Information Retrieval

WS 2016 / 2017

Lecture 12, Tuesday January 24th, 2017

(Linear Classifiers, Perceptrons)

Prof. Dr. Hannah Bast
Chair of Algorithms and Data Structures
Department of Computer Science
University of Freiburg

UNI FREIBURG

Overview of this lecture

- Organizational
  - Experiences with ES 10 — Naïve Bayes
  - Online evaluation for this course — until **January 29**
- Contents
  - Linear classifiers — definition
  - Naïve Bayes (again) — is a linear classifier
  - Perceptrons — another linear classifier
  - **ES12:** prove that Naïve Bayes is a linear classifier

Experiences with ES11 1/2

- Results
  - Overall precision: 73% on genres, 65% on ratings
  - Works well for some classes, not so well for others

    Quite well: Documentary (F1 = 86%), G-Rating (F1 = 79%)

    Not so well: Science Fiction (F1 = 46%), PG-13 (F1 = 38%)
  - Words with largest p_{wc} are mostly stopwords:

    Comedy: the and a film by is of in directed comedy was to ...

    R: the and a film by is of in was … horror … thriller

    However: they don't really hurt, because they occur for all classes … indeed, removing them change results only little

Experiences with ES11 2/2

- Which precision is considered "good" ?
  - **Baseline 1:** guess label uniformly at random

    Overall precision: 20% for genres, 25% for ratings

    Pr(label predicted correctly) = 1 / #classes
  - **Baseline 2:** always pick label most frequent in training set

    Overall precision: 41% for genres, 51% for ratings

    Works quite well when one class is very frequent
  - **Baseline 3:** pick label c with probability $p_c = |T_c| / |T|$

    This is not better than Baseline 2, and generally worse

    Think about why … this might be an exam question

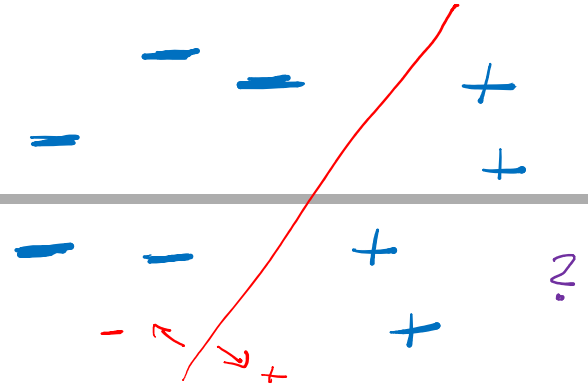
Official course evaluation

- Instructions
  - You should have received an email from EvaSys Admin on Monday, January 16 with a link to an evaluation form
  - We are **very** interested in your feedback
  - Please take your time for this
  - Please be honest and concrete
  - The **free text comments** are most interesting for us

  Please complete by Sunday, January 29

  The evaluation is centralized, and will be closed after that date, and there is nothing we can do about that

Linear Classifiers 1/6



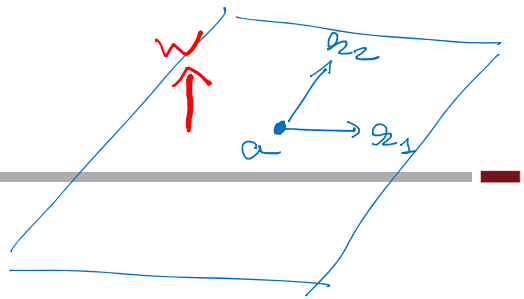
- Framework
  - Objects are (as usual now) vectors in d dimensions
  - Exactly two classes … often denoted +1 and –1

    See slide 10 for how to generalize to more classes
  - A linear classifier tries to separate the data points by a (d-1)-dimensional hyperplane, as defined on next slide

    For d=2 this means: try to separate by a straight line

    Note that the points may not be fully separable
  - Predictions are made based on which side of the hyperplane (for d=2: straight line) the object lies on

Linear Classifiers 2/6



- Hyperplane, definitions
  - Two common ways to define a hyperplane H in $\mathbf{R}^d$

    The two definitions are equivalent … proof on next slide
  - **Definition 1** (by anchor point and basis):

    There is an anchor point $a \in \mathbf{R}^d$ and pairwise orthogonal $h_1, \ldots, h_{d-1} \in \mathbf{R}^d$ such that H consists of all linear combinat. $a + \Sigma_i \alpha_i h_i$ for arbitrary $\alpha_1, \ldots, \alpha_{d-1} \in \mathbf{R}$
  - **Definition 2** (by normal vector and offset):

    There is a normal vector $w \in \mathbf{R}^d$ and an offset $b \in \mathbf{R}$ such that H consists of all points $x \in \mathbf{R}^d$ with $w \bullet x = b$

Linear Classifiers 3/6

$x \cdot x = \sum_{i=1}^{d} x_i^2 = |x|^2$

$x = \begin{pmatrix} x_1 \\ \vdots \\ x_d \end{pmatrix}$

- Hyperplane, equivalence of these definitions
  - For the proof we use that for any pairwise orthogonal $x_1, \ldots, x_k \in \mathbf{R}^d$, we can find $x_{k+1}, \ldots, x_d \in \mathbf{R}^d$ such that $x_1, \ldots, x_d$ are pairwise orthogonal

$w \perp z_i \Leftrightarrow w \cdot z_i = 0$

"$\Rightarrow$": $x = a + \sum_{i=1}^{d-1} \alpha_i \cdot z_i$; pick w orthog. to $z_1, \ldots, z_{d-1}$

$w \cdot x = \underbrace{w \cdot a}_{=:b} + \sum_{i=1}^{d-1} \alpha_i \underbrace{w \cdot z_i}_{=0} = b$ ∎

"$\Leftarrow$": $w \cdot x = b$; pick $z_1, \ldots, z_{d-1}$ s.t. $w, z_1, \ldots, z_{d-1}$ pw $\perp$

$\Rightarrow$ can write $x = \sum_{i=1}^{d-1} \alpha_i \cdot z_i + \alpha_d \cdot w = \sum_{i=1}^{d-1} \alpha_i \cdot z_i + a$

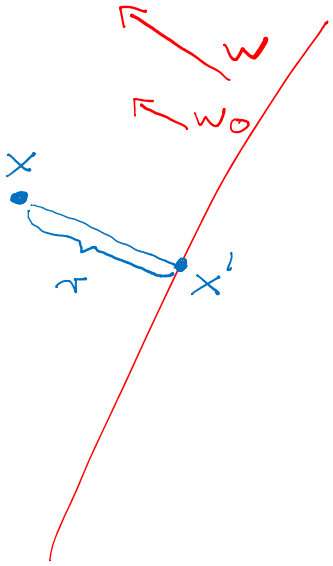
$w \cdot x = \sum_{i=1}^{d-1} \alpha_i \underbrace{w \cdot z_i}_{=0} + \alpha_d \underbrace{w \cdot w}_{=|w|^2} = b$

$\Rightarrow \alpha_d \cdot |w|^2 = b \Rightarrow \alpha_d \cdot |w|^2 \cdot w = w \cdot b$

$\Rightarrow \alpha_d \cdot w = w \cdot b / |w|^2 =: a$

Linear Classifiers 4/6

- Distance from a point to a hyperplane
  - Let $H = \{ x \in \mathbf{R}^d : w \bullet x = b \}$ be a hyperplane in $\mathbf{R}^d$
  - Then the distance of a point $x \in \mathbf{R}^d$ to H is $|w \bullet x - b| / |w|$
  - The sign of $w \bullet x - b$ says on which side of H lies x



$w_0 = w / |w| \Rightarrow |w_0| = 1$ $\qquad w = w_0 \cdot |w|$

$x = x' + r \cdot w_0 \Rightarrow w \bullet x = \underbrace{w \bullet x'}_{=b} + r \cdot \underbrace{w \bullet w_0}_{= r \cdot |w| \cdot \underbrace{|w_0|^2}_{=1}}$

because $x' \in H$

$\Rightarrow w \bullet x = b + r \cdot |w|$

$\Rightarrow r = (w \bullet x - b) / |w|$

This was CASE 1 (x on "right" side of H)

For the other case, $x = x' - r \cdot w_0$

$\Rightarrow \ldots \Rightarrow r = -(w \bullet x - b) / |w|$ ∎

Linear Classifiers 5/6

- Generalization to more classes
  - **Option 1:** Build k classifiers, one for each class, with the i-th one doing the classification: Class i OR not Class i

    Drawback: Need to "vote" when more than one class wins
  - **Option 2:** Build $k \cdot (k - 1) / 2$ classifiers, one for each subset of two classes

    Drawback: For large k, that's a lot of classifiers !
  - **Option 3:** Extend theory of the respective approach to deal with more than two classes directly

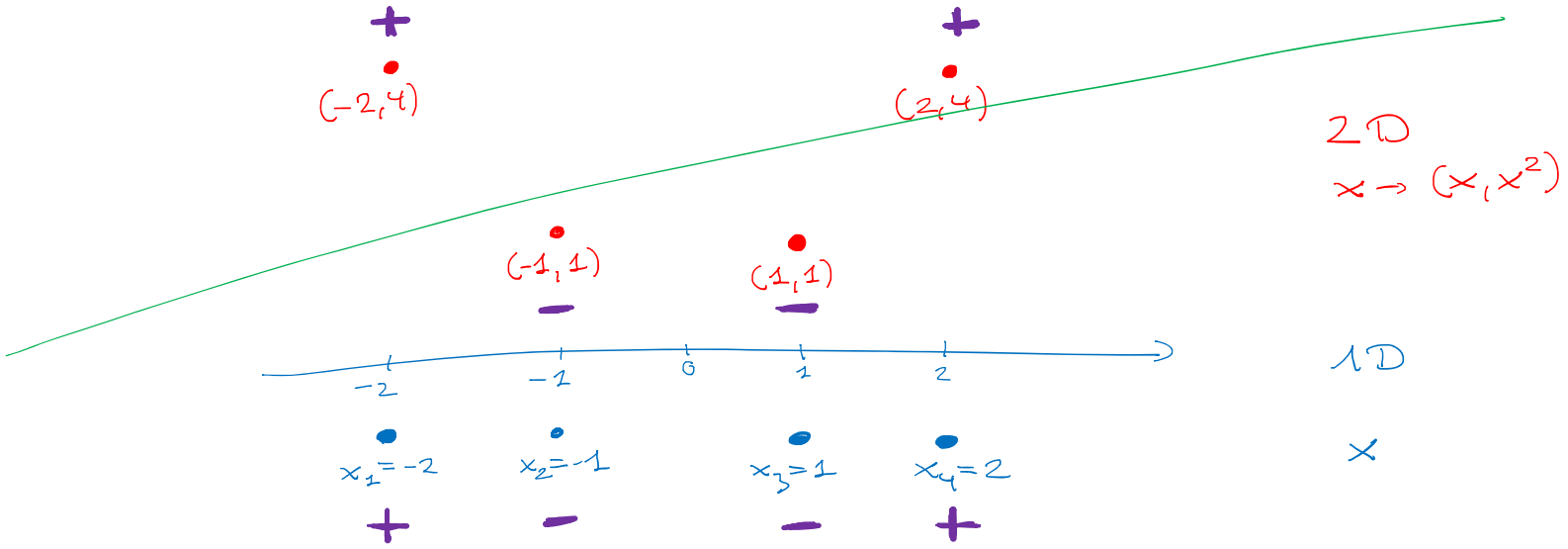
    Drawback: Sometimes hard (not for Naïve Bayes though)

    For ES11 we work with Option 1, but only for one class

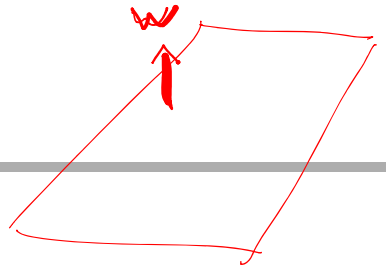
Linear Classifiers 6/6

- When the data is not linearly separable
  - **Option 1:** extend the method to accept "outliers"

    Naïve Bayes does this by definition: it always finds some hyperplane, whether the data is linearly separable or not
  - **Option 2:** suitably transform the data to some higher-dimensional space, where it becomes (better) separable



Naïve Bayes 1/4



- Recap from last lecture
  - Let the vocabulary of all words be $V = \{v_1, \ldots, v_{|V|}\}$

    Beware: in todays lecture, **w** is reserved for normal vectors of hyperplanes, which is why we denote a word by **v** here
  - Recall how NB predicts the probability of a class C for d

    $Pr(C=c \mid D=d) = \Pi_{v_i \text{ in } D} p_{ic} \cdot p_c / Pr(D=d)$

    where $p_{ic} = Pr(W=v_i \mid C=c)$ and the factor is taken multiple times for multiple occurrence of v_i in document d
  - We can equivalently write this as

    $Pr(C=c \mid D=d) = \Pi_{i=1,\ldots,|V|} p_{ic}^{tf_i} \cdot p_c / Pr(D=d)$

    where tf_i is the number of occurrences of v_i in D

Naïve Bayes 2/4

- Two-class NB is a linear classifier
  - Assume our two classes are called A and B, and define $b \in \mathbf{R}$ and $w \in \mathbf{R}^{|V|}$ as follows:

    $b = -\log_2 (p_A / p_B)$, $w_i = \log_2 (p_{iA} / p_{iB})$

    Then NB predicts A if and only if $w \bullet x - b > 0$

    You should prove this yourself in Exercise 11.1

    This is a good exercise for understanding the linear algebra behind linear classifiers. It's not hard, but you have to understand the basic concepts, so perfect exercise :-)

Naïve Bayes 3/4

- Our toy example from Lecture 10
  - Let us recap the math for a general document

aba	A
baabaaa	A
bbaabbab	B
abbaa	A
abbb	B
bbbaab	B

From the training:

$P_A = P_B = 1/2$

$P_{aA} = \frac{2}{3}$, $P_{bA} = \frac{1}{3}$

$P_{aB} = \frac{1}{3}$, $P_{bB} = \frac{2}{3}$

Now arbitrary document with $\#_a$ times a and $\#_b$ times b

$Pr(C=A \mid D=d) = P_{aA}^{\#_a} \cdot P_{bA}^{\#_b} \cdot P_A / Pr(D=d) = \left(\frac{2}{3}\right)^{\#_a} \cdot \left(\frac{1}{3}\right)^{\#_b} \cdot \frac{1}{2} / Pr(D=d)$

$Pr(C=B \mid D=d) = P_{aB}^{\#_a} \cdot P_{bB}^{\#_b} \cdot P_B / Pr(D=d) = \left(\frac{1}{3}\right)^{\#_a} \cdot \left(\frac{2}{3}\right)^{\#_b} \cdot \frac{1}{2} / Pr(D=d)$

$Pr(C=A \mid D=d) > Pr(C=B \mid D=d) \iff \left(\frac{2}{3}\right)^{\#_a} \cdot \left(\frac{1}{3}\right)^{\#_b} > \left(\frac{1}{3}\right)^{\#_a} \cdot \left(\frac{2}{3}\right)^{\#_b}$

$\iff \left(\frac{2}{3}\right)^{\#_a - \#_b} > \left(\frac{1}{3}\right)^{\#_a - \#_b} \iff 2^{\#_a - \#_b} > 1$

$\iff \#_a - \#_b > 0 \iff \#_a > \#_b$

Naïve Bayes 4/4

$P_A = P_B = \frac{1}{2}$

$P_{aA} = \frac{2}{3}$, $P_{bA} = \frac{1}{3}$

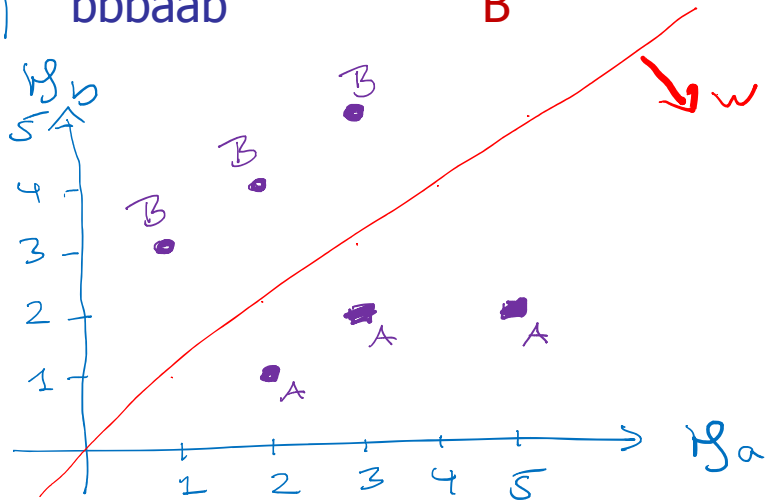
$P_{aB} = \frac{1}{3}$, $P_{bB} = \frac{2}{3}$

- Our toy example from Lecture 10
  - Now express geometrically, in terms of w and b

(y_a, y_b)		
(2,1)	aba	A
(5,2)	baabaaa	A
(3,5)	bbaabbab	B
(3,2)	abbaa	A
(1,3)	abbb	B
(2,4)	bbbaab	B

$$w = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} \log_2(P_{aA}/P_{bA}) \\ \log_2(P_{aB}/P_{bB}) \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

$$b = -\log_2(P_A/P_B) = 0$$



CORRECTION:
The formula for w_1, w_2 above is wrong, the one on slide 13 is correct (the result above is correct anyway though)

Perceptrons 1/6

UNI FREIBURG

- Intuition
  - A perceptron is a linear classifier that **iteratively** computes the w and b that define the hyperplane used for prediction
  - The w and b are greedily improved in each iteration
  - If the training set is linearly separable, the algorithm provably terminates

Perceptrons 2/6

one round = one pass over all training objects (in random order)

- Algorithm
  - Initialization: set $w = 0$ (all-zero vector) and $b = 0$
  - Then iterate over the objects from the training set in random order, and for each object x do the following:

    If $w \bullet x - b$ gives the right prediction, do nothing

    If $w \bullet x - b$ gives the wrong prediction, update w and b:
    - if $w \bullet x - b \leq 0$: $w \leftarrow w + x$ and $b \leftarrow b - 1$
    - if $w \bullet x - b \geq 0$: $w \leftarrow w - x$ and $b \leftarrow b + 1$
  - Repeat until no more change in w or fixed no. of rounds

    For ES11, find out a good termination condition yourself

Perceptrons 3/6

$w \cdot x - b > 0 \Rightarrow A$

$w \cdot x - b < 0 \Rightarrow B$

- First few iterations on our toy example

$x_1 = (2,1)$	aba	A
$x_2 = (5,2)$	baabaaa	A
$x_3 = (3,5)$	bbaabbab	B
$x_4 = (3,2)$	abbaa	A
$x_5 = (1,3)$	abbb	B
$x_6 = (2,4)$	bbbaab	B

$w = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \quad b = 0$

$w \cdot x_1 - b = \begin{pmatrix} 0 \\ 0 \end{pmatrix}\begin{pmatrix} 2 \\ 1 \end{pmatrix} - 0 = 0$

$\Rightarrow$ UPDATE (towards A) $w \leftarrow w + x = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, \; b \leftarrow b - 1 = -1$

$w \cdot x_2 - b = \begin{pmatrix} 2 \\ 1 \end{pmatrix}\begin{pmatrix} 5 \\ 2 \end{pmatrix} + 1 = 13 > 0$

$\Rightarrow$ NO CHANGE

$w \cdot x_3 - b = \begin{pmatrix} 2 \\ 1 \end{pmatrix}\begin{pmatrix} 3 \\ 5 \end{pmatrix} + 1 = 12 > 0$

$\Rightarrow$ UPDATE (towards B) $w \leftarrow w - x = \begin{pmatrix} -1 \\ -4 \end{pmatrix}, \; b \leftarrow 0$

and so on ...

Perceptrons 4/6

$x \bullet x = |x|^2$

UNI FREIBURG

- Convergence, intuition
  - In each iteration, w and b are fixed "towards" the right prediction for the x under consideration:

    Assume $w \bullet x - b \leq 0$ when it should be > 0

    Then we update to $w' = w + x$ and $b' = b - 1$

    $= (w + x) \bullet x$

    Then $w' \bullet x - b' = w \bullet x - b + |x|^2 + 1$

    $= \text{before update}$ $\quad > 0$

    This pushes $w' \bullet x - b'$ in the right direction (towards > 0)

    The hope is that the various updates (for the various x) do not cancel each other out, and eventually all predictions are correct on the training set

    That is, a separating hyperplane is found ... if it exists

Perceptrons 5/6

$\begin{matrix} d \text{ dim.} \\ 1 \text{ more dim} \end{matrix} \left\{ \begin{pmatrix} w \\ b \end{pmatrix} \bullet \begin{pmatrix} x \\ -1 \end{pmatrix} = w \bullet x - b \right.$

b is often called "intercept" term

- Convergence, proof sketch
  - Without loss of generality, we can omit the b

    Just add another dimension d+1, and let the value of all objects be -1 in that dimension … then w_{d+1} is like the b
  - Let $w^{(k)}$ be the w after the k-th correction of w
  - Then we can prove that $|w^{(k+1)}| \geq \varepsilon \cdot k$

    That is, |w| increases by a fixed amount for each correction
  - We can also prove that $|w^{(k+1)}| \leq C \cdot \sqrt{k}$

    That is, |w| increases only sublinearly with k
  - This implies that $k \leq C^2 / \varepsilon^2$ … a fixed #iterations suffice

$\varepsilon \cdot k \leq C \cdot \sqrt{k}$

$\Rightarrow \varepsilon^2 \cdot k^2 \leq C^2 \cdot k$

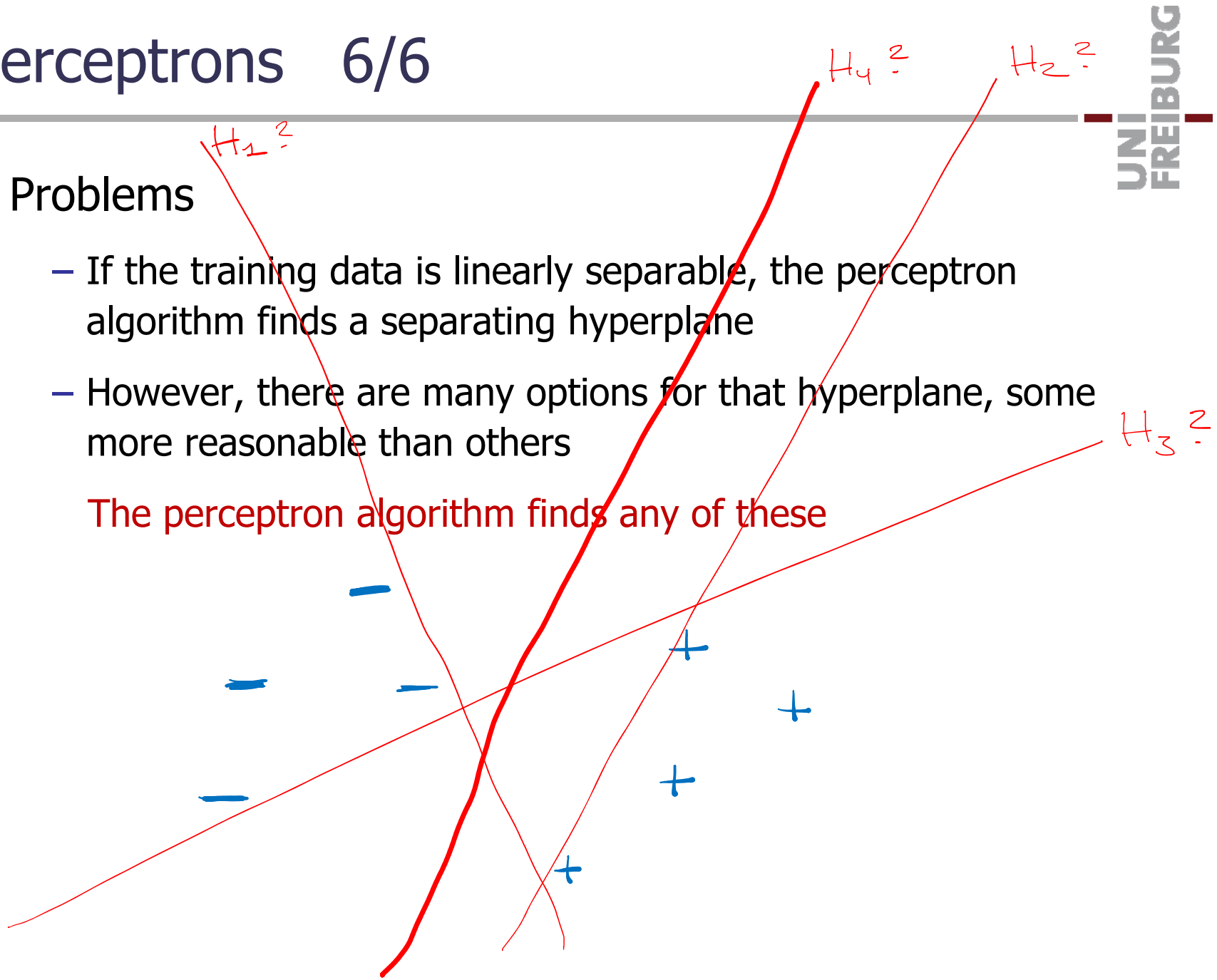
$\Rightarrow \varepsilon^2 \cdot k \leq C^2$

$\Rightarrow k \leq C^2 / \varepsilon^2$ ∎

Perceptrons 6/6

- Problems
  - If the training data is linearly separable, the perceptron algorithm finds a separating hyperplane
  - However, there are many options for that hyperplane, some more reasonable than others

    The perceptron algorithm finds any of these



Perceptron Refinements 1/4

- Refinements we already discussed
  - Change the (pre-determined) number of iterations
  - Terminate when change in precision (on training set) drops below a certain threshold
  - Remove frequent words
  - Use tf.idf instead of tf to represent documents
  - Use different / additional features, e.g. word bigrams

Perceptron Refinements 2/4

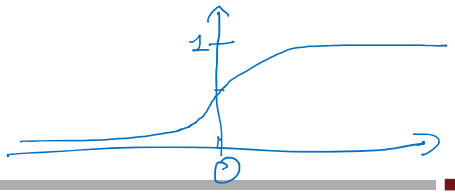
- Averaging
  - Take the average of all w from all iterations ... including all the iterations where w did not change

    That is, if you have 10 iterations and a training set of size 100, you take the average of 1000 w vectors
  - Intuition 1: the final changes to w are due to relative few documents (which are still misclassified)

    Averaging de-emphasizes the w vectors from the end
  - Intuition 2: good values of w are not changed for many iterations (where they classify elements correctly)

    Averaging emphasizes those "good" w vectors

Perceptron Refinements 3/4



sigmoid function

w.l.o.g b=0 (incorporated into w) see L11

- Logistic Regression
  - Let $S(t) = 1 / (1 + e^{-t})$ … then $S(w \bullet x)$ can be interpreted as the probability that x is classified as $+1$
  - We can now try to find the w such that the observed data is most likely … another instance of MLE
  - This gives the following refined update step:

    Class of x is $+1$: $w \leftarrow w + \alpha \cdot a \cdot x$

    Class of x is -1 : $w \leftarrow w - \alpha \cdot a' \cdot x$

    where $a = 1 - S(w \bullet x)$ and $a' = S(w \bullet x)$ and α is a tuning parameter (the so-called learning rate)

Perceptron Refinements 4/4

- Batching
  - Given a w, consider a whole batch B of training elements

    The size of the batch is a parameter to play around with
  - For each $x_i \in B$ compute update term with respect to w

    Simple Perceptron: $+ x$ if class is $+1$, $-x$ otherwise

    "Logistic" Perceptron: $+ \alpha \cdot a \cdot x$ … or … $- \alpha \cdot a' \cdot x$
  - Then add all the update terms to w to obtain a new w

    Batching mainly improves performance (a lot), but it also affects the precision (since it leads to a different w)

References

- Wikipedia
  - http://en.wikipedia.org/wiki/Linear_classifier
  - http://en.wikipedia.org/wiki/Perceptron
  - https://en.wikipedia.org/wiki/Logistic_regression