Support Vector Machines

Find a linear hyperplane (decision boundary) that will separate the data.
Support Vector Machines

- One Possible Solution

Support Vector Machines

- Another possible solution
Support Vector Machines

- Other possible solutions

Which one is better? B1 or B2?
How do you define better?
Support Vector Machines

- Find hyperplane maximizes the margin => B1 is better than B2

\[ \frac{1}{\|w\|} \]

\[ f(\tilde{x}) = \begin{cases} 
1 & \text{if } \tilde{w} \cdot \tilde{x} + b \geq 1 \\
-1 & \text{if } \tilde{w} \cdot \tilde{x} + b \leq -1 
\end{cases} \]
Linear SVM

• Linear model:

\[ f(\tilde{x}) = \begin{cases} 
1 & \text{if } \tilde{w} \cdot \tilde{x} + b \geq 1 \\
-1 & \text{if } \tilde{w} \cdot \tilde{x} + b \leq -1 
\end{cases} \]

• Learning the model is equivalent to determining the values of \( \tilde{w} \) and \( b \)
  – How to find \( \tilde{w} \) and \( b \) from training data?

Learning Linear SVM

• Objective is to maximize: Margin = \( \frac{2}{||\tilde{w}||} \)
  – Which is equivalent to minimizing: \( L(\tilde{w}) = \frac{||\tilde{w}||^2}{2} \)
  – Subject to the following constraints:

\[ y_i = \begin{cases} 
1 & \text{if } \tilde{w} \cdot \tilde{x}_i + b \geq 1 \\
-1 & \text{if } \tilde{w} \cdot \tilde{x}_i + b \leq -1 
\end{cases} \]

or

\[ y_i (w \cdot x_i + b) \geq 1, \quad i = 1, 2, \ldots, N \]

• This is a constrained optimization problem
  – Solve it using Lagrange multiplier method
Example of Linear SVM

Learning Linear SVM

- Decision boundary depends only on support vectors
  - If you have data set with same support vectors, decision boundary will not change
  - How to classify using SVM once \( \mathbf{w} \) and \( b \) are found? Given a test record, \( x_i \)

\[
f(x_i) = \begin{cases} 
1 & \text{if } \mathbf{w} \cdot x_i + b \geq 1 \\
-1 & \text{if } \mathbf{w} \cdot x_i + b \leq -1 
\end{cases}
\]
Support Vector Machines

• What if the problem is not linearly separable?

Introduce slack variables

\[ L(w) = \frac{||\mathbf{w}||^2}{2} + C \sum_{i=1}^{N} \xi_i^k \]

Subject to:

\[ y_i = \begin{cases} 
1 & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b \geq 1 - \xi_i \\
-1 & \text{if } \mathbf{w} \cdot \mathbf{x}_i + b \leq -1 + \xi_i 
\end{cases} \]

If \( k \) is 1 or 2, this leads to similar objective function as linear SVM but with different constraints (see textbook)
Support Vector Machines

- Find the hyperplane that optimizes both factors

Nonlinear Support Vector Machines

- What if decision boundary is not linear?

\[ y(x_1, x_2) = \begin{cases} 
1 & \text{if } \sqrt{(x_1 - 0.5)^2 + (x_2 - 0.5)^2} > 0.2 \\
-1 & \text{otherwise}
\end{cases} \]
Nonlinear Support Vector Machines

- Transform data into higher dimensional space

\[ x_1^2 - x_1 + x_2^2 - x_2 = -0.46. \]
\[ \Phi : (x_1, x_2) \rightarrow (x_1^2, x_2^2, \sqrt{x_1}, \sqrt{x_2}, 1). \]
\[ w_4 x_1^2 + w_3 x_2^2 + w_2 \sqrt{x_1} + w_1 \sqrt{x_2} + w_0 = 0. \]

Decision boundary:
\[ \vec{w} \bullet \Phi(\vec{x}) + b = 0 \]

Learning Nonlinear SVM

- Optimization problem:

\[ \min_{\vec{w}} \frac{||\vec{w}||^2}{2} \text{ subject to } y_i (\vec{w} \cdot \Phi(x_i) + b) \geq 1, \forall \{(x_i, y_i)\} \]

- Which leads to the same set of equations (but involve \( \Phi(x) \) instead of \( x \))

\[ I_D = \sum_{i=1}^{n} \lambda_i - \frac{1}{2} \sum_{i,j} \lambda_i \lambda_j y_i y_j \Phi(x_i) \cdot \Phi(x_j) \quad w = \sum_{i} \lambda_i y_i \Phi(x_i) \]
\[ \lambda_i \{ y_i \left( \sum_{j} \lambda_j y_j \Phi(x_j) \cdot \Phi(x_i) + b \right) - 1 \} = 0, \]
\[ f(x) = \text{sign}(\vec{w} \cdot \Phi(x) + b) = \text{sign}(\sum_{i=1}^{n} \lambda_i y_i \Phi(x_i) \cdot \Phi(x) + b). \]
Learning NonLinear SVM

• Issues:
  – What type of mapping function $\Phi$ should be used?
  – How to do the computation in high dimensional space?
    ◆ Most computations involve dot product $\Phi(x_i) \cdot \Phi(x_j)$
    ◆ Curse of dimensionality?

Learning Nonlinear SVM

• Kernel Trick:
  – $\Phi(x_i) \cdot \Phi(x_j) = K(x_i, x_j)$
  – $K(x_i, x_j)$ is a kernel function (expressed in terms of the coordinates in the original space)
    ◆ Examples:
    $$K(x, y) = (x \cdot y + 1)^p$$
    $$K(x, y) = e^{-\|x - y\|^2/(2\sigma^2)}$$
    $$K(x, y) = \tanh(kx \cdot y - \delta)$$
Learning Nonlinear SVM

- Advantages of using kernel:
  - Don’t have to know the mapping function $\Phi$
  - Computing dot product $\Phi(x_i) \cdot \Phi(x_j)$ in the original space avoids curse of dimensionality

- Not all functions can be kernels
  - Must make sure there is a corresponding $\Phi$ in some high-dimensional space
  - Mercer’s theorem (see textbook)
Characteristics of SVM

- The learning problem is formulated as a convex optimization problem
  - Efficient algorithms are available to find the global minima
  - Many of the other methods use greedy approaches and find locally optimal solutions
  - High computational complexity for building the model

- Robust to noise
- Overfitting is handled by maximizing the margin of the decision boundary,
- SVM can handle irrelevant and redundant attributes better than many other techniques
- The user needs to provide the type of kernel function and cost function
- Difficult to handle missing values

- What about categorical variables?