Sapienza University of Rome

Master in Artificial Intelligence and Robotics
Master in Engineering in Computer Science

Machine Learning

A.Y. 2019/2020

Prof. L. Iocchi, F. Patrizi, V. Ntouskos

6. Probabilistic models for classification

L. Iocchi, F. Patrizi, V. Ntouskos

Overview

- Probabilistic generative models
- Probabilistic discriminative models
- Logistic regression

References

C. Bishop. Pattern Recognition and Machine Learning. Sect. 4.2, 4.3

Probabilistic Models for Classification

- Generative: estimate $P(C_i|\mathbf{x})$ through $P(\mathbf{x}|C_i)$ and Bayes theorem
- Discriminative: estimate $P(C_i|\mathbf{x})$ directly from a model

Probabilistic Generative Models

Consider first the case of two classes.

Find the conditional probability:

$$\begin{aligned} P(C_1|\mathbf{x}) &= \frac{P(\mathbf{x}|C_1)P(C_1)}{P(\mathbf{x}|C_1)P(C_1) + P(\mathbf{x}|C_2)P(C_2)} \\ &= \frac{1}{1+\exp(-a)} = \sigma(a) \end{aligned}$$

with:

$$a = \ln \frac{p(\mathbf{x}|C_1)P(C_1)}{p(\mathbf{x}|C_2)P(C_2)}$$

and

$$\sigma(a) = \frac{1}{1+\exp(-a)} \text{ the } \textit{sigmoid function}.$$

Probabilistic Generative Models

Assume $p(\mathbf{x}|C_i) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \boldsymbol{\Sigma})$ - same covariance matrix

$$a = \ln \frac{p(\mathbf{x}|C_1)P(C_1)}{p(\mathbf{x}|C_2)P(C_2)} = \ln \frac{\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_1, \boldsymbol{\Sigma})P(C_1)}{\mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_2, \boldsymbol{\Sigma})P(C_2)} = \ldots = \mathbf{w}^T\mathbf{x} + w_0$$

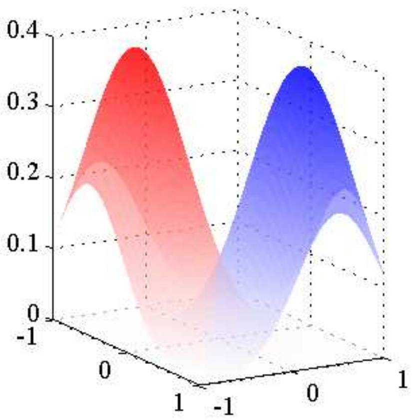
with:

$$\begin{aligned} \mathbf{w} &= \boldsymbol{\Sigma}^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2), \\ w_0 &= -\frac{1}{2}\boldsymbol{\mu}_1^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_1 + \frac{1}{2}\boldsymbol{\mu}_2^T\boldsymbol{\Sigma}^{-1}\boldsymbol{\mu}_2 + \ln\frac{P(C_1)}{P(C_2)}. \end{aligned}$$

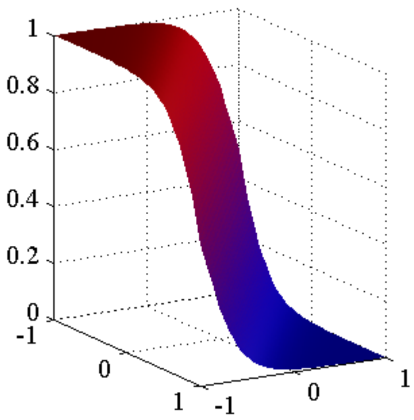
Thus

$$P(C_1|\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x} + w_0),$$

Probabilistic Generative Models



$P(\mathbf{x}|C_1), P(\mathbf{x}|C_2)$



$P(C_1|\mathbf{x})$

Decision rule: $c = C_1 \iff P(c = C_1|\mathbf{x}) > 0.5$

Probabilistic Generative Models Multi-class

$$P(C_k|\mathbf{x}) = \frac{P(\mathbf{x}|C_k)P(C_k)}{\sum_j P(\mathbf{x}|C_j)P(C_j)} = \frac{exp(a_k)}{\sum_j exp(a_j)}$$

(normalized exponential or softmax function)

with $a_k = \ln P(\mathbf{x}|C_k)P(C_k)$

Maximum likelihood

Maximum likelihood solution for 2 classes

Assuming $P(C_1) = \pi$ (thus $P(C_2) = 1 - \pi$), $P(\mathbf{x}|C_i) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_i, \mathbf{\Sigma})$

Given data set $D = \{(\mathbf{x}_n, t_n)_{n=1}^N\}$, $t_n = 1$ if $\mathbf{x}_n$ belongs to class C_1, $t_n = 0$ if $\mathbf{x}_n$ belongs to class C_2

Let N_1 be the number of samples in D belonging to C_1 and N_2 be the number of samples in C_2 ($N_1 + N_2 = N$)

Likelihood function

$$P(\mathbf{t}|\pi, \boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \mathbf{\Sigma}, D) = \prod_{n=1}^{N} [\pi \mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_1, \mathbf{\Sigma})]^{t_n} [(1-\pi)\mathcal{N}(\mathbf{x}_n|\boldsymbol{\mu}_2, \mathbf{\Sigma})]^{(1-t_n)}$$

Unknown π, $\boldsymbol{\mu}_1$, $\boldsymbol{\mu}_2$, $\mathbf{\Sigma}$

Maximum likelihood

Maximum likelihhod solution for 2 classes
Maximizing log likelihood function, we obtain

$$\pi = \frac{N_1}{N}$$

$$\boldsymbol{\mu}_1 = \frac{1}{N_1}\sum_{n=1}^{N} t_n \mathbf{x}_n \qquad \boldsymbol{\mu}_2 = \frac{1}{N_2}\sum_{n=1}^{N} (1-t_n)\mathbf{x}_n$$

$$\mathbf{\Sigma} = \frac{N_1}{N} S_1 + \frac{N_2}{N} S_2$$

with $S_i = \frac{1}{N_i}\sum_{n \in C_i} (\mathbf{x}_n - \boldsymbol{\mu}_i)(\mathbf{x}_n - \boldsymbol{\mu}_i)^T$, $i = 1, 2$

Note: details in C. Bishop. PRML. Section 4.2.2

Maximum likelihood for K classes

Gauusian Naive Bayes

$P(C_k) = \pi_k$, $P(\mathbf{x}|C_k) = \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$

Data set $D = \{(\mathbf{x}_n, \mathbf{t}_n)_{n=1}^N\}$, with $\mathbf{t}_n$ 1-of-K encoding

$$\pi_k = \frac{N_k}{N}$$

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n=1}^{N} t_{nk} \, \mathbf{x}_n$$

$$\boldsymbol{\Sigma} = \sum_{k=1}^{K} \frac{N_k}{N} S_k \; , \qquad S_k = \frac{1}{N_k} \sum_{n=1}^{N} t_{nk} (\mathbf{x}_n - \boldsymbol{\mu}_k)(\mathbf{x}_n - \boldsymbol{\mu}_k)^T$$

Probabilistic Discriminative Models

Estimate directly $P(C_i|\mathbf{x})$

Logistic regression is a *classification* method based on maximum likelihood.

Logistic regression

Two classes

Given data set D, consider $\{\mathbf{x}_n, t_n\}$, with $t_n \in \{0, 1\}$, $n = 1, \ldots, N$

Likelihood function:

$$p(\mathbf{t}|\mathbf{w}) = \prod_{n=1}^{N} y_n^{t_n} (1 - y_n)^{1-t_n}$$

with $y_n = p(C_1|\mathbf{x}_n) = \sigma(\mathbf{w}^T \mathbf{x}_n)$

Sapienza University of Rome, Italy - Machine Learning (2019/2020)

Logistic regression

Cross-entropy error function

$$E(\mathbf{w}) \equiv -\ln p(\mathbf{t}|\mathbf{w}) = -\sum_{n=1}^{N} [t_n \ln y_n + (1 - t_n) \ln(1 - y_n)]$$

Solution concept: solve the optimization problem

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w}} E(\mathbf{w})$$

Many solvers available.

Iterative reweighted least squares

Apply *Newton-Raphson* iterative optimization for minimizing $E(\mathbf{w})$.

Gradient of the error with respect to $\mathbf{w}$

$$\nabla E(\mathbf{w}) = \sum_{n=1}^{N} (y_n - t_n)\mathbf{x}_n$$

Gradient descent step

$$\mathbf{w} \leftarrow \mathbf{w} - \boldsymbol{H}^{-1}\nabla E(\mathbf{w})$$

$\boldsymbol{H} = \nabla\nabla E(\mathbf{w})$ is the Hessian matrix of $E(\mathbf{w})$ (second derivatives with respect to $\mathbf{w}$).

Iterative reweighted least squares

Given
$\mathbf{t} = (t_1, \ldots, t_n)^T$, $\mathbf{y} = (y_1, \ldots, y_n)^T$, $\boldsymbol{R}$: diagonal matrix with
$R_{nn} = y_n(1 - y_n)$,
$\mathbf{X} = \begin{pmatrix} \mathbf{x}_1^T \\ \ldots \\ \mathbf{x}_N^T \end{pmatrix}$

we have

$$\nabla E(\mathbf{w}) = \mathbf{X}^T(\mathbf{y} - \mathbf{t})$$

$$\boldsymbol{H} = \nabla\nabla E(\mathbf{w}) = \sum_{n=1}^{N} y_n(1 - y_n)\mathbf{x}_n\mathbf{x}_n^T = \mathbf{X}^T\boldsymbol{R}\mathbf{X}$$

Iterative reweighted least squares

Iterative method:

1. Initialize $\mathbf{w}$
2. Repeat until termination condition

$$\mathbf{w} \leftarrow \mathbf{w} - (\mathbf{X}^T \boldsymbol{R} \mathbf{X})^{-1} \mathbf{X}^T (\mathbf{y} - \mathbf{t})$$

Multiclass logistic regression

$$P(C_k|\mathbf{x}) = y_k(\mathbf{x}) = \frac{exp(a_k)}{\sum_j exp(a_j)}$$

with $a_k = \mathbf{w}_k^T \mathbf{x}$.

Discriminative model

$$P(\mathbf{T}|\mathbf{w}_1, \ldots \mathbf{w}_K) = \prod_{n=1}^{N} \prod_{k=1}^{K} P(C_k|\mathbf{x}_n)^{t_{nk}} = \prod_{n=1}^{N} \prod_{k=1}^{K} y_{nk}^{t_{nk}}$$

with $y_{nk} = y_k(\mathbf{x}_n)$ and $\mathbf{T}$ $N \times K$ matrix of t_{nk}.

Multiclass logistic regression

Cross-entropy error function

$$E(\mathbf{w}_1, \dots \mathbf{w}_K) = -\ln P(\mathbf{T}|\mathbf{w}_1, \dots \mathbf{w}_K) = -\sum_{n=1}^{N} \sum_{k=1}^{K} t_{nk} \ln y_{nk}$$

Iterative algorithm with gradient $\nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \dots \mathbf{w}_K) = \dots$ and Hessian matrix $\nabla_{\mathbf{w}_k} \nabla_{\mathbf{w}_j} E(\mathbf{w}_1, \dots \mathbf{w}_K) = \dots$

Learning in feature space

All methods described above can be applied in a transformed space of the input (*feature space*).

Given a function $\phi : \mathbf{X} \mapsto \mathbf{\Phi}$ ($\mathbf{\Phi}$ is the *feature space*)
each sample $\mathbf{x}_n$ can be mappet to a feature vector $\boldsymbol{\phi}_n = \phi(\mathbf{x}_n)$

Replacing $\mathbf{x}_n$ with $\boldsymbol{\phi}_n$ in all the equations above, makes the learning system to work in the feature space instead of the input space.

We will see in the next lectures why this trick is useful.