Machine Learning

Daniel Mao

Contents

Chapter 1

Formal Setup

1.1 Definitions

DEFINITION. Let $\mathcal{X}$ be the domain. Let Y be the label set. We define a **classifier**, denoted by h, to be a function from X to Y.

DEFINITION (Error)**.** Let $\mathcal{X}$ denote the domain. Let $\mathcal{D}$ be a probability distribution on X. Let f be the true labeling function on X. Let h be a classifier. We define the **error** of h, with respect to $\mathcal{D}$ and f, denoted by $\mathcal{L}_{\mathcal{D},f}$, to be a probability given by

$$\mathcal{L}_{\mathcal{D},f}(h) := \mathcal{D}\{x \in \mathcal{X} : h(x) \neq f(x)\}.$$

- We are never sure about the true error. We only approximate the true error.
- Note that there is no label set in this definition. What we have is the true labeling function f.

1.2 Empirical Risk Minimization (ERM)

DEFINITION (Empirical Error)**.** Let $\mathcal{S}$ be a training data. Let h be a classifier. We define the **empirical error**, with respect to $\mathcal{S}$, denoted by $L_{\mathcal{S}}$, to be a proportion given by

$$L_{\mathcal{S}}(h) := \frac{|\{i : h(x_i) \neq y_i\}|}{|S|}.$$

DEFINITION (The Realizability Assumption)**.** Assume that $\exists h^* \in \mathcal{H}$ such that $\mathcal{L}_{\mathcal{D},f}(h^*) = 0$.

THEOREM 1.1. Assume that the data are independent and identically distributed. Assume realizability. Then if $\mathcal{H}$ is a finite class of classifiers, the $ERM(H)$ algorithm is guaranteed to succeed.

Intuitively, the training set S is a window through which the learner gets partial information about the distribution D over the world and the labeling function, f. The larger the sample gets, the more likely it is to reflect more accurately the distribution and labeling used to generate it.

Proof Idea. Given some error rate ε and a sample of size m, what is the probability that $ERM(H)$ will have error ε.

$$\Pr\big(\mathcal{L}_{D,f}(A[S]) > \varepsilon\big).$$

Let's call a sample S "BAD" if $\exists h \in H$ such that $L_S(h) = 0$ but $L_{\mathcal{D},f}(h) > \varepsilon$. Let's show that the probability of picking a BAD sample is small. For every given h, such that $L_{D,f}(h) > \varepsilon$, the probability of picking x on which h is correct is $\leq 1-\varepsilon$. So by independence,

$$D^m\{S : S \text{ is BAD w.r.t. this } h\} \leq (1-\varepsilon)^m.$$

Given a BAD h, the probability of S failing to show that this h is BAD $\leq (1-\varepsilon)^m$. The probability that S is BAD, S fails any $h \in H$, by the Union Bound, is $\leq \sum_{BAD h \in H} \Pr[S fails to alert on h]$. $\leq |H| \cdot (1-\varepsilon)^m$.

Conclusion

The prob of picking an S that will mislead $ERM(H)$ to pick some h with true error $> \varepsilon$ is at most $|H| \cdot (1-\varepsilon)^m$. So

$$\Pr\big(\mathcal{L}_{\mathcal{D},f}(ERM(H)(S)) > \varepsilon\big) < |H| \cdot (1-\varepsilon)^m \overset{m\to\infty}{\longrightarrow} 0.$$

■

Formal Proof. Let $\mathcal{X}$ denote the domain. Let $\mathcal{Y}$ denote the labeling set. Let $\mathcal{D}$ be some probability distribution on $\mathcal{X}$. Let f be some true labeling function from $\mathcal{X}$ to $\mathcal{Y}$. Let $\mathcal{H}$ be a finite class of classifiers. Define for each sample S a classifier h_S as $h_S := \operatorname{argmin}_{h\in\mathcal{H}} \mathcal{L}_{S,f}(h)$. The realizability assumption implies that $\mathcal{L}_{S,f}(h_S) = 0$. Define a set $\mathcal{H}_B$ as

$$\mathcal{H}_B := \{h \in \mathcal{H} : \mathcal{L}_{\mathcal{D},f}(h) > \varepsilon\}.$$

i.e., $\mathcal{H}_B$ is the set of bad hypotheses. Define a set M as

$$M := \{S \in \mathcal{X}^m : \exists h \in \mathcal{H}, \mathcal{L}_{\mathcal{D},f}(h) > \varepsilon \text{ and } \mathcal{L}_{S,f}(h) = 0\}.$$

i.e., M is the set of misleading samples. Note that

$$\{S \in \mathcal{X}^m : \mathcal{L}_{\mathcal{D},f}(h_S) > \varepsilon\} \subseteq M.$$

So

$$\begin{aligned}
\mathcal{D}^m\{S \in \mathcal{X}^m : \mathcal{L}_{\mathcal{D},f}(h_S) > \varepsilon\} &\leq \mathcal{D}^m(M) \\
&= \mathcal{D}^m \bigcup_{h\in\mathcal{H}_B} \{S \in \mathcal{X}^m : \mathcal{L}_{S,f}(h) = 0\} \\
&\leq \sum_{h\in\mathcal{H}_B} \mathcal{D}^m\{S \in \mathcal{X}^m : \mathcal{L}_{S,f}(h) = 0\} \\
&= \sum_{h\in\mathcal{H}_B} \mathcal{D}^m\{S \in \mathcal{X}^m : \forall x \in S, h(x) = f(x)\} \\
&= \sum_{h\in\mathcal{H}_B} \mathcal{D}^m \prod_{i=1}^{m}\{x \in \mathcal{X} : h(x) = f(x)\} \\
&= \sum_{h\in\mathcal{H}_B} \prod_{i=1}^{m} \mathcal{D}\{x \in \mathcal{X} : h(x) = f(x)\} \\
&= \sum_{h\in\mathcal{H}_B} \prod_{i=1}^{m} (1 - \mathcal{L}_{\mathcal{D},f}(h)) \\
&\leq \sum_{h\in\mathcal{H}_B} \prod_{i=1}^{m} (1 - \varepsilon) \\
&= \sum_{h\in\mathcal{H}_B} (1 - \varepsilon)^m \\
&= |\mathcal{H}|(1 - \varepsilon)^m.
\end{aligned}$$

■

Chapter 2

Uniform Convergence

2.1 Definitions

DEFINITION (ε Representative)**.** We say a sample S is ε-representative, with respect to H and p, if

$$\forall h \in H, \quad |L_p(h) - L_S(h)| < \varepsilon.$$

Intuitively, this property tells you that you can trust on sample S on estimating p for any h in H.

DEFINITION (Uniform Convergence Property)**.** Let Z be the domain set. We say that a class H has the **uniform convergence property** if $\exists m : (0,1)^2 \to \mathbb{N}$ such that for any probability distribution p over $Z = X \times Y$, for any $m \geq m_0 = m(\varepsilon, \delta)$, we have

$$\Pr_{S \sim p^m} \left[S \text{ is not } \varepsilon\text{-representative with respect to } H \text{ and } p \right] < \delta.$$

A class has the uniform convergence property if when I take samples which are big enough, the probability that it is not an representative is small, no matter what p is.

PROPOSITION 2.1.1. If a sample S is $\frac{\varepsilon}{2}$-representative with respect to p and a class H, then if A is an ERM_H function, we get

$$L_p(A(S)) \leq \min_{h\in H}\{L_p(h)\} + \varepsilon.$$

This connects being representative to the success of an ERM algorithm.

Proof.

$$\begin{aligned} L_p(A(S)) &\leq L_S(A(S)) + \varepsilon/2, \text{ since } S \text{ is } \frac{\varepsilon}{2}\text{-representative} \\ &= \min_{h\in H}\{L_S(h)\} + \varepsilon/2 \\ &\leq \min_{h\in H}\{L_p(h) + \varepsilon/2\} + \varepsilon/2, \text{ since } S \text{ is } \frac{\varepsilon}{2}\text{-representative again} \\ &= \min_{h\in H}\{L_p(h)\} + \varepsilon. \end{aligned}$$

That is,

$$L_p(A(S)) \leq \min_{h\in H}\{L_p(h)\} + \varepsilon.$$

■

PROPOSITION 2.1.2. If a class H has the uniform convergence property, then H is agnostically PAC learnable. Further more, any ERM_H function will be a successful learner in such a case.

THEOREM 2.1. Every finite H has the uniform convergence property.

Proof. Step 1: Showing that for any fixed h, we can find $m(\varepsilon, \delta)$ that guarantees $|L_p(h) - L_S(h)| < \varepsilon$.

Let θ_i denote the loss $\ell(h, (x_i, h_I))$. Then $L_S(h) = \frac{1}{m}\sum_{i=1}^m \theta_i$. Let μ denote $L_p(h)$. By the Hoeffding's inequality, we have

$$\Pr(|L_S(h) - L_p(h)| > \varepsilon) \leq 2\exp(-2m\varepsilon^2).$$

For any fixed h, large enough samples are likely to be ε-representative for that H.
Step 2: Using the union bound we extend this to every finite H. ■

Corollary. *Every finite H is agnostically PAC learnable.*

Chapter 3

Vapnik–Chervonenkis Dimension

3.1 Definitions

3.2 Examples

EXAMPLE 3.2.1. $\mathrm{VCdim}(H_{\text{thresholds}}) = 1$.

EXAMPLE 3.2.2. $\mathrm{VCdim}(H_{\text{intervals}}) = 2$.

EXAMPLE 3.2.3. $\mathrm{VCdim}(H^2_{\text{rectangles}}) = 4$.

3.3 Properties

PROPOSITION 3.3.1. Let H be a hypothesis class. Then $|H| \leq 2^{\mathrm{VCdim}(H)}$.

PROPOSITION 3.3.2 (Monotonicity)**.** Let H_1 and H_2 be two hypothesis classes. Suppose that $H_1 \subseteq H_2$. Then $\mathrm{VCdim}(H_1) \leq \mathrm{VCdim}(H_2)$.

Chapter 4

Probably Approximately Correct Learning

4.1 Definition

4.2 No-Free-Lunch Theorem

THEOREM 4.1 (No-Free-Lunch Theorem)**.** Let $\mathcal{X}$ be a finite domain. Let $\mathcal{H}$ be the class of all functions from $\mathcal{X}$ to $\{0,1\}$. Then $m_{\mathcal{H}}(\frac{1}{8},\frac{1}{8}) \geq \frac{|\mathcal{X}|}{2}$. i.e., No algorithm can succeed in PAC learning H for $\varepsilon = \frac{1}{8}$ and $\delta = \frac{1}{8}$ using less than $\frac{|\mathcal{X}|}{2}$ training samples.

4.3 Relation to Vapnik–Chervonenkis Dimension

PROPOSITION 4.3.1. Let H be a hypothesis class. If $\mathrm{VCdim}(H) = \infty$, then H is not PAC learnable.

Proof. Assume for the sake of contradiction that $\mathcal{H}$ is PAC learnable. Then there is some function $m_{\mathcal{H}} : (0,1) \times (0,1) \to \mathbb{N}$ and some learner A such that for every probability distribution $\mathcal{D}$ over $\mathcal{X}$, $\forall f \in \mathcal{H}$, $\forall \varepsilon > 0$, $\forall \delta > 0$, $\forall m > m_{\mathcal{H}}(\varepsilon, \delta)$, we have

$$\Pr_{S \sim \mathcal{D}^m, f}[L_{P,f}(A(S)) > \varepsilon] < \delta.$$

Define $m_0 := m_{\mathcal{H}}(\frac{1}{8},\frac{1}{8})$. Since $\mathrm{VCdim}(H) = \infty$, $\exists W \subseteq \mathcal{X}$ such that $|W| > 2m_0$ that can be shattered by H. Then $\mathcal{H}$ induces all possible function from W from $\{0,1\}$. However, by the NFL theorem, in such case, $m_0 \geq \frac{|W|}{2} > m_0$, which is a contradiction. So $\mathcal{H}$ is not PAC learnable. ■

4.4 The Fundamental Theorem of Statistical Learning

THEOREM 4.2 (The Fundamental Theorem of Statistical Learning)**.** For ever domain $\mathcal{X}$ and every class $\mathcal{H}$ of functions from $\mathcal{X}$ to $\{0,1\}$, the following statements are equivalent.

(1) $\mathcal{H}$ has the uniform convergence property.

(2) ERM is a successful agnostic PAC learner for $\mathcal{H}$.

(3) $\mathcal{H}$ is agnostic PAC learnable.

(4) ERM is a successful PAC learner for $\mathcal{H}$.

(5) $\mathcal{H}$ is PAC learnable.

(6) $\mathcal{H}$ has finite VC dimension.

Proof. **Proof of (1) $\implies$ (2)**.

proved in lecture 02.

■

Chapter 5

Perceptron

5.1 The Perceptron Algorithm

Algorithm 1: The Perceptron Algorithm (Rosenblatt 1958)

Input: Dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i) \in \mathbb{R}^d \times \{\pm 1\} : i = 1..n\}$, initialization $\boldsymbol{w} \in \mathbb{R}^d$ and $b \in \mathbb{R}$, threshold $\delta \geq 0$.

Output: Approximate solutions $\boldsymbol{w}$ and b so that $\forall i \in \{1..n\}$, $y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) > 0$.

```
while true do
    receive training example index $i \in \{1..n\}$;
    if $y_i(\boldsymbol{w} \cdot \boldsymbol{x}_i + b) \leq \delta$ then
        $\boldsymbol{w} \leftarrow \boldsymbol{w} + y_i \boldsymbol{x}_i$;
        $b \leftarrow b + y_i$;
```

Remark. *We only perform updates when we make a mistake, and this is a necessary rule.*

THEOREM 5.1. Assume there exists some $\boldsymbol{z}$ such that $A^\top \boldsymbol{z} > 0$, then the perceptron algorithm converges to some $\boldsymbol{z}^*$. If each column of A is selected indefinitely often, then $A^\top \boldsymbol{z}^* > \delta$.

Corollary. *Let $\delta = 0$ and $\boldsymbol{z}_0 = \mathbf{0}$. Then the perceptron algorithm converges after at most $(R/\gamma)^2$ steps, where R and γ are dataset properties given by*

$$R := \|A\|_{2,\infty} = \max_i \|\boldsymbol{a}_i\|_2 \quad \textit{and} \quad \gamma := \max_{\|z\|_2 \leq 1} \min_i \langle \boldsymbol{z}, \boldsymbol{a}_i \rangle.$$

Note that the parameters R and γ are independent of the dataset size n.

THEOREM 5.2. The iterate $\boldsymbol{z} = (\boldsymbol{w}; b)$ of the perceptron algorithm is always bounded. In particular, if there is no separating hyperplane, then perceptron cycles.

Chapter 6

Logistic Regression

6.1 Logit Transform

DEFINITION (Logit Transform)**.**

$$\log \frac{p(\boldsymbol{x};\boldsymbol{w})}{1-p(\boldsymbol{x};\boldsymbol{w})} = \boldsymbol{w}^\top \boldsymbol{x}.$$

Or equivalently,

$$p(\boldsymbol{x};\boldsymbol{w}) = \frac{1}{1+\exp(-\boldsymbol{w}^\top \boldsymbol{x})}.$$

This $p(\boldsymbol{x};\boldsymbol{w})$ is the confidence in the prediction.

6.2 Hypothesis Representation

Logistic Regression Model

Want $0 \le h_\theta(x) \le 1$.

$$h_\theta(x) = g(\theta^\top x)$$

where

$$g(z) = \frac{1}{1+e^{-z}}.$$

So

$$h_\theta(x) = \frac{1}{1+e^{-\theta^\top x}}.$$

Sigmoid function (logistic function)

$$g(z) = \frac{1}{1+e^{-z}}.$$

Properties:

- $g(0) = 0.5$.
- $\lim_{z\to+\infty} g(z) = 1$.
- $\lim_{z\to-\infty} g(z) = 0$.

Interpretation of the Hypothesis Output

$h_\theta(x)$ is the probability that $y = 1$ on input x. i.e. $h_\theta(x) = P(y = 1 \mid x;\theta)$, the probability that $y = 1$, given x, parameterizad by θ.

Since y is either 0 or 1,

$$\begin{aligned} &P(y=0 \mid x;\theta) + P(y=1 \mid x;\theta) = 1, \text{ or} \\ &P(y=0 \mid x;\theta) = 1 - P(y=1 \mid x;\theta). \end{aligned}$$

6.3 Decision Boundary

PROPOSITION 6.3.1. The decision boundary $\mathcal{D}$ of logistic regression with parameter $\boldsymbol{w} \in \mathbb{R}^n$ is

$$\mathcal{D} = \{x \in \mathbb{R}^n : \boldsymbol{w}^\top \boldsymbol{x} = 0\}.$$

Proof. Predict $y = 1$ if $h_{\boldsymbol{w}}(x) \geq 0.5$ and predict $y = 0$ if $h_{\boldsymbol{w}}(x) < 0.5$. Notice

$$h_{\boldsymbol{w}}(x) = 0.5 \iff \boldsymbol{w}^\top x = 0.$$

So the model predicts $\hat{y} = \operatorname{sign}(\boldsymbol{w}^\top \boldsymbol{x})$ but with confidence $p(\boldsymbol{x};\boldsymbol{w})$. ■

EXAMPLE 6.3.1. Consider $\theta = \begin{bmatrix} -3 \\ 1 \\ 1 \end{bmatrix}$. Predict "$y = 1$" if $\theta^\top x = -3 + x_1 + x_2 \geq 0$. That is, $x_2 \geq 3 - x_1$. This gives a "half space" in $\mathbb{R}^2$. The line $x_1 + x_2 = 3$ separates $\mathbb{R}^2$ into a region where the model predicts "$y = 0$" and a region where the model predicts "$y = 1$".

6.4 Maximum Likelihood Estimation

PROPOSITION 6.4.1 (Maximum Likelihood Estimation of the Parameter $\boldsymbol{w}$)**.** The maximum likelihood estimation of $\boldsymbol{w}$ is given by the following update rule:

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - (\boldsymbol{X}\boldsymbol{W}\boldsymbol{X}^\top)^{-1}\boldsymbol{X}^\top(\boldsymbol{y} - \boldsymbol{p})$$

where

$$\boldsymbol{p} = \left[\frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_i)}{1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_i)}\right]_{i=1}^n \qquad \text{and} \qquad \boldsymbol{W} = \operatorname{diag}(p_i(1-p_i))_{i=1}^n.$$

Proof. The maximum likelihood function $\mathcal{L}$ is:

$$\begin{aligned}
\mathcal{L}(\boldsymbol{x}_1, ..., \boldsymbol{x}_n; \boldsymbol{w}) &= \prod_{i=1}^n f_i(\boldsymbol{x}_i; \boldsymbol{w}) \\
&= \prod_{i=1}^n \begin{cases} \Pr(y = 1 \mid \boldsymbol{X} = \boldsymbol{x}_i), & \text{if } y_i = 1 \\ \Pr(y = 0 \mid \boldsymbol{X} = \boldsymbol{x}_i), & \text{if } y_i = 0 \end{cases} \\
&= \prod_{i=1}^n \big[\Pr(y = 1 \mid \boldsymbol{X} = \boldsymbol{x}_i)^{y_i} \Pr(y = 0 \mid \boldsymbol{X} = \boldsymbol{x}_i)^{1-y_i}\big].
\end{aligned}$$

So the log maximum likelihood function ℓ is:

$$\begin{aligned}
\ell(\boldsymbol{x}_1, ..., \boldsymbol{x}_n; \boldsymbol{w}) &= \log \mathcal{L}(\boldsymbol{x}_1, ..., \boldsymbol{x}_n; \boldsymbol{w}) \\
&= \log \prod_{i=1}^n \big[\Pr(y = 1 \mid \boldsymbol{X} = \boldsymbol{x}_i)^{y_i} \Pr(y = 0 \mid \boldsymbol{X} = \boldsymbol{x}_i)^{1-y_i}\big] \\
&= \sum_{i=1}^n \log \big[\Pr(y = 1 \mid \boldsymbol{X} = \boldsymbol{x}_i)^{y_i} \Pr(y = 0 \mid \boldsymbol{X} = \boldsymbol{x}_i)^{1-y_i}\big] \\
&= \sum_{i=1}^n \big[y_i \log \Pr(y = 1 \mid \boldsymbol{X} = \boldsymbol{x}_i) + (1-y_i) \log \Pr(y = 0 \mid \boldsymbol{X} = \boldsymbol{x}_i)\big] \\
&= \sum_{i=1}^n \left[y_i \log \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_i)}{1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_i)} + (1-y_i) \log \frac{1}{1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_i)}\right] \\
&= \sum_{i=1}^n \Big[y_i \big[\boldsymbol{w}^\top \boldsymbol{x}_i - \log(1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_i))\big] - (1-y_i)\big[\log(1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_i))\big]\Big] \\
&= \sum_{i=1}^n \Big[y_i \boldsymbol{w}^\top \boldsymbol{x}_i - \log(1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_i))\Big].
\end{aligned}$$

So the derivative of ℓ, with respect to $\boldsymbol{w}$, is:

$$\frac{\partial}{\partial \boldsymbol{w}} \ell(\boldsymbol{x}_1, ..., \boldsymbol{x}_n; \boldsymbol{w}) = \frac{\partial}{\partial \boldsymbol{w}} \sum_{i=1}^n \Big[y_i \boldsymbol{w}^\top \boldsymbol{x}_i - \log(1+\exp(\boldsymbol{w}^\top \boldsymbol{x}_i))\Big]$$

$$
\begin{aligned}
&= \sum_{i=1}^{n} \left[\frac{\partial}{\partial \boldsymbol{w}} y_i \boldsymbol{w}^\top \boldsymbol{x}_i - \frac{\partial}{\partial \boldsymbol{w}} \log(1 + \exp(\boldsymbol{w}^\top \boldsymbol{x}_i)) \right] \\
&= \sum_{i=1}^{n} \left[y_i \boldsymbol{x}_i^\top - \frac{1}{1 + \exp(\boldsymbol{w}^\top \boldsymbol{x}_i)} \exp(\boldsymbol{w}^\top \boldsymbol{x}_i) \boldsymbol{x}_i^\top \right] \\
&= \sum_{i=1}^{n} \left[y_i - \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_i)}{1 + \exp(\boldsymbol{w}^\top \boldsymbol{x}_i)} \right] \boldsymbol{x}_i^\top \\
&= \boldsymbol{X}^\top (\boldsymbol{y} - \boldsymbol{p}).
\end{aligned}
$$

So the second derivative of ℓ, with respect to $\boldsymbol{w}$, is:

$$
\begin{aligned}
\frac{\partial^2}{\partial \boldsymbol{w} \partial \boldsymbol{w}^\top} \ell(\boldsymbol{x}_1, ..., \boldsymbol{x}_n; \boldsymbol{w}) &= \frac{\partial}{\partial \boldsymbol{w}^\top} \frac{\partial}{\partial \boldsymbol{w}} \ell(\boldsymbol{x}_1, ..., \boldsymbol{x}_n; \boldsymbol{w}) \\
&= \frac{\partial}{\partial \boldsymbol{w}^\top} \sum_{i=1}^{n} \left[y_i - \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_i)}{1 + \exp(\boldsymbol{w}^\top \boldsymbol{x}_i)} \right] \boldsymbol{x}_i^\top \\
&= -\sum_{i=1}^{n} \frac{\partial}{\partial \boldsymbol{w}^\top} \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_i)}{1 + \exp(\boldsymbol{w}^\top \boldsymbol{x}_i)} \boldsymbol{x}_i^\top \\
&= -\sum_{i=1}^{n} \frac{\exp(-\boldsymbol{w}^\top \boldsymbol{x}_i) \boldsymbol{x}_i}{(1 + \exp(-\boldsymbol{w}^\top \boldsymbol{x}_i))^2} \boldsymbol{x}_i^\top \\
&= -\sum_{i=1}^{n} \boldsymbol{x}_i \boldsymbol{x}_i^\top \frac{\exp(\boldsymbol{w}^\top \boldsymbol{x}_i)}{1 + \exp(\boldsymbol{w}^\top \boldsymbol{x}_i)} \frac{1}{1 + \exp(\boldsymbol{w}^\top \boldsymbol{x}_i)} \\
&= \boldsymbol{X} \boldsymbol{W} \boldsymbol{X}^\top.
\end{aligned}
$$

So the update rule is:

$$
\begin{aligned}
\boldsymbol{w}_{k+1} &= \boldsymbol{w}_k - \left(\frac{\partial^2 \ell}{\partial \boldsymbol{w} \partial \boldsymbol{w}^\top} \right)^{-1} \left(\frac{\partial \ell}{\partial \boldsymbol{w}} \right) \\
&= \boldsymbol{w}_k - (\boldsymbol{X} \boldsymbol{W} \boldsymbol{X}^\top)^{-1} \boldsymbol{X}^\top (\boldsymbol{y} - \boldsymbol{p}).
\end{aligned}
$$

∎

6.5 Cost Function

Training set: $\left\{ (x^{(1)}, y^{(1)}), ..., (x^{(m)}, y^{(m)}) \right\}$. $x_0 = 1$. $x = \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^{n+1}$. $y \in \{0, 1\}$.

If

$$
cost(h_\theta(x^{(i)}, y^{(i)})) = \frac{1}{2}(h_\theta(x^{(i)}) - y^{(i)})^2
$$

as in linear regression, then the cost function is non-convex.

Logistic Regression Cost Function

$$cost(h_\theta(x), y) := \begin{cases} -\log(h_\theta(x)), & \text{if } y = 1 \\ -\log(1 - h_\theta(x)), & \text{if } y = 0. \end{cases}$$

Property:

- If $y = 1$ and $h_\theta(x) \uparrow 1$, then $cost(h_\theta(x), y) \approx 0$.
- If $y = 1$ and $h_\theta(x) \downarrow 0$, then $cost(h_\theta(x), y) \to +\infty$.
- Similar for $y = 0$.

Captures intuition that if $h_\theta(x) = 0$ (the model predicts $P(y = 1 \mid x; \theta) = 0$), but $y = 1$, we'll penalize the learning algorithm by a very large cost.

Cost Function for the Training Set

$$J(\theta) := \frac{1}{m} \sum_{i=1}^{m} cost(h_\theta(x^{(i)}, y^{(i)}))$$

where *cost* is the function given above.

Rewrite the cost function:

$$cost(h_\theta(x), y) = -y \log(h_\theta(x)) - (1 - y) \log(1 - h_\theta(x)).$$

So

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} \left[y^{(i)} \log\left(h_\theta(x^{(i)})\right) + (1 - y^{(i)}) \log\left(1 - h_\theta(x^{(i)})\right) \right].$$

To fit parameter θ: $\min_\theta J(\theta)$.

6.6 Optimization

Gradient Descent

$$\frac{\partial}{\partial \theta_j} J(\theta) = \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) x_j^{(i)}.$$

This looks identical to linear regression.

Other Optimization Algorithms

- Conjugate Gradient
- BFGS

- L-BFGS

Advantages:

- No need to pick learning rate α.
- Often faster than gradient descent.

6.7 Multiclass Classification

One-vs-rest

$$h_\theta^{(i)}(x) = P(y = i \mid x; \theta) \quad i = 1, 2, 3.$$

Train a logistic regression classifier for each i to predict the probability that $y = i$. On a new input x, pick class $i = \underset{i}{\operatorname{argmax}} h_\theta^{(i)}(x)$.

Softmax

$$\mathbb{P}(Y = k \mid \boldsymbol{x}; \boldsymbol{W}) = \frac{\exp(\boldsymbol{w}_k^\top \boldsymbol{x})}{\sum_{l=1}^C \exp(\boldsymbol{w}_l^\top \boldsymbol{x})}.$$

6.8 Introducing Non-linearity

Consider

$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

and $\theta = \begin{bmatrix} -1 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix}$. Then the model predicts “$y = 1$” if $-1 + x_1^2 + x_2^2 \geq 0$. That is, if $x_1^2 + x_2^2 \geq 1$.

6.9 Beyond Logistic Regression

The logit transform is just one choice among the others. What we need is a function Q from $[0, 1]$ onto $\mathbb{R}$. The inverse of any cumulative distribution function can be used as the Q function. e.g. the probit regression which uses the inverse of the CDF of a standard normal distribution.

DEFINITION (Logistic Distribution)**.** We define the **logistic distribution** to be a probability distribution with CDF given by

$$F(x; \mu; s) := \frac{1}{1 + \exp(-\frac{x-\mu}{s})}.$$

PROPOSITION 6.9.1 (Mean of Logistic Distribution)**.** Let $X \sim F(x;\mu;s)$. Then

$$\mathbb{E}[X] = \mu.$$

PROPOSITION 6.9.2. Let $X \sim F(x;\mu;s)$. Then

$$\mathrm{var}[X] = \frac{s^2\pi^2}{3}.$$

Chapter 7

Dimensionality Reduction

7.1 Motivations

- Data compression.
- Data visualization.

7.2 Principal Component Analysis

PCA is trying to find a lower dimensional surface onto which to project the data so as to minimize squared projection error.

Principal component analysis vs. linear regression:

- Linear regression tries to minimize residuals. PCA tries to minimize projection error.
- In linear regression, we try to predict one feature with other features. In PCA, all features are equivalent.

Algorithm: (from n dimensions to k dimensions)

- Preprocessing: mean normalization and feature scaling (standardization).
- Compute covariance matrix
$$\Sigma = \frac{1}{m}\sum_{i=1}^{m}(x^{(i)})(x^{(i)})^T.$$
- Compute the eigenvalues of Σ.
$$\Sigma = USV.$$
Let U_{reduce} denote the first k columns of U. Then $z = U_{reduce}^T x$.

Choosing k:

- Average squared projection error:

$$\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)} - x^{(i)}_{approx}\|^2.$$

- Total variation in the data:

$$\frac{1}{m}\sum_{i=1}^{m}\|x^{(i)}\|^2.$$

- Typically, we choose k to be the smallest value so that

$$\frac{\text{average squared projection error}}{\text{total variation}} < 0.01,$$

  i.e. 99% of variance is retained.

- The above ratio can be computed in an easier way. Say

$$S = \begin{bmatrix} s_{11} & & O \\ & \ddots & \\ O & & s_{nn} \end{bmatrix}.$$

  Then

$$\frac{\text{average squared projection error}}{\text{total variation}} = 1 - \frac{\sum_{i=1}^{k} s_{ii}}{\sum_{i=1}^{n} s_{ii}}.$$

  So we can start with $k = 1$ and increase k and check whether

$$\frac{\sum_{i=1}^{k} s_{ii}}{\sum_{i=1}^{n} s_{ii}} > 0.99.$$

Chapter 8

Support Vector Machines

Support Vector Machines are also called the maximum-margin classifier.

8.1 Hard-Margin Support Vector Machines

8.1.1 Formulation

PROPOSITION 8.1.1 (Distance to a Hyperplane)**.** Let $\mathcal{X} = \mathbb{R}^d$ be the domain. Let $\mathcal{Y} = \{0, 1\}$ be the labels. Let $\{(x_i, y_i)\}_{i=1}^n$ be a dataset. Let $H := \{x \in \mathbb{R}^d : \boldsymbol{w}^\top x + b = 0\}$ be a hyperplane in $\mathbb{R}^d$. Then for each $i = 1..n$, the distance $d_H(x_i)$ from x_i to H is given by

$$d_H(x_i) = y_i \frac{\boldsymbol{w}^\top x_i + b}{\|\boldsymbol{w}\|}.$$

Proof. Let x be an arbitrary point on H. Then the signed distance d_i from x_i to H can be computed via

$$\begin{aligned} d_i &= \frac{\boldsymbol{w}^\top (x_i - x)}{\|\boldsymbol{w}\|} = \frac{\boldsymbol{w}^\top x_i - \boldsymbol{w}^\top x}{\|\boldsymbol{w}\|} \\ &= \frac{\boldsymbol{w}^\top x_i + b}{\|\boldsymbol{w}\|}, \text{ since } \boldsymbol{w}^\top x + b = 0. \end{aligned}$$

So the distance $d_H(x_i)$ from x_i to H is $y_i \cdot d_i$. That is,

$$d_H(x_i) = y_i \frac{\boldsymbol{w}^\top x_i + b}{\|\boldsymbol{w}\|}.$$

■

PROPOSITION 8.1.2 (Margin)**.**

$$margin := \min_{i=1..n} \{y_i d_H(x_i)\}.$$

Recall that the perceptron algorithm is

$$\min 0$$

$$\text{subject to } \forall i, y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) > 0.$$

This is a feasibility problem.

$$\begin{aligned} \max_{\boldsymbol{w},b} \quad & \frac{1}{\|\boldsymbol{w}\|} \\ \text{subject to} \quad & \forall i \in \{1..n\}, \quad y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1. \end{aligned}$$

The objective function is the margin between the hyperplanes $H_{+1} : \boldsymbol{w}^\top \boldsymbol{x}_i + b = 1$ and $H_{-1} : \boldsymbol{w}^\top \boldsymbol{x}_i + b = -1$.

DEFINITION (Hard-Margin Support Vector Machines)**.**

$$\begin{aligned} \min_{\boldsymbol{w},b} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 \\ \text{subject to} \quad & \forall i \in \{1..n\}, \quad y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) \geq 1. \end{aligned}$$

This is a quadratic programming whereas the perceptron algorithm is a linear programming.

DEFINITION (Support Vectors)**.** We define the **support vectors** to be the points that are on the hyperplanes H_{+1} and H_{-1}.

PROPOSITION 8.1.3 (Existence)**.** Assume that the data points are linearly separable. Then the minimizers $\boldsymbol{w}$ and b exist.

PROPOSITION 8.1.4 (Uniqueness)**.** Assume that the data points are linearly separable. Then the minimizers $\boldsymbol{w}$ and b exist and are unique.

PROPOSITION 8.1.5 (Lagrangian Dual)**.** The Lagrangian dual problem of the hard-margin support vector machine is

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \| \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \|^2 \\ \text{subject to} \quad & \alpha \geq 0 \text{ and } \sum_{i=1}^{n} \alpha_i y_i = 0. \end{aligned}$$

Proof. The Lagrangian function is:

$$\begin{aligned} L(\boldsymbol{w}, b, \alpha) &:= \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{n} \alpha_i [y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) - 1], \qquad \text{for } \alpha \geq 0 \\ &= \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{w}^\top \boldsymbol{x}_i - \sum_{i=1}^{n} \alpha_i y_i b + \sum_{i=1}^{n} \alpha_i. \\ &= \frac{1}{2}\|\boldsymbol{w}\|^2 - \boldsymbol{w}^\top \left(\sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \right) - b \left(\sum_{i=1}^{n} \alpha_i y_i \right) + \sum_{i=1}^{n} \alpha_i. \end{aligned}$$

Then the derivatives of the Lagrangian function are:

$$\begin{aligned} \frac{\partial L(\boldsymbol{w}, b, \alpha)}{\partial \boldsymbol{w}} &= \boldsymbol{w} - \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \\ \frac{\partial L(\boldsymbol{w}, b, \alpha)}{\partial b} &= -\sum_{i=1}^{n} \alpha_i y_i. \end{aligned}$$

Setting all first derivatives of the Lagrangian function to 0, we get

$$\boldsymbol{w} = \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i y_i = 0.$$

Now I have the relation between the primal variables and the dual variables. So I rewrite the Lagrangian function in terms of the dual variables.

$$\begin{aligned} L(\alpha) &= \frac{1}{2} \| \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \|^2 - \left(\sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \right)^\top \left(\sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \right) - b \cdot 0 + \sum_{i=1}^{n} \alpha_i \\ &= \frac{1}{2} \| \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \|^2 - \| \sum_{i=1}^{n} \alpha_i y_i \boldsymbol{x}_i \|^2 - b \cdot 0 + \sum_{i=1}^{n} \alpha_i \end{aligned}$$

$$
\begin{aligned}
&= -\frac{1}{2}\|\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i\|^2 - b\cdot 0 + \sum_{i=1}^{n}\alpha_i \\
&= -\frac{1}{2}\|\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i\|^2 + \sum_{i=1}^{n}\alpha_i \\
&= \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\|\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i\|^2.
\end{aligned}
$$

That is,

$$L(\alpha) = \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\|\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i\|^2.$$

So the Lagrangian dual problem is

$$
\begin{aligned}
\max_{\alpha} \quad & \sum_{i=1}^{n}\alpha_i - \frac{1}{2}\|\sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i\|^2 \\
\text{subject to} \quad & \alpha \geq 0 \text{ and } \sum_{i=1}^{n}\alpha_i y_i = 0.
\end{aligned}
$$

$\blacksquare$

After solving the dual problem, we can solve $\boldsymbol{w}$ via $\boldsymbol{w} = \sum_{i=1}^{n}\alpha_i y_i \boldsymbol{x}_i$ and solve b by solving $y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) = 1$ for the i's such that $\alpha_i \neq 0$.

Both the primal and the dual are quadratic programmings. But the dual is sparse and the primal is not. So the dual problem is easier the solve.

The Lagrangian dual problem is equivalent to

$$
\begin{aligned}
\min_{\alpha} \quad & \frac{1}{2}\sum_{i}\sum_{j}\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j - \sum_{k}\alpha_k \\
\text{subject to} \quad & \alpha \geq 0, \quad \sum_{i}\alpha_i y_i = 0.
\end{aligned}
$$

PROPOSITION 8.1.6. Define for any $d \in \mathbb{N}$ a set $\Delta \subseteq \mathbb{R}^d$ as

$$\Delta := \{\boldsymbol{x} \in \mathbb{R}^d : \sum_{i}\boldsymbol{x}_i = 1\}.$$

$$
\begin{aligned}
\min_{\bar{\alpha}\in 2\Delta} \quad & \frac{1}{2}\|\sum_{i}\bar{\alpha}_i y_i \boldsymbol{x}_i\|_2^2 \\
\text{subject to} \quad & \sum_{i}\bar{\alpha}_i y_i = 0.
\end{aligned}
$$

Define sets P and N as

$$P := \{i : y_i = 1\} \text{ and } N := \{i : y_i = -1\}.$$

Define vectors μ and ν as

$$\mu := [\alpha_i]_{i \in P} \text{ and } \nu := [\alpha_i]_{i \in N}.$$

Then the problem is equivalent to

$$\min_{\mu \in \Delta, \nu \in \Delta} \quad \frac{1}{2} \left\| \sum_{i \in P} \mu_i \boldsymbol{x}_i - \sum_{j \in N} \nu_j \boldsymbol{x}_j \right\|_2 .$$

Note that $\sum_{i \in P} \mu_i \boldsymbol{x}_i \in \operatorname{conv}\{\boldsymbol{x}_i : i \in P\}$ and $\sum_{j \in N} \nu_j \boldsymbol{x}_j \in \operatorname{conv}\{\boldsymbol{x}_i : i \in P\}$. So the objective function is the distance between the two convex hulls. So $\boldsymbol{w}$ is in the direction of the line segment between the pair of points in the positive/negative convex hull with the minimum distance, and the hyperplane is the bisector of this line segment.

8.2 Soft-Margin Support Vector Machines

DEFINITION (Soft-Margin Support Vector Machines)**.**

$$\begin{aligned} \min_{\boldsymbol{w}, b} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + \gamma \sum_{i=1}^n \xi_i \\ \text{subject to} \quad & \forall i \in \{1..n\}, \quad y_i(\boldsymbol{w}^\top x_i + b) \geq 1 - \xi_i \\ & \forall i \in \{1..n\}, \quad \xi_i \geq 0. \end{aligned}$$

PROPOSITION 8.2.1 (Lagrangian Dual)**.** The Lagrangian dual problem of the soft-margin support vector machine is

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^n \alpha_i - \frac{1}{2} \| \sum_{i=1}^n \alpha_i y_i \boldsymbol{x}_i \|^2 \\ \text{subject to} \quad & \alpha_i \in [0, \gamma] \text{ and } \sum_{i=1}^n \alpha_i y_i = 0. \end{aligned}$$

Proof. The Lagrangian function is:

$$L(\boldsymbol{w}, b, \xi, \alpha, \lambda) = \frac{1}{2}\|\boldsymbol{w}\|^2 + \gamma \sum_{i=1}^n \xi_i - \sum_{i=1}^n \alpha_i [y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) - 1 + \xi_i] - \sum_{i=1}^n \lambda_i \xi_i, \text{ for } \alpha, \xi \geq 0$$

$$\begin{aligned}
&= \frac{1}{2}\|\boldsymbol{w}\|^2 + \gamma\sum_{i=1}^n \xi_i - \sum_{i=1}^n \alpha_i y_i \boldsymbol{w}^\top \boldsymbol{x}_i - \sum_{i=1}^n \alpha_i y_i b + \sum_{i=1}^n \alpha_i - \sum_{i=1}^n \alpha_i \xi_i - \sum_{i=1}^n \lambda_i \xi_i \\
&= \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^n \alpha_i y_i \boldsymbol{w}^\top \boldsymbol{x}_i - \sum_{i=1}^n \alpha_i y_i b + \sum_{i=1}^n (\gamma - \alpha_i - \lambda_i)\xi_i + \sum_{i=1}^n \alpha_i.
\end{aligned}$$

Then the derivatives of the Lagrangian function are:

$$\begin{aligned}
\frac{\partial L(\boldsymbol{w}, b, \xi, \alpha, \lambda)}{\partial \boldsymbol{w}} &= \boldsymbol{w} - \sum_{i=1}^n \alpha_i y_i \boldsymbol{x}_i \\
\frac{\partial L(\boldsymbol{w}, b, \xi, \alpha, \lambda)}{\partial b} &= -\sum_{i=1}^n \alpha_i y_i \\
\frac{\partial L(\boldsymbol{w}, b, \xi, \alpha, \lambda)}{\partial \xi_i} &= \gamma - \alpha_i - \lambda_i.
\end{aligned}$$

Setting all first derivatives of the Lagrangian function to 0, we get

$$\begin{aligned}
\boldsymbol{w} &= \sum_{i=1}^n \alpha_i y_i \boldsymbol{x}_i \\
\sum_{i=1}^n \alpha_i y_i &= 0 \\
\alpha_i + \lambda_i &= \gamma.
\end{aligned}$$

So the Lagrangian function in terms of the dual variables is:

$$\begin{aligned}
L(\alpha, \lambda) &= \frac{1}{2}\|\sum_{i=1}^n \alpha_i y_i \boldsymbol{x}_i\|^2 - \left(\sum_{i=1}^n \alpha_i y_i \boldsymbol{x}_i\right)^\top \left(\sum_{i=1}^n \alpha_i y_i \boldsymbol{x}_i\right) + \sum_{i=1}^n \alpha_i \\
&= -\frac{1}{2}\|\sum_{i=1}^n \alpha_i y_i \boldsymbol{x}_i\|^2 + \sum_{i=1}^n \alpha_i.
\end{aligned}$$

That is,

$$L(\alpha, \lambda) = \sum_{i=1}^n \alpha_i - \frac{1}{2}\|\sum_{i=1}^n \alpha_i y_i \boldsymbol{x}_i\|^2.$$

So the Lagrangian dual problem is

$$\begin{aligned}
\max_{\alpha} \quad & \sum_{i=1}^n \alpha_i - \frac{1}{2}\|\sum_{i=1}^n \alpha_i y_i \boldsymbol{x}_i\|^2 \\
\text{subject to} \quad & \alpha_i \in [0, \gamma] \text{ and } \sum_{i=1}^n \alpha_i y_i = 0.
\end{aligned}$$

■

Chapter 9

Mixture Models

9.1 Gaussian Mixture Models

THEOREM 9.1. Any continuous probability distribution can be approximated arbitrarily well by a finite mixture of Gaussian density functions.