Efficient Classification Based on Sparse Regression

Pardis Noorzad

Department of Computer Engineering and Information Technology
Amirkabir University of Technology

Supervisor: **Prof. Mohammad Rahmati**

MSc Thesis Defense – July 17, 2012

Outline

The SVM Optimization Problem

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{subject to} \quad \begin{cases} y_i(\mathbf{w}.\mathbf{x}_i - b) & \geq 1 - \xi_i \\ \xi_i & \geq 0 \end{cases} \qquad \text{for } i = 1, \cdots, n,$$

This can alternatively be written as

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i$$

$$\text{subject to} \quad \xi_i \geq \max\Big(0, 1 - y_i(\mathbf{w}.\mathbf{x}_i - b)\Big) \qquad \text{for } i = 1, \cdots, n,$$

Introduce the notation

$$\xi_i \geq \Big[1 - y_i(\mathbf{w}.\mathbf{x}_i - b)\Big]_+, \text{ where } [x]_+ = \max(0, x)$$

The SVM Optimization Problem: Continued

We will be working with the following formulation:

$$\min \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\Big[1 - y_i(\mathbf{w}.\mathbf{x}_i - b)\Big]_+$$

Which contains the terms:

- ℓ_2-regularization of the weights
- hinge loss

SVM is popular

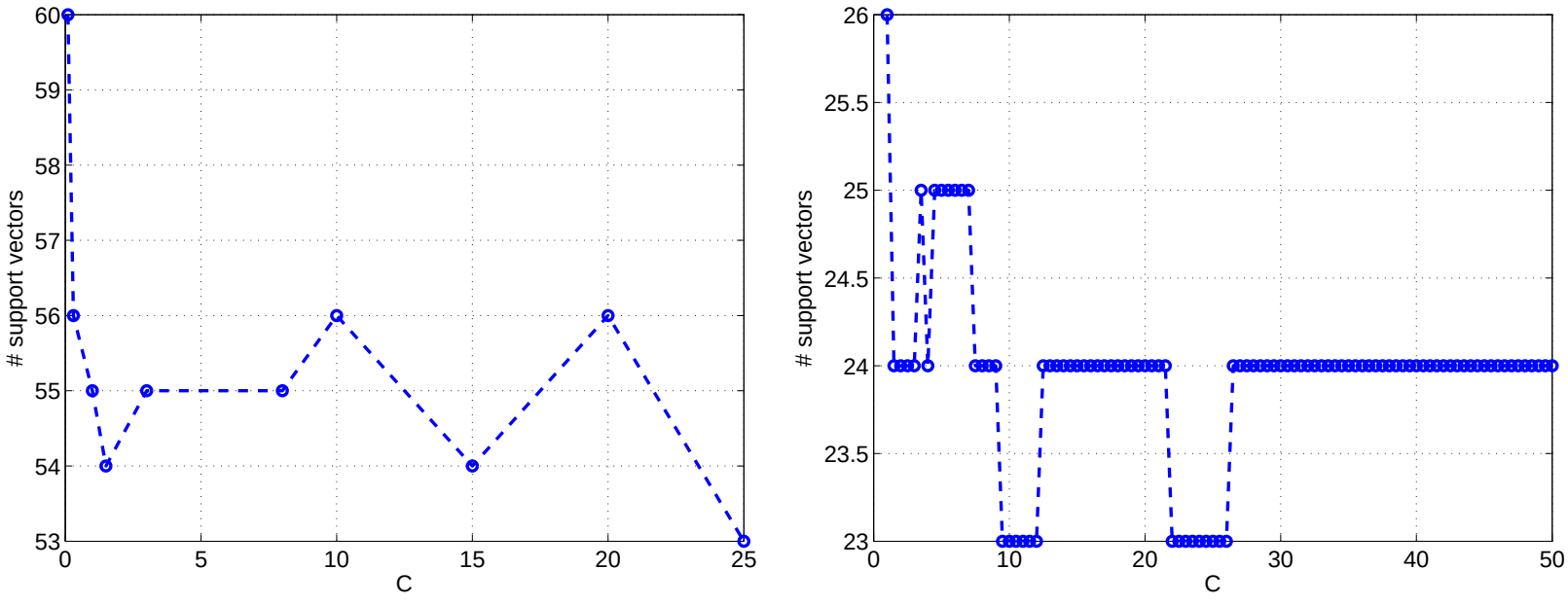
- SVM solution is **sparse**—due to the **hinge loss**
  - SVM uses a subset of training samples for prediction: called **support vectors**

$$f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i \mathbf{x}_i \mathbf{x} - b$$

- SVM can be employed for nonlinear classification —due to the **kernel trick**: $f(\mathbf{x}) = \sum_{i=1}^{n} \alpha_i y_i k(\mathbf{x}_i, \mathbf{x}) - b$
- SVM has excellent generalization performance —due to ℓ_2-regularization on the weights

BUT...

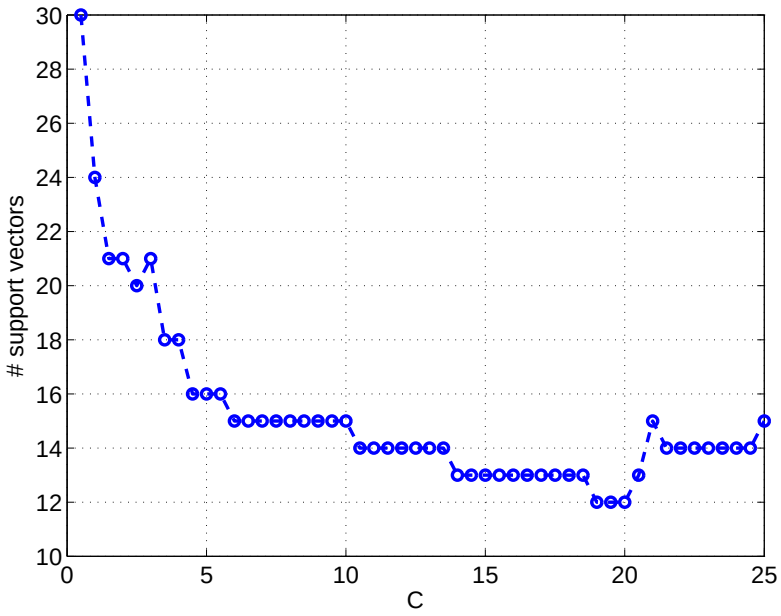
- SVM solution, α, is usually not sparse enough
  - sparsity is an issue because
    1. classification/testing/prediction time
    2. classifier space
  - more importantly, sparsity of α is not easily controlled



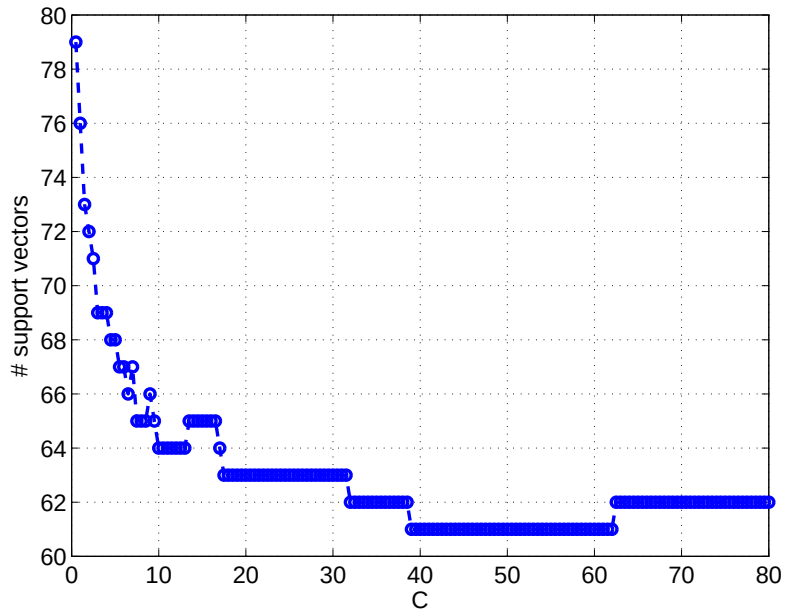
(a) Australian dataset

(b) Heart dataset

Figure: Graphs show that the number of support vectors does not have a meaningful relation to the hyperparameter C in the SVM optimization.



(a) Ionosphere dataset

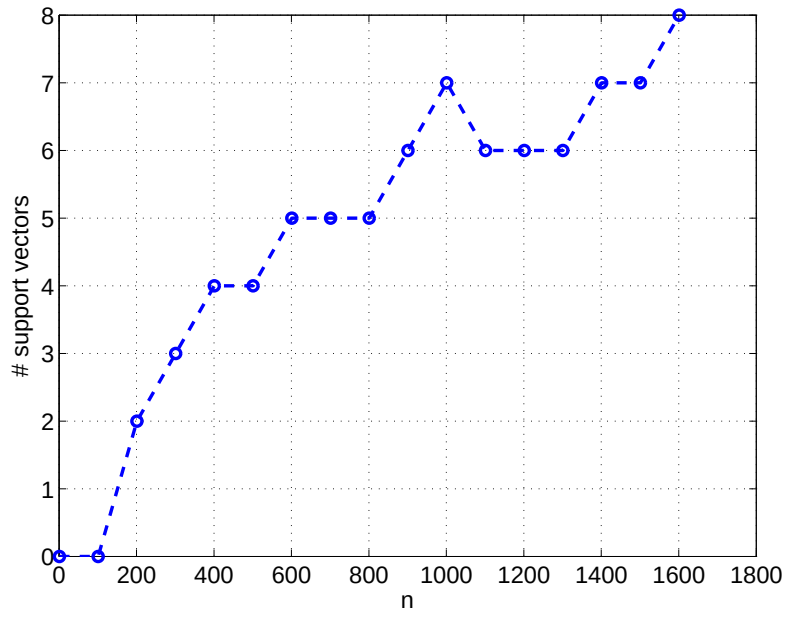


(b) Liver disorders dataset

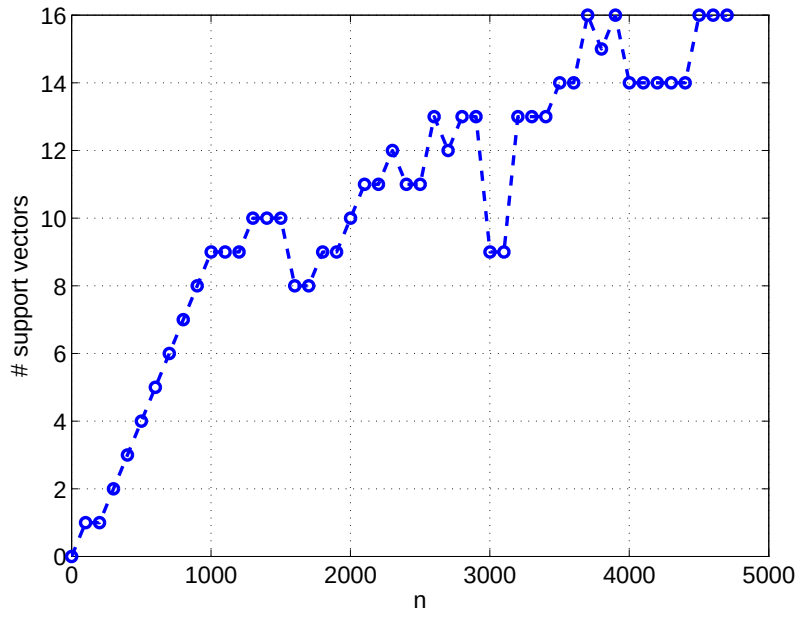
Figure: Graphs show that the number of support vectors does not have a meaningful relation to the hyperparameter C in the SVM optimization.

BUT...

- SVM solution, α, is usually not sparse enough
  - sparsity is an issue because
    1. classification/testing/prediction time
    2. classifier space
  - more importantly, sparsity of α is not controllable
  - number of support vectors grows with sample size

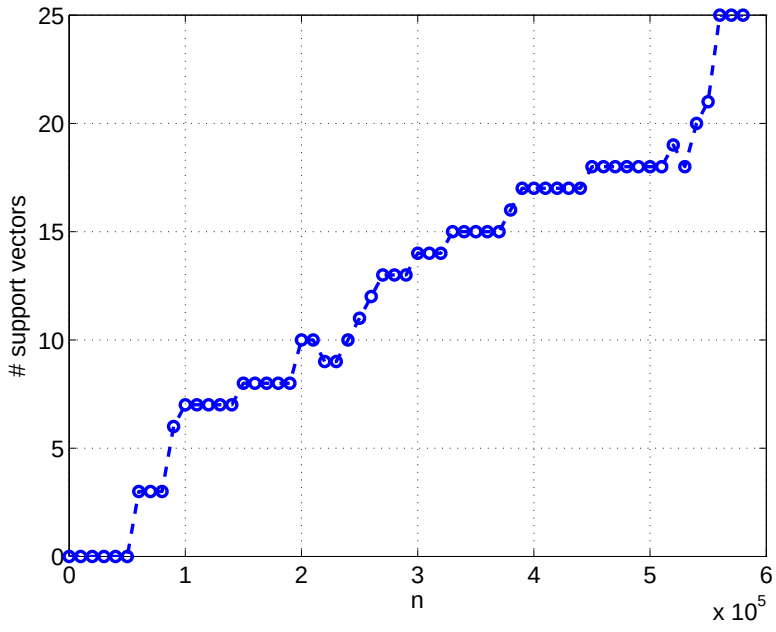


(a) Adult 1 dataset

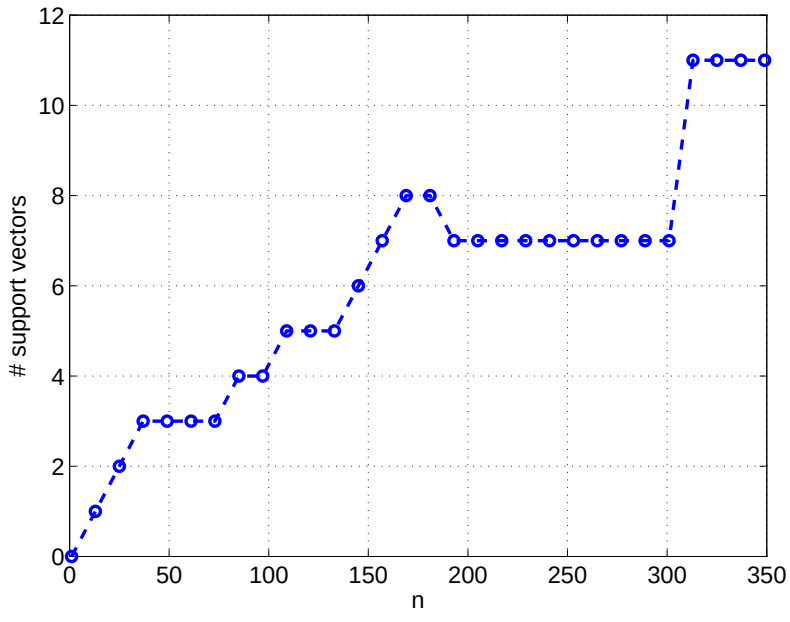


(b) Adult 4 dataset

Figure: Graphs show that the number of support vectors increase as the number of samples grow.



(a) Covertype dataset



(b) Ionosphere dataset

Figure: Graphs show that the number of support vectors increase as the number of samples grow.

BUT...

- SVM solution, α, is usually not sparse enough
  - sparsity is an issue because
    1. classification/testing/prediction time
    2. classifier space
  - more importantly, sparsity of α is not controllable
  - number of support vectors grows with sample size
- for most real applications, linear SVM is used
  - linear SVM is faster to train and test
  - especially for OVA or OVO
  - high-dimensional data is sparse

Related work in square loss minimization for classification

- In his PhD thesis, Rifkin (2002) claims hinge loss **is not the secret** to SVM's success
  - Rifkin proposes **Regularized Least Squares Classification** (RLSC)

$$\min_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 + \lambda \|\mathbf{w}\|^2$$

  - and the nonlinear case

$$\min_{\mathbf{c}} \|\mathbf{y} - \mathbf{K}\mathbf{c}\|^2 + \lambda \mathbf{c}^T \mathbf{K}\mathbf{c}$$

- BUT
  - resulting classifier is not sparse
  - nonlinear RLSC takes longer than SVM to train

Related work in ℓ_1-regularization for sparse classifiers

- Yuan et al. (2010) compare several sparse linear classifiers

$$\min_{\mathbf{w}} \|\mathbf{w}\|_1 + C \sum_{i=1}^{n} \xi(\mathbf{w}; \mathbf{x}_i, y_i)$$

- with the **logistic**, **hinge**, and **square hinge loss**
  - $\xi_{\log}(\mathbf{w}; \mathbf{x}_i, y_i) = \log\left(1 + \exp(-y\mathbf{w}^\mathsf{T}\mathbf{x})\right)$
  - $\xi_{L_1}(\mathbf{w}; \mathbf{x}_i, y_i) = \max\left(1 - y\mathbf{w}^\mathsf{T}\mathbf{x}, 0\right)$
  - $\xi_{L_2}(\mathbf{w}; \mathbf{x}_i, y_i) = \max\left(1 - y\mathbf{w}^\mathsf{T}\mathbf{x}, 0\right)^2$
- BUT
  - they don't consider those optimizing the square loss
  - the square loss has several good **computational** and **statistical** properties

Classification

(Devroye et al., 1996)

- Find a function $g : \mathbb{R}^p \to \{-1, 1\}$ which takes an *observation* $\mathbf{x} \in \mathbb{R}^p$ and assigns it to $y \in \{-1, 1\}$
- g is called a **classifier**
- Probability of error or probability of misclassification

$$L(g) = \mathbb{P}\{g(X) \neq Y\}$$

- The optimal classifier g^* is

$$g^* = \underset{g:\mathbb{R}^p \to \{1,\ldots,M\}}{\arg\min} \mathbb{P}\{g(X) \neq Y\}$$

- and is called the **Bayes classifier**.

Empirical Risk Minimization

- Minimizing $L(g)$ is only possible with the knowledge of the joint distribution of X and Y
- Given $\mathcal{T}_n = \{(\mathbf{x}_i, y_i) : i = 1, \ldots, n\}$ of n observations, assumed to be sampled i.i.d. from the distribution of (X, Y)
- An estimate of $L(g)$ is

$$L_n(g) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{\{g(\mathbf{x}_i) \neq y_i\}}$$

- called the **empirical error**—but it is intractable to compute

Empirical Risk Minimization

Continued

- Consider classifiers of the form

$$g_f(\mathbf{x}) = \begin{cases} -1 & \text{if } f(\mathbf{x}) < 0 \\ 1 & \text{otherwise} \end{cases}$$

- where $f : \mathbb{R}^p \to \mathbb{R}$ is a real-valued function in $\mathcal{F}$
- The probability of error of g_f is

$$\begin{aligned} L(g_f) = L(f) &= \mathbb{P}\{\text{sgn}(f(X)) \neq Y\} \\ &= \mathbb{P}\{Y f(X) \leq 0\} \\ &= \mathbb{E}\{\mathbb{I}_{\{Y f(X) \leq 0\}}\} \end{aligned}$$

- The quantity $yf(\mathbf{x})$ is called the **margin**

Empirical Risk Minimization

Continued

- Given $\mathcal{T}_n$, once can estimate $L(f)$ by $L_n(f)$

$$L_n(f) = \frac{1}{n} \sum_{i=1}^{n} \mathbb{I}_{\{y_i f(\mathbf{x}_i) \leq 0\}}$$

- where $\mathbb{I}_{\{yf(\mathbf{x}) \leq 0\}}$ is the **0-1 loss function**
- minimizing the empirical error is computationally intractable
- we seek to minimize a smooth convex upper bound of the 0-1 loss

Convex Loss

- The cost functional becomes

$$A(f) = \mathbb{E}\{\phi(Y f(X))\}$$

- with its corresponding empirical form being

$$A_n(f) = \frac{1}{n} \sum_{i=1}^{n} \phi(y_i f(\mathbf{x}_i)).$$

Convex Loss

Continued

- One can prove that the minimizer $f^*(x)$ of $A(f)$ is such that the induced classifier g_f

$$g^*(\mathbf{x}) = \begin{cases} -1 & \text{if } f^*(\mathbf{x}) < 0 \\ 1 & \text{otherwise} \end{cases}$$

- is the Bayes classifier (Zhang, 2004; Boucheron et al., 2005)—thereby proving Fisher consistency of convex cost functions

Table: Well-known convex loss functions and their corresponding minimizing function.

Loss function name	Form of $\phi(v)$	Form of $f_\phi^*(\eta)$
Square loss	$(1-v)^2$	$2\eta - 1$
Hinge loss	$\max(0, 1-v)$	$\text{sign}(2\eta - 1)$
Squared hinge loss	$\max(0, 1-v)^2$	$2\eta - 1$
Logistic loss	$\ln(1+\exp(-v))$	$\ln \frac{\eta}{1-\eta}$

Two important insights

- Convex cost functions are all Fisher consistent.
- SVM estimates $\operatorname{sign}(2\eta - 1)$, whereas a least squares classifier estimates $2\eta - 1$
  - thus giving us information about the confidence of its predictions
  - making it more suitable for OVA

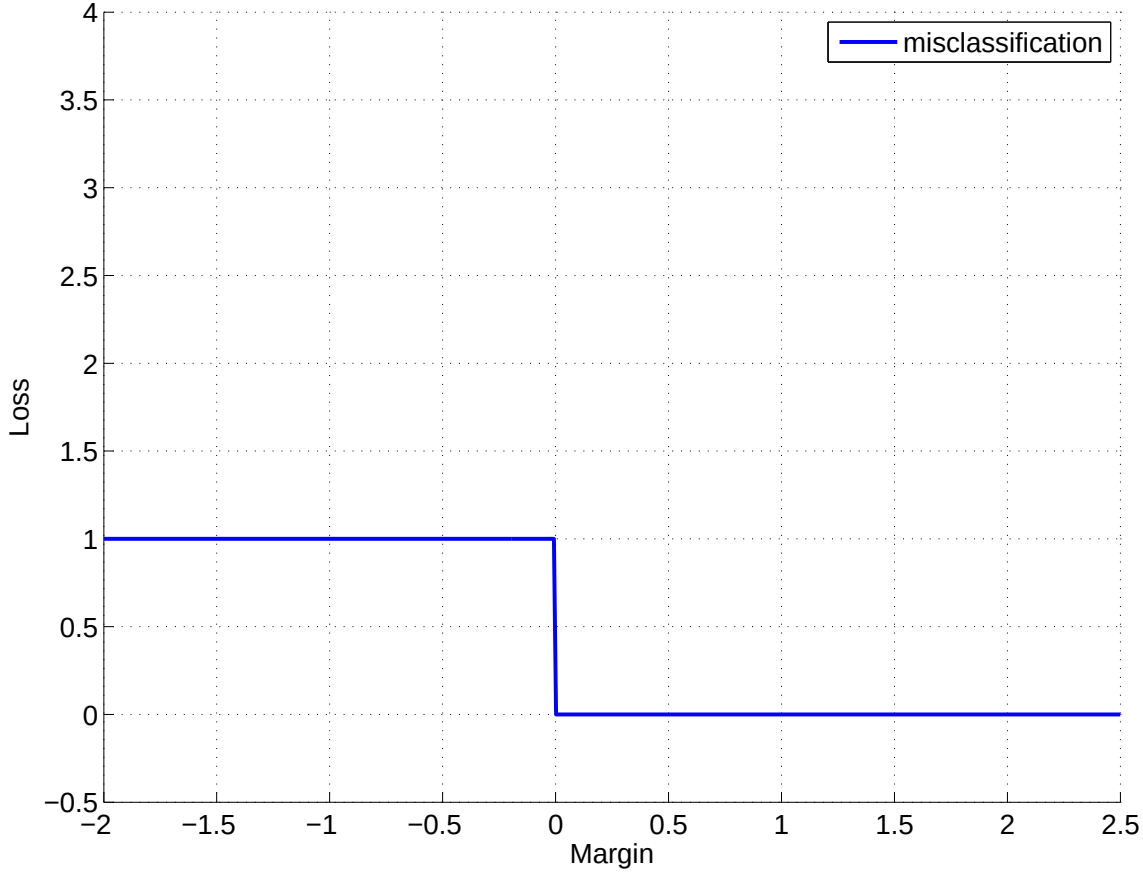


Figure: A comparison of convex loss functions. The misclassification loss is also shown.

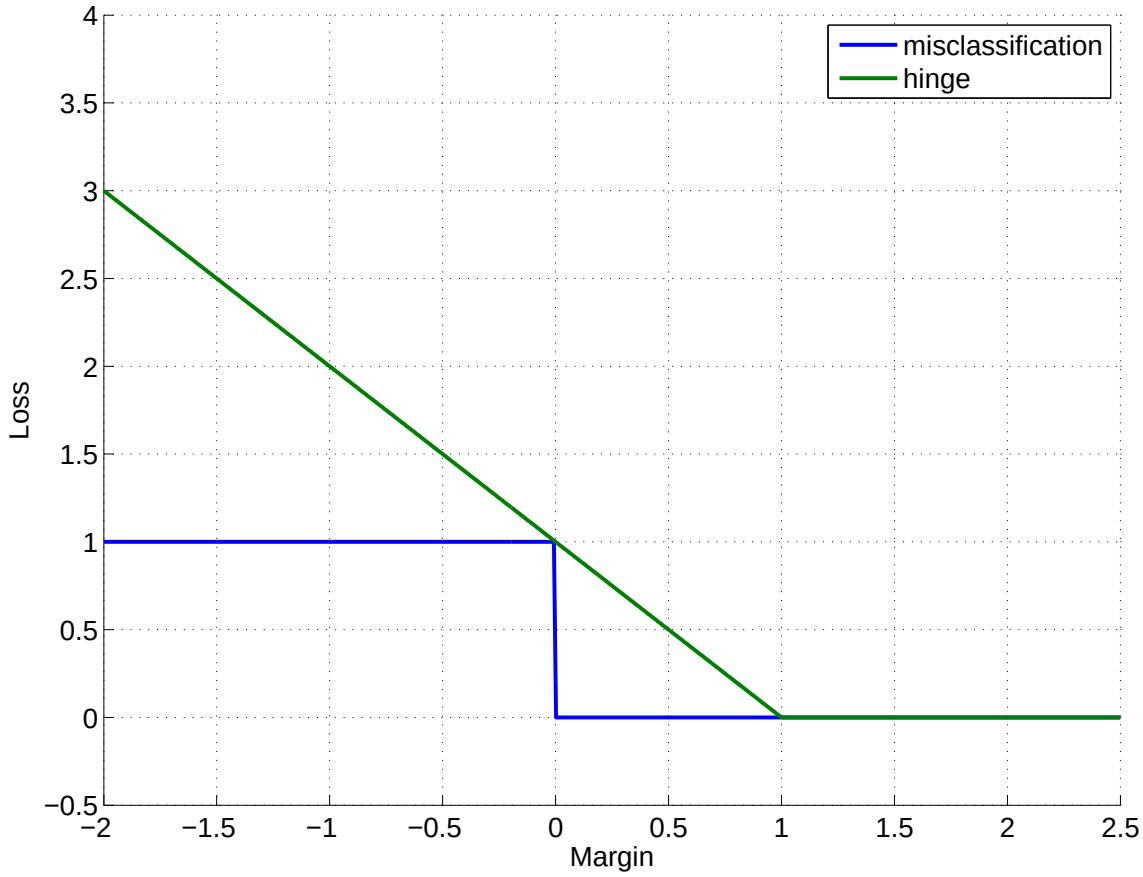


Figure: A comparison of convex loss functions. The misclassification loss is also shown.

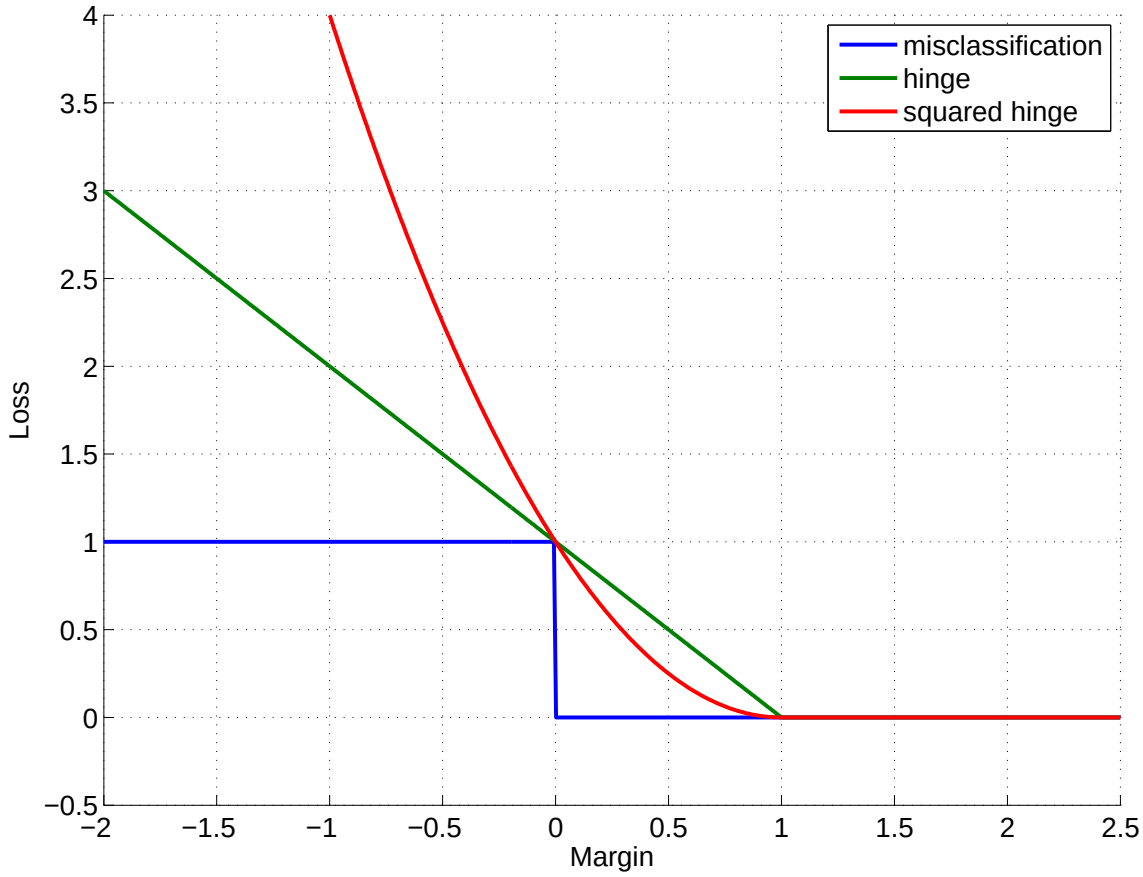


Figure: A comparison of convex loss functions. The misclassification loss is also shown.

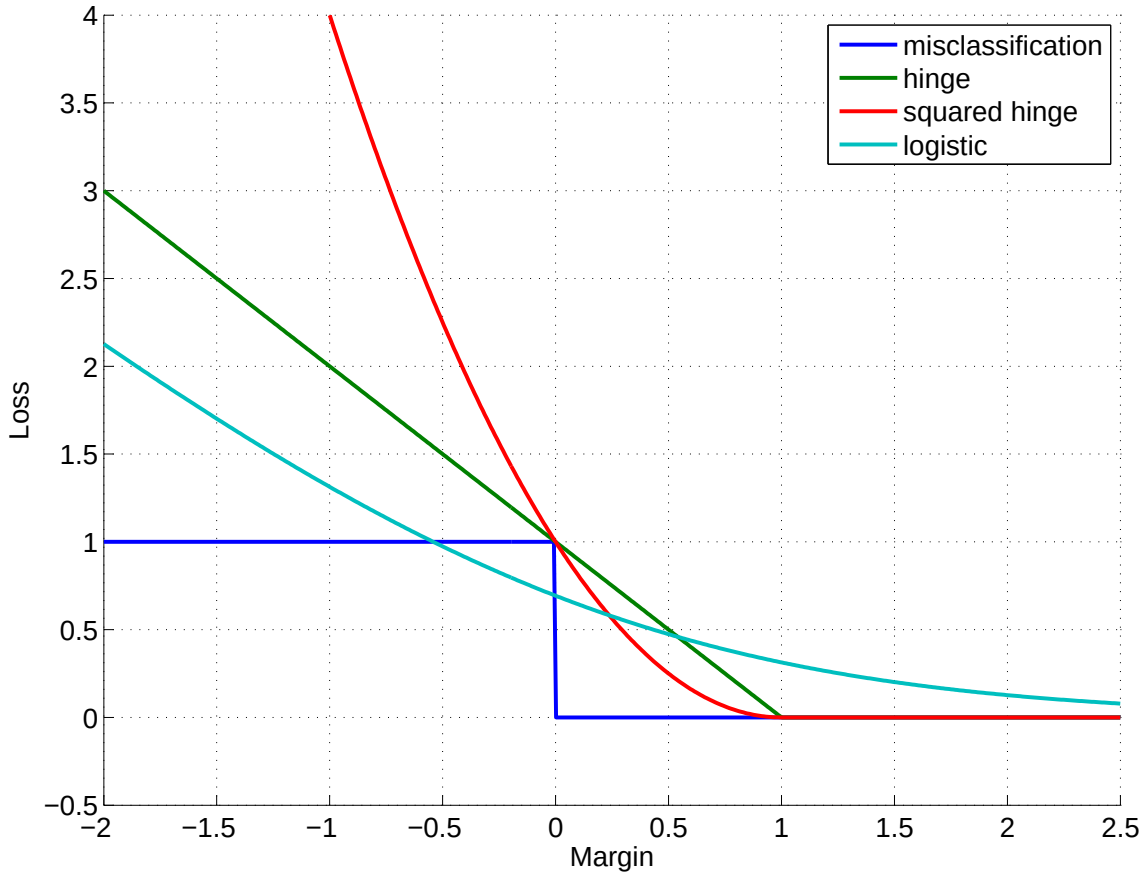


Figure: A comparison of convex loss functions. The misclassification loss is also shown.

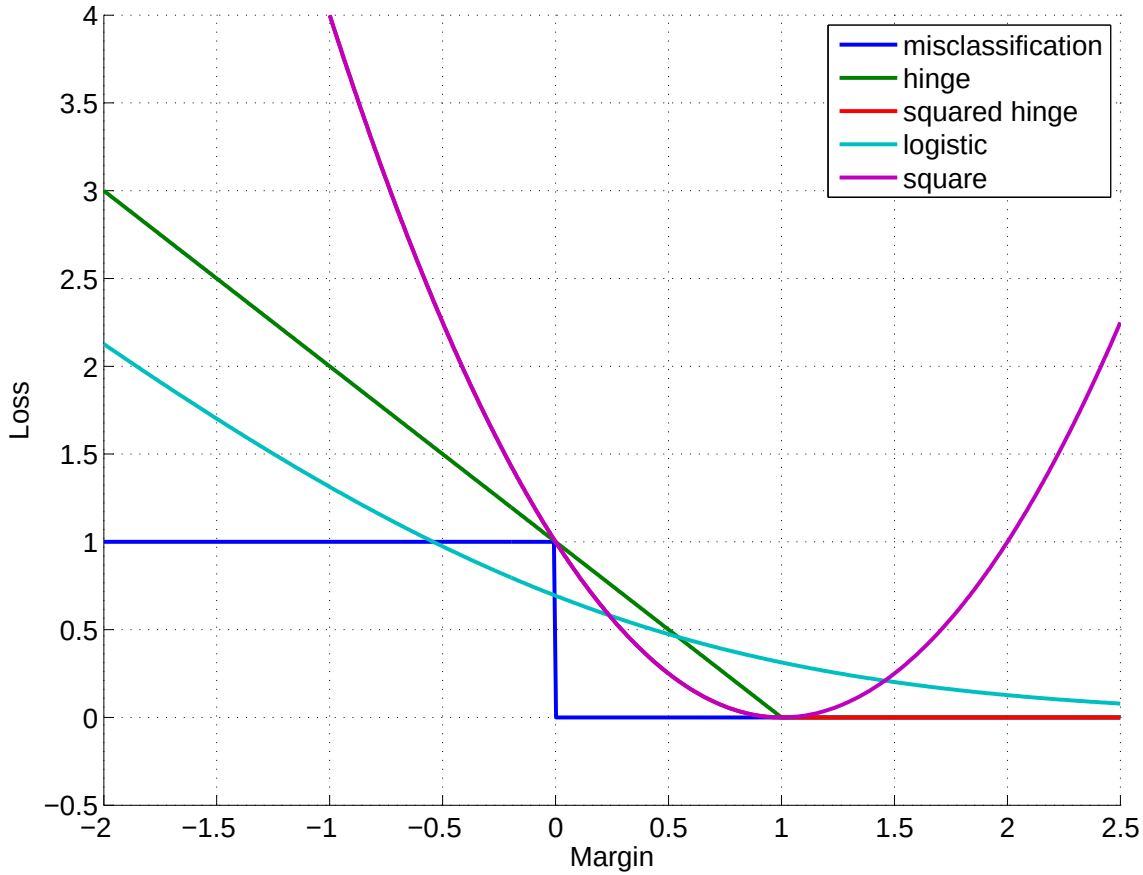


Figure: A comparison of convex loss functions. The misclassification loss is also shown.

The Setting

- We have the linear inverse problem

$$\mathbf{b} = \mathbf{A}\mathbf{x}$$

where $\mathbf{b} \in \mathbb{R}^n, \mathbf{x} \in \mathbb{R}^p, \mathbf{A} \in \mathbb{R}^{n \times p}$; $\mathbf{x}$ is unknown.

- Many problems in ML are linear inverse problems, for e.g.,
  - regression and classification: $\mathbf{y} = \mathbf{X}\mathbf{a}$, $\mathbf{a}$ is unknown;
  - sparse coding: $\mathbf{x} = \mathbf{D}\mathbf{a}$, $\mathbf{a}$ is unknown;

Solution

Take I

$$\mathbf{a} = \mathbf{X}^{-1}\mathbf{y}$$

- What's the problem here?
- $\mathbf{X}$ is almost never **invertible** in our problems:
  - needs to be square
  - needs to have full column rank

Ill-posedness

- **Case I**
  - If $n = p$ or $n > p$, we say that the system of equations is **overdetermined**.
  - In this case, the solution to the inverse problem does **not exist**.
- **Case II**
  - If $n < p$, the system is **underdetermined**,
  - and there exists **infinitely many** solutions.

Solution

Case I, Take II

- Instead of the equations, $\mathbf{y} = \mathbf{X}\mathbf{a}$, only minimize the residual,

$$\min_{\mathbf{a}} \|\mathbf{y} - \mathbf{X}\mathbf{a}\|_2^2$$

- this yields an approximate solution to the inverse problem, i.e.,

$$\mathbf{a} = (\mathbf{X}^\mathsf{T}\mathbf{X})^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}$$

- The solution exists if $\mathbf{X}^\mathsf{T}\mathbf{X}$ is invertible, i.e.,
  - $\mathbf{X}$ must have full column rank
  - o.w., the least squares solution is no better than the original problem, which is the case for **Case II**.

Regularization

- Regularize to incorporate a priori assumptions about the **size** and **smoothness** of the solution.
  - for e.g. by using the ℓ_2 norm as the measure of size
- Regularization is done using one of the following schemes:

$$\min \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{a}\|_2^2 \quad \text{s.t.} \quad \|\mathbf{a}\|_1 \leq T$$

$$\min \|\mathbf{a}\|_1 \quad \text{s.t.} \quad \|\mathbf{y} - \mathbf{X}\mathbf{a}\|_2^2 \leq \epsilon$$

$$\min \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{a}\|_2^2 + \lambda\|\mathbf{a}\|_1 \quad \text{(Lagrangian form)}$$

- Note that the schemes are equivalent in theory but not in practice, since relations between T, ϵ, and λ are unknown.

Solution

Take III

- Regularize, i.e.,

$$\min \frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{a}\|_2^2 + \lambda\|\mathbf{x}\|_2^2$$

- called **ridge regression** with the **unique** solution,

$$\mathbf{a}^* = (\mathbf{X}^\mathsf{T}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^\mathsf{T}\mathbf{y}.$$

- Note that $\mathbf{X}^\mathsf{T}\mathbf{X} + \lambda\mathbf{I}$ is nonsingular even when $\mathbf{X}^\mathsf{T}\mathbf{X}$ is singular.

When $n \ll p$

High-dimensional Problem

- Standard procedure is to constrain with **sparsity**.
- To measure sparsity, we introduce the ℓ_0 quasi-norm,

$$\|\mathbf{a}\|_0 = \#\{i : a_i \neq 0\}.$$

- The problem becomes,

$$\min \|\mathbf{a}\|_0 \quad \text{s.t.} \quad \mathbf{y} = \mathbf{X}\mathbf{a}.$$

- Because of the **combinatorial** aspect of the ℓ_0 norm, the ℓ_0-regularization is intractable.

Solution

Convex Relaxation

- **Basis pursuit** (Chen et al., 1995)

$$\min \|\mathbf{a}\|_1 \quad \text{s.t.} \quad \mathbf{y} = \mathbf{X}\mathbf{a}.$$

- which is a linear program for which a tractable algorithm exists, in this case:
  - primal-dual interior point method
  - solves the **approximate** problem, **exactly**
- To allow for some noise, Chen et al. proposed **basis pursuit de-noising**, also called the **lasso** (Tibshirani, 1996)

$$\frac{1}{2}\|\mathbf{y} - \mathbf{X}\mathbf{a}\|_2^2 + \lambda\|\mathbf{a}\|_1.$$

Power family of penalties

ℓ_p norms raised to the pth power

$$\|\mathbf{a}\|_p^p = \left(\sum_i |a_i|^p \right)$$

- For $1 \leq p < \infty$, the above is convex.
- $0 < p \leq 1$, is the range of p useful for measuring sparsity.

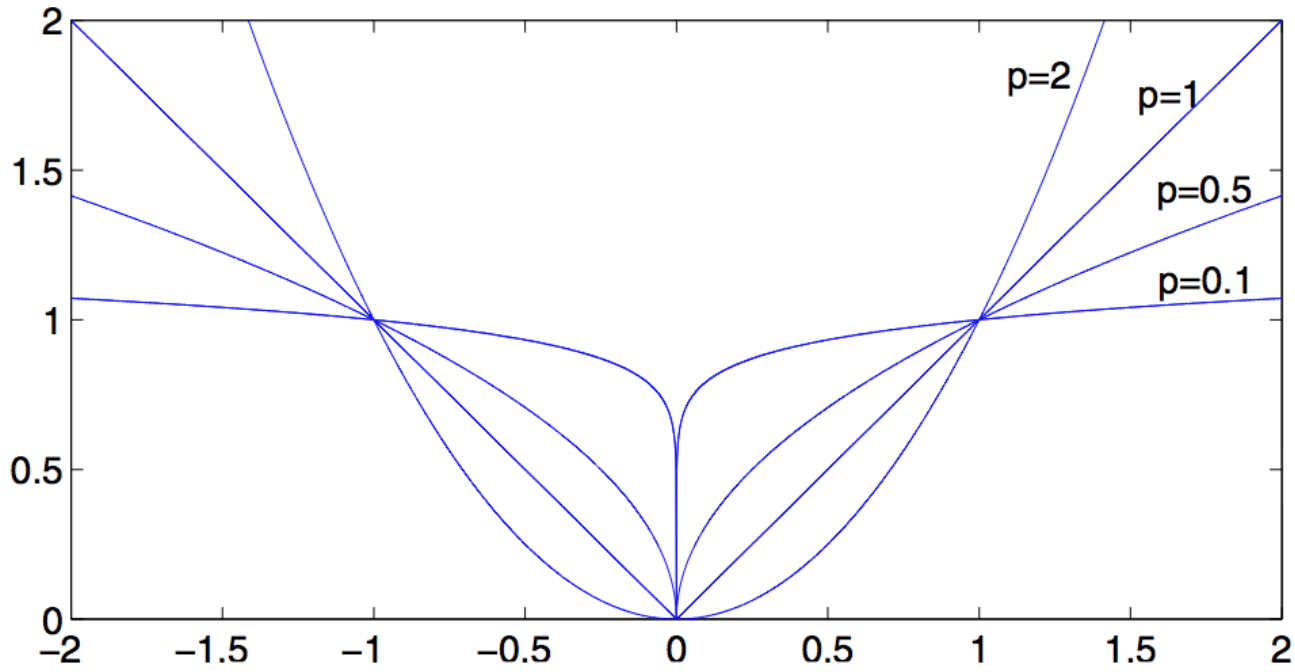
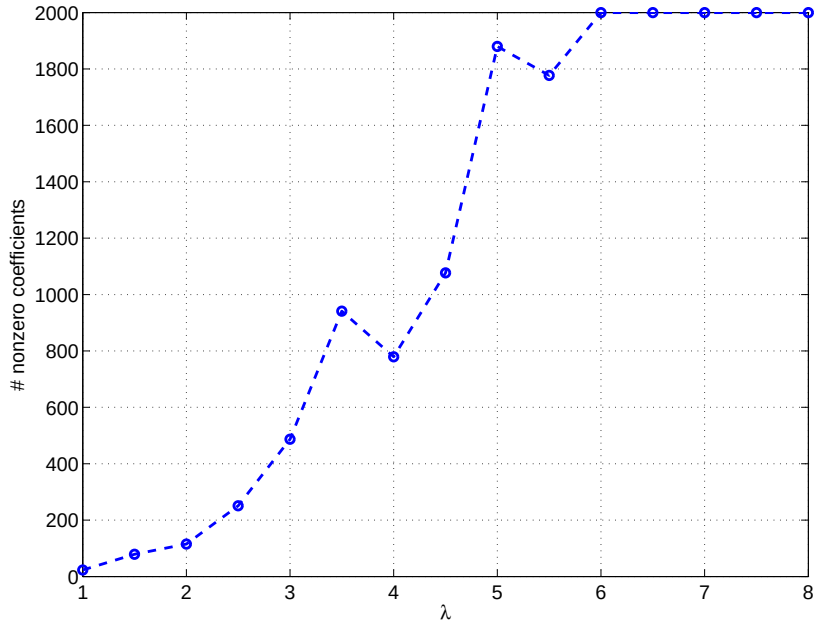
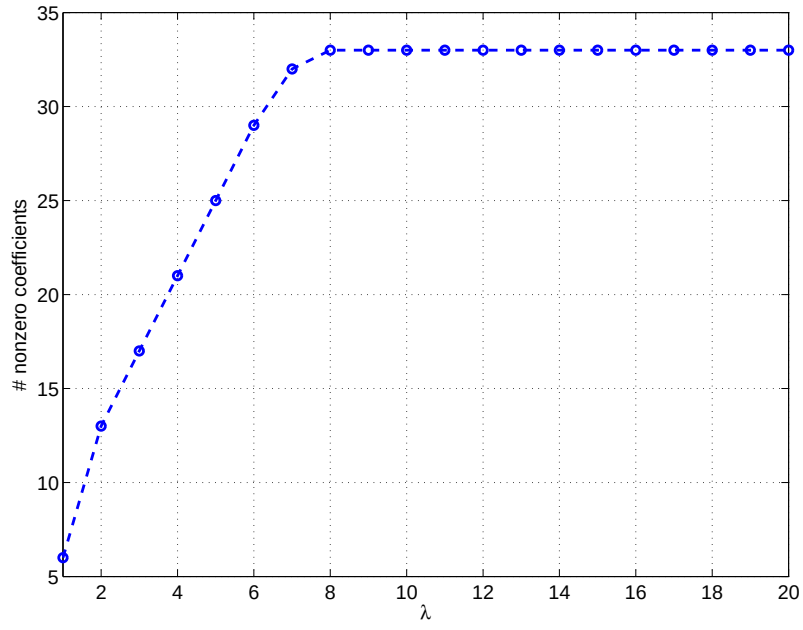


Figure: As p goes to 0, $|x|^p$ becomes the indicator function and $|\mathbf{x}|^p$ becomes a count of the nonzeros in $\mathbf{x}$ (Bruckstein et al., 2009).

- We train a classifier ($sign(f(\mathbf{x}) = \mathbf{w}^\mathsf{T}\mathbf{x} + b)$)
- using the lasso optimization
- we find the best λ using cross-validation on the training set
- we know that if we start with the smallest λ in cross-validation,
  - then we have the most compact classifier possible

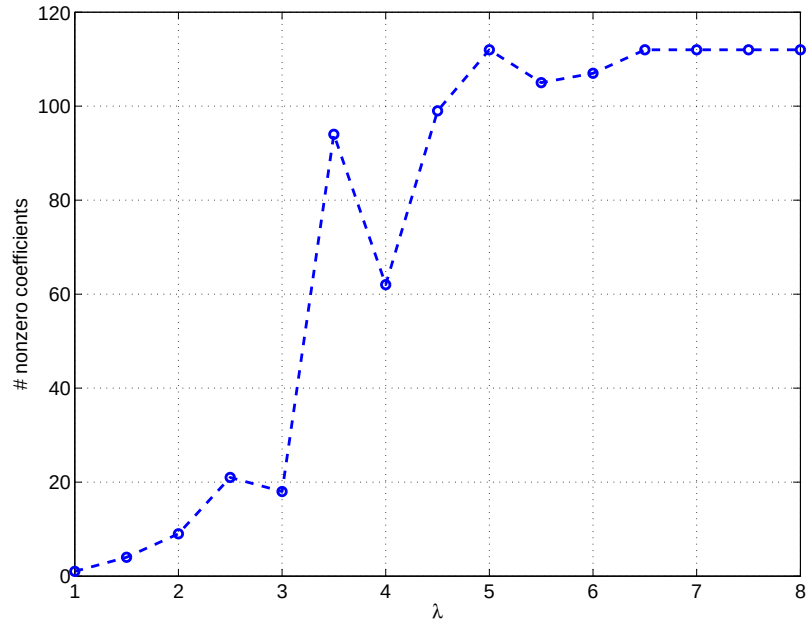


(a) Colon dataset



(b) Ionosphere dataset

Figure: In these figures, we see that the number of nonzero elements of the solution increases as we increase the regularization parameter λ, thus providing us with means to control the sparsity of the solution.



(a) Mushrooms dataset

Figure: In this figure, we see that the number of nonzero elements of the solution increases as we increase the regularization parameter λ, thus providing us with means to control the sparsity of the solution.

Table: Data set information: n denotes the number of samples, p denotes the dimension of the samples, and #nz are the nonzero elements of the $n \times p$ data matrix.

Data set	n	p
adult1	1605	123
adult4	4781	123
adult7	16,100	123
australian	690	14
colon	62	2,000
covertype	581,012	54
diabetes	768	8
heart	270	13
ionosphere	351	34
liverdisorders	8,124	112

Table: In this table we see a comparison of three other classifiers with the lasso on seven data sets. The hyperparameter is denoted by C or λ, depending on the algorithm. The number of nonzero elements in the solution vector is denoted by #nz. The percentage of correctly classified testing samples is denoted by Acc.

Dataset	lasso			SVM			ℓ_1-reg L2SVM			ℓ_1-reg logreg		
	λ	Acc	#nz	C	Acc	#nz	C	Acc	#nz	C	Acc	#nz
australian	10	86	14	2	86	68	20	86	14	5	**87**	14
colon	1	77	16	1	**87**	11	10	75	112	10	83	91
diabetes	10	**80**	8	1	75	105	10	77	8	10	76	8
heart	6	**87**	13	1.5	84	30	10	80	13	5	83	13
ionosphere	1	77	16	1	**87**	11	10	75	28	2	82	31
liverdisorders	2	46	6	2	62	70	5	66	6	5	**67**	6
mushrooms	2	48	13	2	**100**	90	10	**100**	96	20	**100**	95

Table: In this table we present the results for ridge regression. The solution is dense and hence the number of nonzero elemenst equals the number of features.

Dataset	ridge		
	λ	Acc	#nz
australian	30	86	14
colon	6	87	2000
diabetes	40	76	8
heart	9	86	13
ionosphere	10	74	34
liverdisorders	8	35	6
mushrooms	20	49	112

Representation by sparse approximation

- To motivate this idea let's look at
  - feature learning with **sparse coding**, and
  - **sparse representation classification** (SRC)
    - an example of **exemplar-based sparse approximation**

Unsupervised feature learning

Application to image classification

$$\mathbf{x} = \mathbf{D}\mathbf{a}$$

- An example is the recent work by Coates and Ng (2011).
  - where $\mathbf{x}$ is the input feature vector
  - could be a vectorized image patch, or a SIFT descriptor
  - $\mathbf{a}$ is the **higher-dimensional sparse representation** of $\mathbf{x}$
  - $\mathbf{D}$ is usually learned

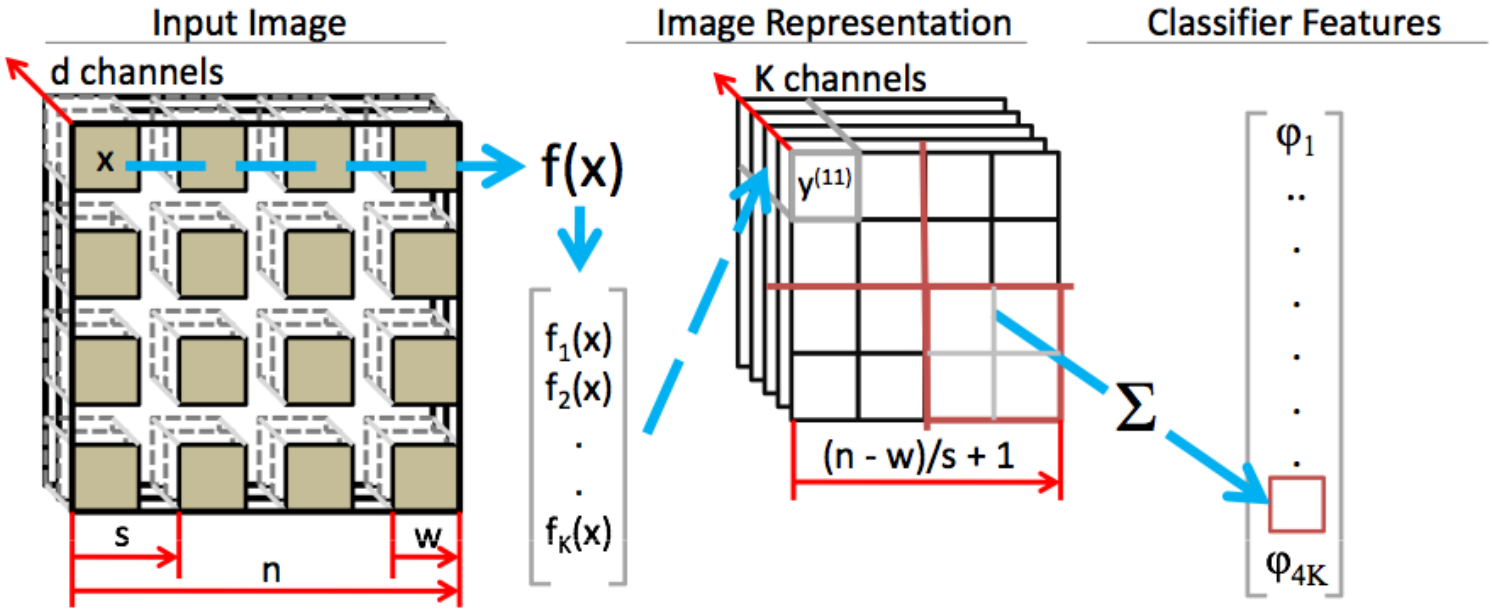


Figure: Image classification (Coates and Ng, 2011).

Multiclass classification

(Wright et al., 2009)

- $\mathcal{D} := \left\{(\mathbf{x}_i, y_i) : \mathbf{x}_i \in \mathbb{R}^m, y_i \in \{1, \ldots, c\}, i \in \{1, \ldots, N\}\right\}$
- Given a test sample $\mathbf{z}$
  1. Solve $\min_{\boldsymbol{\alpha} \in \mathbb{R}^N} \|\boldsymbol{\alpha}\|_1$ subject to $\|\mathbf{z} - \mathbf{D}\boldsymbol{\alpha}\|_2^2 \leq \sigma$
  2. Define $\{\boldsymbol{\alpha}_y : y \in \{1, \ldots, c\}\}$ where $[\boldsymbol{\alpha}_y]_i = \boldsymbol{\alpha}_i$ if $\mathbf{x}_i$ belongs to class y, o.w. 0
  3. Construct $\mathcal{X}(\boldsymbol{\alpha}) := \left\{\hat{\mathbf{x}}_y(\boldsymbol{\alpha}) = \mathbf{D}\boldsymbol{\alpha}_y, y \in \{1, \ldots, c\}\right\}$
  4. Predict $\hat{y} := \arg\min_{y \in \{1, \ldots, c\}} \|\mathbf{z} - \hat{\mathbf{x}}_y(\boldsymbol{\alpha})\|_2^2$

Global methods

- In parametric approaches, the regression function is known
- for e.g., in **multiple linear regression** (MLR) we assume

$$f(\mathbf{z}) = \sum_{j=1}^{M} \beta_j z_j + \epsilon$$

- we can also add higher order terms but still have a model that is linear in the parameters β_j, γ_j

$$f(\mathbf{z}) = \sum_{j=1}^{M} \left(\beta_j z_j + \gamma_j z_j^2\right) + \epsilon$$

Local methods

- A successful nonparametric approach to regression: local estimation (Hastie and Loader, 1993; Härdle and Linton, 1994; Ruppert and Wand, 1994)
- In local methods:

$$f(\mathbf{z}) = \sum_{i=1}^{N} l_i(\mathbf{z}) y_i + \epsilon$$

Local methods

Continued

- For e.g. in k-**nearest neighbor regression** (k-NNR)

$$f(\mathbf{z}) = \sum_{i=1}^{N} \frac{\alpha_i(\mathbf{z})}{\sum_{p=1}^{N} \alpha_p(\mathbf{z})} y_i$$

  - where $\alpha_i(\mathbf{z}) := \mathrm{I}_{\mathcal{N}_k(\mathbf{z})}(\mathbf{x}_i)$
  - $\mathcal{N}_k(\mathbf{z}) \subset \mathcal{D}$ is the set of the k-nearest neighbors of $\mathbf{z}$

Local methods

Continued

- In **weighted** k-**NNR** (Wk-NNR),

$$f(\mathbf{z}) = \sum_{i=1}^{N} \frac{\alpha_i(\mathbf{z})}{\sum_{p=1}^{N} \alpha_p(\mathbf{z})} y_i$$

  - $\alpha_i(\mathbf{z}) := S(\mathbf{z}, \mathbf{x}_i)^{-1} \, \mathrm{I}_{\mathcal{N}_k(\mathbf{z})}(\mathbf{x}_i)$
  - $S(\mathbf{z}, \mathbf{x}_i) = (\mathbf{z} - \mathbf{x}_i)^{\mathsf{T}} \mathbf{V}^{-1} (\mathbf{z} - \mathbf{x}_i)$
    is the scaled Euclidean distance

Local methods

Continued

- In local methods:
  estimate the regression function *locally*
  by a *simple parametric model*
- In **local polynomial regression**:
  estimate the regression function locally,
  by a **Taylor polynomial**
- This is what happens in SPARROW, as we will explain

Sparrow

I meant this sparrow

SPARROW is a local method

- Before we get into the details,
- see a few examples showing benefits of local methods
- then we'll talk about SPARROW

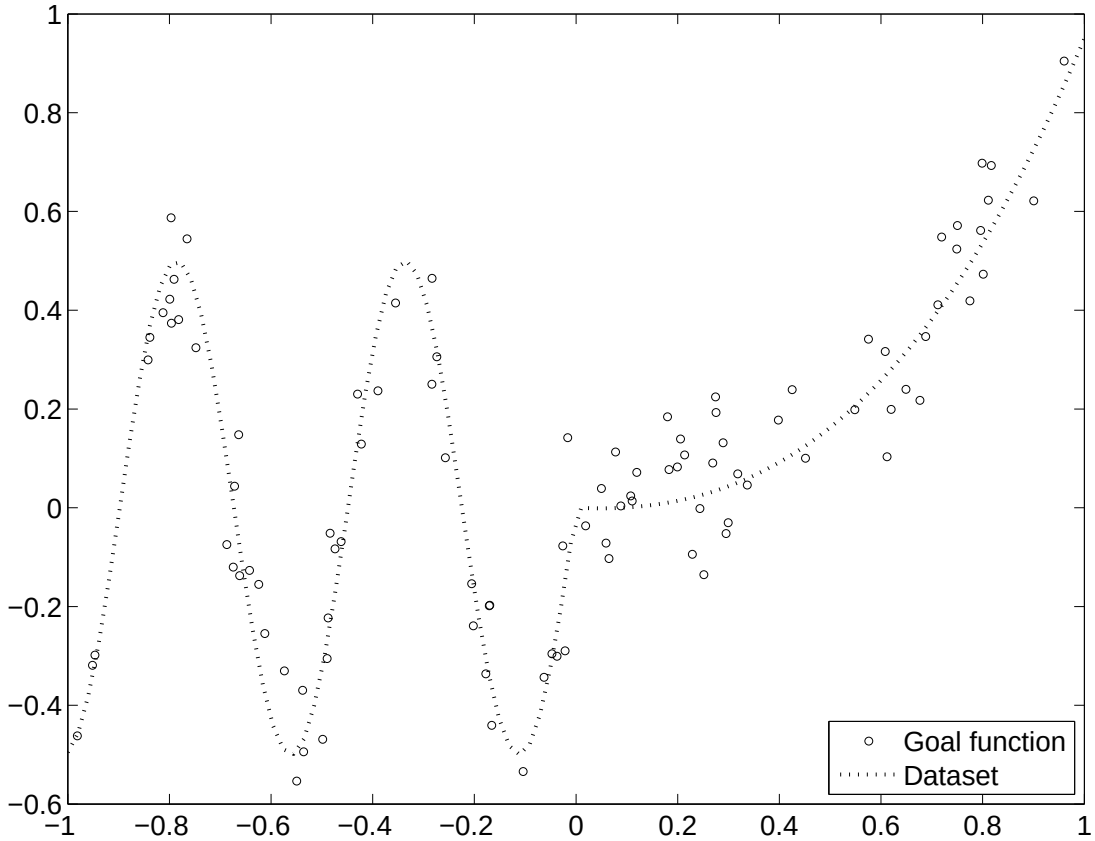


Figure: Our generated dataset. $y_i = f(x_i) + \epsilon_i$, where $f(x) = (x^3 + x^2)\,\mathrm{I}(x) + \sin(x)\,\mathrm{I}(-x)$.

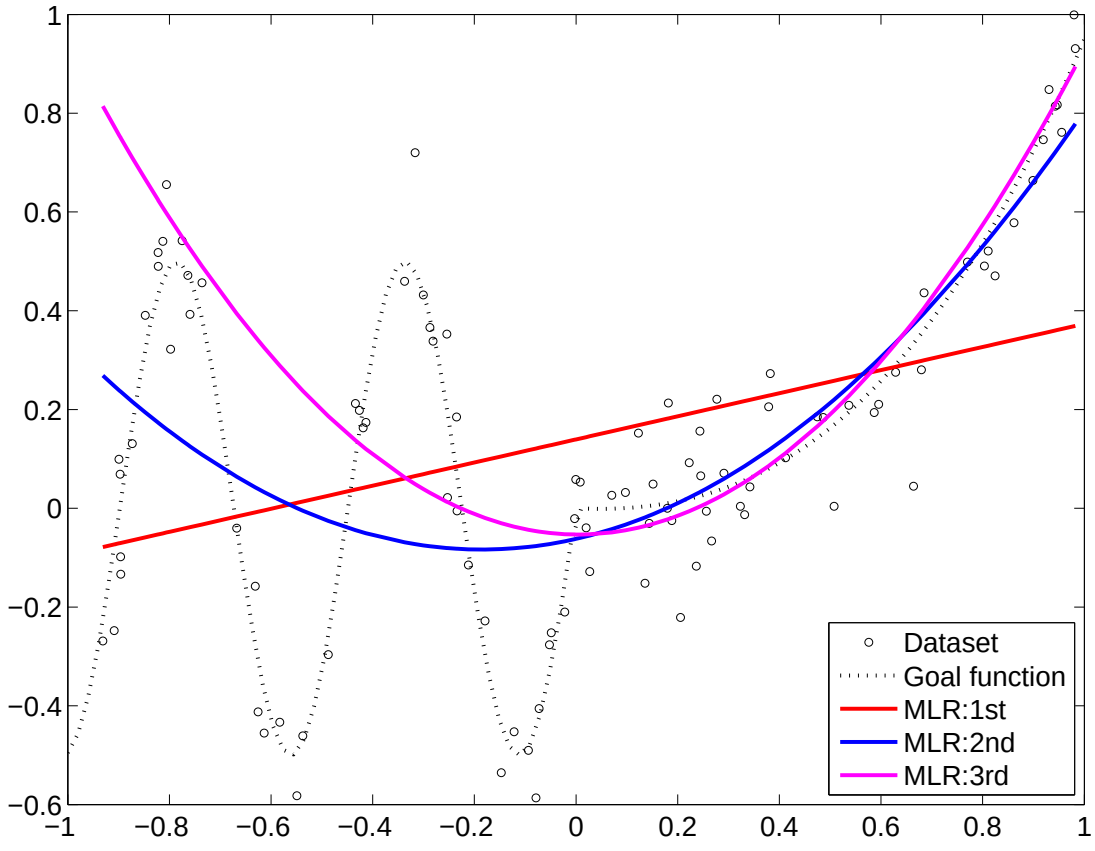


Figure: Multiple linear regression with first-, second-, and third-order terms.

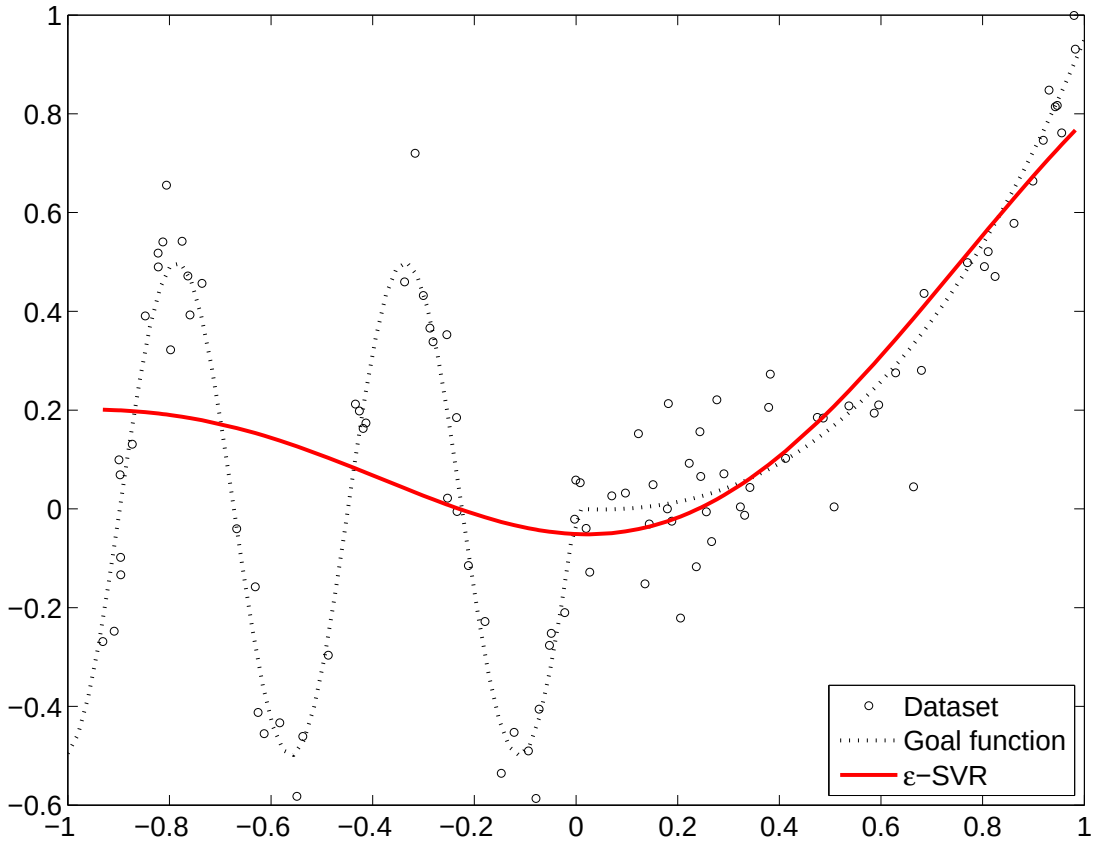


Figure: ϵ-support vector regression with an RBF kernel.

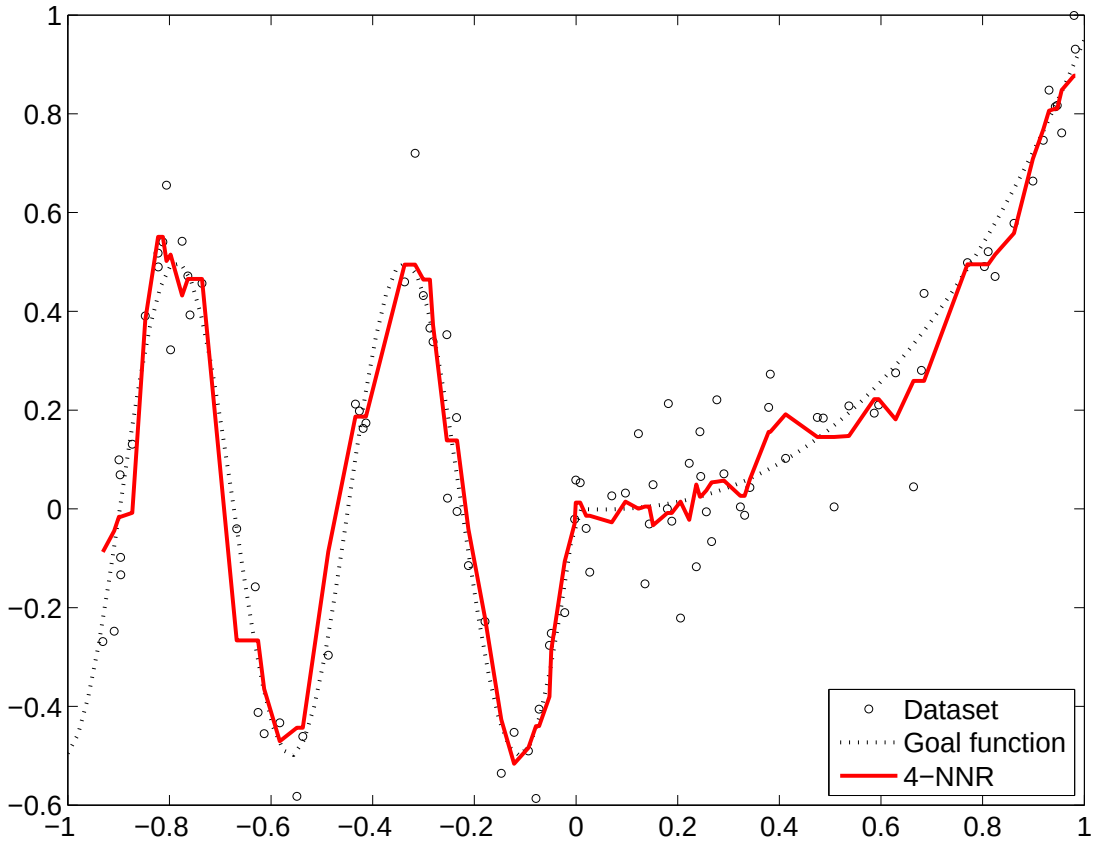


Figure: 4-nearest neighbor regression.

Effective weights in SPARROW

- In local methods:

$$f(\mathbf{z}) = \sum_{i=1}^{N} l_i(\mathbf{z}) y_i$$

- Now we define $l_i(\mathbf{z})$

Local estimation by a Taylor polynomial

- To obtain the local quadratic estimate of the regression function at $\mathbf{z}$
- we can approximate $f(\mathbf{x})$ about $\mathbf{z}$ by a second-degree Taylor polynomial

$$f(\mathbf{x}) \approx f(\mathbf{z}) + (\mathbf{x}-\mathbf{z})^T \boldsymbol{\theta}_\mathbf{z} + \frac{1}{2}(\mathbf{x}-\mathbf{z})^T \mathbf{H}_\mathbf{z} (\mathbf{x}-\mathbf{z})$$

- $\boldsymbol{\theta}_\mathbf{z} := \nabla f(\mathbf{z})$ the gradient of $f(\mathbf{x})$,
  $\mathbf{H}_\mathbf{z} := \nabla^2 f(\mathbf{z})$ is its Hessian
  both evaluated at $\mathbf{z}$

Local estimation by a Taylor polynomial

Continued

We need to solve the locally weighted least squares problem

$$\min_{f(\mathbf{z}),\boldsymbol{\theta}_\mathbf{z},\mathbf{H}_\mathbf{z}} \sum_{i\in\Omega} \alpha_i(\mathbf{z}) \left[y_i - f(\mathbf{z}) - (\mathbf{x}_i - \mathbf{z})^T \boldsymbol{\theta}_\mathbf{z} - \frac{1}{2}(\mathbf{x}_i - \mathbf{z})^T \mathbf{H}_\mathbf{z} (\mathbf{x}_i - \mathbf{z}) \right]^2$$

Local estimation by a Taylor polynomial

Continued

- This can be expressed as

$$\min_{\mathbf{\Theta_z}} \left\| \mathbf{A}_{\mathbf{z}}^{1/2} \left[\mathbf{y} - \mathbf{X_z} \mathbf{\Theta_z} \right] \right\|_2^2$$

- $a_{ii} = \alpha_i$, $\mathbf{y} := [y_1, y_2, \ldots, y_N]^\mathsf{T}$
- $\mathbf{X_z} := \begin{bmatrix} 1 & (\mathbf{x}_1 - \mathbf{z})^T & \operatorname{vech}^T[(\mathbf{x}_1 - \mathbf{z})(\mathbf{x}_1 - \mathbf{z})^T] \\ \vdots & \vdots & \vdots \\ 1 & (\mathbf{x}_N - \mathbf{z})^T & \operatorname{vech}^T[(\mathbf{x}_N - \mathbf{z})(\mathbf{x}_N - \mathbf{z})^T] \end{bmatrix}$
- parameter supervector: $\mathbf{\Theta_z} := \left[f(\mathbf{z}), \boldsymbol{\theta}_\mathbf{z}, \operatorname{vech}(\mathbf{H_z}) \right]^\mathsf{T}$

Local estimation by a Taylor polynomial

Continued

- The parameters defined by the least squares solution:

$$\widehat{\mathbf{\Theta}}_{\mathbf{z}} = \left(\mathbf{X}_{\mathbf{z}}^T \mathbf{A}_{\mathbf{z}} \mathbf{X}_{\mathbf{z}}\right)^{-1} \mathbf{X}_{\mathbf{z}}^T \mathbf{A}_{\mathbf{z}} \mathbf{y}$$

- And so the local quadratic estimate is

$$\hat{f}(\mathbf{z}) = \mathbf{e}_1^T \left(\mathbf{X}_{\mathbf{z}}^T \mathbf{A}_{\mathbf{z}} \mathbf{X}_{\mathbf{z}}\right)^{-1} \mathbf{X}_{\mathbf{z}}^T \mathbf{A}_{\mathbf{z}} \mathbf{y}$$

- Since $f(\mathbf{z}) = \sum_{i=1}^{N} l_i(\mathbf{z}) y_i$,
  the ith effective weight for SPARROW is

$$l_i(\mathbf{z}, \mathcal{D}) = \mathbf{e}_i^T \mathbf{A}_{\mathbf{z}}^T \mathbf{X}_{\mathbf{z}} \left(\mathbf{X}_{\mathbf{z}}^T \mathbf{A}_{\mathbf{z}} \mathbf{X}_{\mathbf{z}}\right)^{-1} \mathbf{e}_1$$

Local estimation by a Taylor polynomial

Continued

- The local constant regression estimate is

$$\hat{f}(\mathbf{z}) = (\mathbf{1}^T \mathbf{A}_\mathbf{z} \mathbf{1})^{-1} \mathbf{1}^T \mathbf{A}_\mathbf{z} \mathbf{y} = \frac{\sum_{i \in \Omega} \alpha_i(\mathbf{z}) y_i}{\sum_{k \in \Omega} \alpha_k(\mathbf{z})}.$$

- Look familiar?

Observation weights in SPARROW

- To find α_i we solve the following problem (Chen et al., 1995)

$$\min_{\mathbf{s}\in\mathbb{R}^N} \|\mathbf{s}\|_1 \quad \text{subject to} \quad \frac{\|\mathbf{z}-\mathbf{D}\mathbf{s}\|_2^2}{\|\mathbf{z}\|_2^2} \leq \epsilon^2$$

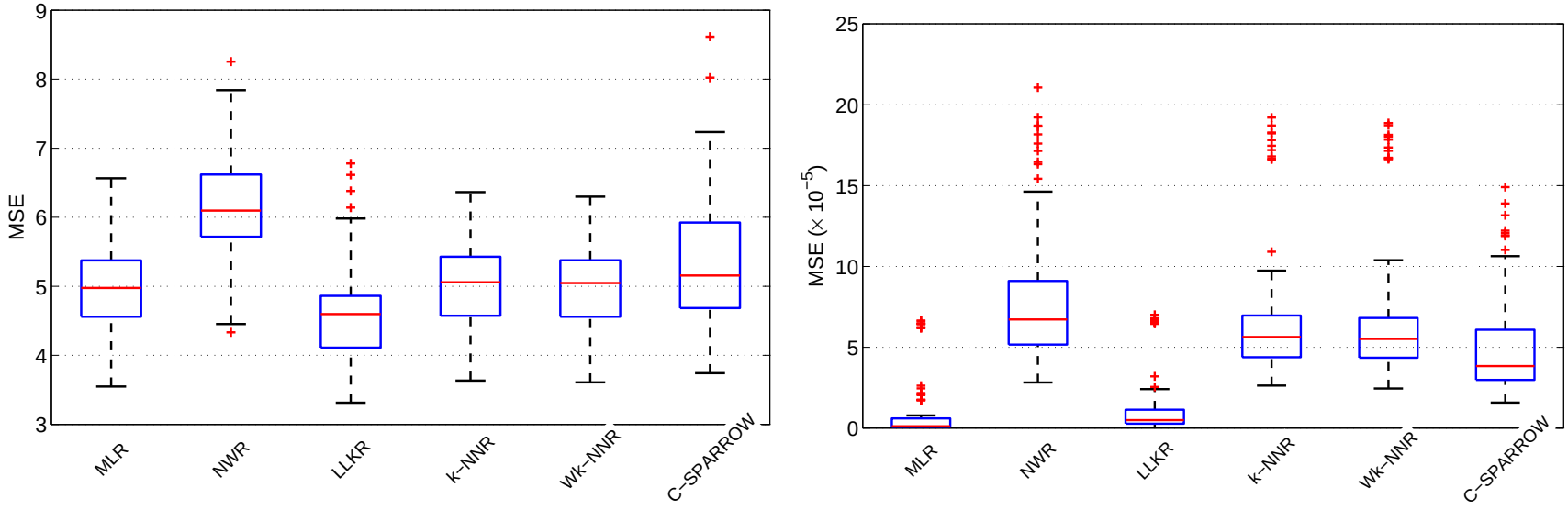
  - $\sigma^2 > 0$ limits signal to approximation error ratio
  - and $\mathbf{D} := \left[\frac{\mathbf{x}_1}{\|\mathbf{x}_1\|_2}, \frac{\mathbf{x}_2}{\|\mathbf{x}_2\|_2}, \ldots, \frac{\mathbf{x}_N}{\|\mathbf{x}_N\|_2}\right]$

- Finally, the ith observation weight in SPARROW is

$$\alpha_i(\mathbf{z}) := \left[\frac{S(\mathbf{z},\mathbf{x}_i)}{\min_{j\in\Omega} S(\mathbf{z},\mathbf{x}_j)}\right]^{-1} \frac{s_i}{\|\mathbf{z}\|_2}$$

Table: Summary of the four datasets we test. The last column indicates the tuned parameter k used in the experiments involving k-NNR and Wk-NNR.

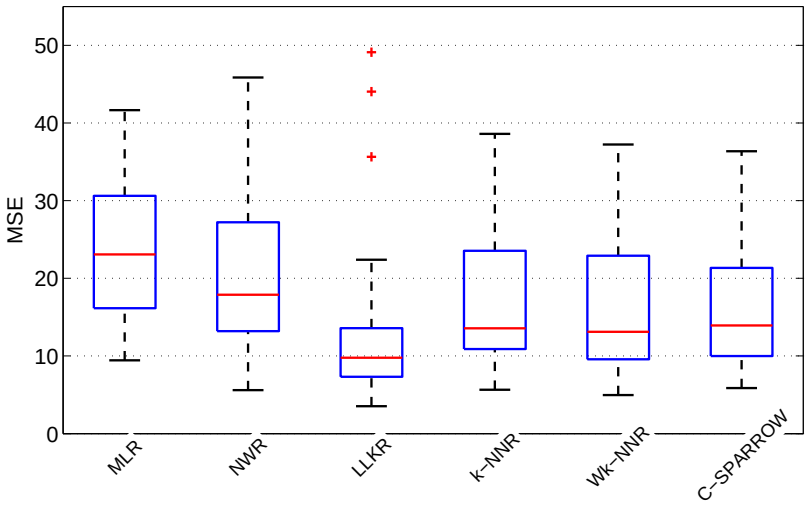
Dataset	# observations (N)	# attributes (M)	k
abalone	4,177	8	9
bodyfat	252	14	4
housing	506	13	2
mpg	392	7	4



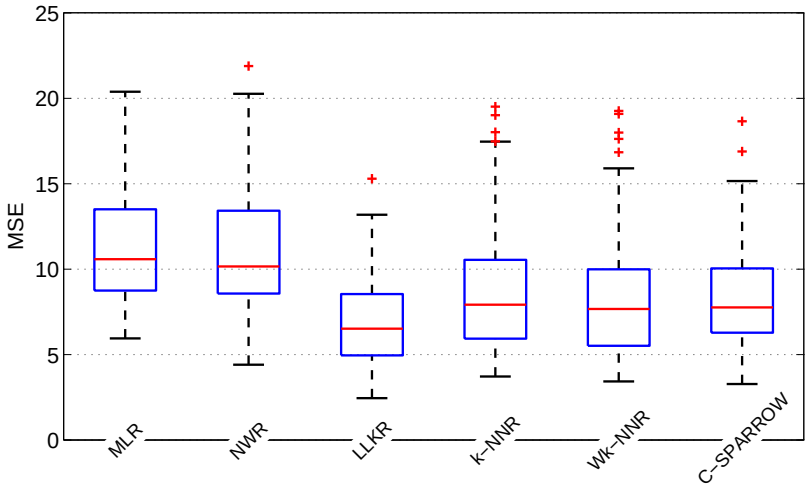
(a) Abalone dataset

(b) Bodyfat dataset

Figure: Boxplots for 10-fold cross-validation estimate of mean squared error (100 independent runs) for four different datasets. Each box delimits 25 to 75 percentiles, and the red line marks median. Extrema are marked by whiskers, and outliers by pluses.



(a) Housing dataset



(b) MPG dataset

Figure: Boxplots for 10-fold cross-validation estimate of mean squared error (100 independent runs) for four different datasets. Each box delimits 25 to 75 percentiles, and the red line marks median. Extrema are marked by whiskers, and outliers by pluses.

Linear SPARROW

- L-SPARROW should perform better than C-SPARROW because it is a higher-order model
- But a problem with higher-order models is that solutions could become unstable
- We resolve the problem by solving

$$\min_{\mathbf{\Theta_z},\lambda} \left\| \mathbf{A_z}^{1/2} \left[\mathbf{y} - \mathbf{X_z \Theta_z} \right] \right\|_2^2 + \lambda \|\mathbf{\Theta_z}\|_2^2$$

- The solution becomes

$$\widehat{\mathbf{\Theta}}(\mathbf{z}) = \left(\mathbf{X_z}^T \mathbf{A_z} \mathbf{X_z} + \lambda \mathbf{I} \right)^{-1} \mathbf{X_z}^T \mathbf{A_z} \mathbf{y}.$$

Table: A comparison of the MSE estimates obtained by 10 trials of 10-fold cross-validation of C-SPARROW and L-SPARROW without and with ridge regression on the four datasets. The last column denotes the ridge parameter used to obtain the L-SPARROW estimate.

Dataset	C-SPAR.	L-SPAR.	L-SPAR. w/ RR	λ
abalone	5	16	988	10^{-3}
bodyfat	5×10^{-5}	35×10^{-5}	960×10^{-5}	10^{-6}
housing	10	45	4304	10^{-4}
mpg	7	8	6335	10^{-3}

Table: In this table we compare kNN and SRC on five multiclass classification data sets.

Dataset	n	p	#classes	k	kNN	SRC
dna	2000	180	3	125	86	86
glass	214	9	6	2	70	65
iris	150	4	3	6	95	72
vowel	528	10	11	2	94	84
wine	178	13	3	7	97	99

Conclusions

- ℓ_1-regularized square loss minimization **for classification** is a **success** both computationally and statistically
- ℓ_1-regularized square loss minimization **for reconstruction** is **not worth it**
  - simpler methods like kNN classification and WkNNR are at least as good

Recommendations for Future Work

- Replace dictionary learning and sparse coding with k-means and kNN for feature learning in image classification tasks
- Replace $\mathbf{x}_i$'s with $\phi(\mathbf{x}_i)$ to get nonlinear classification
- Perform analysis on computational complexity of ℓ_1-regularized square loss minimization methods like (Yuan et al., 2010)
- Try the elastic net (Prof. Rahmati's suggestion)

$$\min \|\mathbf{y} - \mathbf{X}\mathbf{a}\|_2^2 + \lambda_2 \|\mathbf{a}\|_2^2 + \lambda_1 \|va\|_1$$

- Prof. Ebadzadeh's initial proposal on regularizing $\boldsymbol{\alpha}$, the SVM dual variable, has been done before by Osuna and Girosi (1999) in a paper entitled: "Reducing run-time complexity of support vector machines"

References I

Stéphane Boucheron, Olivier Bousquet, and Gabor Lugosi. Theory of classification : A survey of some recent advances. *ESAIM: Probability and Statistics*, 9:323–375, 2005.

Alfred M. Bruckstein, David L. Donoho, and Michael Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM Review*, 51(1):34–81, 2009.

Scott S. Chen, David L. Donoho, and Michael A. Saunders. Atomic decomposition by basis pursuit. Technical Report 479, Department of Statistics, Stanford University, May 1995.

Adam Coates and Andrew Ng. The importance of encoding versus training with sparse coding and vector quantization. In *International Conference on Machine Learning (ICML)*, pages 921–928, 2011.

L. Devroye, L. Györfi, and G. Lugosi. *A Probabilistic Theory of Pattern Recognition*. Springer, 1996.

References II

W. Härdle and O. Linton. Applied nonparametric methods. Technical Report 1069, Yale University, 1994.

T. J. Hastie and C. Loader. Local regression: Automatic kernel carpentry. *Statistical Science*, 8(2):120–129, 1993.

Edgar E. Osuna and Federico Girosi. Reducing the run-time complexity in support vector machines. In Bernhard Schölkopf, Christopher J. C. Burges, and Alexander J. Smola, editors, *Advances in kernel methods*, pages 271–283. MIT Press, Cambridge, MA, USA, 1999.

Ryan Rifkin. *Everything old is new again: a fresh look at historical approaches in machine learning*. PhD thesis, Sloan School of Management, Massachusetts Institute of Technology, 2002.

D. Ruppert and M. P. Wand. Multivariate locally weighted least squares regression. *The Annals of Statistics*, 22:1346–1370, 1994.

Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society (Series B)*, 58:267–288, 1996.

References III

John Wright, Allen Y. Yang, Arvind Ganesh, S. Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31:210–227, 2009.

Guo-Xun Yuan, Kai-Wei Chang, Cho-Jui Hsieh, and Chih-Jen Lin. A comparison of optimization methods and software for large-scale ℓ_1-regularized linear classification. *Journal of Machine Learning Research*, 11:3183–3234, 2010.

Tong Zhang. Statistical behavior and consistency of classification methods based on convex risk minimization. *The Annals of Statistics*, 32:56–134, March 2004.

Acknowledgements

- Committee:
  - Prof. Mohammad Rahmati
  - Prof. Narollah Moghaddam Charkari
  - Prof. Mohammad Mehdi Ebadzadeh
- Prof. Saeed Shiry
- Special thanks to **Sheida Bijani**, **Isaac Nickaein**, and **Mina Shirvani**
- Prof. Bob Sturm
- Last, and certainly not least, my parents and my brother

In memory of Uncle Asadollah Noorzad.