

Information Retrieval

WS 2015 / 2016

Lecture 11, Tuesday January 19th, 2016
(Linear Classifiers, Perceptrons, SVMs)

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

Overview of this lecture

■ Organizational

- Your experiences with ES 10 Naïve Bayes

■ Contents

- Linear classifiers definition
- Naïve Bayes (again) is a linear classifier
- Perceptrons another linear classifier
- Support Vector Machines a better linear classifier
- **Exercise Sheet 11:** prove that Naïve Bayes is a linear classifier + implement a perceptron + compare to NB

Experiences with ES10 1/3

■ Summary / excerpts

- Interesting + fun exercise sheet
- "Linear Algebra (LA) was surprisingly useful"
- "Surprised how fast and simple the LA solution was" ... **YES !**
- "LA is not the problem, but the libraries are"
- "Numpy und Scipy sind doof"
- "Wrote all the code on my own, because easier that way"

■ 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:

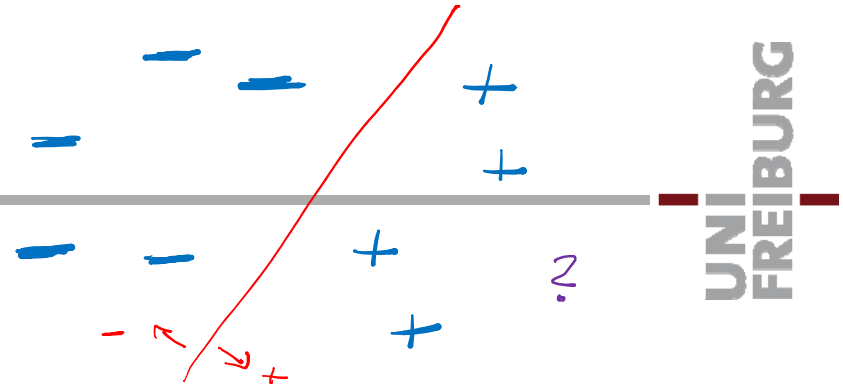
Comedy: the and a film by is of in directed comedy was to ...

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 ES10 3/3

- Which precision is considered "good" ?
 - **Baseline 1:** guess label uniformly at random
Overall precision: 20% for genres, 25% for ratings
 $\text{Pr}(\text{label predicted correctly}) = 1 / \#\text{classes}$
 - **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



■ 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

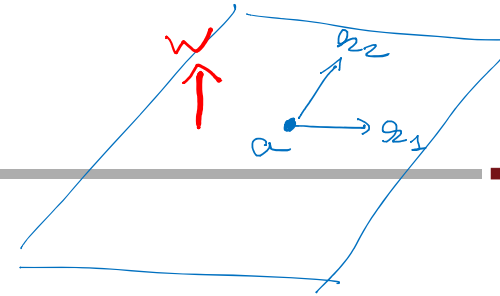
- A linear classifier tries to separate the data points by a $(d-1)$ -dimensional hyperplane, as defined on next slide

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 $h_1, \dots, h_{d-1} \in \mathbf{R}^d$ such that H consists of all linear combinat. $a + \sum_i \alpha_i h_i$ for arbitrary $\alpha_1, \dots, \alpha_{d-1} \in \mathbf{R}$

- **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 $x_1, \dots, x_k \in \mathbb{R}^d$, we can find $x_{k+1}, \dots, x_d \in \mathbb{R}^d$ such that x_1, \dots, x_d are pairwise orthogonal

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

" \Rightarrow " : $x = a + \sum_{i=1}^{d-1} \alpha_i g_i$; pick w orthog. to g_1, \dots, g_{d-1}

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

" \Leftarrow " : $w \cdot x = b$; pick g_1, \dots, g_{d-1} s.t. w, g_1, \dots, g_{d-1} are pairwise orthogonal

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

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

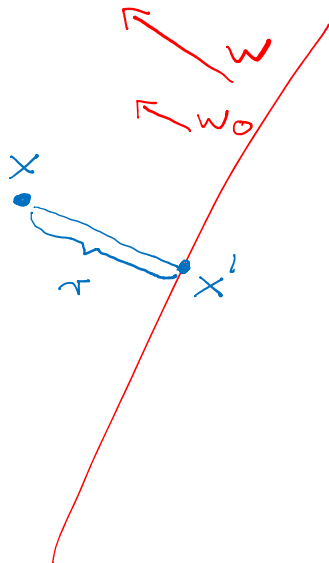
$$\Rightarrow \alpha_d \cdot \|w\|^2 = b \Rightarrow \alpha_d \cdot \|w\|^2 \cdot w = w \cdot b$$

$$\Rightarrow \alpha_d \cdot w = \underbrace{w \cdot b / \|w\|^2}_{=: a}$$

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



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

$$w = w_0 \cdot |w|$$

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

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, $x = x' - r \cdot w_0$

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

■ 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

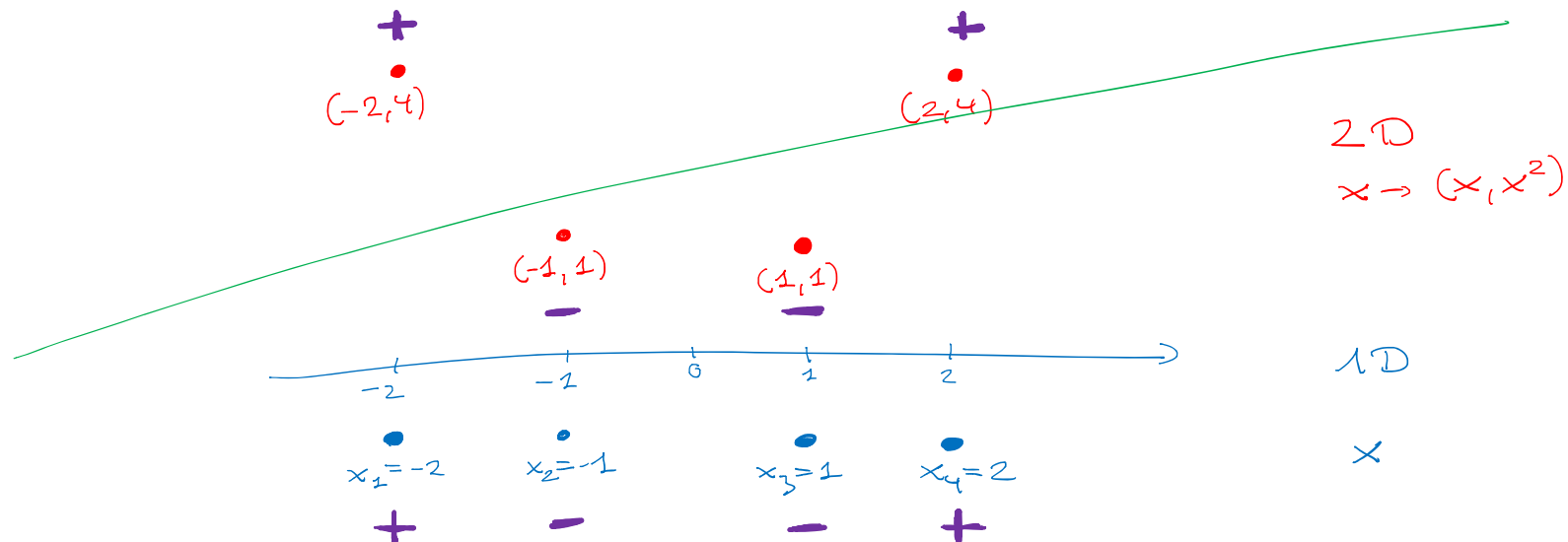
Linear Classifiers 6/6

■ When the data is not linearly separable

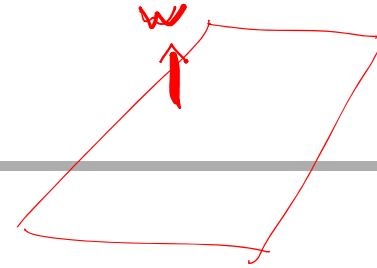
- **Option 1:** extend the method to accept "outliers"

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 $V = \{v_1, \dots, v_{|V|}\}$

Beware: in today's 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

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

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

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

where tf_i is the number of occurrences of v_i in D

- 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:

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

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:

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

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

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

Now arbitrary document
 H_a times a and H_b times b

$$\Pr(C=A|D=d) = P_{aA}^{H_a} \cdot P_{bA}^{H_b} \cdot P_A / \Pr(D=d) = \left(\frac{2}{3}\right)^{H_a} \cdot \left(\frac{1}{3}\right)^{H_b} \cdot \frac{1}{2} / \Pr(D=d)$$

$$\Pr(C=B|D=d) = P_{aB}^{H_a} \cdot P_{bB}^{H_b} \cdot P_B / \Pr(D=d) = \left(\frac{1}{3}\right)^{H_a} \cdot \left(\frac{2}{3}\right)^{H_b} \cdot \frac{1}{2} / \Pr(D=d)$$

$$\Pr(C=A|D=d) > \Pr(C=B|D=d) \iff \left(\frac{2}{3}\right)^{H_a} \cdot \left(\frac{1}{3}\right)^{H_b} > \left(\frac{1}{3}\right)^{H_a} \cdot \left(\frac{2}{3}\right)^{H_b}$$

$$\iff \left(\frac{2}{3}\right)^{H_a - H_b} > \left(\frac{1}{3}\right)^{H_a - H_b} \iff 2^{H_a - H_b} > 1$$

$$\iff H_a - H_b > 0 \iff H_a > H_b$$

Naïve Bayes 4/4

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

$$P_{a|A} = \frac{2}{3}, P_{b|A} = \frac{1}{3}$$

$$P_{a|B} = \frac{1}{3}, P_{b|B} = \frac{2}{3}$$

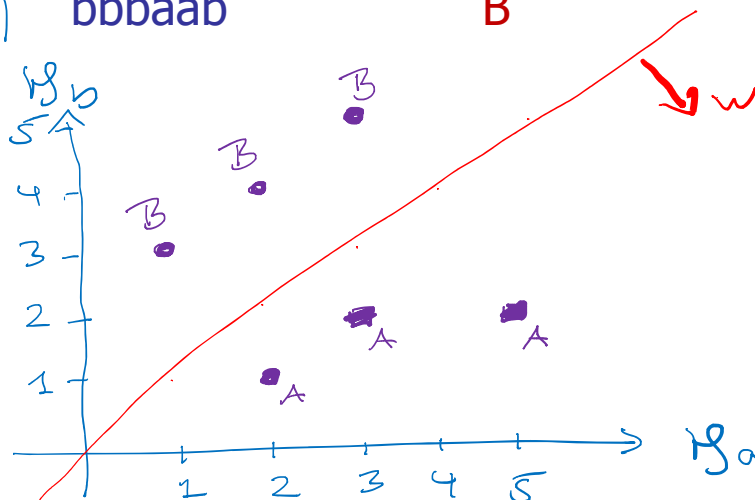
■ Our toy example from Lecture 10

– Now express geometrically, in terms of w and b

(H_a, H_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

$$w = \begin{pmatrix} w_1 \\ w_2 \end{pmatrix} = \begin{pmatrix} \log_2(P_{a|A}/P_{b|A}) \\ \log_2(P_{a|B}/P_{b|B}) \end{pmatrix} = \begin{pmatrix} 1 \\ -1 \end{pmatrix}$$

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



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)

■ 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

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

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

$$\Rightarrow \text{UPDATE } w \leftarrow w + x = \begin{pmatrix} 2 \\ 1 \end{pmatrix}, b \leftarrow b - 1 = -1$$

(towards A)

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

\Rightarrow NO CHANGE

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

$$\Rightarrow \text{UPDATE } w \leftarrow w - x = \begin{pmatrix} -1 \\ -4 \end{pmatrix}, b \leftarrow 0$$

(towards B)

and so on...

$$x \cdot x = |x|^2$$

■ Convergence, intuition

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

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

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

$$\begin{aligned} &= (w+x) \cdot x \\ \text{Then } w' \cdot x - b' &= \underbrace{w \cdot x - b}_{= \text{before update}} + \underbrace{|x|^2 + 1}_{> 0} \end{aligned}$$

This pushes $w' \cdot x - b'$ 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

5/6 $\left. \begin{array}{l} d \text{ dim.} \\ 1 \text{ more dim.} \end{array} \right\} \begin{pmatrix} w \\ b \end{pmatrix} \cdot \begin{pmatrix} x \\ -1 \end{pmatrix} = w \cdot x - b$

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 $|w^{(k+1)}| \geq \epsilon \cdot k$

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 $k \leq C^2 / \epsilon^2$... a fixed #iterations suffice

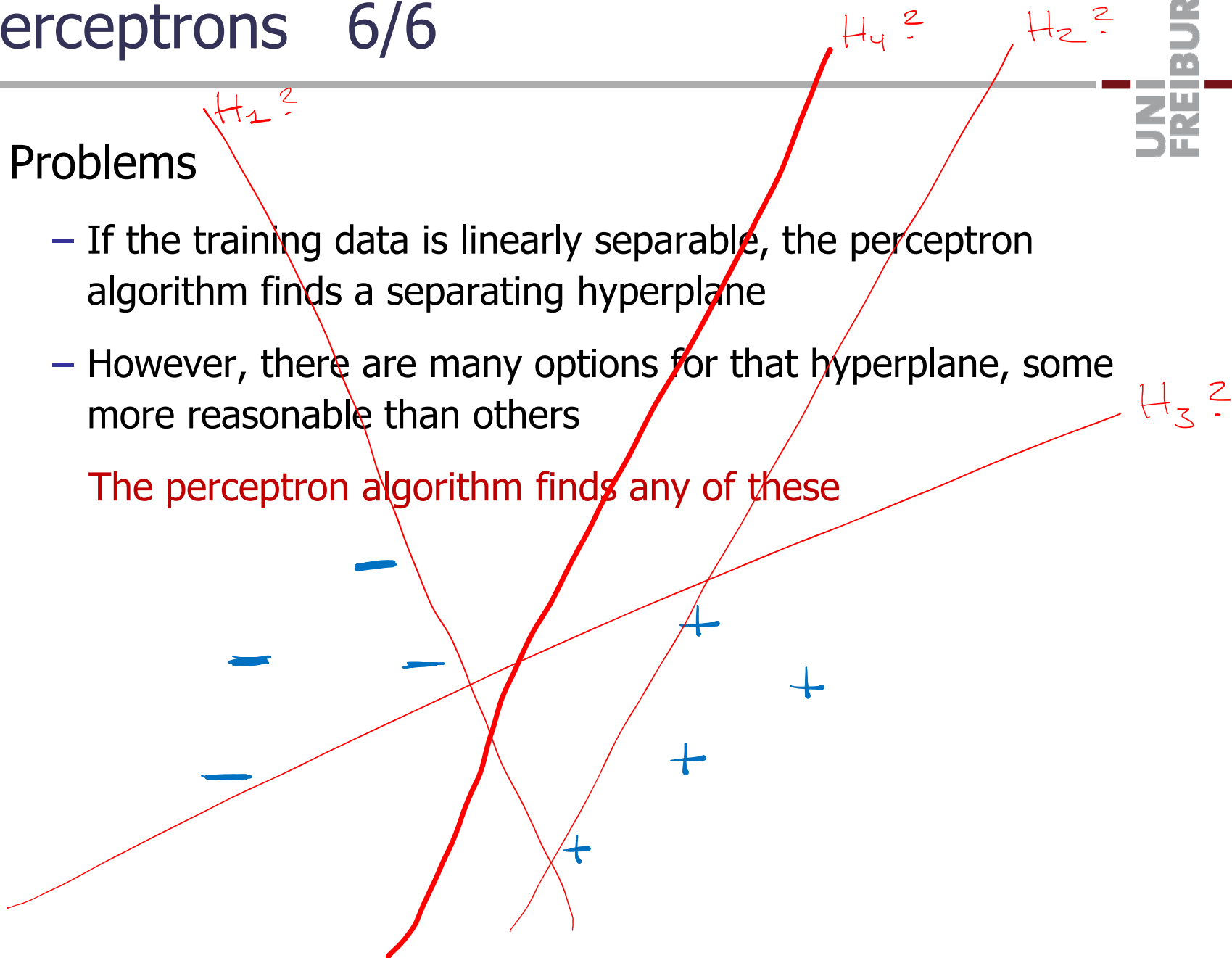
$$\begin{aligned} \epsilon \cdot \mathcal{R} &\leq C \cdot \sqrt{\mathcal{R}} \\ \Rightarrow \epsilon^2 \cdot \mathcal{R}^2 &\leq C^2 \cdot \mathcal{R} \\ \Rightarrow \epsilon^2 \cdot \mathcal{R} &\leq C^2 \\ \Rightarrow \mathcal{R} &\leq \frac{C^2}{\epsilon^2} \quad \square \end{aligned}$$

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

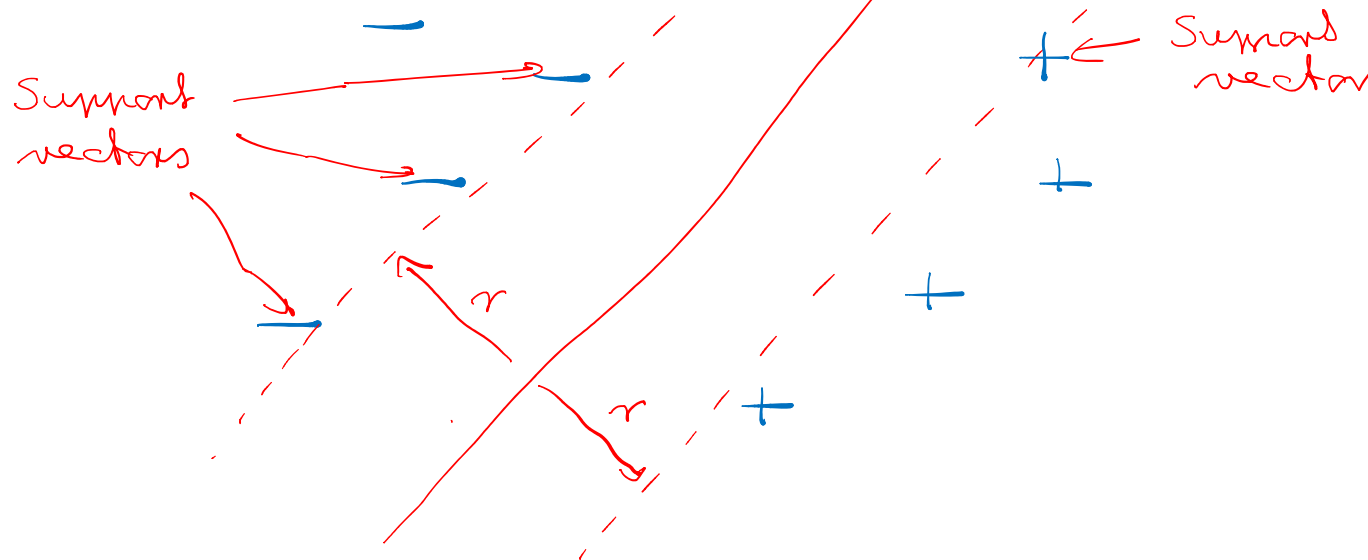
The perceptron algorithm finds any of these



Support Vector Machines 1/3

■ Intuition

- Place the separating hyperplane H such that the symmetric margin around H until the next points is as large as possible
- In \mathbf{R}^2 this means: try to separate the point sets with not just a line, but a "band" of width $2r$, with $r > 0$ as large as possible
- Points on the margin boundary are called **support vectors**



■ Algorithm

- Let $x_1, \dots, x_m \in \mathbf{R}^d$ be the objects from the training set
- Let $y_i = +1$ if x_i is in class A, $y_i = -1$ if x_i is in class B
- Let $H = \{ x \text{ in } \mathbf{R}^d : w \bullet x = b \}$ be a separating hyperplane

Then $\text{dist}(x_i, H) = y_i \cdot (w \bullet x_i - b) / |w|$... see slide 9

- This gives rise to the following maximization problem:

Maximize $2r$, such that $y_i \cdot (w \bullet x_i - b) / |w| \geq r$ for all i

- This can be formulated equivalently as

Minimize $|w|^2$, such that $y_i \cdot (w \bullet x_i - b) \geq 1$ for all i

Proof omitted, see slides from WS 13/14 if you are interested

Support Vector Machines 3/3

- We now have a quadratic optimization problem
 - The $|w|^2 = w \bullet w$ is a **quadratic** objective function
 - The $y_i \cdot (w \bullet x_i - b) \geq 1$ are **linear** constraints

There are established methods for this kind of optimization problem, but which are out scope for this lecture

This is similar to the SVD in Lecture 8, where we resorted to third-party software (numpy and scipy) to solve it

For ES11, we will stick with perceptrons, which are easy to implement from scratch without advanced methods

Some basic linear algebra will be helpful again though

References

- Wikipedia

- http://en.wikipedia.org/wiki/Linear_classifier
- <http://en.wikipedia.org/wiki/Perceptron>
- http://en.wikipedia.org/wiki/Support_vector_machine

- Textbook

- Chapter 15: Support vector machines
<http://nlp.stanford.edu/IR-book/pdf/15svm.pdf>