

Computational Learning Theory

Sommersemester 2017

Prof. Dr. Georg Schnitger
AG Theoretische Informatik

Goethe-Universität Frankfurt am Main

Herzlich willkommen!!!

Einführung

(a) Das PAC-Modell

- ▶ **P**robably **A**pproximately **C**orrect (Supervised) Learning.
 - ★ eine unbekannte Verteilung D erzeugt klassifizierte Beispiele (also überwachtes Lernen),
 - ★ der Lernalgorithmus wird auf D evaluiert.

(b) Die wesentlichen Fragen

- ▶ **Beispielkomplexität**
 - ★ Wie viele Beispiele sind für ein erfolgreiches Lernen hinreichend und notwendig?
 - ★ Die Vapnik-Chervonenkis Dimension.
- ▶ **Algorithmische Komplexität**
 - ★ Für welche Probleme können Hypothesen effizient bestimmt werden?

(a) Gegenbeispiel-Komplexität

- ▶ Ein (böser) Lehrer legt einem Schüler ein Beispiel vor.
- ▶ Der Schüler klassifiziert und erhält danach die richtige Antwort.
- ▶ Wiederhole, solange bis alle Antworten richtig sind.

(b) Expertenauswahl

- ▶ **Weighted-Majority**: Experten geben Online Empfehlungen ab. Können wir, **ohne den Sachverhalt zu kennen**, eine Strategie entwickeln, die fast so gut ist wie der im Nachhinein beste Experte?
- ▶ **Winnow**:
Finde wenige entscheidende aus vielen möglichen Gründen.
- ▶ **Perzeptron**:
Trenne rote von blauen Vektoren im \mathbb{R}^n durch eine Hyperebene.

(a) Grundlegende Methoden

- ▶ Validierung
- ▶ Boosting/Bagging
- ▶ Nicht-binäres Lernen
- ▶ stochastischer Gradientenabstieg

(b) Lernalgorithmen:

- ▶ Nearest Neighbor
- ▶ **Support Vektor Maschinen**
- ▶ **Neuronale Netzwerke**
- ▶ Maximum Likelihood, Expectation-Maximization.

(c) **Aktives** Lernen: Wie stark verbessert sich die Lernleistung, wenn der Schüler Fragen stellen darf?

- **Diskrete Modellierung:**

- ▶ Beweisführung: Direkte und indirekte Beweise, vollständige Induktion.

- **Datenstrukturen und Theoretische Informatik 1:**

- ▶ Laufzeitanalyse: O , o , Ω , ω and Rekursionsgleichungen,
- ▶ Graphalgorithmen,
- ▶ NP-Vollständigkeit,
- ▶ Berechenbarkeit.

- **Mathematik 3:**

- ▶ Grundlagen der Wahrscheinlichkeitstheorie
- ▶ Siehe auch Kapitel 2.4 im Skript.

- (1) S. Shalev-Shwartz und Shai Ben-David, Machine Learning: From Theory to Algorithms, Cambridge University Press 2014. (*PAC Lernen und Support-Vektor-Maschinen*)
- (2) M. Mohri, A. Rostamizadeh und A. Talwalkar, Foundations of Machine Learning, MIT Press 2012. (*PAC Lernen und Support-Vektor-Maschinen*)
- (3) M. J. Kearns und U. V. Vazirani, An Introduction to Computational Learning Theory, MIT Press 1994. (*PAC Lernen*)
- (4) M. H. G. Anthony und N. Biggs, Computational Learning Theory, Cambridge University Press 1997. (*PAC Lernen*)
- (5) I. Goodfellow, Y. Bengio und A. Courville, Deep Learning, MIT Press 2016. (*Lernverfahren, insbesondere Neuronale Netzwerke*)
- (6) Skript zur Vorlesung „Computational Learning Theory“, Goethe-Universität Frankfurt, 2017.

Organisatorisches

<http://thi.cs.uni-frankfurt.de/lehre/clt/sose17.de>

Die Webseite enthält alle wichtigen Informationen zur Veranstaltung:

- Alle Vorlesungsmaterialien (**Skript**, **Folien**, Zugang zu **Extra-Materialien**) finden Sie auf dieser Seite.
- Auch organisatorische Details zum **Übungsbetrieb** werden beschrieben.
- Unter **Aktuelles** finden Sie zum Beispiel:
 - ▶ Anmerkungen zum Übungsbetrieb
 - ▶ und gegebenenfalls Anmerkungen zu aktuellen Übungsaufgaben.
- Im **Logbuch** finden Sie Informationen,
 - ▶ Beamer-Folien, weitere Referenzen zu den einzelnen Vorlesungsstunden.

BITTE, BITTE, BITTE aktiv an den Übungen teilnehmen!

- Wöchentliche Übungszettel auf der Webseite mit **Ausgabe am Montag**. Der 1. Übungszettel wird in der 2. Vorlesungswoche ausgegeben.
- **Rückgabe**, nach 1-wöchiger Bearbeitungszeit, **vor der Montag-Vorlesung**. (Rückgabe auch im Briefkasten neben Büro 312 möglich.)
- Wenn die Prüfung bestanden ist:
 - ▶ Bei mindestens 50% aller Übungspunkte eine Verbesserung um einen Notenschnitt,
 - ▶ bei mindestens 70% eine Verbesserung um zwei Notenschritte.
- Bei nicht zu vielen Teilnehmern werden nur mündliche Prüfungen angeboten, sonst eine Klausur.

BITTE, BITTE, BITTE

- 1 Bitte helfen Sie mir durch
 - ▶ Fragen,
 - ▶ Kommentare
 - ▶ und Antworten!
- 2 Die Vorlesung kann nur durch **Interaktion** interessant werden.
 - ▶ Ich muss wissen, wo der Schuh drückt.
- 3 Sie erreichen mich außerhalb der Vorlesung im Büro 303.
 - ▶ Sprechstunde: Dienstags 10-12.
 - ▶ Kommen Sie vorbei.

- „**Neuer Master Informatik**“ oder „**Master Wirtschaftsinformatik**“:
 - ▶ Computational Learning Theory: Grundlagen (5 CP).
 - ★ Die Veranstaltung findet vom 18.04 bis zum 31.05 statt.
 - ▶ Computational Learning Theory: Weiterführende Themen (5 CP)
 - ★ Die Veranstaltung findet vom 06.06 bis zum 19.07 statt.
 - ▶ Als Veranstaltung vom 18.4 bis zum 19.07 (10 CP).
- „**Alter Master Informatik**“ oder „**Master Bioinformatik**“:
Computational Learning Theory (4+2 SWS, 10 CP).

Worum geht's?

Supervised Learning: Die zentralen Begriffe

Sei X eine nicht notwendigerweise endliche Menge.

- (a) Wir nennen X den **Beispielraum**.
- (b) Ein **Konzept** c (über X) ist eine Teilmenge von X .
- (c) Eine **Hypothese** h (über X) ist ein Konzept (über X) und eine **Hypothesenklasse** \mathcal{H} ist eine Menge von Hypothesen.
- (d) Für ein Konzept c ist $x \in X$ ein **positives** Beispiel für c , falls $x \in c$ und ansonsten ein **negatives** Beispiel. Insbesondere ist

$$\{ (x, b) \in X \times \{0, 1\} : x \in c \iff b = 1 \}$$

die Menge der gemäß c klassifizierten Beispiele.

Die Zielstellung

c ist das unbekannte Konzept.

Der Lernalgorithmus erhält gemäß klassifizierte Beispiele.
Eine Hypothese ist zu bestimmen, die c möglichst gut *erklärt*.

Welche Fragen sind zu klären?

- (1) **Was ist Lernen?** Wann erklärt die Hypothese h das Konzept c ?
Wann können wir behaupten, dass unsere Hypothese gut ist?
- (2) **Algorithmische Komplexität:** Kann eine gute Hypothese bei genügend vielen Beispielen in vertretbarer Zeit gefunden werden?
- (3) **Beispielkomplexität:**
Wieviele Beispiele werden für ein erfolgreiches Lernen benötigt?

- (1) **Handschriftenerkennung:** Für jeden Anwender x und jede Ziffer $z \in \{0, 1, \dots, 9\}$ besteht das Konzept $c_{x,z}$ aus allen Schriftproben des Anwenders x , die der Ziffer z entsprechen.
- (2) **Textinterpretation:** Nachrichten sind zum Beispiel automatisch einer der Kategorien „Politik, Wirtschaft, Kultur, Sport ...“ zuzuordnen. Jede dieser Kategorien entspricht einem Konzept.
- (3) **Bilderkennung:** Eine Pixelmatrix ist einer Kategorie (wie etwa „Sonnenaufgang, Wüste, Meer, Stadt, ...“) zuzuordnen.
- (4) **Bestimmung der Funktion von Proteinen:**
Sage die Funktion eines Proteins voraus. Proteinfamilien mit ähnlicher Funktion entsprechen Konzepten.

Welche Konzeptklassen sind von besonderem Interesse?

Die Konzeptklasse der monotonen Monome

Jedes Beispiel

$$x \in X := \{0, 1\}^n$$

wird als ein Vektor von n Eigenschaften (oder Features) repräsentiert.

- Die unbekannte Konjunktion $\alpha := x_{i_1} \wedge \dots \wedge x_{i_k}$ sei zu lernen. Das entsprechende Zielkonzept c_α ist die Menge

$$c_\alpha = \{x \in \{0, 1\}^n : x \text{ erfüllt } \alpha\}.$$

aller positiven Beispiele.

- Die Konzeptklasse **MONOTON-MONOM** $_n$ besteht aus der Menge aller Konzepte c_α für eine monotone Konjunktion α .

- Und wenn allgemeine Monome zu lernen sind?

Jetzt entsprechen die unbekanntenen Zielkonzepte der Konzeptklasse **MONOM**_{*n*} aller Monome mit den Literalen $x_1, \neg x_1, \dots, x_n, \neg x_n$.

- Darf es ein wenig mehr sein?

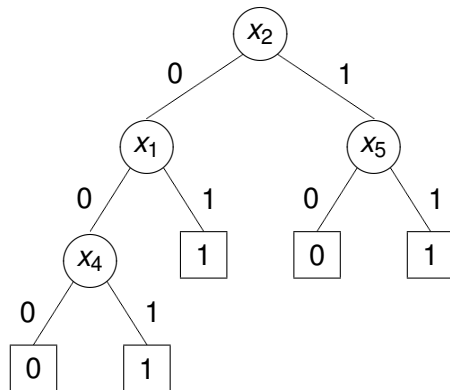
***k*-TERM DNF**_{*n*} ist die Konzeptklasse aller DNF-Formeln mit den Literalen $x_1, \neg x_1, \dots, x_n, \neg x_n$ und höchstens *k* Monomen.

- Und vielleicht noch ein wenig(?) mehr:

BOOLEAN_{*n*} ist die Konzeptklasse aller booleschen Funktionen $f : \{0, 1\}^n \rightarrow \{0, 1\}$.

Entscheidungsbäume

Die Konzeptklasse **DECISION** _{n,s} hat für jeden Entscheidungsbaum B mit höchstens s Knoten (über den Variablen x_1, \dots, x_n) ein Konzept c_B .



AUTOMAT $_{n,\Sigma}$ besitzt für jeden DFA A mit höchstens n Zuständen ein Konzept, nämlich die von A akzeptierte Sprache $L(A)$.

A muss das Eingabealphabet Σ besitzen.

HALBRAUM $_n$ besitzt für alle Gewichte $w_1, \dots, w_n \in \mathbb{R}$ und jeden Schwellenwert (bzw. Threshold) $t \in \mathbb{R}$ das Konzept

$$\{ x \in \mathbb{R}^n \mid \sum_{i=1}^n w_i \cdot x_i \geq t \}$$

HALBRAUM ist unsere erste reellwertige Konzeptklasse.

- Die Konzeptklasse der **neuronalen Netzwerke** besteht aus Schaltungen mit „neuronalen Gattern“ der Form $g(\sum_{i=1}^n w_i \cdot y_i - t)$.
 - ▶ Zum Beispiel werden das Standard-Sigmoid $g(x) = \frac{1}{1+e^{-x}}$
 - ▶ rectified linear units $g(z) = \max\{0, z\}$
 - ▶ oder die binäre Threshold-Funktion $g(x) = \begin{cases} 0 & x < 0, \\ 1 & \text{sonst} \end{cases}$ verwandt.
- In der Methode der **Support Vektor Maschinen** wird eine **Feature-Funktion**

$$\phi : X \rightarrow \mathbb{R}^n$$

auf die Beispiele x_1, \dots, x_s angewandt. Dann wird nach einer Hyperebene gesucht, die – **so weit es geht** – alle positiven Beispiele $\phi(x_i)$ von den negativen Beispielen $\phi(x_j)$ trennt.

- (1) Wieviele Beispiele werden benötigt, um eine Chance zu haben die einzelnen Konzeptklassen erfolgreich zu erlernen?
 - Für **MONOM**_n sollten sehr viel weniger Beispiele ausreichen als für **BOOLEAN**_n.
 - Die Beispielzahl sollte von der Anzahl der „**Freiheitsgrade**“ der Hypothesenklasse abhängen.
 - Wie misst man Freiheitsgrade?
- (2) Können wir tatsächlich lernen, wenn genügend viele Beispiele vorhanden sind?
- (3) Müssen wir ewig herum experimentieren oder können wir sagen, ob wir zumindest **hoch-wahrscheinlich** erfolgreich lernen werden?

PAC-Algorithmen

- Klassifizierte Beispiele werden zufällig und unabhängig voneinander, gemäß einer unbekanntem Verteilung D_1 , gezogen.
- Der Lernalgorithmus \mathcal{L} berechnet eine Hypothese h .
- Wie gut ist h ? Wir messen den Fehler zwischen dem unbekanntem Zielkonzept c und der Hypothese h gemäß einer Verteilung D_2 :

$$\text{fehler}_{D_2}(c, h) = \text{prob}[c \oplus h]$$

(Später betrachten wir allgemeinere Fehlermodelle.)

Fairness-Bedingung: Wenn Beispiele nach der Verteilung D_1 erzeugt werden, dann sollte der Fehler auch gemäß D_1 gemessen werden. Wir fordern deshalb

$$D_1 = D_2.$$

Der Anwender gibt einen **Fehlerparameter** ϵ vor.

- Der Fehler, also der Unterschied zwischen Zielkonzept und Hypothese, soll höchstens ϵ betragen.
- Aber wenn der Lernalgorithmus *zufälligerweise* wenig informative Beispiele erhält, dann wird die Hypothese i. A. nicht gut sein.

Der Anwender gibt einen **Misstrauensparameter** δ vor und fordert:

*Der Fehler darf mit **Wahrscheinlichkeit** mindestens $1 - \delta$ den Fehlerparameter ϵ nicht übersteigen.*

PAC Lernen = probably (mit Wahrscheinlichkeit mindestens $1 - \delta$)
approximately (mit Fehler höchstens ε) correct.

Wir möchten also hoch-wahrscheinlich approximativ korrekt lernen.

Ein Lernalgorithmus A ist genau dann ein **PAC-Algorithmus** für eine Konzeptklasse \mathcal{C} , wenn **für alle** $\varepsilon \in (0, 1]$, **für alle** $\delta \in (0, 1]$, **für alle Konzepte** $c \in \mathcal{C}$ und **für alle Verteilungen** D

$$\text{prob}_D[\text{fehler}_D(c, h_c) \leq \varepsilon] \geq 1 - \delta$$

gilt, wobei h_c die von A bestimmte Hypothese ist.

Die Grundstruktur eines PAC-Algorithmus

1. Die Eingabe besteht aus dem *Misstrauensparameter* δ und dem *Fehlerparameter* ε . Die Hypothesenklasse \mathcal{H} sei vorgegeben:

Ein unbekanntes Konzept c ist mit Hilfe einer Hypothese $h_c \in \mathcal{H}$ zu erlernen.

2. Fordere eine Trainingsmenge $S = \{ (x_1, b_1), \dots, (x_s, b_s) \}$ von $s = s(\delta, \varepsilon)$ klassifizierten Beispielen an.

/ Es gilt $x_i \in c \iff b_i = 1$ für $1 \leq i \leq s$.*

**/*

3. Bestimme eine Hypothese $h_c \in \mathcal{H}$.

- Fordern wir zuviel?

In Anwendungen achtet man darauf, dass die Trainingsmenge aus möglichst informativen Beispielen besteht:

Die Beispielverteilung D_1 ist dann sogar unterstützend.

- Andererseits wäre die Existenz von PAC-Algorithmen A klasse, denn wir können A mit einem Gütesiegel versehen:

Für jede Verteilung D , falls A mindestens $s(\varepsilon, \delta)$ Beispiele erhält, ist approximativ korrektes Lernen hoch-wahrscheinlich.

- Wie groß ist die Beispielzahl $s(\varepsilon, \delta)$ in Abhängigkeit vom erlaubten Fehler ε und vom Vertrauensparameter δ ?
- Angenommen, A ist ein PAC-Algorithmus mit Fehler ε and Vertrauensparameter δ .
 - ▶ Um wieviel steigt die Beispielzahl und Laufzeit an, wenn δ durch δ' mit $0 < \delta' \leq \delta$ ersetzt wird?
 - ▶ Um wieviel steigt die Beispielzahl und Laufzeit an, wenn ε durch ε' mit $0 < \varepsilon' \leq \varepsilon$ ersetzt wird?

Konsistente Hypothesen

- (1) Bestimme $s = s_n(\varepsilon, \delta)$ und fordere s Beispiele an.
- (2) Die erste vorläufige Hypothese ist das Monom

$$h \equiv x_1 \wedge \neg x_1 \wedge x_2 \wedge \neg x_2 \wedge \cdots \wedge x_n \wedge \neg x_n.$$

- (3) Streiche jedes Literal, das irgendeinem positiven Beispiel widerspricht.
- (4) Gib die resultierende Hypothese aus.

h ist mit allen Beispielen konsistent

- (1) Das Literal x_j (bzw. $\neg x_j$) wird entfernt, falls ein positives Beispiel diesem Literal widerspricht.
- (2) Wir bestimmen somit das längste mit allen positiven Beispielen konsistente Monom h .
- (3) Ist y ein negatives Beispiel und ist M das unbekannte Zielmonom, dann
 - ▶ gibt es i , so dass das Literal x_i (bzw. $\neg x_i$) in M vorkommt,
 - ▶ und x_i (bzw. $\neg x_i$) verwirft das negative Beispiel y .

Da h das längste mit allen positiven Beispielen konsistente Monom ist, kommt x_i (bzw. $\neg x_i$) in h vor und h verwirft y .

Wir zeigen jetzt, dass unser Algorithmus ein PAC-Algorithmus für die Konzeptklasse MONOM_n ist, falls

$$s_n(\varepsilon, \delta) = \left\lceil \frac{1}{\varepsilon} \cdot \ln \left(\frac{1}{\delta} \right) + \frac{\ln(3^n)}{\varepsilon} \right\rceil.$$

PAC-Algorithmen für endliche Konzeptklassen

Konzepte einer Klasse \mathcal{C} von endlich vielen Konzepten sind zu lernen. Vertrauensparameter δ und Fehlerparameter ε sind vorgegeben.

- (1) Setze $s = \lceil \frac{1}{\varepsilon} \cdot (\ln |\mathcal{C}| + \ln(\frac{1}{\delta})) \rceil$.
- (2) Wähle **irgendein** Konzept $h \in \mathcal{C}$ als Hypothese, das mit allen s Beispielen **konsistent** ist.
/* Eine Hypothese h ist genau dann konsistent, wenn h alle positiven Beispiele akzeptiert und alle negativen Beispiele verwirft.

Für MONOM_n genügen, wie versprochen, $\lceil \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}) + \frac{\ln(3^n)}{\varepsilon} \rceil$ Beispiele.

Behauptung: Wir erhalten einen PAC-Algorithmus, der $s = \lceil \frac{1}{\epsilon} \cdot (\ln |\mathcal{C}| + \ln(\frac{1}{\delta})) \rceil$ Beispiele anfordert.

- Wir müssen für jedes Konzept $c \in \mathcal{C}$ und jede Verteilung D

$$\text{prob}_D[\text{fehler}_D(c, h_c) \leq \epsilon] \geq 1 - \delta$$

nachweisen.

- Sei $p = \text{prob}_D[\text{fehler}_D(c, h_c) > \epsilon]$. Dann ist

$$\begin{aligned}
 p &\leq \text{prob} \left[\begin{array}{c} \text{Es gibt eine konsistente Hypothese } h \text{ mit} \\ \text{fehler}_D(c, h) > \epsilon \end{array} \right] \\
 &\leq \sum_{h \in \mathcal{C} \text{ mit } \text{fehler}_D(c, h) > \epsilon} \text{prob}[h \text{ ist konsistente Hypothese}].
 \end{aligned}$$

- Wenn $h \in \mathcal{C}$ eine Hypothese mit großem Fehler ist (also $\text{fehler}_D(\mathbf{c}, h) > \varepsilon$), dann wird h ein zufällig gezogenes Beispiel mit Wahrscheinlichkeit höchstens $1 - \varepsilon$ richtig klassifizieren.
- Also ist

$$\begin{aligned}
 p &\leq \sum_{h \in \mathcal{C} \text{ mit } \text{fehler}_D(\mathbf{c}, h) > \varepsilon} \text{prob}[h \text{ ist eine konsistente Hypothese}] \\
 &\leq \sum_{h \in \mathcal{C} \text{ mit } \text{fehler}_D(\mathbf{c}, h) > \varepsilon} (1 - \varepsilon)^s \leq \sum_{h \in \mathcal{C}} (1 - \varepsilon)^s \\
 &= |\mathcal{C}| \cdot (1 - \varepsilon)^s.
 \end{aligned}$$

- Es ist $p = \text{prob}_D[\text{fehler}_D(\mathbf{c}, h_c) > \varepsilon]$. Wir müssen $p \leq \delta$ fordern!

- Wir müssen

$$|C| \cdot (1 - \varepsilon)^s \leq \delta$$

erfüllen.

- Logarithmiere beide Seiten der Ungleichung:

$$\ln |C| + \underbrace{s \cdot \ln(1 - \varepsilon)}_{\leq -\varepsilon} \leq \ln \delta.$$

(Es ist $\ln(1 - \varepsilon) \leq -\varepsilon$. Warum? )

- Somit reicht der Nachweis von $\ln |C| - s \cdot \varepsilon \leq \ln \delta$. Diese Ungleichung ist äquivalent zu

$$\frac{1}{\varepsilon} (\ln |C| - \ln \delta) \leq s$$

und das war zu zeigen.

Weitere Anwendungen

- (a) Beispiel(MONOTON-MONOM_n) = $O(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$,
Beispiel(MONOM_n) = $O(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (b) Beispiel(k -KNF_n) = $O(\frac{n^k}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$ für festes k ,
Beispiel(k -DNF_n) = $O(\frac{n^k}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$ für festes k .
- (c) Beispiel(k -KLAUSEL-KNF_n) = $O(\frac{k \cdot n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$,
Beispiel(k -TERM-DNF_n) = $O(\frac{k \cdot n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (d) Beispiel(KNF_n) = $O(\frac{1}{\varepsilon} \cdot 2^n + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$,
Beispiel(DNF_n) = $O(\frac{1}{\varepsilon} \cdot 2^n + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$,
Beispiel(BOOLEAN_n) = $O(\frac{1}{\varepsilon} \cdot 2^n + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (e) Beispiel(SYM_n) = $O(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (f) Beispiel(AUTOMAT_{n,Σ}) = $O(\frac{n}{\varepsilon} \cdot |\Sigma| \cdot \log_2 n + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.

Die Beispielzahl $s = \lceil \frac{1}{\varepsilon} \cdot (\ln |\mathcal{C}| + \ln(\frac{1}{\delta})) \rceil$.

- Eine große Verlässlichkeit, also ein kleines δ , kann **billig** erzielt werden, denn die Beispielanzahl wächst nur **logarithmisch** mit $1/\delta$.
- Ein kleiner Fehler, also ein kleines ε , ist **teuer**, denn $1/\varepsilon$ geht **linear** in der Beispielanzahl ein.

- Könnte

$$\ln |\mathcal{C}|$$

eine Definition der Anzahl der Freiheitsgrade sein?

- ▶ Nur wenn die Schranke für s scharf ist.
- ▶ Was tun für Konzeptklassen mit unendlich vielen Konzepten?
- Unser PAC-Algorithmus wählt eine **konsistente** Hypothese.
 - ▶ Wie bestimmt man konsistente Hypothesen effizient?
 - ▶ Und wenn es keine effizienten Hypothesen gibt?

Was tun bei unendlich vielen Konzepten?

(Allerdings pessimistische) Schranken für die Beispielzahl:

- Konzepte aus **HALBRAUM** $_n$ werden durch $n + 1$ Parameter beschrieben: Die n Koeffizienten und der eine Schwellenwert.
- Für die Darstellung einer Gleitkommazahl genügen c Bits: Die Anzahl der im Rechner darstellbaren Halbräume ist $\leq 2^{c(n+1)}$

$$\implies \lceil \frac{1}{\varepsilon} \cdot \left(c \cdot (n + 1) \ln 2 + \ln \left(\frac{1}{\delta} \right) \right) \rceil$$

klassifizierte Beispiele reichen, um den wahren Fehler ε mit Wahrscheinlichkeit mindestens $1 - \delta$ nicht zu überschreiten.

Der genaue Wert von c ist rechner-abhängig.
Für präzise Schranken der Beispielzahl benötigen wir neue Ideen.

Das ERM-Problem

Wir möchten ein unbekanntes Konzept c mit der Hypothesenklasse \mathcal{H} lernen und fordern klassifizierte Beispiele $(x_1, b_1), \dots, (x_s, b_s)$ an.

Empirische Risiko Minimierung: Bestimme eine Hypothese $h_c \in \mathcal{H}$, so dass das **empirische Risiko** bzw der **empirische Fehler**

$$|\{i : 1 \leq i \leq s, h_c(x_i) \neq b_i\}|$$

minimal ist. (Wenn möglich, bestimme konsistente Hypothesen.)

- Achtung, droht **Overfitting** (bzw Auswendiglernen)?
 - ▶ Die vielen „Freiheitsgrade“ der Hypothesenklasse werden nur teilweise durch die Beispiele gesetzt.
- Oder **Underfitting**?
 - ▶ Die Hypothesenklasse hat zu wenige Freiheitsgrade.
 - ▶ Die Beispielmenge ist auch näherungsweise nicht erklärbar.

In der

empirischen Risiko Minimierung mit induktivem Bias

fließen Annahmen über wesentliche Eigenschaften der Konzeptklasse in die Definition der Hypothesenklasse ein.

- (a) Die Hypothesenklasse sollte möglichst wenige,
- (b) aber genügend viele Freiheitsgrade beitzen

um einen kleinen empirischen Fehler und damit auch einen kleinen **wahren Fehler** zu garantieren.

Sei \mathcal{H} eine Hypothesenklasse über einer Menge X von Beispielen.

(a) Im **ERM-Problem** für die Beispielmenge

$$S = \{(x_1, b_1), \dots, (x_s, b_s)\} \subseteq X \times \{0, 1\}$$

ist eine Hypothese $h_S \in \mathcal{H}$ zu bestimmen, die den empirischen Fehler (bzw das empirische Risiko)

$$|\{i : 1 \leq i \leq s, h_S(x_i) \neq b_i\}|$$

unter allen Hypothesen in \mathcal{H} minimiert. Wir nennen h_S eine **ERM-Hypothese**.

(b) Ein **ERM-Algorithmus** für \mathcal{H} ist ein Lernalgorithmus, der nur ERM-Hypothesen für \mathcal{H} ausgibt.

(Später messen wir den Fehler auch auf andere Arten.)

Inkonsistente Hypothesen

- Für praktische Anwendungen können wir nicht davon ausgehen, dass jedes Zielkonzept auch in der Hypothesenklasse liegt:
Die Schriftproben einer bestimmten Person werden wir nur approximativ, aber nicht exakt erkennen können!
- Stattdessen sollten wir annehmen, dass es zu jedem Zielkonzept $c \in \mathcal{C}$ eine Hypothese $h \in \mathcal{H}$ mit kleinem Fehler gibt, d.h. es gelte

$$\text{fehler}_D(c, h) \leq \varepsilon.$$

Der Lernalgorithmus:

- (1) Bestimme die Beispielzahl s in Abhängigkeit von ε und δ .
- (3) Wähle *irgendeine* Hypothese $h \in \mathcal{H}$, die mit mindestens $(1 - 2\varepsilon) \cdot s$ Beispielen konsistent ist. Gib eine Fehlermeldung aus, wenn eine solche Hypothese nicht existiert.

Bestimme die Beispielzahl s so, dass gilt

$$\text{prob}_D[\text{Algorithmus findet eine Hypothese } h_c \text{ mit } \text{fehler}_D(c, h_c) \leq 4\epsilon] \geq 1 - \delta$$

- Zwei Fehlerquellen: Es gibt keine „fast konsistenten“ Hypothesen (**Fehlermeldung**) oder der Fehler ist zu groß.
- Bestimme s , so dass


$$\text{prob}_D[\text{Algorithmus gibt eine Fehlermeldung}] \leq \frac{\delta}{2} \text{ und}$$
$$\text{prob}_D[\text{fehler}_D(c, h_c) \geq 4\epsilon] \leq \frac{\delta}{2}$$

Die Situation: Es gibt eine Hypothese $h \in \mathcal{H}$ mit $\text{fehler}_D(c, h) \leq \varepsilon$, aber keine Hypothese, ist mit mindestens $(1 - 2\varepsilon) \cdot s$ Beispielen konsistent.

- h ist also mit mehr als $2\varepsilon \cdot s$ Beispielen inkonsistent.
- Wenn wir aber s Beispiele zufällig ziehen, dann ist die erwartete Anzahl der durch h falsch klassifizierten Beispiele höchstens

$$\varepsilon \cdot s.$$

- Im Falle einer Fehlermeldung verstoßen wir signifikant gegen den Erwartungswert: Statt Inkonsistenz mit höchstens $\varepsilon \cdot s$ Beispielen sogar Inkonsistenz mit mehr als $2\varepsilon \cdot s$ Beispielen.
- Ein heftiger Verstoß gegen den Erwartungswert sollte doch unwahrscheinlich sein!

Wir liegen mit $2\varepsilon s$ zweifach über dem Erwartungswert und die Chernoff-Schranke für $\beta = 1$ liefert 

$$e^{-\frac{\varepsilon s}{3}}.$$

als Wahrscheinlichkeit einer Fehlermeldung.

- Wir müssen die Wahrscheinlichkeit einer Fehlermeldung durch $\delta/2$ nach oben beschränken.
- Wir fordern

$$s \geq \frac{3}{\varepsilon} \cdot \ln \left(\frac{2}{\delta} \right).$$

- Sei $h \in \mathcal{H}$ eine beliebige Hypothese mit zu großem Fehler, also $\text{fehler}_D(c, h) \geq 4 \cdot \varepsilon$.
- Ist es nicht dann unwahrscheinlich, dass h mit höchstens $2\varepsilon \cdot s$ Beispielen inkonsistent ist?
Mehr als $4\varepsilon \cdot s$ inkonsistente Beispiele sind zu erwarten.
- Chernoff muss ran und zwar mit $\beta = 1/2$.

$$p = \text{prob}_D \left[\begin{array}{l} h \text{ ist mit höchstens } 2\varepsilon s \\ \text{Beispielen nicht konsistent} \end{array} \right] \leq e^{-\frac{4\varepsilon \cdot s}{4 \cdot 2}} = e^{-\frac{\varepsilon \cdot s}{2}}$$

Für die Wahrscheinlichkeit q eines zu großen Fehlers gilt:

$$q \leq \sum_{h \in \mathcal{H} \text{ mit } \text{fehler}_D(c,h) \geq 4\epsilon} \text{prob}_D \left[\begin{array}{l} h \text{ ist mit } \leq 2\epsilon s \text{ Beispielen} \\ \text{nicht konsistent} \end{array} \right]$$

- Also

$$q \leq \sum_{h \in \mathcal{H}} e^{-\frac{\epsilon s}{2}} = |\mathcal{H}| \cdot e^{-\frac{\epsilon s}{2}}.$$

- Um $q \leq \delta/2$ zu erhalten, fordere $|\mathcal{H}| \cdot e^{-\frac{\epsilon s}{2}} \leq \delta/2$.
- Nach logarithmieren folgt $\ln |\mathcal{H}| - \frac{\epsilon s}{2} \leq \ln \left(\frac{\delta}{2} \right)$ und äquivalent

$$\frac{2}{\epsilon} \cdot \left[\ln |\mathcal{H}| + \ln \left(\frac{2}{\delta} \right) \right] \leq s.$$

Wir erreichen $\text{fehler}_D(\mathbf{c}, h) \leq 4\varepsilon$ mit Wahrscheinlichkeit mindestens $1 - \delta$, falls es eine Hypothese h' mit $\text{fehler}_d(\mathbf{c}, h') \leq \varepsilon$ gibt und falls

$$s(\delta, \varepsilon) \geq \frac{3}{\varepsilon} \cdot \left[\ln |\mathcal{H}| + \ln \left(\frac{2}{\delta} \right) \right].$$

Die Beispielzahl steigt, im Vergleich zur garantierten Existenz konsistenter Lösungen nur moderat an!

? Sind die Schranken für die Beispielzahl scharf?

Sind $\ln(|\mathcal{C}|)$, bzw. $\ln(|\mathcal{H}|)$ gute Messungen der Anzahl der Freiheitsgrade von Konzept- bzw. Hypothesenklasse?

? Wie bestimmt man (fast-)konsistente Hypothesen?

Das agnostische PAC-Modell

Es wäre schön, wenn wir

1. Klassen \mathcal{C} von Funktionen

$$f : X \rightarrow Y$$

mit **beliebigem** Wertebereich Y lernen könnten (bisher $|Y| = 2$),

2. Verteilungen D auf

$$X \times Y$$

zulassen könnten. (Bisher ist D nur auf dem Beispielraum X definiert.)

- ▶ Wir kennen das äußeren Erscheinungsbild x einer Frucht.
- ▶ Was sind die „Schmeckt-lecker-Wahrscheinlichkeiten“

$$\text{prob}_D[\text{Beispiel} = x, y = 1]?$$

Sei \mathcal{H} eine Hypothesenklasse

(a) Eine Funktion der Form

$$\ell : \mathcal{H} \times X \times Y \rightarrow \mathbb{R}_{\geq 0}$$

wird als **Loss-Funktion** bezeichnet.

- ▶ Der 0-1 Loss $\ell(h, x, y) = \begin{cases} 0 & h(x) = y \\ 1 & h(x) \neq y, \end{cases}$ bestraft, wenn $h(x)$ von y verschieden ist,
- ▶ der quadratische Loss $\ell(h, x, y) := (h(x) - y)^2$ bestraft große Abweichungen von y .

(b) Der **erwartete Loss** einer Hypothese $h \in \mathcal{H}$ (bzgl. D und ℓ) ist

$$\text{Loss}_D(h) := \mathbb{E}_{(x,y) \sim D}[\ell(h, x, y)].$$

($\text{Loss}_D(h)$ verallgemeinert unsere Definition von $\text{fehler}_D(c, h_c)$.)

Sei ℓ der 0-1 Loss.

(a) Für eine Zielfunktion $f : X \rightarrow Y$ und eine Hypothese $h : X \rightarrow Y$ ist

$$\text{Loss}_D(h) = \mathbb{E}_{(x,y) \sim D}[\ell(h, x, y)] = \text{prob}_{(x,y) \sim D}[h(x) \neq y].$$

- ▶ Die Verteilung D kann unterschiedliche Werte $y \neq h(x)$ in Abhängigkeit von x unterschiedlich gewichten (sprich: bestrafen)

(b) Sei $Y = \{0, 1\}$, D eine Verteilung „nur“ über dem Beispielraum X . Wir erhalten auch das alte Fehlermodell wie folgt:

- ▶ Setze D auf $X \times \{0, 1\}$ fort.
- ▶ Das Paar $(x, f(x))$ „übernimmt das Gewicht“ des Beispiels x .
- ▶ Es ist

$$\text{Loss}_D(h) = \text{fehler}_D(h)$$

Das einfache **lineare Regressionsproblem**:

- Experimentelle Beobachtungen $(x_1, y_1), \dots, (x_s, y_s) \in \mathbb{R}^2$ liegen vor: Allerdings sind Meßwerte u. U. verfälscht.
- Es wird nach einem „linearen Zusammenhang“ zwischen den Größen x_i und y_i gesucht.
- Bestimme $a, b \in \mathbb{R}$ mit kleinem „Trainingsfehler“

$$\frac{1}{s} \cdot \sum_{i=1}^s (ax_i + b - y_i)^2.$$

- Hoffentlich ist der erwartete Loss

$$\text{Loss}_D(a, b) := \mathbb{E}_{(x,y) \sim D} [(ax + b - y)^2]$$

(zur quadr. Loss-Funktion und Hypothese $h(x) := ax + b$) klein.

Agnostische PAC-Algorithmen

Wir wollen alles, aber

\mathcal{H} sei eine Konzeptklasse über X und Y

Ein Lernalgorithmus A ist genau dann ein

agnostischer PAC-Algorithmus

für \mathcal{H} (und eine Loss-Funktion ℓ), wenn es

- für alle $\varepsilon \in (0, 1]$ und alle $\delta \in (0, 1]$
- eine Beispielzahl $s = s(\varepsilon, \delta)$ gibt,
- so dass A für alle Verteilungen D über X und Y , nach Erhalt von s Beispielen, eine Hypothese h bestimmt mit

$$\text{prob}_{(x,y) \sim D} \left[\overbrace{\mathbb{E}_D[\ell(h, x, y)]}^{\text{Schätzfehler von } h} - \underbrace{\min_{g \in \mathcal{H}} \mathbb{E}_D[\ell(g, x, y)]}_{\text{Approximationsfehler von } \mathcal{H}} \leq \varepsilon \right] \geq 1 - \delta.$$

$$\text{prob}_{(x,y) \sim D} \left[\overbrace{\mathbb{E}_D[\ell(h, x, y)] - \min_{g \in \mathcal{H}} \mathbb{E}_D[\ell(g, x, y)]}^{\text{Schätzfehler von } h} \leq \varepsilon \right] \geq 1 - \delta.$$

Approximationsfehler von \mathcal{H}

- Der erwartete Loss $\mathbb{E}_D[\ell(h, x, y)]$ ist eine Zufallsvariable, denn
 - ▶ die Hypothese h hängt von den gezeigten Beispielen ab.
- Neue Perspektive:
 - ▶ Der **Approximationsfehler** ist eine Eigenschaft der Hypothesenklasse und lässt sich nicht ändern.
 - ▶ Wir sollten aber versuchen, den **Schätzfehler** klein zu halten.

Gibt es agnostische PAC-Algorithmen?

- (a) Für eine Menge $S \subseteq X \times Y$ von s klassifizierten Beispielen ist

$$\text{Loss}^S(h) := \frac{1}{s} \cdot \sum_{(x,y) \in S} \ell(h, x, y)$$

der **empirische Loss** (oder Trainingsfehler) für eine Loss-Funktion ℓ und die Beispielmenge S .

- (b) Eine Hypothese $h \in \mathcal{H}$ ist eine **ERM-Hypothese bezüglich** ℓ auf S , wenn

$$\text{Loss}^S(h) = \min_{g \in \mathcal{H}} \text{Loss}^S(g).$$

Was wäre toll? Wenn sich empirischer Loss und wahrer Loss nur unwesentlich unterscheiden.

ϵ -repräsentative Beispielmengen

\mathcal{H} sei eine Hypothesenklasse über X und Y .

Eine Beispielmenge S ist

ε -repräsentativ

für \mathcal{H} , eine Loss-Funktion ℓ und eine Verteilung D über X und Y , falls

$$|\text{Loss}^S(h) - \text{Loss}_D(h)| \leq \varepsilon$$

für **alle** Hypothesen $h \in \mathcal{H}$ gilt. ▶

- **Wenn** ε -repräsentative Beispielmengen hoch-wahrscheinlich sind,
- dann gilt $\text{Loss}^S(h) \approx \text{Loss}_D(h)$ hoch-wahrscheinlich.

Zeige:

Für jede Verteilung D ist eine gemäß D zufällig ausgewählte Menge S von $s = s(\varepsilon, \delta)$ Beispielen

mit Wahrscheinlichkeit mindestens $1 - \delta$

ε -repräsentativ.

$$\text{Zeige: } p := \text{prob}_D \left[\exists h \in \mathcal{H} : \left| \text{Loss}^S(h) - \text{Loss}_D(h) \right| \geq \varepsilon \right] \stackrel{!}{\leq} \delta.$$

Für eine beliebige Hypothese $h \in \mathcal{H}$ ist

$$\mathbb{E}[\text{Loss}^S(h)] = \frac{1}{s} \cdot \sum_{(x,y) \in S} \mathbb{E}[\ell(h, x, y)] = \frac{1}{s} \cdot \sum_{(x,y) \in S} \text{Loss}_D(h) = \text{Loss}_D(h).$$

Gute Nachrichten:

Der wahre Loss ist der Erwartungswert des empirischen Loss.

Sind signifikante Abweichungen vom Erwartungswert relativ unwahrscheinlich, haben wir gewonnen!

1. Definiere die Zufallsvariable $X_i^h := \ell(h, x_i, y_i)$.
2. Fordere $a_i := 0 \leq X_i^h \leq 1 =: b_i$.

Die Hoeffding-Ungleichung besagt 

$$\begin{aligned} & \text{prob}\left[\left| \text{Loss}^S(h) - \text{Loss}_D(h) \right| \geq \varepsilon \right] \\ &= \text{prob}\left[\left| \sum_{i=1}^s (X_i^h - \mathbb{E}[X_i^h]) \right| \geq \varepsilon \cdot s \right] \leq 2 \exp^{-2s\varepsilon^2}. \end{aligned}$$

$$\begin{aligned} p &= \text{prob}_D \left[\exists h \in \mathcal{H} : \left| \text{Loss}^S(h) - \text{Loss}_D(h) \right| \geq \varepsilon \right] \\ &\leq \sum_{h \in \mathcal{H}} \text{prob}_D \left[\left| \text{Loss}^S(h) - \text{Loss}_D(h) \right| \geq \varepsilon \right] \\ &\leq |\mathcal{H}| \cdot 2 \exp^{-2s\varepsilon^2}. \end{aligned}$$

Wir müssen $p \stackrel{!}{\leq} \delta$ erreichen und die Forderung

$$|\mathcal{H}| \cdot 2 \exp^{-2s\varepsilon^2} \leq \delta$$

ist zu erfüllen \implies

$$s = \frac{1}{2\varepsilon^2} \cdot \left(\ln(2|\mathcal{H}|) + \ln\left(\frac{1}{\delta}\right) \right) \quad \text{Beispiele sind ausreichend.}$$

Sei \mathcal{H} eine Hypothesenklasse mit endlich vielen Hypothesen und sei

$$\underbrace{\ell : \mathcal{H} \times X \times Y \rightarrow [0, 1]} \quad \text{eine Loss-Funktion. Es gelte}$$

Zu schwere Bestrafungen sind verboten

$$\underbrace{s(\varepsilon, \delta) = \frac{1}{2\varepsilon^2} \cdot \left(\ln(2|\mathcal{H}|) + \ln\left(\frac{1}{\delta}\right) \right)}_{\text{Beispielzahl steigt um Faktor } \approx \frac{1}{\varepsilon} \text{ an}}$$

1. Fordert man eine Menge S von $s(\varepsilon/2, \delta)$ Beispielen an und
2. bestimmt eine ERM-Hypothese $h \in \mathcal{H}$ bezüglich ℓ ,
3. so erhält man einen agnostischen PAC-Algorithmus bezüglich ℓ .

Fazit

Ist die Beispielpzahl genügend groß, dann sagt der Trainingsfehler den Verallgemeinerungsfehler

hoch-wahrscheinlich

voraus und zwar simultan für alle Hypothesen.

Was bedeutet dies für praktische Anwendungen?

Occam-Algorithmen

„Non sunt multiplicanda entia praeter necessitatem“.

In freier Übersetzung:

„Wähle stets eine möglichst einfache Erklärung“.

Was ist eine **einfache** Erklärung (bzw. Hypothese)?

Eine Erklärung, die zu einer Teilklasse $\mathcal{H}' \subseteq \mathcal{H}$ mit möglichst wenigen Freiheitsgraden gehört.

- (*) Hier weisen wir jeder Hypothese einen binären Namen zu.
Wir suchen eine ERM-Hypothese mit möglichst kurzem Namen.

Die Konzeptklasse MONOM_n

Sei m ein Monom mit

$$m = l_1 \wedge l_2 \wedge \cdots \wedge l_k.$$

l_1, \dots, l_k sind Literale mit $l_j \in \{x_{i_j}, \neg x_{i_j}\}$.

- Beschreibe m durch die Binärdarstellung

$$b_1 \cdot \text{code}(i_1) \cdot b_2 \cdot \text{code}(i_2) \cdots b_k \cdot \text{code}(i_k).$$

- ▶ $\text{code}(i)$ ist die Binärdarstellung der Zahl $i \in \{0, \dots, n-1\}$ mit führenden Nullen auf $\lceil \log_2 n \rceil$ Bits aufgefüllt.
- ▶ $b_j \cdot \text{code}(i_j)$ gibt an, ob $l_j = x_{i_j}$ ($b_j = 0$) bzw. $l_j = \neg x_{i_j}$ ($b_j = 1$).
- Die Beschreibungslänge ist $k(\lceil \log_2 n \rceil + 1)$.

Unser PAC-Algorithmus für MONOM_n hat ein **längstes** konsistentes Monom bestimmt. War das eine gute Idee?

Setup:

- Für $h \in \mathcal{H}$ sei $\text{länge}(h)$ die Länge des binären Namens von h .
- \mathcal{H}_r : alle Hypothesen in \mathcal{H} mit Namenslänge kleiner als r .

Für welche Beispielzahl $s_r(\varepsilon, \delta)$ erhalten wir einen PAC-Algorithmus, wenn wir eine konsistente Hypothese in \mathcal{H}_r finden?

- Es ist $|\mathcal{H}_r| \leq \sum_{i=0}^{r-1} 2^i = 2^r - 1$.
- Sei $p_{s,r}$ die Wahrscheinlichkeit, dass es eine Hypothese $h \in \mathcal{H}_r$ gibt, die mit s Beispielen konsistent ist **und einen Fehler $> \varepsilon$ hat**.

Dann ist

$$p_{s,r} \leq (1 - \varepsilon)^s \cdot |\mathcal{H}_r| \leq (1 - \varepsilon)^s \cdot 2^r.$$

(1) Wir möchten $p_{s,r} \leq \delta$ erreichen und müssen fordern

$$(1 - \varepsilon)^s \cdot 2^r \leq \delta.$$

(2) Nach Logarithmieren ist dies äquivalent zu:

$$s \cdot \ln(1 - \varepsilon) + r \cdot \ln 2 \leq \ln \delta.$$

(3) Aber $\ln(1 - \varepsilon) \leq -\varepsilon$ mit der Mutter aller Ungleichungen. Fordere

$$-s \cdot \varepsilon + r \cdot \ln 2 \leq \ln(\delta).$$

Wenn für

$$s_r(\varepsilon, \delta) = \frac{1}{\varepsilon} \cdot \left(r \cdot \ln 2 + \ln \left(\frac{1}{\delta} \right) \right)$$

Beispiele eine Hypothese h der Länge $< r$ gefunden wird, dann hat h mit Wahrscheinlichkeit $\geq 1 - \delta$ einen Fehler $\leq \varepsilon$.

Fazit: Occam-Algorithmen

Erfolgreiches Lernen für Hypothesenlänge $< r$ und

$$s = \frac{1}{\varepsilon} \cdot \left(r \cdot \ln 2 + \ln \left(\frac{1}{\delta} \right) \right) \text{ Beispiele.}$$

- Erfolgreiches Lernen, wenn die konsistente Hypothese wesentlich kürzer ist als die Anzahl der Beispiele, d.h. $r \ll s$.
- Kurze Hypothesen **komprimieren** die Beispiele!

Hat ein unbekanntes Monom höchstens k Literale, dann

- genügen $\approx \frac{1}{\varepsilon} \cdot \left(k \log_2 n + \ln \left(\frac{1}{\delta} \right) \right)$ Beispiele.
- Unser erster Algorithmus benötigt $\approx \frac{1}{\varepsilon} \cdot \left(n + \ln \left(\frac{1}{\delta} \right) \right)$ Beispiele.

Zusammenfassung: Endliche Hypothesenklassen

Was haben wir insgesamt erreicht?

- Das Gütesiegel „hoch-wahrscheinlich kleiner Fehler“ kann bei

$$\Omega\left(\frac{1}{\varepsilon} \cdot (\ln(|\mathcal{H}|) + \ln\left(\frac{1}{\delta}\right))\right)$$

Beispielen an ERM-Hypothesen vergeben werden.

- Occam-Algorithmen erlauben eine Vergabe des Gütesiegels bei mgl. sehr kleiner Beispielmenge.
- Eine ERM-Hypothese erhält das Gütesiegel in Gold („holt hoch-wahrscheinlich raus, was drin ist“) bei

$$\Omega\left(\frac{1}{\varepsilon^2} \cdot (\ln(|\mathcal{H}|) + \ln\left(\frac{1}{\delta}\right))\right)$$

Beispielen: Schafft Verteilungen auf $X \times Y$ und Loss-Funktionen!

Ja, aber

- Werden so viele Beispiele auch tatsächlich benötigt?
Kann man den Begriff *Freiheitsgrad* formalisieren?
- Lassen sich keine scharfen Aussagen über die Beispielzahl für reellwertige Hypothesenklassen wie etwa HALBRAUM_n machen?
- Lassen sich ERM-Hypothesen effizient bestimmen?
- Modellieren wir Lern-Situationen in der Praxis?
 - ▶ Sind die in Anwendungen „gelieferten“ Beispiele „unabhängig“?
 - ▶ Warum wird Lernen „gegen“ **jede** Verteilung gefordert?

VC-Dimension und Freiheitsgrade

Wie misst man Freiheitsgrade?

Sei \mathcal{C} eine Konzeptklasse

(1) \mathcal{C} zertrümmert eine Menge S von Beispielen, falls

$$\mathcal{P}(S) = \{S \cap c \mid c \in \mathcal{C}\}.$$

($\mathcal{P}(S) = \{T \mid T \subseteq S\}$ ist die Potenzmenge von S .)

(2) Die VC-Dimension $\text{VC}(\mathcal{C})$ ist die maximale Mächtigkeit einer Menge S , die von \mathcal{C} zertrümmert wird.

Wenn \mathcal{C} die Menge S zertrümmert, dann erhalten wir jede Teilmenge von S als den Durchschnitt von S mit einem Konzept in \mathcal{C} :

Wenn \mathcal{C} die Menge S zertrümmert, dann muss bei kleinem Fehler die Klassifikation eines jeden Beispiels in S bekannt sein!

Die Anzahl der Freiheitsgrade

Wir definieren $VC(\mathcal{C})$ als die Anzahl der Freiheitsgrade von \mathcal{C} .
Ist diese Definition vernünftig?

BOOLEAN_n besitzt ein Konzept c_f für jede Boolesche Funktion $f : \{0, 1\}^n \rightarrow \{0, 1\}$:

$$c_f = \{x \in \{0, 1\}^n \mid f(x) = 1\}.$$

- **BOOLEAN**_n „sollte“ 2^n Freiheitsgrade besitzen, da der Wert von f auf jedem Argument $x \in \{0, 1\}^n$ bestimmt werden muss.
- **BOOLEAN**_n zertrümmert $\{0, 1\}^n$, denn es gibt für jede Teilmenge $T \subseteq \{0, 1\}^n$ ein Konzept $c \in \mathbf{BOOLEAN}_n$ mit $c = T$.
- Also ist $VC(\mathbf{BOOLEAN}_n) = 2^n$.

RECHTECK $_n$ besitzt für jedes achsenparallele Rechteck $R \subseteq \mathbb{R}^n$ ein Konzept, das aus allen Punkten in R besteht.

- (1) Die Anzahl der Freiheitsgrade von **RECHTECK** $_2$ „sollte“ vier sein, denn jedes Rechteck wird durch die vier Koordinaten diagonal gegenüber liegender Eckpunkte beschrieben.
- (2) Die vier Punkte

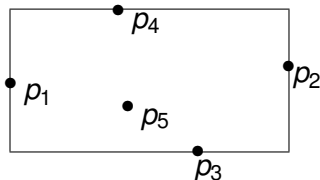
$$e_1 = (-1, 0), \quad e_2 = (0, 1), \quad e_3 = (0, -1), \quad e_4 = (1, 0)$$

werden von RECHTECK_2 zertrümmert und $\text{VC}(\text{RECHTECK}_2) \geq 4$ folgt.

$VC(\text{RECHTECK}_2) = 4$

Wir zeigen $VC(\text{RECHTECK}_2) \leq 4$.

- (1) Angenommen, RECHTECK_2 zertrümmert die fünf Punkte p_1, p_2, p_3, p_4, p_5 .
- (2) p_1 und p_2 seien die Punkte mit minimaler (bzw. maximaler) x -Koordinate. p_3 und p_4 seien die Punkte mit minimaler (bzw. maximaler) y -Koordinate.
- (3) Dann kann die Teilmenge $T = \{p_1, p_2, p_3, p_4\}$ aber nicht von p_5 getrennt werden!



Jedes Konzept in **HALBRAUM_n** besteht aus allen Punkten $x \in \mathbb{R}$, die die Ungleichung $\sum_{i=1}^n w_i \cdot x_i \geq t$ erfüllen.

HALBRAUM_n „sollte“ $n + 1$ Freiheitsgrade besitzen: Jedes Konzept wird durch die n Gewichte und den Schwellenwert beschrieben.

- **HALBRAUM_n** zertrümmert die $n + 1$ Punkte $0, e_1, \dots, e_n$, wobei e_i der i te Einheitsvektor ist. Warum?
 - ▶ Sei $T \subseteq \{0, e_1, \dots, e_n\}$ eine beliebige Teilmenge.
 - ▶ Wenn $0 \notin T$, wähle c als den Halbraum $\sum_{i=1}^n w_i \cdot x_i \geq 1$ mit $w_i = 1$, wenn $e_i \in T$, und $w_i = -1$, wenn $e_i \notin T$.
 - ▶ Wenn $0 \in T$, dann arbeite mit den gleichen Gewichten, setze aber den Schwellenwert auf 0.
- Also ist $VC(\text{HALBRAUM}_n) \geq n + 1$.

- Sei S eine Menge von $n + 2$ Punkten.
- Wie beschafft man sich eine Teilmenge $T \subseteq S$, die nicht von ihrem Komplement $S \setminus T$ getrennt werden kann?

Der Satz von Radon

Für jede Menge $S \subseteq \mathbb{R}^n$ mit $|S| \geq n + 2$ gibt es eine nicht-leere Teilmenge $T \subset S$, so dass

$$\text{konvexe Hülle}(T) \cap \text{konvexe Hülle}(S \setminus T) \neq \emptyset.$$

Zum Beispiel: Vier Punkte im \mathbb{R}^2 besitzen stets eine Teilmenge T , die nicht von ihrem Komplement getrennt werden kann.

Angenommen, $\sum_{i=1}^n w_i \cdot x_i \geq t$ für alle Punkte in T **und**

$\sum_{i=1}^n w_i \cdot x_i < t$ für alle Punkte in $S \setminus T$.

(1) Setze

$$H_1 = \left\{ x \in \mathbb{R}^n \mid \sum_{i=1}^n w_i x_i \geq t \right\}, \quad H_2 = \left\{ x \in \mathbb{R}^n \mid \sum_{i=1}^n w_i x_i < t \right\}.$$

(2) Dann $H_1 \cap H_2 = \emptyset$.

(3) Aber $T \subseteq H_1$ und $S \setminus T \subset H_2$ nach Annahme.

(4) Wir erhalten einen Widerspruch zum Satz von Radon, denn H_1 und H_2 sind konvex.

VC(HALBRAUM_n) ≤ n + 1.

- (a) Die Konzeptklasse **KUGEL**_{*d*} besitzt für jede Kugel

$$K_r(m) = \{y \in \mathbb{R}^d : \|x - m\| \leq r\}$$

in \mathbb{R}^d ein Konzept. Es ist $\text{VC}(\text{KUGEL}_d) = d + 1$.

- (b) Die Konzeptklasse **k-GONS** ist die Klasse der konvexen Polygone in \mathbb{R}^2 mit *höchstens* *k* Ecken. Das Konzept besteht aus allen eingeschlossenen Punkten. Es ist $\text{VC}(\text{k-GONS}) = 2k + 1$.

- Der Beweis wird in den Übungen behandelt.
- *k*-gons werden doch durch *k* Punkte mit jeweils zwei Koordinaten festgelegt?!
(Aber *höchstens* *k* Ecken werden gefordert.)

Und dann sowas!

Für $a \in \mathbb{R}$ definieren wir das Konzept

$$\text{Sinus}(a) \subseteq \mathbb{R}$$

- 1 Es ist $x \in \text{Sinus}(a) \iff \lceil \sin(a \cdot x) \rceil = 1$. (Wir setzen $\lceil -1 \rceil = 0$.)
- 2 Die Konzeptklasse \mathfrak{G} besteht aus allen Konzepten $\text{Sinus}(a)$.

\mathfrak{G} wird durch einen Parameter beschrieben, aber

$$\text{VC}(\mathfrak{G}) = \infty.$$

Welche Sichtweise ist richtig: Kann man \mathfrak{G} mit endlich vielen Beispielen lernen?

- (a) Wenn $\mathcal{C}_1 \subseteq \mathcal{C}_2$, dann ist $VC(\mathcal{C}_1) \leq VC(\mathcal{C}_2)$.
- (b) $\bar{\mathcal{C}} = \{ \bar{c} \mid c \in \mathcal{C} \}$ ist die Komplement-Konzeptklasse von \mathcal{C} .
Dann gilt $VC(\mathcal{C}) = VC(\bar{\mathcal{C}})$.
- (c) Für jede endliche Konzeptklasse \mathcal{C} ist $VC(\mathcal{C}) \leq \lfloor \log_2 |\mathcal{C}| \rfloor$.

Zu Teil (c):

- (*) Angenommen, \mathcal{C} zertrümmert eine Beispielmenge S .
- (*) Dann gibt es zu jeder Teilmenge $T \subseteq S$ ein Konzept $c_T \in \mathcal{C}$ mit $T = S \cap c_T$.
- (*) Also ist die Anzahl der Teilmengen von S höchstens so groß wie die Anzahl der Konzepte in $\mathcal{C} \implies 2^{|S|} \leq |\mathcal{C}|$.
- (*) $\implies VC(\mathcal{C}) \leq \lfloor \log_2 |\mathcal{C}| \rfloor$.

Die VC-Dimension wichtiger Konzeptklassen

- (a) $VC(\text{MONOTON-MONOM}_n) = n$.
- (b) $VC(\text{MONOM}_1) = 2$ und $VC(\text{MONOM}_n) = n$ für $n \geq 2$.
- (c) $\binom{n}{k} \leq VC(k\text{-KNF}_n) = VC(k\text{-DNF}_n) \leq \binom{n}{k} \cdot 2^k$.
- (d) $VC(k\text{-KLAUSEL-KNF}_n) = VC(k\text{-Term-DNF}_n) = \Theta(k \cdot n)$
für $k \leq 2^{n/2}$.
- (e) $VC(\text{BOOLEAN}_n) = 2^n$.
- (f) $VC(\text{SYM}_n) = n + 1$.
- (g) $VC(\text{AUTOMAT}_{n,\Sigma}) = \Theta(n \cdot |\Sigma| \cdot \log_2 n)$.
- (h) $VC(\text{RECHTECK}_n) = 2n$.
- (i) $VC(\text{HALBRAUM}_n) = n + 1$.

Die VC-Dimension von SYM_n

- Eine boolesche Funktion $f : \{0, 1\}^n \rightarrow \{0, 1\}$ heißt **symmetrisch**, wenn der Wert $f(x)$ nur von der Anzahl der Einsen in x abhängt.
 - **SYM_n** besitzt für jede symmetrische Funktion f das Konzept $c_f = \{x \mid f(x) = 1\}$.
 - Behauptung: $\text{VC}(\text{SYM}_n) = n + 1$.
-
- Die Anzahl der symmetrischen Funktionen f stimmt mit der Anzahl der Funktionen $g : \{0, \dots, n\} \rightarrow \{0, 1\}$ überein.
 - $\implies |\text{SYM}_n| = 2^{n+1}$ und $\text{VC}(\text{SYM}_n) \leq n + 1$ folgt.
 - Der Vektor e_i habe Einsen in den Positionen $1, \dots, i$ und sonst nur Nullen. Wir zertrümmern $S = \{e_0, \dots, e_n\}$ mit $e_0 = 0$.
 - ▶ Für $T \subseteq S$ definiere die symmetrische Funktion f mit
$$f(e_i) = \begin{cases} 1 & e_i \in T \\ 0 & \text{sonst,} \end{cases}$$
 - ▶ Dann ist $S \cap c_f = T$.

Konsistente Hypothesen: Untere Schranken für die Beispielzahl

Wieviele Beispiele sind notwendig: Intervalle

Die Konzeptklasse **INTERVALL** besteht aus allen Intervallen $I \subseteq [0, 1]$.
Wieviele Beispiele sind notwendig, um Intervalle mit
Wahrscheinlichkeit $\geq 1 - \delta$ und Fehler höchstens ε zu lernen?

- **INTERVALL** besteht aus überabzählbar unendlich vielen Konzepten. Unsere $\ln |\mathcal{C}|$ -Schranke versagt völlig.
- Angenommen, unser Lernalgorithmus gibt das kleinste Intervall, das alle positiven Beispiele enthält, als Hypothese aus.

Übungen: $s = \Omega\left(\frac{\ln(1/\delta)}{\varepsilon}\right)$ Beispiele sind notwendig, um das Zielkonzept $I = [0, 1]$ zu lernen.

Nenne eine Konzeptklasse \mathcal{C} **trivial**, wenn \mathcal{C} nur aus einem Konzept oder aus disjunkten Konzepten besteht.

- (1) Wenn \mathcal{C} nicht-trivial ist, dann gibt es zwei Beispiele $x \neq y$ und zwei Konzepte $c_1, c_2 \in \mathcal{C}$ mit
 $\{x, y\} \subseteq c_1$ sowie $x \in c_2, y \notin c_2$.
- (2) Mache x hoch-wahrscheinlich (Wahrscheinlichkeit $1 - 2\varepsilon$) und y niedrig-wahrscheinlich (Wahrscheinlichkeit 2ε).
- (3) Ein deterministischer Lernalgorithmus irrt für entweder c_1 oder c_2 , wenn Beispiel y nicht gesehen wird.

Wenn der Lernalgorithmus y nicht sieht, dann folgt der unerlaubt große Fehler 2ε .

- (1) Es muss gewährleistet sein, dass der Algorithmus Beispiel y mit Wahrscheinlichkeit höchstens δ nicht sieht.
- (2) Aber y wird mit Wahrscheinlichkeit $(1 - 2\varepsilon)^s = e^{-\Theta(\varepsilon \cdot s)}$ nicht gesehen.
- (3) Wir müssen $e^{-\Theta(\varepsilon \cdot s)} \leq \delta$, bzw. $1/\delta \leq e^{\Theta(\varepsilon \cdot s)}$ fordern.

Für jede nicht-triviale Konzeptklasse \mathcal{C} werden mindestens

$$\Omega\left(\frac{1}{\varepsilon} \ln\left(\frac{1}{\delta}\right)\right)$$

Beispiele benötigt.

Wieviele Beispiele sind notwendig: Der allgemeine Fall

Es gelte $VC(\mathcal{C}) = d + 1$ und \mathcal{C} zertrümmere die Beispielmenge

$$S = \{x_0, \dots, x_d\}$$

- (1) S wird zertrümmert: Jede Teilmenge von S entspricht einem Konzept.
- (2) Wir definieren eine schwierige Verteilung D auf den Beispielen:
 - ▶ x_0 ist hoch-wahrscheinlich, und wird mit Wahrscheinlichkeit $1 - 4\varepsilon$ gewählt.
 - ▶ Jedes andere x_i erhält die Wahrscheinlichkeit $4\varepsilon/d$.
 - ▶ Beispiele außerhalb S erhalten die Wahrscheinlichkeit 0.
- (3) x_0 wird sehr häufig gewählt: Wann sieht ein Lernalgorithmus genügend viele verschiedene klassifizierte Beispiele in S ?

Wieviele Beispiele sind notwendig: Der allgemeine Fall

Wenn s Beispiele angefordert werden und wenn $d = \text{VC}(\mathcal{C})$:

- (1) Die erwartete Häufigkeit von x_0 ist $s \cdot (1 - 4\varepsilon)$.
- (2) Der Algorithmus sieht nur $s \cdot 4\varepsilon$ Beispiele aus $\{x_1, \dots, x_d\}$.
- (3) Angenommen $s \leq \frac{d}{8\varepsilon}$.
 - ▶ Der Algorithmus sieht nur $d/2$ verschiedene Beispiele aus S .
 - ▶ Die Klassifikation der nicht gesehenen Beispiele ist beliebig (denn S wird zertrümmert) und damit nicht prognostizierbar.
 - ▶ Der Algorithmus muss die Klassifikation der nicht gesehenen $d/2$ Beispiele raten und wird sich für ein Zielkonzept irren.
 - ▶ \implies der erwartete Fehler ist $\geq (d/2) \cdot (4\varepsilon/d) = 2\varepsilon$.

Mindestens

$$\Omega\left(\frac{\text{VC}(\mathcal{C})}{\varepsilon}\right)$$

Beispiele müssen angefordert werden.

Wir kombinieren die untere Schranke bei großer VC-Dimension mit der unteren Schranke für kleines δ :

Für jede nicht-triviale Konzeptklasse \mathcal{C} sind mindestens

$$\Omega\left(\frac{1}{\varepsilon} \cdot \left[\text{VC}(\mathcal{C}) + \ln\left(\frac{1}{\delta}\right) \right]\right)$$

Beispiele notwendig bzw der Fehler beträgt mindestens

$$\Omega\left(\frac{1}{s} \cdot \left[\text{VC}(\mathcal{C}) + \ln\left(\frac{1}{\delta}\right) \right]\right).$$

Diese untere Schranke ist in vielen Fällen asymptotisch exakt!

Konsequenzen der unteren Schranke

Da $\Omega(\frac{1}{\varepsilon} \cdot (\text{VC}(\mathcal{C}_n) + \ln(\frac{1}{\delta}))) = \text{Beispiel}(\mathcal{C}_n) = O(\frac{1}{\varepsilon} \cdot (\ln(|\mathcal{C}_n|) + \ln(\frac{1}{\delta})))$, kann die Beispielzahl exakt vorausgesagt werden, wenn

$$\text{VC}(\mathcal{C}_n) = \Theta(\ln(|\mathcal{C}_n|)).$$

- (a) $\text{Beispiel}(\text{MONOTON-MONOM}_n) = \Theta(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (b) $\text{Beispiel}(\text{MONOM}_n) = \Theta(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (c) $\text{Beispiel}(k\text{-KNF}_n) = \text{Beispiel}(k\text{-DNF}_n) = \Theta(\frac{n^k}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (e) $\text{Beispiel}(k\text{-KLAUSEL-KNF}_n) = \Theta(\frac{k \cdot n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (f) $\text{Beispiel}(k\text{-TERM-DNF}_n) = \Theta(\frac{k \cdot n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (g) $\text{Beispiel}(\text{BOOLEAN}_n) = \Theta(\frac{1}{\varepsilon} \cdot 2^n + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (h) $\text{Beispiel}(\text{SYM}_n) = \Theta(\frac{n}{\varepsilon} + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$.
- (i) $\text{Beispiel}(\text{AUTOMAT}_{n,\Sigma}) = \Theta(\frac{1}{\varepsilon} \cdot n \cdot \log_2 n + \frac{1}{\varepsilon} \cdot \ln(\frac{1}{\delta}))$ für $|\Sigma| = 2$.

No free Lunch

Gibt es universelle Lernalgorithmen?

Sei $X = \{0, 1\}^n$.

- ? Warum wählen wir eine Hypothesenklasse a priori aus?
- ? Warum nicht Algorithmen bauen, die prinzipiell **jede mögliche** Hypothese als Antwort geben können?

Dann müssen wir mit der Hypothesenklasse BOOLEAN_n arbeiten, die für jede Teilmenge von $X = \{0, 1\}^n$ ein Konzept besitzt.

No-Free-Lunch-Theorem

Der Lernalgorithmus A fordere weniger als $|X|/2$ Beispiele an \implies
Es gibt ein Konzept $c \subseteq X$, eine Verteilung D , eine reelle Zahl $\mu > 0$,
so dass für die von A bestimmte Hypothese h_c gilt:

$$\text{prob}[\text{fehler}_D(c, h_c) \geq \mu] \geq \mu.$$

Konsistente Hypothesen: Wieviele Beispiele genügen?

Das Ziel

Die Konzeptklasse \mathcal{C} sei gegeben mit $VC(\mathcal{C}) = d$. Dann gibt es einen PAC-Algorithmus für \mathcal{C} , der höchstens

$$\left\lceil \frac{4}{\varepsilon} \cdot \left(d \cdot \ln \left(\frac{12}{\varepsilon} \right) + \ln \left(\frac{2}{\delta} \right) \right) \right\rceil \text{ Beispiele anfordert.}$$

- Vollständig zertrümmerte Beispielmengen S sind **chaotisch**, denn jede Teilmenge kann als Konzept auftreten.
- Bei genügend vielen Beispielen werden aber (fast) alle Freiheitsgrade „gesetzt“, und wir haben kein Problem.

Was passiert, wenn der Algorithmus eine Beispielmenge S mit $s \gg VC(\mathcal{C})$ Beispielen sieht? Wie chaotisch ist S ?

Das partielle Zertrümmern großer Beispielmengen

\mathcal{C} zertrümmert eine Beispielmenge S mit $|S| = VC(\mathcal{C})$. Für größere Beispielmengen: Wieviele Teilmengen „produziert“ \mathcal{C} höchstens?

- Für eine Beispielmenge S setze

$$\Pi_{\mathcal{C}}(S) := \{ c \cap S \mid c \in \mathcal{C} \}.$$

- Für $s \in \mathbb{N}$ ist

$$\Pi_{\mathcal{C}}(s) := \max\{ |\Pi_{\mathcal{C}}(S)| \mid |S| = s \}.$$

Sauer's Lemma

Für $d = VC(\mathcal{C})$ und jedes $s \in \mathbb{N}$ ist

$$\Pi_{\mathcal{C}}(s) \leq \sum_{i=0}^d \binom{s}{i}.$$

Für $s \geq d$ ist $\Pi_{\mathcal{C}}(s) \leq \left(\frac{e \cdot s}{d}\right)^d$.

Der (einfache) Beweis wird im Skript beschrieben.

Die Beispielmengung S sei gegeben und es gelte $d = VC(\mathcal{C})$.

(a) Wieviele Konzepte der Konzeptklasse \mathcal{C} unterscheiden sich auf S ?

- ▶ Genau $|\Pi_{\mathcal{C}}(S)|$ Konzepte, denn $\Pi_{\mathcal{C}}(S) = \{c \cap S \mid c \in \mathcal{C}\}$.
- ▶ Es gibt höchstens

$$\sum_{i=0}^d \binom{s}{i} \leq \left(\frac{e \cdot s}{d}\right)^d$$

Konzepte aus \mathcal{C} , die sich auf S voneinander unterscheiden.

(b) Wie scharf ist die obere Schranke in Sauer's Lemma?

- Für $s \leq d = VC(\mathcal{C})$ ist $\Pi_{\mathcal{C}}(s) = 2^s$, denn es gibt Beispielmengungen der Größe s , die vollständig zertrümmert werden können.
- Sauer's Lemma ist exakt für $s \leq d$:

$$2^s = \Pi_{\mathcal{C}}(s) \leq \sum_{i=0}^s \binom{s}{i} = \sum_{i=0}^s \binom{s}{i} 1^i \cdot 1^{s-i} = (1 + 1)^s$$

denn $\binom{s}{i} = 0$ für $i > s$.

(c) Es gibt höchstens endlich viele verschiedene Konzepte „über S “, nämlich höchstens $(\frac{e \cdot s}{d})^d$ Konzepte.

► **Reine Spekulation** (?!):

Und wenn wir die Schranke für endliche Konzeptklassen anwenden? Für geeignete Konstanten C, D folgt:

$$s \leq \frac{C}{\varepsilon} \cdot \left(d \cdot \ln\left(\frac{e \cdot s}{d}\right) + \ln\left(\frac{1}{\delta}\right) \right) \leq \frac{D}{\varepsilon} \cdot \left(d \cdot \ln\left(\frac{1}{\varepsilon}\right) + \ln\left(\frac{1}{\delta}\right) \right)$$

Beispiele genügen.

► Genau das möchten wir zeigen!

ε -Netze

Die Situation:

- Eine Beispielmenge S mit s Beispielen wurde zufällig gewählt.
- Das (unbekannte) Zielkonzept sei $c \in \mathcal{C}$ und D sei die Verteilung.
- **Gefahr**: Es gibt konsistentes Konzept $c' \in \mathcal{C}$ mit großem Fehler.
 - ▶ c' ist genau dann mit S konsistent, wenn $S \cap (c \oplus c') = \emptyset$.
 - ▶ Der Fehler ist zu groß, falls $\text{fehler}_D(c, c') > \varepsilon$.


Wir nennen

$$\Delta_\varepsilon(c) = \{c \oplus c' \mid c' \in \mathcal{C} \text{ und } \text{fehler}_D(c, c') > \varepsilon\}$$

die Menge der ε -**Fehlerregionen** für c .

Wir nennen eine Beispielmenge S ein ε -**Netz**, wenn

$$S \cap c^* \neq \emptyset$$

für jede ε -Fehlerregion $c^* \in \Delta_\varepsilon(c)$. 

Angenommen, S ist ein ε -Netz und c' ist eine konsistente Hypothese mit zu großem Fehler:

- Dann ist $c \oplus c'$ eine ε -Fehlerregion $\implies c \oplus c' \in \Delta_\varepsilon(c)$.
- Aber S ist ein ε -Netz $\implies S \cap (c \oplus c') \neq \emptyset$.
- Es gibt ein $x \in S$ mit $x \in c \oplus c' \implies c'$ ist nicht auf S konsistent. ⚡

Ein konsistenter Lernalgorithmus ist ein PAC-Algorithmus, falls

$$\text{prob}[S \text{ ist ein } \varepsilon\text{-Netz}] \geq 1 - \delta.$$

Sei A_s das Ereignis, dass eine Menge von s Beispielen **kein** ε -Netz ist. Was ist die Wahrscheinlichkeit von A_s ?

$$\begin{aligned}\text{prob}[A_s] &= \text{prob}[S \text{ ist } \textit{kein} \varepsilon\text{-Netz}] \\ &= \text{prob}[\text{Es gibt ein } c^* = c \oplus c' \in \Delta_\varepsilon(c) \text{ und } S \cap c^* = \emptyset] \\ &\leq \sum_{c^* \in \Delta_\varepsilon(c)} \text{prob}[S \cap c^* = \emptyset]\end{aligned}$$

Aber die Anzahl der Fehlerregionen in $\Delta_\varepsilon(c)$ kann sogar unendlich groß sein. So geht es leider nicht.

Es ist

$$\text{prob}[A_S] = \text{prob}[\text{Es gibt ein } c^* = c \oplus c' \in \Delta_\varepsilon(c) \text{ und } S \cap c^* = \emptyset].$$

Wir definieren das Ereignis

$$B_S \equiv \exists c^* \in \Delta_\varepsilon(c) \left[c^* \cap S_1 = \emptyset \wedge |c^* \cap S_2| \geq \frac{\varepsilon \cdot S}{2} \right]$$

für Beispielmengen S_1, S_2 von jeweils s Beispielen.

B_S tritt ein, wenn eine schlechte Hypothese c' mit $c^* = c \oplus c'$

- (*) nicht auf den ersten s ,
- (*) wohl aber auf den zweiten s Beispielen auffällt.

Das Ereignis B_s

(1) B_s tritt für (S_1, S_2) ein.

- ▶ die schlechte Hypothese c^* fällt auf S_1 nicht auf
- ▶ S_1 ist kein ε -Netz
- ▶ Ereignis A_s tritt für S_1 ein.

(2) Also folgt $\text{prob}[B_s] = \text{prob}[B_s \mid A_s] \cdot \text{prob}[A_s]$.

- ▶ Eine schlechte Hypothese c^* sollte „wahrscheinlich“ auf S_2 inkonsistent sein $\implies \text{prob}[B_s \mid A_s]$ sollte groß sein.

Wenn $s \geq \frac{8}{\varepsilon}$, dann ist

$$\text{prob}[A_s] \leq 2 \cdot \text{prob}[B_s].$$

Der Beweis (via Chernoff-Schranke) wird im Skript beschrieben.

Zwischenfazit

- (1) Unser Ziel ist der Nachweis, dass eine Menge von s Beispielen hoch-wahrscheinlich ein ε -Netz ist, falls s hinreichend groß ist.
- (2) A_s ist das Ereignis, dass eine Menge von s Beispielen **kein** ε -Netz ist. Wir wissen, dass $\text{prob}[A_s] \leq 2 \cdot \text{prob}[B_s]$ mit

$$B_s \equiv \exists c^* \in \Delta_\varepsilon(c) \left[c^* \cap S_1 = \emptyset \wedge |c^* \cap S_2| \geq \frac{\varepsilon \cdot s}{2} \right].$$

- (3) Zeige, dass B_s unwahrscheinlich ist.
- (4) Für B_s ziehe eine Beispielmenge S von $2s$ Beispielen:
 - ▶ Zuerst die s Beispiele aus S_1 ,
 - ▶ dann die s Beispiele aus S_2 .

- Beachte, dass $(c^* \cap S) \cap S_1 = c^* \cap (S \cap S_1) = c^* \cap S_1$ und $(c^* \cap S) \cap S_2 = c^* \cap (S \cap S_2) = c^* \cap S_2$.
- Ersetze c^* durch $c^* \cap S$.

- (1) Wir fassen die **Einzelexperimente** (ziehe zuerst S_1 , dann S_2) zu **Großexperimenten** S zusammen:
 - ▶ Ziehe zuerst S , dann zerlege S zufällig in S_1 und S_2 .
- (2) Fixiere ab jetzt das Großexperiment S .

Wie groß ist die Wahrscheinlichkeit

$$\text{prob}[B_S \mid S]$$

des Ereignisses B_S , wenn wir das Großexperiment S durchführen?

$$\begin{aligned}
 & \text{prob}[B_S \mid S] \\
 = & \text{prob}[\exists c^* \in \Delta_\varepsilon(c) : (c^* \cap S) \cap S_1 = \emptyset, | (c^* \cap S) \cap S_2 | \geq \frac{\varepsilon \cdot S}{2}] \\
 \leq & \sum_{c^* \cap S, c^* \in \Delta_\varepsilon(c)} \text{prob}[(c^* \cap S) \cap S_1 = \emptyset, | (c^* \cap S) \cap S_2 | \geq \frac{\varepsilon \cdot S}{2}].
 \end{aligned}$$

Wie groß ist die Menge $\{c^* \cap S \mid c^* \in \Delta_\varepsilon(c)\}$?

- (1) Die Menge $\Delta_\varepsilon(c)$ der Fehlerregionen ist eine Konzeptklasse.
 - ▶ Dann ist $\Delta_\varepsilon(c) \subseteq c \oplus \mathcal{C}$
 - ▶ und $VC(\Delta_\varepsilon(c)) \leq VC(c \oplus \mathcal{C}) = VC(\mathcal{C})$ folgt.
- (2) Die Menge $\{c^* \cap S \mid c^* \in \Delta_\varepsilon(c)\}$ ist die Einschränkung der Konzeptklasse $\Delta_\varepsilon(c)$ auf die Beispielmenge S .
- (3) Wende Sauer's Lemma für $d = VC(\mathcal{C})$ an:

$$|\{c^* \cap S \mid c^* \in \Delta_\varepsilon(c)\}| \leq \left(\frac{e \cdot s}{d}\right)^d.$$

Wie groß ist

$$\text{prob} [(\mathbf{c}^* \cap \mathcal{S}) \cap \mathcal{S}_1 = \emptyset, (\mathbf{c}^* \cap \mathcal{S}) \cap \mathcal{S}_2 \mid \geq \frac{\varepsilon \cdot \mathcal{S}}{2}]$$

für eine beliebige Fehlerregion \mathbf{c}^* ?

- (1) Angenommen, $|\mathbf{c}^* \cap \mathcal{S}| = r$.
- (2) Wie groß ist die Wahrscheinlichkeit, dass die Auswahl von \mathcal{S}_1 diese r Elemente auslasst?
 - ▶ Genau 2^{-r} , denn jedes der r Beispiele in $\mathbf{c}^* \cap \mathcal{S}$ erscheint mit Wahrscheinlichkeit genau $\frac{1}{2}$ in \mathcal{S}_2 .
- (3) Fur uns ist aber nur der Fall $r \geq \frac{\varepsilon \cdot \mathcal{S}}{2}$ interessant.

$$\text{Es ist } \text{prob} [(\mathbf{c}^* \cap \mathcal{S}) \cap \mathcal{S}_1 = \emptyset, (\mathbf{c}^* \cap \mathcal{S}) \cap \mathcal{S}_2 \mid \geq \frac{\varepsilon \cdot \mathcal{S}}{2}] \leq 2^{-\frac{\varepsilon \cdot \mathcal{S}}{2}}.$$

$$\begin{aligned}
& \text{prob}[B_S \mid S] \\
& \leq \sum_{\mathbf{c}^* \cap S, \mathbf{c}^* \in \Delta_\varepsilon(c)} \text{prob}[(\mathbf{c}^* \cap S) \cap S_1 = \emptyset, (\mathbf{c}^* \cap S) \cap S_2 \mid \geq \frac{\varepsilon \cdot S}{2}] \\
& \leq \left(\frac{e \cdot S}{d}\right)^d \cdot 2^{-\frac{\varepsilon \cdot S}{2}}.
\end{aligned}$$

A_S war das Ereignis, dass eine Menge von s Beispielen **kein** ε -Netz ist.

Also folgt $\text{prob}[A_S] \leq 2 \cdot \text{prob}[B_S] \leq 2 \cdot \left(\frac{e \cdot S}{d}\right)^d \cdot 2^{-\frac{\varepsilon \cdot S}{2}}$.

Die Forderung $\text{prob}[A_s] \leq \delta$ führt auf

$$2 \cdot \left(\frac{e \cdot 2s}{d} \right)^d \cdot e^{-\frac{\varepsilon \cdot s}{2}} \leq \delta.$$

Jetzt logarithmiere

$$d \cdot \ln \left(\frac{2e \cdot s}{d} \right) - \frac{\varepsilon \cdot s}{2} \leq \ln \frac{\delta}{2}$$

und vertausche Terme

$$d \cdot \ln \left(\frac{2e \cdot s}{d} \right) + \ln \frac{2}{\delta} \leq \frac{\varepsilon \cdot s}{2}$$

Fordere $s = \frac{\alpha}{\varepsilon} \cdot (d \ln(\frac{1}{\varepsilon}) + \ln(\frac{1}{\delta}))$ für eine große Konstante α .

Zusammenfassung

Sei \mathcal{C} eine Konzeptklasse mit $d = \text{VC}(\mathcal{C})$.

- (1) $s = \frac{\alpha}{\epsilon} \cdot (d \ln(\frac{1}{\epsilon}) + \ln(\frac{1}{\delta}))$ Beispiele genügen für ein erfolgreiches PAC-Lernen von \mathcal{C} , solange α hinreichend groß ist, aber
- (2) $s = \frac{\beta}{\epsilon} \cdot (d + \ln(\frac{1}{\delta}))$ Beispiele sind erforderlich, solange β hinreichend klein ist.

Konsequenzen:

- Die VC-Dimension formalisiert den Begriff „Freiheitsgrad“ für das PAC-Modell.
- Für endliche Konzeptklassen: $\text{VC}(\mathcal{C}) = \Theta(\ln(|\mathcal{C}|)) \implies$

$$s = \Theta\left(\frac{1}{\epsilon} \cdot (\ln |\mathcal{C}_n| + \ln(\frac{1}{\delta}))\right)$$

Beispiele sind hinreichend **und** notwendig.

Es gibt Konstanten C_1, C_2 und D_1, D_2 mit

$$\frac{C_1}{\varepsilon} \cdot \left(n + \ln\left(\frac{1}{\delta}\right)\right) \leq \text{Beispiel}_{\text{RECHTECK}_n}(\varepsilon, \delta) \leq \frac{C_2}{\varepsilon} \cdot \left(n \cdot \ln\left(\frac{1}{\varepsilon}\right) + \ln\left(\frac{1}{\delta}\right)\right).$$

$$\frac{D_1}{\varepsilon} \cdot \left(n + \ln\left(\frac{1}{\delta}\right)\right) \leq \text{Beispiel}_{\text{HALBRAUM}_n}(\varepsilon, \delta) \leq \frac{D_2}{\varepsilon} \cdot \left(n \cdot \ln\left(\frac{1}{\varepsilon}\right) + \ln\left(\frac{1}{\delta}\right)\right).$$

Beispielkomplexität: Agnostische PAC-Algorithmen

Sei \mathcal{H} eine Konzeptklasse mit $VC(\mathcal{H}) = d$. Dann gilt für den 0-1 Loss:

- (a) Es gibt eine Konstante C_1 , so dass jeder ERM-Algorithmus, der mindestens

$$\frac{C_1}{\epsilon^2} \cdot \left(d + \ln \left(\frac{1}{\delta} \right) \right)$$

Beispiele anfordert, ein agnostischer PAC-Algorithmus für \mathcal{H} ist.

- (b) Es gibt eine Konstante C_2 , so dass jeder agnostische PAC-Algorithmus für \mathcal{H} mindestens

$$\frac{C_2}{\epsilon^2} \cdot \left(d + \ln \left(\frac{1}{\delta} \right) \right)$$

Beispiele anfordern muss.

Siehe Shalev-Shwartz und Ben-David.

Die folgenden Aussagen sind äquivalent:

- (a) $VC(\mathcal{H}) < \infty$,
- (b) \mathcal{H} ist PAC-lernbar (mit ERM-Hypothesen),
- (c) \mathcal{H} ist agnostisch PAC-lernbar (mit ERM-Hypothesen).

Wie bitte? Ist die Konzeptklasse

Vereinigung von Intervallen

nicht lernbar?

Die algorithmische Komplexität

Das Konsistenzproblem

- (a) Eine **parametrisierte** Konzeptklasse $\mathcal{C} = (\mathcal{C}_n)_{n \in \mathbb{N}}$ besteht aus Konzeptklassen \mathcal{C}_n mit Beispielen der Länge höchstens n .
- (b) Ein PAC-Algorithmus A für \mathcal{C} ist **effizient**, wenn A für \mathcal{C}_n höchstens $s = \text{poly}(n, \frac{1}{\epsilon}, \log_2(\frac{1}{\delta}))$ Beispiele anfordert und in Zeit $\text{poly}(s)$ rechnet.

- (1) Die wichtigsten Konzeptklassen besitzen eine in der Beispiellänge n polynomielle VC-Dimension:

Bis auf **BOOLEAN** _{n} genügen $s = \text{poly}(n, \frac{1}{\epsilon}, \log_2(\frac{1}{\delta}))$ Beispiele.

- (2) Um eine gute Hypothese in Zeit $\text{poly}(s)$ zu bestimmen, sollten wir insbesondere das **Konsistenzproblem**

Gibt es eine konsistente Hypothese für eine Menge klassifizierter Beispiele?

in Zeit $\text{poly}(s)$ lösen können.

Einfache Konzeptklassen

Die folgenden Konzeptklassen besitzen effiziente PAC-Algorithmen:

- (a) MONOTON-MONOM,
MONOM,
 k -DNF für fixiertes k ,
 k -KNF für fixiertes k ,
 k -TERM-DNF für fixiertes k (mit Hypothesenklasse $\mathcal{H} = k$ -KNF),
 k -KLAUSEL-KNF für fixiertes k (mit Hypothesenklasse $\mathcal{H} = k$ -DNF),
- (b) SYM,
- (c) RECHTECK und
- (c) HALBRAUM.

(a1) MONOTON-MONOM_n:

- ▶ Das Monom

$$x_1 \wedge x_2 \wedge \cdots \wedge x_n$$

ist die erste Hypothese.

- ▶ Streiche alle Variablen, die einem positiven Beispiel widersprechen.

(a2) MONOM_n:

- ▶ Das Monom

$$x_1 \wedge \neg x_1 \wedge x_2 \wedge \neg x_2 \wedge \cdots \wedge x_n \wedge \neg x_n$$

ist die erste Hypothese.

- ▶ Streiche alle Literale, die einem positiven Beispiel widersprechen.

(a3) **k-KNF**: In den Übungen.

(a4) **k-DNF**: In den Übungen

(a5) **k-TERM DNF**:

- ▶ Zu jeder **k-Term-DNF**-Formel gibt es eine äquivalente **k-KNF**-Formel.
- ▶ **ACHTUNG**: Wenn **k-TERM DNF** als Hypothesenklasse benutzt wird, dann ist das Konsistenzproblem NP-vollständig und effizientes Lernen ist unmöglich!

(a6) **k-KLAUSEL KNF**: Wie **k-TERM DNF**.

(b) **SYM**_n:

- ▶ Wenn eine Funktion $f : \{0, 1\}^n \rightarrow \{0, 1\}$ symmetrisch ist und die Eingabe x genau i Einsen hat, dann ist $f(x) = f(1^i 0^{n-i})$.
- ▶ Die Hypothese akzeptiert Eingaben mit i Einsen genau dann, wenn es mindestens ein positives Beispiel mit i Einsen gibt.

(c) **RECHTECK**_n:

- ▶ Bestimme das kleinste Rechteck, das alle positiven Beispiele enthält.
- ▶ Wie? Für Beispiele $x_1, \dots, x_s \in \mathbb{R}^n$ und jede Dimension j bestimme das Intervall $I_j = [\min_{i=1}^s x_{i,j}, \max_{i=1}^s x_{i,j}]$.
- ▶ Setze $R = I_1 \times \dots \times I_n$.

(d) HALBRAUM_n:

- ▶ x_1, \dots, x_r seien die positiven und z_1, \dots, z_t seien die negativen Beispiele.
- ▶ Stelle das folgende lineare Programm auf

$$\max \varepsilon \text{ so dass} \quad \begin{array}{l} w^T \cdot x_i \geq t + \varepsilon \text{ für } i = 1, \dots, r \\ w^T \cdot z_j \leq t - \varepsilon \text{ für } j = 1, \dots, t. \end{array}$$

Die Variablen des linearen Programms sind ε , der Schwellenwert t und die Komponenten des Vektors w .

- ▶ Das lineare Programm kann mit Interior Point Verfahren (oder mit dem Simplex Verfahren) effizient gelöst werden.

(e) \wedge -HALBRAUM_n: Alle Konzepte, die durch eine Konjunktion von zwei Threshold-Gattern berechnet werden.

Das Konsistenzproblem ist NP-vollständig,

Wie entscheidend ist das Konsistenzproblem?

Und wenn das Konsistenzproblem für \mathcal{C} nicht effizient lösbar ist?
Kann es trotzdem effiziente PAC-Algorithmen für \mathcal{C} geben?

- (1) **Ja**: wir zeigen später, dass das Konsistenzproblem für **k-TERM DNF** \mathcal{NP} -vollständig ist, aber **k-TERM DNF** kann effizient mit Hypothesenklasse **k-KNF** gelernt werden.
- (2) Wenn das Konsistenzproblem für \mathcal{C} \mathcal{NP} -vollständig ist: Gibt es effiziente PAC-Algorithmen für \mathcal{C} mit \mathcal{C} als Hypothesenklasse?
Wahrscheinlich nicht: Wir konstruieren aus einem effizienten Lernalgorithmus A einen effizienten, randomisierten Algorithmus für das Konsistenzproblem.

Eine effiziente Lösung des Konsistenzproblems für \mathcal{C}_n

- (1) Die klassifizierte Beispielmenge S sei vorgegeben. Wir müssen entscheiden, ob es ein mit S konsistentes Konzept in \mathcal{C}_n gibt.

Setze $\varepsilon = \frac{1}{|S|+1}$ und $\delta = \frac{1}{2}$.

- (2) Der PAC-Algorithmus A möge $s = s(\varepsilon, \delta) = \text{poly}(n)$ Beispiele anfordern. Erzeuge s Beispiele aus S gemäß der Verteilung

$$D(x) = \begin{cases} \frac{1}{|S|} & x \in S \\ 0 & \text{sonst.} \end{cases}$$

- (3) Sei h_c die von A ausgegebene Hypothese.
/* h_c ist eine effiziente Lösung des Konsistenzproblem für S mit Wahrscheinlichkeit mindestens $\frac{1}{2}$. */
- (4) Wenn h_c konsistent ist, antworte „ S ist konsistent mit einem Konzept in \mathcal{C} “ (**fehlerfrei**) und ansonsten „ S hat kein konsistentes Konzept in \mathcal{C} “ (mgl. **falsch**).

PAC-Lernen von \mathcal{C} mit Hypothesenklasse \mathcal{C}

Warum ist h_c eine effiziente Lösung des Konsistenzproblem für S mit Wahrscheinlichkeit mindestens $\frac{1}{2}$?

(1) Als PAC-Algorithmus erfüllt A

$$\text{prob}[\text{fehler}_D(c, h_c) \leq \frac{1}{|S|+1}] \geq \frac{1}{2},$$

(2) Wenn aber nur ein einziges Beispiel aus S missklassifiziert wird, dann ist $\text{fehler}_D(c, h_c) \geq \frac{1}{|S|} > \frac{1}{|S|+1}$.

- ▶ Unser Algorithmus ist fehlerfrei bei positiver Antwort.
- ▶ Eine negative Antwort ist mit Wahrscheinlichkeit $\leq \frac{1}{2}$ fehlerhaft.

Wenn das **Konsistenzproblem für \mathcal{C}** NP-vollständig ist, dann gibt es vermutlich keine effizienten Algorithmen, die \mathcal{C} mit \mathcal{C} lernen.

3-Klausel-KNF: Schwieriges Lernen,
wenn die Hypothesenklasse vorgegeben ist

Das 3-Färbbarkeitsproblem

- Eingabe ist ein ungerichteter Graph $G = (V, E)$ mit Knotenmenge V und Kantenmenge E .
- Ist es möglich die Knoten mit drei Farben zu färben, so dass jede Kante nur verschieden gefärbte Knoten verbindet?

Das 2-Färbbarkeitsproblem ist effizient lösbar, das 3-Färbbarkeitsproblem ist hingegen NP-vollständig.

Konstruiere die polynomielle Reduktion

3-Färbbarkeit \leq_p Konsistenz Problem für **3-KLAUSEL-KNF**.

Die Reduktion

Der ungerichtete Graph $G = (V, E)$ mit $V = \{1, \dots, n\}$ sei die Eingabe für 3-Färbbarkeit.

Welche negativen, welche positiven Beispiele sollen wir wählen?

- (1) Die Beispiele sind Belegungen aus $\{0, 1\}^n$:
Bitposition v „entspricht“ dem Knoten v .
 - ▶ Die Menge S_- der negativen Beispiele besteht aus allen Belegungen mit genau einer Eins.
 - ▶ Die Menge S_+ der positiven Beispiele besteht aus allen Belegungen mit genau zwei Einsen, die einer Kante von G entsprechen.
- (2) Die Beispielmenge ist $S = S_- \cup S_+$.

Zeige: G ist 3-färbbar \iff

es gibt eine mit S konsistente 3-KLAUSEL-KNF.

Angenommen, der Graph G ist 3-färbbar mit der Färbung

$$\text{Farbe} : V \rightarrow \{1, 2, 3\}.$$

$c = k_1 \wedge k_2 \wedge k_3$ ist eine KNF mit den drei Klauseln

$$k_i \equiv \bigvee_{v \in V, \text{Farbe}(v) \neq i} x_v.$$

Frage: Ist c konsistent mit der Beispielmenge S ?

(1) Jede Belegung x mit genau einer Eins verwirft c :

$x_v = 1 \implies x$ falsifiziert die Klausel k_i (mit $i = \text{Farbe}(v)$).

(2) Jede Belegung, die einer Kante entspricht, akzeptiert c :

Denn u und v besitzen verschiedene Farben.

Angenommen die 3-KLAUSEL-KNF_n-Formel $c \equiv k_1 \wedge k_2 \wedge k_3$ ist konsistent mit der Beispielmenge S .

(1) Alle Belegungen x mit genau einer Eins (in Position v) verwerfen c :

- ▶ Es gibt mindestens eine Klausel $k_{f(v)}$, die für x falsch wird.
- ▶ Wähle $f(v) \in \{1, 2, 3\}$ als Farbe für Knoten v .

(2) f definiert eine 3-Färbung.

Warum verbinden Kanten nur verschieden gefärbte Knoten?

- ▶ Sei $\{u, v\}$ eine Kante von G .
- ▶ $f(u) = f(v) \implies$ Klausel $k_{f(u)}$ enthält weder x_u noch x_v
- ▶ \implies die der Kante $\{u, v\}$ entsprechende Belegung verwirft c ⚡

Das Konsistenzproblem für **3-KLAUSEL-KNF** ist NP-vollständig.

Sehr schwierige Konzeptklassen

- Die Konzeptklasse **3-KLAUSEL-KNF** ist nur schwierig, wenn **3-KLAUSEL-KNF** auch als Hypothesenklasse verwandt wird.
 - ▶ In den Übungen: **3-KLAUSEL-KNF** kann mit k -**DNF** (höchstens k Literale pro Monom) gelernt werden.
- Gibt es Konzeptklassen, die effizientes Lernen für **alle** potentiellen Hypothesenklassen verhindern?

Erfahrungsgemäß sehr schwierige Konzeptklassen sind

- **SCHALTKREISE**, sogar bei nur logarithmischer Tiefe.
- **NEURONALE NETZWERKE**, sogar bei kleiner Tiefe
- wie auch **ENDLICHE AUTOMATEN**.

Wir werden zeigen, dass wir das RSA-Kryptosystem brechen können, wenn wir eine dieser Konzeptklassen effizient lernen können.

- **Alice** und **Bob** möchten verschlüsselte Nachrichten über einen öffentlichen Kanal austauschen.
- **Eve** möchte die mitgehörten Nachrichten entschlüsseln.

Zur Verschlüsselung werden **Trapdoor-Funktionen** f_w eingesetzt:

- ▷ $f_w(x)$ soll **einfach**, $f_w^{-1}(y)$ **schwierig** zu berechnen sein.

Die Kommunikation:

- (1) Alice wählt eine Trapdoor-Funktion f_w und veröffentlicht ein Programm zur Berechnung von f_w .
- (2) Bob kodiert seine Nachricht x mit dem veröffentlichten Programm und veröffentlicht seinerseits $f_w(x)$.
- (3) Alice kann $f_w(x)$ effizient dekodieren, da sie w kennt. Eve kennt w nicht und kann deshalb keine effiziente Dekodierung erreichen.

Lernen und Kryptographie

Das RSA-Kryptosystem

Das RSA-Kryptosystem von Rivest, Shamir und Adleman (1978) :

- (1) Alice wählt zwei große Primzahlen p und q mit $p \neq q$, sowie den Kodierungsexponenten e . Sie veröffentlicht $N = p \cdot q$ und e .
- (2) Bob kodiert eine Nachricht x (mit $1 \leq x < N$ und $\text{ggT}(x, N) = 1$):
 - ▶ Zuerst berechnet er (durch fortgesetztes Quadrieren)

$$x^{2^k} \bmod N,$$

für $k = 0, \dots, \log_2 N$,

- ▶ danach

$$x^e \bmod N$$

durch Multiplikation der geeigneten Potenzen und veröffentlicht

$$f_{p,q,e}(x) = x^e \bmod N.$$

Alice kennt die Primfaktoren p und q und dekodiert $f_{p,q,e}(x)$. Wie?

$\phi(N)$ ist die Anzahl der primen Restklassen modulo N , also

$$\phi(N) = | \{ x \in \{1, \dots, N\} \mid \text{ggT}(x, N) = 1 \} |$$

- Welche Zahlen sind nicht teilerfremd zu N ?
 - ▶ die p Vielfachen von q , d.h. $\{q, 2q, \dots, (p-1)q, p \cdot q\}$
 - ▶ und die q Vielfachen von p , d.h. $\{p, 2p, \dots, (q-1) \cdot p, q \cdot p\}$.
- $\phi(N) = p \cdot q - p - q + 1 = (p-1)(q-1)$.
- Alice kennt p und q und kann deshalb $\phi(N)$ leicht bestimmen.

Wie dekodiert Alice?

Der Satz von Euler

Für jede endliche Gruppe G mit Gruppenordnung m gilt

$$g^m \equiv 1$$

für alle $g \in G$.

- (1) Bob sendet die Nachricht $y = x^e \pmod N$.
- (2) Alice bestimmt den Dekodierungsexponenten d so, dass

$$e \cdot d \equiv 1 \pmod{\phi(N)}.$$

Sie arbeitet mit dem Euklidischen Algorithmus (für e und $\phi(N)$).

- (3) Alice berechnet

$$y^d = (x^e)^d \equiv x^{ed} \equiv x^{1+k \cdot \phi(N)} \equiv x^1 \equiv x \pmod N.$$

Die Konzeptklasse **RSA**

(a) Die Parameter einer Konzeptklasse:

- ▶ $N = p \cdot q$ ist das Produkt zweier verschiedener Primzahlen p, q ,
- ▶ $e \in \mathbb{Z}_{\phi(N)}^*$ ist der Kodierungsexponent und
- ▶ i ist eine Bitposition mit $1 \leq i \leq \lfloor \log_2 N \rfloor + 1$.

(b) Das Konzept $c_{N,e,i}$ besteht aus allen Vektoren

$$\left(N, e, x^e \bmod N, x^{e \cdot 2^1} \bmod N, \dots, x^{e \cdot 2^{\lfloor \log_2 N \rfloor}} \bmod N \right),$$

so dass das i te Bit von x mit 1 übereinstimmt.

(c) Die Konzeptklasse **RSA** $_n$ umfasst alle Konzepte $c_{N,e,i}$ mit RSA-Modulen N der Bitlänge höchstens n . Es ist

$$\mathbf{RSA}_n = \left\{ c_{N,e,i} \mid \begin{array}{l} N \text{ ist RSA-Modul mit } 1 \leq N < 2^n, \\ e \in \mathbb{Z}_{\phi(N)}^* \text{ und } 1 \leq i \leq \lfloor \log_2 N \rfloor + 1 \end{array} \right\}.$$

Es ist **RSA** = $(\mathbf{RSA}_n)_{n \in \mathbb{N}}$.

Die RSA Hypothese

Es gibt keinen probabilistischen Algorithmus A , so dass:

- (a) Für jede Zahl $N = p \cdot q$ und jeden Exponenten $e > 1$ mit $\text{ggT}(e, \phi(N)) = 1$ gilt

$$\text{prob} \left[\begin{array}{l} A \text{ berechnet zu gegebenen } N, e, x^e \bmod N \\ \text{das Urbild } x, \text{ sofern } x^e \in \mathbb{Z}_N^* \text{ und } 0 \text{ sonst} \end{array} \right] \geq \frac{3}{4},$$

wobei die Wahrscheinlichkeit über die internen Münzwürfe von A und die gleichverteilte Wahl von $x \in \{1, \dots, N-1\}$ zu bilden ist.

- (b) Algorithmus A läuft in Zeit $\text{poly}(\log_2 N)$.

Die RSA-Hypothese „steht“ seit über 40 Jahren.

Angenommen, es gibt einen **effizienten** PAC-Algorithmus für **RSA**.

- (1) Alice veröffentlicht ihre Kodierung (N, e) .
- (2) Eve knackt das RSA-Kryptosystem wie folgt:
 - ▶ Eve bestimmt zufällig Zahlen $x_1, \dots, x_s \in \{0, \dots, N-1\}$ gemäß der Gleichverteilung und berechnet jeweils $y_i = x_i^e \bmod N$.
 - ▶ Sie klassifiziert genau die Beispiele y_i als positiv für die x_j an der niedrigstwertigsten Position eine 1 hat und $\text{ggT}(x_j, N) = 1$ gilt.
 - ▶ Die klassifizierten Beispiele werden einem PAC-Algorithmus (mit Fehler $\varepsilon = \frac{1}{n^2}$ und $\delta = \frac{1}{n^2}$) übergeben.
- (3) Eve kann das niedrigstwertige Bit der Nachricht entschlüsseln. Sie wiederholt ihr Vorgehen für die verbleibenden Bits.

RSA-Hypothese \implies **RSA** besitzt keine effizienten PAC-Algorithmen, sogar, wenn eingeschränkt auf $D =$ Gleichverteilung.

Konsequenzen

- (a) Die RSA-Hypothese möge gelten.
- (b) $\mathcal{C} = (\mathcal{C}_n \mid n \in \mathbb{N})$ sei eine parametrisierte Konzeptklasse
- (c) Für jedes $n \in \mathbb{N}$ möge es $m = \text{poly}(n)$ geben, so dass alle Konzepte in RSA_n auch Konzepte in \mathcal{C}_m sind.

$\implies \mathcal{C}$ hat keine effizienten PAC-Algorithmen, selbst wenn als Verteilung D nur die Gleichverteilung benutzt wird.

Welche Konzeptklassen sind mächtig genug, um **RSA** zu enthalten?

LOG-SCHALTKREIS_n:

- ▶ Sei S ein $\{\wedge, \vee, \neg\}$ -Schaltkreis mit **Fanin höchstens zwei**, **Eingaben** $x \in \{0, 1\}^n$, **Tiefe höchstens $\log_2 n$** und einem Ausgabegatter.
- ▶ **LOG-SCHALTKREIS_n** enthält die von S berechnete boolesche Funktion als Konzept.

Wie einschneidend ist die Einschränkung auf logarithmische Tiefe?

RSA_n ⊆ LOG-SCHALTKREIS_{poly(n)}.

- (1) Schaltkreise können n Binärzahlen $x_1, \dots, x_n \in \{0, 1\}^n$ in logarithmischer Tiefe $O(\log_2 n)$ miteinander multiplizieren.
- (2) Wenn N und der Dekodierungsexponent d bekannt sind, dann kann in logarithmischer Tiefe
 - ▶ überprüft werden, ob die Eingabe die Form

$$\left(N, e, y \bmod N, y^2 \bmod N, y^{2^2} \bmod N, \dots, y^{2^{\lfloor \log_2 N \rfloor}} \bmod N \right)$$

- ▶ hat und die Nachricht x kann rekonstruiert werden:

$$x \equiv y^d \equiv y^{\sum_{i=0}^k d_i \cdot 2^i} \equiv \prod_{i, d_i = 1} y^{2^i} \bmod N.$$

Reverse Engineering für Schaltkreise ist selbst bei logarithmischer Tiefe äußerst schwierig. Und für Threshold-Schaltkreise **beschränkter Tiefe**?

THRESHOLD-SCHALTKREIS $_{t,n}$:

- ▶ Ein Schaltkreis S ist ein Threshold-Schaltkreis, wenn seine Gatter Threshold-Funktionen der Form

$$f(y_1, \dots, y_m) = \begin{cases} 0 & \sum_{i=1}^m w_i \cdot y_i < t \\ 1 & \text{sonst} \end{cases}$$

berechnen.

- ▶ **THRESHOLD-SCHALTKREIS $_{t,n}$** enthält für jeden Threshold-Schaltkreis der Tiefe t mit n Eingabe-, einem Ausgabegatter sowie höchstens n^2 Gattern
die von S berechnete boolesche Funktion als Konzept.
- ▶ Threshold-Schaltkreise können n Binärzahlen $x_1, \dots, x_n \in \{0, 1\}^n$ in Tiefe 5 mit $\text{poly}(n)$ Gattern miteinander multiplizieren.

$\text{RSA}_n \subseteq \text{THRESHOLD-SCHALTKREIS}_{7,m}$ für $m = \text{poly}(n)$.

Sind neuronale Netze effizient lernbar?

Die RSA-Hypothese gelte. Dann hat weder **LOG-SCHALTKREIS** $_n$ noch **THRESHOLD-SCHALTKREIS** $_{7,n}$ effiziente PAC-Algorithmen.

- Threshold-Schaltkreise mit sieben Schichten sind nicht effizient lernbar, egal mit welcher Hypothesenklasse.
- Neuronale (feedforward-)Netzwerke besitzen als Gattertypen
 - ▶ Threshold-Gatter,
 - ▶ reellwertige Gatter wie
 - ★ Rectifier: $r(x) = \max\{0, x\}$,
 - ★ die Softplus-Funktion: $f(x) = \ln(1 + \exp^x)$ oder
 - ★ das Standard-Sigmoid: $\sigma(x) = \frac{1}{1 + \exp^{-x}}$.

Feedforward Netze beschränkter Tiefe sind mindestens so mächtig wie Threshold-Schaltkreise mit sieben Schichten.

Ausdrucksstärke muss mit **ineffizientem** Lernen bezahlt werden.

⇒ Lernverfahren für neuronale Netzwerke

- sind entweder sehr aufwändig oder
- kommen ohne jegliche Garantie.

Eine Familie von „**Mechanismen**“ $\mathcal{M}_{N,d_0\dots d_k\dots}$ ist zum Beispiel dann ausdrucksstark, wenn

$$\mathcal{M}_{N,d_0\dots d_k\dots}(y \bmod N, \dots, y^{2^k} \bmod N, \dots) = y^{\sum_k d_k 2^k} \bmod N.$$

Sind endliche Automaten effizient lernbar?

Endliche Automaten haben,

wenn die Eingabe häufig genug wiederholt wird,

die Berechnungskraft von Schaltkreisen.

Sei S ein Schaltkreis der Tiefe $t = \log_2 n$.

Konstruiere einen DFA A_S mit $\text{poly}(2^t)$ Zuständen, der

$$\underbrace{x \cdots x}_{2^{t+1}-1 \text{ Mal}} = x^{2^{t+1}-1}$$

genau dann akzeptiert, wenn S die Eingabe x akzeptiert.

Schaltkreise: Eine Simulation durch DFAs

Das **Pebble-Spiel**:

Eine Schaltungsauswertung mit möglichst wenigen Registern:

- Ein Pebble darf auf eine Quelle des Schaltkreises gesetzt werden
 - ▶ (das entsprechende Eingabebit wird in ein Register geladen)
 - „Pebbel“ ein Gatter, wenn alle Vorgänger Pebbles besitzen
 - ▶ (werte das Gatter aus und lade Ergebnis in ein Register).
- Im gleichen Zug dürfen alle „Vorgänger-Pebbles“ gelöscht werden
- ▶ (die entsprechenden Register werden wieder freigegeben).
- Ein Pebble darf jederzeit entfernt werden.

Sei S ein $\{\wedge, \vee, \neg\}$ -Schaltkreis mit **Fan-in höchstens 2** und **Tiefe t** :

- ? S kann mit $\leq t + 1$ Pebbles in Zeit $\leq 2^{t+1} - 1$ gepebbelt werden.
- ? Ein DFA mit $\text{poly}(2^t)$ Zuständen kann S simulieren.

Beispielkomplexität und algorithmische Komplexität, eine Zusammenfassung

(a) Das PAC-Modell:

- ▶ Fairness Bedingung: Werte die Hypothese auf derselben Verteilung aus auf der Beispiele erzeugt wurden.
- ▶ Für Fehler $\leq \varepsilon$ mit Wahrscheinlichkeit $\geq 1 - \delta$ bei konsistenten Hypothesen:
 - ★ mindestens $\Omega\left(\frac{1}{\varepsilon} \cdot [\text{VC}(\mathcal{C}) + \ln\left(\frac{1}{\delta}\right)]\right)$ Beispiele sind erforderlich,
 - ★ höchstens $O\left(\frac{1}{\varepsilon} \cdot [\text{VC}(\mathcal{C}) \cdot \ln\left(\frac{1}{\varepsilon}\right) + \ln\left(\frac{1}{\delta}\right)]\right)$ Beispiele genügen.

\implies VC-Dimension = Anzahl Freiheitsgrade.

(b) Agnostische PAC-Algorithmen müssen rausholen, was drin ist.

(c) Gute Lernalgorithmen bestimmen eine ERM-Hypothese.

- ▶ **Beispielzahl** $\gg \frac{\text{Parameterzahl}}{\varepsilon}$, bzw $\gg \frac{\text{Parameterzahl}}{\varepsilon^2} \implies$ Lernerfolg.
- ▶ Occam Algorithmen suchen eine einfachste ERM-Hypothese.

(d) Schwierige Konzeptklassen.

▶ Repräsentationsabhängige Ergebnisse:

- ★ **k-TERM DNF** kann nicht mit **k-TERM DNF** effizient PAC-gelernt werden, wohl aber mit Hypothesenklasse **k-KNF**.
- ★ Wenn \mathcal{C} der Durchschnitt von zwei Halbräumen ist, dann ist \mathcal{C} nicht effizient mit \mathcal{C} als Hypothesenklasse PAC-lernbar.
- ★ Es gibt keine effizienten agnostischen PAC-Algorithmen für **MONOTON-MONOM** bzw **HALBRAUM**.

▶ Repräsentationsunabhängige Ergebnisse: Weder

- ★ **SCHALTKREIS**_n noch
- ★ neuronale feedforward Netze aus mindestens sieben Schichten noch
- ★ endliche Automaten

sind mit *irgendeiner* Hypothesenklasse effizient PAC-lernbar.

- + Die Beispielauswahl wird durch Beispielverteilungen modelliert.
 - ▶ In „industriellen“ Anwendungen werden Millionen von Beispielen eingesetzt: Eine sorgfältige Auswahl ist nicht möglich.
- ? Ist die Annahme unabhängig voneinander gezogener Beispiele in Anwendungen gerechtfertigt?
 - Erfolgreiches Lernen wird für *jede* Beispielverteilung gefordert.
 - ▶ In Anwendungen versucht, man das Lernen mit der der Auswahl der Beispiele zielgerichtet zu unterstützen.
- +/- Wenige theoretische Ergebnisse über die Kombination von Hypothesenklasse und Beispielauswahl in Abhängigkeit von der zu lernenden Konzeptklasse.

Die Mutter aller Ungleichungen

Für jedes $x > -1$ gilt

$$e^{x/(1+x)} \leq 1 + x \leq e^x.$$

Darüberhinaus gilt $1 + x \leq e^x$ für alle $x \in \mathbb{R}$.

- Der natürliche Logarithmus ist eine konkave Funktion:
Die Steigung der Sekante in den Punkten 1 und $1 + x$ ist
 - ▶ durch die Tangentensteigungen in den Punkten 1 nach oben
 - ▶ und $1 + x$ nach unten beschränkt.
- Als Konsequenz

$$\frac{1}{1+x} \leq \frac{\ln(1+x) - \ln(1)}{(1+x) - 1} = \frac{\ln(1+x)}{x} \leq 1.$$

- Jetzt exponentieren. ◀

Die Chernoff-Schranke

- (*) X_1, \dots, X_n seien **unabhängige, binäre** Zufallsvariablen:
Jedes X_i nimmt also nur die Werte 0 oder 1 an.
- (*) X_i habe die „Erfolgswahrscheinlichkeit“ $p_i = \text{prob}[X_i = 1]$.
- (*) Dann ist $N = \sum_{i=1}^n p_i$ die erwartete Anzahl der Erfolge.

$$\text{prob}\left[\sum_{i=1}^n X_i > (1 + \beta) \cdot N\right] \leq \left(\frac{e^\beta}{(1 + \beta)^{1+\beta}}\right)^N \leq e^{-N \cdot \beta^2/3}$$

$$\text{prob}\left[\sum_{i=1}^n X_i < (1 - \beta) \cdot N\right] \leq \left(\frac{e^{-\beta}}{(1 - \beta)^{1-\beta}}\right)^N \leq e^{-N \cdot \beta^2/2}$$

für jedes $\beta > 0$ (bzw. $0 < \beta \leq 1$ im zweiten Fall). ◀

Die Hoeffding-Ungleichung

X_1, \dots, X_n seien **unabhängige** Zufallsvariablen, so dass für alle i

$$a_i \leq X_i - \mathbb{E}[X_i] \leq b_i$$

mit Wahrscheinlichkeit 1 gelte. Dann folgt für alle $\epsilon > 0$

$$\text{prob}\left[\left|\sum_{i=1}^n \left(X_i - \mathbb{E}[X_i]\right)\right| \geq \epsilon \cdot n\right] \leq 2 \exp^{-\frac{2n^2\epsilon^2}{\sum_{i=1}^n (b_i - a_i)^2}}.$$

