LAGO: A Computationally Efficient Method for Statistical Detection

Mu Zhu

University of Waterloo

Acknowledgment

- Co-authors:
  - Wanhua Su;
  - Hugh A. Chipman.
- Research support:
  - NSERC;
  - MITACS;
  - CFI;
  - Acadia Centre for Mathematical Modelling and Computation.
- Others: William J. Welch, R. Wayne Oldford, Jerry F. Lawless, Mary Thompson, S. Young.

Agenda

1. The statistical detection problem.
2. Average precision.
3. Drug discovery and high throughput screening.
4. LAGO.
5. Radial basis function (RBF) networks.
6. Support vector machines (SVMs).
7. Results.

The Detection Problem

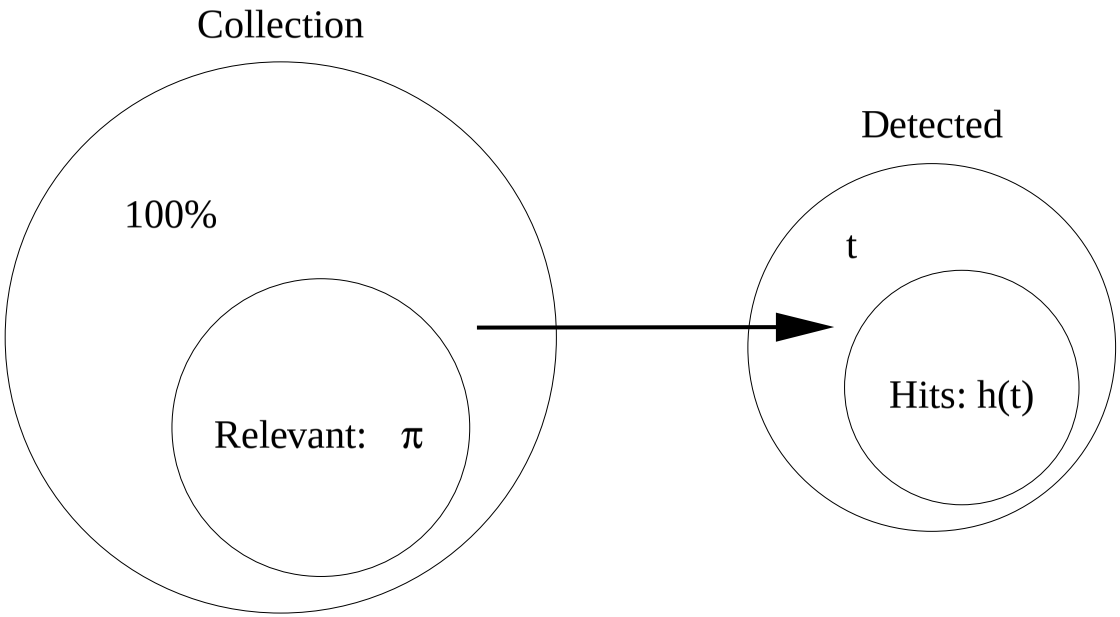


Figure 1: Illustration of a typical detection operation. A small fraction π of the entire collection $\mathcal{C}$ is of interest (relevant). An algorithm detects a fraction t from $\mathcal{C}$, out of which $h(t)$ is relevant.

The Typical Paradigm

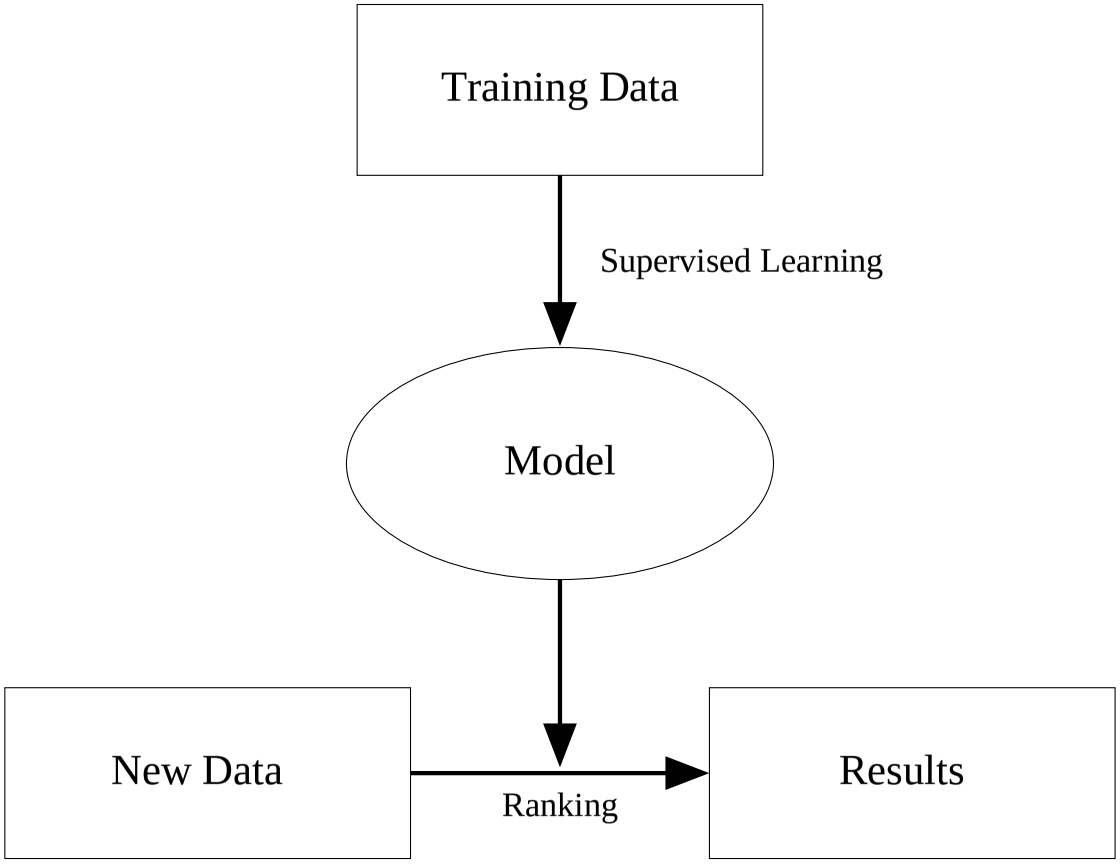


Figure 2: Illustration of the typical modelling and prediction process.

The Hit Curve

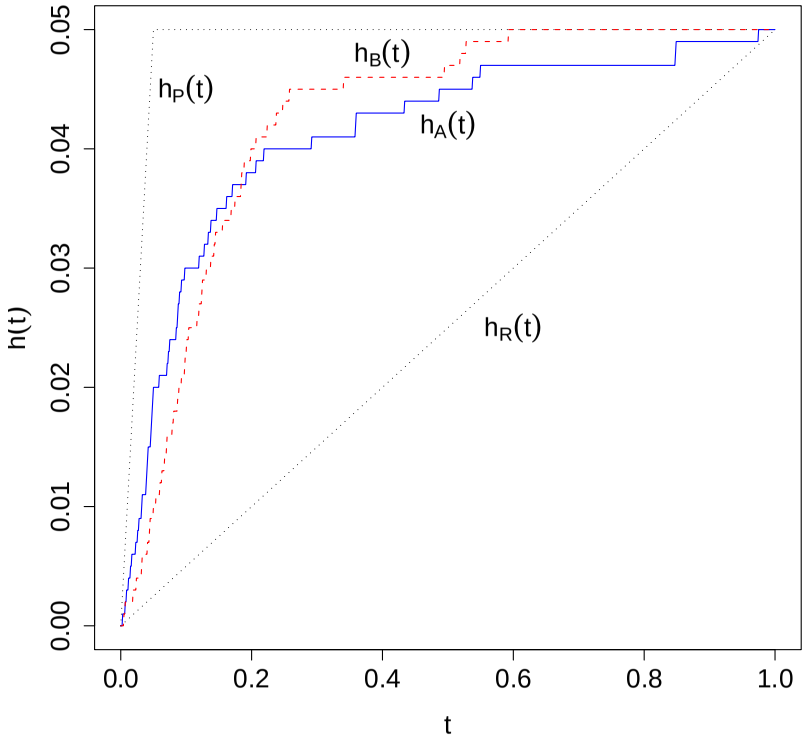


Figure 3: Illustration of some hit curves. Note that $h_A(t)$ and $h_B(t)$ cross each other; $h_P(t)$ is an ideal curve produced by a perfect algorithm; $h_R(t)$ corresponds to the case of random detection.

The Average Precision

Let $h(t)$ be the hit curve; let

$$r(t) = \frac{h(t)}{\pi} \quad \text{and} \quad p(t) = \frac{h(t)}{t}.$$

Then,

$$\text{Average Precision} = \int p(t)dr(t). \tag{1}$$

In practice, $h(t)$ takes values only at a finite number of points $t_i = i/n$, $i = 1, 2, ..., n$. Hence, the integral (1) is replaced with a finite sum

$$\int p(t)dr(t) = \sum_{i=1}^{n} p(t_i)\Delta r(t_i) \tag{2}$$

where $\Delta r(t_i) = r(t_i) - r(t_{i-1})$.

A Simple Example

	Algorithm A			Algorithm B		
Item (i)	Hit	$p(t_i)$	$\Delta r(t_i)$	Hit	$p(t_i)$	$\Delta r(t_i)$
1	1	1/1	1/3	1	1/1	1/3
2	1	2/2	1/3	0	1/2	0
3	0	2/3	0	0	1/3	0
4	1	3/4	1/3	1	2/4	1/3
5	0	3/5	0	1	3/5	1/3

$$\text{AP(A)} = \sum_{i=1}^{5} p(t_i)\Delta r(t_i) = \left(\frac{1}{1} + \frac{2}{2} + \frac{3}{4}\right) \times \frac{1}{3} \approx 0.92.$$

$$\text{AP(B)} = \sum_{i=1}^{5} p(t_i)\Delta r(t_i) = \left(\frac{1}{1} + \frac{2}{4} + \frac{3}{5}\right) \times \frac{1}{3} = 0.70.$$

High Throughput Screening (HTS)

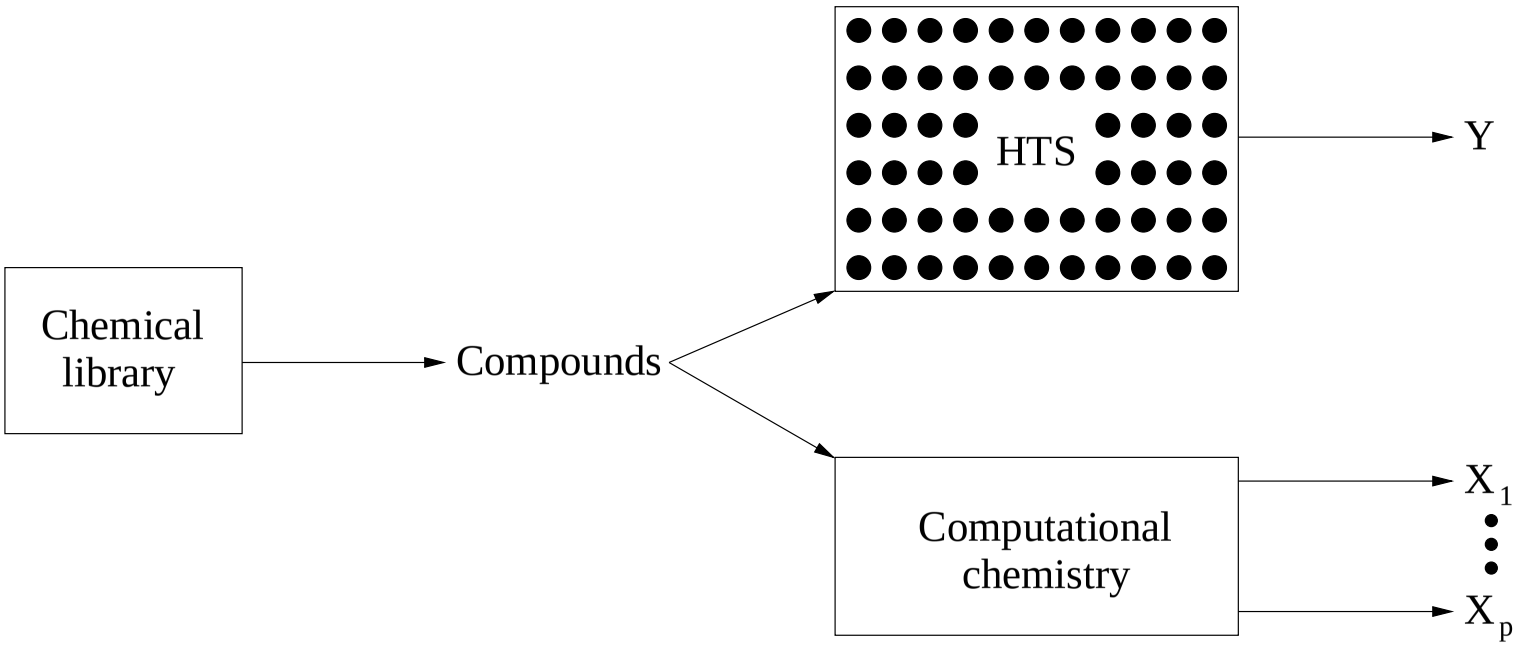


Figure 4: Illustration of the high throughput screening process.

Drug Discovery Data

Original data from National Cancer Institute (NCI) with predictors calculated by GlaxoSmithKline, Inc.

1. $n = 29,812$ chemical compounds, of which only 608 are active against the HIV virus.
2. $d = 6$ chemometric descriptors of the molecular structure, known as BCUT numbers.
3. Using stratified sampling, randomly split of the data to produce a training set and a test set (each with $n = 14,906$ and 304 active compounds).
4. Tuning parameters selected using 5-fold cross-validation on the training set, and compare performance on the test set.

Ranking Functions

1. Given a vector of predictors $\mathbf{x}$, the posterior probability

$$g(\mathbf{x}) \equiv P(y = 1|\mathbf{x}) = \frac{\pi_1 p_1(\mathbf{x})}{\pi_1 p_1(\mathbf{x}) + \pi_0 p_0(\mathbf{x})} \tag{3}$$

   is arguably a good ranking function, i.e., items with a high probability of being relevant should be ranked first.

2. As far as ranking is concerned, all monotonic transformations of g are clearly equivalent, so it suffices to focus on the ratio function

$$f(\mathbf{x}) = \frac{p_1(\mathbf{x})}{p_0(\mathbf{x})} \tag{4}$$

   since the function g is of the form

$$g(\mathbf{x}) = \frac{af(\mathbf{x})}{af(\mathbf{x}) + 1}$$

   for some constant a not depending on $\mathbf{x}$, which is a monotonic transformation of f.

Two Assumptions

A1. For all practical purposes the density function $p_1(\mathbf{x})$ can be assumed to have bounded local support, possibly over a number of disconnected regions, $\mathcal{S}_\gamma \subset \mathbb{R}^d$, $\gamma = 1, 2, ..., \Gamma$, in which case the support of p_1 can be written as

$$\mathcal{S} = \bigcup_{\gamma=1}^{\Gamma} \mathcal{S}_\gamma \subset \mathbb{R}^d.$$

A2. For every observation $\mathbf{x}_i \in C_1$, there are at least a certain number of observations, say m, from C_0 in its immediate local neighborhood; moreover, the density function $p_0(\mathbf{x})$ in that neighborhood can be assumed to be relatively flat in comparison with $p_1(\mathbf{x})$.

In order to build a predictive model for
statistical detection problems,
it suffices to

- estimate $p_1(\mathbf{x})$ alone and
- adjust $p_1(\mathbf{x})$ locally depending on $p_0(\mathbf{x})$ nearby.

Assume $x \in \mathbb{R}$ is a scalar.

$$\Downarrow$$

Generalize to $\mathbf{x} \in \mathbb{R}^d$ for $d > 1$.

Step 1: Estimating p_1

1. Use an adaptive bandwidth kernel estimator:

$$\hat{p}_1(x) = \frac{1}{n_1} \sum_{y_i=1} \mathcal{K}\left(x; x_i, r_i\right). \tag{5}$$

2. For each $x_i \in C_1$, choose r_i adaptively to be the average distance between x_i and its K-nearest neighbors from C_0, i.e.,

$$r_i = \frac{1}{K} \sum_{w_j \in N(x_i,K)} |x_i - w_j|. \tag{6}$$

   The notation $N(x_i, K)$ is used to refer to the set that contains the K-nearest class-0 neighbors of x_i. The number K is a tuning parameter to be selected empirically, e.g., with cross-validation.

Original Inspiration

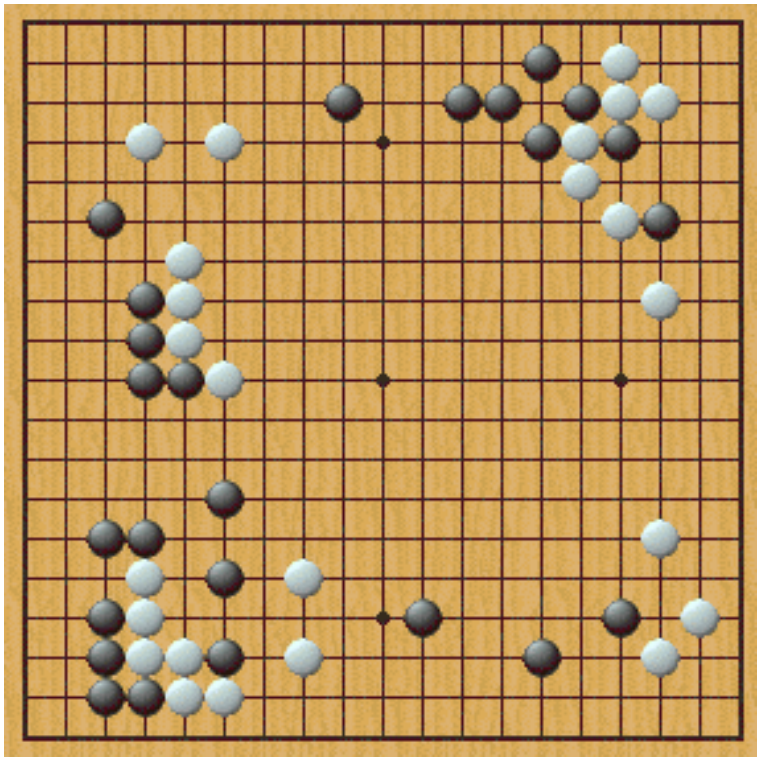


Figure 5: The ancient Chinese game of Go is a game in which each player tries to claim as many territories as possible on the board. Image taken from http://go.arad.ro/Introducere.html.

Step 2: Local Adjustment of p_1

1. View the kernel density estimate (5) as a mixture and adjust each mixture component (centered at x_i) accordingly.

2. Estimate p_0 locally around every $x_i \in C_1$, say $p_0(x; x_i)$, and divide it into $\mathcal{K}(x; x_i, r_i)$. Assumption A2 implies that we can simply estimate $p_0(x; x_i)$ locally as a constant, say c_i. Hence, we obtain

$$\hat{f}(x) = \frac{1}{n_1} \sum_{y_i=1} \frac{\mathcal{K}(x; x_i, r_i)}{c_i} \tag{7}$$

   as an estimate of the ranking function $f(x)$.

An Idealized Situation

Instead of saying $p_0(x; x_i) \approx c_i$, we shall explicitly assume that, for every $x_i \in C_1$, there exist i.i.d. observations $w_1, w_2, ..., w_m$ from C_0 that can be taken to be uniformly distributed on the interval $[x_i - 1/2c_i, x_i + 1/2c_i]$.

Theorem 1 *Let x_0 be a fixed observation from class 1. Suppose $w_1, w_2, ..., w_m$ are i.i.d. observations from class 0 that are uniformly distributed around x_0, say on the interval $[x_0 - 1/2c_0, x_0 + 1/2c_0]$. If r_0 is the average distance between x_0 and its K nearest neighbors from class 0 ($K < m$), then we have*

$$E(r_0) = \frac{K+1}{4(m+1)c_0}.$$

Implications of the Theorem

- Can assume there are at least m observations from C_0 distributed *approximately* uniformly around every $x_i \in C_1$.
- For $K < m$, can conclude r_i is *approximately* proportional to $1/c_i$.
- Since r_i is already computed, there is no need to estimate c_i; we simply use

$$\hat{f}(x) = \frac{1}{n_1} \sum_{y_i=1} r_i \; \mathcal{K}\left(x; x_i, r_i\right). \tag{8}$$

A Short Summary

1. Estimation of p_1:

$$\hat{p}_1(x) = \frac{1}{n_1} \sum_{y_i=1} \mathcal{K}\left(x; x_i, r_i\right).$$

2. Adjustment of p_1 according to p_0 nearby:

$$\hat{f}(x) = \frac{1}{n_1} \sum_{y_i=1} \frac{\mathcal{K}(x; x_i, r_i)}{c_i} \quad \Longrightarrow \quad \hat{f}(x) = \frac{1}{n_1} \sum_{y_i=1} r_i \; \mathcal{K}\left(x; x_i, r_i\right).$$

$$\Downarrow$$

LAGO = "Locally Adjusted GO-kernel density estimator."

Extension to $\mathbb{R}^d$

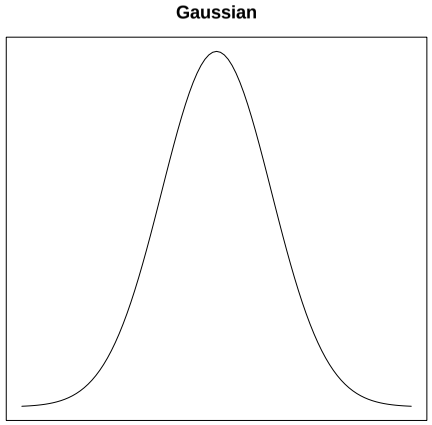
1. For every training observation in class 1, $\mathbf{x}_i \in C_1$, compute a specific bandwidth vector $\mathbf{r}_i = (r_{i1}, r_{i2}, ..., r_{id})^T$, where r_{ij} is the average distance between $\mathbf{x}_i$ and its K-nearest class-0 neighbors in the jth dimension.

2. For every new observation $\mathbf{x} = (x_1, x_2, ..., x_d)^T$ where a prediction is required, score and rank $\mathbf{x}$ according to:

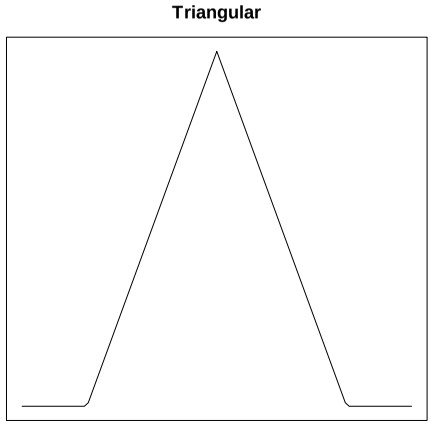
$$f(\mathbf{x}) = \frac{1}{n_1} \sum_{y_i=1} \left\{ \prod_{j=1}^{d} r_{ij} \mathcal{K}\left(x_j; x_{ij}, r_{ij}\right) \right\}, \tag{9}$$

   which uses the Naive Bayes principle (Hastie *et al.* 2001, Section 6.6.3).

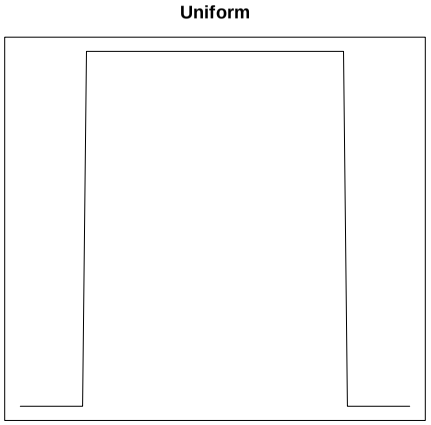
Some Kernel Functions



$$f(u) \propto \exp\left(-\frac{u^2}{2}\right)$$



$$f(u) = 1 - |u|$$
$$|u| \leq 1$$



$$f(u) = 1$$
$$|u| \leq 1$$

Radial Basis Function Networks

A radial basis function (RBF) network has the form:

$$f(\mathbf{x}) = \sum_{i=1}^{n} \beta_i \mathcal{K}(\mathbf{x}; \boldsymbol{\mu}_i, \mathbf{r}_i), \tag{10}$$

where $\mathcal{K}(\mathbf{x}; \boldsymbol{\mu}, \mathbf{r})$ is a kernel function centered at location $\boldsymbol{\mu}$ with radius (or bandwidth) vector $\mathbf{r} = (r_1, r_2, ..., r_d)^T$. Clearly, in order to construct an RBF network we must specify the centers $\boldsymbol{\mu}_i$ and the radii $\mathbf{r}_i$ for $i = 1, 2, ..., n$.

General Parameterization

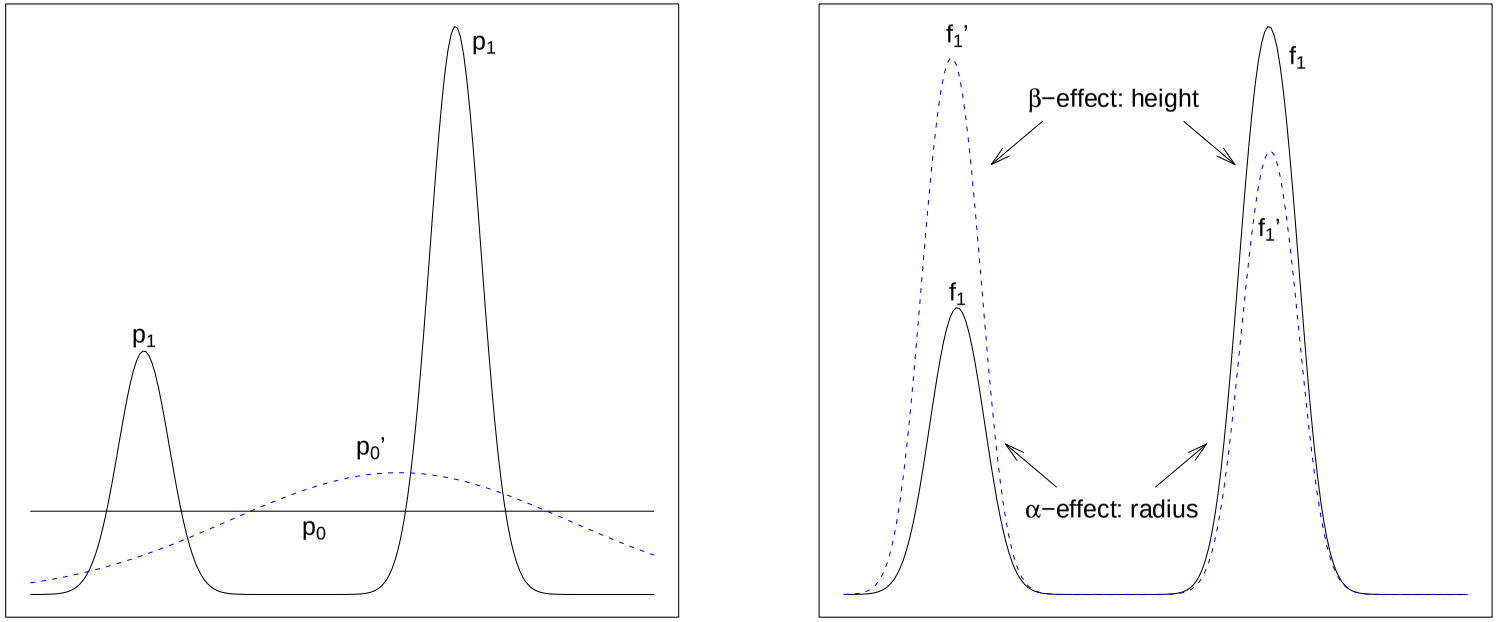


Figure 6: Illustration. Left: Density functions p_0 and p_1. Right: The ratio function $f(x)$.

Parameterizing the α- and β-Effects

- Take a kernel function belonging to a location-scale family:

$$\frac{1}{r_i}\mathcal{K}\left(\frac{x-x_i}{r_i}\right).$$

  Can explicitly parameterize the α- and β-effects as follows:

$$r_i^{\beta'}\frac{1}{\alpha r_i}\mathcal{K}\left(\frac{x-x_i}{\alpha r_i}\right) \quad \propto \quad r_i^{\beta'-1}\mathcal{K}\left(\frac{x-x_i}{\alpha r_i}\right) \quad \equiv \quad r_i^{\beta}\mathcal{K}\left(\frac{x-x_i}{\alpha r_i}\right).$$

- In constructing the LAGO model, we have in effect argued that $\beta = 0$ (or $\beta' = 1$).

The LAGO Model

- For relatively large K, can usually obtain a model with very similar performance by setting $\alpha > 1$ and using a much smaller K.
- Hence by keeping α, can restrict ourselves to a much narrower range when selecting K by cross-validation.
- The final form of the LAGO model is:

$$f(\mathbf{x}) = \frac{1}{n_1} \sum_{y_i=1} \left\{ \prod_{j=1}^{d} r_{ij} \mathcal{K}\left(x_j; x_{ij}, \alpha r_{ij}\right) \right\}, \tag{11}$$

  with two tuning parameters, K and α.

Separating Hyperplanes

- Given $\mathbf{x}_i \in \mathbb{R}^d$, a hyperplane in $\mathbb{R}^d$ is characterized by

$$f(\mathbf{x}) = \boldsymbol{\beta}^T \mathbf{x} + \beta_0 = 0.$$

- Given $y_i \in \{-1, +1\}$ (two classes), a hyperplane is a separating hyperplane if there exists $c > 0$ such that

$$y_i(\boldsymbol{\beta}^T \mathbf{x}_i + \beta_0) \geq c \quad \forall i.$$

- A hyperplane can be reparameterized by scaling, e.g.,

$$\boldsymbol{\beta}^T \mathbf{x} + \beta_0 = 0 \quad \text{is the same as} \quad s(\boldsymbol{\beta}^T \mathbf{x} + \beta_0) = 0.$$

- A separating hyperplane satisfying

$$y_i(\boldsymbol{\beta}^T \mathbf{x}_i + \beta_0) \geq 1 \quad \forall i$$

(i.e., scaled so that $c = 1$) is sometimes called a canonical separating hyperplane (Cristianini and Shawe-Taylor 2000).

Separating Hyperplanes and Margins

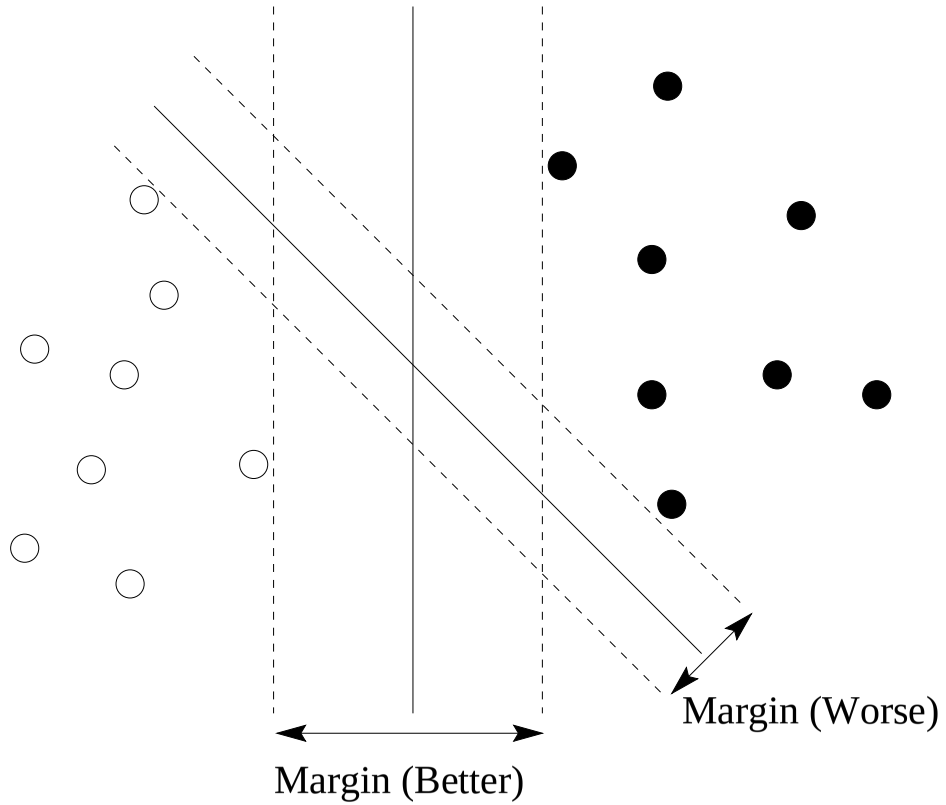


Figure 7: Two separating hyperplanes, one with a larger margin than the other.

The Support Vector Machine

- It can be calculated that a canonical separating hyperplane has margin equal to $\frac{1}{\|\boldsymbol{\beta}\|}$.
- The support vector machine (SVM) finds a "best" (maximal margin) canonical separating hyperplane to separate the two classes (labelled $+1$ and -1) by solving

$$\min \quad \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \gamma \sum_{i=1}^{n} \xi_i$$

$$\text{s.t.} \quad \xi_i \geq 0 \quad \text{and} \quad y_i(\boldsymbol{\beta}^T \mathbf{x}_i + \beta_0) \geq 1 - \xi_i \quad \forall i.$$

ASVM for Unbalanced Classes

Let w_0 and w_1 be class weights; extend the optimization problem to be:

$$\min \quad \frac{1}{2}\|\boldsymbol{\beta}\|^2 + \gamma_1 \sum_{y_i=1} \xi_i + \gamma_0 \sum_{y_i=0} \xi_i$$

$$\text{s.t.} \quad \xi_i \geq 0 \quad \text{and} \quad y_i(\boldsymbol{\beta}^T \mathbf{x}_i + \beta_0) \geq 1 - \xi_i \quad \forall i,$$

where $\gamma_0 = \gamma w_0$ and $\gamma_1 = \gamma w_1$.

SVM: Characterizing the Solution

- The solution for $\boldsymbol{\beta}$ is characterized by

$$\hat{\boldsymbol{\beta}} = \sum_{i \in SV} \hat{\alpha}_i y_i \mathbf{x}_i,$$

  where $\hat{\alpha}_i \geq 0$ $(i = 1, 2, ..., n)$ are solutions to the dual optimization problem and SV, the set of "support vectors" with $\hat{\alpha}_i > 0$ strictly positive.

- This means the resulting hyperplane can be written as

$$\hat{f}(\mathbf{x}) = \hat{\boldsymbol{\beta}}^T \mathbf{x} + \hat{\beta}_0 = \sum_{i \in SV} \hat{\alpha}_i y_i \mathbf{x}_i^T \mathbf{x} + \hat{\beta}_0 = 0.$$

SVMs and RBF Networks

- Can replace the inner product $\mathbf{x}_i^T\mathbf{x}$ with a kernel function $\mathcal{K}(\mathbf{x};\mathbf{x}_i)$ to get a nonlinear decision boundary:

$$\hat{f}(\mathbf{x}) = \sum_{i\in SV} \hat{\alpha}_i y_i \mathcal{K}(\mathbf{x};\mathbf{x}_i) + \hat{\beta}_0 = 0.$$

  The boundary is linear in the space of $h(\mathbf{x})$ where $h(\cdot)$ is such that $\mathcal{K}(\mathbf{u};\mathbf{v}) = \langle h(\mathbf{u}), h(\mathbf{v})\rangle$ is the inner product in the space of $h(\mathbf{x})$.

- Hence SVM can be viewed as an automatic way of constructing an RBF network (Schölkopf *et al.* 1997).

Performance Results: Drug Discovery Data

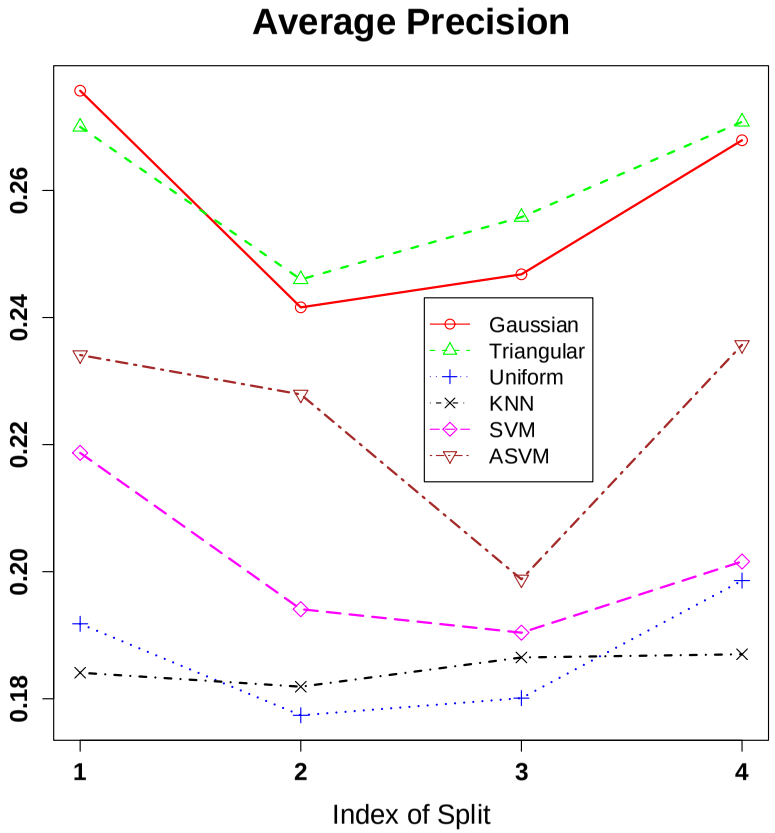


Figure 8: The average precision of all algorithms evaluated on the test data.

Performance Results: ANOVA Set-up

Let $\mu_K, \mu_S, \mu_A, \mu_U, \mu_T$ and μ_G be the average result of K-NN, SVM, ASVM, and LAGO using the uniform kernel, the triangular kernel and the Gaussian kernel, respectively.

Contrast	Expression	Estimate
Cntr1	$\mu_T - \mu_G$	0.0027
Cntr2	$\mu_G - \mu_A$	0.0339
Cntr3	$\mu_A - \mu_S$	0.0230
Cntr4	$\mu_S - (\mu_K + \mu_U)/2$	0.0157
Cntr5	$\mu_U - \mu_K$	0.0014

Performance Results: ANOVA Summary

Source	SS ($\times 10^{-4}$)	df	MS ($\times 10^{-4}$)	F_0	P-Value
Methods	233.504	5	46.701	64.307	<0.0001
Cntr1	*0.140*	*1*	*0.140*	*0.193*	*0.6664*
Cntr2	*22.916*	*1*	*22.916*	*31.556*	*<0.0001*
Cntr3	*10.534*	*1*	*10.534*	*14.505*	*0.0017*
Cntr4	*6.531*	*1*	*6.531*	*8.994*	*0.0090*
Cntr5	*0.036*	*1*	*0.036*	*0.050*	*0.8258*
Splits	18.877	3	6.292	8.664	0.0014
Error	10.893	15	0.726		
Total	263.274	23			

Hit Curves: Drug Discovery Data

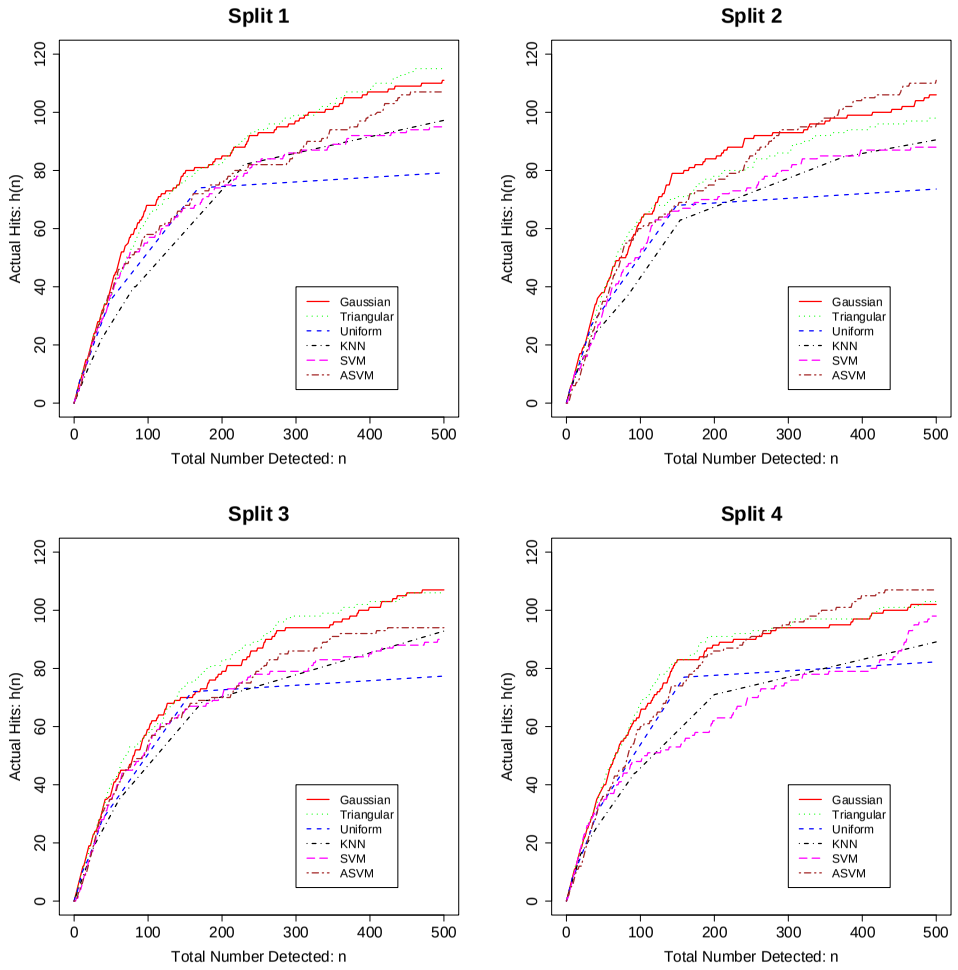


Figure 9: Only the initial part of the curves (up to $n = 500$) are shown.

Main Conclusions

☞

$$(\text{Triangle LAGO} \sim \text{Gaussian LAGO}) \succ \succ \text{ASVM} \succ \text{SVM} \succ \succ (\text{Uniform LAGO} \sim \text{KNN}).$$

☞

Computationally, ASVM is *extremely* expensive.

The Number of SVs

	SVM		ASVM	
	C_0	C_1	C_0	C_1
Split 1	11531	294	5927	291
Split 2	11419	303	3472	284
Split 3	11556	293	11706	290
Split 4	1863	293	6755	281
Total Possible	14602	304	14602	304

Empirical Evidence: $\beta = 0$

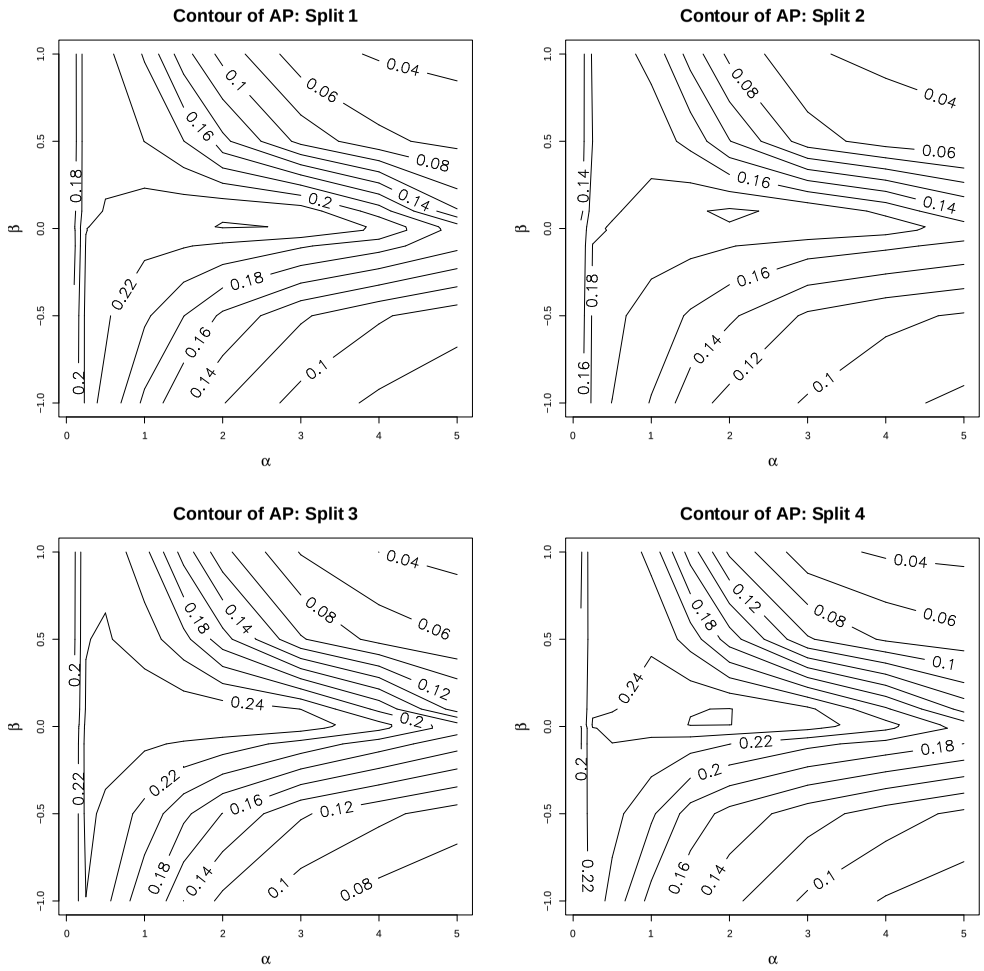


Figure 10: Choosing α and β (while fixing $K = 5$) using 5-fold CV.

References

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

Hastie, T. J., Tibshirani, R. J., and Friedman, J. H. (2001). *The Elements of Statistical Learning: Data-Mining, Inference and Prediction*. Springer-Verlag.

Schölkopf, B., Sung, K. K., Burges, C. J. C., Girosi, F., Niyogi, P., Poggio, T., and Vapnik, V. (1997). Comparing support vector machines with gaussian kernels to radial basis function classifiers. *IEEE Transactions on Signal Processing*, **45**(11), 2758–2765.

Zhu, M., Su, W., and Chipman, H. A. (2005). LAGO: A computationally efficient approach for statistical detection. Working Paper 2005-01, Department of Statistics and Actuarial Science, University of Waterloo. To appear in *Technometrics*.