SVMs II

CMSC 422

SOHEIL FEIZI

sfeizi@cs.umd.edu

Slides adapted from MARINE CARPUAT

Today’s topics

- SVMs

- Final project presentations start on Thursday

- Course evals

https://www.CourseEvalUM.umd.edu

Support Vector Machine (SVM)

A hyperplane based linear classifier defined by $\mathbf{w}$ and b

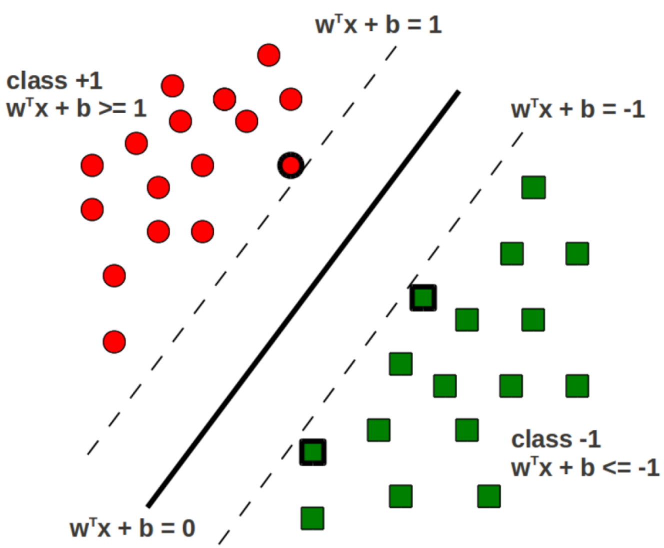
Prediction rule: $y = sign(\mathbf{w}^T\mathbf{x} + b)$

Given: Training data $\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$

Goal: Learn $\mathbf{w}$ and b that achieve the maximum margin

Characterizing the margin

Let's assume the entire training data is correctly classified by (**w**,b) that achieve the maximum margin

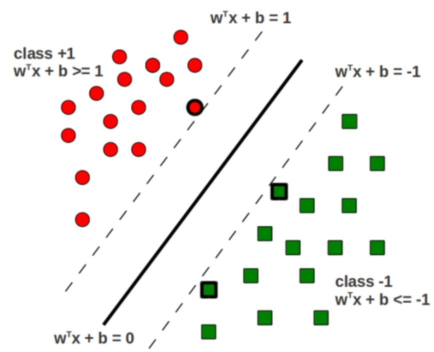


- Assume the hyperplane is such that
  - $\mathbf{w}^T\mathbf{x}_n + b \geq 1$ for $y_n = +1$
  - $\mathbf{w}^T\mathbf{x}_n + b \leq -1$ for $y_n = -1$
  - Equivalently, $y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1$
    $\Rightarrow \min_{1 \leq n \leq N} |\mathbf{w}^T\mathbf{x}_n + b| = 1$
  - The hyperplane's margin:

$$\gamma = \min_{1 \leq n \leq N} \frac{|\mathbf{w}^T\mathbf{x}_n + b|}{||\mathbf{w}||} = \frac{1}{||\mathbf{w}||}$$

The Optimization Problem

We want to maximize the margin $\gamma = \frac{1}{||\mathbf{w}||}$



Maximizing the margin γ = minimizing $||\mathbf{w}||$ (the norm)

Our optimization problem would be:

$$\text{Minimize} \quad f(\mathbf{w}, b) = \frac{||\mathbf{w}||^2}{2}$$

$$\text{subject to} \quad y_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1, \qquad n = 1, \ldots, N$$

Large Margin = Good Generalization

- Intuitively, large margins mean good generalization
  - Large margin => small ||**w**||
  - small ||**w**|| => regularized/simple solutions

- (Learning theory gives a more formal justification)

Solving the SVM Optimization Problem

Our optimization problem is:

$$\text{Minimize} \quad f(\mathbf{w}, b) = \frac{||\mathbf{w}||^2}{2}$$

$$\text{subject to} \quad 1 \leq y_n(\mathbf{w}^T\mathbf{x}_n + b), \qquad n = 1, \ldots, N$$

Introducing Lagrange Multipliers α_n ($n = \{1, \ldots, N\}$), one for each constraint, leads to the **Lagrangian**:

$$\text{Minimize} \quad L(\mathbf{w}, b, \alpha) = \frac{||\mathbf{w}||^2}{2} + \sum_{n=1}^{N} \alpha_n \{1 - y_n(\mathbf{w}^T\mathbf{x}_n + b)\}$$

$$\text{subject to} \quad \alpha_n \geq 0; \quad n = 1, \ldots, N$$

Solving the SVM Optimization Problem

Take (partial) derivatives of L_P w.r.t. $\mathbf{w}$, b and set them to zero

$$\frac{\partial L_P}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n, \quad \frac{\partial L_P}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N} \alpha_n y_n = 0$$

Substituting these in the Primal Lagrangian L_P gives the Dual Lagrangian

Maximize $L_D(\mathbf{w}, b, \alpha) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{m,n=1}^{N} \alpha_m \alpha_n y_m y_n (\mathbf{x}_m^T \mathbf{x}_n)$

subject to $\sum_{n=1}^{N} \alpha_n y_n = 0, \quad \alpha_n \geq 0; \quad n = 1, \ldots, N$

Solving the SVM Optimization Problem

Take (partial) derivatives of L_P w.r.t. $\mathbf{w}$, b and set them to zero

A Quadratic Program for which many off-the-shelf solvers exist

$$= \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n, \qquad \frac{\partial L_P}{\partial b} = 0 \Rightarrow \sum_{n=1}^{N} \alpha_n y_n = 0$$

Substituting thes … the Primal Lagrangian L_P gives the Dual Lagrangian

Maximize $\quad L_D(\mathbf{w}, b, \alpha) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{m,n=1}^{N} \alpha_m \alpha_n y_m y_n (\mathbf{x}_m^T \mathbf{x}_n)$

subject to $\quad \sum_{n=1}^{N} \alpha_n y_n = 0, \quad \alpha_n \geq 0; \quad n = 1, \ldots, N$

SVM: the solution!

Once we have the α_n's, $\mathbf{w}$ and b can be computed as:

$$\mathbf{w} = \sum_{n=1}^{N} \alpha_n y_n \mathbf{x}_n$$

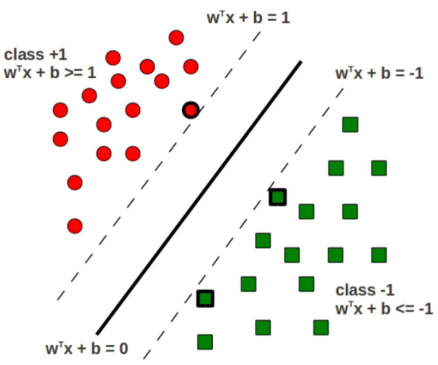
$$b = -\frac{1}{2}\left(\min_{n:y_n=+1} \mathbf{w}^T\mathbf{x}_n + \max_{n:y_n=-1} \mathbf{w}^T\mathbf{x}_n\right)$$

Note: Most α_n's in the solution are zero (sparse solution)

- Reason: Karush-Kuhn-Tucker (KKT) conditions
- For the optimal α_n's

$$\alpha_n\{1 - y_n(\mathbf{w}^T\mathbf{x}_n + b)\} = 0$$

- α_n is non-zero only if $\mathbf{x}_n$ lies on one of the two margin boundaries, i.e., for which $y_n(\mathbf{w}^T\mathbf{x}_n + b) = 1$
- These examples are called support vectors
- Support vectors "support" the margin boundaries

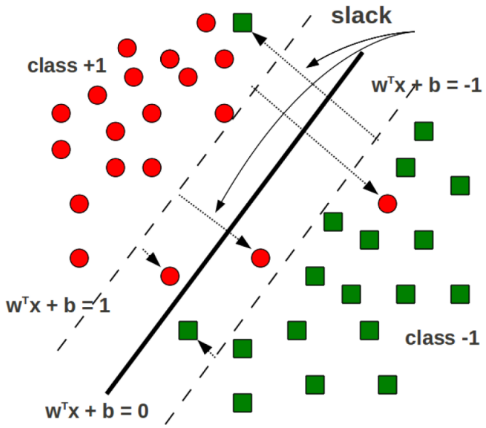


SVM in the non-separable case

- no hyperplane can separate the classes perfectly

- We still want to find the max margin hyperplane, but
  - We will allow some training examples to be **misclassified**
  - We will allow some training examples to fall **within** the margin region

SVM in the non-separable case



Recall: For the separable case (training loss = 0), the constraints were:

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

For the non-separable case, we relax the above constraints as:

$$\boxed{y_n(\mathbf{w}^T \mathbf{x}_n + b) \geq 1 - \xi_n \quad \forall n}$$

ξ_n is called slack variable (distance $\mathbf{x}_n$ goes past the margin boundary)

$\xi_n \geq 0, \forall n$, misclassification when $\xi_n > 1$

SVM Optimization Problem

Non-separable case: We will allow misclassified training examples

- .. but we want their number to be minimized
  $\Rightarrow$ by *minimizing* the sum of slack variables ($\sum_{n=1}^{N} \xi_n$)

The optimization problem for the non-separable case

$$\text{Minimize} \quad f(\mathbf{w}, b) = \frac{||\mathbf{w}||^2}{2} + C \sum_{n=1}^{N} \xi_n$$

$$\text{subject to} \quad y_n(\mathbf{w}^T \mathbf{x}_n + b) \geq 1 - \xi_n, \quad \xi_n \geq 0 \qquad n = 1, \ldots, N$$

C hyperparameter dictates which term dominates the minimization

- Small C => prefer large margins and allows more misclassified examples
- Large C => prefer small number of misclassified examples, but at the expense of a small margin

Soft SVM

- Same optimization as :

$$\min_{\mathbf{w},b} \frac{\|\mathbf{w}\|^2}{2} + C\underbrace{\sum_{n=1}^{N} \max\left\{1 - y_n(\mathbf{w}^t\mathbf{x}_n), 0\right\}}_{\text{Hinge loss!}}$$

- Why?

- Have you seen this loss function before?

Our goal in 422

- Learning is the process of obtaining *expertise* from *experience*
- Our goal: *learning* “Machine Learning”

Beyond 422…

- Machine learning is everywhere
- Many opportunities to create new high impact applications

- But challenging issues arise
  - Fairness
  - Robustness
  - Interpretability
  - Privacy
  - …

What you should know: Linear Models

- What are linear models?
  - a general framework for binary classification
  - how optimization objectives are defined
    - loss functions and regularizers
  - separate model definition from training algorithm (Gradient Descent)

What you should know: Gradient Descent

- Gradient descent
  - a generic algorithm to minimize objective functions
  - what are the properties of the objectives for which it works well?
  - subgradient descent (ie what to do at points where derivative is not defined)
  - why choice of step size, initialization matter

What you should know: Probabilistic Models

- The Naïve Bayes classifier
  - Conditional independence assumption
  - How to train it?
  - How to make predictions?
  - How does it relate to other classifiers we know?

- Fundamental Machine Learning concepts
  - iid assumption
  - Bayes optimal classifier
  - Maximum Likelihood estimation

What you should know: Neural Networks

- What are Neural Networks?
  - Multilayer perceptron
- How to make a prediction given an input?
  - Forward propagation: Matrix operations + non-linearities
- Why are neural networks powerful?
  - Universal function approximators!
- How to train neural networks?
  - The backpropagation algorithm
    - How to step through it, and how to derive update rules

What you should know: PCA

- Principal Components Analysis
  - Goal: Find a **projection** of the data onto directions that **maximize variance** of the original data set
  - PCA **optimization objectives** and resulting **algorithm**
  - Why this is useful!

What you should know: Kernels

- Kernel functions
  - What they are, why they are useful, how they relate to feature combination

- Kernelized perceptron
  - You should be able to derive it and implement it

What you should know: SVMs

- What are Support Vector Machines
  - Hard margin vs. soft margin SVMs
- How to train SVMs
  - Which optimization problem we need to solve
- Geometric interpretation
  - What are support vectors and what is their relation with parameters **w**,b?
- How do SVM relate to the general formulation of linear classifiers
- Why/how can SVMs be kernelized