ORF 523	LECTURE 8	Princeton University
Instructor: A.A. Ahmadi		Scribe: G. Hall

Any typos should be emailed to a_a_a@princeton.edu.

1 Outline

- Convexity-preserving operations
- Convex envelopes, cardinality constrained optimization and LASSO
- An application in supervised learning: support vector machines (SVMs)

2 Operations that preserve convexity

The role of convexity preserving operations is to produce new convex functions out of a set of "atom" functions that are already known to be convex. This is very important for broadening the scope of problems that we can recognize as efficiently solvable via convex optimization. There is a long list of convexity-preserving rules; see section 3.2 of [2]. We present only a few of them here. The software CVX has a lot of these rules built in [1],[4].

2.1 Nonnegative weighted sums

Rule 1. *If $f_1, \ldots, f_m : \mathbb{R}^n \to \mathbb{R}$ are convex functions and $\omega_1, \ldots, \omega_m$ are nonnegative scalars then*

$$f(x) = \omega_1 f_1(x) + \ldots + \omega_m f_m(x)$$

is convex also. Similarly, a nonnegative weighted sum of concave functions is concave.

Exercise: If f_1, f_2 are convex functions,

- is $f_1 - f_2$ convex?
- is $f_1 \cdot f_2$ convex?
- is $\frac{f_1}{f_2}$ convex?

2.2 Composition with an affine mapping

Rule 2. *Suppose $f : \mathbb{R}^n \to \mathbb{R}$, $A \in \mathbb{R}^{n \times m}$, and $b \in \mathbb{R}^n$. Define $g : \mathbb{R}^m \to \mathbb{R}$ as*

$$g(x) = f(Ax + b)$$

with $dom(g) = \{x | Ax + b \in dom(f)\}$. Then, if f is convex, so is g; if f is concave, so is g.

The proof is a simple exercise.
Example: The following function is immediately seen to be convex. (Without knowing the previous rule, it would be much harder to prove convexity.)

$$f(x_1, x_2) = (x_1 - 2x_2)^4 + 2e^{3x_1 + 2x_2 - 5}$$

2.3 Pointwise maximum

Rule 3. *If $f_1, \ldots, f_m$ are convex functions, then their pointwise maximum*

$$f(x) = \max\{f_1(x), \ldots, f_m(x)\},$$

with $dom(f) = dom(f_1) \cap \ldots \cap dom(f_m)$ is also convex.

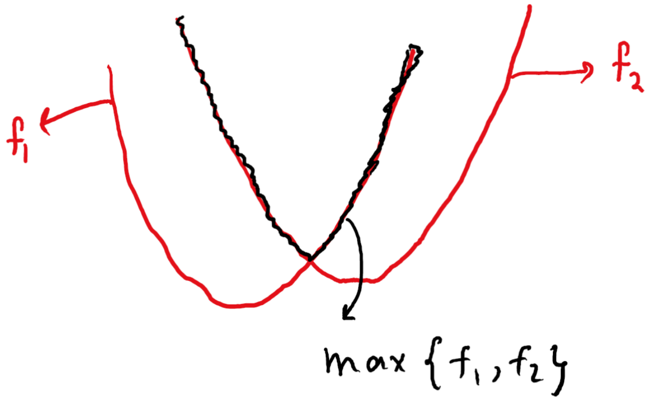


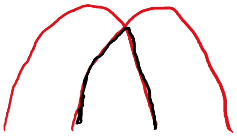
Figure 1: An illustration of the pointwise maximum rule

Proof: Pick any $x, y \in dom(f), \lambda \in [0, 1]$. Then,

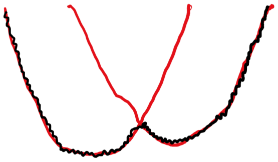
$$\begin{aligned} f(\lambda x + (1-\lambda)y) &= f_j(\lambda x + (1-\lambda)y) \text{ (for some } j \in \{1, \ldots, m\}) \\ &\leq \lambda f_j(x) + (1-\lambda) f_j(y) \\ &\leq \lambda \max\{f_1(x), \ldots, f_m(x)\} + (1-\lambda) \max\{f_1(y), \ldots, f_m(y)\} \\ &= \lambda f(x) + (1-\lambda) f(y). \ \square \end{aligned}$$

- It is also easy to prove this result using epigraphs. Recall that f convex $\Leftrightarrow$ $epi(f)$ is convex. But $epi(f) = \cap_{i=1}^{m} epi(f_i)$, and we know that the intersection of convex sets is convex.

- One can similarly show that the pointwise minimum of two concave functions is concave.



- But the pointwise minimum of two convex functions may not be convex.



2.4 Restriction to a line

Rule 4. *Let $f : \mathbb{R}^n \to \mathbb{R}$ be a convex function and fix some $x, y \in \mathbb{R}^n$. Then the function $g : \mathbb{R} \to \mathbb{R}$ given by $g(\alpha) = f(x + \alpha y)$ is convex.*

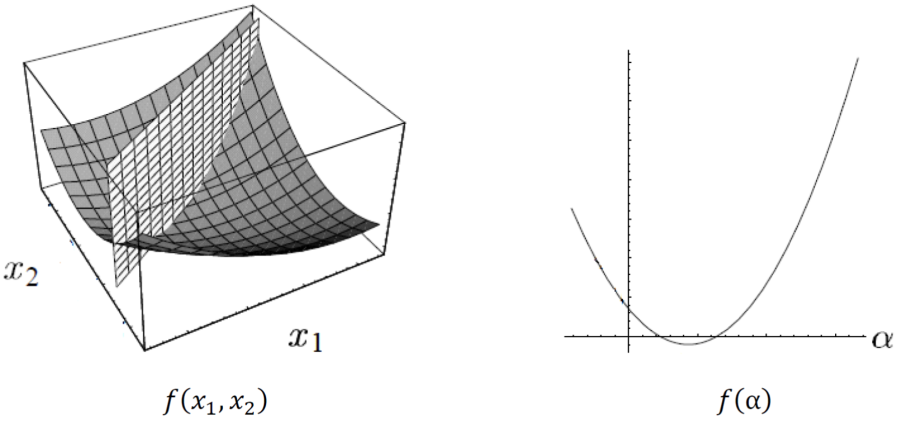


Figure 2: An illustration of the restriction to a line rule (image credit: [6])

Many algorithms for unconstrained convex optimization (e.g., steepest descent with exact line search) work by iteratively minimizing a function over lines. It's useful to remember that the restriction of a convex function to a line remains convex. This tells us that in each subproblem we are faced with a univariate convex minimization problem, and hence we can simply find a global minimum e.g. by finding a zero of the first derivative.

2.5 Power of a nonnegative function

Rule 5. *If f is convex and nonnegative (i.e., $f(x) \geq 0, \ \forall x$) and $k \geq 1$, then f^k is convex.*

<u>Proof:</u> We prove this in the case where f is twice differentiable. Let $g = f^k$. Then

$$\begin{aligned} &\nabla g(x) = k f^{k-1} \nabla f(x) \\ &\nabla^2 g(x) = k \left((k-1) f^{k-2} \nabla f(x) \nabla f^T(x) + f^{k-1} \nabla^2 f(x) \right). \end{aligned}$$

We see that $\nabla^2 g(x) \succeq 0$ for all x (why?). □

Does this result hold if you remove the nonnegativity assumption on f?

3 Convex envelopes

Definition 1. *The* convex envelope *(or* convex hull*) $conv_D \ f$ of a function $f : \mathbb{R}^n \to \mathbb{R}$ over a convex set $D \subseteq \mathbb{R}^n$ is "the largest convex underestimator of f on D"; i.e.,*

$$\textit{if } h(x) \leq f(x) \ \forall x \in D \textit{ and } h \textit{ is convex } \Rightarrow h(x) \leq conv_D \ f(x), \ \forall x \in D.$$

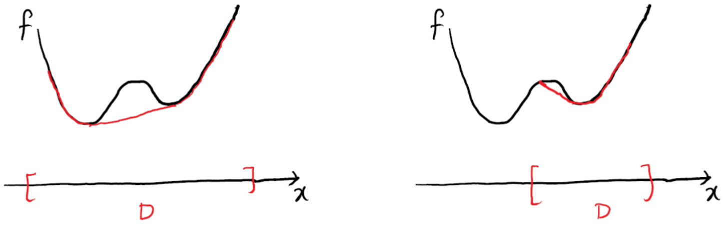


Figure 3: The convex envelope of a function over two different sets

- Equivalently, $conv_D \ f(x)$ is the pointwise maximum of all convex function that lie below f (on D).

- As the pictures suggest, the epigraph of *conv* f is the convex hull of the epigraph of f.
- Computing convex hulls of functions is in general a difficult task; e.g., computing the convex envelope of a mulitilinear function over the unit hypercube is NP-hard [3]. Indeed if we could compute $conv_D\ f$, then we could minimize f over D as the following statement illustrates.

Theorem 1 ([5])**.** *Consider the problem* $\min_{x \in S} f(x)$*, where* S *is a convex set. Then,*

$$f^* := \min_{x \in S} f(x) = \min_{x \in S} conv_S\ f(x) \tag{1}$$

and

$$\{y \in S|\ f(y) = f^*\} \subseteq \{y \in S|\ conv_S\ f(y) = f^*\}. \tag{2}$$

Proof: First we prove (1). As $conv_S\ f$ is an underestimator of f, we clearly have

$$\min_{x \in S} conv_S\ f \leq \min_{x \in S} f(x).$$

To see the converse, note that the constant function $g(x) = f^*$ is a convex underestimator of f. Hence, we must have $conv_S\ f(x) \geq f^*$, $\forall x \in S$.

To prove (2), let $y \in S$ be such that $f(y) = f^*$. Suppose for the sake of contradiction that $conv_S\ f(y) < f^*$. But this means that the function

$$\max\{f^*, conv_S\ f\}$$

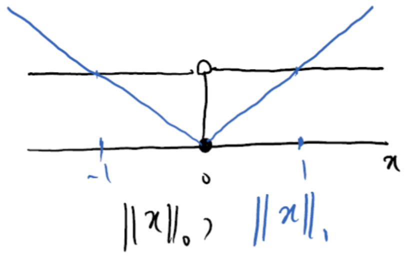
is convex (why?), an underestimator of f on D (why?), but larger than $conv_S\ f$ at y. This contradicts $conv_S\ f$ being the convex envelope. □

Example: In simple cases, the convex envelope of some functions over certain sets can be computed. A well-known example is the envelope of the function $l_0(x) := ||x||_0$, which is the function $l_1(x) = ||x||_1$. The l_0 "pseudonorm", also known as the cardinality function, is defined as

$$||x||_0 = \text{ \# of nonzero elements of } x.$$

This function is not a norm (why?) and is not convex (why?).

Theorem 2. *The convex envelope of the* l_0 *pseudonorm over the set* $\{x|\ ||x||_\infty \leq 1\}$ *is the* l_1 *function.*



This simple observation is the motivation (or one motivation) behind many heuristics for l_0 optimization like compressed sensing, LASSO, etc.

3.1 LASSO [7]

LASSO stands for *least absolute shrinkage and selection operator*. It is simply a least squares problem with an l_1 penalty

$$\min_x ||Ax - b||^2 + \lambda ||x||_1,$$

where $\lambda > 0$ is a fixed parameter. This is a convex optimization problem (why?).

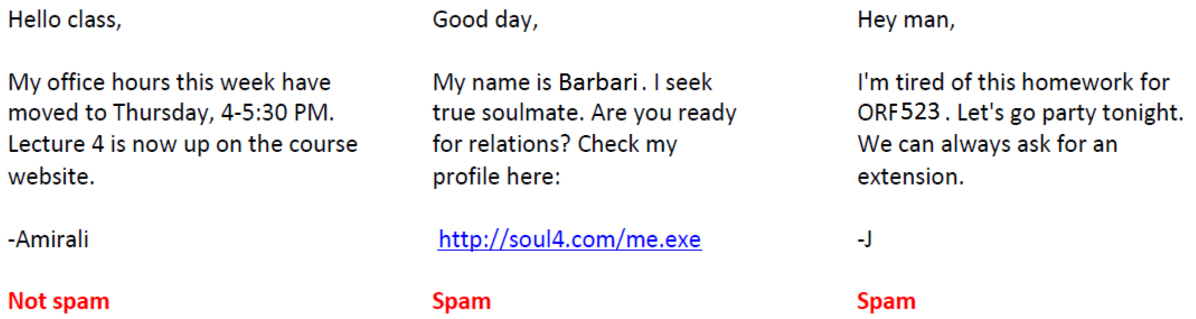
- By increasing λ, we increase our preference for having sparse solutions.
- By decreasing λ, we increase our preference for decreasing the regression error.

Here's the idea behind why this could be useful. Consider a very simple scenario where you are given m data points in $\mathbb{R}^n$ and want to fit a function f to the data that minimizes the sum of the squares of the deviations. The problem is, however, that you don't have a good idea of what function class exactly f belongs to. So you decide to throw in a lot of functions in your basis: maybe you include a term for every monomial up to a certain degree, you add trigonometric functions, exponential functions, etc. After this, you try to write f as a linear combination of this massive set of basis functions by solving an optimization problem that finds the coefficients of the linear combination. Well, if you use all the basis functions (nonzero coefficients everywhere), then you will have very small least squares error but you would be overfitting the data like crazy. What LASSO tries to do, as you increase λ, is to set many of these coefficients equal to zero and tell you (somehow magically) that which of the basis functions were actually important for fitting the data and which weren't.

4 Support vector machines

- Support vector machines (SVM) constitute a prime example of supervised learning. In such a setting, we would like to learn a classifier from a labeled data set (called the training set). The classifier is then used to label future data points.

- A classic example is an email spam filter:
  - Given a large number of emails with correct labels "spam" or "not spam", we would like an algorithm for classifying future emails as spam or not spam.
  - The emails for which we already have the labels constitute the "training set".



Hello class,

My office hours this week have moved to Thursday, 4-5:30 PM. Lecture 4 is now up on the course website.

-Amirali

Not spam

Good day,

My name is Barbari. I seek true soulmate. Are you ready for relations? Check my profile here:

http://soul4.com/me.exe

Spam

Hey man,

I'm tired of this homework for ORF523. Let's go party tonight. We can always ask for an extension.

-J

Spam

Figure 4: An example of a good spam filter

- A basic approach is to associate a pair (x_i, y_i) to each email: y_i is the label, which is either 1 (spam) or -1 (not spam). The vector $x_i \in \mathbb{R}^n$ is called a *feature vector*; it collects some relevant information about email i. For example:
  - How many words are in the email?
  - How many misspelled words?
  - How many links?
  - Is there a \$ sign?
  - Does the word "bank account" appear?
  - Is the sender's email client trustworthy?
  - ...

- If we have m emails, we end up with m vectors in $\mathbb{R}^n$, each with a label ± 1. Here is a toy example in $\mathbb{R}^2$:

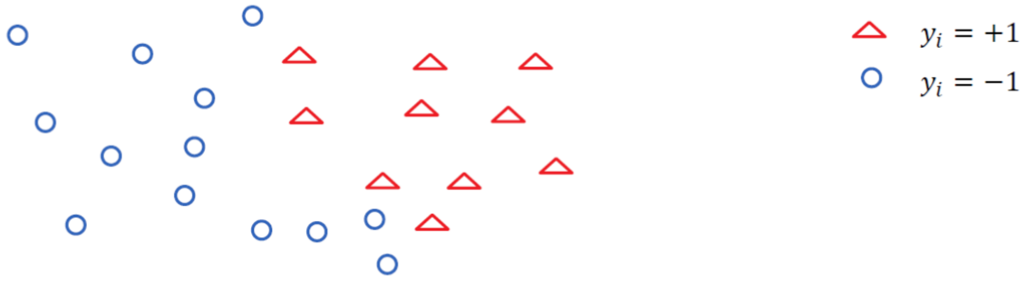
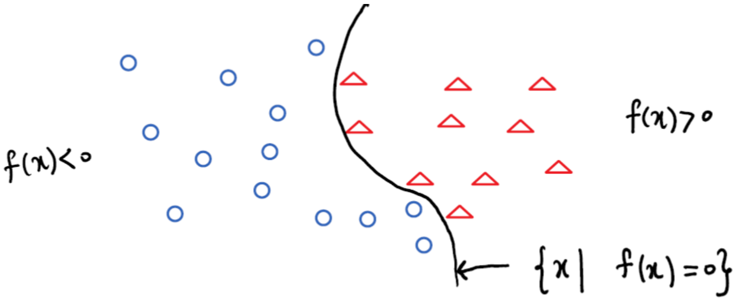


Figure 5: An example of a labeled training with only two features

- The goal is now to find a classifier $f : \mathbb{R}^n \to \mathbb{R}$, which takes a positive value on spam emails and a negative value on non-spam emails.
- The zero level set of f serves as a classifier for future predictions.



- We can search for many classes of classifier functions using convex optimization.
- The simplest one is *linear classification*: $f(x) = a^T x - b$.
- Here, we need to find $a \in \mathbb{R}^n$, $b \in \mathbb{R}$ that satisfy

$$a^T x_i - b > 0 \text{ if } y_i = 1$$
$$a^T x_i - b < 0 \text{ if } y_i = -1$$

- This is equivalent (why?) to finding $a \in \mathbb{R}^n$, $b \in \mathbb{R}$ that satisfy:

$$y_i(a^T x_i - b) \geq 1, \; i = 1, \ldots, m.$$

- This is a convex feasibility problem (in fact a set of linear inequalities). It may or may not be feasible (compare examples above and below). Can you identify the geometric condition for feasibility of linear classification? (Hint: think of convex hulls.)

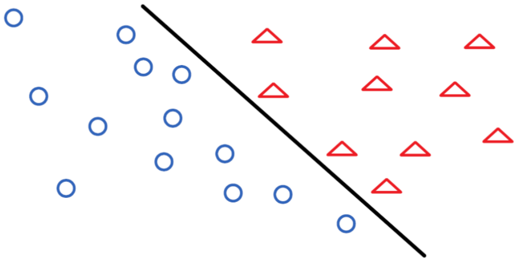
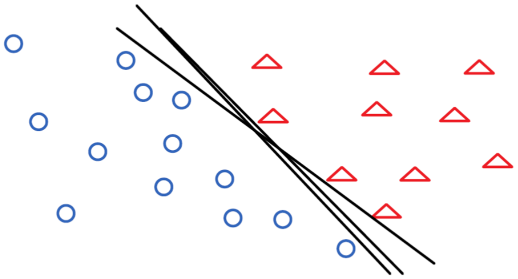


Figure 6: An example of linearly separable data

- When linear separation is possible, there could be many (in fact infinitely many) linear classifiers to choose from. Which one should we pick?



- As we explain next, the following optimization problem (known as *maximum-margin SVM*) tries to find the most "robust" one:

$$\begin{aligned} &\min_{a,b} ||a|| \\ &\text{s.t.} \quad y_i(a^T x_i - b) \geq 1, \; i = 1, \dots, m. \end{aligned} \tag{3}$$

  - This is a convex optimization problem (why?).
  - Its optimal solution is unique (why?).
  - But what exactly is this optimization problem doing?

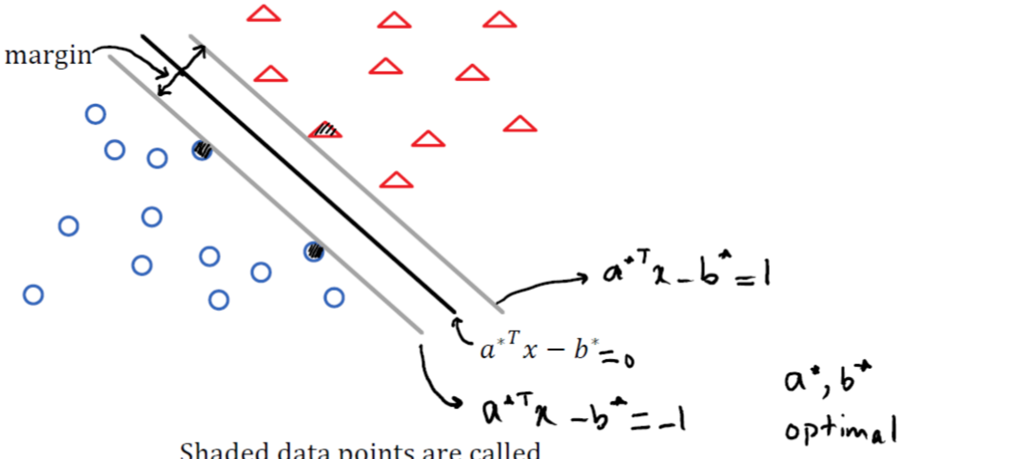
Claim 1. *The optimization problem above is equivalent to*

$$\begin{aligned}
&\max_{a,b,t} t \\
&s.t.\ \ y_i(a^T x_i - b) \geq t,\ \ i = 1, \dots, m, \qquad (4)\\
&||a|| \leq 1.
\end{aligned}$$

Claim 2. *An optimal solution of* (4) *always satisfies* $||a|| = 1$*.*

Claim 3. *The Euclidean distance of a point* $v \in \mathbb{R}^n$ *to a hyperplane* $a^T z = b$ *is given by*

$$\frac{|a^T v - b|}{||a||}.$$



Shaded data points are called
support vectors

- Let's believe these three claims for the moment. What optimization problem (3) is then doing is finding a hyperplane that *maximizes the minimum distance* between the hyperplane (our classifier) and any of our data points. Do you see why?

- We are trying to end up with as wide a margin as possible. Formally, the margin is defined to be the distance between the two gray hyperplanes in the figure above. What is the length of this margin in terms of a^* (ans possibly b^*)?

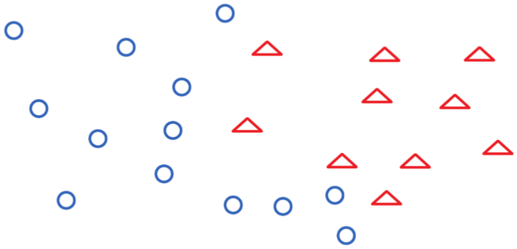
- Having a wide margin helps us be robust to noise, in case the feature vector of our future data points happens to be slightly misspecified.

The proof of the three claims are given as homework. Here are a few hints:

- Claim 1: how would you get feasible solutions to one from the other?
- Claim 2: how would you improve the objective if it didn't?
- Claim 3: good exercise of our optimality conditions.

4.1 Data that is not linearly separable

- What if the data points are not linearly separable?



- Idea: let's try to minimize the number of points misclassified:

$$\begin{aligned}&\min_{a,b,\eta} ||\eta||_0\\ &\text{s.t. } y_i(a^T x_i - b) \geq 1 - \eta_i,\ i = 1, \ldots, m\\ &\eta_i \geq 0,\ i = 1, \ldots, m.\end{aligned}$$

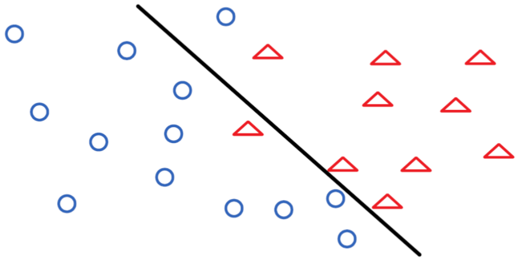
- Here, $||\eta||_0$ denotes th number of nonzero elements of η.
- If $\eta_i = 0$, data point i is correctly classified.
- The optimization problem above is trying to set as many entries of η to zero as possible.
  - Unfortunately, it is a hard problem to solve.
  - Which entries to set to zero? Many different subsets to consider.
  - As a powerful heuristic for this problem, people solve the following problem instead:

$$\begin{aligned}&\min_{a,b,\eta} ||\eta||_1\\ &\text{s.t. } y_i(a^T x_i - b) \geq 1 - \eta_i,\ i = 1, \ldots, m\\ &\eta_i \geq 0,\ i = 1, \ldots, m.\end{aligned}$$

- This is a convex program (why?). We can solve it efficiently.
- The solution with minimum l_1 norm tends to be sparse; i.e., has many entries that are zero.
- Note that when $\eta_i \leq 1$, data point i is still correctly classified but it falls within our margin; hence it is not “robustly classified”.
- When $\eta_i > 1$, data point i is misclassified.
- We can solve a modified optimization problem to balance the tradeoff between the number of missclassified points and the width of our margin:

$$\begin{aligned} &\min_{a,b,\eta} ||a|| + \gamma||\eta||_1 \\ &\text{s.t. } y_i(a^T x_i - b) \geq 1 - \eta_i,\ i = 1, \ldots, m \\ &\eta_i \geq 0,\ i = 1, \ldots, m. \end{aligned}$$

- $\gamma \geq 0$ is a parameter that we fix a priori.
- Larger γ means we assign more importance to reducing number of misclassified points.
- Smaller γ means we assign more importance to having a large margin.
  - Note that the length of our margin (counting both sides) is $\frac{2}{||a||}$ (why?).
- For each γ, the problem is a convex program (why?).
- On your homework, you will run some numerical experiments on this problem.



Notes

Further reading for this lecture can include Chapter 3 of [2]. You can read more about SVMs in Section 8.6 of [2].

References

[1] S. Boyd and M. Grant. *Graph implementations for nonsmooth convex programs. Recent Advances in Learning and Control*. Springer-Verlag, 2008.

[2] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, http://stanford.edu/ boyd/cvxbook/, 2004.

[3] Y. Crama. Recognition problems for special classes of polynomials in 0-1 variables. *Mathematical Programming*, 44, 1989.

[4] Inc. CVX Research. *CVX: Matlab software for disciplined convex programming, version 2.0*. Available online at http://cvxr.com/cvx, 2011.

[5] D.Z. Du, P.M. Pardalos, and W. Weili. *Mathematical Theory of Optimization*. Kluwer Academic Publishers, 2001.

[6] J.R. Shewchuk. *An introduction to the conjugate gradient method without the agonizing pain*. Carnegie-Mellon University. Department of Computer Science, 1994.

[7] R. Tibshirani. Regression shrinkage and selection via the Lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.