Support Vector Machines

Andrew Nobel

April, 2021

Overview: Support Vector Machines (SVM)

- Simplest case: linear classification rule
- Generalizes to non-linear rules through feature maps and kernels
- Good off-the-shelf method for high dimensional data, widely used
- Begins with geometry rather than a statistical model
- Close connections with convex programming
- Early bridge between machine learning and optimization

Notational switch: Code two-valued response Y as -1 or $+1$

Linear Classification Rules

Linear Classification Rules

Setting: Labeled pair (x, y) with predictor $x \in \mathbb{R}^p$ and class $y \in \{-1, +1\}$

Definition: Given $w \in \mathbb{R}^p$ and $b \in \mathbb{R}$ *linear classification rule* has form

$$\phi(x) \;=\; \text{sign}(x^t w - b) \;=\; \begin{cases} +1 & \text{if } w^t x \geq b \\ -1 & \text{if } w^t x < b \end{cases}$$

Decision boundary of ϕ equal to **hyperplane** $H = \{x : w^t x = b\}$

Distance to Decision Boundary

Consider rule $\phi = \text{sign}(x^t w - b)$ with decision boundary $H = \{x : w^t x = b\}$

Given pair (x, y) ask two questions

- Correctness: Is x on the right side of decision boundary H?
- Confidence: How far is x from the decision boundary H?

Fact: The signed distance from x to the decision boundary H is given by

$$\frac{x^t w - b}{||w||}$$

Margin

Definition: The *margin* of linear rule $\phi = \text{sign}(x^t w - b)$ at (x, y) is

$$m_\phi(x, y) \; = \; y \left(\frac{x^t w - b}{||w||} \right)$$

Idea: Margin assesses the fit of ϕ at pair (x, y)

- $m_\phi(x, y) > 0$ iff $\phi(x) = y$ iff x on correct side of H
- $m_\phi(x, y) < 0$ iff $\phi(x) \neq y$ iff x on wrong side of H
- $|m_\phi(x, y)| =$ distance from x to H

Maximum Margin Classifiers and Linearly Separability

General goal: In fitting a linear rule to data, we would like the margins of all the data points to be large and positive (if possible)

Definition: A dataset $D_n = (x_1, y_1), \ldots, (x_n, y_n)$ is *linearly separable* if there is a hyperplane H separating $\{x_i : y_i = 1\}$ and $\{x_i : y_i = -1\}$

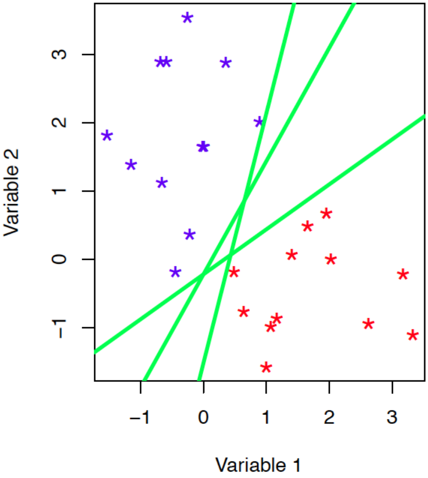
We will consider two cases

1. Data is linearly separable $\Rightarrow$ max margin classifier
2. Data is not linearly separable $\Rightarrow$ soft margin classifier

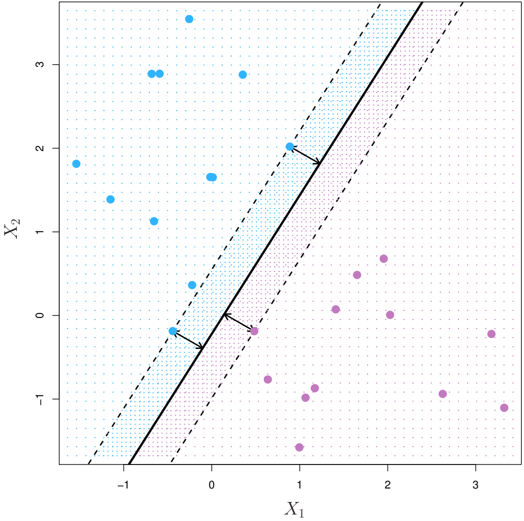
Maximum Margin Classifiers (Support Vector Machine)

Linearly Separable Case

Linearly Separable Data: Multiple Hyperplanes



Max Margin Classifier (from ISL)



Maximizing the Minimum Margin

Max Margin Classifier: Given linearly separable data D_n, find w and b to maximize the minimum margin of $\phi(x) = \text{sign}(x^t w - b)$. Program is

$$\max_{w,b} \Gamma(w,b) \quad \text{where} \quad \Gamma(w,b) = \min_{1 \le i \le n} y_i \left(\frac{x_i^t w - b}{||w||} \right) \qquad (\star)$$

Note that this program is not convex.

Fact: Non-convex program $(\star)$ is equivalent to the convex program

$$p^* = \min_{w,b} \frac{1}{2} ||w||^2 \ \text{subject to} \ y_i(x_i^t w - b) \ge 1 \ \text{for} \ i = 1, \ldots, n$$

Finding p^* is called the *primal problem*

Solving the Problem of Maximizing the Minimum Margin

Approach: Solve primal problem using *Lagrangian function* and *duality*

Definition: The *Lagrangian* $L : \mathbb{R}^p \times \mathbb{R} \times \mathbb{R}^n_+$, with $\mathbb{R}_+ = [0, \infty)$, for the max margin classifier problem is

$$L(w, b, \lambda) \ := \ \frac{1}{2}||w||^2 \ - \sum_{i=1}^{n} \lambda_i \left\{ y_i(w^t x_i - b) - 1 \right\}$$

Note: Lagrangian combines objective and constraints into a single function. New variables λ_i called *Lagrange multipliers*.

Min-Max Formulation and Dual Problem

1. The Lagrangian turns primal problem into min-max problem. Note that

$$\max_{\lambda \geq 0} L(w, b, \lambda) = \begin{cases} ||w||^2/2 & \text{if constraints satisfied} \\ +\infty & \text{otherwise} \end{cases}$$

Therefore the primal problem can be written in **min-max** form

$$p^* = \min_{w,b} \max_{\lambda \geq 0} L(w, b, \lambda)$$

2. Changing the order of the min and the max yields the **dual problem**

$$d^* = \max_{\lambda \geq 0} \min_{w,b} L(w, b, \lambda)$$

The Dual Problem

Note: The dual problem can be written in the equivalent form

$$d^* = \max_{\lambda \geq 0} \tilde{L}(\lambda) \quad \text{where} \quad \tilde{L}(\lambda) = \min_{w,b} L(w, b, \lambda)$$

- The *dual function* $\tilde{L}(\lambda)$ is concave and has a global maximum, so the dual problem has a solution.
- In general, $d^* \leq p^*$. Difference $p^* - d^* \geq 0$ called *duality gap*
- In this case, can show that $d^* = p^*$, so solution of the dual problem gives solution of the primary problem

Solving the Dual Problem

Step 1: Fix $\lambda \geq 0$ and minimize $L(w, b, \lambda)$ over w, b. Differentiation gives

$$w = \sum_{i=1}^{n} \lambda_i \, y_i \, x_i \quad \text{and} \quad \sum_{i=1}^{n} \lambda_i \, y_i = 0$$

Substituting these equations into $L(w, b, \lambda)$ yields quadratic *dual function*

$$\tilde{L}(\lambda) \; = \; \sum_{i=1}^{n} \lambda_i \; - \; \frac{1}{2} \sum_{i,j=1}^{n} \lambda_i \, \lambda_j \, y_i \, y_j \, \langle x_i, x_j \rangle$$

Step 2: Solve concave dual problem using quadratic programming

$$\max \tilde{L}(\lambda) \quad \text{s.t.} \quad \sum_{i=1}^{n} \lambda_i \, y_i = 0 \;\; \text{and} \;\; \lambda_1, \ldots, \lambda_n \geq 0$$

Solving the Problem of Maximizing the Minimum Margin

Step 3: Combine solution λ of dual problem and optimality conditions to get desired values of w and b

$$w = \sum_{i=1}^{n} \lambda_i \, y_i \, x_i \qquad b = \frac{1}{2} \left[\min_{i:y_i=1} x_i^t w + \max_{i:y_i=-1} x_i^t w \right]$$

Upshot: Maximum margin classification rule $\hat{\phi}_n^{\text{SVM}}(x) = \text{sign}(h(x))$ where

$$h(x) \;=\; x^t w - b \;=\; \sum_{i=1}^{n} \lambda_i \, y_i \, \langle x_i, x \rangle - b$$

Inner Products

Note: Observed feature vectors x_i affect $\hat{\phi}_n^{\text{SVM}}$ only through inner products

- Dual $\tilde{L}(\lambda)$ depends on x_i's only through inner products $\langle x_i, x_j \rangle$
- Function $h(x)$ depends on x_i's only through inner products $\langle x_i, x \rangle$

KKT Conditions and Support Vectors

Fact: For each i, optimal w, b, and λ are such that $\lambda_i(y_i\, h(x_i) - 1) = 0$.
This implies that $\lambda_i = 0$ or $y_i\, h(x_i) = 1$

Let $S = \{i : \lambda_i > 0\}$. Note that

1. $h(x) = \sum_{i \in S} \lambda_i\, y_i\, \langle x_i, x \rangle - b$
2. If $i \in S$ then $y_i h(x_i) = 1$ so x_i lies on margin for class y_i

Definition: Training vectors x_i with $i \in S$ called *support vectors*

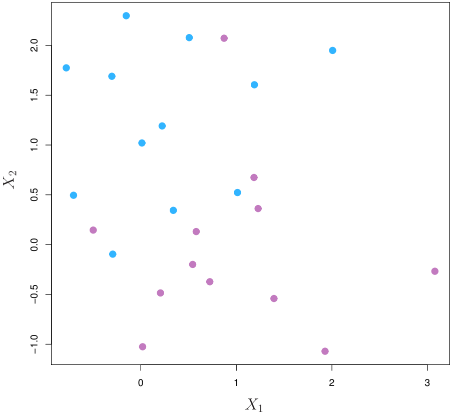
- Changing a support vector with other data fixed would change the decision boundary

Soft Margin Classifiers (Support Vector Machine)

General Case

Extending SVM to Non-Separable Case

Most data sets *not* linearly separable: no hyperplane can separate ± 1's



Question: How to extend maximum margin classifiers to this setting?

SVM: Non-Separable Case

Idea: Reformulate primal problem. For fixed $C > 0$ solve convex program

$$\min_{w,b,\xi} \left\{ \frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \xi_i \right\}$$

$$\text{s.t. } \; y_i(x_i^t w - b) \geq 1 - \xi_i \; \text{ and } \; \xi_i \geq 0$$

- $\xi_1, \ldots, \xi_n$ are called *slack variables*
- ξ_i measures violation of hard constraint $y_i(x_i^t w - b) \geq 1$
- $||w||^2$ small means larger margin
- C controls tradeoff between margin size and total slack

Slack Variables and Margins

Consider linear function $h(x) = x^t w - b$, associated rule $\phi(x) = \text{sign}(h(x))$

- Separating hyperplane $H = \{x : h(x) = 0\}$
- Target half spaces $H^+ = \{x : h(x) \geq 1\}$ and $H^- = \{x : h(x) \leq -1\}$

Consider data point (x_i, y_i) with fit $u_i = y_i h(x_i)$. Three cases

1. If $u_i \geq 1$ then $\phi(x_i) = y_i$ and $x_i \in H^{y_i}$, slack $\xi_i = 0$
2. If $0 \leq u_i < 1$ then $\phi(x_i) = y_i$ but $x_i \notin H^{y_i}$, slack $\xi_i = 1 - m_i \in (0, 1]$
3. If $u_i < 0$ then $\phi(x_i) \neq y_i$ and $x_i \notin H^{y_i}$, slack $\xi_i = 1 - m_i > 1$

Soft Margin Classifier

Upshot: Dual approach similar to separable case yields soft margin classification rule $\hat{\phi}_n^{\text{SVM}}(x) = \text{sign}(h(x))$ where

$$h(x) \;=\; x^t w - b \;=\; \sum_{i \in S} \lambda_i \, y_i \, \langle x_i, x \rangle - b$$

- Optimal λ from dual optimization; support set $S = \{i : \lambda_i > 0\}$

$$w = \sum_{i \in S} \lambda_i \, y_i \, x_i \qquad b = \text{function of } \lambda \text{ and data}$$

- Rule $\hat{\phi}_n^{\text{SVM}}$ depends on vectors x_i, x only through inner products

Effect of Parameter C (from ISL)

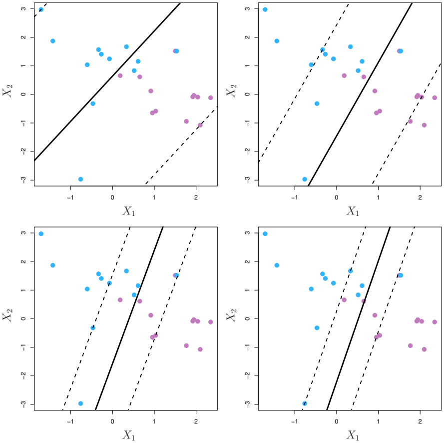


Figure: SVM with small C (the top left) to large C (bottom right). Data non-separable.

Revisiting the Soft Margin Classifier

Recall: Soft margin classifier has primal problem

$$\min_{w,b,\xi} \left\{ \frac{1}{2}||w||^2 + C \sum_{i=1}^{n} \xi_i \right\} \quad \text{s.t.} \quad y_i(x_i^t w - b) \geq 1 - \xi_i \ \text{ and } \ \xi_i \geq 0$$

Equivalent Problem: Primal problem can be written in form

$$\min_{w,b} \left\{ \sum_{i=1}^{n} \ell_h(w^t x_i - b, y_i) + \lambda ||w||^2 \right\}$$

- $\ell_h(s,t) = [1 - st]_+ = \max(1 - st, 0)$ “hinge loss ”
- $\ell_h(s,t)$ convex in s when t fixed, so $\ell_h(w^t x - b, y)$ convex in w, b
- Equivalent problem is a convex program

Revisiting Soft Margin, cont.

Note similarity between hinge-loss problem and ridge regression

$$\min_{\beta} \left\{ \sum_{i=1}^{n} \ell(\beta^t x_i, y_i) + \lambda ||\beta||^2 \right\} \text{ with } \ell(s,t) = (s-t)^2$$

Sparse SVM: Connection with Ridge suggests SVM with ℓ_1-penalty

$$\min_{w,b} \left\{ \sum_{i=1}^{n} \ell_h(w^t x_i - b, y_i) + \lambda ||w||_1 \right\}$$

- The ℓ_1-penalty sets many coefficients of w to zero
- Interpretation: selecting important features
- Similar idea can be applied to logistic regression

Support Vector Machines: Non-Linear Case

Nonlinear SVM: Background

Note: Inner product $\langle x, x' \rangle$ is signed measure of similarity between x and x'

- $\langle x, x' \rangle = ||x||\,||x'||$ if x, x' point in same direction
- $\langle x, x' \rangle = 0$ if x, x' are orthogonal
- $\langle x, x' \rangle = -||x||\,||x'||$ if x, x' point in opposite directions

Goal: Enhance and expand applicability of standard SVM

- Map predictors x to new feature space via nonlinear transformation
- Classify data using similarity between transformed features
- In many cases new features space is high dimensional

Direct Approach to Nonlinear SVM: Feature Maps

Given: Data $(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \{\pm 1\}$

- Define *feature map* $\gamma : \mathcal{X} \to \mathbb{R}^d$ taking predictors to HD features
- Apply SVM to observations $(\gamma(x_1), y_1), \ldots, (\gamma(x_n), y_n)$
- SVM classifier is sign of $h(x) = \sum_{i=1}^n \lambda_i \, y_i \, \langle \gamma(x_i), \gamma(x) \rangle - b$

Example 1: Two-way interactions (polynomials of degree two)

- Predictor space $\mathcal{X} = \mathbb{R}^p$
- Define feature map $\gamma : \mathcal{X} \to \mathbb{R}^d$ by $\gamma(x) = (x_i \, x_j)_{1 \leq i,j \leq p}$
- Computing $\langle \gamma(x), \gamma(x') \rangle$ requires $d = p^2$ operations.

Feature Maps, cont.

Example 2: Bag-of-words representation of documents

- Predictor space $\mathcal{X} = \{\text{English language documents}\}$
- Fix set of words (vocabulary) V of interest
- Define map $\gamma : \mathcal{X} \to \{0, 1, 2, \ldots\}^V$ from docs to word counts by

$$\gamma(x) = \text{\# occurrences of each word } v \in V \text{ in document } x$$

- Computing $\langle \gamma(x), \gamma(x') \rangle$ requires $d = |V|$ operations

Note: Bag-of-words representation common in natural language processing

Nonlinear SVM via Kernels

Basic idea: Replace inner product $\langle \cdot, \cdot \rangle$ by kernel function $K : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ where $K(u, v)$ measures the similarity between u and v. Key assumptions

- $K(u, v) = K(v, u)$
- For all $u_1, \ldots, u_n \in \mathcal{X}$ the matrix $\{K(u_i, u_j) : 1 \leq i, j \leq n\} \geq 0$

Kernel classifier: SVM with kernel K

- Solve Lagrange dual problem, replacing $\langle x_i, x_j \rangle$ by $K(x_i, x_j)$
- Optimal rule rule $\phi(x) = \text{sign}(h(x))$ where

$$h(x) \;=\; \sum_{i \in S} \lambda_i \, y_i \, K(x_i, x) - b$$

Examples of Kernels

1. Feature map. Given $\gamma : \mathcal{X} \to \mathbb{R}^d$ define kernel $K(u, v) = \langle \gamma(u), \gamma(v) \rangle$

2. Polynomial. For $\mathcal{X} = \mathbb{R}^d$ let $K(u, v) = (1 + \langle u, v \rangle)^d$

3. Radial basis. For $\mathcal{X} = \mathbb{R}^d$ let $K(u, v) = \exp\{-c||u - v||^2\}$

4. Neural network. For $\mathcal{X} = \mathbb{R}^d$ let $K(u, v) = \tanh(a\langle u, v \rangle + b)$

Fact: Under appropriate conditions kernel $K(u, v) = \langle \gamma(u), \gamma(v) \rangle$ for a suitable feature map $\gamma : \mathcal{X} \to \mathcal{S}$

- Feature space $\mathcal{S}$ may be infinite dimensional
- Computing $K(u, v)$ may be faster than computing $\langle \gamma(u), \gamma(v) \rangle$