Lecture 1:

Joint distribution, Conditional probability, Hypothesis, Loss function

Lecture 2:

Statistical Learning Theory. (Learning Algorithm)

Empirical Risk Minimization (ERM)

Hoeffding's Inequality (Expected risk vs. empirical risk).

Analysis of ERM. Flawed Version.

Lecture 3:

Empirical Risk Minimization (ERM) / Structure Risk Minimization (SRM).

Incorrect Argument for ERM $(\lambda, \mathcal{H})$

post-hoc generalization guarantee. $(|\mathcal{H}|)$

(needed # of sample)

(relative) learning guarantee

overfitting
- Generalization error $R_{01}(\hat{h}_K)$
- Estimation error $(R_{01}(\hat{h}_K) - R_{01}(h))$
- Approximation error $R_{01}(h_K^*)$
- Training Error $\hat{R}_{01}(\hat{h}_K)$

complexity Level

Lecture 4:

The growth function $\Pi(\mathcal{H}, S)$ - How many behaviors it can have.

Vapnik-Chervonenkis (VC) dimension. (shattering)

Lecture 5-6:

logistic loss : $\log(1+e^{-y h(x)}) \approx -y\,h(x) = -1 \cdot z$ (Lipschitz constant 1).

A: Convex function & Lipschitz function.

Linear predictors : $h(w;x)$ has to be linear

regularization : $\|w\|^2$ to separate points with a margin (Margin loss)

Rademacher Complexity : $RC_m(\mathcal{H}) = E_x E_{\sigma \sim \text{unif}\{\pm 1\}^n}\left[\sup_{h \in \mathcal{H}} \frac{1}{m} \left|\sum_{i=1}^{m} \sigma_i h(x_i)\right|\right]$

Lecture 7-8: (SVM)

loss function: 0-1 loss function, Margin loss function, Logistic loss function

hinge loss function $\text{loss}_h(h(x), y) = \begin{cases} 0, & y h(x) \geq 1 \\ 1 - y h(x), & y h(x) < 1 \end{cases}$ | $\log(1+e^{-y h(x)})$.

lagrangian duality.

KKT condition

Kernel Methods

Lecture 9.

1). Gradient Descent Method.

a). Sufficient Decrease Condition: $f(x^k + t^k d^k) \le f(x) + \alpha t^k \nabla f(x)^T d^k$ ($\nabla f(x)^T d^k < 0$)

b). Rate of Convergence. $f(x^k) - f(x^*) \le C^k (f(x^0) - f(x^*))$

$C = \frac{r-1}{r+1}$, $r = \frac{L}{\mu}$ $\Rightarrow$ exact line search.

2). Steepest descent Vs. Gradient descent.

3). Newton step: second order approximation

Lecture 10.

1). Newton Method: $\Delta x_{nt} = -\nabla^2 f(x)^{-1} \nabla f(x)$

a). Quadratic Approximation Model. (图1)

b). Convergence Analysis

c). Self-concordant. $|f'''(x)| \le 2 f''(x)^{\frac{3}{2}}$.

2). Interior Point Method: (IPM).

a). barrier function: $\min \frac{1}{2} x^T Q x + c^T x$ s.t. $Ax = b$, $x \ge 0$ $\Rightarrow$ $\min \frac{1}{2} x^T Q x + c^T x - \mu \sum_{i=1}^{n} \ln x_i$ s.t. $Ax = b$.

b). Optimality conditions for SVM. (Relaxed KKT conditions)

Lecture 11-12.

1). Lipschitz Continuity.

2). Lipschitz Smooth function $\rightarrow$ { Linear lower approximation; Quadratic upper approximation. $Q(y)$

$\Rightarrow \min Q(y) \Rightarrow y = x - \mu \nabla f(x)$, $\mu \le \frac{1}{L}$

3). Basic Proximal Gradient Method
Algorithm & Rate of Convergence

4). Accelerated Gradient Method

5). Fast iterative shrinkage-thresholding algorithm (FISTA)

6). Deal with the non-smooth function.

Lecture 13.

1). Sparse Optimization (Shrinkage operator)

2). Gradient prox method. (Quadratic) $\leftrightarrow$ ISTA.

3). Fast first-order Method

Lecture 14:

1). Compressed Sensing.

2). Signal recovery (Restricted Isometry Property).

Isometric constant.

3). $\|Ax-b\|_n + \lambda \|x\|_m$

$(n,m) = \begin{cases} (2,1) : \underline{\text{lasso}} \text{ or Sparse regularized regression.} \\ (2,2) : \underline{\text{Ridge regression}} \\ \vdots \end{cases}$

4). Lasso $\leftrightarrow$ Convex QP.

Lecture 15:

1). Coordinate descent method (to solve lasso).

Soft-thresholding operator (shrinkage)

2). First Order method

3). Group Sparsity.

Lecture 16:

1). Stochastic Gradient Descent Method.

2). Error Decomposition.

3). relationship between CD & SGD.

Lecture 17-18:

1). vector norms & matrix norms.

2). Eigenvalues & Eigenvectors.

3). positive semidefinite matrices.

4). trace.

5). singular values, nuclear norm.

6). convex sets, norm cone.

7). Lagrangian duality, KKT condition

8). Proper cone.

9). Dual cone & generalized inequalities, self-dual

Lecture 1.

Data

- **Data and Labels**

In learning we seek a mapping from the initial data $\mathcal{X}$ (the domain of abstract input objects) to some label set $\mathcal{Y}$ (anything we want to predict).

Example: In character recognition $\mathcal{X}$ consists of possible images of letters and $\mathcal{Y}$, consists of the twenty-six letters of the Latin alphabet.

Note: For simplicity we will use binary labels $\{+1, -1\}$. Whether something is the letter "G" $(+1)$ or not the letter "G" (-1), or whether a given image contains a face $(+1)$ or does not contain a face (-1).

Joint distribution

- **Joint distribution** $p_{X,Y}(x, y)$

Future data is coming from some unknown source joint distribution $p_{X,Y}$ over input objects and their corresponding labels, which we write as the joint distribution $p_{X,Y}(x, y)$, where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

Example: Character recognition source distribution would assign much more probability to ("image containing a circular shape", "O") than to ("image containing a circular shape","T").

Conditional probability

- **Conditional probability** $p_{Y|X}(y|x)$

We can define source joint distribution as really having two components:

$$p_{X,Y}(x, y) = p_{Y|X}(y|x) \cdot p_X(x)$$

Where $p_{Y|X}(y|x)$ is the conditional probability of the label random variable Y given the appearance random variable X and $p_X(x)$ is marginal probability of the input image.

Note: $p_{Y|X}(y|x)$, is defined as "correctness" for a predictor.

Example: In character recognition we may have:

$p_{Y|X}(Y = \text{"A"}|X = \text{"image of an } A\text{"}) = 1$

$p_{Y|X}(Y = \text{"C"}|X = \text{"image of an A"}) = 0$

Hypothesis and Loss function

- **Hypothesis** h

A hypothesis (a predictor) h is a function from X to Y, $h : X \mapsto Y$.

- **Loss function** $loss_{01}(h(x), y)$

How we can evaluate the performance of h on a given (input,label) pair (x, y)?

If the label $h(x)$ does not match the provided label y, we incur a loss of 1 and if the prediction $h(x)$ does match the provided label y, we incur 0 loss.

The loss function that represents this measure of performance is called the 01 loss and defined as:

$$loss_{01}(h(x), y) = \begin{cases} 1 & \text{if } h(x) \neq y \\ 0 & \text{if } h(x) = y \end{cases}$$

Expected Risk

- **Expected Risk** $R_{01}[h(\cdot)]$

How well we expect to do (on average) over the entire (admittedly unknown) source joint distribution $p_{X,Y}(x, y)$?

The expected risk $R_{01}[h(\cdot)]$ of a hypothesis h on that distribution $p_{X,Y}(x, y)$, measures the performance of this hypothesis by evaluating its expected loss over pairs (x, y) drawn from the distribution:

$$R_{01}[h(.)] = \mathbb{E}_{(X,Y)\sim P_{X,Y}}[loss_{01}(h(X), Y)] = \sum_{X,Y} P(x, y)\, loss(h(x), y)$$

Note: Other terms with the same meaning are *expected loss*, *generalization error*, or *source-distribution risk*.

Additional property of Expected Risk

- A predictor h is "good" on a particular source joint distribution if it has low risk $R[h(.)]$ on that distribution.
- The 01 risk $R_{01}[h(.)]$ is the probability that the predictor h will incorrectly predict the label for any pair (x, y) drawn at random from the source joint distribution:

$$R_{01}[h(.)] = \mathbb{E}_{(X,Y)\sim P_{X,Y}}[loss_{01}(h(X), Y)] = \mathbb{P}_{(X,Y)\sim P_{X,Y}}\{h(X) \neq Y\}$$

This equivalence between the risk and the probability of incorrect label prediction holds only for the 01 loss.

Note: We will be assuming that the source joint distribution is fixed.

Optimization Methods In Machine Learning

Lecture 2: Finding a "good" hypothesis

Professor Katya Scheinberg

Lehigh University

Spring 2016

Outline

This lecture is taken from a short course at UT Austin taught by N. Srebro and K. Scheinberg in 2011.

Outline

Ideal case

Consider we have sample space of X and space of all possible labels Y. We are trying to find a predictor such that minimize the expected loss.

- Assume that we have complete knowledge of the true source joint distribution $p_{X,Y}(x, y)$.
- Also, we have chosen loss function $\mathbf{loss}(\cdot, \cdot)$.

Question: How does actual practice differ from this ideal setting?

- We do not ever have complete knowledge of the true source joint distribution.

Outline

Sampling

In real world, we just can sample from our population $\mathcal{X}$. In order to move forward with the process of learning a predictor, we need to make some assumptions about how this sample data set is drawn.

- It is possible to make various assumptions about the sampling of the data set.
- We will consider **Statistical Learning Theory**.

We assume:

- we are given a particular observed sample data set s of m (input, label) pairs, written $s = \{(x_1, y_1), \ldots, (x_m, y_m)\}$.
- each point is an observation of the joint random variables (X_i, Y_i).
- They are independently and identically distributed (i.i.d.) according to the source joint distribution $p_{X,Y}(x, y)$.
- i.e. each random variable pair (X_i, Y_i) is sampled independently according to $p_{X,Y}$[1]:

$$(X_i, Y_i) \underset{\text{ind.}}{\sim} p_{X,Y}$$

[1] Often, the sample data set is *not* drawn from the same distribution that we will measure our expected loss on. **But** any type of analysis of machine learning methods assumes that the sample data set is drawn from the same distribution that we use to measure the error

Choosing a predictor

- Assume that we have a loss function that characterizes what we care about.
- The process of learning from a sample data set is a mapping from a particular observed sample data set s and a loss function $\mathbf{loss}(\cdot,\cdot)$ to a predictor h.

$$[s, \mathbf{loss}(\cdot,\cdot)] \mapsto h.$$

- Learning algorithm takes a loss function and a sample data set of labeled examples and returns a predictor.[2]

[2] This is what we consider in this course. For this course, we will primarily be considering just simple supervised learning in which we would like to find a good predictor based on a labeled sample data set.

Empirical risk

- We want to find a predictor that minimizes the expected loss on the true source joint distribution, but ...
- We do not have complete knowledge of the true source joint distribution.
- We should choose our predictor to minimize the expected loss on what we do have access to.
- We hope that this predictor will do well on the true source joint distribution.
- The expected loss of a predictor h on a particular observed sample data set $s = \{(x_1, y_1), \ldots, (x_m, y_m)\}$ could also be referred to as the *empirical risk* $\hat{R}_s[h(\cdot)]$

$$\hat{\mathbf{E}}_s[\mathbf{loss}(h(X), Y)] = \hat{R}_s[h(\cdot)] = \frac{1}{m}\sum_{i=1}^{m} \mathbf{loss}(h(x_i), y_i)$$

Outline

Empirical risk minimizer

- As the first step of Empirical Risk Minimization we choose some hypothesis class $\mathcal{H}$.
- We view $\mathcal{H}$ as a set of predictors, written as

$$\mathcal{H} = \{h \;:\; \mathcal{X} \to \mathcal{Y}\} \subseteq \mathcal{Y}^{\mathcal{X}}$$

- The Empirical Risk Minimization learning rule can be written as:

$$\mathbf{ERM}_{\mathcal{H}}(s) = \hat{h} = \operatorname*{argmin}_{h\in\mathcal{H}} \hat{R}_s(h) = \operatorname*{argmin}_{h\in\mathcal{H}} \frac{1}{m}\sum_{i=1}^{m} \mathbf{loss}(h(x_i), y_i)$$

Example of hypothesis classes

- Consider binary labels, so the label set is $\mathcal{Y} = \{+1, -1\}$. In addition, consider $\mathcal{X} = \mathbb{R}^2$.
- Some specific examples of hypothesis classes:

$$\begin{aligned}
\mathcal{H} &= \{x_i \geq \theta \mid i \in \{1,2\}, \theta \in \mathbb{R}\} \\
\mathcal{H} &= \left\{x \mapsto \mathbf{sign}\left(w^T x + b\right) \mid w \in \mathbb{R}^2, b \in \mathbb{R}\right\} \\
\mathcal{H} &= \left\{\sum_{i=1}^{2} x_i \leq \theta \mid \theta \in \mathbb{R}\right\}
\end{aligned} \tag{1}$$

- We can use any features of x_1 and x_2 to make a new class of hypothesis.

$$\mathcal{H} = \{\phi(x_i) \geq \theta \mid i \in \{1,2\}, \theta \in \mathbb{R}\} \tag{2}$$

- In a learning problem, we limit ourselves only to hypotheses in a certain class.
- We want to make the difference of expected risk and empirical risk, as small as possible.

Outline

Analysis of ERM

- Consider a specific predictor h.
- This predictor has an expected 01 loss $R_{01}(h)$ with respect to the true source joint distribution $p_{X,Y}(x, y)$.
- We only have access to an *estimate* of the expected 01 loss of the predictor h, $R_{01}(h)$, in the form of the sample average 01 loss $\hat{R}_{s,01}(h)$ of the predictor.
- There are many samples of size m that can be drawn from the true source distribution.
- If the sample average 01 loss of the predictor h is usually close to the expected 01 loss of the predictor h, then we can feel more confident about using the sample average 01 loss in place of the expected 01 loss.

Hoeffding's Inequality

- Let $T_1, \ldots, T_m$ be *independent* scalar random variables.
- Assume further that the T_i are *bounded* so that $T_i \in [a_i, b_i]$.
- Then, for the empirical mean of these m bounded variables,

$$\overline{T} = \frac{1}{m}\sum_{i=1}^{m} T_i,$$

- We have the inequality:

$$\mathbb{P}\left\{\left|\overline{T} - \mathbb{E}\left[\overline{T}\right]\right| \geq \varepsilon\right\} \leq 2\exp\left(-\frac{2\varepsilon^2 m^2}{\sum_{i=1}^{m}(b_i - a_i)^2}\right)$$

Comparing Expected Risk and Empirical Risk

- Let define the variables for our problem:

$$\begin{aligned} T_i &= \text{loss}_{01}\left(h\left(X_i\right), Y_i\right) \\ a_i &= 0 \qquad\qquad \text{for } i \in \{1, 2\} \\ b_i &= 1 \end{aligned} \tag{3}$$

- Now we have:

$$\begin{aligned} \mathbb{E}(\hat{R}_{01}(h(x)) &= \mathbb{E}(\frac{1}{m}\sum_{i=1}^{m}\text{loss}_{01}\left(h\left(X_i\right), Y_i\right)) \\ &= \frac{1}{m}\sum_{i=1}^{m}\mathbb{E}(\text{loss}_{01}\left(h\left(X_i\right), Y_i\right)) \\ &= \frac{1}{m}\sum_{i=1}^{m} R_{01}(h(x)) \\ &= R_{01}(h(x) \end{aligned} \tag{4}$$

- The expectation of empirical loss is expected loss.

Comparing Expected Risk and Empirical Risk

- Use the Hoeffding's Inequality:

$$\mathbb{P}\left\{\left|R_{01}(h(.)) - \hat{R}_{01}(h(.))\right| \geq \varepsilon\right\} \leq 2\exp\left(-2\varepsilon^2 m\right) \tag{5}$$

- If we have a sample with size m, the probability of the difference between expected loss and empirical loss be less than ϵ is:

$$\mathbb{P}\left\{\left|R_{01}(h(.)) - \hat{R}_{01}(h(.))\right| < \varepsilon\right\} \geq 1 - 2\exp\left(-2\varepsilon^2 m\right) = 1 - \delta \tag{6}$$

- So the sample size that grantees the accuracy of ϵ with probability of $1 - \delta$ can be found by considering this equation:

$$\epsilon = \sqrt{\frac{\log\frac{2}{\delta}}{2m}} \tag{7}$$

Comparing Expected Risk and Empirical Risk (Cont.)

- So we can rewrite the inequality as follows:

$$\mathbb{P}\left\{\left|R_{01}(h(.)) - \hat{R}_{01}(h(.))\right| < \sqrt{\frac{\log\frac{2}{\delta}}{2m}}\right\} \geq 1 - \delta \tag{8}$$

- This inequality tells us just by changing the sample size, we can control the accuracy.
- What does this mean? What does the probability δ mean? How should we understand this inequality?

ERM (Empirical Risk Minimization).

- We want to find a hypothesis that minimizes the *expected risk*. However, as discussed, we can only find a hypothesis that minimizes empirical risk. Call such a hypothesis $\hat{h}$.

$$\hat{h} = \arg\min_{h \in \mathcal{H}}\{\hat{R}(h(.)) = \frac{1}{m}\sum_{i}\text{loss}\left(h(x_i), y_i\right)\} \tag{9}$$

- We produce a sample set and then search for the $\hat{h}$ that minimizes the empirical risk.
- What can we say about $\hat{h}$? How "good" is it?

Motivation | Real world solution | Hypothesis classes | **Analysis of ERM**

Analysis of ERM. Flawed Version!

- Let's take a close look to the inequality (8). With probability of $1-\delta$ we have:

$$R_{01}(h(.)) < \hat{R}_{01}(h(.)) + \sqrt{\frac{\log\frac{2}{\delta}}{2m}} \tag{10}$$

- This means the expected risk cannot be larger than the empirical risk plus an error. That is exactly what we want. Can we use it?

Analysis of ERM. Flawed Version!

of the distribution.

- The best hypothesis of the class $\mathcal{H}$ is the following:

$$h^* = \arg\min_{h\in\mathcal{H}}\{R(h(.)) = E\{\mathbf{loss}(h(x), y)\}\} \tag{11}$$

- According to 10 and 11, we can find the following relationship:

$$R_{01}(h^*(.)) \le R_{01}(h(.)) < \hat{R}_{01}(h(.)) + \sqrt{\frac{\log\frac{2}{\delta}}{2m}} \tag{12}$$

- And based on (8) and (9), we have:

$$\hat{R}_{01}\left(\hat{h}(.)\right) \le \hat{R}_{01}(h^*(.)) \le R_{01}(h^*(.)) + \sqrt{\frac{\log\frac{2}{\delta}}{2m}} \tag{13}$$

add hat ^ means sample.

$\hat{h}(\cdot)$: best hypothesis of sample ($\hat{R}_{01}$).

Analysis of ERM. Flawed Version!

- By using inequalities 10 and 13, following inequality will achieved:

$$R_{01}\left(\hat{h}(.)\right) < \hat{R}_{01}\left(\hat{h}(.)\right) + \sqrt{\frac{\log\frac{2}{\delta}}{2m}} \le R_{01}(h^*(.)) + 2\sqrt{\frac{\log\frac{2}{\delta}}{2m}} \tag{14}$$

- We now have shown that expected risk of our ERM $R_{01}\left(\hat{h}(.)\right)$ is not very different from expected risk of the expected risk minimizer h^*, which is the best we can hope for. The difference gets smaller if the sample set gets bigger.

- Sounds perfect!! But is it wrong! Why?

X.

Example

- Consider a problem with following data set X and labels Y:

$$X = \{\text{"People in the US"}\} \quad Y = \{\text{"Male","Female"}\} \tag{15}$$

- There is four kinds of different hypothesis classes:

$$\begin{aligned} H_b &= \{\text{"Predictor based only on month and day of birth"}\} \\ H_n &= \{\text{"Predictor based only on nationality"}\} \\ H_m &= \{\text{"Predictor based only on length of hair"}\} \\ H_p &= \{\text{"Predictor based only on last four digits of phone"}\} \end{aligned} \tag{16}$$

- Which of these hypothesis is the best?

- What is wrong with $2\sqrt{\frac{\log\frac{2}{\delta}}{2m}}$?

Hoeffding Inequality Proof: $X_1, X_2, \cdots X_N \in [a,b]$.

$P_Z(f(x)>\varepsilon) = E_Z(1\{f(x)-\varepsilon>0\})$

$e^{\mu x}$ ↑ $1(x)$ → $E_Z(1\{f(x)-\varepsilon>0\}) \le E_Z(\exp(\mu(f(x)-\varepsilon)))$

$P\{\bar{X}-E(\bar{X})>\varepsilon\} = P\{N\bar{X}-NE(\bar{X})>N\varepsilon\}$

$\le E(\exp(N\mu\bar{X}-N\mu E(\bar{X})-N\mu\varepsilon))$

$= \exp(-N\mu\varepsilon)\cdot\prod_{i=1}^{N} E(\exp(\mu X_i-\mu E(X_i)))$

$E(\exp(\mu X_i-\mu E(X_i))) = \exp(-\mu E(X_i))\,E(\exp(\mu X_i))$

$\le \exp(-\mu E(X_i))\left[\frac{b-E(X_i)}{b-a}e^{\mu a}+\frac{E(X_i)-a}{b-a}e^{\mu b}\right]$

$= \exp(-\mu(E(X_i)-a))\left[1-\frac{E(X_i)-a}{b-a}+\frac{E(X_i)-a}{b-a}e^{\mu(b-a)}\right]$ (P_i)

$= \exp(-\mu(b-a)P_i)\left[1-P_i+P_i e^{\mu(b-a)}\right]$ ($=y_i$)

$= \exp\left[-y_i P_i + \log(1-P_i+P_i e^{y_i})\right]$

P_i 是常数. y_i based on μ.

Let $L(y_i) = -y_i P_i + \log(1-P_i+P_i e^{y_i})$

$L'(y_i) = -P_i + \frac{P_i e^{y_i}}{1-P_i+P_i e^{y_i}} = -P_i + \frac{P_i}{(1-P_i)e^{-y_i}+P_i}$

$L''(y_i) = \frac{P_i\times(1-P_i)e^{-y_i}}{\left[(1-P_i)e^{-y_i}+P_i\right]^2} \le \frac{1}{4}$.

$L(y_i) = L(0) + L'(0)y_i + \frac{1}{2}L''(0)y_i^2 \le \frac{1}{8}y_i^2$.

$\therefore P\{\bar{X}-E(\bar{X})>\varepsilon\} \le \exp(-N\mu\varepsilon+\frac{N}{8}\mu^2(b-a)^2)$

Let $M(\mu) = -N\mu\varepsilon+\frac{N}{8}\mu^2(b-a)^2$, $\mu>0$.

$= \frac{N}{8}\mu(\mu(b-a)^2-8\varepsilon)$

$\therefore$ Let $\mu = \frac{4\varepsilon}{(b-a)^2} \Rightarrow M_{min} = -\frac{2\varepsilon^2 N}{(b-a)^2}$.

$\therefore P\{\bar{X}-E(\bar{X})>\varepsilon\} \le \exp(-\frac{2\varepsilon^2 N}{(b-a)^2})$.

$\therefore P\{|\bar{X}-E(\bar{X})|>\varepsilon\} \le 2\exp(-\frac{2\varepsilon^2 N}{(b-a)^2})$.

Optimization Methods In Machine Learning

Lecture 3: Empirical Risk Minimization and Structure Risk Minimization

Professor Katya Scheinberg

Lehigh University

Spring 2016

Outline

This lecture is taken from a short course at UT Austin taught by N. Srebro and K. Scheinberg in 2011.

Outline

Empirical Risk Minimization (ERM)

Incorrect Argument for ERM

- Recall examples of complexity of hypothesis class from the previous lecture:
  - $\mathcal{H}_b$ = {"predictors based only on month and day of birthdate"}, $|\mathcal{H}_b| = 2^{365}$.
  - $\mathcal{H}_n$ = {"predictors based only on nationality"}, $|\mathcal{H}_n| = 2^n$, n is the number of countries.
  - $\mathcal{H}_h$ = {"predictors based only on short or long hair"}, $|\mathcal{H}_h| = 2^2$.
  - $\mathcal{H}_p$ = {"predictors based only on last four digits of phone number"}, $|\mathcal{H}_p| = 2^{10000}$.

Assuming $R \simeq \hat{R}$.
∴ make sure all h, Hoeffding's inequality.

Empirical Risk Minimization (ERM)

Correcting Argument for justifying ERM

- Recall our re-interpretation of Hoeffding's inequality:

只有一个h在这个概率下满足不等式

$$|R_{01}(h) - \hat{R}_{s,01}(h)| \le \sqrt{\frac{\log \frac{2}{\delta}}{2m}} \text{ with probability at least } (1-\delta).$$

Then recall from our previous lecture that

这个时候不一定其他h也满足 Hoeffding's inequality

$$R_{01}(\hat{h}) \le \hat{R}_{s,01}(\hat{h}) + \sqrt{\frac{\log \frac{2}{\delta}}{2m}} \le \hat{R}_{s,01}(h^*) + \sqrt{\frac{\log \frac{2}{\delta}}{2m}}$$
$$\le R_{01}(h^*) + 2\sqrt{\frac{\log \frac{2}{\delta}}{2m}}$$

$|\mathcal{H}|$. $R \simeq \hat{R}$.

- What is flawed in the above argument?
  Reason: the problem is that Hoeffding's inequality holds for each h separately but does NOT hold for all $h \in \mathcal{H}$. We neglected the complexity of the hypothesis class in our previous argument.

所有可能的hypothesis的集合

Empirical Risk Minimization (ERM)

Correcting Argument for justifying ERM

- Recall the knowledge of the union bound from probability theory

Theorem (Boole's inequality, also known as the union bound)
For a countable set of different events $A_1, A_2, \ldots$, we have

$$\mathbb{P}(\bigcup_i A_i) \le \sum_i \mathbb{P}(A_i).$$

error of accuracy

- Correct the flaw by using the union bound, we should have the following inequality hold.

how well accuracy ⇔ complexity.

Theorem

$$\mathbb{P}\{\forall h \in \mathcal{H}, |\hat{R}_{s,01}(h) - R_{01}(h)| \le \sqrt{\frac{\log|\mathcal{H}| + \log \frac{2}{\delta_{bad}}}{2m}}\} \ge 1 - \delta_{bad}.$$

Empirical Risk Minimization (ERM)

Correcting Argument for justifying ERM

Proof Considering the union bound, let event A_i = "hypothesis h_i looks misleadingly good", i.e. "h_i is cheating, explicitly written out as $|R_{01}(h_i) - \hat{R}_{s,01}(h_i)| \geq \epsilon$. Then $\bigcup_{i\in\mathcal{H}} A_i$ = "at least one hypothesis from $\mathcal{H}$ is cheating". So the following inequality holds,

$$\begin{aligned}
&\mathbb{P}(\text{all hypotheses from } \mathcal{H} \text{ "behave well"}) \\
&= 1 - \mathbb{P}(\text{at least one hypothesis from } \mathcal{H} \text{ is cheating}) \\
&= 1 - \mathbb{P}(\bigcup_{i\in\mathcal{H}} A_i) \geq 1 - \sum_{i\in\mathcal{H}} \mathbb{P}(A_i) \\
&= 1 - \sum_{i\in\mathcal{H}} \mathbb{P}\{|R_{01}(h_i) - \hat{R}_{s,01}(h_i)| \geq \epsilon\}
\end{aligned}$$

Empirical Risk Minimization (ERM)

Correcting Argument for justifying ERM

- Event "all hypotheses from $\mathcal{H}$ 'behave well' " can be explicitly written as "$\forall h \in \mathcal{H}, |\hat{R}_{s,01}(h) - R_{01}(h)| \leq \epsilon$. Then

$$\begin{aligned}
&\mathbb{P}\{\forall h \in \mathcal{H}, |\hat{R}_{s,01}(h) - R_{01}(h)| \leq \epsilon\} \\
&\geq 1 - \sum_{i\in\mathcal{H}} \mathbb{P}\{|R_{01}(h_i) - \hat{R}_{s,01}(h_i)| \geq \epsilon\}
\end{aligned}$$

Recall from our previous lecture $\mathbb{P}\{|R_{01}(h) - \hat{R}_{s,01}(h)| \geq \epsilon\} \leq 2e^{-2\epsilon^2 m}$ for any $h \in \mathcal{H}$, the above inequality is equivalent to

$$\mathbb{P}\{\forall h \in \mathcal{H}, |\hat{R}_{s,01}(h) - R_{01}(h)| \leq \epsilon\} \geq 1 - 2|\mathcal{H}|e^{-2\epsilon^2 m}$$

Empirical Risk Minimization (ERM)

Example

- By letting $\delta_{bad} = 2|\mathcal{H}|e^{-2\epsilon^2 m}$, we have $\epsilon = \sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}}$. Thus,

$$\mathbb{P}\{\forall h \in \mathcal{H}, |\hat{R}_{s,01}(h) - R_{01}(h)| \leq \sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}}\} \geq 1 - \delta_{bad}$$

Proof is finished here.

Example (3.1)

Suppose we have an hypothesis class of size $|\mathcal{H}| = 10000$, and we want the probability of there being some "possibly misleading" hypothesis in $\mathcal{H}$ (such that its sample date set performance differs from its source distribution performance by more than $\epsilon = 0.01$) to be no higher than $\delta_{bad} = e^{-7}$. How many samples do we need?

Solution: $\epsilon = 0.01 \leq sqrt\frac{\log 10000 + \log 2/e^{-7}}{2m} \Longrightarrow m \geq \frac{2}{17} \cdot 10^4 \approx 1200.$

Empirical Risk Minimization (ERM)

- Similar as previous lectures, let $h^* := \arg\min_{h\in\mathcal{H}} R_{01}(h)$, $\hat{h} := \arg\min_{h\in\mathcal{H}} \hat{R}_{s,01}(h)$. From the inequality

$$\mathbb{P}\{\forall h \in \mathcal{H}, |\hat{R}_{s,01}(h) - R_{01}(h)| \leq \sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}}\} \geq 1 - \delta_{bad},$$

we have: with probability at least $(1 - \delta_{bad})$,

$$\begin{aligned}
R_{01}(\hat{h}) &\leq \hat{R}_{s,01}(\hat{h}) + \sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}} \\
&\leq \hat{R}_{s,01}(h^*) + \sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}} \\
&\leq R_{01}(h^*) + 2\sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}}.
\end{aligned}$$

needed number of sample. bias variance.

Empirical Risk Minimization (ERM)

Post-hoc Guarantees

- With probability at least $(1 - \delta_{bad})$, we will have gotten a sample set realization s for which the **post-hoc generalization guarantee** holds:

$$R_{01}(\hat{h}) \leq \hat{R}_{s,01}(\hat{h}) + \sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}}.$$

Empirical Risk Minimization (ERM)

Relative Learning Guarantees

- With probability at least $(1 - \delta_{bad})$, we will have the **(relative) learning guarantee**

$$R_{01}(\hat{h}) \leq R_{01}(h^*) + 2\sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}}.$$

Empirical Risk Minimization (ERM)

Examples

Example (3.2)

Suppose for hypothesis class $\mathcal{H}$, $h(x) = \{x_i \geq \theta, i = 1, 2\}$. (Assume the machine is 64-bit.)

Actually infinite possible values

- For θ, there 2^{64} possible values, we have 2 hypotheses for each θ in 2-dimensional space.
  So the complexity of hypothesis class

$$|\mathcal{H}| = 2^{64} + 2^{64} = 2^{65} \Rightarrow \log|\mathcal{H}| \approx 45.$$

Example (3.3)

$h(x) = \{w^T x + b \geq 0\}$, $\mathcal{H} = \{h_{w,b} | w \in \mathbb{R}^d, b \in \mathbb{R}\}$. (Assume the machine is 64-bit.)

- Similarly, the complexity of hypothesis class ($D := d + 1$)

$$|\mathcal{H}| = 2^{64D} + 2^{64D} = 2^{65D} \Rightarrow \log|\mathcal{H}| \approx 45D(\text{linear}).$$

Outline

Structure Risk Minimization (SRM)

Overfitting

Example (3.4)

$x \in \mathbb{R}, y = \{+1, -1\}$. Given a series of data samples (x_i, y_i).

Figure: Given data samples

$\mathcal{H} = \{x \to \text{sign}[\sin(wx + \theta)] \ | w \in \mathbb{R}, \theta \in \mathbb{R}\}$ can overfit any data when w is sufficiently large.

Figure: Example of overfitting

Structure Risk Minimization (SRM)

- Recall from our previous slide of relative learning guarantee, with probability at least $(1 - \delta_{bad})$,

$$R_{01}(\tilde{h}) \leq R_{01}(h^*) + 2\sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}}.$$

where $R_{01}(h^*)$ is called approximation error and $R_{01}(\tilde{h}) - R_{01}(h^*)$ is the estimation error which is bounded by $2\sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}}$.

$h_0^* \to h_1^*$

$h_0^* = \arg\min R_{01}(h), h \in H_0$ $\quad |H_0| < |H_1|$

$h_1^* = \arg\min R_{01}(h), h \in H_1$

Structure Risk Minimization (SRM)

- If we are optimizing $R_{01}(h_k^*)(h_k^* \in \mathcal{H}_k)$ over a hierarchy of hypothesis classes $\mathcal{H}_0 \subseteq \mathcal{H}_5 \subseteq \mathcal{H}_{10} \subseteq \mathcal{H}_{15} \subseteq \ldots$, $R_{01}(h_k^*)$ will get smaller as more complexity parameters can make better model by fitting the data.

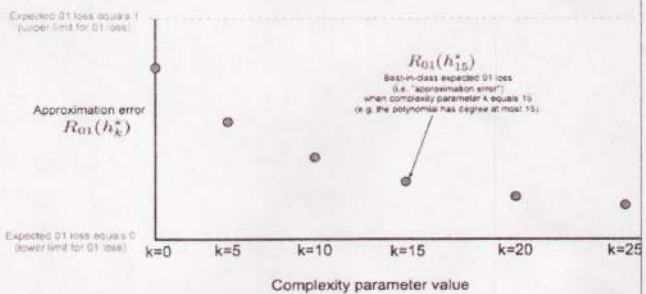


Figure: Optimizing $R_{01}(h_k^*)$ over a hierarchy of classes

Structure Risk Minimization (SRM)

- However, the bound of estimation error $2\sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta_{bad}}}{2m}}$ is increasing as complexity parameter value increases. SRM principle aims to balance the model's complexity against its success at fitting the data.

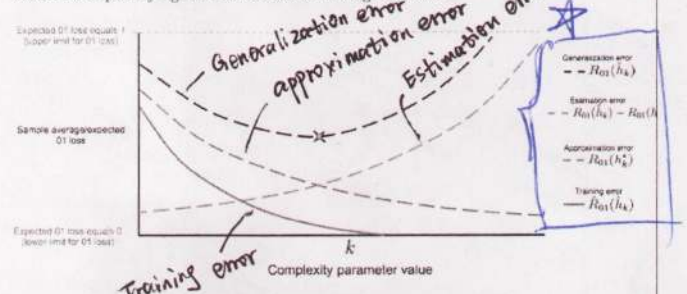


Figure: Behaviors of different errors over a hierarchy of classes

Structure Risk Minimization (SRM)

Choosing a Complexity Level

- There is a trade-off between the approximation error and the estimation error, so how do we know which class of hypothesis is better?
  1. Sample again! And test the predictors on the new sample, compare their behaviors.
  2. Cross-validation: Partition a sample set into complementary subsets, perform analysis on one subset (training set) and validate the analysis on the other subset (testing set). Can do this multiple times with different partitions.

Optimization Methods in Machine Learning
Lecure 4: Vapnik-Chervonenkis (VC) dimension

Katya Scheinberg

Lehigh University

Spring 2016

Overview

The material for this lecture is taken from a short course taught at UT Austin by N. Srebro and K. Scheinberg in 2011.

Motivation

- Recall that the bound on the difference in the expected (R) and empirical ($\hat{R}$) errors: with probability at least $1-\delta$, for all $h \in \mathcal{H}$,

$$\left|R(h) - \hat{R}(h)\right| \leq \sqrt{\frac{\log|\mathcal{H}| + \log\frac{2}{\delta}}{2m}}$$

- Recall that the cardinality of a hypothesis class is difficult to measure, many classes may be infinite in cardinality because of their parameters being continuous.
- We need a more consistent way of measuring complexity of a class of hypothesis.

why we use growth function

Growth Function

- We define the number of different behaviors that a hypothesis class has on a specific set of points as the **growth function**.

Definition

We define the growth function $\Pi(\mathcal{H}, S)$ for a hypothesis class $\mathcal{H}$ and a set of points $S = \{x_1, \ldots, x_m\}$. We look at all the possible labelings we can have $y_1, \ldots, y_m$ such that there exists some classifier in the class that actually gives these labelings.

The **growth function**

$\Pi(\mathcal{H}, S)$ = Number of different behaviors(predictions) the class of hypothesis H can generate on a sample S.

Growth Function

Example (4.1)

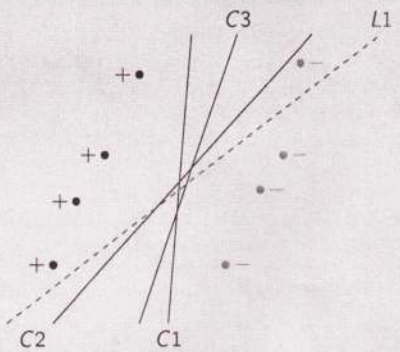


Figure: Linear classifiers with same (C1,C2,C3) and different (L1) behavior on training sample.

Why do we care about the growth function

Lemma

With probability at least $1-\delta$, for all hypotheses $h \in \mathcal{H}$

$$\left|R(h) - \hat{R}(h)\right| \leq \sqrt{\frac{\log\Pi(\mathcal{H}, 2m) + \log\frac{4}{\delta}}{m}}.$$

Growth Function, Simple Bounds

Here, instead of providing labelings to specific data sample, we are looking for the maximum number of growth function for any m data points on given space.
How big can the growth function be for a given hypothesis class $\mathcal{H}$?

- $$\Pi(\mathcal{H}, m) \leq 2^m$$
- $$\Pi(\mathcal{H}, m) \leq |\mathcal{H}|$$

Vapnik-Chervonenkis (VC) dimension

- The VC-dimension of a hypothesis class the maximal number of points for which you can get all possible behaviors.

Definition

VC-dimension($\mathcal{H}$) $\triangleq$maximal number of points m such that $\Pi(\mathcal{H}, m) = 2^m$.

- In terms of the growth function, we can just write the VC-dimension as the maximal m such that $\Pi(\mathcal{H}, m) = 2^m$.
- Being able to achieve any labeling of a given set of points is also known as *shattering* the points.
- We say that if we have m points and we can get all possible behaviors on our hypothesis class for these points then these points are shattered.

The maximum of sample numbers that the $\mathcal{H}$ can shatter.

The Shatter Lemma

The following lemma connects m, VC-dimension and the growth function.

Lemma

$$\Pi(\mathcal{H}, m) \leq \sum_{i=0}^{VC\text{-}Dim(\mathcal{H})} \binom{m}{i} \leq \left(\frac{e \cdot m}{D}\right)^{D} \overset{(D \geq 3)}{\leq} m^D$$

where D is VC-dimension($\mathcal{H}$).

The Deviation bounds using VC-dimension:

Lemma

With probability at least $1 - \delta$, for all hypotheses $h \in \mathcal{H}$

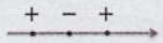
$$\left|R(h) - \hat{R}(h)\right| \leq \sqrt{\frac{D \log(2m) + \log\frac{4}{\delta}}{m}}.$$

Vapnik-Chervonenkis (VC) dimension

Example (4.2)

Let us consider a hypothesis $h = \{x \in [a, b]\}$, which labels x positive if x is in the interval [a,b], negative otherwise.
For the case of two points, we have two labelings:

However, for the case of three points, there exists some labelings that can not be generated, one case is presented below:



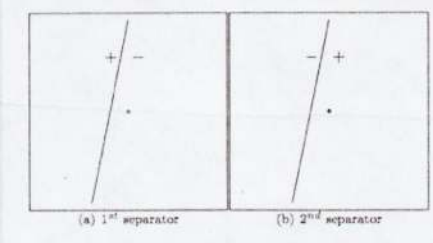
So the VC dimension for this hypothesis is 2, note that it is the same as the number of parameters of the hypothesis.

Vapnik-Chervonenkis (VC) dimension

Let us consider a particular example of linear separators in $\mathbb{R}^2$.

Example (4.3)

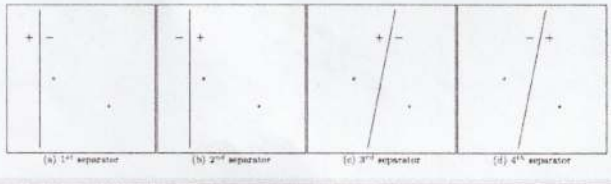
If we have only a single point, we can either label it positive or negative, hence we can label it in all 2^1 different ways. For each such labeling we can find a linear predictor that is consistent with it.



Vapnik-Chervonenkis (VC) dimension

Example (4.4)

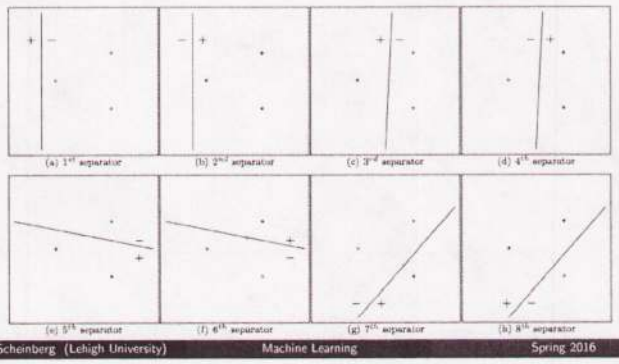
For the case of 2 points, there are $4 = 2^2$ kinds of labeling. For each such labeling we can find a linear predictor that is consistent with it.



Vapnik-Chervonenkis (VC) dimension

Example (4.5)

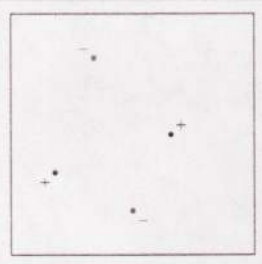
For the case of 3 points, there are $8 = 2^3$ kinds of labeling. For each such labeling we can find a linear predictor that is consistent with it.



Vapnik-Chervonenkis (VC) dimension

Example (4.6)

However, things are a little different with the case of 4 points. For the case of 4 points, there are $2^4 - 2 = 14$ kinds of labeling. As the usual 2^m number of labelings, this time there are two labeling that is not achievable by linear classifiers. Below presents one of them:



SHATTER : ① step : fix the X values (定下值).

Vapnik-Chervonenkis (VC) dimension

From the previous examples, we see that:

- For $m = 1$, $\Pi(\mathcal{H}, m) = 2^1 = 2 = 2^m$;
- For $m = 2$, $\Pi(\mathcal{H}, m) = 2^2 = 4 = 2^m$;
- For $m = 3$, $\Pi(\mathcal{H}, m) = 2^3 = 8 = 2^m$;
- For $m = 4$, $\Pi(\mathcal{H}, m) = 14 = 2^4 - 2 \neq 2^m$.

Therefore, VC-dimension $(\mathcal{H}) = 3$.

$\sup_m \{X_1, X_2, \cdots X_m\} \in X$

Not any X, $\forall x$

☆. but EX

② Label

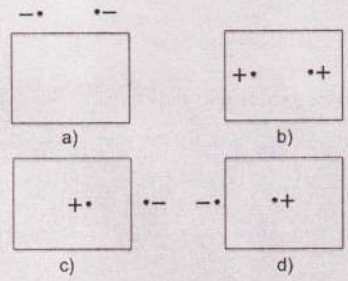
Thus: ①. 找最好分类的摆位.

②. 找最难的 labeling.

Vapnik-Chervonenkis (VC) dimension

Example (4.7)

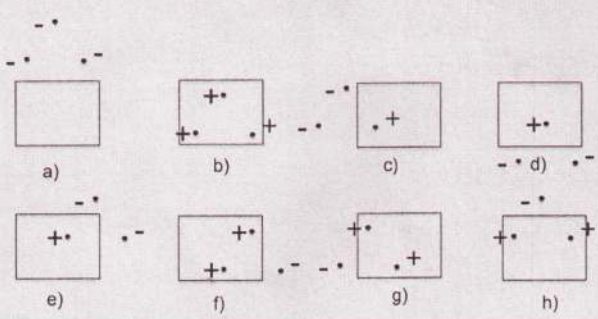
Now we look at another example, where the hypothesis labels the point inside the rectangle decided by the two points (a,b), (c,d) positive, and otherwise negative. The case for two points:



Vapnik-Chervonenkis (VC) dimension

Example (4.8)

The case for three points:



Vapnik-Chervonenkis (VC) dimension

Example (4.9)

The case for four points is a little different; It is not possible to produce all the labeling for certain situations, one of them is presented below:

Vapnik-Chervonenkis (VC) dimension

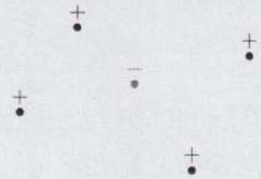
Example (4.10)

On the other hand this configuration can be shattered:

Vapnik-Chervonenkis (VC) dimension

Example (4.11)

But not this one:



Vapnik-Chervonenkis (VC) dimension

From the previous examples, we see that:

- For $m = 1$, $\Pi(\mathcal{H}, m) = 2^1 = 2 = 2^m$;
- For $m = 2$, $\Pi(\mathcal{H}, m) = 2^2 = 4 = 2^m$;
- For $m = 3$, $\Pi(\mathcal{H}, m) = 2^3 = 8 = 2^m$;
- For $m = 4$, $\Pi(\mathcal{H}, m) = 2^4 = 16 = 2^m$.
- For $m = 5$, $\Pi(\mathcal{H}, m) < 2^5 = 2^m$.

Therefore, VC-dimension $(\mathcal{H}) = 4$.

Optimization Methods in Machine Learning
Lecture 5-6: Logistic Loss and regularization

Katya Scheinberg

Lehigh University

Spring 2016

The consequence of 01 loss

- Consider the minimization the empirical error problem subject to some constraints.

$$\min_{w\in\mathbb{R}^d, b\in\mathbb{R}, z\in\mathbb{R}^d} \frac{1}{m}\sum_{i=1}^{m} \text{loss}_{01}(z_i, y_i)$$
$$\text{s.t. } z_i = w^T x_i + b$$

- z is the output of the predictor for x_i
- We care about the sign of y and z.

$$\text{loss}_{01} = \begin{cases} 0 & \text{if } y_i z_i > 0 \\ 1 & \text{if } y_i z_i \leq 0 \end{cases}$$

- If $z = 0$, it incurs a loss of 1 since the prediction can never match the true label y.

The consequence of 01 loss

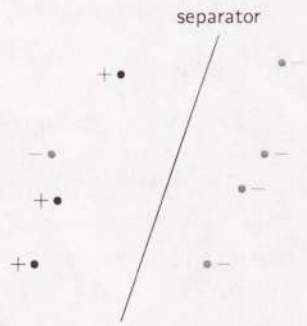
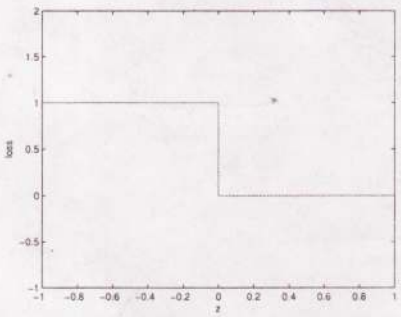


Figure: Affine separator performance in the case of 01 loss. 01 loss is not good at measuring the loss magnitude.

The consequence of 01 loss

- 01 loss function is not convex



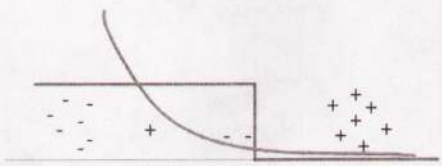
- At 0, the Lipschitz constant is ∞

Logistic Loss

Consider a hypothesis class $\mathcal{H}$ of linear functions $h(x) = w^T x + b$ of size m, and its logistic loss.

$$\min_{w,b} \frac{1}{m}\sum_i \log(1 + e^{-y_i(w^T x_i + b)}).$$

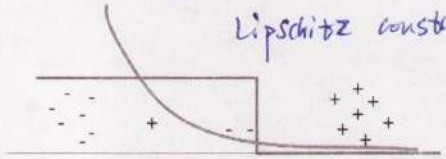
Logistic loss minimizer may give a different separation from 01 loss minimizer.



Logistic Loss

$\log(1+e^{-yh(x)}) \geq 1 - yh(x) = -1 \cdot z$

Lipschitz constant 1.



from the example, we can see logistic loss have some advantages over 01 loss:

- Convex function
- Lipschitz function

Question: Is logistic loss good enough?

Behavior of the smooth loss function

$$\min_w \; f(w) = \frac{1}{m}\sum_{i=1}^{m} \text{loss}_g(h(w,x_i), y_i)$$

- If y_i is $+1$, $\text{loss}_g\left(h\left(w,x_i\right), y_i\right)$ is a monotonically decreasing function of $h\left(w,x_i\right)$.
  - That is, for $y_i = +1$, when $h\left(w,x_i\right)$ increases, $\text{loss}_g\left(h\left(w,x_i\right), y_i\right)$ decreases monotonically.
  - A stronger correct prediction leads to lower loss.
- If y_i is -1, $\text{loss}_g\left(h\left(w,x_i\right), y_i\right)$ is a monotonically increasing function of $h\left(w,x_i\right)$.
  - That is, for $y_i = -1$, when $h\left(w,x_i\right)$ increases, $\text{loss}_g\left(h\left(w,x_i\right), y_i\right)$ increases monotonically.
  - A stronger incorrect prediction leads to higher loss.

Behavior of the smooth loss function

$$\min_w \; f(w) = \frac{1}{m}\sum_{i=1}^{m} \text{loss}_g(h(w,x_i), y_i)$$

- If y_i is $+1$, $\text{loss}_g\left(h\left(w,x_i\right), y_i\right)$ is a monotonically decreasing function of $h\left(w,x_i\right)$.
  - That is, for $y_i = +1$, when $h\left(w,x_i\right)$ increases, $\text{loss}_g\left(h\left(w,x_i\right), y_i\right)$ decreases monotonically.
  - $h(w,x)$ should be a **concave** function in w for $f(w)$ to be convex.
- If y_i is -1, $\text{loss}_g\left(h\left(w,x_i\right), y_i\right)$ is a monotonically increasing function of $h\left(w,x_i\right)$.
  - That is, for $y_i = -1$, when $h\left(w,x_i\right)$ increases, $\text{loss}_g\left(h\left(w,x_i\right), y_i\right)$ increases monotonically.
  - $h(w,x)$ should be a **convex** function in w for $f(w)$ to be convex.

Logistic loss and linear predictors

Conclusion: $h(w,x)$ has to be **linear** in w!

$$\min_{w,b} \; f(w) = \frac{1}{m}\sum_{i=1}^{m} \text{loss}_g(h(x_i), y_i) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + e^{-y_i(w^T x_i + b)})$$

Different mapping $\phi(x_i)$

$$\mathcal{H}: \; x \mapsto w^T \phi(x)$$

or

$$\mathcal{H} = \left\{ h_w(\cdot) \mid h_w(x) = w^T \phi(x), \; x \in \mathbb{R}^D, \; w \in \mathbb{R}^D \right\}.$$

If $\mathcal{X} = \mathbb{R}^2$, $x \in \mathcal{X}$,
$\phi(\cdot) \; : \; \mathcal{X} \to \mathbb{R}^3$, so that now $D = 3$

$$\phi(x) = \begin{bmatrix} x[1] & x[2] & 1 \end{bmatrix}^T$$
$$w = \begin{bmatrix} w_1 & w_2 & b \end{bmatrix}^T$$

If $\mathcal{X} = \mathbb{R}^2$, $x \in \mathcal{X}$
$\phi_2(\cdot) \; : \; \mathcal{X} \to \mathbb{R}^6$, so that now $D = 6$

$$\phi_2(x) = \begin{bmatrix} x[1] & x[2] & (x[1])^2 & (x[2])^2 & x[1]x[2] & 1 \end{bmatrix}^T$$
$$w = \begin{bmatrix} w_1 & w_2 & w_3 & w_4 & w_5 & b \end{bmatrix}^T$$

Logistic loss and regularization

Consider the case when $\hat{w}^T x$ separates the data without error. What does this mean?

$$y_i(\hat{w}^T x_i) > 0, \quad \forall i = 1, \ldots m$$

The consider empirical risk minimizer

$$\min_w \; f(w) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + e^{-y_i(w^T x_i)})$$

Where is the minimum attained?

Logistic loss and regularization

Consider the case when $\hat{w}^T x$ separates the data without error. What does this mean?

$$y_i(\hat{w}^T x_i) > 0, \quad \forall i = 1, \ldots m$$

The consider empirical risk minimizer

$$\min_w \; f(w) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + e^{-y_i(w^T x_i)})$$

Where is the minimum attained? Consider $w = 100\hat{w}$... or $w = 1000000\hat{w}$.
Clearly as $w \to \infty$ $f(w) \to 0$. The means that the problem is not well defined. Also this means that we need to control $\|w\|$.

An Example

For example, $w^T x_i = 0.001$, then the logarithm term could be:

$$\log(1 + e^{-0.001}) \approx \log 2 = 0.301.$$

Without changing the hyperplane, we can multiple scalar on both side of $w^T x_i = 0.001$, say $\tilde{w}^T x_i = 1$, with $\tilde{w}^T = 1000 w^T$. Then

$$\log(1 + e^{-1}) \approx \log 1.36 = 0.134.$$

So we can see, we can minimizing the objective function simply by scaling.

An Example

Consider a extreme case, $\tilde{w}^T x_i = +\infty$. Then

$$\log(1 + e^{-\infty}) = 0.$$

note that 0 cannnot be actually attained.
As our goal is to minimize the total loss, so the optimal objective function value can be very small simply by scaling of the hypothesis.
Logistic loss is sensitive to SCALING

sensitive to Scaling.

Drawbacks

something like overfitting.

One may argue that as the sample problem is separable, it is OK to choose such 'big' w and b. However, sample data being separable doesn't necessarily implies that the whole set is separable.
Consider an example that some actual data violates the hypothesis. If $w^T x + b = 1000$, and there exists a $y_i < 0$ in actual data, then

$$\log(1 + e^{1000}) \approx +\infty,$$

which means the error of such logistic loss function with big 'w' and 'b' can be very huge.

Drawbacks

- There might be huge difference between expected error (R) and empirical error ($\hat{R}$).
- Sample separable doesn't imply actual data separable.

How can we modify logistic loss function to eliminate its drawbacks?

Logistic loss and regularization

Regularized logistic regression

$$\min_{w:\|w\| \le B} f(w) = \frac{1}{m} \sum_{i=1}^{m} \log(1 + e^{-y_i (w^T x_i)})$$

Or

$$\min_{w} f(w) = \frac{1}{m} \sum_{i=1}^{m} \log(1 + e^{-y_i (w^T x_i)}) + \lambda \|w\|^2$$

Large Margin Classification

logistic function

SVM.

Assume that w is normalized so that $\|w\|_2 = 1$.
We do not only want to separate points, but we want to separate them **with a margin**:

$$w^T x_i + b \ge \gamma \text{ if } y_i = 1$$
$$w^T x_i + b \le -\gamma \text{ if } y_i = -1$$

Alternately, we can express both of the statements above as the single requirement

$$y_i \left(w^T x_i + b \right) \ge \gamma \quad i = 1, \ldots, m$$

Large Margin Classification

We want the largest margin (why?)

$$\begin{aligned} &\underset{w,b,\gamma}{\text{maximize}} \ \gamma \\ &\text{subject to} \ \ y_i\left(w^T x_i + b\right) \geq \gamma \quad i = 1, \ldots, m \\ &\|w\|_2 = 1 \end{aligned}$$

Note that as stated, the problem above is not convex. Specifically, because the equality constraint $\|w\|_2 = 1$ is not affine.

Large Margin Classification

We will show that this can be rewritten as

$$\begin{aligned} &\underset{\tilde{w},\tilde{b}}{\text{minimize}} \ \|\tilde{w}\|_2 \\ &\text{subject to} \ \ y_i\left(\tilde{w}^T x_i + \tilde{b}\right) \geq 1 \quad i = 1, \ldots, m \end{aligned}$$

- This is a convex problem.
- Note that there is no loss function, because we assume that we achieve zero loss.
- Moreover we use a different loss function - a margin loss and assume it is equal to zero.
- We will generalize this later.

margin loss

Rademacher Complexity

Given a hypotheses set $\mathcal{H}$, The Rademacher complexity of the function class $\mathcal{H}$ for sample size m is:

$$\mathcal{RC}_m(\mathcal{H}) = \mathbb{E}_x \mathbb{E}_{\sigma \sim unif\{\pm 1\}^n}\left[\sup_{h \in \mathcal{H}} \frac{1}{m}\left|\sum_{i=1}^{m} \sigma_i h(x_i)\right|\right].$$

Some features:

- If $\|w\| \to \infty$, then $\mathcal{RC}_m(\mathcal{H}) \to \infty$.
- Sensitive to scaling.
- Connected with VC-dimension.
- $\mathbb{E}_x$ can be replaced with the worst case to bound $\mathcal{R}_m(\mathcal{H})$.

Rademacher Complexity

Recall $\mathcal{R}(h) = \mathbb{E}(loss(h(x), y))$, $\hat{\mathcal{R}}(h) = \frac{1}{m}\sum loss(h(x_i), y_i)$. And consider $\mathcal{H}_B = \{w : x \to w^T x | \|w\|_2 \leq B\}$, so we have,

$$\mathcal{RC}(\hat{h}) \leq \mathcal{R}(h^*) + \tilde{\mathcal{O}}[L \cdot \mathcal{RC}_m^2(\mathcal{H}) + \sqrt{L \cdot \mathcal{R}(h^*)}\mathcal{RC}_m(\mathcal{H})],$$

where L is the Lipschitz constant of loss function.
We can bound Rademacher complexity

$$\mathcal{RC}_m(\mathcal{H}_B) \leq XB\sqrt{\frac{2}{m}},$$

where $X : \{\|x\|_2 \leq X, \forall x \in \mathcal{X}\}$ and $B : \{\|w\|_2 \leq B, \forall w \in \mathcal{H}_B\|\}$

γ-fat shattering

Now instead of VC-dimension (aka shattering), we will have a "fat-shattering" dimension.

Definition

The points $x_1, \ldots, x_m$ are γ-fat shattered by the hypothesis class $\mathcal{H} \in \mathbb{R}^{\mathcal{X}}$ if for all possible labelings $y_1, \ldots, y_m$, there exists a predictor $h \in \mathcal{H}$ such that every labeling is possible with a margin of γ. That is $y_i\left(h(x_i)\right) \geq \gamma$.

Definition

The γ-fat shattering (or "fat shattering at scale γ") dimension of a hypothesis class $\mathcal{H} \in \mathbb{R}^{\mathcal{X}}$ is the largest number of points m such that there exists some set of points $x_1, \ldots, x_m$ that are γ-fat shattered.

Example of γ-fat shattering for the case of a restricted classifier set.

$$\begin{aligned} \mathcal{H}_{w;B} &= \left\{x \mapsto w^T x \mid \|w\|_2 \leq B\right\} \\ \mathcal{X} &= \left\{x \mid \|x\|_2 \leq X\right\}. \end{aligned}$$

For this setting, the γ-fat shattering dimension $D_\gamma \propto \frac{B^2 X^2}{\gamma^2}$. Note that these parameters (B, X, γ) must scale together.

- Can 1 point be γ-fat shattered with $\gamma = 0.9$? Yes.
- Can 2 points be γ-fat shattered with $\gamma = 0.9$?
- Can 3 points be γ-fat shattered with $\gamma = 0.9$?

Width of the margin over the radius of the data, geometrically $\frac{BX}{\gamma}$. The ratio is what matters.
The relative margin is $\frac{\gamma}{BX}$.

Deviation bounds involving the fat-shattering dimension

For an L-Lipschitz loss function, the γ-fat shattering dimension controls the capacity/complexity.

$$\mathcal{H}_{w;B} = \left\{ x \mapsto w^T x \mid \|w\|_2 \leq B \right\}$$
$$\mathcal{X} = \{ x \mid \|x\|_2 \leq X \}.$$

Lemma

For any loss that is L-Lipschitz, with probability at least $1 - \delta$

$$\left| R(h) - \hat{R}(h) \right| \leq 2 \cdot L \sqrt{\frac{B^2 X^2}{m}} + s_{max} \sqrt{\frac{\log \frac{2}{\delta}}{m}}.$$

Regularized empirical risk minimization

$$\mathcal{H}_{w;B} = \left\{ x \mapsto w^T x \mid \|w\|_2 \leq B \right\}$$
$$\mathcal{X} = \{ x \mid \|x\|_2 \leq X \}.$$

For any loss that is L-Lipschitz,

$$R(h) \leq \hat{R}(h) + 2 \cdot L \sqrt{\frac{B^2 X^2}{m}} + s_{\max} \sqrt{\frac{\log \frac{2}{\delta}}{m}}.$$

R, δ and L are constants beyond our control (more or less). Assume m is also fixed. But B can be controlled

$$R\left(\hat{h}\right) \leq \min_{\|w\| \leq B} \left[\hat{R}(h) + 2 \cdot L \sqrt{\frac{B^2 X^2}{m}} + s_{\max} \sqrt{\frac{\log \frac{2}{\delta}}{m}} \right]$$

Optimization Methods In Machine Learning

Lecture 7-8: Support Vector Machines

Professor Katya Scheinberg

Lehigh University

Spring 2014

Outline

Outline

Review on Loss functions

Recall different type of loss functions:

- **0-1 loss function**
$$loss_{01}(h(x), y) = \begin{cases} 1 & \text{if } yh(x) < 0 \\ 0 & \text{if } yh(x) \geq 0. \end{cases}$$
- **Margin loss function**
$$loss_m(h(x), y) = \begin{cases} 1 & \text{if } yh(x) < 1 \\ 0 & \text{if } yh(x) \geq 1. \end{cases}$$
- **Logistic loss function**
$$loss_g(h(x), y) = log(1 + e^{-yh(x)}).$$

Hinge loss function

- The hinge loss function is used for "Maximum Margin Classification", most notably for "Support Vector Machines".
- For an intended output $y = \{+1, -1\}$ and a classifier $h(x)$, the hinge loss of the $h(x)$ is defined as:
$$loss_h(h(x), y) = \begin{cases} 0 & \text{if } yh(x) \geq 1 \\ 1 - yh(x) & \text{if } yh(x) < 1. \end{cases}$$

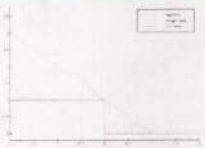


Figure: 0-1-loss upper bounded by log-loss and hinge-loss[1]

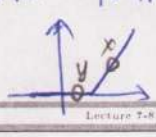
[1] Ben Taskar, Learning Structured Prediction Models:A Large Margin Approach. PhD Thesis,2004.

Properties of hinge loss function

- The hinge loss is a **convex** function, so many of the usual convex optimizers used in machine learning can work with it.
- Hinge loss function is **Lipschitz** with Lipschitz constant $L = 1$.
- Hinge loss is an upper bound on 0-1 loss.

$\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|, \forall x, y \in B$

for hinge loss function



$\Rightarrow L = 1.$

Rademacher Complexity

- Rademacher Complexity is a concept for a particular size of sample set.
- It is upper-bounded by $\sqrt{\frac{W^2X^2}{m}}$.

Where m is the sample size and X is the radius of the ball containing whole possible data (not just sample data), which is $X \geq \|\phi(x_i)\|$.

Note: We also want to keep W small!

Example

Recall logistic loss minimization:

$$\min \frac{1}{m}\sum_{i=1}^{m} loss_g(w^T\phi(x_i), y_i),$$
$$w: \ \|w\| \leq W.$$

- With $W = 10^6$ we will have:

$\hat{w}: \|\hat{w}\| = 10^6 \implies \hat{R}_g(\hat{w}) = 0.001 \implies R_g(w) \leq 0.001 + \sqrt{\frac{W^2X^2}{m}}$
Which means we need a sample size m as: $m = (100 \times 1000 \times 10^6)^2$.

- With $W = 100$ we will have:

$\hat{w}: \|\hat{w}\| = 100 \implies \hat{R}_g(\hat{w}) = 0.002 \implies R_g(w) \leq 0.002 + \sqrt{\frac{W^2X^2}{m}}$
Which means we need a sample size m as: $m = (100 \times 500 \times 100)^2$.

Outline

Linear Separators

- Minimizing following problem over w is the goal:

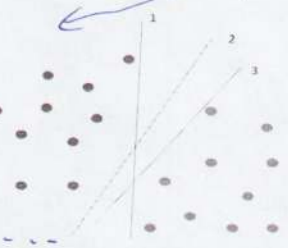
$$\min_w \sum_{i=1}^{m} loss_g(h(x_i), y_i) + \lambda\|w\|^2.$$

Note: The norm of w is not restricted.

- Which of the linear separators is optimal?

penalty for $|H|$.

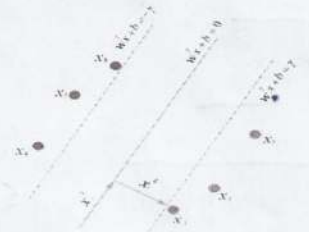
$\|w\| \leq 1 \quad H_0$
$\|w\| \leq 2 \quad H_1$
$\|w\| \leq 3. \quad H_2$
$\vdots$



$H_0 \subseteq H_1 \subseteq \cdots$

Classification Margin

- Vectors closest to the hyperplane are **Support Vectors**.
- **Margin** of the separator is the distance between support vectors.



Maximum Margin Classification

$$\max \gamma$$

$$w^Tx_1 + b \geq \gamma, \qquad w^Tx_4 + b \leq -\gamma,$$
$$w^Tx_2 + b \geq \gamma, \qquad w^Tx_5 + b \leq -\gamma,$$
$$w^Tx_3 + b \geq \gamma, \qquad w^Tx_6 + b \leq -\gamma,$$

$\gamma \leq w^Tx_1 + b = w^Tx_1^\parallel + w^Tx_1^\perp + b = \|w\|\|x_1^\parallel\|.$

- The margin depends on the $\|w\|$, but if we fix $\|w\| = 1$, we will have: $\gamma = \|w\|\|x_1^\parallel\|$.
- Maximizing the margin is equivalent to minimizing the norm of w.

Optimal Weight

$\mathcal{L}_p = \frac{1}{2}w^T w - \sum_i \alpha_i [y_i(w^T x_i + b) - 1]$

$\frac{\partial \mathcal{L}_p}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{N} \lambda_i y^{(i)} x^{(i)}$

$\frac{\partial \mathcal{L}_p}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N} \lambda^{(i)} y^{(i)} = 0.$

- The optimal solution is the following:

$$w^* = \arg\min_w \lambda\|w\|^2 + \sum_{i=1}^{m} loss_h(w^T\phi(x_i), y_i) = \arg\min f(w).$$

- **Representer Theorem:** The final w, always is the linear combination of $\phi(x_i)$, for $i = 1...m$:

$$w^* = \sum_{i=1}^{m} \alpha_i \phi(x_i).$$

$\mathcal{L}(w, b, \lambda) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{M} \lambda_i [y^{(i)}(w^T x^{(i)} + b) - 1]$

$= \frac{1}{2}w^T w - \sum_{i=1}^{M} \lambda_i y^{(i)} w^T x^{(i)} - \sum_{i=1}^{M} \lambda_i y^{(i)} b + \sum_{i=1}^{M} \lambda_i$

$= \frac{1}{2} w^T \sum_{i=1}^{M} \lambda_i y^{(i)} x^{(i)} - w^T \sum_{i=1}^{M} \lambda_i y^{(i)} x^{(i)} - \sum_{i=1}^{M} \lambda_i y^{(i)} b + \sum_{i=1}^{M} \lambda_i$

$= -\frac{1}{2} w^T \sum_{i=1}^{M} \lambda_i - b \sum_{i=1}^{M} \lambda_i y^{(i)} + \sum_{i=1}^{M} \lambda_i$

$= -\frac{1}{2} \left(\sum_{i=1}^{M} \lambda y^{(i)} x^{(i)}\right)^T \left(\sum_{i=1}^{M} \lambda_i y^{(i)} x^{(i)}\right) - b \sum_{i=1}^{M} \lambda_i y^{(i)} + \sum_{i=1}^{M} \lambda_i$

$= -\frac{1}{2} \sum_{i=1}^{M} \lambda y^{(i)} (x^{(i)})^T \sum_{i=1}^{M} \lambda_i y^{(i)} x^{(i)} - b \sum_{i=1}^{M} \lambda_i y^{(i)} + \sum_{i=1}^{M} \lambda_i$

$= -\frac{1}{2} \sum_{i=1, j=1}^{M} \lambda_i y^{(i)} (x^{(i)})^T \lambda_j y^{(j)} x^{(j)} - b \sum_{i=1}^{M} \lambda_i y^{(i)} + \sum_{i=1}^{M} \lambda_i$

$= \sum_{i=1}^{M} \lambda_i - \frac{1}{2} \sum_{i=1, j=1}^{M} \lambda_i \lambda_j y^{(i)} y^{(j)} (x^{(i)})^T x^{(j)}$

only $x^{(i)}$'s are vector.

Proof of "Representer Theorem":

Consider $w^* = w^\perp + w^\parallel$, where:

$$w^\parallel : \quad w^\parallel = \sum_{i=1}^{m} \alpha_i \phi(x_i),$$

$$w^\perp : \quad w^\perp . \phi(x_i) = 0, \quad \forall i = 1...m.$$

Based on definition ($\|w^*\|^2 = \|w^\parallel\|^2 + \|w^\perp\|^2$) we have $\|w^*\|^2 \geq \|w^\parallel\|^2$.

On the other hand:

$$w^{*T}\phi(x_i) = w^{\parallel T}\phi(x_i) + w^{\perp T}\phi(x_i) \implies w^{*T}\phi(x_i) = w^{\parallel T}\phi(x_i).$$

Which implies:

$$loss(w^{*T}\phi(x_i), y_i) = loss(w^{\parallel T}\phi(x_i), y_i).$$

$w^\parallel$ and w^* have the same loss, but the norm of $w^\parallel$ is less than the norm of w^*, which means w^* can not be the minimizer!

So, to have w^* as the minimizer, w^* should consists of just parallel part, which means $w^* = w^\parallel = \sum_{i=1}^{m} \alpha_i \phi(x_i)$.

Outline

Separable Case

- Mathematical formulation:

$$\min_{w,b} \|w\|^2,$$

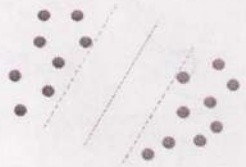
$$s.t \quad w^T x_i + b \geq 1, \quad if \quad y_i = 1,$$

$$w^T x_i + b \leq -1, \quad if \quad y_i = -1.$$

- Which is equivalent to:

$$\min_{w,b} \|w\|^2,$$

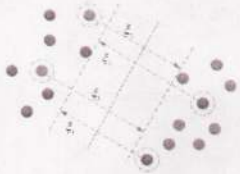
$$s.t \quad y_i(w^T x_i + b) \geq 1, \quad \forall i = 1...m.$$



Non-separable Case

- **What if the points are not linearly separable?**

Error parameters ξ_i can be added to allow misclassification of difficult or noisy examples.



- Mathematical model after constraint relaxation and adding penalty to objective function:

$$\min_{w,b,\xi} \|w\|^2 + C\sum_{i=1}^{m} \xi_i.$$

$$s.t \quad y_i(w^T x_i + b) \geq 1 - \xi_i, \quad \forall i = 1...m.$$

$$\xi_i \geq 0. \quad \forall i = 1...m.$$

Hinge Loss vs. Misclassification Errors

- For any w and b, there is different feasible value for ξ_i but we can optimize ξ_i separately for each sample point and make $\sum_{i=1}^{m} \xi_i$ minimum.

- The optimal value of ξ_i for any x_i and y_i is the following:

$$\xi_i = \max\{1 - y_i(w^T x_i + b), 0\}.$$

- This optimal value of ξ always is equal to to "hinge loss".

Solving Optimization Problem

- Lagrangian with multipliers α and u is the following:

$$L(w,b,\xi,\alpha,u) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i - \sum_{i=1}^{m}\alpha_i\left(y_i(w^Tx_i+b) - 1 + \xi_i\right) - u_i\xi_i.$$

- KKT conditions:

- Derivatives:

$$\nabla_w L(w,b,\xi,\alpha,u) = w - \sum_{i=1}^{m}\alpha_i y_i x_i = 0 \implies w = \sum_{i=1}^{m}\alpha_i y_i x_i$$

Note: This is exactly the result of "Representer Theorem".

$$\nabla_\xi L(w,b,\xi,\alpha,u) = C - \alpha_i - u_i = 0 \quad \forall i \implies u_i = C - \alpha_i \quad \forall i$$

$$\nabla_b L(w,b,\xi,\alpha,u) = \sum_{i=1}^{m}\alpha_i y_i = 0 \quad \forall i$$

$$\alpha_i \geq 0 \quad u_i \geq 0 \implies 0 \leq \alpha_i \leq C \quad \forall i$$

Solving Optimization Problem

- Complementary slackness for inequality constraints:

$$y_i(w^Tx_i+b) \geq 1-\xi_i \quad \longleftrightarrow \quad \alpha_i$$
$$\xi_i \geq 0 \quad \longleftrightarrow \quad C-\alpha_i$$

- $\xi_i = 0$ and $\alpha_i = 0 \implies y_i(w^Tx_i+b) \geq 1$: the point is classified up the margin.
- $\xi_i = 0$ and $\alpha_i > 0 \implies y_i(w^Tx_i+b) = 1$: the point is on the margin (it's support vector).
- $\xi_i > 0$ and $\alpha_i > 0 \implies y_i(w^Tx_i+b) \geq 1-\xi_i$: the point is misclassified and is up the margin.

Note1: Since $w = \sum_{i=1}^{m}\alpha_i y_i x_i$, we can omit points with zero α (which are on the "correct" side of the margin and not on the margin), and have exactly the same solution.

Note2: The points on the margin are support vectors and only these vectors matter; other training vectors are ignorable.

Equivalent representations

- By using the definition of $w = \sum_{i=1}^{m}\alpha_i y_i x_i$, we will have the following optimization problem:

$$\min_{\alpha,\xi} \sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j(y_jx_j)^T(y_ix_i) + C\sum_{i=1}^{m}\xi_i,$$
$$s.t \quad (\sum_{j=1}^{m}\alpha_j y_j x_j)^T y_i x_i \geq 1-\xi_i, \qquad \forall i = 1...m,$$
$$\xi_i \geq 0, \qquad \forall i = 1...m.$$

$Q\alpha \geq 1 - \xi$. α_i, ξ_i $2m$ variables.

$\xi \geq 0$.

larger / computational expensive

Equivalent representations

- By defining matrix $Q \subseteq \mathbb{R}^{m\times m}$ as $Q_{ij} = (y_iy_j)x_i^Tx_j$, we will have:

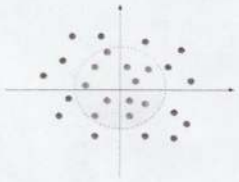
$$\min_{\alpha,b,\xi} \frac{1}{2}\alpha^TQ\alpha + C\sum_{i=1}^{m}\xi_i,$$
$$s.t \quad Q\alpha + y_ib \geq 1-\xi_i, \qquad \forall i = 1...m,$$
$$\xi_i \geq 0, \qquad \forall i = 1...m.$$

- We can also solve the "Dual form" of this optimization problem:

$$\begin{cases} \min_{\alpha} \frac{1}{2}\alpha^TQ\alpha - e^T\alpha, \\ s.t \quad y^T\alpha = 0, \qquad \forall i = 1...m, \\ \quad 0 \leq \alpha \leq C, \qquad \forall i = 1...m. \end{cases}$$

Kernel Methods and Nonlinear classification

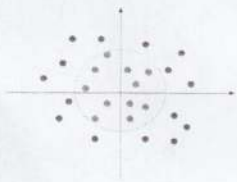
- We often need to deal with nonlinear pattern in data,
- Classes may not be separable by a linear boundary,



- In this case linear SVM is not powerful enough to separate classes accurately.

Kernel Methods and Nonlinear classification

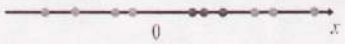
- We often need to deal with nonlinear pattern in data,
- Classes may not be separable by a linear boundary,



- In this case linear SVM is not powerful enough to separate classes accurately.
- Kernels make linear models work in nonlinear setting by mapping data to higher dimensions (via changing the feature representation).
- Linear model can be applied in new feature space.

Classifying Non-Linearly Separable Data via Kernel Trick(Example 1)

- Consider a binary classification problem [2] such that each example is represented by a single feature x.

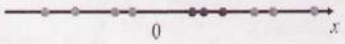


- No linear boundary can separate these data points.

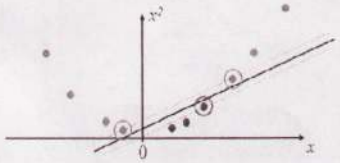
[2] http://nlp.stanford.edu/IR-book/html/htmledition/nonlinear-svms-1.html

Classifying Non-Linearly Separable Data via Kernel Trick(Example 1)

- Consider a binary classification problem [2] such that each example is represented by a single feature x.



- No linear boundary can separate these data points,
- By mapping $x \longrightarrow \{x, x^2\}$, each point has two features x and x^2.



- Data points became linearly separable in the new feature space.

[2] http://nlp.stanford.edu/IR-book/html/htmledition/nonlinear-svms-1.html

Classifying Non-Linearly Separable Data via Kernel Trick(Example 2)

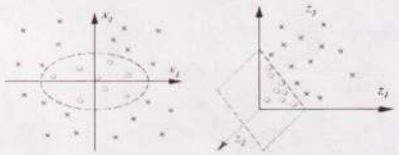
- Consider another problem such that each example is defined by two features $\{x_1, x_2\}$.

- No linear separator can classify these data points.
- Now by defining new feature space $\{x_1, x_2\} \longrightarrow \{x_1^2, \sqrt{2}x_1x_2, x_2^2\}$, each point has three features.

Classifying Non-Linearly Separable Data via Kernel Trick(Example 2)

- Consider another problem such that each example is defined by two features $\{x_1, x_2\}$.

$$\Phi : R^2 \rightarrow R^3$$
$$(x_1, x_2) \mapsto (z_1, z_2, z_3) := (x_1^2, \sqrt{(2)}x_1x_2, x_2^2)$$



- No linear separator can classify these data points.
- Now by defining new feature space $\{x_1, x_2\} \longrightarrow \{x_1^2, \sqrt{2}x_1x_2, x_2^2\}$, each point has three features and points are linearly separable in this new feature space. [3]

[3] http://courses.cs.ut.ee/2011/graphmining/Main/KernelMethodsForGraphs

Feature Mapping

- Consider following **quadratic mapping** for a $d-$dimensional point $x = \{x_1, x_2, ..., x_d\}$, such that each new feature uses a pair of original feature.

$$\phi : x \longrightarrow \{x_1^2, x_2^2, ..., x_d^2, x_1x_2, x_1x_3, ..., x_1x_d, ..., x_{d-1}x_d\}.$$

Feature Mapping

- Consider following **quadratic mapping** for a $d-$dimensional point $x = \{x_1, x_2, ..., x_d\}$, such that each new feature uses a pair of original feature.

$$\phi : x \longrightarrow \{x_1^2, x_2^2, ..., x_d^2, x_1x_2, x_1x_3, ..., x_1x_d, ..., x_{d-1}x_d\}.$$

- In general computing mapping can be expensive and inefficient,
- On the other hand since after this mapping the number of features blow up, using the mapped representation may be inefficient too!
- By using Kernel Trick we can avoid both of these issues, since the mapping does not have to be explicitly computed, and working with new feature space remain efficient.

Kernels as Highe Dimensional Feature Space

- Consider two examples $x = \{x_1, x_2\}$ and $z = \{z_1, z_2\}$,
- Assume we have the kernel function k as $k(x, z) = (x^T z)^2$, then we have

$$\begin{aligned} k(x,z) &= (x^T z)^2 \\ &= (x_1 z_1 + x_2 z_2)^2 \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 \\ &= (x_1^2, \sqrt{2} x_1 x_2, x_2^2)^T (z_1^2, \sqrt{2} z_1 z_2, z_2^2) \\ &= \phi(x)^T \phi(z). \end{aligned}$$

Kernels as Highe Dimensional Feature Space

- Consider two examples $x = \{x_1, x_2\}$ and $z = \{z_1, z_2\}$,
- Assume we have the kernel function k as $k(x, z) = (x^T z)^2$, then we have

$$\begin{aligned} k(x,z) &= (x^T z)^2 \\ &= (x_1 z_1 + x_2 z_2)^2 \\ &= x_1^2 z_1^2 + x_2^2 z_2^2 + 2 x_1 x_2 z_1 z_2 \\ &= (x_1^2, \sqrt{2} x_1 x_2, x_2^2)^T (z_1^2, \sqrt{2} z_1 z_2, z_2^2) \\ &= \phi(x)^T \phi(z). \end{aligned}$$

- The kernel function k, implicitly defines a mapping ϕ to a higher dimensional space as

$$\phi(x) = \{x_1^2, \sqrt{2} x_1 x_2, x_2^2\}.$$

Kernels as Highe Dimensional Feature Space

- We **do not have to** define or compute this mapping,
- Defining kernel function $k(x, z)$ in a certain way, produce the higher dimensional mapping ϕ,
- Note that the kernel function $k(x, z)$ computes the dot product $\phi(x)^T \phi(z)$, (so we don't need to do this expensive computation explicitly).
- All kernel functions have these properties.

The Kernel Matrix

- Kernel function k defines the kernel matrix K over the data,
- Given m examples $\{x_1, ..., x_m\}$, the $(i, j)-$th entry of matrix K is defined as:

$$K_{ij} = k(x_i, x_j) = \phi(x_i)^T \phi(x_j),$$

- Matrix K is a symmetric matrix,
- Matrix K is a positive definite matrix, except for a few exceptions

Kernel's Examples

The following are the most popular kernels,

- Linear Kernel:

$$k(x, z) = x^T z$$

- Quadratic Kernel:

$$k(x, z) = (x^T z)^2 \quad \text{or} \quad (1 + x^T z)^2$$

- Polynomial Kernel of degree d:

$$k(x, z) = (x^T z)^d \quad \text{or} \quad (1 + x^T z)^d$$

- Radial Basis Function (RBF) Kernel:

$$k(x, z) = \exp(-\gamma \|x - z\|^2).$$

- Kernel hyperparametres (e.g. d, γ) chosen via cross-validation.

Kernelized SVM Training

- By using $\phi(x)$ instead of x, the first representation(primal model) of the model will change to the following:

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{m} \xi_i, \\ &s.t \quad y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i, \qquad \forall i = 1...m, \\ &\xi_i \geq 0. \qquad\qquad\qquad\quad \forall i = 1...m. \end{aligned}$$

- The change in second representation(dual problem) will be in substituting dot product $x_i^T x_j$ by.

$$K_{ij} = k(x_i, x_j) = \phi(x_i)^T \phi(x_j).$$

Note: Kernelized SVM learns a linear separator in new feature space which corresponds to a non-linear separator in original feature space.

Refereces

You can find more detail in following materials

- https://www.cs.utah.edu/~piyush/teaching/15-9-slides.pdf

Optimization Methods in Machine Learning
Lecure 9: Unconstrained optimization, logistic regression

Katya Scheinberg

Lehigh University

Spring 2016

Motivation

$$\min f(w) = \frac{1}{m}\sum \log(1+e^{-y_i x_i^T w}) + \lambda \|w\|_2^2 .$$

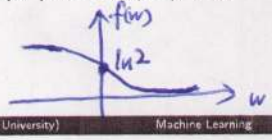
- Recall that we have the optimization problem

$$\min f(w) = \frac{1}{m}\sum_i \log(1+e^{-y_i x_i^T w}) + \lambda \|w\|_2^2 .$$

- We want to have an iterative method to approach the solution. For the optimal solution

$$w^* = \arg\min f(w)$$

we want to make steps $\{w^k\}$ so that as $k \to \infty$, $w^k \to w^*$, $f(w^k) \to f(w^*)$, and $\nabla f(w^k) \to 0$.



Gradient Descent Method

$$\min f(x)$$

Assume that $\nabla f(x)$ and possibly $\nabla^2 f(x)$ exist

- We introduce the gradient descent method as a common iterative method for solving optimization problems of this type.
- A typical iteration of this method will take the form:

$$x^{k+1} = x^k + t^k d^k$$

where t^k is the step size, and d^k is the search direction for the k^{th} step.

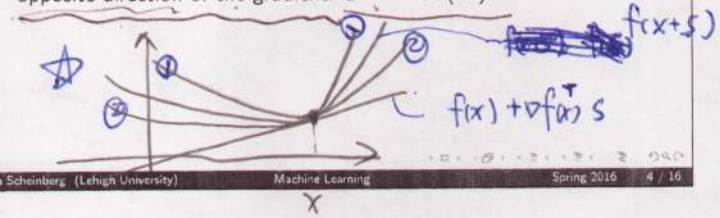
- Assume that $(d^k)^T \nabla f(x^k) < 0$.

Gradient Descent Method

- The Gradient Descent Method implements a conditions to ensure convergence:
  - Sufficient Decrease Condition:

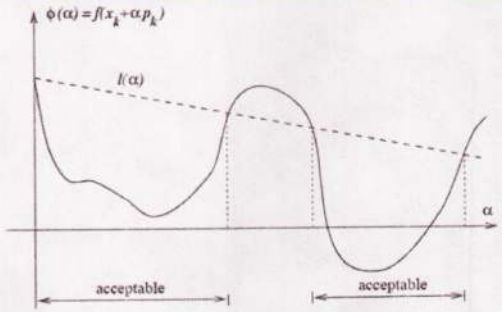
$$f(x^k + t^k d^k) \le f(x) + \alpha t^k \nabla f(x)^T d^k$$

- For Gradient Descent method, the search direction is often the opposite direction of the gradient: $d^k = -\nabla f(x^k)$.



Sufficient Decrease Condition

- The sufficient decrease condition ensures that each iteration will make sufficient progress in terms of reducing objective value.



Rate of Convergence

- Assume that $f(x)$ is Lipschitz smooth : $f(x+s) \le f(x) + \nabla f(x)^T s + \frac{L}{2}\|s\|^2$ (1)
- Assume that $f(x)$ is strongly convex: $f(x+s) \ge f(x) + \nabla f(x)^T s + \frac{\mu}{2}\|s\|^2$ (2).
- The above can be ensured if

$$\nabla^2 f(x) \succeq \mu I$$

for some $\mu > 0$.

- The rate of convergence for the gradient descent method with $d^k = \nabla f(x^k)$ is

$$f(x^k) - f(x^*) \le c^k (f(x^0) - f(x^*)), c \in (0,1) \quad (1)$$

where $C = \frac{\gamma-1}{\gamma+1}$, and $\gamma = \frac{L}{\mu}$ (the ratio of the largest value of Hessian and smallest value of Hessian).

- So if we want an accuracy of $\epsilon = 10^{-3}$, then $c^k = \frac{1}{1000}$. So $c^k \approx \epsilon$, $k \approx \log_c^{\epsilon}$.

Convergence of gradient descent

Slides from L. Vandenberghe http://www.ee.ucla.edu/ vandenbe/ee236c.html

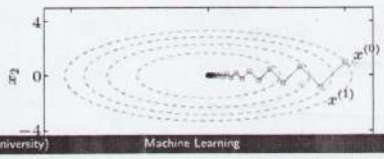
quadratic problem in $\mathbf{R}^2$

$$f(x) = (1/2)(x_1^2 + \gamma x_2^2) \qquad (\gamma > 0)$$

with exact line search, starting at $x^{(0)} = (\gamma, 1)$:

$$x_1^{(k)} = \gamma\left(\frac{\gamma-1}{\gamma+1}\right)^k, \qquad x_2^{(k)} = \left(-\frac{\gamma-1}{\gamma+1}\right)^k$$

- very slow if $\gamma \gg 1$ or $\gamma \ll 1$
- example for $\gamma = 10$:



Steepest descent Vs. Gradient descent

1. Normalized Steepest Descent

$$d_{nsd} = \arg\min\{\nabla f(x)^T v \mid \|v\| = 1\} \qquad (2)$$

and Gradient Descent

$$d^k = -\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|_2} \qquad (3)$$

What is the difference?

Steepest descent Vs. Gradient descent

1. Difference is the choice of norm.

If we use $\|\cdot\|_2$

$$d_{nsd} = -\frac{\nabla f(x^k)}{\|\nabla f(x^k)\|_2} \qquad (4)$$

If we use $\|\cdot\|_1$

$$d_{nsd} = \arg\max \left| \frac{\partial f(x)}{\partial x_i} \right| - \operatorname{sign}\frac{\partial f(x)}{\partial x_{i^*}} e_{i^*} \qquad (5)$$

If we use $\|v\|_M = v^T M v$ where $M \succeq 0$

$$d_{nsd} = -\frac{M^{-1}\nabla f(x^k)}{\|M^{-1}\nabla f(x^k)\|_2} \qquad (6)$$

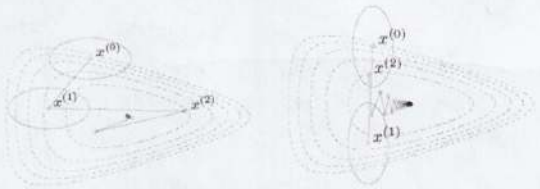
How to pick M?

$$d_{\text{Newton}} = -\nabla^2 f(x)^{-1}\nabla f(x) \qquad (7)$$

Convergence of gradient descent

Slides from L. Vandenberghe http://www.ee.ucla.edu/ vandenbe/ee236c.html

choice of norm for steepest descent



- steepest descent with backtracking line search for two quadratic norms
- ellipses show $\{x \mid \|x - x^{(k)}\|_P = 1\}$
- equivalent interpretation of steepest descent with quadratic norm $\|\cdot\|_P$: gradient descent after change of variables $\bar{x} = P^{1/2}x$

shows choice of P has strong effect on speed of convergence

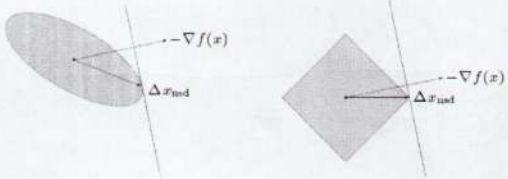
Convergence of gradient descent

Slides from L. Vandenberghe http://www.ee.ucla.edu/ vandenbe/ee236c.html

examples

- Euclidean norm: $\Delta x_{\text{sd}} = -\nabla f(x)$
- quadratic norm $\|x\|_P = (x^T P x)^{1/2}$ $(P \in \mathbf{S}^n_{++})$: $\Delta x_{\text{sd}} = -P^{-1}\nabla f(x)$
- ℓ_1-norm: $\Delta x_{\text{sd}} = -(\partial f(x)/\partial x_i)e_i$, where $|\partial f(x)/\partial x_i| = \|\nabla f(x)\|_\infty$

unit balls and normalized steepest descent directions for a quadratic norm and the ℓ_1-norm:

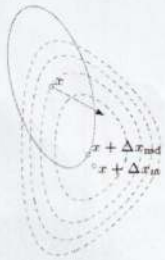


Convergence of gradient descent

Slides from L. Vandenberghe http://www.ee.ucla.edu/ vandenbe/ee236c.html

- Δx_{nt} is steepest descent direction at x in local Hessian norm

$$\|u\|_{\nabla^2 f(x)} = \left(u^T \nabla^2 f(x) u\right)^{1/2}$$



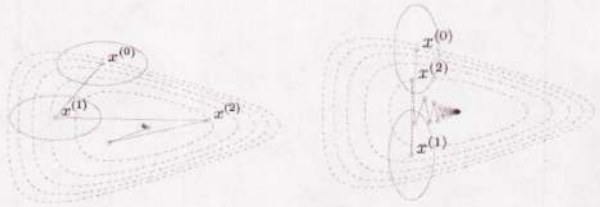
dashed lines are contour lines of f; ellipse is $\{x + v \mid v^T \nabla^2 f(x) v = 1\}$

arrow shows $-\nabla f(x)$

Convergence of gradient descent

Slides from L. Vandenberghe http://www.ee.ucla.edu/ vandenbe/ee236c.html

choice of norm for steepest descent



- steepest descent with backtracking line search for two quadratic norms
- ellipses show $\{x \mid \|x - x^{(k)}\|_P = 1\}$
- equivalent interpretation of steepest descent with quadratic norm $\|\cdot\|_P$: gradient descent after change of variables $\bar{x} = P^{1/2}x$

Convergence of gradient descent

Slides from L. Vandenberghe http://www.ee.ucla.edu/ vandenbe/ee236c.html

Newton step

$$\Delta x_{\rm nt} = -\nabla^2 f(x)^{-1}\nabla f(x)$$

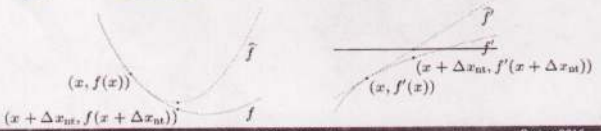
interpretations

- $x + \Delta x_{\rm nt}$ minimizes second order approximation

$$\widehat{f}(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2}v^T\nabla^2 f(x)v$$

- $x + \Delta x_{\rm nt}$ solves linearized optimality condition

$$\nabla f(x+v) \approx \nabla \widehat{f}(x+v) = \nabla f(x) + \nabla^2 f(x)v = 0$$

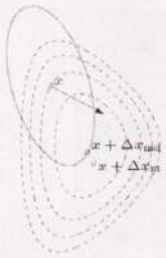


Convergence of gradient descent

Slides from L. Vandenberghe http://www.ee.ucla.edu/ vandenbe/ee236c.html

- $\Delta x_{\rm nt}$ is steepest descent direction at x in local Hessian norm

$$\|u\|_{\nabla^2 f(x)} = \left(u^T\nabla^2 f(x)u\right)^{1/2}$$



dashed lines are contour lines of f; ellipse is $\{x + v \mid v^T\nabla^2 f(x)v = 1\}$

arrow shows $-\nabla f(x)$

Optimization

We come back to the optimization problem

$$\min f(w) = \frac{1}{m}\sum \log(1 + e^{-y_i x_i^T w}) + \lambda\|w\|_2^2 \tag{8}$$

And we derive the gradient

$$\nabla f(w) = \frac{1}{m}\sum \frac{1}{e^{+y_i x_i^T w} + 1}(-y_i x_i) + \lambda w \tag{9}$$

And the Hessian

$$\nabla^2 f(w) = \frac{1}{m}\sum \frac{e^{y_i x_i^T w}}{(1 + e^{y_i x_i^T w})^2}(y_i x_i)(y_i x_i)^T + \lambda I \tag{10}$$

Notice that the time complexity for computing the Hessian is $d^2 m$, size of Hessian is $d \times d$, the complexity for computing the inverse of Hessian is d^3. So computation can be very expensive for large Hessian!

Optimization Methods in Machine Learning
Lecture 10: Newton method and Interior Point Method for SVM

Katya Scheinberg

Lehigh University

Spring 2016

Newton method

Slides from L. Vandenberghe http://www.ee.ucla.edu/ vandenbe/ee236c.html

Newton step (= Hessian Matrix)

$$\Delta x_{\rm nt} = -\nabla^2 f(x)^{-1}\nabla f(x)$$

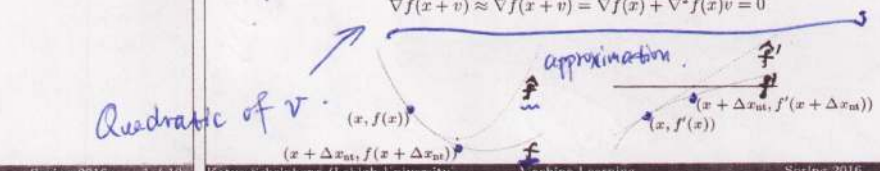
interpretations

- $x + \Delta x_{\rm nt}$ minimizes second order approximation

$$\hat f(x+v) = f(x) + \nabla f(x)^T v + \frac{1}{2} v^T \nabla^2 f(x) v$$

- $x + \Delta x_{\rm nt}$ solves linearized optimality condition

$$\nabla f(x+v) \approx \nabla \hat f(x+v) = \nabla f(x) + \nabla^2 f(x) v = 0$$



Quadratic of v.

objective.

Quadratic Approximation Model

- The problem we deal with is $f(x)$
- The quadratic approximation model is

$$q(x) = f(x_k) + \nabla f(x_k)^T (x - x_k) + \frac{1}{2}(x - x_k)^T \nabla^2 f(x_k)(x - x_k)$$

- Newton Step. $\Delta x_k = -\nabla^2 f(x_k)^{-1}\nabla f(x_k)$
- $x_{k+1} = x_k + \Delta x_k$
- Damped Newton step: not the full step.
- If full step is not good, use line search.

Convergence Analysis

- Assumption:

$$\|\nabla^2 f(x) + \nabla^2 f(y)\|_2 \le L\|y - x\|_2$$

- f is strongly convex with constant m
- We want Hessian to be Lipschitz continuous.
- There exists constants $\eta \in (0, m^2/L)$ and $\gamma > 0$ such that
  - if $\|\nabla f(x^k)\|_2 \ge \eta$, then

$$f(x^{k+1}) - f(x^k) \le -\gamma$$

  - if $\|\nabla f(x^k)\|_2 < \eta$, then

$$\frac{L}{2m^2}\|\nabla f(x^{k+1})\|_2 \le \left(\frac{L}{2m^2}\|\nabla f(x^k)\|_2\right)^2 \quad \text{(fast converge stage)}$$

where m is the smallest eigenvalue of $\nabla^2 f(x)$ for all x.

$-\mu \ln x$

Small μ.

large μ.

for each update, use a smaller μ, otherwise, it doesn't move (in Newton's Method)

Self-concordant

- Newton method is invariant under linear transformation! But the convergence analysis isn't!!
- To have a better analysis, self-concordant functions have been introduced.
- In optimization, a self-concordant function is a function $f : \mathbb{R} \to \mathbb{R}$ such that

$$|f'''(x)| \le 2 f''(x)^{\frac{3}{2}}$$

- $f : \mathbb{R}^n \to \mathbb{R}$ is self-concordant if

$$g(t) = f(x + tv)$$

is self-concordant for all $x \in \mathrm{dom} f, v \in \mathbb{R}^n$

- Examples: linear, convex quadratic, logarithm.

Interior Point Method

- Rewrite the quadratic model

$$\begin{array}{ll} \min & \frac{1}{2}x^T Q x + c^T x \\ & Ax = b \\ & x \ge 0 \end{array} \quad \Rightarrow \quad \begin{array}{ll} \min & \frac{1}{2}x^T Q x + c^T x - \mu \sum_{i=1}^n \ln x_i \\ & Ax = b \end{array}$$

Barrier function.

- KKT conditions are: $\mathcal{L} = \frac{1}{2}x^T Qx + c^T x - \mu \sum_{i=1}^n \log x_i - y(Ax - b)$

wrt y derivative to 0 $\rightarrow Ax = b$

wrt x derivative to 0. $\rightarrow -Qx + A^T y + s = c$

$Xs = \mu e$

$X, s > 0$

$S = X^{-1}\mu e$.

where $X = \mathrm{diag}(x)$ $\begin{bmatrix} x_1 & & \\ & x_2 & \\ & & \ddots \end{bmatrix}$

Interior Point Method

- Given (x, y, s), find the Newton step $(\Delta x, \Delta y, \Delta s)$,

$$A(x + \Delta x) = b$$
$$-Q(x + \Delta x) + A^T(y + \Delta y) + s + \Delta s = c$$
$$s\Delta X + X\Delta s + Xs = \mu e$$

- Then we have

$$S\Delta x + X\Delta s = \mu e - Xs \quad (1)$$
$$A\Delta x = (b - Ax) = r_p \quad (2)$$
$$-Q\Delta x + A^T\Delta y + \Delta s = (c - Qx - A^T y - s) = r_d \quad (3)$$

Interior Point Method

- Augmented system

$$A\Delta x = r_p$$
$$A^T\Delta y - (X^{-1}S + Q)\Delta x = r_d - X^{-1}(\mu e - Xs)$$

- Eventually, we have the following *Normal Equation*

$$A(X^{-1}S + Q)^{-1}A^T\Delta y = r$$

- Important: $A(X^{-1}S + Q)^{-1}A^T$ is positive definite if A is full row rank.

Optimality conditions for SVM

Consider dual form of SVM as following:

$$\min_{\alpha} \ \frac{1}{2}\alpha^T Q\alpha - e^T\alpha$$
$$s.t.$$
$$y^T\alpha = 0$$
$$0 \le \alpha \le c$$

Now consider KKT conditions:

$$\alpha_i s_i = 0 \qquad i = 1, 2, ..., n$$
$$(c - \alpha_i)\xi_i = 0 \qquad i = 1, 2, ..., n$$
$$y^T\alpha = 0$$
$$-Q\alpha + y\beta + s - \xi = -e$$
$$0 \le \alpha \le c$$
$$s \ge 0, \xi \ge 0$$

Relaxed KKT conditions

We want to solve this problem by interior point method, so we rewrite the KKT conditions as following:

$$\alpha_i s_i = \mu \qquad i = 1, 2, ..., n$$
$$(c - \alpha_i)\xi_i = \mu \qquad i = 1, 2, ..., n$$
$$y^T\alpha = 0$$
$$-Q\alpha + y\beta + s - \xi = -e$$
$$0 < \alpha < c$$
$$s > 0$$
$$\xi > 0$$

A Newton step of IPM

- Let $\mathcal{A} = diag(\alpha)$, $\mathcal{S} = diag(s)$ and $\Xi = diag(\xi)$

$$\begin{pmatrix} y^T & 0 \\ -(Q + \mathcal{A}^{-1}\mathcal{S} + (C - \mathcal{A})^{-1}\Xi) & y \end{pmatrix} \begin{pmatrix} \Delta\alpha \\ \Delta\beta \end{pmatrix} = \begin{pmatrix} -y^T\alpha \\ -e + Q\alpha - y\beta - \mathcal{A}^{-1}\mu e + (C - \mathcal{A})^{-1}\mu e \end{pmatrix}$$

- Doing some algebra, we have:

$$y^T(Q + D)^{-1}\Delta\beta = \gamma$$

where

$$D = \mathcal{A}^{-1}\mathcal{S} + (C - \mathcal{A})^{-1}\Xi$$
$$\gamma = -y^T\alpha + y^T(Q + D)^{-1}(-e + Q\alpha - y\beta - \mathcal{A}^{-1}\mu e + (C - \mathcal{A})^{-1}\mu e)$$

Kernel operation

- As we defined before

$$Q_{i,j} = y_i y_j \mathcal{K}(x_i, x_j) \quad (1)$$

where $\mathcal{K}(x_i, x_j)$ is kernel operation of x_i and x_j.

- Some examples for kernel operation:

  Linear kernel: $\mathcal{K}(x_i, x_j) = x_i^T x_j$

  Quadratic kernel: $\mathcal{K}(x_i, x_j) = [a + b(x_i^T x_j)]^2$

  RBF kernel: $\mathcal{K}(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma})$

- Q is a Positive Semi-Definite (p.s.d) matrix.

Computation complexity

Back to SVM, consider Q.

- To solve the Newton system, we have to solve

$$y^T(Q+D)^{-1}\Delta\beta = \gamma$$

where

$D = \mathcal{A}^{-1}\mathcal{S} + (C-\mathcal{A})^{-1}\Xi$

$\gamma = -y^T\alpha + y^T(Q+D)^{-1}(-e + Q\alpha - y\beta - \mathcal{A}^{-1}\mu e + (C-\mathcal{A})^{-1}\mu e)$

- Q is typically a dense matrix.

$$Q = Y^T X X^T Y$$

where Y is a $n \times 1$ and X is a $n \times d$, so rank of Q is at most d.

- We need $\mathcal{O}(n^3)$ operations to invert $(Q+D)$.

Scherman-Morrison-Woodbury formula

- Let $Q = VV^T$.
- We can find $(Q+D)^{-1}$ as following

$$\begin{aligned}(Q+D)^{-1} &= (VV^T + D)^{-1} \\ &= D^{-1} - D^{-1}V(I + V^T D^{-1} V)^{-1} V^T D^{-1}\end{aligned}$$

which needs $\mathcal{O}(nd^2)$ operations and $\mathcal{O}(nd)$ storage amount.

Reminder: Matrix calculus, Symmetric matrix

Definition

Matrix $M_{n\times n}$ is symmetric, if and only if $M = M^T$.

- Consider $Q_{n\times n}$ where n is size of training data. We can define Q as following

$$Q = Y^T X X^T Y$$

- Q is symmetric because:

$$Q^T = (Y^T X X^T Y)^T = Y^T X X^T Y = Q$$

- If a matrix is symmetric, its eigen values are real number.

Positive semi-definite and positive definite

Definition

Symmetric matrix $M_{n\times n}$ is positive semi-definite if for all $z_{n\times 1}$ we have:

$$z^T M z \geq 0$$

M is positive definite if $z^T M z > 0$ for all $z_{n\times 1}$.

- Q is positive semi-definite, because

$$z^T Q z = Y^T X X^T Y z = (X^T Y z)^T (X^T Y z) = \|X^T Y z\|^2 \geq 0$$

Definition

Symmetric matrix $M_{n\times n}$ is positive semi-definite if all of its eigenvalues are nonnegative.

Eigenvalue decomposition

Definition

Consider symmetric matrix $M_{n\times n}$. We have:

$$M = P\Lambda P^T$$

where Λ is a diagonal matrix that elements on its main diagonal corresponds to eigenvalues of M, and P is an orthogonal matrix. Columns of P correspond to eigenvectors of M.

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & 0 & \dots & 0 & 0 \\ 0 & \lambda_2 & 0 & \dots & 0 & 0 \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & \dots & 0 & \lambda_n \end{bmatrix}$$

$$P = \begin{bmatrix} P_1 \mid & P_2 \mid & \dots & \mid P_n \end{bmatrix}$$

Eigenvalue decomposition (continued)

Definition

Rank of a matrix is the number of linear independent columns or linear independent rows of the matrix.

- The rank of p.s.d. matrix is the number of positive eigen values.

$$Q = P\Lambda P^T = \sum_i \lambda_i P_i P_i^T$$

where $P_i P_i^T (i = 1, 2, \dots, n)$ are rank-one matrices.

Optimization Methods in Machine Learning
Lectures 11-12: Proximal and Accelerated Gradient Methods

Katya Scheinberg

Lehigh University

Spring 2016

Logistic Regression, Gradient and Hessian

We come back to the optimization problem

$$\min f(w) = \frac{1}{m}\sum log(1 + e^{-y_i x_i^T w}) + \frac{\lambda}{2}\|w\|_2^2 \quad (1)$$

And we derive the gradient

$$\nabla f(w) = \frac{1}{m}\sum \frac{1}{e^{y_i x_i^T w} + 1}(-y_i x_i) + \lambda w \quad (2)$$

And the Hessian

$$\nabla^2 f(w) = \frac{1}{m}\sum \frac{e^{y_i x_i^T w}}{(1 + e^{y_i x_i^T w})^2}(y_i x_i)(y_i x_i)^T + \lambda I \quad (3)$$

Notice that the time complexity for computing the Hessian is $d^2 m$, size of Hessian is $d \times d$, the complexity for computing the inverse of Hessian is d^3. So computation can be very expensive for large Hessian!

Lipschitz continuity

Definition (Lipschitz Continuity)

Given an open set $B \subseteq \mathbb{R}^n$, we say that f has **Lipschitz continuous gradient** (Lipschitz smooth) on the open subset B if there exists a constant $L \geq 0$ such that

$$\|\nabla f(x) - \nabla f(y)\| \leq L\|x - y\|, \quad \forall x, y \in B.$$

L is called the **Lipschitz constant** of ∇f on B.

Example (10.1)

An example of functions with different Lipschitz constants for $\nabla f(x)$:

Figure: $L = +\infty$

Figure: $L = 4$

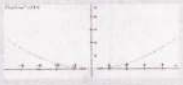


Figure: $L = 0.2$

Lipschitz smooth functions

- Recall that our goal is the minimization problem

$$\min_{x \in \mathbb{R}^n} f(x),$$

where the objective function f is convex and the gradient $\nabla f(x)$ has a Lipschitz constant L.

- By convexity we have the linear under-estimator $f(y) \geq f(x) + \nabla f(x)^T(y - x)$. Instead of using the linear model, we use a quadratic model as our subproblem:

$$Q(y) = f(x) + \nabla f(x)^T(y - x) + \frac{1}{2\mu}\|y - x\|^2,$$

where if $\mu \leq \frac{1}{L}$, then $Q(y) \geq f(y)$.

Over-approximation of a smooth function

- Linear lower approximation

$$f(y) \geq f(x) + \nabla f(x)^T(y - x)$$

- Quadratic upper-approximation

$$f(y) \leq f(x) + \nabla f(x)^T(y - x) + \frac{1}{2\mu}\|y - x\|^2 = Q(y), \quad \mu \leq \frac{1}{L}$$

- Rewrite

$$f(y) \leq f(x) + \frac{1}{2\mu}\|x - \mu\nabla f(x) - y\|^2 - \frac{1}{\mu^2}\|\nabla f(x^k)\|^2 = Q(y)$$

Minimizing $Q(y)$ gives $y = x - \mu\nabla f(x)$.

Basic Proximal Gradient Method

The proximal function of f is defined by:
$\mathbf{prox}_\mu(x^k) = \arg\min_x Q(x; x^k) = f(x^k) + \nabla f(x^k)^T(x - x^k) + \frac{1}{2\mu}\|x - x^k\|^2$.

Algorithm 1 Basic Proximal Gradient Method

Initialization: start with x^0.

for $k = 0$ to maximum iterations **do**

  repeat

    $z \leftarrow \mathbf{prox}_\mu(x^k - \mu\nabla f(x^k))$, and decrease μ,

  until $f(z) \leq Q(z; x^k)$.

  $x^{k+1} \leftarrow z$.

end for

Basic Proximal Gradient Method

Rate of Convergence and Complexity Analysis

Theorem

If $f(x^{i+1}) \le Q(x^{i+1}; x^i)$ for all $i = 1, \ldots, k$, then

$$f(x^k) - f(x^*) \le \frac{1}{2\mu k}\|x^0 - x^*\|^2$$

Chosing $\mu = 1/L$ ensures that $f(x^{i+1}) \le Q(x^{i+1}; x^i)$. By setting $\|x^0 - x^*\| = 1$ and assuming the precision to be ϵ, the computational complexity is estimated to be $k \approx \frac{1}{2\mu\epsilon} \sim \mathcal{O}(\frac{L}{\epsilon})$. Then when $\epsilon \approx 10^{-3}$, $k \approx 1000$ if $L \approx 1$. Compare this to $O(log(1/\epsilon))$.

Converge rate $\frac{\|x^0 - x^*\|}{\sum_{k} \mu_k}$ x2 ?

Proof

- From $f(x^{i+1}) \le Q(x^{i+1}; x^i)$, it can be shown (skipped here)

$$f(x^{i+1}) - f(x^*) \le \frac{1}{2\mu}(\|x^i - x^*\|^2 - \|x^{i+1} - x^*\|^2).$$

- By summing the above equations from $i = 0$ to $(k-1)$, and using $f(x^k) \le f(x^i)$, $\forall i \le k$

$$k(f(x^k) - f(x^*)) \le \sum_{i=0}^{k-1} f(x^{i+1}) - kf(x^*) \le \frac{1}{2\mu}(\|x^0 - x^*\|^2 - \|x^k - x^*\|$$
$$\le \frac{1}{2\mu}\|x^0 - x^*\|^2.$$

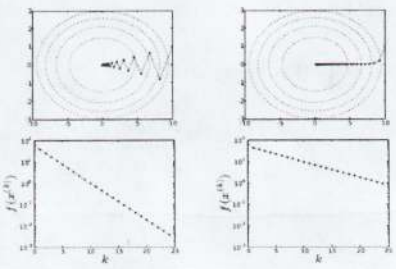
- Hence $f(x^k) - f(x^*) \le \frac{1}{2\mu k}\|x^0 - x^*\|^2$, which proves the algorithm convergence **sub-linearly**.

Convergence depending on choice of μ

Slides from L. Vandenberghe http://www.ee.ucla.edu/ vandenbe/ee236c.html

Quadratic example

$f(x_1, x_2) = (x_1^2 + Lx_2^2)/2$; left: $t_k = 1.8/L$; right: $t_k = 0.8/L$



Accelerated Gradient Method

Algorithm 2 Accelerated Generalized Gradient Method

Initialization: start with x^0 and $y^1 = x^0$.

for $k = 0$ to maximum iterations **do**

repeat

$z \leftarrow \mathbf{prox}_\mu(y^k - \mu\nabla f(y^k))$, and decrease μ,

until $f(z) \le Q(z; x^k)$.

$x^k \leftarrow z$,

$y^{k+1} \leftarrow x^k + \frac{k-1}{k+2}(x^k - x^{k-1})$.

end for

Don't use backtracking on AGD method
the theory breaks down

FISTA (fast iterative shrinkage-thresholding algorithm)

Algorithm 3 FISTA (fast iterative shrinkage-thresholding algorithm)

Initialization: start with x^0, $y^1 = x^0$ and $t^1 = 1$.

for $k = 0$ to maximum iterations **do**

repeat

$z \leftarrow \mathbf{prox}_\mu(y^k - \mu\nabla f(y^k))$, and decrease μ,

until $f(z) \le q(z; x^k)$.

$x^k \leftarrow z$,

$t^{k+1} = (1 + \sqrt{1 + 4(t^k)^2})/2$,

$y^{k+1} \leftarrow x^k + \frac{t^k - 1}{t^{k+1}}(x^k - x^{k-1})$.

end for

Complexity Result for Algorithms 2 & 3

- For Algorithms 2 & 3, the convergence result for smooth function is

$$|f(x^k) - f(x^*)| \le \frac{1}{2\mu k^2}\|x^0 - x^*\|^2,$$

If we pick $\mu = 1/L$ then $f(x^k) \le f(x^*) \le \epsilon$ when $k \ge \mathcal{O}(\sqrt{\frac{L}{\epsilon}})$.

For example, when $\epsilon \approx 10^{-3}$, and $L \approx 1$ then the complexity is approximately $\frac{1}{\sqrt{10^{-3}}} \approx 2^5 = 32$. Compare this to $\mathcal{O}(\frac{L}{\epsilon})$.

- The complexity of standard sub gradient method for non-smooth functions is $\mathcal{O}(\frac{1}{\epsilon^2})$.

For example, when $\epsilon \approx 10^{-3}$, the complexity is approximately $\frac{1}{(10^{-3})^2} \approx 10^6$.

- We can use accelerated algorithm to have an algorithm for non smooth functions with complexity $\mathcal{O}(\frac{1}{\epsilon})$.

Dealing with the non-smooth function

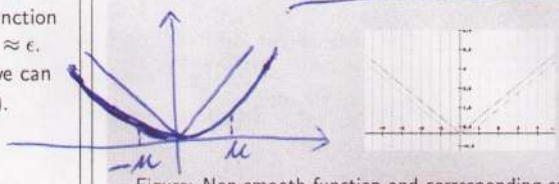
- Our strategy to reduce the computational complexity is to approximate the function with a smooth function.
- For a non-smooth function, consider a smooth approximation function with the Lipschitz constant of the gradient $L \approx \frac{1}{\epsilon}$. Then $\mu \sim \frac{1}{L} \approx \epsilon$. From the previous slide, $\frac{L}{2k^2} \sim \epsilon$, so $k \sim \frac{1}{\epsilon}$, which means that we can reduce the complexity for non-smooth case from $\mathcal{O}(\frac{1}{\epsilon^2})$ to $\mathcal{O}(\frac{1}{\epsilon})$.

Dealing with the non-smooth function

Example (10.2)

An example of approximation for non-smooth function $f(x) = \frac{1}{\mu}|x|$ is the smooth function:

$$\phi_\mu(x) = \begin{cases} \frac{x^2}{2\mu} & \text{if } |x| \le \mu \\ |x| - \frac{\mu}{2} & \text{if } |x| > \mu \end{cases}$$



use it to unconstrained SVM.

Figure: Non-smooth function and corresponding smooth function with $\mu = 1$

$$|x| - \mu \le \phi_\mu(x) \le |x|, \quad \phi''_\mu(x) = \frac{1}{\mu}$$

Unconstrained SVM

Given a training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, $x_i \in R^d$, $y \in \{+1, -1\}$

$$\min_w f(w) = \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_{i=1}^{n} \ell(w, (x_i, y_i))$$

where

$$\ell(w, (x, y)) = \max\{0, 1 - y(w^T x)\}$$

We want to find $f(w) \le f(w^*) + \epsilon$ - ϵ-optimal solution.

Smoothed SVM

Given a training set $S = \{(x_1, y_1), \ldots, (x_n, y_n)\}$, $x_i \in R^d$, $y \in \{+1, -1\}$

$$\min_w f_\mu(w) = \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_{i=1}^{n} \phi_\mu(w, (x_i, y_i))$$

where

$$\phi_\mu(w, (x, y)) = \begin{cases} 0 & y(w^T x) \ge 1 \\ \frac{(y(w^T x) - 1)^2}{2\mu} & 1 - \mu < y_i(w^T x) < 1 \\ 1 - y(w^T x) - \frac{\mu}{2} & y(w^T x) \le 1 - \mu \end{cases}$$

Find $f(w) \le f(w^*) + \epsilon$ - ϵ-optimal solution. Set $\mu = \epsilon/2$ and find $f_\mu(w) \le f_\mu(w^*) + \epsilon/2$ with an accelerated method. The L for $f_\mu(w)$ is $2/\epsilon$, hence we find the solution in $O(\sqrt{\frac{4}{\epsilon^2}})$ iterations.

First order methods for composite functions

Lecture 13.

Prox method with nonsmooth term

- Consider: $\min_x F(x) = f(x) + g(x)$

$$|\nabla f(x) - \nabla f(y)| \le L||x - y||$$

- Quadratic upper approximation

$$f(y)+g(y) \le f(x) + \nabla f(x)^\top (y - x) + \frac{1}{2\mu}||y - x||^2 + g(y) = Q_{f,\mu}(x, y)$$

$$F(y) \le f(x) + \frac{1}{2\mu}||x - \mu\nabla f(x)^\top - y||^2 - \frac{1}{\mu}\|\nabla f(x^k)\|^2 + g(y)$$

Assume that $g(y)$ is such that the above function is easy to optimize over y

Example 1 Sparse optimization

$$\min_x f(x) + ||x||_1$$

1-normal.

- Minimize upper approximation function $Q_{f,\mu}(x,y)$ on each iteration

$$\min_y Q_{f,\mu}(x, y) = \min_y f(x) + \frac{1}{2\mu}||x - \mu\nabla f(x)^\top - y||^2 + \|y\|_1$$

$$\Updownarrow$$

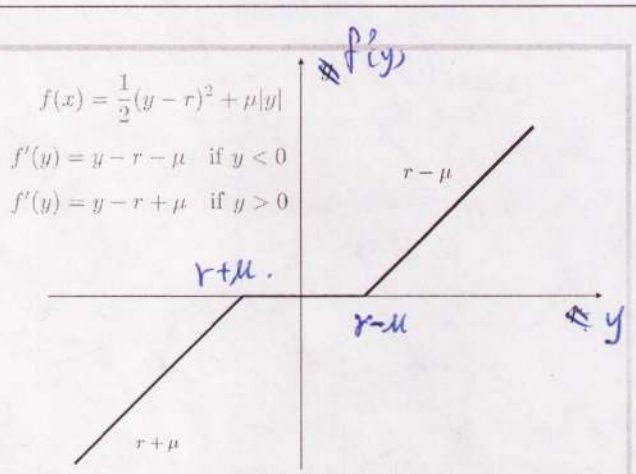
Shrinkage operator

$$\sum_i \min_{y_i} \left[\frac{1}{2\mu}(y_i - r_i)^2 + |y_i|\right]$$

$$\Updownarrow$$

Closed form solution! $O(n)$ effort

$$\min_{y_i} \frac{1}{2}(y_i - r_i)^2 + \mu|y_i| \to y_i^* = \begin{cases} r_i - \mu & \text{if } r_i > \mu \\ 0 & \text{if } -\mu \le r_i \le \mu \\ r_i + \mu & \text{if } r_i < -\mu \end{cases}$$



ISTA/Gradient prox method

$$\min_x F(x) = f(x) + g(x)$$

- Minimize quadratic upper approximation on each iteration

$$x^{k+1} = \text{argmin}_y Q_f(x^k, y)$$

$$Q_{f,\mu}(x, y) = f(x) + \nabla f(x)^\top (y - x) + \frac{1}{2\mu}||y - x||^2 + g(y)$$

- If $\mu \le 1/L$ then in $O(L/\epsilon)$ iterations finds solution

$$\bar{x}: \; F(\bar{x}) \le F(x^*) + \epsilon$$

Fast first-order method

Nesterov, Beck & Teboulle

$$\min_x F(x) = f(x) + g(x)$$

- Minimize upper approximation at an "accelerated" point.

$$x^k = \text{argmin}_y Q_f(y^k, y)$$

$$t_{k+1} := \quad (1 + \sqrt{1 + 4t_k^2})/2$$
$$y^{k+1} := \quad x^k + \frac{t_k - 1}{t_{k+1}}[x^k - x^{k-1}]$$

- If $\mu \le 1/L$ then in $O(\sqrt{L/\epsilon})$ iterations finds solution

$$\bar{x}: \; F(\bar{x}) \le F(x^*) + \epsilon$$

Practical first order algorithms using backtracking search

Iterative Shrinkage Threshholding Algorithm (ISTA)

$$\min_x F(x) = f(x) + g(x)$$

- Minimize quadratic upper relaxation on each iteration

$$x^{k+1} = \text{argmin}_y f(x^k) + \frac{1}{2\mu_k}||x^k - \mu_k \nabla f(x^k)^\top - y||^2 + g(y)$$

- Using line search find μ_k such that

$$F(x^{k+1}) \leq Q_f(x^k, x^{k+1})$$

- In $O(1/\mu_{min}\epsilon)$ iterations finds ϵ-optimal solution (in practice better)

Nesterov, 07
Beck&Teboulle, Tseng,
Auslender&Teboulle, 08

Fast Iterative Shrinkage Threshholding Algorithm (FISTA)

$$\min_x F(x) = f(x) + g(x)$$

- Minimize quadratic upper relaxation on each iteration

$$x^k = \text{argmin}_y Q_f(y^k, y) = f(y^k) + \frac{1}{2\mu_k}||y^k - \mu_k \nabla f(y^k)^\top - y||^2 + g(y)$$

- Using line search find $\mu_k \leq \mu_{k-1}$ such that (Can be restrictive)

$$F(x^k) \leq Q_f(y^k, x^k)$$

$$t_{k+1} := (1 + \sqrt{1 + 4t_k^2})/2$$
$$y^{k+1} := x^k + \frac{t_k - 1}{t_{k+1}}[x^k - x^{k-1}]$$

- In $O(\sqrt{1/\mu_{min}\epsilon})$ iterations finds ϵ-optimal solution

Nesterov, Beck&Teboulle, Tseng

Optimization Methods in Machine Learning
Lecture 13: Proximal and Optimal Proximal Gradient Methods

Katya Scheinberg

Lehigh University

Spring 2015

Logistic Loss

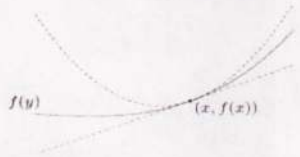
Recall the logisitic loss with L_2 norm:

$$\min_w f(w) = \frac{1}{n}\sum \log(1 + e^{-y_i(w^T x_i)}) + \lambda \|w\|_2^2$$

with L_1 norm:

$$\min_w f(w) = \frac{1}{n}\sum \log(1 + e^{-y_i(w^T x_i)}) + \lambda \|w\|_1$$

Proximal Gradient Method



Use quadratic approximation in each iteration

$$w^k = \arg\min_w : f(u^k) + \nabla f(u)^T(w - u^k) + \frac{1}{2\mu_k}\|w - u^k\|^2$$

Also let

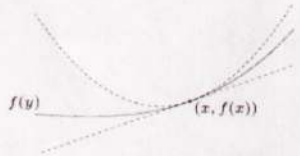
$$q(w) = \frac{1}{2\mu_k}\|w - u^k\|^2$$

Proximal Gradient Method

- Set $u^0 = 0, t_1 = 1$.
- $w^{k+1} = w^k - \mu_k \nabla f(w^k)$
- Use line search to find $\mu_k \geq \mu_{k+1}$ such that $f(w^{k+1}) \leq q(w^{k+1})$

Convergence rate: $\frac{1}{k}$.

Optimal Proximal Gradient Method



Use quadratic approximation in each iteration

$$w^k = \arg\min_w : f(u^k) + \nabla f(u)^T(w - u^k) + \frac{1}{2\mu_k}\|w - u^k\|^2$$

Also let

$$q(w) = \frac{1}{2\mu_k}\|w - u^k\|^2$$

Optimal Proximal Gradient Method

- Set $u^0 = 0, t_1 = 1$.
- $w^k = u^k - \mu_k \nabla f(u^k)$
- Use line search to find $\mu_k \geq \mu_{k+1}$ such that $f(w^{k+1}) \leq q(w^{k+1})$
- $t_{k+1} = (1 + \sqrt{1 + 4t_k^2})/2$
- $u^{k+1} = w^k + \frac{t_k - 1}{t_{k+1}}(w_k - w_{k-1})$

Convergence rate: $\frac{1}{k^2}$.

Comparison

- **Convergence Rate**
  Proximal Gradient Method: $\frac{1}{k}$
  Optimal Proximal Gradient Method: $\frac{1}{k^2}$
  Optimal proximal gradient method is faster than proximal gradient method.
- **Selection of μ_k**
  Selection in optimal proximal gradient method is more restrictive than that in proximal gradient method.

Second Order Method

Let

$$\ell(w) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + e^{-y_i(w^T x_i)}),$$

so

$$f(w) = \ell(w) + \lambda\|w\|_1.$$

We use the second order term to approximate f,

$$q(w) = \ell(u^k) + \nabla\ell(u^k)^T(w-u^k) + \frac{1}{2}(w-u^k)^T\nabla^2\ell(u^k)(w-u^k) + \lambda\|w\|_1,$$

from where we can use Newton method.

Second Order Method

If the Hessian is expensive to compute, then we can use,

$$q(w) = \ell(u^k) + \nabla\ell(u^k)^T(w-u^k) + \frac{1}{2}(w-u^k)^T\nabla^2 H_k(w-u^k) + \lambda\|w\|_1,$$

where H_k is the Hessian approximation in k^{th} iteration.
Normally, second order method works better than first order method.

General Form

We want to solve the following problem,

$$\min_{x\in\mathbb{R}^n} \quad f(x) + g(x),$$

where $f(x)$ is convex and smooth, $g(x)$ is convex and simple. If the problem

$$\min \quad \frac{1}{2}\|z-y\|^2 + \lambda g(y)$$

is easy, then we can state that $g(y)$ is simple.
If $g(y) = \lambda\|x\|_1$, then

$$y^* = \begin{cases} z-\lambda, & \text{if } z > \lambda \\ 0, & \text{if } -\lambda \le z \le \lambda \\ z+\lambda, & \text{if } z < -\lambda \end{cases}$$

General Form

Let

$$q(x) = f(x^k) + \nabla f(x^k)^T(y-x^k) + \frac{1}{2\mu_k}\|y-x^k\|^2 + g(y)$$

The following two problems are equivalent:

$$\min_x \quad q(x).$$

$$\min_x \quad \|x^k - u_k\nabla f(x^k) - y\|^2 + \mu_k g(y)$$

Extensions

Consider $g(y) = \lambda\sum_i \|y^i\|, \forall i$, then the problem

$$\min_y \quad \frac{1}{2}\|z-y\|^2 + \lambda\sum_i \|y^i\|$$

is equivalent to solve for each i separately because all variables y_i are independent, so

$$\min_{y_i} \quad \frac{1}{2}\|z^i - y^i\|^2 + \lambda\|y^i\|,$$

so

$$y^{i*} = \frac{r^i}{\|r^i\|}\max(0, \|r^i\| - \lambda)$$

Extensions

Consider one example that we have a group of identical features and want to study the effect,

$$Ax = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0.01 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = b,$$

where x_1, x_2 are identical.
Obviously, both $(1, 0, 1)^T$ and $(0.5, 0.5, 1.01)$ are solutions, however, if we solve the following problem,

$$\min_x \quad \frac{1}{2}\|Ax - b\|^2 + \lambda\|(x_1, x_2)^T\| + \lambda\|x_3\|,$$

we will find the unique optimal solution $(0.5, 0.5, 1.01)$.

Optimization Methods for Machine Learning
Lecture 14
Sparse Convex Optimization

Katya Scheinberg

Compressed sensing

A short introduction to *Compressed Sensing*

- An imaging perspective

Scene

Picture

- Image compression

Why do we compress images?

Introduction to *Compressed Sensing*

- Images are compressible

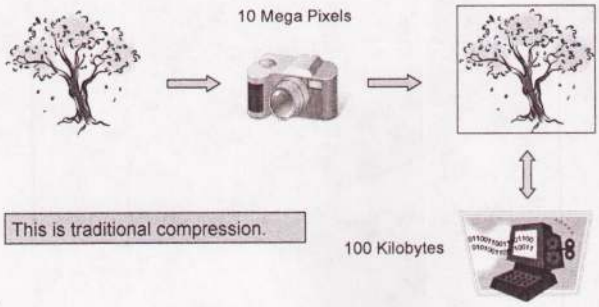
Because
- Only certain part of information is important (e.g. objects and their edges)
- Some information is unwanted (e.g. noise)

- Image compression
  - Take an input image u
  - Pick a good dictionary Φ
  - Find a sparse representation x of u such that $||\Phi x - u||_2$ is small
  - Save x

This is traditional compression.

Introduction to *Compressed Sensing*

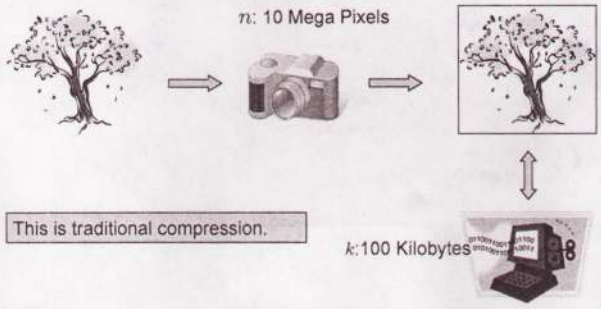
- An imaging perspective



This is traditional compression.

Introduction to *Compressed Sensing*

- An imaging perspective



This is traditional compression.

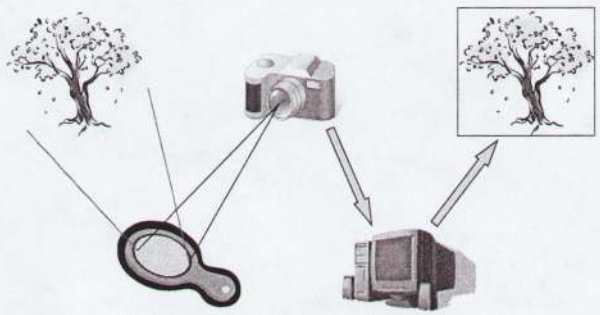
Introduction to *Compressed Sensing*

- If only 100 kilobytes are saved, why do we need a 10-megapixel camera in the first place?
- Answer: a traditional compression algorithm needs the complete image to compute Φ and x
- Can we do better than this?

Introduction to *Compressed Sensing*

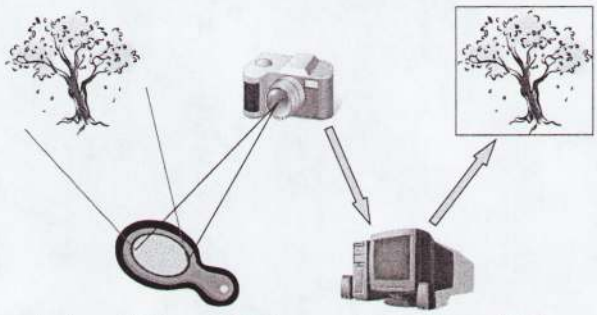
- Let $k=||x||_0$, $n=\dim(x)=\dim(u)$.
- In compressed sensing based on l_1 minimization, the number of measurements is $m=O(k \log(n/k))$ (Donoho, Candés-Tao)

Introduction to *Compressed Sensing*



Input	Linear encoding	Signal acquisition	Signal reconstruction	Signal representation
u	Bu	$b = Bu$	x	Φx

Introduction to *Compressed Sensing*



Input	Linear encoding	Signal acquisition	Signal reconstruction	Signal representation
u	Bu	$b = Bu$	x	$\Phi x \approx u$

known : B, Φ, b

want : $\vec{x}$, $\vec{u}$.

Introduction to *Compressed Sensing*

- Input: $b=Bu=B\Phi x$, $A=B\Phi$
- Output: x

$B = [$ not sparse $]$

$\Phi = [$ sparse $]$.

- In compressed sensing, $m=\dim(b)<<\dim(u)=\dim(x)=n$
- Therefore, $Ax = b$ is an *underdetermined* system
- Approaches for recovering x (hence the image u):
  - Solve min $||x||_0$, subject to $Ax = b$
  - Solve min $||x||_1$, subject to $Ax = b$
  - Other approaches

Difficulties

- Large scales
- Completely dense data: A

However

- Solutions x are expected to be sparse
- The matrices A are often fast transforms

Recovery by using the l_1-norm

Sparse signal reconstruction

$$\begin{aligned} \min \quad & ||x||_0 \\ s.t. \quad & Ax = b. \end{aligned}$$

Sparse signal $x \in \mathbf{R}^n$, matrix $A \in \mathbf{R}^{m \times n}$, $n >> m$

The system is underdetermined, but if card(x)<m, can recover signal.

The problem is NP-hard in general. Typical relaxation,

$$\begin{aligned} \min \quad & ||x||_1 \\ s.t. \quad & Ax = b. \end{aligned}$$

s-Sparsity：非0列数 C_m^s种可能.

Signal recovery

- Shown by Candes & Tao and Donoho that under certain conditions on matrix A the sparse signal

$$\begin{aligned} \min \quad & ||x||_0 \\ s.t. \quad & Ax = b. \end{aligned}$$

is recovered exactly by solving the convex relaxation

$$\begin{aligned} \min \quad & ||x||_1 \\ s.t. \quad & Ax = b. \end{aligned}$$

$\|\cdot\|_0$ can be solved by $\|\cdot\|_1$.

- The matrix property is called "restricted isometry property"

Property of A : R.I.P. 等距，等容，等高.

Restricted Isometry Property

- A vector is said to be s-sparse if it has at most s nonzero entries.
- For a given s the isometric constant δ_s of a matrix A is the smallest constant such that

$$(1 - \delta_s)\|x\|_2^2 \leq \|Ax\|_2^2 \leq (1 + \delta_s)\|x\|_2^2$$

- for any s-sparse x .

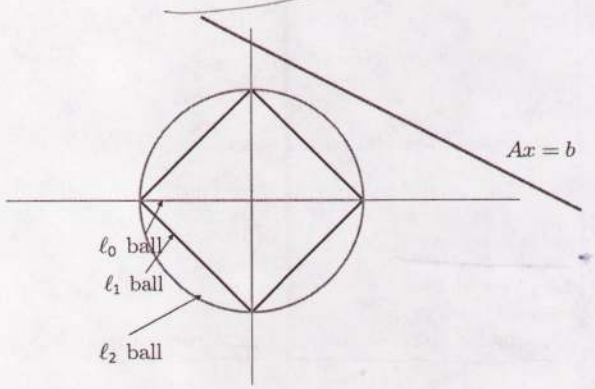
Assume that solution x^* to $\min\{\|x\|_0 : Ax = b\}$ is s-sparse.

If $\delta_{2s}(A) < 1$ then x^* is the unique solution to $\min\{\|x\|_0 : Ax = b\}$.

If $\delta_{2s}(A) < \sqrt{2} - 1$ then x^* is the solution to $\min\{\|x\|_1 : Ax = b\}$

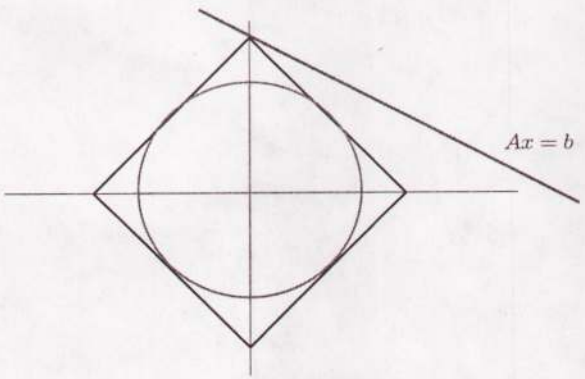
↳ when, we can use restricted isometry property.

Why $\|\cdot\|_1$ norm?

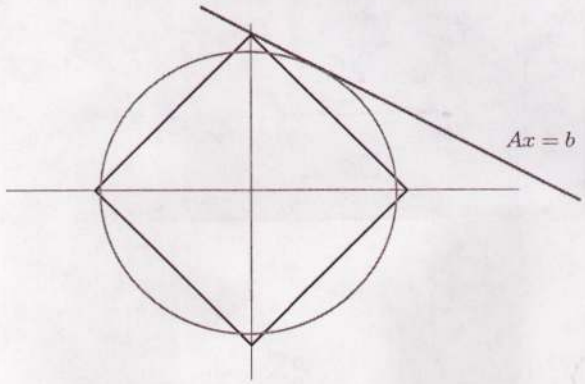


Thus, we want A to be ~~non-sing~~ singular.

Why $\|\cdot\|_1$ norm?



Why $\|\cdot\|_1$ norm?

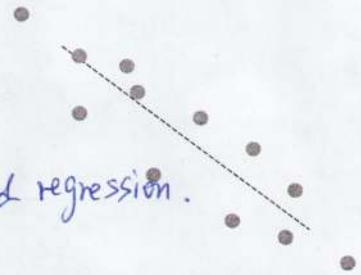


Sparse regularized regression

$||Ax - b||_n + \lambda ||x||_m$.

$(n, m) = \begin{cases} (2,1) & : \text{Lasso or Sparse regularized regression.} \\ (2,2) & : \text{Ridge regression.} \\ (1,2) & : \text{sth like SVM.} \\ (1,1). & \\ (2,0) & : \text{Standard Least Squares Problem.} \end{cases}$

Least Squares Linear Regression



Least squares problem

Standard form of LS problem

$$\min_{x \in \mathbf{R}^n} ||Ax - b||_2^2 \Rightarrow x = (A^\top A)^{-1} A^\top b$$

Includes solution of a system of linear equations Ax=b.

May be used with additional linear constraints, e.g.

$$\min_{l \le x \le u} ||Ax - b||_2^2$$

Ridge regression

$$\min_{x \in \mathbf{R}^n} ||Ax - b||_2^2 + \lambda ||x||_2^2 \Rightarrow x = (A^\top A + I)^{-1} A^\top b$$

λ is the regularization parameter – the trade-off weight.

Robust least squares regression

Assume matrix A is not known exactly, but each column $A_i \in B(A_i^0, r) = \{A_i : \|A_i - A_i^0\| \le r\}$
$\Rightarrow A \in \mathcal{A} = B(A_1^0, r) \otimes \ldots \otimes B(A_n^0, r)$.

$$\min_{x \in \mathbf{R}^n} ||Ax - b||_2^2 \Rightarrow \min_{x \in \mathbf{R}^n} \max_{A \in \mathcal{A}} ||Ax - b||_2^2$$

Less straightforward than for SVM but it is possible to show that the above problem leads to

$$\min_{x \in \mathbf{R}^n} ||A^0 x - b||_2^2 + r||x||_1$$

Another interpretation – feature selection

Lasso and other formulations

Sparse regularized regression or Lasso:

$$\min \quad \frac{1}{2}||Ax - b||^2 + \lambda ||x||_1$$

Sparse regressor selection

$$\begin{aligned} \min \quad & ||Ax - b|| \\ s.t. \quad & ||x||_1 \le t. \end{aligned}$$

Noisy signal recovery

$$\begin{aligned} \min \quad & ||x||_1 \\ s.t. \quad & ||Ax - b|| \le \epsilon. \end{aligned}$$

Connection between different formulations

$$\begin{aligned} \min \quad & ||Ax - b|| \\ s.t. \quad & ||x||_1 \le t. \end{aligned} \iff \begin{aligned} \min \quad & ||Ax - b||^2 \\ s.t. \quad & ||x||_1 \le t. \end{aligned}$$

$$\min \quad \frac{1}{2}||Ax - b|| + \lambda ||x||_1 \not\iff \min \quad \frac{1}{2}||Ax - b||^2 + \lambda ||x||_1$$

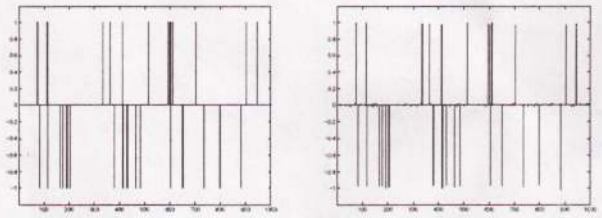
$$\min \quad \frac{1}{2}||Ax - b|| + \lambda ||x||_1 \iff \begin{aligned} \min \quad & ||Ax - b|| \\ s.t. \quad & ||x||_1 \le t. \end{aligned}$$

If solution exists

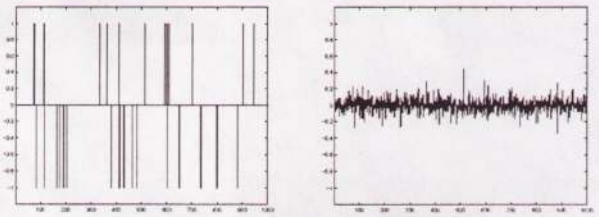
$$\begin{aligned} \min \quad & ||x||_1 \\ s.t. \quad & Ax = b \end{aligned}$$

Example

- signal $x \in \mathbf{R}^n$ with $n = 1000$, $\mathbf{card}(x) = 30$
- $m = 200$ (random) noisy measurements: $y = Ax + v$, $v \sim \mathcal{N}(0, \sigma^2 \mathbf{1})$, $A_{ij} \sim \mathcal{N}(0,1)$
- *left*: original; *right*: ℓ_1 reconstruction with $\gamma = 10^{-3}$



- ℓ_2 reconstruction; minimizes $\|Ax - y\|_2 + \gamma \|x\|_2$, where $\gamma = 10^{-3}$
- *left*: original; *right*: ℓ_2 reconstruction



Types of convex problems

$$\begin{array}{ll} \min & ||x||_1 \\ s.t. & Ax = b \end{array}$$

Variable substitution: $x = x' - x'',\ x' \geq 0,\ x'' \geq 0$

$$\begin{array}{ll} \min & e^\top (x' + x'') \\ \text{s.t.} & A(x' - x'') = b \\ & x' \geq 0, x'' \geq 0 \end{array}$$

Linear programming problem

Types of convex problems

$$\min \quad \frac{1}{2}||Ax - b|| + \lambda ||x||_1$$

Variable substitution: $x = x' - x'',\ x' \geq 0,\ x'' \geq 0$

$$\begin{array}{ll} \min & \frac{1}{2}||A(x' - x'') - b|| + \lambda e^\top (x' + x'') \\ \text{s.t.} & x' \geq 0, x'' \geq 0 \end{array}$$

Convex non-smooth objective with linear inequality constraints

Types of convex problems

Convex QP with linear inequality constraints

$$\begin{array}{ll} \min & ||Ax - b||^2 \\ s.t. & ||x||_1 \leq t. \end{array} \quad \Longleftrightarrow \quad \begin{array}{ll} \min & ||(Ax' - Ax'' - b||^2 \\ s.t. & e^\top (x', x'') \leq t. \\ & x', x'' \geq 0 \end{array}$$

SOCP

$$\begin{array}{ll} \min & ||x||_1 \\ s.t. & ||Ax - b|| \leq \epsilon. \end{array}$$

Optimization approaches

Lasso

Regularized regression or Lasso:

$$\min \quad \frac{1}{2}||Ax-b||^2+\lambda||x||_1$$

$$\min \quad \frac{1}{2}||Ax'-Ax''-b||^2+\lambda e^\top(x'+x'')$$
$$s.t. \quad x', x'' \geq 0$$

Convex QP with nonnegativity constraints

Standard QP formulation

Reformulate as

$$\min \quad \frac{1}{2}||Mz-b||^2+\lambda\sum_{i=1}^{n} z_i$$
$$\text{s.t.} \quad z \geq 0 \qquad M=[A,-A]$$

$$\min \quad \frac{1}{2}z^\top M^\top Mz-b^\top Mz+\lambda\sum_{i=1}^{n} z_i$$
$$\text{s.t.} \quad z \geq 0.$$

How is it different from SVMs dual QP?

Standard QP formulation

Reformulate as

$$\min \quad \frac{1}{2}||Mz-b||^2+\lambda\sum_{i=1}^{n} z_i$$
$$\text{s.t.} \quad z \geq 0 \qquad M=[A,-A]$$

$$\min \quad \frac{1}{2}z^\top M^\top Mz-b^\top Mz+\lambda\sum_{i=1}^{n} z_i$$
$$\text{s.t.} \quad z \geq 0.$$

Standard QP formulation

Reformulate as

$$\min \quad \frac{1}{2}||Mz-b||^2+\lambda\sum_{i=1}^{n} z_i$$
$$\text{s.t.} \quad z \geq 0 \qquad M=[A,-A]$$

$$\min \quad \frac{1}{2}z^\top M^\top Mz-b^\top Mz+\lambda\sum_{i=1}^{n} z_i$$
$$\text{s.t.} \quad z \geq 0.$$

Features of this QP

1. $Q=M^\top M$, where M is $m \times n$, with $n>>m$.
2. Forming Q is $O(m^2n)$, factorizing $Q+D$ is $O(m^3)$
3. There are no upper bound constraints.

IPM complexity is $O(m^3)$ per iteration

Dual Problem

$$\min \quad \frac{1}{2}||Ax'-Ax''-b||^2+\lambda(x'+x'')$$
$$s.t. \quad x', x'' \geq 0$$

$$L(x', x'', s', s'')=\frac{1}{2}||Ax'-Ax''-b||^2+\lambda e^\top(x'+x'')-s'^\top x'-s''^\top x''$$

$$\nabla_{x'} L(x', x'', s', s'')=A^\top(Ax'-Ax''-b)+\lambda e-s'=0$$
$$\nabla_{x''} L(x', x'', s', s'')=-A^\top(Ax'-Ax''-b)+\lambda e-s''=0$$
$$s', s'' \geq 0$$

Dual Problem

Using: Complementary slackness.

$$(x')^\top A^\top(Ax'-Ax''-b)+\lambda^\top x'-s'^\top x'=0$$
$$-(x'')^\top A^\top(Ax'-Ax''-b)+\lambda^\top x''-s''^\top x''=0$$

$$\max_s \min_x L(x', x'', s', s'')=$$
$$\frac{1}{2}(Ax'-Ax''-b)^\top(Ax'-Ax''-b)+\lambda e^\top(x'+x'')-s'^\top x'-s''^\top x''=$$
$$-\frac{1}{2}(Ax'-Ax'')^\top(Ax'-Ax'')=-\frac{1}{2}x^\top A^\top Ax$$

Lasso

Primal-Dual pair of problems

$$\min \quad \frac{1}{2}||Ax-b||^2 + \lambda||x||_1$$

$$\max \quad \min \quad \frac{1}{2}x^\top A^\top Ax$$
$$s.t. \quad \|A^\top(Ax-b)\|_\infty \leq \lambda$$

Optimality Conditions

(i) $x_i < 0$, and $(A^\top(Ax-b))_i = \lambda$,

(ii) $x_i > 0$, and $(A^\top(Ax-b))_i = -\lambda$,

(iii) $x_i = 0$, and $-\lambda \leq A^\top(Ax-b)_i \leq \lambda$

Coordinate descent

Coordinate descent

Choose one variable x_i and column A_i.
Let $\bar{x}$ and $\bar{A}$ correspond to the fixed part

$$\min_{x_i} \quad \frac{1}{2}||A_i x_i + \bar{A}\bar{x} - b||^2 + \lambda|x_i|$$

Soft-thresholding operator

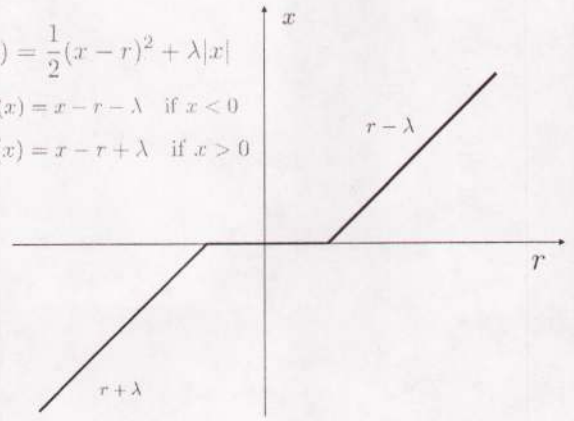
$$\min_{x_i} \frac{1}{2}(x_i - r)^2 + \lambda|x| \rightarrow x_i = \begin{cases} r-\lambda & \text{if } r > \lambda \\ 0 & \text{if } -\lambda \leq r \leq \lambda \\ r+\lambda & \text{if } r < -\lambda \end{cases}$$

$$r = -A_i{}^\top(\bar{A}\bar{x} - b)/||A_i||^2, \ \lambda \rightarrow \lambda/||A_i||^2$$

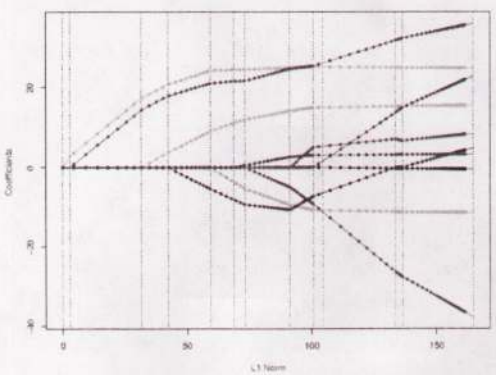
$$f(x) = \frac{1}{2}(x-r)^2 + \lambda|x|$$

$$\nabla_x f(x) = x - r - \lambda \quad \text{if } x < 0$$

$$\nabla_x f(x) = x - r + \lambda \quad \text{if } x > 0$$



LARS and GLMNET



Optimization Methods in Machine Learning
Lecture 15: Optimization approaches to sparse regularized regression

Katya Scheinberg

Lehigh University

Spring 2016

Lasso

$$\begin{aligned} &\min && \frac{1}{2}\|Ax-b\|^2+\lambda\|x\|_1 \\ =&\min && \frac{1}{2}x^TA^TAx-b^TAx+\lambda\|x\|_1 \qquad \text{constant ignored} \end{aligned}$$

- A^TA is a $n \times n$ matrix, where n is the number of feature in one sample.
- Two methods to solve this.
  - Coordinate descent method
  - First order method

Coordinate descent method

- choose one variable x_i and one column A_i.
- Let $\bar{x}$ and $\bar{A}$ be the fixed part.
-

$$\begin{aligned} &\min_{x_i} && \frac{1}{2}\|A_ix_i+\bar{A}\bar{x}-b\|^2+\lambda|x_i| \\ =&\min_{x_i} && \frac{1}{2}(A_i^TA_i)x_i^2+(\bar{x}^T\bar{A}^TA_i)x_i-b^TA_ix_i+\lambda|x_i| \end{aligned}$$

plug into here

Soft-thresholding operator

- Equivalent problem

$$\min_\theta \ \frac{1}{2}(\theta-r)^2+\lambda|\theta|$$

- Taking derivative

$$\nabla_\theta f(\theta)=\begin{cases}\theta-r-\lambda & \text{if } \theta<0\\ \theta-r+\lambda & \text{if } \theta>0\end{cases}$$

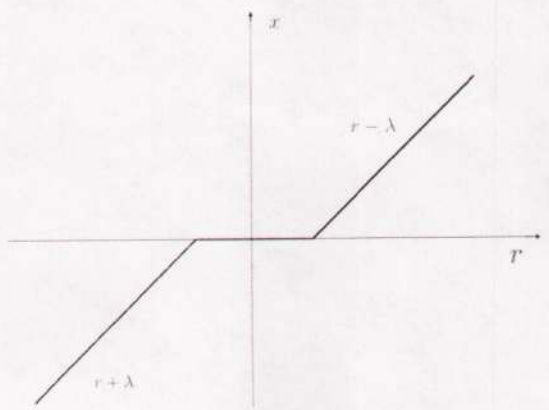
- Then we have

$$\theta^*=\begin{cases}0 & \text{if } -\lambda\le r\le\lambda\\ r-\lambda & \text{if } r>\lambda\\ r+\lambda & \text{if } r<\lambda\end{cases}$$

shrinkage.

intuitive: 0 will go to r.

Illustration



Specific step i

- On iteration i, we solve

$$\min \ \frac{1}{2}(x_i-r_i)^2+\lambda|x_i|$$

- For r:

$$r_i=A_i^T(\bar{A}\bar{x}-b)/\|A_i\|$$

Algorithm

- Pick x_0
- For $k = 0, 1, 2, \ldots$
- compute $\bar{A}\bar{x}$, given Ax_i for given i
- compute $r_i = -A_i^T(\bar{A}\bar{x} - b)/\|A_i\|$
- compute $\mu_i = \lambda/\|A_i\|$
- Solve

$$\min_{\theta} \ \frac{1}{2}(\theta - r_i)^2 + \mu_i|\theta|$$

- update

$$x_{k+1} = x_k + \theta e_i, \quad Ax_{k+1} = Ax_k + \theta A_i$$

- Avoid $n \times d$ operation. How? starting from 0 vector.

Something like stochastic gradient descent. but chose part of parameters to minimize in primal space

convergence rate:
randomize: $O(\frac{1}{k}) \propto \frac{1}{L_i}$ → coordinate Lipschitz Constants.
First order: $O(\frac{1}{k}) \propto \mu_i \propto \frac{1}{L}$.

Computation cost

- $Ax_k - A_i x_i \approx \mathbb{O}(n)$
- $Ax_{k+1} = Ax_k + \theta A_i \approx \mathbb{O}(n)$

First order method

- $$\min \quad \frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_1$$

Given x_k, quadratic over-approximation:

$$Q(y) = f(x_k) + \nabla(A^\top(Ax_k - b))^\top(y - x_k) + \frac{1}{2\mu}\|y - x_k\|^2 + \lambda\|y\|_1$$

- Minimize Q(y)

$$\min_y \quad f(x_k) + \frac{1}{2\mu_k}\|x_k - \mu_k A^\top(Ax_k - b) - y\|^2 + \lambda\|y\|_1$$
$$= \min_{y_i} \quad \sum_i [\frac{1}{2\mu_k}((x_k)_i - \mu_k A_i^\top(Ax_k - b) - y_i)^2 + \lambda|y_i|]$$

- Cost of computing $A^\top(Ax_k - b)$ is $O(m \times n)$. Cost of minimizing $Q(y)$ is $O(n)$.
- For signal processing, the cost of Ax can be $O(n \log n)$ for some cases, but A is not explicitly available.

Accelerated first order method

$$\min \quad \frac{1}{2}\|Ax - b\|^2 + \lambda\|x\|_1$$

Given x_k, quadratic over-approximation:

$$Q(y) = f(x_k) + \nabla(A^\top(Ax_k - b))^\top(y - x_k) + \frac{1}{2\mu}\|y - x_k\|^2 + \lambda\|y\|_1$$

Algorithm 1 FISTA (fast iterative shrinkage-thresholding algorithm)

Initialization: start with x^0, $y^1 = x^0$ and $t^1 = 1$.

for $k = 0$ to maximum iterations **do**
  $x^k \leftarrow \mathbf{prox}_\mu(y^k - \mu A^\top(Ay^k - b), \lambda)$, for appropriate μ,
  $t^{k+1} = (1 + \sqrt{1 + 4(t^k)^2})/2$,
  $y^{k+1} \leftarrow x^k + \frac{t^k - 1}{t^{k+1}}(x^k - x^{k-1})$.
end for

Extensions to group sparsity

Consider $g(y) = \lambda \sum_i \|y^i\|, \forall i$, then the problem

$$\min_y \ \frac{1}{2}\|r - y\|^2 + \lambda \sum_i \|y^i\|$$

# of groups.

is equivalent to solve for each i separately because all variables y_i are independent, so

$$\min_{y_i} \ \frac{1}{2}\|r^i - y^i\|^2 + \lambda\|y^i\|,$$

so

$$y^{i*} = \frac{r^i}{\|r^i\|}\max(0, \|r^i\| - \lambda)$$

$X^* = [[0\ 0], [x_2, x_3], [0, 0]]$

Extensions

Consider one example that we have a group of identical features and want to study the effect,

$$Ax = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0.01 & 1 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = b,$$

where x_1, x_2 are identical.
Obviously, both $(1, 0, 1)^T$ and $(0.5, 0.5, 1.01)$ are solutions, however, if we solve the following problem,

$$\min_x \ \frac{1}{2}\|Ax - b\|^2 + \lambda\|(x_1, x_2)^T\| + \lambda\|x_3\|,$$

we will find the unique optimal solution $(0.5, 0.5, 1.01)$.

Optimization Methods In Machine Learning

Lecture 16: Stochastic Gradient Descent Method

Professor Katya Scheinberg

Lehigh University

Spring 2016

Review

Recall SVM and Logistic regression problems,

what we want

$$\min_w \ f(w) = \min_w \ \frac{\lambda}{2}\|w\|^2 + \mathbb{E}_{x,y}\left(L(w^T x, y)\right).$$

where

$$\left(L(w^T x), y\right) = \max\{0, 1 - y(w^T x)\}. \quad \text{hinge}$$

or

$$\left(L(w^T x), y\right) = \log(1 + e^{-yx^T w}) \quad \text{logistic}$$

estimate

- For a given sample set S with size "m" we can derive approximate loss function,

$$\min_w \ f_S(w) = \min_w \ \frac{\lambda}{2}\|w\|^2 + \frac{1}{n}\sum_{i=1}^{n} L(w^T x_i, y_i).$$

Where $S = \{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$, $x_i \in \mathbb{R}^d$ and $y \in \{+1, -1\}$,

Motivation

As we mentioned before, we can solve SVM problem with "Interior Point Method", which

- Converges very fast with polynomial time convergence rate.
- Is very expensive at each iteration,
- Solves SVM very accurately.

In ML we often do not want to solve the problem very accurately

Huh?!! Why? Because what we really care about is minimizing expected risk, not empirical risk.

First/Second Order Method

The subgradient of approximate function $f_S(w)$ is the following,

$$\nabla f_S(w) = \lambda w - \frac{1}{n}\sum_{i=1}^{n} y_i x_i \mathbb{I}\left((w^T x_i), y_i\right)$$

Where, for SVM

$$\mathbb{I}\left((w^T x_i), y_i\right) = \begin{cases} 1 & \text{if } (w^T x_i) y_i < 1 \\ 0 & \text{if else.} \end{cases}$$

or for Log. Reg.

$$\mathbb{I}\left((w^T x_i), y_i\right) = \frac{1}{e^{y_i x_i^T w} + 1}$$

At each iteration, given w_k, we need $O(nd)$ operations. To compute Hessian is $O(nd^2)$ operations.

$$\nabla^2 f(w) = \frac{1}{n}\sum \frac{e^{y_i x_i^T w}}{(1 + e^{y_i x_i^T w})^2}(y_i x_i)(y_i x_i)^T + \lambda I \qquad (1)$$

- We want n to be large so that $f_S(w) \approx f(w) = \mathbb{E}_\xi(f_S(w))$!

Stochastic Gradient

- Suppose instead of using all n sample points, we pick just a subsample $A_k \subset S$ of size m_k and compute

$$\nabla f_{A_k}(w) = \lambda w - \frac{1}{m_k}\sum_{j=1}^{m_k} y_i x_i \mathbb{I}\left((w^T x_j), y_j\right).$$

sample is random.

Do we have

$$\nabla f_{A_k}(w) \approx \nabla f_S(x)? \quad \mathbb{E}(\nabla f_{A_k}(w)) \approx \mathbb{E}(\nabla f_S(w))? \quad \mathbb{E}(\nabla f_{A_k}(w)) \approx \mathbb{E}(\nabla f_S(w))?$$

= 0 for sure

- For a given function $f(w)$ we define stochastic gradient as a **random** vector function $g(w, \xi)$ such that

$$\mathbb{E}_\xi g(w, \xi) = \nabla f(w).$$

Hence, $\nabla f_{A_k}(w)$ is a stochastic gradient for both $\nabla f_S(w)$ and $\nabla f(w)$!

need to go far.

like $\frac{1}{2}, \frac{1}{4}, \frac{1}{8}$ ... will never to 1.

Stochastic gradient method

We need to update the value of w at each iteration,

$$w_{k+1} = w_k - t_k \nabla f_{A_k}(w).$$

- How we can compute the step size at each iteration?
- Stochastic gradient descent method can not use "Line Search" to compute the step size!Why?! Not constant t_k

There is different methods to define the step size of stochastic gradient method. The sequence of steps has to satisfy

$$\sum_{k=1}^{\infty} t_k = \infty \text{ and } \sum_{k=1}^{\infty} t_k^2 < \infty.$$

choose them carefully.

- In ML methods the usual choice is

$$t_k = \frac{a}{k + b}.$$

a scalar

initial step

SGD Algorithm for SVM (Pegasos)

Choose w_1, such that $\|w_1\| \le \frac{1}{\lambda}$.

For $k = 1, 2, \cdots$,

1. Choose $A_k \subset \{1, \cdots, m\}$, where $|A_k| = p$.
2. Set $A_k^+ = \{(x, y) \in A_k,\ \ y(w_k^T x) < 1\}$.
3. $\mu_k = \frac{1}{\lambda k}$.
4. $w_{k+\frac{1}{2}} = w_k - \mu_k \lambda w_k + \frac{\mu_k}{p} \sum_{(x,y) \in A_k^+} yx$.
5. $w_{k+1} = \min\{1, \frac{\frac{1}{\sqrt{\lambda}}}{\|w_{k+\frac{1}{2}}\|}\} w_{k+\frac{1}{2}}$.

grow with iteration.

non-zero gradient

put A_k^+ into gradient descent

Convergence

- With constant step size (what size??), stochastic gradient method is not convergent, but function values converge at $\bar{w} = \frac{1}{k} \sum_{i=1}^{k} w_i$, which is the average of whole k iterates.
- The difference between function value which we obtained and the best function value is the following,

SGD: $E[f(\sum_{i=1}^{k} w_i) - f(w^*)] \le \tilde{O}(\frac{R^2}{\lambda k})$, where $R = \max_i \|x_i\|$.

$k \sim \tilde{O}(\log \frac{1}{\epsilon})$

- For basic gradient descent with constant step size we had,

BGD: $[f(w_k) - f(w^*)] \le (\frac{L-\mu}{L+\mu})^k [f(w_0) - f(w^*)]$, where L is Lipschitz constant.

- For a given accuracy ϵ, computation at each iteration of gradient descent and stochastic gradient descent are respectively $\{nd\}$ and $\{pd\}$.

$k \sim \tilde{O}(\frac{1}{\epsilon})$

Note: Computation at each iteration of stochastic gradient descent does not depend on the sample size!

in the beginning, don't need to have small variance

$V = \|\nabla f_{w_k} - \nabla f_{A_k}(w_k)\|$ by $|A_k| \uparrow$, $V \downarrow$. big V is good to find a good direction.

Error Decomposition

The solution of original classification problem is a classifier with an error, which is consist of three components,

$$\mathcal{E} = \mathcal{E}_{app} + \mathcal{E}_{est} + \mathcal{E}_{opt} = \mathcal{E}_{app} + \sqrt{\frac{\log n}{n}} + \epsilon$$

- **Approximation error:** Error of population minimizer. limited by class
- **Estimation error:** Extra error due to replacing expected loss with empirical loss. limited by Data Size Bound by Hoeffding
- **Optimization error:** Extra error due to only optimizing to within finite precision. Numeric Method.

$\xi \sim \tilde{O}(\sqrt{\frac{\log n}{n}}) \to n \sim \tilde{O}(\frac{1}{\xi^2} \log \frac{1}{\xi})$.

Error Decomposition - More Data - less Work

When the size of data increases, why should it be harder to get the same answer?

How much work is there to reach ϵ optimization error by a method?

- First Order Method: $O(n \log(\frac{1}{\epsilon}))$
- Second Order Method: $O(n \log(\log(\frac{1}{\epsilon})))$
- SGD Method: $O(\frac{1}{\epsilon})$

How much work is there to reach $\mathcal{E}$ total error by a method?
Assuming that, say, $\mathcal{E}_{app} \le \frac{1}{2}\mathcal{E}$, let $\epsilon = \frac{1}{4}\mathcal{E}$, then $n = O(\frac{1}{\epsilon^2} \log(\frac{1}{\epsilon}))$ (why?):
So what is the overall complexity?

- First Order Method: $O(\frac{1}{\epsilon^2} \log^2(\frac{1}{\epsilon}))$
- Second Order Method: $O(\frac{1}{\epsilon^2} \log(\frac{1}{\epsilon})))$
- SGD Method: $O(\frac{1}{\epsilon})$

Connection of SGD and coordinate descent

- Consider the SVM problem:

$$\min_w \ \frac{1}{2}\lambda \|w\|_2^2 + \frac{1}{m} \sum_{i=1}^{m} L(w^T x_i, y_i).$$

- Dual formulation:

$$\min_\alpha f(\alpha) := \frac{1}{2}\alpha^T Q \alpha - e^T \alpha$$
$$0 \le \alpha \le c.$$

Coordinate Descent

- Given the gradient $\nabla f(\alpha) = Q\alpha - e$, consider the i^{th} coordinate

$$\nabla_i f(\alpha) = (Q\alpha)_i - 1.$$

- The updating criterions are:

$$\nabla_i f(\alpha) < 0 \ \Rightarrow \ \text{increase } \alpha_i \ s.t. \ \alpha_i \le c;$$
$$\nabla_i f(\alpha) = 0 \ \Rightarrow \ \text{keep } \alpha_i;$$
$$\nabla_i f(\alpha) > 0 \ \Rightarrow \ \text{decrease } \alpha_i \ s.t. \ \alpha_i > 0.$$

Relationship with Stochastic Gradient Descent

- Denote Q_i as the i^{th} column of Q where $Q_{ij} = x_i^T x_j y_i y_j$, then by previous lectures, $w := \sum_i \alpha_i x_i y_j$, so

$$\langle Q\alpha \rangle_i = Q_i \alpha = \sum_j x_i^T x_j y_i y_j \alpha_j = (w^T x_i) y_i$$

- Consider the case when $w^T x_i y_i < 1$, we update α_i:

$$\alpha_i^{k+1/2} = \alpha_i^{(k)} - \eta_k((w^{(k)})^T x_i y_i - 1),$$

and $\alpha_j (j \neq i)$:

$$\alpha_j^{k+1/2} = \alpha_j^{(k)}.$$

The superscripts denote iterations.

Relationship with Stochastic Gradient Descent

- Multiply the previous equations by $x_i y_i$ and $x_j y_j$, respectively, and sum them up over indexes, we have

$$\begin{aligned} w^{k+1/2} &= w^{(k)} - \eta_k[((w^{(k)})^T x_i y_i) x_i y_i - x_i y_i] \\ &= w^{(k)} - \eta_k[((w^{(k)})^T x_i) x_i y_i^2 - x_i y_i] \\ &= w^{(k)} - \eta_k[((w^{(k)})^T x_i) x_i - x_i y_i] \end{aligned}$$

which is the similar as performing the stochastic gradient descent algorithm (recall from last lecture) with only sample $\{i\}$ where $w^T x_i y_i < 1$:

$$w^{(k+1)/2} = w^{(k)} - \eta_k(\lambda w^{(k)} - x_i y_i).$$

- Obviously, stochastic gradient descent is same as coordinate descent in dual space.

Subproblem

- Let α be the vector α without updating the i^{th} coordinate and let d be the displacement we are updating on the i^{th} coordinate α_i. Then min $f(\alpha)$ w.r.t. the i^{th} coordinate is equivalent to solving the subproblem:

$$\min_d \ \frac{1}{2} Q_{ii} d^2 + (Q\alpha)_i d - d$$

with $Q_{ii} = x_i^T x_i y_i^2 = x_i^T x_i, (Q\alpha)_i = Q_i \alpha$ where Q_i is the i^{th} row of Q, and then update $\alpha_i : \alpha_i \leftarrow \alpha_i + d$.

- Considering the complexity, assuming d is the dimension of x_i and m is the number of data points, $Q_{ii} = x_i^T x_i$ is $\mathcal{O}(d)$, $(Q\alpha)_i = Q_i \alpha = \sum_j x_i^T x_j y_i y_j \alpha_j$ is $\mathcal{O}(n)$! Hence too expensive for large n.

Optimization Methods in Machine Learning

Lectures 17-18

Basic linear algebra, convexity, duality in convex programming.

Start with some basics

Outline

- Basic linear algebra
- Convex and nonconvex optimization
- Optimality conditions

Basic linear and matrix algebra

Vector norms

Vector $x = (x_1, \ldots, x_n) \in \mathbf{R}^n$

- $\|x\|_p = (\sum_{i=1}^n x_i^p)^{1/p}$ - l_p-norm
- $\|x\|_2 = \sqrt{\sum_{i=1}^n x_i^2}$ - Euclidean norm
- $\|x\|_1 = \sum_{i=1}^n |x_i|$ - l_1-norm
- $\|x\|_\infty = \max_i |x_i|$ - l_∞-norm
- $\|x\|_0 = card(x)$ - l_0-norm - not a norm.

Note that l_0-norm is not a norm. For what values of p is l_p-norm a norm?

Matrix norms

Matrix $M = \begin{bmatrix} M_{11} & \ldots & M_{1m} \\ \vdots & \ddots & \vdots \\ M_{n1} & \ldots & M_{nm} \end{bmatrix} \in \mathbf{R}^{n \times m}$

- $\|M\|_p = \max_{\|x\|_p = 1} \|Mx\|_p$ - matrix l_p-norm
- $\|M\|_F = \sqrt{\sum_{i=1,j=1}^{n,m} M_{i,j}^2}$ - Frobenius norm
- $\|M\|_1 = \max_j \sum_{i=1}^n |M_{ij}|$ - matrix l_1-norm
- $\|M\|_\infty = \max_i \sum_{j=1}^m |M_{ij}|$ - l_∞-norm

Eigenvalues and Eigenvectors

If M is a square $n \times n$ matrix, then λ is an eigenvalue of M with corresponding eigenvector q if

$$Mq = \lambda q \text{ and } q \neq 0 .$$

Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ enumerate the eigenvalues of M.

The corresponding eigenvectors $q^1, q^2, \ldots, q^n$ of M can be chosen so that they are orthonormal, namely

$$(q^i)^T (q^j) = 0 \text{ for } i \neq j, \text{ and } (q^i)^T (q^i) = 1$$

Define:

$$Q := [q^1\, q^2\, \cdots\, q^n]$$

Then Q is an *orthonormal* matrix:

$$Q^T Q = I, \text{ equivalently } Q^T = Q^{-1}$$

$\lambda_1, \lambda_2, \ldots, \lambda_n$ are the eigenvalues of M

$q^1, q^2, \ldots, q^n$ are the corresponding orthonormal eigenvectors of M

$$Q := [q^1\, q^2\, \cdots\, q^n]$$
$$Q^T Q = I, \text{ equivalently } Q^T = Q^{-1}$$

Define D:

$$D := \begin{pmatrix} \lambda_1 & 0 & & 0 \\ 0 & \lambda_2 & & \\ & & \ddots & \\ 0 & & & \lambda_n \end{pmatrix} .$$

Property: $M = QDQ^T$.

The decomposition of M into $M = QDQ^T$ is called its *eigendecomposition*.

- If $X \in S^n$, then $X = QDQ^T$ for some orthonormal matrix Q and some diagonal matrix D. The columns of Q form a set of n orthogonal eigenvectors of X, whose eigenvalues are the corresponding entries of the diagonal matrix D.
- $X \succeq 0$ if and only if $X = QDQ^T$ where the eigenvalues (i.e., the diagonal entries of D) are all nonnegative.
- $X \succ 0$ if and only if $X = QDQ^T$ where the eigenvalues (i.e., the diagonal entries of D) are all positive.
- If M is symmetric, then

$$\det(M) = \prod_{j=1}^{n} \lambda_j$$

Symmetric positive semidefinite matrices

$X \in \mathbf{S}^n$ - symmetric $n \times n$ matrix.

$\lambda_i(X)$, $i = 1, \ldots, n$ - the eigenvalues of X (all real).

$X \succ 0 (\succeq 0) \;\Rightarrow\; \lambda_i(x) > 0 (\geq 0)\ \forall i = 1, \ldots, n$.

$\mathrm{tr}(X) = \sum_{i=1}^n \lambda_i(X) = \sum_{i=1}^n X_{ii}$ - trace of the matrix X.

$C \in \mathbf{S}^n$: $\mathrm{tr}(CX) = \sum_{i,j=1}^n C_{ij} X_{ij}$ - linear function of X.

Matrix $M = \begin{bmatrix} M_{11} & \ldots & M_{n1} \\ \vdots & \ddots & \vdots \\ M_{n1} & \ldots & M_{nn} \end{bmatrix} \in \mathbf{S}^{n \times n}$

- $\mathrm{trace}(M) = \sum_{i=1}^n M_{ii} = \sum_{i=1}^n \lambda_i(M)$
- $\mathrm{trace}(AM) = \sum_{i=1,j=1}^n A_{ij} M_{ij}$
- $\mathrm{trace}(MM^\top) = \|M\|_F^2$ - square of Frobenius norm
- $\|M\|_F = \sqrt{\sum_{i=1}^n \lambda_i^2(M)}$
- If $M \succ 0$ then $\sqrt{x^\top M x}$ is a norm

Singular values

Matrix $M = \begin{bmatrix} M_{11} & \dots & M_{1m} \\ \vdots & \ddots & \vdots \\ M_{n1} & \dots & M_{nm} \end{bmatrix} \in \mathbf{R}^{n \times m}$

- $\sigma_i(M) = \sqrt{\lambda_i(MM^\top)}$ - i-th singular value
- $\|M\|_* = \sum_i \sigma_i(M)$ - nuclear norm of M.
- $\text{rank}(M) = card(\sigma(M))$ - the number of nonzero singular values.
- $\text{rank}(M) \le \min\{n, m\}$
- For vectors $u \in \mathbf{R}^n$ and $v \in \mathbf{R}^m$ $\text{rank}(uv^\top) = 1$ - rank-one matrix

Convex Optimization Problem

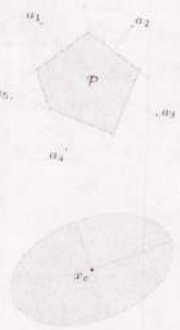
$\min f(x)$, f is convex
s.t. $x \in S$, S is convex

Convex set S:
$y, x \in S \Rightarrow \forall \theta \in [0,1]: \theta x + (1-\theta) y \in S$
Convex function $f(x)$:
$\forall \theta \in [0,1]: \theta f(x) + (1-\theta) f(y) \ge f(\theta x + (1-\theta) y)$

Convex Sets

$S = \{x \in \mathbf{R}^n : f_i(x) \le 0, i = 1, \dots, m\}$, where $f_i(x)$ are convex.

- Affine sets {Ax=b, x∈ R^n}.
- Polyhedral sets {Ax≤ b, x∈ R^n}.
- Ellipsoid {$(x-x_c)^T A(x-x_c) \le 1$}, A- p.s.d. matrix



Convex Sets

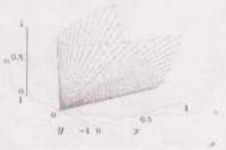
- Norm cone {(x,t): ||x||≤ t}, ||·|| is a norm.

Note that for any norm the norm cone is convex

- Positive semidefinite cone {X∈ $S^n \succeq 0$}.

S^n is a space of symmetric n×n matrices and $\succeq 0$ means that the matrix is positive semidefinite.

- Positive semidefinite set: {y∈ R^m: $B \succeq \sum_i y_i A_i$, $B, A_i \in S^n$}.



Duality in convex optimization

Duality

$$\begin{aligned} \min \quad & f_0(x), \\ \text{s.t.} \quad & f_i(x) = 0, \ i = 1, \dots, m \end{aligned}$$

Lagrangian function

$$L(x, y) = f_0(x) + \sum_{i=1}^{m} y_i f_i(x),$$

Primal problem

$$\min_x (\max_y L(x, y)) = \min_x (\max_y f_0(x) + \sum_{i=1}^{m} y_i f_i(x))$$

Dual problem

$$\max_y (\min_x L(x, y)) = \max_y (\min_x f_0(x) + \sum_{i=1}^{m} y_i f_i(x))$$

Duality

$$\begin{aligned} \min \quad & f_0(x), \\ \text{s.t.} \quad & f_i(x) = 0, \ i = 1, \ldots, m \end{aligned}$$

Lagrangian function

$$L(x, y) = f_0(x) + \sum_{i=1}^{m} y_i f_i(x)$$

Optimality Conditions

$$(x^*, y^*) : \min_x \max_y L(x, y) = \max_y \min_x L(x, y)$$

KKT conditions

$$\nabla_x L(x, y) = \nabla_x f_0(x) + \sum_{i=1}^{m} y_i \nabla_x f_i(x) = 0$$

$$f_i(x) = 0, \ i = 1, \ldots, m$$

Duality

$$\begin{aligned} \min \quad & f_0(x), \\ \text{s.t.} \quad & f_i(x) \le 0, \ i = 1, \ldots, m \end{aligned}$$

Lagrangian function

$$L(x, y) = f_0(x) + \sum_{i=1}^{m} y_i f_i(x), \ y \ge 0$$

Primal problem

$$\min_x (\max_{y \ge 0} L(x, y)) = \min_x (\max_{y \ge 0} f_0(x) + \sum_{i=1}^{m} y_i f_i(x))$$

Dual problem

$$\max_{y \ge 0} (\min_x L(x, y)) = \max_{y \ge 0} (\min_x f_0(x) + \sum_{i=1}^{m} y_i f_i(x))$$

Duality

$$\begin{aligned} \min \quad & f_0(x), \\ \text{s.t.} \quad & f_i(x) \le 0, \ i = 1, \ldots, m \end{aligned}$$

Lagrangian function

$$L(x, y) = f_0(x) + \sum_{i=1}^{m} y_i f_i(x)$$

Optimality Conditions

$$(x^*, y^*) : \min_x \max_{y \ge 0} L(x, y) = \max_{y \ge 0} \min_x L(x, y)$$

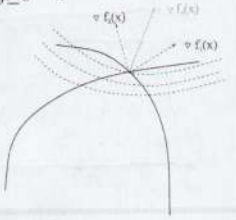
KKT conditions

$$\nabla_x L(x, y) = \nabla_x f_0(x) + \sum_{i=1}^{m} y_i \nabla_x f_i(x) = 0$$

$$y_i \ge 0, \ i = 1, \ldots, m$$

$$f_i(x) \le 0, \ i = 1, \ldots, m$$

$$f_i(x) y_i = 0$$



Generalized inequalities

a convex cone $K \subseteq \mathbf{R}^n$ is a **proper cone** if

- K is closed (contains its boundary)
- K is solid (has nonempty interior)
- K is pointed (contains no line)

examples

- nonnegative orthant $K = \mathbf{R}^n_+ = \{x \in \mathbf{R}^n \mid x_i \ge 0, i = 1, \ldots, n\}$
- positive semidefinite cone $K = \mathbf{S}^n_+$
- nonnegative polynomials on $[0, 1]$:

$$K = \{x \in \mathbf{R}^n \mid x_1 + x_2 t + x_3 t^2 + \cdots + x_n t^{n-1} \ge 0 \text{ for } t \in [0, 1]\}$$

Slides from L. Vandenberghe
http://www.ee.ucla.edu/~vandenbe/ee236c.html

generalized inequality defined by a proper cone K:

$$x \preceq_K y \iff y - x \in K, \qquad x \prec_K y \iff y - x \in \operatorname{int} K$$

examples

- componentwise inequality ($K = \mathbf{R}^n_+$)

$$x \preceq_{\mathbf{R}^n_+} y \iff x_i \le y_i, \quad i = 1, \ldots, n$$

- matrix inequality ($K = \mathbf{S}^n_+$)

$$X \preceq_{\mathbf{S}^n_+} Y \iff Y - X \text{ positive semidefinite}$$

these two types are so common that we drop the subscript in $\preceq_K$

properties: many properties of $\preceq_K$ are similar to $\le$ on $\mathbf{R}$, e.g.,

$$x \preceq_K y, \quad u \preceq_K v \implies x + u \preceq_K y + v$$

Slides from L. Vandenberghe
http://www.ee.ucla.edu/~vandenbe/ee236c.html

Dual cones and generalized inequalities

dual cone of a cone K:

$$K^* = \{y \mid y^T x \ge 0 \text{ for all } x \in K\}$$

examples

- $K = \mathbf{R}^n_+$: $K^* = \mathbf{R}^n_+$
- $K = \mathbf{S}^n_+$: $K^* = \mathbf{S}^n_+$
- $K = \{(x, t) \mid \|x\|_2 \le t\}$: $K^* = \{(x, t) \mid \|x\|_2 \le t\}$
- $K = \{(x, t) \mid \|x\|_1 \le t\}$: $K^* = \{(x, t) \mid \|x\|_\infty \le t\}$

first three examples are **self-dual** cones

dual cones of proper cones are proper, hence define generalized inequalities

$$y \succeq_{K^*} 0 \iff y^T x \ge 0 \text{ for all } x \succeq_K 0$$

Slides from L. Vandenberghe
http://www.ee.ucla.edu/~vandenbe/ee236c.html

Duality

$$\begin{aligned} \min \quad & f_0(x), \\ \text{s.t.} \quad & f_i(x) \leq_k 0, \ i = 1, \ldots, m \end{aligned}$$

Lagrangian function

$$L(x, y) = f_0(x) + \sum_{i=1}^{m} y_i f_i(x), \ y \succeq_{k^*} 0$$

Primal problem

$$\min_x (\max_{y \succeq_{k^*} 0} L(x, y)) = \min_x (\max_{y \succeq_{k^*} 0} f_0(x) + \sum_{i=1}^{m} y_i f_i(x))$$

Dual problem

$$\max_{y \succeq_{k^*} 0} (\min_x L(x, y)) = \max_{y \succeq_{k^*} 0} (\min_x f_0(x) + \sum_{i=1}^{m} y_i f_i(x))$$

Conic Primal Problem

$$\begin{aligned} \min \quad & c^T x, \\ \text{s.t.} \quad & Ax = b, x \in K \\ & K \subset \mathbf{R}^n \text{ is a convex cone,} \\ & A \in \mathbf{R}^{m \times n}, b \in \mathbf{R}^m. \end{aligned}$$

Conic Dual Problem

$$\begin{aligned} \max \quad & b^T y, \\ \text{s.t.} \quad & c - A^T y \in K^* \\ & K^* \subset \mathbf{R}^n \text{ is a convex dual cone,} \end{aligned}$$

Examples of convex conic problems

Primal Linear Programming Problem

$$\begin{aligned} \min \quad & c^T x, \\ \text{s.t.} \quad & Ax = b, \\ & x \in \mathbf{R}^n \ x \geq 0 \\ & A \in \mathbf{R}^{m \times n}, b \in \mathbf{R}^m. \end{aligned}$$

Positive orthant cone $K = \{x \in \mathbf{R}^n : x \geq 0\}$ - self dual.

$$\max_{y, s \geq 0} \ \min_x L(x, y, s) = c^\top x - y^\top (Ax - b) - s^\top x,$$

$$\nabla_x L(x, y, s) = c + y^\top A + s = 0,$$

Primal Linear Programming Problem

$$\begin{aligned} \min \quad & c^T x, \\ \text{s.t.} \quad & Ax = b, \\ & x \in \mathbf{R}^n \ x \geq 0 \\ & A \in \mathbf{R}^{m \times n}, b \in \mathbf{R}^m. \end{aligned}$$

Positive orthant cone $K = \{x \in \mathbf{R}^n : x \geq 0\}$ - self dual.

Dual Linear Programming Problem

$$\begin{aligned} \max \quad & b^T y, \\ \text{s.t.} \quad & A^T y \leq c \end{aligned}$$

Primal Semidefinite Programming Problem

$$\begin{aligned} \min \quad & \text{trace}(CX), \\ \text{s.t.} \quad & \text{trace}(A_i X) = b_i, \ i = 1, \ldots, m \\ & X \in \mathbf{S}^n \ X \succeq 0 \\ & C, A_i \in \mathbf{S}^n, b \in \mathbf{R}^m. \end{aligned}$$

SDP cone $K = \{x \in \mathbf{S}^n : X \succeq 0\}$ - self dual.

$$\max_{y, S \succeq 0} \ \min_X L(X, y, S) =$$

$$\text{trace}(CX) - \sum_{i=1}^{m} y_i (\text{trace}(A_i X) - b_i) - \text{trace}(SX)$$

Lecture 19 – Matrix rank minimization

Optimization Methods in Machine Learning

Katya Scheinberg

Semidefinite programming

A reminder

Primal Semidefinite Programming Problem

$$\begin{aligned} \min \quad & \mathrm{trace}(CX), \\ \text{s.t.} \quad & \mathrm{trace}(A_i X) = b_i, \ i = 1, \ldots, m \\ & X \in \mathbf{S}^n \ X \succeq 0 \\ & C, A_i \in \mathbf{S}^n, b \in \mathbf{R}^m. \end{aligned}$$

SDP cone $K = \{x \in \mathbf{S}^n : X \succeq 0\}$ - self dual.

Dual Semidefinite Programming Problem

$$\begin{aligned} \max \quad & b^T y, \\ \text{s.t.} \quad & \sum_{i=1}^{m} y_i A_i + S = C \\ & S \succeq 0. \end{aligned}$$

- Some users rate some movies they watched (or didn't!)
- Predict the rating (1..5) for each user/ movie pair.
- Use this prediction to recommend users the movies that they would like

Matrix completion problem, collaborative filtering

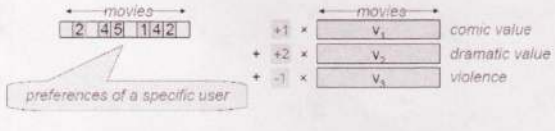
Collaborative filtering: famous Netflix challenge

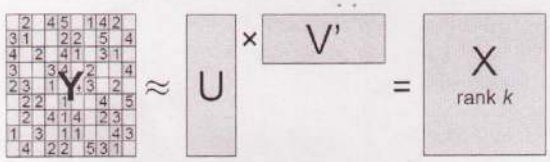
Will user i like movie j?

Complete the matrix based on partially filled information.

Slides from N. Srebro, 2005

Linear factor model





Slides from N. Srebro, 2005

Convex approximation via nuclear norm

- Given the values for a subset of entries, find the matrix with these entries and the smallest (or given) rank.

$$\begin{aligned} \min_{X \in \mathbf{R}^{m \times n}} \quad & \mathbf{rank}(X) \\ \text{s.t.} \quad & X_{ij} = M_{ij}, \ (i,j) \in I \end{aligned}$$

- NP-hard problem.

$\mathbf{rank}(X) = \|\sigma(X)\|_0$,
where $\sigma(X)$ is the vector of the singular values.

$\|\cdot\|_0 \Rightarrow \|\cdot\|_1$ - the tightest convex "relaxation".

Convex objective function: $\|\sigma(X)\|_1 = \sum_{i=1}^n \sigma_i(X)$

Under suitable randomness hypothesis
ACMRM

- Recht, Fazel and Parrilo, 2007:
  For fixed $0 < \delta < 1$, when $p = O((m+n)r\log(mn))$, with high probability, $\mathcal{A}$ satisfies the Restricted Isometry Property (RIP):

$$(1 - \delta_r(\mathcal{A}))\|X\|_F \le \|\mathcal{A}X\|_2 \le (1 + \delta_r(\mathcal{A}))\|X\|_F,$$

  with $\delta_r(\mathcal{A}) \le \delta$ for all matrices X of rank r.

Matrix Completion

- Candès and Recht, 2008: $O(n^{1.2} r \log n)$
- Candès and Tao, 2009: $O(nr\,\mathrm{poly}\log n)$

Convex approximation via nuclear norm

- Given the values for a subset of entries, find the matrix with these entries and the smallest "nuclear norm".

$$\begin{aligned} \min_{X \in \mathbf{R}^{m \times n}} \quad & \|X\|_* \\ \text{s.t.} \quad & X_{ij} = M_{ij}, \ (i,j) \in I \end{aligned}$$

- Convex problem

$$\|X\|_* = \|\sigma(X)\|_1 = \sum_{i=1}^n \sigma_i(X)$$

- Convex Cone

$$\|X\|_* \le t$$

nuclear norm.

Trace norm properties

Definition 1. *The trace norm*[1] $\|X\|_\Sigma$ *is the sum of the singular values of* X.

Lemma 1. $\|X\|_\Sigma = \min_{X=UV^*} \|U\|_{Fro}\|V\|_{Fro} = \min_{X=UV^*} \frac{1}{2}(\|U\|_{Fro}^2 + \|V\|_{Fro}^2)$

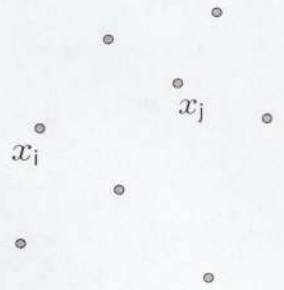
Lemma 3 ([7, Lemma 1]). *For any* $X \in \mathbb{R}^{n\times m}$ *and* $t \in \mathbb{R}$, $\|X\|_\Sigma \le t$ *iff there exists* $A \in \mathbb{R}^{n\times n}$ *and* $B \in \mathbb{R}^{m\times m}$ *such that* $\begin{bmatrix} A & X \\ X' & B \end{bmatrix} \succeq 0$ *and* $\mathrm{tr}\,A + \mathrm{tr}\,B \le 2t$.

Proof. Note that for any matrix W, $\|W\|_{Fro} = \mathrm{tr}\,WW^*$. If $\begin{bmatrix} A & X \\ X' & B \end{bmatrix} \succeq 0$, we can write it as a product $\begin{bmatrix} U \\ V \end{bmatrix}\begin{bmatrix} U^* & V^* \end{bmatrix}$. We have $X = UV^*$ and $\frac{1}{2}(\|U\|_{Fro}^2 + \|V\|_{Fro}^2) = \frac{1}{2}(\mathrm{tr}\,A + \mathrm{tr}\,B) \le t$, establishing $\|X\|_\Sigma \le t$. Conversely, if $\|X\|_\Sigma \le t$ we can write it as $X = UV^*$ with $\mathrm{tr}\,UU^* + \mathrm{tr}\,VV^* \le 2t$ and consider the p.s.d. matrix $\begin{bmatrix} UU^* & X \\ X' & VV^* \end{bmatrix}$. □

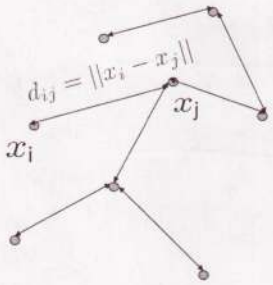
Matrix Completion formulation

$$\begin{aligned} \min \quad & \mathrm{trace}\begin{pmatrix} W_1 & X \\ X^\top & W_2 \end{pmatrix} \\ \text{s.t.} \quad & X_{ij} = M_{ij}, \ (i,j) \in I \\ & \begin{pmatrix} W_1 & X \\ X^\top & W_2 \end{pmatrix} \succeq 0 \\ & X \in \mathbf{R}^{m\times n}, \\ & W_1 \in \mathbf{R}^{m\times m}, \ W_2 \in \mathbf{R}^{n\times n} \end{aligned}$$

Sensor network localization



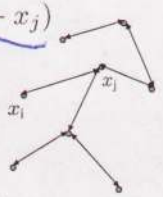
Sensor network localization



SDP relaxation of sensor network localization problem

- Given partial information on pair-wise distances find all distances and the exact locations of sensors.

$$d_{ij} = \|x_i - x_j\|^2 = (x_i - x_j)^\top (x_i - x_j)$$



Looking for matrix $X \in \mathbf{R}^{n\times 3}$
such that for all pairs (i,j)
for which d_{ij} is known
$\|x_i - x_j\|^2 = \|x_i\|^2 + \|x_j\|^2 - 2x_i{}^\top x_j$

Find $Y = XX^\top$: $Y_{ij} = x_i{}^\top x_j$
linear constaints $Y_{ii} + Y_{jj} - 2Y_{ij} = d_{ij}$, Y-low rank

Convex relaxation via trace

- Given the distances between some elements, find the matrix with a given rank (2 or 3)

$$\begin{aligned} & \mathbf{rank}(Y) = 2 \ (3) \\ \text{s.t.} \quad & Y_{ii} + Y_{jj} - 2Y_{ij} = d_{ij}, \ (i,j) \in I \\ & Y \succeq 0 \end{aligned}$$

- NP-hard problem.

$\mathbf{rank}(Y) = \|\lambda(Y)\|_0$,
where $\lambda(Y)$ is the vector of eignevalues values of Y.

Convex relaxation: $\|\lambda(Y)\|_1 = \sum_{i=1}^n \lambda_i(Y) = \mathrm{trace}(Y)$

SDP relaxation for sensor network localization

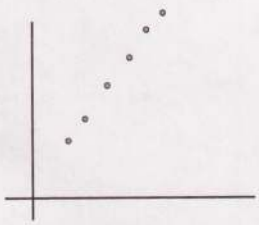
$$\begin{aligned} \min \quad & \mathrm{trace}(Y) \\ & \sum_{ij} Y_{ij} = 0 \\ \text{s.t.} \quad & Y_{ii} + Y_{jj} - 2Y_{ij} = d_{ij}, \ (i,j) \in I \\ & Y \succeq 0 \\ & Y \in \mathbf{R}^{n\times n} \end{aligned}$$

- Unsupervised learning,
- principal component analysis,
- low dimensional embedding,
- sparse and robust PCA.

Principal component analysis

Let us take three points in $\mathbf{R}^2$:

$$\begin{aligned} x_1 &= (2,1) \\ x_2 &= (4,2) \\ x_3 &= (6,3) \end{aligned}$$



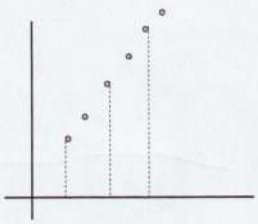
$$Y = \frac{1}{3}\sum_{i=1}^3 x_i x_i{}^\top = \frac{1}{3}\left(\begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} + \begin{bmatrix} 16 & 8 \\ 8 & 4 \end{bmatrix} + \begin{bmatrix} 36 & 18 \\ 18 & 9 \end{bmatrix} \right)$$

$$Y = \frac{1}{3}\sum_{i=1}^3 x_i x_i{}^\top = \frac{1}{3}\begin{bmatrix} 56 & 28 \\ 28 & 14 \end{bmatrix} = \frac{14}{3}\begin{bmatrix} 2 \\ 1 \end{bmatrix}\begin{bmatrix} 2 & 1 \end{bmatrix}$$

Metric multidimensional scaling

Let us take three points in $\mathbf{R}^2$:

$$\begin{aligned} x_1 &= (2,1) \\ x_2 &= (4,2) \\ x_3 &= (6,3) \end{aligned}$$



$$Y = XX^\top = \begin{bmatrix} 5 & 10 & 15 \\ 10 & 20 & 30 \\ 15 & 30 & 45 \end{bmatrix} = \begin{bmatrix} \sqrt{5} \\ 2\sqrt{5} \\ 3\sqrt{5} \end{bmatrix} \begin{bmatrix} \sqrt{5} & 2\sqrt{5} & 3\sqrt{5} \end{bmatrix}$$

Preserving the inner products $x_i^\top x_j$ we preserve distances $\|x_i - x_j\|^2$

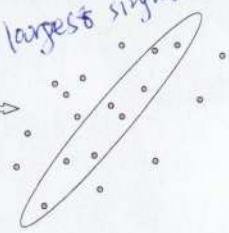
Dimensionality reduction

Principal Component Analysis = largest singular values of X

Select few largest eigenvectors of $Y = X^\top X/n$

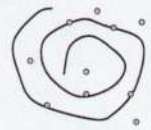
Multidimensional Metric Scaling

Select few largest eigenvectors of $Y = XX^\top$



difference ?

?



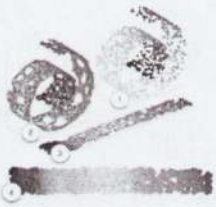


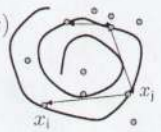
Figure 1. The problem of manifold learning, illustrated for $N = 800$ data points sampled from a "Swiss roll". (1). A discretized manifold is revealed by connecting each data point and its $k = 6$ nearest neighbors (2). An unsupervised learning algorithm unfolds the Swiss roll while preserving the local geometry of nearby data points (3). Finally, the data points are projected onto the two dimensional subspace that maximizes their variance, yielding a faithful embedding of the original manifold (4).

Nonlinear dimensionality reduction

- Preserve pair-wise distances between neighboring points.

$$d_{ij} = \|x_i - x_j\|^2 = (x_i - x_j)^\top (x_i - x_j), \text{ for } (i,j) \in I$$

Looking for matrix $Z \in \mathbf{R}^{n \times k}$ (for some k) such that for all pairs $(i,j) \in I$
$\|z_i - z_j\|^2 = \|z_i\|^2 + \|z_j\|^2 - 2z_i{}^\top z_j$



Find $Y = ZZ^\top$: $Y_{ij} = z_i{}^\top z_j$
constaints $Y_{ii} + Y_{jj} - 2Y_{ij} = d_{ij}$, $Y \succeq 0$

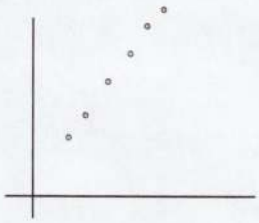
SDP formulation for nonlinear dimensionality reduction

$$\begin{aligned} \max \quad & \operatorname{trace}(Y) \\ & \sum_{ij} Y_{ij} = 0 \\ \text{s.t.} \quad & Y_{ii} + Y_{jj} - 2Y_{ij} = d_{ij}, \ (i,j) \in I \\ & Y \succeq 0 \\ & Y \in \mathbf{R}^{n \times n} \end{aligned}$$

Principal component analysis

Let us take three points in $\mathbf{R}^2$:

$$\begin{aligned} x_1 &= (2,1) \\ x_2 &= (4,2) \\ x_3 &= (6,3) \end{aligned}$$



$$Y = \frac{1}{3}\sum_{i=1}^{3} x_i x_i^\top = \frac{1}{3}\left(\begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix} + \begin{bmatrix} 16 & 8 \\ 8 & 4 \end{bmatrix} + \begin{bmatrix} 36 & 18 \\ 18 & 9 \end{bmatrix} \right)$$

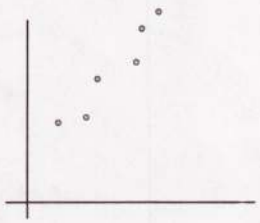
$$Y = \frac{1}{3}\sum_{i=1}^{3} x_i x_i^\top = \frac{1}{3}\begin{bmatrix} 56 & 28 \\ 28 & 14 \end{bmatrix} = \frac{14}{3}\begin{bmatrix} 2 \\ 1 \end{bmatrix}\begin{bmatrix} 2 & 1 \end{bmatrix}$$

Principal component analysis

Find direction of largest variance

Let us take three points in $\mathbf{R}^2$:

$$\begin{aligned} y_1 &= (2,1) + (O(\epsilon), O(\epsilon)) \\ y_2 &= (4,2) + (O(\epsilon), O(\epsilon)) \\ y_3 &= (6,3) + (O(\epsilon), O(\epsilon)) \end{aligned}$$



$$A = \frac{14}{\sqrt{3}} \begin{bmatrix} 2/\sqrt{3} \\ 1/\sqrt{3} \end{bmatrix} \begin{bmatrix} 2/\sqrt{3} & 1/\sqrt{3} \end{bmatrix} + \begin{bmatrix} O(\epsilon) & O(\epsilon) \\ O(\epsilon) & O(\epsilon) \end{bmatrix}$$

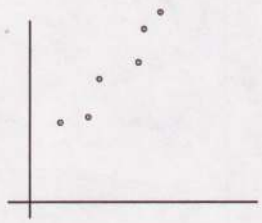
$$\begin{bmatrix} 2/\sqrt{3} \\ 1/\sqrt{3} \end{bmatrix} = \mathrm{argmax}_{x \in \mathbf{R}^2,\ \|x\|=1} x^\top A x$$

Sparse principal component analysis

Find a sparse direction of largest variance

Let us take three points in $\mathbf{R}^2$:

$$\begin{aligned} y_1 &= (2,1) + (O(\epsilon), O(\epsilon)) \\ y_2 &= (4,2) + (O(\epsilon), O(\epsilon)) \\ y_3 &= (6,3) + (O(\epsilon), O(\epsilon)) \end{aligned}$$



$$A = \frac{1}{3} \begin{bmatrix} 56 + O(\epsilon) & 28 + O(\epsilon) \\ 28 + O(\epsilon) & 14 + O(\epsilon) \end{bmatrix} \approx \frac{56}{3} \begin{bmatrix} 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 & 0 \end{bmatrix} + \begin{bmatrix} O(\epsilon) & O(\epsilon) \\ O(\epsilon) & O(\epsilon) \end{bmatrix}$$

$$\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \mathrm{argmax}_{x \in \mathbf{R}^2,\ \|x\|=1,\ \mathrm{card}(x)=1} x^\top A x$$

Introduction

Clustering of gene expression data in PCA versus sparse PCA, on 500 genes.

PCA | Sparse PCA

The PCA factors f_i on the left are dense and each use all 500 genes.
The sparse factors g_1, g_2 and g_3 on the right involve 6, 4 and 4 genes respectively.

Sparse PCA

Given a set $Y \in \mathbf{R}^{m \times n}$ compute empirical covariance matrix $A = \frac{1}{m} Y^\top Y$

Principal component analysis

Maximize the variance explained by factor x

$$\begin{array}{ll} \max\limits_{x \in \mathbf{R}^n} & x^\top A x \\ \text{s.t.} & \|x\|_2 = 1 \end{array}$$

Sparse principal component analysis

Maximize the variance explained by a factor x with bounded cardinality

$$\begin{array}{ll} \max_{x \in \mathbf{R}^n} & x^\top A x \\ \text{s.t.} & card(x) = k \\ & \|x\|_2 = 1 \end{array}$$

Semidefinite relaxation

Start from:

$$\begin{array}{ll} \text{maximize} & x^T A x \\ \text{subject to} & \|x\|_2 = 1 \\ & \mathbf{Card}(x) \leq k, \end{array}$$

where $x \in \mathbf{R}^n$. Let $X = xx^T$ and write everything in terms of the matrix X:

$$\begin{array}{ll} \text{maximize} & \mathrm{Tr}(AX) \\ \text{subject to} & \mathrm{Tr}(X) = 1 \\ & \mathbf{Card}(X) \leq k^2 \\ & X = xx^T, \end{array}$$

Replace $X = xx^T$ by the equivalent $X \succeq 0$, $\mathbf{Rank}(X) = 1$:

$$\begin{array}{ll} \text{maximize} & \mathrm{Tr}(AX) \\ \text{subject to} & \mathrm{Tr}(X) = 1 \\ & \mathbf{Card}(X) \leq k^2 \\ & X \succeq 0,\ \mathbf{Rank}(X) = 1, \end{array}$$

again, this is the same problem.

Semidefinite relaxation

We have made some progress:

- The objective $\mathrm{Tr}(AX)$ is now linear in X
- The (non-convex) constraint $\|x\|_2 = 1$ became a linear constraint $\mathrm{Tr}(X) = 1$.

But this is still a hard problem:

- The $\mathbf{Card}(X) \leq k^2$ is still non-convex.
- So is the constraint $\mathbf{Rank}(X) = 1$.

We still need to relax the two non-convex constraints above:

- If $u \in \mathbf{R}^p$, $\mathbf{Card}(u) = q$ implies $\|u\|_1 \leq \sqrt{q}\|u\|_2$. So we can replace $\mathbf{Card}(X) \leq k^2$ by the weaker (but convex): $\mathbf{1}^T |X| \mathbf{1} \leq k$.
- We simply drop the rank constraint

Semidefinite Programming

Semidefinite relaxation:

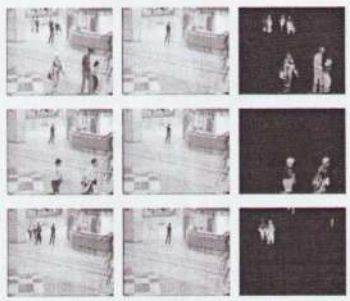
$$\begin{array}{ll} \text{maximize} & x^T A x \\ \text{subject to} & \|x\|_2 = 1 \\ & \mathbf{Card}(x) \leq k, \end{array} \quad \text{becomes} \quad \begin{array}{ll} \text{maximize} & \mathbf{Tr}(AX) \\ \text{subject to} & \mathbf{Tr}(X) = 1 \\ & 1^T |X| 1 \leq k \\ & X \succeq 0, \end{array}$$

- This is a semidefinite program in the variable $X \in \mathbf{S}^n$. . .
- Solve small problems (a few hundred variables) using IP solvers, etc.
- Dimensionality reduction apps: solve very large instances.

Solution: use first order algorithm. . .

Robust PCA

- Robust PCA $\min_{X,Y \in \mathbf{R}^{n \times m}} \|X + Y - M\|_F^2 + \lambda \|X\|_1 + \delta \|Y\|_*$



(a) Original frames (b) Low-rank L (c) Sparse S

Composite functions for matrix rank optimization

- Matrix Completion $\min_{X \in \mathbf{R}^{n \times m}} \sum_{(i,j) \in I} (X_{ij} - M_{ij})^2 + \lambda ||X||_*$
- Robust PCA $\min_{X,Y \in \mathbf{R}^{n \times m}} \|X + Y - M\|_F^2 + \lambda \|X\|_1 + \delta \|Y\|_*$

  $\min_{X \in \mathbf{R}^{n \times m}} \lambda \|X_{ij} - M_{ij}\|_1 + ||X||_*$
- Low dimensional embedding

$$\min_{X \in \mathbf{S}^{n \times n}} \sum_{(i,j) \in I} (X_{ii} + X_{jj} - 2X_{ij} - D_{ij})^2 + \lambda ||X||_*$$

Shrinkage for nuclear norm

$$\min_{X \in \mathbf{R}^{n \times m}} f(X) + ||X||_*$$

$$\min_Y Q_f(X, Y)$$

$$\Updownarrow$$

$$\min_Y \left[\frac{1}{2\mu} \|Y - Z\|_F^2 + \|Y\|_* \right]$$

$$\Updownarrow$$

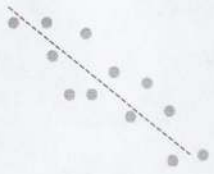
$$Z = P \mathrm{diag}\{\sigma_1, \sigma_2, \ldots, \sigma_n\} Q^\top$$

Closed form solution! O(n^3) effort

$$Y^* = P \mathrm{diag}\{\sigma_1^*, \sigma_2^*, \ldots, \sigma_n^*\} Q^\top, \; \sigma_i^* = \begin{cases} \sigma_i - \mu & \text{if } \sigma_i > \mu \\ 0 & \text{if } -\mu \leq \sigma_i \leq \mu \\ \sigma_i + \mu & \text{if } \sigma_i < -\mu \end{cases}$$

Group Lasso regression

- Problem: $\min_x \; \frac{1}{2}||Ax - b||^2 + \lambda \sum_i ||x_i||_2$



- Assume that columns of A form groups of correlated features.
- Find sparse vector x where nonzeros are selected according to groups
- x_i is a subvector of x corresponding to the i-th group of features.

Group Sparsity Shrinkage Operator

$$\min_x f(x) + \sum_i ||x_i||, \; x_i \in \mathbf{R}^{n_i}$$

- Very similar to the previous case, but with ||.|| instead of |.|

$$\sum_i \min_{y_i \in \mathbf{R}^{n_i}} \left[\frac{1}{2\mu} \|y_i - r_i\|^2 + \|y_i\| \right]$$

$$\Updownarrow$$

$$y_i^* = \frac{r_i}{\|r_i\|} \max(0, \|r_i\| - \mu)$$

Closed form solution! O(n) effort

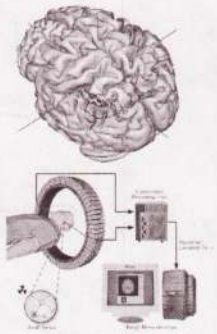
Lecture 20 – Matrix optimization in ML

Sparse inverse covariance selection

Neuroimaging: MRI, fMRI, PET

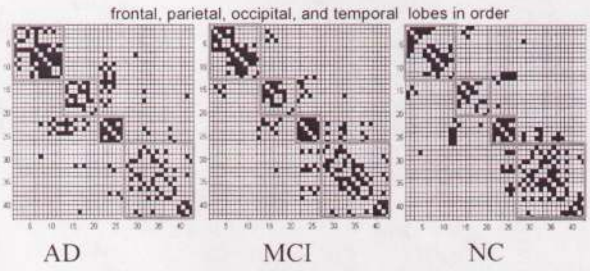
Detecting Alzheimer, Schizophrenia, etc by constructing and exploring connectivity network of the brain.

- **AD**: Grady et al. 2001, Heun et al. 2006, Celone et al 2006, Rombouts et al. 2005, Lustig et al. 2006.
- **Schizophrenia:** Cecchi, G. et al (2009), Carroll, M. K., et al(2009) Neuroimage, M. Plaze et al. (2006), Schizophrenia Research, . V.M. Eguiluz et al(2005), Phys. Rev. Letters , Y. Liu et al. (2008). Brain, Feb. 2008.



From Shuai Huang, Jing Li, Liang Sun, Jieping Ye, Adam Fleisher, Teresa Wu, Kewei Chen, and Eric Reiman. Learning Brain Connectivity of Alzheimer's Disease by Sparse Inverse Covariance Estimation. **NeuroImage**, 50, 935-949, 2010.

- There is significant, quantifiable difference in brain connectivity between AD and normal brains.

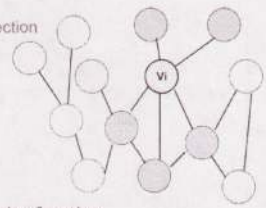


Why Sparse Inverse Covariance?

- Employ sparse inverse covariance estimation for brain region connectivity identification.
- The covariance matrix can be estimated robustly when many entries of the inverse covariance matrix are zero.
- The sparse inverse covariance matrix can be interpreted from the perspective of undirected graphical model.
  - If the ijth component of Θ is zero, then variables i and j are conditionally independent, given the other variables in the multivariate Gaussian distribution.
- Many real-world networks are sparse.
  - Gene interaction network

From Jieping Ye, KDD'09 presentation

Sparse inverse covariance selection



p random varibles

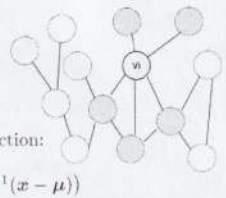
$x = \{x_1, ..., x_n\}$

Multivariate Gaussian probability density function:

$$P(\mathbf{x}) = (2\pi)^{-\frac{n}{2}} \det(\Sigma)^{-\frac{1}{2}} \exp\left(-\tfrac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

- $\Sigma \in R^{n \times n}$ - covariance matrix
- Zeros in Σ^{-1} : conditional independence
- Sparsity of Σ^{-1} : better interpretability

Sparse inverse covariance selection

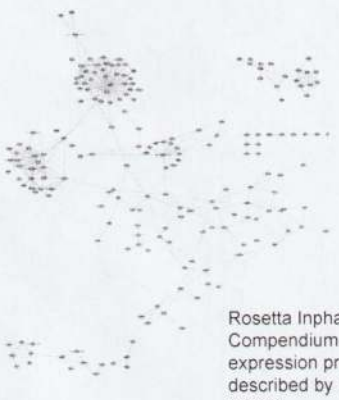


Multivariate Gaussian probability density function:

$$P(\mathbf{x}) = (2\pi)^{-\frac{n}{2}} \det(\Sigma)^{-\frac{1}{2}} \exp\left(-\tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right)$$

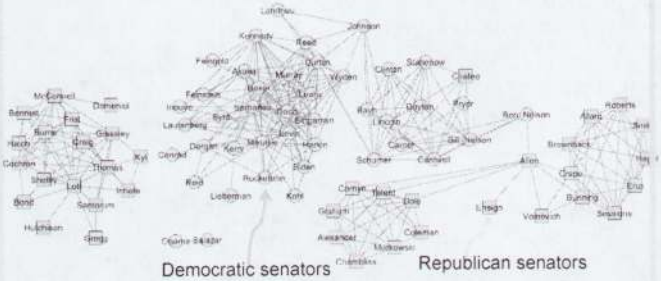
- Given X - m realizations of $\boldsymbol{x}$, $(\mu = 0)$
- $\max_\Sigma \log(P(X)) = \max_\Sigma \frac{m}{2}\log(\det(\Sigma^{-1})) - \frac{1}{2}Tr((XX^\top)\Sigma^{-1})$
- Can compute Σ^{-1} maximing log-likelihood

Example: Gene Network



Rosetta Inpharmatics Compendium of gene expression profiles described by Hughes et al. (2000)

Example: Senate Voting Records Data (2004-06)



Democratic senators Republican senators

O. Banerjee, L. El Ghaoui, and A. d' Aspremont. Model selection through sparse maximum likelihood estimation for multivariate Gaussian or binary data. *Journal of Machine Learning Research, 9:485–516*, 2008.

Optimizing log likelihood

- $\max_\Sigma \log(P(X)) = \max_\Sigma \frac{m}{2}\log(\det(\Sigma^{-1})) - \frac{1}{2}Tr((XX^\top)\Sigma^{-1})$
- Let $A = \frac{1}{m}XX^\top$
- $\Sigma^{-1} = \arg\max_C \frac{m}{2}(\log\det C - Tr(AC))$
- Solution $\Sigma^{-1} = A^{-1}$ - typically not sparse.
- Need to enforce sparsity of Σ^{-1}: Penalize for nonzeros

Enforcing sparsity

- NP-hard formulation
  $\Sigma^{-1} = \arg\max_C \left(\frac{m}{2}(\log\det C - Tr(AC)) - \lambda Card(C)\right)$
- Convex relaxation

  $\Sigma^{-1} = \arg\max_C \frac{m}{2}(\log\det C - Tr(AC)) - \lambda\|C\|_1$

  $(\|C\|_1 = \sum_{ij}|C_{ij}|)$
- Convex optimization problem with unique solution for each λ

1 norm → 0 norm

tightest relaxation.

Primal-dual pair of problems

Primal problem

$$\max_{C\succ 0} \frac{m}{2}(\ln\det(C) - Tr(AC)) - \lambda\|C\|_1$$

Reformulate using constraints

$$\begin{array}{ll}\max\limits_{C',C''} & \frac{m}{2}[\ln\det(C'-C'') - Tr(A(C'-C''))] - \lambda Tr(E(C'+C'')),\\ \text{s. t.} & C'\geq 0,\ C''\geq 0,\ C'-C''\succ 0\end{array}$$

Lagrangian

$$L(C',C'',U,V) = \frac{m}{2}[\ln\det(C'-C'') - Tr(A(C'-C''))] - \lambda Tr(E(C'+C'')) + U.*C' + V.*C''$$
$$U,V,C',C''\geq 0$$

Deriving the dual

$$\nabla_{C'} L(C', C'', U, V) = \frac{m}{2}[(C' - C'')^{-1} - A] - \lambda E + U = 0$$
$$U \geq 0$$
$$\nabla_{C''} L(C', C'', U, V) = \frac{m}{2}[-(C' - C'')^{-1} + A] - \lambda E + V = 0$$
$$V \geq 0$$

$$\Downarrow$$

$$W = (C' - C'')^{-1}$$
$$-\lambda E + V = \frac{m}{2} W - A = \lambda E - U$$
$$U, V \geq 0$$

$$\Downarrow$$

$$\frac{m}{2} \|W - A\|_\infty \leq \lambda$$

Primal-dual pair of problems

Primal problem

$$\max_{C \succ 0} \tfrac{m}{2}(\ln\det(C) - Tr(AC)) - \lambda \|C\|_1$$

Dual problem

$$\max_{W \succ 0} \{\tfrac{m}{2} \ln(\det(W)) - mp/2 : \text{s.t. } \tfrac{m}{2} \|(W - A)\|_\infty \leq \lambda\}$$

Interior point method – $O(n^3)$ operations/iter

Similar to Lasso and sparse signal recovery

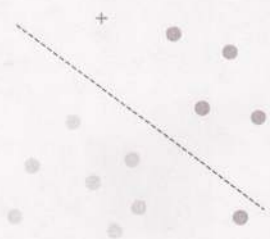
Primal-Dual pair of problems

$$\min \quad \frac{1}{2}||Ax - b||^2 + \lambda ||x||_1$$

$$\min \quad \frac{1}{2} x^\top A^\top A x$$
$$s.t. \quad \|A^\top (Ax - b)\|_\infty \leq \lambda$$

Distance metric learning

Support Vector Machines

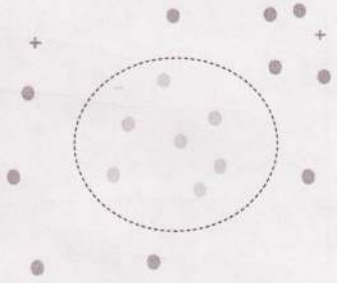


Kernel SVM

$Q_{ij} = y_i y_j x_i^\top x_j \rightarrow Q_{ij} = y_i y_j \phi(x_i)^\top \phi(x_j) = y_i y_j K(x_i, x_j)$
Kernel operation: $K(x_i, x_j) = \phi(x_i)^\top \phi(x_j)$

Examples:

- $K(x_i, x_j) = \exp^{-\|x_i - x_j\|^2 / 2\sigma^2}$
- $K(x_i, x_j) = (x_i^\top x_j / a_1 + a_2)^d$



Gaussian Kernel

$$K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma}}$$

Euclidean distance $\|x_i - x_j\|^2 = (x_i - x_j)^\top (x_i - x_j)$

Gaussian Kernel

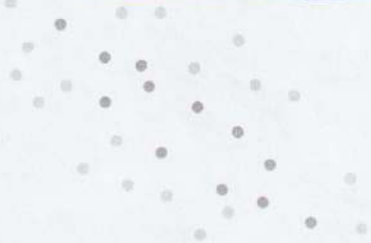
$$K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma}}$$

Euclidean distance $\|x_i - x_j\|^2 = (x_i - x_j)^\top (x_i - x_j)$

Gaussian Kernel

$$K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|_M^2}{2\sigma}}$$

Mahalanobis distance $\|x_i - x_j\|_M^2 = (x_i - x_j)^\top M (x_i - x_j)$



Distance Metric Learning

$$\begin{aligned} \max_M \quad & \sum_{(i,j)\in D} (x_i - x_j)^\top M (x_i - x_j) \\ \text{s.t.} \quad & \sum_{(i,j)\in S} (x_i - x_j)^\top M (x_i - x_j) \leqslant 1 \\ & M \succeq 0 \end{aligned}$$

S – the set of similarly labeled examples, card(S)~ $O(n^2)$

D – the set of differently labeled examples, card(D)~$O(n^2)$

IPM is too expensive, need a first order method approach

Multiple Kernel Learning

Modified from Gert Lanckriet's (UCSD) slides

Maximal margin classification

- Training: convex optimization problem (QP)
- Dual problem:

$$\max_\alpha \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \quad \text{s.t.} \quad \sum_{i=1}^n \alpha_i y_i = 0,\ 0 \leqslant \alpha_i \leqslant C$$

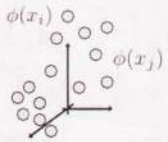
$$K_{ij} = \phi(x_i)^T \phi(x_j)$$

$$\max_\alpha \ \alpha^\top e - \frac{1}{2} \alpha^\top D_y K D_y \alpha \quad \text{s.t.} \quad \alpha^\top y = 0,\ 0 \leqslant \alpha \leqslant C$$

- Optimality condition: $w = \sum_{i=1}^n \alpha_i y_i \phi(x_i)$

Kernel-based learning

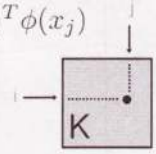
Embed data



IMPLICITLY: Inner product measures similarity

$$k(x_i, x_j) = \phi(x_i)^T \phi(x_j)$$

$$K_{ij} = \phi(x_i)^T \phi(x_j)$$



Property: Any **symmetric positive definite** matrix specifies a kernel matrix & every kernel matrix is **symmetric positive definite**

Optimizing over the kernel

- Primal problem:

$$\min_{K \succeq 0} \min_{\alpha} \quad \frac{1}{2}\alpha^\top D_y K D_y \alpha + C\sum_{i=1}^{n} \xi_i$$
$$\text{s.t.} \quad D_y K D_y \alpha + y\beta + s - \xi = -e,$$
$$0 \leqslant \alpha_i \leqslant C,\ \xi \geqslant 0$$

- Can we do this?

Optimizing over the kernel?

- Primal problem:

$$\min_{K \succeq 0} \min_{\alpha} \quad \frac{1}{2}\alpha^\top D_y K D_y \alpha + C\sum_{i=1}^{n} \xi_i$$
$$\text{s.t.} \quad D_y K D_y \alpha + y\beta + s - \xi = -e,$$
$$0 \leqslant \alpha_i \leqslant C, \xi \geqslant 0$$

- Consider $K = yy^\top \succeq 0$

$$K(x_i, x_j) = \begin{cases} 1 & \text{if } x_i,\ x_j \text{ in the same class} \\ -1 & \text{if } x_i,\ x_j \text{ in different classes} \end{cases}$$

Classification using the kernel

- Training:

$$\max_{\alpha} \quad \alpha^\top 1 - \frac{1}{2}\alpha^\top D_y K D_y \alpha \quad \text{s.t.} \quad \alpha^\top y = 0,\ 0 \leqslant \alpha \leqslant C$$

- Classification rule: classify new data point x:

$$\begin{aligned} f(\phi(x)) &= \text{sign}\left(w^T\phi(x) + b\right) \\ &= \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \phi(x_i)^T \phi(x) + b\right) \end{aligned}$$

Classification using the kernel

- Training:

$$\max_{\alpha} \quad \alpha^\top 1 - \frac{1}{2}\alpha^\top D_y K D_y \alpha \quad \text{s.t.} \quad \alpha^\top y = 0,\ 0 \leqslant \alpha \leqslant C$$

- Classification rule: classify new data point x:

$$\begin{aligned} f(\phi(x)) &= \text{sign}\left(w^T\phi(x) + b\right) \\ &= \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i \phi(x_i)^T \phi(x) + b\right) \\ &= \text{sign}\left(\sum_{i=1}^{n} \alpha_i y_i k(x_i, x) + b\right) \end{aligned}$$

Optimizing over the kernel?

- Primal problem:

$$\min_{K \succeq 0} \min_{\alpha} \quad \frac{1}{2}\alpha^\top D_y K D_y \alpha + C\sum_{i=1}^{n} \xi_i$$
$$\text{s.t.} \quad D_y K D_y \alpha + y\beta + s - \xi = -e,$$
$$0 \leqslant \alpha_i \leqslant C, \xi \geqslant 0$$

- Need additional conditions on K

When the unlabeled data is given

- Primal problem:

$$\min_{K \succeq 0} \min_{\alpha} \quad \frac{1}{2}\alpha^\top D_y K_{tr} D_y \alpha + C\sum_{i=1}^{n}\xi_i$$
$$\text{s.t.} \quad D_y K_{tr} D_y \alpha + y\beta + s - \xi = -e,$$
$$0 \leqslant \alpha_i \leqslant C,\ \xi \geqslant 0$$

K_{tr}	$K_{ts;tr}$
$K_{tr;ts}$	K_{ts}

- Still need more conditions

Kernel methods with heterogeneous data

1. First focus on every single source k of information individually
   - Extract relevant information from source j into K_j

→ Focus on kernel design for specific types of information

2. Design algorithm that learns the optimal K, by "mixing" any number of kernel matrices K_j for a given learning problem

→ Homogeneous, standardized input

→ Flexibility

→ Can ignore information irrelevant for learning task

$x^{(i)} = [\ x^{(i)}_{text}, x^{(i)}_{pic}, x^{(i)}_{sound}\ ]$.

suppose, reasonably:

$$K(x^{(i)}, x^{(j)}) = \sum_{source} \varphi_{source}(x^{(i)}_{source}, x^{(j)}_{source})$$

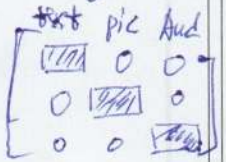
But, φ_{source} could be different a lot on scale

Classification with multiple kernels

- Consider a convex sets of kernels

$$K = \sum_{j=1}^{m} \eta_j K_{j,tr}$$
$$\sum_{j=1}^{m} \eta_j = c$$
$$\sum_{j=1}^{m} \eta_j K_j \succeq 0,\ \eta \geq 0$$

text pic sound



it's a simple case, could be dense.

$K = K_1 + K_2 + K_3$.

Can reformulate this as an SOCP

second order cone problem.

Convex combination of kernels

So, $K_{ij} = \sum_k \eta_k \varphi_k(x^{(i)}_k, x^{(j)}_k)$

$$K_{tr} = \sum_{j=1} \eta_j K_{j,tr}$$
$$\sum_j \eta_j = c$$
$$K_j \succeq 0,\ \eta \geq 0$$

$$\min_{\eta_j \geqslant 0, \sum_j \eta_j = c} \left(\max_{\alpha, \alpha^\top y = 0} \alpha^\top e - \frac{1}{2}\alpha^\top D_y \left(\sum_j \eta_j K_j\right) D_y \alpha \quad \text{s.t.} \quad 0 \leqslant \alpha \leqslant C \right)$$

Convex combination of kernels

$$\min_{\eta_j \geqslant 0, \sum_j \eta_j = c} \left(\max_{\alpha, \alpha^\top y = 0} \alpha^\top e - \frac{1}{2}\alpha^\top \left(\sum_j \eta_j K_j\right) \alpha \quad \text{s.t.} \quad 0 \leqslant \alpha \leqslant C \right)$$

Omit D_y for simplicity

Because both problems are convex and have strictly feasible solutions

$$\left(\max_{\alpha, \alpha^\top y = 0} \alpha^\top e - \max_{\eta_j \geqslant 0, \sum_j \eta_h = c} \frac{1}{2}\alpha^\top \left(\sum_j \eta_j K_j\right) \alpha \quad \text{s.t.} \quad 0 \leqslant \alpha \leqslant C \right)$$

Optimum of the linear function is achieved at the corners

consider about simplex.

$$\left(\max_{\alpha, \alpha^\top y = 0} \alpha^\top e - c \max_j \frac{1}{2}\alpha^\top (K_j) \alpha \quad \text{s.t.} \quad 0 \leqslant \alpha \leqslant C \right)$$

Convex combination of kernels

$$\left(\max_{\alpha, \alpha^\top y = 0} \alpha^\top e - c \max_j \frac{1}{2}\alpha^\top (K_j) \alpha \quad \text{s.t.} \quad 0 \leqslant \alpha \leqslant C \right)$$

$$\Updownarrow$$

$$\max_{t, \alpha} \quad \alpha^\top e - ct$$
$$\text{s.t.} \quad t \geq \frac{1}{2}\alpha^\top K_j \alpha$$
$$y^\top \alpha = 0$$
$$0 \leqslant \alpha \leqslant C$$

Convex combination of kernels

$$\left(\max_{\alpha,\alpha^\top y=0} \alpha^\top e - c\max_j \frac{1}{2}\alpha^\top (K_j)\alpha \quad \text{s.t.} \quad 0 \leqslant \alpha \leqslant C\right)$$

$$\Updownarrow$$

$$\begin{aligned} \max_{t,\alpha} \quad & \alpha^\top e - ct \\ \text{s.t.} \quad & t \geq \frac{1}{2}\alpha^\top K_j \alpha \\ & y^\top \alpha = 0 \\ & 0 \leqslant \alpha \leqslant C \end{aligned}$$

This is a QCQP

Convex combination of kernels

$$\left(\max_{\alpha,\alpha^\top y=0} \alpha^\top e - c\max_j \frac{1}{2}\alpha^\top (K_j)\alpha \quad \text{s.t.} \quad 0 \leqslant \alpha \leqslant C\right)$$

A first order method

$$\begin{aligned} \max_{t,\alpha} \quad & \alpha^\top e - ct \\ \text{s.t.} \quad & t \geq \frac{1}{2}\alpha^\top K_j \alpha \\ & y^\top \alpha = 0 \\ & 0 \leqslant \alpha \leqslant C \end{aligned}$$

An ASM

An IPM

Multiple kernels: primal problem

$$x \mapsto \begin{pmatrix} \phi_1(x) \\ \vdots \\ \phi_m(x) \end{pmatrix} \mapsto w = \begin{pmatrix} w_1 \\ \vdots \\ w_m \end{pmatrix}$$

- Primal problem

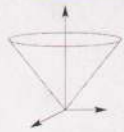
$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\left(\sum_j \|w_j\|_2\right)^2 + C\sum_i \xi_i \\ s.t. \quad & y_i(w^\top \phi(x_i) + b) \geq 1 - \xi_i, \quad i = 1,\ldots,n \\ & \xi_i \geq 0, \quad i = 1,\ldots,n \end{aligned}$$

Multiple kernels: dual problem

- Reformulation as an SOCP

$$\min_{w,b,t} \quad \frac{1}{2}\left(\sum_j t_j\right)^2 + C\sum_i \xi_i \quad \text{s.t.} \quad \forall j, \|w_j\|_2 \leqslant t_j$$

- Constraint of type $\|u\|_2 \leqslant t$
  - Second-order cone (Lorentz cone, "ice-cream cone")



  - Self-dual cone

Multiple kernels: dual problem

- Dual problem

$$\max_\alpha \quad \alpha^\top 1 - \frac{1}{2}\max_j \alpha^\top K_j \alpha \quad \text{s.t.} \quad \alpha^\top y = 0,\ 0 \leqslant \alpha \leqslant C$$

- KKT conditions
  - α is the solution of the SVM with $K = \Sigma_j \eta_j K_j$
    - η_j's: from conic duality
    - equivalent to previously obtained QCQP (for combining kernels)
  - "Support vectors": x_i for which $\alpha_i > 0$
  - "Support kernels": K_j for which $\eta_j > 0$

→ **SKM: Support kernel machine**

Methods for solving sparse inverse covariance selection problem

Block coordinate ascent

Update one row and one column of the dual matrix W at each step

$$W = \begin{bmatrix} W_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}$$

$$\max_{W \succ 0} \{ \tfrac{m}{2} \ln(\det(W)) - mp/2 : \text{s.t. } \tfrac{m}{2} \|W - A\|_\infty \leq \lambda \}$$

$$\ln \det W = \ln(\det(W_{11})(w_{22} - w_{12}{}^T W_{11}^{-1} w_{12}))$$

Block coordinate ascent subproblem

Update one row and one column of the dual matrix W at each step

$$W = \begin{bmatrix} W_{11} & w_{12} \\ w_{21} & w_{22} \end{bmatrix}$$

$$\begin{aligned} \max_{w_{12}, w_{22}} \quad & \ln(w_{22} - w_{12}{}^T W_{11}^{-1} w_{12})) \\ \text{s.t.} \quad & \|w_{12} - a_{12}\|_\infty \leq \frac{2}{m}\lambda, |w_{22} - a_{22}| \leq \frac{2}{m}\lambda \end{aligned}$$

$$\min_{w_{12}} \{ w_{12}^\top W_{11}^{-1} w_{12} : \quad \text{s.t.} \quad \|w_{12} - a_{12}\|_\infty \leq \frac{2}{m}\lambda,$$

Subproblem reformulation

$$\min_{w_{12}} \{ w_{12}^\top W_{11}^{-1} w_{12} : \quad \text{s.t.} \quad \|w_{12} - a_{12}\|_\infty \leq \frac{2}{m}\lambda,$$

$$w_{12} = W_{11}\beta$$

$$\min_{\beta} \{ \beta^\top W_{11} \beta : \quad \text{s.t.} \quad \|W_{11}\beta - a_{12}\|_\infty \leq \frac{2}{m}\lambda \}$$

Remember Lasso!

Primal-Dual pair of problems

$$\min \quad \frac{1}{2}||Ax - b||^2 + \lambda ||x||_1$$

$$\begin{aligned} \min \quad & \frac{1}{2} x^\top A^\top A x \\ s.t. \quad & \|A^\top (Ax - b)\|_\infty \leq \lambda \end{aligned}$$

Dual subproblem

$$\min_{w_{12}} \{ w_{12}^\top W_{11}^{-1} w_{12} : \quad \text{s.t.} \quad \|w_{12} - a_{12}\|_\infty \leq \frac{2}{m}\lambda,$$

$$w_{12} = W_{11}\beta$$

$$\min_{\beta} \{ \beta^\top W_{11} \beta : \quad \text{s.t.} \quad \|W_{11}\beta - a_{12}\|_\infty \leq \frac{2}{m}\lambda \}$$

$$\min_{\beta} \{ \|W_{11}^{1/2}\beta - W_{11}^{-1/2} a_{12}\|^2 + \frac{4}{m}\lambda \|\beta\|_1$$

The dual subproblem is the Lasso problem

Remember coordinate descent for Lasso

$$\min_{x_i} \quad \frac{1}{2}\|Ax - b\|^2 + \lambda \|x\|_1$$

Choose one variable x_i and column A_i.
Let $\bar{x}$ and $\bar{A}$ correspond to the fixed part

$$\min_{x_i} \quad \frac{1}{2}(A_i x_i + \bar{A}\bar{x} - b)^2 + \lambda |x_i|$$

Soft-thresholding operator

$$\min_{x_i} \frac{1}{2}(x_i - r)^2 + \lambda |x| \rightarrow x_i = \begin{cases} r - \lambda & \text{if } r > \lambda \\ 0 & \text{if } -\lambda \leq r \leq \lambda \\ r + \lambda & \text{if } r < -\lambda \end{cases}$$

$$r = -A_i{}^\top (\bar{A}\bar{x} - b)/||A_i||^2, \ \lambda \rightarrow \lambda/||A_i||^2$$

Remember coordinate descent for Lasso

$$\min_{x_i} \quad \frac{1}{2}\|W_{11}^{1/2}\beta - W_{11}^{-1/2}a_{12}\|^2 + \lambda\|\beta\|_1$$

$$\min_{\beta_i} \frac{1}{2}(\beta_i - r)^2 + \lambda|x| \to \beta_i = \begin{cases} r - \lambda & \text{if } r > \lambda \\ 0 & \text{if } -\lambda \leq r \leq \lambda \\ r + \lambda & \text{if } r < -\lambda \end{cases}$$

$$r = -((W_{11})_i{}^\top \bar{\beta} - (a_{12})_i)/(W_{11})_{ii}, \ \lambda \to \lambda/(W_{11})_{ii}$$

No need to compute $W^{1/2}$