SVM

- Said to start in 1979 with Vladimir Vapnik’s paper
- Major developments throughout 1990’s
- Elegant theory
  - Has good generalization properties
- Have been applied to diverse problems very successfully in the last 10-15 years
- One of the most important developments in pattern recognition in the last 10 years
Linear Discriminant Functions

- A discriminant function is linear if it can be written as
  \[ g(x) = w^T x + w_0 \]
  \[ g(x) > 0 \Rightarrow x \in \text{class 1} \]
  \[ g(x) < 0 \Rightarrow x \in \text{class 2} \]

- which separating hyperplane should we choose?

Linear Discriminant Functions

- Training data is just a subset of all possible data
- Suppose hyperplane is close to sample \( x_i \)
- If we see new sample close to sample \( i \), it is likely to be on the wrong side of the hyperplane

- Poor generalization (performance on unseen data)
**Linear Discriminant Functions**

- Hyperplane as far as possible from any sample

New samples close to the old samples will be classified correctly
- Good generalization

**SVM**

- Idea: maximize distance to the closest example

For the optimal hyperplane
- distance to the closest negative example = distance to the closest positive example
SVM: Linearly Separable Case

- SVM: maximize the *margin*

  ![Diagram](image)

  - *margin* is twice the absolute value of distance $b$ of the closest example to the separating hyperplane
  - Better generalization (performance on test data)
    - in practice
    - and in theory

SVM: Linearly Separable Case

- *Support vectors* are the samples closest to the separating hyperplane
  - they are the most difficult patterns to classify
  - Optimal hyperplane is completely defined by support vectors
    - of course, we do not know which samples are support vectors without finding the optimal hyperplane
SVM: Formula for the Margin

- $g(x) = w^T x + w_0$
- absolute distance between $x$ and the boundary $g(x) = 0$
  $$\frac{|w^T x + w_0|}{|w|}$$
- distance is unchanged for hyperplane $g_1(x) = \alpha g(x)$
  $$\frac{|\alpha w^T x + \alpha w_0|}{|\alpha w|} = \frac{|w^T x + w_0|}{|w|}$$
- Let $x_i$ be an example closest to the boundary. Set $|w^T x_i + w_0| = 1$
- Now the largest margin hyperplane is unique

SVM: Formula for the Margin

- For uniqueness, set $|w^T x_i + w_0| = 1$ for any example $x_i$ closest to the boundary
- now distance from closest sample $x_i$ to $g(x) = 0$ is
  $$\frac{|w^T x_i + w_0|}{|w|} = \frac{1}{|w|}$$
- Thus the margin is
  $$m = \frac{2}{|w|}$$
**SVM: Optimal Hyperplane**

- Maximize margin $m = \frac{2}{||w||}$
- subject to constraints
  
  \[
  \begin{cases}
  w^T x_i + w_0 \geq 1 & \text{if } x_i \text{ is positive example} \\
  w^T x_i + w_0 \leq -1 & \text{if } x_i \text{ is negative example}
  \end{cases}
  \]

- Let \[
  \begin{cases}
  z_i = 1 & \text{if } x_i \text{ is positive example} \\
  z_i = -1 & \text{if } x_i \text{ is negative example}
  \end{cases}
  \]

- Can convert our problem to
  
  \[
  \text{minimize } J(w) = \frac{1}{2}||w||^2
  \]
  
  constrained to $z_i(w^T x_i + w_0) \geq 1 \ \forall i$

- $J(w)$ is a quadratic function, thus there is a single global minimum

**SVM: Optimal Hyperplane**

- Use Kuhn-Tucker theorem to convert our problem to:

  \[
  \begin{align*}
  & \text{maximize } L_0(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j x_i^T x_j \\
  & \text{subject to } \alpha_i \geq 0 \ \forall i \ \text{and } \sum_{i=1}^{n} \alpha_i z_i = 0
  \end{align*}
  \]

- $\alpha = \{\alpha_1, \ldots, \alpha_n\}$ are new variables, one for each sample

- Can rewrite $L_D(\alpha)$ using $n$ by $n$ matrix $H$:

  \[
  L_0(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}^T \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_n \end{bmatrix}
  \]

  \[
  \text{where the value in the } i\text{th row and } j\text{th column of } H \text{ is } H_{ij} = z_i z_j x_i^T x_j
  \]
SVM: Optimal Hyperplane

- Use Kuhn-Tucker theorem to convert our problem to:

  \[
  \begin{align*}
  \text{maximize} & \quad L_0(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j x_i^t x_j \\
  \text{constrained to} & \quad \alpha_i \geq 0 \quad \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i z_i = 0
  \end{align*}
  \]

- \(\alpha =\{\alpha_1, \ldots, \alpha_n\}\) are new variables, one for each sample
- \(L_D(\alpha)\) can be optimized by quadratic programming
- \(L_D(\alpha)\) formulated in terms of \(\alpha\)
  - it depends on \(w\) and \(w_0\) indirectly

SVM: Optimal Hyperplane

- After finding the optimal \(\alpha =\{\alpha_1, \ldots, \alpha_n\}\)
  - For every sample \(i\), one of the following must hold
    - \(\alpha_i = 0\) (sample \(i\) is not a support vector)
    - \(\alpha_i \neq 0\) and \(z_i (\mathbf{w}^t \mathbf{x}_i + w_0 - 1) = 0\) (sample \(i\) is a support vector)
  - can find \(\mathbf{w}\) using \(\mathbf{w} = \sum_{i=1}^{n} \alpha_i z_i \mathbf{x}_i\)
  - can solve for \(w_0\) using any \(\alpha_i > 0\) and \(\alpha_i [z_i (\mathbf{w}^t \mathbf{x}_i + w_0) - 1] = 0\)
    \[w_0 = \frac{1}{z_i} - \mathbf{w}^t \mathbf{x}_i\]
  - Final discriminant function:
    \[g(x) = \left(\sum_{x_i \in S} \alpha_i z_i x_i \right)^t x + w_0\]
  - where \(S\) is the set of support vectors
    \[S = \{x_i \mid \alpha_i \neq 0\}\]
**SVM: Optimal Hyperplane**

maximize \( L_0(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j x_i^t x_j \)

consstrained to \( \alpha_i \geq 0 \quad \forall i \) and \( \sum_{i=1}^{n} \alpha_i z_i = 0 \)

- \( L_D(\alpha) \) depends on the number of samples, not on dimension of samples
- samples appear only through the dot products \( x_i^t x_j \)
- This will become important when looking for a **nonlinear** discriminant function, as we will see soon
- Code available on the web to optimize

**SVM: Non Separable Case**

- Data is most likely to be not linearly separable, but linear classifier may still be appropriate

Can apply SVM in non linearly separable case
- data should be “almost” linearly separable for good performance
SVM: Non Separable Case

- Use non-negative slack variables $\xi_1, \ldots, \xi_n$ (one for each sample)
- Change constraints from $z_i(w^T x_i + w_0) \geq 1 \ \forall i$ to $z_i(w^T x_i + w_0) \geq 1 - \xi_i \ \forall i$

$\xi_i$ is a measure of deviation from the ideal for sample $i$
- $\xi_i > 1$ sample $i$ is on the wrong side of the separating hyperplane
- $0 < \xi_i < 1$ sample $i$ is on the right side of separating hyperplane but within the region of maximum margin

SVM: Non Separable Case

- Would like to minimize

$$J(w, \xi_1, \ldots, \xi_n) = \frac{1}{2} \|w\|^2 + \beta \sum_{i=1}^{n} l(\xi_i > 0)$$

where $l(\xi_i > 0) = \begin{cases} 1 & \text{if } \xi_i > 0 \\ 0 & \text{if } \xi_i \leq 0 \end{cases}$

- constrained to $z_i(w^T x_i + w_0) \geq 1 - \xi_i$ and $\xi_i \geq 0 \ \forall i$

$\beta$ is a constant which measures relative weight of the first and second terms
- if $\beta$ is small, we allow a lot of samples not in ideal position
- if $\beta$ is large, we want to have very few samples not in ideal positon
SVM: Non Separable Case

\[ J(w, \xi_1, \ldots, \xi_n) = \frac{1}{2} ||w||^2 + \beta \sum_{i=1}^{n} I(\xi_i > 0) \]

# of examples not in ideal location

\[ \sum \sum \text{not in ideal location} \]

large \( \beta \), few samples not in ideal position

small \( \beta \), a lot of samples not in ideal position

Unfortunately this minimization problem is NP-hard due to discontinuity of functions \( I(\xi_i) \)

\[ J(w, \xi_1, \ldots, \xi_n) = \frac{1}{2} ||w||^2 + \beta \sum_{i=1}^{n} I(\xi_i > 0) \]

# of examples not in ideal location

- where \( I(\xi_i > 0) = \begin{cases} 1 & \text{if } \xi_i > 0 \\ 0 & \text{if } \xi_i \leq 0 \end{cases} \)

- constrained to \( z_i (w^T x_i + w_0) \geq 1 - \xi_i \) and \( \xi_i \geq 0 \ \forall i \)
**SVM: Non Separable Case**

- Instead we minimize
  \[
  J(w, \xi_1, \ldots, \xi_n) = \frac{1}{2} \|w\|^2 + \beta \sum_{i=1}^{n} \xi_i
  \]
  a measure of
  # of misclassified examples

- constrained to
  \[
  \begin{cases}
  z_i(w^\top x_i + w_0) \geq 1 - \xi_i & \forall i \\
  \xi_i \geq 0 & \forall i
  \end{cases}
  \]

- Can use Kuhn-Tucker theorem to converted to

  maximize
  \[
  L_0(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j x_i^\top x_j
  \]

  constrained to
  \[
  0 \leq \alpha_i \leq \beta \quad \forall i \quad \text{and} \sum_{i=1}^{n} \alpha_i z_i = 0
  \]

- find \( w \) using
  \[
  w = \sum_{i=1}^{n} \alpha_i z_i x_i
  \]

- solve for \( w_0 \) using any \( 0 < \alpha_i < \beta \) and
  \[
  \alpha_i [z_i (w^\top x_i + w_0) - 1] = 0
  \]

**Non Linear Mapping**

- Cover’s theorem:
  - “pattern-classification problem cast in a high dimensional space non-linearly is more likely to be linearly separable than in a low-dimensional space”

- One dimensional space, not linearly separable

- Lift to two dimensional space with
  \[
  \phi(x) = (x, x^2)
  \]

- Diagram showing one-dimensional data not separable, lifted to two-dimensional space.
**Non Linear Mapping**

- To solve a non linear classification problem with a linear classifier
  1. Project data $x$ to high dimension using function $\varphi(x)$
  2. Find a linear discriminant function for transformed data $\varphi(x)$
  3. Final nonlinear discriminant function is $g(x) = w^t \varphi(x) + w_0$

$\varphi(x) = (x, x^2)$

- In 2D, discriminant function is linear
  $$g\left(\begin{bmatrix} x^{(1)} \\ x^{(2)} \end{bmatrix}\right) = [w_1, w_2] \begin{bmatrix} x^{(1)} \\ x^{(2)} \end{bmatrix} + w_0$$

- In 1D, discriminant function is not linear
  $$g(x) = w_1 x + w_2 x^2 + w_0$$

**Non Linear Mapping: Another Example**
Non Linear SVM

- Can use any linear classifier after lifting data into a higher dimensional space. However we will have to deal with the “curse of dimensionality”
  1. poor generalization to test data
  2. computationally expensive

- SVM avoids the “curse of dimensionality” problems by
  1. enforcing largest margin permits good generalization
     - It can be shown that generalization in SVM is a function of the margin, independent of the dimensionality
  2. computation in the higher dimensional case is performed only implicitly through the use of kernel functions

Non Linear SVM: Kernels

- Recall SVM optimization
  \[
  \max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j
  \]

  Note this optimization depends on samples \( x_i \) only through the dot product \( x_i^T x_j \)

- If we lift \( x_i \) to high dimension using \( \phi(x) \), need to compute high dimensional product \( \phi(x_i)^T \phi(x_j) \)
  \[
  \max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j)
  \]

- Idea: find kernel function \( K(x_i, x_j) \) s.t.
  \[
  K(x_i, x_j) = \phi(x_i)^T \phi(x_j)
  \]
Non Linear SVM: Kernels

maximize \( L_0(\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(x_i, x_j) \%

- Then we only need to compute \( K(x_i, x_j) \) instead of \( \phi(x_i)^T \phi(x_j) \%
  - “kernel trick”: do not need to perform operations in high dimensional space explicitly

Non Linear SVM: Kernels

- Suppose we have 2 features and \( K(x, y) = (x^T y)^2 \%
- Which mapping \( \phi(x) \) does it correspond to?
  \[
  K(x, y) = (x^T y)^2 = \left( \begin{bmatrix} x^{(1)} & x^{(2)} \\ y^{(1)} & y^{(2)} \end{bmatrix} \right)^2 = (x^{(1)} y^{(1)} + x^{(2)} y^{(2)})^2 \\
  = (x^{(1)} y^{(1)})^2 + 2(x^{(1)} y^{(1)})(x^{(2)} y^{(2)}) + (x^{(2)} y^{(2)})^2 \\
  = \left( \begin{bmatrix} x^{(1)} \\ y^{(1)} \end{bmatrix} \right)^2 \sqrt{2} x^{(1)} x^{(2)} \left( \begin{bmatrix} y^{(1)} \\ y^{(2)} \end{bmatrix} \right)^2 \\
  = \left( \begin{bmatrix} x^{(1)} \\ \sqrt{2} x^{(1)} x^{(2)} \\ \sqrt{2} x^{(2)} \end{bmatrix} \right)^T \%
  
- Thus \( \phi(x) = \left( \begin{bmatrix} x^{(1)} \\ \sqrt{2} x^{(1)} x^{(2)} \\ x^{(2)} \end{bmatrix} \right) \%
Non Linear SVM: Kernels

- How to choose kernel function \( K(x_i, x_j) \)?
  - \( K(x_i, x_j) \) should correspond to product \( \varphi(x_i)'\varphi(x_j) \) in a higher dimensional space
  - Mercer’s condition tells us which kernel function can be expressed as dot product of two vectors
  - Kernel's not satisfying Mercer’s condition can be sometimes used, but no geometrical interpretation

- Some common choices (satisfying Mercer’s condition):
  - Polynomial kernel \( K(x_i, x_j) = (x_i'x_j + 1)^p \)
  - Gaussian radial Basis kernel (data is lifted in infinite dimension)
    \[
    K(x_i, x_j) = \exp\left(-\frac{1}{2\sigma^2}||x_i - x_j||^2\right)
    \]

Non Linear SVM

- search for separating hyperplane in high dimension
  \[
  w\varphi(x) + w_0 = 0
  \]

- Choose \( \varphi(x) \) so that the first (“0”th) dimension is the augmented dimension with feature value fixed to 1
  \[
  \varphi(x) = \begin{bmatrix} 1 & x^{(1)} & x^{(2)} & x^{(1)}x^{(2)} \end{bmatrix}'
  \]

- Threshold parameter \( w_0 \) gets folded into the weight vector \( w \)
  \[
  \begin{bmatrix} w_0 \ v \end{bmatrix} \begin{bmatrix} 1 \\ \varphi(x) \end{bmatrix} = 0
  \]
**Non Linear SVM**

- Will not use notation \( a = [w_0 \ w] \), we'll use old notation \( w \) and seek hyperplane through the origin \( w \phi(x) = 0 \)

- If the first component of \( \phi(x) \) is not 1, the above is equivalent to saying that the hyperplane has to go through the origin in high dimension
  - removes only one degree of freedom
  - But we have introduced many new degrees when we lifted the data in high dimension

**Non Linear SVM Recipe**

- Start with data \( x_1, \ldots, x_n \) which lives in feature space of dimension \( d \)
- Choose kernel \( K(x_i, x_j) \) or function \( \phi(x_i) \) which takes sample \( x_i \) to a higher dimensional space
- Find the largest margin linear discriminant function in the higher dimensional space by using quadratic programming package to solve:

\[
\text{maximize} \quad L_0(\alpha) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j z_i z_j K(x_i, x_j)
\]

constrained to \( 0 \leq \alpha_i \leq \beta \quad \forall i \quad \text{and} \quad \sum_{i=1}^{n} \alpha_i z_i = 0 \)
Non Linear SVM Recipe

- Weight vector $\mathbf{w}$ in the high dimensional space:
  \[
  \mathbf{w} = \sum_{x_i \in \mathbf{S}} \alpha_i z_i \varphi(x_i)
  \]
  where $\mathbf{S}$ is the set of support vectors $\mathbf{S} = \{x_i \mid \alpha_i \neq 0\}$
- Linear discriminant function of largest margin in the high dimensional space:
  \[
  g(\varphi(x)) = \mathbf{w}'\varphi(x) = \left(\sum_{x_i \in \mathbf{S}} \alpha_i z_i \varphi(x_i)\right)' \varphi(x)
  \]
- Non linear discriminant function in the original space:
  \[
  g(x) = \left(\sum_{x_i \in \mathbf{S}} \alpha_i z_i \varphi(x_i)\right)' \varphi(x) = \sum_{x_i \in \mathbf{S}} \alpha_i z_i \varphi'(x_i) \varphi(x) = \sum_{x_i \in \mathbf{S}} \alpha_i z_i K(x_i, x)
  \]
  decide class 1 if $g(x) > 0$, otherwise decide class 2

Non Linear SVM

- Nonlinear discriminant function
  \[
  g(x) = \sum_{x_i \in \mathbf{S}} \alpha_i z_i K(x_i, x)
  \]
  \[
  g(x) = \sum \quad \text{weight of support vector } x_i \quad \text{"inverse distance" from } x \text{ to support vector } x_i
  \]

  most important training samples, i.e. support vectors
  \[
  K(x_i, x) = \exp\left(-\frac{1}{2\sigma^2} \|x_i - x\|^2\right)
  \]
**SVM Example: XOR Problem**

- Class 1: \(x_1 = [1, -1], x_2 = [-1, 1]\)
- Class 2: \(x_3 = [1, 1], x_4 = [-1, -1]\)
- Use polynomial kernel of degree 2:
  - \(K(x_i, x_j) = (x_i^t x_j + 1)^2\)
  - This kernel corresponds to mapping \(\phi(x) = [1, \sqrt{2}x^{(t)}, \sqrt{2}x^{(o)}, \sqrt{2}x^{(t)}x^{(o)}, (x^{(t)})^2, (x^{(o)})^2]\)
- Need to maximize
  \[L_0(\alpha) = \sum_{i=1}^{4} \alpha_i - \frac{1}{2} \sum_{i=1}^{4} \sum_{j=1}^{4} \alpha_i \alpha_j z_i z_j (x_i^t x_j + 1)^2\]
  constrained to \(0 \leq \alpha_i \quad \forall i \quad \text{and} \quad \alpha_i + \alpha_2 - \alpha_3 - \alpha_4 = 0\)

**SVM Example: XOR Problem**

- Can rewrite \(L_0(\alpha) = \sum_{i=1}^{4} \alpha_i - \frac{1}{2} \alpha^t H \alpha\)
  - where \(\alpha = [\alpha_1, \alpha_2, \alpha_3, \alpha_4]^t\) and \(H = \begin{bmatrix} 9 & 1 & -1 & -1 \\ 1 & 9 & -1 & -1 \\ -1 & -1 & 9 & 1 \\ -1 & -1 & 1 & 9 \end{bmatrix}\)
- Take derivative with respect to \(\alpha\) and set it to \(0\)
  \[
  \frac{d}{d\alpha} L_0(\alpha) = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 9 & 1 & -1 & -1 \\ 1 & 9 & -1 & -1 \\ -1 & -1 & 9 & 1 \\ -1 & -1 & 1 & 9 \end{bmatrix} \alpha = 0
  \]
- Solution to the above is \(\alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = 0.25\)
  - satisfies the constraints \(\forall i, \ 0 \leq \alpha_i \quad \text{and} \quad \alpha_i + \alpha_2 - \alpha_3 - \alpha_4 = 0\)
  - all samples are support vectors
**SVM Example: XOR Problem**

\[ \varphi(x) = \begin{bmatrix} 1 & \sqrt{2}x^{(1)} & \sqrt{2}x^{(2)} & \sqrt{2}x^{(1)}x^{(2)} & (x^{(1)})^2 & (x^{(2)})^2 \end{bmatrix} \]

- Weight vector \( w \) is:
  \[
w = \sum_{i=1}^{d} \alpha_i z_i \varphi(x_i) = 0.25(\varphi(x_1) + \varphi(x_2) - \varphi(x_3) - \varphi(x_4))
  = [0 \ 0 \ 0 \ -\sqrt{2} \ 0 \ 0]
\]

- Thus the nonlinear discriminant function is:
  \[g(x) = w\varphi(x) = \sum_{i=1}^{d} w_i \varphi_i(x) = -\sqrt{2}(\sqrt{2}x^{(1)}x^{(2)}) = -2x^{(1)}x^{(2)}\]
Degree 3 Polynomial Kernel

- In linearly separable case (on the left), decision boundary is roughly linear, indicating that dimensionality is controlled
- Nonseparable case (on the right) is handled by a polynomial of degree 3

SVM Summary

Advantages:
- Based on nice theory
- excellent generalization properties
- objective function has no local minima
- can be used to find non linear discriminant functions
- Complexity of the classifier is characterized by the number of support vectors rather than the dimensionality of the transformed space

Disadvantages:
- tends to be slower than other methods
- quadratic programming is computationally expensive
- Not clear how to choose the Kernel