Applied Machine Learning

Naive Bayes

Reihaneh Rabbany

McGill
School of Computer Science

Learning objectives

- the assumption of Naive Bayes classifier
- what does learning and prediction steps involve?
- different likelihood functions
- Bayesian parameter learning in Naive Bayes
- practical considerations

Bayes rule for classification

given

- the prior probability of each class
- likelihood of observations given the class

use Bayes rule for classification

prior class probability:

frequency of observing this label

likelihood of input features given the class label

(input features for each label come from a different distribution)

posterior class probability

$$\underbrace{p(y = c \mid x)}_{\text{posterior class probability}} = \frac{\overbrace{p(c)}^{\text{prior}}\,\overbrace{p(x|c)}^{\text{likelihood}}}{\underbrace{p(x)}_{\text{evidence}}}$$

evidence

don't worry about the evidence

it simply normalizes the posterior class probabilities

Bayes rule for classification

example

$x \in \{-, +\}$ input: test results, a single binary feature

$y \in \{\text{yes}, \text{no}\}$ label: patient has cancer

prior: 1% of population has cancer $p(\text{yes}) = .01$

likelihood: $p(+|\text{yes}) = .9$ TP rate of the test (90%)

$$p(c \mid x) = \frac{p(c)p(x|c)}{p(x)}$$

posterior: $p(\text{yes}|+) = .08$

FP rate of the test (5%)

evidence: $p(+) = p(\text{yes})p(+|\text{yes}) + p(\text{no})p(+|\text{no}) = .01 \times .9 + .99 \times .05 = .189$

Generative classification

training learn the following distributions from the data $\mathcal{D} = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$

prior probability of each class $p(y = c) \forall c \in \{1, \ldots, C\}$

likelihood of data for each class $p(x|y = c)$

prediction use the Bayes rule to get the posterior class probability $p(y = c \mid x) \propto p(c)p(x|c)$

generative classifier because we are learning the joint data distribution $p(x, y) = p(y)p(x|y)$

we can *generate* new data from this joint distribution

in a **discriminative classifier** we directly learn $p(y|x)$

more on this in the future

Generative classification

prior class probability: frequency of observing this label

likelihood of input features given the class label
(input features for each label come from a different distribution)

$$p(y = c \mid x) = \frac{p(c)p(x|c)}{p(x)}$$

posterior probability
of a given class

marginal probability of the input (evidence)
$\sum_{c'=1}^{C} p(x, c')$

Some generative classifiers:

- Gaussian Discriminant Analysis: the likelihood is multivariate Gaussian
- Naive Bayes: decomposed likelihood

Naive Bayes **model**

number of input features

assumption about the likelihood $p(x|y) = \prod_{d=1}^{D} p(x_d|y)$

when is this assumption correct?

when features are **conditionally independent** given the label $x_i \perp\!\!\!\perp x_j \mid y$

knowing the label, the value of one input feature gives us no information about the other input features

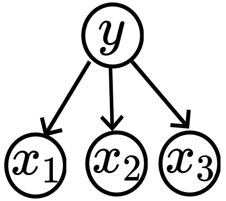
How is the likelihood derived from this independence assumption?

chain rule of probability (true for any distribution)

$$p(x|y) = p(x_1|y)p(x_2|y, x_1)p(x_3|y, x_1, x_2)\dots p(x_D|y, x_1, \dots, x_{D-1})$$

conditional independence assumption

x_1, x_2 give no extra information, so $p(x_3|y, x_1, x_2) = p(x_3|y)$



Naive Bayes: objective

given the training dataset $\mathcal{D} = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$

a generative classifier maximizes the joint likelihood (or log-likelihood)

$$L(\pi, \theta; \mathcal{D}) = \prod_{n \in \mathcal{D}} p(x^{(n)}, y^{(n)}; \pi, \theta)$$

π, θ are the model parameters

$$\begin{aligned}
\ell(\pi, \theta) &= \sum_n \log p(x^{(n)}, y^{(n)}; \pi, \theta) \\
&= \sum_n \left[\log p(y^{(n)}; \pi) + \log p(x^{(n)} | y^{(n)}; \theta)\right] \\
&= \sum_n \log p(y^{(n)}; \pi) + \sum_n \log p(x^{(n)} | y^{(n)}; \theta) \\
&= \sum_n \log p(y^{(n)}; \pi) + \sum_n \log \prod_d p(x_d^{(n)} | y^{(n)}; \theta_d) \\
&= \sum_n \log p(y^{(n)}; \pi) + \sum_d \sum_n \log p(x_d^{(n)} | y^{(n)}; \theta_d)
\end{aligned}$$

$p(x, y) = p(y) p(x|y)$

using Naive Bayes assumption here

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

$\log p(x|y) = \sum_{d=1}^{D} \log p(x_d|y)$

separate max-likelihood problems for prior and each feature x_d given the label

Prior class probabilities

class probabilities prior to looking at the features

for binary classification, class probability is given by Bernoulli $p(y;\pi) = \pi^y (1-\pi)^{1-y}$

recall the max-likelihood estimate for Bernoulli

$$\arg\max_\pi \sum_n \log p(y^{(n)};\pi) = \frac{1}{N}\sum_n y^{(n)}$$

for multi-class classification, class probability is given by categorical distribution

$$p(y;\pi) = \prod_{c=1}^{C} {\pi_c}^{\mathbb{I}(y=c)} = \pi_y$$

note that in this case π is a vector

max-likelihood estimate is again given by empirical frequencies

$$\arg\max_{\pi_c} \sum_n \log p(y^{(n)};\pi) = \frac{N(y=c)}{N}$$

s.t. $\sum_c \pi_c = 1$

frequency of class c in our dataset

$$\pi^* = [\frac{N_1}{N}, \ldots, \frac{N_C}{N}]$$

In both cases we learn the prior simply as the class frequencies in the training data

Naive Bayes: objective

given the training dataset $\mathcal{D} = \{(x^{(1)}, y^{(1)}), \ldots, (x^{(N)}, y^{(N)})\}$

a generative classifier maximizes the joint likelihood (or log-likelihood)

$$\ell(\pi, \theta) = \sum_n \log p(x^{(n)}, y^{(n)}; \pi, \theta)$$

$$= \sum_n \log p(y^{(n)}; \pi) + \log p(x^{(n)} | y^{(n)}; \theta)$$

$$= \sum_n \log p(y^{(n)}; \pi) + \sum_n \log p(x^{(n)} | y^{(n)}; \theta)$$

$$= \sum_n \log p(y^{(n)}; \pi) + \sum_n \log \prod_d p(x_d^{(n)} | y^{(n)}; \theta_d)$$

so far we know how to maximize this part

$$= \sum_n \log p(y^{(n)}; \pi) + \sum_d \sum_n \log p(x_d^{(n)} | y^{(n)}; \theta_d)$$

Next, how to maximize this part

separate max-likelihood problems for prior and each feature x_d given the label

Likelihood terms

likelihood terms $p(x_d|y;\theta_d)$

- encode our assumption about the *generative process*
- different types of features require different forms of likelihood
  - Bernoulli for binary features
  - Categorical for categorical features
  - Multinomial for "count" features
  - Gaussian is one option for continuous feature
- each feature x_d may use a different likelihood form
- separate maximum conditional likelihood estimate for each feature

$$\arg\max_{\theta_d} \sum_{n=1}^{N} \log p(x_d^{(n)} \mid y^{(n)}; \theta_d)$$

Bernoulli Naive Bayes

for a binary **feature** likelihood is Bernoulli

$$\begin{cases} p(x_d \mid y=0; \theta_d) = \text{Bernoulli}(x_d; \theta_{d,0}) \\ p(x_d \mid y=1; \theta_d) = \text{Bernoulli}(x_d; \theta_{d,1}) \end{cases}$$

one parameter per label

short form: $p(x_d \mid y; \theta_d) = \text{Bernoulli}(x_d; \theta_{d,y})$

max-likelihood estimation is similar to what we saw for the prior

closed form solution of MLE $\rightarrow$ $\theta_{d,c}^{MLE} = \frac{N(y=c, x_d=1)}{N(y=c)}$ number of training instances satisfying this condition

Covid-19 classification example

each patient has seven binary features $x \in \{0,1\}^7$

we have a dataset of N=1000 patients, where 200 had covid-19

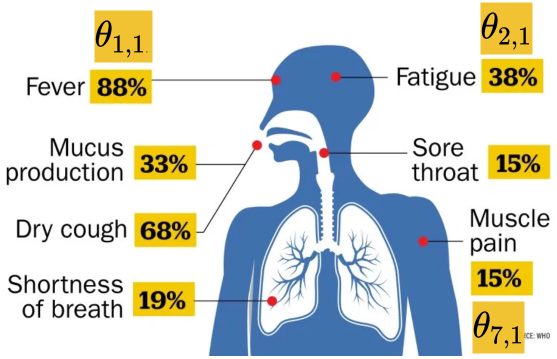


image credit: Time magazine

learning:

learn the prior: $\pi = \frac{N(y=1)}{N} = .2$ $\quad \text{Bernoulli}(y; \pi)$

for each symptom d:

probability of symptom $x_d = 1$ given $y = \begin{cases} 1 & \theta_{d,1} = \frac{N(y=1, x_d=1)}{N(y=1)} \\ 0 & \theta_{d,0} = \frac{N(y=0, x_d=1)}{N(y=0)} \end{cases}$

$\text{Bernoulli}(x_d | y = 1; \theta_{d,1})$

$\text{Bernoulli}(x_d | y = 0; \theta_{d,0})$

prediction:

for a new patient x calculate unnormalized posterior

$$\begin{cases} \tilde{p}(y=0|x) = \text{Bernoulli}(0; \pi) \prod_d \text{Bernoulli}(x_d; \theta_{d,0}) \\ \tilde{p}(y=1|x) = \text{Bernoulli}(1; \pi) \prod_d \text{Bernoulli}(x_d; \theta_{d,1}) \end{cases}$$

normalize it $p(y=1|x) = \frac{\tilde{p}(y=1|x)}{\tilde{p}(y=0|x) + \tilde{p}(y=1|x)}$

Disease diagnosis example

what changes in **multi-class** setting?

$p(y; \pi)$ learn the prior: $\pi_c = \frac{N(y=c)}{N}$

for each symptom d:

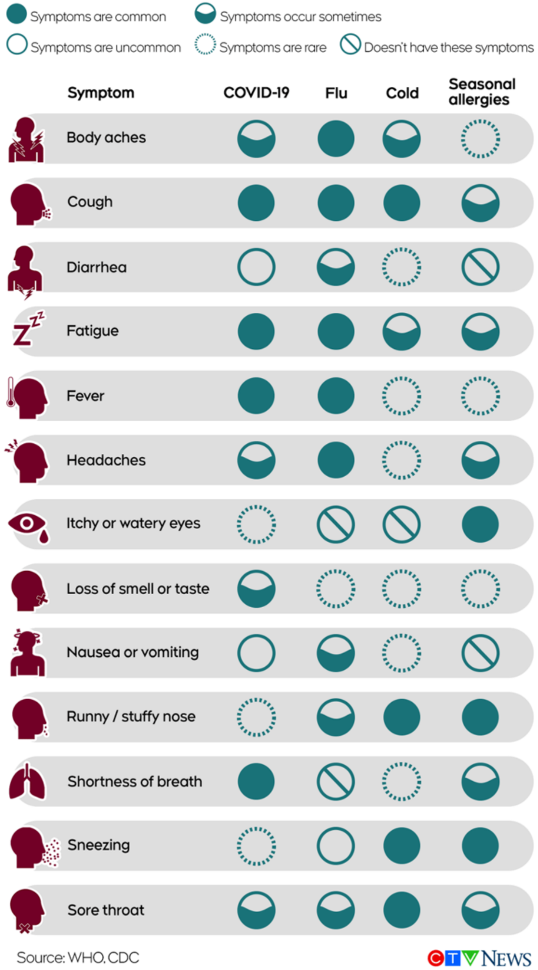
$p(x_d | y; \theta_d)$ learn the conditional likelihood:

probability of symptom $x_d = 1$ given $y = \begin{cases} 0 \\ \vdots \\ C \end{cases}$

how many parameters in our model?

binary classification, binary features $1 + 2D$

multi-class classification, binary features $C + CD$



Document classification

example

e.g., spam filtering

words in our vocabulary

each document (email) is one instance $x^{(n)} \in \{0,1\}^D$

$x_d^{(n)} = 1$ if the word d appears in document n

classify the documents based on this bag of words representation

$D = 7$

$N = 5$

	it	is	puppy	cat	pen	a	this
it is a puppy	1	1	1	0	0	1	0
it is a kitten	1	1	0	0	0	1	0
it is a cat	1	1	0	1	0	1	0
that is a dog and this is a pen	0	1	0	0	1	1	1
it is a matrix	1	1	0	0	0	1	0

document-term matrix

learning:

MLE for the prior $\mathrm{Bernoulli}(y; \pi)$ (spam frequency in our dataset)

MLE for the likelihood terms $\mathrm{Bernoulli}(x; \theta_{d,y})$ (frequency of word (d) in spam/non-spam documents)

prediction:

calculate the posterior $p(y|x) \propto \mathrm{Bernoulli}(y;\pi) \prod_d \mathrm{Bernoulli}(x_d; \theta_{d,y})$

image: Feature Engineering for Machine Learning

Document classification

example

let's learn the Naive Bayes for the following data
the label y=1 if the sentence is about animals

	it	is	puppy	cat	pen	a	this	label
it is a puppy	1	1	1	0	0	1	0	1
it is a kitten	1	1	0	0	0	1	0	1
it is a cat	1	1	0	1	0	1	0	1
that is a dog and this is a pen	0	1	0	0	1	1	1	1
it is a matrix	1	1	0	0	0	1	0	0

prior parameter: $\pi = \frac{4}{5}$

class conditional parameters: $\theta_{d,y}$

	$d=1$						$d=7$	
$y=0$	$\frac{1}{1}$	$\frac{1}{1}$	$\frac{0}{1}$	$\frac{0}{1}$	$\frac{0}{1}$	$\frac{1}{1}$	$\frac{0}{1}$	$\theta_{7,0}$?
$y=1$	$\frac{3}{4}$	$\frac{4}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{4}{4}$	$\frac{1}{4}$	$\theta_{7,1}$?

we get a new sentence: `it is a random sentence`

$x = [1,1,0,0,0,1,0]$

$$\tilde{p}(y=1|x) = \frac{4}{5} \times \frac{3}{4} \times \frac{4}{4} \times \frac{3}{4} \times \frac{3}{4} \times \frac{3}{4} \times \frac{4}{4} \times \frac{3}{4} \approx .19$$

$$\tilde{p}(y=0|x) = \frac{1}{5} \times \frac{1}{1} \times \frac{1}{1} \times \frac{1}{1} \times \frac{1}{1} \times \frac{1}{1} \times \frac{1}{1} \times \frac{1}{1} = .2$$

$$\rightarrow \quad p(y=1|x) = \frac{.19}{.2+.19} \approx .49$$

image: Feature Engineering for Machine Learning.

Why Naive Bayes assumption?

Naive Bayes assumption $p(x|y) = \prod_d p(x_d|y)$

what if we did not make this assumption?

consider the **spam filtering example:**

- D can be very large
- **with** the Naive Bayes assumption: learn the frequency of each word (d) in spam/non-spam documents
- **without** it: learn the frequency of each possible subset of words in spam/non-spam documents

e.g., for $x = [1,1,0,0,0,1,0]$ we need to estimate $p(x|y)$

problems

- many combinations of words may not appear in even one document
- we need exponentially more parameters
- even for large datasets, this could lead to *overfitting*

	it	is	puppy	cat	pen	a	this
it is a puppy	1	1	1	0	0	1	0
it is a kitten	1	1	0	0	0	1	0
it is a cat	1	1	0	1	0	1	0
that is a dog and this is a pen	0	1	0	0	1	1	1
it is a matrix	1	1	0	0	0	1	0

image: Feature Engineering for Machine Learning

Bayesian Naive Bayes

using MLE in Naive Bayes can lead to overfitting

	it	is	puppy	cat	pen	a	this
it is a puppy	1	1	1	0	0	1	0
it is a kitten	1	1	0	0	0	1	0
it is a cat	1	1	0	1	0	1	0
that is a dog and this is a pen	0	1	0	0	1	1	1
it is a matrix	1	1	0	0	0	1	0

example let's classify this new sentence:

`that dog was my puppy`

$$\tilde{p}(y=1|x) = \frac{4}{5} \times \frac{1}{4} \times \frac{0}{4} \times \ldots = 0$$

$$\tilde{p}(y=0|x) = \frac{1}{5} \times \frac{0}{1} \times \frac{0}{1} \times \ldots = 0$$

the problem is that the word "is" appears in all instances

max-likelihood estimate $\theta_{1,1} = \theta_{1,0} = 1$

we can solve this by being Bayesian in parameter learning:

instead of maintaining a point estimates $\pi, \theta_{d,y}$ we maintain distributions $p(\pi), p(\theta_{d,y})$ for $y \in \{0,1\}, d$

start from separate prior for each parameter $p(\pi), p(\theta_{d,y})$

calculate the likelihood $\prod_n p(y^{(n)}|\pi)$

update with observed frequencies in the dataset

image: Feature Engineering for Machine Learning

Bayesian Naive Bayes

start from separate prior for each parameter $p(\pi) = \mathrm{Beta}(\pi; \alpha^\pi, \beta^\pi)$ $\quad p(\theta_{d,y}) = \mathrm{Beta}(\theta; \alpha^\theta, \beta^\theta)$

calculate the posterior $p(\pi|\mathcal{D}) = \mathrm{Beta}(\pi; \alpha^\pi + N(y=1), \beta^\pi + N(y=0))$

$$p(\theta_{d,\bar{y}}|\mathcal{D}) = \mathrm{Beta}(\theta_{d,\bar{y}}; \alpha^\theta + N(y=\bar{y}, x_d = 1), \beta^\theta + N(y=\bar{y}, x_d = 0))$$

use posterior predictive for a new instance (x) $p(y=1|x,\mathcal{D}) = \int_{\theta,\pi} p(y=1|\pi)p(\pi|\mathcal{D}) \prod_d p(x_d|\theta_{d,1})p(\theta_{d,1}|\mathcal{D}) \mathrm{d}\theta \mathrm{d}\pi$

individual posterior predictives $\rightarrow$ $= \left(\int_\pi p(y=1|\pi)p(\pi|\mathcal{D})\mathrm{d}\pi\right) \prod_d \left(\int_{\theta_{d,1}} p(x_d|\theta_{d,1})p(\theta_{d,1}|\mathcal{D})\mathrm{d}\theta\right)$

for Beta distribution, we simply used the posterior mean (and dropped the integral)

$$\tilde{p}(y=1|x,\mathcal{D}) = \frac{\alpha^\pi + N(y=1)}{\alpha^\pi + \beta^\pi + N} \prod_d \left(\frac{\alpha^\theta + N(y=1, x_d=1)}{\alpha^\theta + \beta^\theta + N(y=1)}\right)^{x_d} \left(\frac{\beta^\theta + N(y=1, x_d=0)}{\alpha^\theta + \beta^\theta + N(y=1)}\right)^{(1-x_d)}$$

recall: Laplace smoothing

compare with our previous prediction (using MLE)

$$\tilde{p}(y=1|x,\mathcal{D}) = \frac{N(y=1)}{N} \prod_d \left(\frac{N(y=1, x_d=1)}{N(y=1)}\right)^{x_d} \left(\frac{N(y=1, x_d=0)}{N(y=1)}\right)^{(1-x_d)}$$

we are simply adding a constant to various frequencies

Bayesian Naive Bayes

example

let's classify this new sentence using **Laplace smoothing**:

`this dog was my puppy`

$\alpha^\pi = \beta^\pi = \alpha^\theta = \beta^\theta = 1$

$$\tilde{p}(y=1|x) = \frac{4+1}{5+2} \times \frac{1+1}{4+2} \times \frac{0+1}{4+2} \times \frac{1+1}{4+2} \times \frac{3+1}{4+2} \times \frac{3+1}{4+2} \times \frac{0+1}{4+2} \times \frac{1+1}{4+2} \approx .00032$$

$$\tilde{p}(y=0|x) = \frac{1+1}{5+2} \times \frac{0+1}{1+2} \times \frac{0+1}{1+2} \times \frac{0+1}{1+2} \times \frac{1+1}{1+2} \times \frac{1+1}{1+2} \times \frac{0+1}{1+2} \times \frac{0+1}{1+2} \approx .00052$$

$$p(y=0|x) = \frac{.00052}{.00032+.00052} \approx .62$$

	it	is	puppy	cat	pen	a	this
it is a puppy	1	1	1	0	0	1	0
it is a kitten	1	1	0	0	0	1	0
it is a cat	1	1	0	1	0	1	0
that is a dog and this is a pen	0	1	0	0	1	1	1
it is a matrix	1	1	0	0	0	1	0
$y=0$	$\frac{1}{1}$	$\frac{1}{1}$	$\frac{0}{1}$	$\frac{0}{1}$	$\frac{0}{1}$	$\frac{1}{1}$	$\frac{0}{1}$
$y=1$	$\frac{3}{4}$	$\frac{4}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{1}{4}$	$\frac{4}{4}$	$\frac{1}{4}$

$d=1$ $d=7$

note that if D is large we have to calculate the product of many terms → numerical problems!

image: Feature Engineering for Machine Learning

Log-Sum-Exp trick

In estimating unnormalized posteriors we could get numerical problems (underflow)

when calculating the posterior for new instances, we work with in the **log-domain**:

$$\log \tilde{p}(y|x;\pi,\theta) = \log p(y;\pi) + \sum_d \log p(x_d|y;\theta_d)$$

to get the final probabilities we need to normalize $\tilde{p}$

$$p(y|x;\pi,\theta) = \frac{\tilde{p}(y|x;\pi,\theta)}{\sum_{c=1}^{C} \tilde{p}(c|x;\pi,\theta)}$$

we can do this **normalization in the log domain** as well:

$$\log p(y|x;\pi,\theta) = \log \tilde{p}(y|x;\pi,\theta) - \log \sum_{c=1}^{C} \exp(\log \tilde{p}(c|x;\pi,\theta))$$

we could run into very large or small numbers inside the exponential

Log-Sum-Exp trick

we can do this **normalization in the log domain** as well:

$$\log p(y|x;\pi,\theta) = \log \tilde{p}(y|x;\pi,\theta) - \log \sum_{c=1}^{C} \exp(\log \tilde{p}(c|x;\pi,\theta))$$

observation $\log \sum_c \exp a_c = \log\left(\exp(a_0)\left(\sum_c \exp(a_c - a_0)\right)\right) = a_0 + \log \sum_c \exp(a_c - a_0)$

to make log-sum-exp numerically stable, bring the numbers a_c close to zero

for example choose $a_0 \leftarrow \max_c a_c$

Multinomial likelihood

what if we wanted to use word frequencies in document classification?

$x_d^{(n)}$ is the number of times word d appears in document n

	it	is	puppy	cat	pen	a	this
it is a puppy	1	1	1	0	0	1	0
it is a kitten	1	1	0	0	0	1	0
it is a cat	1	1	0	1	0	1	0
that is a dog and this is a pen	0	2	0	0	1	2	1
it is a matrix	1	1	0	0	0	1	0

Multinomial likelihood

$$p(x|y) = \mathrm{Mult}(x;\theta_y) = \frac{(\sum_d x_d)!}{\prod_{d=1}^D x_d!} \prod_{d=1}^D \theta_{d,y}^{x_d}$$

probability of word d appearing x_d time

the max-likelihood estimate is again given by the relative frequency

$$\theta_{d,c}^{MLE} = \frac{\sum_n x_d^{(n)} \mathbb{I}(y^{(n)}=c)}{\sum_n \sum_{d'} x_{d'}^{(n)} \mathbb{I}(y^{(n)}=c)}$$

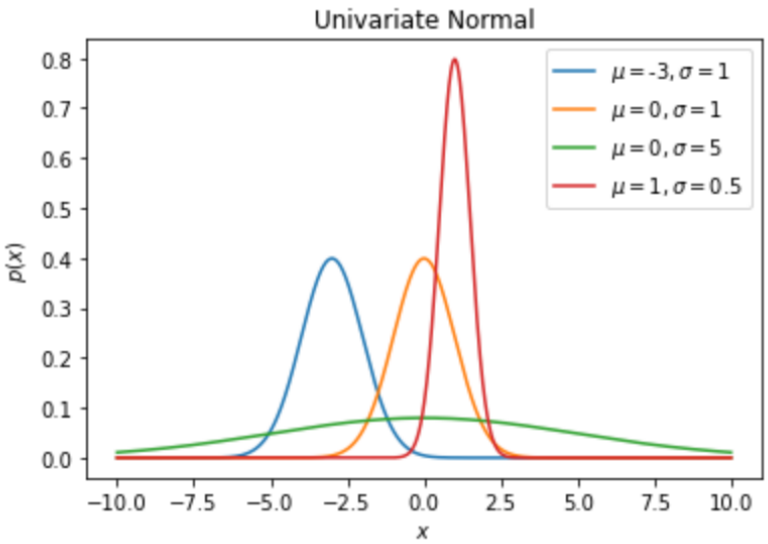
counts of word d in all documents labelled **c**

total word count in all documents labelled **c**

Univariate Gaussian density

Gaussian probability density function (pdf)

$$\mathcal{N}(x;\mu,\sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$



two parameters are μ, σ^2

turn out to be the mean and variance

$$\mathbb{E}[x] = \mu$$

$$\mathbb{E}[(x-\mu)^2] = \sigma^2$$

this is a random variable; we are using the same notation for a random variable and a particular value of that variable

Univariate Gaussian density

Gaussian probability density function (pdf)

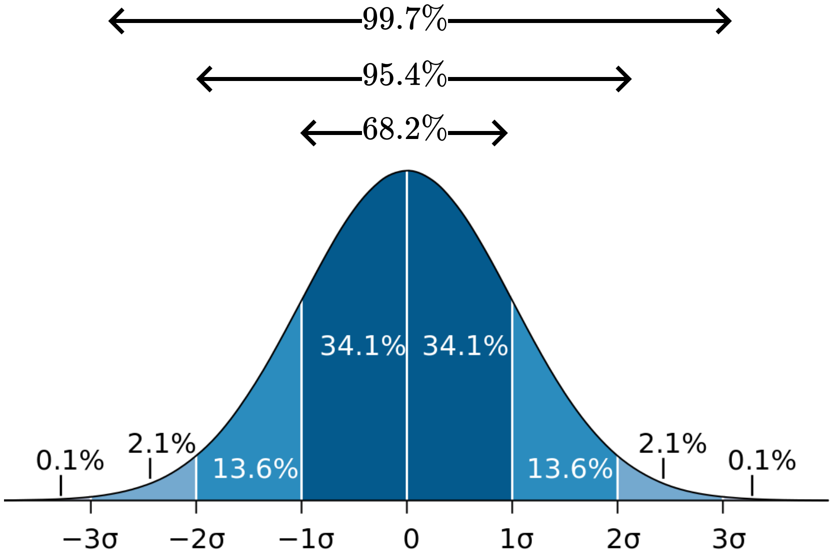
$$\mathcal{N}(x;\mu,\sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

given a dataset $\mathcal{D} = \{x^{(1)}, \ldots, x^{(N)}\}$

maximum likelihood estimate of μ, σ^2
are empirical mean and variance

$$\mu^{MLE} = \frac{1}{N}\sum_n x^{(n)}$$

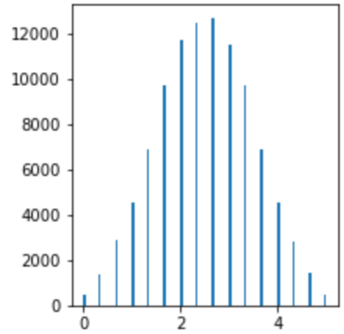
$$\sigma^{2\,MLE} = \frac{1}{N}\sum_n (x^{(n)} - \mu^{MLE})^2$$



Univariate Gaussian density

two reasons why Gaussian is an important dist.

- maximum entropy dist. with a fixed variance
- **central limit theorem**

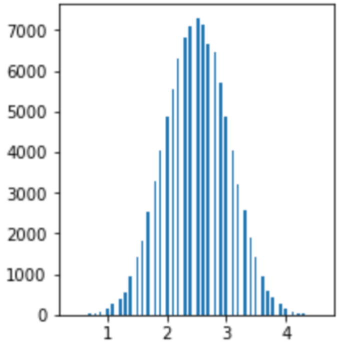


let's throw three dice, repeatedly

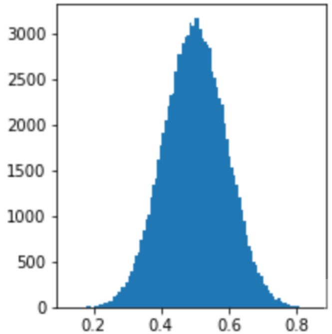
plot the histogram of the average outcome

looks familiar?

lets use 10 dice



let's replace the dice with uniformly distributed values in [0,1]



the average (and sum) of IID random variables has a Gaussian distribution

justifies use of Gaussian for observations that are mean or sum of some random values

Gaussian Naive Bayes

for continuous features one option is the Gaussian conditional likelihood

$$p(x_d \mid y) = \mathcal{N}(x_d; \mu_{d,y}, \sigma^2_{d,y}) = \frac{1}{\sqrt{2\pi{\sigma_{d,y}}^2}} e^{-\frac{(x_d - \mu_{d,y})^2}{2{\sigma_{d,y}}^2}}$$

corresponds to what we previously called $\theta_{d,y}$

Maximum likelihood estimates:

empirical mean & variance of feature x_d across instances with label y

$$\mu_{d,c} = \frac{1}{N(y=c)} \sum_{n=1}^{N} x_d^{(n)} \mathbb{I}(y^{(n)} = c)$$

$$\sigma^2_{d,c} = \frac{1}{N(y=c)} \sum_{n=1}^{N} \mathbb{I}(y^{(n)} = c)(x_d^{(n)} - \mu_{d,y})^2$$

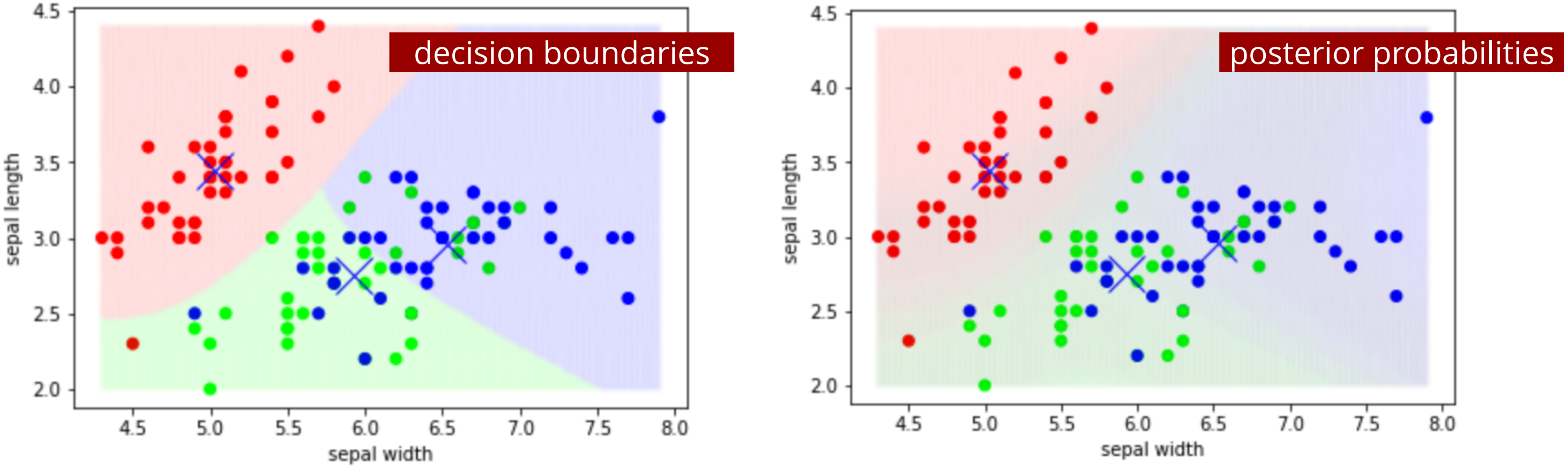
Gaussian Naive Bayes example

classification on **Iris flowers dataset**:

- we use categorical class prior (3 classes)
- Gaussian likelihood since the features are continuous (we use D=2 features)

the **decision boundary** found by Gaussian NB
three means are identified using X

- note that we have a mean $\mu_{d,c}$ and variance $\sigma^2_{d,c}$ for each class-feature combination
- in the plot each X is showing the *combined* mean of two features, sepal length and sepal width.



Summary

- generative classification:
  - learn the class prior and likelihood
  - Bayes rule for conditional class probability
- Naive Bayes
  - assumes conditional independence
    - *e.g., word appearances indep. of each other given document type*
  - class prior: Bernoulli or Categorical
  - likelihood: Bernoulli, Gaussian, Multinomial...
  - MLE has closed-form, estimated separately for each feature and each label
  - Bayesian Naive Bayes helps with overfitting
    - with frequent or rare feature values