Statistical Machine Learning

https://cvml.ist.ac.at/courses/SML_W18

Christoph Lampert

IST AUSTRIA

Institute of Science and Technology

Spring Semester 2018/2019

Lecture 3

Overview (tentative)

Date		no.	Topic
Oct 08	Mon	1	A Hands-On Introduction
Oct 10	Wed	–	self-study (Christoph traveling)
Oct 15	Mon	2	Bayesian Decision Theory
			Generative Probabilistic Models
Oct 17	Wed	3	Discriminative Probabilistic Models
			Maximum Margin Classifiers
Oct 22	Mon	4	Generalized Linear Classifiers, Optimization
Oct 24	Wed	5	Evaluating Predictors; Model Selection
Oct 29	Mon	–	self-study (Christoph traveling)
Oct 31	Wed	6	Overfitting/Underfitting, Regularization
Nov 05	Mon	7	Learning Theory I: classical/Rademacher bounds
Nov 07	Wed	8	Learning Theory II: miscellaneous
Nov 12	Mon	9	Probabilistic Graphical Models I
Nov 14	Wed	10	Probabilistic Graphical Models II
Nov 19	Mon	11	Probabilistic Graphical Models III
Nov 21	Wed	12	Probabilistic Graphical Models IV
until Nov 25			final project

Learning from Data

In the real world, $p(x, y)$ is unknown, but we have a training set $\mathcal{D}$. There's at least 3 approaches:

Definition

Given a training set $\mathcal{D}$, we call it

- a **generative probabilistic approach**:
  if we use $\mathcal{D}$ to build a model $\hat{p}(x, y)$ of $p(x, y)$, and then define

  $$c(x) := \operatorname*{\mathbf{argmax}}_{y \in \mathcal{Y}} \hat{p}(x, y) \quad \text{or} \quad c_\ell(x) := \operatorname*{\mathbf{argmin}}_{y \in \mathcal{Y}} \mathop{\mathbb{E}}_{\bar{y} \sim \hat{p}(x, \bar{y})} \ell(\,\bar{y}, y\,).$$

- a **discriminative probabilistic approach**:
  if we use $\mathcal{D}$ to build a model $\hat{p}(y|x)$ of $p(y|x)$ and define

  $$c(x) := \operatorname*{\mathbf{argmax}}_{y \in \mathcal{Y}} \hat{p}(y|x) \quad \text{or} \quad c_\ell(x) := \operatorname*{\mathbf{argmin}}_{y \in \mathcal{Y}} \mathop{\mathbb{E}}_{\bar{y} \sim \hat{p}(\bar{y}|x)} \ell(\,\bar{y}, y\,).$$

- a **decision theoretic approach**: if we use $\mathcal{D}$ to directly seach for a classifier c in a hypothesis class $\mathcal{H}$.

Discriminative Probabilistic Models

Observation

Task: spam classification, $\mathcal{X} = \{\text{all possible emails}\}, \mathcal{Y} = \{\texttt{spam}, \texttt{ham}\}$.
What's, e.g., $p(x|\texttt{ham})$?
For every possible email, a value how likely it is to see that email, including:

- all possible languages,
- all possbile topics,
- an arbitrary length,
- all possible spelling mistakes, etc.

This is much more general (and much harder) than just deciding if an email is spam or not!

"When solving a problem, do not solve a more general problem as an intermediate step."
(Vladimir Vapnik, 1998)

Observation

Instead of $p(x, y) = p(x|y)p(y)$, we can also use $p(x, y) = p(y|x)p(x)$. Since $\mathbf{argmax}_y\, p(x, y) = \mathbf{argmax}_y\, p(y|x)$, we don't need to model $p(x)$, only $p(y|x)$.

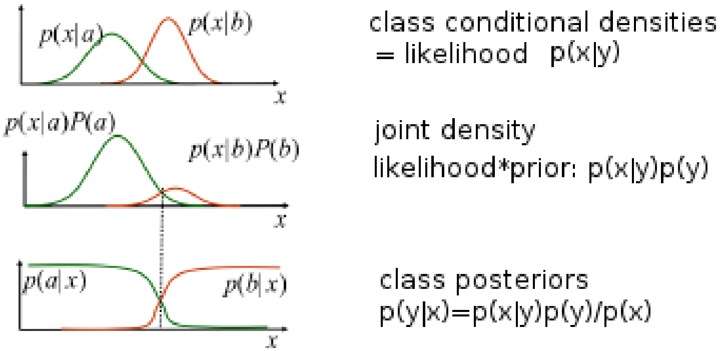
$$\boxed{\textbf{Let's use } \mathcal{D} \textbf{ to estimate } p(y|x)\textbf{.}}$$

Observation

Instead of $p(x, y) = p(x|y)p(y)$, we can also use $p(x, y) = p(y|x)p(x)$. Since $\mathbf{argmax}_y\, p(x, y) = \mathbf{argmax}_y\, p(y|x)$, we don't need to model $p(x)$, only $p(y|x)$.

$$\boxed{\textbf{Let's use } \mathcal{D} \textbf{ to estimate } p(y|x)\textbf{.}}$$

Visual intuition:



Observation

Instead of $p(x, y) = p(x|y)p(y)$, we can also use $p(x, y) = p(y|x)p(x)$. Since $\mathbf{argmax}_y\, p(x, y) = \mathbf{argmax}_y\, p(y|x)$, we don't need to model $p(x)$, only $p(y|x)$.

$\boxed{\textbf{Let's use } \mathcal{D} \textbf{ to estimate } p(y|x)\textbf{.}}$

Example (Spam Classification)

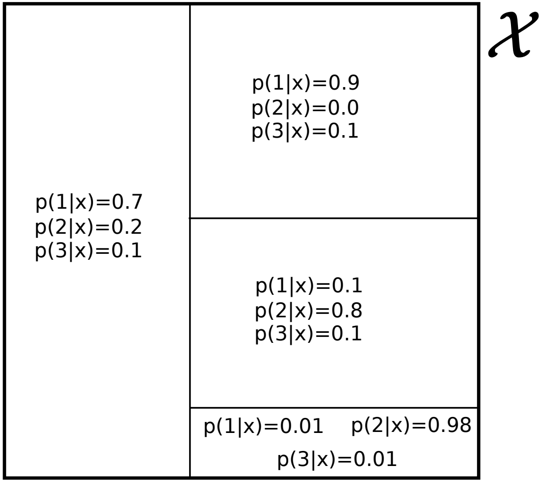
Is $p(y|x)$ really easier than, e.g., $p(x|y)$?

- $p($*"v1agra"*$|\texttt{spam})$ is some positive value (not every spam is viagra)
- $p(\texttt{spam}|$*"v1agra"*$)$ is almost surely 1.

For $p(y|x)$ we treat x as *given*, we don't need to know its probability.

Nonparametric Discriminative Model

Idea: split $\mathcal{X}$ into regions, for each region store an estimate $\hat{p}(y|x)$.



Nonparametric Discriminative Model

Idea: split $\mathcal{X}$ into regions, for each region store an estimate $\hat{p}(y|x)$.

For example, using a **decision tree**:

- training: build a tree
- prediction: for new example x, find its leaf
- output $\hat{p}(y|x) = \frac{n_y}{n}$, where
  - n is the number of examples in the leaf,
  - n_y is the number of example of label y in the leaf.

Nonparametric Discriminative Model

Idea: split $\mathcal{X}$ into regions, for each region store an estimate $\hat{p}(y|x)$.

For example, using a **decision tree**:

- training: build a tree
- prediction: for new example x, find its leaf
- output $\hat{p}(y|x) = \frac{n_y}{n}$, where
  - n is the number of examples in the leaf,
  - n_y is the number of example of label y in the leaf.

Note: prediction rule

$$c(x) = \underset{y}{\mathbf{argmax}}\, \hat{p}(y|x)$$

is predicts the most frequent label in each leaf (same as in first lecture).

Parametric Discriminative Model: Logistic Regression

Setting. We assume $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} = \{-1, +1\}$.

Definition (Logistic Regression (LogReg) Model)

Modeling

$$\hat{p}(y|x; w) = \frac{1}{1 + \exp(-y\langle w, x\rangle)},$$

with parameter vector $w \in \mathbb{R}^d$ is called a *logistic regression* model.

Parametric Discriminative Model: Logistic Regression

Setting. We assume $\mathcal{X} \subseteq \mathbb{R}^d$ and $\mathcal{Y} = \{-1, +1\}$.

Definition (Logistic Regression (LogReg) Model)

Modeling

$$\hat{p}(y|x;w) = \frac{1}{1 + \exp(-y\langle w, x\rangle)},$$

with parameter vector $w \in \mathbb{R}^d$ is called a *logistic regression* model.

Lemma

$\hat{p}(y|x;w)$ *is a well defined probability density w.r.t.* y *for any* $w \in \mathbb{R}^d$*.*

Proof. elementary.

How to set the weight vector w (based on $\mathcal{D}$)

Logistic Regression Training

Given a training set $\mathcal{D} = \{(x^1, y^1), \dots, (x^n, y^n)\}$, *logistic regression training* sets the free parameter vector as

$$w_{LR} \ = \ \operatorname*{\mathbf{argmin}}_{w \in \mathbb{R}^d} \ \sum_{i=1}^{n} \log\left(1 + \exp(-y^i \langle w, x^i \rangle)\right)$$

Lemma (Conditional Likelihood Maximization)

w_{LR} *from Logistic Regression training maximizes the conditional data likelihood w.r.t. the LogReg model,*

$$w_{LR} \ = \ \operatorname*{\mathbf{argmax}}_{w \in \mathbb{R}^d} \ \hat{p}(y^1, \dots, y^n | x^1, \dots, x^n, w)$$

Proof.

Maximizing

$$\hat{p}(\mathcal{D}^Y|\mathcal{D}^X, w) \overset{i.i.d.}{=} \prod_{i=1}^{n} \hat{p}(y^i|x^i, w)$$

is equivalent to minimizing its negative logarithm

$$\begin{aligned}
-\log\ \hat{p}(\mathcal{D}^Y|\mathcal{D}^X, w) &= -\log \prod_{i=1}^{n} \hat{p}(y^i|x^i, w) = -\sum_{i=1}^{n} \log \hat{p}(y^i|x^i, w) \\
&= -\sum_{i=1}^{n} \log \frac{1}{1+\exp(-y^i\langle w, x^i\rangle)}, \\
&= -\sum_{i=1}^{n} [\log 1 - \log(1+\exp(-y^i\langle w, x^i\rangle))], \\
&= \sum_{i=1}^{n} \log(1+\exp(-y^i\langle w, x^i\rangle)).
\end{aligned}$$

□

Alternative Explanation

Definition (Kullback-Leibler (KL) divergence)

Let p and q be two probability distributions (for discrete $\mathcal{Z}$) or probabilitiy densities with respect to a measure dλ (for continuous $\mathcal{Z}$). The **Kullbach-Leibler (KL)-divergence** between p and q is defined as

$$\mathrm{KL}(p\,\|q) = \sum_{z\in\mathcal{Z}} p(z)\log\frac{p(z)}{q(z)}, \quad \text{or} \quad \mathrm{KL}(p\,\|q) = \int_{z\in\mathcal{Z}} p(z)\log\frac{p(z)}{q(z)}\ \mathsf{d}\lambda(\mathsf{z}),$$

(with convention $0\log 0 = 0$, and $a\log\frac{a}{0} = \infty$ for $a > 0$).

Alternative Explanation

Definition (Kullback-Leibler (KL) divergence)

Let p and q be two probability distributions (for discrete $\mathcal{Z}$) or probabilitiy densities with respect to a measure dλ (for continuous $\mathcal{Z}$). The **Kullbach-Leibler (KL)-divergence** between p and q is defined as

$$\mathrm{KL}(p\,\|q) = \sum_{z\in\mathcal{Z}} p(z)\log\frac{p(z)}{q(z)}, \quad \text{or} \quad \mathrm{KL}(p\,\|q) = \int_{z\in\mathcal{Z}} p(z)\log\frac{p(z)}{q(z)}\,\mathrm{d}\lambda(\mathrm{z}),$$

(with convention $0\log 0 = 0$, and $a\log\frac{a}{0} = \infty$ for $a > 0$).

KL is a similarity measure between probability distributions. It fulfills

$$0 \le KL(p\,\|q) \le \infty, \qquad \text{and} \qquad KL(p\,\|q) = 0 \;\Leftrightarrow\; p = q.$$

However, KL is **not a metric**.

- it is in general not symmetric, $KL(q\,\|p) \neq KL(p\,\|q)$,
- it does not fulfill the triangle inequality.

Alternative Explanation of Logistic Regression Training

Definition (Expected Kullback-Leibler (KL) divergence)

Let $p(x, y)$ be a probability distribution over $(x, y) \in \mathcal{X} \times \mathcal{Y}$ and let $\hat{p}(y|x)$ be an approximation of $p(y|x)$.
We measure the approximation quality by the **expected KL-divergence between** p **and** q over all $x \in \mathcal{X}$:

$$\mathrm{KL}_{\textit{exp}}(p\,\|q) = \mathop{\mathbb{E}}_{x \sim p(x)} \{\, \mathrm{KL}(p(\cdot|x) \| q(\cdot|x)) \,\}$$

Theorem

The parameter w_{LR} obtained by logistic regression training approximately minimizes the KL divergence between $\hat{p}(y|x; w)$ and $p(y|x)$.

Proof.

We show how maximizing the conditional likelihood relates to KL_{exp}:

$$\begin{aligned}\mathrm{KL}_{exp}(p\|\hat{p}) &= \mathop{\mathbb{E}}_{x\sim p(x)} \sum_{y\in\mathcal{Y}} p(y|x)\log\frac{p(y|x)}{\hat{p}(y|x,w)} \\ &= \underbrace{\mathop{\mathbb{E}}_{(x,y)\sim p(x,y)} \log p(y|x)}_{\text{indep. of } w} - \mathop{\mathbb{E}}_{(x,y)\sim p(x,y)} \log \hat{p}(y|x,w)\end{aligned}$$

We can't maximize $\mathbb{E}_{(x,y)\sim p(x,y)} \log \hat{p}(y|x,w)$ directly, because $p(x,y)$ is unknown. But we can maximimize its empirical estimate based on $\mathcal{D}$:

$$\mathop{\mathbb{E}}_{(x,y)\sim p(x,y)} \log \hat{p}(y|x,w) \;\approx\; \underbrace{\sum_{(x^i,y^i)\in\mathcal{D}} \log \hat{p}(y^i|x^i,w)}_{\log \text{ of conditional data likelihood}} \quad .$$

The approximation will get better the more data we have. □

Solving Logistic Regression numerically – Optimization I

Theorem

Logistic Regression training,

$$w_{LR} \;=\; \operatorname*{\mathbf{argmin}}_{w\in\mathbb{R}^d}\; \mathcal{L}(w) \quad \textit{for} \quad \mathcal{L}(w) = \sum_{i=1}^{n} \log\big(1 + \exp(-y^i\langle w, x^i\rangle)\big),$$

is a C^∞-smooth, unconstrained, convex optimization problem.

Proof.

1. it's an optimization problem,
2. it's unconstrained,
3. it's smooth (the objective function is C^∞ differentiable),
4. remains to show: the objective function is a *convex* function.
   Since $\mathcal{L}$ is smooth, it's enough to show that its *Hessian matrix* (the matrix of 2nd partial derivatives) is everywhere *positive definite.*

We compute first the gradient and then the Hessian of

$$\mathcal{L}(w) = \sum_{i=1}^{n} \log(1 + \exp(-y^i \langle w, x^i \rangle).$$

$$\nabla_w \ \mathcal{L}(w) = \sum_{i=1}^{n} \nabla \log(1 + \exp(-y^i \langle w, x^i \rangle).$$

use the chain rule, $\nabla f(g(w)) = \frac{df}{dt}(g(w)) \nabla g(w)$, and $\frac{d \log(t)}{dt} = \frac{1}{t}$

$$= \sum_{i=1}^{n} \frac{\nabla [1 + \exp(-y^i \langle w, x^i \rangle]}{1 + \exp(-y^i \langle w, x^i \rangle}$$

$$= \sum_{i=1}^{n} \underbrace{\frac{\exp(-y^i \langle w, x^i \rangle)}{1 + \exp(-y^i \langle w, x^i \rangle)}}_{= \hat{p}(-y^i | x^i, w)} \nabla (-y^i \langle w, x^i \rangle)$$

use the chain rule again, $\frac{d}{dt} \exp(t) = \exp(t)$, and $\nabla_w \langle w, x^i \rangle = x^i$

$$= - \sum_{i=1}^{n} [\hat{p}(-y^i | x^i, w)] \ y^i x^i$$

$$H_w\mathcal{L}(w) = \nabla\nabla^\top\mathcal{L}(w) = -\sum_{i=1}^{n}[\nabla\hat{p}(-y^i|x^i,w)]\ y^i x^i$$

$$\nabla\hat{p}(-y^i|x^i,w) = \nabla\frac{1}{1+\exp(y^i\langle w,x^i\rangle)}$$

$$= -\frac{\nabla[1+\exp(y^i\langle w,x^i\rangle)]}{[1+\exp(y^i\langle w,x^i\rangle)]^2}$$

use quotient rule, $\nabla\frac{1}{f(w)} = -\frac{\nabla f(w)}{f^2(w)}$, and chain rule,

$$= -\frac{\exp(y^i\langle w,x^i\rangle)}{[1+\exp(y^i\langle w,x^i\rangle)]^2}\nabla y^i\langle w,x^i\rangle$$

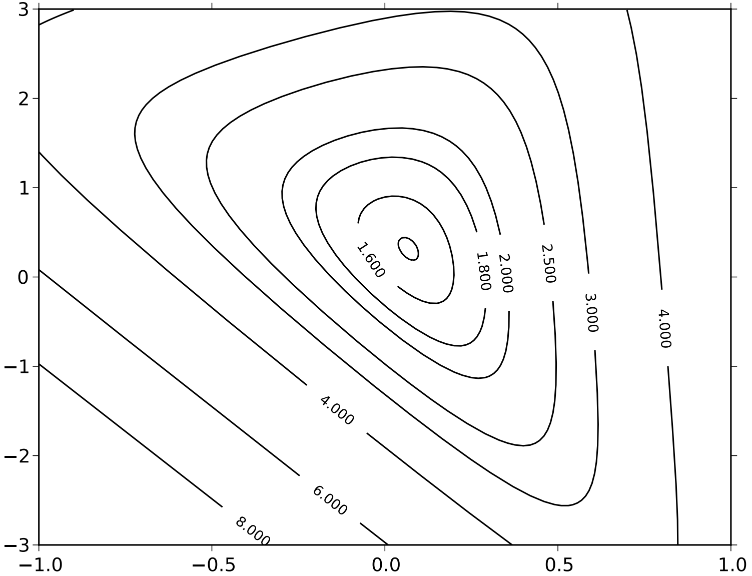
$$= -(\hat{p}(-y^i|x^i))\hat{p}(y^i|x^i,w)y^i x^i$$

insert into above expression for $H_w\mathcal{L}(w)$

$$H = \sum_{i=1}^{n}\underbrace{\hat{p}(-y^i|x^i)\hat{p}(y^i|x^i,w)}_{>0}\ \underbrace{x^i x^{i\top}}_{sym.pos.def.}$$

A positively weighted linear combination of pos.def. matrices is pos.def.

Example plot: LogReg objective for three examples in $\mathbb{R}^2$



Numeric Optimization

Convex optimization is a well understood field. We can use, e.g., *gradient descent* will converge to the globally optimal solution!

Steepest Descent Minimization with Line Search

```
input   $\epsilon > 0$ tolerance (for stopping criterion)
  1: $w \leftarrow 0$
  2: repeat
  3:     $v \leftarrow -\nabla_w \mathcal{L}(w)$                  {descent direction}
  4:     $\eta \leftarrow \mathbf{argmin}_{\eta>0} \mathcal{L}(w + \eta v)$        {1D line search}
  5:     $w \leftarrow w + \eta v$
  6: until $\|v\| < \epsilon$
output  $w \in \mathbb{R}^d$ learned weight vector
```

Faster conference from methods that use second-order information, e.g., *conjugate gradients* or *(L-)BFGS* $\rightarrow$ *convex optimization lecture*

Binary classification with a LogReg Models

A discriminative probability model, $\hat{p}(y|x)$, is enough to make decisions:

$$c(x) = \operatorname*{\mathbf{argmax}}_{y\in\mathcal{Y}} \hat{p}(y|x) \quad \text{or} \quad c(x) = \operatorname*{\mathbf{argmin}}_{y\in\mathcal{Y}} \mathop{\mathbb{E}}_{\bar{y}\sim\hat{p}(y|x)} \ell(\bar{y}, y).$$

For Logistic Regression, this is particularly simple:

Lemma

The LogReg classification rule for $0/1$-loss is

$$c(x) = \operatorname{sign} \langle w, x\rangle.$$

For a loss function $\ell = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ the rule is

$$c_\ell(x) = \operatorname{sign}[\ \langle w, x\rangle + \log \frac{c-d}{b-a}\ ],$$

*In particular, **the decision boundaries is linear (or affine).***

Proof. Elementary, since $\log \frac{\hat{p}(+1|x;w)}{p(-1|x;w)} = \langle w, x\rangle$

Multiclass Logistic Regression

For $\mathcal{Y} = \{1, \ldots, M\}$, we can do two things:

- Parametrize $\hat{p}(y|x; \vec{w})$ using $M-1$ vectors, $w_1, \ldots, w_{M-1} \in \mathbb{R}^d$, as

$$\hat{p}(y|x, w) = \frac{\exp(\langle w_y, x\rangle)}{1 + \sum_{j=1}^{M-1} \exp(\langle w_j, x\rangle)} \quad \text{for } y = 1, \ldots, M-1,$$
$$\hat{p}(M|x, w) = \frac{1}{1 + \sum_{j=1}^{M-1} \exp(\langle w_j, x\rangle)}.$$

- Parametrize $\hat{p}(y|x; \vec{w})$ using M vectors, $w_1, \ldots, w_M \in \mathbb{R}^d$, as

$$\hat{p}(y|x, w) = \frac{\exp(\langle w_y, x\rangle)}{\sum_{j=1}^{M} \exp(\langle w_j, x\rangle)} \quad \text{for } y = 1, \ldots, M,$$

Second is more popular, since it's easier to implement and analyze.

Decision boundaries are still *piecewise linear*, $c(x) = \mathbf{argmax}_y \langle w_y, x\rangle$.

Summary: Discriminative Models

Discriminative models treats the input data, x, as fixed and only model the distribution of the output labels $p(y|x)$.

Discriminative models, in particular LogReg, are popular, because

- they often need less training data than generative models,
- they provide an estimate of the uncertainty of a decision $p(c(x)|x)$,
- training them is often efficient, e.g. big companies train LogReg models routinely from billions of examples.

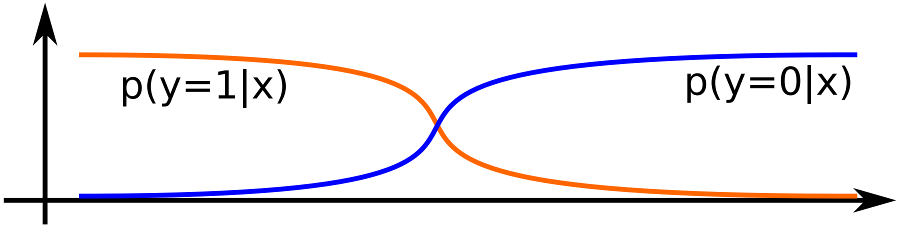
But: they also have drawbacks

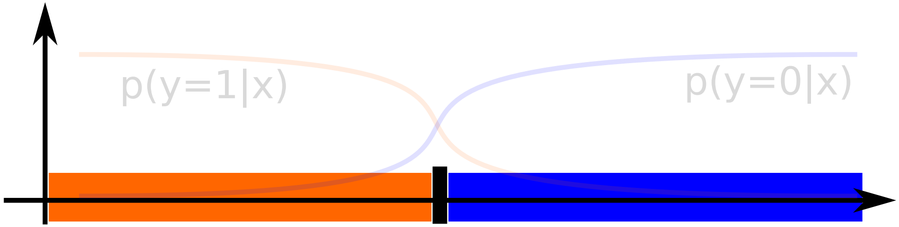
- often $\hat{p}_{LR}(y|x) \not\to p(y|x)$, even for $n \to \infty$,
- they usually are good for *prediction*, but they do not reflect the actual *mechanism*.

Note: there are much more complex discriminative models than LogReg, e.g. Conditional Random Fields (maybe later).

Observation

Even easier than estimating $p(y|x)$ (or $p(x, y)$) should be to just estimate the decision boundary between classes.





Maximum Margin Classifiers

$\boxed{\textbf{Let's use } \mathcal{D} \textbf{ to estimate a classifier } c : \mathcal{X} \to \mathcal{Y} \textbf{ directly.}}$

Maximum Margin Classifiers

$\boxed{\textbf{Let's use } \mathcal{D} \textbf{ to estimate a classifier } c : \mathcal{X} \to \mathcal{Y} \textbf{ directly.}}$

For a start, we fix

- $\mathcal{D} = \{(x^1, y^1), \ldots, (x^n, y^n)\}$,
- $\mathcal{Y} = \{+1, -1\}$,
- we look for classifiers with linear decision boundary.

Several of the classifiers we saw had *linear* decision boundaries:

- Perceptron
- Generative classifiers for Gaussian class-conditional densities with shared covariance matrix
- Logistic Regression

What's the **best linear classifier**?

Linear classifiers

Definition

Let

$$\mathcal{F} = \{\ f : \mathbb{R}^d \to \mathbb{R} \text{ with } f(x) = b + a_1 x_1 + \cdots + a_d x_d = b + \langle w, x\rangle\ \}$$

be the set of linear (affine) function from $\mathbb{R}^d \to \mathbb{R}$.

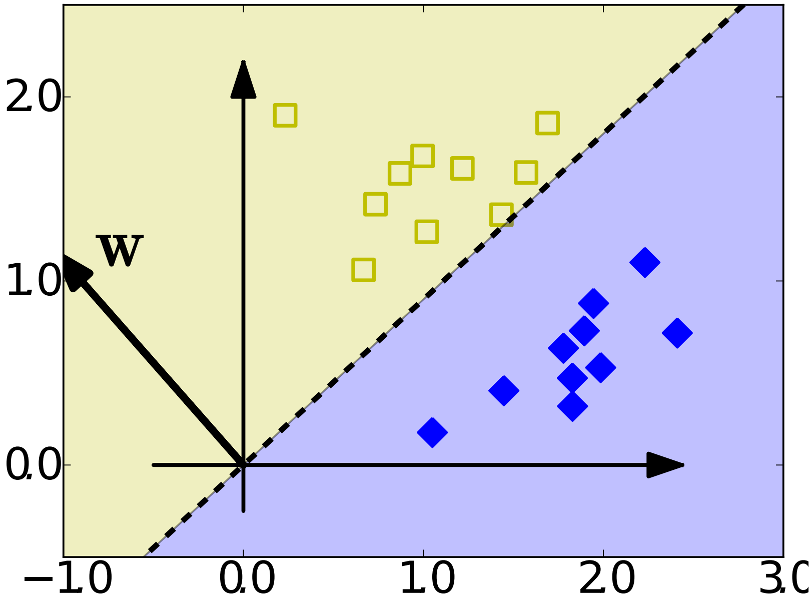
A classifier $g : \mathcal{X} \to \mathcal{Y}$ is called **linear**, if it can be written as

$$g(x) = \operatorname{sign} f(x)$$

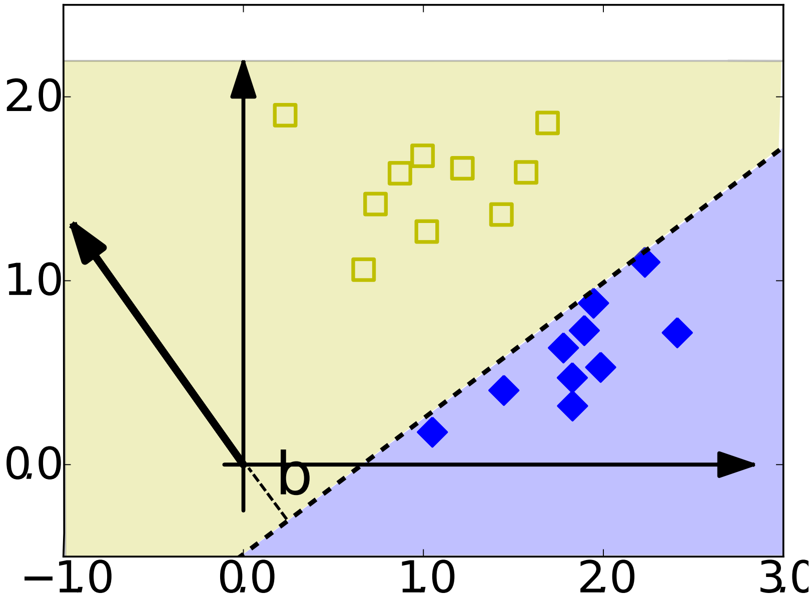
for some $f \in \mathcal{F}$.

We write $\mathcal{G}$ for the set of all linear classifiers.

A linear classifier, $g(x) = \mathrm{sign}\langle w, x\rangle$, with $b = 0$



A linear classifier $g(x) = \operatorname{sign}(\langle w, x \rangle + b)$, with $b > 0$



Feature augmentation

The bias term is good for intuition, but annoying in analysis:

Useful trick: feature augmentation

Adding a constant feature allows us to avoid models with explicit bias term:

- instead of $x = (x^1, \dots, x^d) \in \mathbb{R}^d$, use $\tilde{x} = (x^1, \dots, x^d, 1) \in \mathbb{R}^{d+1}$
- for any $\tilde{w} \in \mathbb{R}^{d+1}$, think $\tilde{w} = (w, b)$ with $w \in \mathbb{R}^d$ and $b \in \mathbb{R}$

Linear function in $\mathbb{R}^{d+1}$:

$$f(\tilde{x}) = \langle \tilde{w}, \tilde{x} \rangle = \sum_{i=1}^{d+1} \tilde{w}_i \tilde{x}_i = \sum_{i=1}^{d} \tilde{w}_i \tilde{x}_i + \tilde{w}_{d+1} \tilde{x}_{d+1} = \langle w, x \rangle + b$$

$$\boxed{\text{Linear classifier with bias in } \mathbb{R}^d \; \equiv \; \text{linear classifier with no bias in } \mathbb{R}^{d+1}}$$

Augmenting with other (larger) values than 1 can make sense, see later...

Linear classifiers

Definition (Ad hoc)

We call a classifier, g, **correct** (for a training set $\mathcal{D}$), if it predicts the correct labels for all training examples:

$$g(x^i) = y^i \qquad \text{for } i = 1, \dots, n.$$

Example (Perceptron)

- if the *Perceptron* converges, the result is an *correct* classifier.
- any classifier with zero training error is *correct*.

Linear classifiers

Definition (Ad hoc)

We call a classifier, g, **correct** (for a training set $\mathcal{D}$), if it predicts the correct labels for all training examples:

$$g(x^i) = y^i \qquad \text{for } i = 1, \dots, n.$$

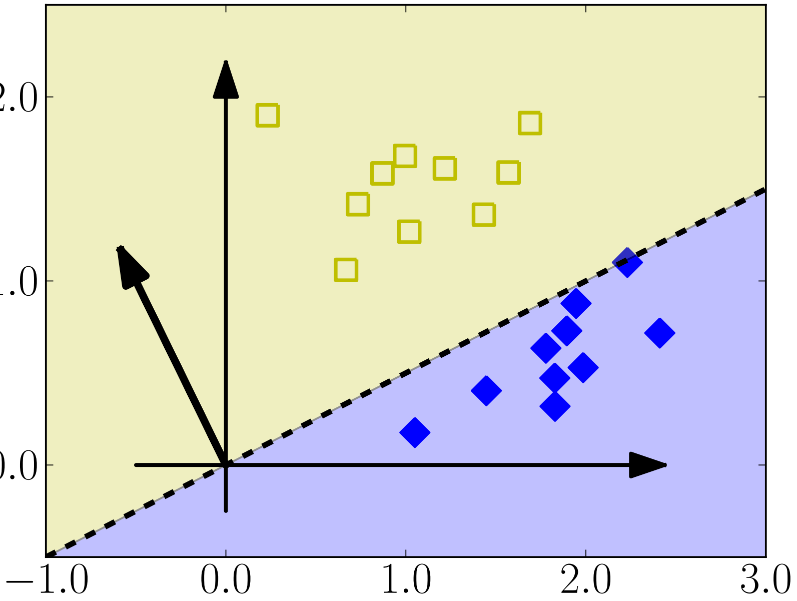
Example (Perceptron)

- if the *Perceptron* converges, the result is an *correct* classifier.
- any classifier with zero training error is *correct*.

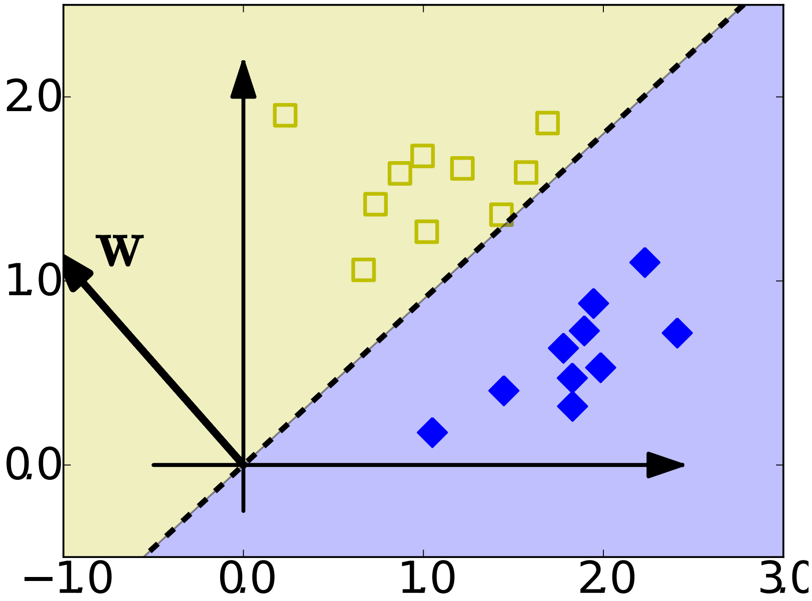
Definition (Linear Separability)

A training set $\mathcal{D}$ is called **linearly separable**, if it allows a correct linear classifier (i.e. the classes can be separated by a hyperplane).

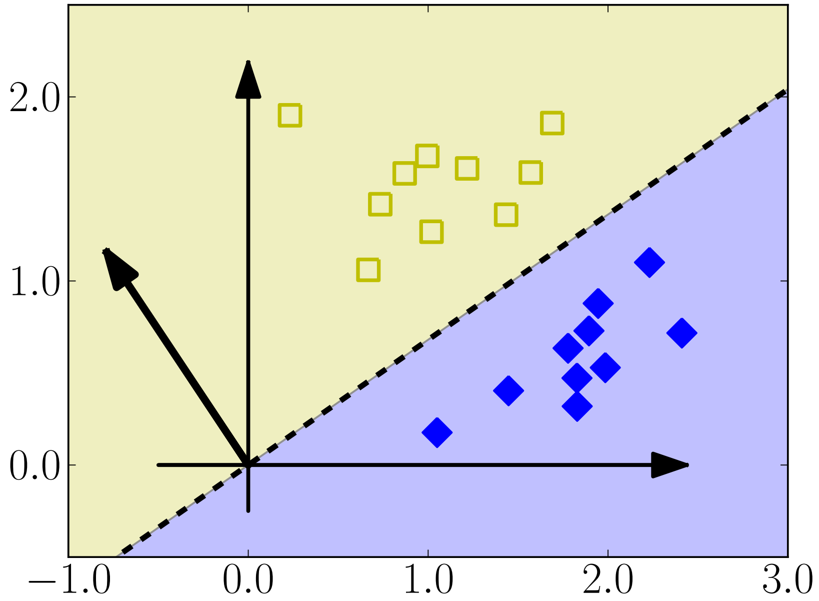
A linearly separable dataset and a correct classifier



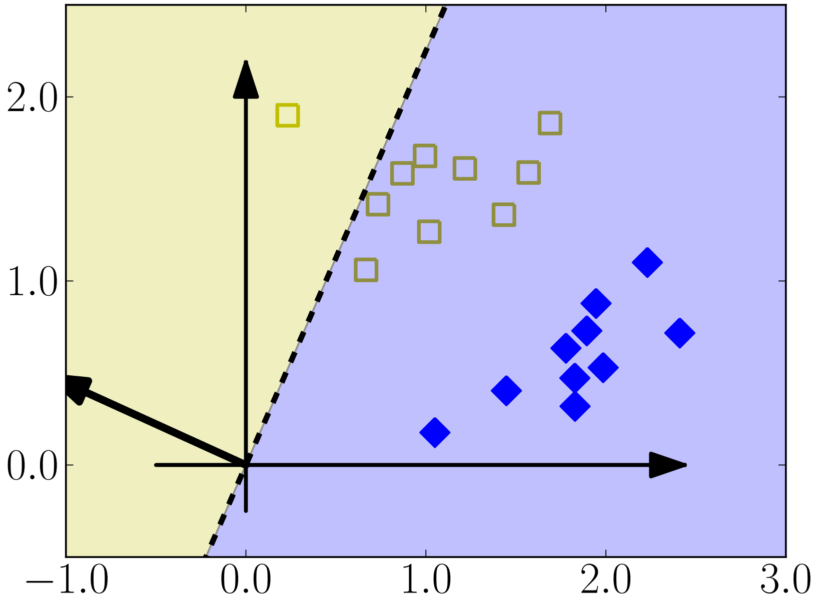
A linearly separable dataset and a correct classifier



A linearly separable dataset and a correct classifier



An incorrect classifier



Linear Classifiers

Definition (Ad hoc)

The **robustness** of a classifier g (with respect to $\mathcal{D}$) is the largest amount, ρ, by which we can perturb the training samples without changing the predictions of g.

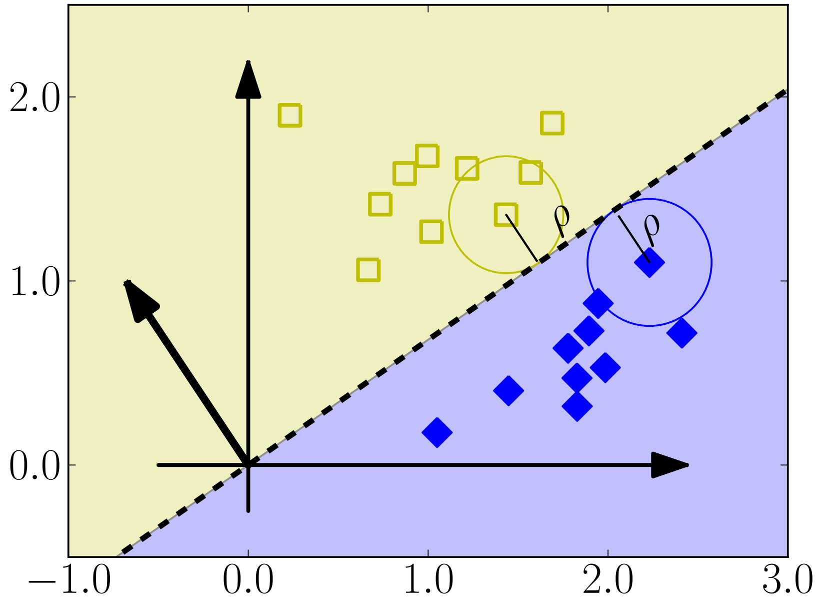
$$g(x^i + \epsilon) = g(x^i), \qquad \text{for all } i = 1, \dots, n.$$

for any $\epsilon \in \mathbb{R}^d$ with $\|\epsilon\| < \rho$.

Example

- constant classifier, e.g. $c(x) \equiv 1$: very robust ($\rho = \infty$), (but it is not *correct*, in the sense of the previous definition)
- robustness of the *Perceptron*: can be arbitrarily small (see Exercise...)

Robustness, ρ, of a linear classifier



Definition (Margin)

Let $f(x) = \langle w, x \rangle + b$ define a *correct* linear classifier.
Then the smallest (Euclidean) distance of any training example from the decision hyperplane is called the **margin** of f (with respect to $\mathcal{D}$).

Lemma

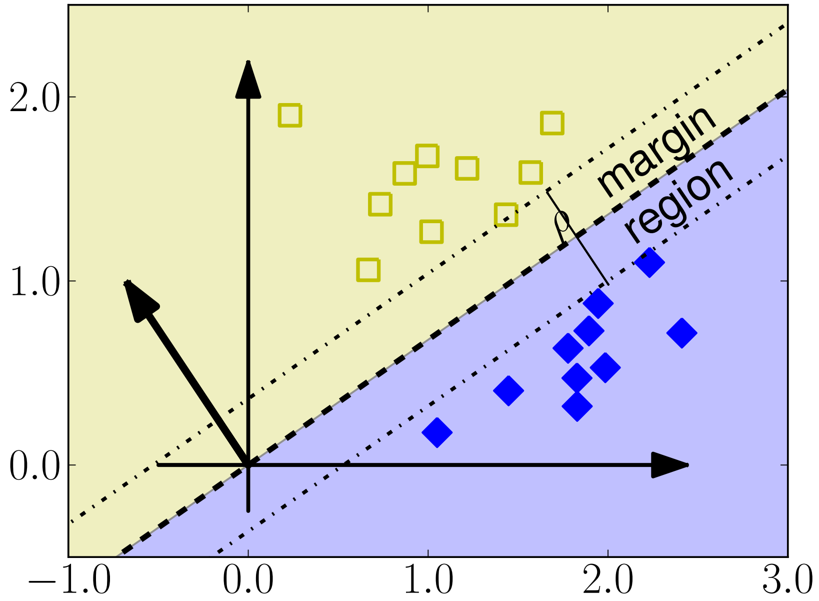
We can compute the margin of a linear classifier $f = \langle w, x \rangle + b$ *as*

$$\rho = \min_{i=1,\dots,n} \Big| \langle \frac{w}{\|w\|}, x^i \rangle + b \Big|.$$

Proof.

High school maths: distance between a points and a hyperplane in *Hessian normal form.*

Margin, ρ, of a linear classifier



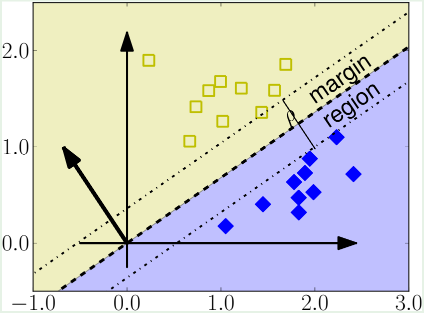
Theorem

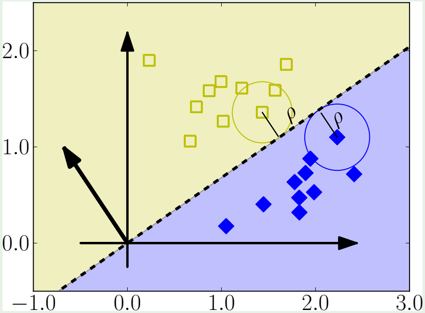
The robustness of a linear classifier function $g(x) = \operatorname{sign} f(x)$ with $f(x) = \langle w, x \rangle$ is identical to the margin of f.

Theorem

The robustness of a linear classifier function $g(x) = \operatorname{sign} f(x)$ *with* $f(x) = \langle w, x \rangle$ *is identical to the margin of* f*.*

Proof by Picture





Proof (blackboard). For any $i = 1, \dots, n$ and any $\epsilon \in \mathbb{R}^d$

$$f(x^i + \epsilon) = \langle w, x^i + \epsilon \rangle = \langle w, x^i \rangle + \langle w, \epsilon \rangle = f(x^i) + \langle w, \epsilon \rangle,$$

so it follows (Cauchy-Schwarz inequality) that

$$f(x^i) - \|w\|\|\epsilon\| \quad \leq \quad f(x^i + \epsilon) \quad \leq \quad f(x^i) + \|w\|\|\epsilon\|.$$

Checking the cases $\epsilon = \pm \frac{\|\epsilon\|}{\|w\|} w$, we see that these inequalities are sharp.

To ensure $g(x^i + \epsilon) = g(x^i)$ for all training samples, $f(x^i)$ and $f(x^i + \epsilon)$ have the same sign for all ϵ, i.e. $|f(x^i)| \geq \|w\|\|\epsilon\|$ for $i = 1, \dots, n$.

This inequality holds for all samples, so in particular it holds for the one of minimal score, and $\mathbf{min}_i \, |f(x^i)| = \mathbf{min}_i \, |\langle w, x^i \rangle| = \rho$.

$\square$

Maximum-Margin Classifier

Theorem

Let $\mathcal{D}$ be a linearly separable training set. Then the ***most robust, correct linear classifier*** *(without bias term) is given by* $g(x) = \operatorname{sign}\langle w^*, x\rangle$ *where w^* are the solution to*

$$\min_{w \in \mathbb{R}^d} \frac{1}{2}\|w\|^2$$

subject to

$$y^i(\langle w, x^i\rangle) \geq 1, \quad \textit{for } i = 1, \ldots, n.$$

Remark

- The classifier defined above is call **Maximum (Hard) Margin Classifier**, or **Hard-Margin Support Vector Machine (SVM)**
- It is unique (follows from strictly convex optimization problem).

Proof.

1. All w that fulfill the inequalities yield *correct* classifiers.
2. Since $\mathcal{D}$ is linearly separable, there exists some v with

$$\operatorname{sign}\langle v, x^i \rangle = y_i, \quad \text{i.e.} \quad y_i \langle v, x^i \rangle \geq \gamma > 0.$$

for $\gamma = \mathbf{min}_i \, y_i \langle v, x^i \rangle$. So $\tilde{v} = v/\gamma$, fulfills the inequalities and we see that the constraint set is at least not empty.

Proof.

1. All w that fulfill the inequalities yield *correct* classifiers.
2. Since $\mathcal{D}$ is linearly separable, there exists some v with

$$\operatorname{sign}\langle v, x^i\rangle = y_i, \quad \text{i.e.} \quad y_i\langle v, x^i\rangle \geq \gamma > 0.$$

   for $\gamma = \mathbf{min}_i\, y_i\langle v, x^i\rangle$. So $\tilde{v} = v/\gamma$, fulfills the inequalities and we see that the constraint set is at least not empty.

3. Now we check (with $i = 1, \ldots, n$):

$$\begin{aligned}
& \mathop{\mathbf{min}}_{w\in\mathbb{R}^d} \frac{1}{2}\|w\|^2 \text{ sb.t. } y^i\langle w, x^i\rangle \geq 1 \\
\Leftrightarrow \quad & \mathop{\mathbf{max}}_{w\in\mathbb{R}^d} \frac{1}{\|w\|} \quad \text{sb.t. } y^i\langle w, x^i\rangle \geq 1 \\
\Leftrightarrow \quad & \mathop{\mathbf{max}}_{\{w':|w'\|=1\},\rho\in\mathbb{R}} \quad \rho \quad \text{sb.t. } y^i\langle \frac{w'}{\rho}, x^i\rangle \geq 1 \\
\Leftrightarrow \quad & \mathop{\mathbf{max}}_{\{w':|w'\|=1\},\rho\in\mathbb{R}} \quad \rho \quad \text{sb.t. } y^i\langle w', x^i\rangle \geq \rho \\
\Leftrightarrow \quad & \mathop{\mathbf{max}}_{\{w':|w'\|=1\},\rho\in\mathbb{R}} \quad \rho \quad \text{sb.t. } |\langle w', x^i\rangle| \geq \rho \text{ and } \operatorname{sign}\langle w', x^i\rangle = y_i
\end{aligned}$$

Proof.

1. All w that fulfill the inequalities yield *correct* classifiers.
2. Since $\mathcal{D}$ is linearly separable, there exists some v with

$$\operatorname{sign}\langle v, x^i\rangle = y_i, \quad \text{i.e.} \quad y_i\langle v, x^i\rangle \geq \gamma > 0.$$

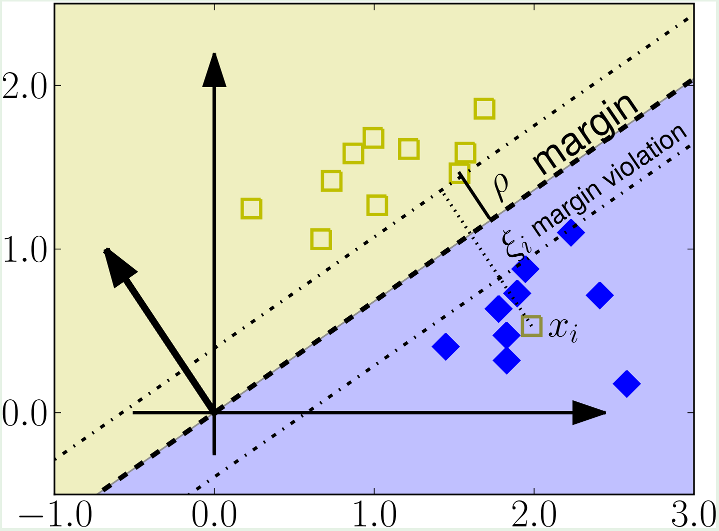
   for $\gamma = \mathbf{min}_i\, y_i\langle v, x^i\rangle$. So $\tilde{v} = v/\gamma$, fulfills the inequalities and we see that the constraint set is at least not empty.

3. Now we check (with $i = 1, \ldots, n$):

$$\begin{aligned}
& \min_{w\in\mathbb{R}^d} \frac{1}{2}\|w\|^2 \text{ sb.t. } y^i\langle w, x^i\rangle \geq 1 \\
\Leftrightarrow \quad & \max_{w\in\mathbb{R}^d} \frac{1}{\|w\|} \quad \text{sb.t. } y^i\langle w, x^i\rangle \geq 1 \\
\Leftrightarrow \quad & \max_{\{w':|w'\|=1\},\rho\in\mathbb{R}} \rho \quad \text{sb.t. } y^i\langle \frac{w'}{\rho}, x^i\rangle \geq 1 \\
\Leftrightarrow \quad & \max_{\{w':|w'\|=1\},\rho\in\mathbb{R}} \rho \quad \text{sb.t. } y^i\langle w', x^i\rangle \geq \rho \\
\Leftrightarrow \quad & \underbrace{\max_{\{w':|w'\|=1\},\rho\in\mathbb{R}} \rho \quad \text{sb.t. } |\langle w', x^i\rangle| \geq \rho}_{} \text{ and } \underbrace{\operatorname{sign}\langle w', x^i\rangle = y_i}_{\text{and correct}}
\end{aligned}$$

Non-Separable Training Sets

Observation (Not all training sets are linearly separable.)



Definition (Maximum Soft-Margin Classifier)

Let $\mathcal{D}$ be a training set, not necessarily linearly separable. Let $C > 0$. Then the classifier $g(x) = \operatorname{sign}\langle w^*, x\rangle + b)$ where (w^*, b^*) are the solution to

$$\min_{w\in\mathbb{R}^d, \xi\in\mathbb{R}^n} \quad \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}\xi^i$$

subject to

$$\begin{aligned} y^i(\langle w, x^i\rangle + b) \geq 1 - \xi^i, &\quad \text{for } i = 1, \ldots, n. \\ \xi^i \geq 0, &\quad \text{for } i = 1, \ldots, n. \end{aligned}$$

is called **Maximum (Soft-)Margin Classifier** or **Linear Support Vector Machine**.

Maximum Soft-Margin Classifier

Theorem

The maximum soft-margin classifier exists and is unique for any $C > 0$.

Proof. optimization problem is strictly convex

Remark

The constant $C > 0$ is called **regularization** parameter.

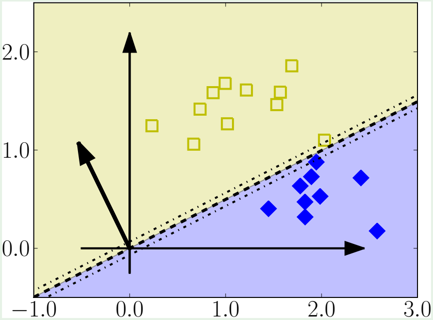
It balances the wishes for robustness and for correctness

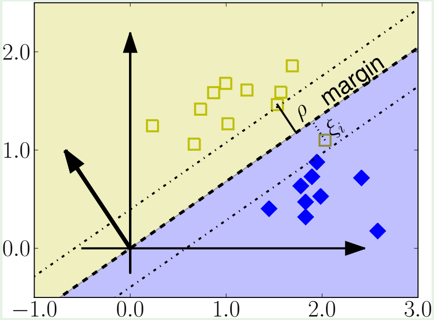
- $C \to 0$: mistakes don't matter much, emphasis on short w
- $C \to \infty$: as few errors as possible, might not be robust

We'll see more about this in the next lecture.

Remark

Sometimes, a soft margin is better even for linearly separable datasets!





Left: small margin, no errors) Right: large margin, but 1 error