Naive Bayes

The MNIST images in these slides were borrowed from these slides.

Probabilistic Classification Models

- Discriminative Model
  - Learns Pr[*C* | *X*] directly
  - Example: Logistic Regression
- Generative Model
  - Learns Pr[*C* | *X*] indirectly
    - P[*C* | *X*] = P[*X*, *C*] / P[*X*] = P[*X* | *C*] P[*C*] / P[*X*]
    - Depends on prior knowledge: P[*C*] is called the prior probability
    - Calculate the likelihood Pr[*X* | *C*]: probability of the features, given the labels
    - Finally, apply Bayes' rule to calculate Pr[*C* | *X*] (the posterier probability)
    - MAP (maximum *a posteriori*) principle: Choose the most likely class *C*
  - Example: Naive Bayes

Probabilistic Generative Model (cont'd)

- A probabilistic generative model tells us: P[*X* | *C*]
- A classification problem: Given *X*, what is *C*?
- Bayes' Rule to the rescue!
- P[*C* | *X*] = P[*X* | *C*] P[*C*] / P[*X*] is called the posterior probability
  - Depends on prior knowledge: P[*C*] is called the prior probability
  - Depends on the likelihood Pr[*X* | *C*]: probability of the feature values, given the labels
  - P[*X*]: total probability of *X*
    - The sum over all classes *c* of the probability of *c* times the probability of *X* given c
  - P[*X*] is difficult to calculate, but happily, we don't need to know it's value …

MNIST Database

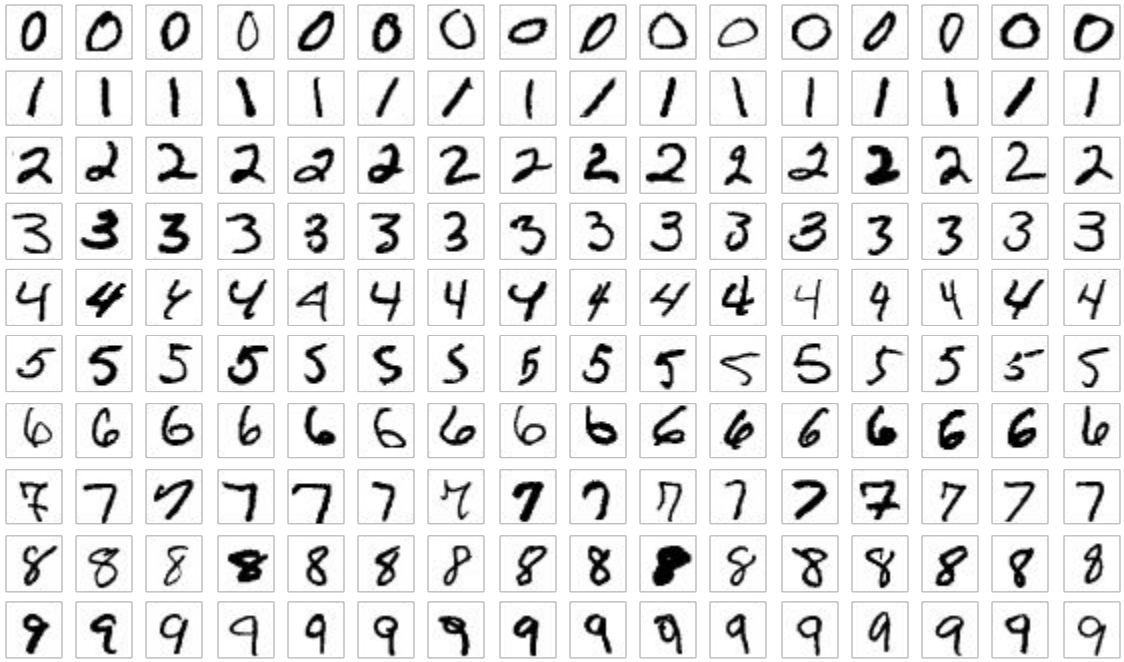
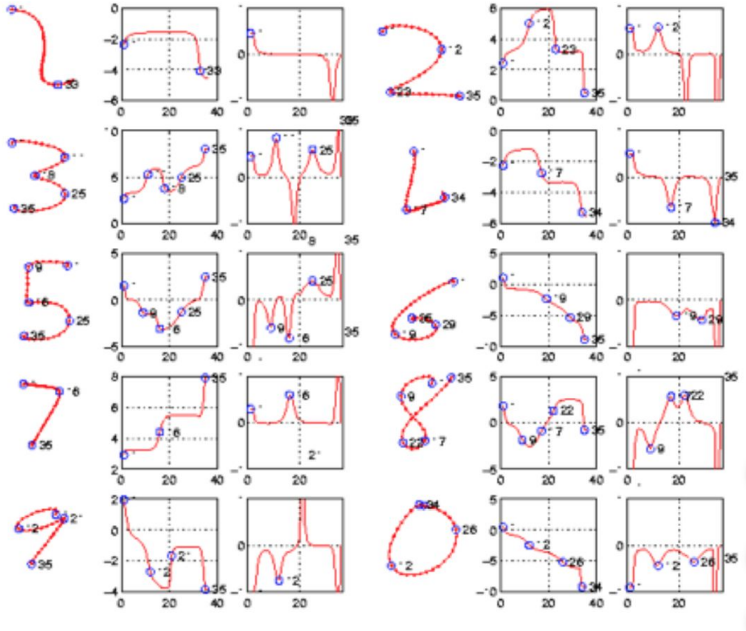


Image Source

MNIST Engineered Generative Model

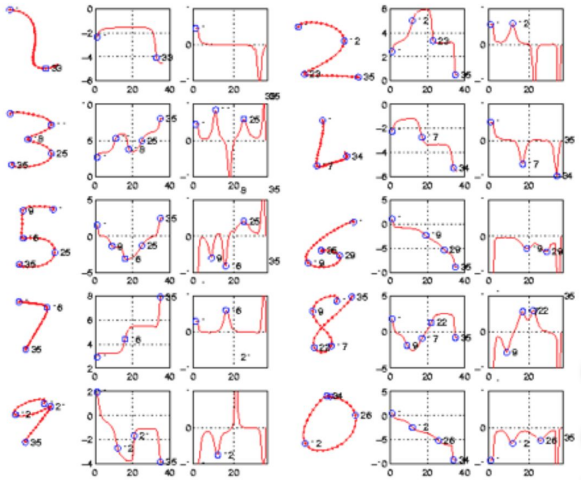
- 0 is a loop
  8 is two loops
  1 is a line
  Etc.
- Difficult
  Lots of exceptions to our rules



(M. Kamvysselis, “Digit recognition in curvature space”, 1999)

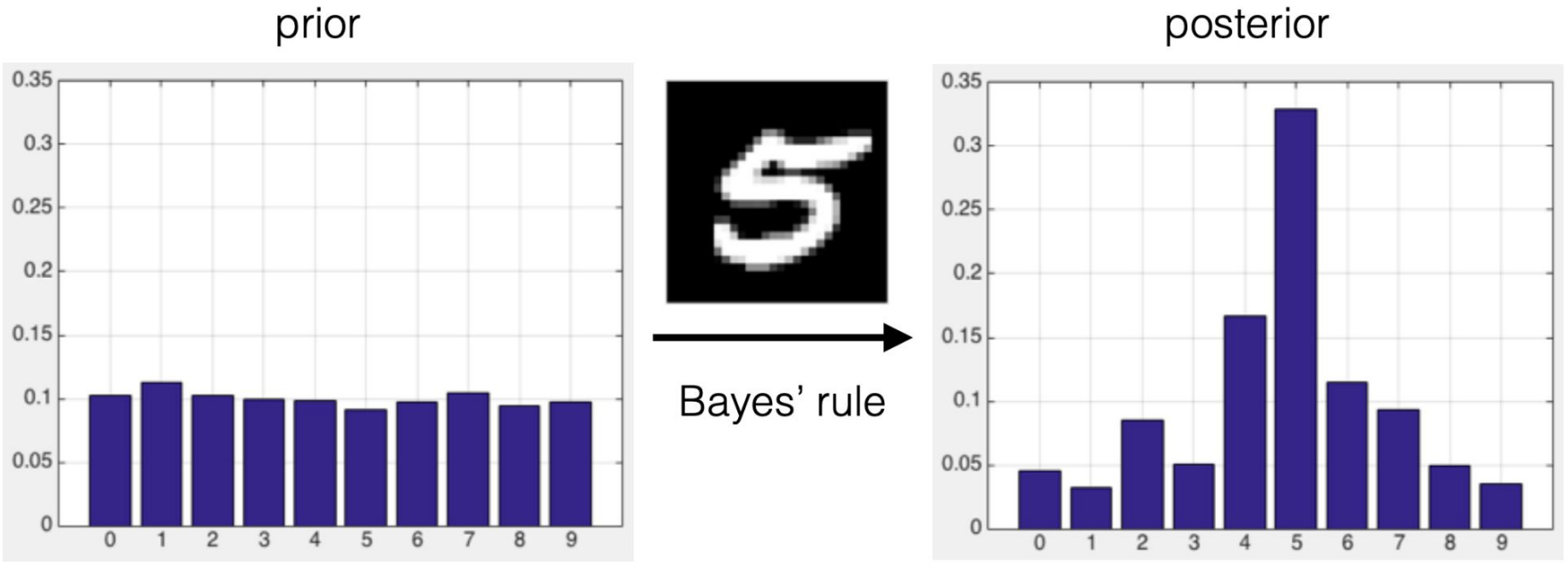
MNIST Engineered Generative Model

- Assume $|C|$ models, one that generates each class $c \in C$
- Such a generative model is a classifier. How?
  - 0 is a loop
    8 is two loops
    1 is a line
    Etc.
  - Difficult
    Lots of exceptions to our rules
- We have a model of each class c
- Given a new observation $X = x$, compute $P[X \mid C]$
- MAP principle: Choose the class c for which x is most probable



(M. Kamvysselis, “Digit recognition in curvature space”, 1999)

MAP (Maximum *a posteriori*) Principle

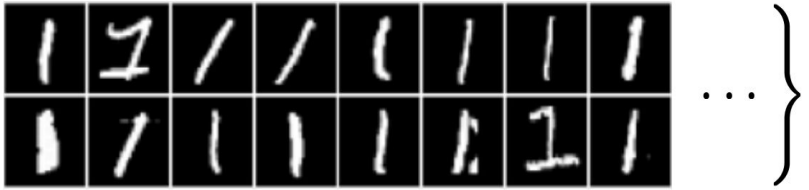


Probabilistic Classification Models

- Learn Pr[$C \mid X, \theta$]
  - Model parameters θ imply a probability of class *C,* given feature values *X*
  - Learn θ that minimizes error / maximizes accuracy
- Maximum *a posteriori* (MAP) principle
  - To classify, choose a class *C* that maximizes P[$C \mid X, \theta$]

$$
\begin{aligned}
c_{\mathrm{MAP}} &\in \operatorname*{argmax}_{c \in C} P(C \mid X) \\
&= \operatorname*{argmax}_{c \in C} \frac{P(X \mid C)\, P(C)}{P(X)} \\
&\propto \operatorname*{argmax}_{c \in C} P(X \mid C)\, P(C)
\end{aligned}
$$

MNIST Learned Generative Model

$P[X \mid 1] =$ 

$P[X \mid 2] =$ 

$P[X \mid 8] =$ 

MNIST Learned Generative Model

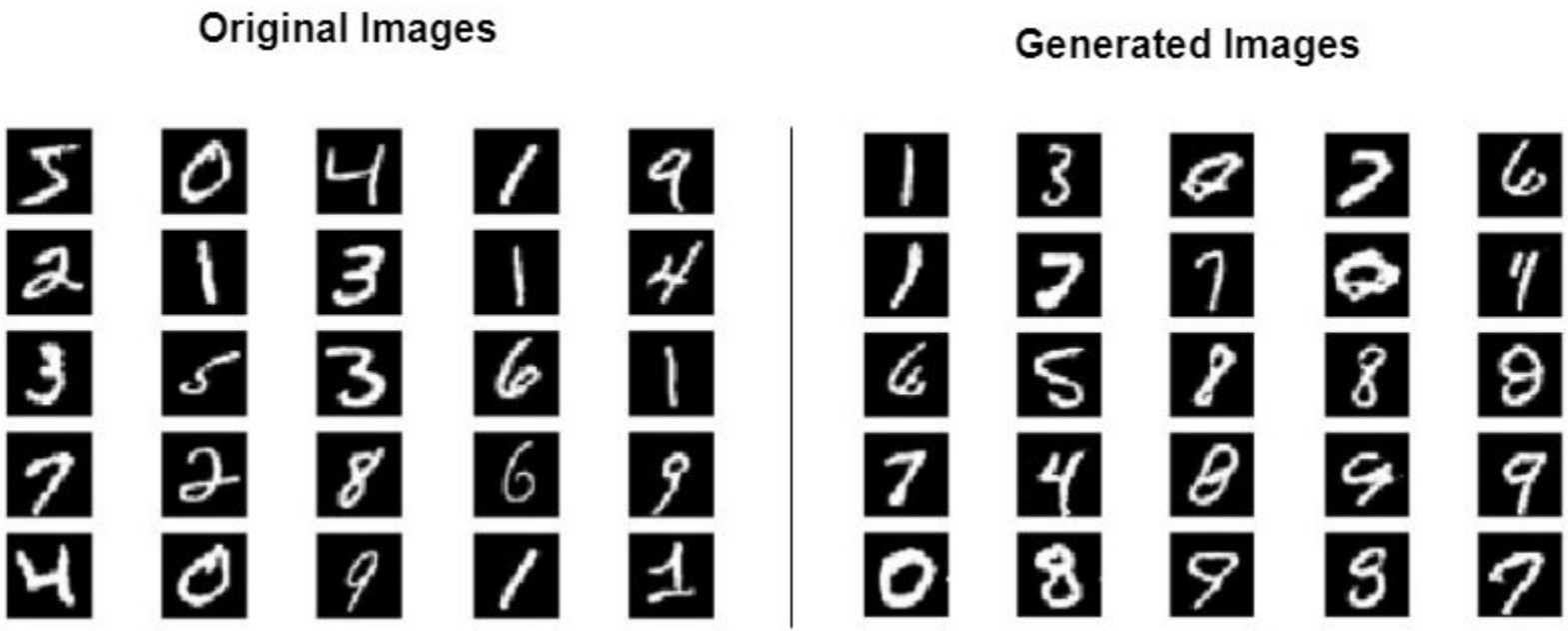


Image Source

Bayesian Networks

Day	Fever?	Coughing?	Headache?	Bodyache?	Flu?
1	Low	None	No	Yes	No
2	Low	None	No	No	No
3	Low	A lot	No	Yes	Yes
4	Mild	A little	No	Yes	Yes
5	High	A little	Yes	Yes	Yes
6	High	A little	Yes	No	No
7	High	A lot	Yes	No	Yes
8	Mild	None	No	Yes	No
9	High	None	Yes	Yes	Yes
10	Mild	A little	Yes	Yes	Yes
11	Mild	None	Yes	No	Yes
12	Mild	A lot	No	No	Yes
13	Low	A lot	Yes	Yes	Yes
14	Mild	A little	No	No	No
15	High	None	No	No	?

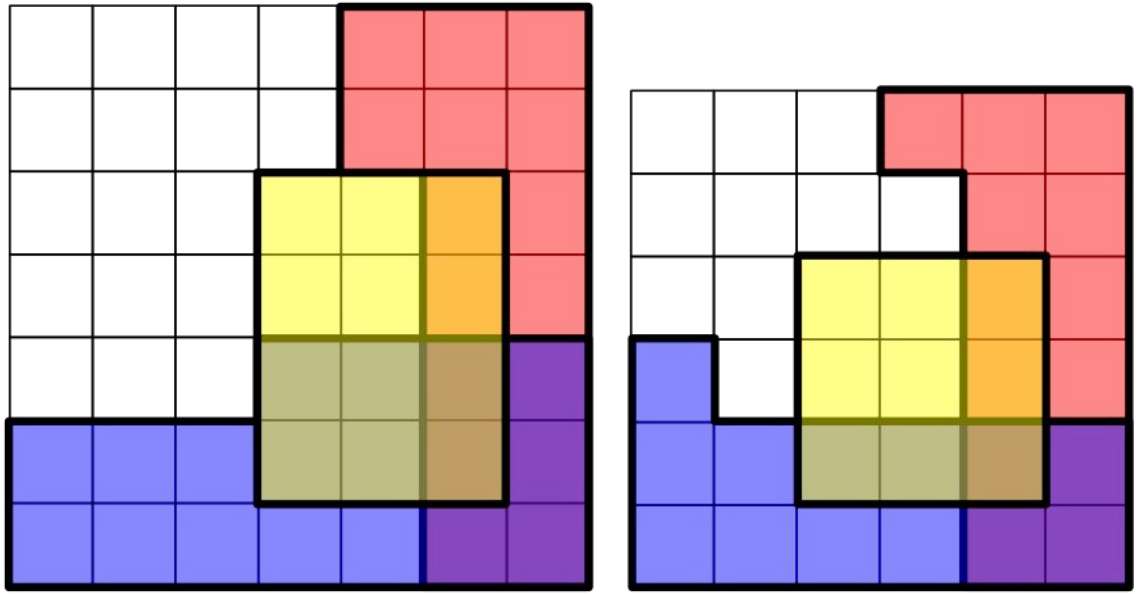
Joint Probability Model

$P[X_1, \ldots, X_n \mid C]$

$P[X \mid C]$ = P[Fever = High, Coughing = None, Headache = No, Bodyache = No | C = Flu]
$P[X \mid C]$ = P[Fever = Low, Coughing = A lot, Headache = No, Bodyache = No | C = Flu]
etc.

3 values of Fever, 3 values of Coughing, 2 values of Headache, 2 values of Bodyache
In total, 3*3*2*2 = 36 probabilities, when C = Flu
Likewise, 36 probabilities, when C = No Flu
In total, 72 probabilities: i.e., 72 model parameters

Conditional Independence



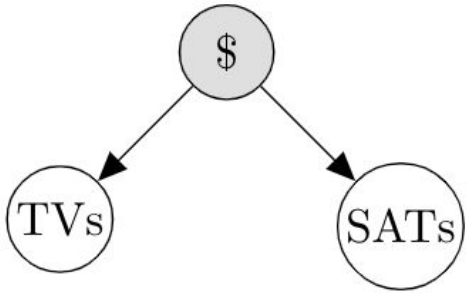
$$\Pr(R \cap B \mid Y) = \Pr(R \mid Y) \Pr(B \mid Y)$$

Image Source

Conditional Independence

Joint Probability Distribution

$$P(\text{SATs}, \text{TVs}, \$) = \text{P}(\text{SATs} \mid \text{TVs}, \$)\,\text{P}(\text{TVs} \mid \$)\,\text{P}(\$)$$



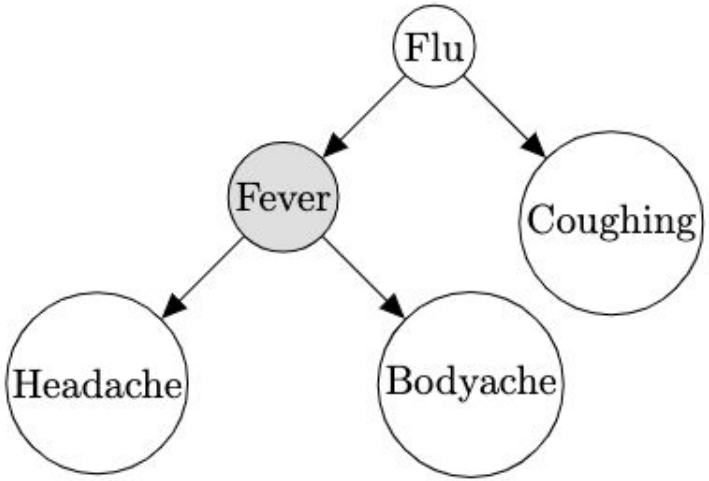
Conditional Independence Assumption

$$P(\text{SATs} \mid \text{TVs}, \$) = \text{P}(\text{SATs} \mid \$)$$

Compact Representation

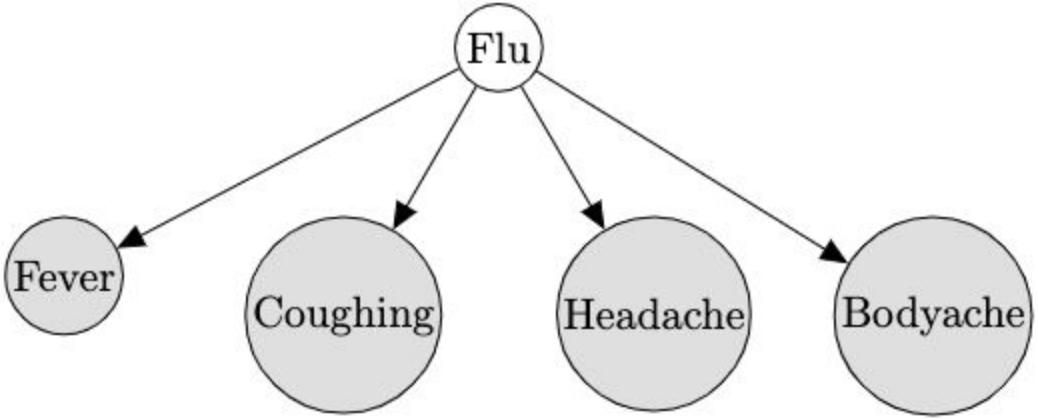
$$P(\text{SATs}, \text{TVs}, \$) = \text{P}(\text{SATs} \mid \$)\,\text{P}(\text{TVs} \mid \$)\,\text{P}(\$)$$

Conditional Independence



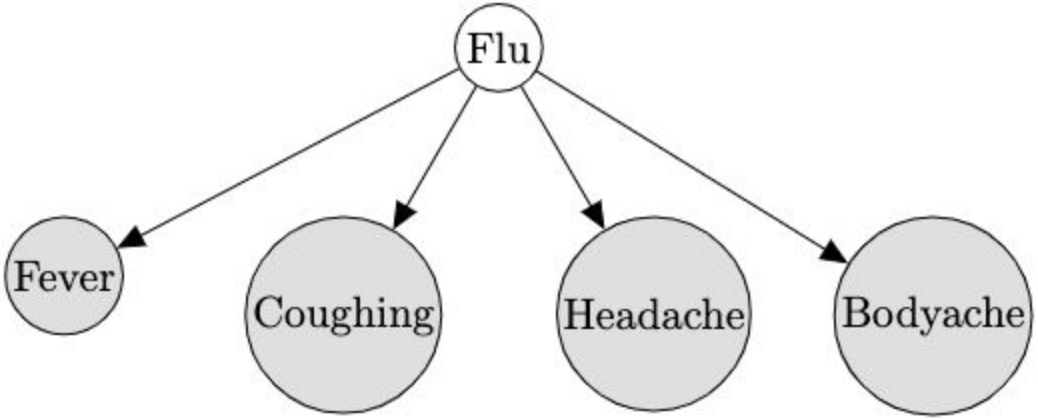
$$
\begin{aligned}
& P(\text{Headache}, \text{Bodyache}, \text{Fever}, \text{Coughing}, \text{Flu}) \\
= & P(\text{Headache} \mid \text{Fever})\, \mathrm{P}(\text{Bodyache} \mid \text{Fever})\, \mathrm{P}(\text{Fever} \mid \text{Flu})\, \mathrm{P}(\text{Coughing} \mid \text{Flu})\, \mathrm{P}(\text{Flu})
\end{aligned}
$$

Naive Bayes' Assumption



$$
\begin{aligned}
& P(\text{Headache}, \text{Bodyache}, \text{Fever}, \text{Coughing}, \text{Flu}) \\
= & P(\text{Fever} \mid \text{Flu})\, \text{P}(\text{Coughing} \mid \text{Flu})\, \text{P}(\text{Headache} \mid \text{Flu})\, \text{P}(\text{Bodyache} \mid \text{Flu})\, \text{P}(\text{Flu})
\end{aligned}
$$

Naive Bayes' Assumption



$$
\begin{aligned}
& P(\text{Headache}, \text{Bodyache}, \text{Fever}, \text{Coughing} \mid \text{Flu}) \\
= & P(\text{Fever} \mid \text{Flu})\, \mathrm{P}(\text{Coughing} \mid \text{Flu})\, \mathrm{P}(\text{Headache} \mid \text{Flu})\, \mathrm{P}(\text{Bodyache} \mid \text{Flu})
\end{aligned}
$$

Naive Bayes' Assumption

$P[X_1, \ldots, X_n \mid C]$

$P[X \mid C]$ = P[Fever = High, Coughing = None, Headache = No, Bodyache = No | C = Flu]
= P[Fever = High | C = Flu] P[Coughing = None | C = Flu]
P[Headache = No | C = Flu] P[Bodyache = No | C = Flu]

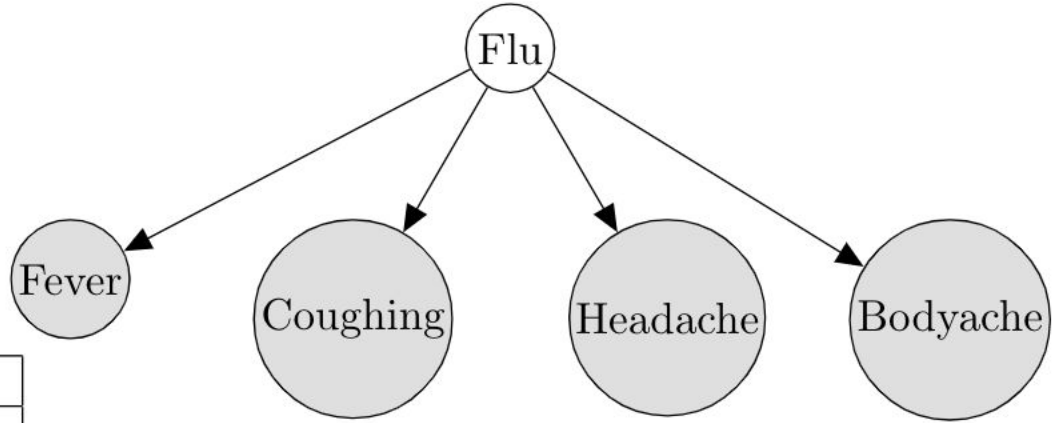
3 values of Fever, 3 values of Coughing, 2 values of Headache, 2 values of Bodyache
In total, 3+3+2+2 = 10 probabilities, when C = Flu
Likewise, 10 probabilities, when C = No Flu
In total, 20 probabilities: i.e., 20 model parameters (down from 72)

Day	Fever?	Coughing?	Headache?	Bodyache?	Flu?
1	Low	None	No	Yes	No
2	Low	None	No	No	No
3	Low	A lot	No	Yes	Yes
4	Mild	A little	No	Yes	Yes
5	High	A little	Yes	Yes	Yes
6	High	A little	Yes	No	No
7	High	A lot	Yes	No	Yes
8	Mild	None	No	Yes	No
9	High	None	Yes	Yes	Yes
10	Mild	A little	Yes	Yes	Yes
11	Mild	None	Yes	No	Yes
12	Mild	A lot	No	No	Yes
13	Low	A lot	Yes	Yes	Yes
14	Mild	A little	No	No	No
15	High	None	No	No	?

P[Flu] = 0.64	P[No Flu] = 0.36
P[Bodyache | Flu] = 0.67	P[Bodyache | No Flu] = 0.40
P[No Bodyache | Flu] = 0.33	P[No Bodyache | No Flu] = 0.60
P[Headache | Flu] = 0.67	P[Headache | No Flu] = 0.20
P[No Headache | Flu] = 0.33	P[No Headache | No Flu] = 0.80
P[A lot | Flu] = 0.44	P[A lot | No Flu] = 0.00
P[A little | Flu] = 0.33	P[A little | No Flu] = 0.40
P[None | Flu] = 0.22	P[None | No Flu] = 0.60
P[High| Flu] = 0.33	P[High | No Flu] = 0.20
P[Mild| Flu] = 0.44	P[Mild | No Flu] = 0.40
P[None| Flu] = 0.22	P[None | No Flu] = 0.40

Priors

Flu	$P(\text{Flu})$
T	0.64
F	0.36



Flu	F	$P(F)$
T	High	0.33
T	Mild	0.44
T	None	0.22
F	High	0.2
F	Mild	0.4
F	None	0.4

Flu	C	$P(C)$
T	A lot	0.44
T	A little	0.33
T	None	0.22
F	A lot	0.0
F	A little	0.4
F	None	0.6

Flu	H	$P(H)$
T	T	0.67
T	F	0.33
F	T	0.20
F	F	0.80

Flu	B	$P(B)$
T	T	0.67
T	F	0.33
F	T	0.40
F	F	0.60

X = [High, No (Cough), No (Headache), No (Bodyache)]

- P[Flu | X] = P[High | Flu] P[None | Flu] P[No | Yes] P[No | Flu] P[Flu]
  = [0.33] [0.22] [0.33] [0.33] [0.64]
  = 0.0051
- P[NF | X] = P[High | NF] P[None | NF] P[No | NF] P[No | NoFlu] P[NF]
  = [0.20] [0.60] [0.80] [0.60] [0.36]
  = 0.0069

P[No Flu | X] = 0.0069 > 0.0051 = P[Flu | X]

So our Naive Bayes classifier outputs No Flu

MLE for Naive Bayes

MLE for Naive Bayes

$$
\begin{aligned}
& \underset{\theta}{\operatorname{argmax}} \prod_{i=1}^{n} P(\boldsymbol{x}_i, y_i \mid \theta) \\
= \; & \underset{\theta}{\operatorname{argmax}} \prod_{i=1}^{n} P(\boldsymbol{x}_i \mid y_i, \theta) P(y_i \mid \theta) \\
= \; & \underset{\theta}{\operatorname{argmax}} \prod_{i=1}^{n} \prod_{j=1}^{m} P(x_{ij} \mid y_i, \theta) P(y_i \mid \theta) \\
= \; & \underset{\theta}{\operatorname{argmax}} \log \prod_{i=1}^{n} \prod_{j=1}^{m} P(x_{ij} \mid y_i, \theta) P(y_i \mid \theta) \\
= \; & \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^{n} \sum_{j=1}^{m} \log P(x_{ij} \mid y_i, \theta) + \log P(y_i \mid \theta)
\end{aligned}
$$

Naive Bayes Assumption

MLE for Naive Bayes (cont'd)

Likelihood of feature values, given class labels:

$$P(x_{ij} = 1 \mid y_i = 1) = a_j \quad \text{and} \quad P(x_{ij} = 0 \mid y_i = 1) = 1 - a_j$$

$$P(x_{ij} = 1 \mid y_i = 0) = b_j \quad \text{and} \quad P(x_{ij} = 0 \mid y_i = 0) = 1 - b_j$$

$$P(x_{ij} \mid y_i, a_j, b_j) = a_j^{y_i x_{ij}} (1 - a_j)^{y_i (1 - x_{ij})} b_j^{(1 - y_i) x_{ij}} (1 - b_j)^{(1 - y_i)(1 - x_{ij})}$$

MLE for Naive Bayes (cont'd)

Likelihood of class labels:

$$P(y_i = 1) = p \quad \text{and} \quad P(y_i = 0) = 1 - p$$

$$P(y_i \mid p) = p^{y_i}(1-p)^{(1-y_i)}$$

MLE for Naive Bayes (cont'd)

Plug P(x_{ij}, y_i | θ) and P(y_i | θ) back into the log likelihood function.

Use calculus to solve for the optimal θs: i.e., take derivatives & set equal to zero.

$$p = \frac{\text{Count}\,(y_i = 1)}{\text{Count}\,(y_i = 1) + \text{Count}\,(y_i = 0)}$$

$$a_j = \frac{\text{Count}\,(y_i = 1, x_{ij} = 1)}{\text{Count}\,(y_i = 1)}$$

$$b_j = \frac{\text{Count}\,(y_i = 0, x_{ij} = 1)}{\text{Count}\,(y_i = 0)}$$

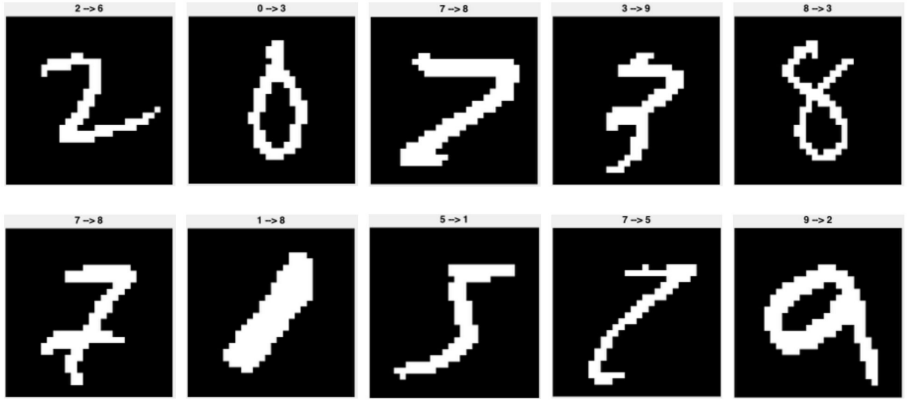
Back to MNIST: Bernoulli Model

The parameter vector θ consists of p, a_j, and b_j, for all $j \in \{1, \ldots, m\}$.

There are 784 features, so there are 784 a_j and 784 b_j parameters, so 1569 in total.

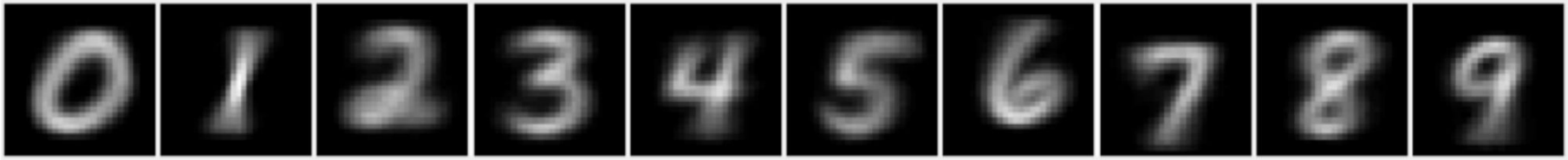
In the full joint, there are $2^{784} - 1$ parameters (per class, and there are 10 classes).

15% error rate on
10,000 test images

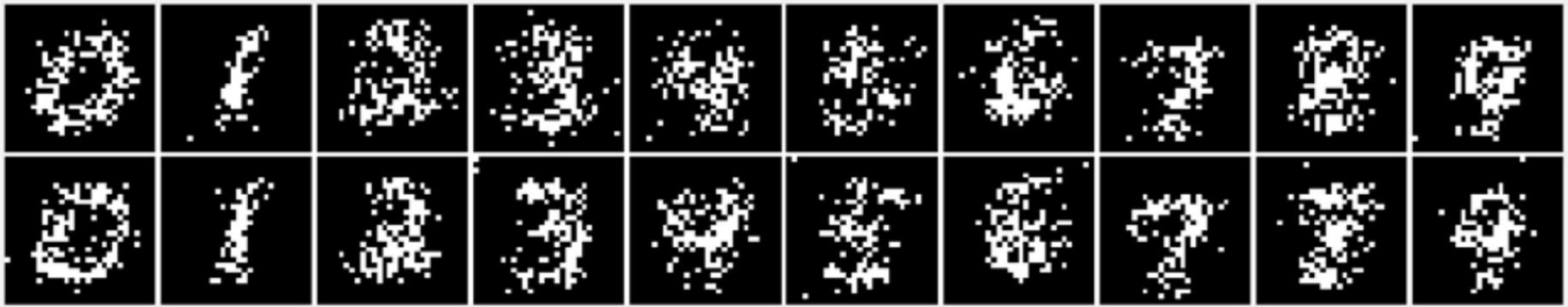


Naive Bayes

Means:

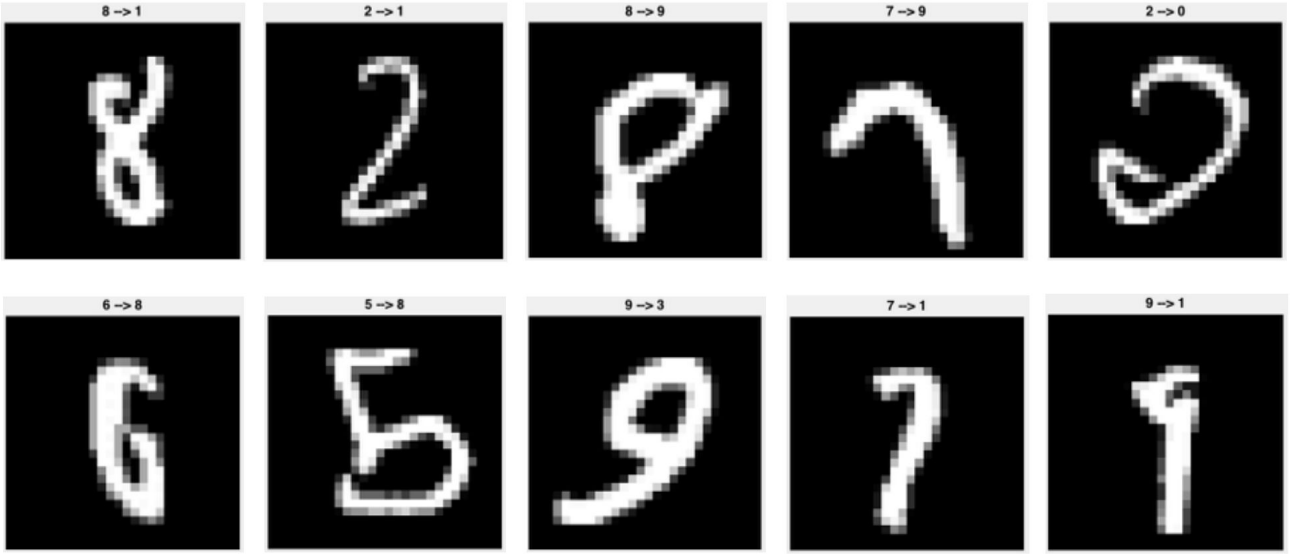


Samples:



Back to MNIST: Gaussian Model

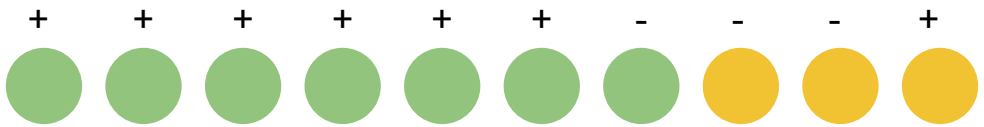
< 5% error rate on
10,000 test images



Extras

Naive Bayes

We estimate the requisite probabilities by counting.

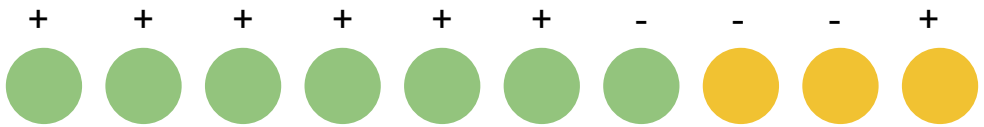


Naive Bayes (cont'd)

We estimate the requisite probabilities by counting.

P[•] = ? P[•] = ? P[+] = ? P[-] = ?

P[•|+] = ? P[•|-] = ? P[•|+] = ? P[•|-] = ?

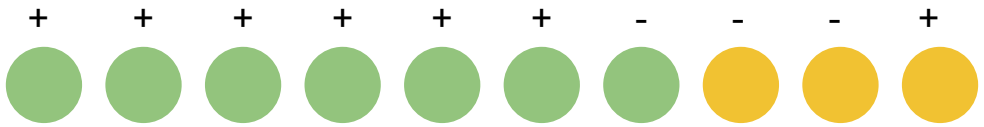


Naive Bayes (cont’d)

We estimate the requisite probabilities by counting.

P[•] = 7/10 P[•] = 3/10 P[+] = 7/10 P[-] = 3/10

P[•|+] = 6/7 P[•|-] = 1/3 P[•|+] = 1/7 P[•|-] = 2/3



Naive Bayes (cont'd)

We estimate the requisite probabilities by counting.

P[•] = 7/10 P[•] = 3/10 P[+] = 7/10 P[-] = 3/10

P[•|+] = 6/7 P[•|-] = 1/3 P[•|+] = 1/7 P[•|-] = 2/3

Great!

Now how do we classify a new •?

Naive Bayes (cont'd)

We estimate the requisite probabilities by counting.

P[•] = 7/10 P[•] = 3/10 P[+] = 7/10 P[-] = 3/10

P[•|+] = 6/7 P[•|-] = 1/3 P[•|+] = 1/7 P[•|-] = 2/3

P[+|•] = P[•|+] P[+] = (6/7)(7/10) = 6/10.

P[-|•] = P[•|-] P[-] = (1/3)(3/10) = 1/10.

So • is classified as a +.

Naive Bayes (cont'd)

We estimate the requisite probabilities by counting.

P[●] = 7/10 P[●] = 3/10 P[+] = 7/10 P[-] = 3/10

P[●|+] = 6/7 P[●|-] = 1/3 P[●|+] = 1/7 P[●|-] = 2/3

P[+|●] = P[●|+] P[+] = (1/7)(7/10) = 1/10.

P[-|●] = P[●|-] P[-] = (2/3)(3/10) = 2/10.

So ● is classified as a -.

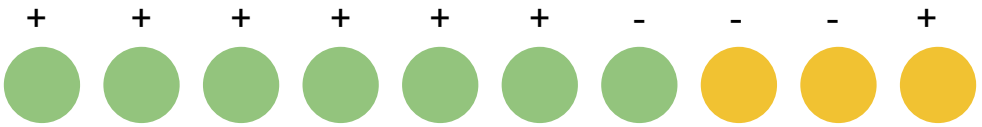
Sanity Check

We estimate the requisite probabilities by counting.

P[•] = 7/10 P[•] = 3/10 P[+] = 7/10 P[-] = 3/10

P[•|+] = 6/7 P[•|-] = 1/3 P[•|+] = 1/7 P[•|-] = 2/3

P[+|•] = 6/7 P[-|•] = 1/7 P[+|•] = 1/3 P[-|•] = ⅔



P[+|•] > P[-|•], so • is classified as a +, and P[-|•] > P[+|•] so • is classified as a -.

In-class Activity

Training data:

Day	Outlook	Temp	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Test data:

D15	Sunny	Cool	High	Strong	?

P[Yes] = 0.64	P[No] = 0.36
P[Weak | Yes] = 0.67	P[Weak | No] = 0.40
P[Strong | Yes] = 0.33	P[Strong | No] = 0.60
P[High | Yes] = 0.33	P[High | No] = 0.80
P[Normal | Yes] = 0.67	P[Normal | No] = 0.20
P[Hot | Yes] = 0.22	P[Hot | No] = 0.40
P[Mild | Yes] = 0.44	P[Mild | No] = 0.40
P[Cool | Yes] = 0.33	P[Cool | No] = 0.20
P[Sunny | Yes] = 0.22	P[Sunny | No] = 0.60
P[Overcast | Yes] = 0.44	P[Overcast | No] = 0.00
P[Rain | Yes] = 0.33	P[Rain | No] = 0.40

X = [Sunny, Cool, High, Strong]

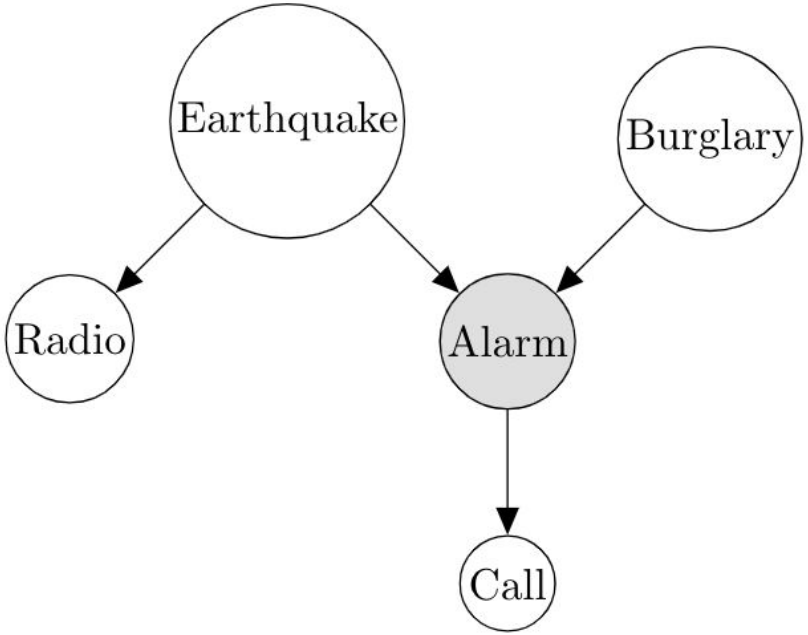
- P[Yes | X] = P[Sunny | Yes] P[Cool | Yes] P[High | Yes] P[Strong | Yes] P[Yes]
  = [0.22] [0.33] [0.33] [0.33] [0.64] = 0.0051
- P[No | X] = P[Sunny | No] P[Cool | No] P[High | No] P[Strong | No] P[No]
  = [0.60] [0.20] [0.80] [0.60] [0.36] = 0.0069

P[No | X] > P[Yes | X]

So our NB classifier outputs No

Skipped Slides

Conditional Independence



$$P(\text{Call} \mid \text{Radio}, \text{Alarm}) = \text{P}(\text{Call} \mid \text{Alarm})$$