Aalto University
School of Science
and Technology

Molecular classification

Hongyu Su

Department of Information and Computer Science
Aalto University, School of Science and Technology
hongyu.su@aalto.fi

March 14, 2015

Content

- Learning for classification
  - Statistic learning
  - Empirical risk minimization
  - Regularization
  - Support vector machines for single-task classification
  - Kernel methods
- Molecular classification problem
  - Predicting anti-cancer potentials of molecules
  - Hierarchical molecule classification
- Multi-task classification
  -

Learning for classification

Example: dog vs. cat?

- We have 5000 pictures of dog and 5000 pictures of cat.


- Each picture is digitalized into 100×100 pixels, $\mathbf{x} \in R^{10000}$.
- Given a new picture, we want to answer: is it a dog or a cat?
- This is a Kaggle competition (https://www.kaggle.com/c/dogs-vs-cats)
- The idea is to test the CAPTCHA system is safe from machine learning attack.
- Simple for human, dog, or cat. It used to be difficult for machines, with 82.7% accuracy (Golle, 2008) before competition and 98.9% afterwards.

Statistical learning

- Two random variable $\mathbf{x} \in \boldsymbol{\mathcal{X}}$ and $y \in \mathcal{Y}$ are jointly distributed according to some fixed but unknown distribution $P(\mathbf{x}, y)$.
- $\boldsymbol{\mathcal{X}} = \mathbb{R}^d$ is an input (instance) space, and $\mathcal{Y} = \{-1, +1\}$ is an output (label) space.
- Examples are given in pair $(\mathbf{x}, y) \in \boldsymbol{\mathcal{X}} \times \mathcal{Y}$ generated by sampling from distribution $P(\mathbf{x}, y)$.
- A hypothesis class $\mathcal{H}$ is a set of function to be explored, for example, a set of hyperplanes in the feature space $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = 0, \mathbf{w} \in \mathbb{R}^d$.
- The goal of statistical learning is to provide an estimator $f \in \mathcal{H} : \boldsymbol{\mathcal{X}} \to \mathcal{Y}$ that predicts the best output y for an input $\mathbf{x}$.

Loss function

- Loss function measures the goodness of an estimator $f(\mathbf{x})$ on a single training example $(\mathbf{x}_i, y_i)$

$$\ell(f(\mathbf{x}_i), y_i) : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}^+.$$

- It is a monotonic bounded function defined on an estimation and the ground truth.
- The simplest one is hamming loss, which return 1 if the estimation is the same as the ground truth, 0 otherwise.

$$\ell_{hamming}(f(\mathbf{x}_i), y_i) = \mathbf{1}_{\{\mathbf{y}_i = f(\mathbf{x}_i)\}}$$

- Many other loss functions: hinge loss in support vector machines (Cortes and Vapnik, 1995), 0/1 loss in structured SVM (Tsochantaridis et al., 2004), squared loss in ridge regression (Hoerl and Kennard, 2000), exponential loss in ADABOOST (Schapire and Singer, 1999), logistic loss in logistic regression (Chen and Rosenfeld, 1999).

Empirical risk minimization

- In the end, we want to minimize the true risk of an estimator over a domain $\mathcal{X} \times \mathcal{Y}$

$$R_{true}(f) = \int_{(\mathbf{x},y)\in\mathcal{X}\times\mathcal{Y}} \ell(y, f(\mathbf{x}))P(\mathbf{x}, y)\, d_{\mathbf{x}} d_y. \quad \text{Test error}$$

- It is impossible to compute the true risk directly, as $P(\mathcal{X}, \mathcal{Y})$ is known.
- We are given a set of m training examples $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$.
- The empirical risk of an estimator $f \in \mathcal{H}$ over $\mathcal{S}$ is

$$R_{emp}(f) = \frac{1}{m}\sum_{i=1}^{m} \ell(y_i, f(\mathbf{x}_i)). \quad \text{Training error}$$

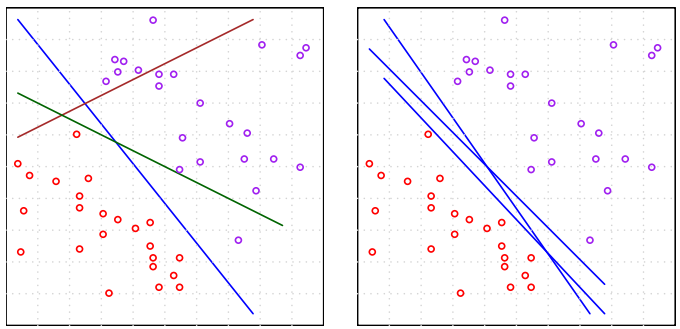
- Empirical risk minimization strategy (Vapnik, 1992) suggests we should search for an estimator that minimizes the empirical risk with regards to the training data.
- Vapnik (1992) showed that true risk is upper bounded by empirical risk

$$R_{true}(f) \leq R_{emp}(f) + \sqrt{\frac{h(\log(\frac{2m}{h}+1) - \log(\frac{\eta}{4}))}{m}}.$$

Empirical risk minimization

An example

- Training examples from two classes are located in 2 dimensional space.
- We are searching for a linear function (hyperplane) to separate two classes.
- Empirical minimization principle is applied.



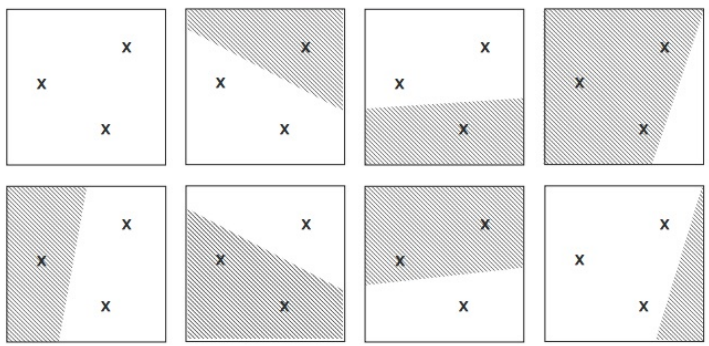
- We will not get a unique solution!

Regularized learning

- Empirical risk minimization is
  - ill-posed, no unique solution.
  - overfitting, high dimensional feature space, small number of examples.
- Regularization theory (Evgeniou et al., 1999, 2002) aims to minimize

$$\mathcal{J}(f) = \underbrace{\mathcal{R}_{emp}(f)}_{\text{empirical risk}} + \underbrace{\lambda \cdot \Omega(f)}_{\text{regularization}},$$

  the latter part controls the complexity (capacity) of the function f.
- The complexity is defined by its VC dimension as the maximum number of points that can be separated by all possible way by the set of functions.
- For example, the VC dimension of a hyperplane in $\mathbb{R}^2$ is 3.



Regularizer

- Vapnik (1992) shown that, the set of decision functions $f(\mathbf{x}) = \langle \mathbf{w}, \mathbf{x} \rangle + b = 0$ defined on $\boldsymbol{\mathcal{X}}$ such that $||\mathbf{w}|| \leq A$ has a VC dimension $h \leq R^2 A^2$.
- This suggests we should minimize the norm of the feature weight $||\mathbf{w}||$ in order to control the complexity of the linear function class.
- Popular regularizers to control the complexity
  - L_2-norm regularizer, ridge regression (Hoerl and Kennard, 2000), logistic regression (Chen and Rosenfeld, 2000), and support vector machines (Cortes and Vapnik, 1995)

$$\Omega_{L_2}(f) = ||\mathbf{w}||_2 = \left(\sum_{i=1}^{d} |\mathbf{w}[i]|^2 \right)^{\frac{1}{2}}.$$

  - L_1-norm regularizer, LASSO (Tibshirani, 1994)

$$\Omega_{L_1}(f) = ||\mathbf{w}||_1 = \sum_{i=1}^{d} |\mathbf{w}[i]|.$$

- Other regularizer: graph regularizer (Zhou et al., 2004).

Support Vector Machines

History

- Support Vector Machines (SVMs) was introduced by Boser et al. (1992); Cortes and Vapnik (1995).
- SVM is firmly rooted from statistical learning theory (Vapnik, 1998).
- Good empirical performance in many applications (e.g., bioinformatics, computer vision, text classification).
- A large and diverse communities: e.g., machine learning, optimization, learning theory, statistics, applied computer science.
- Centralized website at http://www.kernel-machines.org
- Text books e.g.,
  - An introduction to support vector machines and other kernel-based learning methods by Cristinini and Shawe-Taylor (2000).
  - Kernel methods for pattern analysis by Shawe-Taylor and Cristianini (2004).

Support Vector Machine

Preliminaries

- We focus on classification, though SVMs can do more than classification.
- Examples are given in pairs $(\mathbf{x}, y) \in \boldsymbol{\mathcal{X}} \times \mathcal{Y}$, where $\mathbf{x} \in \boldsymbol{\mathcal{X}}$ is an object and $y \in \mathcal{Y}$ is a class label.
- $\boldsymbol{\mathcal{X}}$ is an input space $\boldsymbol{\mathcal{X}} = \mathbb{R}^d$.
- We consider binary classification for now, setting $\mathcal{Y} = \{-1, +1\}$.
- A training set of m training example $\mathcal{S} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$.
- We want to learn a mapping function $f : \boldsymbol{\mathcal{X}} \to \mathcal{Y}$.
- The mapping function is a hyperplane in the d-dimension feature space $\boldsymbol{\mathcal{F}}$

$$f(x) = \langle \mathbf{w}, \mathbf{x} \rangle + b = 0.$$

- For an unknown example $\mathbf{x}_0$, the predicted label is given by

$$\mathbf{sign}(\langle \mathbf{w}, \mathbf{x}_0 \rangle + b).$$

Support Vector Machines

Regularized risk minimization

- We want to minimize a function

$$\mathcal{J}(f) = \underbrace{\mathcal{R}_{emp}(f)}_{\text{empirical risk}} + \underbrace{\lambda \cdot \Omega(f)}_{\text{regularization}}$$

- Use hamming loss $\ell_{hamming}$ and L_2 regularizer

$$\mathcal{J}(f) = \underbrace{\frac{1}{m}\sum_{i=1}^{m} \ell_{hamming}(f(\mathbf{x}_i), y_i)}_{\text{empirical risk}} + \underbrace{||\mathbf{w}||^2}_{regularization} .$$

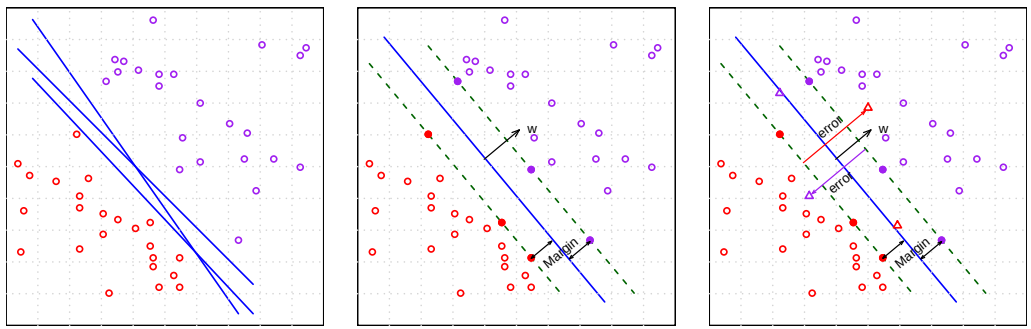
- $h_{hamming}(f(\mathbf{x}_i), y_i) = 1$ if $f(\mathbf{x}_i) \neq y_i$, 0 otherwise.
- Assume we can perfectly separate the data, we can rewrite the optimization problem as

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 \\ \text{s.t.} \quad & y_i(\langle \mathbf{w}_i, \mathbf{x}_i \rangle + b) \geq 1, \forall i \in \{1, \cdots, m\}. \end{aligned}$$

- This is known as hard-margin SVMs.

Geometric interpretation

- We are actually looking for a hyperplane with maximum margin between examples of two classes.
- Max-margin principle
  - Robustness: small perturbation in training data does not change the classification, decision boundary only depends on the support vectors.
  - Performance: large margin lead to low error on unseen data.
- What if there is no perfect separation?



Soft-margin SVMs

- For non-separable case, we write the optimization problem as

$$\begin{aligned} \min_{\mathbf{w},b} \quad & \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{m}\xi_i \\ \text{s.t.} \quad & y_i\left(\langle \mathbf{w}, \mathbf{x}_i\rangle + b\right) \geq 1-\xi_i, i \in \{1,\cdots,m\}, \xi \geq 0. \end{aligned}$$

- This is known as soft-margin SVMs.
- It is the same as the original objective

$$\mathcal{J}(f) = \underbrace{\frac{1}{m}\sum_{i=1}^{m}\ell_{hinge}(f(\mathbf{x}_i), y_i)}_{\text{empirical risk}} + \underbrace{||\mathbf{w}||^2}_{\textit{regularization}} .$$

- Hamming loss is replaced by hinge loss

$$\ell_{hinge}(f(\mathbf{x}_i), y_i) = \mathbf{max}(0, 1 - y_i(f(\mathbf{x}_i) + b)).$$

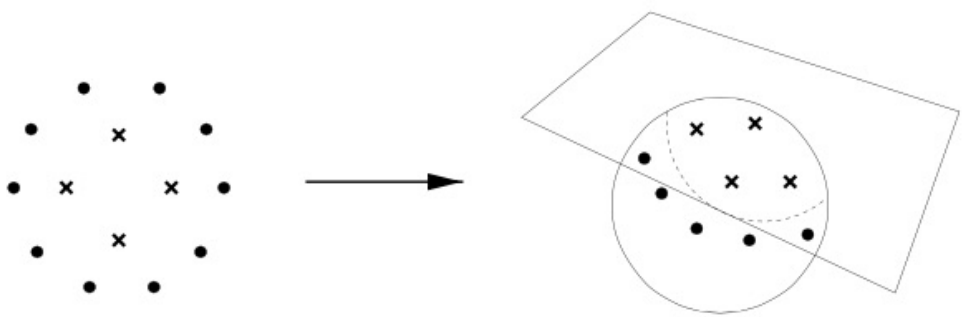
Non-linear SVMs

- Linear functions are sometimes not powerful enough.
- Data are not linearly separable in the original feature space.
- SVMs solution: map data into a richer feature space including non-linear features, then construct a hyperplane in that space.
- Define a input feature map

$$\mathbf{x} \in \boldsymbol{\mathcal{X}} = \mathbb{R}^d \xrightarrow{\varphi} \varphi(\mathbf{w}) \in \boldsymbol{\mathcal{H}} = \mathbb{R}^{d'}.$$

- Learn the map from $\varphi(\mathbf{x})$ to y

$$f(\mathbf{x}) = \langle \mathbf{w}, \varphi(\mathbf{x}) \rangle + b, \mathbf{w} \in \mathbb{R}^{d'}$$



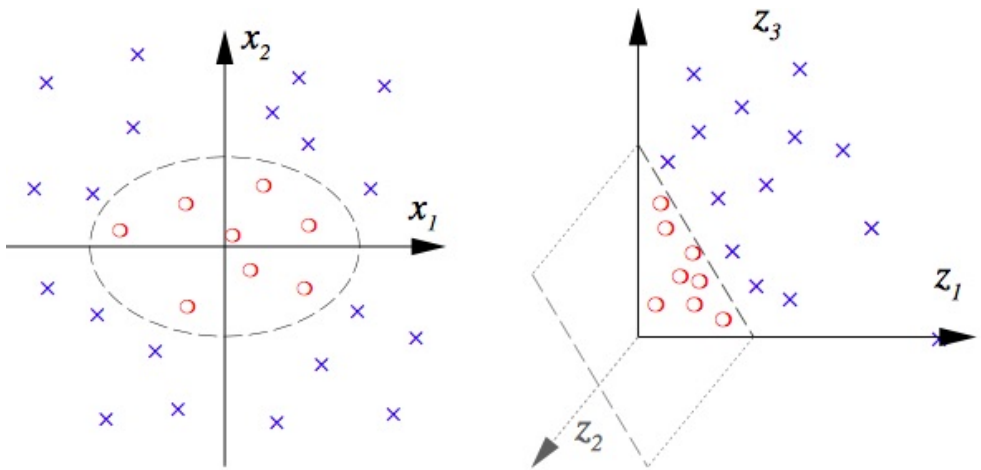
Example: polynomial feature map

- Data points locate in two dimensional space, $\mathbf{x} \in \mathbb{R}^2$.
- We cannot separate the data point with a line.
- Define a feature map:

$$\boldsymbol{\varphi} : \mathbb{R}^2 \to \mathbb{R}^3$$

$$\mathbf{x} = (x_1, x_2) \to \boldsymbol{\varphi}(\mathbf{x}) \stackrel{\text{def}}{=} (x_1^2, \sqrt{2}x_1x_2, x_2^2)$$

- We can separate the data points with a hyperplane in the high dimensional feature space $\mathbb{R}^3$.



Kernel methods

- Dual form of SVMs

$$\max_{\alpha} \quad \sum_{i=1}^{m} \alpha_i - \sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \underbrace{\langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle}_{k(\mathbf{x}_i, \mathbf{x}_j)}$$

$$\textbf{s.t.} \quad 0 \leq \alpha_i \leq C, \forall i \in \{1, \cdots, m\}.$$

- We do not need to know $\varphi(\mathbf{x}_i)$ and $\varphi(\mathbf{x}_j)$, we only need the results of the inner product $\langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle$.
- Kernel is a function $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$

$$k(\mathbf{x}_i, \mathbf{x}_j) = \langle \varphi(\mathbf{x}_i), \varphi(\mathbf{x}_j) \rangle,$$

  $\varphi : \mathcal{X} \to \mathcal{H}$ is the input feature map.
- Kernel function is used to make (implicit) nonlinear feature map.
- Kernel function can be evaluated in original space $\mathcal{X}$ without working explicitly in high the dimensional space $\mathcal{H}$.

An example of kernel function

- Degree two polynomial kernel

$$k(\mathbf{x}, \mathbf{z}) = (\langle \mathbf{x}, \mathbf{z} \rangle)^2, \mathbf{x}, \mathbf{z} \in \mathbb{R}^2.$$

- Evaluate kernel function in original feature space $\mathbb{R}^2$

$$\begin{aligned} k(\mathbf{x}, \mathbf{z}) &= (\langle (x_1, x_2), (z_1, z_2) \rangle)^2 \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 \end{aligned}$$

- Explicit feature map

$$\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2 \xrightarrow{\boldsymbol{\varphi}} \boldsymbol{\varphi}(\mathbf{x}) = (x_1^2, \sqrt{2} x_1 x_2, x_2^2) \in \mathbb{R}^3$$
$$\mathbf{z} = (z_1, z_2) \in \mathbb{R}^2 \xrightarrow{\boldsymbol{\varphi}} \boldsymbol{\varphi}(\mathbf{z}) = (z_1^2, \sqrt{2} z_1 z_2, z_2^2) \in \mathbb{R}^3$$

- Evaluate kernel function with explicit feature map

$$\begin{aligned} \langle \boldsymbol{\varphi}(\mathbf{x}), \boldsymbol{\varphi}(\mathbf{z}) \rangle &= \langle (x_1^2, \sqrt{2} x_1 x_2, x_2^2), (z_1^2, \sqrt{2} z_1 z_2, z_2^2) \rangle \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 \end{aligned}$$

Popular kernels

- Linear kernel

$$k_{linear}(\mathbf{x}_i, \mathbf{x}_j) = \langle \mathbf{x}_i, \mathbf{x}_j \rangle.$$

- Polynomial kernel

$$k_{poly}(\mathbf{x}_i, \mathbf{x}_j) = (\langle \mathbf{x}_i, \mathbf{x}_j \rangle + b)^d.$$

- Gaussian kernel

$$k_{gaussian}(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(\frac{||\mathbf{x}_i - \mathbf{x}_j||^2}{2\delta^2}\right).$$

SVMs Softwares

- LibSVM, in C++, by Chang and Lin (2011)
- SVMLight, in C, by Joachims (1998)
- Torch (http://bengio.abracadoudou.com/SVMTorch.html)
- Spider (http://people.kyb.tuebingen.mpg.de/spider/)
- Weka (http://www.cs.waikato.ac.nz/ml/weka/index.html)

Molecular classification

-

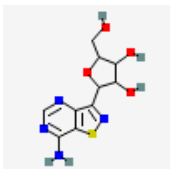
Multi-task classification

-

Max-margin conditional random field

Preliminaries

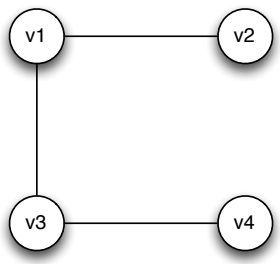
- Training examples come in pairs $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \times \mathcal{Y}$ sampled from some unknown joint distribution $P(\mathbf{x}, \mathbf{y})$.
- $\mathbf{x} \in \mathcal{X}$ is an instant from input domain $\mathcal{X}$ (e.g., a document, a molecule, a picture).
- $\mathbf{y} = (y_1, \cdots, y_k) \in \mathcal{Y}$ is a vector of labels from output domain (e.g., categories of a documents, potentials of a molecules, annotations of a picture).
- The output domain $\mathcal{Y} = \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_k$, $\mathcal{Y}_j = \{-1, +1\}$ and $\mathcal{Y} = \{-1, +1\}^k$.
- $\mathbf{y}$ is called a multilabel, y_i is called a microlabel of the multilabel.
- The goal is to obtain a function $f : \mathcal{X} \to \mathcal{Y}$.



$\rightarrow \qquad \mathbf{y} = (+1, -1, +1, +1)$

Output graph and feature maps

- Observe an output $G = (E, V)$, a vertex $v_i \in V$ correspond to a microlabel y_i, an edge $(v_i, v_j) \in E$ correspond to the pairwise correlation of the microlabel y_i and y_j.
- Feature maps
  - $\boldsymbol{\varphi}(\mathbf{x})$ is the input feature map, $\boldsymbol{\varphi}(\mathbf{x}) \in \mathbb{R}^d$.
  - $\boldsymbol{\psi}(\mathbf{y})$ is the output feature map that maps the multilabel $\mathbf{y}$ into a set of edges and labels.
  - Joint feature map $\boldsymbol{\phi}(\mathbf{x}, \mathbf{y}) = \varphi(\mathbf{x}) \times \psi(\mathbf{y})$, $\times$ is Kronecker tensor.



$$\mathbf{y} = (+1, -1, +1, +1) \qquad \boldsymbol{\psi}(\mathbf{y}) =$$

Bibliography

Boser, B. E., Guyon, I. M., and Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of the 5th Annual Workshop on Computational Learning Theory*, pages 144–152. ACM.

Chang, C.-C. and Lin, C.-J. (2011). Libsvm: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27:1–27:27.

Chen, S. and Rosenfeld, R. (1999). *A Gaussian Prior for Smoothing Maximum Entropy Models*. PhD thesis, Computer Science Department, Carnegie Mellon University. Technical Report CMU-CS-99-108.

Chen, S. and Rosenfeld, R. (2000). A survey of smoothing techniques for me models. *IEEE Transactions on Speech and Audio Processing*, 8(1):37–50.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297.

Cristinini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

Bibliography (cont.)

Evgeniou, T., Poggio, T., Pontil, M., and Verri, A. (2002). Regularization and statistical learning theory for data analysis. *Comput. Stat. Data Anal.*, 38(4):421–432.

Evgeniou, T., Pontil, M., and Poggio, T. (1999). A unified framework for regularization networks and support vector machines. Technical report, Cambridge, MA, USA.

Golle, P. (2008). Machine learning attacks against the asirra captcha. In *Proceedings of the 15th ACM Conference on Computer and Communications Security*, CCS '08, pages 535–542, New York, NY, USA. ACM.

Hoerl, A. E. and Kennard, R. W. (2000). Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 42(1):80–86.

Joachims, T. (1998). Making large-scale svm learning practical. LS8-Report 24, Universität Dortmund, LS VIII-Report.

Schapire, R. and Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336.

Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Bibliography (cont.)

Tibshirani, R. (1994). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society, Series B*, 58:267–288.

Tsochantaridis, I., Hofmann, T., Joachims, T., and Altun, Y. (2004). Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21th International Conference on Machine Learning (ICML 2004)*, pages 823–830. ACM.

Vapnik, V. (1992). Principles of risk minimization for learning theory. In Moody, J., Hanson, S., and Lippmann, R., editors, *Advances in Neural Information Processing Systems 4*, pages 831–838. Morgan-Kaufmann.

Vapnik, V. N. (1998). *Statistical Learning Theory*. Wiley-Interscience.

Zhou, D., Bousquet, O., Lal, T. N., Weston, J., and Schölkopf, B. (2004). Learning with local and global consistency. In Thrun, S., Saul, L., and Schölkopf, B., editors, *Advances in Neural Information Processing Systems 16*, pages 321–328. MIT Press.