Notes of Machine Learning
Supervised Learning

Xiangli Chen

Computer Science Department
University of Illinois at Chicago

June 30, 2017

Outline

1. Decision Tree
2. K Nearest Neighbore
3. Naive Bayes
4. Logistic Regression
5. Support Vector Machine
6. Linear Regression

Outline

Outline

Naive Bayes Formulation

Training data: $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$ and $x = (x^{(1)}, \cdots, x^{(d)})$
Classifer (Bayesian approach):

$$\begin{aligned} h(x) &= \underset{y}{\operatorname{argmax}}\, p(y|x) \\ &= \underset{y}{\operatorname{argmax}}\, p(x|y)p(y) \end{aligned}$$

Suppose $x^{(j)}$: M different values; y : K different class lables
$p(x|y)$ requires estimate $(M^d - 1)K$ values
Naive Bayesian assumption (conditional independence):

$$p(x^{(1)}, \cdots, x^{(d)}|y) = \prod_{j=1}^{d} p(x^{(j)}|y)$$

$p(x|y)$ requires estimate only $(M - 1)dK$ values

Naive Bayes - Discrete Features

Estimation via MLE of joint distribution (Generative approach)

$$(\hat{\theta}, \hat{\pi}) = \underset{\theta,\pi}{\operatorname{argmax}}\, p(\mathcal{D}|\theta, \pi) = \underset{\theta,\pi}{\operatorname{argmax}} \prod_{i=1}^{N} \prod_{j=1}^{d} p(x_i^{(j)}|y_i)p(y_i)$$

For class prior $\hat{\pi}_k = \hat{p}(y_k) = \dfrac{\#\mathcal{D}\{Y = y_k\}}{\#\mathcal{D}}$

For likelihood $\hat{\theta}_{mk}^{(j)} = \hat{p}(x_m^{(j)}|y_k) = \dfrac{\#\mathcal{D}\{X^{(j)} = x_m^{(j)} \wedge Y = y_k\}}{\#\mathcal{D}\{Y = y_k\}}$

Naive Bayesian classifier:

$$h_{NB}(x) = \underset{y}{\operatorname{argmax}}\, \hat{p}(y) \prod_{j=1}^{d} \hat{p}(x^{(j)}|y)$$

Naive Bayes - Smoothing

Issue: insufficient training data

Never see a training data where $X^{(j)} = a$ when $Y = b$

$$p(X^{(j)} = a | Y = b) = 0$$

Smoothing estimation method

$$\hat{\theta}_{mk}^{(j)} = \frac{\#\mathcal{D}\{X^{(j)} = x_m^{(j)} \wedge Y = y_k\} + l_{mk}^{(j)}}{\#\mathcal{D}\{Y = y_k\} + \sum_{m=1}^{M} l_m^{(j)}}$$

Similarly,

$$\hat{\pi}_k = \hat{p}(y_k) = \frac{\#\mathcal{D}\{Y = y_k\} + l_k}{\#\mathcal{D} + \sum_{k=1}^{K} l_k}$$

Naive Bayes - Smoothing - Interpretation

Two ways to interpret smoothing:

- add $\sum_{m=1}^{M} l_{mk}^{(j)}$ "virtual" examples for estimating $\hat{\theta}_{mk}^{(j)}$.
- MAP estimate for $\hat{\theta}_{mk}^{(j)}$ assuming a Dirichlet prior distribution over $\hat{\theta}_{mk}^{(j)}$. If $l_{mk}^{(j)} = 1$, known as Laplace smoothing.

Gaussian Naive Bayes - Continuous Features

Gaussian Naive Bayes (GNB):

$$p(X^{(j)} = x | Y = y_k) = \frac{1}{\sigma_k^{(j)}\sqrt{2\pi}} e^{\frac{-(x-\mu_k^{(j)})^2}{2(\sigma_k^{(j)})^2}}$$

There are $2dK$ unknown parameters: $\mu_k^{(j)}$ and $\sigma_k^{(j)}$.

Gaussian Naive Bayes - MLE

MLE for Gaussian distribution $X \sim \mathcal{N}(\mu, \sigma^2)$

$$\hat{\mu}_{MLE} = \frac{1}{N}\sum_{i=1}^{N} x_i \qquad \hat{\sigma}^2_{unbiased} = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \hat{\mu})^2$$

MLE estimate for GNB

$$\hat{\mu}_k^{(j)} = \frac{\sum_{i=1}^{N} x_i^{(j)} \delta(Y_i = y_k)}{\sum_{i=1}^{N} \delta(Y_i = y_k)} \quad (\hat{\sigma}_k^{(j)})^2 = \frac{\sum_{i=1}^{N} (x_i^{(j)} - \hat{\mu}_k^{(j)})^2 \delta(Y_i = y_k)}{\sum_{i=1}^{N} \delta(Y_i = y_k) - 1}$$

GNB classifier:

$$h_{GNB}(x) = \operatorname*{argmax}_{y_k} \hat{p}(y_k) \prod_{j=1}^{d} \mathcal{N}(\hat{\mu}_k^{(j)}, \hat{\sigma}_k^{(j)})$$

Gaussian Naive Bayes - Decision Boundary

Decision boundary of a binary Naive Bayes classifier (x's):

$$\frac{\prod_{i=1}^{d} p(x_i|y=0)p(y=0)}{\prod_{i=1}^{d} p(x_i|y=1)p(y=1)} = 1$$

Decision boundary of a binary GNB classifier with class independent variance is linear (i.e. $\sigma_{i,0}^2 = \sigma_{i,1}^2$) (log form)

$$\log \frac{\prod_{i=1}^{d} p(x_i|y=0)p(y=0)}{\prod_{i=1}^{d} p(x_i|y=1)p(y=1)} = \log \frac{p(y=0)}{p(y=1)} + \sum_{i=1}^{d} \log \frac{p(x_i|y=0)}{p(x_i|y=1)}$$

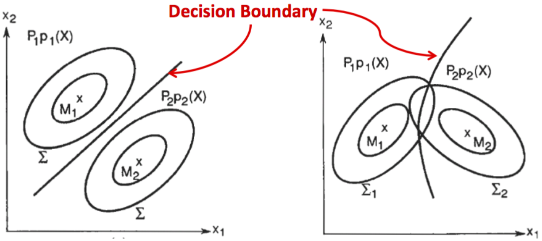
$$= \log \frac{p(y=0)}{p(y=1)} + \sum_{i=1}^{d} \frac{\mu_{i,1}^2 - \mu_{i,0}^2}{2\sigma_i^2} + \sum_{i=1}^{d} \frac{\mu_{i,0} - \mu_{i,1}}{\sigma_i^2} x_i = \underbrace{\omega_0 + \sum_{i=1}^{d} \omega_i x_i}_{\text{a linear classifier}} = 0$$

Gaussian Naive Bayes - Decision Boundary

Decision boundary of binary GNB classifiers

$$X = (x_1, x_2) \quad P_1 = p(y=0) \quad P_2 = p(y=1)$$

$$p_1(X) = p(x|y=1) \sim \mathcal{N}(M_1, \Sigma_1) \quad p_2(X) = p(x|y=2) \sim \mathcal{N}(M_2, \Sigma_2)$$



- $\Sigma_1 = \Sigma_2 = \Sigma$ (class independent variance) - linear
- $\Sigma_1 \neq \Sigma_2$ - non-linear

Outline

1. Decision Tree
2. K Nearest Neighbore
3. Naive Bayes
4. Logistic Regression
5. Support Vector Machine
6. Linear Regression

Logistic Function

Logistic function

$$f(z) = \frac{1}{1 + \exp(-z)}$$

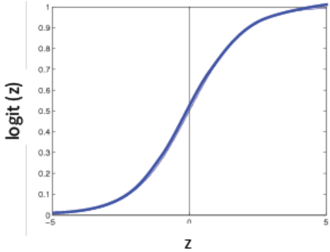


Figure 1: Logistic function

Logistic regression for binary class

$$p(y = 0|x) = \frac{1}{1 + \exp(\omega_0 + \sum_j \omega_j x_j)}$$

$$p(y = 1|x) = \frac{\exp(\omega_0 + \sum_j \omega_j x_j)}{1 + \exp(\omega_0 + \sum_j \omega_j x_j)}$$

Decision Boundary

Decision boundary is linear

$$\frac{p(y=1|x)}{p(y=0|x)} = \exp\left(\omega_0 + \sum_j \omega_j x_j\right) = 1$$

Log form

$$\omega_0 + \sum_j \omega_j x_j = 0$$

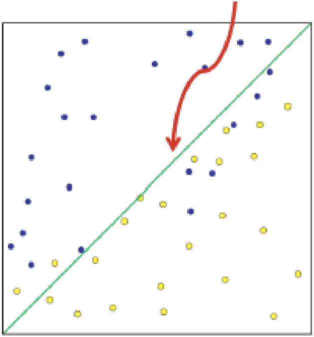


Figure 2: Decision boundary

Training Logistic Regression Model

Data

$$\{(x_i, y_i)_{i=1}^n\} \qquad x_i = (x_{i1}, \cdots, x_{id})$$

Maximum conditional log likelihood (discriminative approach)

$$\hat{\omega}_{MLE} = \arg\max_{\omega} \log \prod_{i=1}^{n} p(y_i | x_i, \omega)$$

Optimization Problem

Conditional log likelihood

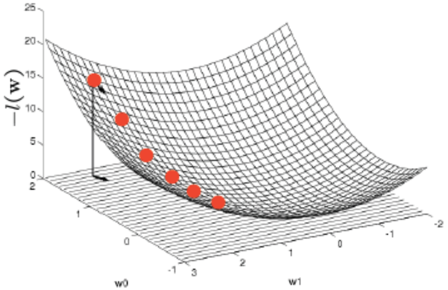
$$
\begin{aligned}
l(\omega) &= \sum_{i=1}^{n} \log p(y_i|x_i, \omega) \\
&= \sum_{i} y_i \log p(y_i = 1|x_i, \omega) + (1 - y_i) \log p(y_i = 0|x_i, \omega) \\
&= \sum_{i} y_i(\omega_0 + \sum_{j} \omega_j x_{ij}) - \log(1 + \exp(\omega_0 + \sum_{j} \omega_j x_{ij}))
\end{aligned}
$$

- no closed form solution
- $l(\omega)$ is a concave function of ω ($-l(\omega)$ is the convex function)

Gradient Descent Method

Gradient descent ($\eta > 0$)

$$\omega^{(t+1)} \leftarrow \omega^{(t)} - \triangle\omega$$



$$\triangle\omega = \eta\nabla_\omega(-l(\omega)) = -\eta\nabla_\omega(l(\omega))$$

$$\omega^{(t+1)} \leftarrow \omega^{(t)} + \eta\nabla_\omega(l(\omega))$$

$$\nabla_\omega l(\omega) = \left[\frac{\partial l(\omega)}{\partial \omega_0} \cdots \frac{\partial l(\omega)}{\partial \omega_d}\right]^T$$

$$\frac{\partial l(\omega)}{\partial \omega_0} = \sum_i (y_i - \hat{p}(y_i = 1|x_i, \omega))$$

$$\frac{\partial l(\omega)}{\partial \omega_j} = \sum_i x_{ij}(y_i - \hat{p}(y_i = 1|x_i, \omega))$$

Multi-Class Logistic Regression

Logistic regression of multi-class - $Y \in \{y_1, \cdots, y_K\}$
For $k < K$

$$p(y = y_k|x) = \frac{\exp(\omega_{0k} + \sum_{j=1}^{d} \omega_{jk} x_j)}{1 + \sum_{m=1}^{K-1} \exp(\omega_{0m} + \sum_{j=1}^{d} \omega_{jm} x_j)}$$

For $k = K$

$$p(y = y_K|x) = \frac{1}{1 + \sum_{m=1}^{K-1} \exp(\omega_{0m} + \sum_{j=1}^{d} \omega_{jm} x_j)}$$

Classifier

$$f_{MCLR}(x) = \arg\max_{y_k} p(y = y_k|x)$$

Decision Boundary of Multi-Class

linear

Training Multi-Class Logistic Regression

Conditional log likelihood

$$
\begin{aligned}
l(\omega) &= \sum_{i=1}^{n} \log p(y_i|x_i,\omega) = \sum_{i=1}^{n}\sum_{k=1}^{K} 1_{(y_i=y_k)} \log p(y_i = y_k|x_i,\omega) \\
&= \sum_{i=1}^{n}\sum_{k=1}^{K-1} 1_{(y_i=y_k)} \left(\omega_{0k} + \sum_{j=1}^{d} \omega_{jk} x_{ij}\right) \\
&\qquad\qquad - \log\left(1 + \sum_{m=1}^{K-1} \exp\left(\omega_{0m} + \sum_{j=1}^{d} \omega_{jm} x_{ij}\right)\right)
\end{aligned}
$$

Gradient ($(K-1)\times(d+1)$ matrix, $\omega_{(0 \text{ or } j)K}$ doesn't exist)

$$
\frac{\partial l(\omega)}{\partial \omega_{0k}} = \sum_i (1_{(y_i=y_k)} - \hat{p}(y_i = y_k|x_i,\omega))
$$

$$
\frac{\partial l(\omega)}{\partial \omega_{jk}} = \sum_i x_{ij}(1_{(y_i=y_k)} - \hat{p}(y_i = y_k|x_i,\omega))
$$

Outline

Outline