22 Nov 2021 / 29 Nov 2021

Chap II – Regression

1 Motivation

(1) So far we have looked at classification problems where the results of hypotheses do not have any interesting structures – their domains are just sets, typically finite sets such as $\{0,1\}$. The lack of an assumed structure is reflected in the definition of risk, which treats all outputs except a single target the same and penalizes them equally. That is, the risk uses $\mathbb{1}\{h(x) \neq y\}$, and all y' with $y' \neq y$ are equally bad as the output of $h(x)$.

(2) But there are problems where the output domain has a useful structure, in particular, a metric, and this structure lets us say when one incorrect answer is actually better than another incorrect answer. In this chapter, we will study most well-known such problems, called regression problem.

(3) The regression problem asks us to learn a good function from $\mathcal{X}$ to $\mathcal{Y}$ for $\mathcal{Y} \subseteq \mathbb{R}$ from data. The key aspect of the problem is that $\mathcal{Y}$ has a notion of distance (i.e. $|y-y'|$), and we consider it when formalizing what we mean by good fns. for a regression problem. even when $h(x) \neq y$ and $h'(x) \neq y$, we say that h is better than h' for the example (x,y) if $|h(x)-y| < |h'(x)-y|$. The objective of learning is then designed so as to reflect this changed and less rigid attitude towards incorrect outputs.

More generally, we may say that a learning problem is a regression task if the output space $\mathcal{Y}$ is a metric space and the notion of error (or risk) used for learning takes this metric into account.

(cf.) In this chapter, we will study the learning theory for regression and well-known algorithms for regression.

2. Problem of Regression

(1) Setup: $\mathcal{X}$ ... space for the inputs. , $\mathcal{X} \subseteq \mathbb{R}^N$.

$\mathcal{Y}$ ... space for the outputs. , $\mathcal{Y} \subseteq \mathbb{R}$.

$D \in Pr(\mathcal{X} \times \mathcal{Y})$ .... distribution that expresses what to learn

$\mathcal{H} \subseteq \{ h : \mathcal{X} \to \mathcal{Y} \mid h \text{ is measurable} \}$.

.... hypothesis set.

(2) Learning problem

① Input: $S \sim D^m$, $S = ((x_1, y_1), \cdots, (x_m, y_m))$

drawn independently from D.

② Loss function: $L : \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$

- e.g., $L(y, y') = |y - y'|^2$ or $L(y, y') = |y - y'|^p$ for $p \geq 1$.

③ Ultimate goal: Using S, find $h \in \mathcal{H}$ that minimizes or almost minimizes. $R(h) = \mathop{\mathbb{E}}_{(x,y) \sim D} [L(h(x), y)]$.

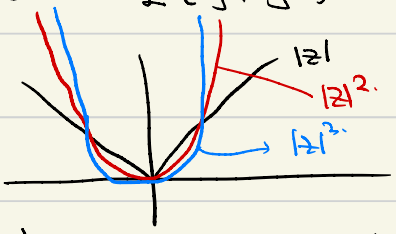
As we explained before, using this R instead of true one for classification is what makes regression very different from classification. By using L, R here can differentiate h, h' even when they have the same generalization error in the sense of classification: $\mathbb{E}_{(x,y)\sim D}[\mathbb{1}_{\{h(x)\neq y\}}] = \mathbb{E}_{(x,y)\sim D}[\mathbb{1}_{\{h'(x)\neq y\}}]$

(4) Most regression algorithms use the following empirical counterpart of R since true latter cannot be computed:

$$\hat{R}_S(h) = \frac{1}{m}\sum_{i=1}^{m} L(h(x_i), y_i).$$

(5) Assume that $L(y, y') = |y - y'|^p$ for $p \geq 1$.

Note



So, using big p amounts to the decision that we penalize large deviation from the target y more, and small deviation from y less.

3. Generalization Bounds for Bounded Regression Problems.

(1) Before studying concrete regression algorithms, we look at generalisation bounds for bounded regression problems. Results we get here will be used to analyse true algorithms that we study later.

(2) Let's start with the case of a finite hypothesis set, one of the simplest cases in true context of learning theory.

Thm 11.1 $\exists M > 0$ s.t. $L(y, y') \le M$ for all $y, y' \in \mathcal{Y}$.
Assume that $\mathcal{H}$ is finite.

$D \in Pr(\mathcal{X} \times \mathcal{Y})$ $\quad$ $\delta > 0$ $\quad$ $m \in \mathbb{N}$.

$\Rightarrow$

$$\mathbb{P}_{S \sim D} \left[\forall h \in \mathcal{H}. \quad R(h) \le \hat{R}_S(h) + M \sqrt{\frac{\log|\mathcal{H}| + \log\frac{1}{\delta}}{2m}} \right] \ge 1 - \delta.$$

The thm is very similar to Thm 2.13. ($^2/_\delta$ there should be $^1/_\delta$). except the presence of M. Both thms have essentially the same proof for. (due to the fact that the loss at (x,y) can be at most M, instead of 1).

Pf. Let $\varepsilon \overset{\text{def}}{=} M \sqrt{\frac{1}{2m}(\log|\mathcal{H}| + \log\frac{1}{\delta})}$

$$\mathbb{P}_{S \sim D^m} [\exists h \in \mathcal{H}. \; R(h) - \hat{R}_S(h) > \varepsilon].$$

$$\le \sum_{h \in \mathcal{H}} \mathbb{P}_{S \sim D^m} \left[\sum_{i=1}^m L(h(x_i), y_i) - \left(m \mathbb{E}_{(x,y) \sim D}[L(h(x), y)] \right) < -m\varepsilon \right]$$

↑ union-bound

$$\le \sum_{h \in \mathcal{H}} \exp\left(- \frac{2(m^2\varepsilon^2)}{\sum_{i=1}^m M^2} \right) = \sum_{h \in \mathcal{H}} \exp\left(-\frac{2m}{M^2} \times \left(\frac{M^2}{2m} \log(|\mathcal{H}|/\delta) \right) \right)$$

↑ Hoeffding

$$= \sum_{h \in \mathcal{H}} \frac{\delta}{|\mathcal{H}|} = \delta.$$

□

As in the case of classification, this bound suggests the use of the simple hypo. set (Occam's razor principle).

(3) Usually, though, $\mathcal{H}$ is not finite. In those cases, we will have to try what we will present next, which is derived from the Rademacher complexity bound of Thm 3.3, or other more powerful tools from learning theory.

Prop 11.2.

$L(-,y): \mathcal{Y} \to [0,\infty)$ is μ-Lipschitz. for $\mu > 0$.

$S = ((x_1,y_1), \cdots, (x_m,y_m))$

$\mathcal{G} = \{(x,y) \mapsto L(h(x),y) \mid h \in \mathcal{H}\}$.

$\Rightarrow$

$$\widehat{R}_S(\mathcal{G}) \leq \mu \widehat{R}_{S_{\mathcal{X}}}(\mathcal{H})$$

$S_{\mathcal{X}} = (x_1, \cdots, x_m)$.

Proof.

$$\widehat{R}_S(\mathcal{G}) = \frac{1}{m} \mathop{\mathbb{E}}_{\vec{\sigma} \sim \mathrm{Unif}\{\pm 1\}^m} \left[\sup_{h \in \mathcal{H}} \sum_{i=1}^m \sigma_i L(h(x_i),y_i) \right]$$

$$\leq \frac{\mu}{m} \mathop{\mathbb{E}}_{\vec{\sigma}} \left[\sup_{h \in \mathcal{H}} \sum_{i=1}^m \sigma_i h(x_i) \right] = \mu \widehat{R}_{S_{\mathcal{X}}}(\mathcal{H}).$$

$\Phi_i(-) = L(-,y_i)$, $\ell = \mu$

□

Lemma 5.7. [Talagrand's lemma].

$\Phi_1, \cdots, \Phi_m : \mathbb{R} \to \mathbb{R}$, all ℓ-Lipschitz.

$\mathcal{H} \subseteq [\mathcal{X} \to \mathbb{R}]$ measurable

$\Rightarrow$

$$\frac{1}{m} \mathop{\mathbb{E}}_{\vec{\sigma} \in \mathrm{Unif}\{\pm 1\}^m} \left[\sup_{h \in \mathcal{H}} \sum_{i=1}^m \sigma_i \left(\Phi_i(h(x_i))\right) \right]$$

$$\leq \frac{\ell}{m} \mathop{\mathbb{E}}_{\vec{\sigma}} \left[\sup_{h \in \mathcal{H}} \sum_{i=1}^m \sigma_i h(x_i) \right].$$

Thm 11.3 $L : \mathcal{Y} \times \mathcal{Y} \to [0, M]$. $L(-, y)$ is μ-Lipschitz. $D \in Pr(\mathcal{X} \times \mathcal{Y})$. $\delta > 0$. $m \in \mathbb{N}$.

$\Rightarrow$

$$\mathbb{P}_{S \sim D^m}\left[\forall h \in \mathcal{H}.\ R(h) \le \hat{R}_S(h) + 2\mu R_m(\mathcal{H}) + M\sqrt{\frac{\log 1/\delta}{2m}}\right] \ge 1-\delta.$$

and

$$\mathbb{P}_{S \sim D^m}\left[\forall h \in \mathcal{H}.\ R(h) \le \hat{R}_S(h) + 2\mu \hat{R}_S(\mathcal{H}) + 3M\sqrt{\frac{\log 2/\delta}{2m}}\right] \ge 1-\delta.$$

Proof We will show only the first. Let

$$\mathcal{G}' = \left\{ (x,y) \longmapsto \frac{1}{M} L(h(x), y) \;\middle|\; h \in \mathcal{H} \right\}.$$

Then, $\mathcal{G}' \subseteq [(\mathcal{X} \times \mathcal{Y}) \to [0,1]]$, so that we can apply.

Thm 3.3. If we do so, we get:

$$\mathbb{P}_{S \sim D^m}\left[\forall g' \in \mathcal{G}'.\ \mathbb{E}_{(x,y)\sim D}[g'(x,y)] \le \frac{1}{m}\sum_{i=1}^{m} g'(x_i,y_i) + 2R_m(\mathcal{G}') + \sqrt{\frac{\log 1/\delta}{2m}}\right] \ge 1-\delta.$$

Since $\mathcal{G}' = \frac{1}{M}\mathcal{G}$ (i.e. $\{\frac{1}{M} g \mid g \in \mathcal{G}\}$), the above implies:

$$\mathbb{P}_{S}\left[\forall g \in \mathcal{G}.\ \mathbb{E}_{(x,y)\sim D}[g(x,y)] \le \frac{1}{m}\sum_{i=1}^{m} g(x_i,y_i) + 2M R_m(\mathcal{G}') + M\sqrt{\frac{\log 1/\delta}{2m}}\right] \ge 1-\delta.$$

But $R_m(\mathcal{G}') \le \frac{1}{M} R_m(\mathcal{G})$ by Talagrand's lemma

$\le \frac{\mu}{M} R_m(\mathcal{H})$ by Prop 11.2.

So,

$$\mathbb{P}_{S}\left[\forall h \in \mathcal{H}.\ R(h) \le R_S(h) + 2\mu R_m(\mathcal{H}) + M\sqrt{\frac{\log 1/\delta}{2m}}\right] \ge 1-\delta.$$

※ Without the bound M, this proof doesn't work. R.

※ Example: $L(y,y') = |y-y'|^p$ and $|y-y'| \le M_0$ for all $y, y' \in \mathcal{Y}$.

Then, $M = M_0^p$ and $\mu = pM_0^{p-1}$ ($\because \frac{d}{dy}|y-y'|^p \le pM_0^{p-1}$).

4. Regression algorithm 1 – Linear Regression.

Linear regression is

(1) the most basic algorithm for regression. It finds a linear fn that minimizes the empirical mean squared error.

In a sense, the algorithm is similar to ERM. for classification

(2). ① Hypothesis set used. ... $\mathcal{H} = \{ x \mapsto \langle w, \Phi(x) \rangle + b \mid w \in \mathbb{R}^N, b \in \mathbb{R} \}$.

where. Φ is a map from $\mathcal{X}$ to $\mathbb{R}^N$ and intuitively, it computes. N features for a given input x.

② Loss fn : $L(y, y') = (y - y')^2$.

③ Linear regression solves the following optimization problem.

$$\min_{h \in \mathcal{H}} \widehat{R}_S(h)$$

where $S = ((x_1, y_1), \cdots, (x_m, y_m))$ is a sample. (or a training set).

Note.

$$= \min_{w,b} \frac{1}{m} \sum_{i=1}^{m} (\langle w, \Phi(x_i) \rangle + b - y_i)^2.$$

$$= \min_{W} F(W)$$

where.

$$X = \begin{bmatrix} \Phi(x_1) & \Phi(x_2) & \cdots & \Phi(x_m) \\ 1 & 1 & & 1 \end{bmatrix},$$

$$W = \begin{bmatrix} w_1 \\ \vdots \\ w_N \\ b \end{bmatrix}, \quad Y = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix},$$

$$F(W) = \frac{1}{m} \| X^T W - Y \|^2.$$

(4) The optimal w can be obtained analytically. To see this, note that $F(w)$ is differentiable and convex. So, F admits a global minimum at w iff. $\nabla F(w) = 0$.

But: $\nabla F(w) = \frac{2}{m} X(X^T w - Y) = 0$ is equivalent to.

$$XX^T w = XY.$$

Thus, the optimal w can be characterized as follows:

- if (XX^T) is invertible, $w = (XX^T)^{-1} XY$;
- if (XX^T) is not invertible, $w = (XX^T)^{\dagger} XY + (I - (XX^T)^{\dagger}(XX^T)) w_0$ for some w_0.

① Here $(XX^T)^{\dagger}$ is the pseudo inverse of XX^T. It is

$$U \Sigma^{\dagger} U^T$$

when $U \Sigma U^T$ is the eigen-decomposition of XX^T and $\Sigma^{\dagger}$ is the diagonal matrix with

$$(\Sigma^{\dagger})_{ii} = \begin{cases} 0 & \text{if } \Sigma_{ii} = 0 \\ \frac{1}{\Sigma_{ii}} & \text{if } \Sigma_{ii} \neq 0 \end{cases}$$

More generally, $A^{\dagger}$ is the unique matrix s.t.

(i) $A^{\dagger} A A^{\dagger} = A^{\dagger}$ (ii) $A A^{\dagger} A = A$ (iii) $(AA^{\dagger})^T = AA^{\dagger}$

(iv) $(A^{\dagger} A)^T = A^{\dagger} A$

② We can check $(XX^T) w = XY$ in the non-invertible case:

$$(XX^T) w = (XX^T)(XX^T)^{\dagger} XY + (XX^T)\left(I - (XX^T)^{\dagger}(XX^T)\right) w_0$$

$$= XX^T (X^T)^{\dagger} X^{\dagger} XY + \left((XX^T) - (XX^T)(XX^T)^{\dagger}(XX^T)\right) w_0$$

$$= X\left(X^T (X^{\dagger})^T (X^{\dagger} X) Y\right) + \left((XX^T) - (XX^T)\right) w_0$$

$$= X\left((X^{\dagger} X)^T (X^{\dagger} X) Y\right) = X(X^{\dagger} X) X^{\dagger} XY = XX^{\dagger} XY = XY.$$

③ In the non-invertible case, we often use:

$$w = (XX^T)^\dagger X Y$$

that has the minimum norm.

(5) ① $O(mN^2 + N^3)$ time complexity.

XX^T ↗ ↳ inversion.

② Not so good at generalization, partly due to the lack of any effort for reducing model complexity during training.

5. Regression Algorithm 2 — Kernel Ridge Regression

(1) How should we try to reduce model complexity? The next thm gives a hint, and following the hint leads to kernel ridge regression (KRR). Assume that $L(y, y') = (y - y')^2$.

Thm 11.11 $k : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ ... PDS kernel.

$\mathbb{H}$, Φ ... RKHS, feature map of k.

$\mathcal{H} = \{ w \mapsto \langle w, \Phi(x) \rangle \mid \|w\|_{\mathbb{H}} \le \Lambda \}$.

$\exists r, M$ s.t. $K(x,x) \le r^2$ for all x. and.

$|h(x) - y| < M$ for all $h \in \mathcal{H}$, $y \in \mathcal{Y}$.

$D \in Pr(\mathcal{X} \times \mathcal{Y})$. $\delta > 0$. $m \in \mathbb{N}$.

$\Rightarrow$

$$\mathbb{P}_{S \sim D^m}\left[\forall h \in \mathcal{H}.\ R(h) \le \hat{R}_S(h) + 4M\sqrt{\frac{r^2\Lambda^2}{m}} + M^2\sqrt{\frac{\log\frac{1}{\delta}}{2m}} \right] \ge 1 - \delta.$$

and

$$\mathbb{P}_{S \sim D^m}\left[\forall h \in \mathcal{H}.\ R(h) \le \hat{R}_S(h) + \frac{4M\Lambda\sqrt{\mathrm{Tr}[K]}}{m} + 3M^2\sqrt{\frac{\log\frac{2}{\delta}}{2m}} \right] \ge 1 - \delta.$$

where $K = [k(x_i, x_j)]_{ij}$.

Proof: Immediate consequence of Thm 11.3 and Thm 6.12. □

$$\hat{R}_S(\mathcal{H}) \leq \frac{\Lambda\sqrt{\mathrm{Tr}[K]}}{m} \leq \sqrt{\frac{\Lambda^2 r^2}{m}}$$

(2) One lesson of the thm is that one way to reduce model complexity is to use small Λ. We can turn this lesson to an algorithm, which becomes KRR. KRR solves the following optimization pb:

$$\min_w F(w)$$

where:

$$F(w) = \lambda\|w\|^2 + \sum_{i=1}^{m} \left(\langle w, \Phi(x_i)\rangle - y_i\right)^2.$$

$$= \lambda\|w\|^2 + \|X^T w - Y\|^2,$$

$$X = \begin{bmatrix} \Phi(x_1) & \cdots & \Phi(x_m) \end{bmatrix}, \quad Y = \begin{bmatrix} y_1 \\ \vdots \\ y_m \end{bmatrix}.$$

(3) The solution of this problem can be computed analytically. Since F is convex and differentiable, F attains a global minimum at w iff $\nabla F(w) = 0$. But:

$$\nabla F(w) = \lambda w + X(X^T w - Y).$$

So that: $\nabla F(w) = 0$ is equivalent to

$$(XX^T + \lambda I)\, w = XY.$$

$XX^T + \lambda I$ is invertible because its eigenvalues are those of XX^T, which are ≥ 0, and $\lambda > 0$ and so they are all positive. (The determinant is the product of the sums of eigenvalues). Thus, the optimal w satisfies:

$$w = (XX^T + \lambda I)^{-1} XY.$$

(4) To obtain a kernelized variant of the above algorithm, we consider a related optimization problem.

$$\min_{w} \sum_{i=1}^{m} \left(\langle w, \Phi(x_i) \rangle - y_i \right)^2$$
$$\text{s.t.} \quad \|w\|^2 \leq \Lambda^2.$$

$$= \min_{\xi, w} \sum_{i=1}^{m} \xi_i^2$$
$$\text{s.t.} \quad \|w\|^2 \leq \Lambda^2 \quad \wedge \quad \xi_i = y_i - \langle w, \Phi(x_i) \rangle \quad \text{for } \forall i \in [m]$$

(no condition $\alpha_i' \geq 0$ because α_i' is introduced for the equality = constraint)

$$= \min_{\xi, w} \max_{\alpha', \lambda \geq 0} \mathcal{L}(\xi, w, \alpha', \lambda).$$

where

$$\mathcal{L}(\xi, w, \alpha', \lambda) = \sum_{i=1}^{m} \xi_i^2 + \sum_{i=1}^{m} \alpha_i' \left(y_i - \langle w, \Phi(x_i) \rangle - \xi_i \right) + \lambda \left(\|w\|^2 - \Lambda^2 \right)$$

$$= \max_{\alpha', \lambda \geq 0} \min_{\xi, w} \mathcal{L}(\xi, w, \alpha', \lambda)$$

KKT condition:

$$\nabla_w \mathcal{L} = 0 \implies w = \frac{1}{2\lambda} \sum_{j=1}^{m} \alpha_j' \Phi(x_j)$$

$$\nabla_{\xi_i} \mathcal{L} = 0 \implies \xi_i = \alpha_i'/2.$$

$$\forall i \in [m]. \quad \alpha_i' \left(y_i - \langle w, \Phi(x_i) \rangle - \xi_i \right) = 0.$$

$$\lambda \left(\|w\|^2 - \Lambda^2 \right) = 0.$$

$$= \max_{\alpha', \lambda \geq 0} \sum_{i=1}^{m} \frac{\alpha_i'^2}{4} + \sum_{i=1}^{m} \alpha_i' \left(y_i - \frac{1}{2\lambda} \sum_{j=1}^{m} \alpha_j' k(x_j, x_i) - \frac{\alpha_i'}{2} \right) + \frac{1}{4\lambda} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i' \alpha_j' k(x_i, x_j) - \lambda \Lambda^2$$

$$= \max_{\alpha', \lambda \geq 0} -\frac{1}{4}\sum_{i=1}^{m} \alpha_i'^2 + \sum_{i=1}^{m} \alpha_i' y_i - \frac{1}{4\lambda}\sum_{i,j=1}^{m} \alpha_i' \alpha_j' k(x_i, x_j) - \lambda\Lambda^2.$$

Let $\alpha_i' = 2\sqrt{\lambda}\,\alpha_i$

$$= \max_{\alpha, \lambda \geq 0} -\lambda \sum_{i=1}^{m} \alpha_i^2 + 2\sqrt{\lambda}\sum_{i=1}^{m} \alpha_i y_i - \sum_{i,j=1}^{m} \alpha_i \alpha_j k(x_i, x_j) - \lambda\Lambda^2$$

$$= \max_{\alpha, \lambda \geq 0} -\lambda \alpha^T\alpha + 2\sqrt{\lambda}\,\alpha^T Y - \alpha^T K \alpha - \lambda\Lambda^2.$$

(and ignore $-\lambda\Lambda^2$)

Now if we treat λ as a given constant, this opti. is equivalent to the dual of KRR.

Thus, the kernelized dual version of KRR solves:

$$\underset{\alpha \in \mathbb{R}^m}{\operatorname{argmax}} \underbrace{-\lambda \alpha^T\alpha + 2\sqrt{\lambda}\,\alpha^T Y - \alpha^T K \alpha}_{= G(\alpha)}.$$

G is concave and differentiable. Thus, it has a global optimum at α iff $\nabla_\alpha G(\alpha) = 0$.

But $\nabla_\alpha G(\alpha) = -2\lambda\alpha + 2\sqrt{\lambda}\,Y - 2K\alpha$.

Thus, $\nabla_\alpha G(\alpha) = 0$ is equivalent to

$$\alpha = \sqrt{\lambda}\,(\lambda I + K)^{-1} Y$$

In this case,

$$\langle w, \Phi(x)\rangle = \frac{1}{2\lambda}\sum_{i=1}^{m} 2\sqrt{\lambda}\,\alpha_i \langle \Phi(x_i), \Phi(x)\rangle = \frac{1}{\sqrt{\lambda}}\sum_{i=1}^{m} \alpha_i k(x_i, x)$$

$$= \frac{1}{\sqrt{\lambda}} K_{x, x_{1:m}} \alpha = \frac{1}{\sqrt{\lambda}} K_{x, x_{1:m}} \sqrt{\lambda}\,(\lambda I + K)^{-1} Y$$

$$= K_{x, x_{1:m}} (\lambda I + K)^{-1} Y.$$

almost.

(5) KRR with bias term b can be covered by what we discussed so far if we use $\Phi'(x) = \begin{bmatrix} \Phi(x) \\ 1 \end{bmatrix} \in \mathbb{R}^{N+1}$.

But in this case, we optimize $\lambda \left\| \begin{bmatrix} w \\ b \end{bmatrix} \right\|^2 + \dots$ instead of $\lambda\|w\|^2 + \dots$

(6) Computational Complexity

m examples. $\quad$ K .. cost of computing $k(x, x')$.

N features.

	Learning	Prediction
primal	$O(mN^2 + N^3)$	$O(N)$
dual	$O(Km^2 + m^3)$	$O(mK)$

So, depending on whether $N > m$ or $N < m$, we may prefer primal version or dual version.

(7) One drawback of KRR is that its solution is not sparse. The next two algorithms are designed to address this issue.

6. Regression Algorithm 3 – Support Vector Regression (SVR)

(1) The support vector regression is a regression algorithm that often finds a sparse solution, i.e., a fn that uses a small number of features or dimensions in the input. The key idea of the SVR is not to penalise a prediction if it is close enough (ε-close) to the ground truth. It lets the SVR achieve sparsity and be analysed using concepts from the SVM, such as support vector.

(2) The SVR solves the following optimization problem:

① Input : $S = ((x_1, y_1), \cdots, (x_m, y_m))$

Hyperparameters : $C > 0$, $\varepsilon > 0$.

② $\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} |y_i - (\langle w, \Phi(x_i)\rangle + b)|_\varepsilon$

where $|y|_\varepsilon = \max(0, |y| - \varepsilon)$.

Φ: feature map

=

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

subject to. $\xi_i' \geq 0 \;\wedge\; \xi_i' \geq y_i - (\langle w, \Phi(x_i)\rangle + b) - \varepsilon$

$\xi_i \geq 0 \;\wedge\; \xi_i \geq (\langle w, \Phi(x_i)\rangle + b) - y_i - \varepsilon$

for all $i \in [m]$.

Convex quadratic optimisation pb. with linear constraints.

Can be solved by an off-the-shelf optimizer.

(3) The optimization from above can be dualized as shown below, and can be converted into a kernelized algorithm.

① $L(w, b, \xi, \xi', \alpha, \alpha', \beta, \beta')$

$$= \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}(\xi_i + \xi_i') + \sum_{i=1}^{m}\left(\beta_i(-\xi_i) + \beta_i'(-\xi_i')\right)$$

$$+ \sum_{i=1}^{m}\Big(\alpha_i\big((\langle w, \Phi(x_i)\rangle + b) - y_i - \varepsilon\big) + \alpha_i'\big(y_i - (\langle w, \Phi(x_i)\rangle + b) - \varepsilon\big)\Big)$$

($-\xi_i$ with α_i; $-\xi_i'$ with α_i')

$$\min \ldots = \inf_{w,b,\xi,\xi'} \sup_{\alpha,\alpha',\beta,\beta' \in [0,\infty)^m} L(w, b, \xi, \xi', \alpha, \alpha', \beta, \beta')$$

$$= \sup_{\alpha,\alpha',\beta,\beta' \in [0,\infty)^m} \inf_{w,b,\xi,\xi'} L(w, b, \xi, \xi', \alpha, \alpha', \beta, \beta').$$

KKT cond. $\nabla_w L = 0 \Rightarrow w = -\sum_{i=1}^{m}(\alpha_i - \alpha_i')\Phi(x_i)$.

$\nabla_b L = 0 \Rightarrow \sum_{i=1}^{m}(\alpha_i - \alpha_i') = 0$. $\quad \nabla_{\xi_i} L = 0 \Rightarrow C = \alpha_i + \beta_i$.

$\nabla_{\xi_i'} L = 0 \Rightarrow C = \alpha_i' + \beta_i'$.

Let $k(x, x') = \langle \Phi(x), \Phi(x') \rangle$ and $K_{ij} = k(x_i, x_j)$

sup inf $\mathcal{L}$ ... from above. ↳ kernel or Gram matrix

$$= \sup_{\alpha, \alpha'} -\frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} (\alpha_i - \alpha_i')(\alpha_j - \alpha_j') \langle \Phi(x_i), \Phi(x_j) \rangle$$
$$- \sum_{i=1}^{m} (\alpha_i - \alpha_i') y_i - \varepsilon \sum_{i=1}^{m} (\alpha_i + \alpha_i')$$

s.t. $0 \le \alpha_i \le C \wedge 0 \le \alpha_i' \le C \wedge \sum_{i=1}^{m} (\alpha_i - \alpha_i') = 0$

$$= \sup_{\alpha, \alpha'} -\varepsilon \langle \alpha + \alpha', 1 \rangle - \langle \alpha - \alpha', y \rangle - \frac{1}{2} \langle \alpha - \alpha', K(\alpha - \alpha') \rangle$$

s.t. $0 \le \alpha_i \le C \wedge 0 \le \alpha_i' \le C \wedge \langle \alpha - \alpha', 1 \rangle = 0$. for all $i \in [m]$

② Hypothesis $h^*(x) = \langle w^*, \Phi(x) \rangle + b$.

$$= \langle \sum_{i=1}^{m} (\alpha_i^* - \alpha_i'^*) \Phi(x_i), \Phi(x) \rangle + b^*$$
$$= -\sum_{i=1}^{m} (\alpha_i^* - \alpha_i'^*) k(x_i, x) + b^*$$

For b^*, we pick i_0 with $0 < \alpha_{i_0}^* < C$ or

$0 < \alpha_{i_0}'^* < C$.

In the former case, $\xi_{i_0} = 0$ and $\langle w^*, \Phi(x_{i_0}) \rangle + b^* - y_{i_0} - \varepsilon = 0$

by KKT. So, $b^* = y_{i_0} + \varepsilon + \sum_{i=1}^{m} (\alpha_i^* - \alpha_i'^*) k(x_i, x_{i_0})$.

We can derive b^* similarly in the latter case.

③ Note that only the i's with $\alpha_i^* - \alpha_i'^* \neq 0$ matter for h^*. So, x_i with such i plays the role of support vectors in the SVM algo.

If (x_i, y_i) is ε-close to h^*, then by KKT, $\alpha_i^* = \alpha_i'^* = 0$.

($|h^*(x_i) - y_i| = |\langle w^*, \Phi(x_i) \rangle + b^* - y_i| < \varepsilon$)

Thus, such input doesn't contribute to the def'n of h^*.

④ Compare this dual opti. pb. with the dual objective of KRR

SVR: $\max_{\alpha,\alpha'} \; -\varepsilon(\alpha+\alpha')^T 1 + (\alpha'-\alpha)^T y - \frac{1}{2}(\alpha'-\alpha) K (\alpha'-\alpha)$

s.t. $0 \le \alpha \le C \;\wedge\; 0 \le \alpha' \le C \;\wedge\; (\alpha'-\alpha)^T 1 = 0.$

($0 \le \alpha_i \le C$ for all $i \in [m]$).

KRR: $\max_{\alpha \in \mathbb{R}^m} \; 2\alpha^T y - \alpha^T (K+\lambda I)\alpha.$

$\alpha'-\alpha$ in SVR corresponds to α in KRR. So, the SVR objective has an extra term $-\varepsilon(\alpha+\alpha')^T 1$ and more constraints, but it also lacks the λI part. The missing λI part is due to the use of the L_1 norm when we compute the empirical loss. If we used the L_2 norm instead, that is, we use the following objective

$$\max_{w,b} \; \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m} |y_i - \langle w, \Phi(x_i)\rangle - b|_\varepsilon^2$$

then the dualization would give the following objective:

$$\max_{\alpha,\alpha'} \; -\varepsilon(\alpha+\alpha')^T 1 + (\alpha'-\alpha)^T y - \frac{1}{2}(\alpha'-\alpha)\left(K + \frac{1}{C} I\right)(\alpha'-\alpha).$$

s.t. $0 \le \alpha \;\wedge\; 0 \le \alpha' \;\wedge\; (\alpha'-\alpha)^T 1 = 0.$

Note that if $\varepsilon = 0$ and we ignore the $(\alpha'-\alpha)^T 1 = 0$ constraint, which appears due to the presence of the bias term b. (in our analysis of KRR, we ignored the bias term), then the above objective becomes identical to that of KRR. (modulo the correspondence between $\alpha'-\alpha$ in SVR and α in KRR).

(4). Generalization bounds. We ignore the bias term b here.

① SVR with the L_1 empirical loss.

Thm 11.13. $k: \mathcal{X}\times\mathcal{X} \to \mathbb{R}$ ... PDS kernel. (Gram matrix for k and S: $K_{ij} = k(x_i, x_j)$).

$\mathcal{H}$, Φ ... RKHS, feature map of k.

$\mathcal{H} = \{x \mapsto \langle w, \Phi(x)\rangle_{\mathcal{H}} \mid \|w\|_{\mathcal{H}} \le \Lambda\}$.

$\exists r>0, M>0$ s.t. $K(x,x) \le r^2$ for all $x\in\mathcal{X}$.

and $|y-y'| \le M$ for all $y, y' \in \mathcal{Y}$.

$D \in \Pr(\mathcal{X}\times\mathcal{Y})$. $\varepsilon > 0$. $\delta > 0$. $m \in \mathbb{N}$.

$$\Rightarrow \Pr_{S\sim D^m}\left[\forall h\in\mathcal{H}.\ \mathop{\mathbb{E}}_{(x,y)\sim D}[|h(x)-y|_\varepsilon] \le \frac{1}{m}\sum_{i=1}^m |h(x_i)-y_i|_\varepsilon + 2\sqrt{\frac{r^2\Lambda^2}{m}} + M\sqrt{\frac{\log 1/\delta}{2m}}\right] \ge 1-\delta$$

and

$$\Pr_{S\sim D^m}\left[\forall h\in\mathcal{H}.\ \mathop{\mathbb{E}}_{(x,y)\sim D}[|h(x)-y|_\varepsilon] \le \frac{1}{m}\sum_{i=1}^m |h(x_i)-y_i|_\varepsilon + \frac{2\Lambda\sqrt{\mathrm{Tr}[K]}}{m} + 3M\sqrt{\frac{\log 2/\delta}{2m}}\right] \ge 1-\delta$$

Pf. Let $L: \mathcal{Y}\times\mathcal{Y} \to [0,\infty)$ be $L(y,y') \overset{\text{def.}}{=} |y-y'|_\varepsilon$.

Then, L is 1-Lip and bounded by M. Also, by Thm 6.12,

$\widehat{\mathcal{R}}_S(\mathcal{H}) \le \frac{\Lambda\sqrt{\mathrm{Tr}[K]}}{m} \le \sqrt{\frac{\Lambda^2 r^2}{m}}$. So, by Thm 11.3,

we have the bounds in the theorem. □

② SVR with the L_2 empirical loss. Essentially by the same reasoning, we can get the following generalization bounds.

Thm 11.14. Same setup as the one for Thm 11.13. Then,

$$\Pr_{S\sim D^m}\left[\forall h\in\mathcal{H}.\ \mathop{\mathbb{E}}_{(x,y)\sim D}[|h(x)-y|_\varepsilon^2] \le \frac{1}{m}\sum_{i=1}^m |h(x_i)-y_i|_\varepsilon^2 + 4M\sqrt{\frac{r^2\Lambda^2}{m}} + M^2\sqrt{\frac{\log 1/\delta}{2m}}\right] \ge 1-\delta \text{ and.}$$

$$\Pr_{S\sim D^m}\left[\forall h\in\mathcal{H}.\ \mathop{\mathbb{E}}_{(x,y)\sim D}[|h(x)-y|_\varepsilon^2] \le \frac{1}{m}\sum_{i=1}^m |h(x_i)-y_i|_\varepsilon^2 + \frac{4M\Lambda\sqrt{\mathrm{Tr}[K]}}{m} + 3M^2\sqrt{\frac{\log 2/\delta}{2m}}\right] \ge 1-\delta.$$

Proof. The same proof as the one for Thm 11.13 except that the loss fn $L(y,y') = \max(0, |y-y'| - \varepsilon)^2$ is bound by M^2 and is $2M$-Lipschitz. R.

(5) Any problem with the SVR? Yes. First, we have to find appropriate values for the hyperparameters C and ε. Second, the algorithm and the learnt classifier are slow when m is large. The next algorithm, Lasso, addresses these issues at least partially, which computing sparse solutions.

7. Regression Algorithm 4 – Lasso.

(1) The key idea of Lasso is to use the L_1 regularization on weights, instead of L_2. Lasso solves the following opti. pb:

$$\min_{w,b} \lambda \|w\|_1 + \sum_{i=1}^{m} \left(\langle w, \Phi(x_i) \rangle + b - y_i \right)^2$$

$$\lambda \sum_{k=1}^{N} |w_k|$$

Solutions of Lasso tend to be sparse. That is, $\forall i \in [N]$ $|w_i^*|$ tends to be small. The reason is that the optimal values are often obtained at corners. See Fig 11.6 in the textbook.

(2) How to solve the above optimization problem? One issue is that $\|w\|_1$ is the sum of absolute values, and the absolute value fn is not as nice as polynomials. The next transformation provides a way to get rid of the absolute values.

$$\min_{w,b} \lambda \|w\|_1 + \sum_{i=1}^{m} \left(\langle w, \Phi(x_i) \rangle + b - y_i \right)^2.$$

$$= \min_{w^+ \geq 0, w^- \geq 0, b} \lambda \sum_{k=1}^{N} (w_k^+ + w_k^-) + \sum_{i=1}^{m} \left(\langle w^+ - w^-, \Phi(x_i) \rangle + b - y_i \right)^2.$$

The new optimization problem is an instance of so called quadratic programming, and it can be solved by an off-the-shelf solver.

(3) We can use a generalisation bound for Lasso again using Thm 11.3. But this time we want to bound the Rademacher complexity part in the conclusion of Thm 11.3 differently. We want the L_1 norm to feature there, instead of the L_2 norm. The next thm is crucial to achieve what we want. By the way, in this theoretical analysis of Lasso, we ignore the bias term b.

Thm 11.15: $S = (x_1, \ldots, x_m) \in \mathcal{X}^m$.

$\exists r_\infty > 0$ s.t. $\forall i \in [m]$. $\|\Phi(x_i)\|_\infty < r_\infty$

$\|\Phi(x_i)\|_\infty = \max_{j \in [N]} |\Phi(x_i)_j|$.

$\mathcal{H} = \{ x \mapsto \langle w, \Phi(x) \rangle \mid \|w\|_1 \leq \Lambda_1 \}$.

$\Rightarrow$

$$\widehat{R}_S(\mathcal{H}) \leq \sqrt{\frac{2 r_\infty^2 \Lambda_1^2 \log(2N)}{m}}.$$

Proof

$$\hat{R}_S(\mathcal{H}) = \frac{1}{m} \mathop{\mathbb{E}}_{\vec{\sigma} \in \mathrm{Unif}\{-1,1\}^m} \left[\sup_{h \in \mathcal{H}} \sum_{i=1}^m \sigma_i h(x_i) \right]$$

$$= \frac{1}{m} \mathop{\mathbb{E}}_{\vec{\sigma}} \left[\sup_{\|w\|_1 \le \Lambda_1} \sum_{i=1}^m \sigma_i \langle w, \Phi(x_i) \rangle \right]$$

$$= \frac{1}{m} \mathop{\mathbb{E}}_{\vec{\sigma}} \left[\sup_{\|w\|_1 \le \Lambda_1} \left\langle w, \sum_{i=1}^m \sigma_i \Phi(x_i) \right\rangle \right]$$

$$\le \frac{1}{m} \mathop{\mathbb{E}}_{\vec{\sigma}} \left[\sup_{\|w\|_1 \le \Lambda_1} \|w\|_1 \left\| \sum_{i=1}^m \sigma_i \Phi(x_i) \right\|_\infty \right]$$

$$= \frac{\Lambda_1}{m} \mathop{\mathbb{E}}_{\vec{\sigma}} \left[\max_{j \in [N]} \left| \sum_{i=1}^m \sigma_i \Phi(x_i)_j \right| \right]$$

$$= \frac{\Lambda_1}{m} \mathop{\mathbb{E}}_{\vec{\sigma}} \left[\max_{j \in [N]} \max_{s \in \{-1,+1\}} \sum_{i=1}^m \sigma_i \left(s \Phi(x_i)_j \right) \right]$$

Let $A = \left\{ \left(s\Phi(x_1)_j, s\Phi(x_2)_j, \cdots, s\Phi(x_m)_j \right) \mid s \in \{-1,+1\}, j \in [N] \right\}$.

$$= \frac{\Lambda_1}{m} \mathop{\mathbb{E}}_{\vec{\sigma}} \left[\sup_{z \in A} \sum_{i=1}^m \sigma_i z_i \right]$$

Since for any $z \in A$, $\|z\|_2 \le \sqrt{m r_\infty^2}$ and $|A| \le 2N$,

by Massart's lemma (Thm 3.7),

$$\le \frac{\Lambda_1}{m} \cdot \sqrt{m r_\infty^2} \sqrt{2 \log 2N} = \Lambda_1 r_\infty \sqrt{\frac{2 \log 2N}{m}} \quad \square$$

Thm 3.7. $A \subseteq \mathbb{R}^m$ finite set. $r = \max_{x \in A} \|x\|_2$.

$$\Rightarrow \mathop{\mathbb{E}}_{\vec{\sigma} \sim \mathrm{Unif}\{-1,1\}^m} \left[\frac{1}{m} \sup_{x \in A} \sum_{i=1}^m \sigma_i x_i \right] \le r/m \times \sqrt{2 \log |A|}$$

Thm 11.16 $\mathcal{H} = \{x \mapsto \langle w, \Phi(x) \rangle \mid \|w\|_1 \leq \Lambda_1\}$.

$\exists r_\infty > 0$ s.t. $\forall x \in \mathcal{X}$ $\|x\|_\infty \leq r_\infty$.

$\exists M > 0$ s.t. $\forall y, y' \in Y$ $|y - y'| \leq M$.

$D \in \mathcal{P}(X \times Y)$. $m \in \mathbb{N}$ $\delta > 0$.

$\Rightarrow$

$$\underset{S \sim D^m}{\mathbb{P}}\left[\forall h \in \mathcal{H},\ R(h) \leq \widehat{R}_S(h) + 4 r_\infty \Lambda_1 M \sqrt{\frac{2 \log 2N}{m}} + M^2 \sqrt{\frac{\log 1/\delta}{2m}}\right] \geq 1 - \delta.$$

$R(h) = \mathbb{E}_{(x,y) \sim D}[(h(x) - y)^2]$

Proof By Thm 11.3 and Thm 11.15. We use the fact that the loss fn $L(y, y') = (y - y')^2$ is bounded by M^2 and $2M$-Lipschitz. D.

(4) Any drawback for Lasso? First, it doesn't permit a natural use of PDS kernels. So, kernelizing it and generalizing it to the non-linear case are hard. Second, it doesn't admit a closed-form solution. This is not an issue in practice, but it makes the theoretical analysis of Lasso difficult.