}$ normalisiert die bedingten Wahrscheinlichkeiten auf 1

- $p(\ominus | [1]) = \frac{p(\ominus)p([1]|\ominus)}{Z_{[1]}} = \frac{\frac{5}{8} \frac{3}{5}}{Z_{[1]}} = \frac{0.375}{Z_{[1]}} = 0.75$
- $p(\oplus | [1]) = \frac{p(\oplus)p([1]|\oplus)}{Z_{[1]}} = \frac{\frac{3}{8} \frac{1}{3}}{Z_{[1]}} = \frac{0.125}{Z_{[1]}} = 0.25$
- $Z_{[1]} = 0.375 + 0.125 = 0.5$

- $p(\ominus | [0]) = \frac{p(\ominus)p([0]|\ominus)}{Z_{[0]}} = \frac{\frac{5}{8} \frac{2}{5}}{Z_{[0]}} = \frac{0.25}{Z_{[0]}} = 0.5$
- $p(\oplus | [0]) = \frac{p(\oplus)p([0]|\oplus)}{Z_{[0]}} = \frac{\frac{3}{8} \frac{2}{3}}{Z_{[0]}} = \frac{0.25}{Z_{[0]}} = 0.5$
- $Z_{[0]} = 0.25 + 0.25 = 0.5$

Beispiel: Filmkritik-Klassifikation/1: Maximum Entropy

Maximum Entropy Features f_i

$$f_1(\vec{x}, c) = \begin{cases} 1 & \text{if } c = \text{⊗} \wedge x_0 = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$f_2(\vec{x}, c) = \begin{cases} 1 & \text{if } c = \text{⊖} \wedge x_0 = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$f_3(\vec{x}, c) = \begin{cases} 1 & \text{if } c = \text{⊕} \wedge x_0 = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$f_4(\vec{x}, c) = \begin{cases} 1 & \text{if } c = \text{⊖} \wedge x_0 = 0 \\ 0 & \text{otherwise} \end{cases}$$

Erwartungswerte $E(f_i)$

$$E(f_i) = \frac{1}{|D|} \sum_{(\vec{x}, c) \in D} f_i(\vec{x}, c))$$

- $E(f_1) = 1/8$
- $E(f_2) = 3/8$
- $E(f_3) = 1/4$
- $E(f_4) = 1/4$

Iterativ geschätzte Parameter λ_i

- $\lambda_1 = -1$
- $\lambda_2 = 0.585$
- $\lambda_3 = 0$
- $\lambda_4 = 0$

Beispiel: Filmkritik-Klassifikation/1: Maximum Entropy

$$p(c \mid \vec{x}) = \frac{\exp_2 \sum_i \lambda_i f_i(\vec{x}, c)}{\sum_{c'} \exp_2 \sum_i \lambda_i f_i(\vec{x}, c')} = \frac{2^{(\sum_i \lambda_i f_i(\vec{x}, c))}}{Z_{\vec{x}}}$$

- $Z_{\vec{x}}$ normalisiert die bedingten Wahrscheinlichkeiten auf 1
- $p(\text{⊖} \mid [1]) = \frac{2^{0.585}}{Z_{[1]}} = \frac{1.5}{Z_{[1]}} = 0.75$
- $p(\text{⊕} \mid [1]) = \frac{2^{-1}}{Z_{[1]}} = \frac{0.5}{Z_{[1]}} = 0.25$
- $Z_{[1]} = 1.5 + 0.5 = 2$

$$\bullet p(\text{⊖} \mid [0]) = \frac{2^0}{Z_{[0]}} = \frac{1}{Z_{[0]}} = 0.5$$

$$\bullet p(\text{⊕} \mid [0]) = \frac{2^0}{Z_{[0]}} = \frac{1}{Z_{[0]}} = 0.5$$

$$\bullet Z_{[0]} = 0.5 + 0.5 = 1$$

Beispiel: Filmkritik-Klassifikation/2

Featurevektor \vec{x}

x_0 = Dokument enthält das Wort ‘schlecht’ x_1 = Dokument enthält das Lemma ‘schlecht’

Trainingsinstanzen (\vec{x}, c)

$[([1, 1], \text{⊖}), ([1, 1], \text{⊖}), ([1, 1], \text{⊖}), ([1, 1], \text{⊖}), ([0, 0], \text{⊕}), ([0, 0], \text{⊕}), ([0, 0], \text{⊕}), ([0, 0], \text{⊕})]$

Klassen	
c	Bedeutung
⊖	negative Kritik
\oplus	positive Kritik

Erwartungswerte $E(p(c | \vec{x}))$

- $E(p(\text{⊖} | [1, 1])) = 0.75$
- $E(p(\oplus | [1, 1])) = 0.25$
- $E(p(\oplus | [0, 0])) = 0.5$
- $E(p(\text{⊖} | [0, 0])) = 0.5$

Beispiel: Filmkritik-Klassifikation/2: Naive Bayes

$$p(c | \vec{x}) = \frac{p(c) \prod_i p(x_i | c)}{\sum_{c'} p(c') \prod_i p(x_i | c')} = \frac{p(c) \prod_i p(x_i | c)}{Z_{\vec{x}}}$$

$$\bullet \quad p(\text{⊖} | [1, 1]) = \frac{p(\text{⊖})p([1, 1] | \text{⊖})}{Z_{[1, 1]}} = \frac{\frac{5}{8} \frac{3}{5} \frac{3}{5}}{Z_{[1, 1]}} = \frac{0.225}{Z_{[1, 1]}} = 0.8437$$

$$\bullet \quad p(\oplus | [1, 1]) = \frac{p(\oplus)p([1, 1] | \oplus)}{Z_{[1, 1]}} = \frac{\frac{3}{8} \frac{1}{3} \frac{1}{3}}{Z_{[1, 1]}} = \frac{0.042}{Z_{[1, 1]}} = 0.1563$$

$$\bullet \quad Z_{[1, 1]} = 0.225 + 0.042 = 0.267$$

$$\bullet \quad p(\text{⊖} | [0, 0]) = \frac{p(\text{⊖})p([0, 0] | \text{⊖})}{Z_{[0, 0]}} = \frac{\frac{5}{8} \frac{2}{5} \frac{2}{5}}{Z_{[0, 0]}} = \frac{0.1}{0.267} = 0.375$$

$$\bullet \quad p(\oplus | [0, 0]) = \frac{p(\oplus)p([0, 0] | \oplus)}{Z_{[0, 0]}} = \frac{\frac{3}{8} \frac{2}{3} \frac{2}{3}}{Z_{[0, 0]}} = \frac{0.167}{0.267} = 0.625$$

$$\bullet \quad p(\text{⊖} | [0, 1]) = \frac{p(\text{⊖})p([0, 1] | \text{⊖})}{Z_{[0, 1]}} = \frac{\frac{5}{8} \frac{2}{5} \frac{3}{5}}{Z_{[0, 1]}} = \frac{0.15}{0.233} = 0.643$$

$$\bullet \quad p(\oplus | [0, 1]) = \frac{p(\oplus)p([0, 1] | \oplus)}{Z_{[0, 1]}} = \frac{\frac{3}{8} \frac{2}{3} \frac{1}{3}}{Z_{[0, 1]}} = \frac{0.083}{0.233} = 0.357$$

Beispiel: Filmkritik-Klassifikation/2: Maximum Entropy

Maximum Entropy Features f_i

$$f_1(\vec{x}, c) = \begin{cases} 1 & \text{if } c = \text{⊗} \wedge x_0 = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$f_2(\vec{x}, c) = \begin{cases} 1 & \text{if } c = \text{⊗} \wedge x_1 = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$f_3(\vec{x}, c) = \begin{cases} 1 & \text{if } c = \text{⊖} \wedge x_0 = 1 \\ 0 & \text{otherwise} \end{cases}$$

$$f_4(\vec{x}, c) = \begin{cases} 1 & \text{if } c = \text{⊖} \wedge x_1 = 1 \\ 0 & \text{otherwise} \end{cases}$$

...

Iterativ geschätzte Parameter λ_i

- $\lambda_1 = -0.5$
- $\lambda_2 = -0.5$
- $\lambda_3 = 0.292$
- $\lambda_4 = 0.292$
- $\lambda_5 = 0$
- $\lambda_6 = 0$
- $\lambda_7 = 0$
- $\lambda_8 = 0$

Beispiel: Filmkritik-Klassifikation/2: Maximum Entropy

$$p(c \mid \vec{x}) = \frac{\exp_2 \sum_i \lambda_i f_i(\vec{x}, c)}{\sum_{c'} \exp_2 \sum_i \lambda_i f_i(\vec{x}, c')} = \frac{2^{(\sum_i \lambda_i f_i(\vec{x}, c))}}{Z_{\vec{x}}}$$

$$\bullet p(\text{⊖} \mid [1, 1]) = \frac{2^{0.292+0.292}}{Z_{[1,1]}} = \frac{1.5}{2} = 0.75$$

$$\bullet p(\text{⊗} \mid [1, 1]) = \frac{2^{-0.5-0.5}}{Z_{[1,1]}} = \frac{0.5}{2} = 0.25$$

$$\bullet p(\text{⊖} \mid [0, 0]) = \frac{2^{0+0}}{Z_{[0,0]}} = \frac{1}{2} = 0.5$$

$$\bullet p(\text{⊗} \mid [0, 0]) = \frac{2^{0+0}}{Z_{[0,0]}} = \frac{1}{2} = 0.5$$

$$\bullet p(\text{⊖} \mid [0, 1]) = \frac{2^{0+0.292}}{Z_{[0,1]}} = \frac{1.224}{1.931} = 0.634$$

$$\bullet p(\text{⊗} \mid [0, 1]) = \frac{2^{0-0.5}}{Z_{[0,1]}} = \frac{0.707}{1.931} = 0.366$$

Beispiel: Fazit

Fazit

- Für gesehene Ereignisse nähert sich die Schätzung des Maximum-Entropy-Modells der empirischen Wahrscheinlichkeitsverteilung an.
- Bei Naive Bayes ist dies nicht unbedingt der Fall; vor allem dann nicht, wenn Merkmale voneinander abhängig sind.
- Maximum-Entropy-Modelle kommen besser mit voneinander abhängigen Merkmalen zu recht; λ -Parameter werden angepasst.

8.2.4 Fazit

Maximum-Entropy-Tools

- NLTK-Python-Implementation mit anschaulichem Erklärungsmodus ²
 - keine
 - beinhaltet Schnittstelle zu MEGAM
- MEGAM³ [DAUMÉ III 2008]
 - Training mit LM-BFGS
 - sehr schnell und effizient mit Millionen von Merkmalen
 - verhindert Overfitting bei Sparse-Data mit *Gaussian Prior* auf Lambda ($\mu = 0$): Maximum-Entropy-Modelling mit MAP beschränkt den Einfluss der Trainingsdaten (L2-Regularisierung)
 - beinhaltet optional eine Merkmalsselektion
 - kann via NLTK verwendet werden
- Liblinear: Logistic Regression und lineare SVMs
- sklearn benutzt unter anderem liblinear
- OpenNLP
 - Java-basiert; erlaubt ausgereiftere Merkmalsfunktionen

Iterativer Algorithmus zur Gewichtsbestimmung

²<http://www.nltk.org/api/nltk.classify.html#module-nltk.classify.maxent>

³<https://www.umiacs.umd.edu/~hal/megam/>

- **Inputs:** A collection \mathcal{D} of labeled documents and a set of feature functions f_i .
- Set the constraints (Equation 2). For every feature f_i , estimate its expected value on the training documents.
- Initialize all the λ_i 's to be zero.
- Iterate until convergence:
 - Calculate the expected class labels for each document with the current parameters, $P_\Lambda(c|d)$ (Equation 3).
 - For each parameter λ_i :
 - Set $\frac{\partial B}{\partial \delta_i} = 0$ and solve for δ_i (Equation 9).
 - set $\lambda_i = \lambda_i + \delta_i$
- **Output:** A text classifier that takes an unlabeled document and predicts a class label.

Table 1: An outline of the Improved Iterative Scaling algorithm for estimating the parameters for maximum entropy.

[NIGAM et al. 1999]

MaxEnt mit MAP (Gaussian Prior on λ)

To integrate a prior into maximum entropy, we use maximum a posteriori estimation for the exponential model, instead of maximum likelihood estimation. We use a Gaussian prior for the model, with the mean at zero, and a diagonal covariance matrix. This prior favors feature weightings that are closer to zero, that is, are less extreme. The prior probability of the model is just the product over the Gaussian of each feature value λ_i with variance σ_i^2 :

$$P(\Lambda) = \prod_i \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{\lambda_i^2}{2\sigma_i^2}\right). \quad (10)$$

[NIGAM et al. 1999]

Moderne ML-Formulierung des Optimierungsziels [YU et al. 2010]

$$\mathcal{P}_w(y|x) \equiv \frac{\exp(\mathbf{w}^T \mathbf{f}(x, y))}{\sum_{y'} \exp(\mathbf{w}^T \mathbf{f}(x, y'))},$$

where x denotes a context, y is the label of the context, and $\mathbf{w} \in R^n$ is the weight vector. A function vector $\mathbf{f}(x, y) \in R^n$ indicates features extracted from the context x and the label y . Assume N training samples $\{(x, y)\}$ are given, and we have grouped x 's to l unique contexts $\{x_i\}$ and calculate the empirical probability distribution $\tilde{\mathcal{P}}(x_i, y) = N_{x_i, y}/N$, where $N_{x_i, y}$ is the number of times that (x_i, y) occurs in the training data. ME minimizes the following regularized negative log-likelihood:

$$\begin{aligned} \min_{\mathbf{w}} P^{\text{ME}}(\mathbf{w}) &= - \sum_{i=1}^l \sum_y \tilde{\mathcal{P}}(x_i, y) \log \mathcal{P}_w(y|x_i) + \frac{1}{2\sigma^2} \mathbf{w}^T \mathbf{w} \\ &= \sum_{i=1}^l \tilde{\mathcal{P}}(x_i) \log \left(\sum_y \exp(\mathbf{w}^T \mathbf{f}(x_i, y)) \right) - \mathbf{w}^T \tilde{\mathbf{f}} + \frac{1}{2\sigma^2} \mathbf{w}^T \mathbf{w}, \end{aligned} \quad (5)$$

where σ is the penalty parameter similar to C in (1), $\tilde{\mathcal{P}}(x_i) = \sum_y \tilde{\mathcal{P}}(x_i, y)$ is the marginal probability of x_i , and

$$\tilde{\mathbf{f}} = \sum_{i=1}^l \sum_y \tilde{\mathcal{P}}(x_i, y) \mathbf{f}(x_i, y) \quad (6)$$

is the expected vector of $\mathbf{f}(x_i, y)$. For convenience, we assume that

$$y_i \in Y \equiv \{1, 2, \dots, |Y|\}.$$

Wofür MEMs?

Log-lineare Modelle und SMT

- Loglineare Modellierung ist dominante Methode in Statistischer Maschineller Übersetzung zur Kombination verschiedener Modelle (Übersetzungsmodelle, Sprachmodelle).
- Parameterschätzung nicht über Maximum-Entropy-Kriterium, sondern Optimierung der Übersetzungsqualität auf Entwicklungsdaten
- Minimum Error Rate Training (MERT); MIRA

Vertiefung

- Pflichtlektüre: Kapitel 16.2 “Maximum Entropy Modelling” in [MANNING und SCHÜTZE 1999] (ohne technische Details zum IIS-Algorithmus)
- Blogseintrag zur Äquivalenz von *Multinomial Logistic Regression* und ME: ⁴

⁴www.win-vector.com/blog/2011/09/the-simpler-derivation-of-logistic-regression/

Kapitel 9

Support Vector Machines

Lernziele

- Idee hinter linearen *Maximal Margin Classifiers* verstehen
- Idee der Stützvektoren verstehen (*Support Vector Classifier*)
- Verstehen, wie Kernel-Funktionen die Daten transformieren und dadurch nicht-lineare Entscheidungsgrenzen entstehen können (*Support Vector Machine*)
- Verstehen, wie binäre Klassifikationsverfahren auf n-äre Klassifikationsprobleme angewendet werden können
- Verstehen, dass unterschiedliche Klassifikationsverfahren unterschiedliche Entscheidungsgrenzen beherrschen

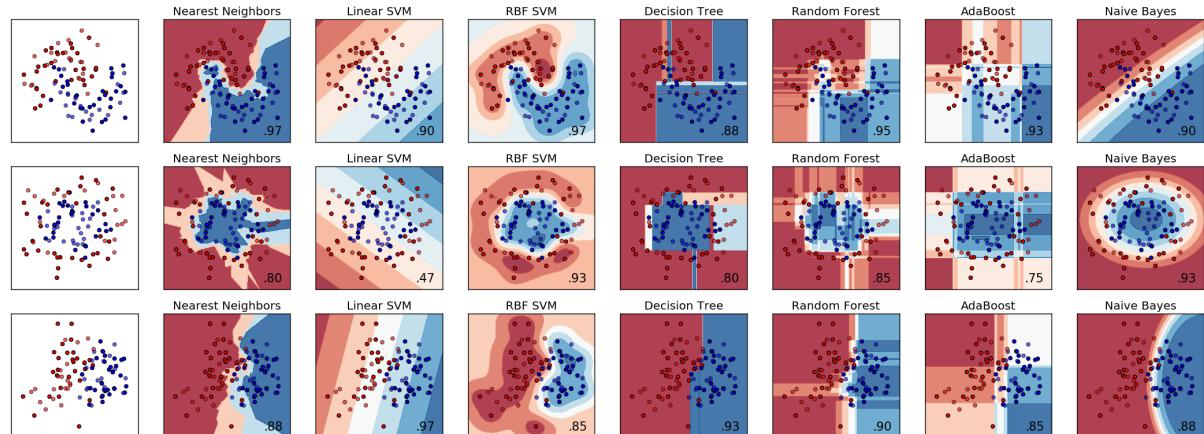
9.1 Support Vector Machines

Selbständige Lektüre

- Kapitel 9.1-9.5 “Support Vector Machines” in [JAMES et al. 2013]
- Offene Fragen bitte ins Olat-Feedback-Forum posten!

9.2 Entscheidungsgrenzen

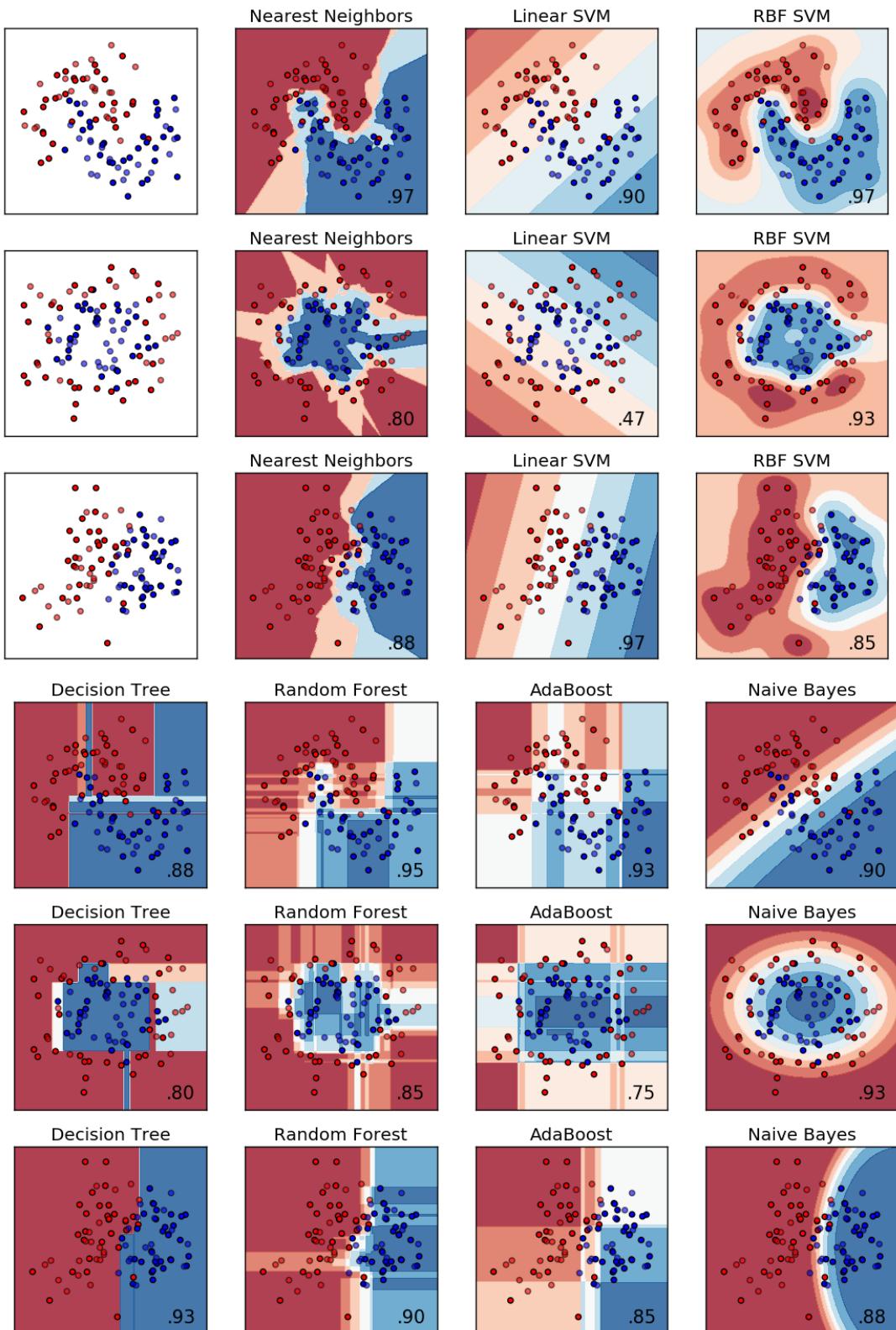
Verschiedene Klassifikatoren = verschiedene Grenzen



http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

- Trainingspunkte sind in deckender Farbe, Testpunkte sind halbtransparent.

• Datenset ist synthetisch erstellt



Vertiefung

- Pflichtlektüre: Kapitel 9.1-9.5 “Support Vector Machines” in [JAMES et al. 2013]

- Offene Fragen bitte ins Feedback-Forum posten!
- Beispielprogramme für Klassifikationsvergleich in sklearn: [http://scikit-learn.org/stable/
auto_examples/classification/plot_classifier_comparison.html](http://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html)

Kapitel 10

Sequenzklassifikation und CRFs

Lernziele

- Erweiterung von Maximum-Entropie-Klassifikation (aka *Logistic Regression*) auf Sequenzen verstehen
- *Linear Chain Conditional Random Fields* (CRF) formal verstehen
- Praktisches Werkzeug *wapiti* für Linear Chain CRFs mit Merkmalsmustern anwenden können

Der grössere Zusammenhang: CRF und HMM

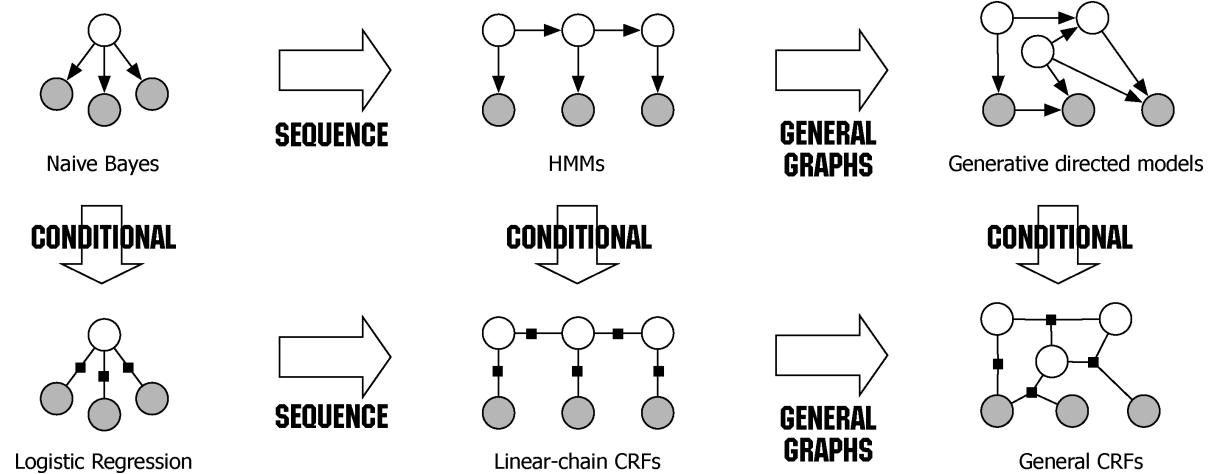


Fig. 2.4 Diagram of the relationship between naive Bayes, logistic regression, HMMs, linear-chain CRFs, generative models, and general CRFs.

Quelle: [SUTTON und McCALLUM 2012, 286]

Hinweise: Grau gefüllte Variablen sind gegeben bei der Vorhersage. Siehe [KSCHISCHANG et al. 2001] für Bedeutung von schwarzen Kästchen.

10.1 HMM

Sequenzklassifikation: POS-Tagging-Problem

Wahrscheinlichkeitsmodell

Gesucht ist die wahrscheinlichste Wortartfolge t_{1n}^* für eine gegebene Wortfolge $w_{1n} = w_1, \dots, w_n$

$$t_{1n}^* = \arg \max_{t_{1n}} p(t_{1n} | w_{1n})$$

Nach Anwendung des Bayes'schen Theorems, ergibt sich:

$$t_{1n}^* = \arg \max_{t_{1n}} \frac{p(t_{1n})p(w_{1n}|t_{1n})}{p(w_{1n})} = \arg \max_{t_{1n}} p(t_{1n})p(w_{1n}|t_{1n})$$

Nach Zerlegung in ein Produkt bedingter Wahrscheinlichkeiten:

$$t_{1n}^* = \arg \max_{t_{1n}} \prod_{i=1}^n p(t_i | t_1, \dots, t_{i-1}) p(w_i | w_1, \dots, w_{i-1}, t_1^n)$$

Sequenzklassifikation: HMM-Ansatz (Markov Chains \uparrow)

Hidden-Markow-Modelle

HMMs machen folgende vereinfachenden Annahmen:

- Das Wortart-Tag t_i hängt nur von den k vorherigen Tags ab.
- Das Wort w_i hängt nur von seiner Wortart t_i ab.
- Die Wahrscheinlichkeiten sind unabhängig von der Position.

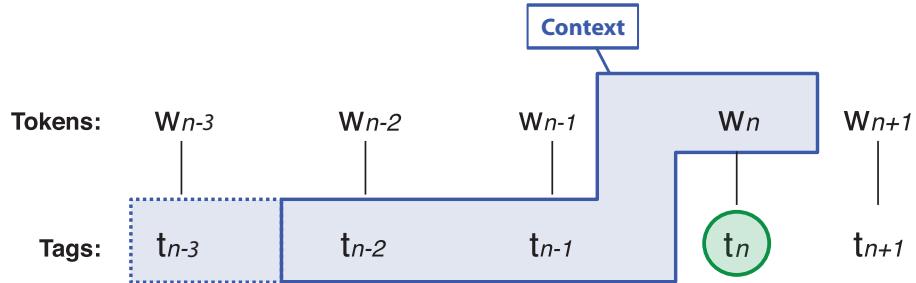
$$\begin{aligned} t_{1n}^* &= \arg \max_{t_{1n}} \prod_{i=1}^n p(t_i | t_1, \dots, t_{i-1}) p(w_i | t_1, \dots, t_{i-1}) \\ &= \arg \max_{t_{1n}} \prod_{i=1}^n \underbrace{p(t_i | t_{i-k}, \dots, t_{i-1})}_{\text{Kontextwahrsch.}} \underbrace{p(w_i | t_i)}_{\text{lexikalische Wk.}} \end{aligned}$$

Dieses Modell heißt **Hidden-Markow-Modell**, weil die Zustände (Tags) nicht direkt beobachtbar sind.

Für $k = 2$ erhält man einen Trigramm-Tagger.

Context Model of HMM-Based Taggers

Transition Order (hunpos, rft)



- Trigram (c2): $P(t_n | t_{n-1}, t_{n-2})$
- Quadrigram (c3): $P(t_n | t_{n-1}, t_{n-2}, t_{n-3})$

Emission Order (hunpos)

- Classical order of 1 (e1): $P(w_n | t_n)$
- Emission order of 2 (e2): $P(w_n | t_{n-1}, t_n)$

10.2 CRF

10.2.1 Case Study

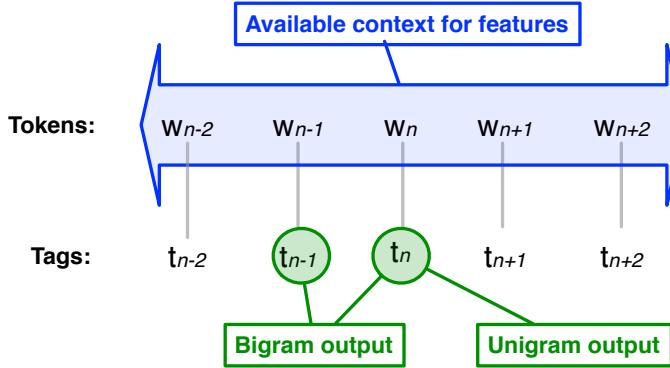
Case Tagging Task: Assign Correct Case Labels in Context

English	German	Case	GERTWOL
<i>The</i>	Die	Nom(inative)	Nom,Acc
<i>exact</i>	genaue	Nom(inative)	Nom,Acc
<i>determination</i>	Bestimmung	Nom(inative)	Nom,Acc,Dat,Gen
<i>of</i>	von	Dat(ive)	Dat
<i>case</i>	Kasus	Dat(ive)	Nom,Acc,Dat,Gen
<i>represents</i>	stellt	-	-
<i>a</i>	eine	Acc(usative)	Nom,Acc
<i>basic</i>	grundlegende	Acc(usative)	Nom,Acc
<i>morpho-syntactic</i>	morpho-syntaktische	Acc(usative)	UNKNOWN
<i>requirement</i>	Anforderung	Acc(usative)	Nom,Acc,Dat,Gen
<i>for</i>	der	Gen(itive)	Nom,Dat,Gen
<i>language processing</i>	Sprachverarbeitung	Gen(itive)	Nom,Acc,Dat,Gen
	dar	-	-
.	.	.	.

GERTWOL: Typical morphological analyzer output concerning case.

CRF Model: Hand-Crafted Feature Templates

Any property from the token level can serve as (part of) a feature. These properties can be related to bigram output tuples (B: *bigram feature*) or simple output classes (U: *unigram features*) or both (*).



Feature space in sequential CRFs

$\%X[,0]$	$\%X[,1]$	$\%X[n,]$
Mit	-	-3
dieser	Dat	-2
neuen	Dat	-1
Praxis	Dat	0 ($=w/t_n$)
reagiert	-	1
das	Nom	2
Gericht	Nom	3

Matrix notation for feature templates

Feature Templates in wapiti and Similar Tools

Template instantiation operators ($\%_ [...]$) and feature template names

$\%X U:wrd-L=%X[-1,0]$ is instantiated to $U:wrd-L=neuen$ with unigram output class Dat.

$\%m$ Regex match for first and last character $U:PS=%m[0,0,"^ ."] / %m[0,0," .\$"]$ is instantiated to $U:PS-1=P/s$.

$\%t$ Matches yes/no: bigram template $B:upper=%t[0,0,"^\u00d6"]$ instantiates to $B:upper=True$ and is related to the bigram output tuple (Dat,Dat).

Verticalized input

0	1
Mit	- -3
dieser	Dat -2
neuen	Dat -1
Praxis	Dat 0 (=tn)
reagiert	- 1
das	Nom 2
Gericht	Nom 3

Feature Template for morphological tagging in German

```

U:word LL=%X[-2,0]
U:word L=%X[-1,0]
U:word O=%X[ 0,0]
U:word R=%X[ 1,0]
U:word RR=%X[ 2,0]
U:suf-1 O=%m[ 0,0,".?"]
U:suf-2 O=%m[ 0,0,"?.?"]
U:suf-3 O=%m[ 0,0,".?.?.?"]
U:pre-1 O=%m[ 0,0,"^.?"]
U:pre-2 O=%m[ 0,0,"^.?.?"]
U:pre-3 O=%m[ 0,0,"^.?.?.?"]
U:word R/O=%X[ 1,0] / %X[0,0]
U:word L/O=%X[-1,0] / %X[0,0]

```

```

U:word R/L=%X[1,0]/%X[-1,0]
*:is-upper X=%t[ 0,0,"^\\u"]
U:suf-2 L/O/R=%m[-1,0,".?.?"]/%m[ 0,0,".?.?"]/%m[1,0,".?.?"]
U:presuf-1 O=%m[ 0,0,"^."]/%m[ 0,0,".$"]

```

Features and Training

Feature Templates with highest performance

- Current token, and its prefixes and suffixes of length 1 to 3
- Combination of first and last character of current token
- Upper case or not?
- Neighbor tokens up to 2 positions to the right/left
- Token bigrams of current token and neighbors
- Combinations of the final 2 characters of current and neighbor tokens

Training of CRFs = determining the weights of features

- CRF training determines for each feature a *weight* that maximizes the probability of the tagged sequences from the training material wapiti:
- A *development set* (18% in our case) is needed to prevent overfitting.

Unknown Words

- HMM taggers have built-in support to guess tags of tokens that were not seen in the training data.
- Typically, a suffix trie derived from infrequent token is used.
- General CRF tools do not support that.
- We need suffix features to “simulate” the effect of suffix tries.

Tagging Accuracy: All Words

TIGER							
System:+Internal Tags	Mean	SD	Δ_{abs}	Δ_{rel}	$\Delta_{rel_{bs}}$	P val	ΔCI_l
tnt:num,pos	90.47	0.41					
hp-c3-e1:num,pos	91.01	0.39	+0.54	+0.59	+0.59	0.00	+0.51
rft-c5:gend,num,pers,pos	91.50	0.40	+0.49	+0.54	+1.14	0.00	+0.44
wapiti: num,pos	93.77	0.29	+2.27	+2.48	+3.65	0.00	+2.14
TUEBA							
tnt:num,pos	90.82	0.27					
hp-c3-e1:num,pos	91.35	0.29	+0.54	+0.59	+0.59	0.00	+0.49
rft-c4:gend,num,pers,pos	91.87	0.24	+0.52	+0.56	+1.16	0.00	+0.47
wapiti:num,pos	93.84	0.16	+1.97	+2.14	+3.33	0.00	+1.88

- rel_{bs} : Relative performance improvement compared to baseline in table
- P val: P value on exact pairwise t test between mean differences of cross-validation folds
- ΔCI_l : lower confidence interval (95%)

Overall Improvement over Vanilla Trigram Tagger

System:Tagset	Mean	SD	Δ_{abs}	Δ_{rel}	$\Delta_{rel_{bs}}$	P val	ΔCI_l
TIGER							
tnt:case	84.36	0.44					
wapiti:case,num,pos	93.77	0.29	+9.41	+11.16	+11.16	0.00	+9.29
TUEBA							
tnt:case	84.68	0.36					
wapiti:case,num,pos	93.84	0.16	+9.15	+10.81	+10.81	0.00	+9.02

10.2.2 Formal

Vorhersage der wahrscheinlichsten Sequenz

Conditional Random Fields:
Modelling the Conditional Distribution

Model the Conditional Distribution:

$$P(\mathbf{y} \mid \mathbf{x})$$

To predict a sequence compute:

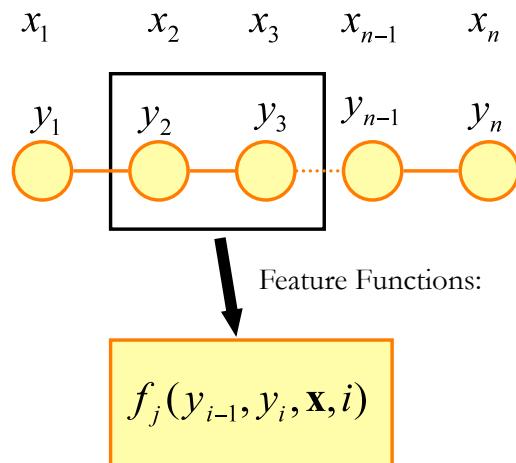
$$\mathbf{y}^* = \arg \max_{\mathbf{y}} P(\mathbf{y} \mid \mathbf{x})$$



Must be able to compute it efficiently.

Bigramm-Merkmalsfunktionen

Conditional Random Fields: Feature Functions

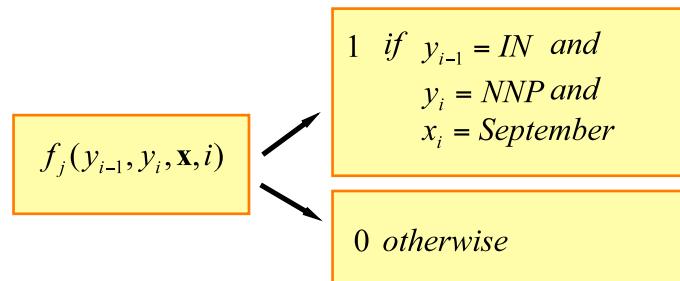


Quelle: [QUATTTONI 2013]

Bigramm-Feature-Funktion

Feature Functions

Express some characteristic of the empirical distribution
that we wish to hold in the model distribution



Quelle: [QUATTTONI 2013]

Exponential Conditional Sequence Model

Conditional Random Fields:: Distribution

Label sequence modelled as a normalized product of feature functions:

$$P(\mathbf{y} | \mathbf{x}, \boldsymbol{\lambda}) = \frac{1}{Z(\mathbf{x})} \exp \sum_{i=1}^n \sum_j \lambda_j f_j(y_{i-1}, y_i, \mathbf{x}, i)$$

$$Z(\mathbf{x}) = \sum_{\mathbf{y} \in \mathcal{Y}} \sum_{i=1}^n \sum_j \lambda_j f_j(y_{i-1}, y_i, \mathbf{x}, i)$$

The model is log-linear on the Feature Functions

Quelle: [QUATTTONI 2013]

Trainingsmaterial und Likelihood

Parameter Estimation: Maximum Likelihood

IID training samples:

$$D = [(\mathbf{x}^1, \mathbf{y}^1), (\mathbf{x}^2, \mathbf{y}^2), \dots, (\mathbf{x}^m, \mathbf{y}^m)]$$

(negative) Conditional Log-Likelihood:

$$\begin{aligned} L(\boldsymbol{\lambda}, D) &= -\log \left(\prod_{k=1}^m P(\mathbf{y}^k | \mathbf{x}^k, \boldsymbol{\lambda}) \right) \\ &= -\sum_{k=1}^m \log \left[\frac{1}{Z(\mathbf{x}_m)} \exp \sum_{i=1}^n \sum_j \lambda_j f_j(y_{i-1}^k, y_i^k, \mathbf{x}^m, i) \right] \end{aligned}$$

Quelle: [QUATTTONI 2013]

Optimierungsziel

Parameter Estimation: Maximum Likelihood

Maximum Likelihood Estimation

Set optimal parameters to be:

$$\lambda^* = \arg \min_{\lambda} L(\lambda, D) + C \frac{1}{2} \|\lambda\|^2$$

This function is convex, i.e. no local minimums

Quelle: [QUATTTONI 2013]

Optimierungsziel inklusive L2-Regularisierung

Parameter Estimation: Maximum Likelihood

Maximum Likelihood Estimation

Set optimal parameters to be:

$$\lambda^* = \arg \min_{\lambda} L(\lambda, D) + C \frac{1}{2} \|\lambda\|^2$$

This function is convex, i.e. no local minimums

Quelle: [QUATTTONI 2013]

Berechnung der optimalen Parameter

Parameter Estimation: Optimization

Let: $F_j(\mathbf{y}, \mathbf{x}) = \sum_{i=1}^n f_j(y_{i-1}, y_i, \mathbf{x}, i)$

Differentiating the log-likelihood with respect to parameter λ_j

$$\frac{\partial L(\boldsymbol{\lambda}, D)}{\partial \lambda_j} = \frac{-1}{m} \sum_{k=1}^m F_j(\mathbf{y}^k, \mathbf{x}^k) + \sum_{k=1}^m E_{P(\mathbf{y}|\mathbf{x}^k, \boldsymbol{\lambda})} [F_j(\mathbf{y}, \mathbf{x}^k)]$$

↓
Observed Mean
Feature Value

↓
Expected Feature
Value Under
The Model

Quelle: [QUATTTONI 2013]

Verbindung zu Maximum Entropie

Maximum Entropy Interpretation

Notice that at the optimal solution of:

$$\boldsymbol{\lambda}^* = \arg \min_{\boldsymbol{\lambda}} L(\boldsymbol{\lambda}, D) + C \frac{1}{2} \|\boldsymbol{\lambda}\|^2$$

We must have that:

$$\frac{1}{m} \sum_{k=1}^m F_j(\mathbf{y}^k, \mathbf{x}^k) = \sum_{k=1}^m E_{P(\mathbf{y}|\mathbf{x}^k, \boldsymbol{\lambda})} [F_j(\mathbf{y}, \mathbf{x}^k)]$$

Maximizing log-likelihood	\approx	Finding max-entropy distribution that satisfies the set of constraints defined by the feature functions
---------------------------	-----------	---

Quelle: [QUATTTONI 2013]

“Decoding”: Berechnen der wahrscheinlichsten Sequenz

CRF's Inference

Given a model, i.e. parameter values

Can we compute the following efficiently?

Best Label Sequence

$$\mathbf{y}^* = \arg \max_{\mathbf{y}} P(\mathbf{y} | \mathbf{x}, \lambda)$$

Expected Values

$$\begin{aligned} \sum_{k=1}^m E_{P(\mathbf{y}|\mathbf{x}^k, \lambda)} [F_j(\mathbf{y}, \mathbf{x}^k)] &= \sum_{k=1}^m \sum_{\mathbf{y}} p(\mathbf{y} | \mathbf{x}^k, \lambda) F_j(\mathbf{y}, \mathbf{x}^k) \\ &= \sum_{k=1}^m \sum_{i=1}^n \sum_{\mathbf{y}: y_{i-1}=a, y_i=b} p(y_{i-1} = a, y_i = b | \mathbf{x}^k, \lambda) f_j(a, b, \mathbf{x}^k, i) \end{aligned}$$

Both can be computed using dynamic programming.

Quelle: [QUATTTONI 2013]

Fazit

- CRFs sind State-of-the-Art Sequenzklassifikatoren
- Viel aufwändiger im Training (weil die Gewichte in Abhängigkeit der wahrscheinlichsten Sequenz (!) optimiert werden) im Vergleich zu HMM
- Schnell in der Anwendung
- Mit *Elastic-Net*-Regularisierung ($\rho_1 * |\lambda|_1 + \rho_2 / 2.0 * \|\lambda\|_2^2$) wie in wapiti werden die relevanten Merkmale beim Training automatisch ausgewählt (L1 *feature selection*) und mit einem mit ρ_2 gewichteten Prior versehen (L2)
- CRFs ohne Bigramm-Merkmale über Sequenzen der Länge 1 sind Maximum-Entropie-Modelle!

Kapitel 11

Unsupervisede Verfahren

Lernziele

- Verschiedene unsupervisede Verfahren für Ähnlichkeitsmessung
- Flaches und hierarchisches Clustering
- *Soft* und *Hard Clustering*
- *Topic Modeling*
- *Latent Semantic Indexing*
- Verschiedene Arten von *Embeddings* kennen: für Wörter (monolingual und bilingual) und für Sätze

11.1 Clustering

11.1.1 Flach

K-Means: Einfaches Expectation-Maximization-Verfahren

Schnelles, hartes, flaches Clustering-Verfahren, wo der Abstand (euklidisch oder besser L1-Norm) zum Mittelpunkt als Homogenitätskriterium zählt.

- Centroid: Mittelpunkt (Durchschnittswert aller Vektoren eines Clusters); normalerweise kein Datenpunkt
- K: Anzahl Cluster = Anzahl Centroiden

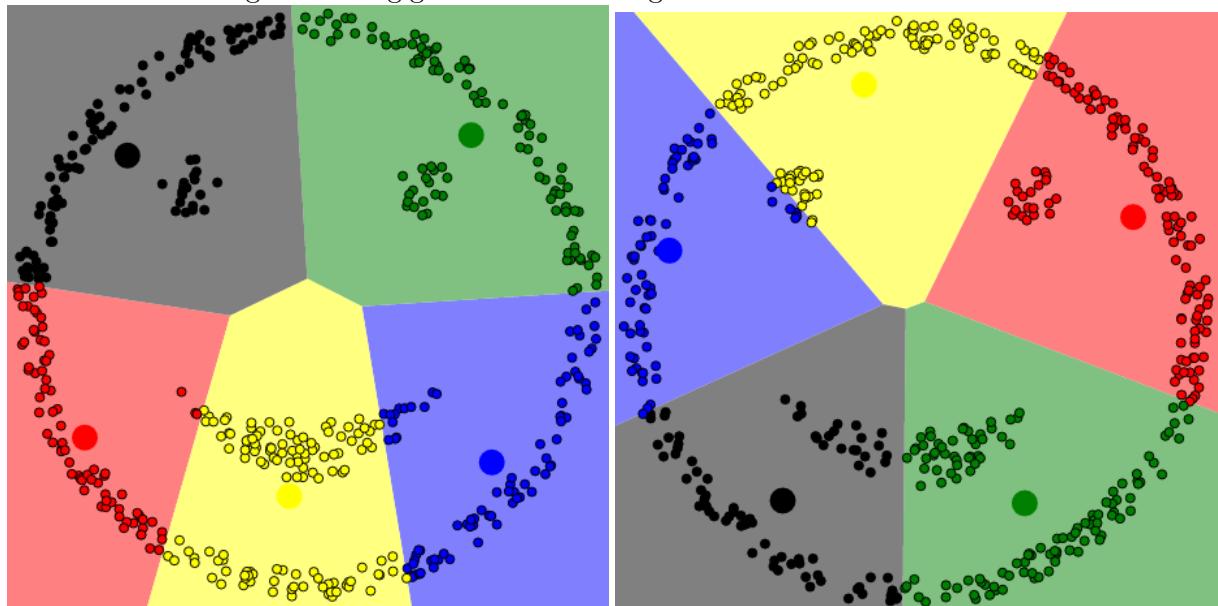
K-Means ausgehend von zufällig gesetzten Centroiden

1. Berechne Clusters über minimale Distanz zum Centroid
2. Berechne Mittelpunkte aller Datenpunkte im Cluster und setze ihn als neuen Centroid

Wiederhole 1-2 bis keine Veränderung mehr eintritt oder n-Mal!

K-Means¹: Lokale Minima

K-Means-Clustering ist abhängig von Initialisierung!



11.1.2 Hierarchisch

Hierarchisches Clustering

- Hierarchie von Clustern
- Topdown: Splitte Datengruppe mit niedrigster Kohärenz
- Bottomup: Vereinige Datengruppe mit höchster Ähnlichkeit

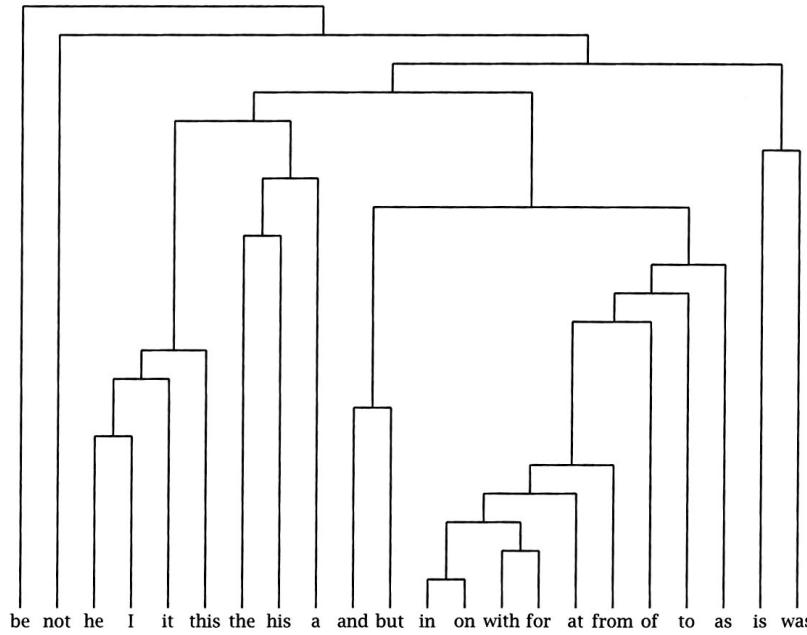
Varianten der Ähnlichkeitsmessung

- Single Link: Ähnlichkeit der ähnlichsten Elemente
- Complete Link: Ähnlichkeit der unähnlichsten Elemente
- Group-Average: Ähnlichkeit des Durchschnitts

Dendogramme von hierarchischen Clustern

Englische Stopwörter (*linker und rechter Nachbar als Merkmale*)

¹<http://www.naftaliharris.com/blog/visualizing-k-means-clustering>



Quelle: [MANNING und SCHÜTZE 1999, 496]

Globale semantische Merkmale

Codes für hierarchische Hard-Clusters [DERCZYNSKI et al. 2015]

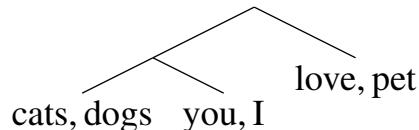


Figure 1: A binary, hierarchical clustering of semantically similar entries. Each leaf corresponds to a cluster of words (i.e., a “class”) and leaves near to their common ancestors correspond to clusters that are similar to each other.

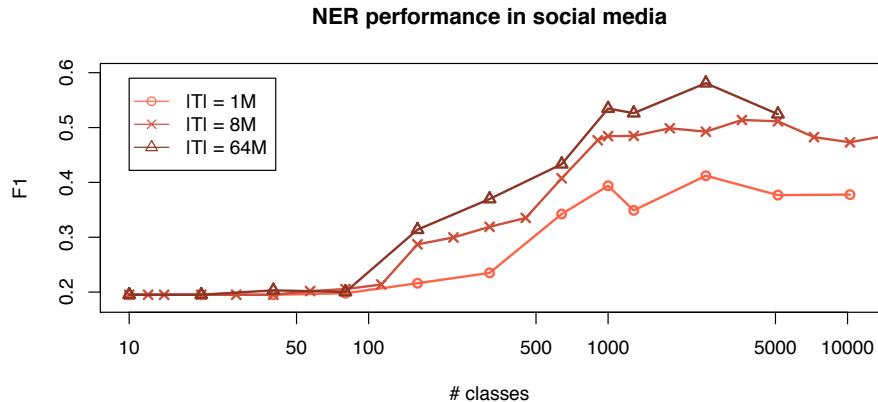
Bit path	Word types
00111001	can cn cann caan cannn ckan shalll ccan caaan cannnn caaaaan
001011111001	ii id ion iv ll iii ud wd uma ul idnt provoking hed 1+1 ididnt hast ine 2+2 idw #thingsblackpeople doledo iiiii #onlywhitepeople dost doan uon apt-get

Table 1: Sample Brown clusters over English tweets.¹ Each set of terms is a leaf in the hierarchy.

- Die Cluster-ID eines Worts ergibt sich aus der Pfad in der Clusterhierarchie!
- Eindeutige ID für jedes Wort, über dem geclustert wurde!
- Nützlich in vielen Anwendungen!

- Anzahl Cluster muss sorgfältig auf Applikation abgestimmt werden [DERCZYNSKI et al. 2015]

Nutzen von Brown-Cluster-IDs für NER



Quelle: [DERCZYNSKI et al. 2015]

Sowohl Textmenge wie Cluster-Anzahl spielen eine wichtige Rolle!

Hard vs. Soft Clustering

- Hart: Jedes Element ist in genau einer Gruppe!
- Weich: Jedem Element ist eine Gruppendiftribution zugeordnet!

Beispiele von Soft-Clustering-Verfahren

- Topic-Modeling ist Soft-Clustering von Dokumenten
- Gaussian Mixture Models: Cluster als Normalverteilungen

11.1.3 Topic Modeling

Probabilistisches Topic Modeling

Ziel

Automatisches Zuordnen von Textdokumenten zu automatisch und unsupervisert erstellten *Topics* (Themenbereichen). Dokumente enthalten normalerweise *mehr als ein Topic*.

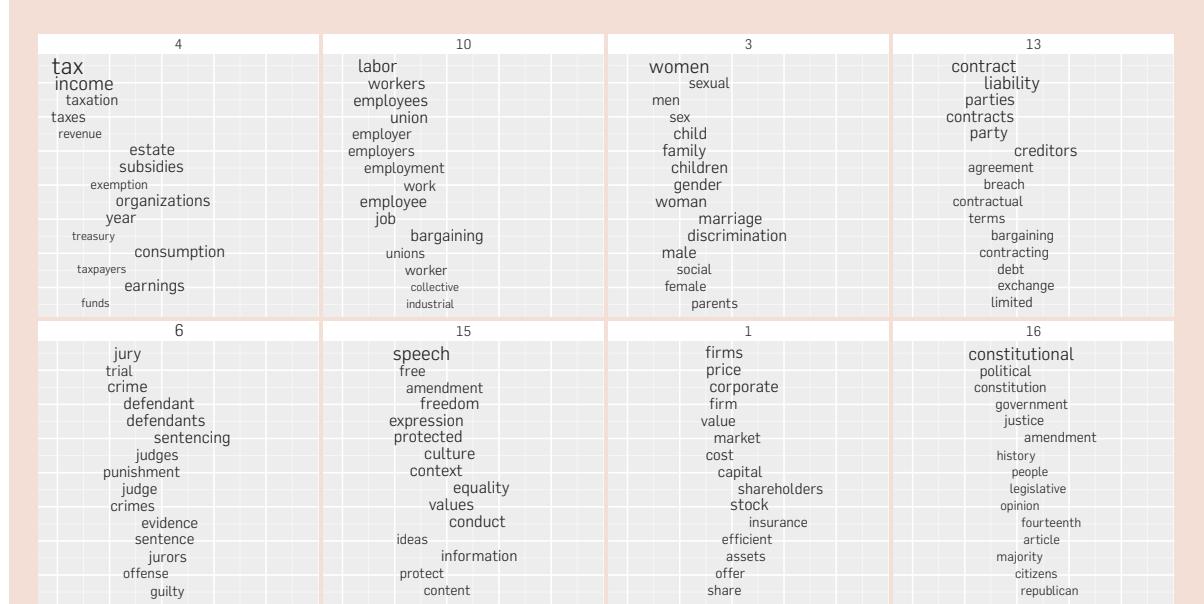
Methoden

Fortgeschrittene statistische maschinelle Lernverfahren wie LDA (*Latent Dirichlet Allocation*).

Ausgangsmaterial

- Möglichst grosse Textsammlung mit möglichst vielen Dokumenten und eher heterogenen Themenbereichen
- Ungefähr Vorstellung der Anzahl Topics, die man unterscheiden möchte

Figure 3. A topic model fit to the Yale Law Journal. Here, there are 20 topics (the top eight are plotted). Each topic is illustrated with its top-most frequent words. Each word's position along the x-axis denotes its specificity to the documents. For example "estate" in the first topic is more specific than "tax."



Topic Modeling: Intuitiv erklärt

Was sind Dokumente?

Dokumente sind eine statistische Verteilung (Mischung) von Topics. Typischerweise ein Haupt- und mehrere Neben-Themenbereiche, welche anteilmässig für den Inhalt charakterisieren.

Was sind Topics?

Eine statistische Verteilung von Wörtern. Intuitiv: Jedes Topic ist ein grosser unfairer Würfel mit vielen Seiten, wo jeweils ein Wort drauf steht.

Generative Story von LDA Topic Modeling

Würfle für jedes Dokument anteilmässig mit jedem Topic-Model-Würfel die Wörter, die den Bag-Of-Word des Dokuments ausmachen.

Illustration inkl. parametrisierte Distributionen

LDA als graphisches Modell

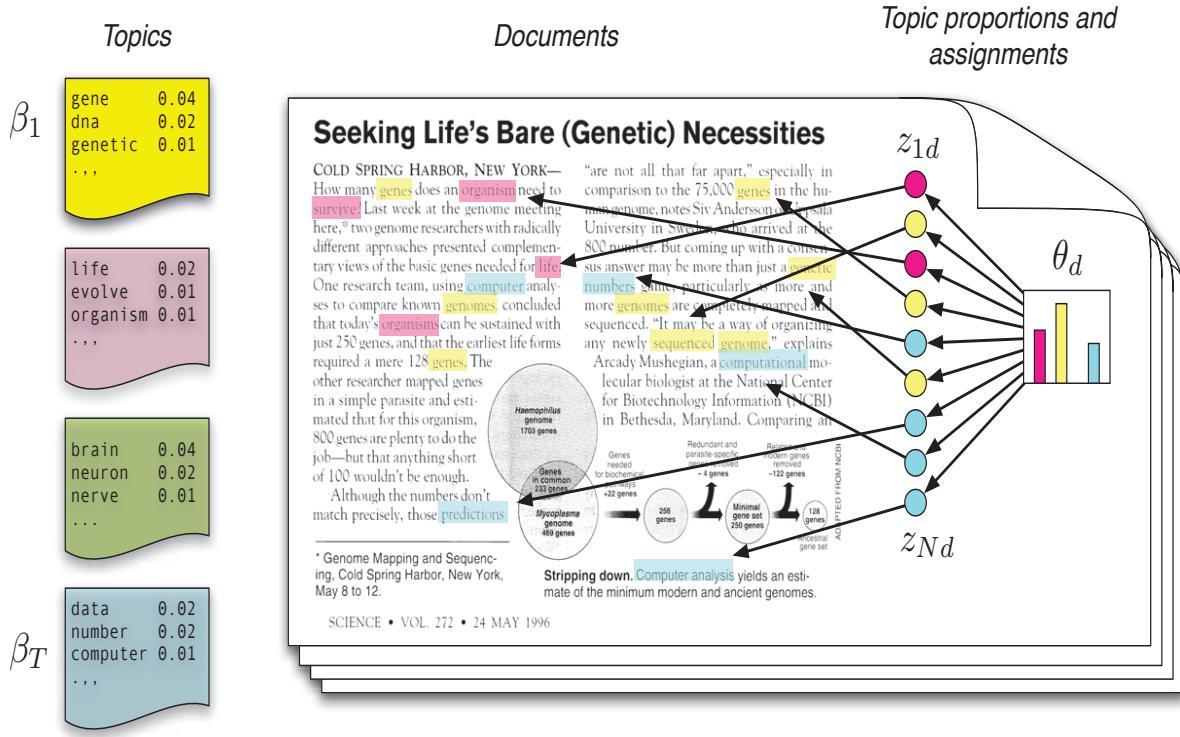
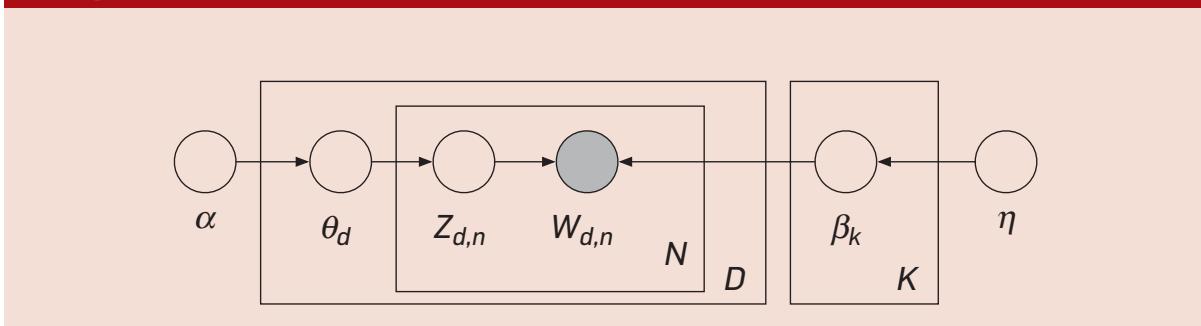


Abbildung 11.1: Zutaten für probabilistisches *Topic Modeling* nach [BLEI 2012, 78]

Figure 4. The graphical model for latent Dirichlet allocation. Each node is a random variable and is labeled according to its role in the generative process (see Figure 1). The hidden nodes—the topic proportions, assignments, and topics—are unshaded. The observed nodes—the words of the documents—are shaded. The rectangles are “plate” notation, which denotes replication. The N plate denotes the collection words within documents; the D plate denotes the collection of documents within the collection.



Frage: Was unterscheidet die Story von Naive-Bayes von Topic-Modeling?

Fazit Probabilistisches Topic Modeling

- Aktuell sehr populäre Methode zur automatischen inhaltlichen Strukturierung grosser Dokumentensammlung
- Ergibt eine Art Soft-Clustering: Dokument gehören anteilmässig zu Themenbereichen

- Topics lassen sich dem Menschen relativ anschaulich als Wortverteilungen darstellen
- Neue Dokumente lassen sich in einem bestehenden Topic-Model einordnen
- Ähnliche Dokumente haben ähnliche Topic-Verteilungen, aber nicht zwingend dieselben Wörter
- Relativ rechenintensiv, aber auf heutiger Hardware gut machbar

11.2 LSA

Wort-Dokument-Matrix: Numerische Vektoren

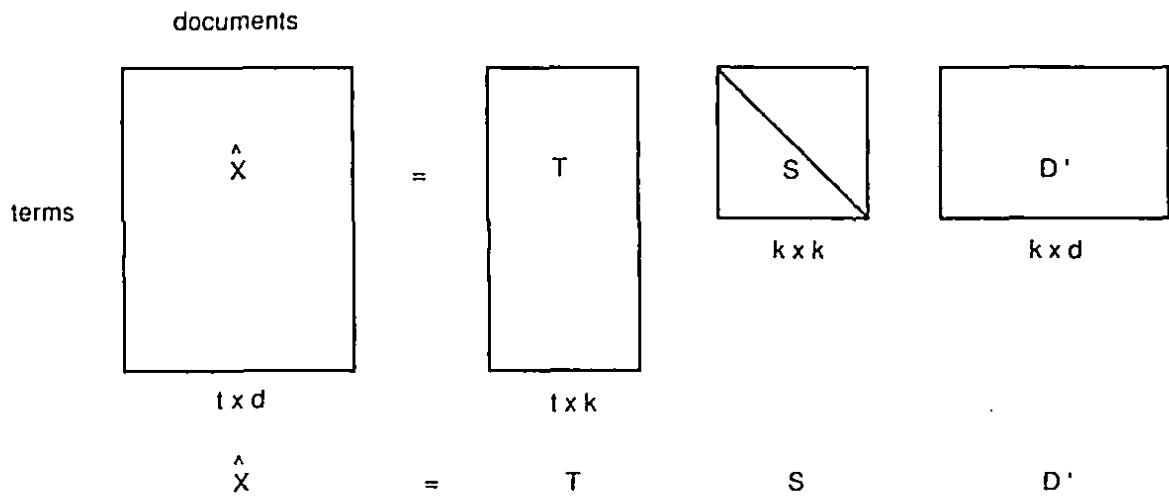
c1:	<i>Human machine interface for Lab ABC computer applications</i>
c2:	<i>A survey of user opinion of computer system response time</i>
c3:	<i>The EPS user interface management system</i>
c4:	<i>System and human system engineering testing of EPS</i>
c5:	<i>Relation of user-perceived response time to error measurement</i>
m1:	<i>The generation of random, binary, unordered trees</i>
m2:	<i>The intersection graph of paths in trees</i>
m3:	<i>Graph minors IV: Widths of trees and well-quasi-ordering</i>
m4:	<i>Graph minors: A survey</i>

Terms	Documents								
	c1	c2	c3	c4	c5	m1	m2	m3	m4
human	1	0	0	1	0	0	0	0	0
interface	1	0	1	0	0	0	0	0	0
computer	1	1	0	0	0	0	0	0	0
user	0	1	1	0	1	0	0	0	0
system	0	1	1	2	0	0	0	0	0
response	0	1	0	0	1	0	0	0	0
time	0	1	0	0	1	0	0	0	0
EPS	0	0	1	1	0	0	0	0	0
survey	0	1	0	0	0	0	0	0	1
trees	0	0	0	0	0	1	1	1	0
graph	0	0	0	0	0	0	1	1	1
minors	0	0	0	0	0	0	0	1	1

Quelle:[DEERWESTER et al. 1990]

Dimensionreduktion: Mathematisch

Zehntausende von Wörtern werden auf 50 bis 500 Konzepte (k Dimensionen) reduziert mit Hilfe von Singular Value Decomposition!



Notationshinweis: $D' = D^T$

Aus der reduzierten Matrix sollte die ursprüngliche möglichst gut rekonstruierbar sein: Reduktion aufs Wesentliche, d.h. orthogonale Spaltenvektoren!

Eigenschaften: $\mathbf{T} \cdot \mathbf{T}^T = I$ und $\mathbf{D} \cdot \mathbf{D}^T = I$ und \mathbf{S} ist eine Diagonalmatrix mit positiven Zahlen nach Zeilen absteigend geordnet

Beispiel für SVD-Dimensionsreduktion: $\mathbf{X} \approx \hat{\mathbf{X}} = \mathbf{T} \cdot \mathbf{S} \cdot \mathbf{D}^T$

X

Approximierungsgröbeit: $\mathbf{X} \approx \hat{\mathbf{X}} = \mathbf{T} \cdot \mathbf{S} \cdot \mathbf{D}^T$

$X =$	$\hat{X} =$
1 0 0 1 0 0 0 0 0	0.16 0.40 0.38 0.47 0.18 -0.05 -0.12 -0.16 -0.09
1 0 1 0 0 0 0 0 0	0.14 0.37 0.33 0.40 0.16 -0.03 -0.07 -0.10 -0.04
1 1 0 0 0 0 0 0 0	0.15 0.51 0.36 0.41 0.24 0.02 0.06 0.09 0.12
0 1 1 0 1 0 0 0 0	0.26 0.84 0.61 0.70 0.39 0.03 0.08 0.12 0.19
0 1 1 2 0 0 0 0 0	0.45 1.23 1.05 1.27 0.56 -0.07 -0.15 -0.21 -0.05
0 1 0 0 1 0 0 0 0	0.16 0.58 0.38 0.42 0.28 0.06 0.13 0.19 0.22
0 1 0 0 1 0 0 0 0	0.16 0.58 0.38 0.42 0.28 0.06 0.13 0.19 0.22
0 0 1 1 0 0 0 0 0	0.22 0.55 0.51 0.63 0.24 -0.07 -0.14 -0.20 -0.11
0 1 0 0 0 0 0 0 1	0.10 0.53 0.23 0.21 0.27 0.14 0.31 0.44 0.42
0 0 0 0 0 1 1 1 0	-0.06 0.23 -0.14 -0.27 0.14 0.24 0.55 0.77 0.66
0 0 0 0 0 0 1 1 1	-0.06 0.34 -0.15 -0.30 0.20 0.31 0.69 0.98 0.85
0 0 0 0 0 0 0 1 1	-0.04 0.25 -0.10 -0.21 0.15 0.22 0.50 0.71 0.62

Dimensionreduktion: Intuitiv

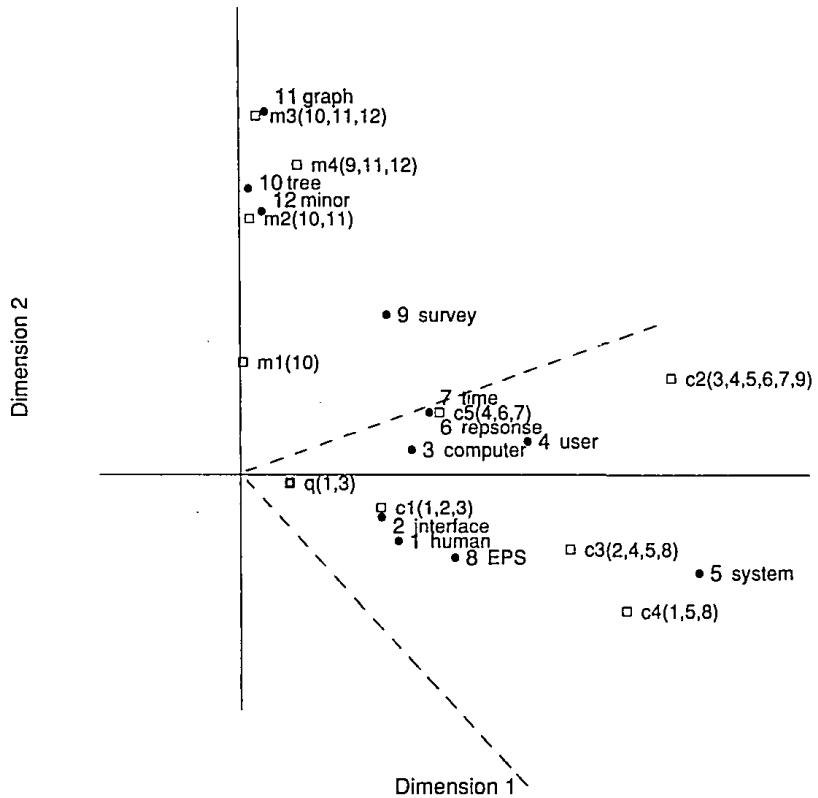
- Abstraktion von Wörtern (Verbalisierungen) zu distributionellen Konzepten (semantischer Raum)
- Wörter, welche oft zusammen vorkommen, werden durch gemeinsame Konzepte repräsentiert im reduzierten Raum
- Jedes Konzept ist eine Dimension in einem Dokument!
- Jedes Konzept ist eine Dimension im semantischen Raum!
- Jedes Dokument hat für jedes Konzept eine gewisse Stärke!
- Ähnliche Dokumente haben für dieselben Konzepte eine ähnliche Stärke!
- Sie liegen nahe zusammen im semantischen Raum!

Kein Wundermittel gegen Mehrdeutigkeit

- Mehrdeutige Wörter können nicht auseinandergehalten werden!
- Aber: In eingegrenzten Anwendungsbereichen ist Wortmehrdeutigkeit seltener!

Ähnlichkeit im semantischen Raum messen

2-D Plot of Terms and Docs from Example



Quelle: [DEERWESTER et al. 1990]

- Jedes Dokument ist ein Punkt im k-dimensionalen semantischen Raum.
- Unähnlichkeit ist der Distanz (ausgedrückt als Kosinus) zwischen 2 Dokumenten
- Similarity = $1 - \text{Cosinus}$
- Wichtig: Auch jedes Wort ist ein Punkt im Semantischen Raum!

Welcher semantische Raum ist optimal?

Viele Möglichkeiten

- Texte, welche benutzt werden
- Vorverarbeitung (Stoppwörter, Stemming)
- “Dokumentlänge” (Sätze, Paragraphen, thematisch kohärente Abschnitte, Dokumente)
- Anzahl Dimensionen im semantischen Raum
- Inhalt der Matrixzellen (Worthäufigkeit, TFIDF, IG, ...)

Evaluierung und Optimierung

- Extrinsische Evaluation über Applikationsleistung
- Intrinsische Evaluation mit intellektuell erstellten Ähnlichkeitsurteilen

11.3 Embeddings

Embeddings: word2vec

- Unsupervisiert und dimensionsreduziertes kontinuierliches Vektormodell
- Jedes Wort besteht aus d Dimensionen (reelle Zahlen), d.h. *word embeddings* $\Phi: w \mapsto \mathbb{R}^d$
- Geordnet (Collobert & Weston 2008): Die Reihenfolge der Kookkurenzen spielt eine Rolle: damit ist syntaktische Information berücksichtigt
- vs Bag-Of-Words, wo die Kookkurenzen als (Multi-)Menge betrachtet werden (CBOW)
- Oder Zwischenformen (Skip-Gram), wo linke und rechte Kontexte separat modelliert werden
- Tutorial mit Tensorflow ²

Embeddings: word2vec, doc2vec, bivec

- **doc2vec**: Embeddings für Sätze oder Paragraphen (unterschiedlicher Länge) für Satzähnlichkeiten [LE und MIKOLOV 2014]
- **bivec**: Embeddings über Sprachgrenzen hinweg mit Hilfe von parallelen Korpora [LUONG et al. 2015]
- Verschiedene x2vec-Modelle mit **multivec**-Implementation [BERARD et al. 2016]

11.3.1 multivec

multivec in Aktion

```
$ multivec-bi --train-src data/europarl-v7.de-en.en --train-trg data/europarl-v7.de-en.de \
--save models/europarl.en-de.bin --threads 18
```

Trainingsparameter

```
dimension: 100
window size: 5
min count: 5
alpha: 0.05
iterations: 5
threads: 18
subsampling: 0.001
skip-gram: false
HS: false
negative: 5
sent vector: false
freeze: false
beta: 1
Training time: 1242.87
```

Python-Modul: Ähnlichste Wörter

²<http://www.tensorflow.org/tutorials/word2vec>

```

from multivec import BiModel

model = BiModel()
model.load('../models/europarl.en-de.bin')

for w, sim in model.closest('love',3):
    print "%s(%2.2f) " % (w, sim),
print
for w, sim in model.closest('hate',3):
    print "%s(%2.2f) " % (w, sim),
print

lieben,(0.72) Liebe(0.68) Liebe,(0.66)
Hasstiraden(0.77) Hass(0.76) Hassreden(0.75)

```

Nutzen von Word-Embeddings für NER [GUO et al. 2014]

Kontinuierliche, distributionelle Wortrepräsentationen

- Word-Embeddings = dimensionsreduzierte kontinuierliche Wortrepräsentationen (word2vec, glove etc.)
- Extrem populäre Methode in letzter Zeit
- Kann als flaches Soft-Clustering betrachtetet werden!

Merkmalifizierung von Embeddings

- Wie kann man numerische Vektoren in CRF einbringen?
- Beste Methode nach [GUO et al. 2014]: Prototypen-Kodierung über Distanz in Wortvektoren

NE Type	Prototypes
B-PER	Mark, Michael, David, Paul
I-PER	Akram, Ahmed, Khan, Younis
B-ORG	Reuters, U.N., Ajax, PSV
I-ORG	Newsroom, Inc, Corp, Party
B-LOC	U.S., Germany, Britain, Australia
I-LOC	States, Republic, Africa, Lanka
B-MISC	Russian, German, French, British
I-MISC	Cup, Open, League, OPEN
O	., ,, the, to

Table 1: Prototypes extracted from the CoNLL-2003 NER training data using NPMI.

NER mit globalen semantischen Merkmalen

Setting	F1
Baseline	83.43
+DenseEmb [†]	86.21
+BinarizedEmb	86.75
+ClusterEmb	86.90
+DistPrototype	87.44
+BinarizedEmb+ClusterEmb	87.56
+BinarizedEmb+DistPrototype	87.46
+ClusterEmb+DistPrototype	88.11
+Brown	87.49
+Brown+ClusterEmb	88.17
+Brown+DistPrototype	88.04
+Brown+ClusterEmb+DistPrototype	88.58
Finkel et al. (2005)	86.86
Krishnan and Manning (2006)	87.24
Ando and Zhang (2005)	89.31
Collobert et al. (2011)	88.67

Table 3: The performance of semi-supervised NER on the CoNLL-2003 test data, using various embedding features. [†] DenseEmb refers to the method used by Turian et al. (2010), i.e., the direct use of the dense and continuous embeddings.

Weiterführendes

- Kapitel 14 “Clustering” in [MANNING und SCHÜTZE 1999]

Literaturverzeichnis

- [BERARD et al. 2016] BERARD, ALEXANDRE, C. SERVAN, O. PIETQUIN und L. BESACIER (2016). *MultiVec: a Multilingual and Multilevel Representation Learning Toolkit for NLP*, In: *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*.
- [BERGER et al. 1996] BERGER, ADAM L., S. A. D. PIETRA und V. D. PIETRA (1996). *A Maximum Entropy Approach to Natural Language Processing*, Computational Linguistics, 22(1):39–71.
- [BIRD et al. 2009] BIRD, STEVEN, E. KLEIN und E. LOPER (2009). *Natural Language Processing with Python*. O'Reilly.
- [BISHOP 2006] BISHOP, CHRISTOPHER M (2006). *Pattern recognition and machine learning*. Springer, New York.
- [BLEI 2012] BLEI, DAVID M. (2012). *Probabilistic topic models*, Commun. ACM, 55(4):77–84, <http://doi.acm.org/10.1145/2133806.2133826>.
- [CARSTENSEN et al. 2004] CARSTENSEN, KAI-UWE, C. EBERT, C. ENDRISS, S. JEKAT, R. KLABUNDE und H. LANGER, Hrsg. (2004). *Computerlinguistik und Sprachtechnologie : Eine Einführung*. Elsevier, München.
- [CHEN und GOODMAN 1998] CHEN, STANLEY und J. GOODMAN (1998). *An Empirical Study of Smoothing Techniques for Language Modeling*, Technical Report TR-10-98, Center for Research in Computing Technology, private Kopie Gute Einführung, mathematisch sauber.
- [CHURCH 1988] CHURCH, KENNETH WARD (1988). *A stochastic parts program and a noun phrase parser for unrestricted text*, In: *Proceedings of the Second Conference on Applied Natural Language Processing*, S. 136–143, Austin, Texas.
- [DAUMÉ III 2015] DAUMÉ III, HAL (2015). *A Course in Machine Learning*, <http://ciml.info>.
- [DAUMÉ III 2008] DAUMÉ III, HAL (2008). *MegaM: Maximum Entropy Model Optimization Package*, ACL Data and Code Repository, ADCR2008C003.
- [DEERWESTER et al. 1990] DEERWESTER, SCOTT C., S. T. DUMAIS, T. K. LANDAUER, G. W. FURNAS und R. A. HARSHMAN (1990). *Indexing by Latent Semantic Analysis*, JASIS, 41(6):391–407, [http://dx.doi.org/10.1002/\(SICI\)1097-4571\(199009\)41:6<391::AID-ASI1>3.0.CO;2-9](http://dx.doi.org/10.1002/(SICI)1097-4571(199009)41:6<391::AID-ASI1>3.0.CO;2-9).

- [DERCZYNSKI et al. 2015] DERCZYNSKI, LEON, S. CHESTER und K. BOGH (2015). *Tune Your Brown Clustering, Please*, In: *Proceedings of the International Conference Recent Advances in Natural Language Processing*, S. 110–117, Hissar, Bulgaria. <http://www.aclweb.org/anthology/R15-1016>.
- [DOMINGOS und PAZZANI 1997] DOMINGOS, PEDRO und M. PAZZANI (1997). *On the Optimality of the Simple Bayesian Classifier under Zero-One Loss*, Machine Learning, 29(2):103–130, <http://dx.doi.org/10.1023/A:1007413511361>.
- [FIELD und MILES 2010] FIELD, ANDY und J. MILES (2010). *Discovering Statistics Using SAS*. SAGE.
- [GUO et al. 2014] GUO, JIANG, W. CHE, H. WANG und T. LIU (2014). *Revisiting Embedding Features for Simple Semi-supervised Learning*, In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, S. 110–120, Doha, Qatar. Association for Computational Linguistics, <http://www.aclweb.org/anthology/D14-1012>.
- [HARRIS 1954] HARRIS, ZELLIG (1954). *Distributional structure*, Word, 23(10):146–162.
- [JAMES et al. 2013] JAMES, GARETH, D. WITTEN, T. HASTIE und R. TIBSHIRANI (2013). *An Introduction to Statistical Learning: with Applications in R*. Springer, Corr. 6th printing 2015 Aufl., <http://www-bcf.usc.edu/~gareth/ISL/>.
- [JURAFSKY und MARTIN 2008] JURAFSKY, DANIEL und J. H. MARTIN (2008). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Prentice-Hall, 2. Aufl.
- [KSCHISCHANG et al. 2001] KSCHISCHANG, FRANK R., B. J. FREY und H. LOELIGER (2001). *Factor graphs and the sum-product algorithm*, IEEE Trans. Information Theory, 47(2):498–519, <http://dx.doi.org/10.1109/18.910572>.
- [LE und MIKOLOV 2014] LE, QUOC V. und T. MIKOLOV (2014). *Distributed Representations of Sentences and Documents*, In: *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, Bd. 32 d. Reihe *JMLR Proceedings*, S. 1188–1196. JMLR.org, <http://jmlr.org/proceedings/papers/v32/le14.html>.
- [LUONG et al. 2015] LUONG, THANG, H. PHAM und D. C. MANNING (2015). *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, Kap. Bilingual Word Representations with Monolingual Quality in Mind, S. 151–159. Association for Computational Linguistics, <http://aclweb.org/anthology/W15-1521>.
- [MALOUF 2002] MALOUF, ROBERT (2002). *A comparison of algorithms for maximum entropy parameter estimation*, In: *Proceedings of CoNLL-2002*, S. 49–55. Taipei, Taiwan.
- [MANNING und SCHÜTZE 1999] MANNING, CHRISTOPHER D. und H. SCHÜTZE (1999). *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, 3rd printing Aufl.
- [MANNING und SCHÜTZE 2000] MANNING, CHRISTOPHER D. und H. SCHÜTZE (2000). *Foundations of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, 3rd printing Aufl.
- [MC CALLUM und NIGAM 1998] MC CALLUM, ANDREW und K. NIGAM (1998). *A Comparison of Event Models for Naive Bayes Text Classification*, In: *AAAI Technical Report WS-98-05*. AAAI, <http://www.aaai.org/Papers/Workshops/1998/WS-98-05/WS98-05-007.pdf>.

- [METSIS et al. 2006] METSIS, VANGELIS, I. ANDROUTSOPoulos und G. PALIOURAS (2006). *Spam Filtering with Naive Bayes – Which Naive Bayes?*, In: *CEAS 2006 – Third Conference on Email and Anti-Spam*. http://nlp.cs.aueb.gr/pubs/ceas2006_paper.pdf.
- [MIKOLOV 2013] MIKOLOV, TOMAS (2013). *Learning Representations of Text using Neural Networks*, <https://drive.google.com/file/d/0B7XkCwpI5KDYRWRnd1RzWXQ2TWc/edit>.
- [MIKOLOV et al. 2011] MIKOLOV, TOMAS, A. DEORAS, S. KOMBRINK, L. BURGET und J. H. CERNOCKY (2011). *Empirical Evaluation and Combination of Advanced Language Modeling Techniques*, In: *Interspeech*. ISCA, <http://research.microsoft.com/apps/pubs/default.aspx?id=175560>.
- [MURPHY 2012] MURPHY, KEVIN P (2012). *Machine Learning: A Probabilistic Perspective*. MIT Press.
- [NIGAM et al. 1999] NIGAM, KAMAL, J. LAFFERTY und A. McCALLUM (1999). *Using Maximum Entropy for Text Classification*.
- [PENNINGTON et al. 2014] PENNINGTON, JEFFREY, R. SOCHER und C. D. MANNING (2014). *GloVe: Global Vectors for Word Representation*, In: *Empirical Methods in Natural Language Processing (EMNLP)*, S. 1532–1543. <http://www.aclweb.org/anthology/D14-1162>.
- [PEREIRA 2000] PEREIRA, FERNANDO (2000). *Formal grammar and information theory: together again?*, Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences, 358(1769):1239–1253, <http://rsta.royalsocietypublishing.org/content/358/1769/1239>.
- [POTTS 2013] POTTS, CHRIS (2013). *Distributional approaches to word meanings*, <http://web.stanford.edu/class/linguist236/materials/ling236-handout-05-09-vsm.pdf>.
- [QUATTINI 2013] QUATTINI, ARIADNA (2013). *Tutorial on Conditional Random Fields for Sequence Prediction*, http://www.cs.upc.edu/~aquattoni/AllMyPapers/crf_tutorial_talk.pdf.
- [ROBINSON 2012] ROBINSON, PETER N (2012). *Parameter Estimation: ML vs. MAP*, <http://www.mi.fu-berlin.de/wiki/pub/ABI/Genomics12/MLvsMAP.pdf>.
- [SAHLGREN 2008] SAHLGREN, M (2008). *The distributional hypothesis*, Italian Journal of Linguistics, 20(1):33–54.
- [SCHMID 1994] SCHMID, HELMUT (1994). *Probabilistic Part-of-Speech Tagging Using Decision Trees*, In: JONES, DANIEL, Hrsg.: *Proceedings of International Conference on New Methods in Language Processing*, Studies in Computational Linguistics, S. 44–49. University of Stuttgart, <http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.28.1139>.
- [SCHNEIDER 2003] SCHNEIDER, KARL-MICHAEL (2003). *A Comparison of Event Models for Naive Bayes Anti-Spam E-Mail Filtering*, In: *10th Conference of the European Chapter of the Association for Computational Linguistics*. <http://aclweb.org/anthology/E03-1059>.
- [SHANNON 1951] SHANNON, C E (1951). *Prediction and Entropy of Printed English*, Bell Systems Technical Journal, 30(January):50–64.
- [SUTTON und McCALLUM 2012] SUTTON, CHARLES A. und A. McCALLUM (2012). *An Introduction to Conditional Random Fields*, Foundations and Trends in Machine Learning, 4(4):267–373.

[TURNEY und PANTEL 2010] TURNEY, PETER D. und P. PANTEL (2010). *From Frequency to Meaning: Vector Space Models of Semantics*, J. Artif. Int. Res., 37(1):141–188, <http://dl.acm.org/citation.cfm?id=1861751.1861756>.

[WESTON und BORDES 2014] WESTON, JASON und A. BORDES (2014). *Embedding Methods for NLP: Part 1: Unsupervised and Supervised Embeddings*, EMNLP Tutorial - October 29, 2014, http://emnlp2014.org/tutorials/8_notes.pdf.

[WITTEN et al. 2011] WITTEN, I. H, E. FRANK und M. A. HALL (2011). *Data mining: practical machine learning tools and techniques*. Morgan Kaufmann, Burlington, MA, 3rd Aufl.

[YU et al. 2010] YU, HSIANG-FU, F.-L. HUANG und C.-J. LIN (2010). *Dual coordinate descent methods for logistic regression and maximum entropy models*, Machine Learning, 85(1):41–75, <http://dx.doi.org/10.1007/s10994-010-5221-8>.

Index

- Alignierungsproblem, 13
- argmax, 52
- Bag-of-Words, 50
- Bagging, 62
- Beta-Distribution, 74
- bivec, 119
- Boosting, 63
- Bootstrap-Sampling, 62
- Brown-Cluster-IDs, 111
- CBOW, 43
- Classification, Binary, 49
- Classification, Multiclass, 49
- Closed Form, 72
- Clusterähnlichkeit, Complete Link, 110
- Clusterähnlichkeit, Group Average, 110
- Clusterähnlichkeit, Single Link, 110
- Clustering, Hard, 112
- Clustering, Hierarchisch, 110
- Clustering, K-Means, 12, 109
- Clustering, Soft, 112
- Confidence Intervall, 18
- Conjugate Prior, 75
- Cosinus-Ähnlichkeit, 117
- CRF, 98, 103
- Decoding, 107
- Dendogramm, 110
- Deviance, 18
- Dimensionsreduktion, 117
- Discriminativ, 98
- Discriminative, 82
- Distributionalismus, 40
- Distributionen, 68
- doc2vec, 119
- Effect Size, 18
- Emeddings, 119
- Entropie, 27, 32
- Entropie, Relative, 30
- Entropy, Joint, 31
- Entscheidungsbaum, 11, 53
- Entscheidungsgrenzen, 94
- Erwartungswert, 27
- Erwartungswert, MaxEnt-Merkmalsfunktion, 88
- Evidence, 46
- Feature-Templates, 100
- Feeedforward Neural Net LM, 41
- Fitting, 19
- Gaussian Prior, 92
- Generativ, 98
- Generative, 82
- Generative Story, Naive Bayes, 83
- Gradient Ascent, 71
- Gruppierungsproblem, 12
- HMM, 98
- Improved Iterative Scaling, 92
- Impurity, 58
- Informationsgehalt, 27
- Interpretierbarkeit, 49
- KL-Divergenz, 30
- Klassifikation, formal, 46
- Klassifikation, Probabilistisch, 49
- Klassifikatoren, 94
- KNN, 67
- Kreuzentropie, 29, 38
- Lernen, maschinell, 9
- Log-Linear Modelling, 81
- Log-lineares Modell, Anwendungen, 93
- Log-odds, 81
- Logistic Regression, 12
- LSA, 116
- MAP-Dekodierung, 85
- Maximal Margin Classifier, 94
- Maximum-A-Posteriori, 72
- Maximum-A-Posteriori-Klasse, 85
- Maximum-Entropy, Idee, 84

Maximum-Entropy, MAP, 92
 Maximum-Entropy-Classification, 81
 Maximum-Entropy-Modell, 12
 ME-Indikatorfunktion, 85
 Mean, Population, 18
 Mean, Sample, 18
 Merkmalstypen, 49
 Merkmalsvektoren, 85
 MI, 30
 MLE, 21, 70
 Modell, 18
 Mutual Information, 30

 N-Gramm-Sprachmodell, 8
 N-Gramm-Wahrscheinlichkeiten, 34
 Naive Bayes, 51, 98
 Naive Bayes, Gaussian, 67
 Naive Bayes, Problems, 26
 NLP-Methode, statistisch, 8
 Non-Parametric, 66

 Objective Function, 105
 Odds, 81
 OOV, 35
 Optimierungsproblem, 13
 Outcome, 46
 Overfitting, 15

 Parameter, 19, 68
 Parametric, 66
 Perplexität, 34, 39
 Perplexity, 31
 Probability Mass Function, 24
 Prototypen-Kodierung, 120
 Pruning, 61

 Random Forests, 62
 Randverteilung, 26
 Recurrent Neural Net LM, 42
 Regellernen, 50
 Regression, Linear, 80
 Regression, linear, 47
 Regression, Logistitic, 80
 Regularisierung, L2, 106

 Shannon Game, 32
 Skip-Gram, 43
 Smoothing, Backoff, 34
 Smoothing, Good-Turing, 34
 Smoothing, Interpolation, 34
 Smoothing, Kneser-Ney, 35

 Smoothing, Laplace, 34
 Split, Recursive, 56
 Sprachmodell, aggregiert, 7
 Sprachmodell, Kontinuierlich, 39
 Sprachmodell-Evaluation, 38
 Standard Deviation, 18
 Stoppkriterium, 60
 Sum of Squared Errors, 18
 Support Vector Classifier, 94
 Support Vector Machine, 11, 94
 SVD, 116
 System, datenbasiert, 15
 System, regelbasiert, 15

 Topic Modeling, 112
 Training, 9

 Unabhängigkeitssannahme, 21
 Underfitting, 15

 Variable, Dependent, 46
 Variable, Independent, 46
 Variance, 18
 Vektorraum, 43
 Vorhersage, 9
 Vorhersagetypen, 14

 Wahrscheinlichkeit, Bedingt, 25
 Wahrscheinlichkeitsraum, Diskrete, 22
 Wahrscheinlichkeitsraum, Kontinuierlich, 23
 Weights, 19
 word2vec, 119
 Worthypothesengraph, 8

 Zufallsvariable, 24, 68
 Zufallsvariable, Bernoulli, 69