Maschinelles Lernen

Prof. Dr. Katharina Morik

22. April 2009

Inhaltsverzeichnis

1 Anwendungen maschinellen Lernens

Bekannte Anwendungen

- Google ordnet die Suchergebnisse nach der Anzahl der auf sie verweisenden Hyperlinks an.
- Amazon empfiehlt einem Kunden, der A gekauft hat, das Produkt B, weil alle (viele) Kunden, die A kauften, auch B kauften.
- Die Post sortiert handbeschriftete Briefe per Schrifterkennung.
- Firmen ordnen ihre eingehende Post automatisch der zuständigen Abteilung zu.
- Aktienkurse oder Verkaufszahlen werden vorhergesagt.

Interesse an Anwendungen

- Business Reporting soll automatisiert werden. On-line Analytical Processing beantwortet nur einfache Fragen. Zusätzlich sollen Vorhersagen getroffen werden.
- Wissenschaftliche Daten sind so umfangreich, dass Menschen sie nicht mehr analysieren können, um Gesetzmäßigkeiten zu entdecken.
- Geräte sollen besser gesteuert werden, indem aus den log-Dateien gelernt wird.
- Roboter sollen sich besser an menschliche Umgebung und Kommunikation anpassen.
- Das Internet soll nicht nur gesamte Dokumente liefern, sondern Fragen beantworten.
- Multimedia-Daten sollen personalisiert strukturiert und gezielter zugreifbar sein.

2 Lernen beim Menschen

Was ist Lernen beim Menschen?
Menschen lernen durch:

- Auswendig lernen.
- Einüben. (Fertigkeiten)
- Logisch schließen:
  - Alle Menschen sind sterblich.
    Sokrates ist ein Mensch.
    Sokrates ist sterblich. (Deduktion)
  - Sokrates, Uta, Udo, Veronika, Volker, ... sind Menschen.
    Sokrates, Uta, Udo, Veronika, Volker, ... sind sterblich.
    Alle Menschen sind sterblich. (Induktion)
- Begriffe bilden.
- Grammatiken lernen.
- Gesetze entdecken.
- Theorien entwickeln. (Wissen)

2.1 Begriffsbildung

Begriffsbildung

- Eins von diesen Dingen gehört nicht zu den anderen!

Clustering Kategorisierung

- Alle Beobachtungen, die sich ähneln, werden zu einer Gruppe zusammengefasst.
- Auf diese Weise strukturiert man die vielen Beobachtungen.
- Von den vielen Merkmalen wählt man zur Ähnlichkeitsbestimmung eine möglichst kleine Anzahl aus.
- Die ausgewählten Merkmale sind immer erkennbar (operational).

Die Kategorisierung ordnet jede Beobachtung mindestens einer Gruppe zu. Die Gruppen können sich überlappen. Menschen kategorisieren immer, ob sie wollen oder nicht! Es ist ein unbewusster kognitiver Prozess.

Einige Gründe für die Kategorisierung

- Handlungen können nicht auf der Gesamtheit der Beobachtungen ausgeführt werden. Menschen haben eine beschränkte Wahrnehmungs- und Aktionskapazität.
  - Menschen können nur 5-7 kognitive Objekte gleichzeitig beachten (ansehen, hören, merken).
  - Hände können nur eine begrenzte Anzahl physikalischer Objekte fassen.
  - Deshalb muss eine große Grundgesamtheit für Menschen in kleine, wahrnehmbare, handhabbare Untermengen aufgeteilt werden.
- Es gibt schon ein Wort dafür.
  - Jemand nennt ein Objekt x *Tasse*.
  - Alle Objekte, die von jemandem als *Tasse* bezeichnet wurden, gehören in eine Gruppe mit dem Titel *Tasse*.

Positive Beispiele

- Dies sind Tassen.

Negative Beispiele

- Dies sind keine Tassen.

Klassifikation

- Eine Funktion ordnet einer Wahrnehmung eine Klasse zu.
  - Dem Wort *Tasse* entspricht eine Erkennungsfunktion, die jeder Wahrnehmung die Klasse *Tasse* oder *Nicht-Tasse* zuordnet.
- Die einfachste Funktion ist das Aufzählen. Dies begrenzt aber die Klassifikation auf bereits gesehene Objekte.
- Als Wissenschaftler verwenden Menschen gern numerische Funktionen.
- Besonders verständlich sind logische Funktionen. Dies sind meist Definitionen.

Definitionen

Eine Definition ist eine Erkennungs- und Ergänzungsfunktion (hinreichende und notwendige Bedingungen).

Definition: Eine Tasse ist ein Behälter mit flachem Boden und einem Henkel an der Seite.

Erkennungsfunktion: Aha, konkav und undurchlässig, flacher Boden, Henkel an der Seite – eine Tasse!
$konkav(x), opak(x), hatBoden(x, y), flach(y), hatHenkel(x, z) \rightarrow tasse(x)$

Ergänzungsfunktion: Kann ich eine Tasse hinstellen? – Ja, denn eine Tasse hat einen flachen Boden und Objekte mit flachem Boden stehen sicher!
$tasse(x) \rightarrow kannStehen(x)$

Ein Begriff erleichtert oft die Definition anderer Begriffe.

- Wer nicht weiß, was ein *Boden* oder ein *Henkel* ist, hat Probleme, eine *Tasse* zu definieren.
- Die Definition für *Boden* und *Henkel*
  $... \rightarrow hatBoden(x, y)$
  $... \rightarrow hatHenkel(x, z)$
  erlaubt die Definition von *Tasse*:
  $konkav(x), opak(x), hatBoden(x, y), flach(y), hatHenkel(x, z) \rightarrow tasse(x)$

Menschliches Lernen

- Die kognitive Psychologie untersucht das menschliche Lernen.
- Die Entwicklungspsychologie untersucht das Lernen über die Alterstufen hinweg [4].
- Einflüsse auf das Lernen werden untersucht:
  - Reihenfolge der Beobachtungen oder Lernschritte [3]
  - Umgebung beim Lernen [1]
  - Soziale Zusammenarbeit (kollaboratives Lernen) [2]
  - ...

Literatur zu menschlichem Lernen

Literatur

[1] J. Bliss, R. Saljo, and P. Light, editors. *Learning Sites – Social and technological Resources for Learning.*

[2] P. Dillenbourg, editor. *Collaborative Learning – Cognitive and Computational Approaches.* Pergamon Press, 1998.

[3] Frank E. Ritter, Erno Lehtinen, Josef Nerb, and Timothy O'Shea, editors. *In Order to Learn – How the Sequence of Topics Influences Learning.* Oxford University Press, 2007.

[4] R.S. Siegler. *Children's Thinking.* Prentice-Hall, 2nd edition, 1991.

3 Maschinelle Lernaufgaben

Maschinelles Lernen – generische Aufgabe

Population: Eine Menge von Objekten, um die es geht.

Merkmale: Eine Menge von Merkmalen (quantitativ oder qualitativ) beschreibt die Objekte.

Ausgabe: Ein quantitativer Wert (Messwert) oder ein qualitativer (label, z.B. *Tasse*) gehört zu jeder Beobachtung.

Ein Lernverfahren findet eine Funktion, die Objekten einen Ausgabewert zuordnet. Oft *minimiert* die Funktion einen *Fehler*.

Modell: Das Lernergebnis (die gelernte Funktion) wird auch als *Modell* bezeichnet.

Notation

- Der Raum möglicher Beobachtungen wird als p-dimensionale Zufallsvariable X geschrieben.
- Jede Dimension der Beobachtungen wird als X_i notiert (Merkmal).
- Die einzelnen Beobachtungen werden als $\vec{x_1}, ..., \vec{x_N}$ notiert.
- Die Zufallsvariable Y ist die Ausgabe (label).
- N Beobachtungen von Vektoren mit p Komponenten ergeben also eine $N \times p$-Matrix.

ExampleSet

Meta Data View | Data View | Plot View

ExampleSet (14 examples, 1 special attribute, 4 regular attributes)

row no.	Play	Outlook	Temperat...	Humidity	Wind
1	no	sunny	85	85	false
2	no	sunny	80	90	true
3	yes	overcast	83	78	false
4	yes	rain	70	96	false
5	yes	rain	68	80	false
6	no	rain	65	70	true
7	yes	overcast	64	65	true
8	no	sunny	72	95	false
9	yes	sunny	69	70	false
10	yes	rain	75	80	false
11	yes	sunny	75	70	true
12	yes	overcast	72	90	true
13	yes	overcast	81	75	false
14	no	rain	71	80	true

Lernaufgabe Clustering
Gegeben

- eine Menge $\mathcal{T} = \{\vec{x_1}, ..., \vec{x_N}\} \subset X$ von Beobachtungen,
- eine Anzahl K zu findender Gruppen $C_1, ..., C_K$,
- eine Abstandsfunktion $d(\vec{x}, \vec{x'})$ und
- eine Qualitätsfunktion.

Finde

- Gruppen $C_1, ..., C_K$, so dass
- alle $\vec{x} \in X$ einer Gruppe zugeordnet sind und
- die Qualitätsfunktion optimiert wird: Der Abstand zwischen Beobachtungen der selben Gruppe soll minimal sein; der Abstand zwischen den Gruppen soll maximal sein.

Lernaufgabe Klassifikation
Gegeben

- Klassen Y, oft $y \in \{+1, -1\}$,
- eine Menge $\mathcal{T} = \{(\vec{x_1}, y_1), ..., (\vec{x_N}, y_N)\} \subset X \times Y$ von Beispielen,
- eine Qualitätsfunktion.

Finde

- eine Funktion $f : X \to Y$, die die Qualitätsfunktion optimiert.

Lernaufgabe Regression
Gegeben

- Zielwerte Y mit Werten $y \in \mathcal{R}$,
- eine Menge $\mathcal{T} = \{(\vec{x_1}, y_1), ..., (\vec{x_N}, y_N)\} \subset X \times Y$ von Beispielen,
- eine Qualitätsfunktion.

Finde

- eine Funktion $f : X \to Y$, die die Qualitätsfunktion optimiert.

Funktionsapproximation
Wir schätzen die wahre, den Beispielen unterliegende Funktion. Gegeben

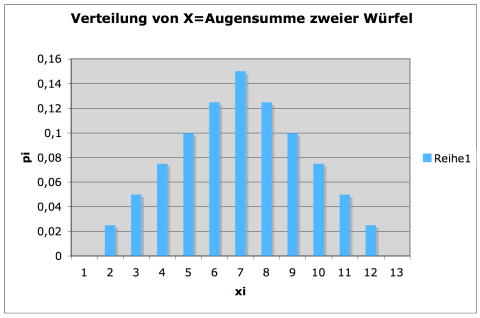
- eine Menge von Beispielen $\mathcal{T} = \{(\vec{x_1}, y_1), ..., (\vec{x_N}, y_N)\} \subset X \times Y$,
- eine Klasse zulässiger Funktionen f_θ (Hypothesensprache),
- eine Qualitätsfunktion,
- eine feste, unbekannte Wahrscheinlichkeitsverteilung $P(X)$.

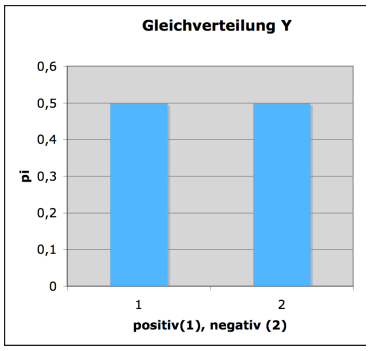
Finde

- eine Funktion $f_\theta : X \to Y$, die die Qualitätsfunktion optimiert.

Zur Erinnerung: Verteilung

Eine Zufallsvariable X heißt *diskret*, wenn sie nur endlich oder abzählbar unendlich viele Werte $x_1, ..., x_m$ annehmen kann. Zu jedem Wert gehört ein Ereignis, das mit der Wahrscheinlichkeit $P(X = x_i)$ eintreten kann. Die Realisationen x_i gemeinsam mit den zugehörigen Wahrscheinlichkeiten heißen *(Wahrscheinlichkeits-)Verteilung* von X.





Verteilungsfunktion

Sei X eine diskrete oder stetige Zufallsvariable. Die Funktion
$D(x) = P(X \leq x), x \in \mathcal{R}$
heißt *Verteilungsfunktion* von X.
Bei diskreten Zufallsvariablen gilt: $D(x) = \sum_{i:x_i \leq x} p_i$
Eine Zufallsvariable heißt *stetige Zufallsvariable*, wenn ihre Verteilungsfunktion stetig ist.

Dichtefunktion

Die Ableitung $D'(x)$ wird *Dichtefunktion* genannt. Umgekehrt erhält man die Verteilungsfunktion durch Integration der Dichtefunktion: $D(x) = \int_{-\infty}^{x} h(t)dt$
Funktionen, die eine Dichte haben, sind absolut stetig.
Die Gesamtfläche unter dem Graphen von h ist gleich 1.

Wenn wir die Verteilung kennen, können wir eine gute Prognose machen!

- Wenn wir wissen, dass $p_i = 0,01$ ist, dann ist es nicht so schlimm, wenn wir uns bei x_i irren – wir irren uns dann selten.
- Wenn wir wissen, dass $P(Y = +1) = 0,99$ ist, dann sagen wir immer $+1$ voraus und sind in 99% der Fälle richtig. Wir haben nur ein Risiko von 1%, uns zu irren.

Qualitätsfunktion – Fehlerfunktion

Fehlerrisiko:

$$R(Y, f(X)) = \sum_{i=1}^{N} Q(y_i, \vec{x_i}) p(\vec{x_i}) \tag{1}$$

wobei $p(\vec{x_i})$ die Wahrscheinlichkeit ist, dass das Beispiel $\vec{x_i}$ aus X gezogen wird.

Mittlerer Quadratischer Fehler:

$$MSE(Y, f(X)) = \frac{1}{N} \sum_{i=1}^{N} (y_i - f(\vec{x_i}))^2 \tag{2}$$

Mittlerer 0-1-Verlust: $Q(Y, f(X)) = \frac{1}{N} \sum_{i=1}^{N} Q(\vec{x_i}, f)$, wobei

$$Q(y_i, f(\vec{x_i})) = \begin{cases} 0, falls & f(\vec{x_i}) = y \\ 1, falls & f(\vec{x_i}) \neq y \end{cases}$$

Problem

- Wir haben nur eine endliche Menge von Beispielen. Alle Funktionen, deren Werte durch die Beispiele verlaufen, haben einen kleinen Fehler.
- Wir wollen aber für *alle* Beobachtungen das richtige y voraussagen. Dann sind nicht mehr alle Funktionen, die auf die Beispiele gepasst haben, gut.
- Wir kennen nicht die wahre Verteilung der Beispiele.
- Wie beurteilen wir da die Qualität unseres Lernergebnisses?

Lern- und Testmenge
Wir teilen die Daten, die wir haben, auf:

Lernmenge: Einen Teil der Daten übergeben wir unserem Lernalgorithmus. Daraus lernt er seine Funktion $f(x) = \hat{y}$.

Testmenge: Bei den restlichen Daten vergleichen wir $\hat{y}$ mit y.

Aufteilung in Lern- und Testmenge

- Vielleicht haben wir zufällig aus lauter Ausnahmen gelernt und testen dann an den normalen Fällen. Um das zu vermeiden, verändern wir die Aufteilung mehrfach.

  leave-one-out: Der Algorithmus lernt aus $N - 1$ Beispielen und testet auf dem ausgelassenen. Dies wird N mal gemacht, die Fehler addiert.

- Aus Zeitgründen wollen wir den Algorithmus nicht zu oft anwenden.

  Kreuzvalidierung: Die Lernmenge wird zufällig in n Mengen aufgeteilt. Der Algorithmus lernt aus $n - 1$ Mengen und testet auf der ausgelassenen Menge. Dies wird n mal gemacht.

Kreuzvalidierung

- Man teile alle verfügbaren Beispiele in n Mengen auf. z.B. $n = 10$.
- Für i=1 bis i=n:
  - Wähle die i-te Menge als Testmenge,
  - die restlichen $n - 1$ Mengen als Lernmenge.
  - Messe die Qualität auf der Testmenge.
- Bilde das Mittel der gemessenen Qualität über allen n Lernläufen. Das Ergebnis gibt die Qualität des Lernergebnisses an.

Fragestellungen des maschinellen Lernens

- Welche Zusicherungen kann ich meinen Kunden geben? (Fehlerschranken)
- Wieviele Beispiele brauche ich?
- Welche Eigenschaften sollen die Beispiele haben, um gut vorherzusagen und wie finde (erzeuge) ich sie?
- Welche Modellklasse soll ich wählen?
- Welcher Algorithmus wird mit vielen Beispielen und vielen Dimensionen in kurzer Zeit fertig?

Zusammenfassung

Was wissen Sie jetzt?

- Sie haben Clustering (Kategorisierung) und Klassifikation als menschliches Lernen gesehen.
- Die Lernaufgaben *Clustering, Klassifikation, Regression* haben Sie auch als Aufgaben des maschinellen Lernens gesehen.
- Sie wissen, was die *Kreuzvalidierung* ist.

Was wissen Sie noch nicht?

- Es gibt viele verschiedene *Modellklassen*. Damit werden die Lernaufgaben spezialisiert.
- Es gibt unterschiedliche *Qualitätsfunktionen*. Damit werden die Lernaufgaben als Optimierungsaufgaben definiert.
- Die *Algorithmen* zur Lösung der Lernaufgaben werden Sie in der Vorlesung kennenlernen und ihre Kernmethoden in den Übungen *selbst implementieren*.

4 Themen, Übungen, Scheine

Themen

- k neirest Neighbor und least squares und das Problem von *bias* und *variance*
- Entscheidungsbäume
- naive Bayes
- logistische Regression
- Stützvektormethode (SVM) und strukturelle Risikominimierung
- K-Means Clustering
- verteiltes Clustering
- Subgruppenentdeckung (KBS) und Boosting
- Merkmalsextraktion und sogar lernende Merkmalsextraktion

Grundidee der Vorlesung

Die Vorlesung behandelt die Themen unter drei Aspekten:

- Theorie: abstrakte Darstellung der Lernaufgabe, ihrer Annahmen, Eigenschaften. Dies gründet sich auf die statistische Lerntheorie [2]. Als Mathe-Buch kann man dazu verwenden [3] und [1].
- Algorithmik: wie löst man nun also die Lernaufgabe?
- Praxis: Algorithmen werden in Java programmiert – zum Teil selbst in den Übungen.

Übungen

Christian Bockermann betreut die Übungen und steht auch für Fragen zur Verfügung.
Wir verwenden das System RapidMiner und können damit

- (fast) alle Lernverfahren und Transformationen der Daten durchführen
- den Kern bestimmter Lernverfahren selbst implementieren und in der RapidMiner-Umgebung ablaufen lassen.

Durch das eigene Implementieren in Java wird die Theorie mit eigener Praxis verbunden. So versteht man sie besser, behält sie auch und besteht sehr gut eine Prüfung!

Wofür bekommen Sie einen Schein?

- Kommen Sie in jede Vorlesung – dann können Sie auch das Tempo bestimmen und Fragen stellen.
- Gehen Sie in die Übungsgruppe!
- Lösen Sie jede Übungsaufgabe: Werden 80% der Punkte erreicht, bekommt man einen Schein.
- Nutzen Sie die Vorlesung/Übung zur Vorbereitung auf eine Fachprüfung!

Wir sehen uns...
In der ersten Übung wird RapidMiner vorgestellt. Sie findet statt:
Am Donnerstag 16.10.2008
In GB IV Raum 113

Literatur

Literatur

[1] Gerald Farin and Dianne Hansford. *Lineare Algebra – Ein geometrischer Zugang.* Springer, 2003.

[2] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer series in statistics. Springer, New York, USA, 2001.

[3] Gerald Teschl and Susanne Teschl. *Mathematik für Informatiker.* Springer, 2006.

5 Lineare Modelle zur Klassifikation und Regression

Grundlagen

Sei $X = \{X_1, \ldots, X_p\}$ eine Menge von Zufallsvariablen und $Y \neq \emptyset$ eine Menge. Ein *Beispiel* (oder *Beobachtung*) $\vec{x}$ ist ein konkreter p-dimensionaler Vektor über diesen Zufallsvariablen. Eine *Menge von n Beispielen* $\mathbf{X} = \{\vec{x}_1, \ldots, \vec{x}_N\}$ können wir dann als $(N \times p)$-Matrix auffassen:

$$\mathbf{X} = \begin{pmatrix} x_{1,1} & x_{1,2} & \ldots & x_{1,p} \\ x_{2,1} & \ddots & & \vdots \\ \vdots & & \ddots & \vdots \\ x_{N,1} & x_{N,2} & \ldots & x_{N,p} \end{pmatrix}$$

Dabei entspricht jede Zeile $\vec{x}_i$ der Matrix $\mathbf{X}$ einem Beispiel.

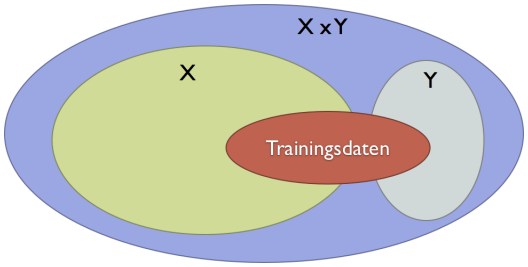
5.1 Klassifikation und Regression

Klassifikation und Regression

Beim *überwachten Lernen* (darum geht es hier), ist zusätzlich zu jeder Beobachtung $\vec{x}$ ein *Label* (*Klasse*) y gegeben, d.h. wir haben Beobachtungen $(\vec{x}, y) \in X \times Y$. Y kann sowohl eine *qualitative*, als auch eine *quantitative* Beschreibung von $\vec{x}$ sein.Für den quantitativen Fall ist z.B. $Y = \mathbb{R}$ und wir versuchen für unbekanntes $\vec{x}$ den Wert y vorherzusagen: *Regression*. Im Falle qualitativer Beschreibungen ist Y eine diskrete Menge und wir nutzen f zur *Klassifikation*.

Lernen auf Trainingsdaten

Wovon gehen wir also aus? Was ist unser Ziel?



- Wir suchen *die wahre Funktion* $f : X \to Y$ mit

$$f(\vec{x}) = y \quad \forall\, (\vec{x}, y) \in X \times Y$$

- Wir haben jedoch nur eine Teilmenge der Beobachtungen gegeben (Trainingsdaten)

Klassifikation und Regression

Auf Grundlage der Trainingsdaten suchen wir eine möglichst gute Annäherung $\hat{f}$ an die *wahre Funktion* f. Die Funktion $\hat{f}$ bezeichnen wir auch als das gelernte *Modell*. Haben wir ein Modell $\hat{f}$ gelernt, so liefert uns dieses Modell mit

$$\hat{y} = \hat{f}\,(\vec{x})$$

für *neue Daten* $\vec{x} \in X$ eine Vorhersage $\hat{y} \in Y$.

Klassifikation und Regression

Im Falle der *Regression* läßt sich so für zuvor unbekannte $\vec{x} \in X$ der Wert

$$\hat{y} = \hat{f}\,(\vec{x})$$

mit $\hat{y} \in \mathbb{R}$ vorhersagen. Dieses Modell $\hat{f}$ läßt sich auch für die Klassifikation nutzen, bei der z.B. $\hat{y} \in \{-1, +1\}$ vorhergesagt werden sollen:

$$\hat{y} = \begin{cases} +1, & \text{falls } \hat{f}(\vec{x}) \geq \theta \\ -1, & \text{sonst} \end{cases}$$

Hier ist θ ein vorgegebener Schwellwert.

Beispiel

Gegeben seien Gewicht (X_1) und Größe (X_2) einiger Personen und ein Label $y \in \{m, w\}$:

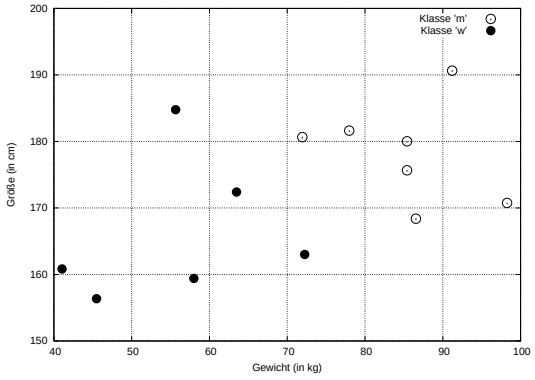
	X_1	X_2	Y
x_1	91	190	m
x_2	60	170	w
x_3	41	160	w
$\vdots$	$\vdots$	$\vdots$	$\vdots$

Die Tabelle enthält die zur Verfügung stehenden Trainingsdaten, also

$$\mathbf{X} = \begin{pmatrix} 91 & 190 \\ 60 & 170 \\ 41 & 160 \\ \vdots & \vdots \end{pmatrix}$$

Es wird nun eine Funktion $\hat{f}$ gesucht, die für neue Daten $\vec{x}$ das Attribut Y (Geschlecht) voraussagt, also

$$\hat{y} = \begin{cases} \text{m, falls } \hat{f}(x) > \theta \\ \text{w, sonst} \end{cases}$$



5.2 Lineare Modelle

Lineare Modelle

Welche Art von Funktionen sind denkbar? *Lineare Funktionen* als einfachste Funktionenklasse:

$$y = f(x) = mx + b \quad \text{Gerade im } \mathbb{R}^2$$

Allerdings betrachten wir als Beispielraum den $\mathbb{R}^p$, d.h. wir brauchen eine verallgemeinerte Form:

$$y = f(\vec{x}) = \sum_{i=1}^{p} \beta_i x_i + \beta_0 \quad \text{mit } \beta_0 \in \mathbb{R}, \vec{x}, \vec{\beta} \in \mathbb{R}^p \tag{3}$$

Die Funktion f wird also durch $\vec{\beta}$ und β_0 festgelegt und sagt uns für ein gegebenes $\vec{x}$ das entsprechende y voraus

Notation, Vereinbarungen

Bei genauerer Betrachtung von Formel (3) läßt sich $\sum_{i=1}^{p} \beta_i x_i$ als Matrizenmultiplikation schreiben, also

$$y = \sum_{i=1}^{p} \beta_i x_i + \beta_0 = \vec{x}^T \vec{\beta} + \beta_0$$

Zur einfacheren Darstellung von f, wird β_0 in den Vektor $\vec{\beta}$ codiert, indem jedes Beispiel $x = (x_1, \ldots, x_p)$ aufgefasst wird als $(p+1)$-dimensionaler Vektor

$$(x_1, \ldots, x_p) \mapsto (1, x_1, \ldots, x_p)$$

Dies ermöglicht die Darstellung von f als:

$$y = f(\vec{x}) = \sum_{i=0}^{p} \beta_i x_i = \vec{x}^T \vec{\beta}$$

Was haben wir nun gemacht?

Wir haben (bei der Beschränkung auf lineare Modelle) nun eine Darstellung für das, was wir *lernen* wollen:

$$y = \hat{f}(\vec{x}) = \vec{x}^T \vec{\beta}$$

Wir haben die Zielfunktion $\hat{f}$ in Abhängigkeit von $\vec{\beta}$ geschrieben und müssen *nur noch* das passende $\vec{\beta}$ finden.

5.3 Geometrie linearer Modelle: Hyperebenen

Veranschaulichung

Bevor wir uns an die Wahl des passenden $\vec{\beta}$ machen, zunächst einige Vorüberlegungen. Betrachten wir dazu die binäre Klassifikation ($Y = \{-1, +1\}$):

- Was passiert dabei eigentlich anschaulich?
- Wie klassifiziert unser $\hat{f}$ die Daten?
- Wie wirkt sich die Wahl von $\vec{\beta}$ aus?

Zur Erinnerung: Hyperebene

Sei $V = \mathbb{R}^p$ ein Vektorraum, dann ist eine Hyperebene H ein $(p-1)$-dimensionaler affiner Untervektorraum. H lässt sich über einen Stützvektor $\vec{a}$ und einen Normalenvektor $\vec{\beta}$ schreiben als

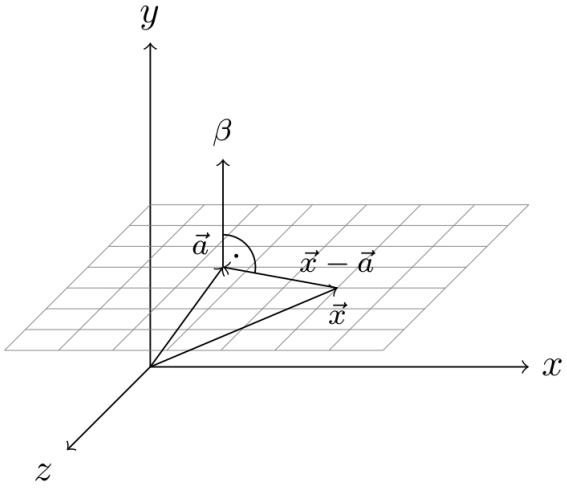
$$H = \left\{ x \in \mathbb{R}^p \mid \vec{\beta}(\vec{x} - \vec{a}) = 0 \right\}$$

Definition 1 (Hesse Normalform)**.** Die Ebenengleichung

$$\vec{\beta}(\vec{x} - \vec{a}) = 0$$

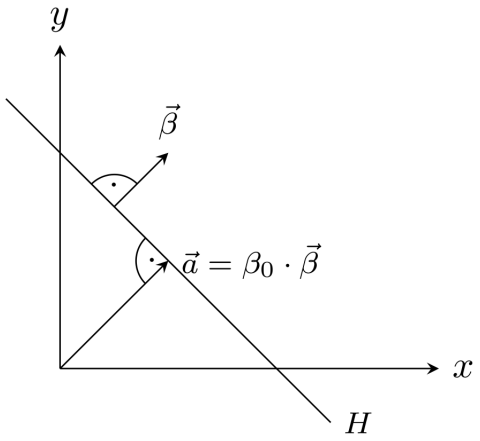
ist in *Hesse Normalform*, falls $||\vec{\beta}|| = 1$.

Beispiel



(Hyper-) Ebene im $\mathbb{R}^3$ mit Normalenvektor $\vec{\beta}$ und Stützvektor $\vec{a}$.

Beispiel



Zur Erinnerung: Euklidsche Länge

Euklidsche Länge oder Norm $\| \vec{x} \| = \sqrt{\sum_{i=1}^{p} x_i^2} = \sqrt{\vec{x}^T \vec{x}}$ weil $\| \vec{x} \|^2 = x_1^2 + ...x_p^2$ (Pythagoras)

Beispiel: $\vec{x} = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}$ $\| \vec{x} \| = \sqrt{1^2 + 2^2 + 3^2} = \sqrt{14}$

Normiert heisst ein Vektor, wenn er die (Euklidsche) Länge 1 hat.

Zur Erinnerung: Skalarprodukt

Skalarprodukt: $\langle \vec{v}, \vec{w} \rangle = \sum_{i=1}^{p} v_i w_i = \vec{v}^T \vec{w}$

Beispiel:

	$\vec{w}$:
	4
	5
	6
$\vec{v}^T$: 1 2 3	$1 \cdot 4 + 2 \cdot 5 + 3 \cdot 6 =$
	32

Winkel zweier Vektoren: $\cos(\measuredangle(\vec{v},\vec{w})) = \frac{\langle \vec{v},\vec{w}\rangle}{\|\vec{v}\|\cdot\|\vec{v}\|}$

Also drückt das Skalarprodukt auch den Winkel aus:

$$\langle \vec{v},\vec{w}\rangle = \| \vec{v} \| \cdot \| \vec{v} \| \cdot \cos(\measuredangle(\vec{v},\vec{w}))$$

Normalisierung

Der Vektor $\vec{\beta}$ soll die Euklidsche Länge 1 haben. Falls das noch nicht der Fall ist, *normalisieren* wir:

$$\vec{\beta} := \frac{\vec{\beta'}}{\| \vec{\beta'} \|}$$

Beispiel

$$\vec{\beta'} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\vec{\beta_0} = 4$$

Die Ebenengleichung ist nicht in Hesse Normalform, weil $\| \vec{\beta'} \| = \sqrt{3} \neq 1$. Wir normalisieren

$$\vec{\beta} = \frac{\vec{\beta'}}{\| \vec{\beta'} \|} = \begin{pmatrix} \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \\ \frac{1}{\sqrt{3}} \end{pmatrix}.$$

Jetzt ist $\beta_0 = \frac{-4}{\sqrt{3}}$ der Abstand der Ebene zum Ursprung.

Der Normalenvektor ist hier $\beta^* = \frac{\beta}{\|\beta\|}$

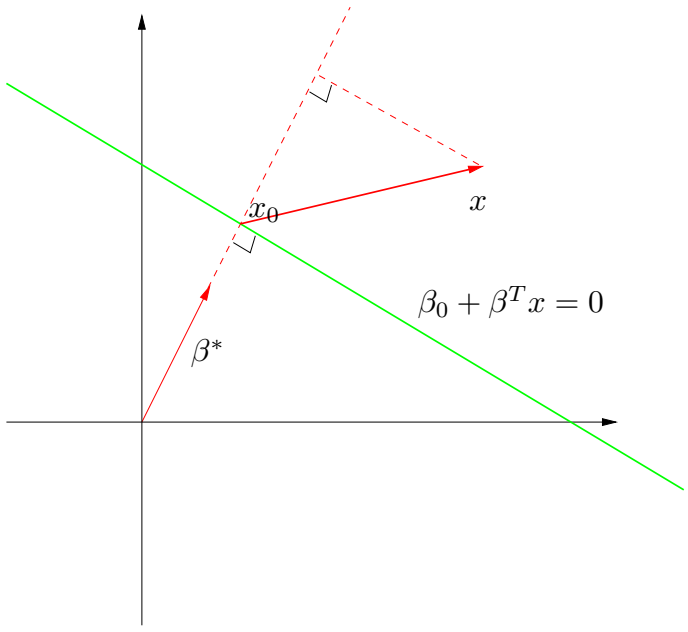
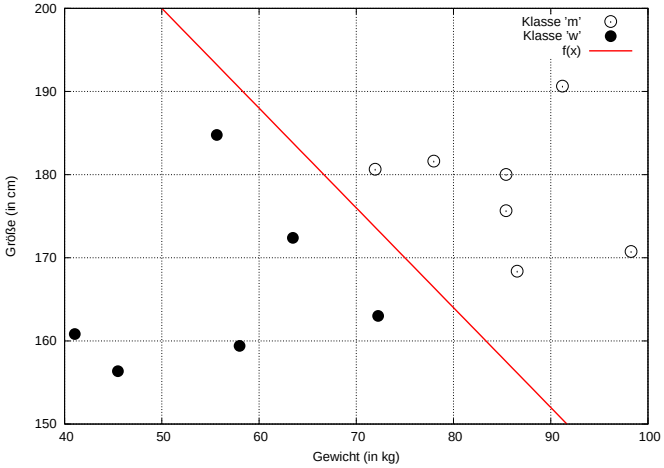


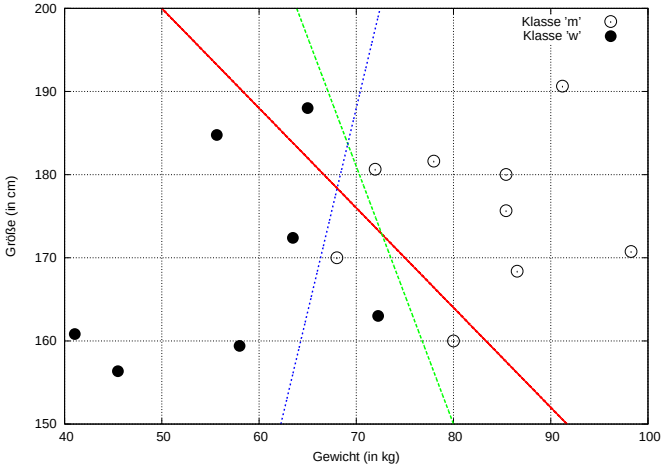
Figure 4.14: *The linear algebra of a hyperplane (affine set).*

Beispiel: Ein mögliches $\vec{\beta}$



$$f(\vec{x}) = \vec{x}^T \hat{\vec{\beta}} \quad \text{mit} \quad \hat{\vec{\beta}} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \end{pmatrix} = \begin{pmatrix} 260 \\ 1 \\ 1.2 \end{pmatrix}$$

Es ist nicht garantiert, dass $\vec{\beta}$ immer paßt!



Modell-Anpassung

Unsere linearen Modelle sind durch $\vec{\beta}$ parametrisiert, das Lernen eines Modells haben wir also auf die Wahl eines $\vec{\beta}$ abgewälzt. Das wirft eine Reihe von Fragen auf:

- Was ist ein gutes $\vec{\beta}$?
- Gibt es ein optimales $\vec{\beta}$?
- Welche Möglichkeiten haben wir, unser Modell zu beurteilen?

Eine Möglichkeit: Berechne den *Trainingsfehler*

$$Err(\vec{\beta}) = \sum_{i=1}^{N} |y_i - \hat{f}(\vec{x}_i)| = \sum_{i=1}^{N} |y_i - x_i^T \vec{\beta}|$$

Modell-Anpassung

Häufig wird als Fehlerfunktion die *quadratische Fehlersumme* (RSS) verwendet:

$$\begin{aligned} RSS(\vec{\beta}) &= \sum_{i=1}^{N} (y_i - \vec{x_i}^T \vec{\beta})^2 \\ &= (\vec{y} - \mathbf{X}\vec{\beta})^T (\vec{y} - \mathbf{X}\vec{\beta}) \end{aligned}$$

Wir wählen jetzt $\vec{\beta}$ derart, dass der Fehler minimiert wird:

$$\min_{\vec{\beta} \in \mathbb{R}^p} RSS(\vec{\beta})$$

$\Rightarrow$ Konvexes Minimierungsproblem!

Minimierung von RSS($\vec{\beta}$)

Um $RSS(\vec{\beta})$ zu minimieren, bilden wir die partielle Ableitung nach $\vec{\beta}$:

$$\frac{\partial RSS(\vec{\beta})}{\partial \beta} = \mathbf{X}^T(\mathbf{y} - \mathbf{X}\vec{\beta})$$

Notwendige Bedingung für die Existenz eines (lokalen) Minimums von RSS ist

$$\frac{\partial RSS(\vec{\beta})}{\partial \beta} = \mathbf{X}^T(\mathbf{y} - \mathbf{X}\vec{\beta}) = 0$$

Ist $\mathbf{X}^T\mathbf{X}$ regulär, so erhalten wir

$$\hat{\vec{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y} \tag{4}$$

Optimales $\hat{\vec{\beta}}$?

Mit Hilfe der Minimierung der (quadratischen) Fehlerfunktion RSS auf unseren Trainingsdaten haben wir ein (bzgl. RSS) optimales $\hat{\vec{\beta}}$ gefunden. Damit liefert unser Modell Voraussagen $\hat{y}$ für $\vec{x} \in X$:

$$\hat{y} = \hat{f}(\vec{x}) = x^T \hat{\vec{\beta}}$$

Sind wir schon fertig?

Schön wär's! Aber drei Gründe sprechen für weitere Arbeit:

1. Es ist nicht immer so einfach, z.B. dann nicht, wenn wir viele Dimensionen haben (Fluch der hohen Dimension).
2. Vielleicht lassen sich die Beispiele nicht linear trennen!
3. Nur den Fehler zu minimieren reicht nicht aus, wir suchen noch nach weiteren Beschränkungen, die zu besseren Lösungen führen.

Also schauen wir uns den Fehler noch einmal genauer an, stoßen auf Bias und Varianz und merken, dass wir noch keine perfekte Lösung haben.

6 Bias-Varianz

Fehler

- Bisher haben wir mit RSS die Fehler einfach summiert.
- Wir wollen aber einbeziehen, wie wahrscheinlich der Fehler ist – vielleicht ist er ja ganz unwahrscheinlich!
- Wann können wir denn einen Fehler erwarten?

6.1 Exkurs: Erwartungswert

Zur Erinnerung: Erwartungswert

Definition 2 (Erwartungswert)**.** Sei X eine *diskrete Zufallsvariable*, mit Werten $x_1, \ldots, x_n$ und p_i die Wahrscheinlichkeit für x_i. Der Erwartungswert von X ist

$$E(X) = \sum_i x_i p_i = \sum_i x_i P(X = x_i)$$

Ist X eine *stetige Zufallsvariable* und f die zugehörige Wahrscheinlichkeitsdichtefunktion, so ist der Erwartungswert von X

$$E(X) = \int_{-\infty}^{\infty} x\, f(x) dx$$

Definition 3 (Erwartungswert (Eigenschaften))**.** Seien X, Y und $X_1, \ldots, X_n$ Zufallsvariablen, dann gilt:

- Der Erwartungswert ist additiv, d.h. es gilt

$$E\left(\sum_{i=1}^{n} X_i\right) = \sum_{i=1}^{n} E(X_i) \tag{5}$$

- Ist $Y = kX + d$, so gilt für den Erwartungswert

$$E(Y) = E(kX + d) = kE(X) + d$$

- Sind die Zufallsvariablen X_i *stochastisch unabhängig*, gilt

$$E\left(\prod_{i=1}^{n} X_i\right) = \prod_{i=1}^{n} E(X_i)$$

Varianz und Standardabweichung

über den Erwartungswert einer Zufallsvariablen X sind mehrere Eigenschaften von X definiert, die helfen, X zu charakterisieren:

Definition 4 (Varianz)**.** Sei X eine Zufallsvariable mit $\mu = E(X)$. Die *Varianz* $Var(X)$ ist definiert als

$$Var(X) := E\left((X - \mu)^2\right).$$

Definition 5 (Standardabweichung)**.** Die *Standardabweichung* σ einer Zufallsvariable X ist definiert als

$$\sigma := \sqrt{Var(X)}$$

Die Varianz wird häufig auch mit σ^2 bezeichnet.

Definition 6 (Verschiebungssatz)**.** Sei X eine Zufallsvariable, für die Varianz gilt

$$Var(X) = E(X - E(X))^2 = E(X^2) - (E(X))^2 \tag{6}$$

Eine weitere Charakteristik, die häufig zur Beschreibung von erwarteten Fehlern verwendet wird, ist die Verzerrung:

Definition 7 (Verzerrung (Bias))**.** Sei X eine Zufallsvariable, dann ist die Verzerrung definiert als der erwartete Schätzfehler für X

$$Bias(\hat{x}) = E(X - \hat{x}) \tag{7}$$

Erwartungswert: Sei X eine diskrete Zufallsvariable, mit Werten $x_1, \ldots, x_n$ und p_i die Wahrscheinlichkeit für x_i. Der Erwartungswert von X ist

$$E(X) = \sum_i x_i p_i = \sum_i x_i P(X = x_i)$$

Ist X eine stetige Zufallsvariable und f die zugehörige Wahrscheinlichkeitsdichtefunktion, so ist der Erwartungswert von X

$$E(X) = \int_{-\infty}^{\infty} x\, f(x) dx$$

Satz: Ist die Dichtefunktion f einer Zufallsvariablen symmetrisch um einen Wert c, also

$$f(c - x) = f(c + x)\ \forall x$$

dann ist $E(X) = c$.

Erwartungswert des Fehlers einer Regression

Erwarteter quadratischer Fehler: Gelernte Funktion $\hat{f} : X \to Y$, der Erwartungswert ihres Fehlers ist:

$$EPE(f) = E(Y - \hat{f}(X))^2 \tag{8}$$

Optimierungsproblem: Wähle $\hat{f}$ so, dass der erwartete Fehler minimiert wird!

$$\hat{f}(x) = argmin_c E_{Y|X}((Y - c)^2 | X = x)$$

Lösung (Regressionsfunktion): $f(x) = E(Y|X = x)$

Bias und Varianz

Zwei Aspekte machen den erwarteten Fehler aus, die Verzerrung (Bias) und die Varianz. Wir wollen den Fehler an einem Testpunkt $x_0 = 0$ angeben und mitteln über allen Trainingsmengen $\mathcal{T}$.

$$\begin{aligned} MSE(\vec{x_0}) &= E_{\mathcal{T}}[y_0 - \hat{y_0}]^2 \\ &= E_{\mathcal{T}}[\hat{y_0} - E_{\mathcal{T}}(\hat{y_0})]^2 + [E_{\mathcal{T}}(\hat{y_0} - y_0)]^2 \\ &= E_{\mathcal{T}}[y_0^2] - E_{\mathcal{T}}[2y_0\hat{y}_0] + E_{\mathcal{T}}[\hat{y}_0^2] \\ &= Var_{\mathcal{T}}(\hat{y_0}) + Bias^2(\hat{y_0}) \end{aligned}$$

Wie das?

Herleitung der Varianz in MSE

Nach dem Verschiebungssatz (6) gilt

$$\begin{aligned} Var_{\mathcal{T}}(y_0) =& E_{\mathcal{T}}[\hat{y}_0^2] - (E_{\mathcal{T}}[\hat{y}_0])^2 \\ \Leftrightarrow& E_{\mathcal{T}}[\hat{y}_0^2] = Var_{\mathcal{T}}(y_0) + (E_{\mathcal{T}}[\hat{y}_0])^2 \end{aligned} \tag{9}$$

Damit folgt

$$\begin{aligned} MSE(\vec{x}_0) &= E_{\mathcal{T}}[y_0 - \hat{y}_0]^2 = E_{\mathcal{T}}\left[y_0^2 - 2y_0\hat{y}_0 + \hat{y}_0^2\right] \\ &\overset{(5)}{=} E_{\mathcal{T}}[y_0^2] - E_{\mathcal{T}}[2y\hat{y}_0] + E_{\mathcal{T}}[\hat{y}_0^2] \\ &\overset{(9)}{=} E_{\mathcal{T}}[y_0^2] - E_{\mathcal{T}}[2y_0\hat{y}_0] + Var_{\mathcal{T}}(\hat{y}_0) + (E_{\mathcal{T}}[\hat{y}_0])^2 \\ &= E_{\mathcal{T}}[y_0^2 - 2y\hat{y}_0 + \hat{y}_0^2] + Var_{\mathcal{T}}(\hat{y}_0) \\ &= E_{\mathcal{T}}[y_0 - \hat{y}_0]^2 + Var_{\mathcal{T}}(\hat{y}_0) \\ &\overset{(7)}{=} Bias^2(\hat{y}_0) + Var_{\mathcal{T}}(\hat{y}_0) \end{aligned}$$

Herleitung des Bias in MSE

Somit gilt

$$MSE(\vec{x_0}) = Var_{\mathcal{T}}(\hat{y_0}) + Bias^2(\hat{y}_0)$$

Die Dekomposition des MSE in Bias und Varianz abstrahiert so, dass wir besser über Modelle nachdenken können.

Frage: Wie wirken sich Bias und Varianz nun auf unsere linearen Modelle aus?

6.2 Bias und Varianz bei linearen Modellen

Erwartungswert des Fehlers bei linearen Modellen

Unter der *Annahme*, dass unsere Beispiele Messfehler enhalten, aber X und Y wirklich linear voneinander abhängen *(Bias=0)*, passen wir das Modell $Y = X^T\beta + \epsilon$ durch Minimieren des quadratischen Fehlers an.

Der erwartete Fehler der $\hat{y}$-Vorhersage für ein beliebiges $\vec{x_0}$ ist:

$$\begin{aligned} EPE(\vec{x_0}) &= E_{y_0|\vec{x_0}} E_{\mathcal{T}}(y_0 - \hat{y_0})^2 \\ &= Var(y_0|\vec{x_0}) + E_{\mathcal{T}}(\hat{y_0} - E_{\mathcal{T}}(y_0))^2 + (E_{\mathcal{T}}(\hat{y_0}) - E_{\mathcal{T}}(y_0))^2 \\ &= Var(y_0|\vec{x_0}) + Var_{\mathcal{T}}(\hat{y_0}) + Bias^2(\hat{y_0}) \\ &= \sigma^2 + E_{\mathcal{T}}(\vec{x_0}^T(\mathbf{X}^T\mathbf{X})^{-1}\vec{x_0}\sigma^2) + 0^2 \end{aligned}$$

Die zusätzliche Varianz kommt durch das Rauschen.

Zusammenhang zwischen Anzahl der Beispiele, der Attribute und erwartetem Fehler

Beziehen wir den Erwartungswert von $\vec{x}$ ein, erhalten wir

$$E_{\vec{x}} EPE(\vec{x}) = \sigma^2(p/N) + \sigma^2$$

Bei kleinem σ^2 und großem N klappt alles auch bei großem p, *wenn* das lineare Modell perfekt passt, d.h. die Beispiele sind linear trennbar.

Fluch der hohen Dimension bei linearen Modellen

- Leider mussten wir annehmen, dass das Modell genau passt, um den erwarteten Fehler klein zu halten.
- Wir wissen aber nicht, welche Art von Funktion gut zu unseren Daten passt! *Modellselektion* ist schwierig!
- Das Modell muss immer komplizierter werden, je mehr Dimensionen es gibt.
- Bei linearen Modellen entspricht die Komplexität des Modells direkt p, denn β hat so viele Komponenten wie p bzw. $p + 1$.

Bias und Varianz bei linearen Modellen

Das lineare Modell wird an die Daten angepasst durch

$$\hat{f}_p(\vec{x}) = \hat{\beta}^T\vec{x}$$

Der Fehler ist dann für ein beliebiges $\vec{x}$:

$$\begin{aligned} Err(\vec{x}) &= E[(Y - \hat{f}_p(\vec{x}))^2 | X = \vec{x}] && (10) \\ &= \sigma_\epsilon^2 + Var(\hat{f}_p(\vec{x}) + \left[f(\vec{x}) - E\hat{f}_p(\vec{x})\right]^2 && (11) \end{aligned}$$

Im Mittel über allen $\vec{x_i}$ ist $Var(\hat{f_p}) = (p/N)\sigma^2$. Modellkomplexität und Varianz hängen bei linearen Modellen direkt zusammen.

Der Trainingsfehler linearer Modelle ist:

$$\frac{1}{N}\sum_{i=1}^{N} Err(x_i) = \sigma_\epsilon^2 + \frac{p}{N}\sigma_\epsilon^2 \frac{1}{N}\sum_{i=1}^{N}\left[f(\vec{x_i}) - E\hat{f}(\vec{x_i})\right]^2 \quad (12)$$

Lineare Modelle

Die grünen und roten Datenpunkte werden durch eine Ebene getrennt.

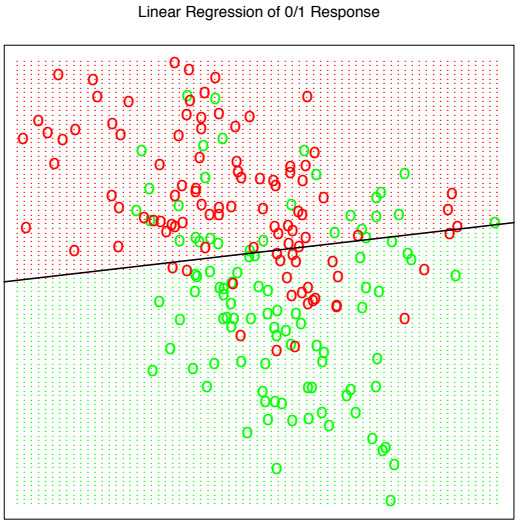


Figure 2.1: *A classification example in two dimensions. The classes are coded as a binary variable—* GREEN $= 0$, RED $= 1$ *—and then fit by linear regression. The line is the decision boundary defined by* $x^T\hat{\beta} = 0.5$. *The red shaded region denotes that part of input space classified as* RED, *while the green region is classified as* GREEN.

Was wissen Sie jetzt?

- Sie haben theoretisch lineare Modelle für Klassifikation und Regression kennengelernt.
- Sie kennen das *Optimierungsproblem* der kleinsten Quadrate RSS für lineare Modelle (Gleichung 4).
- Sie kennen den erwarteten Fehler EPE bei linearen Modellen.
- Sie kennen den *Fluch der hohen Dimension* bei linearen Modellen: Komplexität und Varianz hängen an der Dimension! Der Bias kann sehr hoch sein, wenn die Beispiele tatsächlich nicht linear separierbar sind.

Bis zum nächsten Mal...

- Gehen Sie alle Folien noch einmal in Ruhe durch.
- Vertiefen Sie sich noch einmal in die Ebenengleichung! Die lineare Algebra wird immer wieder vorkommen. Sie können auch die partiellen Ableitungen für RSS mit der Normalengleichung vornehmen.
- Rechnen Sie mal ein Beispiel durch mit Gleichung (4), (12)...
- Diskutieren Sie, warum Bias und Varianz so wichtig sind!
- Probieren Sie lineare Regression in RapidMiner aus!

7 kNN zur Klassifikation, Regression

Globale und lokale Modelle

- Lineare Modelle finden eine trennende Hyperebene.
- Die durch $\vec{\beta}$ angegebene Hyperebene wurde durch *alle* Beispiele bestimmt.
- Deshalb sind lineare Modelle *globale Modelle*.
- Klassifiziert man ein Beispiel nur anhand der Beispiele seiner *Umgebung*, spricht man von einem *lokalen Modell*.
- Nächste Nachbarn sind ein lokales Modell.

Nächste Nachbarn

- Das kNN-Modell betrachtet nur noch die k nächsten Nachbarn eines Beispiel $\vec{x}$:

$$\hat{f}(\vec{x}) = \frac{1}{k} \sum_{\vec{x}_i \in N_k(\vec{x})} y_i \tag{13}$$

- Die Nachbarschaft $N_k(\vec{x})$ wird durch ein Abstandsmaß, z.B. den Euklidschen Abstand bestimmt.
- Es gibt maximal $\frac{N}{k}$ Nachbarschaften und in jeder bestimmen wir den Durchschnitt (13).

Regression und Klassifikation

Gleichung (13) gibt als Regressionsfunktion den Mittelwert der y_i zurück.

$$\hat{f}(\vec{x}) = \frac{1}{k} \sum_{\vec{x}_i \in N_k(\vec{x})} y_i$$

Wie schon bei den linearen Modellen können wir durch einen Schwellwert aus der Regression eine Klassifikation machen:

$$\hat{y} = \begin{cases} 1, & falls\ \hat{f}(\vec{x}) \geq 0,5 \\ 0, & sonst \end{cases}$$

Die grünen und roten Datenpunkte werden in Nachbarschaften gruppiert

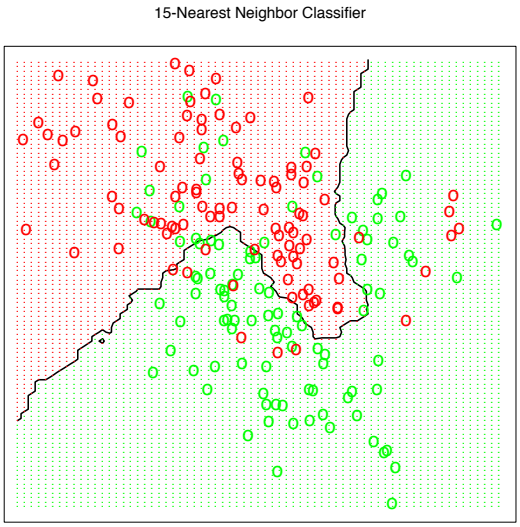


Figure 2.2: *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable* (GREEN $= 0$, RED $= 1$) *and then fit by* 15*-nearest-neighbor averaging as in (2.8). The predicted class is hence chosen by majority vote amongst the* 15*-nearest neighbors.*

Bei k=1 wird nur auswendig gelernt.

- Falls $\vec{x} = \vec{x}' \rightarrow y = y'$, gibt es bei $k = 1$ keinen Trainingsfehler.
- Wenn allein der Trainingsfehler das Optimierungskriterium ist, würden wir stets $k = 1$ nehmen und nur auswendig lernen.
- Vermutlich ergibt das auf den Testdaten einen großen Fehler!

Overfitting

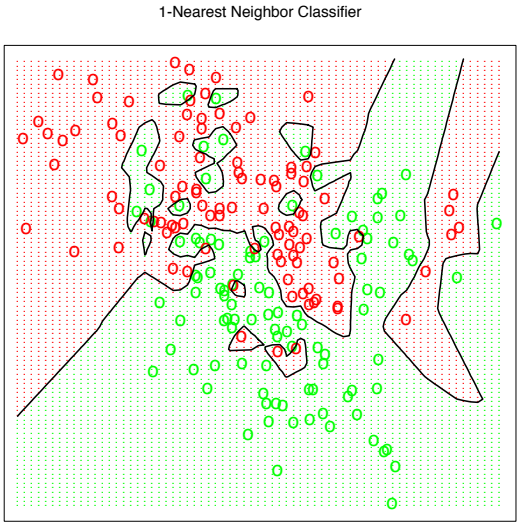


Figure 2.3: *The same classification example in two dimensions as in Figure 2.1. The classes are coded as a binary variable* (GREEN $= 0$, RED $= 1$)*, and then predicted by* 1*-nearest-neighbor classification.*

Training- und Testfehler bei verschiedenen k

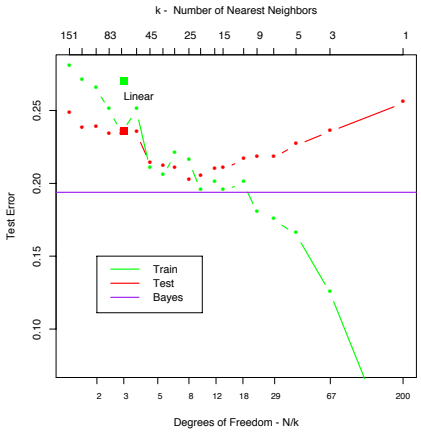


Figure 2.4: *Misclassification curves for the simulation example used in Figures 2.1, 2.2 and 2.3. A single training sample of size* 200 *was used, and a test sample of size* $10,000$*. The red curves are test and the green are training error for* k*-nearest-neighbor classification. The results for linear regression are the bigger green and red dots at three degrees of freedom. The purple line is the optimal Bayes Error Rate.*

7.1 Bias und Varianz bei kNN

Erwartungswert von Y bei k Nächsten Nachbarn

- Der *Erwartungswert* von Y, $E(Y) = \sum_{i=1}^{N} y_i p_i$, geht bei linearen Modellen in den Fehler ein: $EPE(\hat{f}) = E(Y - \hat{f}(X))^2$.
- kNN verwendet den Erwartungswert von Y direkt zur Vorhersage, allerdings beschränkt auf die Nachbarschaft $E(Y) = \frac{1}{k} \sum_{\vec{x}_i \in N_k(\vec{x})}^{k} y_i$.
- Für die Vorhersage sind wir an bedingten Wahrscheinlichkeiten interessiert $P(Y|X = \vec{x})$.
- Bei kNN wird die bedingte Wahrscheinlichkeit auf die Nachbarschaft begrenzt $E(Y|\vec{x}_i \in N_k(\vec{x}))$.
- Gerechnet wird dies mit Hilfe Gleichung (13).

Asymptotisches Ergebnis zu kNN

- Wenn k/N gegen 0 und N, k gegen ∞ konvergieren, konvergiert auch $\hat{f}(x)$ gegen $E(Y|X = x)$. (Hastie/etal/2001, S. 19)
- Haben wir also schon (wieder) den perfekten Lernalgorithmus gefunden?

Fluch der hohen Dimension bei kNN

- Die Dichte der Beispiele ist proportional zu $N^{\frac{1}{p}}$.
- Schon bei $p = 10$ brauchen wir 80% der möglichen Werte jedes Attributs X_i, um wenigstens 10% der Daten in einer Nachbarschaft gesehen zu haben!
- Die Dichte der Datenpunkte in der Nachbarschaft ist bei hoher Dimension furchtbar spärlich.
  - $N^{\frac{1}{p}}$ ist bei 100 Beispielen und $p = 10$ nur $100^{1/10} = \sqrt[5]{10}$.
  - Wenn 100 Beispiele bei $p = 1$ einen dichten Raum ergeben, muss man für die selbe Dichte bei $p = 10$ schon 100^{10} Beispiele sammeln: $100^{1/1} = 100, 100^{10 \cdot \frac{1}{10}} = 100$

Bias und Varianz bei kNN

- Wenn man die richtige, dicht besetzte Nachbarschaft hat, verzerrt kNN die Vorhersage nicht (*kleiner Bias*).
- Wenn - wie bei hohen Dimensionen - die Nachbarschaft wild variiert, schwankt auch die Güte der Vorhersage (*große Varianz*).

Bias und Varianz – bildlich

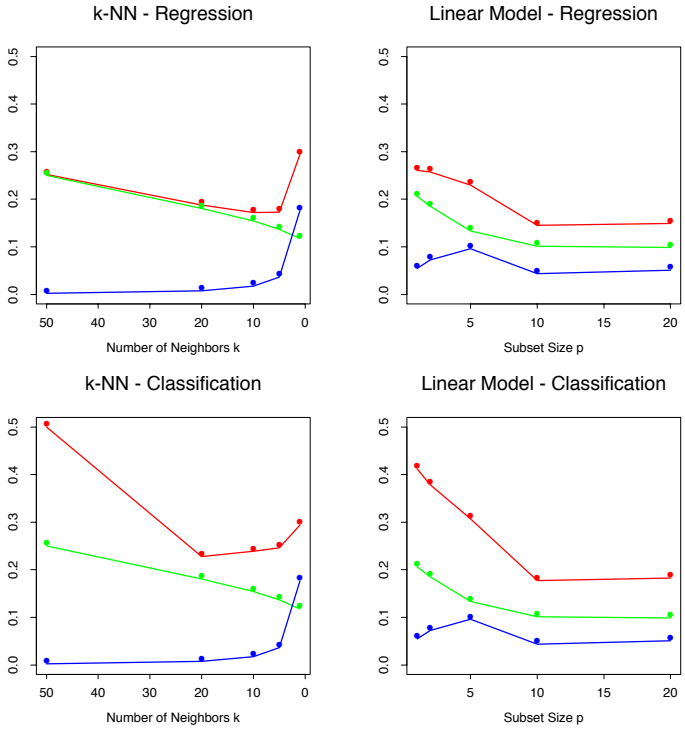


Figure 7.3: *Prediction error (red), squared bias (green) and variance (blue) for a simulated example. The top row is regression with squared error loss; the bottom row is classification with* 0–1 *loss. The models are k-nearest neighbors (left) and best subset regression of size p (right). The variance and bias curves are the same in regression and classification, but the prediction error curve is different.*

Bias, Varianz und Modellkomplexität – bildlich

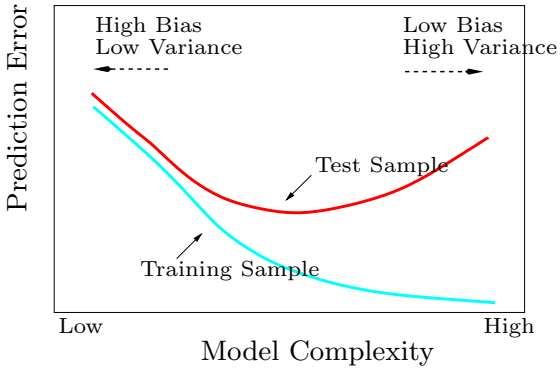


Figure 7.1: *Behavior of test sample and training sample error as the model complexity is varied.*

- Dieser Zusammenhang von Training-/Testfehler und Komplexität bestimmt alle Lernverfahren.

- Kreuzvalidierung lasst abschätzen, wie gut das Modell zu den Daten passt (Modellselektion).

Was wissen Sie jetzt?

- Sie kennen den Fluch der hohen Dimension bei kNN: kleiner Bias, aber hohe Varianz.
- Bei linearen Modellen war es umgekehrt: kleine Varianz, aber hoher Bias (falls die Annahme des linearen Zusammenhangs von X, Y nicht stimmt).

7.2 kNN implementieren

Rahmen – Instanzbasiertes Lernen

- Alle Beispiele werden gespeichert.
  - Geschickt indexieren?
  - Typische Beispiele auswählen?
- Zu einer neuen Beobachtung $\vec{x}$ werden die k ähnlichsten Beispiele $\vec{x}'$ gefunden: $N_k(\vec{x})$.
  - Ähnlichkeitsmaß $Sim(\vec{x}, \vec{x}')$?
- und daraus gemäß einer Entscheidungsfunktion der y-Wert ermittelt.
  - Maximum, Mehrheit, Mittelwert?

Entscheidungsfunktionen zur Klassifikation

- Mehrheitsentscheidung (Gleichung (13)):

$$f(\vec{x}) = \frac{1}{k} \sum_{\vec{x_i} \in N_k(\vec{x})} y_i$$

- Gewichtete Entscheidung nach tatsächlicher Ähnlichkeit $w_i = Sim(\vec{x}, \vec{x_i})$:

$$f(\vec{x}) = \frac{1}{k} \sum_{\vec{x_i} \in N_k(\vec{x})} w_i \cdot y_i$$

- Bewahrt ein Schwellwert $Sim(\vec{x}, \vec{x_i}) \leq \theta \rightarrow \vec{x_i} \notin N_k(\vec{x})$ vor großer Varianz?
- Probieren Sie und vergleichen Sie die Ergebnisse der Kreuzvalidierung!

7.3 Ähnlichkeitsmaße

Ähnlichkeit – Maße

- Ähnlichkeit oder Distanz sollte stets Werte in $[0, 1]$ haben.
- $dist(\vec{x}_1, \vec{x}_2) = 1 - sim(\vec{x}_1, \vec{x}_2)$
- Eine Metrik erfüllt die Bedingungen
  1. $Metrik(x, x) = 0$
  2. $Metrik(x_1, x_2) = Metrik(x_2, x_1)$
  3. $Metrik(x_1, x_3) \leq Metrik(x_1, x_2) + Metrik(x_2, x_3)$

sim: **Ähnlichkeit für einzelne Attribute**

Numerische Attribute: Sei max_j der höchste Wert von X_j und min_j der niedrigste, sei $x_{i,j}$ der Wert des j-ten Attributs in der i-ten Beobachtung, dann ist z.B.

$$sim_j(x_{1,j}, x_{2,j}) = 1 - \frac{|x_{1,j} - x_{2,j}|}{max_j - min_j}$$

ein Ähnlichkeitsmaß für X_j.

Nominale Attribute: Ganz einfach:

$$sim_j(x_{1,j}, x_{2,j}) = \begin{cases} 1 & falls \quad x_{1,j} = x_{2,j} \\ 0 & sonst \end{cases}$$

Sim: **Ähnlichkeit der Beispiele als Kombination der Attributähnlichkeiten**
Im einfachsten Fall mitteln wir die Einzelähnlichkeiten:

$$Sim(\vec{x_1}, \vec{x_2}) = \frac{1}{p} \sum_{j=1}^{p} sim(x_{1,j}, x_{2,j})$$

Vielleicht sind einige Attribute wichtiger als andere?

$$Sim(\vec{x_1}, \vec{x_2}) = \frac{\sum_{j=1}^{p} w_j sim(x_{1,j}, x_{2,j})}{\sum_{j=1}^{p} w_j}$$

Vielleicht ist der quadratische Abstand besser?

$$Sim(\vec{x_1}, \vec{x_2}) = 1 - \sum_{j=1}^{p} w_j (x_{1,j} - x_{2,j})^2$$

Wie bestimmt man w_j?

Spielräume
Sie sehen, es gibt viele Varianten in einer *Klasse* von Verfahren, die instanzbasiert oder nächste Nachbarn heißt. Das bedeutet für

die Implementierung: Es sollte eine Rahmenklasse geben, der Ähnlichkeits- und Fehlermaße übergeben werden – nicht für jedes Kriterium ein neues Programm!

die Anwendung: Bei jedem neuen Datensatz müssen die Kriterien wohl überlegt werden – Modellselektion!

die Theorie: Wir wollen Eigenschaften ermitteln, die für alle Verfahren einer Klasse gültig sind, z.B. Bias und Varianz.

8 Funktionsapproximation

Funktionsapproximation

- Die beiden vorgestellten Verfahren zu maschinellem Lernen, lineare Modelle und k-nächste Nachbarn, sind Instanzen der Funktionsapproximation.
- Gegeben sind die Trainingsbeispiele $\mathcal{T}$, gesucht ist eine Funktion

$$f_\theta(\vec{x}) = \sum_{k=1}^{K} h_k(\vec{x}) \theta_k$$

.

- Dabei gibt es Parameter θ, die abzuschätzen sind, bei den linearen Modellen ist dies β.
- Darüber hinaus können die Daten transformiert werden in einen Raum, der für das Lernen besser geeignet ist: $h_k(\vec{x})$.
- Optimiert wird ein Qualitätskriterium, z.B. wird eine Verlustfunktion minimiert oder die Wahrscheinlichkeit maximiert.

Wege der Funktionsapproximation

Verlustfunktion: Fehler minimieren als Abstand zwischen wahrem Wert und Ergebnis der gelernten Funktion, z.B. $RSS(\theta)$ minimieren. Das haben wir bisher gesehen.

Likelihood: Wahrscheinlichkeit der wahren Werte maximieren! Das schauen wir uns jetzt an.

8.1 Likelihood

Maximum Likelihood

Gegeben eine Verteilung $Pr_\theta(y)$ und eine Stichprobe dieser Verteilung $y_1, ..., y_N$, ist die logarithmierte Wahrscheinlichkeit:

$$L(\theta) = \sum_{i=1}^{N} \log Pr_\theta(y_i) \tag{14}$$

Genau das θ, das y_i am wahrscheinlichsten macht, ist gut – $L(\theta)$ maximieren!

- Wir können dafür eine Verteilung annehmen, da wir die wahre Verteilung nicht kennen.
- Meist ist die Normalverteilung eine gute Annahme.

Normalverteilung $\mathcal{N}(\mu, \sigma)$

Definition 8 (Normalverteilt)**.** Eine Zufallsvariable X heißt **normalverteilt** mit den Parametern μ, σ, wenn sie die Dichtefunktion

$$g(x) = \frac{1}{\sigma\sqrt{2\pi}} \cdot e^{-\frac{1}{2}((x-\mu)/\sigma)^2} \tag{15}$$

besitzt.

Definition 9 (Normalverteilung)**.** Die zugehörige Wahrscheinlichkeitsverteilung $X \sim \mathcal{N}(\mu, \sigma^2)$ heißt **Normalverteilung**, der Graph ihrer Dichtefunktion wird Gaußsche Glockenkurve genannt.

Bei linearen Modellen ist die Maximum Likelihood gleich der Minimierung von RSS

Wir wollen θ schätzen, so dass die richtige Ausprägung von Y auch die wahrscheinlichste ist, gegeben X, θ. Unter der *Annahme der Normalverteilung*:

$$Pr(Y|X, \theta) = \mathcal{N}(f_\theta(X), \sigma^2)$$

Nun entspricht die log-likelihood der Daten gerade $RSS(\theta)$:

$$\begin{aligned} L(\theta) &= \sum_{i=1}^{N} log(\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{y_i - f_\theta(\vec{x}_i)}{\sigma})^2}) \\ &= C_2 + C_1 \cdot \sum_{i=1}^{N} (y_i - f_\theta(\vec{x}_i))^2 \end{aligned}$$

Wie das?

Herleitung von $L(\theta) = RSS(\theta) \cdot C_1 + C_2$ bei Normalverteilung

$$\begin{aligned}
L(\theta) &= \sum_{i=1}^{N} log(\frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{y_i - f_\theta(\vec{x}_i)}{\sigma})^2}) \\
&= \sum_{i=1}^{N} \Big(log(1) - log(\sigma\sqrt{2\pi}) + log(e^{-\frac{1}{2}(\frac{y_i - f_\theta(\vec{x}_i)}{\sigma})^2}) \Big) \\
&= \sum_{i=1}^{N} \left(0 - log(\sigma) - log(\sqrt{2\pi}) - \frac{1}{2}(\frac{y_i - f_\theta(\vec{x}_i)}{\sigma})^2 \right) \\
&= \underbrace{-N \cdot log(\sigma) - \frac{N}{2} log(2\pi)}_{=:C_2} \underbrace{- \frac{1}{2\sigma^2}}_{=:C_1} \sum_{i=1}^{N} (y_i - f_\theta(\vec{x}_i))^2 \\
&= RSS(\theta) \cdot C_1 + C_2
\end{aligned}$$

N, σ sind konstant für einen Datensatz.

Zur Erinnerung:

$$log(a \cdot b) = log(a) + log(b)$$
$$log(\frac{a}{b}) = log(a) - log(b)$$
$$log(\sqrt{x}) = \frac{1}{2} x$$
$$log(e^x) = x$$

Log-likelihood bei nominalem Y ist Entropie

Definition 10 (Cross-Entropie)**.** Sei Y eine Zufallsvariable, die als Werte die Namen von K verschiedenen Klassen annimmt. $Pr(Y = y_k | X = \vec{x}) = p_{k,\theta}(\vec{x}), k = 1, ..., K$

$$L(\theta) = \sum_{i=1}^{N} log(p_{y_i,\theta}(\vec{x}_i)) \tag{16}$$

Wenn man $L(\theta)$ maximiert, passt θ gut zu den Daten im Sinne der Likelihood.

9 Modellselektion

Modellselektion

- Wir haben zwei Modellklassen gesehen: lineare Modelle und Nächste Nachbarn.
- Bei der Verallgemeinerung zur Funktionsapproximation haben wir außerdem Basisfunktionen zur Vorverarbeitung gesehen, die ebenfalls Modellklassen induzieren.
- Wie wählen wir nun Modelle aus?

Verfahren zur Modellselektion

- Kreuzvalidierung für verschiedene Modelle – das mit dem geringsten durchschnittlichen Fehler nehmen! (Minimierung der Verlustfunktion jetzt auf der Ebene der Modelle)
- Direkt anhand der a posteriori Wahrscheinlichkeit Modelle vergleichen. (Maximierung der Wahrscheinlichkeit jetzt auf der Ebene der Modelle)
  - Bayes Information Criterion
  - Minimum Description Length

9.0.1 Kreuzvalidierung zur Modellselektion

Kreuzvalidierung zur Modellselektion

Gegeben eine Klasse von Modellen $f(\vec{x}, \alpha)$, wobei α ein Modell der Klasse indiziert, eine Verlustfunktion $L(y, f(\vec{x}, \alpha))$, N Beispiele und eine Aufteilung der Beispiele in K Partitionen mit der Indexfunktion $\kappa : \{1, ..., N\} \rightarrow \{1, ..., K\}$, die für jede Beobachtung die zugehörige Partition angibt.

Kreuzvalidierung für alle Modelle:

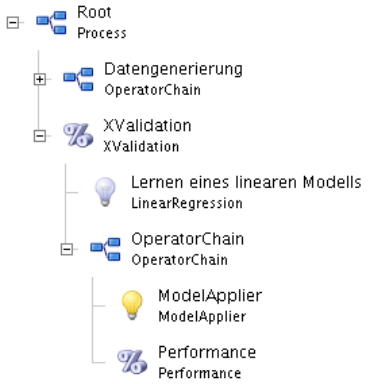
- Lasse die $\kappa(i)$-te Partition aus,
- lerne das α-te Modell: $\hat{f}^{-\kappa(i)}(\vec{x}, \alpha)$.
- rechne den Fehler aus:

$$CV(\alpha) = \frac{1}{N} \sum_{i=1}^{N} L(y_i, \hat{f}^{-\kappa(i)}(\vec{x}_i, \alpha))$$

- Minimiere $CV(\alpha)$, wähle also das Modell mit dem geringsten Verlust.

Modellselektion über Kreuzvalidierung praktisch

In RapidMiner wird die Kreuzvalidierungsschleife schon angeboten.



Es geht aber auch anders...

9.0.2 Bayes Kriterien zur Modellselektion

Bayes Statistik

Definition 11 (A posteriori Wahrscheinlichkeit)**.** Gegeben eine beliebige Einteilung von X in Klassen $y_1, y_2, ..., y_K$ und eine Beobachtung $\vec{x} \in X$. Die Wahrscheinlichkeit von y_j unter der Bedingung, dass $\vec{x}$ beobachtet wird, ist

$$Pr(y_j|\vec{x}) = \frac{Pr(y_j)Pr(\vec{x}|y_j)}{Pr(\vec{x})} \tag{17}$$

$Pr(y_j)$ ist die **a priori** Wahrscheinlichkeit der Klasse. $Pr(y_j|\vec{x})$ ist die **a posteriori** Wahrscheinlichkeit der Klasse.

Bayes Modellselektion

Gegeben eine Menge von Modellen $\mathcal{M}_m, m = 1, ..., M$ mit entsprechenden Parametern θ_m, Trainingsdaten $\mathcal{T}$ und eine Verteilung $Pr(\theta_m|\mathcal{M}_m)$, dann ist die a posteriori Wahrscheinlichkeit eines Modells

$$Pr(\mathcal{M}_m|\mathcal{T}) \sim Pr(\mathcal{M}_m) \cdot Pr(\mathcal{T}|\mathcal{M}_m)$$

Gegeben dass $Pr(\mathcal{M}_l|\mathcal{T}) \neq 0, Pr(\mathcal{T}|\mathcal{M}_l) \neq 0, Pr(\mathcal{M}_l) \neq 0$:

Zum Vergleich zweier Modelle $\mathcal{M}_j, \mathcal{M}_l$ berechnen wir den Quotienten:

$$\frac{Pr(\mathcal{M}_m|\mathcal{T})}{Pr(\mathcal{M}_l|\mathcal{T})} = \frac{Pr(\mathcal{M}_m)}{Pr(\mathcal{M}_l)} \cdot \frac{Pr(\mathcal{T}|\mathcal{M}_m)}{Pr(\mathcal{T}|\mathcal{M}_l)}$$

Ist das Ergebnis > 1, nehmen wir $\mathcal{M}_m$, sonst $\mathcal{M}_l$.

Approximieren der a posteriori Wahrscheinlichkeit

Wenn alle Modelle a priori gleich wahrscheinlich sind, müssen wir nur $Pr(\mathcal{T}|\mathcal{M}_i)$ approximieren.

- Mit Maximum Likelihood schätzen wir $\hat{\theta}_i$.
- Die Anzahl freier Parameter in $\mathcal{M}_i$ nennen wir d_i. Das ist z.B. die Dimension der Beispiele, kann aber wegen $h_k(\vec{x})$ oder einiger Eigenschaften des Lernverfahrens auch etwas anderes sein.
- Als Wahrscheinlichkeit nähern wir an:

$$log\ Pr(\mathcal{T}|\mathcal{M}_i) = log\ Pr(\mathcal{T}|\hat{\theta}_i, \mathcal{M}_i) - \frac{d_i}{2} \cdot log\ N + O(1) \tag{18}$$

Maximale a posteriori Wahrscheinlichkeit und BIC

Definition 12 (Bayes Informationskriterium)**.** Sei d die Anzahl der Parameter eines Modells und N die Anzahl der Beispiele, dann ist das Bayes Informationskriterium BIC

$$BIC = -2\ loglik + (log\ N) \cdot d \tag{19}$$

Dabei ist $loglik = \sum_{i=1}^{N} \log Pr_{\hat{\theta}}(y_i)$.

BIC als Qualitätskriterium bei Likelihood Maximierung wählt eher einfache Modelle. Unter einer Gaußschen Verteilung und bei bekannter Varianz σ^2 rechnen wir

$$-2\ loglik \sim \sum_i \frac{(y_i - \hat{y_i})^2}{\sigma^2}$$

Die Wahl des Modells mit kleinstem BIC entspricht der Wahl des Modells mit größter a posteriori Wahrscheinlichkeit.

Relative Qualität der Modelle per BIC

- Die Wahl des Modells mit kleinstem BIC ist zuverlässig. Gegeben eine Familie von Modellen, darunter das richtige, konvergiert die Wahrscheinlichkeit, dass BIC das richtige wählt, gegen 1, wenn die Anzahl der Beispiele gegen ∞ konvergiert.
- Wenn wir für jedes Modell $\mathcal{M}_m, m = 1, ..., M$ den BIC ausrechnen, können wir (wie bei Kreuzvalidierung auch) die Modelle relativ zueinander bewerten, hier:

$$\frac{e^{-\frac{1}{2} \cdot BIC_m}}{\sum_{l=1}^{M} e^{-\frac{1}{2} \cdot BIC_l}} \tag{20}$$

Minimum Description Length

Ein Modell *kodiert* eine Menge von Beispielen. Wir können Nachrichten so kodieren, dass keine Nachricht Präfix einer anderen ist, z.B.

Nachricht	z1	z2	z3	z4
Code	0	10	110	111

Wir wollen den kürzesten Code für die häufigste Nachricht. Der Code des Beispiels ist optimal, wenn $Pr(z1) = 1/2$, $Pr(z2) = 1/4$, $Pr(z3) = 1/8$, $Pr(z4) = 1/8$.

Wieso das?

Shannon/Weaver Theorem

Code-Länge als Entropie
Wählen wir die Code-Länge l_i einer Nachricht z_i als

$$l_i = -log_2 Pr(z_i)$$

so ist die durchschnittliche Nachrichtenlänge

$$length \geq -\sum Pr(z_i) log_2(Pr(z_i)) \tag{21}$$

Wenn $p_i = A^{-l_i}$, wobei A die Anzahl der verwendeten Zeichen ist, gilt sogar die Gleichheit (s. Beispiel): $Pr(z_1) = 1/2 = 2^{-1} = A^{-l_1}, A = 2, l_1 = 1$

Minimum Description Length zur Modellselektion

Gegeben ein Modell $\mathcal{M}$ mit Parametern θ und Beispiele $\mathcal{T} = (\mathbf{X}, \mathbf{y})$, der Empfänger kennt alle $\mathbf{X}$ und soll die $\mathbf{y}$ empfangen. Dazu müssen wir den Unterschied zwischen Modell und wahren Werten sowie die Modellparameter übermitteln.

Prinzip der Minimum Description Length MDL
Wähle immer das Modell mit der kürzesten Nachrichtenlänge!

$$length = -log\ Pr(\mathbf{y}|\theta, \mathcal{M}, \mathbf{X}) - log\ Pr(\theta|\mathcal{M}) \tag{22}$$

Eigenschaften von MDL

- Bei normalverteilten y, θ, wenn wir $\mathbf{X}$ zur Einfachheit weglassen, sehen wir den Einfluss von σ:

$$length = log\ \sigma + \frac{(y-\theta)^2}{\sigma^2} + \frac{\theta^2}{2}$$

- Je kleiner σ desto kürzer die Nachricht und einfacher das Modell!

Bezug zwischen MDL und BIC

- Wenn wir die Länge (Gleichung 22) minimieren

$$length = -log\ Pr(\mathbf{y}|\theta, \mathcal{M}, \mathbf{X}) - log\ Pr(\theta|\mathcal{M})$$

  maximieren wir auch die a posteriori Wahrscheinlichkeit (vgl. Gleichung 17) $Pr(\mathbf{y}|\mathbf{X})$.

- Mit Hilfe des BIC haben wir Modelle für die Funktionsapproximation durch Maximum Likelihood ausgewählt: das Modell mit dem kleinsten BIC entspricht dem Modell mit größter a posteriori Wahrscheinlichkeit.

- Also kann man das Modell mit der kleinsten Code-Länge (MDL-Prinzip) auch durch die Minimierung des BIC finden.

Was wissen Sie jetzt?

- Funktionsapproximation optimiert eine *Qualitätsfunktion*.
  - *Fehlerminimierung*, z.B. RSS, MSE
  - *Maximierung der Likelihood*, z.B. durch Approximation der a posteriori Wahrscheinlichkeit
    - Fehlerminimierung RSS entspricht Maximum Likelihood, falls Normalverteilung gegeben (Regression).
- Für die *Modellselektion* kann man
  - die Kreuzvalidierung mit Fehlerminimierung und
  - die Kriterien nach Bayes (BIC, MDL) nutzen.

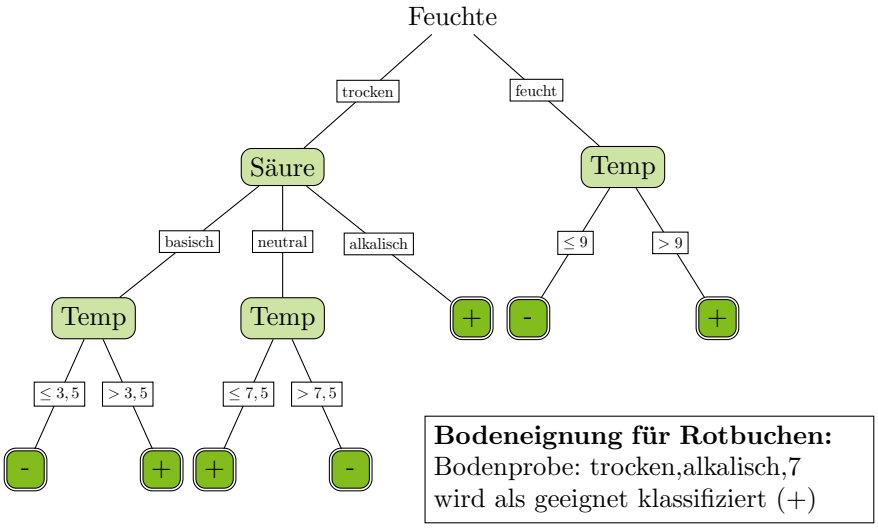
10 Baumlerner

Aufteilen der Beispiele und Modellierung jeder Region

Von globalen zu lokalen Modellen:

- Lineare Modelle können als Vorverarbeitung Basisfunktionen für einzelne Merkmale verwenden.
- Generelle additive Modelle passen die Merkmale einzeln an die Daten an.
- *Baumlerner* teilen den Merkmalsraum in Rechtecke auf und passen in jedem ein Modell an. Dabei wird die Wahl des Merkmals in der rekursiven Aufteilung automatisch bestimmt.
- kNN teilt den Raum der Beispiele bei einer Anfrage x in die Nachbarschaft von x und den Rest auf.

Klassifizieren mit Entscheidungsbäumen



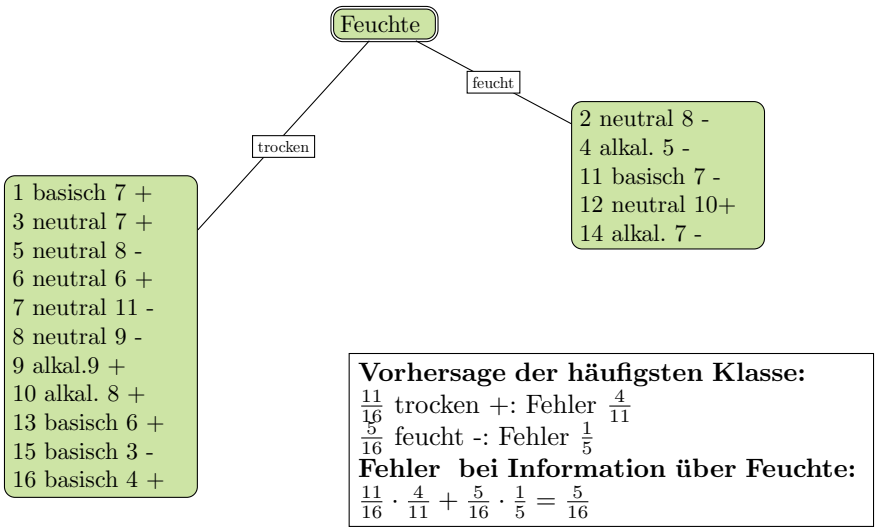
Bodeneignung für Rotbuchen:
Bodenprobe: trocken,alkalisch,7
wird als geeignet klassifiziert (+)

Lernen aus Beispielen

+				-			
ID	Feuchte	Säure	Temp	ID	Feuchte	Säure	Temp
1	trocken	basisch	7	2	feucht	neutral	8
3	trocken	neutral	7	4	feucht	alkal.	5
6	trocken	neutral	6	5	trocken	neutral	8
9	trocken	alkal.	9	7	trocken	neutral	11
10	trocken	alkal.	8	8	trocken	neutral	9
12	feucht	neutral	10	11	feucht	basisch	7
13	trocken	basisch	6	14	feucht	alkal.	7
16	trocken	basisch	4	15	trocken	basisch	3

Ohne weiteres Wissen können wir als Vorhersage immer - sagen. Der Fehler ist dann 8/16.

Aufteilen nach Bodenfeuchte



10.1 Merkmalsauswahl

Bedingte Wahrscheinlichkeit

- Wahrscheinlichkeit, dass ein Beispiel zu einer Klasse gehört, gegeben der Merkmalswert

$$P(Y|X_j) = P(Y \cap X_j)/P(X_j)$$

- Annäherung der Wahrscheinlichkeit über die Häufigkeit
- Gewichtung bezüglich der Oberklasse
- Beispiel: $Y = \{+, -\}, X_j = \{feucht, trocken\}$

$$P(+|feucht) = 1/5,\ P(-|feucht) = 4/5 \text{ gewichtet mit } 5/16$$

$$P(+|trocken) = 7/11,\ P(-|trocken) = 4/11 \text{ gewichtet mit } 11/16$$

Wahl des Merkmals mit dem höchsten Wert (kleinsten Fehler)

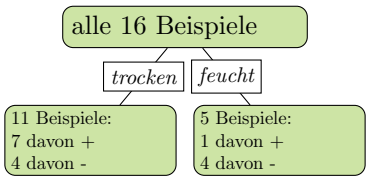
Information eines Merkmals

- Wir betrachten ein Merkmal als Information.
- Wahrscheinlichkeit p_+, dass das Beispiel der Klasse + entstammt. $I(p_+, p_-) = (-p_+ \log p_+) + (-p_- \log p_-)$ Entropie
- Ein Merkmal X_j mit k Werten teilt eine Menge von Beispielen $\mathbf{X}$ in k Untermengen $\mathbf{X}_1, ..., \mathbf{X}_k$ auf. Für jede dieser Mengen berechnen wir die Entropie.

$$Information(X_j, \mathbf{X}) := -\sum_{i=1}^{k} \frac{|\mathbf{X}_i|}{|\mathbf{X}|} I(p_+, p_-)$$

- Der *Informationsgewinn* ist die Differenz zwischen der Entropie der Beispiele mit und ohne die Aufteilung durch X_j.

Feuchte

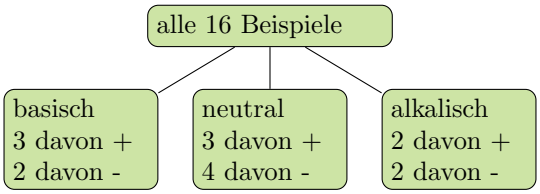


$$-\left[\underbrace{\frac{11}{16}\cdot I(+,-)}_{trocken}+\underbrace{\frac{5}{16}\cdot I(+,-)}_{feucht}\right]$$

$$= -\left[\underbrace{\frac{11}{16}\cdot\left(-\frac{7}{11}\cdot\log\left(\frac{7}{11}\right)-\frac{4}{11}\cdot\log\left(\frac{4}{11}\right)\right)}_{trocken}\right.$$
$$\left.+\underbrace{\frac{5}{16}\left(-\frac{1}{5}\cdot\log\left(\frac{1}{5}\right)-\frac{4}{5}\cdot\log\left(\frac{4}{5}\right)\right)}_{feucht}\right]=-0,27$$

Säure

Güte des Attributs Säure mit den 3 Werten basisch, neutral und alkalisch:



$$-\left(\underbrace{\frac{5}{16}\cdot I(+,-)}_{\text{basisch}}+\underbrace{\frac{7}{16}\cdot I(+,-)}_{\text{neutral}}+\underbrace{\frac{4}{16}\cdot I(+,-)}_{\text{alkalisch}}\right)$$

$$=-0,3$$

basisch $-\frac{3}{5}\cdot\log\left(\frac{3}{5}\right)+-\frac{2}{5}\cdot\log\left(\frac{2}{5}\right)$

neutral $-\frac{3}{7}\cdot\log\left(\frac{3}{7}\right)+-\frac{4}{7}\cdot\log\left(\frac{4}{7}\right)$

alkalisch $-\frac{2}{4}\cdot\log\left(\frac{2}{4}\right)+-\frac{2}{4}\cdot\log\left(\frac{2}{4}\right)$

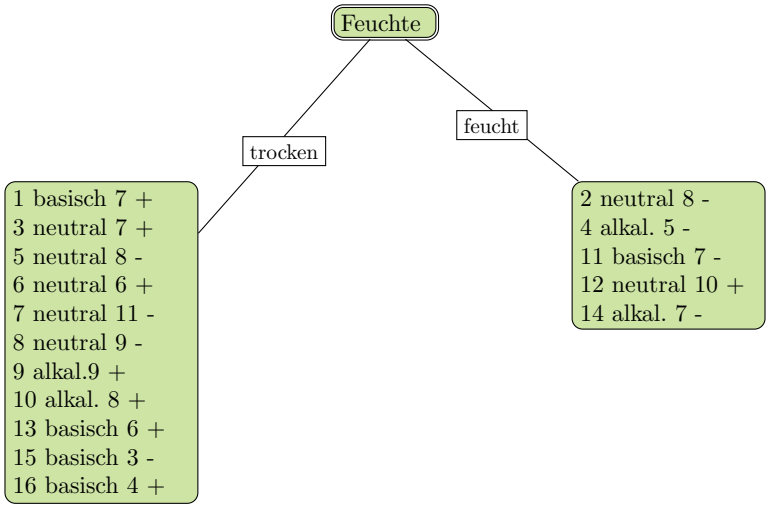
Temperatur

- Numerische Merkmalswerte werden nach Schwellwerten eingeteilt.
  - 9 verschiedene Werte in der Beispielmenge, also 8 Möglichkeiten zu trennen.
  - Wert mit der kleinsten Fehlerrate bei Vorhersage der Mehrheitsklasse liegt bei 7.
  - 5 Beispiele mit Temp ¡ 7, davon 3 in +, 11 Beispiele Temp $\gtrsim$ 7, davon 6 in -.
- Die Güte der Temperatur als Merkmal ist $-0,29$.

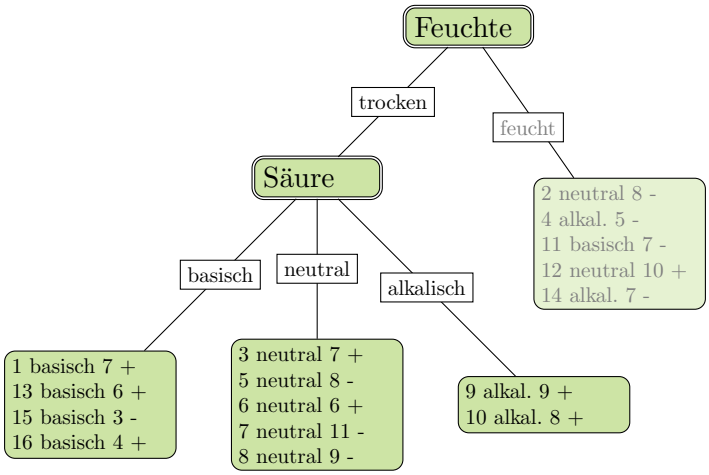
Merkmalsauswahl

- Gewählt wird das Merkmal X_j, dessen Werte am besten in (Unter-)mengen $\mathbf{X}_i$ aufteilen, die geordnet sind.
- Das Gütekriterium *Information* (Entropie) bestimmt die Ordnung der Mengen.
- Im Beispiel hat *Feuchte* den höchsten Gütewert.

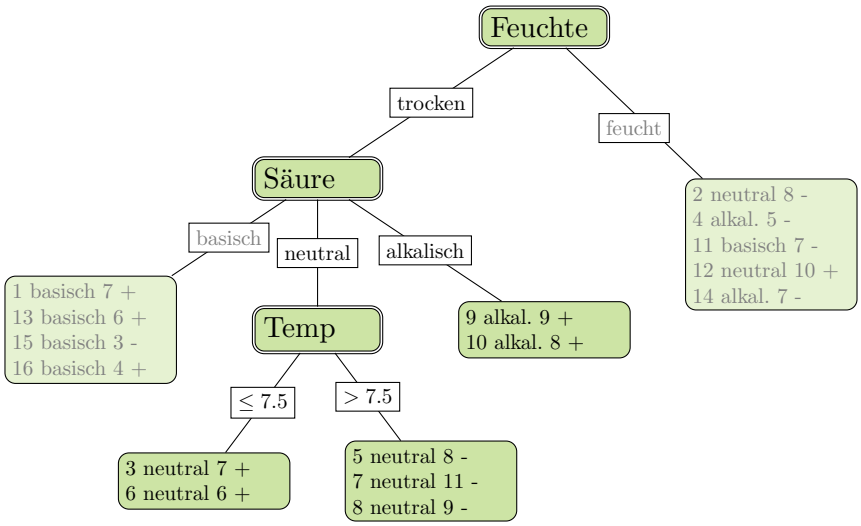
Algorithmus Top Down Induction of Decision Trees (TDIDT, hier: ID3) am Beispiel



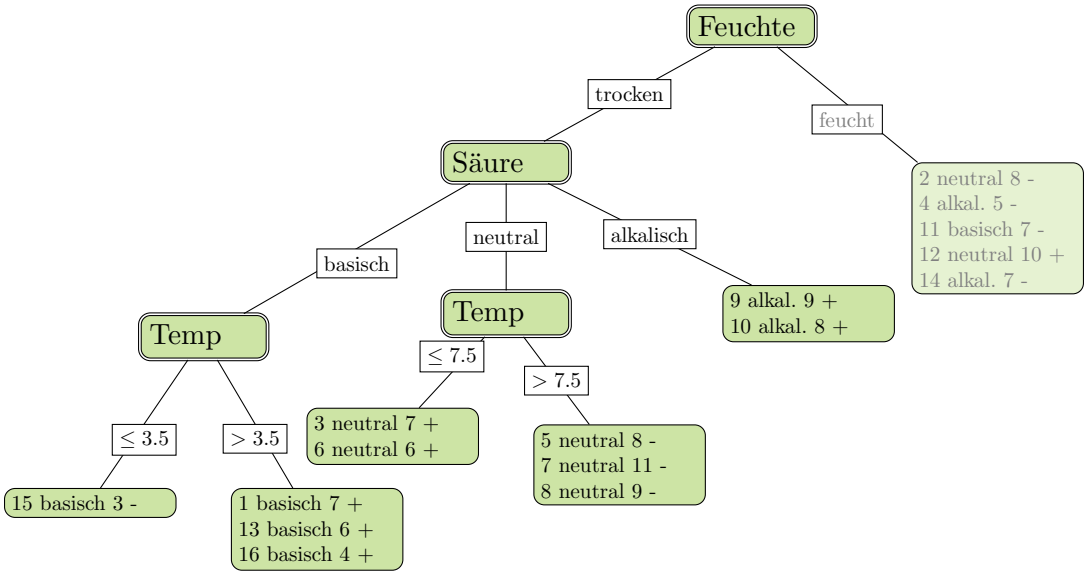
Algorithmus TDIDT (ID3) am Beispiel



Algorithmus TDIDT (ID3) am Beispiel



Algorithmus TDIDT (ID3) am Beispiel



Algorithmus ID3 (TDIDT)

Rekursive Aufteilung der Beispielmenge nach Merkmalsauswahl:

1. $TDIDT(\mathbf{X}, \{X_1, ...X_p\})$
2. $\mathbf{X}$ enthält nur Beispiele einer Klasse → fertig
3. $\mathbf{X}$ enthält Beispiele verschiedener Klassen:
   - $Güte(X_1, .., X_p, \mathbf{X})$
   - Wahl des besten Merkmals X_j mit k Werten
     - Aufteilung von $\mathbf{X}$ in $\mathbf{X}_1, \mathbf{X}_2, ..., \mathbf{X}_k$
     - für $i = 1, ..., k$: $TDIDT(\mathbf{X}_i, \{X_1, ..., X_p\} n X_j)$
   - Resultat ist aktueller Knoten mit den Teilbäumen $T_1, ..., T_k$

Komplexität TDIDT ohne Pruning

Rekursive Aufteilung der Beispielmenge nach Merkmalsauswahl:

- Bei p (nicht-numerischen) Merkmalen und N Beispielen ist die Komplexität $\mathcal{O}(pN \log N)$
  - Die Tiefe des Baums sei in $\mathcal{O}(\log N)$.
  - $\mathcal{O}(N \log N)$ alle Beispiele müssen "in die Tiefe verteilt" werden, also: $\mathcal{O}(N \log N)$ für ein Merkmal.
  - p mal bei p Merkmalen!

10.2 Implementierung

Was muss man implementieren?

import com.rapidminer.example.Attribute;

import com.rapidminer.example.ExampleSet;

split(ExampleSet exampleSet, Attribute attribute);

.

- Die Beispielmenge gemäß der Attributwerte aufteilen.
- Das Attribut auswählen, das zur Partitionierung einer Beispielmenge genutzt wird.
  - Information (Entropie) für alle Attribute berechnen.
- Bei numerischen Attributen den numerischen Wert bestimmen, der die Beispiele am besten aufteilt.

Kleiner Trick

- Wenn es nur nominale Werte gibt, so können diese durchgezählt werden.
  - Wenn der Vergleich beim Aufteilen gemäß eines Merkmalwertes nur nach Gleichheit erfolgt,
  - dann hat das Array für die Nachfolgeknoten gerade den Index der Merkmalswerte.

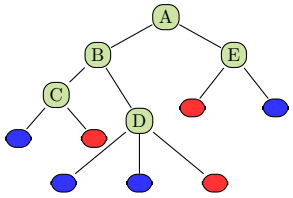
Implementieren in RapidMiner

- **X**: ExampleSet mit den Methoden u.a.
  - `size()` – gibt die Anzahl der Beispiele zurück
  - `getAttributes()` – liefert die Attribute zurück, über `getAttributes().size()` läßt sich die Anzahl ermitteln
  - `iterator()` – liefert einen Iterator über die Beispiele
- $\vec{x_i}$: ein Beispiel (Example) mit den Methoden u.a.
  - `getValue(a)` – gibt den Wert des Attributs a
  - Mit `getAttributes().iterator()` läßt sich über die Attribute eines Examples iterieren
- X_j: Methoden für Werte nominaler Merkmale :
  - Nominale Merkmale werden durch ein Mapping von double-Werten auf Strings realisiert. Für ein nominales Attribut liefert `getMapping()` das Mapping für dieses Attribut.
  - `getMapping().size()` liefert die Anzahl der unterschiedlichen Werte des Attributs
  - `getLabel()` – liefert den Wert des Zielmerkmals als double

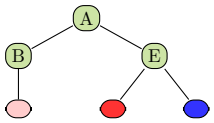
10.3 Gütemaße und Fehlerabschätzung

Stutzen

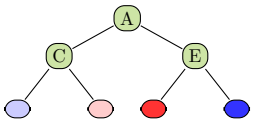
- Überanpassung des Baums an die Trainingsdaten verringern!
- Verständlichkeit erhöhen!
- Stutzen (Pruning):
  1. Knoten an Stelle eines Teilbaums setzen
  2. Einen Teilbaum eine Ebene höher ziehen
- Schätzen, wie sich der wahre Fehler beim Stutzen entwickelt.



1. Knoten an Stelle eines Teilbaums setzen



2. Einen Teilbaum eine Ebene höher ziehen



Stutzen durch Fehlerschätzen

- Wenn der Fehler eines Knotens kleiner ist als die Summe der Fehler seiner Unterknoten, können die Unterknoten weggestutzt werden.
- Dazu müssen wir (bottom-up) die Fehler an allen Knoten schätzen.
- Obendrein sollten wir berücksichtigen, wie genau unsere Schätzung ist. Dazu bestimmen wir ein Konfidenzintervall.
- Wenn die obere Schranke der Konfidenz in den Fehler beim oberen Knoten kleiner ist als bei allen Unterknoten zusammen, werden die Unterknoten gestutzt.

Was ist ein Konfidenzintervall?

Definition 13 (Konfidenzintervall)**.** Vorgegeben eine tolerierte Irrtumswahrscheinlichkeit α, gibt das Konfidenzintervall

$$P(u \leq X \leq o) = 1 - \alpha$$

an, dass X mit der Wahrscheinlichkeit $1 - \alpha$ im Intervall [u, o] liegt und mit der Wahrscheinlichkeit α nicht in [u, o] liegt.

Meist wird das Konfidenzintervall für den Erwartungswert gebildet. Beispiel $\alpha = 0,1$: Mit 90% iger Wahrscheinlichkeit liegt der Mittelwert $\bar{X}$ im Intervall [u, o], nur 10% der Beobachtungen liefern einen Wert außerhalb des Intervalls.

z-Transformation in eine standard-normalverteilte Zufallsvariable

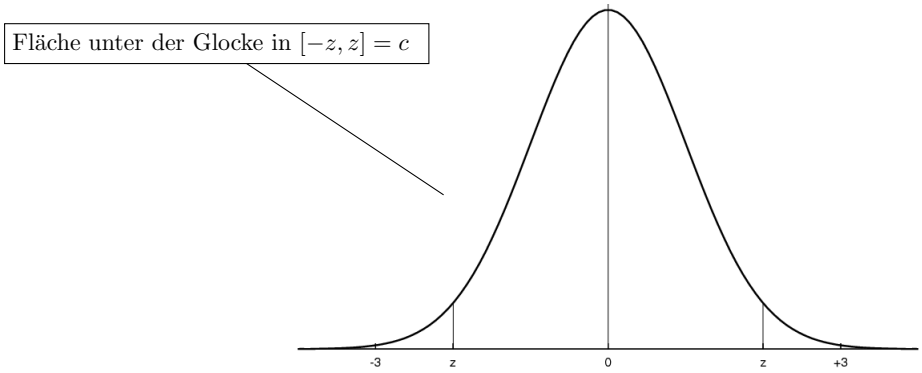
Die Zufallsvariable X wird bezüglich ihres Mittelwerts $\bar{X}$ standardisiert unter der Annahme einer Normalverteilung:

$$Z = \frac{\bar{X} - \mu}{\frac{\sigma}{\sqrt{N}}} \sim \mathcal{N}(0; 1)$$

Die Wahrscheinlichkeit dafür, dass der Mittelwert im Intervall liegt, ist nun:

$$P\left(-z\left(1 - \frac{\alpha}{2}\right) \leq \frac{\bar{X} - \mu}{\frac{\sigma}{\sqrt{N}}} \leq z\left(1 - \frac{\alpha}{2}\right)\right) = 1 - \alpha$$

Verteilung mit z-Werten



- $P(-z \leq X \leq z) = 1-\alpha$ Konfidenzniveau Wahrscheinlichkeit, dass X mit Mittelwert 0 im Intervall der Breite $2z$ liegt ist $1 - \alpha$.
- z kann nachgeschlagen werden (z.B. Bronstein), wobei wegen Symmetrie nur angegeben ist: $P(X \geq z)$

Rechnung für reellwertige Beobachtungen und Mittelwert

Wir wollen ein bestimmtes Konfidenzniveau erreichen, z.B. 0,8.

- $P(X \geq -z)$ $P(X \leq z)$ ist dann $(1 - 0,8)/2 = 0,1$.
- Der z-Wert, für den die Fläche der Glockenkurve zwischen $-z$ und z genau $1 - \alpha = 0,8$ beträgt, ist das $(1 - \frac{\alpha}{2})$-Quantil der Standardnormalverteilung, hier: $1,28$ (nachschlagen).
- Das standardisierte Stichprobenmittel liegt mit der Wahrscheinlichkeit 0,8 zwischen -1,28 und +1,28.

$$\begin{aligned} 0,8 &= P(-1,28 \leq \frac{\bar{X} - \mu}{\frac{\sigma}{\sqrt{N}}} \leq 1,28) \\ &= P(-1,28\frac{\sigma}{\sqrt{N}} \leq \bar{X} - \mu \leq 1,28\frac{\sigma}{\sqrt{N}}) \\ &= P(\bar{X} - 1,28\frac{\sigma}{\sqrt{N}} \leq \mu \leq \bar{X} - 1,28\frac{\sigma}{\sqrt{N}}) \end{aligned}$$

Das Intervall ist $[\bar{X} - 1,28\frac{\sigma}{\sqrt{N}}; \bar{X} + 1,28\frac{\sigma}{\sqrt{N}}]$.

Fehler oder Erfolg schätzen

- Bei den Entscheidungsbäumen beobachten wir nur zwei Werte $Y \in \{+, -\}$.
- Wir haben eine Binomialverteilung mit wahrer Wahrscheinlichkeit p_+ für $y = +$ (Erfolg).
- Beobachtung der Häufigkeit f_+ bei N Versuchen. Varianz:

$$\sigma^2 = \frac{f_+(1 - f_+)}{N}$$

  Erwartungswert:

$$E(p_+) = f_+/N$$

- In das allgemeine Konfidenzintervall $[\bar{X} - z(1 - \alpha/2)\frac{\sigma}{\sqrt{N}}; \bar{X} + 1,28\frac{\sigma}{\sqrt{N}}]$ setzen wir diese Varianz ein und erhalten:

$$\left[f_+ - z(1 - \alpha/2)\frac{\sqrt{f_+(1 - f_+)}}{N}; f_+ z(1 - \alpha/2)\frac{\sqrt{f_+(1 - f_+)}}{N}\right]$$

Konfidenz bei Binomialverteilung

Allgemein berechnet man die obere und untere Schranke der Konfidenz bei einer Binomialverteilung für ein Bernoulli-Experiment:

$$p_+ = \frac{f_+ + \frac{z^2}{2N} \pm z\sqrt{\frac{f_+}{N} - \frac{f_+^2}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}}$$

Hierzu muss lediglich die Häufigkeit f_+ gezählt werden, N, z bekannt sein. Diese Abschätzung für den Erfolg können wir symmetrisch für den Fehler (p_-) durchführen.

Anwendung zum Stutzen

- Für jeden Knoten nehmen wir die obere Schranke (pessimistisch):

$$p_- = \frac{f_- + \frac{z^2}{2N} + z\sqrt{\frac{f_-}{N} - \frac{f_-^2}{N} + \frac{z^2}{4N^2}}}{1 + \frac{z^2}{N}}$$

- Wenn der Schätzfehler eines Knotens kleiner ist als die Kombination der Schätzfehler seiner Unterknoten, werden die Unterknoten weggestutzt. Die Kombination wird gewichtet mit der Anzahl der subsumierten Beispiele.

Gütemaße

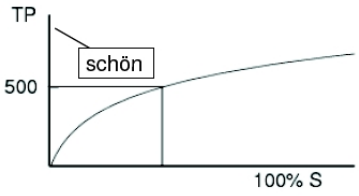
- Konfusionsmatrix:

tatsächlich	Vorhergesagt +	Vorhergesagt −	
+	True positives TP	False negatives FN	Recall: $TP/(TP + FN)$
−	False positives FP	True negatives TN	
	Precision: $TP/(TP + FP)$		

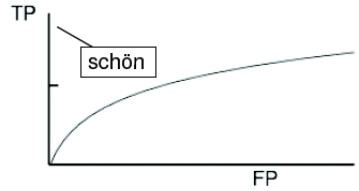
- Accuracy: $P(\hat{f}(x) = y)$ geschätzt als $(TP + TN)/total$

Balance von FP und FN

- F-measure:
$$\frac{\beta \cdot recall \cdot precision}{recall + precision} = \frac{\beta TP}{\beta TP + FP + FN}$$
- Verlaufsformen:
  - Lift: TP für verschiedene Stichprobengrößen S

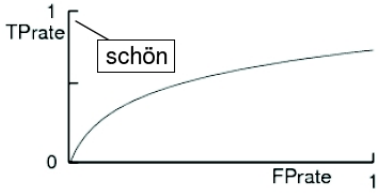


  - Receiver Operating Characteristic (ROC): für verschiedene TP jeweils die FP anzeigen



ROC genauer

- Statt der absoluten Anzahl TP nimm die Raten von true oder false positives – ergibt eine glatte Kurve.
  - Für jeden Prozentsatz von falschen Positiven nimm eine Hypothese h, deren Extension diese Anzahl von FP hat und zähle die TP.
  - $TP_{rate} := TP/P \sim recall$ bezogen auf eine Untermenge
  - $FP_{rate} := FP/N \sim FP/FP + TN$ bezogen auf Untermenge



Kosten von Fehlern

- Nicht immer sind FP so schlimm wie FN
  - medizinische Anwendungen: lieber ein Alarm zu viel als einen zu wenig!
- Gewichtung der Beispiele:
  - Wenn FN 3x so schlimm ist wie FP, dann gewichte negative Beispiele 3x höher als positive.
  - Wenn FP 10x so schlimm ist wie FN, dann gewichte positive Beispiele 10x höher als negative.
- Lerne den Klassifikator mit den gewichteten Beispielen wie üblich. So kann jeder Lerner Kosten berücksichtigen!

Was wissen Sie jetzt?

- Sie kennen den Algorithmus ID3 als Beispiel für TDIDT.
- Für das Lernen verwendet ID3 das Gütemaß des Informationsgewinns auf Basis der Entropie.
- Man kann abschätzen, wie nah das Lernergebnis der unbekannten Wahrheit kommt → Konfidenz
- Man kann abschätzen, wie groß der Fehler sein wird und dies zum Stutzen des gelernten Baums nutzen.
- Lernergebnisse werden evaluiert:
  - Einzelwerte: accuracy, precision, recall, F-measure
  - Verläufe: Lift, ROC

  Diese Evaluationsmethoden gelten nicht nur für Entscheidungsbäume!

11 Basisexpansionen und Strafterm

Ausgangspunkt: Funktionsapproximation

- Die bisher vorgestellten Lernverfahren, sind Instanzen der Funktionsapproximation.
- Gegeben sind die Trainingsbeispiele $\mathcal{T}$, gesucht ist eine Funktion

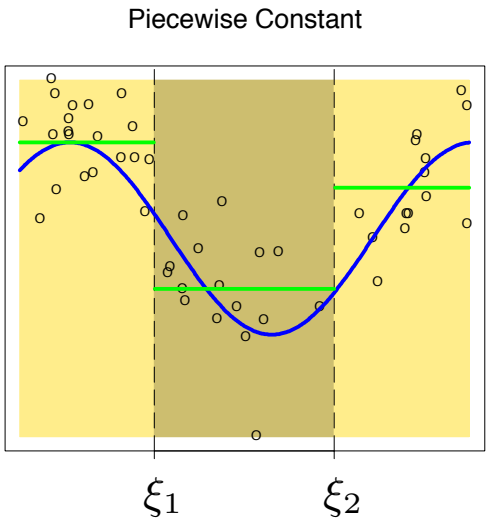
$$f_\theta(x) = \sum_{m=1}^{M} h_m(x)\theta_m$$

  .
- Dabei gibt es Parameter θ, die abzuschätzen sind, bei den linearen Modellen ist dies $\hat{\beta}$.
- Darüber hinaus können die Daten durch Basisfunktionen in einen Raum transformiert werden, der für das Lernen besser geeignet ist: $h_m(x)$.
- Jetzt gehen wir auf $h_m(X) : \mathcal{R}^p \rightarrow \mathcal{R}$ ein.

11.1 Stückweise Funktionen

Einfachste Basisfunktion: Stückweise Konstant

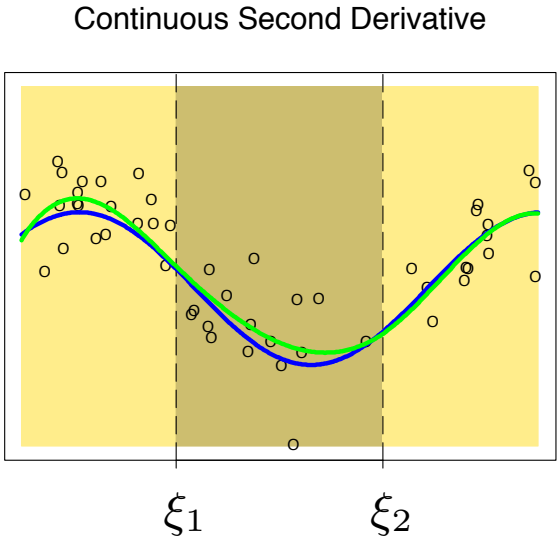
Einteilung von X in Intervalle durch
$h_1(X) = I(X < \xi_1), h_2(X) = I(\xi_1 \leq X < \xi_2)$, $h_3(X) = I(\xi_2 \leq X)$.
Als lineares Modell ergibt sich der Durchschnitt von Y im jeweiligen Intervall: $f(X) = \sum_{m=1}^{3} \hat{\beta}_m h_m(X)$



Stückweise kubisches Polynom

Kontinuierliche, differenzierbare Funktionen (1. und 2. Ableitung) ergeben glattere Annäherung:

$$h_1(X) = X^0, h_3(X) = X^2, h_5(X) = (X - \xi_1)^3_+$$
$$h_2(X) = X^1, h_4(X) = X^3, h_6(X) = (X - \xi_2)^3_+$$



Kubische Splines und Verallgemeinerung

- Für ein Polynom 3. Grades (Ordnung $M = 4$) brauchen wir 4 Basisfunktionen h_i.
- Dazu kommen Basisfunktionen für die Stützstellen. Beim kubischen Polynom hatten wir $K = 2$ Stützstellen ξ mit jeweils einer kubischen Funktion $h_i(X)$.
- Allgemein haben die polynomielle Basisfunktionen die Form

$$\begin{aligned} h_j(X) &= X^{j-1}, j = 1, ..., M \\ h_{M+l}(X) &= (X - \xi_l)^{M-1}_+, l = 1, .., K \end{aligned}$$

- Polynomielle Basisfunktionen heißen *Splines*.

Regression Splines

- Funktionen, die sich an Werte in vorgegebenen Intervallen anpassen, heißen *Regression Splines*.
- Die Anzahl und Lage der Stützstellen ξ_i muss vorgegeben werden.
- Die Funktionen weichen jenseits der Stützstellen sehr vom wahren Wert ab.
- Verbessung: *natürliche Splines*, bei denen jede Funktion jenseits der Intervallgrenzen als linear angenommen wird.

Natürliche kubische Splines

- Das Modell mit kubischem Spline:

$$f(X) = \sum_{j=0}^{3} \beta_j X^j + \sum_{k=1}^{K} \theta_k (X - \xi_k)^3_+$$

- Die Bedingung der Linearität bedeutet: jenseits der Intervallgrenzen darf nur X^1 betrachtet werden. Dies impliziert Beschränkungen (constraints):

$$\begin{aligned} \beta_2 = 0, \qquad & \beta_3 = 0 \\ \sum_{k=1}^{K} \theta_k = 0, \qquad & \sum_{k=1}^{K} \xi_k \theta_k = 0 \end{aligned}$$

- Dadurch reduziert sich die Menge der Basisfunktionen.

Basisfunktionen der natürlichen kubischen Splines

Der natürliche kubische Spline mit K Stützstellen ist durch K Basisfunktionen gegeben.

$$\begin{aligned} N_1(X) &= X^0, \\ N_2(X) &= X^1, \\ N_{k+2}(X) &= d_k(X) - d_{K-1}(X), \ k = 1, ..., K \\ d_k(X) &= \frac{(X - \xi_k)_+^3 - (X - \xi_K)_+^3}{\xi_K - \xi_k} \end{aligned}$$

11.2 Glätten

Glätten erfordert keine Wahl und Platzierung der Trennungen

- Natürliche kubische Splines mit allen Beispielen $x_i, i = 1, ..., N$ als Trennungen hätten zu viele Freiheitsgrade zu bestimmen.
- Mit einem Strafterm für die Krümmung wird aber die Komplexität begrenzt.
- Wir minimieren

$$RSS(f, \lambda) = \sum_{i=1}^{N} (y_i - f(x_i))^2 + \lambda \int (f''(t))^2 \, dt \tag{23}$$

λ gewichtet den Strafterm: $\lambda = 0$ erlaubt alle Funktionen, $\lambda = \infty$ erlaubt nur noch das lineare Modell mit kleinstem RSS – also gar keine Basisfunktionen.

Optimierungsproblem mit Glättung

$$\hat{f}(x) = \sum_{j=1}^{N} N_j(x) \hat{\theta}_j$$

wobei $N_j(x)$ eine Menge von N Basisfunktionen für das Beispiel x ist. Es gibt ein eindeutiges Optimum für natürliche kubische Splines mit allen x_i als Trennstellen. Wir erhalten eine $N \times N$-Matrix: eine Zeile je Beispiel; da jetzt $K = N$ ist, eine Spalte je Basisfunktion.

$$\mathbf{N} = \begin{pmatrix} N_1(x_1) & N_2(x_1) & ... & N_N(x_1) \\ ... & ... & ... & ... \\ N_1(x_i) & ... & ... & N_N(x_i) \\ ... & ... & ... & ... \\ N_1(x_N) & ... & ... & N_N(x_N) \end{pmatrix}$$

$RSS(f, \lambda)$ soll minimiert werden.

Lösung des Optimierungsproblems mit Glättung

Das Qualitätskriterium (Gleichung 23)

$$RSS(f,\lambda) = \sum_{i=1}^{N} (y_i - f(x_i))^2 + \lambda \int (f''(t))^2 \, dt$$

lässt sich vereinfachen zu

$$RSS(\theta,\lambda) = (\mathbf{y} - \mathbf{N}\theta)^T(\mathbf{y} - \mathbf{N}\theta) + \lambda\theta^T\mathbf{\Omega}_N\theta \tag{24}$$

wobei $\{\mathbf{N}\}_{ij} = N_j(x_i)$ und $\{\mathbf{\Omega}_N\}_{jk} = \int N_j''(t)N_k''(t)dt$

Die Lösung ist dann

$$\hat{\theta} = (\mathbf{N}^T\mathbf{N} + \lambda\mathbf{\Omega}_N)^{-1}\mathbf{N}^T\mathbf{y} \tag{25}$$

Beispiel

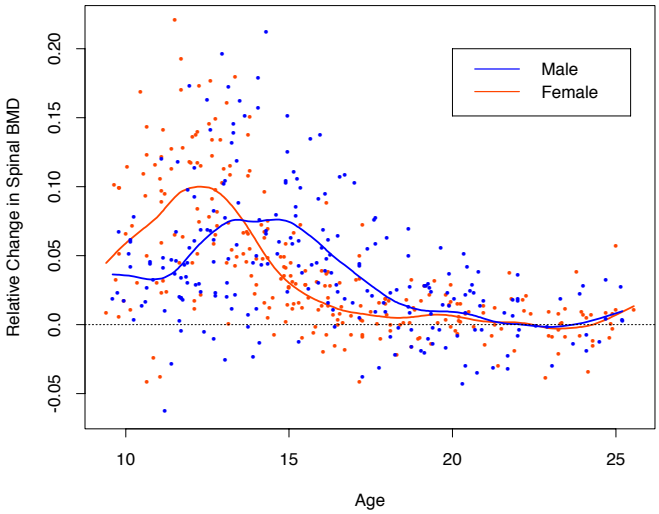


Figure 5.6: *The response is the relative change in bone mineral density measured at the spine in adolescents, as a function of age. A separate smoothing spline was fit to the males and females, with* $\lambda \approx 0.00022$. *This choice corresponds to about* 12 *degrees of freedom.*

Glättungsmatrix $\mathbf{S}_\lambda\mathbf{y}$

Eine Glättung mit vorher bestimmtem λ ist ein linearer Glättungsoperator.

$$\mathbf{S}_\lambda\mathbf{y} = \hat{\mathbf{f}} \quad = \quad \mathbf{N}(\mathbf{N}^T\mathbf{N} + \lambda\mathbf{\Omega}_N)^{-1}\mathbf{N}^T\mathbf{y} \tag{26}$$

$\mathbf{S}_\lambda$ ist die Glättungsmatrix.

- $\mathbf{S}_\lambda$ ist eine symmetrische und semidefinite Matrix.
- $\mathbf{S}_\lambda$ hängt nur von x_i und λ ab.
- $\mathbf{S}_\lambda$ ist linear in $\mathbf{y}$.
- Der Freiheitsgrad ist die Summe der Diagonalelemente von $\mathbf{S}_\lambda$, bezeichnet $df_\lambda = trace(\mathbf{S}_\lambda)$.

Was wissen Sie jetzt?

- Wir haben eine Methode gesehen, Nichtlinearität zu berücksichtigen. Die Daten werden durch Basisexpansionen umgeformt und erst danach linear modelliert.
- Insbesondere haben wir das kubische Polynom gesehen – noch höhere Exponenten ergeben für das menschliche Auge keine Verbesserung der Glättung.
- Die Fehlerminimierung mit Basisexpansion und Strafterm (Gleichungen (23) und (24)) ergibt bei fester Gewichtung λ des Strafterms eine *Glättungsmatrix* $\mathbf{S}_\lambda$.

12 Generelle Additive Modelle

Generelle additive Modelle

- Lineare Modelle passen eine Hyperebene an alle Daten an. Die Hyperebene wird dann auf verschiedene Weisen zur Vorhersage genutzt.
- Basisfunktionen können Nichtlinearität ausdrücken: nach ihrer Anwendung wird dann mit einem linearen Modell vorhergesagt.
- Das Modell selbst kann aber auch nichtlinear sein! Die allgemeine Form genereller additiver Modelle für die Regression:

$$E(Y|X_1, X_2, ..., X_p) = \alpha + f_1(X_1) + f_2(X_2) + ... + f_p(X_p) \tag{27}$$

- Jedes f_i sei hier ein kubischer Spline.

Fehlerminimierung bei generellen additiven Modellen

Eben haben wir das Glätten jeweils für ein Merkmal bei der Funktionsapproximation gesehen mit der Fehlerminimierung beim Glätten einer Funktion (Gleichung 23):

$$RSS(f, \lambda) = \sum_{i=1}^{N} (y_i - f(x_i))^2 + \lambda \int (f''(t))^2 \, dt$$

Bei generellen additiven Modellen müssen wir parallel p Funktionen anpassen:

$$PRSS(\alpha, f_1, ..., f_p) = \sum_{i=1}^{N} \left[y_i - \alpha - \sum_{j=1}^{p} f_j(x_{ij}) \right]^2 + \sum_{j=1}^{p} \lambda_j \int f_j''(t_j)^2 dt_j \tag{28}$$

Jede Funktion f_j ist ein natürlicher kubischer Spline für X_j mit Trennungen an den Werten $x_{ij}, i = 1, ..., N$.

Annahmen für die Optimierung

Um eine eindeutige Lösung der Fehlerminimierung zu finden, nehmen wir an:

$$\forall j : \sum_{i=1}^{N} f_j(x_{ij}) = 0$$

Dann ist $\hat{\alpha} = Mittelwert(y_i)$. Falls die $N \times N$-Matrix der Beispiele nichtsingulär ist (invertierbar, die Determinante der Matrix ist $det(\mathbf{N}) \neq 0$), hat Gleichung (28) eine eindeutige Lösung. Das Optimierungsproblem ist dann konvex.

Backfitting Verfahren ($\mathbf{X}, \mathbf{S}, \tau$,)

1. $\hat{\alpha} := \frac{1}{N} \sum_{i=1}^{N} y_i$; For j=1 until p do $stable_j$:=0;
2. Iterator j über allen Merkmalen $M \backslash Fertig$
   - If $stable_j > \tau$, return $\hat{f}_j$; $Fertig$:=$Fertig \cup \hat{f}_j$; Goto 2;
   - For i=1 until N
   $$\hat{f}_j := S_j \left[y_i - \hat{\alpha} - \sum_{k=1, k \neq j}^{p} \hat{f}_k(x_{ik}) \right]$$
   % Bei Anpassung von $\hat{f}_j$ alle anderen $\hat{f}_k$ verwenden!
   - If $\hat{f}_j$ did not change, $stable_j$++;
3. If $M \neq \{\}$, Goto 2; else stop.

Was wissen Sie jetzt?

- Sie haben gesehen, dass auch das Modell selbst zusammengesetzt sein kann aus an die Beispiele angepassten Glättungsfunktionen.
- Solche Modelle heißen *additive Modelle*.
- Diese Modelle müssen die Glättungsfunktionen für alle Merkmale gleichzeitig anpassen.
- Sie haben den *Backfitting Algorithmus* dafür gesehen.
- Es gibt noch andere additive Modelle und deren Lernverfahren, z.B. additive logistische Regression.

Generalisierte additive logistische Regression
http://de.wikipedia.org/wiki/Newton-Verfahren

13 Support Vector Machine

13.1 Hinführungen zur SVM

Übersicht über die Stützvektormethode (SVM)
Eigenschaften der Stützvektormethode (SVM) (Support Vector Machine)

- Maximieren der Breite einer separierenden Hyperebene – maximum margin method – ergibt eindeutige, optimale trennende Hyperebene.
- Transformation des Datenraums durch Kernfunktion behandelt Nichtlinearität.
- Strukturelle Risikominimierung minimiert nicht nur den Fehler, sondern auch die Komplexität des Modells.

Einführende Literatur

- Vladimir Vapnik "The Nature of Statistical Learning Theory" Springer Vg. 1995
- W.N. Wapnik, A. Tscherwonenkis "Theorie der Zeichenerkennung" Akademie Vg. 1979
- Christopher Burges "A Tutorial on Support Vector Machines for Pattern Recognition" in: Data Mining and Knowledge Discovery 2, 1998, 121-167

Vertiefung: Bernhard Schölkopf, Alexander Smola "Learning with Kernels", MIT Press, 2002

Probleme der Empirischen Risikominimierung

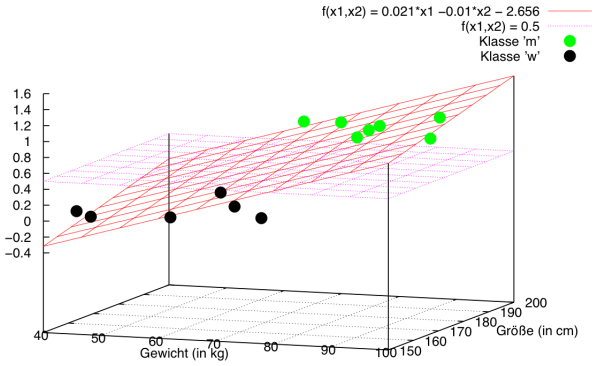
Empirische Risikominimierung: Bisher haben wir lineare Modelle

$$\hat{Y} = \hat{\beta}_0 + \sum_{j=1}^{p} X_j \hat{\beta}_j$$

auf die Fehlerminimierung hin optimiert:

$$RSS(\hat{\vec{\beta}}) = \sum_{i=1}^{N} (y_i - \vec{x}_i^T \hat{\beta})^2$$

Wo trennen wir die Daten?



Problem: Mehrere Funktionen mit minimalem Fehler existieren. Welche wählen?

- *1. Ausweg:* Verbessertes Kriterium: *maximum margin.*
- *2. Ausweg:* Zusätzliches Kriterium: möglichst geringe Komplexität des Modells (*Strukturelle Risikominimierung*)

Klassifikationsproblem

Gegeben sei ein Klassifikationsproblem mit $Y = \{-1; +1\}$ und $\mathbf{X} \subseteq \mathbb{R}^p$. Sei $\mathbf{X} = C_+ \dot{\cup} C_-$ die Menge der Trainingsbeispiele mit

$$C_+ = \{(\vec{x}, y) \mid y = +1\} \quad \text{und} \quad C_- = \{(\vec{x}, y) \mid y = -1\}$$

Zur Klassifikation ist nun eine Hyperebene

$$H = \left\{ \vec{x} \mid \beta_0 + \langle \vec{x}, \vec{\beta} \rangle = 0 \right\}$$

gesucht, die die Mengen C_+ und C_- *bestmöglichst* trennt Für eine gegebene Hyperebene H erfolgt die Klassifikation dann durch

$$\hat{y} = \text{sign}\left(\beta_0 + \langle \vec{x}, \vec{\beta} \rangle\right)$$

Notationen...

Und warum jetzt $\langle \vec{x}, \vec{\beta} \rangle$ statt $\vec{x}^T \vec{\beta}$? *[2ex]

Wir bewegen uns derzeit in einem $\mathbb{R}$-Vektorraum der Beispiele mit dem Standardskalarprodukt

$$\langle \vec{x}, \vec{\beta} \rangle = \underbrace{\vec{x}^T \vec{\beta}}_{Matrixmultiplikation} = \underbrace{\vec{x}\vec{\beta}}_{ImplizitesSkalarprodukt}$$

Die Notation $\langle \vec{x}, \vec{\beta} \rangle$ sollte aus der linearen Algebra (Schule?) bekannt sein.

Klassifikation mit Hyperebenen

Ist eine Ebene $\tilde{H}$ mit

$$\tilde{H} = \left\{ \vec{x} \mid \beta_0 + \langle \vec{x}, \vec{\beta} \rangle = 0 \right\}$$

gegeben, können wir diese in Hesse-Normalenform überführen

$$H = \left\{ \vec{x} \mid \beta_0^* + \langle \vec{x}, \vec{\beta}^* \rangle = 0 \right\} \quad \text{mit } \vec{\beta}^* := \frac{\vec{\beta}}{||\vec{\beta}||}, \beta_0^* := \frac{\beta_0}{||\vec{\beta}||}$$

und erhalten die vorzeichenbehaftete Distanz eines Punktes $\vec{x}$ zu H durch

$$d(\vec{x}, H) \quad = \quad \langle \vec{x} - \vec{x}_0, \vec{\beta}^* \rangle = \frac{1}{||\vec{\beta}||} \left(\langle \vec{x}, \vec{\beta} \rangle + \beta_0 \right)$$

(Übungsaufgabe)

Klassifikation mit Hyperebenen

Die vorzeichenbehaftete Distanz $d(\vec{x}, H)$ drückt aus

1. den Abstand $|d(\vec{x}, H)|$ von $\vec{x}$ zu Ebene H
2. die Lage von $\vec{x}$ relativ zur Orientierung $(\vec{\beta})$ von H, d.h.

$$\text{sign}\left(d(\vec{x}, H)\right) = \begin{cases} +1 & , \text{ falls } \cos \measuredangle(\vec{x}, \vec{\beta}) \geq 0 \\ -1 & , \text{ sonst} \end{cases}$$

Auf diese Weise lassen sich die Punkte klassifizieren mit

$$\hat{y} = \text{sign}\left(\beta_0 + \langle \vec{x}, \vec{\beta} \rangle \right)$$

Einfacher Ansatz nach Schölkopf/Smola

Ein einfacher Ansatz zu einer separierenden Hyperebene zu kommen, geht über die Zentroiden von C_+ und C_-. Seien

$$\vec{c}_+ := \frac{1}{|C_+|} \sum_{(\vec{x},y) \in C_+} \vec{x} \quad \text{und} \quad \vec{c}_- := \frac{1}{|C_-|} \sum_{(\vec{x},y) \in C_-} \vec{x}$$

Wähle nun

$$\vec{x_0} := \frac{\vec{c}_+ + \vec{c}_-}{2} \quad \text{und} \quad \vec{\beta} := \vec{c}_+ - \vec{c}_-$$

als Hyperebene mit Normalenvektor $\vec{\beta}$ durch den Punkt $\vec{x}_0$

Separierende Hyperebene über Zentroiden

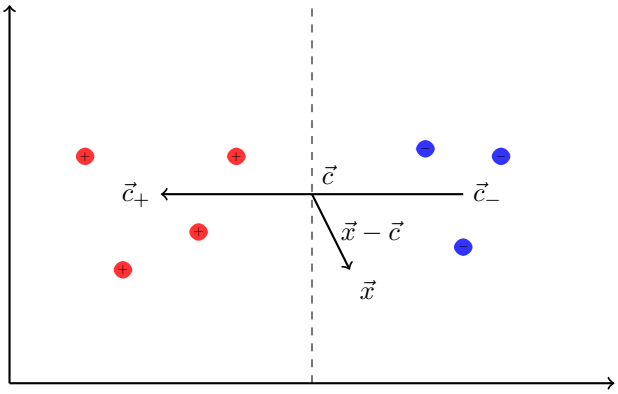
Durch $\vec{\beta}$ und $\vec{x}_0$ ist die Hyperebene gegeben als

$$\tilde{H} = \left\{ \vec{x} \mid \langle \vec{x} - \vec{x}_0, \vec{\beta} \rangle = 0 \right\} = \left\{ \vec{x} \mid \langle \vec{x}, \vec{\beta} \rangle - \underbrace{\langle \vec{x}_0, \vec{\beta} \rangle}_{=: -\beta_0} = 0 \right\}$$

Damit erfolgt die Klassifikation durch

$$\begin{aligned} \hat{y} &= \text{sign}\left(\langle \vec{x} - \vec{c}, \vec{\beta} \rangle \right) \\ &= \text{sign}\left(\langle \vec{x}, \vec{c}_+ \rangle - \langle \vec{x}, \vec{c}_- \rangle + \beta_0 \right) \end{aligned}$$

Lernalgorithmus im Bild



Fast...

... wäre das schon die Stützvektormethode. Aber:

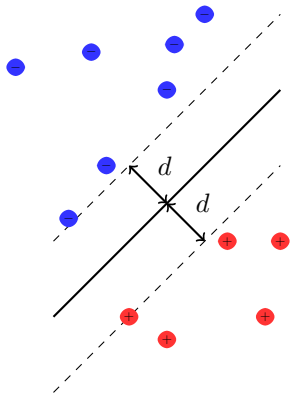
- Einfach den Mittelpunkt der Beispiele einer Klasse zu berechnen ist zu einfach, um ein ordentliches $\vec{\beta}$ zu bekommen.
- Man erhält so nicht die optimale Hyperebene.

Die optimale Hyperebene

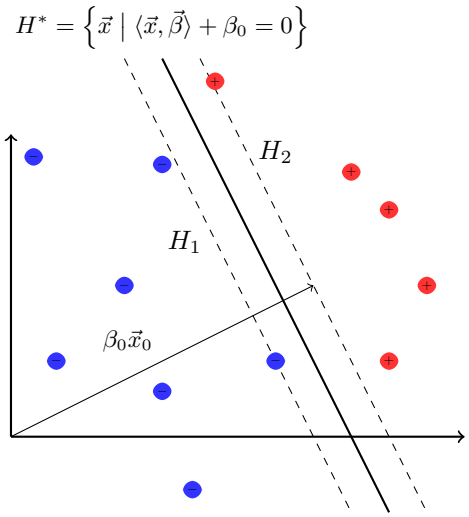
Eine Menge von Beispielen heißt *linear trennbar*, falls es eine Hyperebene H gibt, die die positiven und negativen Beispiele trennt.

Definition 13.1 (Optimale Hyperebene)**.** *Eine separierende Hyperebene H heißt optimal, wenn ihr Abstand d zum nächsten positiven und nächsten negativen Beispiel maximal ist.*

Satz 13.1 (Eindeutigkeit)**.** *Es existiert eine eindeutig bestimmte optimale Hyperebene.*



13.2 Maximum Margin Methode



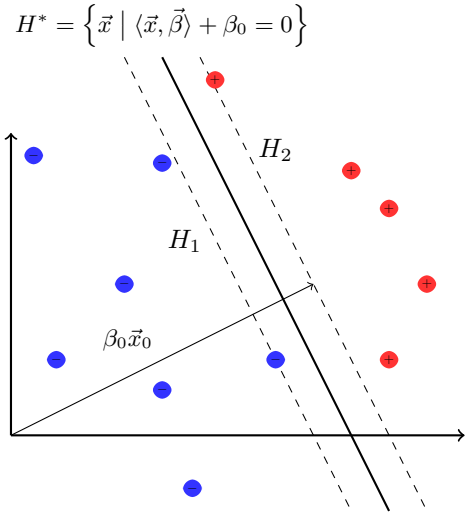
Nach 13.1 wird die optimale Hyperebene durch die nächstliegenende Punkte aus C_+ und C_- bestimmt. Skalierung von $\vec{\beta}$ und β_0, so dass

$$|\langle \vec{\beta}, \vec{x} \rangle + \beta_0| = 1$$

für alle Beispiele am nächsten zur Hyperebene liefert die Hyperebenen H_1 und H_2

$$H_j = \left\{\vec{x} \mid \langle \vec{x}, \vec{\beta} \rangle + \beta_0 = (-1)^j\right\}$$

Abstand der Hyperebenen zum Ursprung



Der Abstand der mittleren Ebene H^* zum Ursprung beträgt

$$d(\vec{0}, H^*) = \frac{\beta_0}{||\vec{\beta}||}$$

Die Abstände der grauen Ebenen H_1 und H_2 sind

$$d(\vec{0}, H_j) = \frac{\beta_0 + (-1)^j}{\|\vec{\beta}\|}$$

$$\overset{H_1 || H_2}{\Rightarrow} d(H_1, H_2) = \frac{2}{||\vec{\beta}||}$$

Margin

Nach Konstruktion liegt kein Beispiel zwischen H_1 und H_2, d.h.

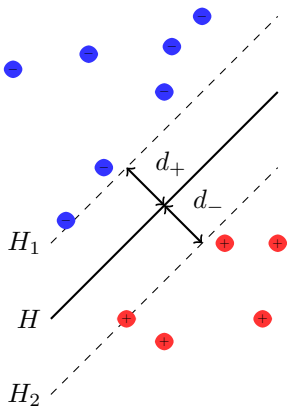
$$\langle \vec{x}, \vec{\beta} \rangle + \beta_0 \geq +1 \forall \vec{x} \in C_+ \tag{29}$$

$$\langle \vec{x}, \vec{\beta} \rangle + \beta_0 \leq -1 \forall \vec{x} \in C_- \tag{30}$$

Der Abstand

$$d(H_1, H_2) = \frac{2}{||\vec{\beta}||}$$

heißt *Margin* und soll maximiert werden!



Maximimum Margin

Mit der Maximierung des Margin finden wir eine *optimale Hyperebene* innerhalb der Menge der möglichen trennenden Hyperebenen. Durch die Minimierung von $\frac{1}{2}||\vec{\beta}||^2$ erhalten wir ein konvexes, quadratisches Optimierungsproblem, d.h.

- Es existiert eine eindeutig bestimmte, optimale Hyperebene

$$H^* = \left\{ \vec{x} \mid \langle \vec{x}, \vec{\beta} \rangle + \beta_0 = 0 \right\}$$

Das quadratische Optimierungsproblem läßt sich in Zeit $O(N^3)$ lösen.

Optimierungsaufgabe

Nach diesen Vorüberlegungen haben wir also (nur noch) die folgende Optimierungsaufgabe zu lösen:

Definition 13.2 (Optimierungsaufgabe)**.** *Minimiere*

$$\frac{1}{2}||\vec{\beta}||^2$$

unter den Nebenbedingungen

$$\langle \vec{x}, \vec{\beta} \rangle + \beta_0 \geq +1 \quad \forall \vec{x} \in C_+$$
$$\langle \vec{x}, \vec{\beta} \rangle + \beta_0 \leq -1 \quad \forall \vec{x} \in C_-$$

Die Nebenbedingungen lassen sich zusammenfassen zu

$$y(\langle \vec{x}, \vec{\beta} \rangle + \beta_0) - 1 \geq 0 \quad \forall (\vec{x}, y) \in \mathbf{X} \tag{31}$$

Optimierung mit Nebenbedingungen

Sei die optimierende Funktion $f : \mathbb{R} \to \mathbb{R}$ gegeben als

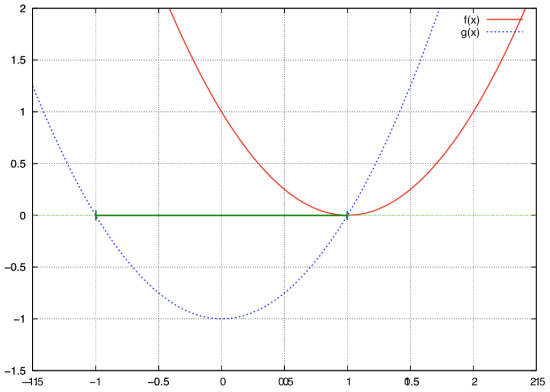
$$f(x) = (x-1)^2$$

unter der einzigen Nebenbedingung

$$g(x) = x^2 - 1,$$

d.h. für die möglichen Lösungen $\tilde{x}$ muss gelten

$$\tilde{x} \in \{x \in \mathbb{R} \mid g(x) \leq 0\}$$



Optimierung mit Lagrange

Die Optimierung nach Lagrange ermöglicht die Optimierung einer Funktion $f(x)$ unter Nebenbedingungen durch *Relaxation*. Mit der Lagrange-Methode lassen sich Nebenbedingungen g_i und h_j der Art

$$g_i(x) \leq 0 \quad \text{und} \quad h_j(x) = 0$$

behandeln, indem diese zur zu optimierenden Funktion f hinzugefügt werden, im Falle eines Minimierungsproblems als

$$\min f(x) + \sum_i \alpha_i g_i(x) + \sum_j \mu_j h_j(x) \quad \text{mit } \alpha_i, \mu_j \geq 0 \, \forall \, i, j$$

Die α_i und μ_j heißen auch *Lagrange-Multiplikatoren*.

13.3 Lagrange-Optimierung

Lagrange-Funktion

Die Umformung der Nebenbedingungen (31) erlaubt nun die Anwendung von Lagrange (nur Ungleichheitsbedingungen):

Definition 13.3 (Lagrange-Funktion)**.** *Sei das Optimierungsproblem gegeben, $f(\vec{\beta})$ zu minimieren unter den Nebenbedingungen $g_i(\vec{\beta}) \geq 0, i = 1, ..., m$ dann ist die Lagrange-Funktion:*

$$L\left(\vec{\beta}, \vec{\alpha}\right) = f(\vec{\beta}) - \sum_{i=1}^{m} \alpha_i g_i(\vec{\beta}) \tag{32}$$

Dabei muss gelten $\alpha_i \geq 0$, Gleichheitsbedingungen sind nicht gegeben.

Optimierungsfunktion als Lagrange

Die Nebenbedingungen g_i sind gegeben durch

$$g_i(\vec{\beta}, \beta_0) = y_i \left(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0\right) - 1 \geq 0 \, \forall \, \vec{x_i} \in \mathbf{X}$$

Die Formulierung des Optimierungsproblems nach Lagrange wird auch als *Primales Problem* bezeichnet:

Definition 13.4 (Primales Problem)**.** *Die Funktion*

$$L_P(\vec{\beta}, \beta_0, \vec{\alpha}) = \frac{1}{2}\|\vec{\beta}\|^2 - \sum_{i=1}^{N} \alpha_i \left(y_i \left(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 \right) - 1 \right) \tag{33}$$

soll L_P bezüglich $\vec{\beta}$ und β_0 minimiert *und bezüglich $\vec{\alpha}$* maximiert *werden!*

Karush-Kuhn-Tucker Bedingungen

Durch die partiellen Ableitung nach $\vec{\beta}$ und β_0 erhalten wir

$$\frac{\partial}{\partial \vec{\beta}} L_P(\vec{\beta}, \beta_0, \vec{\alpha}) = \vec{\beta} - \sum_i \alpha_i y_i \vec{x}_i \quad \text{und} \quad \frac{\partial}{\partial \beta_0} L_P(\vec{\beta}, \beta_0, \vec{\alpha}) = -\sum_i \alpha_i y_i$$

Nullsetzen der Ableitungen und die Berücksichtigung der Nebenbedingungen führt zu den KKT-Bedingungen für eine Lösung für L_P:

$$\vec{\beta} = \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i \quad \text{und} \quad \sum_{i=1}^{N} \alpha_i y_i = 0 \tag{34}$$

$$\alpha_i \geq 0 \; \forall \, i = 1, \ldots, N \tag{35}$$

$$\alpha_i \left(y_i \left(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 \right) - 1 \right) = 0 \; \forall \, i = 1, \ldots, N \tag{36}$$

Duales Problem

Das primale Problem soll bezüglich $\vec{\beta}$ und β_0 minimiert und bezüglich $\vec{\alpha}$ maximiert werden. Mit den Bedingungen aus $\frac{\partial L_P}{\partial \vec{\beta}}$ und $\frac{\partial L_P}{\partial \beta_0}$ erhalten wir den *dualen Lagrange-Ausdruck* $L_D(\vec{\alpha})$

- Der duale Lagrange-Ausdruck $L(\vec{\alpha})$ soll maximiert werden.
- Das Minimum des ursprünglichen Optimierungsproblems tritt genau bei jenen Werten von $\vec{\beta}$,β_0,$\vec{\alpha}$ auf wie das Maximum des dualen Problems.

Umformung des primalen in das duale Problem

$$\begin{aligned}
&\frac{1}{2}||\vec{\beta}||^2 - \sum_{i=1}^{N} \alpha_i && \left[y_i \left(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 \right) - 1 \right] && \\
=&\frac{1}{2}||\vec{\beta}||^2 - \sum_{i=1}^{N} \alpha_i && y_i \left(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 \right) && + \sum_{i=1}^{N} \alpha_i \\
=&\frac{1}{2}||\vec{\beta}||^2 - \sum_{i=1}^{N} \alpha_i y_i \langle \vec{x}_i, \vec{\beta} \rangle && - \sum_{i=1}^{N} \alpha_i y_i \beta_0 && + \sum_{i=1}^{N} \alpha_i \\
\overset{(34)}{=}&\frac{1}{2}||\vec{\beta}||^2 - \sum_{i=1}^{N} \alpha_i y_i \langle \vec{x}_i, \vec{\beta} \rangle && && + \sum_{i=1}^{N} \alpha_i
\end{aligned}$$

Einsetzen von $\vec{\beta} = \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i$ führt zu

$$\begin{aligned}
&\frac{1}{2}||\vec{\beta}||^2 && - \sum_{i=1}^{N} \alpha_i y_i \langle \vec{x}_i, \vec{\beta} \rangle && + \sum_{i=1}^{N} \alpha_i \\
=&\frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \langle \vec{x}_i, \vec{x}_j \rangle && - \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \langle \vec{x}_i, \vec{x}_j \rangle && + \sum_{i=1}^{N} \alpha_i \\
=&+ \sum_{i=1}^{N} \alpha_i && - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j \langle \vec{x}_i, \vec{x}_j \rangle &&
\end{aligned}$$

unter den Nebenbedingungen $0 = \sum_{i=1}^{N} \alpha_i y_i$ und $\alpha_i \geq 0 \; \forall i$

SVM Optimierungsproblem (Duales Problem)

Die Umformungen führen nach Einsetzen der KKT-Bedingungen zum *dualen Problem*:

Definition 13.5 (Duales Problem)**.** *Maximiere*

$$L_D(\vec{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \vec{x}_i, \vec{x}_j \rangle \tag{37}$$

unter den Bedingungen

$$\alpha_i \geq 0 \; \forall \, i = 1, \ldots, N \quad \textit{und} \quad \sum_{i=1}^{N} \alpha_i y_i = 0$$

Stützvektoren

Die Lösung $\vec{\alpha}^*$ des dualen Problems

$$L_D(\vec{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \vec{x}_i, \vec{x}_j \rangle$$

muss die KKT-Bedingungen erfüllen, d.h. es gilt unter anderem

$$\alpha_i \left(y_i \left(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 \right) - 1 \right) = 0 \; \forall \, i = 1, \ldots, N$$

$\vec{\alpha}^*$ enthält für jedes Beispiel $\vec{x}_i$ genau ein α_i mit

$$\begin{aligned}
\alpha_i = 0 \quad &, \quad \text{falls } \vec{x}_i \text{ im richtigen Halbraum liegt} \\
\alpha_i > 0 \quad &, \quad \text{falls } \vec{x}_i \text{ auf der Hyperebene } H_1 \text{ oder } H_2 \text{ liegt}
\end{aligned}$$

Ein Beispiel $\vec{x}_i$ mit $\alpha_i > 0$ heißt Stützvektor.

Optimale Hyperebene

Haben wir das optimale $\vec{\alpha}^*$ bestimmt, erhalten wir unsere optimale Hyperebene. Nach (34) gilt

$$\vec{\beta} = \sum \alpha_i y_i \vec{x}_i$$

d.h. der optimale Normalenvektor $\vec{\beta}$ ist eine Linearkombination von Stützvektoren. Um β_0 zu bestimmen können wir

$$\alpha_i \left(y_i \left(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 \right) - 1 \right) = 0$$

für ein beliebiges i und unser berechnetes $\vec{\beta}$ nutzen.

Berechnung der α_i?

Das prinzipielle Vorgehen ist bei der SVM wie bei anderen Lernverfahren auch:

- Parametrisierung der Modelle, hier über Umwege durch $\vec{\alpha}$
- Festlegung eines Optimalitätskriteriums, hier: *Maximum Margin*
- Formulierung als Optimierungsproblem

Das finale Optimierungsproblem läßt sich mit unterschiedlichen Ansätzen lösen

- Numerische Verfahren (*quadratic problem solver*)
- *Sequential Minimal Optimization* (SMO, [J. C. Platt, 1998])
- Evolutionäre Algorithmen (EvoSVM, [I. Mierswa, 2006])

Zusammenfassung der Lagrange-Optimierung für SVM

Das Lagrange-Optimierungs-Problem (33) ist definiert als:

$$L_P = \frac{1}{2}\|\vec{\beta}\|^2 - \sum_{i=1}^{N} \alpha_i \left[y_i(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0) - 1 \right]$$

mit den *Lagrange-Multiplikatoren* $\vec{\alpha}_i \geq 0$. Notwendige Bedingung für ein Minimum liefern die Ableitungen nach $\vec{\beta}$ und β_0

$$\frac{\partial L_P}{\partial \vec{\beta}} = \vec{\beta} - \sum_{i=1}^{N} \alpha_i y_i \vec{x}_i \quad \text{und} \quad \frac{\partial L_P}{\partial \beta_0} = \sum_{i=1}^{N} \alpha_i y_i$$

Diese führen zum *dualen Problem* (37)

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} \langle \vec{x}_i, \vec{x}_{i'} \rangle$$

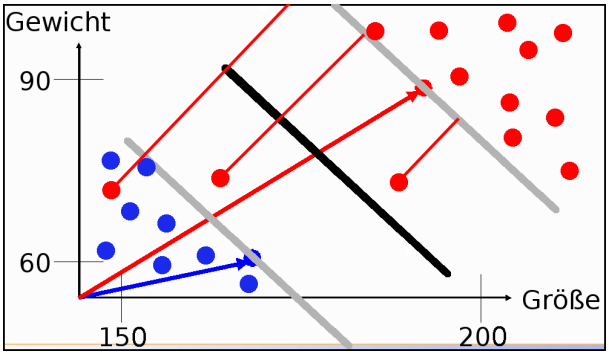
Was wissen wir jetzt?

- Maximieren des Margins einer Hyperebene ergibt eine eindeutige Festlegung der optimalen trennenden Hyperebene.
- Dazu minimieren wir die Länge des Normalenvektors $\vec{\beta}$
  - Formulierung als Lagrange-Funktion
  - Formulierung als duales Optimierungsproblem
- Das Lernergebnis ist eine Linearkombination von Stützvektoren.
- Mit den Beispielen müssen wir nur noch das Skalarprodukt rechnen.

13.4 Weich trennende SVM

SVM mit Ausnahmen

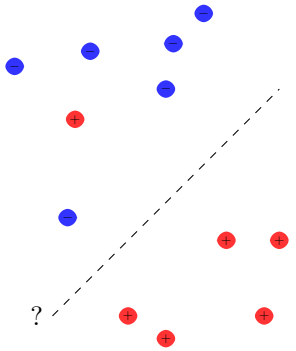
- Was passiert, wenn die Beispiele nicht komplett trennbar sind?



Nicht linear trennbare Daten

In der Praxis sind linear trennbare Daten selten:

- 1. Ansatz: Entferne eine minimale Menge von Datenpunkten, so dass die Daten linear trennbar werden (minimale Fehlklassifikation).
- Problem: Algorithmus wird exponentiell.



SVM mit Ausnahmen

Ein anderer Ansatz basiert wieder auf einer Relaxation:

- Punkte, die nicht am Rand oder auf der richtigen Seite der Ebene liegen, bekommen einen Strafterm $\xi_j > 0$.
- Korrekt klassifizierte Punkte erhalten eine Variable $\xi_j = 0$.

Dies führt zu folgenden Minimierungsproblem

$$\frac{1}{2}\|\vec{\beta}\|^2 + C\sum_{j=1}^{N}\xi_j \quad \text{für ein festes } C \in \mathbb{R}_{>0} \tag{38}$$

Daraus folgt insbesondere

$$0 \leq \alpha_i \leq C$$

Weich trennende Hyperebene

Relaxiertes Optimierungsproblem

Sei $C \in \mathbb{R}$ mit $C > 0$ fest. Minimiere

$$||\vec{\beta}||^2 + C\sum_{i=1}^{N}\xi_i$$

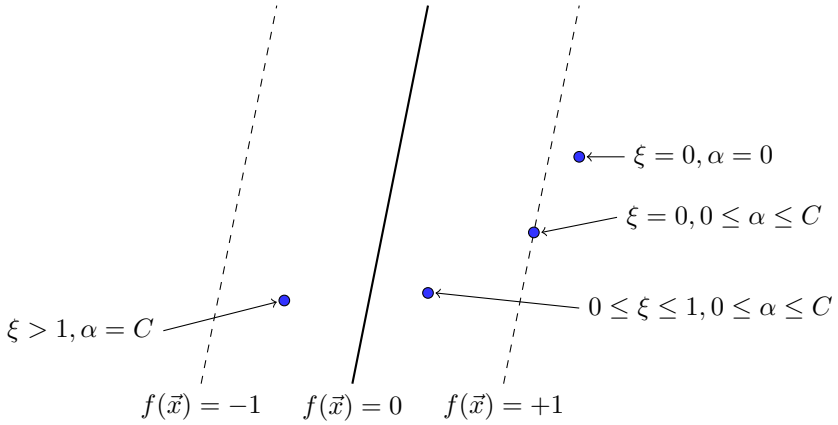
unter den Nebenbedingungen

$$\begin{aligned} \langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 &\geq +1 - \xi_i \quad \text{für } \vec{y}_i = +1 \\ \langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 &\leq -1 + \xi_i \quad \text{für } \vec{y}_i = -1 \end{aligned}$$

Durch Umformung erhalten wir wieder Bedingungen für die Lagrange-Optimierung:

$$y_i(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0) \geq 1 - \xi_i \quad \forall\, i = 1, \ldots, N$$

Bedeutung von ξ und $\vec{\alpha}$



Beispiele $\vec{x}_i$ mit $\alpha_i > 0$ sind Stützvektoren.

Wo sind wir?

- Maximieren der Breite einer separierenden Hyperebene (*maximum margin method*) ergibt eindeutige, optimale trennende Hyperebene.
  - Das haben wir heute in der Theorie für linear separierbare Beispielmengen und mit weicher Trennung gesehen – wie es praktisch geht, sehen wir nächstes Mal.
  - Die Grundlagen waren die selben wie bei den *linearen Modellen*.
- Transformation des Datenraums durch Kernfunktion behandelt Nichtlinearität.
  - Das kam nur einmal am Rande vor. Wir sehen es nächstes Mal genauer.
  - Es baut auf die Behandlung der Nichtlinearität durch die *Basisexpansion* auf.
- Strukturelle Risikominimierung minimiert nicht nur den Fehler, sondern auch die Komplexität des Modells. Später!

13.5 Lösung des Optimierungsproblems mit SMO

Optimierungsproblem der SVM

Die Lösung $\vec{\alpha}^*$ des dualen Problems

$$L_D(\vec{\alpha}) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle \vec{x}_i, \vec{x}_j \rangle$$

muss die KKT-Bedingungen erfüllen, d.h. es gilt unter anderem

$$\alpha_i \left(y_i \left(\langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 \right) - 1 \right) = 0 \; \forall\, i = 1, \ldots, N$$

$\vec{\alpha}^*$ enthält für jedes Beispiel $\vec{x}_i$ genau ein α_i mit

$$\begin{aligned} \alpha_i = 0 \quad &, \quad \text{falls } \vec{x}_i \text{ im richtigen Halbraum liegt} \\ \alpha_i > 0 \quad &, \quad \text{falls } \vec{x}_i \text{ auf der Hyperebene } H_1 \text{ oder } H_2 \text{ liegt} \end{aligned}$$

Ein Beispiel $\vec{x}_i$ mit $\alpha_i > 0$ heißt Stützvektor.

Optimierungsproblem für weiche Trennung

Sei $C \in \mathbb{R}$ mit $C > 0$ fest. Minimiere

$$||\vec{\beta}||^2 + C \sum_{i=1}^{N} \xi_i$$

unter den Nebenbedingungen

$$\begin{aligned} \langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 &\geq +1 - \xi_i \quad \text{für } \vec{y}_i = +1 \\ \langle \vec{x}_i, \vec{\beta} \rangle + \beta_0 &\leq -1 + \xi_i \quad \text{für } \vec{y}_i = -1 \end{aligned}$$

Optimierungsproblem zur Minimierung

- Erst minimierten wir $\vec{\beta}$ (primales Problem), dann maximierten wir α (duales Problem), jetzt minimieren wir das duale Problem, indem wir alles mit -1 multiplizieren...
- Minimiere $L'_D(\alpha)$

  $$\frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} y_i y_j K(x_i, x_j) \alpha_i \alpha_j - \sum_{i=1}^{m} \alpha_i$$

  unter den Nebenbedingungen $0 \leq \alpha_i \leq C$

  $$\sum_{i=1}^{m} y_i \alpha_i = 0$$

Algorithmus?

- Berechnen wir $L'_D(\alpha)$ durch Gradientensuche!
  - Naiver Ansatz berechnet Gradienten an einem Startpunkt und sucht in angegebener Richtung ... Bis kleinster Wert gefunden ist. Dabei wird immer die Nebenbedingung eingehalten. Bei m Beispielen hat α m Komponenten, nach denen es optimiert werden muss. Alle Komponenten von α auf einmal optimieren? m^2 Terme!
  - Eine Komponente von *alpha* ändern? Nebenbedingung verletzt.
  - Zwei Komponenten α_1, α_2 im Bereich $[0, C] \times [0, C]$ verändern!

Sequential Minimal Optimization

- Wir verändern α_1, α_2 , lassen alle anderen α_i fest. Die Nebenbedingung wird zu:

  $$\alpha_1 y_i + \alpha_2 y_2 = -\sum_{i=3}^{m} \alpha_i y_i$$

- Zulässige α_1, α_2 liegen im Bereich $[0, C] \times [0, C]$ auf der Geraden $W = \alpha_1 y_1 + \alpha_2 y_2$ äquivalent $\alpha_1 + s\alpha_2$ mit $s = \frac{y_2}{y_1}$
- Wir optimieren α_2
- Aus dem optimalen $\hat{\alpha}_2$ können wir das optimale $\hat{\alpha}_1$ herleiten:

  $$\hat{\alpha}_1 = \alpha_1 + y_1 y_2 (\alpha_2 - \hat{\alpha}_2)$$

- Dann kommen die nächsten zwei α_i dran...

α_2 optimieren

- Maximum der Funktion $L'_D(\alpha)$ entlang der Geraden $s\alpha_2 + \alpha_1 = d$.
- Wenn $y_1 = y_2$ ist $s = 1$, also steigt die Gerade. Sonst $s = -1$, also fällt die Gerade.
- Schnittpunkte der Geraden mit dem Bereich $[0, C] \times [0, C]$:
  - Falls s steigt: $\max(0; \alpha_2 + \alpha_1 - C)$ und $\min(C; \alpha_2 + \alpha_1)$
  - Sonst: $\max(0; \alpha_2 - \alpha_1)$ und $\min(C; \alpha_2 - \alpha_1 + C)$
  - Optimales α_2 ist höchstens max-Term, mindestens min-Term.

Bestimmen der αs

- $k = \alpha_1^{old} + s\alpha_2^{old} = \alpha_1^{new} + s\alpha_2^{new}$
- Mit Hilfe der Optimierungsquadrate lassen sich untere und obere Schranken für α_2 bestimmen:
  - $y_1 = y_2 : L = max(0, \alpha_1^{old} + \alpha_2^{old} - C)\ H = min(C, \alpha_1^{old} + \alpha_2^{old})$
  - $y_1 \neq y_2 : L = max(0, \alpha_2^{old} - \alpha_1^{old})\ H = min(C, C + \alpha_2^{old} - \alpha_1^{old})$
- Ableiten des Dualen Problems nach α_2 ergibt das Optimum für α_2^{new}
  - $\alpha_2^{new} = \alpha_2^{old} + \frac{y_2((f(\vec{x_1}) - y_1) - (f(\vec{x_2}) - y_2))}{\eta}$
  - $= \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta}$
  - $\eta = x_1^T x_1 + x_2^T x_2 - 2x_1^T x_2$

Optimales α_2

- Sei $\alpha = (\alpha_1, \ldots, \alpha_N)$ eine Lösung des Optimierungsproblems. Wir wählen zum update:

$$\hat{\alpha}_2 = \alpha_2 + \frac{y_2\left((f(x_1) - y_1) - (f(x_2) - y_2)\right)}{K(x_1, x_1) - 2K(x_1, x_2) + K(x_2, x_2)}$$

- Optimales $\hat{\alpha}_1 = \alpha_1 + y_1 y_2 \left(\alpha_2 - \hat{\alpha}_2\right)$
- Prinzip des Optimierens: Nullsetzen der ersten Ableitung...

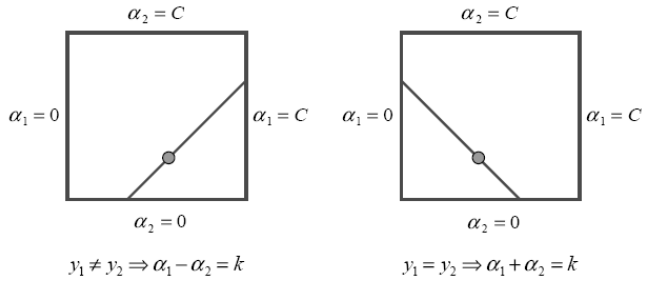
Optimierungsalgorithmus

1:	g = Gradient von $L'_D(\alpha)$	$g_i = \sum \alpha_k y_k y_i (x_k * x_i) - 1$
2:	WHILE nicht konvergiert(g)	auf ϵ genau
3:	WS=working set(g)	suche k "gute" Variablen
4:	α'=optimiere(WS)	k neue α-Werte
5:	g=aktualisiere(g, α')	g = Gradient von $L'_D(\alpha')$

- Gradientensuchverfahren
- Stützvektoren allein definieren die Lösung
- Tricks: Shrinking und Caching von $x_i * x_j$

Ermitteln der αs im Bild

- Alle αs zu optimieren ist zu komplex.
- Nur ein α zur Zeit zu optimieren, verletzt $0 = \sum_{i=1}^{N} \alpha_i y_i$
- Also: zwei αs gleichzeitig optimieren!
- Man optimiert beide innerhalb eines Quadrates...

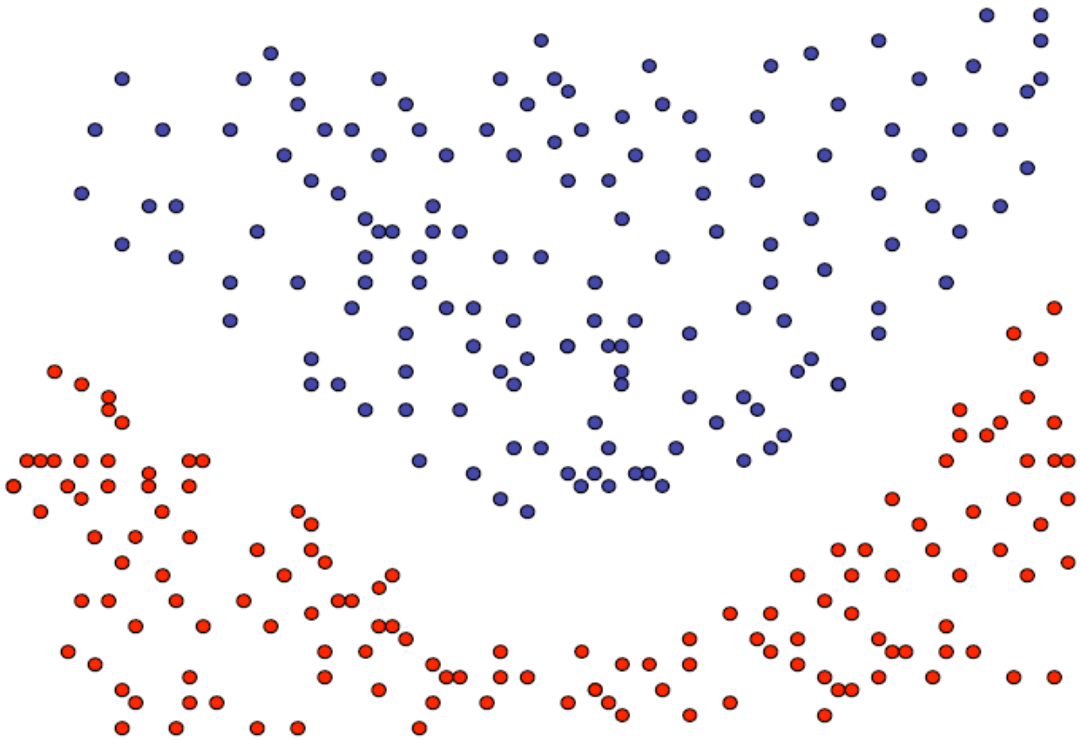


Was wissen wir jetzt?

- Der SMO-Algorithmus ist *einer* der Optimierungsalgorithmen für das duale Problem.
- Man kann auch z.B. per Evolutionsalgorithmus optimieren (Mierswa 2006).
- Oder mit der *cutting plane* Methode (Kelley 1960) (Joachims 2006)
- ...

13.6 Kernfunktionen

Nicht-lineare Daten

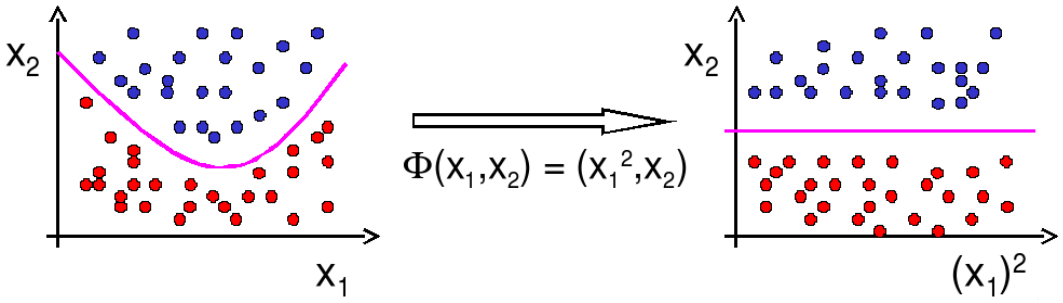


Nicht-lineare Daten

- Neue SVM-Theorie entwickeln? (Neeee!)
- Lineare SVM benutzen?

  If all you've got is a hammer, every problem looks like a nail

- Transformation in lineares Problem!



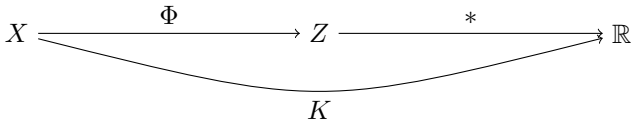
Kernfunktionen

- Erinnerung:

$$L_D(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} y_i y_j \alpha_i \alpha_j \, (\vec{x_i} * \vec{x_j})$$

$$f(\vec{x}) = \sum \alpha_i y_i \, (\vec{x_i} * \vec{x}) + \beta_0$$

- SVM hängt von $\vec{x}$ nur über Skalarprodukt $\vec{x} * \vec{x'}$ ab.
- Ersetze Transformation Φ und Skalarprodukt $*$ durch Kernfunktion $K(\vec{x_1}, \vec{x_2}) = \Phi(\vec{x_1}) * \Phi(\vec{x_2})$

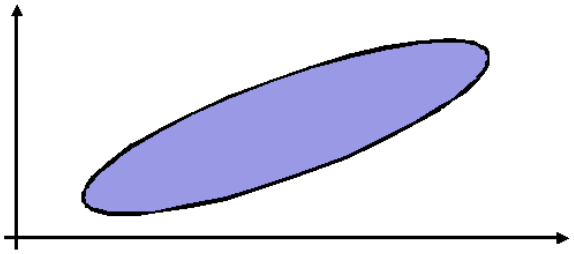


Kernfunktionen II

- Angabe von ϕ nicht nötig, einzige Bedingung: Kernmatrix $(K(\vec{x_i}, \vec{x_j}))_{i,j=1\ldots N}$ muss positiv definit sein.
- Radial-Basisfunktion: $K(\vec{x_i}, \vec{x_j}) = \exp(-\gamma \|\vec{x_i} - \vec{x_j}\|^2)$
- Polynom: $K(\vec{x_i}, \vec{x_j}) = (\vec{x_i} * \vec{x_j})^d$
- Neuronale Netze: $K(\vec{x_i}, \vec{x_j}) = \tanh(\alpha \vec{x_i} * \vec{x_j} + b)$
- Konstruktion von Spezialkernen durch Summen und Produkte von Kernfunktionen, Multiplikation mit positiver Zahl, Weglassen von Attributen

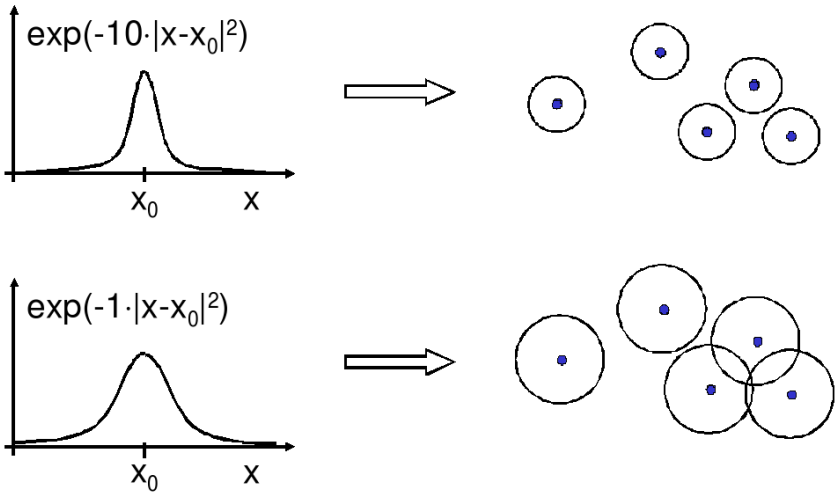
Polynom-Kernfunktionen

- $K_d(\vec{x_i}, \vec{x_j}) = (\vec{x_i} * \vec{x_j})^d$
- Beispiel: $d = 2, \vec{x_i}, \vec{x_j} \in \mathbb{R}^2$.



$$
\begin{aligned}
K_2(\vec{x_i}, \vec{x_j}) &= (\vec{x_i} * \vec{x_j})^2 \\
&= ((x_{i_1}, x_{i_2}) * (x_{j_1}, x_{j_2}))^2 = (x_{i_1}x_{j_1} + x_{i_2}x_{j_2})^2 \\
&= x_{i_1}^2 x_{j_1}^2 + 2x_{i_1}x_{j_1}x_{i_2}x_{j_2} + x_{i_2}^2 x_{j_2}^2 \\
&= (x_{i_1}^2, \sqrt{2}x_{i_1}x_{i_2}, x_{i_2}^2) * (x_{j_1}^2, \sqrt{2}x_{j_1}x_{j_2}, x_{j_2}^2) \\
&=: \phi(\vec{x_i}) * \phi(\vec{x_j})
\end{aligned}
$$

RBF-Kernfunktion



Kernfunktionen – Basisexpansionen

- Die Basisexpansionen waren ein tatsächlicher Schritt der Vorverarbeitung.
- Die Kernfunktionen werden nicht als Vorverarbeitungsschritt durchgeführt.
- Man muss lediglich bei der Berechnung des Skalarprodukts die Kernfunktion berücksichtigen.
- Allerdings kann $\vec{\beta}$ jetzt nicht mehr so einfach interpretiert werden als Bedeutung der Variablen (Merkmale) X_i.

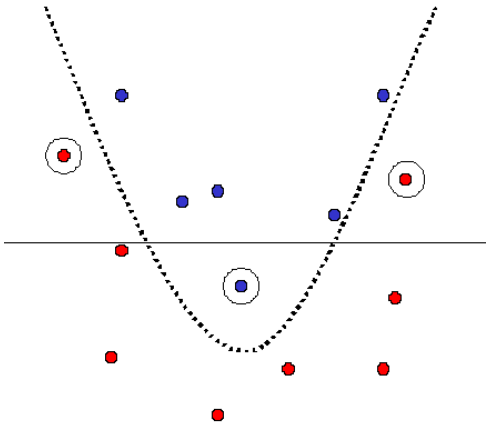
13.7 Bias und Varianz bei SVM

Was ist gutes Lernen?

- Fauler Botaniker: “klar ist das ein Baum - ist ja grün.”
  - Übergeneralisierung
  - Wenig Kapazität
  - Bias
- Botaniker mit fotografischem Gedächtnis: “nein, dies ist kein Baum, er hat 15 267 Blätter und kein anderer hatte genau so viele.”
  - Overfitting
  - Viel Kapazität
  - Varianz
- Kontrolle der Kapazität!

Bias-Varianz-Problem

- Zu kleiner Hypothesenraum: Zielfunktion nicht gut genug approximierbar (Bias)
- Zu großer Hypothesenraum: Zuviel Einfluss zufälliger Abweichungen (Varianz)
- Lösung: Minimiere obere Schranke des Fehlers: $R(\alpha) \leq_\eta R_{emp}(\alpha) + Var(\alpha)$



Risikoschranke nach Vapnik

Definition 13.6 (Strukturelles Risiko)**.** *Gegeben eine unbekannte Wahrscheinlichkeitsverteilung $P(\vec{x}, y)$, nach der Daten gezogen werden. Die Abbildungen $\vec{x} \to f(\vec{x}, \vec{\alpha})$ werden dadurch gelernt, dass $\vec{\alpha}$ bestimmt wird. Mit einer Wahrscheinlichkeit $1 - \mu$ ist das Risiko $R(\vec{\alpha})$ nach dem Sehen von N Beispielen beschränkt:*

$$R(\vec{\alpha}) \leq R_{emp}(\vec{\alpha}) + \underbrace{\sqrt{\frac{\eta\left(\log\left(\frac{2N}{\eta}\right)+1\right) - \log\left(\frac{\mu}{4}\right)}{N}}}_{VC\ confidence}$$

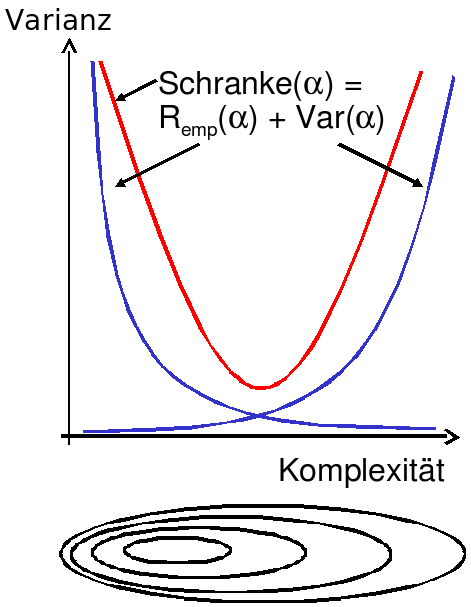
Bevor wir η ergründen (Vapnik-Chervonenkis-Dimension), erst einmal festhalten, was die Bedeutung dieser Schranke ist!

Strukturelle Risikoschranke

- Unabhängig von einer Verteilungsannahme. Alles, was die Schranke braucht, ist, dass Trainings- und Testdaten gemäß der selben Wahrscheinlichkeitsverteilung gezogen werden.
- Das tatsächliche Risiko können wir nicht berechnen.
- Die rechte Seite der Ungleichung können wir berechnen, sobald wir η kennen, die Vapnik-Chervonenkis-Dimension.
- Gegeben eine Menge Hypothesen für $f(\vec{x}, \vec{\alpha})$, wähle immer die mit dem niedrigsten Wert für die rechte Seite der Schranke (R_{emp} oder VC confidence niedrig).

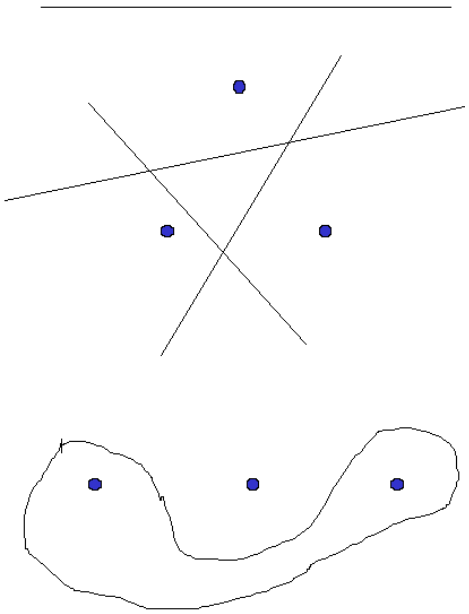
Strukturelle Risikominimierung

1. Ordne die Hypothesen in Teilmengen gemäß ihrer Komplexität.
2. Wähle in jeder Teilmenge die Hypothese mit dem geringsten empirischen Fehler.
3. Wähle insgesamt die Hypothese mit minimaler Risikoschranke.



Vapnik-Chervonenkis-Dimension

- Definition: Eine Menge H von Hypothesen zerschmettert eine Menge E von Beispielen, wenn jede Teilmenge von E durch ein $h \in H$ abgetrennt werden kann.
- Definition: Die VC-Dimension einer Menge von Hypothesen H ist die maximale Anzahl von Beispielen E, die von H zerschmettert wird.
- Eine Menge von 3 Punkten kann von geraden Linien zerschmettert werden, keine Menge von 4 Punkten kann von geraden Linien zerschmettert werden.



ACHTUNG

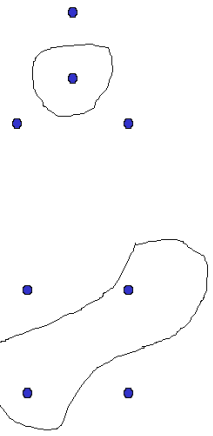
- Für eine Klasse von Lernaufgaben gibt es mindestens eine Menge E, die zerschmettert werden kann - NICHT jede Menge E kann zerschmettert werden!
- Zum Beweis der VC Dimension n muss man also zeigen:
  - Es gibt eine Menge E aus n Punkten, die von H zerschmettert werden kann. $VCdim(H) \geq n$
  - Es kann keine Menge E' aus $n+1$ Punkten geben, die von H zerschmettert werden könnte. $VCdim(H) \leq n$

VC-Dimension von Hyperebenen

Satz 13.2 (VC-Dimension von Hyperebenen)**.** *Satz: Die VC-Dimension der Hyperebenen im R^p ist $p+1$.*

Beweis:

- $VCdim(R^p) \geq p+1$: Wähle $\vec{x_0} = 0$ und $\vec{x_i} = (0, \ldots, 0, 1, 0, \ldots 0)$. Für eine beliebige Teilmenge A von $(\vec{x_0}, \ldots, \vec{x_n})$ setze $y_i = 1$, falls $\vec{x_i} \in A$, sonst $y_i = -1$. Definiere $\vec{\beta} = \sum y_k \vec{x_k}$ und $\beta_0 = \frac{y_0}{2}$. Dann gilt $\vec{\beta}\vec{x_0} + \beta_0 = \frac{y_0}{2}$ und $\vec{\beta}\vec{x_i} + \beta_0 = y_i + \frac{y_0}{2}$. Also: $\vec{\beta}\vec{x} + \beta_0$ trennt A.
- $VCdim(R^p) \leq p+1$: Zurückführen auf die beiden Fälle unten.



VCdim misst Kapazität

- Eine Funktion mit nur 1 Parameter kann unendliche $VCdim$ haben: H kann Mengen von n Punkten zerschmettern, egal wie groß n ist.
- H kann unendliche $VCdim$ haben und trotzdem kann ich eine kleine Zahl von Punkten finden, die H nicht zerschmettern kann.
- $VCdim$ ist also nicht groß, wenn die Anzahl der Parameter bei der Klasse von Funktionen H groß ist.

VC-Dimension der SVM

- Gegeben seien Beispiele $\vec{x_1}, \ldots, \vec{x_N} \in \mathcal{R}^p$ mit $\|\vec{x_i}\| < D$ für alle i. Für die VC-Dimension der durch den Vektor $\vec{\beta}$ gegebenen optimalen Hyperebene H gilt:

$$VCdim(H) \leq \min\left\{D^2\|\vec{\beta}\|^2, p\right\} + 1$$

- Die Komplexität einer SVM ist auch durch die Struktur der Lösung begrenzt!
- Die SVM minimiert nicht nur das empirische Risiko, sondern auch das strukturelle.

Zusicherungen

- Strukturelle Risikominimierung garantiert, dass die einfachste Hypothese gewählt wird, die noch an die Daten anpassbar ist.
- Strukturelle Risikominimierung kontrolliert die Kapazität des Lernens (weder fauler noch fotografischer Botaniker).
- Die Strukturen von Klassen von Funktionen werden durch die $VCdim$ ausgedrückt. Große $VCdim \rightarrow$ große VC-confidence.
- Wir haben nun also ein Verfahren, das ohne zusätzlichen Aufwand die Komplexität regularisiert, wie wir es bei der *Modellselektion* für lineare und lokale Modelle mal wollten.

Performanzschätzer

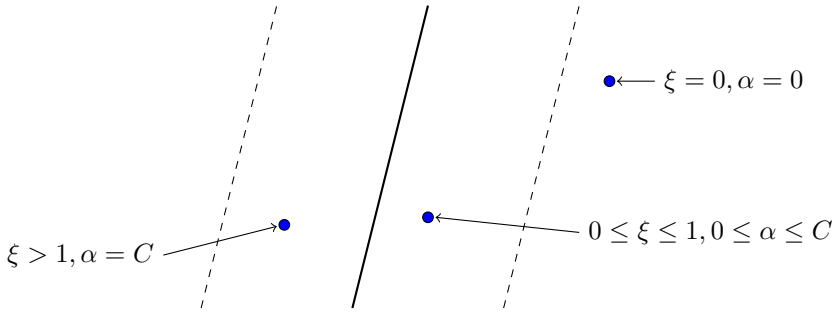
- Welches erwartete Risiko $R(\alpha)$ erreicht SVM?
- $R(\vec{\alpha})$ selbst nicht berechenbar
- Trainingsfehler (zu optimistisch - Overfitting)
- Obere Schranke mittels VC-Dimension (zu locker)
- Kreuzvalidierung / Leave-One-Out-Schätzer (ineffizient)

Performanzschätzer II

- Satz: Der Leave-One-Out-Fehler einer SVM ist beschränkt durch $R_{l1o} \leq \frac{|SV|}{N}$
- Beweis (Skizze):
  - Falsch klassifizierte Beispiele werden Stützvektoren (SV).
  - Also: Nicht-Stützvektoren werden korrekt klassifiziert. Weglassen eines Nicht-Stützvektors ändert die Hyperebene nicht, daher wird es auch beim $l1o$-Test richtig klassifiziert.
  - Nur der Anteil der Stützvektoren an den Beispielen macht den Fehler aus.

Performanzschätzer III

- Satz: Der Leave-One-Out-Fehler einer SVM ist beschränkt durch $R_{l1o} \leq \frac{|\{i:(2\alpha_i D^2+\xi_i)\geq 1\}|}{N}$ (D = Radius des Umkreises um die Beispiele im transformierten Raum).
- Beweis: Betrachte folgende drei Fälle:



Was wissen wir jetzt?

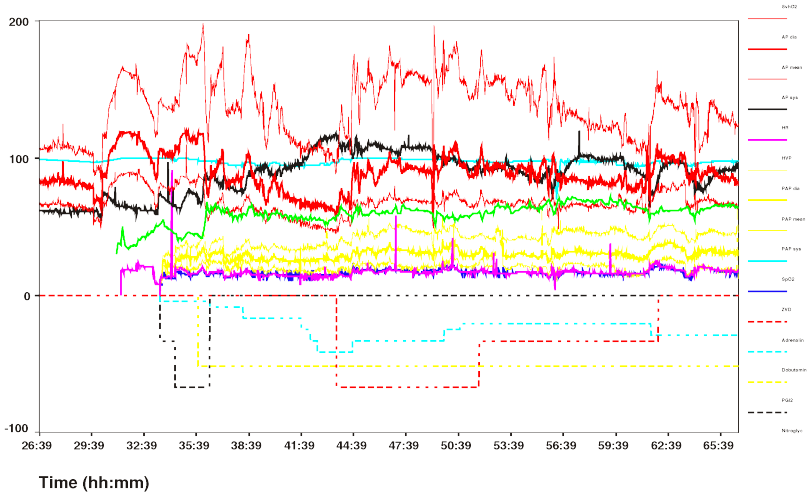
- Kernfunktionen - eine Transformation, die man nicht erst durchführen und dann mit ihr rechnen muss, sondern bei der nur das Skalarprodukt gerechnet wird.
- Idee der strukturellen Risikominimierung (SRM):
  - obere Schranke für das Risiko
  - Schrittweise Steigerung der Komplexität
- Formalisierung der Komplexität: VC-Dimension
- SRM als Prinzip der SVM
- Garantie für die Korrektheit der Lernstrategie

13.8 Anwendungen

Fallstudie Intensivmedizin

- Städtische Kliniken Dortmund, Intensivmedizin 16 Betten, Prof. Dr. Michael Imhoff (Ruhr-Universität Bochum)
- Hämodynamisches Monitoring, minütliche Messungen
  - Diastolischer, systolischer, mittlerer arterieller Druck
  - Diastolischer, systolischer, mittlerer pulmonarer Druck
  - Herzrate
  - Zentralvenöser Druck
- Therapie, Medikamente:
  - Dobutamine, adrenaline, glycerol trinitrate, noradrenaline, dopamine, nifedipine

Patient G.C., male, 60 years old - Hemihepatektomie right



Wann wird Medikament gegeben?

- Mehrklassenproblem in mehrere 2-Klassen-Probleme umwandeln:
  - Für jedes Medikament entscheide, ob es gegeben werden soll oder nicht.
  - Positive Beispiele: alle Minuten, in denen das Medikament gegeben wurde
  - Negative Beispiele: alle Minuten, in denen das Medikament nicht gegeben wurde

Parameter: Kosten falscher Positiver = Kosten falscher Negativer

Ergebnis: Gewichte der Vitalwerte $\vec{\beta}$, so dass positive und negative Beispiele maximal getrennt werden (SVM).

Beispiel: Intensivmedizin

$$f(\vec{x}) = \left[\begin{pmatrix} 0.014 \\ 0.019 \\ -0.001 \\ -0.015 \\ -0.016 \\ 0.026 \\ 0.134 \\ -0.177 \\ \vdots \end{pmatrix} \begin{pmatrix} artsys = 174.00 \\ artdia = 86.00 \\ artmn = 121.00 \\ cvp = 8.00 \\ hr = 79.00 \\ papsys = 26.00 \\ papdia = 13.00 \\ papmn = 15.00 \\ \vdots \end{pmatrix} - 4.368 \right]$$

Wie wird ein Medikament dosiert ?

- Mehrklassenproblem in mehrere 2 Klassenprobleme umwandeln: für jedes Medikament und jede Richtung (increase, decrease, equal), 2 Mengen von Patienten-daten:
  - Positive Beispiele: alle Minuten, in denen die Dosierung in der betreffenden Richtung geändert wurde
  - Negative Beispiele: alle Minuten, in denen die Dosierung nicht in der betreffenden Richtung geändert wurde.

Steigern von Dobutamine

Vektor $\vec{\beta}$ für p Attribute

$ARTEREN$:	-0.05108108119
$SUPRA$:	0.00892807538657973
$DOBUTREX$:	-0.100650806786886
$WEIGHT$:	-0.0393531801046265
AGE :	-0.00378828681071417
$ARTSYS$:	-0.323407537252192
$ARTDIA$:	-0.0394565333019493
$ARTMN$:	-0.180425080906375
HR :	-0.10010405264306
$PAPSYS$:	-0.0252641188531731
$PAPDIA$:	0.0454843337112765
$PAPMN$:	0.00429504963736522
$PULS$:	-0.0313501236399881

Anwendung des Gelernten für Dobutamin

- Patientwerte pat46, artmn 95, min. 2231 ... pat46, artmn 90, min. 2619
- Gelernte Gewichte β_i: $artmn - 0,18$ ...

$$svm\_calc = \sum_{i=1}^{p} \beta_i x_i$$

$$decision = sign(svm\_calc + \beta_0)$$

- $svm\_calc(pat46, dobutrex, up, min.2231, 39)$
- $svm\_calc(pat46, dobutrex, up, min.2619, 25)$
- $\beta_0 = -26$, i.e. increase in minute 2231, not increase in minute 2619.

Steigern von Glyceroltrinitrat (nitro)

$$f(x) = \left[\begin{pmatrix} 0.014 \\ 0.019 \\ -0.001 \\ -0.015 \\ -0.016 \\ 0.026 \\ 0.134 \\ -0.177 \\ -9.543 \\ -1.047 \\ -0.185 \\ 0.542 \\ -0.017 \\ 2.391 \\ 0.033 \\ 0.334 \\ 0.784 \\ 0.015 \end{pmatrix} \begin{pmatrix} artsys = 174.00 \\ artdia = 86.00 \\ artmn = 121.00 \\ cvp = 8.00 \\ hr = 79.00 \\ papsys = 26.00 \\ papdia = 13.00 \\ papmn = 15.00 \\ nifedipine = 0 \\ noradrenaline = 0 \\ dobutamie = 0 \\ dopamie = 0 \\ glyceroltrinitrate = 0 \\ adrenaline = 0 \\ age = 77.91 \\ emergency = 0 \\ bsa = 1.79 \\ broca = 1.02 \end{pmatrix} - 4.368 \right]$$

- Jedes Medikament hat einen Dosierungsschritt. Für Glyceroltrinitrat ist es 1, für *Suprarenin* (adrenalin) 0,01. Die Dosis wird um einen Schritt erhöht oder gesenkt.
- Vorhersage:$pred\_interv(pat49, min.32, nitro, 1, 0)$

Evaluierung

- Blind test über 95 noch nicht gesehener Patientendaten.
  - Experte stimmte überein mit tatsächlichen Medikamentengaben in 52 Fällen
  - SVM Ergebnis stimmte überein mit tatsächlichen Medikamentengaben in 58 Fällen

Dobutamine	*Actual up*	*Actual equal*	*Actual down*
Predicted up	**10 (9)**	12 (8)	0 (0)
Predicted equal	7 (9)	**35 (31)**	9 (9)
Predicted down	2 (1)	7 (15)	**13 (12)**

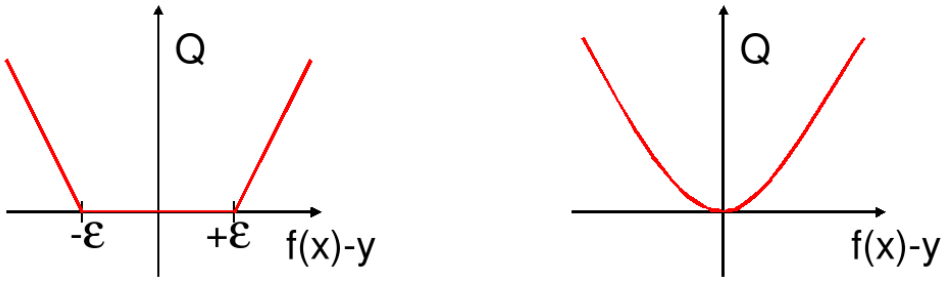
SVMs für Regession

Durch Einführung einer anderen *Loss-Funktion* läßt sich die SVM zur Regression nutzen. Sei $\varepsilon \in \mathbb{R}_{>0}$ und

$$L_k(y, f(\vec{x}, \alpha)) = \begin{cases} 0 & \text{, falls } y - f(\vec{x}, \alpha) \leq \varepsilon \\ (y - f(\vec{x}, \alpha) - \varepsilon)^k & \text{, sonst} \end{cases}$$

Die *Loss-Funktion* L_1 gibt den Abstand der Funktion f von den Trainingsdaten an, alternativ quadratische Loss-Funktion L_2:

lineare Verlustfunktion quadratische Verlustfunktion



SVMs für Regression

Dadurch ergibt sich das Optimierungsproblem:

Definition 13.7 (Regressions-SVM)**.** *Minimiere*

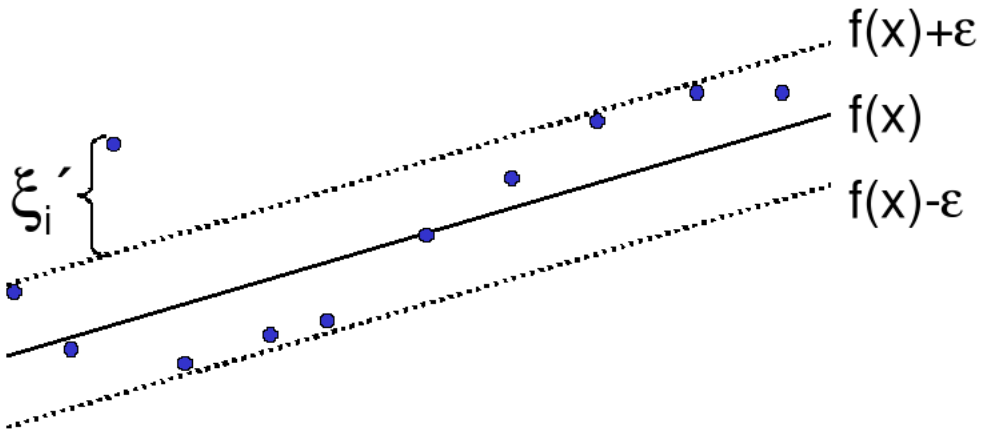
$$\|\vec{\beta}\|^2 + C\left(\sum_{i=1}^{N} \xi_i + \sum_{i=1}^{N} \xi_i'\right)$$

unter den Nebenbedingungen

$$\begin{aligned} f(\vec{x_i}) &= \langle \vec{\beta}, \vec{x_i} \rangle + \beta_0 \leq y_i + \epsilon + \xi_i' \\ f(\vec{x_i}) &= \langle \vec{\beta}, \vec{x_i} \rangle + \beta_0 \geq y_i - \epsilon - \xi_i \end{aligned}$$

SVMs für Regression

Die ξ_i bzw. ξ_i' geben für jedes Beispiel Schranken an, innerhalb derer der vorhergesagte Funktionswert für jedes Beispiel liegen soll:



Bei der Lösung des Optimierungsproblems mit Lagrange führt dies zu *zwei* α-Werten je Beispiel!

SVMs für Regression

Das duale Problem enthält für jedes $\vec{x}_i$ je zwei α-Werte α_i und α_i', je einen für ξ_i und ξ_i', d.h.

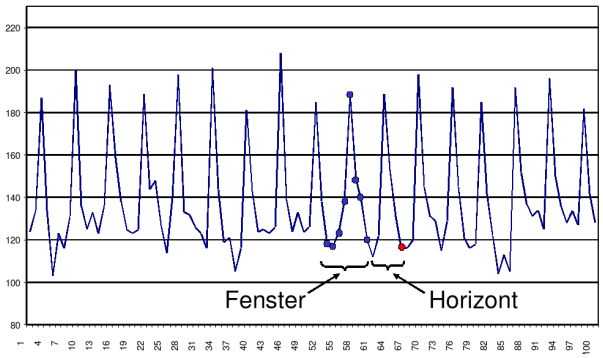
Definition 13.8 (Duales Problem für die Regressions-SVM)**.** *Maximiere*

$$\begin{aligned} L_D(\vec{\alpha}, \vec{\alpha}') &= \sum_{i=1}^{N} y_i \left(\alpha_i' - \alpha_i\right) - \epsilon \sum_{i=1}^{N} y_i \left(\alpha_i' - \alpha_i\right) \\ &\quad -\frac{1}{2} \sum_{i,j=1}^{n} y_i \left(\alpha_i' - \alpha_i\right) \left(\alpha_j' - \alpha_j\right) K(\vec{x_i}, \vec{x_j}) \end{aligned}$$

unter den Nebenbedingungen

$$0 \leq \alpha_i, \alpha_i' \leq C \,\forall\, i = 1, \ldots, N \quad \textit{und} \quad \sum_{i=1}^{N} \alpha_i' = \sum_{i=1}^{N} \alpha_i$$

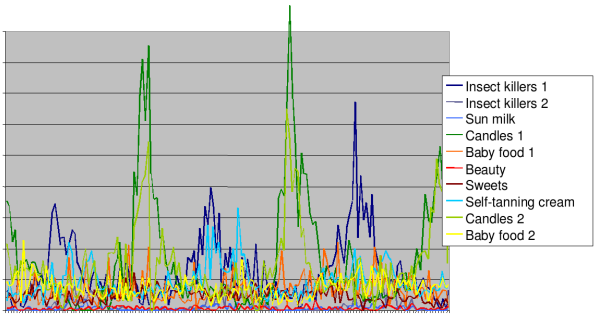
Beispiel: Prognose von Zeitreihen



Prognose von Zeitreihen

- Trend
- Zyklen
- Besondere Ereignisse (Weihnachten, Werbung, ...)
- Wieviel vergangene Beobachtungen?
- Ausreißer

Abverkauf Drogerieartikel



Vorhersage Abverkauf

Gegeben: Verkaufsdaten von 50 Artikeln in 20 Läden über 104 Wochen

Vorhersage: Verkäufe eines Artikels, so dass

- Die Vorhersage niemals den Verkauf unterschätzt,
- Die Vorhersage überschätzt weniger als eine Faustregel.

Beobachtung: 90% der Artikel werden weniger als 10 mal pro Woche verkauft.

Anforderung: Vorhersagehorizont von mehr als 4 Wochen.

Verkaufsdaten – multivariate Zeitreihen

Shop	Week	Item1	...	Item50
Dm1	1	4	...	12
Dm1	...	...	...	...
Dm1	104	9	...	16
Dm2	1	3	...	19
...	...	...	...	...
Dm20	104	12	...	16

Vorverarbeitung: multivariat nach univariat

Quasi-SQL: For all shops for all items: Create view Univariate as Select shop, week, $item_i$ Where shop=“dm_j“ From Source;

- Multiples Lernen für alle univariaten Zeitreihen

Shop_Item	Week	Sale	Week	Sale
Dm1_Item1	1	4...	104	9
...				
Dm1_Item50	1	12...	104	16
...				
Dm20_Item50	1	14...	104	16

Vorverarbeitung II

- Problem: eine Zeitreihe ist nur 1 Beispiel!
- Das ist für das Lernen zu wenig.
- Lösung: Viele Vektoren aus einer Reihe gewinnen durch Fenster der Breite (Anzahl Zeitpunkte) w, bewege Fenster um m Zeitpunkte weiter.

Shop_Item_Window	Week	Sale	Week	Sale
Dm1_Item1_1	1	4...	5	7
Dm1_Item1_2	2	4...	6	8
...	...	...	...	...
Dm1_Item1_100	100	6...	104	9
...	...	...	...	...
Dm20_Item50_100	100	12...	104	16

SVM im Regressionfall

- Multiples Lernen: für jeden Laden und jeden Artikel, wende die SVM an. Die gelernte Regressionsfunktion wird zur Vorhersage genutzt.
- Asymmetrische Verlustfunktion :
  - Unterschätzung wird mit 20 multipliziert, d.h. 3 Verkäufe zu wenig vorhergesagt – 60 Verlust
  - Überschätzung zählt unverändert, d.h. 3 Verkäufe zu viel vorhergesagt – 3 Verlust

  (Diplomarbeit Stefan Rüping 1999)

Vergleich mit Exponential Smoothing

Horizont	SVM	exp. smoothing
1	56.764	52.40
2	57.044	59.04
3	57.855	65.62
4	58.670	71.21
8	60.286	88.44
13	59.475	102.24

Verlust, nicht normiert auf $[0, 1]$!

Was wissen wir jetzt?

- Anwendung der SVM für die Medikamentenverordnung
- Idee der Regressions-SVM
- Anwendung der SVM für die Verkaufsvorhersage
  - Umwandlung multivariater Zeitreihen in mehrere univariate
  - Gewinnung vieler Vektoren durch gleitende Fenster
  - Asymmetrische Verlustfunktion

13.9 Textkategorisierung

World Wide Web

- Seit 1993 wächst die Anzahl der Dokumente – 12,9 Milliarden Seiten (geschätzt für 2005)
- Ständig wechselnder Inhalt ohne Kontrolle, Pflege
  - Neue URLs
  - Neue Inhalte
  - URLs verschwinden
  - Inhalte werden verschoben oder gelöscht
- Verweisstruktur der Seiten untereinander
- Verschiedene Sprachen
- Unstrukturierte Daten

Aufgaben

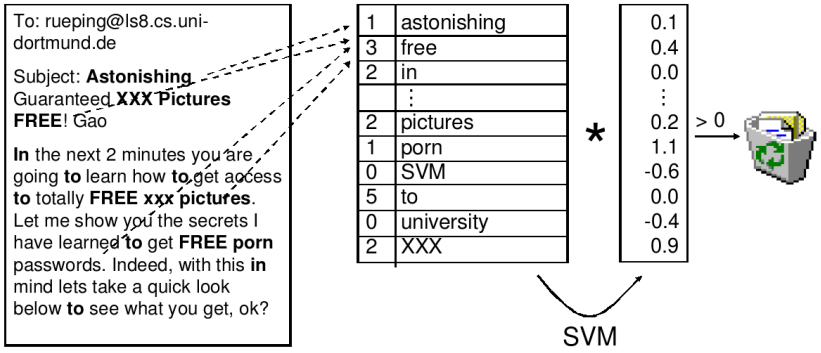
- Indexierung möglichst vieler Seiten (Google)
- Suche nach Dokumenten, ranking der Ergebnisse z.B. nach Häufigkeit der Verweise auf das Dokument (PageLink – Google)
- Kategorisierung (Klassifikation) der Seiten manuell (Yahoo), automatisch
- Strukturierung von Dokumentkollektionen (Clustering)
- Personalisierung:
  - Navigation durch das Web an Benutzer anpassen
  - Ranking der Suchergebnisse an Benutzer anpassen
- Extraktion von Fakten aus Texten

13.9.1 Information Retrieval

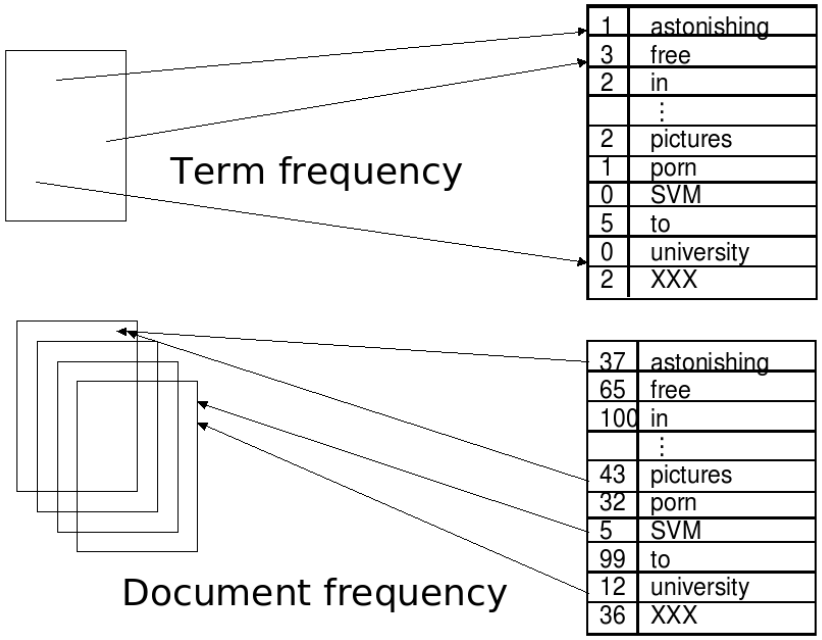
Information Retrieval

- Ein Dokument besteht aus einer Menge von Termen (Wörtern)
  - Bag of words: Vektor, dessen Komponenten die Häufigkeit eines Wortes im Dokument angeben.
- Für alle Dokumente gibt es eine Termliste mit Verweis auf die Dokumente.
  - Anzahl der Dokumente, in denen das Wort vorkommt.

Beispiel zur Klassifikation



Texte als Daten



TFIDF

- Term Frequenz: wie häufig kommt ein Wort w_i in einem Dokument d vor?

$$TF(w_i, d)$$

- Dokumentenfrequenz: in wie vielen Dokumenten einer Kollektion D kommt ein Wort w_i vor?

$$DF(w_i)$$

- Inverse Dokumentenfrequenz:

$$IDF(D, w_i) = \log \frac{|D|}{DF(w_i)}$$

- Bewährte Repräsentation:

$$TFIDF(w_i, D) = \frac{TF(w_i, d)IDF(w_i, D)}{\sqrt{\sum_j [TF(w_j, d)IDF(w_j, D)]^2}}$$

13.9.2 Textklassifikation

Textklassifikation

- Thorsten Joachims "The Maximum-Margin Approach to Learning Text Classifiers Kluwer", 2001
- Modell der Textklassifikation TCat
- Verbindung zur SVM-Theorie

→ theoretisch begründete Performanzabschätzung

Eigenschaften der Textklassifikation 1

- Hochdimensionaler Merkmalsraum
  - Reuters Datensatz mit 9603 Dokumenten: verschiedene Wörter

$$V = 27658$$

  - Heapes Gesetz: Anzahl aller Wörter

$$(s)V = ks^{\beta}$$

  - Beispiel:
    - Konkatenieren von 10 000 Dokumenten mit je 50 Wörtern zu einem,
    - $k = 15$ und $\beta = 0,5$
    - ergibt $V = 35000 \rightarrow$ stimmt!

Eigenschaften der Textklassifikation 2

- Heterogener Wortgebrauch
  - Dokumente der selben Klasse haben manchmal nur Stoppwörter gemeinsam!
  - Es gibt keine relevanten Terme, die in allen positiven Beispielen vorkommen.
  - Familienähnlichkeit (Wittgenstein): A und B haben ähnliche Nasen, B und C haben ähnliche Ohren und Stirn, A und C haben ähnliche Augen.

Eigenschaften der Textklassifikation 3

- Redundanz der Merkmale
  - Ein Dokument enthält mehrere die Klasse anzeigende Wörter.
  - Experiment:
    - Ranking der Wörter nach ihrer Korrelation mit der Klasse.
    - Trainieren von Naive Bayes für Merkmale von Rang

1 - 200	(90% precision/recall)
201 - 500	(75%)
601 - 1000	(63%)
1001- 2000	(59%)
2001- 4000	(57%)
4001- 9947	(51%) – zufällige Klassifikation (22%)

Eigenschaften der Textklassifikation 4

- Dünn besetzte Vektoren
- Reuters Dokumente durchschnittlich 152 Wörter lang
  - mit 74 verschiedenen Wörtern
  - also meist bei etwa 78 Wörtern 0
- Euklidsche Länge der Vektoren klein!

Eigenschaften der Textklassifikation 5

- Zipfs Gesetz: Verteilung von Wörtern in Dokumentkollektionen ist ziemlich stabil.
  - Ranking der Wörter nach Häufigkeit (r)
  - Häufigkeit des häufigsten Wortes (max)
  - $\frac{1}{r}max$ häufig kommt ein Wort des Rangs r vor.
- Generalisierte Verteilung von Häufigkeit nach Rang (Mandelbrot): v ist Größe der Dokumentkollektion in Wortvorkommen

$$\frac{v}{(k+r)^{\phi}}$$

Plausibilität guter Textklassifikation durch SVM

- R sei Radius des Balles, der die Daten enthält. Dokumente werden auf einheitliche Länge normiert, so dass $R = 1$
- Margin sei δ, so dass großes δ kleinem $\frac{R^2}{\delta^2}$ entspricht.

Reuters	$\frac{R^2}{\delta^2}$	$\sum_{i=1}^{n} \xi$
Earn	1143	0
acquisition	1848	0
money-fx	1489	27
grain	585	0
crude	810	4

Reuters	$\frac{R^2}{\delta^2}$	$\sum_{i=1}^{n} \xi$
trade	869	9
interest	2082	33
ship	458	0
wheat	405	2
corn	378	0

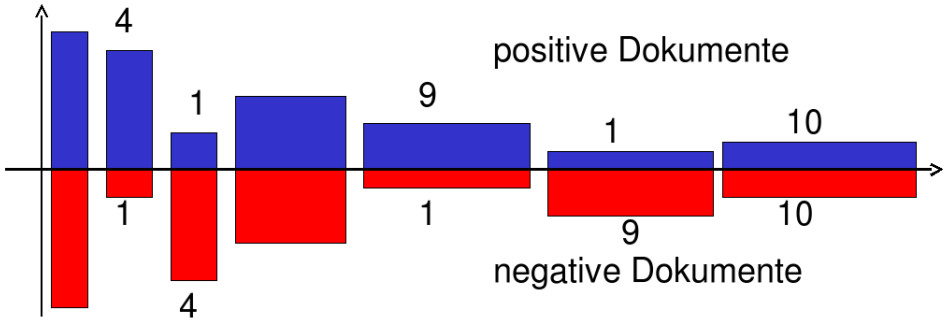
TCat Modell – Prototyp

- Hochdimensionaler Raum: $V = 11100$ Wörter im Lexikon
- Dünn besetzt: Jedes Dokument hat nur 50 Wörter, also mindestens 11050 Nullen
- Redundanz: Es gibt 4 mittelhäufige und 9 seltene Wörter, die die Klasse anzeigen
- Verteilung der Worthäufigkeit nach Zipf/Mandelbrot.
- Linear separierbar mit $\beta_0 = 0$, $\sum_{i=1}^{11100} \beta_i x_i$

$$\beta_i = \begin{cases} 0,23 & \text{für mittelhäufige Wörter in } POS, \\ -0,23 & \text{für mittelhäufige Wörter in } NEG, \\ 0,04 & \text{für seltene Wörter in } POS, \\ -0,04 & \text{für seltene Wörter in } NEG, \\ 0 & \text{sonst} \end{cases}$$

TCat im Bild

- 20 aus 100 Stoppwörtern, 5 aus 600 mittelhäufigen und 10 aus seltenen Wörtern kommen in POS- und NEG-Dokumenten vor; 4 aus 200 mittelhäufigen Wörtern in POS, 1 in NEG, 9 aus 3000 seltenen Wörtern in POS, 1 in NEG (Es müssen nicht immer die selben Wörter sein!)



TCat

"The TCat concept

$$TCat\left([p_1 : n_1 : f_1], \ldots, [p_s : n_s : f_s]\right)$$

describes a binary classification task with s sets of disjoint features. The i-th set includes f_i features. Each positive example contains p_i occurences of features from the respective set and each negative example contains n_i occurrences. The same feature can occur multiple times in one document. " (Joachims 2002)

TCat zum Bild

7 disjunkte Wortmengen; bei einem zur Klasse gehörigen Dokument kommt 20 mal eines der 100 Wörter der ersten Wortmenge vor, 4 mal eines der 200 Wörter der zweiten Wortmenge, ...; bei einem nicht zur Klasse gehörigen Dokument gibt es 20 Auftreten von Wörtern aus der ersten Wortmenge,... Es sind also nicht bestimmte Wörter, die die Klassenzugehörigkeit anzeigen!

$$TCat(\underbrace{[20:20:100]}_{\text{sehr häufig}}$$
$$\underbrace{[4:1:200][1:4:200][5:5:600]}_{\text{mittel häufig}}$$
$$\underbrace{[9:1:3000][1:9:3000][10:10:4000]}_{selten})$$

Lernbarkeit von TCat durch SVM

(Joachims 2002) "Der erwartete Fehler einer SVM ist nach oben beschränkt durch:"

$$\frac{R^2}{n+1}\frac{a+2b+c}{ac-b^2}$$

$a = \sum_{i=1}^{s} \frac{p_i^2}{f_i}$

$b = \sum_{i=1}^{s} \frac{p_i^2 n_i}{f_i}$

$c = \sum_{i=1}^{s} \frac{n_i^2}{f_i}$

$R^2 = \sum_{r=1}^{d} \left(\frac{v}{(r+k)^{\phi}}\right)^2$

Es gibt l Wörter, s Merkmalsmengen, für einige i: $p_i \neq n_i$ und die Termhäufigkeit befolgt Zipfs Gesetz. Wähle d so, dass:

$$\sum_{r=1}^{d} \frac{v}{(r+k)^{\phi}} = l$$

Was wissen Sie jetzt?

- Die automatische Klassifikation von Texten ist durch das WWW besonders wichtig geworden.
- Texte können als Wortvektoren mit TFIDF dargestellt werden. Die Formel für TFIDF können Sie auch!
- Textkollektionen haben bzgl. der Klassifikation die Eigenschaften: hochdimensional, dünn besetzt, heterogen, redundant, Zipfs Gesetz.
- Sie sind mit breitem margin linear trennbar.
- Das TCat-Modell kann zur Beschränkung des erwarteten Fehlers eingesetzt werden. Die Definition von TCat kennen Sie mindestens, besser wäre noch die Fehlerschranke zu kennen.

13.9.3 Verwendung des Modells zur Textklassifikation für zeitgestempelte Daten

Verwendung des TCat Modells für zeitgestempelte Daten
Und jetzt wenden wir das Gelernte auf ein Gebiet fernab von Texten an!

Lokale Muster

- Lokale Muster beschreiben seltene Ereignisse.
- Gegeben ein Datensatz, für den ein globales Modell bestimmt wurde, weichen lokale Muster davon ab.
  - Lokale Muster beschreiben Daten mit einer internen Struktur, z.B. Redundanz, Heterogenität

Zeit-gestempelte Daten

- Zeit-gestempelte Daten können transformiert werden in:
  - Eine Menge von Ereignissen,
  - Zeitintervalle,
  - Zeitreihen.

Klassische Methoden

- Zeitreihenanalyse für Vorhersage, Trend und Zyklus Erkennung
- Indexing und clustering von Zeitreihen (time warping)
- Segmentierung (motif detection)
- Entdeckung von Episoden
  - frequent sets,
  - chain logic programs (grammars)
- Regression

Beispielrepräsentation

- Die Beispielrepräsentation X bestimmt die Anwendbarkeit der Methoden: welche Variablen, was sind Beispiele?
- Bedeutung der Repräsentation lange unterschätzt.
- Suche nach guter Repräsentation ist aufwändig.
- Transformieren der Rohdaten in die Repräsentation auch.

Einige Repräsentationen für zeitgestempelte Daten

- Schnappschuss: ignoriere Zeit, nimm nur den aktuellen Zustand. (So war es bei der Intensivmedizin-Anwendung.)
- Ereignisse mit Zeitintervallen: aggregiere Zeitpunkte zu Intervallen, wende frequent set mining an. (Das machen wir in dieser Vorlesung nicht.)
- Generierte Merkmale: hier: transformiere Zeitinformation in Häufigkeitsmerkmale!

Häufigkeitsmerkmale für Zeitaspekte

- Term frequency: wie oft änderte Attribut A seinen Wert a_i für ein Objekt c_j.

$$tf(a_i, c_j) = \| \{x \in timepoints | a_i \text{ of } c_j \text{ changed } \} \|$$

- Document frequency: in wie vielen Objekten c_j änderte Attribut A seinen Wert a_i.

$$df(a_i) = \| \{c_j \in C | a_i \text{ of } c_j \text{ changed } \} \|$$

- TF/IDF:

$$tfidf(a_i) = tf(a_i, c_j) \log \frac{\|C\|}{df(a_i)}$$

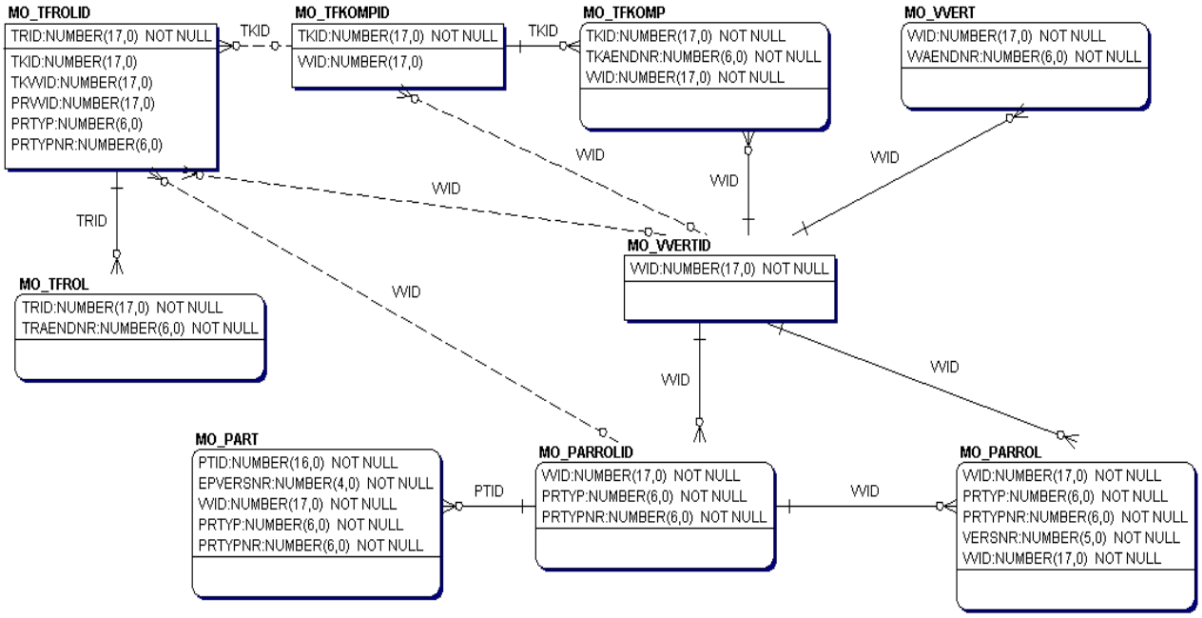
Fallstudie SwissLife

- Lokale Muster
  - Seltenes Ereignis der Kündigung
  - Lokales Muster weicht ab vom generellen Modell
  - Interne Struktur in lokalen Mustern
- Zeit-gestempelte Daten
  - Schnappschuss
  - Zeitintervall
  - Generierte Merkmale: $TFIDF$

Lokale Muster in Versicherungsdaten

- Nur 7.7% der Verträge enden vorzeitig (customer churn).
- Für einige Attribute weicht die likelihood in der churn-Klasse von der globalen ab.
- Interne Struktur:
  - Überlappung: häufige Mengen in churn Verträgen sind auch häufig in fortgesetzten Verträgen.
  - Redundanz: in jedem Vertrag gibt es mehrere Attribute, die auf Fortsetzung oder Kündigung hinweisen.
  - Heterogenität: Es gibt gekündigte Verträge, die nicht ein einziges Attribut gemeinsam haben.

Database



Contract Table

VVID	VVAENDNR	VVWIVON	VVWIBIS	VVAENDAT	VVAENDART	...
16423	1	1946	1998	1946	1000	
16423	2	1998	1998	1998	27	
16423	3	1998	1998	1998	4	
16423	4	1998	1998	1998	54	
16423	5	1998	1998	1998	4	
16423	6	1998	9999	1998	61	
5016	1	1997	1999	1997	33	
5016	2	1999	2001	1999	33	
5016	3	2001	2001	2001	33	
5016	4	2001	2001	2001	33	
5016	5	2001	2002	2001	81	
5016	6	2002	9999	2001	94	
...	...	...	...	...	...	...

Datensatz

- Tabellen enthalten Informationen über
  - 217586 Komponenten and
  - 163745 Kunden
- Attribute:
  - 14 Attributes ausgewählt
  - Eines der Attribute gibt den Grund an für einen Wechsel. Es gibt 121 Gründe. Daraus werden 121 Boolean Attribute.
  - 134 Attribute mit $TFIDF$ Werten.

Erste Experimente

- Bei SwissLife wurde die Abweichung der Wahrscheinlichkeit bestimmter Attributwerte in gekündigten und fortgesetzten Verträgen festgestellt anhand der Schnappschussrepräsentation → keine operationale Vorhersage.

Calculating Term Frequency

VVID	...	VVSTACD	VVPRFIN	VVPRZA	VVINKZWEI	VVBEG	VVEND	VVINKPRL	...
16423		4	1	2	2	1946	1998	295.29	
16423		4	1	2	2	1946	1998	295.29	
16423		4	5	2	0	1946	2028	0	
16423		5	3	2	0	1946	2028	0	
16423		4	1	2	2	1946	1998	295.29	
16423		5	3	2	0	1946	1998	0	

3	VVSTACD
4	VVPRFIN
0	VVPRZA
3	VVINKZWEI
0	VVBEG
2	VVEND
3	VVINKPRL

Experimente mit der TFIDF Repräsentation

- Vergleich der originalen Repräsentation und der TFIDF
  - 10fold cross validation
    - Apriori mit Konklusion 'churn'
    - Entscheidungsbaumlerner J4.8
    - Naive Bayes
    - mySVM mit linearem Kern
  - F-measure balanciert precision und recall gleich.

Alle Lernalgorithmen werden besser mit der $TFIDF$- Repräsentation.

Resultate (F-measure)

Lerner	**TF/IDF repr.**	**Original repr.**
Apriori	63.35	30.24
J4.8	99.22	81.21
Naive Bayes	51.8	45.41
mySVM	97.95	16.06

Erklärung?

- TF/IDF stammt aus Lernen über Texten.
- Dazu gibt es eine Theorie – TCat.
- Können wir die auch hier einsetzen??

Datenbeschreibung im TCat Modell

$$TCat(\underbrace{[2:0:2],[1:4:3],}_{\text{high frequency}}$$
$$\underbrace{[3:1:3],[0:1:4],}_{\text{medium frequency}}$$
$$\underbrace{[1:0:19],[0:1:64],}_{\text{low frequency}})$$
$$\underbrace{[1:1:39]}_{\text{rest}}))$$

[1 : 4 : 3] : Aus der Menge von 3 Merkmale finden wir ein Auftreten in positiven und 4 in negativen Beispielen.

Learnability of TCat

Error bound (Joachims 2002)

$$\frac{R^2}{n+1}\frac{a+2b+c}{ac-b^2}$$

$$\begin{aligned} a &= \textstyle\sum_{i=1}^{s} \frac{p_i^2}{f_i} && = 5.41 \\ b &= \textstyle\sum_{i=1}^{s} \frac{p_i^2 n_i}{f_i} && = 2.326 \\ c &= \textstyle\sum_{i=1}^{s} \frac{n_i^2}{f_i} && = 5.952 \\ R^2 &= \textstyle\sum_{r=1}^{d} \left(\frac{c}{(r+k)^{\phi}}\right)^2 && \leq 37 \end{aligned}$$

Nach 1000 Beispielen erwarteter Fehler $\leq 2.2\%$ Tatsächlicher Fehler 2.05%

Experimente zu lokalen Mustern

- Durch TCat-Konzepte Daten künstlich generieren.
- Lokale Muster als seltene Ereignisse mit interner Struktur.

Lokale Muster: Verzerrte Verteilung

- 10 000 Beispiele mit 100 Attributen
- SVM runs mit 10 fold cross validation

Repr.	$Targetconcept:$	Verzerrung:
TF/IDF	1. change of a particular attribute	50%, 25%,
Boolean	2. frequency of changes	12.5%, 6.25%

Lokale Muster: Strukturen

- 10 000 Beispiele mit 100 Attributen
- 20 Attribute wechseln pro Beispiel (dünn besetzt)
- Variieren:
  - Heterogenität: $\frac{f_i}{p_i}$ Beispiele der selben Klasse haben kein gemeinsames Attribut $4, 5, 10, 20$
  - Redundanz: $\frac{p_i}{f_i}$ oder $\frac{n_i}{f_i}$ für die Redundanz innerhalb einer Klasse $0.5, 0.2, 0.1$
  - Überlappung: einige Attribute sind häufig in beiden Klassen $0.25, 0.66$

Resultate

- Für alle Kombinationen ohne Überlappung sind die Lernergebnisse 100% in Boolean und im TF/IDF- Format.
- Mehr Überlappung verschlechtert das Lernen bei Boolean auf 68.57% F-measure.
- Für alle Kombinationen (auch mit großer Überlappung) erreicht das Lernen mit TF/IDF Daten 100% precision und recall.

Navigation im Raum der Beispiele

- Zunehmende Größe des Datensatzes zeitgestempelter Daten: Schnappschuss ¡ Intervalle ¡ Boolean ¡ TF/IDF
- TF/IDF ist günstig für lokale Muster, wenn diese Redundanz, Heterogenität als Eigenschaft aufweisen.
- Berechnung des TCat Modells für gegebene Daten implementiert → Fehlerschranke angebbar.

Was wissen Sie jetzt?

- Lokale Muster haben manchmal die typische TCat-Struktur.
- Sie haben gesehen, wie manche zeitgestempelte Datenbanken in TCat-Modelle transformiert werden können.
- Die Lernbarkeit mit linearer SVM der so transformierten Daten können Sie ausrechnen.

14 SVMstruct

14.1 Überblick Lernaufgaben

Jenseits des Bag of Words

- Bisher haben wir Texte als Anzahl und Häufigkeit von Wörtern repräsentiert.
- Damit haben wir die Struktur der Sprache ignoriert.
  - Grammatik
  - Koreferenz
  - Eigennamen
  - Semantische Relationen
- Es gibt eine ganze Reihe von Ansätzen des maschinellen Lernens, um (sprachliche) Strukturen zu behandeln.
- Wir besprechen hier nur die SVM bezogenen Ansätze.

Lernaufgabe Named Entity Recognition

- Wortfolgen, die sich auf ein individuelles Objekt beziehen, werden *Named Entities* (NE) genannt.
- Eigennamen, Ortsnamen, Firmennamen sind z.B. NEs.
- Gegeben Beispiele von Sätzen, in denen NEs annotiert sind, lerne die Entscheidungsfunktion, die für jedes Wort angibt, ob es zu einer bestimmten NE gehört, oder nicht.
- Beispiel:

Johann	Sebastian	Bach	publiziert	im	Henle	Verlag	München.
Per	Per	Per	0	0	Org	Org	Place

Anwendungen

Wenn wir in Dokumenten die NEs automatisch annotieren,

- können wir sie im Text markieren, so dass die Benutzer schneller interessante Stellen auffinden;
- können wir alle Sätze zu einer Person, Firma, einem Ort herausschreiben und so eine Zusammenfassung für einen Text erstellen;
- eine weitere Lernaufgabe aufsetzen: *Relationen* zwischen NEs lernen, z.B. Fusion von Firmen $fusion(Org_1, Org_2)$, Mitglied im Aufsichtsrat $aufsicht(Org, Per)$.

Letztlich erstellen wir eine Datenbank aus einer Dokumentensammlung. Auf diese Datenbank wenden wir dann unsere Lernverfahren wie gehabt an.

Part of Speech Tagging

- Unter *Part-of-speech Tagging* versteht man die Zuordnung von Wörtern eines Textes zu Wortarten (engl.: part of speech).
- Beispiel:

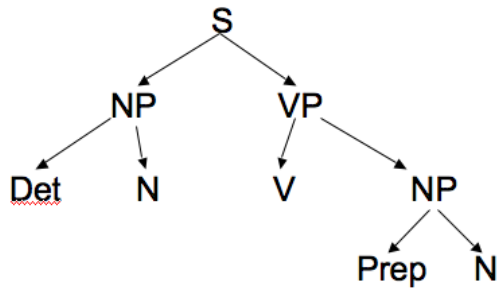
Die	Noten	erscheinen	bei	Henle.
Det	N	V	Prep	N

Shallow Parsing

- Syntaxregeln produzieren einen Syntaxbaum für einen Satz, dessen Wurzel das Startsymbol S ist und die Blätter sind die Wortarten (präterminalen Knoten), denen dann die Wörter zugeordnet sind. *Shallow Parsing* erstellt Syntaxbäume für Sätze.

x: Die Noten erscheinen bei Henle.

y:



- Beispiel: $S \rightarrow NP, VP$ $VP \rightarrow V, NP$ $NP \rightarrow Det, N$ $NP \rightarrow Prep, N$

Lernaufgaben Part of Speech Tagging, Shallow Parsing

- *Part of Speech Tagging*: Gegeben eine Menge von Sätzen, bei denen zu jedem Wort die Wortart angegeben ist, lerne eine Entscheidungsfunktion, die bei beliebigen Sätzen jedem Wort eine Wortart zuordnet.
- *Shallow Parsing Learning*: Gegeben eine Menge von Sätzen, eine Menge von Syntaxregeln und die Anzahl von Regelanwendungen für jeden Satz, lerne eine Entscheidungsfunktion, die beliebigen Sätzen die Anzahl von Regelanwendungen zuordnet (discriminant model).
- *Zur Abgrenzung*: Gegeben eine Menge von syntaktisch korrekten Sätzen (positive Beispiele) und eine Menge von synaktisch falschen Sätzen (negative Sätze), bei denen jeweils die Wortarten annotiert sind, lerne Syntaxregeln, die gerade die syntaktisch korrekten Sätze produzieren (generative model).

14.2 Einführung SVMstruct

SVMstruct

Definition 14.1 (Strukturelle Modelle)**.** *Sei X die Menge der Beispiele. Ist die Ausgabe-Variable Y nicht nur eine Zahl aus irgendeinem Zahlenraum, sondern eine Struktur (z.B. eine Folge, ein Baum, ein Graph), so heißt das Modell*

$$f : X \to Y$$

strukturelles Modell.

- Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, Yasemin Altun "Large Margin Methods for Structured and Interdependent Output Variables", J. of Machine Learning Research, Vol. 6, p. 1453 – 1484, 2005
- Thorsten Joachims "Training Linear SVMs in Linear Time", Proc. KDD 2006

Lernaufgabe der strukturellen SVM

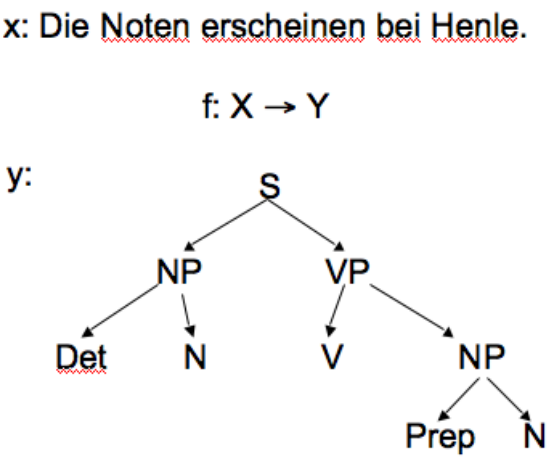
Gegeben eine Menge von Paaren $\mathcal{T} = \{(\vec{x_1}, \vec{y_1}), ..., (\vec{x_n}, \vec{y_n})$, wobei

- die Beobachtungen $\vec{x} \in X$ durch Zufallsvariablen $X_1, ..., X_p$ beschrieben sind und
- die Ausgabe (label) $\vec{y} \in Y$ eine Konfiguration von Zuständen $Y_1, ..., Y_q$ ist, die von einander abhängig sein können.

Finde eine Funktion $F : X \times Y \to \mathcal{R}$, so dass für eine (neue) Beobachtung $\vec{x}$ die richtige Vorhersage durch Maximierung über der Ausgabe $\vec{y}$ getroffen wird:

$$f(\vec{x}; \vec{\beta}) = argmax_{\vec{y} \in Y} F(\vec{x}, \vec{y}; \vec{\beta}) \tag{39}$$

Merkmalsabbildung



$$\Psi(\vec{x}, \vec{y}) = \begin{pmatrix} 1 \\ 1 \\ 1 \\ 0 \\ ... \\ 0 \\ 1 \end{pmatrix} \begin{matrix} S \to NP, VP \\ NP \to Det, N \\ NP \to Prep, N \\ NP \to Adj, N \\ ... \\ Prep \to in \\ Prep \to bei \end{matrix}$$

Annahme:
$F(\vec{x}, \vec{y}; \vec{\beta}) = \langle \vec{\beta}, \Psi(\vec{x}, \vec{y}) \rangle$ Wir lernen also über input-/output-Kombinationen die Ranking-Funktion $F : X \times Y \to \mathcal{R}$.

14.3 Primales Problem

Primales Problem – $\vec{\beta}$

- Wie bei der SVM mit einfacher Ausgabevariable y, muss $\vec{\beta}$ bestimmt werden:

$$min_{\vec{\beta}} \frac{1}{2}\|\vec{\beta}\|^2 \tag{40}$$

- Aber was sind hier die Nebenbedingungen? Grundidee:
  - Alle $\langle \vec{\beta}, \Psi(\vec{x}, \vec{y_i})\rangle$ erhalten ein Gewicht, so dass eine Rangfolge (ranking) entsteht.
  - Der Abstand zwischen der besten und der zweitbesten Lösung soll maximal sein!
- Obwohl wir auf diese Weise $n \mid Y \mid -n$ Nebenbedingungen erhalten, interessieren uns eigentlich immer nur die erst- und zweitbesten $\vec{y}$ und $\vec{y_i}$.

Primales Problem – Nebenbedingungen

- Für jedes der N Beispiele $\vec{x_i}$ mit jedem der $\mid Y \mid$ möglichen $\vec{y_i}$ müssen wir feststellen, wie groß der Abstand zu allen anderen $\Psi(\vec{x_i}, \vec{y})$ ist. Diesen Abstand notieren wir:

$$\delta\Psi_i(\vec{y}) \equiv \Psi(\vec{x_i}, \vec{y_i}) - \Psi(\vec{x_i}, \vec{y})$$

- Wir erhalten $N \mid Y \mid -N$ lineare Nebenbedingungen für das primale Problem:

$$\forall i, \forall \vec{y} \in Y \backslash \vec{y_i} : \langle \vec{\beta}, \delta\Psi_i(\vec{y}\rangle \geq 1 \tag{41}$$

Primales Problem SVMstruct

Bei linear trennbaren Daten ist das primale Optimierungsproblem der SVMstruct:

Definition 14.2 (Primales Problem SVM)**.**

$$\begin{aligned} SVM_0 : \quad & min_{\vec{\beta}} \tfrac{1}{2}\|\vec{\beta}\|^2 && (40)\\ & \textit{unter den Bedingungen} \\ & \forall i, \forall \vec{y} \in Y \backslash \vec{y_i} : \langle \vec{\beta}, \delta\Psi_i(\vec{y}\rangle \geq 1 && (41) \end{aligned}$$

SVMstruct mit Ausnahmen – slack rescaling

Ein Strafterm C für alle Beispiele $\vec{x}$, bei denen die Nebenbedingungen verletzt sind, und die Relaxierung durch ξ führt zu dem Minimierungsproblem:

Definition 14.3 (Slack Rescaling SVM)**.**

$$\begin{aligned} SVM_1 : \quad & min_{\vec{\beta},\xi} \tfrac{1}{2}\|\vec{\beta}\|^2 + \tfrac{C}{N}\sum_{i=1}^{N}\xi_i && (42)\\ & \textit{unter den Bedingungen} \\ & \forall i, \forall \vec{y} \in Y \backslash \vec{y_i} : \langle \vec{\beta}, \delta\Psi_i(\vec{y}\rangle \geq 1 - \xi_i, \xi_i \geq 0 && (43) \end{aligned}$$

C ist linear in den ξ_i.

SVMstruct mit Ausnahmen – margin rescaling

Verletzungen der Nebenbedingungen können auch durch einen quadratischen Term bestraft werden.

Definition 14.4 (Margin rescaling SVM)**.**

$$\begin{aligned} SVM_2 : \quad & min_{\vec{\beta},\xi} \tfrac{1}{2}\|\vec{\beta}\|^2 + \tfrac{C}{2N}\sum_{i=1}^{N}\xi_i^2 && (44)\\ & \textit{unter den Bedingungen} \\ & \forall i, \forall \vec{y} \in Y \backslash \vec{y_i} : \langle \vec{\beta}, \delta\Psi_i(\vec{y}\rangle \geq 1 - \xi_i, \xi_i \geq 0 && (45) \end{aligned}$$

Empirisches Risiko und Slack Rescaling

Auch bei strukturellen Modellen geht es darum, den Fehler zu minimieren. Der erwartete Fehler bei irgend einer Verlustfunktion Δ ist für eine Menge von Beispielen $\mathcal{T}$

$$R_{\mathcal{T}}(f) = \frac{1}{2}\sum_{i=1}^{N} \Delta(\vec{y_i}, f(\vec{x_i})) \tag{46}$$

Um die Verletzung der Nebenbedingung für ein $\vec{y} \neq \vec{y_i}$ bei großem Verlust $\Delta(\vec{y_i}, \vec{y})$ stärker zu bestrafen als bei geringerem, passen wir die Nebenbedingungen an:

$$\begin{aligned} SVM_1: \quad & min_{\vec{\beta},\xi}\ \tfrac{1}{2}\|\vec{\beta}\|^2 + \tfrac{C}{N}\textstyle\sum_{i=1}^{N}\xi_i \\ & \forall i, \forall \vec{y} \in Y \backslash \vec{y_i} : \langle \vec{\beta}, \delta\Psi_i(\vec{y}\rangle \geq 1 - \tfrac{\xi_i}{\Delta(\vec{y_i},\vec{y})} \end{aligned} \tag{47}$$

$$\begin{aligned} SVM_2: \quad & min_{\vec{\beta},\xi}\ \tfrac{1}{2}\|\vec{\beta}\|^2 + \tfrac{C}{2N}\textstyle\sum_{i=1}^{N}\xi_i^2 \\ & \forall i, \forall \vec{y} \in Y \backslash \vec{y_i} : \langle \vec{\beta}, \delta\Psi_i(\vec{y}\rangle \geq 1 - \tfrac{\xi_i}{\sqrt{\Delta(\vec{y_i},\vec{y})}} \end{aligned} \tag{48}$$

Obere Schranke des empirischen Fehlers

Satz 14.1 (Obere Schranke des empirischen Fehlers)**.** *Seien $\xi_i^*(\vec{\beta})$ die optimalen Schlupfvariablen für ein gegebenes $\vec{\beta}$, dann ist*

$$R_{\mathcal{T}}(\vec{\beta}) \leq \frac{1}{N}\sum_{i=1}^{N}\xi_i^*$$

Beweis Wir wissen: $\xi_i^* = max\{0, max_{\vec{y}\neq\vec{y_i}}\{\Delta(\vec{y_i},\vec{y})(1 - \langle\vec{\beta}, \delta\Psi_i(\vec{y})\rangle)\}\}$.

Alles gut: $f(\vec{x_i};\vec{\beta}) = \vec{y_i}$, also $\Delta(\vec{y_i}, f(\vec{x_i};\vec{\beta})) = 0 \leq \xi_i^*$

Fehler: $\vec{y^*} \equiv f(\vec{x_i};\vec{\beta}) \neq \vec{y_i}$, also $\langle\vec{\beta}, \delta\Psi_i(\vec{y^*})\rangle \leq 0$, also $\frac{\xi_i^*}{\Delta(\vec{y_i},\vec{y})} \geq 1$. Dies entspricht $\Delta(\vec{y_i},\vec{y}) \leq \xi_i^*$.

Da die Schranke für jedes Beispiel gilt, gilt sie auch für den Durchschnitt.

14.4 Duales Problem

Hinführung zum dualen Problem

- Wir wollen auch hier das duale Problem formulieren.
- Bisher war es:
$$L_D(\vec{\alpha}) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \langle\vec{x}_i, \vec{x}_j\rangle$$
- Jetzt sind $\vec{y_i}, \vec{y_j}$ nicht mehr einfach Werte, sondern Strukturen.
- Für jedes der N Beispiele $\vec{x_i}$ mit jedem der $\mid Y \mid$ möglichen $\vec{y_i}$ müssen wir feststellen, wie groß der Abstand zu allen anderen $\Psi(\vec{x_i},\vec{y})$ ist.
- Wir haben also nicht mehr ein α je Beispiel in X, sondern ein α für jedes Paar in $X \times Y$.
- Erst sehen wir uns den Raum an, in dem optimiert wird, dann die α.

Raum, in dem SVMstruct optimiert

- Bei der klassischen SVM haben wir beim dualen Problem für $i, j = 1, ..., N$ das Skalarprodukt $\langle \vec{x_i}, \vec{x_j} \rangle$ gerechnet.
- Jetzt müssen wir für $i, j = 1, ..., N$ rechnen $J_{(i\vec{y})(j\vec{y'})} \equiv \langle \delta\Psi_i(\vec{y}), \delta\Psi_j(\vec{y'}) \rangle$.

	$\vec{x_1}$	...	$\vec{x_j}$	...	$\vec{x_N}$
$\vec{x_1}$	$-$	...	...	...	...
...	...	...	...	...	...
$\vec{x_i}$	...	...	$\langle \delta\Psi_i(\vec{y}), \delta\Psi_j(\vec{y'}) \rangle$	...	...
...	...	...	...	...	...
$\vec{x_N}$	...	...	...	...	$-$

- Dabei ist $\langle \delta\Psi_j(\vec{y}), \delta\Psi_i(\vec{y'}) \rangle$ wieder eine Matrix!

Matrix J

- In der $N \times N$ Matrix sind die Einträge $\langle \delta\Psi_i(\vec{y}), \delta\Psi_j(\vec{y'}) \rangle$.
- Wir erinnern uns: $\delta\Psi_i(\vec{y}) \equiv \Psi(\vec{x_i}, \vec{y_i}) - \Psi(\vec{x_i}, \vec{y})$
- Statt einer einfachen Matrix haben wir einen Tensor, d.h. der Eintrag in die $N \times N$ Matrix ist eine $\mid Y \mid \times \mid Y \mid$ Matrix **J**:

	$\vec{y_1}$	...	$y_{\vec{\mid Y \mid}}$
$\vec{y_1}$	$-$	...	$\Psi(\vec{x}, \vec{y_1}) - \Psi(\vec{x}, y_{\vec{\mid Y \mid}})$
...	...	...	...
$y_{\vec{\mid Y \mid}}$	$\Psi(\vec{x}, y_{\vec{\mid Y \mid}}) - \Psi(\vec{x}, \vec{y_1})$	...	$-$

- **J** ist eine Kernfunktion über $X \times Y$: $J_{(i\vec{y})(j\vec{y'})} = \langle \delta\Psi_i(\vec{y}), \delta\Psi_j(\vec{y'}) \rangle$

$\alpha_{i\vec{y}}$ und optimales β

- Statt α_i für $\vec{x_i}$, haben wir α_{ij} mit $j = 1, ..., \mid Y \mid$

$$\begin{pmatrix} \alpha_{i1} \\ ... \\ \alpha_{im} \end{pmatrix}$$

- Das optimale $\vec{\beta}$ ist

$$\begin{aligned} \hat{\vec{\beta}} &= \sum_{i=1}^{N} \sum_{\vec{y} \neq \vec{y_i}}^{|Y|} \alpha_{(i\vec{y})} (\Psi(\vec{x_i}, \vec{y_i}) - \Psi(\vec{x_i}, \vec{y})) \\ &= \sum_{i=1}^{N} \sum_{\vec{y} \neq \vec{y_i}}^{|Y|} \alpha_{(i\vec{y})} \delta\Psi_i(\vec{y}) \end{aligned} \tag{49}$$

Duales Problem der SVMstruct

- SVMstruct bei linear separierbaren Beispielen:

$$L_D(\alpha) = -\frac{1}{2} \sum_{i,\vec{y} \neq \vec{y_i}}^{N} \sum_{j,\vec{y'} \neq \vec{y_i}}^{N} \alpha_{i\vec{y}} \alpha_{j,\vec{y'}} J_{(i\vec{y})(j\vec{y'})} + \sum_{i,\vec{y} \neq \vec{y_i}}^{N} \alpha_{i\vec{y}} \tag{50}$$

- Für die Slack Rescaling SVM_1 mit Ausnahmen muss zusätzlich gelten:

$$\sum_{\vec{y} \neq \vec{y_i}}^{N} \alpha_{i\vec{y}} \leq \frac{C}{N}, \forall i = 1, ..., N$$

- Für die Margin Rescaling SVM_2 mit Ausnahmen wird $J_{(i\vec{y})(j\vec{y'})}$ unter Verwendung der Indikatorfunktion $I(a, b) = 1$ falls $a = b$, sonst 0 zu:

$$\langle \delta\Psi_i(\vec{y}), \delta\Psi_j(\vec{y'}) \rangle + I(i, j)\frac{N}{C}$$

- Immer soll $\vec{\alpha}$ maximiert werden.

14.5 Optimierung der SVMstruct

Die SVMstruct stellt ein schwieriges Optimierungsproblem!

- Bei $N \mid Y \mid -N$ Nebenbedingungen und vermutlich sehr großem $\mid Y \mid$ ist normale Optimierung durch quadratische Programmierung nicht möglich.
- Es sollen nun deutlich weniger Nebenbedingungen wirklich bearbeitet werden.
- Beobachtung: Es gibt immer eine **Teilmenge** von Nebenbedingungen, so dass die damit errechnete Lösung auch **alle** Nebenbedingungen erfüllt mit einer Ungenauigkeit von nur ϵ.

SVMstruct: Algorithmus zum Optimieren – Idee

- Für jedes Beispiel $\vec{x_i}$ gibt es einen working set S_i, in dem die verletzten Nebenbedingungen gespeichert sind. Zunächst sind S_i leer, das Problem unbeschränkt.
- Für jedes Beispiel $\vec{x_i}$ wird die am schlimmsten verletzte Nebenbedingung bzgl. $\vec{y_i^*}$ festgestellt und S_i hinzugefügt. Das Problem wird zunehmend stärker beschränkt.
- Optimiere α
  - bezüglich aller working sets gemeinsam oder
  - nur für ein S_i, wobei die $\alpha_{j\vec{y}}$ mit $j \neq i$ eingefroren werden.
- Wenn kein S_i mehr verändert wurde, STOP.

SVMstruct: Algorithmus zum Optimieren

1. Input: $\mathcal{T} = \{(\vec{x_1}, \vec{y_1}), ..., (\vec{x_N}, \vec{y_N})\}, C, \epsilon$
2. $S_i := \{\}$ für alle $i = 1, ..., N$
3. Solange ein S_i sich in der Iteration ändert:
4. **for** $i = 1, ..., N$ **do**
5. Kosten: $H(\vec{y}) \begin{cases} 1 - \langle \delta\Psi_i(\vec{y}), \vec{\beta} \rangle & SVM_0 \\ (1 - \langle \delta\Psi_i(\vec{y}), \vec{\beta} \rangle)\Delta(\vec{y_i}, \vec{y}) & SVM_1 \ (s.47) \\ (1 - \langle \delta\Psi_i(\vec{y}), \vec{\beta} \rangle)\Delta(\vec{y_i}, \vec{y}) & SVM_2 \ (s.48) \end{cases}$ wobei $\vec{\beta} \equiv \sum_j \sum_{\vec{y'} \in S_j} \alpha_{j\vec{y'}} \delta\Psi_j(\vec{y'})$
6. $\vec{y^*} := argmax_{\vec{y} \in Y} H(\vec{y})$ – schwieriger Schritt!
7. $\xi_i := max\{0, max_{\vec{y} \in S_i} H(\vec{y})\}$
8. **if** $H(\vec{y^*}) > \xi_i + \epsilon$ **then**
9. $S_i := S_i \cup \{\vec{y^*}\}$
10. α_S :=optimiere duales Problem für $S = \cup S_i$

14.6 Anwendungen

Shallow Parsing Learning mit SVMstruct

- Probabilistische kontextfreie Grammatik: Regeln $n_l[C_i \rightarrow C_j, C_k], \beta_l$. Dabei gibt β_l die logarithmierte Wahrscheinlichkeit dafür an, dass ein Knoten C_i mit Regel n_l expandiert wird.
- Lernaufgabe: Gegeben Paare $(\vec{x}, \vec{y})$, wobei $\vec{x} = x_1, ..., x_p$ ein Satz (Kette von Wortarten) ist und $\vec{y}$ ein Syntaxbaum, lerne $X \rightarrow Y$, wobei nur in den Beispielen vorkommende Regeln verwendet werden.
- Formuliert als Maximierungsproblem:

$$h(\vec{x}) = argmax_{\vec{y} \in Y} P(\vec{y}|\vec{x}) = argmax_{\vec{y} \in Y} \left\{ \sum_{n_l \in rules(\vec{y})} \beta_l \right\}$$

  $rules(\vec{y})$ ist die Menge der Regeln, die in $\vec{y}$ verwendet sind.

Shallow Parsing Learning mit SVMstruct

- Es ergibt sich: $\langle \vec{\beta}, \Psi(\vec{x}, \vec{y}) \rangle = \sum_{n_l \in rules(\vec{y})} \beta_l$
- Den schwierigen Schritt $\vec{y^*} := argmax_{\vec{y} \in Y} \langle \vec{\beta}, \Psi(\vec{x}, \vec{y}) \rangle$ löst nun ein Parser, der sowohl den besten als auch den zweitbesten Syntaxbaum für $\vec{x}$ liefert. Somit können die *Beispiele* bei der Optimierung (Schritt 6) effizient bearbeitet werden.
- Das Lernergebnis ordnet *bisher nicht gesehenen Sätzen* $\vec{x}$ die richtigen Syntaxbäume zu. Dabei erweitert es die Fähigkeit der Grammatik – nicht die Menge der Regeln.

Experiment

- Trainingsmenge: 4098 Sätze mit maximal $p = 10$ Wörtern (dargestellt durch ihre Wortart)
- Testmenge: 163 Sätze mit maximal $p = 10$
- Maximum likelihood zum Lernen ergibt: $86,8\%$ precision, $85,2\%$ recall, $86\% F_1$ measure
- SVM_2 mit slack rescaling ergibt: $88,9\%$ precision, $88,1\%$ recall, $88,5\% F_1$ measure
- Der Unterschied des F-measures ist signifikant.
- SVM_2 hat in 12 Iterationen insgesamt 8043 Nebenbedingungen behandelt.
- Das Lernen dauerte insgesamt 3,4 Stunden, wovon die SVM_2 $10,5\%$ verwendete.

Andere Anwendungen der SVMstruct

- Wenn man die SVMstruct anwenden will, muss man
  - die Merkmalsabbildung $\Psi(\vec{x}, \vec{y})$ definieren und ggf. implementieren
  - die Verlustfunktion implementieren $\Delta(\vec{y_i}, \vec{y})$
  - die Selektion verletzter Bedingungen (Schritt 6 des Algorithmus') implementieren.
- Klassifikation mit Taxonomien
- Named Entity Recognition
- Mehrklassen-Klassifikation
- ...

Was wissen Sie jetzt?

- Sie wissen, was strukturelle Modelle sind: Y kann mehr sein als nur ein Wert.
- $\Psi(\vec{x}, \vec{y})$ erweitert die üblichen Beispiele so, dass nun wieder ein Skalarprodukt $\langle \vec{\beta}, \Psi(\vec{x}, \vec{y}) \rangle$ gerechnet werden kann.
- Das Problem sind die $N \times \mid Y \mid -N$ Nebenbedingungen, weil wir jedes Beispiel mit jedem anderen nicht nur bezüglich eines y, sondern bezüglich der $\mid Y \mid$ möglichen $\vec{y}$ vergleichen müssen.
- Dabei wird dieser Vergleich als *joint kernel* aufgefasst: $\langle \delta\Psi_i(\vec{y}), \delta\Psi_j(\vec{y'}) \rangle$. Es gibt noch viele andere Arbeiten zu *string kernels, tree kernels*, die Sie hier nicht kennen gelernt haben.

Sie wissen noch mehr!

- Der Ansatz von Joachims besteht darin,
  - dass als margin der Abstand zwischen der besten und der zweitbesten Lösung maximiert wird,
  - dass nur wenige der Nebenbedingungen wirklich behandelt werden müssen,
  - dass beliebige Verlustfunktionen Δ in den Nebenbedingungen und in der Auswahl der am stärksten verletzten Nebenbedingung verwendet werden können.

15 Cluster-Analyse

Lernaufgabe Clustering

- Gegeben
  - eine Menge $\mathcal{T} = \{\vec{x_1}, ..., \vec{x_N}\} \subset X$ von Beobachtungen,
  - eine Anzahl K zu findender Gruppen $C_1, ..., C_K$,
  - eine Abstandsfunktion $d(\vec{x}, \vec{x'})$ und
  - eine Qualitätsfunktion.
- Finde
  - Gruppen $C_1, ..., C_K$, so dass
  - alle $\vec{x} \in X$ einer Gruppe zugeordnet sind und
  - die Qualitätsfunktion optimiert wird: Der Abstand zwischen Beobachtungen der selben Gruppe soll minimal sein; der Abstand zwischen den Gruppen soll maximal sein.

Bild

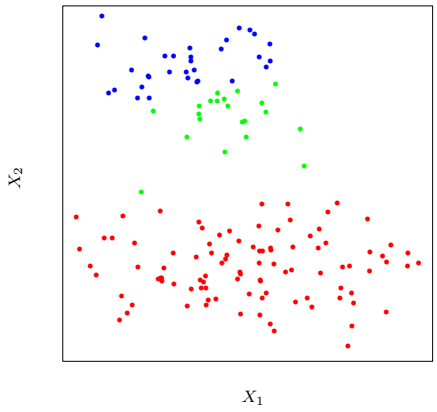


Figure 14.4: *Simulated data in the plane, clustered into three classes (represented by red, blue and green), by the K-means clustering algorithm*

Der Abstand wurde zum Cluster-Zentrum gemessen. Dadurch ergibt sich der grüne Punkt neben den roten.

- Könnte ein besseres Abstandsmaß den grünen Punkt dem roten Cluster zuweisen?
- Wenn nicht nur ein Punkt als Repräsentation eines Clusters gewählt wird, würde das Clustering dann besser?
- Wie kann man die Cluster verständlich beschreiben?
- Wäre $K = 2$ besser gewesen?

Die Probleme der Cluster-Analyse

1. Bestimmung des Abstandsmaßes
2. Formulierung des Optimierungsproblems
3. Repräsentation der Cluster
4. Bestimmung von K

15.1 Abstandsmaße

Bestimmung des Abstandsmaßes

- Ähnlichkeitsmaße haben wir schon bei kNN gesehen.
- Im Allgemeinen ist der Abstand invers zur Ähnlichkeit:

$$D(\vec{x_1}, \vec{x_2}) = 1 - Sim(\vec{x_1}, \vec{x_2})$$

- Man kann aber irgendeine geeignete monoton absteigende Funktion zur Überführung der Ähnlichkeiten in Abstände wählen.

***sim*: Ähnlichkeit für einzelne Attribute (Erinnerung)**

Numerische Attribute: Sei max_j der höchste Wert von X_j und min_j der niedrigste, sei x_{ij} der Wert des j-ten Attributs in der i-ten Beobachtung, dann ist die normalisierte Ähnlichkeit:

$$sim_j(x_{1j}, x_{2j}) = 1 - \frac{|x_{1j} - x_{2j}|}{max_j - min_j}$$

Nominale Attribute: Ganz einfach:

$$sim_j(x_{1j}, x_{2j}) = \begin{cases} 1 & falls \quad x_{1j} = x_{2j} \\ 0 & sonst \end{cases}$$

***d*: Abstand für einzelne Attribute**

Numerische Attribute: Ohne Normalisierung durch $max_j - min_j$ ist der Betrag der Differenz:

$$d_j(x_{ij}, x_{i'j}) =\mid x_{ij} - x_{i'j} \mid$$

Der quadratische Abstand zwischen Beobachtungen x_i und x'_i bezüglich des Merkmals X_j gewichtet große Abstände stärker als kleine:

$$d_j(x_{ij}, x_{i'j}) = (x_{ij} - x_{i'j})^2 \tag{51}$$

Nominale Attribute: Man kann für jede Variable X_j mit M Attributwerten eine $M \times M$ Abstandsmatrix angeben oder einfach:

$$d_j(x_{1j}, x_{2j}) = \begin{cases} 1 & falls \quad x_{1j} \neq x_{2j} \\ 0 & sonst \end{cases}$$

Sim: **Ähnlichkeit der Beobachtungen als Kombination der Attributähnlichkeiten**

Im einfachsten Fall mitteln wir die Einzelähnlichkeiten:

$$Sim(\vec{x_1}, \vec{x_2}) = \frac{1}{p} \sum_{j=1}^{p} sim(x_{1j}, x_{2j})$$

Die *Korrelation* verwendet das Mittel $\bar{x}_i$ über allen p Variablen:

$$Sim(\vec{x_1}, \vec{x_2}) = \frac{\sum_{j=1}^{p}(x_{1j} - \bar{x}_1)(x_{2j} - \bar{x}_2)}{\sqrt{\sum_{j=1}^{p}(x_{1j} - \bar{x}_1)^2 \sum_{j=1}^{p}(x_{2j} - \bar{x}_2)^2}} \tag{52}$$

Vielleicht sind einige Attribute wichtiger als andere?

$$Sim(\vec{x_1}, \vec{x_2}) = \frac{\sum_{j=1}^{p} w_j sim(x_{1,j}, x_{2,j})}{\sum_{j=1}^{p} w_j}$$

Wie bestimmt man w_j?

Abstandsmaß

- Verwendet wird eine $N \times N$ Matrix $\mathbf{D}$ für die N Beobachtungen, wobei d_{12} der Eintrag für $D(\vec{x_1}, \vec{x_2})$ ist.
- Die Matrix hat keine negativen Einträge.
- Die Diagonale der Matrix: $d_{ii} = 0$
- Der Abstand soll symmetrisch sein – falls nicht: $(\mathbf{D} + \mathbf{D}^T)/2$.

D: **Abstand der Beobachtungen als Kombination der Attributabstände**

- Gewichteter Durchschnitt:

$$D(\vec{x_1}, \vec{x_2}) = \sum_{j=1}^{p} w_j d_j(x_{1j}, x_{2j}); \sum_{j=1}^{p} w_j = 1 \tag{53}$$

- Bei quadratischem Abstand d_{12} ergibt sich:

$$D(\vec{x_1}, \vec{x_2}) = \sum_{j=1}^{p} w_j (x_{1j} - x_{2j})^2 \tag{54}$$

- Man kann die Korrelation (Gleichung 52) verwenden:

$$1 - Sim(\vec{x_1}, \vec{x_2}) \tag{55}$$

Einfluss einer Variablen auf das Clustering

- Wenn für alle Variablen $w_j = 1$ wäre, hätten doch nicht alle Variablen den gleichen Einfluss auf das Clustering!
- Der Einfluss einer Variable X_j richtet sich vielmehr nach ihrer durchschnittlichen Unähnlichkeit:

$$\bar{d}_j = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} d_j(x_{ij}, x_{i'j}) \tag{56}$$

- Beim gewichteten quadratischen Abstand

$$\bar{d}_j = \frac{1}{N^2} \sum_{i=1}^{N} \sum_{i'=1}^{N} (x_{ij} - x_{i'j})^2 = 2 \cdot var_j \tag{57}$$

  wobei var_j die anhand der Beobachtungmenge $\mathcal{T}$ geschätzte Varianz von X_j ist.
- Der Einfluss einer Variablen auf das Clustering richtet sich also nach der Varianz! Der relative Einfluss ist $w_j \ \bar{d}_j$.

Beispiel für Nachteil gleichen Einflusses der Variablen

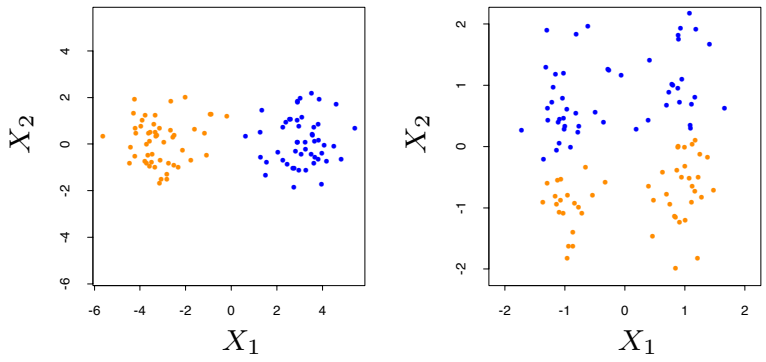


Figure 14.5: *Simulated data: on the left, K-means clustering (with K=2) has been applied to the raw data. The two colors indicate the cluster memberships. One the right, the features were first standardized before clustering. This is equivalent to using feature weights* $1/[2 \cdot \text{var}(X_j)]$. *The standardization has obscured the two well-separated groups. Note that each plot uses the same units in the horizontal and vertical axes.*

- Alle Variablen haben den selben Einfluss auf das Clustering, wenn $w_j \sim 1/\bar{d}_j$.
- Wenn als Gewichte $w_j = \frac{1}{2 \cdot var_j}$ gewählt wird, hat man den Einfluss der Varianz ausgeschaltet und erhält manchmal keine gute Separierung mehr.

Es hängt von der Anwendung ab, wie man w_j wählt!
Für eine Anwendung kann man vor dem Clustern

1. gar nichts tun, d.h. die Rohdaten ohne Gewichtung und ohne Normalisierung clustern,
2. die Rohdaten *normalisieren* (Werte im selben Wertebereich, z.B. $[0, 1]$, oder jeweils $max_j - min_j$ in den Abständen),
3. $\bar{d}_j$ für jedes Merkmal berechnen (Varianz-Gleichung 57),
4. die Rohdaten *standardisieren*, so dass alle Variablen den gleichen Einfuss haben,
5. Gewichte w_j, die dem Sachbereich entsprechen könnten oder dem Clustering-Ziel, direkt auf die Daten als Transformation der Eingabe anzuwenden. (*Implizites* w_j!)
6. Dann die Ergebnisse vergleichen!

15.2 Optimierungsprobleme

Qualitätsfunktionen

Sei die Anzahl K der Cluster gegeben und jedes Cluster durch eine ganze Zahl $k \in \{1, 2, ..., K\}$ eindeutig ausgezeichnet. Die Abbildung $C(i) = k$ weist der i-ten Beobachtung das k-te Cluster zu.

Innerer Abstand Within: Minimiert werden soll der Abstand innerhalb eines Clusters C:

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} D(\vec{x_i}, \vec{x_{i'}}) \tag{58}$$

Zwischenunähnlichkeit Between: Maximiert werden soll der Abstand zwischen Clustern:

$$B(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')\neq k} D(\vec{x_i}, \vec{x_{i'}}) \tag{59}$$

Optimierungsproblem der Cluster-Analyse

- Gegeben die Summe aller Abstände $T = \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} d_{ii'}$, ergänzen sich $W(C) + B(C) = T$, so dass die Minimierung von $W(C)$ der Maximierung von $B(C)$ entspricht.
- Man hat so nur *ein* Optimierungsproblem.
- Sei $\bar{x}_k = (\bar{x}_{1k}, ..., \bar{x}_{pk})$ der Vektor der Mittelwerte aller Variablen in Cluster k und $N_k = \sum_{i=1}^{N} I(C(i) = k)$, dann ist das Optimierungsproblem:

$$C^* = min_C \sum_{k=1}^{K} N_k \sum_{C(i)=k} \parallel \vec{x_i} - \bar{x_k} \parallel^2 \tag{60}$$

16 K-Means

Iteratives Lösen des Optimierungsproblems – K-Means

Definition 16.1 (Algorithmus K-Means($\mathcal{T}, K$))**.** *Ein Algorithus in 4 Schritten:*

1. *Wähle K Beobachtungen aus $\mathcal{T}$ zufällig als Mittelpunkte $\vec{m_1}, ..., \vec{m_K}$ von Clustern aus.*
2. *Berechne das Clustering anhand der Mittelpunkte:*

$$C(i) = argmin_{1 \leq k \leq K} \parallel \vec{x_i} - \vec{m_k} \parallel^2 \tag{61}$$

3. *Berechne die Mittelpunkte entsprechend $C(i)$:*

$$\vec{m_i} := argmin_m \sum_{i=1}^{N} \parallel \vec{x_i} - \vec{m} \parallel^2 \tag{62}$$

4. *Wiederhole Schritt 2 und 3 bis die Zuweisungen sich nicht mehr ändern. Gib zurück $C(1), ..., C(K)$.*

K-Means im Bild

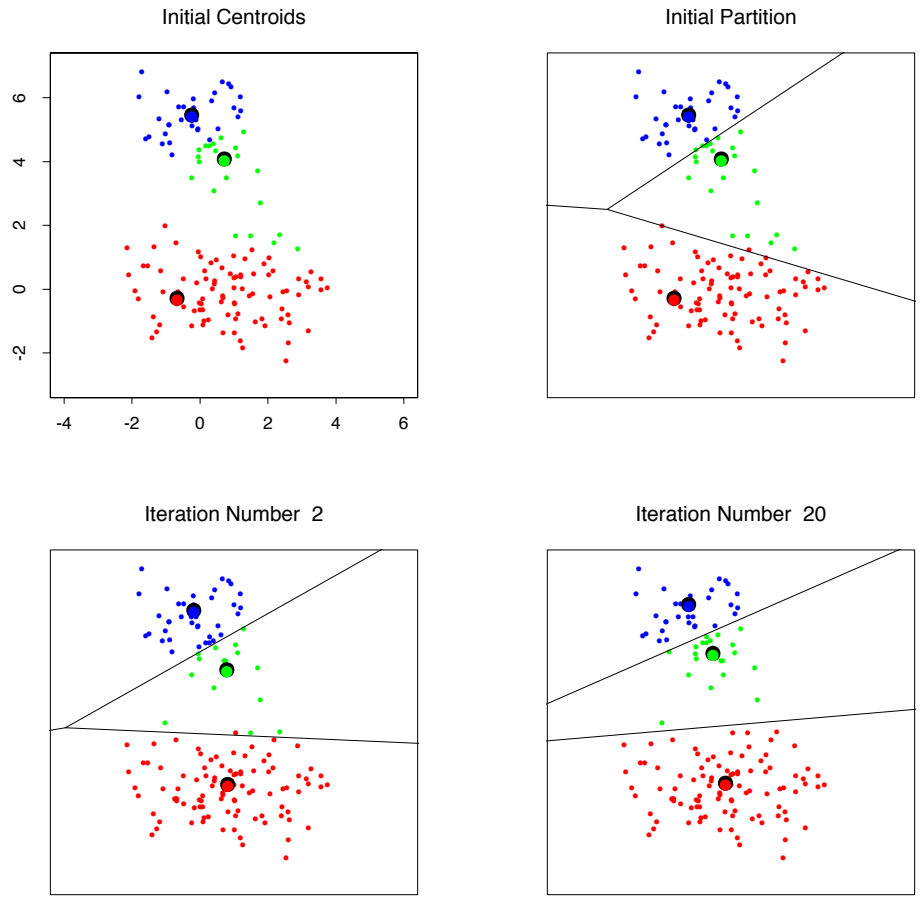


Figure 14.6: *Successive iterations of the K-means clustering algorithm for the simulated data of Figure 14.4.*

Eigenschaften von K-Means

- K-Means ist für numerische Variablen gemacht.
- Als Abstandsmaß wird der quadratische Euklidsche Abstand genutzt.
  - Den größten Einfluss haben Datenpunkte mit dem größten Abstand.
  - Das Verfahren ist daher anfällig für Ausreißer.
- Der Aufwand ist proportional zu $N \cdot K$.
  - Für jedes Cluster wird der Mittelpunkt berechnet anhand der zugeordneten Beobachtungen. Ein Cluster ist also nur durch einen Punkt repräsentiert.
  - Für alle Beobachtungen wird der Abstand zu den K Mittelpunkten berechnet.
- Es kann sein, dass die Lösung von K-Means nicht optimal ist (lokales Optimum).

Repräsentation der Cluster

- K-Means repräsentiert ein Cluster durch einen errechneten Punkt. Dies ist effizient.
- K-Medoid wählt eine Beobachtung als Repräsentation eines Clusters. Dafür muss über allen Punkten optimiert werden – ineffizient.
- Rajeev Rastogi hat vorgeschlagen *einige* Punkte als Repräsentation eines Clusters zu wählen (well scattered points).
- Oft möchte man eine interpretierbare Charaktierisierung der Cluster haben.
  - Aufgabe des *labeling*: finde eine (logische) Charakterisierung der Cluster. Man betrachtet die Cluster als Klassen und wendet z.B. Entscheidungsbaumlernen an.
  - Ryszard Michalski hat ein logisches Cluster-Verfahren vorgeschlagen, die Star-Methode (AQ-Algorithmus), bei dem direkt über den nominalen Werten der Beobachtungen gearbeitet wird.

Bestimmung der vorgegebenen Mittelpunkte

Die Lösung von K-Means hängt von den gewählten Start- mittelpunkten ab. Dafür gibt es mindestens zwei Auswege:

- Mehrfach mit zufällig gewählten Startmittelpunkten den Algorithmus starten!
- Optimierungskriterium

$$min_{C,\{m_k\}_1^K} \sum_{k=1}^{K} N_k \sum_{C(i)=k} \parallel \vec{x_i} - m_k \parallel^2$$

  Für $k = 1, ..., K$: Wähle einen Mittelpunkt i_k so, dass das Kriterium minimiert wird gegeben $i_1, ..., i_{k-1}$. Starte K-Means mit den so gefundenen K Mittelpunkten.

16.1 Bestimmung von K

Wie viele Cluster sollen gebildet werden?

- Vielleicht geht aus der Anwendung hervor, wie viele Cluster nötig sind. Z.B. sollen Kunden so auf K Vertriebsmitarbeiter aufgeteilt werden, dass ein Mitarbeiter ähnliche Fälle bearbeitet.
- Oft soll K^* anhand der Daten so ermittelt werden, dass die Clustering-Qualität optimiert wird (Gleichung 58).

$$W(C) = \frac{1}{2} \sum_{k=1}^{K} \sum_{C(i)=k} \sum_{C(i')=k} D(\vec{x_i}, \vec{x_{i'}})$$

  Man bestimmt $W_1, ..., W_{Kmax}$ für $K = 1, ..., Kmax$.

Daten-gestützte Bestimmung von K

- Wenn $K < K^*$, dann ist meist eine Teilmenge der Beobachtungen in einem Cluster schon richtig zugeordnet, das Cluster müsste aber weiter aufgeteilt werden.
  - $W_{K+1} << W_K$
- Wenn $K > K^*$, dann ist ein 'richtiges' Cluster zerteilt worden.
  - $W_{K+1} < W_K$.
- Man sucht also nach einem Knick in der Kurve der $W_1, ..., W_{Kmax}$-Werte und wählt als K den Wert mit dem geringsten Abstieg $W_K - W_{K+1}$.
  - $\{W_K - W_{K+1} \mid K < K^*\} >> \{W_K - W_{K+1} \mid K \geq K^*\}$

Gap Heuristik

- Tibshirani et al. (2001) vergleichen die Kurve der anhand der Daten gemessenen W-Werte mit einer "normalen".
- Es werden n Mal zufällig Datenpunkte erzeugt, die innerhalb einer Hülle um die Beobachtungen gleichmäßig verteilt sind.
- Für die simulierten Daten werden die W-Werte ausgerechnet und der Erwartungswert bestimmt.
- Die Kurven werden auf einer logarithmisierten Skala aufgetragen und verglichen: wo der Abstand zwischen den Kurven (gap) am größten ist, liegt das richtige K^*.

Gap Heuristik im Bild

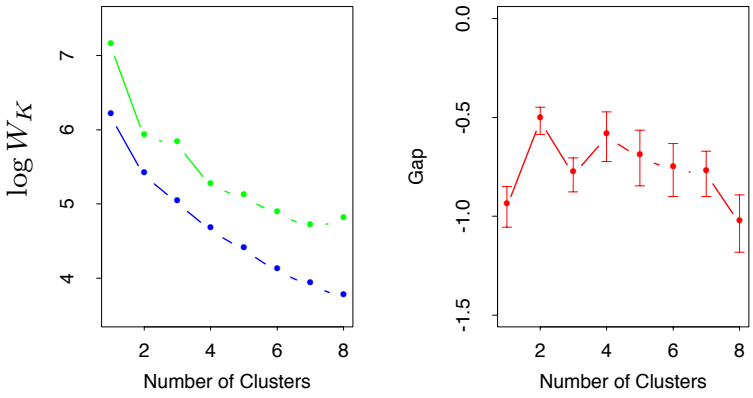


Figure 14.11: *Left panel: observed (green) and expected (blue) values of* $\log W_K$ *for the simulated data of Figure 14.4. Right panel: Gap curve, equal to the difference between the observed and expected values of* $\log W_K$*. The Gap estimate* K^* *is the smallest* K *producing a gap within one standard deviation of the maximum; here* $K^* = 2$*.*

Was wissen Sie jetzt?

- Sie haben die Abstandsmaße kennengelernt und sich dabei an die Ähnlichkeit bei kNN erinnert.
- Sie kennen das Optimierungsproblem des Clusterings (Gleichung 60).
- Sie kennen das Qualitätskriterium des inneren Abstands (Gleichung 58).
- Die Repräsentation eines Clusters kann durch alle zugeordneten Punkte, einige zugeordnete Punkte, einen zentralen zugeordneten Punkt oder ein berechnetes Zentrum sowie durch logische Formeln erfolgen.

- Zur Lösung des Optimierungsproblems kennen Sie K-Means: Euklidscher Abstand, Repräsentation durch berechnete Mittelpunkte, iteratives Vorgehen.
- Als Vorgehen zur Wahl der Anzahl K und zur Initialisierung der K Mittelpunkte haben Sie Heuristiken gesehen.

17 Hierarchisches Clustering

Hierarchisches Clustering

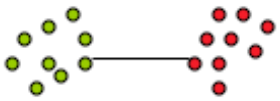
- Die Cluster sollen nicht auf einer Ebene liegen, sondern eine Taxonomie bilden.
- Die unterste Ebene enthält einzelne Beobachtungen.
- Jede Ebene enthält Cluster, die (zwei) Cluster der Ebene darunter subsummieren.
- Die oberste Ebene enthält ein Cluster mit allen Beobachtungen.
- Man unterscheidet ein Vorgehen bottom-up (agglomerativ) und top-down (aufteilend).

Agglomeratives Clustering

- Stufenweise werden Beobachtungen zu übergeordneten Clustern verschmolzen.
- Oft wird ein binärer Baum erzeugt, d.h. immer je 2 Cluster werden verschmolzen.
- Der Benutzer sucht die aussagekräftigste Ebene aus.
- Grundlage ist die *Unähnlichkeit von Clustern*: solche mit geringster Unähnlichkeit werden verschmolzen.
- Die Unähnlichkeit $d(G, H)$ der Cluster G, H wird berechnet durch den Abstand $d_{gh} = D(\vec{x_g}, \vec{x_h})$, wobei $\vec{x_g} \in G, \vec{x_h} \in H$.
- Welche Beobachtungen genutzt werden, macht den Unterschied zwischen den 3 wichtigsten Maßen zur Cluster-Unähnlichkeiten aus.

Single Linkage Clustering

Die Unähnlichkeit zwischen Cluster G und H ist die Unähnlichkeit der nächsten Punkte.



$$\begin{aligned} d_{SL}(G,H) &= min_{\vec{x_g} \in G, \vec{x_h} \in H} D(\vec{x_g}, \vec{x_h}) \\ &= min_{g \in G, h \in H}\ d_{gh} \end{aligned}$$

- Problem: Single Linkage ergibt eventuell Cluster, die nicht kompakt sind mit großer Unähnlichkeit innerhalb eines Clusters.

Complete Linkage Clustering

Die Unähnlichkeit zwischen Cluster G und H ist die Unähnlichkeit der entferntesten Punkte.

$$\begin{aligned} d_{CL}(G,H) &= max_{\vec{x_g} \in G, \vec{x_h} \in H} D(\vec{x_g}, \vec{x_h}) \\ &= max_{g \in G, h \in H}\ d_{gh} \end{aligned}$$

- Problem: Complete Linkage produziert kompakte Cluster, aber eventuell sind die Beobachtungen eines Clusters G näher zu denen eines anderen H als zu denen in G.

Average Linkage Clustering

Die Unähnlichkeit zwischen Cluster G und H ist die durchschnittliche Unähnlichkeit aller Punkte in G von allen in H.

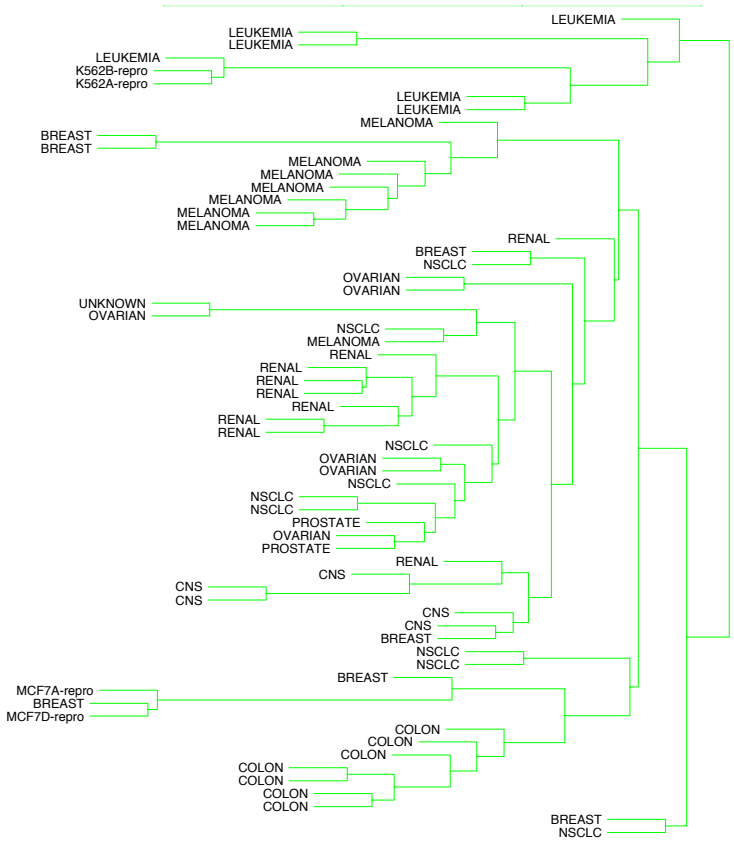
$$d_{AL}(G,H) = \frac{1}{N_G N_H} \sum_{g \in G} \sum_{h \in H} d_{gh}$$

- Kompromiss zwischen Single und Complete Linkage: relativ kompakte Cluster, die relativ weit von einander entfernt sind.
- Problem: Eine strikt monoton aufsteigende Transformation des Abstandsmaßes $h(d_{gh})$ kann das Ergebnis stark verändern.

Beispiel MicroArray-Daten über Krebs

Figure 1.3: *DNA microarray data: expression matrix of 6830 genes (rows) and 64 samples (columns), for the human tumor data. Only a random sample of 100 rows are shown. The display is a heat map, ranging from bright green (negative, under expressed) to bright red (positive, over expressed). Missing values are gray. The rows and columns are displayed in a randomly chosen order.*

Beispiel Average Linkage bei MicroArray-Daten über Krebs



Dendrogramme für agglomeratives Clustering der MicroArray-Daten über Krebs mit Average, Complete, Single Linkage

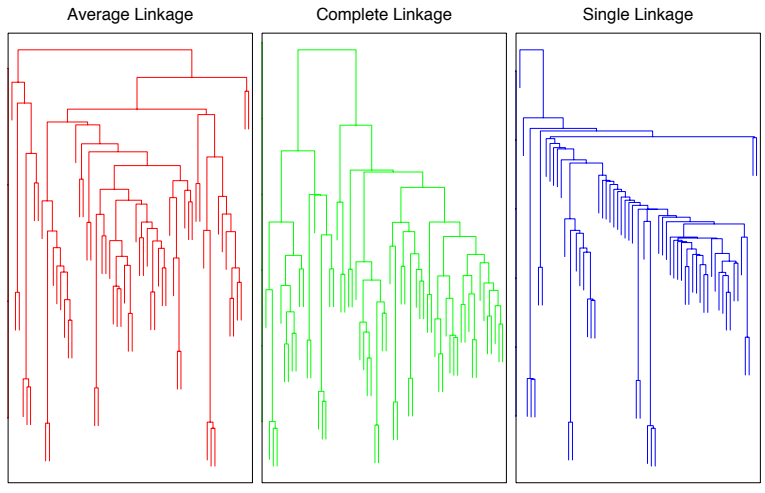


Figure 14.13: *Dendrograms from agglomerative hierarchical clustering of human tumor microarray data.*

Dendogramme

- Monotonie: Die Unähnlichkeit steigt über die Ebenen von unten nach oben monoton an.
- Ein Dendogramm ist so angeordnet, dass die Höhe eines Knoten (Clusters) gerade proportional zur Unähnlichkeit zwischen den beiden Unterknoten ist.
- Deshalb kann der Benutzer eine Ebene auswählen, bei der die Unähnlichkeit zwischen Clustern einen Schwellwert übersteigt.

Aufteilendes Clustering durch rekursives K-Means

- Die rekursive Anwendung von K-Means mit $K = 2$ ergibt ein aufteilendes Verfahren.
- Allerdings ist das Ergebnis dann kein Dendogramm, bei dem die Unähnlichkeit mit den Ebenen immer monoton ansteigt.
- Deshalb gibt es ein anderes Verfahren.

Aufteilendes Clustering durch iteratives Verringern der Unähnlichkeit in einem Cluster

- Alle Beobachtungen sind im Wurzelknoten G.
- Aufteilung(G)
  1. Initialisierung:Wähle den Punkt $\vec{x_h}$ in G, der am unähnlichsten zu allen anderen ist. Dieser wird dem neuen Cluster H zugeordnet.
  2. Teile iterativ G auf solange es ein $\vec{x_i} \in G$ gibt, das im Durchschnitt ähnlicher zu allen $\vec{x_j} \in H$ ist als zu allen $\vec{x_g} \in G$:$H := H \cup \{\vec{x_i}\}$; $G := G \backslash \{\vec{x_i}\}$;
  3. Wähle Cluster zur Aufteilung aus:Solange $\mid G \mid > 1$ und $d_{ij} > 0$ für alle $\vec{x_i}, \vec{x_j} \in G$ Aufteilung(G). Solange $\mid H \mid > 1$ und $d_{ij} > 0$ für alle $\vec{x_i}, \vec{x_j} \in H$ Aufteilung(H).

Was wissen Sie jetzt?

- Top-down Clustering kann durch rekursives K-Means realisiert werden, ist aber aufwändig.
- Optimieren der Average Linkage $d_{AL}(G, H)$ für alle möglichen Aufteilungen wird angenähert durch ein iteratives Verfahren, bei dem in jeder Iteration eine Beobachtung von dem Ausgangscluster G dem neuen Cluster H zugeordnet wird.
- Kann man das effizienter machen?

18 Organisation von Sammlungen

Organisation von Sammlungen

Sammlungen von Fotos, Musik, Filmen bevölkern PCs und das Internet. Sie sind organisiert

- in Taxonomien nach vorgegebenen Kriterien
  - iTunes: Genre, Artist, Album, Jahr
- in Taxonomien nach eigenen Kriterien
  - flickR: Sammlung, Album, Gruppen – annotiert wird mit eigenen tags.
- einfache Dateien, evtl. mit Benutzeroberfläche
  - iPhoto: Ereignisse, jedes Bild kann annotiert werden.

Wie organisieren Menschen Medien?

- Studie von Jones, Cunningham, Jones (2004): Studenten wurden befragt, wie sie ihre CDs, DVDs, Bücher organisieren.
  - Nachttisch, spezieller Schrank, Auto, Küche
  - Gelegenheiten zur Nutzung
  - Aktualität, Anschaffungszeitpunkt
- Studie von Vignoli (2004): Ordnung digitaler Musik auf PCs wurden untersucht.
  - Meist wurden hierarchische Strukturen aufgebaut.
  - Es gibt immer einen Ordner mit nicht einsortierter Musik.
- Studie PG 461 "Kollaboratives Strukturieren von Multimediadaten für Peer-to-Peer-Netze"
  - Verschiedene Aspekte: Gelegenheiten ("beim Autofahren", "Dinner", "Party"), Personen ("für Susie"), Erinnerungen ("Sommer03"), Stimmungen, Tempi, Genres
  - Wieder gibt es Ordner mit nicht einsortierter Musik.

Automatisches Sortieren von Mediensammlungen

- Medien sollen hierarchisch strukturiert werden.
- Die Taxonomien sollen personalisiert sein.
  - Die Bezeichner sind unterschiedlich: was dem einen "fröhliche Tanzmusik", gehört bei dem anderen unter "Depression" (The Cure).
  - Bereiche, die einer fein strukturiert, fasst der andere zusammen.
  - Verschiedene Benutzer stellen verschiedene Mengen als ähnlich betrachteter Medien zusammen.
- Derselbe Benutzer verwendet mehrere, unterschiedliche Hierarchien (*Aspekte*), die teilweise gleiche Medien abdecken.
- Die Einsortierung neuer Medien soll automatisch erfolgen.
- Die Struktur soll automatisch erweitert werden, ohne den Benutzer zur bevormunden.

18.1 Web 2.0

Web 2.0

- Semantic Web:
  - Semantische Beschreibung
  - Vorgegebene, allgemeine Ontologie
  - Logische Beschreibungssprache
  - top-down Modellierung
- Web 2.0
  - Freies Tagging der Benutzer
  - Entstehende *Folksonomies*
  - Statistische Methoden
  - Empfehlungssysteme

Sammlungen im Web 2.0

- Verschiedene Benutzer laden ihre Medien hoch.
- Verschiedene Benutzer annotieren ihre Medien.
- Kollaborative Empfehlung:
  - Ein Benutzer sind einander ähnlich, wenn sie ähnliche Mengen von Medien ausgewählt haben.
  - Medien sind einander ähnlich, wenn sie in Sammlungen ähnlicher Benutzer vorkommen.
  - Meist werden nur flache Medienmengen betrachtet (Amazon, Last.fm). Es werden auch nur Listen von Medien empfohlen.
- Für die automatische Unterstützung der Strukturierung reicht das nicht.

18.2 Clustering verteilter Daten

Clustering Mediensammlungen

- **Ziel:** Hierarchisches Clustering erzeugt für einen Benutzer anhand seiner und der Clusterings anderer Benutzer je Aspekt mehrere Taxonomien zur Auswahl.
  - Wie kann das Benutzer gegebene Clustering beibehalten und nur ergänzt werden?
    → Supervised Clustering
  - Wie kann ein Benutzer von den Strukturierungen anderer Benutzer profitieren?
    → Distributed Clustering, Ensemble Clustering
  - Wie kann das Verfahren mehrere alternative Clusterings zur Auswahl anbieten?
    → Nonredundant Clustering

Supervised Clustering

- **Constraint Clustering** (Cohn, Caruana, McCallum 2003) beachtet bei der Optimierung vom Benutzer gegebene Nebenbedingungen
  - $must - link(\vec{x_g}, \vec{x_g'})$, d.h. $\vec{x_g}, \vec{x_g'}$ müssen im selben Cluster sein;
  - $cannot - link(\vec{x_g}, \vec{x_h})$, d.h. $\vec{x_g}, \vec{x_h}$ dürfen nicht im selben Cluster sein.
- **Supervised Clustering** (Finley, Joachims 2005) beachtet bei der Optimierung als Nebenbedingungen, dass einige Cluster mit zugeordneten Beobachtungen vorgegeben sind:

$$C(i) = k \text{ für } \vec{x_i}, i = 1, ..., M, M < N$$

$$C_k, k = 1, ..., L, L \leq K$$

- Leider nur für flache Clusterings und nicht für mehrere, verteilte gegebene Clusterings!

Distributed Clustering

- Verteilte Daten sollen gruppiert werden.
- Horizontale Verteilung:
  - Alle Daten haben die selben Merkmale, sind aber auf verschiedene Rechner verteilt.
  - Kein Datum ist mehr als einem Rechner zugeordnet.
  - Typisches Beispiel: Filialen eines Geschäfts.
- Vertikale Verteilung:
  - Daten der verschiedenen Rechner haben unterschiedliche Merkmale.

  - Das selbe Objekt ist auf mehreren Rechnern zu finden.
  - Typisches Beispiel: Mediensammlungen Web 2.0.
- Ziel ist ein *Konsens-Modell* als gemeinsames Clustering für alle Daten.
- Das ist nicht das Ziel bei der Strukturierung persönlicher Mediensammlungen!

Ensemble Clustering

- Ensemble Clustering kombiniert eine Menge gegebener Clusterings (Strehl, Ghosh 2002).
- Alle Clusterings decken die selbe Menge von Beobachtungen ab.
  - Zusätzliches Ähnlichkeitsmaß: kommen gemeinsam in einem Cluster vor (Topchy, Jain, Punch 2003);
  - Zuordnung zu einem gegebenen Cluster als zusätzliches Merkmal einer Beobachtung – dann in diesem Raum k-Means anwenden!
- Wieder wird ein Konsens-Modell erzeugt!

Nonredundant Clustering

- Gegeben ein Clustering $C(i) = k$ für Beobachtungen $\vec{x_i}, i = 1, ..., N$ und Cluster $C_k, k = 1, ..., K$
- finde ein alternatives Clustering C', das möglichst orthogonal zu C ist. (Gondek, Hofmann 2004)
- Das Verfahren erhält keine gegebenen Strukturierungen, sondern bietet Alternativen zum gesamten Clustering an.

Es gibt noch kein geeignetes Verfahren für das Strukturieren persönlicher Sammlungen im Web 2.0

- Bisherige Ansätze reichen nicht aus:
  - Supervised clustering ist noch nicht geeignet für hierarchische Strukturen und die Eingabe mehrerer Clusterings.
  - Distributed clustering und Ensemble Clustering erstellen ein Konsens-Modell, das die eigene Annotation von Benutzern überschreiben würde.
  - Nonredundant clustering erhält in den Alternativen nicht das gegebene Clustering.
- Wir mussten also ein eigenes Verfahren entwickeln: Localized Alternative Clustering of Ensembles

19 LACE

Lernaufgabe Localized Alternative Clustering of Ensembles

- Wir sprechen jetzt statt von der Zuordnung $C(i) = k$ einer Beobachtung $\vec{x_i}$ zu einem Cluster C_k von dem Clustering φ_i von einer Menge von Beobachtungen S_i auf ein Cluster G_i.
- Gegeben eine Menge $S \subseteq X$, eine Menge von Clusterings $I \subseteq \{\varphi_i : S_i \to G_i\}$ und eine Qualitätsfunktion

$$q : 2^{\Phi} \times 2^{\Phi} \times 2^{S} \to \mathcal{R} \tag{63}$$

  localized alternative clustering ensembles findet Clusterings $O \subseteq \{\varphi_i | \varphi_i : S_i \to G_i\}$ so dass die Qualität $q(I, O, S)$ maximiert wird und für jedes $\varphi_i \in O$ gilt, dass S Teil seines Ursprungsbereichs ist: $S \subseteq D_{\varphi_i}$.

φ als hierarchisches Clustering

- Die Cluster sollen nicht auf einer Ebene liegen, sondern eine Taxonomie bilden.
- Die unterste Ebene enthält Mengen von Beobachtungen.
- Jede Ebene enthält Cluster, die die Cluster der Ebene darunter subsummieren: jeder Teilbaum von Clustern ist eine Taxonomie.
- Die oberste Ebene enthält ein Cluster mit allen Beobachtungen.
- Man unterscheidet ein Vorgehen bottom-up (agglomerativ) und top-down (aufteilend).
- $\varphi_i : S_i \to G_i$ soll die Menge S_i hierarchisch aufteilen, d.h. G_i soll eine Hierarchie von Clustern sein.

Zur Erinnerung: Agglomeratives Clustering

- Stufenweise werden Beobachtungen zu übergeordneten Clustern verschmolzen.
- Grundlage ist die *Unähnlichkeit von Clustern*: solche mit geringster Unähnlichkeit werden verschmolzen.
- Die Unähnlichkeit $d(G, H)$ der Cluster G, H wird berechnet durch den Abstand $d_{gh} = D(\vec{x_g}, \vec{x_h})$, wobei $\vec{x_g} \in G, \vec{x_h} \in H$.
- Welche Beobachtungen genutzt werden, macht den Unterschied zwischen den 3 wichtigsten Maßen zur Cluster-Unähnlichkeiten aus.
  - Single Linkage Clustering: Die Unähnlichkeit zwischen Cluster G und H ist die Unähnlichkeit der nächsten Punkte.
  - Complete Linkage Clustering: Die Unähnlichkeit zwischen Cluster G und H ist die Unähnlichkeit der entferntesten Punkte.
  - Average Linkage Clustering: Die Unähnlichkeit zwischen Cluster G und H ist die durchschnittliche Unähnlichkeit aller Punkte in G von allen in H.

Erweiterung eines Clustering

Wir wollen ein gegebenes Clustering erweitern, d.h.:

- Bestehende Zuordnungen bleiben.
- Bisher abgedeckte Beobachtungen bleiben abgedeckt.
- Zusätzliche Beobachtungen werden abgedeckt.

Definition 19.1 (Erweiterte Funktion)**.** *$\varphi'_i : S'_i \to G_i$ ist die erweiterte Funktion für $\varphi_i : S_i \to G_i$, wenn $S_i \subset S'_i$ und $\forall \vec{x} \in S_i : \varphi_i(\vec{x}) = \varphi'_i(\vec{x})$.*

Beutel von Clusterings

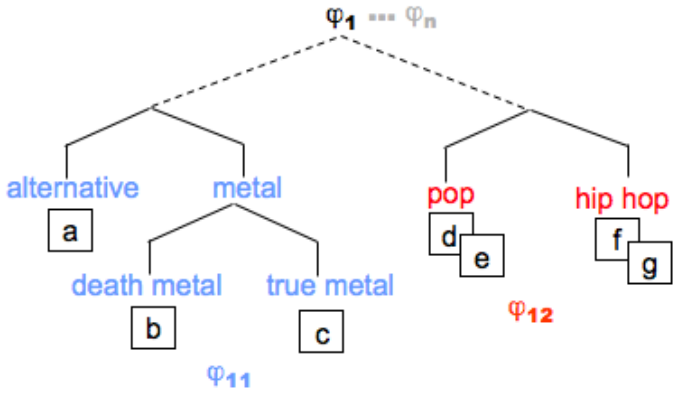
Wir wollen die noch nicht strukturierten Beobachtungen in S durch vorhandene Clusterings $\varphi_1, ..., \varphi_m$ abdecken.

Definition 19.2 (Beutel von Clusterings)**.** *Sei I eine Menge von Clusterings. Ein Beutel von Clusterings ist eine Funktion*

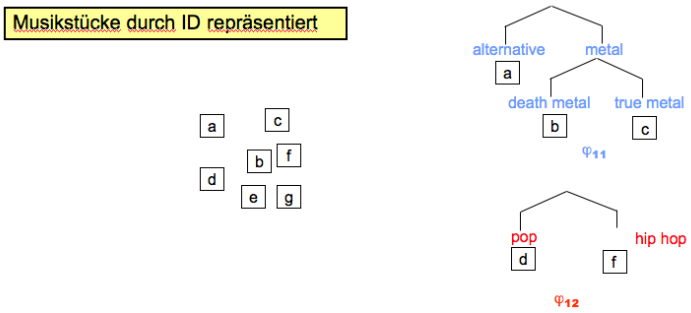
$$\varphi_i(\vec{x}) = \begin{cases} \varphi'_{i1}(x), & \text{wenn } \vec{x} \in S'_{i1} \\ \vdots & \vdots \\ \varphi'_{ij}(x), & \text{wenn } \vec{x} \in S'_{ij} \\ \vdots & \vdots \\ \varphi'_{im}(x), & \text{wenn } \vec{x} \in S'_{im} \end{cases} \tag{64}$$

wobei jedes φ'_{ij} eine Erweiterung eines $\varphi_{ij} \in I$ ist und $\{S'_{i1}, \ldots, S'_{im}\}$ ist eine Partitionierung von S.

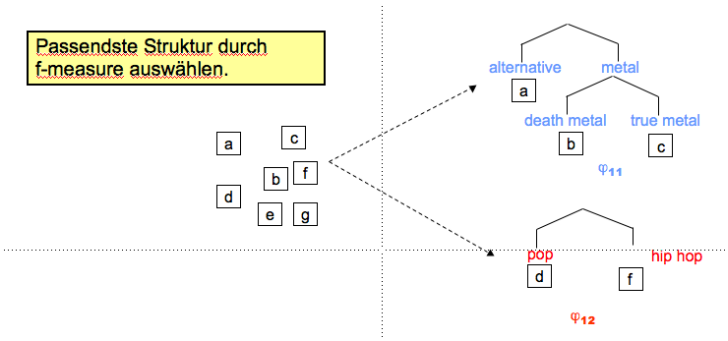
Beutel von Clusterings im Bild



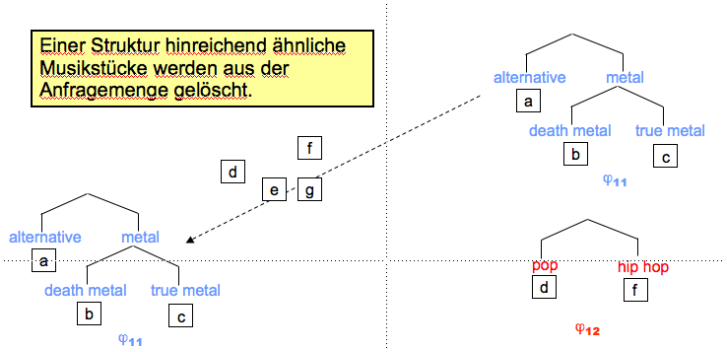
LACE in Bildern - 1: Nicht eingeordnete Stücke, Clusterings anderer Benutzer



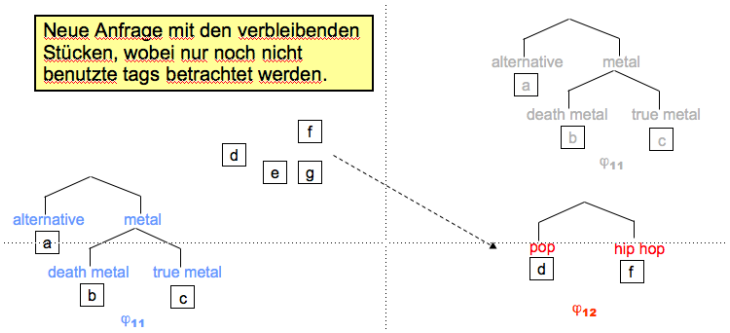
LACE in Bildern - 2: Finden passender Clusterings



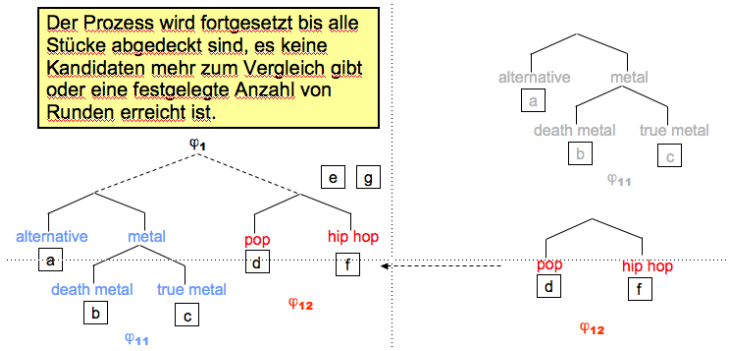
LACE in Bildern - 3: Löschen abgedeckter Stücke



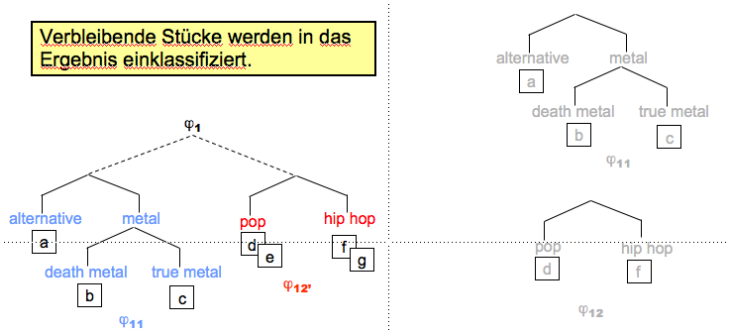
LACE in Bildern - 4: Finden passender Clusterings für den Rest



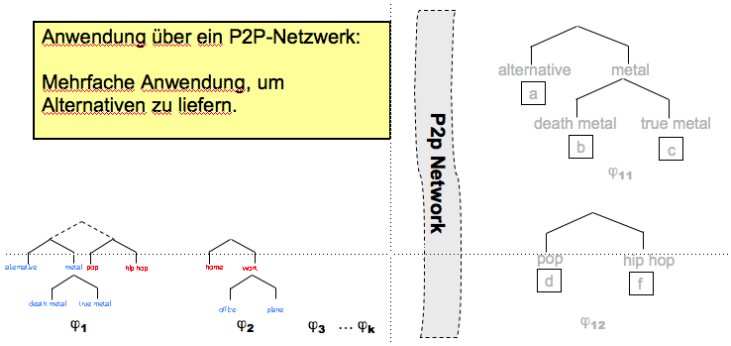
LACE in Bildern - 5: Abbruchbedingung für das sequentielle Abdecken



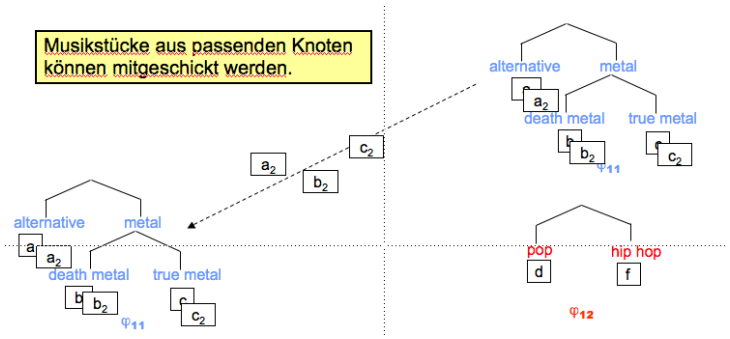
LACE in Bildern - 6: Klassifkation von Stücken in neue Struktur



LACE in Bildern - 7: Posten der abzudeckenden Stücke ins P2P-Netz, Empfangen der passenden Clusterings



Personalisierte Empfehlungen



Qualitätsfunktion für Clustering und Menge von Objekten

- Bei der Repräsentation eines Clusters durch *well-scattered points* ist Z_{φ_i} die Menge von Beobachtungen, die φ_i beschreibt. β sei eine Gewichtung, die Precision und Recall ins Verhältnis setzt:
- Precision:
$$prec(Z_{\varphi_i}, S) = \frac{1}{|Z_{\varphi_i}|} \sum_{\vec{z} \in Z_{\varphi_i}} max\{sim(\vec{x}, \vec{z}) | \vec{x} \in S\}.$$
- Recall:
$$rec(Z_{\varphi_i}, S) = \frac{1}{|S|} \sum_{\vec{x} \in S} max\{sim(\vec{x}, \vec{z}) | \vec{z} \in Z_{\varphi_i}\}.$$
- F-Measure:
$$q_f^*(Z_{\varphi_i}, S) = \frac{(\beta^2 + 1) rec(Z_{\varphi_i}, S) prec(Z_{\varphi_i}, S)}{\beta^2 rec(Z_{\varphi_i}, S) + prec(Z_{\varphi_i}, S)}. \quad (65)$$

Basisalgorithmus Sequenzielles Abdecken

- $O = \emptyset$, $J = I$
- WHILE($|O| < max_{alt}$)
  - $S_u = S$, $B = \emptyset$, $step = 0$
  - WHILE($(S_u \neq \emptyset) \wedge (step < max_{steps})$)
    - $\varphi_i = \arg\max_{\varphi \in J} q_f^*(Z_\varphi, S_u)$
    - $S_u = S_u \setminus \{\vec{x} \in S_u | \vec{x} \sqsubset_\alpha \varphi_i\}$
    - $B = B \cup \{\varphi_i\}$
    - $step = step + 1$
  - $O = O \cup \{bag(B, S)\}$
- Wobei max_{alt} die maximale Anzahl an Alternativen angibt, die Funktion $bag(B, S)$ einen Beutel von Clusterings angibt, der jedem Stück $\vec{x} \in S$ das Clustering $\varphi_i \in B$ zuweist, das die zu $\vec{x}$ ähnlichsten Objekte enthält.

Hierarchisches Vorgehen: Rekursiv Precision und Recall berechnen!

$$prec(Z_{\varphi_i}, S) = \begin{array}{ll} \frac{|Z^*_{\varphi_i}|}{|Z_{\varphi_i}|} prec(Z^*_{\varphi_i}, S) + & updateSchritt \\ \sum_{\varphi_j \prec \varphi_i} \frac{|Z_{\varphi_j}|}{|Z_{\varphi_i}|} prec(Z_{\varphi_j}, S) & direkterNachfolger \end{array}$$

wobei $Z^*_{\varphi_i} = Z_{\varphi_i} \setminus \bigcup_{\varphi_j \prec \varphi_i} Z_{\varphi_j}$ *nur Oberknoten.*

- Die hierarchischen Funktionen φ_j und φ_i, sind in direkter Nachfolgerelation $\varphi_j \prec \varphi_i$, gdw.

$$G_j \subset G_i$$
$$\forall \vec{x} \in S_i : \varphi_j(\vec{x}) = \varphi_i(\vec{x}) \cap G_j$$
$$\neg \exists \varphi_i' : G_j \subset G_j' \subset G_i$$

- Wenn eine optimistische Schätzung des F-measure schon am Wurzelknoten schlechter als ein Schwellwert ist, muss das Clustering nicht weiter untersucht werden!

19.1 Experimente mit LACE

Daten

- $\varphi_1, ..., \varphi_{39}$ sind 39 Taxonomien für eine Musiksammlung von 1886 Stücken.
- Es wird immer eine Taxonomie weggelassen und auf die restlichen LACE angewandt.
- Das Ergebnis wird mit der weggelassenen Taxonomie verglichen. Differenz der absoluten Tree Distance zwischen zwei Beobachtungen in beiden Taxonomien:

S	x_1	x_2	...	x_m	sum of differences
x_1	-	φ:1;φ':3			2+
x_2		-		φ:1;φ':2	1+
...			-		
x_m				-	
Total					3+

Andere Kriterien und Verfahren

- Andere Kriterien: Korrelation zwischen den Tree Distances FScore:
  - Jedes Cluster der weggelassenen Taxonomie wird mit jedem Cluster der gelernten verglichen (Precision und Recall –¿ F-measure) und das jeweils beste ausgewählt. Der Durchschnitt ergibt den FScore.
- Single-linkage agglomeratives Clustering
- TD: Rekursives top-down K-Means (Guan, Kulis 2004)
- Mehrfaches Starten, um zu Ensembles zu kommen, von denen stets das beste ausgesucht wird.

Ergebnisse

Method	Correlation	Absolute distance	FScore
LACE	0.44	0.68	0.63
TD ensemble	0.23	2.5	0.55
single-link ensemble	0.17	9.9	0.60
random	0.09	1.8	0.5

Representation	Correlation	Absolute distance	FScore
all points	0.44	0.68	0.63
$\|Z\| = 10$	0.44	0.68	0.63
$\|Z\| = 5$	0.41	0.69	0.63
$\|Z\| = 3$	0.40	0.69	0.62
centroid	0.19	1.1	0.42

Was wissen Sie jetzt?

- Sie haben das Feld der Strukturierung von Sammlungen im Web 2.0 kennen gelernt.
- Sie kennen eine neue Lernaufgabe: lokale alternative Cluster Ensembles und einen Algorithmus dazu.
- Insbesondere haben Sie dabei gesehen, dass man aus der unüberwachten Lernaufgabe des Clusterings manchmal eine halb-überwachte machen kann:
  - Für einzelne Beobachtungen ist angegeben, ob sie im selben oder in verschiedenen Clustern landen sollen (Constraint Clustering).
  - Es soll eine bestimmte Menge von Objekten abgedeckt (strukturiert) werden (LACE).
  - Es soll eine bestimmte Struktur erhalten, aber erweitert werden (Supervised Clustering, LACE).
- Und Sie haben gesehen, wie man Strukturen anderer Benutzer (über ein P2P Netz) nutzen kann.

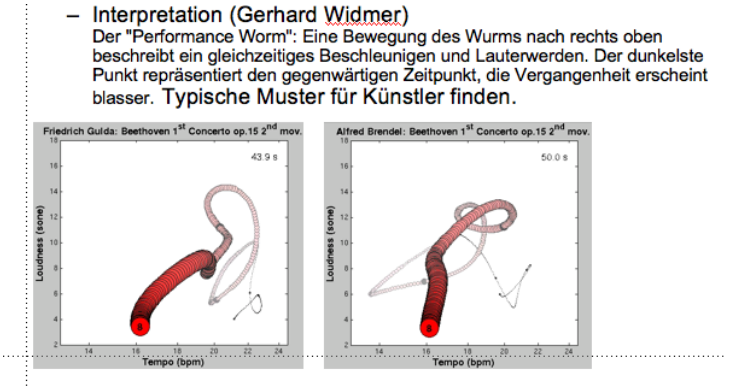
20 Musik als Daten

Technische Grundlagen

- Moving Pictures Expert Group Audio Layer 3 Karlheinz Brandenburg, TU Ilmenau, Fraunhofer Institut Standard für Musik und Filme, min. 1/12 komprimiert
- Tauschbörsen für Musik:
  - Napster 80 Mio. Benutzer, Nachfolger: Morpheus, Gnutella, KaZaA
  - KaZaA 500 Mio. Musikstücke
  - Privatsammlungen oft mehr als 10 000 Musikstücke
- Speichern, Abspielen, GUI zum Anbieten von Musik

Arbeitsfelder – Musik

Wissenschaftliche Untersuchung von Musik



Arbeitsfelder – Music Information Retrieval

- Anfragen: über ID3 tags (Metadaten), query by humming
- Indexierung: über Metadaten, über tags der Benutzer
- Navigation in Sammlungen gemäß Ähnlichkeit
- Klassifikation von Musik
- Empfehlungen

Arbeitsfelder – Intelligente Systeme

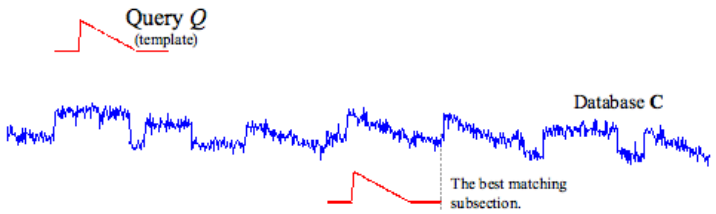
- Automatische Annotation von Musik
- Automatische Klassifikation von Musik nach
  - Genre (nur noch als Benchmark)
  - Benutzerpräferenzen
  - arbiträren tags (Aspekten)
- Automatische Organisation von Sammlungen
- Empfehlungen

Technischer Kern

- Musikdaten sind Zeitreihen der Elongation.
- Wir müssen Ähnlichkeiten von Zeitreihen erkennen. Das ist der technische Kern in fast allen Lernverfahren.
- Ähnlichkeit von Zeitreihen bisher:
  - Ähnlichkeit der Kurven
  - Dynamic Time Warping: Ähnlichkeit mit Verzerrung
- Achtung: Zeitreihenanalyse untersucht *eine* Zeitreihe und sagt neue Werte in der Zukunft voraus. Hier geht es aber um die Klassifikation oder das Clustering von *vielen* Zeitreihen. (Eamonn Keough)

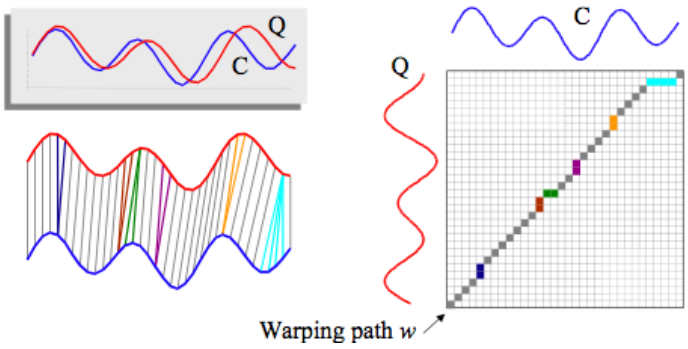
Ähnlichkeit von Zeitreihen

- Gegeben eine Anfrage Q, eine Datenbank mit Zeitreihen C und ein Abstandsmaß,
- finde den Ort in einer Reihe in C, der Q am ähnlichsten ist.



Dynamic Time Warping

$$\gamma(i,j) = d(q_i,c_j) + \min\{ \gamma(i-1,j-1), \gamma(i-1,j), \gamma(i,j-1) \}$$



So geht es nicht! Nötig ist die Merkmalsextraktion.

- Musikdaten geben die Ähnlichkeit von Musik nicht wieder. Musik ist nicht ähnlich, wenn die Elongation ähnlich ist.
- Aus den Elongationsdaten müssen Merkmale extrahiert werden, nach denen die Ähnlichkeit bestimmt werden kann.
- Merkmalsextraktion ist die Voraussetzung für:
  - Annotation
  - Indexing
  - Clustering
  - Kassifikation

Merkmalsextraktion

- Eine Reihe von *low level descriptors* wird extrahiert:
  - Lautstärke
  - Peaks, Verhältnis vom höchsten zum zweithöchsten Peak, ...
  - Zero Crossing Rate
  - Spectral Centroid (Cepstral)
  - Mel Frequency Cepstral Coefficient (MFCC)
- Es gibt einen Merkmalssatz, der sich häufig bewährt: Tzanetakis, Dissertation 2002

Ergebnis von Pohle et al. 2005: je Lernaufgabe ist anderer Merkmalssatz nötig!

- Gegeben eine Menge low level descriptors, klassifiziere nach einem Aspekt
  - Genre
  - Stimmung
  - Tempo
  - Instrument vs. Gesang vs. beides
- Es gibt keine Menge von Merkmalen, die alle Klassifikationsaufgaben lösen hilft.
- Je Lernziel (Aspekt) ist ein anderer Merkmalssatz nötig.
- Tzanetakis' Merkmale sind immer einigermaßen gut.

Mierswa Diplomarbeit 2004

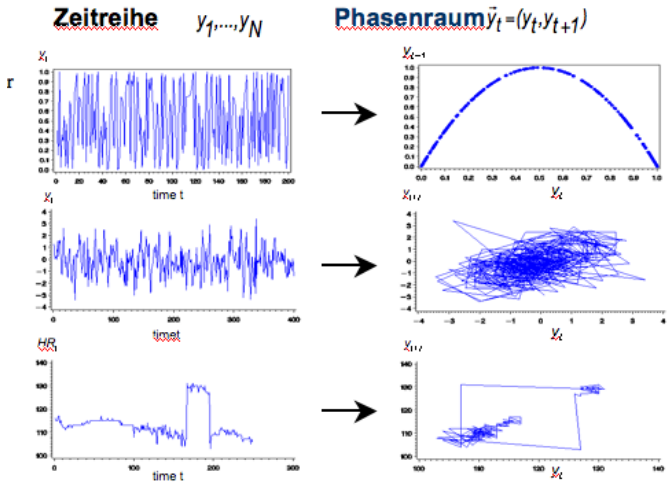
- Jeder Mensch achtet auf Unterschiedliches, um Musik zu beurteilen.
- Dieselbe abstrakte Eigenschaft wird anhand völlig unterschiedlicher Merkmale der physikalischen Ebene zugeschrieben.
- Für persönliche Empfehlungen sind auch persönliche Merkmale nötig.
- Also: lernende Merkmalsextraktion für automatische Klassifikation!

20.1 Lernende, adaptive Merkmalsextraktion

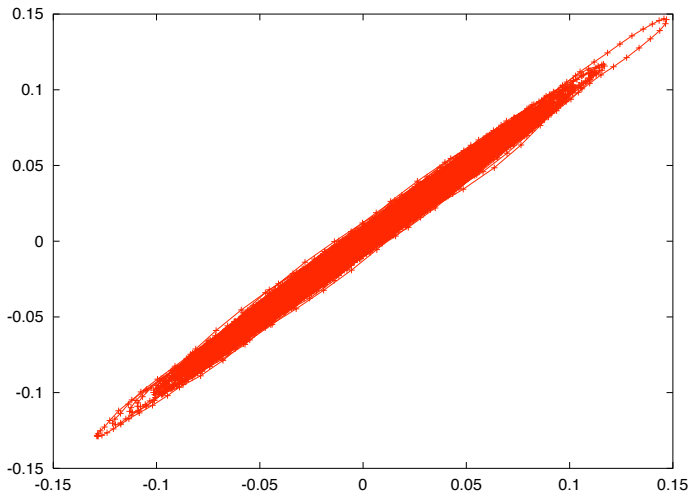
Merkmalsraum strukturieren

- Zeitraum (index)
  - Mittlere Lautstärke: $LS(\vec{x}) = \frac{1}{N} \sum_{i=1} \mid x_i \mid$
  - Tempobestimmung durch Autokorrelation verschobener Reihen: für alle Geschwindigkeiten 90 - 170 bpm: Verschiebung der Reihe um einen Takt, berechnen der Differenz zum Original, wenn die Differenz minimal ist, ist das richtige Tempo bestimmt.
- Frequenzraum
  - Für uns ist die diskrete Fourier-Transformation interessant, insbesondere die schnelle (FFT). Dafür muss die Anzahl der Abtastpunkte eine Zweierpotenz sein. Bei FFT geht die Information verloren, wann die Frequenzen auftreten. Also wird ein Zeitfenster über die Reihe verschoben, innerhalb dessen FFT angewandt wird.
- Phasenraum: gegeben die Messwerte $y_1, ...y_N$ für die Zeitpunkte $1, ..., N$, bilde eine neue Reihe mit den Werten y_{i-1} für die Punkte y_i.

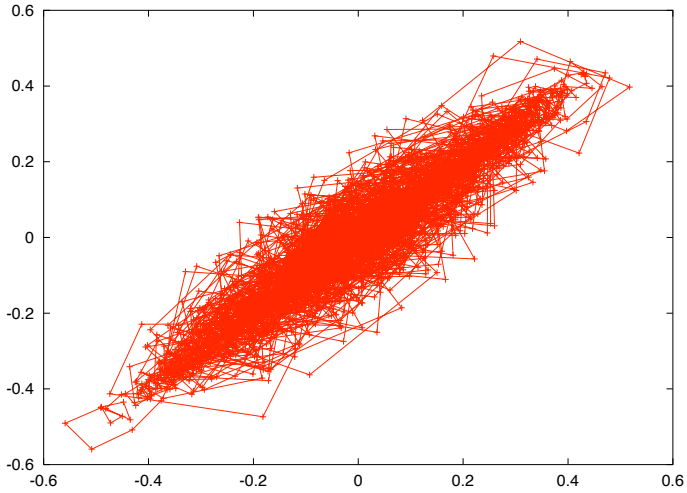
Phasenraum



Phasenraum zur Klassifikation von Genre: Klassik



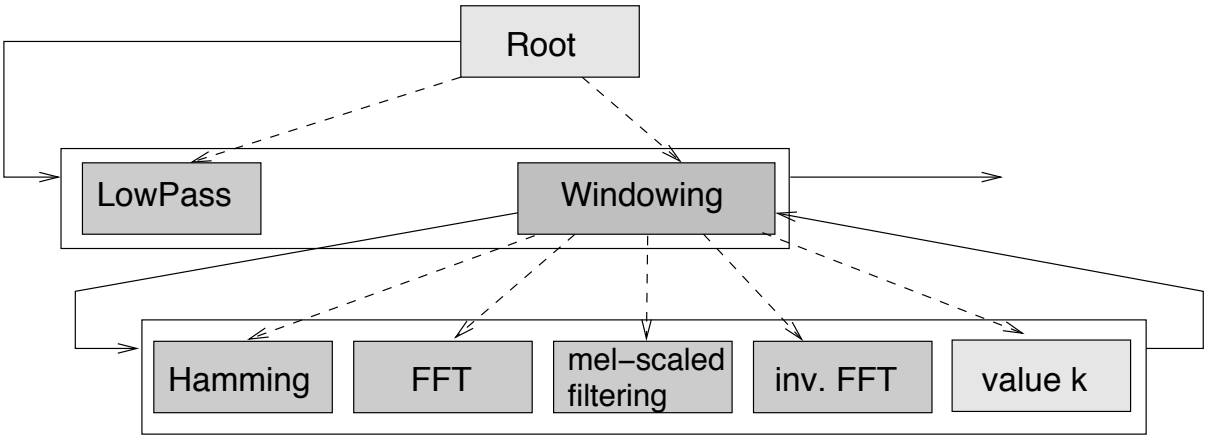
Phasenraum zur Klassifikation von Genre: Pop



Merkmalsraum weiter strukturieren

- Wir haben die Transformationen im Zeit-, Frequenz-, Phasenraum gesehen.
- Außerdem gibt es Filter und Annotationen von Segmenten.
- Das generalisierte Fenstern trennt die Funktion, die auf Messwerte in einem Fenster angewandt wird, von dem Fenstern selbst. Beim generalisierten Fenstern können beliebig viele beliebige Funktionen auf Werte in einem Fenster angewandt werden.
- Während bei allen vorigen Funktionen wieder eine Reihe zurückgegeben wird, liefert ein Funktional für eine Reihe nur einen Wert zurück.
- Aus diesen modularen Elementen können nun beliebige Merkmalsextraktionen zusammengestellt werden.

Methodenbaum zur Extraktion von MFCC



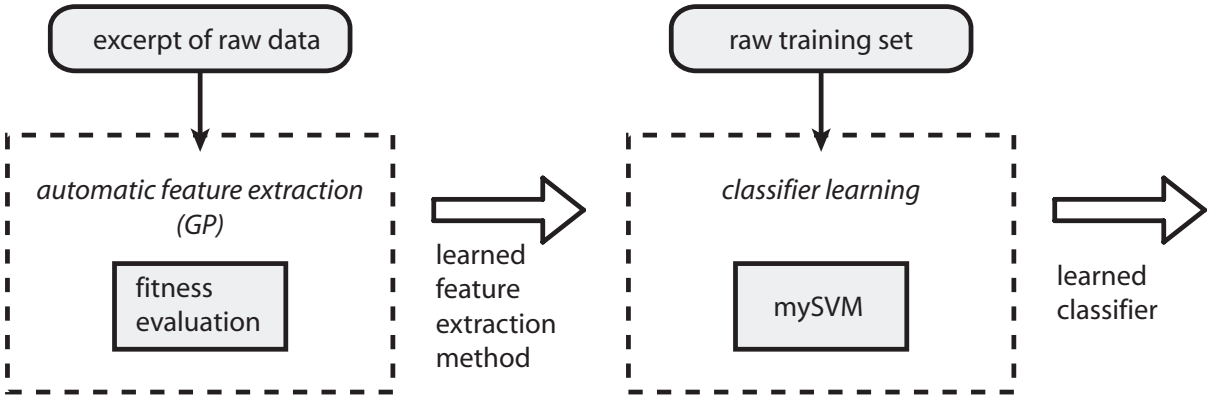
Überblick über den Lernprozess

	Classic/pop	Techno/pop	Hiphop/pop
Accuracy	100%	93.12%	82.50%
Precision	100%	94.80%	85.27%
Recall	100%	93.22%	79.41%
Error	0%	6.88%	17.50%

Tabelle 1: Klassifikation (lineare SVM) mit gelernten Merkmalen.

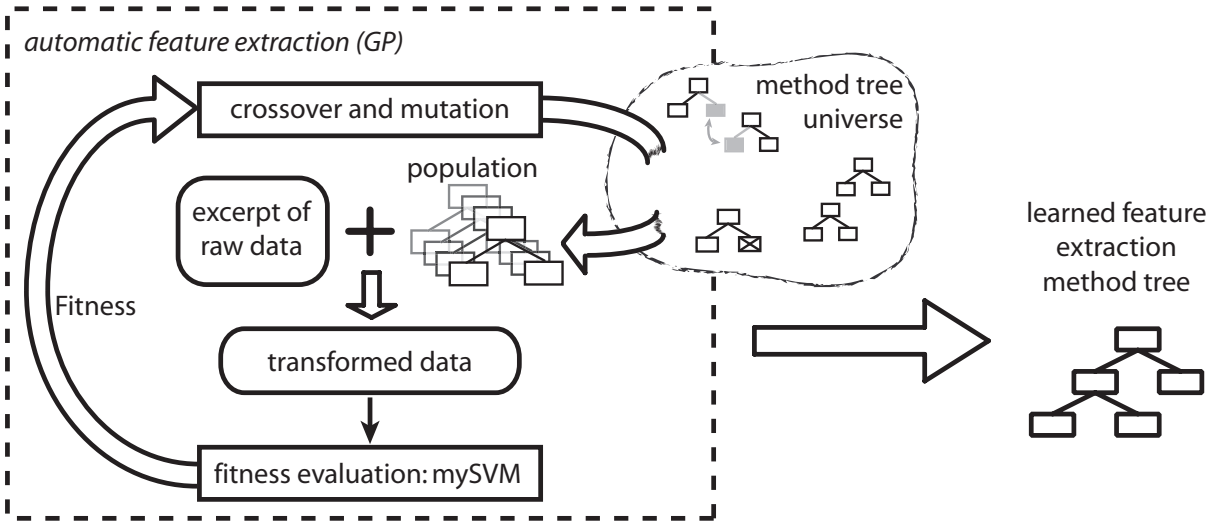
	Classic/pop	Techno/pop	Hiphop/pop
Accuracy	96.50%	64.38%	72.08%
Precision	94.12%	60.38%	70.41%
Recall	95.31%	64.00%	67.65%
Error	3.50%	35.63%	27.92%

Tabelle 2: Klassifikation mit dem selben Merkmalssatz für alle Aufgaben (lineare SVM).



Mierswa, Morik 2005

Lernen von Methodenbäumen mit genetischer Programmierung



Aufgabenspezifisches Lernen der Merkmale verbessert das Ergebnis

41 Merkmale wurden insgesamt gelernt.

	$\mathbf{User_1}$	$\mathbf{User_2}$	$\mathbf{User_3}$	$\mathbf{User_4}$
Accuracy	95.19%	92.14%	90.56%	84.55%
Precision	92.70%	98.33%	90.83%	85.87%
Recall	99.00%	84.67%	93.00%	83.74%
Error	4.81%	7.86%	9.44%	15.45%

Klassifikation nach Benutzerpräferenz

- 50 to 80 Stücke Lieblingsmusik
- Die selbe Anzahl negativer Beispiele.

Alles implementiert im Value-Series Plugin von RapidMiner.
Verwendbar für alle Wertereihen!

Eigenschaften lernender Merkmalsextraktion

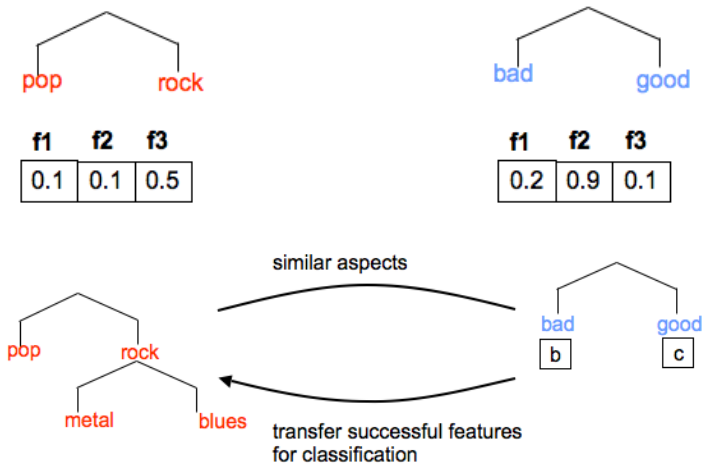
- Sehr gute Lernergebnisse
- Aufwand des Benutzers, die Beispielmengen zusammenzustellen → automatisch (aus Hörverhalten) extrahieren!
- Aufwand der genetischen Programmierung
- Merkmale werden aus einem Musikstück (Sample) extrahiert – funktioniert nicht inkrementell (online).

20.2 Merkmalsübertragung

Merkmalsübertragung

- Wenn das Trainieren der Merkmalsextraktion so lange dauert (1 Woche), sollte für ähnliche Lernaufgaben auch bereits gelernte Merkmalssätze verwendet werden (Mierswa/Wurst 2005, Wurst/Morik 2006).
- Charakterisierung einer Merkmalsmenge durch Gewichtung von Basismerkmalen.
- Feststellen der Eignung von Basismerkmalen für eine Klassifikationsaufgabe.
- Ähnliche Gewichte der Basismerkmale → ähnliche Lernaufgaben und Transfer des gesamten Merkmalssatzes.

Merkmalstransfer im Bild



Eignung von Merkmalen für eine Lernaufgabe

- Ein Merkmal X_{ik} ist **irrelevant** für eine Klassifikationsaufgabe t_i, wenn es nicht mit Y_i korreliert ist: $Pr(Y_i \mid X_{ik}) = Pr(Y_i)$. Die Menge irrelevanter Merkmale für t_i ist IF_i.
- Zwei Merkmale X_{ik} und X_{ir} heißen **alternativ** bzgl. einer Lernaufgabe t_i, $X_{ik} \sim X_{ir}$, gdw. $X_{ir} = a + b \cdot X_{ik}, b > 0$. Die Menge alternativer Merkmale für t_i ist AF_i.
- X_B sei eine Menge von Basismerkmalen.
- Die Merkmale sollen nun so gewichtet werden, wie es ihrer Eignung für die Lösung einer Lernaufgabe enspricht $w : X_B \to \mathcal{R}$.

Bedingungen für Merkmalsgewichtungen, die die Charakterisierung von Lernaufgaben erlauben

1. $w(X_{ik}) = 0$, wenn $X_{ik} \in X_B$ irrelevant ist. Irrelevante Merkmale sind mit 0 gewichtet.
2. Für $AF_i \subseteq X_B$ gilt: $\forall S \subset AF_i, S \neq \{\} : \sum_{X_i \in S} w(X_k) = \sum_{X_k \in AF_i} w(X_k) = \hat{w}$
   Die Gewichtsumme alternativer Merkmale ist unabhängig von der Anzahl alternativer Merkmale.
3. $X_{ik} \sim X_{ir} \Rightarrow w(X_{ik}) = w(X_{ir})$ Alternative Merkmale sind gleich gewichtet.
4. $\forall X_{ik} \in AF_i : X_{ir} \in IF_i \vee \exists X_{ir} \in X_B : X_{ik}X_{ik} \sim X_{ir} \Rightarrow \forall X_{ir} \in X_B :\not\exists X_{ik} \in AF_i : X_{ir} \sim X_{ik} \wedge w'(X_{ir}) = w(X_{ik})$ mit $w' : X_B \cup AF \to \mathcal{R}$. Eine Menge alternativer Merkmale ist nicht stärker gewichtet als ein einzelnes Merkmal.

Die Bedingungen gelten nicht immer!

- Alle Methoden der Merkmalsauswahl, die Merkmale binär gewichten, verletzen Bedingung 2 oder 3, sobald ein alternatives Merkmal hinzugefügt wird. $X'_B = X_B \cup \{X_{ir}\}, X_{ir} \sim X_{ik}, X_{ik} \in X_B \Rightarrow w'(X_{ir}) = w'(X_{ik}) = w(X_{il}) = 1$ weil ein ausgewähltes Merkmal in X_B Gewicht 1 hat; Verletzung 2. Bedingung: die Summe wäre 2! oder $w'(X_{ir}) \neq w(X_{ik})$ Verletzung 3. Bedingung (Alternativen sind gleichgewichtet).
- Jede Methode, die die Merkmale unabhängig voneinander gewichtet, verletzt Bedingung 2. Bei $X'_B = X_B \cup \{X_{ir}\}$ bleiben alle Gewichte für Merkmale in X_B gleich. Wenn $X_{ir} \sim X_{ik}, X_{ik} \in X_B$ verändert sich die Summe, so dass 2. Bedingung verletzt ist.

Die lineare SVM erfüllt alle Bedingungen

Die Merkmalsgewichtung durch die lineare SVM, $\vec{\beta}$, erfüllt alle Bedingungen.

- Bedingung 1: Die Euklidsche Länge von $\vec{\beta}$ soll minimiert werden, also werden möglichst Merkmale mit 0 gewichtet, wenn dadurch nicht der Fehler steigt. Also werden irrelevante Merkmale mit 0 gewichtet.
- Bedingung 2: Fügen wir einfach das selbe Merkmal mehrfach hinzu, so ergibt sich $(\beta_{i1} + ... + \beta_{im})\vec{x}$ in $\vec{\beta}\vec{x} + \beta_0$. Die optimale Hyperebene ändert sich nicht und die Summe der Gewichte bei allen anderen Merkmalen bleibt unverändert.
- Bedingung 3: Die Summe der alternativen Merkmale verteilt sich gleichmäßig auf die Alternativen.
- Bedingung 4: Folglich ist die Menge alternativer Merkmale nicht stärker gewichtet als ein einzelnes Merkmal.

Geeignete Abstandsmaße für die Gewichtung der Basismerkmale als Ähnlichkeit von Lernaufgaben

Das Abstandsmaß $d : T \times T \rightarrow \mathcal{R}^+$ soll erfüllen:

1. $d(\vec{t_1}, \vec{t_2}) = 0 \Leftrightarrow \vec{t_1} = \vec{t_2}$
2. $d(\vec{t_1}, \vec{t_2}) = d(\vec{t_2}, \vec{t_1})$
3. $d(\vec{t_1}, \vec{t_2}) = d(\vec{t_1}', \vec{t_2}'), \vec{t_1}' = \vec{t_1}, \vec{t_1}' \in X_B^2 \cup IF_1^2$ und $\vec{t_2}' = \vec{t_2}, \vec{t_2}' \in X_B^2 \cup IF_2^2$ gleiche Gewichtsvektoren behalten im erweiterten Bereich gleichen Abstand.
4. $d(\vec{t_1}, \vec{t_2}) = d(\vec{t_1}', \vec{t_2}'), \vec{t_1}' = \vec{t_1}, \vec{t_1}' \in X_B^2 \cup AF_1^2$ und $\vec{t_2}' = \vec{t_2}, \vec{t_2}' \in X_B^2 \cup AF_2^2$ gleiche Gewichtsvektoren behalten im erweiterten Bereich gleichen Abstand.

Die Bedingungen gelten nicht immer!

Bei Euklidschem Abstand wird Bedingung 5 nicht eingehalten, d.h. das Hinzufügen alternativer Merkmale verändert den Abstand.

- Das alternative Merkmal X_r wird X_B hinzugefügt und ist alternativ zu $X_k \in X_B$. Wenn die Bedingungen an die Merkmalsgewichtung eingehalten sind, gilt: $w'(X_{sk}) = w'(X_{sr}) = \frac{w(X_{sk})}{2} = \frac{w(X_{sr})}{2}$ für $s = 1, 2$
- Seien alle anderen Merkmalsabstände S, dann ist

$$\begin{aligned} d(\vec{t_1}', \vec{t_2}') = & \sqrt{S + 2(w'(X_{ik}) - w'(X_{jk}))^2} \\ = & \sqrt{S + 2(\frac{w(X_{ik})}{2} - \frac{w(X_{jk})}{2})^2} \\ = & \sqrt{S + \frac{1}{2}(w(X_{ik}) - w(X_{jk}))^2} \\ \neq & \sqrt{S + (w(X_{ik}) - w(X_{jk}))^2} \\ = & d(\vec{t_1}, \vec{t_2}) \end{aligned}$$

Manhattan Abstand hält alle Bedingungen ein

- Bedingungen 1 - 3 sind die einer Metrik.
- Bedingung 4: Wir fügen ein für beide Lernaufgaben $\vec{t_1}, \vec{t_2}$ irrelevantes Merkmal X_{k+1} hinzu. Wenn die Bedingung 4 an die Gewichtung eingehalten ist, gilt: $\mid w'(X_{1,k+1}) - w'(X_{2,k+1}) \mid = 0$. Also:

$$\begin{aligned} d(\vec{t_1}', \vec{t_2}') &= \sum_{r=1}^{k} \mid w'(X_{1,r}) - w'(X_{2,r}) \mid + 0 \\ &= d(\vec{t_1}, \vec{t_2}) \end{aligned}$$

Manhattan Fortsetzung

- Bedingung 5: Das alternative Merkmal X_{k+1} wird X_B hinzugefügt und ist alternativ zu $X_k \in X_B$. Wenn die Bedingungen an die Merkmalsgewichtung eingehalten sind, gilt: $w'(X_{s,k+1}) = w'(X_{s,k}) = \frac{w(X_{s,k+1})}{2} = \frac{w(X_{s,k})}{2}$ für $s = 1, 2$

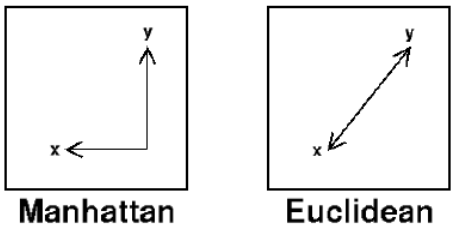
$$\begin{aligned} d(\vec{t_1}', \vec{t_2}') &= (\sum_{r=1}^{k-1} \mid w'(X_{1,r}) - w'(X_{2,r}) \mid) + \\ &\quad 2(\mid w'(X_{1,k+1}) - w'(X_{2,k+1}) \mid) \\ &= (\sum_{r=1}^{k-1} \mid w(X_{1,r}) - w(X_{2,r}) \mid) + \\ &\quad \mid w(X_{1,k}) - w(X_{2,k}) \mid \\ &= d(\vec{t_1}, \vec{t_2}) \end{aligned}$$

	Accuracy	Time	Optimization cycles
base features	0.79	-	-
optimal features	0.92	42s	3970
cbfc (k = 1)	0.85	3s	257
cbfc (k = 3)	0.88	5s	389
cbfc (k = 9)	0.89	8s	678

Tabelle 3: Durchschnittliche accuracy und Gesamtaufwand auf einem Testset von 11 Taxonomien für Lernen mit Basismerkmalen, optimierten Merkmalssätzen und Merkmalstransfer von den k ähnlichsten Lernaufgaben (cbfc).

Unterschied der Abstandsmaße Manhattan und Euklid

$d(x, y)$



Anwendung der Merkmalsübertragung

- Gegeben die 39 Taxonomien zur Musikorganisation. Je Knoten sei die Lernaufgabe, in die Unterknoten zu klassifizieren.
- Wir optimieren Musikmerkmale für jede Lernaufgabe.
- Als Basismerkmale werden 10 gewählt, die für die meisten Lernaufgaben erzeugt wurden.
- Anwendung der linearen SVM auf jede Lernaufgabe liefert $\vec{\beta}$ und damit auch eine Gewichtung der Basismerkmale. $O(\mid X_B \mid\mid T \mid N^3)$
- Gemäß der gewichteten Basismerkmale wird die Ähnlichkeit der Lernaufgaben festgestellt. $O(\mid X_B \mid\mid T \mid^2)$
- Bei ähnlichen Lernaufgaben wird der komplette Merkmalssatz transferiert.

Ergebnis des Merkmalsübertragung

Was wissen Sie jetzt?

- Merkmale können aus Basisfunktionen und -transformationen per Genetischer Programmierung hergestellt werden, wobei die Qualität des Lernergebnisses optimiert wird.
- Merkmale werden von Entscheidungsbaumlernern und der SVM sehr unerschiedlich behandelt. Wichtigster Unterschied ist die Behandlung irrelevanter oder alternativer Merkmale.
- Nur die SVM-Merkmalsgewichtung im Zusammenhang mit der Manhattan-Distanz ermöglicht, anhand der Gewichtung von Basismerkmalen die Ähnlichkeit von Lernaufgaben festzustellen.
- Antrainierte Merkmalssätze können auf ähnliche Lernaufgaben übertragen werden und liefern dann mit viel weniger Aufwand fast gleich gute Ergebnisse.

21 Subgruppenentdeckung

Lernaufgabe Subgruppenentdeckung

- Gegeben
  - X der Raum möglicher Beobachtungen mit einer Wahrscheinlichkeitsverteilung D,
  - $S \subseteq X$ eine gemäß D gezogene Stichprobe,
  - L_H der Raum möglicherweise gültiger Regeln, wobei jeder Regel $h \in L_H$ eine Extension zugeordnet ist: $ext(h) \subseteq X$ und
  - eine Qualitätsfunktion
$$q : L_H \to \mathcal{R}$$
- finde
  - eine Menge $H \subseteq L_H, \mid H \mid = k$
  - und es gibt keine $h' \in L_H \backslash H, h \in H$, für die gilt $q(h') \geq q(h)$

Beispiel der Subgruppenentdeckung

Es werden Gruppen beschrieben, die sich abweichend von der Gesamtpopulation verhalten. Es geht nicht notwendigerweise um Vorhersage, sondern um Beschreibung! Trotzdem ist meist eine Hypothese eine Abbildung $h : X \to Y$.

- Unter den alleinstehenden jungen Männern in ländlichen Regionen ist der Anteil an Lebensversicherungskinden signifikant niedriger als im gesamten Kundenbestand.
- Verheiratete Männer mit Pkws der Luxusklasse machen nur 2 Prozent der Kunden aus, erzeugen aber 14 Prozent der Lebensversicherungsabschlusssumme.

Ansätze zur Subgruppenentdeckung

- Aufzählend: vollständige Suche im strukturierten Raum L_H mit Pruning – Garantie, dass die k besten Regeln gefunden werden. Explora (Klösgen 1996), Midos (Wrobel 1997)
- Heuristisch: ein Entscheidungsbaumlerner wird so verändert, dass seine Qualitätsfunktion die der Subgruppenentdeckung wird und Beispiele ein veränderliches Gewicht erhalten – keinerlei Garantie. CN2-SD (Lavrac et al. 2004)
- *Probabilistisch*: Stichproben-bezogene Fehler werden während das Scans der Daten abgeschätzt – probabilistische Garantie, dass die k besten Regeln gefunden werden. (Scheffer, Wrobel 2002)

Modellselektion

- Die Menge H der gewählten Hypothesen kann auch als Modell betrachtet werden.
- Die Subgruppenentdeckung ist dann ein Problem der Modellselektion.
- Dabei geht es immer um Gütekriterien.
- Wir hatten ja schon:
  - Accuracy
  - Precision
  - Recall
  - Mittlerer quadratischer Fehler, quadratische Fehlersumme, erwarteter quadratischer Fehler, 0-1-Verlust
  - Maximum Likelihood
  - Entropie
  - Bayes Information Criterion
  - Minimum Description Length

21.1 Qualitätsfunktionen

Lift

- Für eine Regel $h = A \rightarrow Y$, wobei A eine Menge von Literalen ist und $Y = \{0, 1\}$ ist

$$Lift(A \rightarrow Y) = \frac{Pr[A, Y]}{Pr[Y]} = \frac{precision(A \rightarrow Y)}{Pr[Y]} \tag{66}$$

- Bei $Lift(A \rightarrow Y) = 1$ sind A und Y unabhängig.
- Bei $Lift(A \rightarrow Y) > 1$ steigt die bedingte Wahrscheinlichkeit für Y gegeben A.
- Bei $Lift(A \rightarrow Y) < 1$ sinkt die bedingte Wahrscheinlichkeit für Y gegeben A.
- Lift normalisiert die precision gegenüber einer verzerrten Verteilung der Klassen!

Coverage und Bias von Regeln

- Die Wahrscheinlichkeit, dass der Antezedens A der Regel auf ein Beispiel zutrifft bei einer Verteilung D der Beispiele ist:

$$Cov(A \rightarrow Y) = Pr[A]$$

- Die Differenz zwischen der bedingten Wahrscheinlichkeit von Y gegeben A und der a priori Wahrscheinlichkeit für Y ist der Bias:

$$Bias(A \rightarrow Y) = Pr[Y \mid A] - Pr[Y] = Pr[Y] \cdot (Lift(A \rightarrow Y) - 1)$$

Weighted relative accuracy WRAcc

- Man kann Bias und Coverage für eine Anwendung mit einem Parameter α geeignet gewichten.
  - Vielleicht will man auf jeden Fall alles abdecken, weil man alle Beispiele irgendwie behandeln muss. Dann gewichtet man Coverage hoch.
  - Vielleicht findet man nur Abweichungen von der a priori Wahrscheinlichkeit interessant. Dann gewichtet man Bias hoch.
  - Bei gleichgewichteten Coverage und Bias $\alpha = 0,5$ erhält man das selbe Ergebnis wie beim binominalen Test, der die Nullhypothese (A hat keinen Einfluss) testet.
- Für eine Regel h und eine Gewichtung $\alpha \in [0, 1]$ ist

$$WRAcc(\alpha, h) = Cov(h) \cdot Bias(h)$$

Wofür die Maße?

- Jetzt wissen wir, wie wir Regeln auswählen können.
- Wir wollen aber auch noch wissen, wie gut das Modell, also die gesamten Regeln, ist.
- Dann können wir die Regelmenge auswählen, die am besten ist.

Sensitivität und Spezifität – ROC

- Sensitivität (Recall): Wahrscheinlichkeit, dass ein positives Beispiel auch als positiv erkannt wird. (TP: true positives)
- Spezifizität: Wahrscheinlichkeit, dass ein negatives Beispiel auch als negativ erkannt wird. (TN: true negatives)
- Die *Receiver Operator Characteristic (ROC)* Kurve setzt Sensitivität und Spezifizität in Beziehung für verschiedene Parameter. Je nach Benutzerinteresse (TP wichtiger? TF wichtiger? Beides?) wird das Modell gewählt.

Beispiel

$y = 1$ für Spam, Fehler insgesamt 9%

	Predicted	
True	email	spam
email	57,3	4,0
spam	5,3	33,4

Sensitivität:

$$100 \cdot \frac{33,4}{33,4+5,3} = 86,3\%$$

Spezifizität:

$$100 \cdot \frac{57,3}{57,3+4,0} = 93,4\%$$

ROC im Bild

Ein Parameter wurde zwischen 0,1 und 10 variiert.

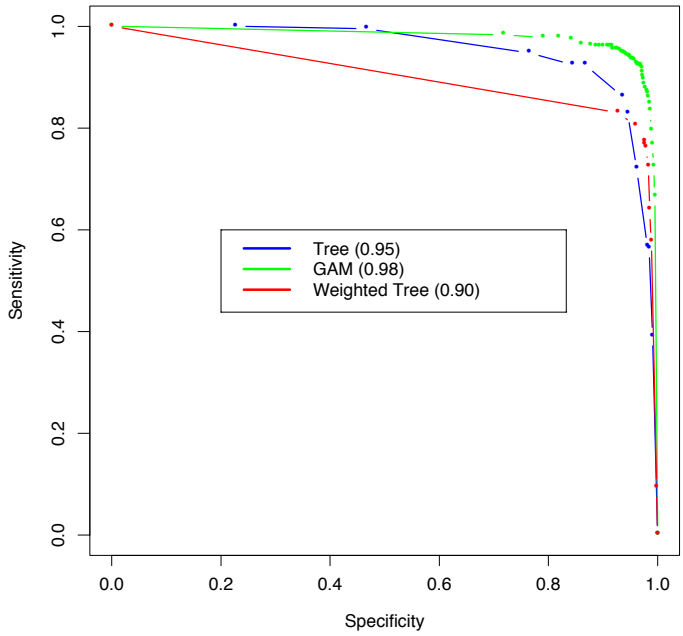


Figure 9.6: *ROC curves for the classification rules fit to the* `spam` *data. Curves that are closer to the north-east corner represent better classifiers. In this case the GAM classifier dominates the trees. The weighted tree achieves better sensitivity for higher specificity than the unweighted tree. The numbers in the legend represent the area under the curve.*

Area Under the Curve

Definition 21.1. *AUC Für $h : X \to Y, Y \in \{0,1\}$ und $D : X \times Y \to \mathcal{R}^+$ ist die Area Under the ROC Curve (AUC) die Wahrscheinlichkeit*

$$AUC(h) = Pr[h(\vec{x}) \geq_q h(\vec{x}') \mid y = 1, y' = 0]$$

dass ein zufällig gezogenes positives Beispiel höher bewertet wird gemäß einer Qualitätsfunktion q als ein zufällig gezogenes negatives Beispiel.

AUC ist invariant gegenüber monotonen Transformationen von h.

Idee eines Algorithmus', der AUC berechnet

- Schätze für jedes Beispiel in S die Wahrscheinlichkeit, ein positives zu sein.
- Ordne die Beispiele nach dieser Wahrscheinlichkeit (ranking).
- Bewerte ein Lernergebnis nach der Anzahl $\Lambda(h, S)$ der notwendigen Vertauschungen der Reihenfolge (des rankings).
- Sei S_+ die Menge der positiven Beispiele, S_- die Menge der negativen, dann ist

$$AUC(h, S) = \frac{\Lambda(h, S)}{\mid S_+ \mid \cdot \mid S_- \mid}$$

22 Sampling

Abhängigkeit des Lernergebnisses von S

- Eigentlich wollen wir ja ein optimales (oder wenigstens angenähert optimales) Lernergebnis auch für noch nicht gesehene Beispiele haben.
- Die ROC Kurve bezieht sich wie auch AUC nur auf die Stichprobe S.
- Meist sind die Datenmengen so groß, dass wir nur eine Stichprobe behandeln können.
- Wir wollen jetzt eine Stichprobe ziehen, die ungefähr so verteilt ist wie die Gesamtmenge.
- Leider haben wir keine Ahnung, was die wahre Verteilung ist!

i.i.d. erhaltende Stichprobe

- Die Daten insgesamt, $\mathbf{X}$, und die Stichprobe S sollen i.i.d. verteilt sein.
- Folgen von Zufallsvariablen, die sowohl unabhängig als auch identisch verteilt sind werden üblicherweise mit i.i.d. (für independent and identically distributed) bezeichnet.
  - Beispiel dreimaliges Würfeln:
    - X_1 1. Wurf, X_2 2. Wurf, X_3 3. Wurf sind i.i.d. verteilt.
    - $X_4 = X_1 + X_2$ und $X_5 = X_2 + X_3$ sind zwar identisch verteilt, aber nicht unabhängig.
    - X_4 und X_3 sind unabhängig, aber nicht identisch verteilt.
- Wenn die Daten in der Datenbank in zufälliger Reihenfolge gespeichert sind, ergibt das Ziehen der m ersten Daten eine i.i.d. erhaltende Stichprobe.

Ziehen der Stichprobe mit/ohne Zurücklegen

- Zufällig ein Beispiel ziehen ist Ziehen mit Zurücklegen. Dabei kann es Doppelte geben und damit eine Verzerrug (Bias). Die Wahrscheinlichkeit für Doppelte beim Ziehen von m Beispielen aus N ist:

$$p_m = \frac{N!}{(N-m)! \cdot N^m}$$

  Also sinkt die Wahrscheinlichkeit, keine Doppelten zu haben, $1 - p_m$, exponentiell mit Steigen von m.
- Zufällig ein Beispiel ziehen und es nicht Zurücklegen verfälscht nicht: jedes Beispiel hat die selbe Wahrscheinlichkeit, gezogen zu werden m/N. Leider ist dies aufwändig: man muss prüfen, ob ein Beispiel der Datenbank schon gezogen wurde, logarithmische Laufzeit.

Konfidenz

- Wir möchten gern wissen, bei wie vielen Beispielen wir wie sicher sein können, uns nicht zu verschätzen.
- Dazu nehmen wir einen Konfidenzwert δ und Schranken für die Wahrscheinlichkeit.
- Dann können wir nach und nach imer größere Stichproben ziehen, bis wir uns sicher genug sind. Und dann aufhören!

Chernoff-Schranke

- Sei p die Wahrscheinlichkeit, dass ein Beispiel gezogen wird, das von einer Regel h korrekt klassifiziert wird.
- Bei i.i.d. Stichproben ist p konstant für alle Regeln.
- Die Zufallsvariable X_i, mit $i = 1, ..., m$ sei 1 für die korrekte Klassifikation, 0 sonst.
- Der Erwartungswert für $\bar{Y} = 1/m \sum X_i$ ist gerade p: $E(\bar{X}) = p$
- Die Standardabweichung ist $\sigma(\bar{Y}) = \sqrt{\frac{p(1-p)}{m}}$
- Die *Chernoff-Schranke* sagt für beliebigen Parameter λ:

$$Pr[\bar{Y} \geq (1+\lambda)p] \leq exp(-\lambda^2 mp/3) \tag{67}$$
$$Pr[\bar{Y} \leq (1-\lambda)p] \leq exp(-\lambda^2 mp/2) \tag{68}$$

Chernoff-Schranke zur Abschätzung der geeigneten Stichprobengröße – Beispiel

- Wie wahrscheinlich ist es, dass Regeln mit der wahren Accuracy $Acc = p = 75\%$ bei einer Stichprobe der Größe m nicht besser als reiner Zufall abschneiden?
- Sei $\bar{Y} = \widehat{Acc}$ der Anteil korrekter Klassifikationen und der reine Zufall 50%. $\lambda = 1/3$, weil $(1 - 1/3) \cdot Acc = 50\%$.
- Wegen Gleichung (67) ergibt sich:

$$Pr[\widehat{Acc} \leq (1-1/3) \cdot Acc] \leq exp(-(1/3)^2 m \cdot Acc/2)$$
$$\Leftrightarrow Pr[\widehat{Acc} \leq 1/2] \leq exp(-1/9m3/8) = exp(-\frac{m}{24})$$

- Risiko $\leq \delta = 5\%$, dass bei $m \geq 72$ Beispielen ein 75% gutes h die Hälfte falsch klassifiziert:

$$exp(-\frac{m}{24}) \leq \delta \Leftrightarrow -\frac{m}{24} \leq ln\ \delta = -ln\frac{1}{\delta} \Leftrightarrow m \geq 24ln\frac{1}{\delta} = 24ln20$$

Hoeffding-Schranke

- Die *Hoeffding-Schranke* ist unabhängig von Acc definiert.

$$\begin{aligned} Pr[\bar{Y} - p \geq \epsilon] &\leq exp(-2\epsilon^2 m) \\ Pr[\bar{Y} - p \leq -\epsilon] &\leq exp(-2\epsilon^2 m) \\ Pr[\mid \bar{Y} - -p \mid \geq \epsilon] &\leq 2exp(-2\epsilon^2 m) \end{aligned} \tag{69}$$

- Die wahre Acc soll um nicht mehr als 10% über- oder unterschätzt werden. Wegen Gleichung (69) ergibt sich:

$$Pr[\mid \widehat{Acc} - Acc \mid \geq 0,1] \leq 2exp(-2 \cdot (0,1)^2 m) \leq 2exp(0,02m)$$

- Risiko $\leq \delta = 5\%$ dafür bei $m \sim 184$ Beispielen:

$$2exp(-0,02m) \leq 0,05 \Leftrightarrow -0,02m \leq ln\frac{1}{40}$$
$$\Leftrightarrow 0,02m \geq ln\ 40 \Leftrightarrow m \geq 50ln\ 40 \sim 184$$

Stichprobengröße für Subgruppenentdeckung

- Sei Konfidenzparameter $\delta \in [0,1]$ und höchster geduldeter Fehler $\epsilon \in \mathcal{R}^+$, es sollen die k besten Regeln $H \in L_H$ gemäß einer Qualitätsfunktion q so gelernt werden, dass mit einer Wahrscheinlichkeit von mindestens $1-\delta$ eine i.i.d. Stichprobe $\mid S \mid = m$ die wahre Qualität $\hat{q}$ höchstens um $E(m,\delta)$ verfälscht.
- In (Scheffer, Wrobel 2002) wird für die verschiedenen Qualitätskriterien aufgeführt, was $E(m,\delta)$ ist.
- Für *Acc* kann man die worst case Größenordnung der Stichprobe durch die Menge betrachteter Regeln L_H angeben:

$$m = O(\frac{1}{\epsilon^2} log \frac{\mid L_H \mid}{\delta})$$

Generic Sequential Sampling (Scheffer, Wrobel 2002)

- Durchgehen der Beispiele (scan) bis höchstens $m = O(\frac{1}{\epsilon^2} log \frac{|L_H|}{\delta})$ betrachtet wurden;
  1. Cov für positive und negative Beispiele bestimmen;
  2. Anordnen der Regeln nach dem Qualitätskriterium (ranking);
  3. Alle Regeln aussortieren aus dem Lernergebnis H, wenn sie häufiger als $\delta(2m \mid L_H \mid)$ falsch waren; die Wahrscheinlichkeit, eine gute Regel auszusortieren, ist dann höchstens $\delta/2$.
  4. Wenn $\mid H \mid \leq k$, wird H ausgegeben und die Regeln sind mit einer Wahrscheinlichkeit von mindestens $1-\delta$ bis auf eine Abweichung von höchstens ϵ optimal.

Stratifizierte Stichproben

Definition 22.1. *Stratifizierte Dichtefunktion Für $D : X \times Y \rightarrow \mathcal{R}^+$ ist die stratifizierte Dichtefunktion D' definiert als*

$$D'(x,y) = \frac{D(x,y)}{\mid Y \mid \cdot Pr[y = y']}$$

und falls wir klassifizieren mit $f : X \rightarrow Y$ als

$$D'(x,y) = \frac{D(x)}{\mid Y \mid \cdot Pr[f(x)]}$$

.

Es wird also die gegebene Verteilung D so geändert, dass die Verteilung der Klassen in D' gleich ist.

Ergebnis von Scholz 2005

- Wenn stratifizierte Stichproben gezogen, d.h. die Verteilung ensprechend geändert wird, entspricht die Subgruppenentdeckung mit der Qualitätsfunktion $WRAcc$ genau einer Klassifikation mit der Gütefunktion *Acc*.
- Man kann also in Ruhe die Lernalgorithmen für Klassifikation verwenden und braucht keine neuen zu erfinden.
- Allerdings muss man eine Stratifizierung, also Veränderung der Verteilung algorithmisch formulieren.
- Idee: Das tut man beim Ziehen von Stichproben.
- Folge: das Lernen auch aus großen Datenmengen geht schnell!

23 Knowledge Based Sampling

Knowledge-Based Sampling for Subgroup Discovery

- Wir wollen Vorwissen berücksichtigen, insbesondere nicht redundante Regelmengen H lernen. Dabei ist die Redundanz der Extension wichtig, nicht, dass sie durch verschiedene Merkmale ausgedrückt werden.
- Auch bereits gelernte Regeln $h \in H$ sind Vorwissen.
- Wir wollen wenig Beispiele bearbeiten müssen.
- Wir wollen vorhandene Algorithmen nutzen.
- Wir wollen diejenigen Subgruppen zurückliefern, die von der Allgemeinheit abweichen.
- Meist interessiert den Anwender die Extension einer solchen abweichenden Gruppe.

Martin Scholz *Scalable and Accurate Knowledge Discovery in Real-World Databases*, Dissertation am LS8, TU Dortmund, 2006

Ansatz: die Verteilung verändern

Die neue Verteilung D' soll nichts Wesentliches verändern:

$$Pr_{D'}[x \mid A, Y] = Pr_D[x \mid A, Y] \tag{70}$$
$$Pr_{D'}[x \mid A, \neg Y] = Pr_D[x \mid A, \neg Y] \tag{71}$$
$$Pr_{D'}[x \mid \neg A, Y] = [x \mid \neg A, Y] \tag{72}$$
$$Pr_{D'}[x \mid \neg A, \neg Y] = [x \mid \neg A, \neg Y] \tag{73}$$

Die Beschränkungen (70 – 73) bestimmen die neue Verteilung $D' : X \to \mathcal{R}^+$ eindeutig:

$$Pr_{D'}(x) = Pr_D(x) \cdot (Lift_D(h, x))^{-1} \tag{74}$$

$Lift(h, x)$

Der Lift eines Beispiels $x \in X$ ist für eine Regel $A \to Y$:

$$Lift(A \to Y, x) = \begin{cases} Lift(A \to Y), falls & x \in ext(A) \cap ext(Y) \\ Lift(A \to \neg Y), falls & x \in ext(A) \cap ext(\neg Y) \\ Lift(\neg A \to Y), falls & x \in ext(\neg A) \cap ext(Y) \\ Lift(\neg A \to \neg Y), falls & x \in ext(\neg A) \cap ext(\neg Y) \end{cases} \tag{75}$$

Lift drückt genau aus, wie weit eine Gruppe A von der allgemeinen Verteilung von Y abweicht.

Knowledge-Based Sampling für Subgruppenentdeckung

Gegeben $X = \{(x_1, y_1), ..., (x_N, y_N)\}$ und k, finde eine Menge $H = \{h_1, ..., h_k\}$

1. Stelle die a priori Verteilung $\pi(y)$ für jedes $y \in Y$ fest.
2. Stratifizieren der Verteilung: $D_1(x_i) = \pi(y_i)^{-1}$ für $i = 1, ..., N$
3. für $t = 1$ bis k do
   - $h_t = RegelLernen(D_t, X)$
   - Kontingenzmatrix für h_t mit Gewichten gemäß D_t
   - Lift-Bewertung für h_t gemäß der Kontingenzmatrix
   - $D_{t+1}(x_i) = D_t(x_i) \cdot (Lift_{Dt}(h_t, x))^{-1}$ für $i \in \{1, ..., N\}$
4. Ausgabe $\{h_1, ..., h_k\}$ mit $Lift(h_i)$ (Definition 66)

Subgruppen für die Vorhersage

- Die Regeln können mit ihrer Gewichtung zu einem Ensemble zusammengefasst werden.
- LiftRatio LR:

$$LR(A \to Y, x) = \begin{cases} \frac{Lift(A \to Y)}{Lift(A \to \neg Y)}, falls & x \in ext(A) \\ \frac{Lift(\neg A \to Y)}{Lift(\neg A \to \neg Y)}, falls & x \in ext(\neg A) \end{cases} \tag{76}$$

- Für alle Regeln, wobei D_0 die uniforme Verteilung über X ist:

$$\hat{\beta}(x) = \frac{Pr_{D0}[Y]}{Pr_{D0}[\neg Y]} \cdot \prod_{1 \leq i \leq k} LR_{Di}[(A^i \to Y), x] \tag{77}$$

Was wissen Sie jetzt?

- Sie haben eine neue Lernaufgabe kennengelernt: Subgruppenendeckung.
- Wie bisher bei (fast) jeder Lernaufgabe, ging es gleich um Modellselektion. Hier für eine Menge von Hypothesen (Regeln), nicht eine Funktion.
- Sie haben neue Gütekritereien kennengelernt: Lift, WRAcc, Spezifizität und Sensitivität
- Für eine Reihe von Experimenten haben Sie ROC und AUC kennengelernt.
- Die Größe von Stichproben in Bezug auf das Risiko, dass das Lernergebnis falsch ist, wurde mt Chernoff und Hoeffding beschränkt.
- Zwei effiziente Ansätze zur Subgruppenentdeckung, von Wrobel und von Scholz, beruhen darauf, dass man nicht alle Beispiele zu betrachten braucht.
- Sie kennen Knowledge-Based Sampling für Subgruppenentdeckung und wie man das Ergebnis für die Klassifikation verwenden kann.