Teilweise-Überwachtes Lernen

Übersicht

- Voll überwachtes Lernen (Klassifikation)
- Teilweises (Semi-) überwachtes Lernen
  - Lernen mit einer kleinen Menge klassifizierter Beispiele und einer großen Menge unklassifizierter Beispiele (LU-Lernen)
  - Lernen mit positiven und unklassifizierten Beispielen (keine klassifizierten negativen Beispiele) (PU-Lernen).

Lernen mit einer kleinen Menge klassifizierter Beispiele und einer großen Menge unklassifizierter Beispiele

LU-Lernen

Unklassifizierte Daten

- Ein Flaschenhals der Klassifikation ist das Klassifizieren einer großen Menge von Beispielen (Datentupel oder Text-Dokumente).
  - Oft manuell
  - Zeitaufwändig
- Kann auch nur eine kleine Menge von klassifizíerten Beispielen zusammen mit einer großen Menge unklassifizierter Beispiele verwendet werden?
- Möglich in vielen Fällen.

Warum sind unklassifizierte Beispiele hilfreich?

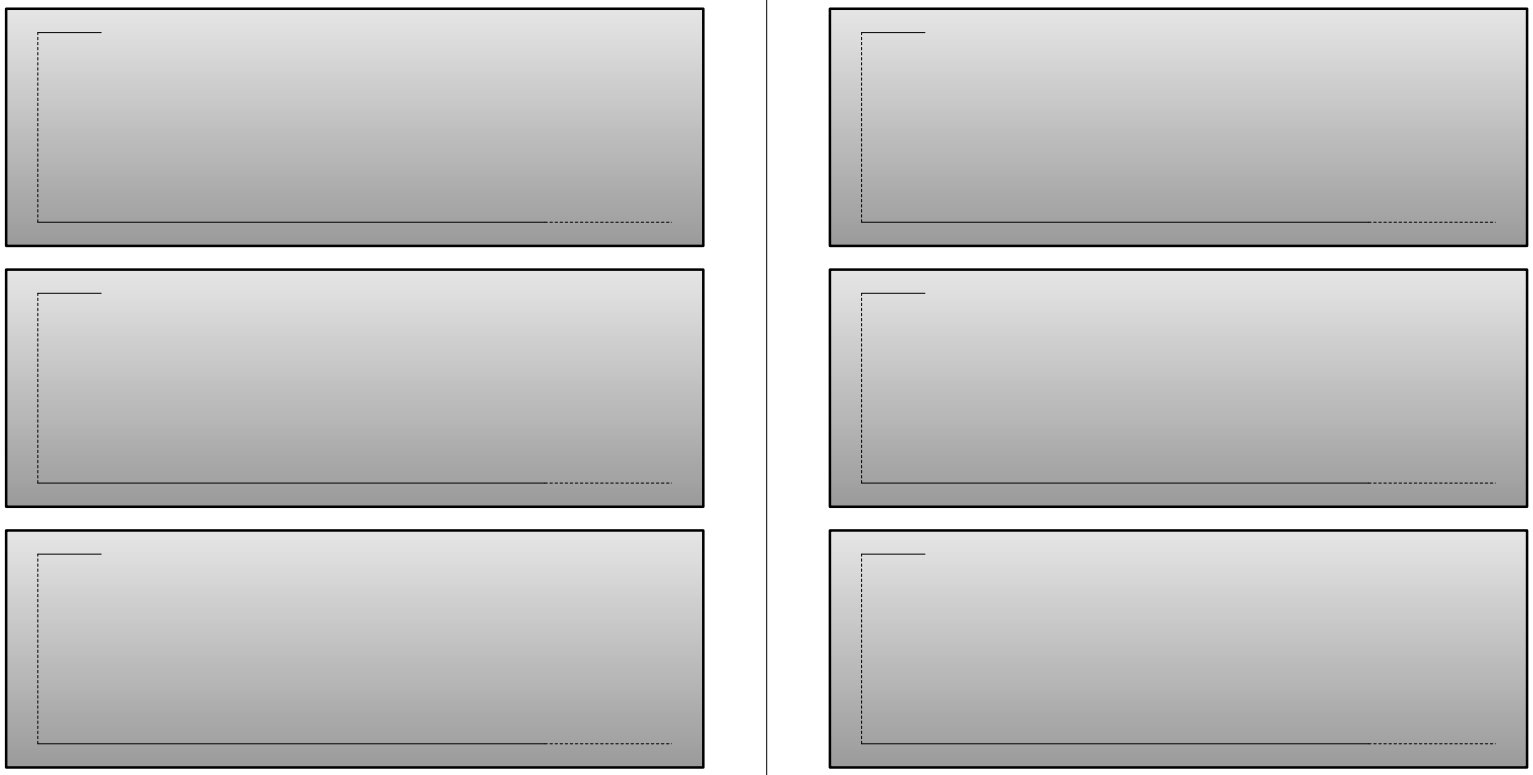
- Unklassifizierte Beispiele sind meist billiger als klassifizierte Beispiele.
- Unklassifizierte Daten enthalten Informationen über die Verbundverteilung von Wörtern und Dokumenten sowie über das gemeinsame Auftreten von Worten. (in Texten).

- Text-Klassifikation als Beispiel für das Problem.

Labeled Data

Unlabeled Data

Documents containing “homework”
tend to belong to the positive class



Wie kann unklassifizierte Daten nutzen

- Eine Möglichkeit ist der EM-Algorithmus
  - EM: Expectation Maximization
- Der EM-Algorithmus ist ein iterative Methode zur Maximum-Likelihood-Schätzung bei Problemen mit fehlenden Daten.
- Der EM-Algorithm besteht aus zwei Schritten,
  - *Expectation-Schritt*, berechnen des Erwartungswertes für fehlende oder versteckte Daten
  - *Maximization-Schritt* – berechnen einen neuen Maximums der Likelyhood bezüglich de Parameter.

Einbeziehen von unklassifizierten Daten im EM (Nigam et al, 2000)

- Einfacher EM
- Erweiterter EM mit gewichteten unklassifizierten Daten
- Ertweiterter EM mit mehreren Misch-Komponenten proKlasse

Algorithmus Übersicht

1. Trainiere einen Klassifikator nur mit den klassifizierten Dokumenten.
2. Dieser wird genutzt um unklassifizierte Dokumente probabilistisch zu klassifizieren.
3. Alle Dokumente werden genutzt um einen neuen Klassifikator zu trainieren.
4. Wiederhole Schritte 2 und 3 bis zur Konvergenz.

Grund-Algorithmus

Algorithm EM(L, U)

1 Learn an initial naïve Bayesian classifier f from only the labeled set L (using Equations (27) and (28) in Chap. 3);

2 **repeat**

// E-Step

3 **for** each example d_i in U **do**

4 Using the current classifier f to compute $\Pr(c_j|d_i)$ (using Equation (29) in Chap. 3).

5 **end**

// M-Step

6 learn a new naïve Bayesian classifier f from $L \cup U$ by computing $\Pr(c_j)$ and $\Pr(w_t|c_j)$ (using Equations (27) and (28) in Chap. 3).

7 **until** the classifier parameters stabilize

Return the classifier f from the last iteration.

Fig. 5.1. The EM algorithm with naïve Bayesian classification

Grund-Algorithmus : E-Schritt & M-Schritt

E-Schritt:

$$\Pr(c_j \mid d_i; \hat{\Theta}) = \frac{\Pr(c_j \mid \hat{\Theta}) \Pr(d_i \mid c_j; \hat{\Theta})}{\Pr(d_i \mid \hat{\Theta})} \tag{29}$$

$$= \frac{\Pr(c_j \mid \hat{\Theta}) \prod_{k=1}^{|d_i|} \Pr(w_{d_i,k} \mid c_j; \hat{\Theta})}{\sum_{r=1}^{|C|} \Pr(c_r \mid \hat{\Theta}) \prod_{k=1}^{|d_i|} \Pr(w_{d_i,k} \mid c_r; \hat{\Theta})},$$

M-Schritt:

$$\Pr(w_t \mid c_j; \hat{\Theta}) = \frac{\lambda + \sum_{i=1}^{|D|} N_{ti} \Pr(c_j \mid d_i)}{\lambda |V| + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} N_{si} \Pr(c_j \mid d_i)}. \tag{27}$$

$$\Pr(c_j \mid \hat{\Theta}) = \frac{\sum_{i=1}^{|D|} \Pr(c_j \mid d_i)}{|D|}. \tag{28}$$

Das Problem

- Der EM-Algorithmus, Abb. 5.1, funktioniert gut, wenn
  - die Mischmodellannahme mit zwei Komponenten (positive und negative Klasse) für die spezielle Datenmenge gilt.
- Diese Annahme kann aber zu Problemen führen, wenn sie verletzt wird. Bei realen Datenmenge kann dies der Fall sein.
- Oft besteht eine Klasse aus weiteren Unterklassen.
  - Zum Beispiel, die Klasse Sport kann Dokumente über verschiedene Unterklassen von Sport enthalten, wie Baseball, Basketball, und Tennis.

Den Einfluss der unklassifizierten Beispiele mit einem Faktor μ gewichten

Neuer M-Schritt:

$$\Pr(w_t \mid c_j) = \frac{\lambda + \sum_{i=1}^{|D|} \Lambda(i) N_{ti} \Pr(c_j \mid d_i)}{\lambda \mid V \mid + \sum_{s=1}^{|V|} \sum_{i=1}^{|D|} \Lambda(i) N_{ti} \Pr(c_j \mid d_i)}, \quad (1)$$

where

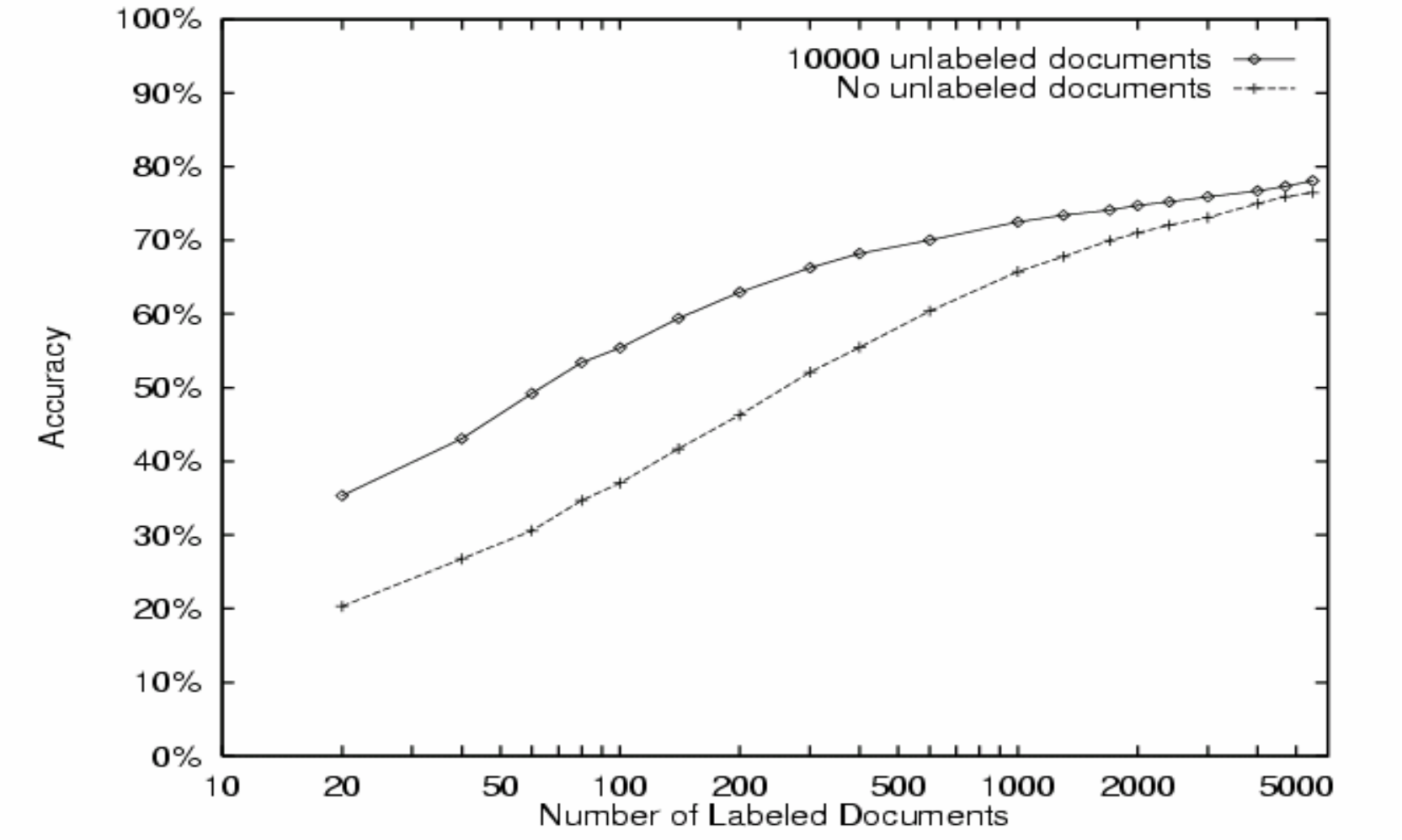
$$\Lambda(i) = \begin{cases} \mu & \text{if } d_i \in U \\ 1 & \text{if } d_i \in L. \end{cases} \quad (2)$$

Die Prior-Wahrscheinlichkeit muss auch gewichtet werden.

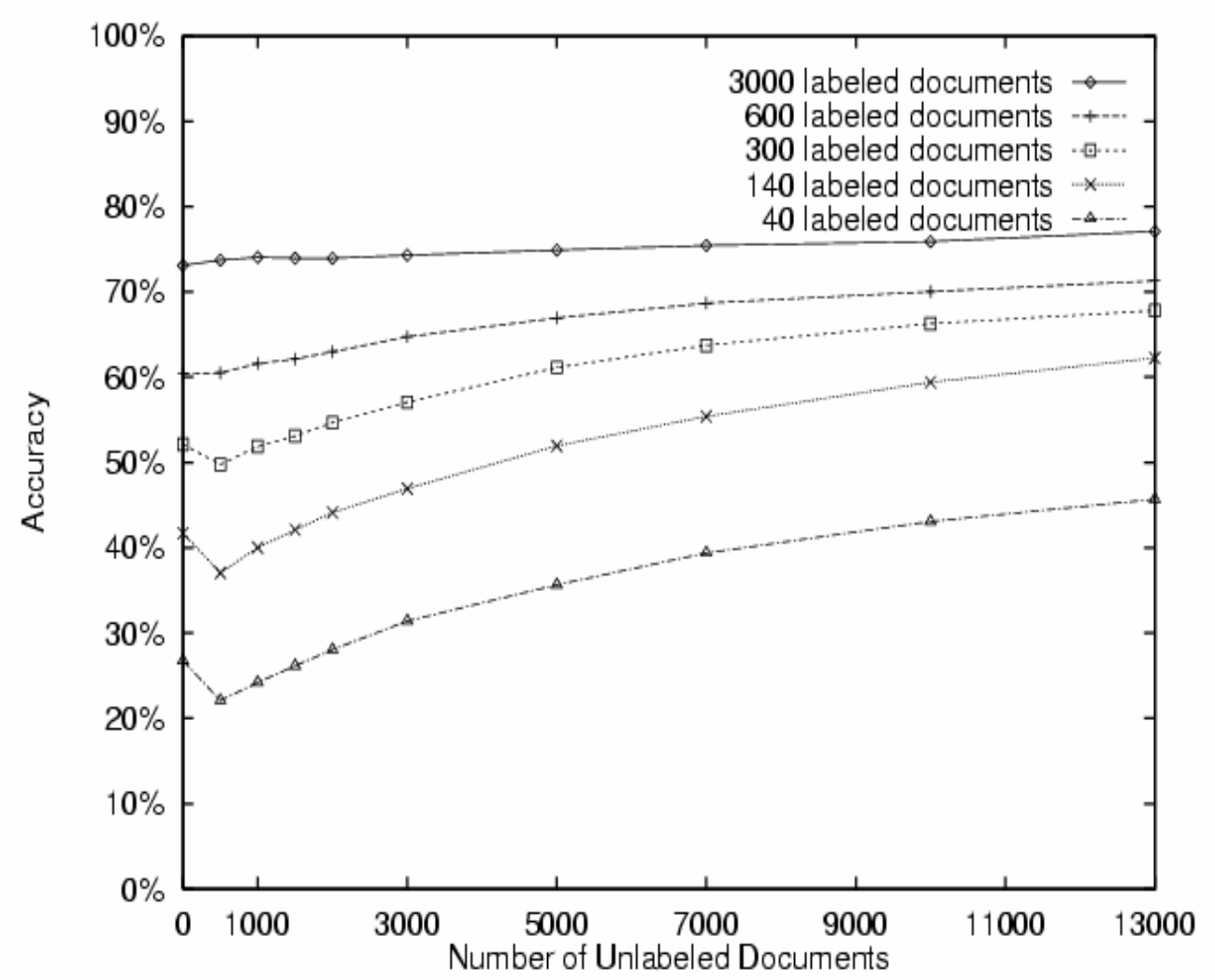
Experimentelle Auswertung

- Newsgroup-Postings
  - 20 Newsgroups, 1000/Gruppe
- Web-Seiten-Klassifikation
  - Student, Dozent, Veranstaltung, Projekt
  - 4199 Web-Seiten
- Reuters Nachrichtenartikel
  - 12,902 Artikel
  - 10 Hauptthemenkategorieen

20 Newsgroups



20 Newsgroups



Anderer Ansatz: Co-Training

- Wieder wird das Lernen mit einer kleinen klassifizierten Menge begonnen.
- Die beschreibenden Attribute werden in zwei Teile partitioniert. Jede Attributteilmenge reicht aus um die Klassifikation zu lernen.
  - z.B., Hyperlinks und Seiteninhalt bei Web-Seiten Klassifikation.
- Zwei Klassifikatoren können auf denselben Daten trainiert werden.

Co-training Algorithmus
[Blum und Mitchell, 1998]

Gegeben: klassifizierte Daten L,

unklassifizierte Daten U

Wiederhole:

Trainiere h1 (z.B., Hyperlink-Klassifikator) mit L

Trainiere h2 (z.B., Seiten-Klassifikator) mit L

Klassifiziere mit h1 Beispiele aus U

Klassifiziere mit h2 Beispiele aus U

Füge die am sichersten klassifizierten Beispiele zu L hinzu, die von h1 und h2 gleich klassifiziert wurden

Co-Training: Experimentelle Ergebnisse

- starte mit 12 klassifizierten Web-Seiten (Academische Veranstaltungen)
- Nutze 1,000 zusätzliche unklassifizierte Web-Seiten
- Durchschnittlicher Fehler: Lernen mit klassifizierten Daten 11.1%;
- Durchschnittlicher Fehler: Co-Training 5.0%

	Page-base classifier	Link-based classifier	Combined classifier
Supervised training	12.9	12.4	11.1
Co-training	6.2	11.6	5.0

Wann ist das Generative Modell nicht passend

- Mehrere Misch-Komponenten proKlasse (M-EM). z.B., eine Klasse – mehrere Unterklassen oder Sub-Cluster.
- Ergebnis für eine Beispiel mit den 20 Newsgroup-Daten
  - 40 klassifizierte; 2360 unklassifizierte; 1600 Test
  - Genauigkeit
    - NB 68%
    - EM 59.6%
- Lösungen
  - M-EM (Nigam et al, 2000): Kreuzvalidierung auf den Trainingsdaten um die Anzahl der Komponenten zu ermitteln.
  - Partitionierter-EM (Cong, et al, 2004): nutzt hierarchisches Clustering. Funktioniert deutlich besser als M-EM.

Zusammenfassung

- Unklassifizierte Daten können die Genauigkeit eines Klassifikators verbessern, wenn die Daten zum generativen Modell passen.
- Partitionierter EM und der EM Klassifikator mit mehreren Mischkomponenten (M-EM) sind besser für reale Daten, wenn mehrere Unterklassen in einer Klasse enthalten sind.
- Co-Training ist eine andere effektive Technik, wenn redundante beschreibende Attribute verfügbar sind.

Lernen mit Positiven und Unklassifizierten Beispielen

PU-Lernen

Lernen mit Positiven und Unklassifizierten Beispielen

- Positive Beispiele: verfügbar ist eine Menge von Beispielen aus der Klasse *P*, und
- Unklassifizierte Beispiele: eine Menge von Beispielen *U* der Klasse unbekannt ist, d.h. es können Positive und negative Beispiele enthalten.
- Erzeugung des Klassifikators: Erzeuge einen Klassifikator um die Beispiele in U und zukünfige Testdaten zu klassifizieren.
- Haupteigenschaft des Problems: keine klassifizierten Beispiele der negativen Klasse verfügbar.
- Problem heißt PU-Lernen.

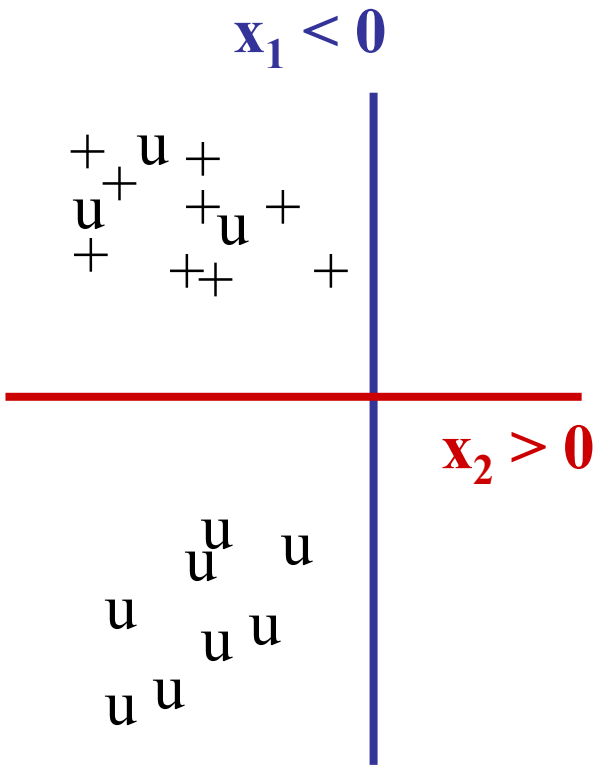
Anwendungen des Problems

- Mit der wachsenden Menge von elektronischen Texten möchte man oft die relevanten Texte zur eigenen Arbeit finden.
- Beispiel, gegeben ICML Tagungsband,
  - finde alle Artikel zu maschinellem Lernen von AAAI, IJCAI, KDD
  - Keine negativen Beispiele aus diesen Sammlungen vorhanden.
- Andere Anwendung
  - Gegeben die Bookmarks (positive Dokumente), identifiziere diejenigen Dokumente die diesen Anwender noch interessieren.

Direct-Marketing

- Firma hat eine Datenbank mit Details der Kunden – positive Beispiele, aber keine Informationen über Menschen, die keine Kunden der Firma sind, keine negativen Beispiele.
- Es sollen Menschen, die ähnlich zu den Kunden sind, gefunden werden
- Ansatz
  - kaufe eine Datenbank mit Details über Menschen, die potentielle Kunden sein könnten – unklassifizierte Beispiele.

Sind Unklassifizierte Beispiele hilfreich?



- Funktion ist entweder $x_1 < 0$ oder $x_2 > 0$
- Welche ist es?

"Nicht lernbar" wenn nur positive Beispiele vorhanden sind. Durch zusätzliche unklassifizierte Beispiele wird das Problem lernbar.

Theoretische Begründung

- (X, Y): X - Eingabevektor, $Y \in \{1, -1\}$ - Klassenattribut.
- f : Klassifikationsfunktion
- Fehlerwahrscheinlichkeit

$$\Pr[f(X) \neq Y] = \Pr[f(X) = 1 \text{ and } Y = -1] + \Pr[f(X) = -1 \text{ and } Y = 1] \quad (1)$$

Es gilt $\Pr[f(X) = 1 \text{ and } Y = -1]$

$= \Pr[f(X) = 1] - \Pr[f(X) = 1 \text{ and } Y = 1]$

$= \Pr[f(X) = 1] - (\Pr[Y = 1] - \Pr[f(X) = -1 \text{ and } Y = 1])$.

Durch Einsetzen in (1) ergibt sich

$$\Pr[f(X) \neq Y] = \Pr[f(X) = 1] - \Pr[Y = 1] + 2\Pr[f(X) = -1 | Y = 1]\Pr[Y = 1] \quad (2)$$

Theoretische Begründung

- $\Pr[f(X) \neq Y] = \Pr[f(X) = 1] - \Pr[Y = 1]$ (2)
  $+ 2\Pr[f(X) = -1 | Y = 1] \Pr[Y = 1]$
- Beachte das $\Pr[Y = 1]$ konstant ist.
- Wenn $\Pr[f(X) = -1 | Y = 1]$ klein ist, dann ist Lernen in etwas das gleiche, wie das Minimieren von $\Pr[f(X) = 1]$.
- Kleinehalten von $\Pr[f(X) = -1 | Y = 1]$ und Mininmieren von $\Pr[f(X) = 1]$ ist in angenähert das gleiche wie
  - Minimieren von $\Pr_u[f(X) = 1]$
  - Während $\Pr_P[f(X) = 1] \geq r$ gehalten wird (wobei r die Ausbaute ist Pr[f(X)=1| Y=1]) was das gleiche ist wie $(\Pr_p[f(X) = -1] \leq 1 - r)$

  wenn die Menge der positiven Beispiele P und die Menge der unklassifizierten Beispiele U groß genug sind.
- Theorem 1 und Theorem 2 in [Liu et al 2002] beweise dies formal für die Fälle ohne und mit Rauschen.

Einfach gesagt

- Bedingtes Optimierungsproblem.
- Ein hinreichend gutes Ergebnis kann erzielt werden wenn
  - der Algorithmus die Anzahl der als positiv klassifizierten unklassifizierten Beispiele mininmiert
  - wird durch die Bedingung erreicht , dass der Anteil der Fehler auf den positiven Beispielen nicht größer als 1-*r* wird

Eine Illustration

- Angenommen ein lineare Klassifikator ist gesucht. Line 3 ist die beste Lösung.

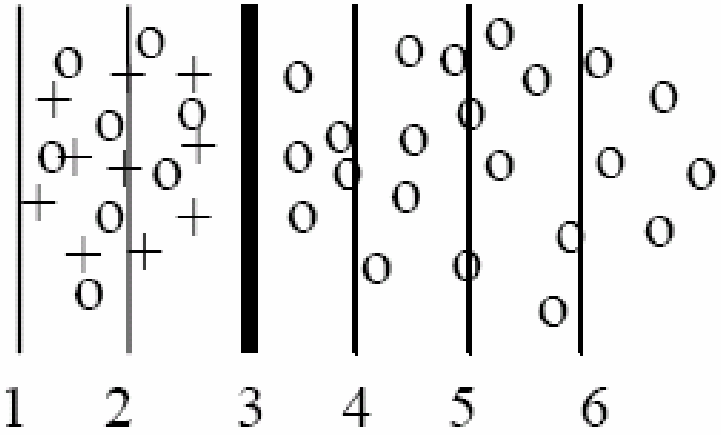
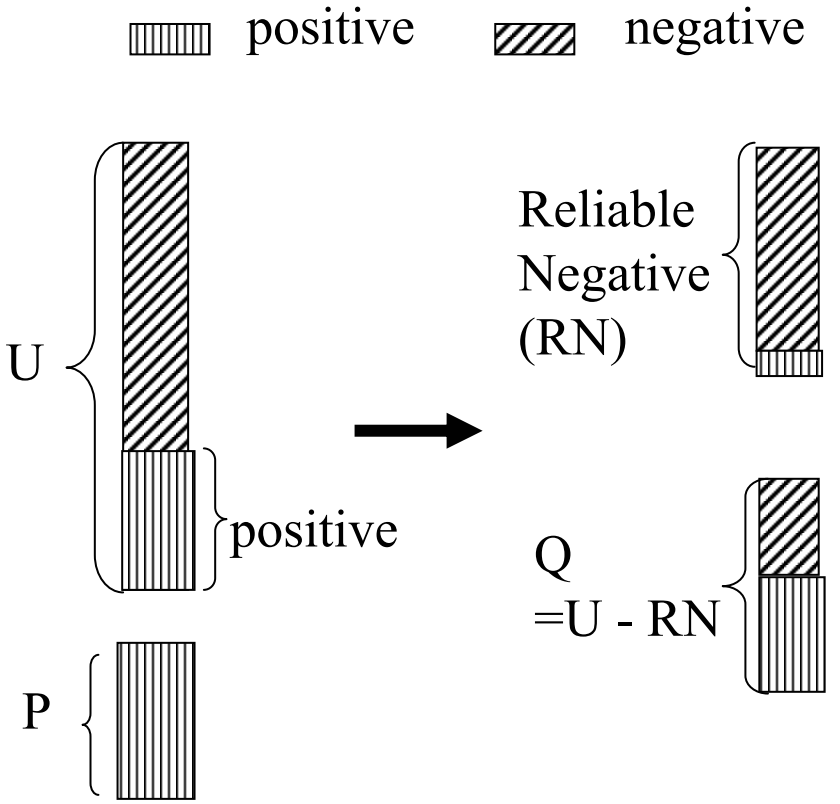


Fig. 5.6. An illustration of the constrained optimization problem

2-Schritt Strategie

- Schritt 1: Identifiziere eine Menge zuverlässlicher negativer Dokumente aus der unklassifizierten Menge.
  - S-EM [Liu et al, 2002] nutzt eine Spion-Technik,
  - PEBL [Yu et al, 2002] nutzt eine 1-DNF-Technik
  - Roc-SVM [Li & Liu, 2003] nutzt den Rocchio-Algorithmus.
  - ...
- Schritt 2: Erzeuge eine Sequenz von Klassifikatoren durch iterative Anwendung eines Lernalgorithmus und selektiere einen guten Klassifikator.
  - S-EM nutzt den Expectation Maximization (EM) Algorithmus mit einem Fehlerbasiertem Klassifikator-Auswahl-Mechanismus
  - PEBL nutzt SVM, keine Klassifikator-Auswahl.
  - Roc-SVM nutzt eine SVM mit einer heuristischen Methode zur Auswahl den endgültigen Klassifikators.

Schritt 1



Schritt 2

Nutzte P, RN und Q um den endgültigen Klassifikator zu iterativ bauen

oder

Nutze nur P und RN um den Klassifikator zu bauen

Schritt 1: Spion Technik

- Wähle zufällig x% positiver Beispiele und füge sie zur unklassifizierten Menge als “Spione”.
- Trainiere einen Klassifikator unter den Annahme alle unklassifizierten Beispiele sind negativ,
  - Das Verhalten der tatsächlich positiven Beispiele in der unklassifizierten Menge wird durch die Spione erkannt.
- Extrahiere zuverlässige negative Beispiele aus der unklassifizierten Menge mit größerer Genauigkeit.

Schritt 1: Andere Methoden

- 1-DNF Methode:
  - Finde eine Menge von Worten W die in den positiven Dokumenten häufiger auftritt als in den unklassifizierten.
  - Extrahierte solche Dokumente aus der unklassifizierten Menge die kein Wort aus W enthalten. Diese Dokumente bilden die zuverlässigen negativen Dokumente.
- Rocchio-Methode aus dem Information Retrieval.
- Naïve Bayes Methode.

Schritt 2: iteratives Anwenden von EM oder SVM

(1) iteratives Anwenden eines Lernalgorithmus

- Wende EM mit P, RN und Q an bis das Verfahren konvergiert, oder
- Wende SVM iterativ an mit P, RN und Q bis kein Dokument aus Q mehr als negativ klassifiziert wird. RN und Q werden in jeder Iteration aktualisiert,
- …

(2) Klassifikator-Auswahl.

Do they follow the theory?

- Yes, heuristic methods because
  - Step 1 tries to find some initial reliable negative examples from the unlabeled set.
  - Step 2 tried to identify more and more negative examples iteratively.
- The two steps together form an iterative strategy of increasing the number of unlabeled examples that are classified as negative while maintaining the positive examples correctly classified.

Can SVM be applied directly?

- Can we use SVM to directly deal with the problem of learning with positive and unlabeled examples, without using two steps?
- Yes, with a little re-formulation.

Support Vector Machines

- Support vector machines (SVM) are linear functions of the form $f(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$, where $\mathbf{w}$ is the weight vector and $\mathbf{x}$ is the input vector.
- Let the set of training examples be $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$, where $\mathbf{x}_i$ is an input vector and y_i is its class label, $y_i \in \{1, -1\}$.
- To find the linear function:

Minimize:
$$\frac{1}{2}\mathbf{w}^T\mathbf{w}$$

Subject to:
$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \ldots, n$$

Soft margin SVM

- To deal with cases where there may be no separating hyperplane due to noisy labels of both positive and negative training examples, the soft margin SVM is proposed:

Minimize: $$\frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\sum_{i=1}^{n}\xi_i$$

Subject to: $$y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, ..., n$$

where $C \geq 0$ is a parameter that controls the amount of training errors allowed.

Biased SVM (noiseless case)

- Assume that the first *k*-1 examples are positive examples (labeled 1), while the rest are unlabeled examples, which we label negative (-1).

Minimize: $\frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C\sum_{i=k}^{n}\xi_i$

Subject to: $\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b \geq 1, \quad i = 1,2,...,k-1$

$$-1(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = k, k+1,...,n$$

$$\xi_i \geq 0,\ i = k,\ k+1...,\ n$$

Biased SVM (noisy case)

- If we also allow positive set to have some noisy negative examples, then we have:

Minimize: $$\frac{1}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w} + C_{+}\sum_{i=1}^{k-1}\xi_i + C_{-}\sum_{i=k}^{n}\xi_i$$

Subject to: $$y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1,2...,n$$

$$\xi_i \geq 0,\ i = 1, 2, ..., n.$$

- This turns out to be the same as the asymmetric cost SVM for dealing with unbalanced data. Of course, we have a different motivation.

Estimating performance

- We need to estimate the performance in order to select the parameters.
- Since learning from positive and negative examples often arise in retrieval situations, we use F score as the classification performance measure $F = 2pr / (p+r)$ (p: precision, r: recall).
- To get a high F score, both precision and recall have to be high.
- However, without labeled negative examples, we do not know how to estimate the F score.

A performance criterion

- Performance criteria *pr*/Pr[Y=1]: It can be estimated directly from the validation set as r^2/Pr[f(X) = 1]
  - Recall *r* = Pr[f(X)=1| Y=1]
  - Precision *p* = Pr[Y=1| f(X)=1]

To see this

Pr[f(X)=1|Y=1] Pr[Y=1] = Pr[Y=1|f(X)=1] Pr[f(X)=1]

$$\Leftrightarrow \quad \frac{r}{\Pr[f(X)=1]} = \frac{p}{\Pr[Y=1]}$$ //both side times r

- Behavior similar to the F-score (= 2*pr* / (*p*+*r*))

A performance criterion (cont …)

- $r^2/\Pr[f(X) = 1]$
- r can be estimated from positive examples in the validation set.
- $\Pr[f(X) = 1]$ can be obtained using the full validation set.
- This criterion actually reflects the theory very well.

Empirical Evaluation

- Two-step strategy: We implemented a benchmark system, called LPU, which is available at http://www.cs.uic.edu/~liub/LPU/LPU-download.html
  - Step 1:
    - Spy
    - 1-DNF
    - Rocchio
    - Naïve Bayesian (NB)
  - Step 2:
    - EM with classifier selection
    - SVM: Run SVM once.
    - SVM-I: Run SVM iteratively and give converged classifier.
    - SVM-IS: Run SVM iteratively with classifier selection
- Biased-SVM (we used SVM$_{light}$ package)

Table 1: Average F scores on Reuters collection

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Step1	1-DNF	1-DNF		1-DNF		Spy	Spy	Spy	Rocchio	Rocchio	Rocchio		NB	NB	NB	NB	
Step2	EM	SVM	PEBL	SVM-IS	S-EM	SVM	SVM-I	SVM-IS	EM	SVM	SVM-I	Roc-SVM	EM	SVM	SVM-I	SVM-IS	NB
0.1	0.187	0.423	0.001	0.423	0.547	0.329	0.006	0.328	0.644	0.589	0.001	0.589	0.547	0.115	0.006	0.115	0.514
0.2	0.177	0.242	0.071	0.242	0.674	0.507	0.047	0.507	0.631	0.737	0.124	0.737	0.693	0.428	0.077	0.428	0.681
0.3	0.182	0.269	0.250	0.268	0.659	0.733	0.235	0.733	0.623	0.780	0.242	0.780	0.695	0.664	0.235	0.664	0.699
0.4	0.178	0.190	0.582	0.228	0.661	0.782	0.549	0.780	0.617	0.805	0.561	0.784	0.693	0.784	0.557	0.782	0.708
0.5	0.179	0.196	0.742	0.358	0.673	0.807	0.715	0.799	0.614	0.790	0.737	0.799	0.685	0.797	0.721	0.789	0.707
0.6	0.180	0.211	0.810	0.573	0.669	0.833	0.804	0.820	0.597	0.793	0.813	0.811	0.670	0.832	0.808	0.824	0.694
0.7	0.175	0.179	0.824	0.425	0.667	0.843	0.821	0.842	0.585	0.793	0.823	0.834	0.664	0.845	0.822	0.843	0.687
0.8	0.175	0.178	0.868	0.650	0.649	0.861	0.865	0.858	0.575	0.787	0.867	0.864	0.651	0.859	0.865	0.858	0.677
0.9	0.172	0.190	0.860	0.716	0.658	0.859	0.859	0.853	0.580	0.776	0.861	0.861	0.651	0.846	0.858	0.845	0.674

Table 2: Average F scores on 20Newsgroup collection

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Step1	1-DNF	1-DNF		1-DNF		Spy	Spy	Spy	Rocchio	Rocchio	Rocchio		NB	NB	NB	NB	
Step2	EM	SVM	PEBL	SVM-IS	S-EM	SVM	SVM-I	SVM-IS	EM	SVM	SVM-I	Roc-SVM	EM	SVM	SVM-I	SVM-IS	NB
0.1	0.145	0.545	0.039	0.545	0.460	0.097	0.003	0.097	0.557	0.295	0.003	0.295	0.368	0.020	0.003	0.020	0.333
0.2	0.125	0.371	0.074	0.371	0.640	0.408	0.014	0.408	0.670	0.546	0.014	0.546	0.649	0.232	0.013	0.232	0.611
0.3	0.123	0.288	0.201	0.288	0.665	0.625	0.154	0.625	0.673	0.644	0.121	0.644	0.689	0.469	0.120	0.469	0.674
0.4	0.122	0.260	0.342	0.258	0.683	0.684	0.354	0.684	0.671	0.690	0.385	0.682	0.705	0.610	0.354	0.603	0.704
0.5	0.121	0.248	0.563	0.306	0.685	0.715	0.560	0.707	0.663	0.716	0.565	0.708	0.702	0.680	0.554	0.672	0.707
0.6	0.123	0.209	0.646	0.419	0.689	0.758	0.674	0.746	0.663	0.747	0.683	0.738	0.701	0.737	0.670	0.724	0.715
0.7	0.119	0.196	0.715	0.563	0.681	0.774	0.731	0.757	0.660	0.754	0.731	0.746	0.699	0.763	0.728	0.749	0.717
0.8	0.124	0.189	0.689	0.508	0.680	0.789	0.760	0.783	0.654	0.761	0.763	0.766	0.688	0.780	0.758	0.774	0.707
0.9	0.123	0.177	0.716	0.577	0.684	0.807	0.797	0.798	0.654	0.775	0.798	0.790	0.691	0.806	0.797	0.798	0.714

Results of Biased SVM

Table 3: Average F scores on the two collections

	γ	Average F score of Biased-SVM	Previous best F score
Reuters	0.3	0.785	0.78
	0.7	0.856	0.845
20Newsgroup	0.3	0.742	0.689
	0.7	0.805	0.774

Summary

- Gave an overview of the theory on learning with positive and unlabeled examples.
- Described the existing two-step strategy for learning.
- Presented an more principled approach to solve the problem based on a biased SVM formulation.
- Presented a performance measure pr/P(Y=1) that can be estimated from data.
- Experimental results using text classification show the superior classification power of Biased-SVM.
- Some more experimental work are being performed to compare Biased-SVM with weighted logistic regression method [Lee & Liu 2003].