Statistical Learning IV (23-11-2017)

1. Linear Discriminant Analysis (LDA)
2. Surrogate Losses
3. Log loss
4. Logistic Regression
5. Discriminative vs Generative Classification
   ↳ Logistic Regression vs Naive Bayes

5. Linear Discriminant Analysis (LDA)

Model: $f_k(x)$ is $N(\mu_k, \Sigma_k)$ is multivariate Gaussian (see fig. 4.5)

Take $\Sigma_k = \Sigma$ the same for all k.

Parameter estimates from T:

$\hat{\pi}_k = \frac{n_k}{N}$ where n_k is number of observations in class k.

$\hat{\mu}_k = \frac{1}{n_k} \sum_{i: y_i = k} x_i$ is mean of x_i in class k

$$\hat{\Sigma} = \frac{1}{N-K} \sum_{k=1}^{K} \sum_{i: y_i = k} (x_i - \hat{\mu}_k)(x_i - \hat{\mu}_k)^T$$

$$\begin{aligned}
\underset{k}{\operatorname{argmax}}\ \hat{f}_k(x) \cdot \hat{\pi}_k &= \underset{k}{\operatorname{argmax}}\ \log \hat{f}_k(x) + \log \hat{\pi}_k \\
&= \underset{k}{\operatorname{argmax}}\ \log\left(\frac{1}{(2\pi)^{p/2} |\hat{\Sigma}|^{1/2}} e^{-\frac{1}{2}(x-\hat{\mu}_k)^T \hat{\Sigma}^{-1} (x-\hat{\mu}_k)} \right) + \log \hat{\pi}_k \\
&= \underset{k}{\operatorname{argmax}}\ -\frac{1}{2}(x-\hat{\mu}_k)^T \hat{\Sigma}^{-1}(x-\hat{\mu}_k) + \log \hat{\pi}_k \\
&= \underset{k}{\operatorname{argmax}}\ -\frac{1}{2} x^T \hat{\Sigma}^{-1} x + x^T \hat{\Sigma}^{-1} \hat{\mu}_k - \frac{1}{2} \hat{\mu}_k^T \hat{\Sigma}^{-1} \hat{\mu}_k + \log \hat{\pi}_k
\end{aligned}$$

$\delta_k(x)$

linear in x, so decision boundary between any two classes also linear in x. Hence name LDA

Decision boundary: (between classes k_1 and k_2)

x s.t. $\delta_{k_1}(x) - \delta_{k_2}(x) = 0$

4. Surrogate Losses

Classification: find $\hat{f}$ with small expected 0/1-loss

$$EPE(\hat{f}) = \mathbb{E}_{X,Y}[L(Y, \hat{f}(X))]$$

0 if $\hat{f}(X)$ predicts Y correctly
1 if $\hat{f}(X)$ predicts Y incorrectly

Empirical Risk Minimization:

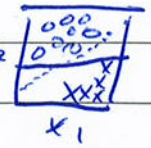
$$\hat{f} = \underset{f \in \mathcal{F}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i))$$

Suppose $\mathcal{F}$ is set of all linear functions: $\mathcal{F} = \{f_\beta(x) = x^T\beta \mid \beta \in \mathbb{R}^p\}$

$y \in \{-1, +1\}$ $\quad L(y, f(x)) = \begin{cases} 0 & \text{if } \operatorname{sign}(f(x)) = y \\ 1 & \text{if } \operatorname{sign}(f(x)) \neq y \end{cases}$

Difficulties:

- ERM solution not unique for linearly separable data:



- Computational problems for non-separable data.

Surrogate/proxy loss L':

Care about loss L, but train classifier with loss L'.

Want: L large $\Leftrightarrow$ L' large

L small $\Leftrightarrow$ L' small

Usually: surrogate L' convex in parameters, so can optimize efficiently

Two most important surrogates:

logistic loss ↑ today, hinge loss ↑ used in support vector machines (next week)

For 0/1-loss, predict correctly iff

$$\text{sign}(x^T\hat{\beta}) = y \Leftrightarrow yx^T\hat{\beta} > 0$$

Most surrogate losses are therefore decreasing as a function of $yx^T\hat{\beta}$.

(see slide)

Exception: squared error N.B: $y^2 = 1$

$$(y - x^T\hat{\beta})^2 = y^2(y - x^T\hat{\beta})^2 = (y^2 - yx^T\hat{\beta})^2 = (1 - yx^T\hat{\beta})^2$$

increases for $yx^T\hat{\beta} > 1$!

This is undesirable!

5. Log(arithmic) Loss

* My predictions are <u>probability distributions</u> P for y

* $L(y, P) = -\log P(y)$
  - ↳ small if I gave large probability to y
  - ↳ large if I gave small probability to y

Maximum Likelihood = ERM for log loss:

$$\text{ML: } \arg\max_\beta \prod_{i=1}^N p_\beta(y_i|x_i) = \arg\min_\beta \sum_{i=1}^N -\log p_\beta(y_i|x_i)$$
$$= \arg\min_\beta \sum_{i=1}^N L(y_i, p_\beta(\cdot|x_i))$$

Bayes optimal predictor for log loss is true distribution:

$$\arg\min_p \mathbb{E}_y[L(y, p)] = p^*(\cdot|x)$$

6. Logistic Regression (for two classes for simplicity)

(How can we use the power of linear regression for classification?)

Bad idea: $\hat{p}(y=1|x) = x^T\hat{\beta}$

E.g. y_i is opening the window during i-th lecture
x_{i3} is avg temperature in the room in °C.
$\hat{\beta}_3 = \frac{1}{5}$
Then $+10°C \Rightarrow x^T\hat{\beta} + 2 \Rightarrow \hat{p}(y=1|x) + 2 > 1$! ($10 \cdot \frac{1}{5}$)

Probabilities do <u>not</u> behave linearly.

Better: $\log P_\beta(Y=1|x) = x^T\beta \iff P_\beta(y=1|x) = e^{x^T\beta}$

$x^T\hat{\beta}+2 \Rightarrow P_{\hat{\beta}}(y=1|x)$ increases by <u>factor</u> e^2,
but can still go outside $[0,1]$

Logistic Regression:

$$\log \frac{P_\beta(Y=1|x)}{P_\beta(Y=-1|x)} = x^T\beta \iff \begin{cases} P_\beta(y=1|x) = \frac{e^{x^T\beta}}{1+e^{x^T\beta}} = \frac{1}{1+e^{-x^T\beta}} \\ P_\beta(y=-1|x) = \frac{1}{1+e^{x^T\beta}} \end{cases}$$

(log odds is linear)

$$P_\beta(y|x) = \frac{1}{1+e^{-yx^T\beta}}$$

decision boundary:

$$P_{\hat{\beta}}(Y=1|x) > \tfrac{1}{2} \iff x^T\hat{\beta} > 0 \Rightarrow \quad x \text{ s.t } x^T\hat{\beta} = 0$$

is linear

logistic loss = log loss for logistic regression:

$$L(y,x^T\beta) = -\log P_\beta(y|x) = \log(1+e^{-yx^T\beta})$$

Train with maximum likelihood = ERM for logistic loss

$$\underset{\beta}{\text{minimize}} \sum_{i=1}^{N} \log(1+e^{-y_i x_i^T\beta})$$

No formula for minimum, but can find it by convex optimization

<u>Important extensions:</u>

$$\text{minimize} \sum_{i=1}^{N} \log(1+e^{-y_i x_i^T\beta}) + \lambda \text{pen}(\beta)$$

L_2-penalty: $\text{pen}(\beta) = \sum_{j=2}^{p} \beta_j^2$ (like ridge regression)

L_1-penalty: $\text{pen}(\beta) = \sum_{j=2}^{p} |\beta_j|$ (like lasso)

Considerations for penalization similar as for squared error.

3. Discriminative vs Generative Methods

Probabilistic model: $\mathcal{P} = \{ P_\beta(x,y) \mid \beta \in R^p \}$

Generative

(imagine data generated by first choosing class y, then features x given y.)
e.g. spam

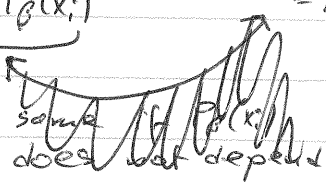
estimate $P(x,y)$

LDA, Naive Bayes, ...

overkill for classification

$$\underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} -\log P_\beta(x_i, y_i)$$

$$= \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} -\log P_\beta(y_i \mid x_i) + \sum_{i=1}^{N} -\log P_\beta(x_i)$$



acts as regularizer, so reduces variance $\Rightarrow$ better for small N

Discriminative

estimate $P(y \mid x)$

Logistic regression, ...

less overkill, closer to class boundary, which is only thing we are interested in

$$\underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} -\log P_\beta(y_i \mid x_i)$$

$$= \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{N} -\log P_\beta(y_i \mid x_i) + \sum_{i=1}^{N} -\log P^*(x_i)$$

P^*: true distribution of x

better fit of true distribution for large N if $P^* \notin \mathcal{P}$

Two methods are generative-discriminative pair if one estimates $P_\beta(x,y)$ and other $P_\beta(y \mid x)$ in same model $\mathcal{P}$.

E.g.: LDA - logistic regression for continuous features x_j;
Naive Bayes - logistic regression for binary features

Logistic Regression vs Naive Bayes

LR: $\log \frac{\hat{P}(y=1|X)}{\hat{P}(y=-1|X)} = X^T\hat{\beta} + \hat{\beta}_0$ (writing intercept separately)

$$\Updownarrow$$

$$\hat{P}(y=1|X) = \frac{e^{X^T\hat{\beta}+\hat{\beta}_0}}{1+e^{X^T\hat{\beta}+\hat{\beta}_0}}$$

Will show that

(*) $\log \frac{\hat{P}(y=1|X)}{\hat{P}(y=-1|X)} = X^T\hat{\alpha} + \hat{\alpha}_0$ for Naive Bayes

($\hat{\alpha}$, $\hat{\alpha}_0$ determined by NB parameters $\hat{\theta}_{1j}, \hat{\theta}_{-1j}, \hat{\pi}_1, \hat{\pi}_{-1}$)

Hence Naive Bayes is working with a probabilistic model $\mathcal{P} = \{P_\alpha(X,y) \mid \alpha \in \mathbb{R}^p\}$ such that $P_\alpha(y|X)$ is the same as in the model logistic regression is using.

So NB-LR form a generative-discriminative pair

Therefore expect NB better for smaller N, LR better for larger N.

NG, Jordan, NIPS 2001 make this precise:

* Asymptotic performance as $N \to \infty$: LR always better than NB
* But LR needs $N \geq O(p)$ to reach asymptotic performance

whereas (under some technical assumptions) NB needs $N \geq O(\log p)$ to reach asymptotic performance

To show (*):

$$\log \frac{\hat{P}(y=1|x)}{\hat{P}(y=-1|x)} = \log \frac{\hat{P}(x|y=1)\hat{P}(y=1)}{\hat{P}(x|y=-1)\hat{P}(y=-1)} = \log \frac{\hat{f}_1(x)\hat{\pi}_1}{\hat{f}_{-1}(x)\hat{\pi}_{-1}}$$

$$x = \begin{pmatrix} 0 \leftarrow \text{UNIVERSITY} \\ \vdots \\ 1 \leftarrow \text{BIG} \\ 2 \leftarrow \text{MONEY} \end{pmatrix} = \begin{pmatrix} n_1 \\ \vdots \\ n_p \end{pmatrix} \qquad f_k(x) = \prod_{j=1}^{p} \theta_{k,j}^{n_j}$$

$$\log \frac{\hat{f}_1(x)\hat{\pi}_1}{\hat{f}_{-1}(x)\hat{\pi}_{-1}} = \log \frac{\prod_{j=1}^{p} \hat{\theta}_{1,j}^{n_j}}{\prod_{j=1}^{p} \hat{\theta}_{-1,j}^{n_j}} + \log \frac{\hat{\pi}_1}{\hat{\pi}_{-1}}$$

$$= \log \prod_{j=1}^{p} \left(\frac{\hat{\theta}_{1,j}}{\hat{\theta}_{-1,j}} \right)^{n_j} + \log \frac{\hat{\pi}_1}{\hat{\pi}_{-1}}$$

$$= \sum_{j=1}^{p} n_j \log \frac{\hat{\theta}_{1,j}}{\hat{\theta}_{-1,j}} + \log \frac{\hat{\pi}_1}{\hat{\pi}_{-1}}$$

$$= x^T \hat{\alpha} + \hat{\alpha}_0 \quad \text{for } \hat{\alpha} = \begin{pmatrix} \log \frac{\hat{\theta}_{1,1}}{\hat{\theta}_{-1,1}} \\ \vdots \\ \log \frac{\hat{\theta}_{1,p}}{\hat{\theta}_{-1,p}} \end{pmatrix} \qquad \hat{\alpha}_0 = \log \frac{\hat{\pi}_1}{\hat{\pi}_{-1}}$$

Possible to choose $\hat{\theta}_{k,j}$, $\hat{\pi}_k$ such that $\hat{\alpha} = \hat{\beta}$, $\hat{\alpha}_0 = \hat{\beta}_0$, but NB does not do that $\forall_0$