Non-Standard-Datenbanken und Data Mining

Informationsrecherche (Information Retrieval)

Prof. Dr. Ralf Möller

Universität zu Lübeck

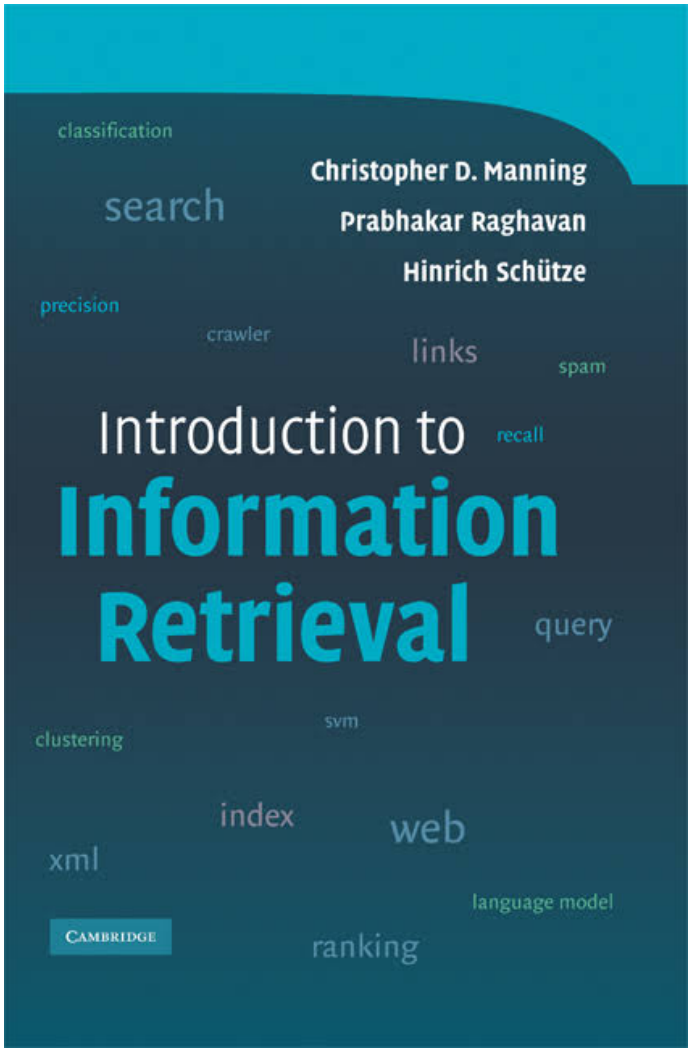
Institut für Informationssysteme

Übersicht

- Semistrukturierte Datenbanken (JSON, XML) und Volltextsuche
- Information Retrieval
- Mehrdimensionale Indexstrukturen
- Cluster-Bildung
- Einbettungstechniken
- Array-Datenbanken
- First-n-, Top-k-, und Skyline-Anfragen
- Probabilistische Datenbanken, Anfragebeantwortung, Top-k-Anfragen und Open-World-Annahme
- Probabilistische Modellierung, Bayes-Netze, Anfragebeantwortungsalgorithmen, Lernverfahren, Verallgemeinerung: Belief Functions, Dempster-Shafer Theorie der Evidenz
- Temporale Datenbanken und das relationale Modell, Zeitreihen in Array-Datenbanken, TimeScaleDB
- Data Mining auf Zeitreihen (SAX, Matrix Product), Probabilistische Temporale Datenbanken
- Dynamische Bayessche Netze, Inferenzalgorithmen und Lernverfahren
- Stromdatenbanken, Prinzipien der Fenster-orientierten inkrementellen Verarbeitung
- Approximationstechniken für Stromdatenverarbeitung, Stream-Mining
- Probabilistische raum-zeitliche Datenbanken und Stromdatenverarbeitungsssysteme: Anfragen und Indexstrukturen, Raum-zeitliches Data Mining
- Von NoSQL- zu NewSQL-Datenbanken, CAP-Theorem, Blockchain-Datenbanken
- Analyse von Graphdaten

Danksagung

- Präsentationen sind aus Vorlesungen zu dem folgenden Buch entnommen:

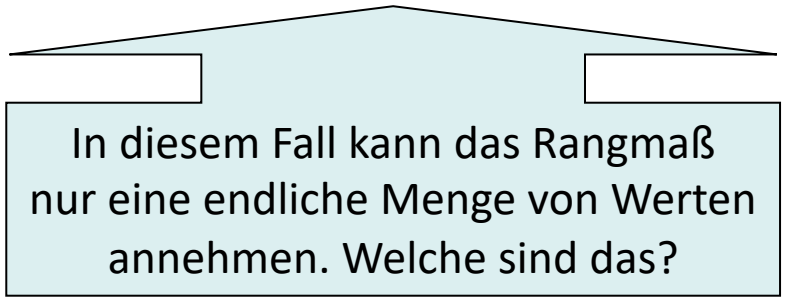


Text-basierte Anfragen

- Finde Dokumente geordnet nach “Nützlichkeit”
- Rangmaß (engl. score) aus [0,1]
  - Für jedes Dokument und jede Anfrage

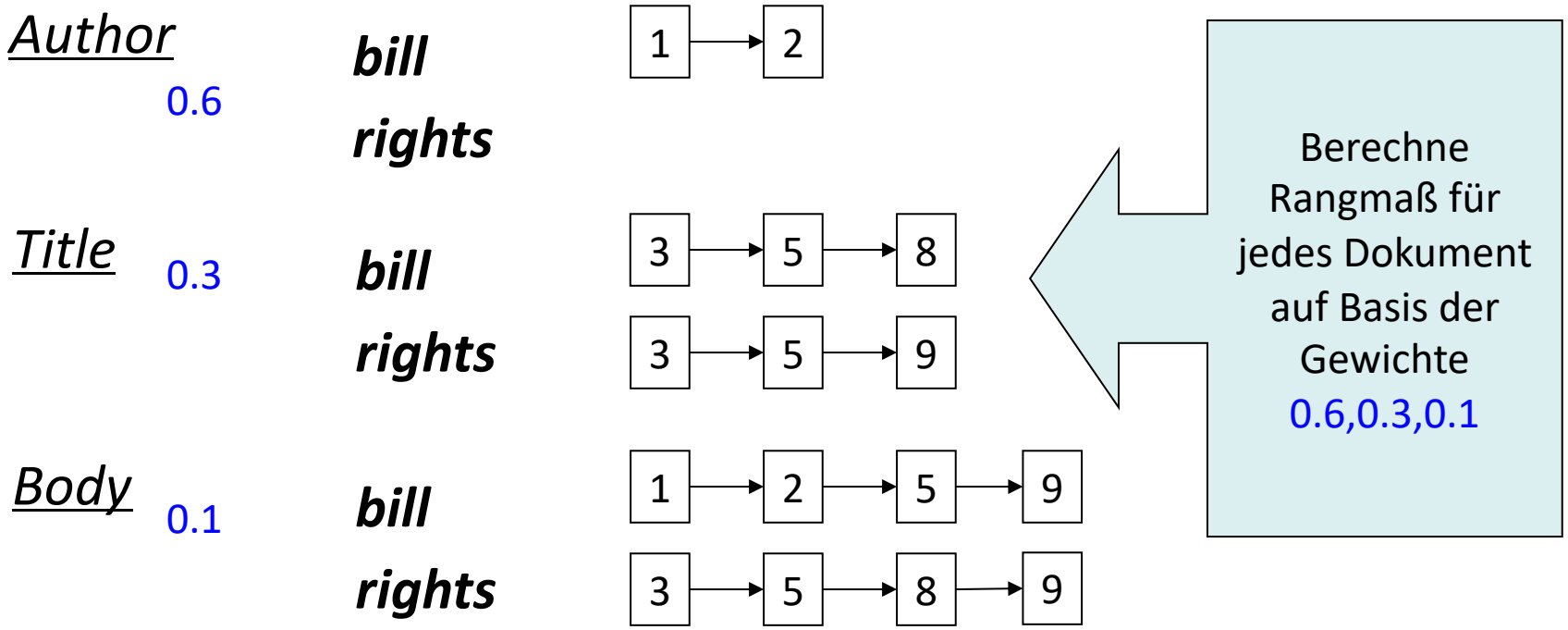
Lineare Kombination von Rangmaßen

- Erste Generation der Rangbestimmung mit verschiedenen Maßen für verschiedene Bereiche:
  - Beispiel für Linearkombination
    0.6*<**"sorting "**in Title> + 0.3*<**"sorting"** in Abstract> +
    0.05*<**"sorting"** in Body> + 0.05*<**"sorting"** in Fettschrift>
  - Jeder Ausdruck wie <**sorting** in Title>
    ergibt einen Wert aus {0,1}
  - Dann liegt der Gesamtwert in [0,1]



Postings-Listen für jeden Bereich

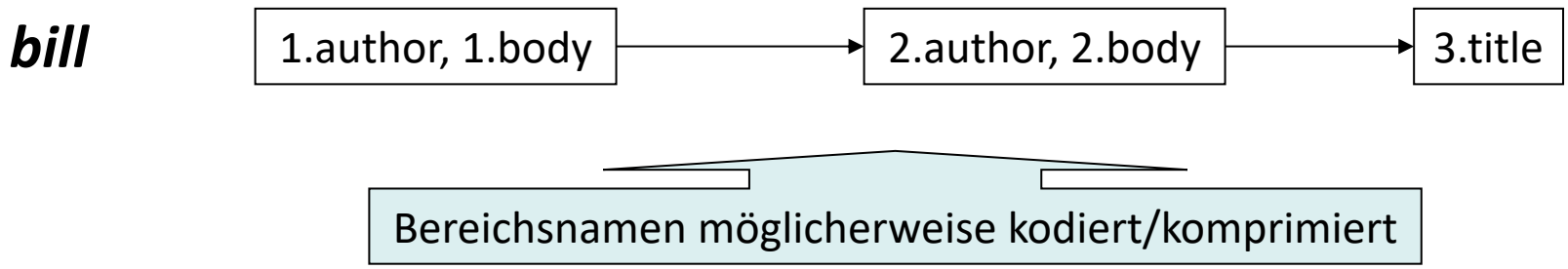
- Für die Anfrage ***bill*** *OR* ***rights*** nehmen wir folgende Listen an:



- Typischerweise ist man nur an den k höchstbewerteten Dokumenten interessiert

Bereichskombinationsindex

- Platz sparen mit Bereichen in den Postings



- Zur Anfragezeit werden Beiträge zum Gesamtmaß eines Dokuments akkumuliert

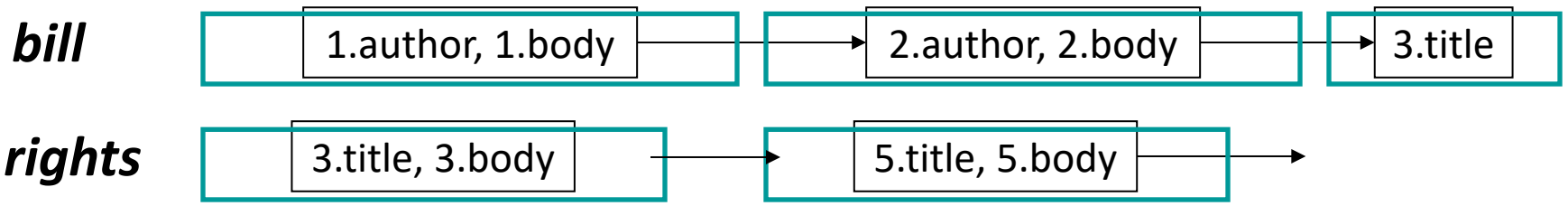
Rangmaßakkumulierung

Author 0.6

Title 0.3

Body 0.1

1	*0.7*
2	*0.7*
3	*0.4*
5	*0.4*



- Anmerkung: ***bill*** und ***rights*** tritt *beides* im Bereich Title von Dokument 3 auf, was sich aber nicht auswirkt
- Sollte hier das Gewicht erhöht werden?

Wo kommen die Gewichte her?

- Gegeben
  - Ein Testkorpus
  - Ein Satz von Testanfragen
  - Relevanzangaben zur den jeweiligen Antworten
- Bestimme einen Satz von Gewichten, so dass Relevanzangaben (besser) passen
- Data Mining nach passende Gewichten:
  - Lösen eines Optimierungsproblems
- Alternative: Bereiche isoliert betrachten
  - Pareto-optimale Lösungen

Mining und QA eng verwoben

Wir kommen darauf zurück

Inzidenzmatrizen und Rangmaße

- Bag-of-words-Modell: Mengen = Bitvektoren
- Dokument (oder ein Bereich darin) als binärer Vektor X in $\{0,1\}^v$
- Anfrage als Vektor Y
- Rangmaß: Überlappungsmaß: $\left|X \cap Y\right| := X \cdot Y$

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0
...	...	...	...	...	...	...

Beispiel

- Anfrage ***ides of march***:
  - Shakespeares *Julius Caesar* hat Rangmaß 3
  - Andere Shakespeare-Stücke haben Rangmaß 2 (für ***march***) oder 1
- In der Rangfolge kommt *Julius Caesar* zuerst

Überlappungsmaß

- Was läuft schief?
- Nicht betrachtet wird:
  - Term-Häufigkeit im Dokument ignoriert
  - Term-Seltenheit in der Sammlung nicht beachtet
    - ***of*** häufiger als ***ides*** oder ***march***
  - Länge von Dokumenten ignoriert
- Normalisierung notwendig

Überlappungsmaß: Normalisierung

- Jaccard-Maß (Mengen bzw. Bitvektoren):

$$|X \cap Y| / |X \cup Y|$$

X: Dokument im Bestand
Y: Anfrage

- „Cosinusmaß“ (nur für Bitvektoren):

$$(X \cdot Y) / \sqrt{|X| \times |Y|}$$

- Diskutiere: Jaccard-Maß für Anfrage mit raren Begriffen und mit häufigen Begriffen. Probleme bei häufigen Begriffen?
- Löst das Cosinusmaß dieses Problem?
- Gewichtung von Termen über Zähler betrachten

Term-Dokument Zählerfelder

- Betrachte Anzahl der Vorkommen eines Terms in einem Dokument:
  - Bag of words-Modell
  - Dokument ist ein Vektor in $\mathbb{N}^v$: eine Spalte in der Matrix

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	157	73	0	0	0	0
Brutus	4	157	0	1	0	0
Caesar	232	227	0	2	1	1
Calpurnia	0	10	0	0	0	0
Cleopatra	57	0	0	0	0	0
mercy	2	0	3	5	5	1
worser	2	0	1	1	1	0

Berechnung des Rangmaßes

- Addition des Maßes für jedes Einzelwort (möglicherweise bereichsgewichtet)
- Betrachten wir die Anfrage **ides of march**
  - Julius Caesar hat 5 Vorkommen von **ides**
  - Kein anderes Stück erwähnt **ides**
  - **march** kommt in dutzenden Dokumenten vor
  - Alle Stücke enthalten **of**
- Durch ein einfaches Zähl-Rangmaß wird das Stück mit den meisten Vorkommen von **of** bestbewertet

Term-Frequenz tf

- Lange Dokumente enthalten per se hohe Zähler
- Normalisierung durch Dokumentlänge
- Verwendung von relativen Häufigkeiten (Term-Frequenz tf)
- Ist das schon ausreichend?

Gewichtete Term-Frequenz

- Anzahl Vorkommen mit linearem Einfluss?
  - 0 vs. 1 Vorkommen eines Terms in einem Dokument
  - 1 vs. 2 Vorkommen
  - 2 vs. 3 Vorkommen …
- Viel hilft viel, aber richtig viel hilft nicht richtig viel mehr

$$wf_{t,d} = 0 \text{ if } tf_{t,d} = 0,\ 1 + \log tf_{t,d} \text{ sonst}$$

- Terme, die selten sind, zeichnen ein Dokument aber gewissenmaßen aus
  - 10 Vorkommen von **hernia** vs
  - 10 Vorkommen von **the**

tf x idf Termgewichtung

- Kombiniertes Maß $w_{i,d}$
  - Term-Frequenz $tf_{i,d}$ (oder gewichtet: $wtf_{i,d}$)
  - Inverse Dokumentfrequenz (idf)
    - Maß der Information:
      Seltenheit eines Terms im Korpus

$$idf_i = \log\left(\frac{n}{df_i}\right)$$

$$w_{i,d} = tf_{i,d} \times \log(n / df_i)$$

TF.IDF-Maß

$tf_{i,d}$ = frequency of term i in document d

n = total number of documents

df_i = the number of documents that contain term i

K. Spärck Jones. A Statistical Interpretation of Term Specificity and Its Application in Retrieval. Journal of Documentation 28: 11–21, **1972**

Siehe auch Kishore Papineni, NAACL 2, **2002**
für theoretische Rechtfertigung

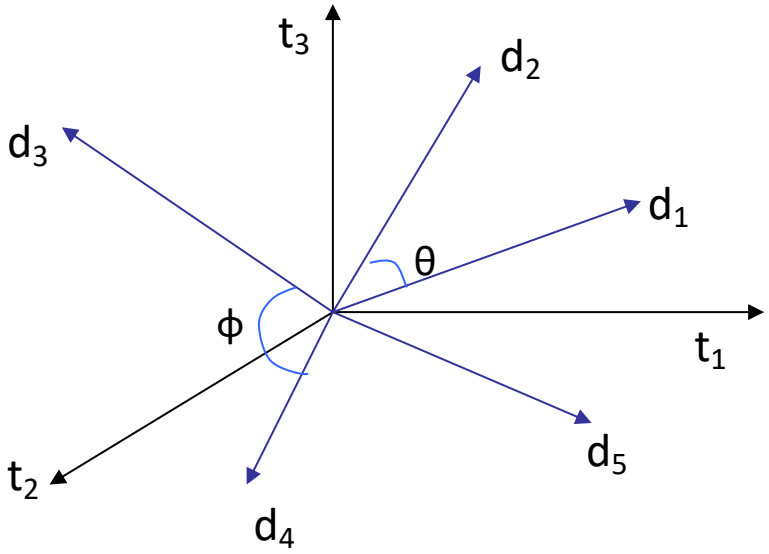
Beispiel mit Tf.idf-Maßen

- Dokument sind Vektoren
- Vektorraum durch Korpus vorgegeben
  - Terme als Achsen (20000+ Dimensionen, selbst mit Wortstämmen)
  - Dokumente als Punkte

NB Maße können >1 sein!

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	13.1	11.4	0.0	0.0	0.0	0.0
Brutus	3.0	8.3	0.0	1.0	0.0	0.0
Caesar	2.3	2.3	0.0	0.5	0.3	0.3
Calpurnia	0.0	11.2	0.0	0.0	0.0	0.0
Cleopatra	17.7	0.0	0.0	0.0	0.0	0.0
mercy	0.5	0.0	0.7	0.9	0.9	0.3
worser	1.2	0.0	0.6	0.6	0.6	0.0

Information Retrieval



- Dokumente die „ähnlich“ sind, sprechen über ähnliche Dinge
- Definition 1 Unähnlichkeit als Euklidischer Abstand der Endpunkte

$$\left|d_j - d_k\right| = \sqrt{\sum_{i=1}^{n}\left(d_{i,j} - d_{i,k}\right)^2}$$

- Normalisierung nötig (Lange der Vektoren = 1), ginge aber
- Je großer die Distanz desto unähnlicher

Cosinusähnlichkeit

- Definition 2: Ähnlichkeit zwischen Dokumenten d_1 und d_2 eingefangen durch Cosinus des Winkels zwischen d_1 und d_2

$$sim(d_j, d_k) = cos(\angle(d_j, d_k)) = \frac{\vec{d}_j \cdot \vec{d}_k}{\left|\vec{d}_j\right|\left|\vec{d}_k\right|} = \frac{\sum_{i=1}^{n} w_{i,j} w_{i,k}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2} \sqrt{\sum_{i=1}^{n} w_{i,k}^2}}$$

- Normalisierung würde Rechenaufwand weiter reduzieren
- Je ähnlicher sich zwei Objekte sind, desto größer ist das Ähnlichkeitsmaß

Anfragen im Vektorraummodell

- Zentrale Idee: Anfrage ist kleines Dokument
- Ergebnis: Dokumente sortiert nach Cosinus des Winkels der zugeordneten Vektoren zum Anfragevektor

$$sim(d_j, d_q) = cos(\angle(d_j, d_{q)}))$$

- Beachte: Vektor d_q ist dünn besetzt!

Siehe auch: Salton, G., *Automatic information organization and retrieval.* New York: McGraw-Hill Book Company. **1968**.

Effiziente Berechnungen: Ausblick

- Finde die k besten Dokumente im Korpus in der Nähe der Anfrage
  - Nächste-Nachbarn-Anfrage
    bzgl. Anfragevektor und Dokumentvektoren
  - Multidimensionale Indexstrukturen + Dimensionsreduktion
  - Top-k-Anfragebeantwortung
- Gruppiere ähnliche Dokumente
  - Clusterbildung
- Nehmen wir an, die Bereiche sollen isoliert betrachtet werden (also keine Linearkombination)
  - Muss man alle Dokumente ansehen?
  - Fagins Algorithmus

Polysemie und Kontext

- Wort hat mehrere Bedeutungen, vom Kontext abhängig
- Beispiel: Pferd = Tier, Turngerät, Schachfigur
- Vektorraummodell unterscheidet Bedeutungen nicht

$$\mathrm{sim}_{\mathrm{true}}(d, q) < \cos(\angle(\vec{d}, \vec{q}))$$

Synonymie und Kontext

- Inhaltliche Übereinstimmung von verschiedenen Wörtern oder Konstruktionen in derselben Sprache
- Beispiel: Geschenk, Mitbringsel
- Fehlende Assoziation im Vektorraummodell

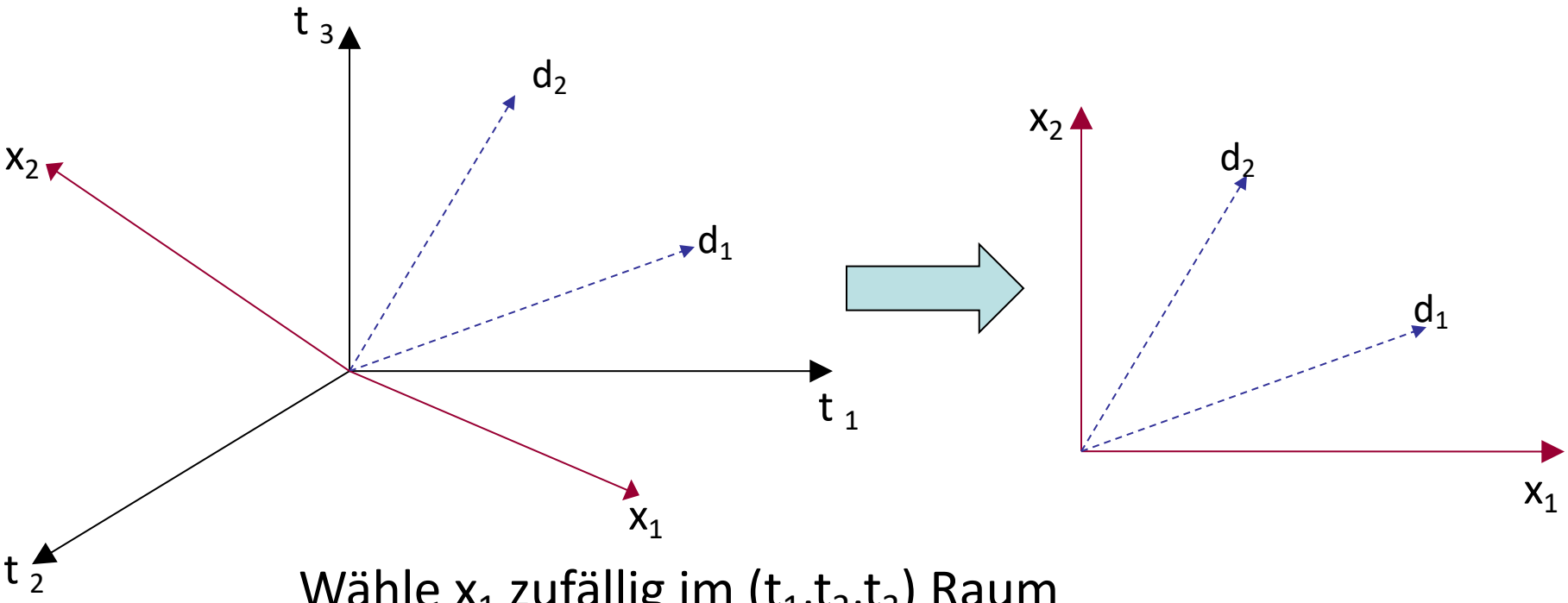
$$\mathrm{sim}_{\mathrm{true}}(d, q) > \cos(\angle(\vec{d}, \vec{q}))$$

- Wenn Bedeutungen verschiedener Worte gleich, wird doch die Verwendung von Worten im Umfeld ähnlich sein
- Wie können wir das einfangen?

Dimensionalitätsreduktion

- Statt m=20000 Terme für Vektoren zu verwenden, Reduktion auf ca. k=100 neue Dimensionen, so dass Ähnlichkeiten beibehalten werden
- 2 Methoden
  - Zufällige Projektion auf $k<<m$ Achsen
  - Latente semantische Indexierung

Beispiel: Projektion von 3 auf 2 Achsen



Wähle x_1 zufällig im (t_1,t_2,t_3) Raum

Wähle x_2 zufällig aber orthogonal zu x_1

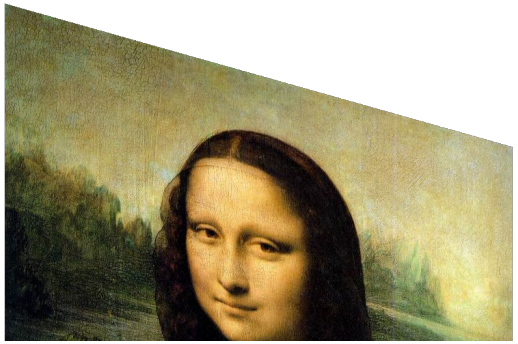
Skalarprodukt von x_1 und x_2 gleich 0

Allgemein: Projektion auf $k<<m$ Achsen

- Wähle zufällige Richtung x_1 im Vektorraum
- For i from 2 to k

  Wähle zufällig eine Richtung x_i orthogonal zu $x_1, x_2, \ldots x_{i-1}$
- Projizieren eines jeden Vektors in den Unterraum aufgespannt durch $\{x_1, x_2, \ldots, x_k\}$
- Mit hoher Wahrscheinlichkeit bleiben relative Distanzen erhalten
- Aber: Relativ aufwendige Berechnungen

Wiederholung: Abbildung von Daten



- Beispiel: Scherung
- Der rote Pfeil ändert sich

Matrixdarstellung [Bearbeiten | Quelltext bearbeiten]

Wählt man in der Ebene ein kartesisches Koordinatensystem, bei dem die x-Achse mit der Achse der Scherung zusammenfällt, dann wird diese Scherung durch die lineare Abbildung

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x + my \\ y \end{pmatrix} = \begin{pmatrix} 1 & m \\ 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} x \\ y \end{pmatrix}$$

mit der Abbildungsmatrix

$$\begin{pmatrix} 1 & m \\ 0 & 1 \end{pmatrix}$$

dargestellt. Ist die Achse der Scherung hingegen die y-Achse, tauschen 0 und m in der Abbildungsmatrix ihre Plätze. Beide Abbildungen verändern den Winkel zwischen den Koordinatenachsen jeweils um $\arctan m$.

[Wikipedia]

Eigenwerte und Eigenvektoren

- Eigenvektoren (für eine quadratische m×m Matrix **S**)

$$\mathbf{S}\mathbf{v} = \lambda\mathbf{v}$$

(rechter) Eigenvektor $\mathbf{v} \in \mathbb{R}^m \neq \mathbf{0}$

Eigenwert $\lambda \in \mathbb{R}$

Beispiel

$$\begin{pmatrix} 6 & -2 \\ 4 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 2 \end{pmatrix} = \begin{pmatrix} 2 \\ 4 \end{pmatrix} = 2 \begin{pmatrix} 1 \\ 2 \end{pmatrix}$$

- Wie viele Eigenwerte gibt es maximal?

$$\mathbf{S}\mathbf{v} = \lambda\mathbf{v} \iff (\mathbf{S} - \lambda\mathbf{I})\,\mathbf{v} = \mathbf{0}$$

Determinante

Hat eine von 0 verschiedene Lösung falls $|\mathbf{S} - \lambda\mathbf{I}| = 0$

Gleichung m-ter Ordnung in λ mit maximal m verschiedenen Lösungen (Nullstellen des charakteristischen Polynoms)
– möglicherweise komplex, obwohl **S** real ist.

Singulärwertzerlegung

Für eine m×n Matrix **A** vom Rang r gibt es eine Faktorisierung (Singulärwertzerlegung, engl. Singular Value Decomposition = **SVD**) wie folgt:

$$A = U\Sigma V^T$$

m×m m×n n×n

Spalten von **U**: links-singuläre Eigenvektoren von $\mathbf{AA^T}$

Spalten von **V**: rechts-singuläre Eigenvektoren von $\mathbf{A^TA}$

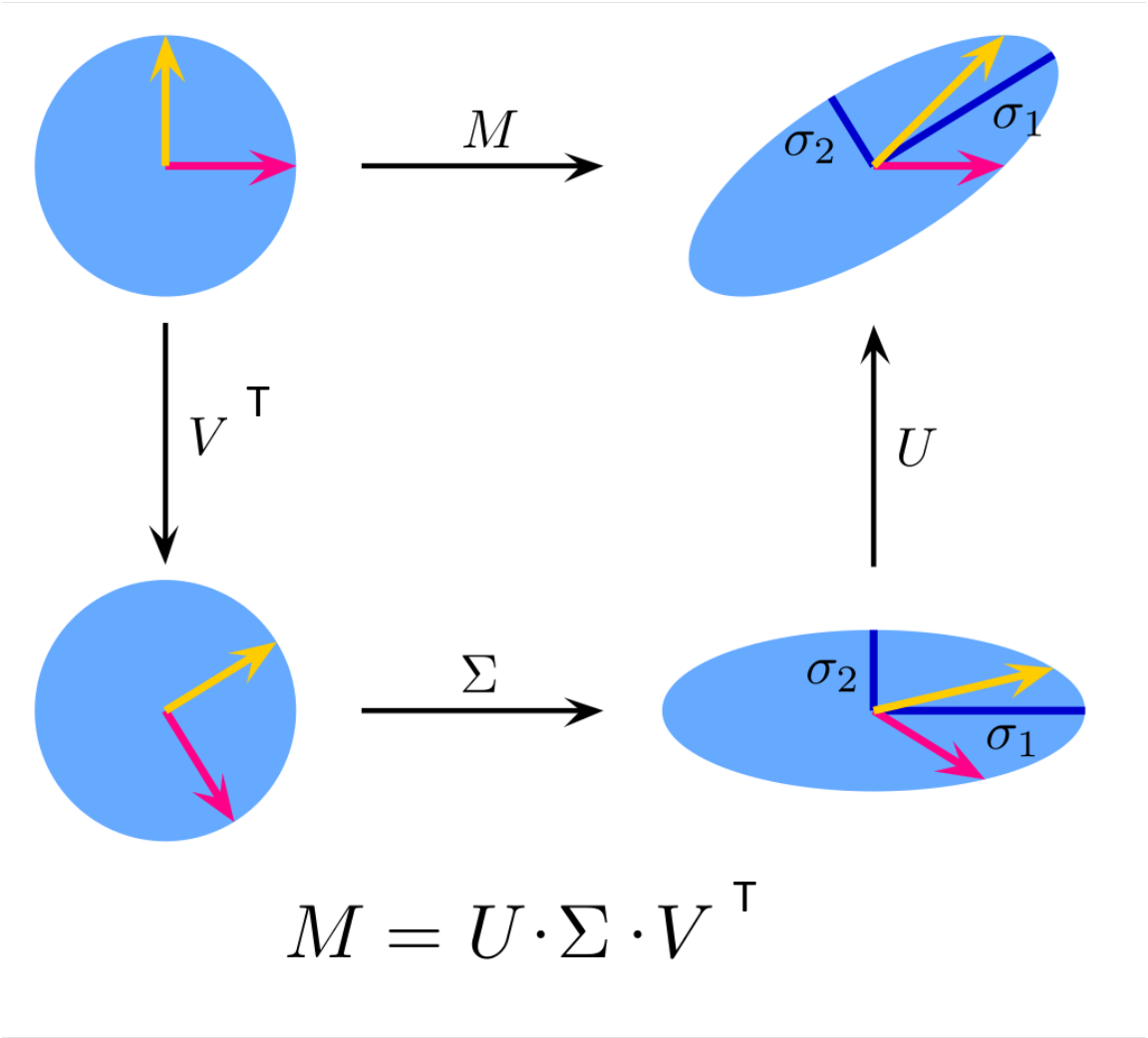
Eigenwerte $\lambda_1 \ldots \lambda_r$ von $\mathbf{AA^T}$ sind Eigenwerte von $\mathbf{A^TA}$

$$\sigma_i = \sqrt{\lambda_i}$$

$$\Sigma = diag(\sigma_1 \ldots \sigma_r)$$

← Singulärwerte

Scherung mit Einheitsvektoren



$$M = U \cdot \Sigma \cdot V^{T}$$

[Wikipedia]

SVD Beispiel

Sei $$A = \begin{bmatrix} 1 & -1 \\ 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Also m=3, n=2. Die SVD ist

$$\begin{bmatrix} 0 & 2/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & -1/\sqrt{6} & 1/\sqrt{3} \\ 1/\sqrt{2} & 1/\sqrt{6} & -1/\sqrt{3} \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & \sqrt{3} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ 1/\sqrt{2} & -1/\sqrt{2} \end{bmatrix}$$

Singulärwerte werden üblicherweise noch in absteigender Reihenfolge angeordnet

Approximation durch Matrix mit kleinem Rang

- SVD kann zur Berechnung einer optimalen Approximation einer Matrix A vom Rang r durch ein Matrix A_k mit kleinerem Rang k verstanden werden

$$A_k = \underset{X:rank(X)=k}{\arg\min} \|A - X\|_F$$

Frobenius-Norm

$$\|A\|_F \equiv \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n} |a_{ij}|^2}.$$

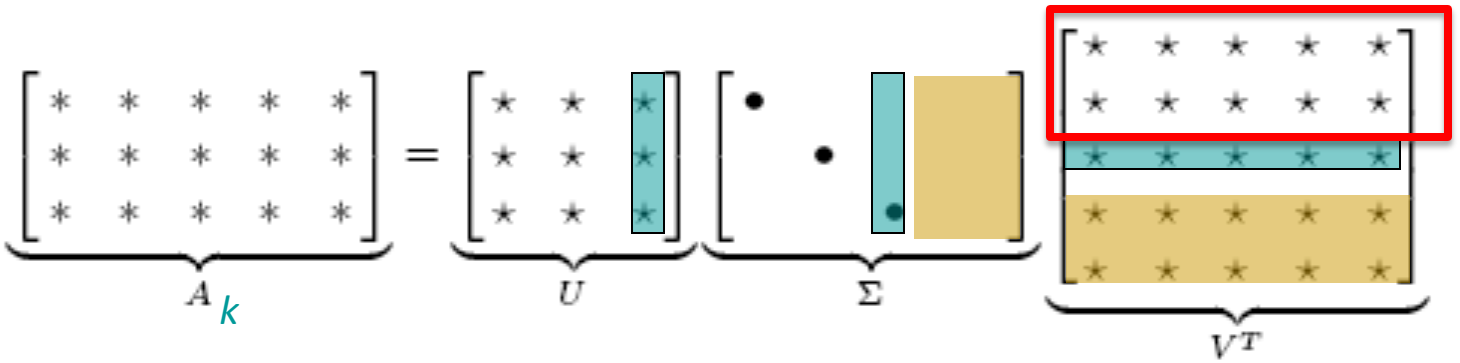
- A_k und X sind beides m×n Matrizen
- Typischerweise k << r

Approximation durch Matrix mit kleinem Rang

- Lösung mittels SVD

$$A_k = U\ \mathrm{diag}(\sigma_1,\ldots,\sigma_k,\underbrace{0,\ldots,0}_{\text{Setze kleinste r-k Eigenwerte auf 0}})V^T$$

Neue Dokumente



C. Eckart, G. Young, The approximation of a matrix by another of lower rank. Psychometrika, 1, 211-218, **1936**

Anwendung zur Informationsrecherche

- Eine Term-Dokument-Matrix kann m=50000, n=10 Millionen Einträge haben (Rang nah bei 50000)
- Wir können eine Approximation A_{100} konstruieren mit Rang 100 und kleinstem Frobenius-Fehler (Hauptkomponentenanalyse, engl. Principle Component Analysis oder PCA)
- Die neue Matrix definiert latente Merkmale (keine verstehbaren Terme mehr) fürdie Informationsrecherche (Latent Semantic Indexing)
- Text to Matrix Generator (TMG) in MATLAB®

Wie behandeln wir Anfragen?

- Eine Anfrage q wird wie folgt in den LSI-Raum abgebildet

$$q_k = q^T U_k \Sigma_k^{-1} T$$

- Anfrage q_k ist nicht (mehr) dünn besetzt

LSI: Zusammenfassung

- Deutliche Reduktion der Vektorraumdimensionen
  - Verringerung des Speicherbedarfs
  - Schnellere Verarbeitung
  - Reduktion von Rauschen
- „Semantische Clusterbildung“
  - Ähnliche Themen auf die gleiche Dimension abgebildet
  - Synonymie und Polysemie besser handhabbar (viele Tests in der Literatur)

Scott Deerwester, Susan Dumais, George Furnas, Thomas Landauer, Richard Harshman: *Indexing by Latent Semantic Analysis.* In: *Journal of the American society for information science.* **1990**.

Landauer, Thomas; Foltz, Peter W.; Laham, Darrell. "Introduction to Latent Semantic Analysis". Discourse Processes. 25 (2–3): 259–284, **1998**.

Anwendungen der Hauptkomponentenanalyse

- Bildverarbeitung
  - Beispiel m Gesichter jeweils als nxn Pixelmatrix
  - Jedes Bild $X = (x_1, \ldots x_{n2})$ Pixel
  - Bilde Matrix M aus m Bildern
  - Reduziere Dimensionen von M: Bilde M_k
  - Finde Bild: Transformiere Bild q nach q_k
  - Bestimme k nächste Nachbarn: passende Bilder
- Spektralanalyse (MS, NMR)
  - Erstelle Referenzspektren R (mit Benennungen für Stoffe)
  - Bilde daraus Matrix R_k
  - Bestimme k nächste Nachbarn zu Q_k, liefere Labels

Externe Evaluierung von Anfrageergebnissen

Precision/Recall (Präzision / Trefferquote)

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

F-measure

The weighted harmonic mean of precision and recall, the traditional F-measure or balanced F-score is:

$$F = 2 \cdot (\text{precision} \cdot \text{recall})/(\text{precision} + \text{recall}).$$

Recall heißt auch Sensitivity

Recherche-Evaluation ohne Rangmaß

- Precision: Anteil der gefundenen Dokumente, die relevant sind = P(retr&rel|retrieved)
- Recall: Anteil der relevanten Dokumente, die gefunden wurden = P(retr&rel|relevant in repos)

	Relevant	Not Relevant
Retrieved	true positives (tp)	false positives (fp)
Not Retrieved	false negatives (fn)	true negatives (tn)

- Precision: $P = tp/(tp + fp)$
- Recall: $R = tp/(tp + fn)$

Evaluationsmaße: Übersicht

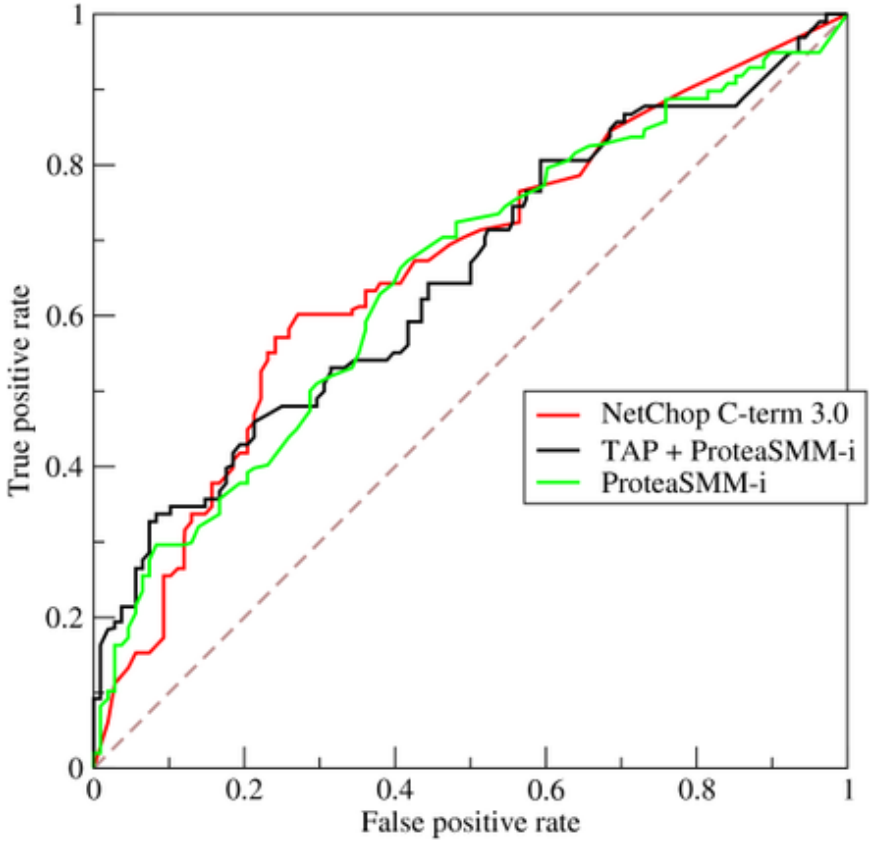
		True condition			
	Total population	Condition positive	Condition negative	Prevalence $= \frac{\Sigma \text{ Condition positive}}{\Sigma \text{ Total population}}$	
Predicted condition	Predicted condition positive	**True positive**	**False positive** (Type I error)	Positive predictive value (PPV), Precision $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Test outcome positive}}$	False discovery rate (FDR) $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Test outcome positive}}$
	Predicted condition negative	**False negative** (Type II error)	**True negative**	False omission rate (FOR) $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Test outcome negative}}$	Negative predictive value (NPV) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Test outcome negative}}$
	Accuracy (ACC) $= \frac{\Sigma \text{ True positive} + \Sigma \text{ True negative}}{\Sigma \text{ Total population}}$	True positive rate (TPR), Sensitivity, Recall $= \frac{\Sigma \text{ True positive}}{\Sigma \text{ Condition positive}}$	False positive rate (FPR), Fall-out $= \frac{\Sigma \text{ False positive}}{\Sigma \text{ Condition negative}}$	Positive likelihood ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic odds ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$
		False negative rate (FNR), Miss rate $= \frac{\Sigma \text{ False negative}}{\Sigma \text{ Condition positive}}$	True negative rate (TNR), Specificity (SPC) $= \frac{\Sigma \text{ True negative}}{\Sigma \text{ Condition negative}}$	Negative likelihood ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$	

[Wikipedia]

Relative Operating Characteristic (ROC)

- Untersuche Effekt von Parametereinstellungen
- Vergleiche TP- und FP-Rate
- Beispiel mit 3 Klassifizierern

- Rangmaß: Area under curve (AUC)



Wikipedia

Wo liegen ggf. Probleme? Konfusionsmatrix

In the example confusion matrix below, of the 8 actual cats, the system predicted that three were dogs, and of the six dogs, it predicted that one was a rabbit and two were cats. We can see from the matrix that the system in question has trouble distinguishing between cats and dogs, but can make the distinction between rabbits and other types of animals pretty well.

Example confusion matrix

	Cat	Dog	Rabbit
Cat	5	3	0
Dog	2	3	1
Rabbit	0	2	11

Ziel: Analyse wo Probleme liegen

Idee: Bei problematischen Anfragen, Relevanz der Ergebnisse erfragen und nachjustieren

Relevanz-Rückkopplung: Rocchio Algorithm

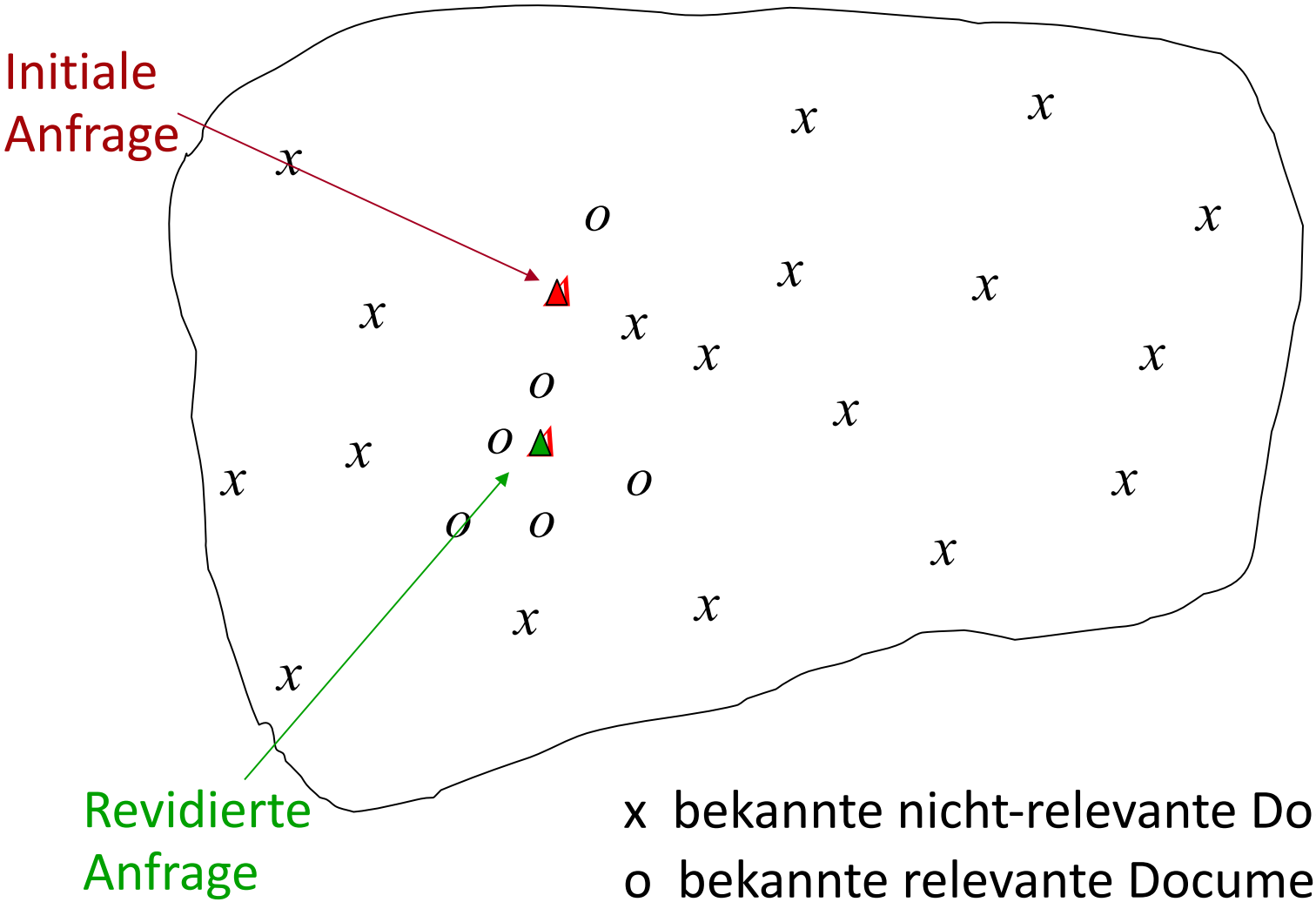
- Relevanzrückkopplung im Vektorraum-Modell
- Wir wollen sim(Q, C_r) - sim(Q, C_{nr}) maximieren, wobei C_r und C_{nr} relevante und nicht relevante Vektoren denotieren
- Der optimale Anfragevektor liegt inmitten der relevanten Vektoren oder sie separiert die nicht relevanten:

$$\vec{Q}_{opt} = \frac{1}{|C_r|} \sum_{\vec{d}_j \in C_r} \vec{d}_j - \frac{1}{N - |C_r|} \sum_{\vec{d}_j \notin C_r} \vec{d}_j$$

Q_{opt} = optimale Anfrage; C_r = Menge relevanter Vektoren im Korpus; N = Korpusgröße

- Leider eine unrealistische Definition: Wir kennen die relevanten Dokumente im Korpus nicht

Beispiel: Relevanzrückkopplung für Anfrage



x bekannte nicht-relevante Documente

o bekannte relevante Documente

Rocchio-Algorithmus (SMART System, 1971)

- In der Praxis:

$$\vec{q}_m = \alpha\vec{q}_0 + \beta\frac{1}{|D_r|}\sum_{\vec{d}_j \in D_r}\vec{d}_j - \gamma\frac{1}{|D_{nr}|}\sum_{\vec{d}_j \in D_{nr}}\vec{d}_j$$

q_m = modifizierter Anfragevektor
q_0 = originaler Anfragevektor
α,β,γ: Gewichte (manuell gesetzt oder empirisch bestimmt)
D_r = Menge von bekannten relevanten Dokumentvektoren
D_{nr} = Menge von bekannten nicht-relevanten Dokumentvektoren

- Neue Anfragen: q_m bewegt sich auf die relevanten Dokumente zu, weg von den nicht relevanten
- Abwägung α vs. β/γ : Falls viele bewertete Dokumente vorhanden, sollte höheres β/γ gewählt werden

Wikipedia: Gerard Salton, The **SMART** (**S**ystem for the **M**echanical **A**nalysis and **R**etrieval of **T**ext or Salton's Magic Automatic Retriever of Text) Information Retrieval System is an information retrieval system, developed at Cornell University in the **1960s**. Many important concepts in information retrieval were developed as part of research on the SMART system, including the vector space model, relevance feedback, and Rocchio algorithm.

Salton, G. (Ed.). *The SMART retrieval system: Experiments in automatic document processing.* Englewood Cliffs, NJ: Prentice-Hall. 1971.

Positive vs Negative Rückkopplung

Positive Rückkopplung wertvoller als negative (also, setze $\gamma < \beta$;
z.B. $\gamma = 0.25$, $\beta = 0.75$).

Viele Systeme erlauben nur positive Rückkopplung (γ=0)