Elective Course
Course Code: CS4103
Autumn 2025-26

Lecture #46

Artificial Intelligence for Data Science

Week-13:
MACHINE LEARNING (Part XIV)
Support Vector Machine (SVM)
Naïve Bayes Classifier

Course Instructor:

Dr. Monidipa Das
Assistant Professor
Department of Computational and Data Sciences
Indian Institute of Science Education and Research Kolkata, India 741246

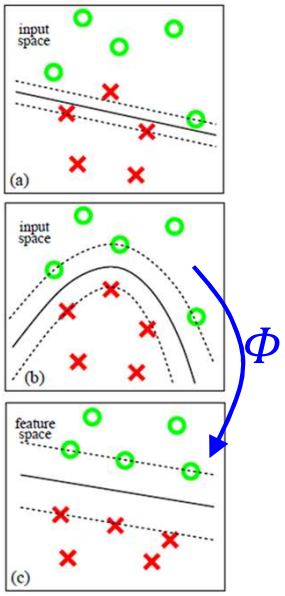
The Kernel Trick

- Recall the SVM optimization problem

Find $\alpha_1 \ldots \alpha_N$ such that
$\mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T\mathbf{x}_j$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $0 \le \alpha_i \le C$ for all α_i

- The data points only appear as inner product
- As long as we can calculate the inner product in the feature space, we do not need the mapping explicitly
- Define the kernel function K by

$$K(\mathbf{x}_i, \mathbf{x}_j) = \phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$



Modification Due to Kernel Function

- Change all inner products to kernel functions
- For training,

Original

Find $\alpha_1 \ldots \alpha_N$ such that
$\mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $0 \le \alpha_i \le C$ for all α_i

With kernel function

Find $\alpha_1 \ldots \alpha_N$ such that
$\mathbf{Q}(\boldsymbol{\alpha}) = \Sigma\alpha_i - \frac{1}{2}\Sigma\Sigma\alpha_i\alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j)$ is maximized and
(1) $\Sigma\alpha_i y_i = 0$
(2) $0 \le \alpha_i \le C$ for all α_i

Modification Due to Kernel Function

- For testing, the new data $\boldsymbol{x}$ is classified as class 1 if $f \geq 0$, and as class 2 if $f < 0$

Original

$$\boldsymbol{w} = \Sigma\alpha_i y_i \boldsymbol{x_i}$$

$$f(\boldsymbol{x}) = \Sigma\alpha_i y_i \boldsymbol{x_i}^T \boldsymbol{x} + b$$

With kernel function

$$\boldsymbol{w} = \Sigma\alpha_i y_i \Phi(\boldsymbol{x_i})$$

$$f(\boldsymbol{\Phi}(\boldsymbol{x})) = \Sigma\alpha_i y_i \boldsymbol{\Phi}(\boldsymbol{x_i})^T \boldsymbol{\Phi}(\boldsymbol{x}) + b$$

$$f(\boldsymbol{\Phi}(\boldsymbol{x})) = \Sigma\alpha_i y_i \boldsymbol{K}(\boldsymbol{x_i}, \boldsymbol{x}) + b$$

Examples of Kernel Functions

Linear kernel $K(\mathbf{x}_1, \mathbf{x}_2) = \mathbf{x}_1^\top \mathbf{x}_2$

Polynomial kernel $K(\mathbf{x}_1, \mathbf{x}_2) = (\gamma \cdot \mathbf{x}_1^\top \mathbf{x}_2 + r)^d$

RBF kernel / Gaussian kernel

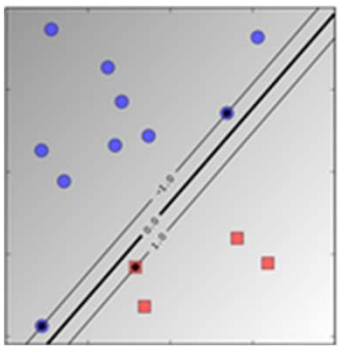
$$K(\mathbf{x}_1, \mathbf{x}_2) = \exp\left(-\gamma \cdot \|\mathbf{x}_1 - \mathbf{x}_2\|^2\right)$$

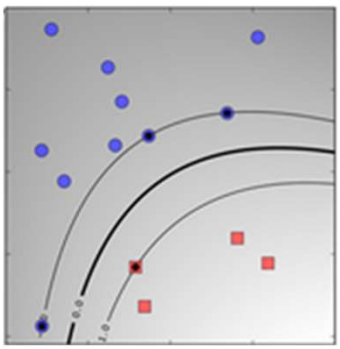
$$\boldsymbol{K(x_1, x_2) = \exp\left(-||x_1 - x_2||^2 / 2\sigma^2\right)}$$

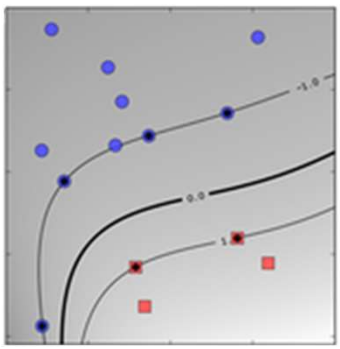
Sigmoid kernel $K(\mathbf{x}_1, \mathbf{x}_2) = \tanh(\gamma \cdot \mathbf{x}_1^\top \mathbf{x}_2 + r)$

Ploynomial kernel: Effect of d

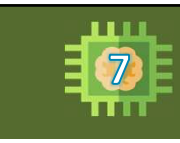
$$K(\mathbf{x}_1, \mathbf{x}_2) = (\mathbf{x}_1^\top \mathbf{x}_2 + \mathbf{1})^d$$



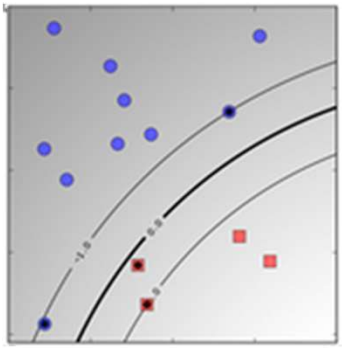


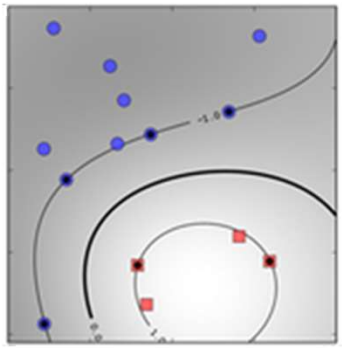


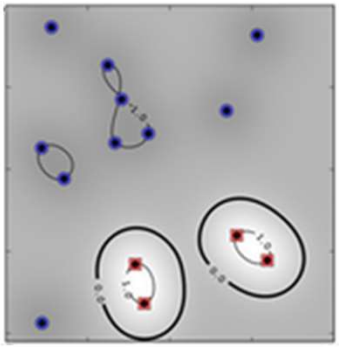
Gaussian RBF kernel: Effect of σ



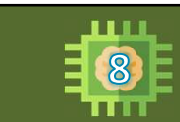
$$K(x_1, x_2) = \exp\left(-||x_1 - x_2||^2 / 2\sigma^2\right)$$







SVM for Multi-Class Classification



- How to use SVM for multi-class classification?
  - One can change the QP formulation to become multi-class
  - More often, multiple binary classifiers are combined
    - **One-versus-Rest (OvR) or One-versus-All (OvA)**
    - **One-versus-One (OvO)**

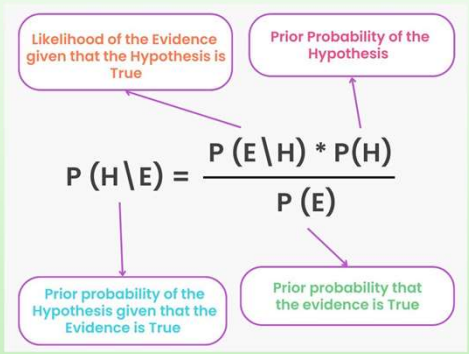
SVM: Strengths and Weaknesses

- Strengths
  - Training is relatively easy
    - No local optimal, unlike in neural networks
  - It scales relatively well to high dimensional data
  - Tradeoff between classifier complexity and error can be controlled explicitly
- Weaknesses
  - Need to choose a "good" kernel function.

Building new kernels

- If $k_1(x, y)$ and $k_2(x, y)$ are two valid kernels then the following kernels are valid
  - *Linear Combination* $k(x, y) = c_1 k_1(x, y) + c_2 k_2(x, y)$
  - *Exponential* $k(x, y) = \exp[k_1(x, y)]$
  - *Product* $k(x, y) = k_1(x, y) \cdot k_2(x, y)$
  - *Polymomial tranfsormation (Q: polymonial with non negative coeffients)* $k(x, y) = Q[k_1(x, y)]$
  - *Function product (f: any function)* $k(x, y) = f(x) k_1(x, y) f(y)$

11



Naïve Bayes

$$P(C \mid \mathbf{X}) \propto P(\mathbf{X} \mid C)P(C) = P(X_1, \cdots, X_n \mid C)P(C)$$

$$\begin{aligned} P(X_1, X_2, \cdots, X_n \mid C) &= P(X_1 \mid X_2, \cdots, X_n; C)P(X_2, \cdots, X_n \mid C) \\ &= P(X_1 \mid C)P(X_2, \cdots, X_n \mid C) \\ &= P(X_1 \mid C)P(X_2 \mid C) \cdots P(X_n \mid C) \end{aligned}$$

Probabilistic Classification

12

- Establishing a probabilistic model for classification
  - Discriminative model $P(C \mid \mathbf{X}) \quad C = c_1, \cdots, c_L, \mathbf{X} = (X_1, \cdots, X_n)$
  - Generative model $P(\mathbf{X} \mid C) \quad C = c_1, \cdots, c_L, \mathbf{X} = (X_1, \cdots, X_n)$
- MAP classification rule
  - MAP: Maximum A Posterior
  - Assign x to c^* if $P(C = c^* \mid \mathbf{X} = \mathbf{x}) > P(C = c \mid \mathbf{X} = \mathbf{x}) \quad c \neq c^*, \qquad c = c_1, \cdots, c_L$
- Generative classification with the MAP rule
  - Apply Bayesian rule to convert: $P(C \mid \mathbf{X}) = \frac{P(\mathbf{X} \mid C)P(C)}{P(\mathbf{X})} \propto P(\mathbf{X} \mid C)P(C)$

Naïve Bayes

- Bayes classification

$$P(C \mid \mathbf{X}) \propto P(\mathbf{X} \mid C)P(C) = P(X_1, \cdots, X_n \mid C)P(C)$$

Difficulty: learning the joint probability $P(X_1, \cdots, X_n \mid C)$

- Naïve Bayes classification
  - Making the assumption that all input attributes are **conditionally** independent

$$\begin{aligned} P(X_1, X_2, \cdots, X_n \mid C) &= P(X_1 \mid X_2, \cdots, X_n; C)P(X_2, \cdots, X_n \mid C) \\ &= P(X_1 \mid C)P(X_2, \cdots, X_n \mid C) \\ &= P(X_1 \mid C)P(X_2 \mid C) \cdots P(X_n \mid C) \end{aligned}$$

  - MAP classification rule

$$[P(x_1 \mid c^*) \cdots P(x_n \mid c^*)]P(c^*) > [P(x_1 \mid c) \cdots P(x_n \mid c)]P(c), \quad c \neq c^*, c = c_1, \cdots, c_L$$

Naïve Bayes

- Naïve Bayes Algorithm (for discrete input attributes)
  - Learning Phase: Given a training set **S**,

For each target value of c_i $(c_i = c_1, \cdots, c_L)$

$\hat{P}(C = c_i) \leftarrow$ estimate $P(C = c_i)$ with examples in **S**;

For every attribute value a_{jk} of each attribute x_j $(j = 1, \cdots, n; k = 1, \cdots, N_j)$

$\hat{P}(X_j = a_{jk} \mid C = c_i) \leftarrow$ estimate $P(X_j = a_{jk} \mid C = c_i)$ with examples in **S**;

Output: conditional probability tables; for x_j, $N_j \times L$ elements

  - Test Phase: Given an unknown instance, $\mathbf{X}' = (a_1', \cdots, a_n')$

Look up tables to assign the label c^* to $\mathbf{X}'$ if

$$[\hat{P}(a_1' \mid c^*) \cdots \hat{P}(a_n' \mid c^*)]\hat{P}(c^*) > [\hat{P}(a_1' \mid c) \cdots \hat{P}(a_n' \mid c)]\hat{P}(c), \quad c \neq c^*, c = c_1, \cdots, c_L$$

Example

- Example: Play Tennis

PlayTennis: training examples

Day	Outlook	Temperature	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

Example

- Learning Phase

Outlook	Play=*Yes*	Play=*No*
Sunny	2/9	3/5
Overcast	4/9	0/5
Rain	3/9	2/5

Temperature	Play=*Yes*	Play=*No*
Hot	2/9	2/5
Mild	4/9	2/5
Cool	3/9	1/5

Humidity	Play=*Yes*	Play=*No*
High	3/9	4/5
Normal	6/9	1/5

Wind	Play=*Yes*	Play=*No*
Strong	3/9	3/5
Weak	6/9	2/5

$P(\text{Play}=Yes) = 9/14$ $P(\text{Play}=No) = 5/14$

Example

- Test Phase
  - Given a new instance,

    $\mathbf{x}'$=(Outlook=*Sunny*, Temperature=*Cool*, Humidity=*High*, Wind=*Strong*)
  - Look up tables

P(Outlook=*Sunny*|Play=*Yes*) = 2/9
P(Temperature=*Cool*|Play=*Yes*) = 3/9
P(Huminity=*High*|Play=*Yes*) = 3/9
P(Wind=*Strong*|Play=*Yes*) = 3/9
P(Play=*Yes*) = 9/14

P(Outlook=S*unny*|Play=*No*) = 3/5
P(Temperature=*Cool*|Play=*No*) = 1/5
P(Huminity=*High*|Play=*No*) = 4/5
P(Wind=*Strong*|Play=*No*) = 3/5
P(Play=*No*) = 5/14

  - MAP rule

P(*Yes*|$\mathbf{x}'$): [P(*Sunny*|*Yes*)P(*Cool*|*Yes*)P(*High*|*Yes*)P(*Strong*|*Yes*)]P(Play=*Yes*) = 0.0053
P(*No*|$\mathbf{x}'$): [P(*Sunny*|No) P(*Cool*|No)P(*High*|*No*)P(*Strong*|*No*)]P(Play=*No*) = 0.0206

Given the fact P(*Yes*|$\mathbf{x}'$) < P(*No*|$\mathbf{x}'$), we label $\mathbf{x}'$ to be "*No*".

Naïve Bayes: Relevant Issues

- Violation of Independence Assumption
  - For many real-world tasks, $P(X_1,\cdots,X_n|C) \neq P(X_1|C)\cdots P(X_n|C)$
  - Nevertheless, naïve Bayes works surprisingly well anyway!
- Zero conditional probability Problem
  - If no example contains the attribute value $X_j = a_{jk}, \hat{P}(X_j = a_{jk}|C = c_i) = 0$
  - In this circumstance, $\hat{P}(x_1|c_i)\cdots\hat{P}(a_{jk}|c_i)\cdots\hat{P}(x_n|c_i) = 0$ during test
  - For a remedy, conditional probabilities estimated with

$$\hat{P}(X_j = a_{jk}|C = c_i) = \frac{n_c + mp}{n + m}$$

n_c : number of training examples for which $X_j = a_{jk}$ and C = c_i
n : number of training examples for which $C = c_i$
p : prior estimate (usually, $p = 1/t$ for t possible values of X_j)
m : weight to prior (number of "virtual" examples, $m \geq 1$)

Naïve Bayes: Relevant Issues

- Continuous-valued Input Attributes
  - Numberless values for an attribute
  - Conditional probability modeled with the normal distribution

$$\hat{P}(X_j \mid C = c_i) = \frac{1}{\sqrt{2\pi}\sigma_{ji}} \exp\left(-\frac{(X_j - \mu_{ji})^2}{2\sigma_{ji}^2}\right)$$

μ_{ji} : mean (avearage) of attribute values X_j of examples for which $C = c_i$

σ_{ji} : standard deviation of attribute values X_j of examples for which $C = c_i$

  - Learning Phase: for $\mathbf{X} = (X_1, \cdots, X_n)$, $C = c_1, \cdots, c_L$
    Output: $n \times L$ normal distributions and $P(C = c_i)$ $i = 1, \cdots, L$
  - Test Phase: for $\mathbf{X}' = (X_1', \cdots, X_n')$
    - Calculate conditional probabilities with all the normal distributions
    - Apply the MAP rule to make a decision

Naïve Bayes: Advantages

- **Simplicity and speed**
- **Small datasets**
- **High-dimensional data**
- **Irrelevant features**
- **Real-time predictions**
- **Handles missing data**

Naïve Bayes: Disadvantages

- **Independence assumption**
- **Unreliable probability estimates**
- **Correlated features**
- **Continuous data limitations**

Questions?