CSCI567 Machine Learning (Fall 2018)

Prof. Haipeng Luo

U of Southern California

Oct 24, 2018

Administration

HW 4 is available and is due on 11/04.

Today's plan: first finish clustering, then move on to more unsupervised learning problems

Outline

1. Density estimation

2. Naive Bayes

Outline

1. Density estimation
   - Parametric methods
   - Nonparametric methods

2. Naive Bayes

Density estimation

Observe what we have done indirectly for clustering with GMMs is:

Given a training set $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$, **estimate a density function p that could have generated this dataset** (via $\boldsymbol{x}_n \overset{i.i.d.}{\sim} p$).

This is exactly the problem of *density estimation*, another important unsupervised learning problem.

Useful for many downstream applications

- we have seen clustering already, will see more today
- these applications also *provide a way to measure quality of the density estimator*

Parametric methods: generative models

Parametric estimation assumes a generative model parametrized by $\boldsymbol{\theta}$:

$$p(\boldsymbol{x}) = p(\boldsymbol{x}\,; \boldsymbol{\theta})$$

Examples:

- GMM: $p(\boldsymbol{x} \mid \boldsymbol{\theta}) = \sum_{k=1}^{K} \omega_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$ where $\boldsymbol{\theta} = \{\omega_k, \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k\}$
- Multinomial for 1D examples with K possible values

$$p(x = k\,; \boldsymbol{\theta}) = \theta_k$$

where $\boldsymbol{\theta}$ is a distribution over K elements.

Size of $\boldsymbol{\theta}$ is independent of the training set size, so it's parametric.

Parametric methods: estimation

Again, we apply **MLE** to learn the parameters $\boldsymbol{\theta}$:

$$\underset{\boldsymbol{\theta}}{\operatorname{argmax}} = \sum_{n=1}^{N} \ln p(x_n\,; \boldsymbol{\theta})$$

For some cases this is intractable and we can use EM to approximately solve MLE (e.g. GMMs).

For some other cases this admits a simple closed-form solution (e.g. multinomial).

MLE for multinomial

$$\begin{aligned}\underset{\boldsymbol{\theta}}{\operatorname{argmax}} &= \sum_{n=1}^{N} \ln p(x = x_n\,; \boldsymbol{\theta}) = \sum_{n=1}^{N} \ln \theta_{x_n} \\ &= \sum_{k=1}^{K} \sum_{n:x_n=k} \ln \theta_k = \sum_{k=1}^{K} z_k \ln \theta_k\end{aligned}$$

where $z_k = |\{n : x_n = k\}|$ is **the number of examples with value k**.

The solution is simply

$$\theta_k = \frac{z_k}{N} \propto z_k,$$

i.e. the fraction of examples with value k.

Nonparametric methods

Can we estimate *without assuming a fixed generative model?*

Yes, **kernel density estimation (KDE)** is a common approach

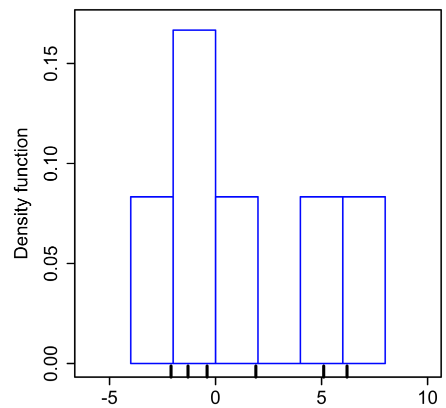
- here "kernel" means something different from what we have seen for "kernel function" (in fact it refers to several different things in ML)
- the approach is nonparametric: it keeps the entire training set
- we focus on the 1D (continuous) case

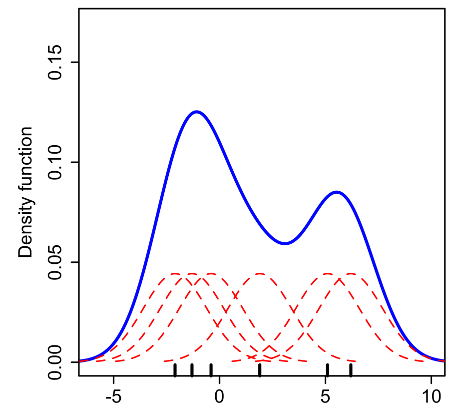
High level idea

picture from Wikipedia

Construct something similar to a **histogram**:

- for each data point, create a "bump" (via a Kernel)
- sum up all the bumps





Kernel

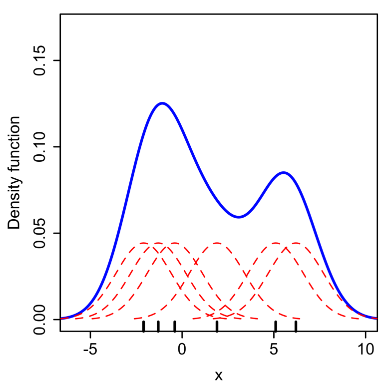
KDE with **a kernel** $K: \mathbb{R} \rightarrow \mathbb{R}$:

$$p(x) = \frac{1}{N} \sum_{n=1}^{N} K(x - x_n)$$

e.g. $K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}}$, the standard Gaussian density

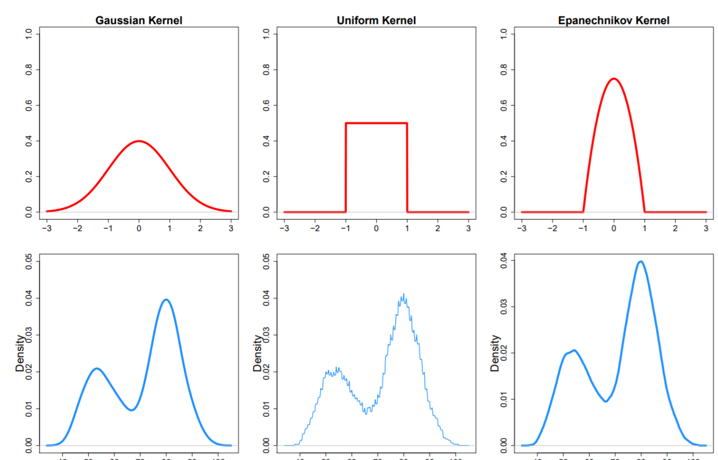
Kernel needs to satisfy:

- symmetry: $K(u) = K(-u)$
- $\int_{-\infty}^{\infty} K(u)du = 1$, makes sure p is a density function.



Different kernels $K(u)$

$$\frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}} \qquad \frac{1}{2}\mathbb{I}[|u| \leq 1] \qquad \frac{3}{4}\max\{1 - x^2, 0\}$$



Bandwidth

If $K(u)$ is a kernel, then for any $h > 0$

$$K_h(u) \triangleq \frac{1}{h} K\left(\frac{u}{h}\right) \qquad \text{(stretching the kernel)}$$

can be used as a kernel too (verify the two properties yourself)

So general KDE is determined by both the kernel K and the bandwidth h

$$p(x) = \frac{1}{N}\sum_{n=1}^{N} K_h(x - x_n) = \frac{1}{Nh}\sum_{n=1}^{N} K\left(\frac{x - x_n}{h}\right)$$

- x_n controls the center of each bump
- h controls the width/variance of the bumps

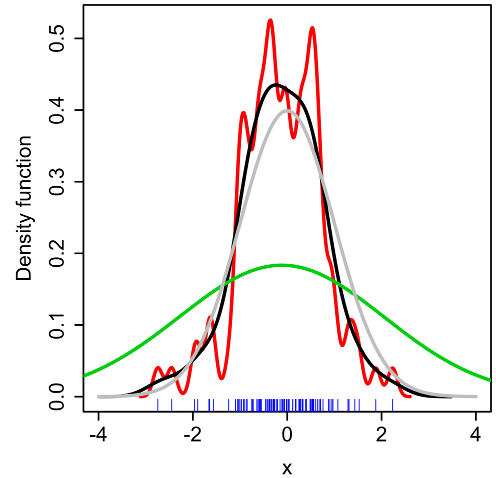
Effect of bandwidth

picture from Wikipedia

Larger h means larger variance and also smoother density

Gray curve is ground-truth

- Red: $h = 0.05$
- Black: $h = 0.337$
- Green: $h = 2$



Bandwidth selection

Selecting h is a deep topic

- there are theoretically-motivated approaches
- one can also do cross-validation based on downstream applications

Outline

Naive Bayes

Naive Bayes

- a simple yet surprisingly powerful **classification** algorithm
- **density estimation** is one important part of the algorithm

Bayes optimal classifier

Recall: suppose the data $(\boldsymbol{x}_n, y_n)$ is drawn from a joint distribution p, the **Bayes optimal classifier** is

$$f^*(\boldsymbol{x}) = \operatorname*{argmax}_{c \in [\mathsf{C}]} p(c \mid \boldsymbol{x})$$

i.e. predict the class with the largest conditional probability.

p is of course unknown, but we can estimate it, which is *exactly a density estimation problem!*

Estimation

How to estimate a joint distribution? Observe we always have

$$p(\boldsymbol{x}, y) = p(y)p(\boldsymbol{x} \mid y)$$

We know how to estimate $p(y)$ by now.

To estimate $p(\boldsymbol{x} \mid y = c)$ for some $c \in [\mathsf{C}]$, we are doing density estimation using data $\{n : y_n = c\}$.

This is *not a 1D problem* in general.

A "naive" assumption

Naive Bayes assumption:
conditioning on a label, features are independent, which means

$$p(\boldsymbol{x} \mid y = c) = \prod_{d=1}^{\mathsf{D}} p(x_d \mid y = c)$$

Now for each d and c we have a simple 1D density estimation problem!

Is this a reasonable assumption? Sometimes yes, e.g.

- use $x =$ (Height, Vocabulary) to predict $y =$ Age
- Height and Vocabulary are dependent
- but condition on Age, they are independent!

More often this assumption is *unrealistic and "naive"*, but still Naive Bayes can work very well even if the assumption is wrong.

Example: discrete features

Height: ≤3', 3'-4', 4'-5', 5'-6', ≥6'
Vocabulary: ≤5K, 5K-10K, 10K-15K, 15K-20K, ≥20K
Age: ≤5, 5-10, 10-15, 15-20, 20-25, ≥25

MLE estimation: e.g.

$$p(\text{Age} = \text{10-15}) = \frac{\#\text{examples with age 10-15}}{\#\text{examples}}$$

$$p(\text{Height} = \text{5'-6'} \mid \text{Age} = \text{10-15}) = \frac{\#\text{examples with height 5'-6' and age 10-15}}{\#\text{examples with age 10-15}}$$

More formally

For a label $c \in [\mathsf{C}]$,

$$p(y = c) = \frac{|\{n : y_n = c\}|}{N}$$

For each possible value k of a discrete feature d,

$$p(x_d = k \mid y = c) = \frac{|\{n : x_{nd} = k, y_n = c\}|}{|\{n : y_n = c\}|}$$

Continuous features

If the feature is continuous, we can do

- parametric estimation, e.g. via a Gaussian

$$p(x_d = x \mid y = c) = \frac{1}{\sqrt{2\pi}\sigma_{cd}} \exp\left(-\frac{(x-\mu_{cd})^2}{2\sigma_{cd}^2}\right)$$

  where μ_{cd} and σ_{cd}^2 are the empirical mean and variance of feature d among all examples with label c (verified in W4).

- or nonparametric estimation, e.g. via a Kernel K and bandwidth h:

$$p(x_d = x \mid y = c) = \frac{1}{|\{n : y_n = c\}|} \sum_{n: y_n = c} K_h(x - x_{nd})$$

How to predict?

After learning the model

$$p(x, y) = p(y) \prod_{d=1}^{\mathsf{D}} p(x_d \mid y)$$

the **prediction** for a new example x is

$$\begin{aligned}
\operatorname*{argmax}_{c \in [\mathsf{C}]} p(y = c \mid x) &= \operatorname*{argmax}_{c \in [\mathsf{C}]} p(x, y = c) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(p(y = c) \prod_{d=1}^{\mathsf{D}} p(x_d \mid y = c) \right) \\
&= \operatorname*{argmax}_{c \in [\mathsf{C}]} \left(\ln p(y = c) + \sum_{d=1}^{\mathsf{D}} \ln p(x_d \mid y = c) \right)
\end{aligned}$$

Examples

For **discrete features**, plugging in previous MLE estimations gives

$$
\begin{aligned}
&\operatorname*{argmax}_{c\in[\mathsf{C}]} \; p(y=c \mid x) \\
&= \operatorname*{argmax}_{c\in[\mathsf{C}]} \left(\ln p(y=c) + \sum_{d=1}^{\mathsf{D}} \ln p(x_d \mid y=c) \right) \\
&= \operatorname*{argmax}_{c\in[\mathsf{C}]} \left(\ln |\{n : y_n = c\}| + \sum_{d=1}^{\mathsf{D}} \ln \frac{|\{n : x_{nd} = x_d, y_n = c\}|}{|\{n : y_n = c\}|} \right)
\end{aligned}
$$

Examples

For **continuous features** with a Gaussian model,

$$
\begin{aligned}
&\operatorname*{argmax}_{c\in[\mathsf{C}]} \; p(y=c \mid x) \\
&= \operatorname*{argmax}_{c\in[\mathsf{C}]} \left(\ln p(y=c) + \sum_{d=1}^{\mathsf{D}} \ln p(x_d \mid y=c) \right) \\
&= \operatorname*{argmax}_{c\in[\mathsf{C}]} \left(\ln |\{n : y_n = c\}| + \sum_{d=1}^{\mathsf{D}} \ln \left(\frac{1}{\sqrt{2\pi}\sigma_{cd}} \exp\left(-\frac{(x_d - \mu_{cd})^2}{2\sigma_{cd}^2} \right) \right) \right) \\
&= \operatorname*{argmax}_{c\in[\mathsf{C}]} \left(\ln |\{n : y_n = c\}| - \sum_{d=1}^{\mathsf{D}} \left(\ln \sigma_{cd} + \frac{(x_d - \mu_{cd})^2}{2\sigma_{cd}^2} \right) \right)
\end{aligned}
$$

which is *quadratic* in the feature $\boldsymbol{x}$.

What naive Bayes is learning?

Observe again the case for continuous features with a Gaussian model, if we **fix the variance for each feature to be σ** (i.e. not a parameter of the model any more), then the prediction becomes

$$
\begin{aligned}
&\operatorname*{argmax}_{c\in[\mathsf{C}]} \; p(y=c \mid x) \\
&= \operatorname*{argmax}_{c\in[\mathsf{C}]} \left(\ln |\{n : y_n = c\}| - \sum_{d=1}^{\mathsf{D}} \left(\ln \sigma + \frac{(x_d - \mu_{cd})^2}{2\sigma^2} \right) \right) \\
&= \operatorname*{argmax}_{c\in[\mathsf{C}]} \left(\ln |\{n : y_n = c\}| - \sum_{d=1}^{\mathsf{D}} \frac{\mu_{cd}^2}{2\sigma^2} + \sum_{d=1}^{\mathsf{D}} \frac{\mu_{cd}}{\sigma^2} x_d \right) \\
&= \operatorname*{argmax}_{c\in[\mathsf{C}]} \left(w_{c0} + \sum_{d=1}^{\mathsf{D}} w_{cd} x_d \right) = \operatorname*{argmax}_{c\in[\mathsf{C}]} \; \boldsymbol{w}_c^{\mathrm{T}} \boldsymbol{x} \qquad (\textit{linear classifier!})
\end{aligned}
$$

where we denote $w_{c0} = \ln |\{n : y_n = c\}| - \sum_{d=1}^{\mathsf{D}} \frac{\mu_{cd}^2}{2\sigma^2}$ and $w_{cd} = \frac{\mu_{cd}}{\sigma^2}$.

Connection to logistic regression

Moreover by similar calculation one can verify

$$p(y=c \mid x) \propto e^{\boldsymbol{w}_c^{\mathrm{T}} \boldsymbol{x}}$$

This is exactly the **softmax** function, *the same model we used for a probabilistic interpretation of logistic regression!*

So what is different then? They **learn the parameters in different ways**:

- both via MLE, one on $p(y=c \mid x)$, the other on $p(x, y)$
- solutions are different: logistic regression has no closed-form, naive Bayes admits a simple closed-form

Generative model v.s discriminative model

	Discriminative model	Generative model
Example	logistic regression	naive Bayes
Model	conditional $p(y \mid x)$	joint $p(x, y)$ (might have same $p(y \mid x)$)
Learning	MLE	MLE
Accuracy	usually better for large N	usually better for small N
Remark		more flexible, can generate data after learning