Lecture 16:
Naïve Bayes
Fall 2022

Kai-Wei Chang

CS @ UCLA

kw+cm146@kwchang.net

The instructor gratefully acknowledges Dan Roth, Vivek Srikuar, Sriram Sankararaman, Fei Sha, Ameet Talwalkar, Eric Eaton, and Jessica Wu whose slides are heavily used, and the many others who made their course material freely available online.

Bayes Theorem

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Posterior probability: What is the probability of Y given that X is observed?

Likelihood: What is the likelihood of observing X given a specific Y?

Prior probability: What is our belief in Y before we see X?

MAP prediction

Let's be use the Bayes rule for predicting y given an input **x**

$$P(Y = y|X = \mathbf{x}) = \frac{P(X = \mathbf{x}|Y = y)P(Y = y)}{P(X = \mathbf{x})}$$

Predict y for the input **x** using

$$\arg\max_y P(X = \mathbf{x}|Y = y)P(Y = y)$$

MAP prediction

Don't confuse with *MAP learning*: finds hypothesis by

$$h_{MAP} = \arg\max_{h \in H} P(D|h)P(h)$$

Let's be use the Bayes rule for predicting y given an input **x**

$$P(Y = y|X = \mathbf{x}) = \frac{P(X = \mathbf{x}|Y = y)P(Y = y)}{P(X = \mathbf{x})}$$

Predict y for the input **x** using

$$\arg\max_{y} P(X = \mathbf{x}|Y = y)P(Y = y)$$

MAP prediction

Predict y for the input **x** using

$$\arg\max_y \underbrace{P(X = \mathbf{x}|Y = y)}_{\text{Likelihood of observing this input } \mathbf{x} \text{ when the label is y}} \underbrace{P(Y = y)}_{\text{Prior probability of the label being y}}$$

Likelihood of observing this input **x** when the label is y

Prior probability of the label being y

All we need are these two sets of probabilities

Example: Tennis

Prior

Play tennis	P(Play tennis)
Yes	0.3
No	0.7

Without any other information, what is the prior probability that I should play tennis?

Likelihood

Temperature	Wind	P(T, W |Tennis = Yes)
Hot	Strong	0.15
Hot	Weak	0.4
Cold	Strong	0.1
Cold	Weak	0.35

On days that I do play tennis, what is the probability that the temperature is T and the wind is W?

Temperature	Wind	P(T, W |Tennis = No)
Hot	Strong	0.4
Hot	Weak	0.1
Cold	Strong	0.3
Cold	Weak	0.2

On days that I don't play tennis, what is the probability that the temperature is T and the wind is W?

Example: Tennis

Prior

Play tennis	P(Play tennis)
Yes	0.3
No	0.7

Likelihood

Temperature	Wind	P(T, W |Tennis = Yes)
Hot	Strong	0.15
Hot	Weak	0.4
Cold	Strong	0.1
Cold	Weak	0.35

Temperature	Wind	P(T, W |Tennis = No)
Hot	Strong	0.4
Hot	Weak	0.1
Cold	Strong	0.3
Cold	Weak	0.2

Input:
Temperature = Hot (H)
Wind = Weak (W)

Should I play tennis?

Example: Tennis

Prior

Play tennis	P(Play tennis)
Yes	0.3
No	0.7

Likelihood

Temperature	Wind	P(T, W |Tennis = Yes)
Hot	Strong	0.15
Hot	Weak	0.4
Cold	Strong	0.1
Cold	Weak	0.35

Temperature	Wind	P(T, W |Tennis = No)
Hot	Strong	0.4
Hot	Weak	0.1
Cold	Strong	0.3
Cold	Weak	0.2

Input:
Temperature = Hot (H)
Wind = Weak (W)

Should I play tennis?

argmax_y P(H, W | play?) P (play?)

P(H, W | Yes) P(Yes) = 0.4 x 0.3
= 0.12

P(H, W | No) P(No) = 0.1 x 0.7
= 0.07

MAP prediction = Yes

How hard is it to learn probabilistic models?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Outlook: S(unny), O(vercast), R(ainy)

Temperature: H(ot), M(edium), C(ool)

Humidity: H(igh), N(ormal), L(ow)

Wind: S(trong), W(eak)

How hard is it to learn probabilistic models?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Outlook: S(unny), O(vercast), R(ainy)

Te

Hu

N(ormal), L(ow)

Wind: S(trong), W(eak)

We need to learn

1. The prior P(Play?)
2. The likelihoods P(X | Play?)

How hard is it to learn probabilistic models?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Prior P(play?)

- A single number

How hard is it to learn probabilistic models?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Prior P(play?)

- A single number (Why only one?)

Likelihood P(**X** | Play?)

- There are 4 features
- For each value of Play? (+/-), we need a value for each possible assignment: $P(x_1, x_2, x_3, x_4 \mid \text{Play?})$

How hard is it to learn probabilistic models?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-
	3	3	3	2	

Values for this feature

Prior P(play?)

- A single number (Why only one?)

Likelihood P(**X** | Play?)

- There are 4 features
- For each value of Play? (+/-), we need a value for each possible assignment: $P(x_1, x_2, x_3, x_4 \mid Play?)$

How hard is it to learn probabilistic models?

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-
	3	3	3	2	

Values for this feature

Prior P(play?)

- A single number (Why only one?)

Likelihood P(**X** | Play?)

- There are 4 features
- For each value of Play? (+/-), we need a value for each possible assignment: $P(x_1, x_2, x_3, x_4 \mid \text{Play?})$
- $\boxed{(3 \cdot 3 \cdot 3 \cdot 2\ - 1)}$ parameters in each case

One for each assignment

How hard is it to learn probabilistic models?

Prior P(Y)

- If there are k labels, then k – 1 parameters

Likelihood P(**X** | Y)

- We need a value for each possible $P(x_1, x_2, \cdots, x_d \mid y)$ for each y
- Assume we have K binary features, how many parameters we need?

Need a lot of data to estimate these many numbers!

High model complexity

If there is very limited data, high variance in the parameters

How can we deal with this?

Answer: Make independence assumptions

How hard is it to learn probabilistic models?

Prior P(Y)

- If there are k labels, then k – 1 parameters

Likelihood P(**X** | Y)

- We need a value for each possible $P(x_1, x_2, \cdots, x_d \mid y)$ for each y
- Assume we have n binary features, how many parameters we need? $(2^n - 1)K$

Need a lot of data to estimate these many numbers!

High model complexity

If there is very limited data, high variance in the parameters

How can we deal with this?

Answer: Make independence assumptions

Recall: Conditional independence

Suppose X, Y and Z are random variables

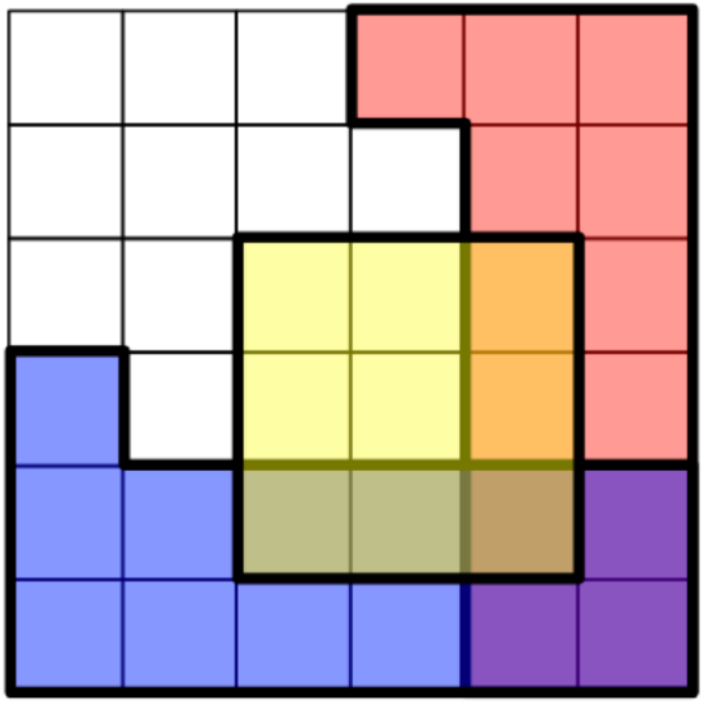
X is *conditionally independent* of Y given Z if the probability distribution of X is independent of the value of Y when Z is observed

$$P(X|Y,Z) = P(X|Z)$$

Or equivalently

$$P(X,Y|Z) = P(X|Z)P(Y|Z)$$

conditionally independent != independent



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

$$\Pr(R, B) \neq \Pr(R)\Pr(B)$$

https://en.wikipedia.org/wiki/Conditional_independence

Modeling the features

$P(x_1, x_2, \cdots, x_d | y)$ required a lot of parameters

Consider we have d Boolean features for k class, we need $k(2^d - 1)$ parameters

What if all the features were conditionally independent given the label?

Modeling the features

$P(x_1, x_2, \cdots, x_d | y)$ required $k(2^d - 1)$ parameters

What if all the features were conditionally independent given the label?

The Naïve Bayes Assumption

That is,

$$P(x_1, x_2, \cdots, x_d | y) = P(x_1|y)P(x_2|y) \cdots P(x_d|y)$$

Requires only d numbers for each label. kd parameters overall. Not bad!

The Naïve Bayes Classifier

Assumption: Features are conditionally independent given the label Y

To predict, we need two sets of probabilities

- Prior P(y)
- For each x_j, we have the likelihood $P(x_j \mid y)$

The Naïve Bayes Classifier

Assumption: Features are conditionally independent given the label Y

To predict, we need two sets of probabilities

- Prior P(y)
- For each x_j, we have the likelihood $P(x_j \mid y)$

Decision rule

$$h_{NB}(\boldsymbol{x}) = \underset{y}{\operatorname{argmax}}\, P(y) P(x_1, x_2, \cdots, x_d | y)$$

The Naïve Bayes Classifier

Assumption: Features are conditionally independent given the label Y

To predict, we need two sets of probabilities

- Prior P(y)
- For each x_j, we have the likelihood $P(x_j \mid y)$

Decision rule

$$h_{NB}(\boldsymbol{x}) = \underset{y}{\operatorname{argmax}}\, P(y) P(x_1, x_2, \cdots, x_d | y)$$
$$= \underset{y}{\operatorname{argmax}}\, P(y) \prod_j P(x_j | y)$$

Decision boundaries of naïve Bayes

What is the decision boundary of the naïve Bayes classifier?

Consider the two class case. We predict the label to be + if

$$P(y=+)\prod_j P(x_j|y=+) > P(y=-)\prod_j P(x_j|y=-)$$

Decision boundaries of naïve Bayes

What is the decision boundary of the naïve Bayes classifier?

Consider the two class case. We predict the label to be + if

$$P(y=+)\prod_j P(x_j|y=+) > P(y=-)\prod_j P(x_j|y=-)$$

$$\frac{P(y=+)\prod_j P(x_j|y=+)}{P(y=-)\prod_j P(x_j|y=-)} > 1$$

Decision boundaries of naïve Bayes

What is the decision boundary of the naïve Bayes classifier?

Taking log and simplifying, we can show that the decision boundary of naïve Bayes is a linear function

$$\log \frac{P(y=-|\boldsymbol{x})}{P(y=+|\boldsymbol{x})} = \log \frac{P(y=+)\prod_j P(x_j|y=+)}{P(y=-)\prod_j P(x_j|y=-)}$$

$$= \log P(y=+) - \log P(y=-) + \sum_j (\log P(x_j|y=+) - \log P(x_j|y=-))$$

This is a linear function of the feature space!

Today's lecture

- The naïve Bayes Classifier
- Learning the naïve Bayes Classifier
- Generative model

Learning the naïve Bayes Classifier

- What is the hypothesis function h defined by?
  - A collection of probabilities
    - Prior for each label: $P(y)$
    - Likelihoods for feature x_j given a label: $P(x_j | y)$

Suppose we have a data set $D = \{(\boldsymbol{x}_i, y_i)\}$ with m examples

How we estimate $P(y)$ and $P(x_j | y)$

For example, suppose we have $y \in \{1,0\}$ and all features are binary

- **Prior**: $P(y = 1) = p$ and $P(y = 0) = 1 - p$
- **Likelihood** for each feature given a label
  - $P(x_j = 1 \mid y = 1) = a_j$ and $P(x_j = 0 \mid y = 1) = 1 - a_j$
  - $P(x_j = 1 \mid y = 0) = b_j$ and $P(x_j = 0 \mid y = 0) = 1 - b_j$

Learning the naïve Bayes Classifier

Maximum likelihood estimation

$$h_{ML} = \arg\max_{h \in H} P(D|h)$$

Here h is defined by all the probabilities used to construct the naïve Bayes decision

Maximum likelihood estimation

Given a dataset $D = \{(\boldsymbol{x}_i, y_i)\}$ with m examples

$$h_{ML} = \arg\max_{h \in H} P(D|h)$$

$$h_{ML} = \arg\max_{h} \prod_{i=1}^{m} P((\mathbf{x}_i, y_i)|h)$$

Each example in the dataset is independent and identically distributed

So we can represent P(D| h) as this product

Maximum likelihood estimation

Given a dataset $D = \{(\boldsymbol{x}_i, y_i)\}$ with m examples

$$h_{ML} = \arg\max_h \prod_{i=1}^{m} \boxed{P((\mathbf{x}_i, y_i)|h)}$$

Each example in the dataset is independent and identically distributed

So we can represent P(D| h) as this product

Asks "What probability would this particular h assign to the pair ($\mathbf{x}_i$, y_i)?"

Maximum likelihood estimation

Given a dataset D = {(x_i, y_i)} with m examples

$$
\begin{aligned}
h_{ML} &= \arg\max_h \prod_{i=1}^{m} P((\mathbf{x}_i, y_i)|h) \\
&= \arg\max_h \prod_{i=1}^{m} P(\mathbf{x}_i|y_i, h)P(y_i|h)
\end{aligned}
$$

Maximum likelihood estimation

Given a dataset D = {(x_i, y_i)} with m examples

$$
\begin{aligned}
h_{ML} &= \arg\max_h \prod_{i=1}^{m} P((\mathbf{x}_i, y_i)|h) \\
&= \arg\max_h \prod_{i=1}^{m} \boxed{P(\mathbf{x}_i|y_i, h)} P(y_i|h) \\
&= \arg\max_h \prod_{i=1}^{m} P(y_i|h) \prod_j \boxed{P(x_{i,j}|y_i, h)}
\end{aligned}
$$

$\mathbf{x}_{ij}$ is the j^{th} feature of $\mathbf{x}_i$

The Naïve Bayes assumption

Maximum likelihood estimation

Given a dataset D = {(x_i, y_i)} with m examples

$$
\begin{aligned}
h_{ML} &= \arg\max_h \prod_{i=1}^{m} P((\mathbf{x}_i, y_i)|h) \\
&= \arg\max_h \prod_{i=1}^{m} P(\mathbf{x}_i|y_i, h) P(y_i|h) \\
&= \arg\max_h \prod_{i=1}^{m} P(y_i|h) \prod_j P(x_{i,j}|y_i, h)
\end{aligned}
$$

How do we proceed?

Maximum likelihood estimation

Given a dataset D = {(x_i, y_i)} with m examples

$$\begin{aligned}
h_{ML} &= \arg\max_h \prod_{i=1}^{m} P((\mathbf{x}_i, y_i)|h) \\
&= \arg\max_h \prod_{i=1}^{m} P(\mathbf{x}_i|y_i, h)P(y_i|h) \\
&= \arg\max_h \prod_{i=1}^{m} P(y_i|h) \prod_j P(x_{i,j}|y_i, h) \\
&= \arg\max_h \sum_{i=1}^{m} \log P(y_i|h) + \sum_i \sum_j \log P(x_{i,j}|y_i, h)
\end{aligned}$$

Learning the naïve Bayes Classifier

Maximum likelihood estimation

$$h_{ML} = \arg\max_h \sum_{i=1}^{m} \log P(y_i|h) + \sum_i \sum_j \log P(x_{i,j}|y_i, h)$$

What next?

Learning the naïve Bayes Classifier

Maximum likelihood estimation

$$h_{ML} = \arg\max_h \sum_{i=1}^{m} \log P(y_i|h) + \sum_i \sum_j \log P(x_{i,j}|y_i, h)$$

Assume

- **Prior**: $P(y = 1) = p$ and $P(y = 0) = 1 - p$
- **Likelihood** for each feature given a label
  - $P(x_j = 1 \mid y = 1) = a_j$ and $P(x_j = 0 \mid y = 1) = 1 - a_j$
  - $P(x_j = 1 \mid y = 0) = b_j$ and $P(x_j = 0 \mid y = 0) = 1 - b_j$

Learning the naïve Bayes Classifier

Maximum likelihood estimation

$$h_{ML} = \arg\max_h \sum_{i=1}^{m} \log P(y_i|h) + \sum_i \sum_j \log P(x_{i,j}|y_i, h)$$

For simplicity, suppose there are two labels 1 and 0 and all features are binary

- **Prior**: P(y = 1) = p and P (y = 0) = 1 – p
- **Likelihood** for each feature given a label
  - $P(x_j = 1 \mid y = 1) = a_j$ and $P(x_j = 0 \mid y = 1) = 1 - a_j$
  - $P(x_j = 1 \mid y = 0) = b_j$ and $P(x_j = 0 \mid y = 0) = 1 - b_j$

Learning the naïve Bayes Classifier

Maximum likelihood estimation

$$h_{ML} = \arg\max_h \sum_{i=1}^{m} \log P(y_i|h) + \sum_i \sum_j \log P(x_{i,j}|y_i, h)$$

- Prior: P(y = 1) = p and P (y = 0) = 1 – p

$$P(y_i|h) = p^{[y_i=1]}(1-p)^{[y_i=0]}$$

[z] is called the indicator function or the Iverson bracket

Its value is 1 if the argument z is true and zero otherwise

Learning the naïve Bayes Classifier

Maximum likelihood estimation

$$h_{ML} = \arg\max_h \sum_{i=1}^{m} \log P(y_i|h) + \sum_i \sum_j \log P(x_{i,j}|y_i, h)$$

Likelihood for each feature given a label

- $P(x_j = 1 \mid y = 1) = a_j$ and $P(x_j = 0 \mid y = 1) = 1 - a_j$
- $P(x_j = 1 \mid y = 0) = b_j$ and $P(x_j = 0 \mid y = 0) = 1 - b_j$

$$P(x_{ij}|y_i, h) = a_j^{[y_i=1, x_{ij}=1]} \times (1 - a_j)^{[y_i=1, x_{ij}=0]} \times b_j^{[y_i=0, x_{ij}=1]} \times (1 - b_j)^{[y_i=0, x_{ij}=0]}$$

Learning the naïve Bayes Classifier

Substituting and deriving the argmax, we get

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

$\leftarrow$ P(y = 1) = p

Learning the naïve Bayes Classifier

Substituting and deriving the argmax, we get

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

$$a_j = \frac{\text{Count}\,(y_i = 1, x_{ij} = 1)}{\text{Count}\,(y_i = 1)} \longleftarrow P(x_j = 1 \mid y = 1) = a_j$$

Learning the naïve Bayes Classifier

Substituting and deriving the argmax, we get

$$p = \frac{\text{Count}\left(y_i = 1\right)}{\text{Count}\left(y_i = 1\right) + \text{Count}\left(y_i = 0\right)}$$ ← $P(y = 1) = p$

$$a_j = \frac{\text{Count}\left(y_i = 1, x_{ij} = 1\right)}{\text{Count}\left(y_i = 1\right)}$$ ← $P(x_j = 1 \mid y = 1) = a_j$

$$b_j = \frac{\text{Count}\left(y_i = 0, x_{ij} = 1\right)}{\text{Count}\left(y_i = 0\right)}$$ ← $P(x_j = 1 \mid y = 0) = b_j$

Let's learn a naïve Bayes classifier

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

Let's learn a naïve Bayes classifier

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

$P(\text{Play} = +) = 9/14 \qquad P(\text{Play} = -) = 5/14$

Let's learn a naïve Bayes classifier

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

P(Play = +) = 9/14   P(Play = -) = 5/14

P(**O** = S | Play = +) = 2/9

Let's learn a naïve Bayes classifier

	O	**T**	**H**	**W**	**Play?**
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

P(Play = +) = 9/14   P(Play = -) = 5/14

P(**O** = S | Play = +) = 2/9

P(**O** = R | Play = +) = 3/9

Let's learn a naïve Bayes classifier

	O	T	H	W	Play?
1	S	H	H	W	-
2	S	H	H	S	-
3	O	H	H	W	+
4	R	M	H	W	+
5	R	C	N	W	+
6	R	C	N	S	-
7	O	C	N	S	+
8	S	M	H	W	-
9	S	C	N	W	+
10	R	M	N	W	+
11	S	M	N	S	+
12	O	M	H	S	+
13	O	H	N	W	+
14	R	M	H	S	-

P(Play = +) = 9/14   P(Play = -) = 5/14

P(**O** = S | Play = +) = 2/9

P(**O** = R | Play = +) = 3/9

P(**O** = O | Play = +) = 4/9

And so on, for other attributes and also for Play = -

Naïve Bayes: Learning and Prediction

- Learning
  - Count how often features occur with each label. Normalize to get likelihoods
  - Priors from fraction of examples with each label
  - Generalizes to multiclass

- Prediction
  - Use learned probabilities to find highest scoring label

Important caveats with Naïve Bayes

1. In practice, features may not be conditionally independent given the label
   - Just because we assume that they are doesn't mean that that's how they behave in nature
   - We made a modeling assumption because it makes computation and learning easier

Important caveats with Naïve Bayes

2. Not enough training data to get good estimates of the probabilities from counts

The basic operation for learning likelihoods is counting how often a feature occurs with a label.

What if we never see a particular feature with a particular label?

Should we treat those counts as zero?

$$P(y)\prod_j P(x_j|y)$$

Important caveats with Naïve Bayes

2. Not enough training data to get good estimates of the probabilities from counts

The basic operation for learning likelihoods is counting how often a feature occurs with a label.

What if we never see a particular feature with a particular label?

That will make the probabilities zero

Should we treat those counts as zero?

Answer: Smoothing

- Add fake counts (very small numbers so that the counts are not zero)

Example: Classifying text

- Instance space: Text documents
- Labels: Spam or NotSpam

- Goal: To learn a function that can predict whether a new document is Spam or NotSpam

How would you build a Naïve Bayes classifier?

Let us brainstorm

How to represent documents?
How to estimate probabilities?
How to classify?

Example: Classifying text

1. Represent documents by a vector of words

   A sparse vector consisting of one feature per word

2. Learning from N labeled documents

   1. Priors $$P(\text{Spam}) = \frac{\text{Count}\,(\text{Spam})}{N}; P(\text{NotSpam}) = 1 - P(\text{Spam})$$

   2. For each word w in vocabulary :

$$P(\text{w}|\text{Spam}) = \frac{\text{Count}\,(\text{w}, \text{Spam}) + 1}{\text{Count}\,(\text{Spam}) + |\text{Vocabulary}|}$$

$$P(\text{w}|\text{NotSpam}) = \frac{\text{Count}\,(\text{w}, \text{NotSpam}) + 1}{\text{Count}\,(\text{NotSpam}) + |\text{Vocabulary}|}$$

Example: Classifying text

1. Represent documents by a vector of words

    A sparse vector consisting of one feature per word

2. Learning from N labeled documents

    1. Priors $P(\text{Spam}) = \frac{\text{Count}\,(\text{Spam})}{N}; P(\text{NotSpam}) = 1 - P(\text{Spam})$

    2. For each word w in vocabulary :

$$P(\text{w}|\text{Spam}) = \frac{\text{Count}\,(\text{w}, \text{Spam}) + 1}{\text{Count}\,(\text{Spam}) + |\text{Vocabulary}|}$$

$$P(\text{w}|\text{NotSpam}) = \frac{\text{Count}\,(\text{w}, \text{NotSpam}) + 1}{\text{Count}\,(\text{NotSpam}) + |\text{Vocabulary}|}$$

How often does a word occur with a label?

Example: Classifying text

1. Represent documents by a vector of words

   A sparse vector consisting of one feature per word

2. Learning from N labeled documents

   1. Priors $P(\text{Spam}) = \frac{\text{Count}\,(\text{Spam})}{N}; P(\text{NotSpam}) = 1 - P(\text{Spam})$

   2. For each word w in vocabulary :

Smoothing

$$P(\text{w}|\text{Spam}) = \frac{\text{Count}\,(\text{w}, \text{Spam}) + 1}{\text{Count}\,(\text{Spam}) + |\text{Vocabulary}|}$$

$$P(\text{w}|\text{NotSpam}) = \frac{\text{Count}\,(\text{w}, \text{NotSpam}) + 1}{\text{Count}\,(\text{NotSpam}) + |\text{Vocabulary}|}$$

Clustering

Hogwarts (Harry Potter)

- Sorting Hat – cluster kids into four groups based on four underlying prototypes

Godric Gryffindor

Helga Hufflepuff

Rowena Ravenclaw

Salazar Slytherin

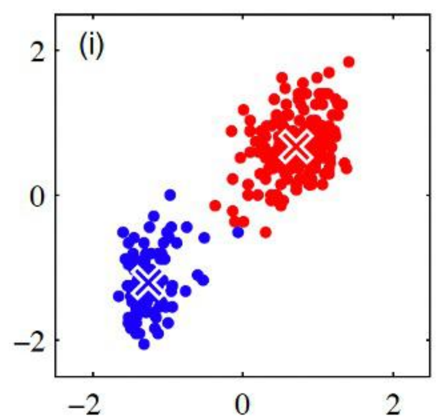
Goal of Clustering

- Given a collection of data points, the goal is to find structure in the data: organize that data into sensible groups.

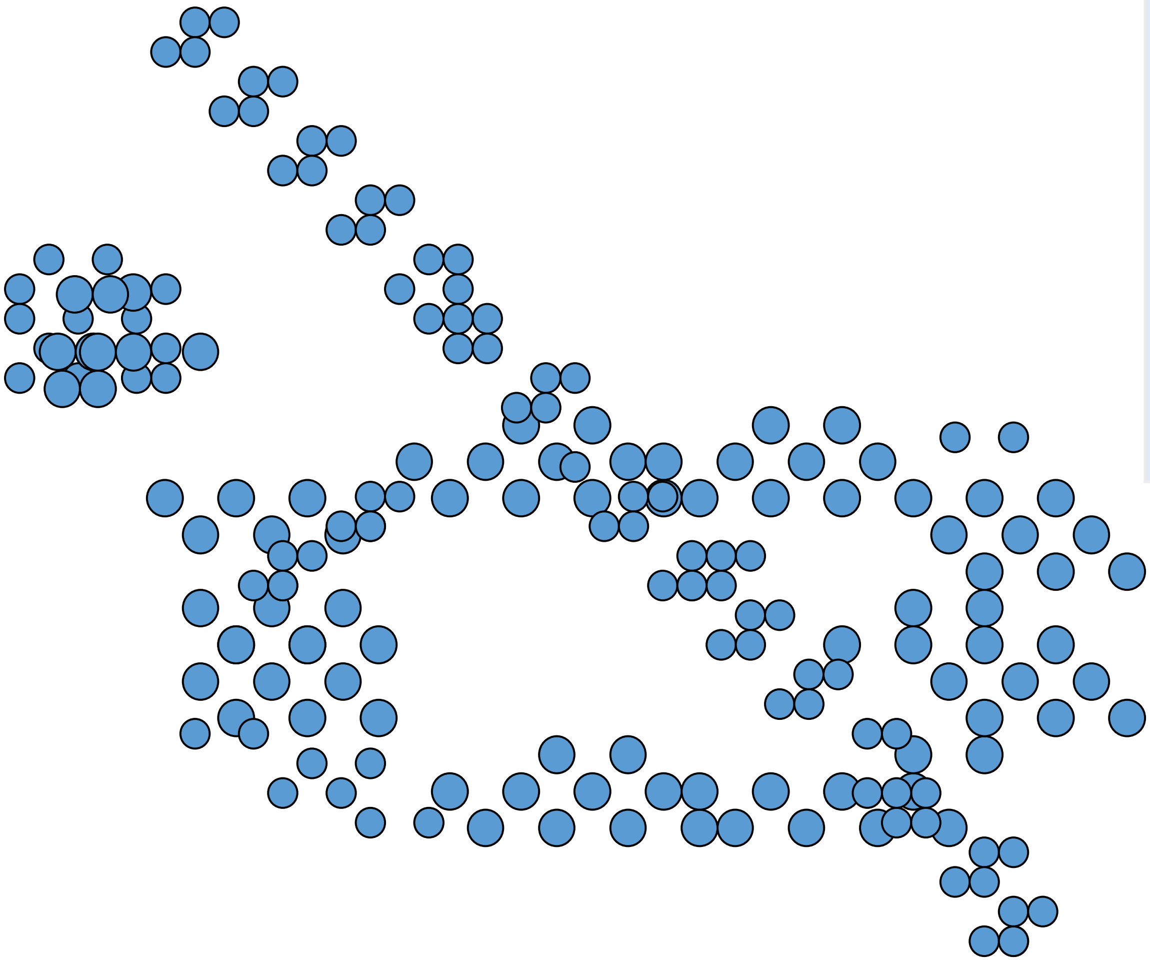
- Applications
  - Topics in news articles
  - Identify communities within social networks

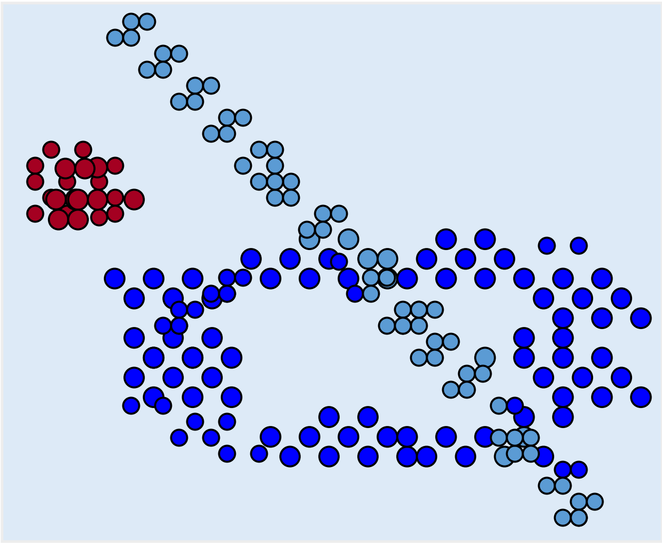
How to define clusters

- A set of entities which are “alike”
- May be described as connected regions of a multi-dimensional space
- ”We recognize a cluster when we see it”



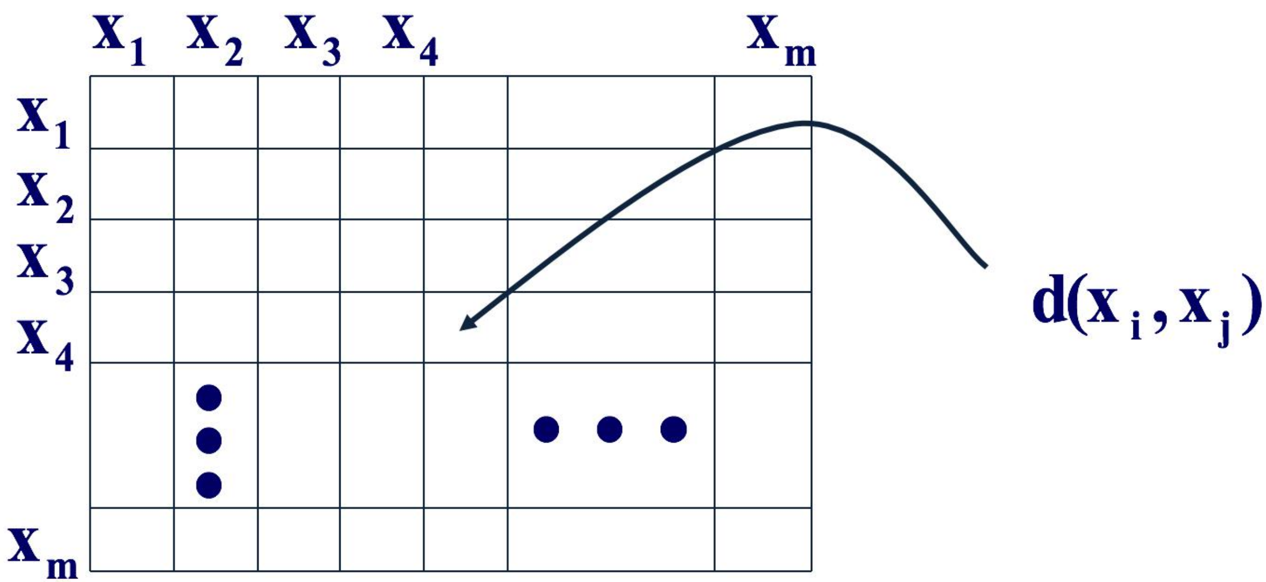
How many clusters are there?





Pairwise distance

- The pairwise distances are given
  - We assume assume that the input to the problem is a matrix of distances between all pairs



Clustering

- An optimization problem:
  - Given a set of points and a pairwise distance, devise an algorithm f that splits the data so that it optimizes some conditions.

Setup Given $\mathcal{D} = \{\boldsymbol{x}_n\}_{n=1}^N$ and K, we want to output

- $\{\boldsymbol{\mu}_k\}_{k=1}^K$: prototypes (or centroids) of clusters
- $A(\boldsymbol{x}_n) \in \{1, 2, \ldots, K\}$: the cluster membership, i.e., the cluster ID assigned to $\boldsymbol{x}_n$

K-Means

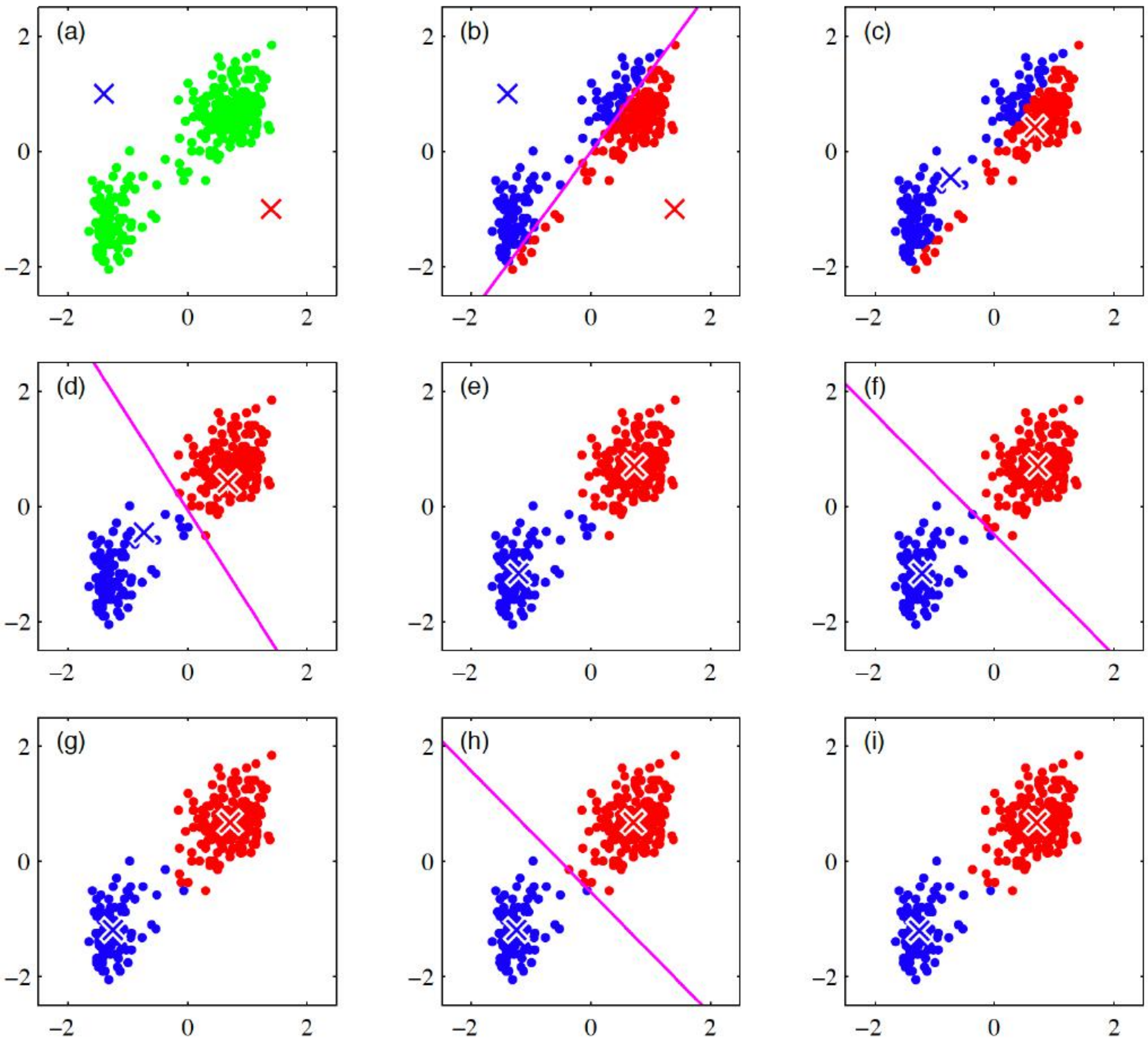
K-Means Intuition

- Sorting Hat – cluster kids into four groups based on four underlying prototypes

- The prototype of each house is *the average of all kids of the house*
- Algorithm*:*
  Alternatively, updating the prototype & the cluster assignment

http://shabal.in/visuals/kmeans/6.html

Intuition of K-Means



K-means clustering

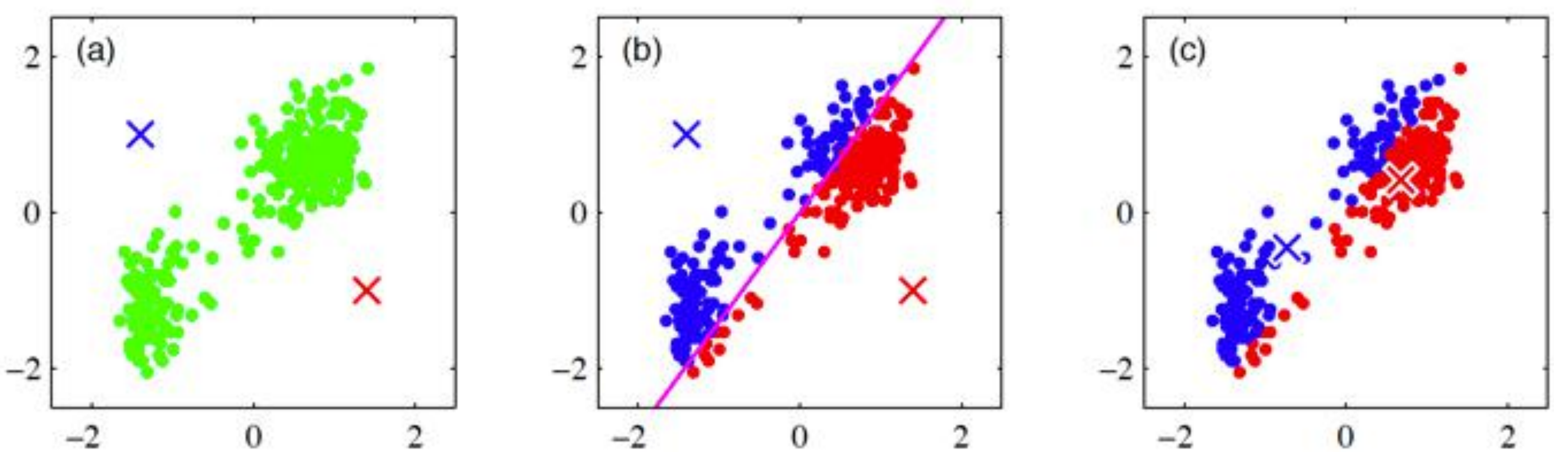
Sum of distances of all the points to their cluster center

- Distortion measure (loss function for clustering)

$$J(\{r_{nk}\}, \{\boldsymbol{\mu}_k\}) = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

where $r_{nk} \in \{0, 1\}$ is an indicator variable

$$r_{nk} = 1 \quad \text{if and only if} \quad A(\boldsymbol{x}_n) = k$$



K-means objective

$$argmin_{\{r_n|_k\},\{\boldsymbol{\mu}_k\}} J(\{r_{nk}\},\{\boldsymbol{\mu}_k\}) = \sum_{n=1}^{N}\sum_{k=1}^{K} r_{nk}\|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2$$

where $r_{nk} \in \{0,1\}$ is an indicator variable

$$r_{nk} = 1 \quad \text{if and only if} \quad A(\boldsymbol{x}_n) = k$$

- It is a non-convex objective function
- Minimizing the above objective is NP-hard.

K-means algorithm
a.k.a Llyod's algorithm

- A greedy algorithm for minimizing K-means objective – alternative update $\{r_{nk}\}, \{\boldsymbol{\mu}_k\}$
- Step 0: randomly assign the cluster centers $\{\mu_k\}$
- Step 1: Minimize J over $\{r_{nk}\}$ -- reassign cluster member
- Step 2: Minimize J over $\{\mu_k\}$ -- update the cluster centers
- Loop until it converges

$$\boxed{J(\{r_{nk}\}, \{\boldsymbol{\mu}_k\}) = \sum_{n=1}^{N} \sum_{k=1}^{K} r_{nk} \|\boldsymbol{x}_n - \boldsymbol{\mu}_k\|_2^2}$$

K-means algorithm a.k.a Llyod's algorithm

- Step 0: randomly assign the cluster centers $\{\mu_k\}$
- Step 1: Minimize J over $\{r_{nk}\}$ -- Assign every point to the closest cluster center

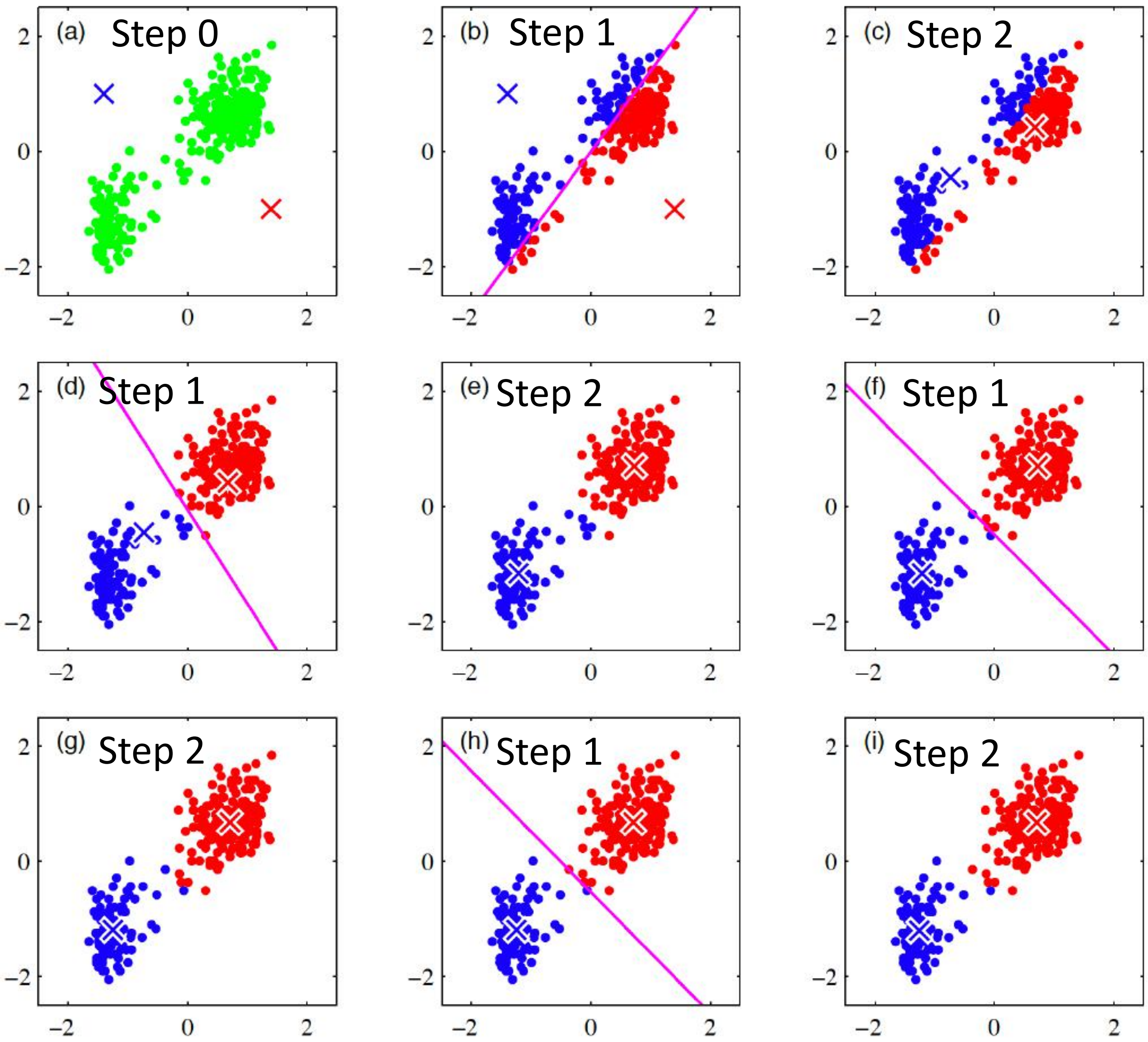
$$r_{nk} = \begin{cases} 1 & \text{if } k = \arg\min_j \|\boldsymbol{x}_n - \boldsymbol{\mu}_j\|_2^2 \\ 0 & \text{otherwise} \end{cases}$$

- Step 2: Minimize J over $\{\mu_k\}$ -- update the cluster centers

$$\boldsymbol{\mu}_k = \frac{\sum_n r_{nk}\boldsymbol{x}_n}{\sum_n r_{nk}}$$

- Loop until it converges

Example



(a) Step 0
(b) Step 1
(c) Step 2
(d) Step 1
(e) Step 2
(f) Step 1
(g) Step 2
(h) Step 1
(i) Step 2

Remarks

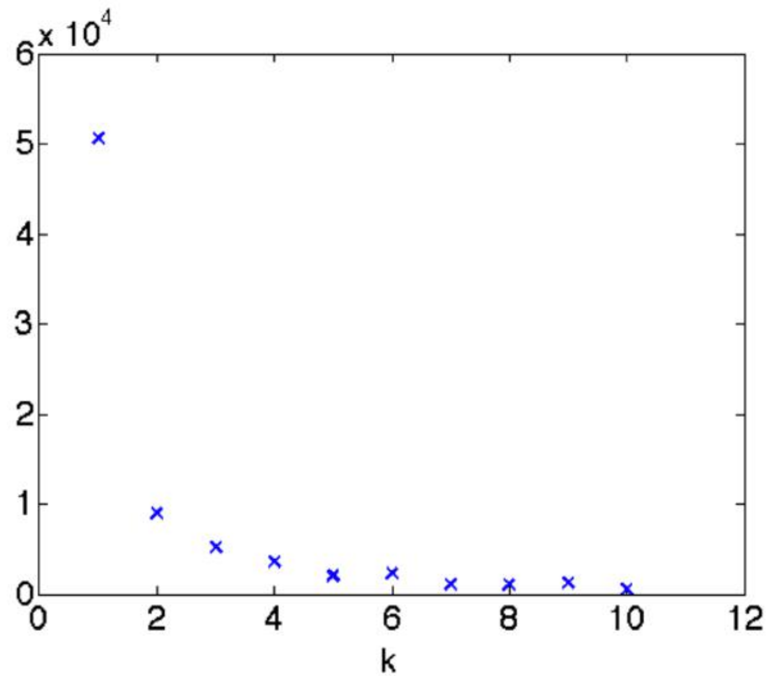
- Prototype μ_k is the mean of data points assigned to the cluster k, hence `K-means'
- μ_k may not in the training set
- Need to pre-define k
  - There are some other approaches for the case k is unknown – not cover in class
- The procedure reduces J in both Step 1 and Step 2 and thus makes improvements or stay the same on each iteration

Properties of the K-means algorithm

- Does the K-means algorithm converge
  - Yes
- How long does it take to converge?
  - In the worst case, exponential in the number of data points
  - In practice, usually quick
- How good is its solution?
  - Local minimum (depends on the initialization)

Choosing K

- Increasing K will always decrease the optimal value of the K-means objective
  - It doesn't mean a better clustering
  - Analogous to overfitting in supervised learning.



K-means can be sensitive to the outlier

- One data point can make the center shift