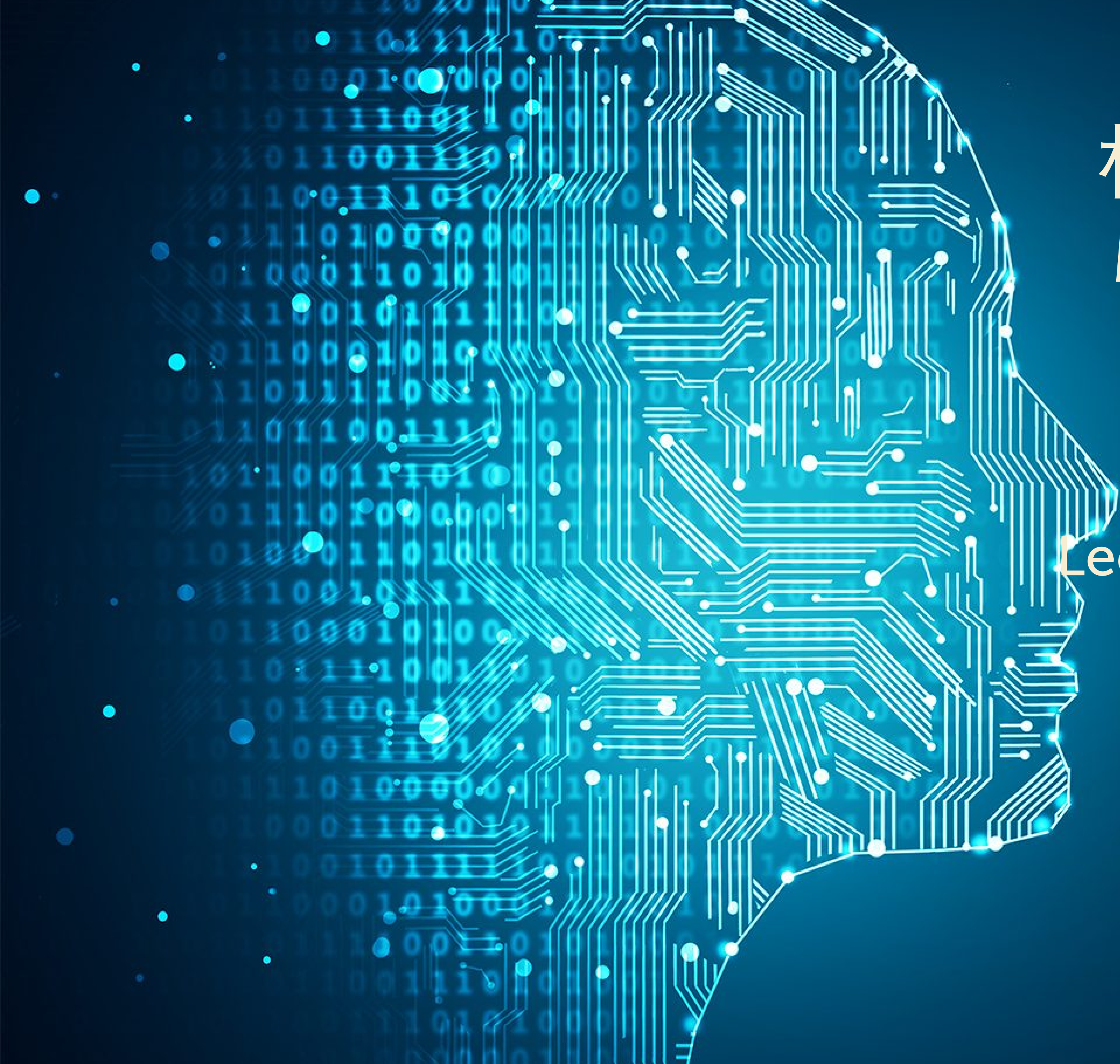


机器学习与人工智能
Machine Learning and Artificial Intelligence

Lecture 5 SVM and Naïve Bayes

Yingjie Zhang (张颖婕)
Peking University
yingjiezhang@gsm.pku.edu.cn
2021 Fall

Missing Value in DT

- Data D and attribute a
- $\tilde{D}$ is the data that do not have missing values on a
- Value of a: $\{a^1, a^2, \dots, a^V\}$
- $\tilde{D}^v$,$\tilde{D}_k$ where $k = 1,2, \dots, |Y|$
- $\rho = \frac{\sum_{x \in \tilde{D}} w_x}{\sum_{x \in D} w_x}$; $\quad \widetilde{p_k} = \frac{\sum_{x \in \tilde{D}_k} w_x}{\sum_{x \in \tilde{D}} w_x}$; $\quad \widetilde{r_v} = \frac{\sum_{x \in \tilde{D}^v} w_x}{\sum_{x \in \tilde{D}} w_x}$
- $Gain(D, a) = \rho \times Gain(\tilde{D}, a) = \rho \times \left(Ent(\tilde{D}) - \sum_{v=1}^{V} \widetilde{r_v} Ent(\tilde{D}^v)\right)$

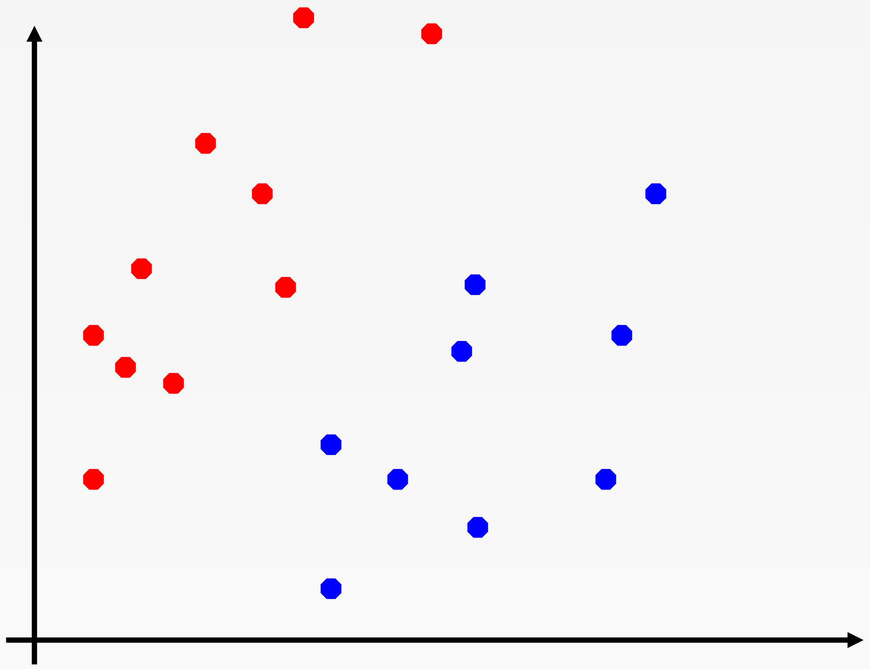
$$Ent(\tilde{D}) = -\sum_{k=1}^{|Y|} \widetilde{p_k} \log_2 \widetilde{p_k}$$

- *Split info* is calculated the same as before but with the missing values considered a separate state that an attribute can take.

Support Vector Machine

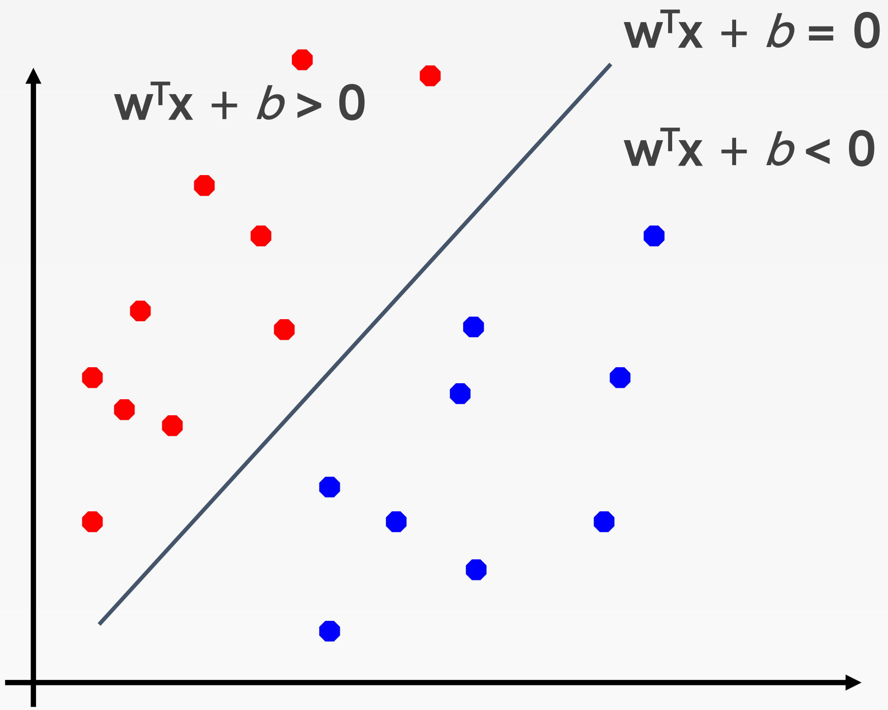
Linear SVM Classification

- Binary classification can be viewed as the task of separating classes in feature space:



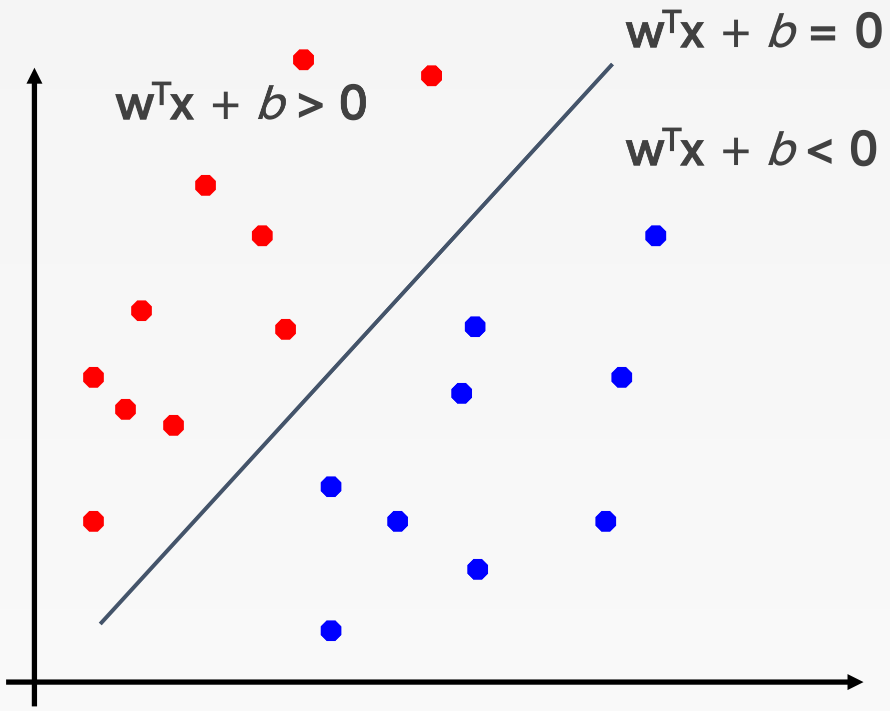
Linear SVM Classification

- Binary classification can be viewed as the task of separating classes in feature space:

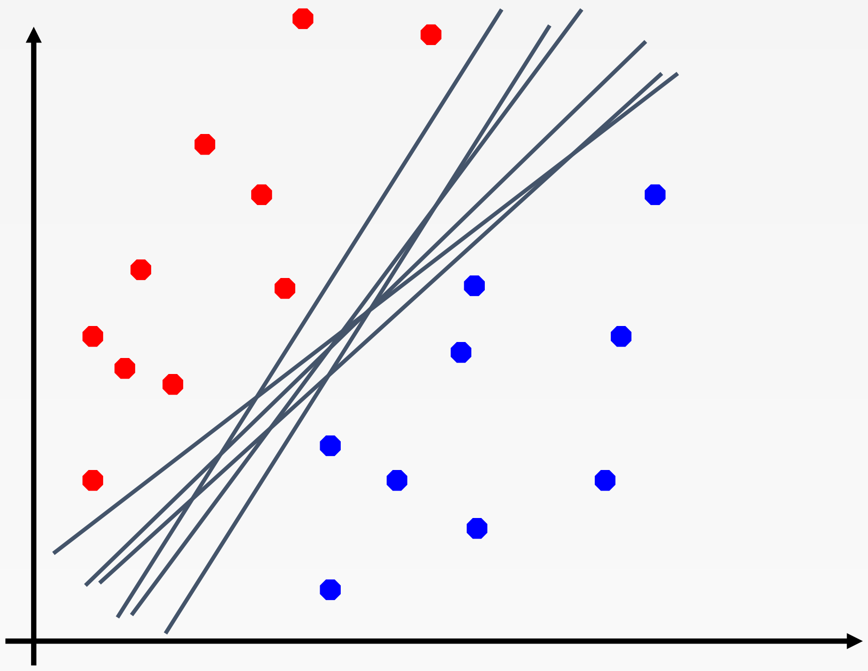


Linear SVM Classification

- Binary classification can be viewed as the task of separating classes in feature space:

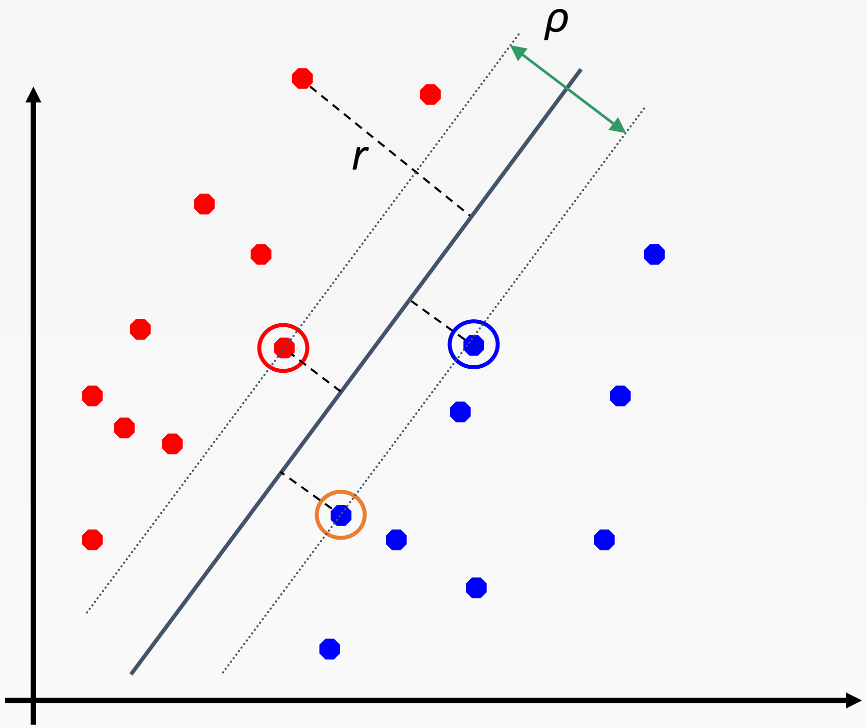


Which one is the optimal?



Classification Margin

- Distance from example X_i to the separator is $r = \frac{|\mathbf{w}^T X_i + b|}{\|\mathbf{w}\|}$
- Examples closest to the hyperplane are ***support vectors***.
- ***Margin*** ρ of the separator is the distance between support vectors.



Goal: maximize the margin

SVM Optimization

Hard-margin SVM (Primal)

$$\min_{\boldsymbol{w},b} \frac{1}{2} \|w\|_2^2$$

$$\text{s.t. } y^{(i)}\left(w^T \boldsymbol{x}^{(i)} + b\right) \geq 1,$$

$$\forall i = 1, \dots, N$$

SVM Optimization

Hard-margin SVM (Primal)	Hard-margin SVM (Lagrangian Dual)
$\min_{\boldsymbol{w},b} \frac{1}{2} \Vert w \Vert_2^2$ s.t. $y^{(i)}\left(w^T \boldsymbol{x}^{(i)} + b\right) \geq 1,$ $\forall i = 1, \dots, N$	$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y^{(i)} y^{(j)} \boldsymbol{x}^{(i)} \cdot \boldsymbol{x}^{(j)}$ s.t. $\alpha_i \geq 0, \forall i = 1, \dots, N$ $\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$

Definition: **support vectors** are those points $x^{(i)}$ for which $\alpha_i \neq 0$

Method of Lagrange Multipliers

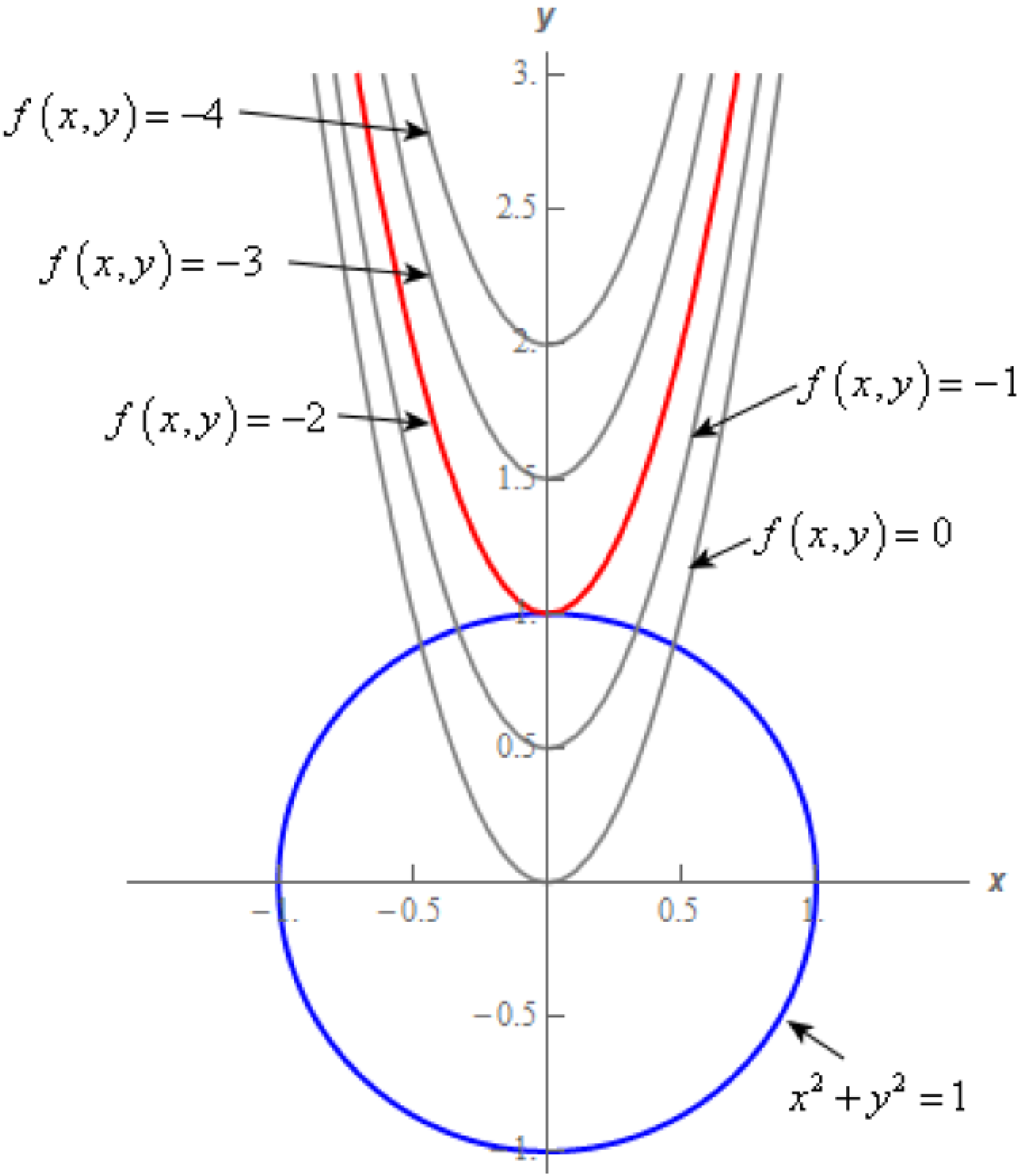
- Goal: $\min f(\boldsymbol{x})$ s.t., $g(\boldsymbol{x}) \leq c$

- Step 1: construct Lagrangian

$$L(\boldsymbol{x}, \lambda) = f(\boldsymbol{x}) + \lambda(g(\boldsymbol{x}) - c)$$

- Step 2: Solve $\min_{x} \max_{\lambda} L(\boldsymbol{x}, \lambda)$

$$\nabla f(\boldsymbol{x}) = \lambda \nabla g(\boldsymbol{x}), \text{ s.t. } \lambda \geq 0, g(\boldsymbol{x}) \leq c$$



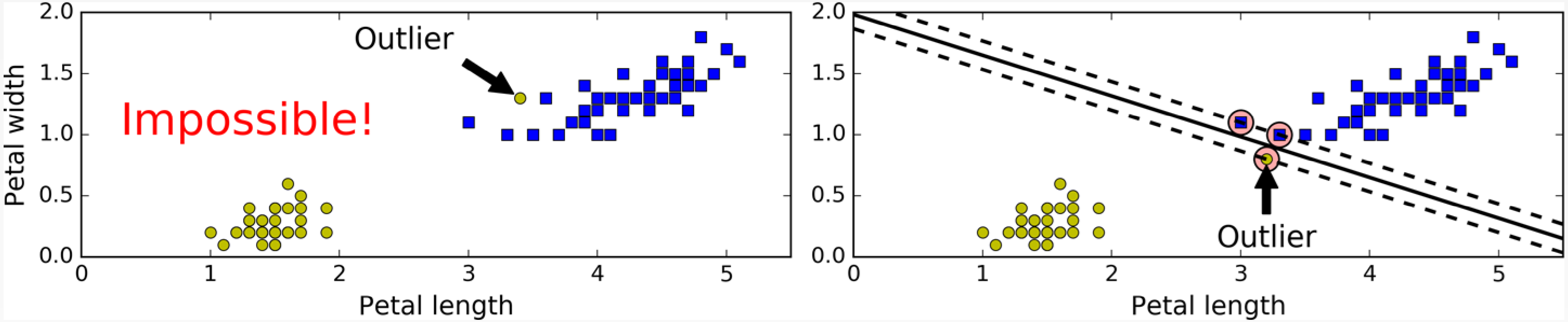
y
x
3.
2.5
2.
1.5
1.
0.5
−0.5
−1.
−1.
−0.5
0.5
1.
$f(x,y) = -4$
$f(x,y) = -3$
$f(x,y) = -2$
$f(x,y) = -1$
$f(x,y) = 0$
$x^2 + y^2 = 1$

Hard Margin Classification

- Hard margin classification: all instances be off the decision boundary

Hard Margin Classification

- Hard margin classification: all instances be off the decision boundary
- Potential issues:
  - Only works if the data is linearly separable
  - Sensitive to outliers



Soft Margin Classification

- **Key idea**: balance between keeping the decision boundary as large as possible and limiting the margin violations

SVM Optimization

Hard-margin SVM (Primal)

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|w\|_2^2$$

$$\text{s.t. } y^{(i)}\left(w^T\boldsymbol{x}^{(i)} + b\right) \geq 1,$$
$$\forall i = 1, \dots, N$$

Hard-margin SVM (Lagrangian Dual)

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y^{(i)} y^{(j)} \boldsymbol{x}^{(i)} \cdot \boldsymbol{x}^{(j)}$$

$$\text{s.t. } \alpha_i \geq 0, \forall i = 1, \dots, N$$

$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

SVM Optimization

Hard-margin SVM (Primal)

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|w\|_2^2$$

$$\text{s.t. } y^{(i)}\left(w^T\boldsymbol{x}^{(i)} + b\right) \geq 1,$$
$$\forall i = 1, \dots, N$$

Hard-margin SVM (Lagrangian Dual)

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y^{(i)} y^{(j)} \boldsymbol{x}^{(i)} \cdot \boldsymbol{x}^{(j)}$$

$$\text{s.t. } \alpha_i \geq 0, \forall i = 1, \dots, N$$
$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

Soft-margin SVM (Primal)

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|w\|_2^2 + C\left(\sum_{i=1}^{N} e_i\right)$$

$$\text{s.t. } y^{(i)}\left(w^T\boldsymbol{x}^{(i)} + b\right) \geq 1 - e_i,$$
$$e_i \geq 0$$
$$\forall i = 1, \dots, N$$

SVM Optimization

Hard-margin SVM (Primal)

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|w\|_2^2$$

s.t. $y^{(i)}\left(w^T\boldsymbol{x}^{(i)} + b\right) \geq 1,$

$\forall i = 1, \dots, N$

Hard-margin SVM (Lagrangian Dual)

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y^{(i)}y^{(j)}\boldsymbol{x}^{(i)} \cdot \boldsymbol{x}^{(j)}$$

s.t. $\alpha_i \geq 0, \forall i = 1, \dots, N$

$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$

Soft-margin SVM (Primal)

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|w\|_2^2 + C\left(\sum_{i=1}^{N} e_i\right)$$

s.t. $y^{(i)}\left(w^T\boldsymbol{x}^{(i)} + b\right) \geq 1 - e_i,$

$e_i \geq 0$

$\forall i = 1, \dots, N$

Hard-margin SVM (Lagrangian Dual)

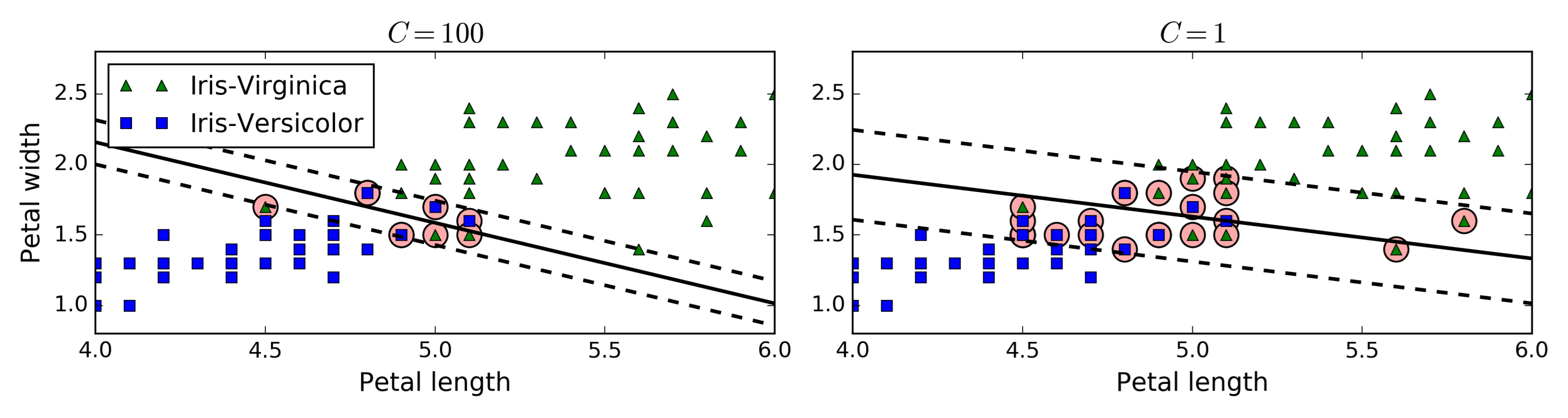
$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i\alpha_j y^{(i)}y^{(j)}\boldsymbol{x}^{(i)} \cdot \boldsymbol{x}^{(j)}$$

s.t. $0 \leq \alpha_i \leq C, \forall i = 1, \dots, N$

$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$

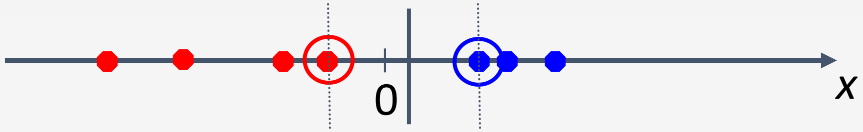
Soft Margin Classification

- **Key idea**: balance between keeping the decision boundary as large as possible and limiting the margin violations
- **C: Regularization parameter**
  - Small C → large margin
  - Large C → narrow margin
  - $C = \infty$ → hard margin



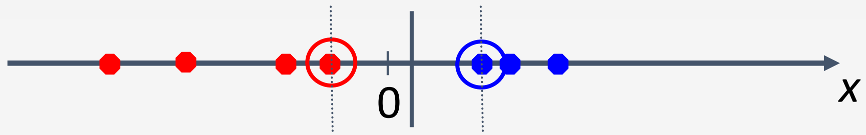
Non-linear SVMs

Datasets that are linearly separable with some noise work out great

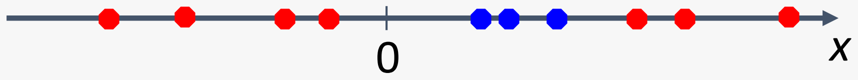


Non-linear SVMs

Datasets that are linearly separable with some noise work out great

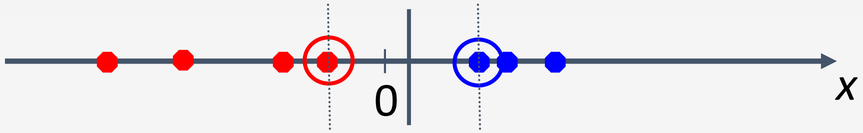


But what if the dataset is not that perfect?

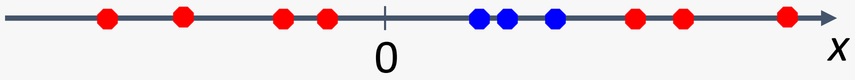


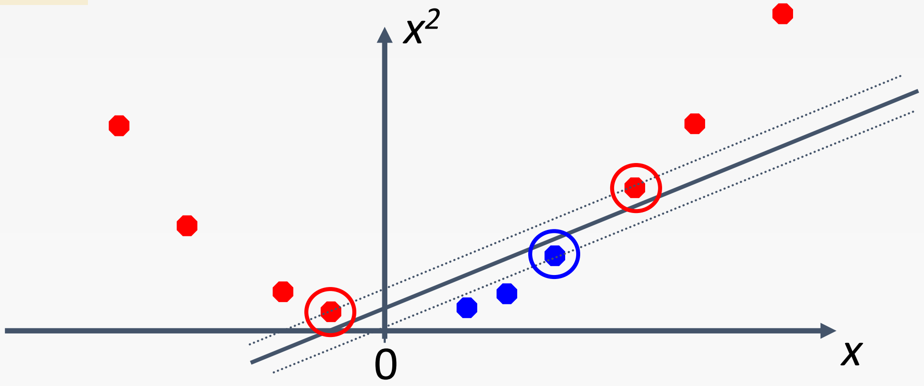
Non-linear SVMs

Datasets that are linearly separable with some noise work out great



But what if the dataset is not that perfect?





General idea: the original feature space can always be mapped to some higher-dimensional feature space where the training set is separable

Kernel Method

- Motivation #1: Inefficient Features
  - Non-linearly separable data requires high dimensional representation
  - Might be prohibitively expensive to compute or store
- Motivation #2: Memory-based Methods
  - KNN

- Key idea:
  - Rewrite the algorithm so that we only work with dot product $x^T z$ of feature vectors
  - Replace the dot products $x^T z$ with a kernel function $k(x, z)$

SVM Optimization

Hard-margin SVM (Primal)

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|w\|_2^2$$

$$\text{s.t. } y^{(i)}\left(w^T \boldsymbol{x}^{(i)} + b\right) \geq 1, \quad \forall i = 1, \dots, N$$

$$\min_{\boldsymbol{w},b} \frac{1}{2}\|w\|_2^2$$

$$\text{s.t. } y^{(i)}\left(w^T \phi\left(\boldsymbol{x}^{(i)}\right) + b\right) \geq 1, \quad \forall i = 1, \dots, N$$

Hard-margin SVM (Lagrangian Dual)

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} \boldsymbol{x}^{(i)} \cdot \boldsymbol{x}^{(j)}$$

$$\text{s.t. } \alpha_i \geq 0, \forall i = 1, \dots, N$$

$$\sum_{i=1}^N \alpha_i y^{(i)} = 0$$

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} \phi\left(\boldsymbol{x}^{(i)}\right) \cdot \phi\left(\boldsymbol{x}^{(j)}\right)$$

$$\text{s.t. } \alpha_i \geq 0, \forall i = 1, \dots, N$$

$$\sum_{i=1}^N \alpha_i y^{(i)} = 0$$

SVM Kernel Trick

Hard-margin SVM (Lagrangian Dual)

$$\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y^{(i)} y^{(j)} k\left(\boldsymbol{x}^{(i)}, \boldsymbol{x}^{(j)}\right)$$

s.t. $\alpha_i \geq 0, \forall i = 1, \dots, N$

$$\sum_{i=1}^{N} \alpha_i y^{(i)} = 0$$

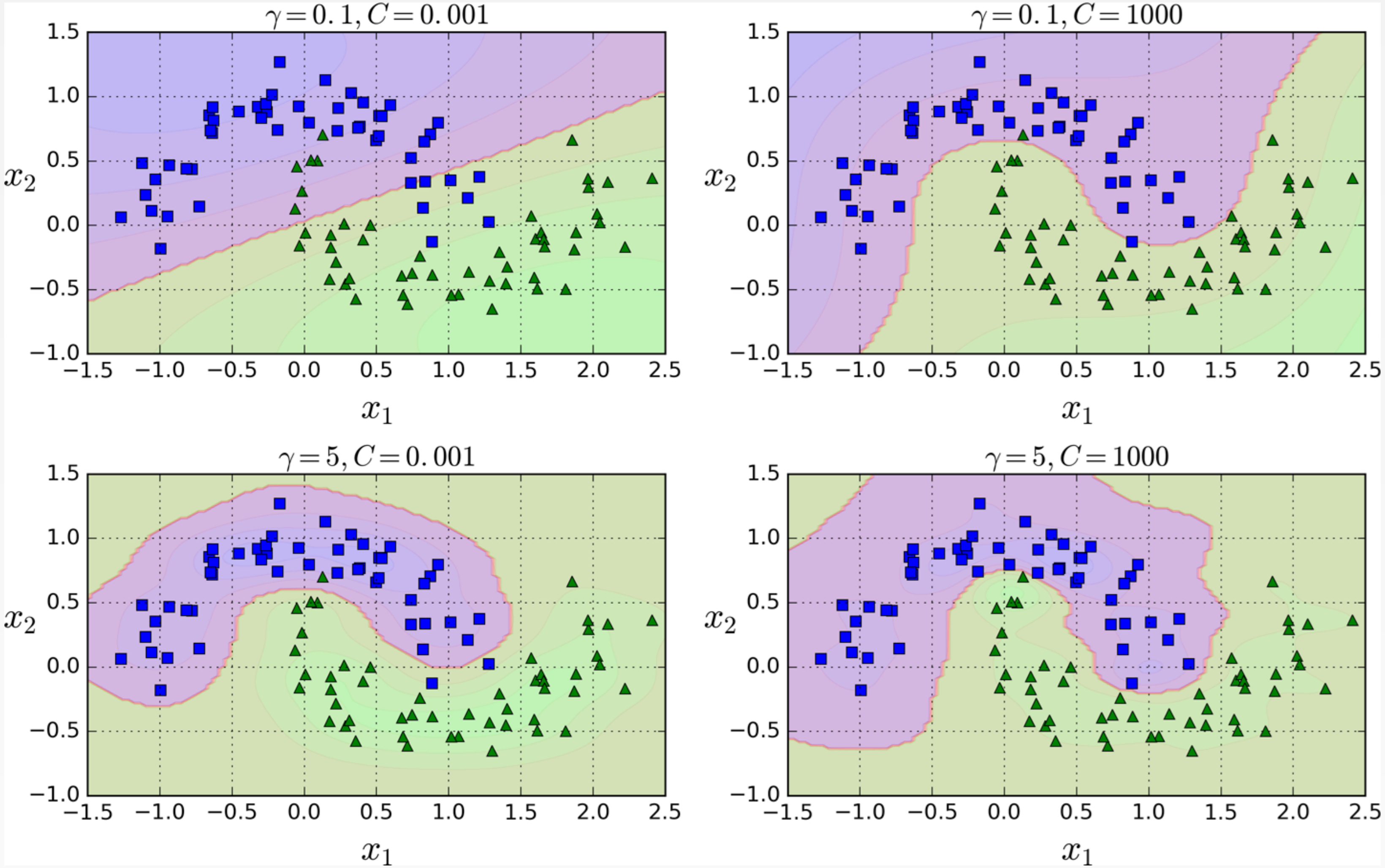
The "Kernel Trick"

- If every data point is mapped into high-dimensional space via some transformation: $\Phi: x \to \psi(x)$, the inner product becomes:
$$K(\boldsymbol{x}, \boldsymbol{z}) = \psi(\boldsymbol{x})^T \psi(\boldsymbol{z})$$
- A kernel function implicitly maps data to a high-dimensional space (without the need to compute each $\psi(x)$ explicitly)
- Can be applied to many algorithms:
  - Classification: SVM, …
  - Regression: ridge regression, …
  - Clustering: K-means,…

Kernel Example

Name	Kernel Function (Implicit dot product)	Feature Space (Explicit dot product)
Linear	$K(\boldsymbol{x},\boldsymbol{z}) = \boldsymbol{x}^T\boldsymbol{z}$	Same as original input
Polynomial	$K(\boldsymbol{x},\boldsymbol{z}) = (\boldsymbol{x}^T\boldsymbol{z})^d$	All polynomials of degree d
Gaussian	$K(\boldsymbol{x},\boldsymbol{z}) = \exp\left(-\frac{\lVert x-z\rVert_2^2}{2\sigma^2}\right)$	Infinite dimensional space
Sigmoid Kernel	$K(\boldsymbol{x},\boldsymbol{z}) = tanh(\alpha\boldsymbol{x}^T\boldsymbol{z} + c)$	With SVM, this is equivalent to a 2-layer neural network

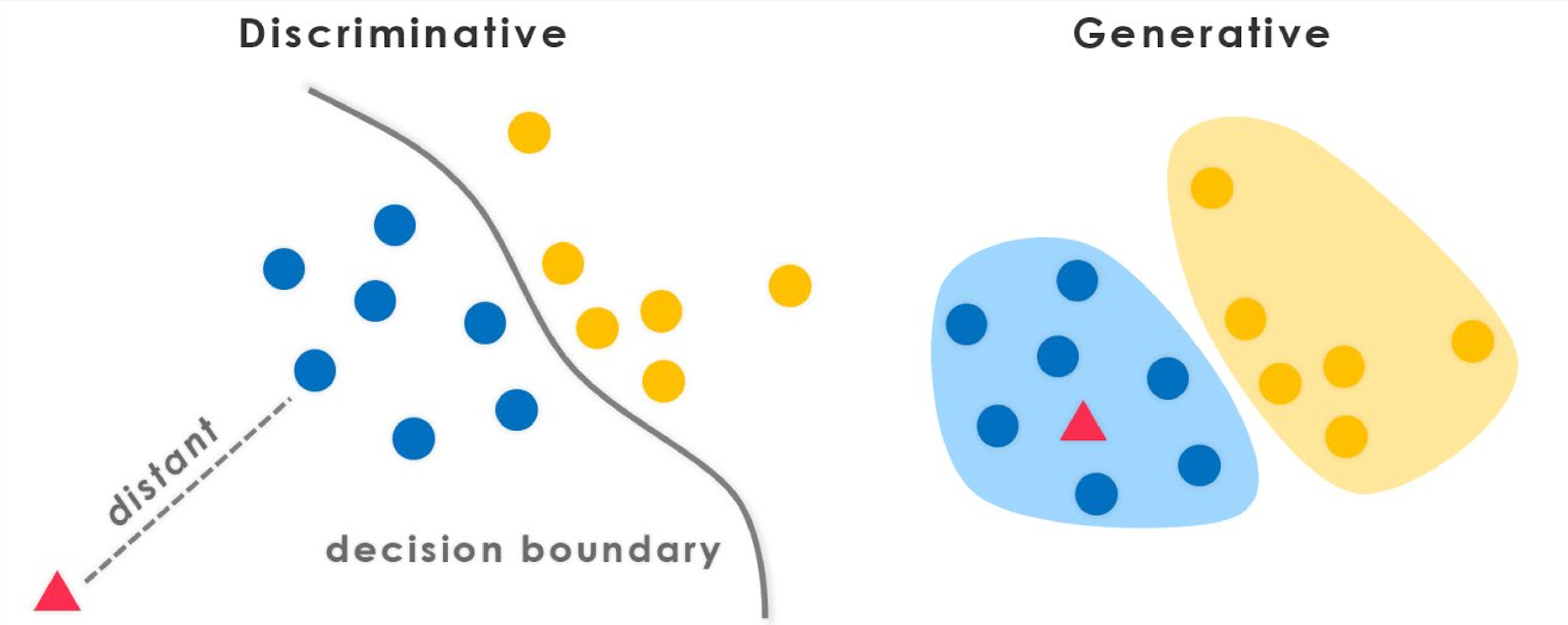
RBF Kernel



RBF Kernel: $K(x, z) = \exp(-\gamma \|x - z\|_2^2)$

Naïve Bayes

Generative vs. Discriminative



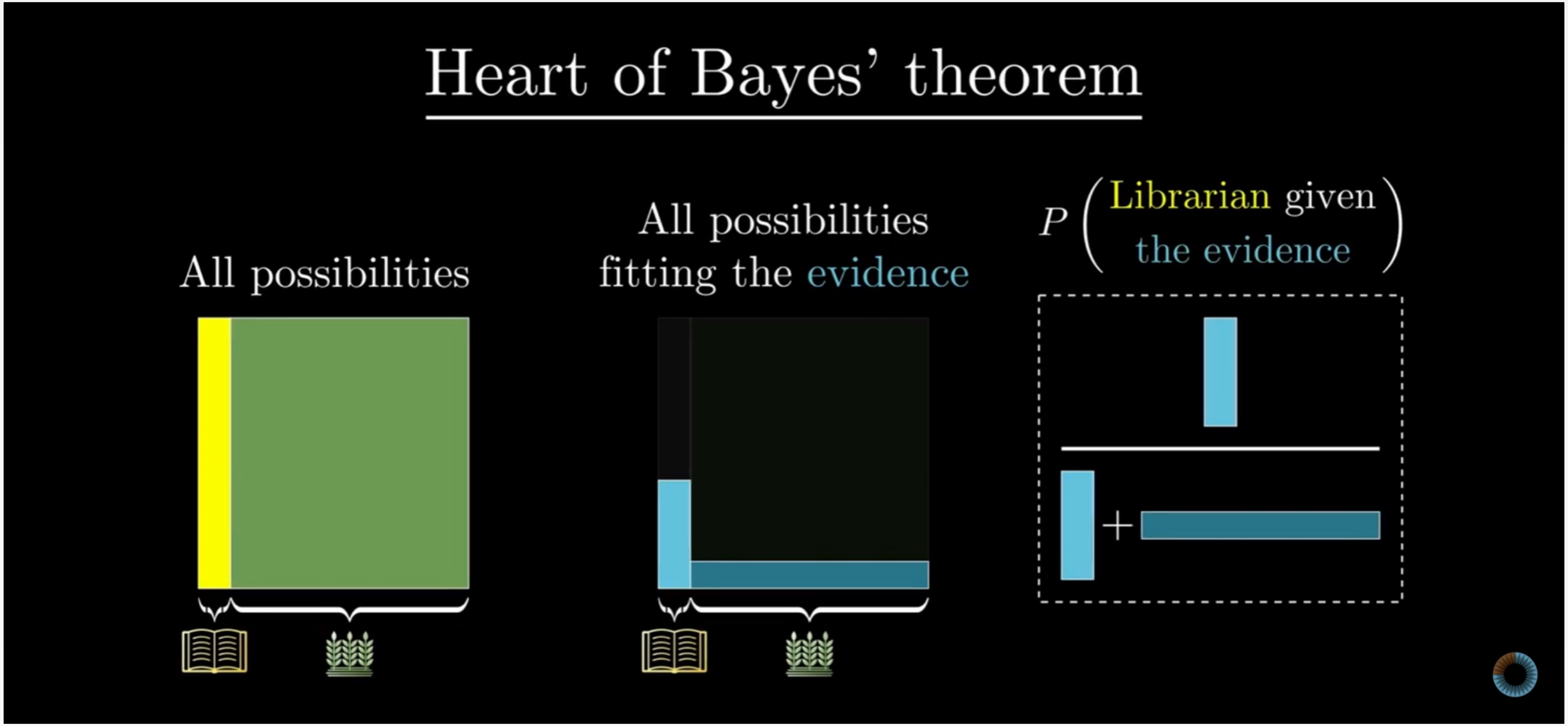
Probability Review

- $P(A) + P(\neg A) = 1$
- $0 \leq P(A) \leq 1$
- $P(A \vee B) = P(A) + P(B) - P(A \wedge B)$
- $P(A) = P(A \vee B) + (A \wedge \neg B)$ $\qquad P(A) = \sum_{i=1}^{k} P(A \wedge \ B = v_i)$
- $P(A|B) = \frac{P(A \wedge B)}{P(B)}$

  $\rightarrow P(A \wedge B) = P(A|B) \times P(B)$
- Independence: $P(A \wedge B) = P(A) \times P(B)$

  $P(A|B) = P(A)$
- Bayes' Rule: $P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}$

Bayes Theorem



Naïve Bayes Assumption

Naïve Bayes classifiers assume that the effect of a variable value on a given class membership is independent of the values of other variables

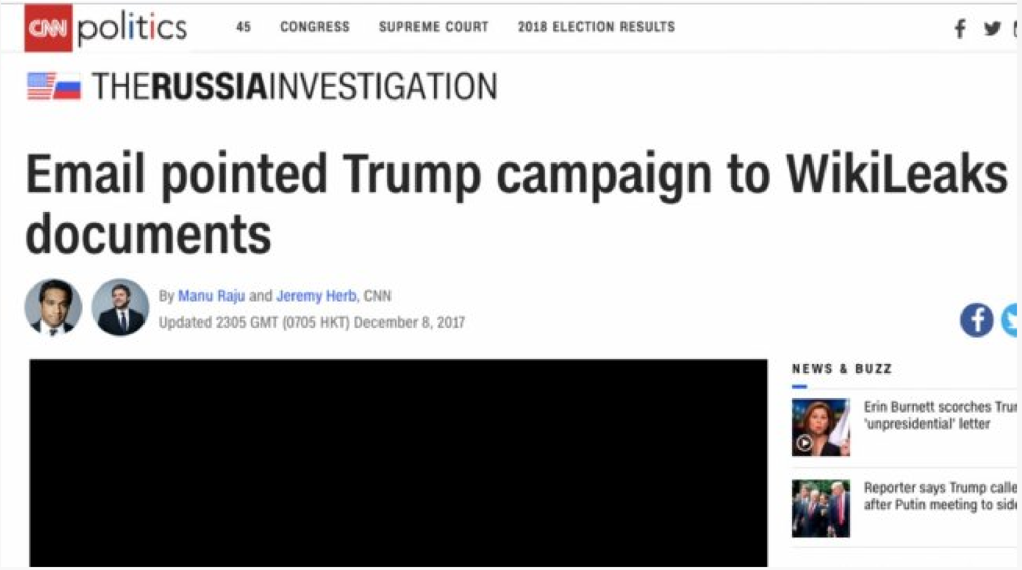
$$P(X_1, X_2|Y) = P(X_1|X_2, Y)P(X_2|Y) = P(X_1|Y)P(X_2|Y)$$

More generally, $P(X_1, \dots, X_n|Y) = \prod_i P(X_i|Y)$

Use Bayes' Rule: $P\left(Y_j \middle| X_1, \dots, X_N\right) = \frac{P(Y_j) \cdot \prod_i P(X_i|Y_j)}{\sum_k P(Y_k) \cdot \prod_i P(X_i|Y_k)}$

Fake News Detector

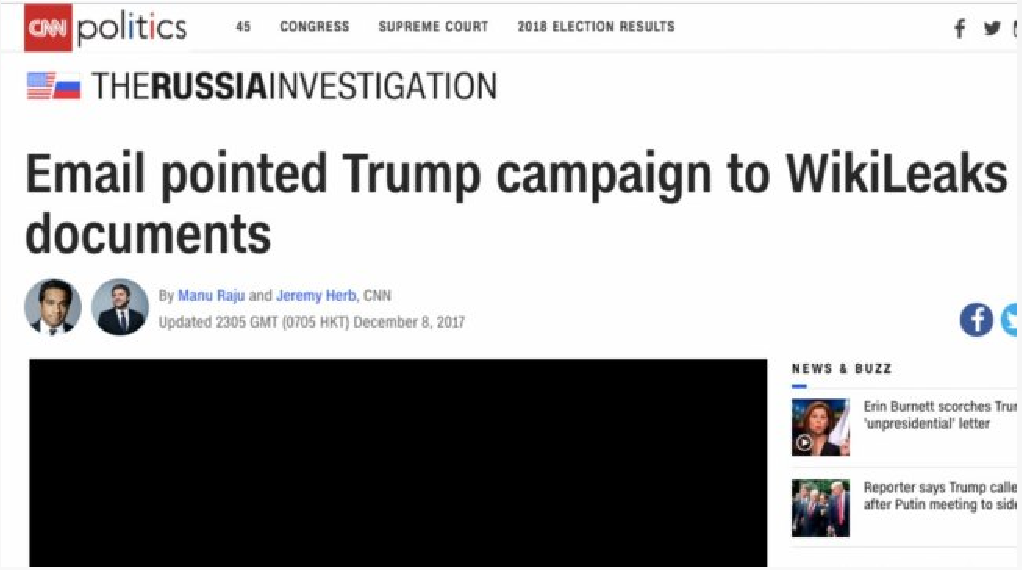
CNN News



Fake News

Fake News Detector

CNN News



Fake News

Bag of words

the dog is on the table

are	cat	dog	is	now	on	table	the
0	0	1	1	0	1	1	1

Model 1: Bernoulli Naïve Bayes

Flip a weighted coin

Model 1: Bernoulli Naïve Bayes

Flip a weighted coin

If HEADS, flip each red coin

$y \quad x_1 \quad x_2 \quad x_3 \quad \dots \quad x_M$

If TAILS, flip each blue coin

Model 1: Bernoulli Naïve Bayes

Flip a weighted coin

If HEADS, flip each red coin

If TAILS, flip each blue coin

y		x_1	x_2	x_3	…	x_M
0		1	0	1	…	1

Model 1: Bernoulli Naïve Bayes

Flip a weighted coin

If HEADS, flip each red coin

If TAILS, flip each blue coin

y	x_1	x_2	x_3	…	x_M
0	1	0	1	…	1
1	0	1	0	…	1

Model 1: Bernoulli Naïve Bayes

Flip a weighted coin

If HEADS, flip each red coin

If TAILS, flip each blue coin

y	x_1	x_2	x_3	…	x_M
0	1	0	1	…	1
1	0	1	0	…	1
1	1	1	1	…	1
0	0	0	1	…	1
0	1	0	1	…	0
1	1	0	1	…	0

What's wrong with the Naïve Bayes Assumption?

The features might not be independent!!

- Example 1:
  - If a document contains the word "Donald", it's extremely likely to contain the word "Trump" – These are not independent!
- Example 2:
  - If the petal width is very high, the petal length is also likely to be very high

Model 1: Bernoulli Naïve Bayes

- Data: $x \in \{0,1\}^M$, $y \in \{0,1\}$

Generative Process:

$$y \sim Bernoulli(\phi)$$

$$x_1 \sim Bernoulli(\theta_{y,1})$$

$$x_2 \sim Bernoulli(\theta_{y,2})$$

…

$$x_M \sim Bernoulli(\theta_{y,M})$$

Model:

$$p_{\phi,\boldsymbol{\theta}}(\boldsymbol{x}, y) = p_{\phi,\boldsymbol{\theta}}(x_1, x_2, \ldots, x_M, y)$$

$$= p_{\phi}(y) \prod_{\boldsymbol{m-1}}^{\boldsymbol{M}} p_{\boldsymbol{\theta}}(x_m|y)$$

$$= \left[(\phi)^y (1-\phi)^{(1-y)} \prod_{m=1}^{M} (\theta_{y,m})^{x_m} (1-\theta_{y,m})^{(1-x_m)} \right]$$

MLE

Training: Find the class-conditional MLE parameters

Count Variables

$$N_{y=1} = \sum_{i=1}^{N} \mathbb{I}\left(y^{(i)} = 1\right)$$

$$N_{y=0} = \sum_{i=1}^{N} \mathbb{I}\left(y^{(i)} = 0\right)$$

$$N_{y=0,x_m=1} = \sum_{i=1}^{N} \mathbb{I}\left(y^{(i)} = 0 \wedge x_m^{(i)} = 1\right)$$

Maximum Likelihood Estimators

$$\phi = \frac{N_{y=1}}{N}$$

$$\phi_{0,m} = \frac{N_{y=0,x_m=1}}{N_{y=0}}$$

$$\phi_{1,m} = \frac{N_{y=1,x_m=1}}{N_{y=1}}$$

$$\forall m \in \{1, \dots, M\}$$

An Illustrative Example

ID	Charges?	Size	Outcome
1	Y	Small	Truthful
2	N	Small	Truthful
3	N	Large	Truthful
4	N	Large	Truthful
5	N	Small	Truthful
6	N	Small	Truthful
7	Y	Small	Fraud
8	Y	Large	Fraud
9	N	Large	Fraud
10	Y	Large	Fraud

Goal: new record: small firm, charges = yes

An Illustrative Example

ID	Charges?	Size	Outcome
1	Y	Small	Truthful
2	N	Small	Truthful
3	N	Large	Truthful
4	N	Large	Truthful
5	N	Small	Truthful
6	N	Small	Truthful
7	Y	Small	Fraud
8	Y	Large	Fraud
9	N	Large	Fraud
10	Y	Large	Fraud

Goal: new record: small firm, charges = yes

$$P(size = small|Fraund) = 0.25$$
$$P(charge = Y|Fraud) = 0.75$$
$$P(size = small|Truthful) = 4/6$$
$$P(charge = Y|Truthful) = 1/6$$

$$P(Fraud) \times 0.25 \times 0.75 = 0.075$$
$$P(Truthful) \times \left(\frac{4}{6}\right) \times \left(\frac{1}{6}\right) = 0.067$$
$$P(Fraud|small, yes) = \frac{0.075}{0.075 + 0.067} = \mathbf{0.53}$$

Naïve Bayes Model

- **Suppose**: Depends on the choice of event model $P(X_k|Y)$
- **Model**: Product of prior and the model

$$P(\boldsymbol{X}, Y) = P(Y)\prod_{k=1}^{K} P(X_k|Y)$$

- **Training**: Find the class-conditional MLE parameters
  - For $P(Y)$, we find the MLE using all the data.
  - For each $P(X_k|Y)$, we condition on the data with the corresponding
- **Classification**: Find the class that maximizes the posterior

$$\hat{y} = argmax_y p(y|\boldsymbol{x})$$

$$= argmax_y p(\boldsymbol{x}|y)p(y)/p(x)$$

$$= argmax_y p(\boldsymbol{x}|y)p(y)$$

A shortcoming of MLE

- suppose we never observe the word "unicorn" in a real news article?

A shortcoming of MLE

- suppose we never observe the word "unicorn" in a real news article?

- Add-1 Smoothing

$$D = \left\{\left(\boldsymbol{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}, D' = D \cup \{(\mathbf{0}, 0), (\mathbf{0}, 1), (\mathbf{1}, 0), (\mathbf{1}, 1)\}$$

$$\theta_{k,0} = \frac{1 + \sum_{i=1}^{N} \mathbb{I}\left(y^{(i)} = 0 \wedge x_k^{(i)} = 1\right)}{2 + \sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0)}$$

Other NB Models

- Bernoulli Naïve Bayes:
  - For binary features
- Multinomial Naïve Bayes:
  - For integer features
- Gaussian Naïve Bayes
  - For continuous features

Model 2: Multinomial Naïve Bayes

- Data: $\boldsymbol{x} = [x_1, x_2, \dots, x_M]$, where $x_m \in \{1, \dots, K\}$

Generative Process:

for $i \in \{1, \dots, N\}$:

$y \sim Bernoulli(\phi)$

for $j \in \{1, \dots, M_i\}$:

$x_j^{(i)} \sim Multinomial\left(\boldsymbol{\theta}_{y^{(i)}}, 1\right)$

Model:

$$p_{\phi,\boldsymbol{\theta}}(\boldsymbol{x}, y)$$

$$= \left[(\phi)^y (1-\phi)^{(1-y)} \prod_{j=1}^{M_i} \theta_{y,x_j}\right]$$

Model 3: Gaussian Naïve Bayes

- Data: $x \in \mathbb{R}^M$

Model:

$$p(\boldsymbol{x}, y) = p(x_1, x_2, \dots, x_M, y)$$

$$= p(y) \prod_{k=1}^{M} p(x_k | y)$$

Gaussian Naïve Bayes assumes that $p(x_k|y)$ is given by a normal distribution.