Recap 5105

1. Learning Framework

→ Experience E + task T + Performance measure P

↓

① $\mathcal{D} = \mathcal{D}_{train} + \mathcal{D}_{validation} + \mathcal{D}_{test}$

② $\mathcal{D} = \mathcal{D}_{train} + \mathcal{D}_{test}$ → Cross-Validation

Related to Model Selection ← ①

↓

Another approach is: use AIC/BIC to penalize the likelihood and select model according to AIC/BIC

→ we only require: $\mathcal{D} = \mathcal{D}_{train} + \mathcal{D}_{test}$

---

2. Supervised Learning

✡

① Assumption: (strong form) deterministic

y_i is determined by x_i according to $y_i = f^*(x_i)$

$f^* \leftrightarrow$ oracle

(weak form) stochastic

$(X, Y) \sim P^*$ $\quad P^* \leftrightarrow$ 'Oracle'

⇕

y_i is a sample from $P^*(\cdot \mid x_i) \leftrightarrow y \sim P^*(\cdot \mid X)$

② Population Risk Minimization Versus Empirical Risk Minimization

ERM → attainable

$$\hat{f} \leftarrow \min_{f \in \mathcal{H}} \hat{R}_n(f, f^*) = \frac{1}{n} \sum_{i=1}^{n} L(f(x_i), f^*(x_i)) \qquad x_i \sim \mu$$

PRM → unattainable

$$\tilde{f} \leftarrow \min_{f \in \mathcal{H}} R(f, f^*) = \mathbb{E}_{x \sim \mu} [L(f(x), f^*(x))]$$

Decomposition

$$R(\hat{f}, f^*) - R(\tilde{f}, f^*)$$

$$= R(\hat{f}, f^*) - \hat{R}_n(\hat{f}, f^*) \rightarrow \text{uniform bound}$$

$$+ \hat{R}_n(\hat{f}, f^*) - \hat{R}_n(\tilde{f}, f^*) \rightarrow \text{optimization}$$

$$+ \hat{R}_n(\tilde{f}, f^*) - R(\tilde{f}, f^*) \rightarrow \text{point-wise bound}$$

③ Three Paradigms of Supervised Learning

Approximation	+	Optimization	+	Generalization
$f^* \leftrightarrow \tilde{f}$		$\hat{f}$		$\hat{f} \leftrightarrow f^*$

---

3. Linear Models

① Linear Regression

1. Model: $\mathcal{H} = \{ f : f(x) = w^T x + b \}$

2. Optimization: $\min_w \frac{1}{2N} \sum_{i=1}^{N} \| y_i - w^T x_i \|_2^2$

$\rightarrow \ell_2$-loss

$$\Rightarrow \hat{w} = (X^T X)^{-1} X^T y$$

Remarks: Actually there are other choices for regression problem

One example is Huber Loss $L(y, y') := \begin{cases} \frac{1}{2}(y-y')^2, & |y-y'| \le \delta \\ \delta|y-y'| - \frac{1}{2}\delta^2, & \text{o/w} \end{cases}$

② Linear Basis Model $\longrightarrow$ feature maps

1. Model: $\mathcal{H}_M = \{ f : f(x) = w^T \phi(x) \}$

$\phi : x \in \mathbb{R}^d \longrightarrow \phi(x) \in \mathbb{R}^M$ $\rightarrow$ pre-defined function

2. Optimization: $\min_w \frac{1}{2N} \sum_{i=1}^{N} \| y_i - w^T \phi(x) \|_2^2$.

$$\Rightarrow \hat{w} = (\Phi^T \Phi)^{-1} \Phi^T y$$

where $\Phi := \begin{pmatrix} \phi(x_1)^T \\ \vdots \\ \phi(x_N)^T \end{pmatrix} \in \mathbb{R}^{N \times m}$

3. When Φ is singular matrix $\longleftrightarrow$ $\det(\Phi^T \Phi) = 0$

$\downarrow$

Regularization

$$\min_w \frac{1}{2N} \sum_{i=1}^{N} \| y_i - w^T \phi(x_i) \|_2^2 + \lambda C(w)$$

choices of $C(w)$:

- l_1 – regularization: $C(w) = \|w\|_1$
- l_2 – regularization: $C(w) = \|w\|_2^2$

③ Linear Basis Model + Classification

1. Architecture

$$x \in \mathbb{R}^d \xrightarrow{\phi} \phi(x) \in \mathbb{R}^M \xrightarrow{W} W\phi(x) \in \mathbb{R}^K$$

2. Model

$$\mathcal{H} = \left\{ f : f(x) = g\left(\sum_{i=0}^{M-1} w_i \cdot \phi_i(x) \right),\ w_i \in \mathbb{R}^K \right\}$$

$$g(\cdot) \longrightarrow \boxed{\text{soft max}} \longrightarrow g_k(z) := \frac{\exp(z_k)}{\sum_i \exp(z_i)}$$

3. Optimization

$$\min_{W \in \mathbb{R}^{M \times K}} \frac{1}{N} \sum_{i=1}^{N} L\left(g(W^T \phi(x_i)), y_i \right)$$

Here, for classification task, we often apply cross-entropy loss

$$\longrightarrow \boxed{L(y, y') = -\sum_{k=1}^{K} y_k' \log y_k} \longrightarrow \text{non-symmetric}$$

Remarks: Actually, this comes from the KL-divergence, which can be viewed as the distance between 2 probability dist.

$$KL(p \| q) = \mathbb{E}_{x \sim p}\left[\log \frac{p(x)}{q(x)} \right]$$

$$= \int_x \log \frac{p(x)}{q(x)} \cdot p(x)\, dx$$

$$= \underbrace{\int_x p(x) \log p(x)\, dx}_{\text{Entropy of } p(x)} - \underbrace{\int_x p(x) \log q(x)\, dx}_{\text{cross-entropy term}}$$

---

4. Kernel Tricks

①. Kernel Ridge Regression

1. Model $\mathcal{H} = \{ f : f(x) = w^T \phi(x) \quad w \in \mathbb{R}^M \}$

2. Linear Basis Model + l_2-reg

$$\min_w \frac{1}{2N} \left(\sum_{i=1}^{N} \| y_i - w^T \phi(x_i) \|_2^2 + \lambda \| w \|_2^2 \right)$$

$$\Rightarrow \hat{w} = (\Phi^T \Phi + \lambda I_M)^{-1} \Phi^T y$$

Extension: Observation: $\hat{w} = (\Phi^T \Phi + \lambda I_M)^{-1} \Phi^T y$

$$= \Phi^T (\Phi \Phi^T + \lambda I_N)^{-1} y$$

Therefore, $\hat{f}(x) = \phi(x)^T \hat{w}$

$$= \phi(x)^T \Phi^T (\Phi \Phi^T + \lambda I_N)^{-1} y$$

$$\Phi \Phi^T = \begin{pmatrix} \phi(x_1) \\ \vdots \\ \phi(x_N) \end{pmatrix} (\phi(x_1)^T \cdots \phi(x_N)^T)$$

Gram Matrix

$$\Phi^T = (\phi(x_1), \ldots, \phi(x_N)) \quad \alpha = \begin{pmatrix} \alpha_1 \\ \vdots \\ \alpha_N \end{pmatrix}$$

$$= \sum_{i=1}^{N} \alpha_i \langle \phi(x_i), \phi(x) \rangle$$

$$= \sum_{i=1}^{N} \alpha_i \, k(x_i, x)$$

3. Kernel Trick

idea: feature maps can be computed expensively

↓

define kernel functions directly

Issue: what is a Valid Kernel Function ?

→ **Answer:** Mercer's Theorem (SPD Kernel)

$k: \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a valid kernel

$\Leftrightarrow$ for $\forall n \in \mathbb{N}^+$, $\forall \{x_1, \dots, x_n\}$ $x_i \in \mathbb{R}^d$

$(G)_{ij} = K(x_i, x_j)$ is symmetric + PSD

$$\to \begin{cases} G_{ij} = G_{ji} \\ \forall c \in \mathbb{R}^n, \; c^T G c \geq 0 \end{cases}$$

4. Re-formulation

a) choose a valid kernel $k(x, y)$

b) construct the Gram Matrix $G_{ij} = k(x_i, x_j)$

c) $\hat{f}(x) = \sum_{i=1}^{n} [(G + \lambda I_N)^{-1} y]_i \, k(x, x_i)$

5. Property of Kernel Function

→ if $K_1(\cdot,\cdot)$, $K_2(\cdot,\cdot)$ is valid kernel

then a) $k(x, y) = \lambda K_1(x, y)$

b) $k(x, y) = k_1(x, y) + k_2(x, y)$

c) $k(x,y) = g(x)\, k_1(x,y)\, g(y)$

d) $k(x,y) = \lim_{n\to\infty} k_n(x,y)$

e) $k(x,y) = k_1(x,y)\, k_2(x,y)$

all above is valid kernel

From these 5 properties, it is convenient to deduce that:

$$k(x,y) = \exp\left\{-\frac{1}{2\sigma^2}\|x-y\|_2^2\right\}$$

---

5. SVM ⟶ we only consider the linear separable case (hard margin)

① Primal formulation

a) linearly separable

$\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ is linearly separable

$\Leftrightarrow \exists\, w, b,$ s.t $y_i(w^T x_i + b) > 0$

margin

$$\begin{cases} \gamma_i^{\mathcal{D}}(w,b) = \dfrac{|w^T x_i + b|}{\|w\|} \\ \gamma^{\mathcal{D}}(w,b) = \min\limits_{i\in[N]} \gamma_i^{\mathcal{D}}(w,b) \end{cases}$$

b) ✡ formulation

$$\begin{cases} \max\limits_{w,b} \ \gamma^{\mathcal{D}}(w,b) \\ \text{s.t} \quad \text{'Hyperplane'} \quad \text{can separate those 2 classes} \end{cases}$$

$$\Longleftrightarrow \begin{cases} \max\limits_{w,b} & \gamma^g(w,b) \\ \text{s.t} & \underline{f(x)=w^Tx+b} \text{ can separate those 2 classes} \end{cases}$$

$$\Longleftrightarrow \begin{cases} \max\limits_{w,b} \min\limits_{i\in[N]} & \gamma_i^g(w,b) = \dfrac{|w^Tx_i+b|}{\|w\|} \\ \text{s.t} & y_i(w^Tx_i+b)>0 \end{cases}$$

$$\Longleftrightarrow \begin{cases} \max\limits_{w,b,\gamma} & \dfrac{1}{\|w\|}\cdot\gamma \\ \text{s.t} & y_i(w^Tx_i+b)>0 \end{cases}$$

$$\min_{i\in[N]} y_i(w^Tx_i+b)=\gamma$$

$$\Longleftrightarrow \begin{cases} \max\limits_{w,b,\gamma} & \dfrac{1}{\|w\|}\cdot\gamma \\ \text{s.t} & y_i(w^Tx_i+b)\geq\gamma \end{cases}$$

$$\rightsquigarrow \begin{cases} \max\limits_{w,b} & \dfrac{1}{\|w\|} \\ \text{s.t} & y_i(w^Tx_i+b)\geq 1 \end{cases}$$

$$\Longleftrightarrow \begin{cases} \min\limits_{w,b} & \frac{1}{2}w^Tw \\ \text{s.t} & y_i(w^Tx_i+b)\geq 1 \quad i\in[N] \end{cases} \qquad (*)$$

② KKT Condition & Duality

$$L(w,b;\lambda)=\frac{1}{2}w^Tw+\sum_{i=1}^{N}\lambda_i\left(1-y_i(w^Tx_i+b)\right)$$

the Dual Form is $\begin{cases} \max\limits_{\lambda} \min\limits_{w,b} & L(w,b;\lambda) \\ \text{s.t} & \lambda\geq 0 \end{cases}$ $(**)$

$$Q(\lambda):=\min_{w,b} L(w,b;\lambda)$$

→ since (*) is a convex program,

then we have :

$(\hat{w}, \hat{b}; \hat{\lambda})$ is the optimal solution with respect to Primal / Dual

$$\Leftrightarrow \begin{cases} \nabla_{w,b} L(\hat{w}, \hat{b}; \hat{\lambda}) = 0 & \rightarrow \text{Stationary} \\ \hat{\lambda}_i \geq 0 \\ 1 - y_i(\hat{w}^T x_i + \hat{b}) \leq 0 & \rightarrow \text{Feasibility} \\ \hat{\lambda}_i (1 - y_i(\hat{w}^T x_i + b)) = 0 & \rightarrow \text{Complemetary Slackness} \end{cases}$$

③ Dual Form of Hard Margin SVM

$$\begin{cases} \nabla_w L(w, b; \lambda) = w - \sum_{i=1}^{N} \lambda_i y_i x_i \\ \nabla_b L(w, b; \lambda) = - \sum_{i=1}^{N} \lambda_i y_i \end{cases}$$

$$\Rightarrow \begin{cases} \hat{w} = \sum_{i=1}^{N} \hat{\lambda}_i y_i x_i \\ \sum_{i=1}^{N} \hat{\lambda}_i y_i = 0 \end{cases}$$

Dual Form

$$\begin{cases} \max_{\lambda} \min_{w,b} L(w, b; \lambda) \\ s.t \quad \lambda \geq 0 \end{cases}$$

$$L(w, b; \lambda) = \frac{1}{2} w^T w + \sum_{i=1}^{N} \lambda_i (1 - y_i(w^T x_i + b))$$

$$\Leftrightarrow \begin{cases} \max_{\lambda} \quad \frac{1}{2} \| \sum_{i=1}^{N} \lambda_i y_i x_i \|_2^2 + \sum_{i=1}^{N} \lambda_i - w^T \sum_{i=1}^{N} \lambda_i y_i x_i \\ s.t \quad \lambda \geq 0 \\ \quad \sum_{i=1}^{N} \lambda_i y_i = 0 \end{cases}$$

$$\Longleftrightarrow \begin{cases} \max\limits_{\lambda} & -\frac{1}{2}\sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N}\lambda_i\lambda_j y_i y_j x_i^T x_j + \sum\limits_{i=1}^{N}\lambda_i \\ \text{s.t} & \lambda_i \geq 0 \quad i \in [N] \\ & \sum\limits_{i=1}^{N}\lambda_i y_i = 0 \end{cases}$$

when achieving $(\hat{\lambda}_1, \ldots, \hat{\lambda}_N)$, then

$$\begin{cases} \hat{w} = \sum\limits_{i=1}^{N}\hat{\lambda}_i y_i x_i \\ \text{for } \alpha \in SV,\ y_\alpha(\hat{w}^T x_\alpha + \hat{b}) = 1 \end{cases}$$

$$\Rightarrow \hat{w}^T x_\alpha + \hat{b} = y_\alpha$$

$$\Rightarrow \hat{b} = y_\alpha - \hat{w}^T x_\alpha$$

$$= y_\alpha - \sum_{i=1}^{N}\hat{\lambda}_i y_i x_i^T x_\alpha$$

to make prediction, $\hat{f}(x) = \text{sgn}(\hat{w}^T x + \hat{b})$

$$= \text{sgn}\left(\sum_{i=1}^{n}\hat{\lambda}_i y_i \langle x_i, x\rangle + \hat{b}\right)$$

④ Kernel Form

Recap: the dual formulation is

$$\begin{cases} \max\limits_{\lambda} & -\frac{1}{2}\sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N}\lambda_i\lambda_j y_i y_j x_i^T x_j + \sum\limits_{i=1}^{N}\lambda_i \\ \text{s.t} & \lambda_i \geq 0 \quad i \in [N] \\ & \sum\limits_{i=1}^{N}\lambda_i y_i = 0 \end{cases}$$

$$\longrightarrow \phi: x \in \mathbb{R}^d \longmapsto \phi(x) \in \mathbb{R}^D$$

$$\updownarrow$$

$$\phi(x_i)^T \phi(x_j) = \langle \phi(x_i), \phi(x_j)\rangle$$

$$= k(x_i, x_j)$$

$\Rightarrow$ Kenerl formulation :

$$\begin{cases} \max\limits_{\lambda} & -\frac{1}{2}\sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N}\lambda_i\lambda_j y_i y_j k(x_i,x_j) + \sum\limits_{i=1}^{N}\lambda_i \\ \text{s.t} & \lambda_i \geq 0 \quad i\in[N] \\ & \sum\limits_{i=1}^{N}\lambda_i y_i = 0 \end{cases}$$

$$\hat{w} = \sum_{i=1}^{N}\hat{\lambda}_i y_i \phi(x_i)$$

$$\Rightarrow \hat{f}(x) = \text{sgn}\left(\sum_{i=1}^{N}\hat{\lambda}_i y_i k(x_i,x) + \hat{b}\right)$$

where $\hat{b}$ is determined by $\hat{b} = y_\alpha - \sum\limits_{i=1}^{N}\hat{\lambda}_i y_i k(x_i,x_\alpha) \quad \alpha\in SV$

---

6. Decision Tree

① Model :

$$\mathcal{H} = \{ f : f(x) = \sum_{j=1}^{J} a_j \mathbb{1}_{R_j}(x),\ \{R_j\}_{j=1}^{J} \text{ is partition of } \mathcal{X} \}$$

② Optimization :

$$\begin{cases} \text{classification} & \min\limits_{R_j} \sum\limits_{j=1}^{J}\left\{\sum\limits_{k=1}^{K} -p(k|j)\log p(k|j)\right\} \\ \text{regression:} & \min\limits_{a_j, R_j} \frac{1}{2}\sum\limits_{i=1}^{N}\left(\sum\limits_{j=1}^{J} a_j \mathbb{1}_{R_j}(x_i) - y_i\right)^2 \end{cases}$$

Issue → minimizing all partition of $\mathcal{X}$ ⇒ Difficult / NP-Hard

$\Rightarrow$ Strategy : greedy algorithm (Recursive Binary Search)

③ Ad. and Dis-ad.

Advantages: 1. visualization

2. Implicit feature selection through the reduction of error / impurity

3. Non-linear relationships

4. Robust to data types

Disadvantages : 1. prone to over-fitting

2. sensitive to { data imbalancing
                 data variation

3. Greedy Algo → sub-optima

---

7. Ensemble { Bagging
              Boosting

a) Bagging (Bootstrap Aggregating)

↳ { regression → average
    classification → majority vote

Algo

1. $\mathcal{D} = \{(X_i, Y_i)\}_{i=1}^{N}$

2. for $j = 1, 2, \ldots . B$

   sample $\mathcal{D}_j$ from $\mathcal{D}$ with replacement

   (use ECDF to approximate unknown CDF)

   train f_j from $\mathcal{D}_j$

3. $\bar{f}(x) = \frac{1}{B}\sum_{b=1}^{B} f_b(x)$

Idea (Naive calculation)

assume $f_b(x) = f^*(x) + \varepsilon_b(x) \quad b = 1, 2, \dots, B$

where $\varepsilon_b(x) \rightarrow \begin{cases} \mathbb{E}[\varepsilon_b(x)] = 0 \\ \mathrm{Var}[\varepsilon_b(x)] = \sigma^2 \end{cases}$

then analyze the power of $\bar{f}(x) = \frac{1}{B}\sum_{b=1}^{B} f_b(x)$

① $\mathrm{MSE}(f_i) \rightarrow \mathbb{E}[(f_i(x) - f^*(x))^2]$

$= \mathbb{E}[(f_i(x) - \mathbb{E}[f_i(x)])^2]$

$= \mathrm{var}[f_i(x)]$

$= \sigma^2$

② $\mathrm{MSE}(\bar{f}) \rightarrow \mathbb{E}[(\bar{f}(x) - f^*(x))^2]$

$= \mathbb{E}[(\frac{1}{B}\sum_{b=1}^{B} f_b(x) - f^*(x))^2]$

$= \mathbb{E}[(\frac{1}{B}\sum_{b=1}^{B} \varepsilon_b(x))^2]$

$= \frac{1}{B^2}[\sum_{b=1}^{B} \varepsilon_b^2(x) + \sum_{i \neq j} \varepsilon_i(x)\varepsilon_j(x)]$

if incorrelated $\frac{1}{B} \cdot \sigma^2$

Remarks: In real application, this significant shrink of variance is not apparent. That is because, through sub-sampling, 2 things will happen:

1) # of sub-sampling large: highly correlated;

2) # of sub-sampling small: σ^2 is large.

b) Boosting (AdaBoost) → sequential model

idea: re-weight the data { correct → less weight; incorrect → larger weight

Formulation 1. Model (additive forward function)

$$F_m(x) = \sum_{i=1}^{m} \alpha_i f_i(x)$$

$$= \sum_{i=1}^{m-1} \alpha_i f_i(x) + \alpha_m f_m(x) =$$

$$= F_{m-1}(x) + \alpha_m f_m(x)$$

2. Loss function (exp loss)

$$L(f(x), y) = \exp\{-y f(x)\}$$

$$\Rightarrow L(F_m(x), y) = \exp\{-y F_m(x)\}$$

$$= \exp\{-y(F_{m-1}(x) + \alpha_m f_m(x))\}$$

$$= \exp\{-y F_{m-1}(x) - y \alpha_m f_m(x)\}$$

$$= \exp\{-y F_{m-1}(x)\} \exp\{-y \alpha_m f_m(x)\}$$

$$\text{Empirical Risk} = \frac{1}{N} \sum_{i=1}^{N} \exp\{-y_i F_{m-1}(x_i)\} \exp\{-y_i \alpha_m f_m(x_i)\}$$

3. Greedy Algo involved → (idea) Fix $F_{m-1}(x)$, find (α_m, f_m)

Define $w_i^{(m)} = \exp\{-y_i F_{m-1}(x_i)\}$

then Empirical Risk $= \frac{1}{N} \sum_{i=1}^{N} w_i^{(m)} \exp\{-y_i \alpha_m f_m(x_i)\}$

$$= \frac{1}{N} \left(\sum_{i \in C} w_i^{(m)} \exp\{-\alpha_m\} + \sum_{i \in IC} w_i^{(m)} \exp\{\alpha_m\} \right)$$

$$= \frac{1}{N} \sum_{i} w_i^{(m)} \exp\{-\alpha_m\} \cdot \mathbb{1}\{y_i = f_m(x_i)\}$$

$$+\frac{1}{N}\sum_i w_i^{(m)} \exp\{\alpha_m\}\, \mathbb{1}\{y_i \neq f_m(x_i)\}$$

$$= \underbrace{\frac{1}{N}\exp(-\alpha_m)\sum_{i=1}^{N} w_i^{(m)}}$$

$$+ \underbrace{\frac{1}{N}\left(\exp\{\alpha_m\} - \exp\{-\alpha_m\}\right)\sum_{i=1}^{N} w_i^{(m)}\, \mathbb{1}\{y_i \neq f_m(x_i)\}}$$

4. Summary

$$(\hat{f}_m, \hat{\alpha}_m) = \underset{f_m, \alpha_m}{\operatorname{argmin}} \text{ Empirical Risk on Exp Loss}$$

$$= \underset{f_m, \alpha_m}{\operatorname{argmin}} \exp(-\alpha_m)\sum_{i=1}^{N} w_i^{(m)} + \left[\exp(\alpha_m) - \exp(-\alpha_m)\right]\sum_{i=1}^{N} w_i^{(m)}\, \mathbb{1}\{y_i \neq f_m(x_i)\}$$

$$\Rightarrow \text{①}\quad \hat{f}_m = \underset{f_m}{\operatorname{argmin}} \sum_{i=1}^{N} w_i^{(m)}\, \mathbb{1}\{y_i \neq f_m(x_i)\}$$

$$\text{②}\quad \hat{\alpha}_m = \underset{\alpha_m}{\operatorname{argmin}} \exp\{\alpha_m\}\sum_{i \in IC} w_i^{(m)} + \exp\{-\alpha_m\}\sum_{i \in C} w_i^{(m)}$$

$$= \frac{1}{2}\log\left(\frac{1-\hat{\varepsilon}_m}{\hat{\varepsilon}_m}\right)$$

$$\hat{\varepsilon}_m := \frac{\sum_{i=1}^{N} w_i^{(m)}\, \mathbb{1}\{y_i \neq \hat{f}_m(x_i)\}}{\sum_{i=1}^{N} w_i^{(m)}}$$

---

8. Neural Network

- MLP → LEC5
- CNN → LEC6
- RNN → LEC7

Summary : As for supervised learning task, we mainly introduce

a) Linear Model

Linear Regression

Linear Basis Model { regression / classification

Regularization Technique

b) Kernel Trick

Actually Linear Basis Model can also achieve!

Ridge Regression (with l_2-regularization) + kernel form

→ exact solution can be determined by $K(x,y) = \langle \phi(x), \phi(y) \rangle$

→ exact predictive function

SVM + kernel

→ 1. optimization problem can be formulated by kernel function, instead of feature map;

2. predictive function can be expressed by kernel function;

c) SVM

Hard-margin SVM { PRIMAL / DUAL

Soft-margin SVM { PRIMAL / DUAL

Kernel SVM { PRIMAL / DUAL

d) Decision Tree { Regression / Classification

Trick: Greedy algorithm involved → recursive binary search

e) Ensemble

1. Bagging → Bootstrap the Dataset
↓
Aggregating through average

2. Boosting → Adaboost

→ Re-weight + training + voted average

f) Neural Network { FCNN / RNN / CNN