Machine Learning in Bioinformatics

From Logistic Regression to SVMs

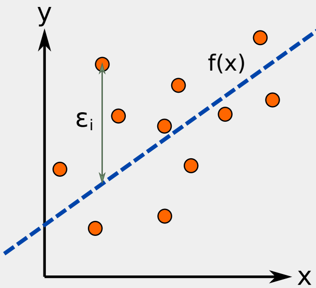
Philipp Benner
philipp.benner@bam.de

VP.1 - eScience
Federal Institute of Materials Research and Testing (BAM)

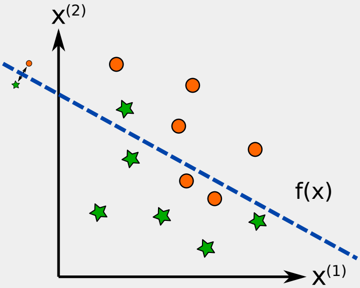
April 25, 2024

Logistic Regression (Classification)

Linear regression



Logistic regression



$$y = X\theta + \epsilon\,, \quad y \in \mathbb{R}$$

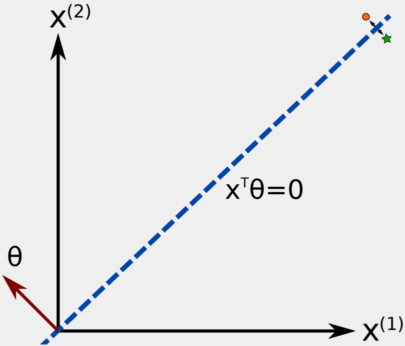
$$y \stackrel{?}{=} \sigma(X\theta) + \epsilon\,, \quad y \in \{0, 1\}$$

How is the hyperplane defined? What is σ?

- We use the properties of the dot product to define the separating hyperplane:

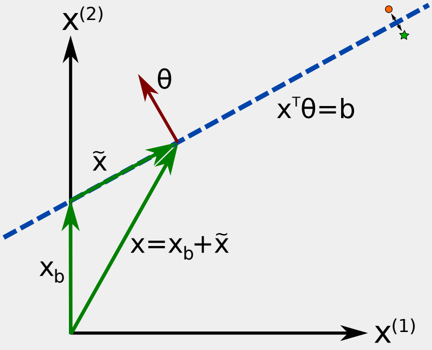
$$x^\top \theta = \|x\| \, \|\theta\| \cos \measuredangle$$

- For vectors x perpendicular to θ we have $\cos \measuredangle = 0$



- For hyperplanes with bias b we use $x^\top \theta = b$

$$
\begin{aligned}
x^\top \theta &= (x_b + \tilde{x})^\top \theta \\
&= \underbrace{x_b^\top \theta}_{=b} + \underbrace{\tilde{x}^\top \theta}_{=0}
\end{aligned}
$$



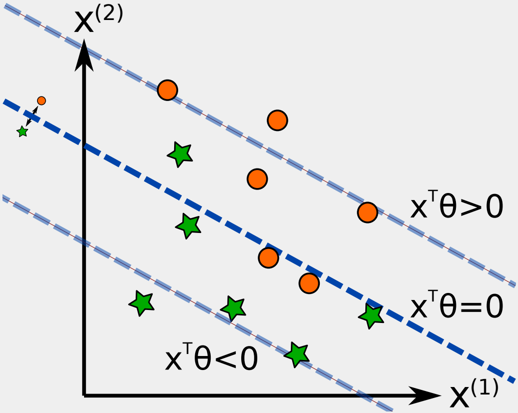
Defining hyperplanes

- Remember our convention:

$$x = \begin{bmatrix} 1 \\ x^{(2)} \\ \vdots \\ x^{(p)} \end{bmatrix}, \qquad \theta = \begin{bmatrix} \theta_1 \\ \theta_2 \\ \vdots \\ \theta_p \end{bmatrix}$$

- Hence, instead of $x^\top\theta = b$ we can write $x^\top\theta = 0$, because $\theta_1 = -b$

- $x^\top\theta > 0$: predicting positive class
- $x^\top\theta < 0$: predicting negative class

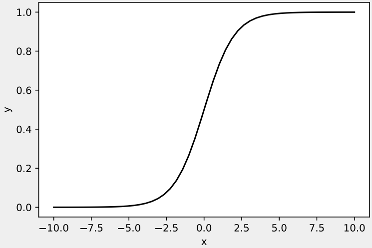


- We convert $x^\top\theta$ to probabilities

$$\mathrm{pr}(Y = 1 \mid x) = \sigma(x^\top\theta)$$

- The function σ denotes the sigmoid function

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$



- Given a training set (X, y) how do we estimate θ?
- Option 1: Minimizing squared error (similar to OLS)

$$\hat{\theta} = \arg\min_{\theta} \sum_{i=1}^{n} \left[y_i - \sigma(x_i^\top \theta) \right]$$

Problem: Not convex!

- Remember how we justified OLS for linear models?
- Option 2: Maximum likelihood

$$\hat{\theta} = \arg\max_{\theta} \mathrm{pr}(y \mid X, \theta)$$

- What is the probability of (X, y)?
- Remember a Bernoulli experiment (coin flip) with outcomes H (head) and T (tail)
- H is observed with probability p
- T is observed with probability $1 - p$
- The sequence HHTHT has probability

$$\mathrm{pr}(\mathrm{HHTHT}) = pp(1 - p)p(1 - p)$$

- Remember the following rule of thumb:

$$\times = \text{"and"}$$
$$+ = \text{"or"}$$

- For logistic regression, assume $y = (1, 1, 0, 1)$, hence

$$\mathrm{pr}(1, 1, 0, 1 \,|\, X, \theta) = \sigma(x_1^\top \theta)\sigma(x_2^\top \theta)(1 - \sigma(x_3^\top \theta))\sigma(x_4^\top \theta)$$

- Write it nicely in general form:

$$\mathrm{pr}(y \,|\, X, \theta) = \prod_{i=1}^{n} \sigma(x_i^\top \theta)^{y_i}(1 - \sigma(x_i^\top \theta))^{1-y_i}$$

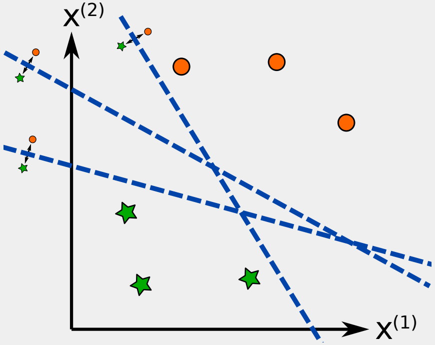
- Maximum likelihood

$$\begin{aligned}\hat{\theta} &= \arg\max_{\theta} \prod_{i=1}^{n} \sigma(x_i^\top \theta)^{y_i}(1 - \sigma(x_i^\top \theta))^{1-y_i} \\ &= \arg\max_{\theta} \sum_{i=1}^{n} y_i \log \sigma(x_i^\top \theta) + (1 - y_i) \log(1 - \sigma(x_i^\top \theta))\end{aligned}$$

- Convex optimization problem, but must be solved numerically

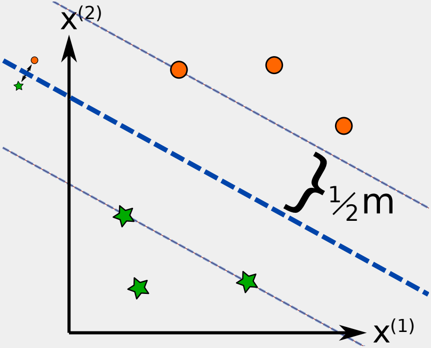
Support Vector Machines (SVMs)

- Support vector machines (SVMs) are similar to logistic regression, however, their learning algorithm is geometrically motivated:



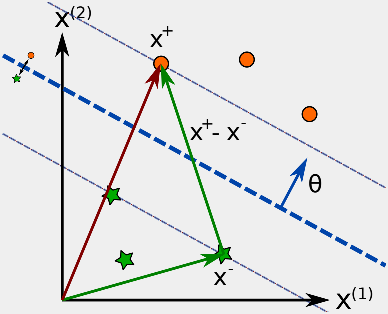
- What is the *best* separating hyperplane?

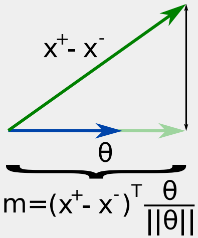
- SVMs take the hyperplane with maximum margin *m*:



- Data points touching the margin are called *support vectors*
- What is *m* and how can we maximize it?

- Computing the margin *m* given a fixed hyperplane:





- Hence, the margin is determined by the scalar projection of $x^+ - x^-$ onto $\theta / \|\theta\|$:

$$m = (x^+ - x^-)^\top \frac{\theta}{\|\theta\|_2}$$

- So far we did not enforce any constraints on θ
- There are infinitely many θ for the same separating hyperplane
- We apply the constraint

$$(x^+)^\top \theta = 1\,, \quad (x^-)^\top \theta = -1$$

for positive x^+ and negative x^- support vectors

- This definition leads to a simplified margin:

$$\begin{aligned} m &= (x^+ - x^-)^\top \frac{\theta}{\|\theta\|_2} \\ &= \frac{2}{\|\theta\|_2} \end{aligned}$$

SVM optimization problem

Let $(x_i, y_i)_i$ denote a training set such that $y_i \in \{-1, 1\}$. The parameters of the SVM are estimated as follows:

$$\hat{\theta} = \arg\min_{\theta} \ \|\theta\|_2$$
$$\text{s.t.} \quad x_i^\top \theta y_i \geq 1$$

- Note that minimizing $\|\theta\|_2$ is equivalent to maximizing the margin $2/\|\theta\|_2$
- The solution can be computed using the Lagrangian

$$L(\theta, \lambda) = \frac{1}{2}\|\theta\|_2 - \sum_{i=1}^{n} \lambda_i (x_i^\top \theta y_i - 1)$$

- The solution is a *saddle point* of the Lagrangian $L(\theta, \lambda)$

SVM dual problem

The solution of the SVM is obtained by maximizing the dual problem

$$
\begin{aligned}
Q(\lambda) &= \min_{\theta} L(\theta, \lambda) \\
&= \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \lambda_i \lambda_j y_i y_j x_i^\top x_j
\end{aligned}
$$

subject to $\lambda_i \geq 0$.

- We have a Lagrange multiplier λ_i for each data point
- λ_i is zero except for support vectors
- The dual problem is solved using the *Sequential minimal optimization (SMO)* algorithm [Cristianini et al., 2000]
- The dual representation depends on $x_i^\top x_j$

- What if the data is not linearly separable?
  Option 1: Slack variables

SVM optimization problem for non-linearly separable data

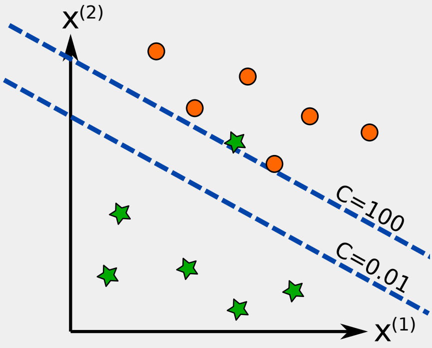
Let $(x_i, y_i)_i$ denote a training set such that $y_i \in \{-1, 1\}$. The parameters of the SVM are estimated as follows:

$$\hat{\theta} = \arg\min_{\theta} \; \|\theta\|_2 + C \sum_{i=1}^{n} \xi_i$$
$$\text{s.t.} \quad x_i^\top \theta y_i \geq 1 - \xi_i$$

where C is the slack panelty.

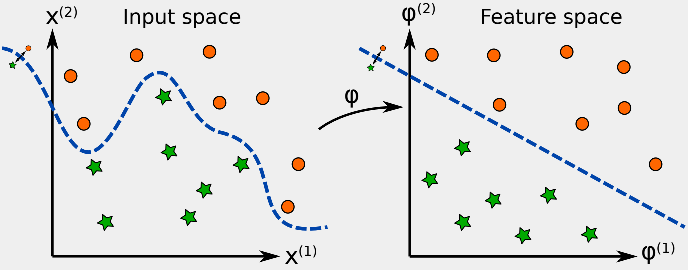
- $C = \infty$: Data must be linearly separated. $C = 0$: Ignore data.
- The dual problem is almost identical to the case of linearly separable data

- The effect of the slack penalty:



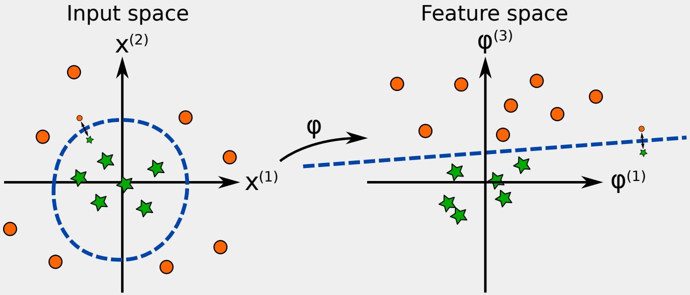
- $C = 0.001$: Some misclassified points are almost ignored
- $C = 100.0$: Get as close as possible to misclassified points

- What if the data is not linearly separable?
  Option 2: Feature space



- $x_i^\top x_j$ measures similarity in input space
- $\phi(x_i)^\top \phi(x_j)$ measures similarity in feature space
- Dimension of feature space is typically much larger
- Data often becomes linearly separable

- Example: $\phi(x^{(1)}, x^{(2)}) = (x^{(1)}, x^{(2)}, x^{(1)}x^{(1)} + x^{(2)}x^{(2)})$



- Projection of the hyperplane back to input space will result in a non-linear decision boundary

Definition: Kernel function

A function $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ is called a *kernel* if there exists a feature map $\phi : \mathcal{X} \to \mathcal{F}$ such that

$$\kappa(x_i, x_j) = \phi(x_i)^\top \phi(x_j)$$

$K = (\kappa(x_i, x_j))_{x_i \in \mathcal{X}, x_j \in \mathcal{X}}$ is called the kernel matrix.

- $\mathcal{X}$ can be an arbitrary space, for instance DNA sequences
- $\kappa(x_i, x_j)$ is interpreted as a similarity measure in feature space
- Evaluating $\kappa(x_i, x_j)$ does not always require to explicitly compute $\phi(x)$
- Not having to map data into feature space is called the kernel trick

Support vector machines - RBF Kernel

- The Gaussian or radial basis function (RBF) kernel:

$$k(x_i, x_j) = \exp\left\{-\frac{\left\|x_i - x_j\right\|_2^2}{2\sigma^2}\right\}$$

- Instead of the dot product $x_i^\top x_j$ we use the difference $x_i - x_j$ as the similarity measure
- What is the corresponding feature map ϕ?
- Taylor expansion of the kernel leads to

$$\phi(x) = \exp\left(-\frac{x^2}{2\sigma^2}\right)\left[1, \sqrt{\frac{1}{1!\sigma^2}}x, \sqrt{\frac{1}{2!\sigma^4}}x^2, \sqrt{\frac{1}{3!\sigma^6}}x^3, \dots\right]$$

- Feature space of the RBF kernel has infinite dimensions

Support vector machines - κ-spectrum kernel

- Suppose our input data is DNA sequences or any other type of strings
- How would we measure the similarity of two strings x_i and x_j?
- Feature map ϕ of the κ-spectrum kernel counts the number of occurrences of substrings of length κ
- Example:

$$x_1 = \text{"statistics"}$$
$$x_2 = \text{"computation"}$$

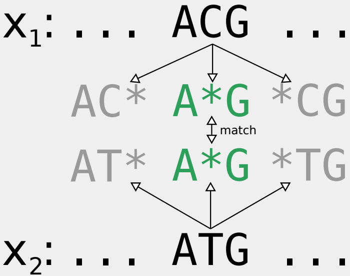
- For $\kappa = 3$ we get:

$$\phi(x_1) = \begin{bmatrix} \text{aaa} & \text{aab} & \dots & \text{sta} & \dots & \text{tat} & \dots \\ 0 & 0 & \dots & 1 & \dots & 1 & \dots \end{bmatrix}$$

- $k(x_1, x_2) = \phi(x_1)^\top \phi(x_2) = 1 \cdot 1 + 1 \cdot 1 = 2$
- We don't have to compute ϕ explicitly, only count the common substrings

Support vector machines - gapped κ-mers

- l: word or substring length
- k: number of non-gaps



- Number of gapped κ-mers:

$$\binom{l}{\kappa} 4^{\kappa}$$

- Requires very efficient implementation (e.g. gkmSVM [Ghandi et al., 2014])

- Support vector machines and logistic regression have different learning objectives
- SVMs maximize the margin between positive and negative samples
- Two approaches to deal with non-linearly separable data:
  - Slack variables to weaken the separability objectives
  - Implicit mapping into high-dimensional feature space with Kernels
- SVMs and logistic regression have different number of parameters:
  - SVMs: One parameter for each training point
  - Logistic regression: One parameter for each feature
- Evaluation of the kernel matrix takes $\mathcal{O}(n^2)$ steps

- Reading: Chapter 12 [Hastie et al., 2009], Section 6.1 [Cristianini et al., 2000]
- Advanced reading: *Representer Theorem*

References

Cristianini, N., Shawe-Taylor, J., et al. (2000).
An introduction to support vector machines and other kernel-based learning methods.
Cambridge university press.

Ghandi, M., Lee, D., Mohammad-Noori, M., and Beer, M. A. (2014).
Enhanced regulatory sequence prediction using gapped k-mer features.
PLoS computational biology, 10(7):e1003711.

Hastie, T., Tibshirani, R., and Friedman, J. (2009).
The elements of statistical learning: data mining, inference, and prediction.
Springer Science & Business Media.