Kernel Support Vector Machine

Li-Hsien Sun
National Central University

July 30, 2018

SVM Revisted

- The SVM problem is

$$\min_{b,\mathbf{w}} \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to

$$y_n(\mathbf{w}^T z_n + b) \geq 1, \quad \forall n,$$

where z_n is $\tilde{d}$-dimension.

- Notice that in the SVM problem
  - $\tilde{d} + 1$ variables
  - N constraints

SVM Revisted

- The SVM problem is

$$\min_{b,\mathbf{w}} \frac{1}{2}\mathbf{w}^T\mathbf{w}$$

subject to

$$y_n(\mathbf{w}^T z_n + b) \geq 1, \quad \forall n,$$

where z_n is $\tilde{d}$-dimension.

- Notice that in the SVM problem
  - $\tilde{d} + 1$ variables
  - N constraints

Dual SVM Revisted

- The dual SVM problem is rewritten as the Lagrangian given by

$$\max_{\alpha_n \geq 0, \forall n} \min_{b,\mathbf{w}} L(b, \mathbf{w}, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{n=1}^{N} \alpha_n (1 - y_n(\mathbf{w}^T\mathbf{z}_n + b))$$

- Dual SVM
  - N variables
  - $N + 1$ constraints

Dual SVM Revisted

- The dual SVM problem is rewritten as the Lagrangian given by

$$\max_{\alpha_n \geq 0, \forall n} \min_{b,\mathbf{w}} L(b, \mathbf{w}, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + \sum_{n=1}^{N} \alpha_n (1 - y_n(\mathbf{w}^T\mathbf{z}_n + b))$$

- Dual SVM
  - N variables
  - $N + 1$ constraints

Dual SVM Revisted

- The first order condition gives

$$\partial_b L|_{b=\hat{b}} = 0, \quad \sum_{n=1}^{N} \alpha_n y_n = 0$$

and

$$\partial_{\mathbf{w}} L|_{\mathbf{w}=\hat{\mathbf{w}}} = 0, \quad \hat{\mathbf{w}} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n$$

and $\hat{b} = y_n - \hat{\mathbf{w}}^T \mathbf{z}_n$ ($\alpha_n(1 - y_n(\hat{\mathbf{w}}^T \mathbf{z}_n + \hat{b})) = 0$. $\hat{b}$ is determined by the vectors with $\alpha_n > 0$ (support vectors).)

- Plugging $\hat{w}$ into L and using $\sum_{n=1}^{N} \alpha_n y_n = 0$, we obtain

$$\min_{\forall n, \alpha_n \geq 0,\ \sum_{n=1}^{N} \alpha_n y_n = 0,\ \hat{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n} \frac{1}{2} \| \sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n \|^2 - \sum_{n=1}^{N} \alpha_n$$

Dual SVM Revisted

- The first order condition gives

$$\partial_b L|_{b=\hat{b}} = 0, \quad \sum_{n=1}^{N} \alpha_n y_n = 0$$

and

$$\partial_{\mathbf{w}} L|_{\mathbf{w}=\hat{\mathbf{w}}} = 0, \quad \hat{\mathbf{w}} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n$$

and $\hat{b} = y_n - \hat{\mathbf{w}}^T \mathbf{z}_n$ ($\alpha_n(1 - y_n(\hat{\mathbf{w}}^T \mathbf{z}_n + \hat{b})) = 0$. $\hat{b}$ is determined by the vectors with $\alpha_n > 0$ (support vectors).)

- Plugging $\hat{w}$ into L and using $\sum_{n=1}^{N} \alpha_n y_n = 0$, we obtain

$$\min_{\forall n, \alpha_n \geq 0,\ \sum_{n=1}^{N} \alpha_n y_n = 0,\ \hat{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n} \frac{1}{2} \| \sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n \|^2 - \sum_{n=1}^{N} \alpha_n$$

Dual SVM Revisted

The standard hard margin dual is written as

$$\min_{\alpha} \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m y_n y_m \mathbf{z}_n^T \mathbf{z}_m - \sum_{n=1}^{M} \alpha_n$$

subject to

$$\sum_{n=1}^{N} \alpha_n y_n = 0, \quad \alpha_n \geq 0, \quad n = 1, 2, \cdots, N.$$

Kernel SVM

Dual SVM to the Quadratic programming problem

$$\min_{\alpha} \alpha^T Q_D \alpha - \mathbf{1}^T \alpha$$

subject to

$$\mathbf{y}^T \alpha = 0, \quad \alpha_n \geq 0, \quad n = 1, 2, \cdots, N.$$

where $q_{n,m} = y_n y_m \mathbf{z}_n^T \mathbf{z}_m$.

- Notice that $\mathbf{z}_n^T \mathbf{z}_m$ is the inner product in $R^{\tilde{d}}$.
- Find the way to obtain $\mathbf{z}_n^T \mathbf{z}_m = \Phi(\mathbf{x}_n)^T \Phi(\mathbf{x}_m)$ faster than $\mathcal{O}(\tilde{d})$.

Kernel SVM

Dual SVM to the Quadratic programming problem

$$\min_{\alpha} \alpha^T Q_D \alpha - \mathbf{1}^T \alpha$$

subject to

$$\mathbf{y}^T \alpha = 0, \quad \alpha_n \geq 0, \quad n = 1, 2, \cdots, N.$$

where $q_{n,m} = y_n y_m \mathbf{z}_n^T \mathbf{z}_m$.

- Notice that $\mathbf{z}_n^T \mathbf{z}_m$ is the inner product in $R^{\tilde{d}}$.
- Find the way to obtain $\mathbf{z}_n^T \mathbf{z}_m = \Phi(\mathbf{x}_n)^T \Phi(\mathbf{x}_m)$ faster than $\mathcal{O}(\tilde{d})$.

Kernel SVM

The solution:

Plug in efficient Kernel functions to avoid the dependence of $\tilde{d}$.

Second Order Polynomial Transform

- Define the 2-nd order transform written as

$$\Phi_2(\mathbf{x}) = (1, x_1, x_2, \cdots, x_d, x_1^2, x_1x_2, \cdots, x_1x_d, x_2^2, x_2x_1, \cdots, x_2x_d \cdots, x_d^2)$$

- Transform + inner product is

$$\begin{aligned}
\Phi_2(\mathbf{x})^T\Phi(\mathbf{x}') &= 1 + \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d}\sum_{j=1}^{d} x_i x_j x_i' x_j' \\
&= 1 + \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d} x_i x_i' \sum_{j=1}^{d} x_j x_j' \\
&= 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2
\end{aligned}$$

Second Order Polynomial Transform

- Define the 2-nd order transform written as

$$\Phi_2(\mathbf{x}) = (1, x_1, x_2, \cdots, x_d, x_1^2, x_1x_2, \cdots, x_1x_d, x_2^2, x_2x_1, \cdots, x_2x_d \cdots, x_d^2)$$

- Transform + inner product is

$$\begin{aligned}
\Phi_2(\mathbf{x})^T\Phi(\mathbf{x}') &= 1 + \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d}\sum_{j=1}^{d} x_i x_j x_i' x_j' \\
&= 1 + \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d} x_i x_i' \sum_{j=1}^{d} x_j x_j' \\
&= 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2
\end{aligned}$$

- Hence, in the case of Φ_2, transform + inner product can be obtain in $\mathcal{O}(\tilde{d})$ instead of $\mathcal{O}(\tilde{d}^2)$

Second Order Polynomial Transform

- Define the 2-nd order transform written as

$$\Phi_2(\mathbf{x}) = (1, x_1, x_2, \cdots, x_d, x_1^2, x_1x_2, \cdots, x_1x_d, x_2^2, x_2x_1, \cdots, x_2x_d \cdots, x_d^2)$$

- Transform + inner product is

$$\begin{aligned}
\Phi_2(\mathbf{x})^T\Phi(\mathbf{x}') &= 1 + \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d}\sum_{j=1}^{d} x_i x_j x_i' x_j' \\
&= 1 + \sum_{i=1}^{d} x_i x_i' + \sum_{i=1}^{d} x_i x_i' \sum_{j=1}^{d} x_j x_j' \\
&= 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2
\end{aligned}$$

- Hence, in the case of Φ_2, transform + inner product can be obtain in $\mathcal{O}(\tilde{d})$ instead of $\mathcal{O}(\tilde{d}^2)$

Kernel Functions

- The kernel function for transform Φ is defined as

$$K_\Phi(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^T \Phi(\mathbf{x}')$$

- For instance, for Φ_2, we define the corresponding Kernel function $K_{\Phi_2}(\mathbf{x}, \mathbf{x}') = 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2$

Kernel Functions

- The kernel function for transform Φ is defined as

$$K_\Phi(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^T \Phi(\mathbf{x}')$$

- For instance, for Φ_2, we define the corresponding Kernel function $K_{\Phi_2}(\mathbf{x}, \mathbf{x}') = 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2$
- Hence, we have $q_{n,m} = y_n y_m \mathbf{z}_n^T \mathbf{z}_m = y_n y_m K(\mathbf{x}_n, \mathbf{x}_m)$ leading to

Kernel Functions

- The kernel function for transform Φ is defined as

$$K_\Phi(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^T \Phi(\mathbf{x}')$$

- For instance, for Φ_2, we define the corresponding Kernel function $K_{\Phi_2}(\mathbf{x}, \mathbf{x}') = 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2$
- Hence, we have $q_{n,m} = y_n y_m \mathbf{z}_n^T \mathbf{z}_m = y_n y_m K(\mathbf{x}_n, \mathbf{x}_m)$ leading to
  - The optimal bias from support vectors $(\mathbf{z}_s, y_s)$ is

$$b = y_s - w^T \mathbf{z}_s = y_s - \left(\sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n\right)^T \mathbf{z}_s = y_s - \sum_{n=1}^{N} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}_s)$$

Kernel Functions

- The kernel function for transform Φ is defined as

$$K_\Phi(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^T \Phi(\mathbf{x}')$$

- For instance, for Φ_2, we define the corresponding Kernel function $K_{\Phi_2}(\mathbf{x}, \mathbf{x}') = 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2$
- Hence, we have $q_{n,m} = y_n y_m \mathbf{z}_n^T \mathbf{z}_m = y_n y_m K(\mathbf{x}_n, \mathbf{x}_m)$ leading to
  - The optimal bias from support vectors $(\mathbf{z}_s, y_s)$ is

$$b = y_s - w^T \mathbf{z}_s = y_s - \left(\sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n \right)^T \mathbf{z}_s = y_s - \sum_{n=1}^{N} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}_s)$$

  - The optimal hypothesis for a test input $\mathbf{x}$ is

$$g_{SVM}(\mathbf{x}) = \text{sign}\left(w^T \Phi(\mathbf{x}) + b\right) = \text{sign}\left(\sum_{n=1}^{N} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b \right)$$

Kernel Functions

- The kernel function for transform Φ is defined as

$$K_\Phi(\mathbf{x}, \mathbf{x}') = \Phi(\mathbf{x})^T \Phi(\mathbf{x}')$$

- For instance, for Φ_2, we define the corresponding Kernel function $K_{\Phi_2}(\mathbf{x}, \mathbf{x}') = 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2$
- Hence, we have $q_{n,m} = y_n y_m \mathbf{z}_n^T \mathbf{z}_m = y_n y_m K(\mathbf{x}_n, \mathbf{x}_m)$ leading to
  - The optimal bias from support vectors $(\mathbf{z}_s, y_s)$ is

$$b = y_s - w^T \mathbf{z}_s = y_s - \left(\sum_{n=1}^{N} \alpha_n y_n \mathbf{z}_n \right)^T \mathbf{z}_s = y_s - \sum_{n=1}^{N} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}_s)$$

  - The optimal hypothesis for a test input $\mathbf{x}$ is

$$g_{SVM}(\mathbf{x}) = \text{sign}\left(w^T \Phi(\mathbf{x}) + b\right) = \text{sign}\left(\sum_{n=1}^{N} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b \right)$$

Kernel SVM with QP

1. $q_{n,m} = y_n y_m K(x_n, x_m)$, $\mathbf{p} = -\mathbf{1}_N$, A and $\mathbf{c}$ for the constraints
2. $\alpha \leftarrow QP(Q_D, \mathbf{p}, A, \mathbf{c})$
3. $b \leftarrow y_s - \sum_{\text{SV indices n}} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}_s)$ with SV $(\mathbf{x}_s, y_s)$
4. For new $\mathbf{x}$, the optimal hypothesis is

$$g_{SVM}(\mathbf{x}) = \text{sign}\left(\sum_{\text{SV indices n}} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b\right)$$

General Poly-2 Kernel

$$\Phi_2(\mathbf{x}) = (1, x_1, \cdots, x_d, x_1^2, \cdots, x_d^2) \Longleftrightarrow K_{\Phi_2} = 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2$$

$$\tilde{\Phi}_2(x) = (1, \sqrt{2\gamma}x_1, \cdots, \sqrt{2\gamma}x_d, \gamma x_1^2, \cdots, \gamma x_d^2) \Longleftrightarrow$$

$$K_2 = 1 + 2\gamma\mathbf{x}^T\mathbf{x}' + \gamma^2(\mathbf{x}^T\mathbf{x}')^2 = (1 + \gamma\mathbf{x}^T\mathbf{x}')^2, \quad \gamma > 0$$

General Poly-2 Kernel

$$\Phi_2(\mathbf{x}) = (1, x_1, \cdots, x_d, x_1^2, \cdots, x_d^2) \Longleftrightarrow K_{\Phi_2} = 1 + \mathbf{x}^T\mathbf{x}' + (\mathbf{x}^T\mathbf{x}')^2$$

$$\tilde{\Phi}_2(x) = (1, \sqrt{2\gamma}x_1, \cdots, \sqrt{2\gamma}x_d, \gamma x_1^2, \cdots, \gamma x_d^2) \Longleftrightarrow$$

$$K_2 = 1 + 2\gamma\mathbf{x}^T\mathbf{x}' + \gamma^2(\mathbf{x}^T\mathbf{x}')^2 = (1 + \gamma\mathbf{x}^T\mathbf{x}')^2, \quad \gamma > 0$$

Remark

- K_2 is commonly used. (Somewhat K_2 is 'easier' to calculate than K_{Φ_2})
- Φ_2 and $\tilde{\Phi}_2$ have the equivalent power

Remark

- K_2 is commonly used. (Somewhat K_2 is 'easier' to calculate than K_{Φ_2})
- Φ_2 and $\tilde{\Phi}_2$ have the equivalent power
  Different inner product $\Rightarrow$ Different geometry

Remark

- K_2 is commonly used. (Somewhat K_2 is 'easier' to calculate than K_{Φ_2})
- Φ_2 and $\tilde{\Phi}_2$ have the equivalent power
  Different inner product $\Rightarrow$ Different geometry
- g_{SVM} is different implying that SVs is different

Remark

- K_2 is commonly used. (Somewhat K_2 is 'easier' to calculate than K_{Φ_2})
- Φ_2 and $\tilde{\Phi}_2$ have the equivalent power
  Different inner product $\Rightarrow$ Different geometry
- g_{SVM} is different implying that SVs is different
- Change of Kernel functions $\leftrightarrow$ Change of the margin definition

Remark

- K_2 is commonly used. (Somewhat K_2 is 'easier' to calculate than K_{Φ_2})
- Φ_2 and $\tilde{\Phi}_2$ have the equivalent power
  Different inner product $\Rightarrow$ Different geometry
- g_{SVM} is different implying that SVs is different
- Change of Kernel functions $\leftrightarrow$ Change of the margin definition

Need selecting K as well as selecting Φ

Remark

- K_2 is commonly used. (Somewhat K_2 is 'easier' to calculate than K_{Φ_2})
- Φ_2 and $\tilde{\Phi}_2$ have the equivalent power
  Different inner product $\Rightarrow$ Different geometry
- g_{SVM} is different implying that SVs is different
- Change of Kernel functions $\leftrightarrow$ Change of the margin definition

Need selecting K as well as selecting Φ

General Polynomial Kernel

The general polynomial Kernel functions are

$$
\begin{aligned}
K_1(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}'), \quad \gamma > 0, \quad \xi \geq 0 \\
K_2(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^2, \quad \gamma > 0, \quad \xi \geq 0 \\
K_3(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^3, \quad \gamma > 0, \quad \xi \geq 0 \\
&\vdots \\
K_Q(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^Q, \quad \gamma > 0, \quad \xi \geq 0
\end{aligned}
$$

SVM + Polynomial Kernel = Polynomial SVM

General Polynomial Kernel

The general polynomial Kernel functions are

$$\begin{aligned} K_1(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}'), \quad \gamma > 0, \quad \xi \geq 0 \\ K_2(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^2, \quad \gamma > 0, \quad \xi \geq 0 \\ K_3(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^3, \quad \gamma > 0, \quad \xi \geq 0 \\ &\vdots \\ K_Q(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^Q, \quad \gamma > 0, \quad \xi \geq 0 \end{aligned}$$

SVM + Polynomial Kernel = Polynomial SVM

- Embed Φ_Q specially with parameters (γ, ξ)
- Allow considering large-margin polynomial classification with dependence on $\tilde{d}$

General Polynomial Kernel

The general polynomial Kernel functions are

$$\begin{aligned} K_1(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}'), \quad \gamma > 0, \quad \xi \geq 0 \\ K_2(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^2, \quad \gamma > 0, \quad \xi \geq 0 \\ K_3(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^3, \quad \gamma > 0, \quad \xi \geq 0 \\ &\vdots \\ K_Q(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^Q, \quad \gamma > 0, \quad \xi \geq 0 \end{aligned}$$

SVM + Polynomial Kernel = Polynomial SVM

- Embed Φ_Q specially with parameters (γ, ξ)
- Allow considering large-margin polynomial classification with dependence on $\tilde{d}$

Overfitting is controlled roughly by the largest margin (criteria)

General Polynomial Kernel

The general polynomial Kernel functions are

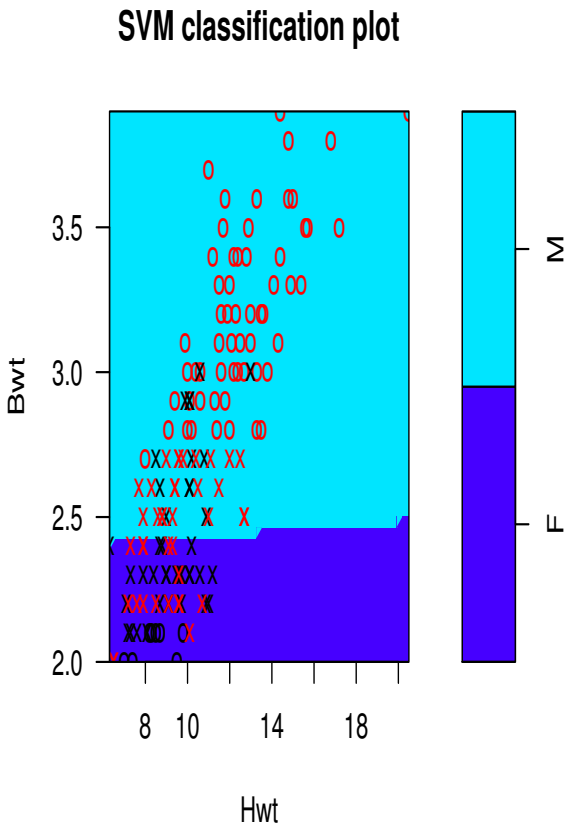
$$\begin{aligned} K_1(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}'), \quad \gamma > 0, \quad \xi \geq 0 \\ K_2(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^2, \quad \gamma > 0, \quad \xi \geq 0 \\ K_3(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^3, \quad \gamma > 0, \quad \xi \geq 0 \\ &\vdots \\ K_Q(\mathbf{x}, \mathbf{x}') &= (\xi + \gamma \mathbf{x}^T \mathbf{x}')^Q, \quad \gamma > 0, \quad \xi \geq 0 \end{aligned}$$

SVM + Polynomial Kernel = Polynomial SVM

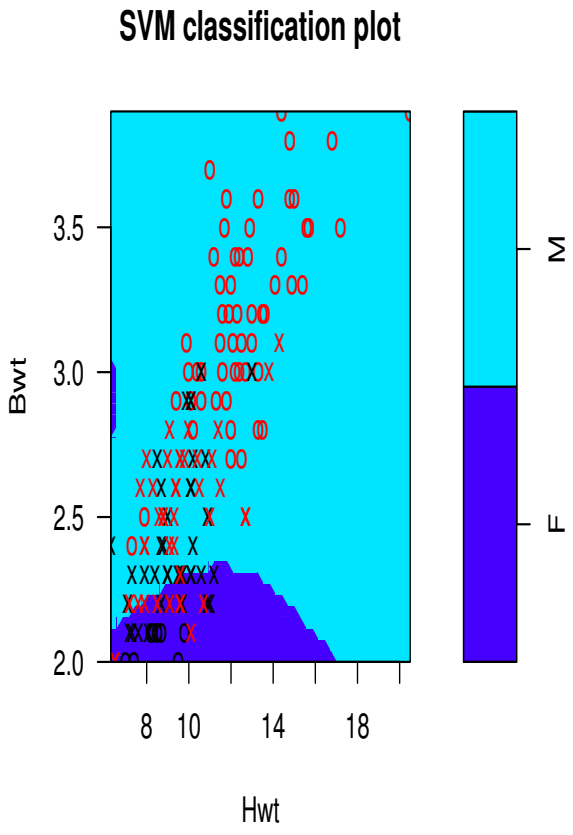
- Embed Φ_Q specially with parameters (γ, ξ)
- Allow considering large-margin polynomial classification with dependence on $\tilde{d}$

Overfitting is controlled roughly by the largest margin (criteria)

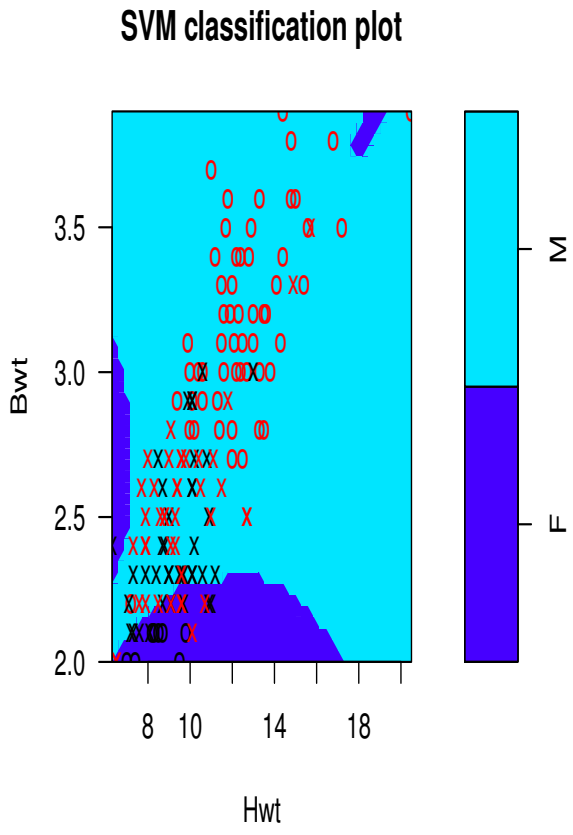
Polynomial SVM with $\gamma = 1$, $\zeta = 0$



$Q = 1$



$Q = 3$



$Q = 5$

Gaussian SVM (Infinite Dimensional Transform)

We first obtain

$$\begin{aligned}
K(x,x') &= \exp\left\{-(x-x')^2\right\} \\
&= \exp\{-x^2\}\exp\{-(x')^2\}\exp\{2xx'\} \\
&= \exp\{-x^2\}\exp\{-(x')^2\}\left(\sum_{i=0}^{\infty}\frac{(2xx')^i}{i!}\right) \\
&= \sum_{i=0}^{\infty}\left(\exp\{-x^2\}\sqrt{\frac{2^i}{i!}}x^i\exp\{-(x')^2\}\sqrt{\frac{2^i}{i!}}x'^i\right) \\
&= \Phi(x)^T\Phi(x')
\end{aligned}$$

where $\Phi(x) = \exp\{-x^2\}\left(1, \sqrt{\frac{2}{1!}}x, \sqrt{\frac{2^2}{2!}}x^2, \cdots\right)$.

Gaussian SVM

- Gaussian SVM Kernel is given by

$$K_G(\mathbf{x}, \mathbf{x}') = \exp\{-\gamma \|\mathbf{x} - \mathbf{x}'\|^2\}$$

with $\gamma > 0$.

- The optimal hypothesis for input $\mathbf{x}$ is

$$\begin{aligned} g_{SVM}(\mathbf{x}) &= \text{sign}\left(\sum_{n \in \mathcal{I}} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b\right) \\ &= \text{sign}\left(\sum_{n \in \mathcal{I}} \alpha_n y_n \exp\{-\gamma \|\mathbf{x} - \mathbf{x}_n\|^2\} + b\right) \end{aligned}$$

where $\mathcal{I} = \{n : \alpha_n > 0\}$.

Gaussian SVM

- Gaussian SVM Kernel is given by

$$K_G(\mathbf{x}, \mathbf{x}') = \exp\{-\gamma \|\mathbf{x} - \mathbf{x}'\|^2\}$$

with $\gamma > 0$.

- The optimal hypothesis for input $\mathbf{x}$ is

$$\begin{aligned} g_{SVM}(\mathbf{x}) &= \text{sign}\left(\sum_{n \in \mathcal{I}} \alpha_n y_n K(\mathbf{x}_n, \mathbf{x}) + b\right) \\ &= \text{sign}\left(\sum_{n \in \mathcal{I}} \alpha_n y_n \exp\{-\gamma \|\mathbf{x} - \mathbf{x}_n\|^2\} + b\right) \end{aligned}$$

where $\mathcal{I} = \{n : \alpha_n > 0\}$.

Remarks

- Gaussian SVM:
  - find α_n to combine Gaussians centered at $\mathbf{x}_n$
  - achieve the large margin in infinite dimensional space
- Also called Radius Basis Function (RBF) kernel

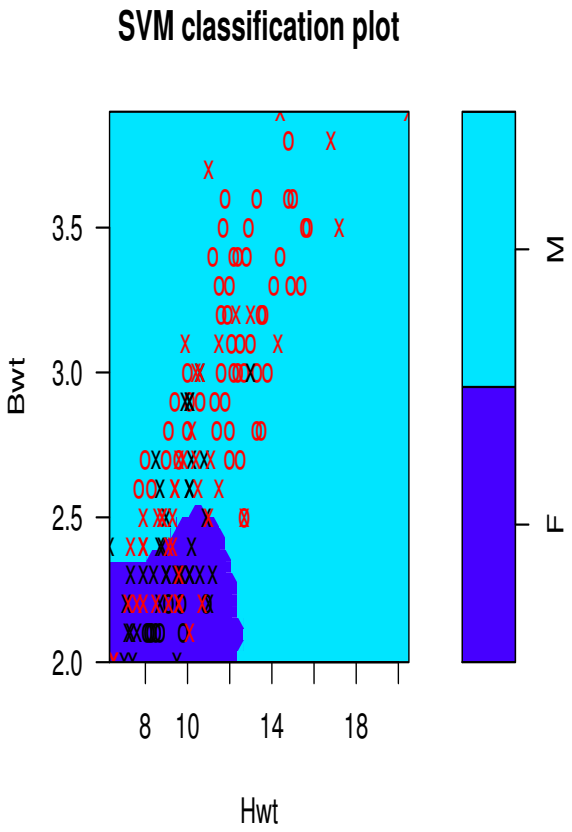
Remarks

- Gaussian SVM:
  - find α_n to combine Gaussians centered at $\mathbf{x}_n$
  - achieve the large margin in infinite dimensional space
- Also called Radius Basis Function (RBF) kernel
- Must select γ carefully (overfitting)

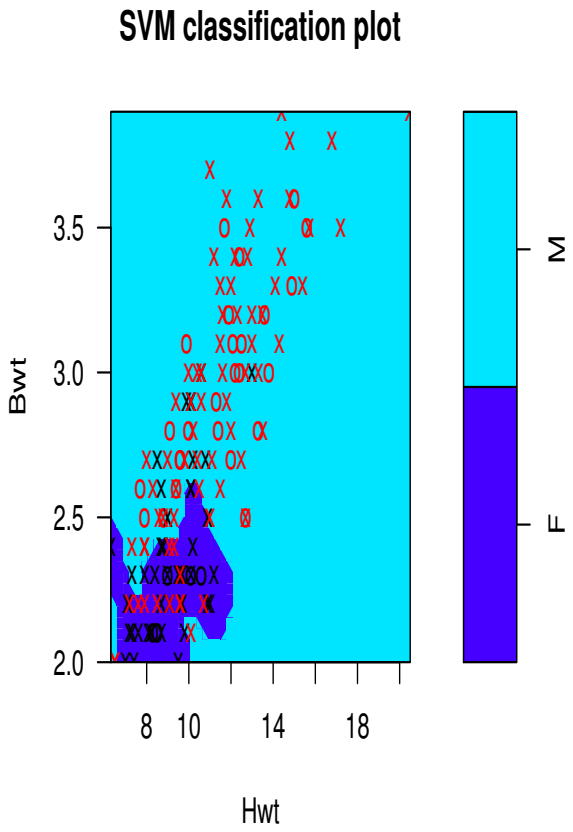
Remarks

- Gaussian SVM:
  - find α_n to combine Gaussians centered at $\mathbf{x}_n$
  - achieve the large margin in infinite dimensional space
- Also called Radius Basis Function (RBF) kernel
- Must select γ carefully (overfitting)

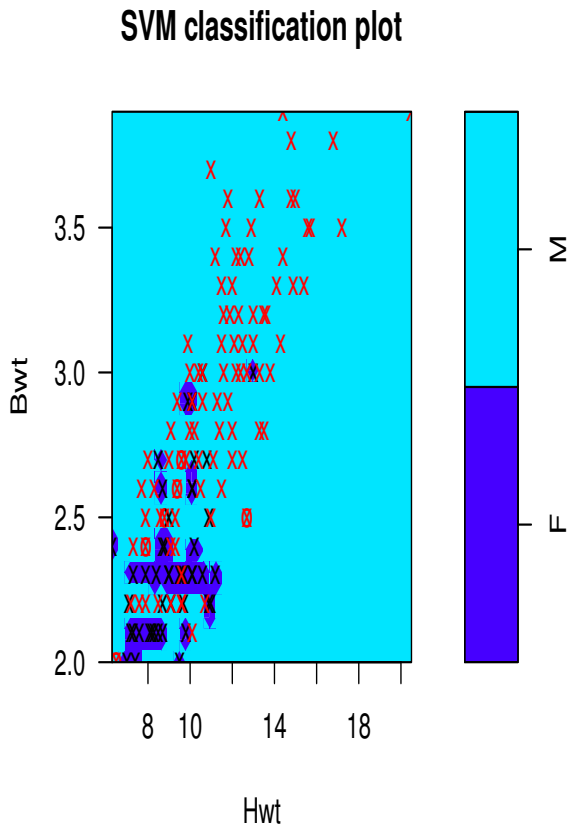
Gaussian Kernel



$\gamma = 1$



$\gamma = 10$



$\gamma = 100$

Summary

- Linear kernel
  - **Pros**: safe, fast (QP solvers), explainable (by $\mathbf{w}$ and SVs)
  - **Cons**: restricted (not always separable)
  - A basic tool

Summary

- Linear kernel
  - **Pros**: safe, fast (QP solvers), explainable (by $\mathbf{w}$ and SVs)
  - **Cons**: restricted (not always separable)
  - A basic tool
- Polynomial kernel
  - **Pros**: less restricted than linear, strong physical control by knowing the degree Q
  - **Cons**: numerical difficulty for large Q ($|\xi + \gamma \mathbf{x}^T \mathbf{x}| >> 1$, K may explode), three parameters (γ, ξ, Q) to be selected

Summary

- Linear kernel
  - **Pros**: safe, fast (QP solvers), explainable (by $\mathbf{w}$ and SVs)
  - **Cons**: restricted (not always separable)
  - A basic tool
- Polynomial kernel
  - **Pros**: less restricted than linear, strong physical control by knowing the degree Q
  - **Cons**: numerical difficulty for large Q ($|\xi + \gamma \mathbf{x}^T \mathbf{x}| >> 1$, K may explode), three parameters (γ, ξ, Q) to be selected
  - Perhaps small Q only

Summary

- Linear kernel
  - **Pros**: safe, fast (QP solvers), explainable (by $\mathbf{w}$ and SVs)
  - **Cons**: restricted (not always separable)
  - A basic tool
- Polynomial kernel
  - **Pros**: less restricted than linear, strong physical control by knowing the degree Q
  - **Cons**: numerical difficulty for large Q ($|\xi + \gamma \mathbf{x}^T \mathbf{x}| >> 1$, K may explode), three parameters (γ, ξ, Q) to be selected
  - Perhaps small Q only
- Gaussian kernel
  - **Pros**: more powerful than linear and poly ones, bounded (less numerical difficulty than Poly one), only one parameter to be selected
  - **Cons**: mysterious (no $\mathbf{w}$), slower than linear, overfitting problem

Summary

- Linear kernel
  - **Pros**: safe, fast (QP solvers), explainable (by $\mathbf{w}$ and SVs)
  - **Cons**: restricted (not always separable)
  - A basic tool
- Polynomial kernel
  - **Pros**: less restricted than linear, strong physical control by knowing the degree Q
  - **Cons**: numerical difficulty for large Q ($|\xi + \gamma \mathbf{x}^T \mathbf{x}| >> 1$, K may explode), three parameters (γ, ξ, Q) to be selected
  - Perhaps small Q only
- Gaussian kernel
  - **Pros**: more powerful than linear and poly ones, bounded (less numerical difficulty than Poly one), only one parameter to be selected
  - **Cons**: mysterious (no $\mathbf{w}$), slower than linear, overfitting problem
  - Popular but be used carefully

Summary

- Linear kernel
  - **Pros**: safe, fast (QP solvers), explainable (by $\mathbf{w}$ and SVs)
  - **Cons**: restricted (not always separable)
  - A basic tool
- Polynomial kernel
  - **Pros**: less restricted than linear, strong physical control by knowing the degree Q
  - **Cons**: numerical difficulty for large Q ($|\xi + \gamma \mathbf{x}^T \mathbf{x}| >> 1$, K may explode), three parameters (γ, ξ, Q) to be selected
  - Perhaps small Q only
- Gaussian kernel
  - **Pros**: more powerful than linear and poly ones, bounded (less numerical difficulty than Poly one), only one parameter to be selected
  - **Cons**: mysterious (no $\mathbf{w}$), slower than linear, overfitting problem
  - Popular but be used carefully

Discussions

- Can we avoid the overfitting problem?

Possible: Soft Margin SVM

Discussions

- Can we avoid the overfitting problem?

Possible: Soft Margin SVM

- The maximum-margin Criteria is

$$\min_{b,\mathbf{w},\xi} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\xi_n$$

subject to

$$y_n(\mathbf{w}^T z_n + b) \geq 1 - \xi_n, \quad \xi_n \geq 0, \quad \forall n,$$

where z_n is $\tilde{d}$-dimension.

Discussions

- Can we avoid the overfitting problem?

Possible: Soft Margin SVM

- The maximum-margin Criteria is

$$\min_{b,\mathbf{w},\xi} \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{n=1}^{N}\xi_n$$

subject to

$$y_n(\mathbf{w}^T z_n + b) \geq 1 - \xi_n, \quad \xi_n \geq 0, \quad \forall n,$$

where z_n is $\tilde{d}$-dimension.

Kernel SVM in R

Package **e1071**

- S3 method for formula
  svm(formula, data = NULL, ..., subset, na.action = na.omit, scale = TRUE)
- S3 method for default
  svm(x, y = NULL, scale = TRUE, type = NULL, kernel = "radial", degree = 3, gamma = if (is.vector(x)) 1 else 1 / ncol(x), coef0 = 0, cost = 1, nu = 0.5, class.weights = NULL, cachesize = 40, tolerance = 0.001, epsilon = 0.1, shrinking = TRUE, cross = 0, probability = FALSE, fitted = TRUE, ..., subset, na.action = na.omit)

The details on
https://www.rdocumentation.org/packages/e1071/versions/1.6-8/topics/svm

Kernel SVM in R

There are 4 Kernels in LIBSVM:

- linear: $K(u, v) =< u, v >$
- polynomial: $K(u, v) = (\gamma < u, v > + coef0)^{degree}$
- radial basis: $K(u, v) = e^{\gamma |u-v|^2}$
- sigmoid: $K(u, v) = \tanh(\gamma < u, v > + coef0)$

The defaults of SVM are as follows

- kernel = "radial",
- degree = 3,
- gamma = 1/(data dimension),
- coef0 = 0,
- cost = 1

Kernel SVM in R

There are 4 Kernels in LIBSVM:

- linear: $K(u, v) =< u, v >$
- polynomial: $K(u, v) = (\gamma < u, v > + coef0)^{degree}$
- radial basis: $K(u, v) = e^{\gamma |u-v|^2}$
- sigmoid: $K(u, v) = \tanh(\gamma < u, v > + coef0)$

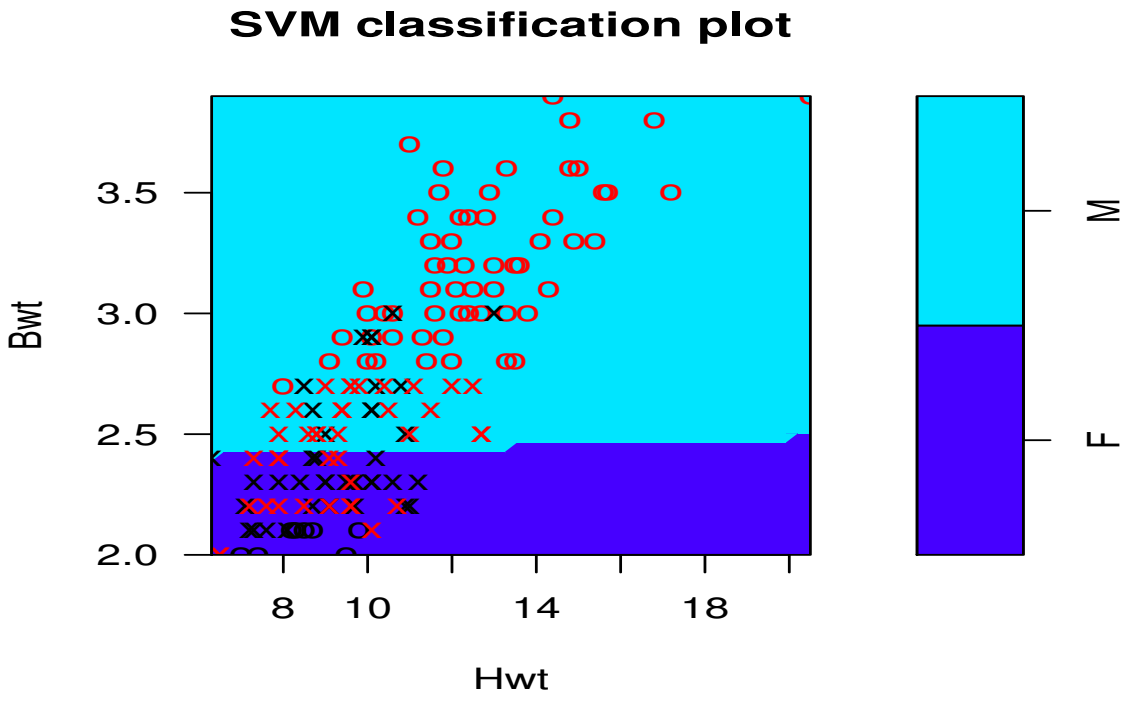
The defaults of SVM are as follows

- kernel = "radial",
- degree = 3,
- gamma = 1/(data dimension),
- coef0 = 0,
- cost = 1

Kernel SVM in R

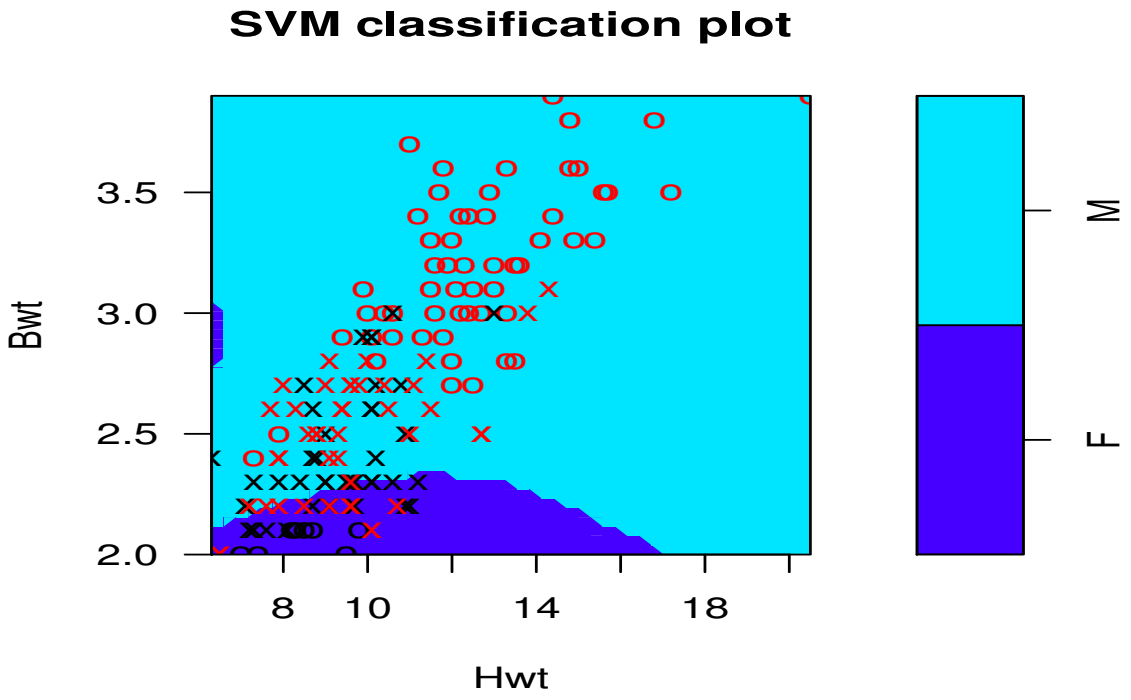
```
library(MASS)
data(cats)
library(e1071)

SVMRBFModel = svm(Sex~., data = cats, gamma = 1,cost=1)
plot(SVMRBFModel,data=cats,color.palette = topo.colors)
predresult =predict(SVMRBFModel , cats)
y=cats$Sex
table(predresult,y)

SVMLinearModel=svm(Sex~., data = cats,kernel="linear")
plot(SVMLinearModel,data=cats,color.palette = topo.colors)
predresult = predict(SVMLinearModel , cats)
y=cats$Sex
table(predresult,y)
```

Linear Kernel



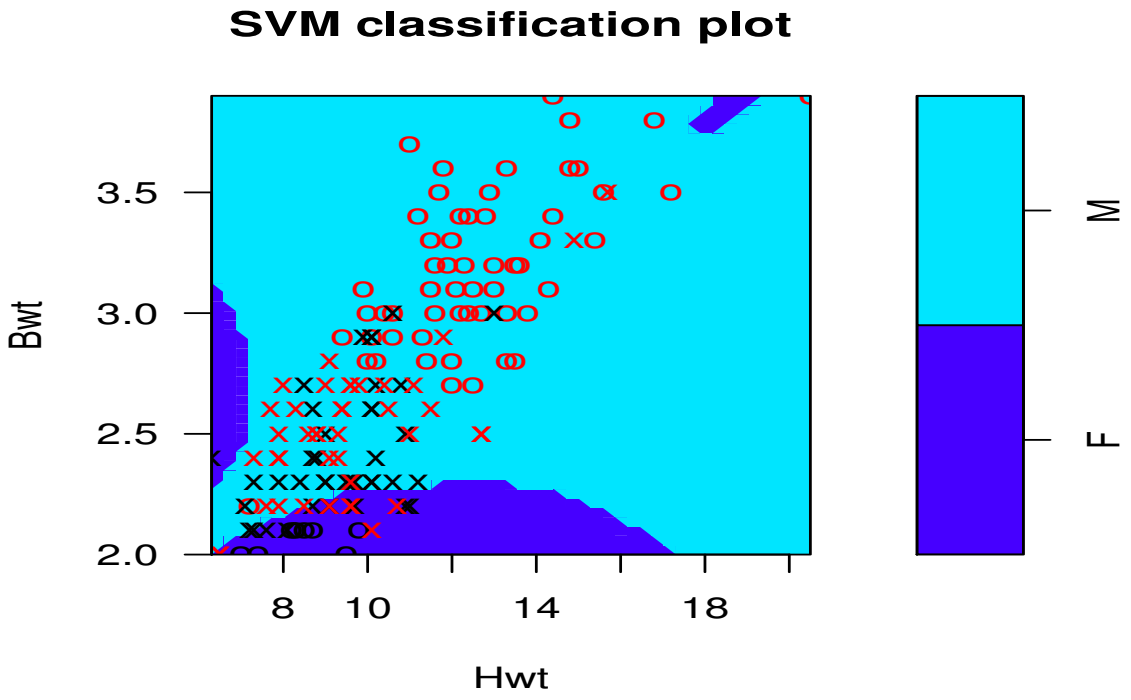
	F	M
F	33	14
M	14	83

Poly Kernel with $Q = 3$, $\gamma = 1$, and $C = 1$



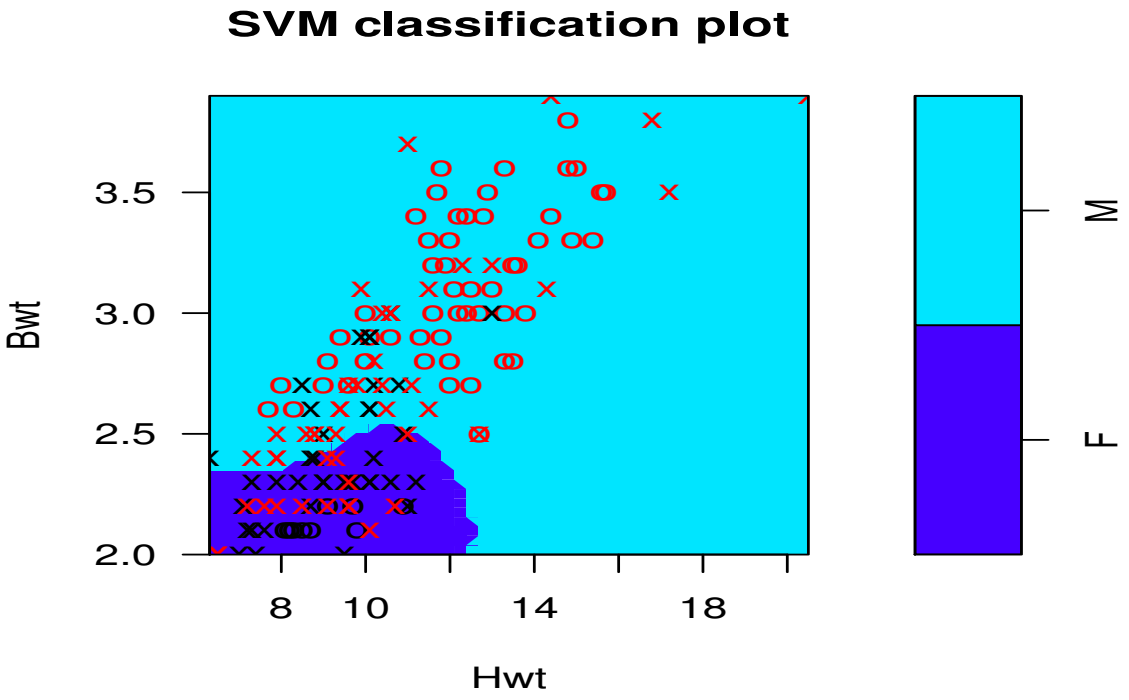
	F	M
F	19	8
M	28	89

Poly Kernel with $Q = 5$, $\gamma = 1$, and $C = 1$



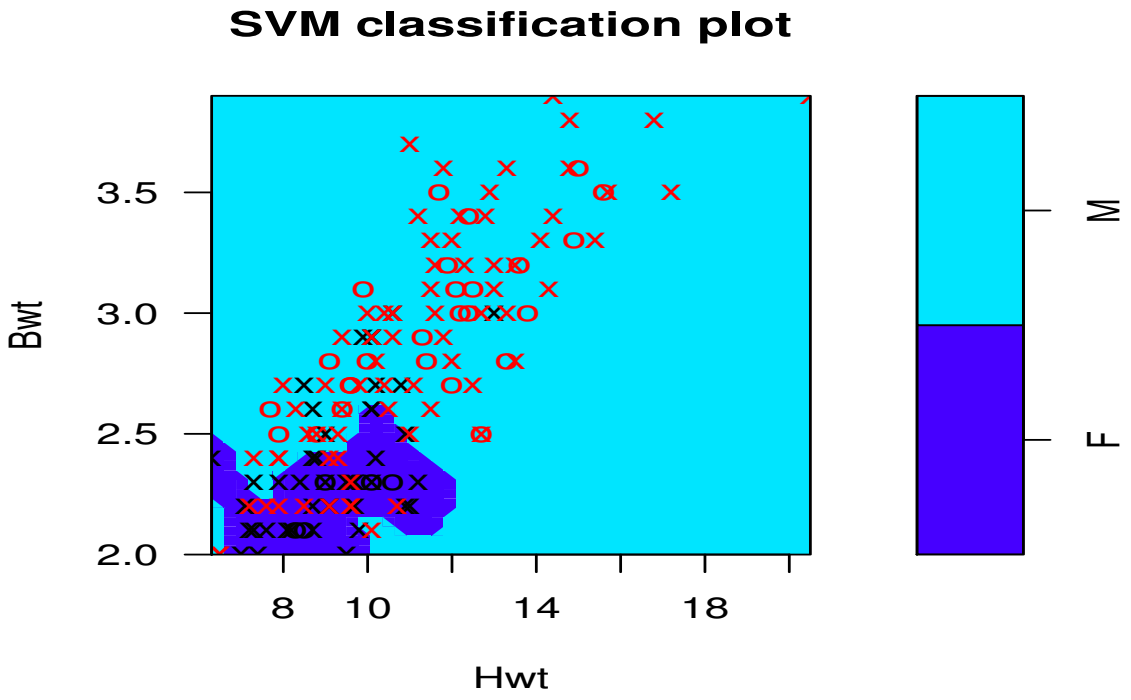
	F	M
F	15	7
M	32	90

Gaussian Kernel with $\gamma = 1$ and $C = 1$



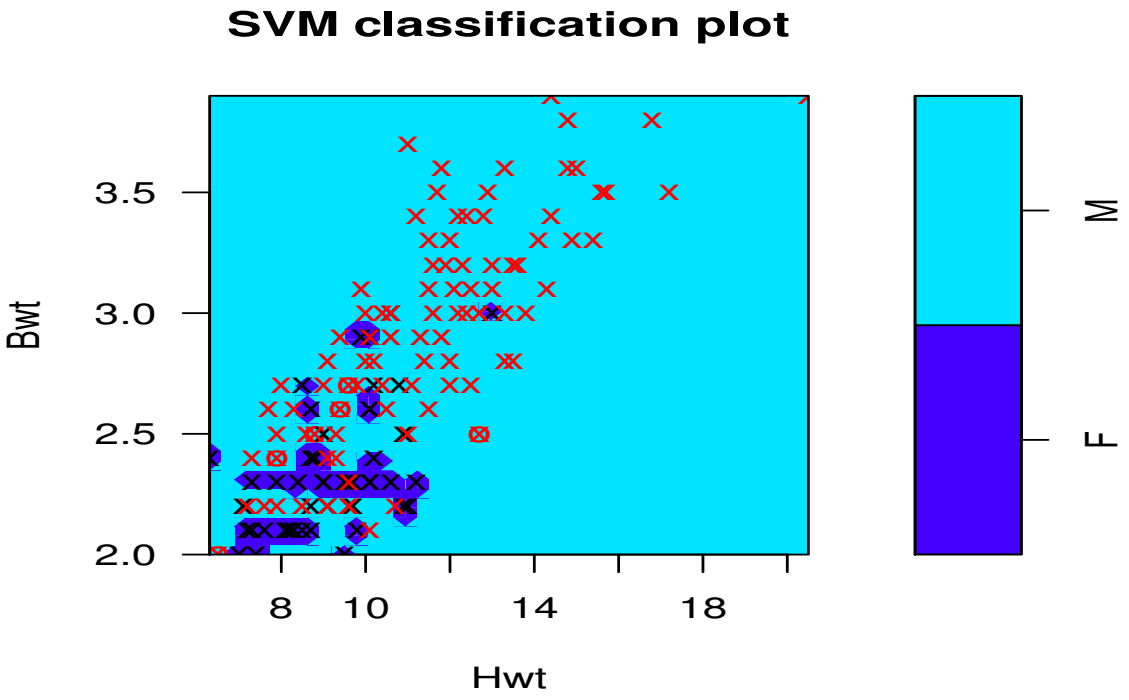
	F	M
F	32	14
M	15	83

Gaussian Kernel with $\gamma = 10$ and $C = 1$



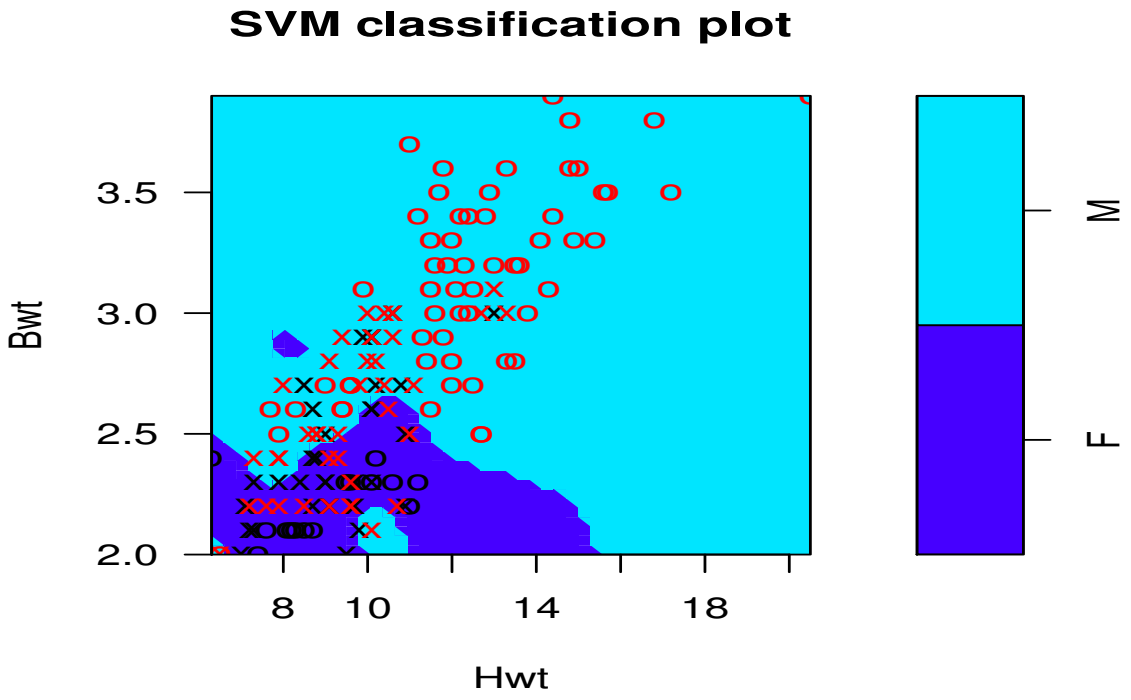
	F	M
F	33	8
M	14	89

Gaussian Kernel with $\gamma = 100$ and $C = 1$



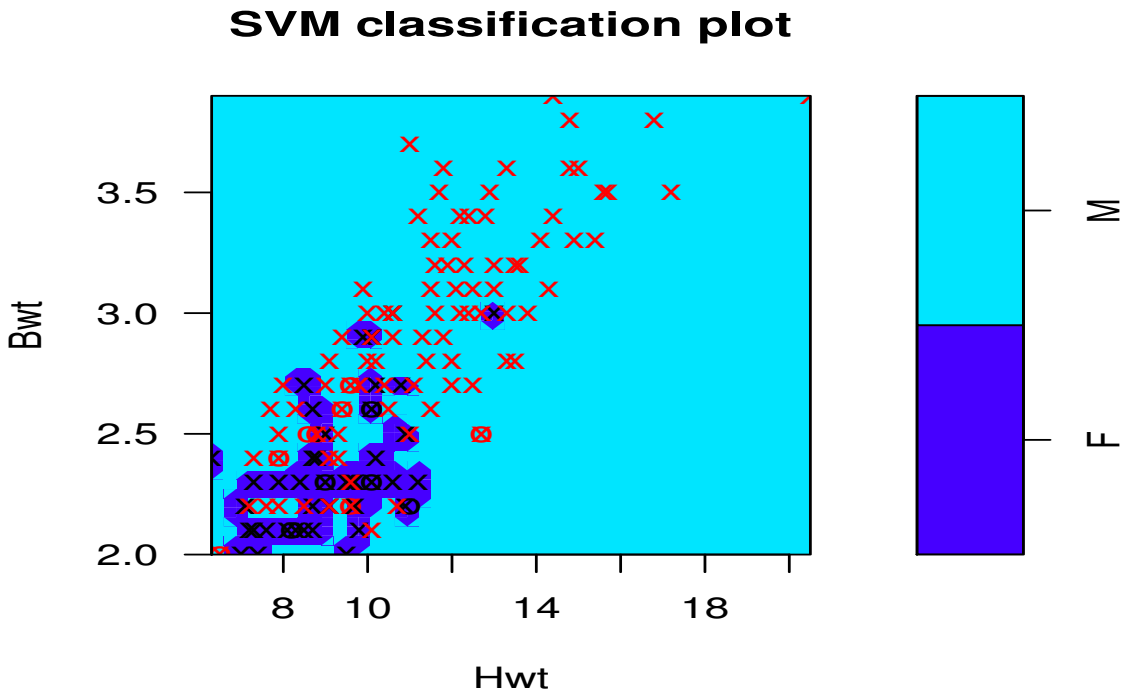
	F	M
F	39	2
M	8	95

Gaussian Kernel with $\gamma = 1$ and $C = 1000$



	F	M
F	35	13
M	12	84

Gaussian Kernel with $\gamma = 100$ and $C = 1000$



	F	M
F	45	2
M	2	95

Refernces

Thanks to Prof. Hsuan-Tien Lin
https://www.csie.ntu.edu.tw/ htlin/mooc/