SVMs with Slack

Nicholas Ruozzi
University of Texas at Dallas

Based roughly on the slides of David Sontag
Dual SVM

\[
\max_{\lambda \geq 0} -\frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j y_i y_j x^{(i)T} x^{(j)} + \sum_i \lambda_i
\]

such that

\[
\sum_i \lambda_i y_i = 0
\]

• The dual formulation only depends on inner products between the data points

• Same thing is true if we use feature vectors instead
Dual SVM

\[
\max_{\lambda \geq 0} -\frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j y_i y_j \Phi(x^{(i)})^T \Phi(x^{(j)}) + \sum_i \lambda_i
\]

such that

\[
\sum_i \lambda_i y_i = 0
\]

• The dual formulation only depends on inner products between the data points

• Same thing is true if we use feature vectors instead
The Kernel Trick

• For some feature vectors, we can compute the inner products quickly, even if the feature vectors are very large

• This is best illustrated by example

  \[
  \phi(x_1, x_2) = \begin{bmatrix}
  x_1 x_2 \\
  x_2 x_1 \\
  x_1^2 \\
  x_2^2
  \end{bmatrix}
  \]

  • Let \( \phi(x_1, x_2) = (x_1^T z)^2 \)

  \[
  \phi(x_1, x_2)^T \phi(z_1, z_2) = x_1^2 z_1^2 + 2x_1 x_2 z_1 z_2 + x_2^2 z_2^2 \\
  = (x_1 z_1 + x_2 z_2)^2 \\
  = (x^T z)^2
  \]
The Kernel Trick

• For some feature vectors, we can compute the inner products quickly, even if the feature vectors are very large

• This is best illustrated by example

  • Let \( \phi(x_1, x_2) = \begin{bmatrix} x_1x_2 \\ x_2x_1 \\ x_1^2 \\ x_2^2 \end{bmatrix} \)

  • \( \phi(x_1, x_2)^T \phi(z_1, z_2) = x_1^2 z_1^2 + 2x_1x_2z_1z_2 + x_2^2 z_2^2 = (x_1z_1 + x_2z_2)^2 = (x^T z)^2 \)

Reduces to a dot product in the original space
The Kernel Trick

• The same idea can be applied for the feature vector \( \phi \) of all polynomials of degree (exactly) \( d \)

  \[ \phi(x)^T \phi(z) = (x^T z)^d \]

• More generally, a kernel is a function

  \[ k(x, z) = \phi(x)^T \phi(z) \] for some feature map \( \phi \)

• Rewrite the dual objective

  \[
  \max_{\lambda \geq 0, \sum_i \lambda_i y_i = 0} -\frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j y_i y_j k(x^{(i)}, x^{(j)}) + \sum_i \lambda_i
  \]
Examples of Kernels

• Polynomial kernel of degree exactly $d$
  
  \[ k(x, z) = (x^T z)^d \]

• General polynomial kernel of degree $d$ for some $c$
  
  \[ k(x, z) = (x^T z + c)^d \]

• Gaussian kernel for some $\sigma$
  
  \[ k(x, z) = \exp \left( \frac{-\|x-z\|^2}{2\sigma^2} \right) \]

  • The corresponding $\phi$ is infinite dimensional!

• So many more…
Gaussian Kernels

• Consider the Gaussian kernel

\[
\exp \left( -\frac{\|x - z\|^2}{2\sigma^2} \right) = \exp \left( -\frac{(x - z)^T(x - z)}{2\sigma^2} \right)
\]

\[
= \exp \left( -\frac{\|x\|^2 + 2x^Tz - \|z\|^2}{2\sigma^2} \right)
\]

\[
= \exp \left( -\frac{\|x\|^2}{2\sigma^2} \right) \exp \left( -\frac{\|z\|^2}{2\sigma^2} \right) \exp \left( \frac{x^Tz}{\sigma^2} \right)
\]

• Use the Taylor expansion for \(\exp()\)

\[
\exp \left( \frac{x^Tz}{\sigma^2} \right) = \sum_{n=0}^{\infty} \frac{(x^Tz)^n}{\sigma^{2n}n!}
\]
Gaussian Kernels

• Consider the Gaussian kernel

\[
\exp\left(\frac{-\|x - z\|^2}{2\sigma^2}\right) = \exp\left(\frac{-(x - z)^T(x - z)}{2\sigma^2}\right)
\]

\[
= \exp\left(\frac{-\|x\|^2 + 2x^Tz - \|z\|^2}{2\sigma^2}\right)
\]

\[
= \exp\left(-\frac{\|x\|^2}{2\sigma^2}\right) \exp\left(-\frac{\|z\|^2}{2\sigma^2}\right) \exp\left(\frac{x^Tz}{\sigma^2}\right)
\]

• Use the Taylor expansion for \(\exp()\)

\[
\exp\left(\frac{x^Tz}{\sigma^2}\right) = \sum_{n=0}^{\infty} \frac{(x^Tz)^n}{\sigma^{2n}n!}
\]

Polynomial kernels of every degree!
Kernels

• Bigger feature space increases the possibility of overfitting
  • Large margin solutions may still generalize reasonably well
• Alternative: add “penalties” to the objective to disincentivize complicated solutions

$$\min_w \frac{1}{2} \|w\|^2 + c \cdot (\# \ of \ misclassifications)$$

• Not a quadratic program anymore (in fact, it’s NP-hard)
• Similar problem to counting the number of misclassifications, no notion of how badly the data is misclassified
SVMs with Slack

- Allow misclassification
  - Penalize misclassification linearly (just like in the perceptron algorithm)
    - Again, easier to work with than counting misclassifications
  - Objective stays convex
- Will let us handle data that isn’t linearly separable!
SVMs with Slack

\[
\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + c \sum_i \xi_i
\]

such that

\[
y_i (w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i
\]

\[
\xi_i \geq 0, \text{ for all } i
\]
SVMs with Slack

\[
\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + c \sum_i \xi_i
\]

such that

\[
y_i(w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i
\]

\[
\xi_i \geq 0, \text{ for all } i
\]

Potentially allows some points to be misclassified/inside the margin
SVMs with Slack

\[
\begin{align*}
\min_{w, b, \xi} & \quad \frac{1}{2} \|w\|^2 + c \sum_i \xi_i \\
\text{such that} & \quad y_i (w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i \\
& \quad \xi_i \geq 0, \text{ for all } i
\end{align*}
\]

Constant $c$ determines degree to which slack is penalized
SVMs with Slack

\[
\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + c \sum_i \xi_i
\]

such that

\[
y_i(w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i
\]

\[
\xi_i \geq 0, \text{ for all } i
\]

• How does this objective change with \( c \)?
SVMs with Slack

\[
\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + c \sum_i \xi_i
\]

such that

\[
y_i(w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i
\]

\[
\xi_i \geq 0, \text{ for all } i
\]

• How does this objective change with \( c \)?
  
• As \( c \to \infty \), requires a perfect classifier
  
• As \( c \to 0 \), allows arbitrary classifiers (i.e., ignores the data)
SVMs with Slack

\[
\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + c \sum \xi_i
\]

such that

\[
y_i(w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i
\]

\[
\xi_i \geq 0, \text{ for all } i
\]

• How should we pick \(c\)?
SVMs with Slack

$$\min_{w, b, \xi} \frac{1}{2} \|w\|^2 + c \sum_{i} \xi_i$$

such that

$$y_i (w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i$$

$$\xi_i \geq 0, \text{ for all } i$$

• How should we pick $c$?

• Divide the data into three pieces training, testing, and validation

• Use the validation set to tune the value of the hyperparameter $c$
Evaluation Methodology

• General learning strategy
  • Build a classifier using the training data
  • Select hyperparameters using validation data
  • Evaluate the chosen model with the selected hyperparameters on the test data

How can we tell if we overfit the training data?
ML in Practice

• Gather Data + Labels
• Select feature vectors
• Randomly split into three groups
  • Training set
  • Validation set
  • Test set
• Experimentation cycle
  • Select a “good” hypothesis from the hypothesis space
  • Tune hyperparameters using validation set
  • Compute accuracy on test set (fraction of correctly classified instances)
SVMs with Slack

\[ \min_{w,b,\xi} \frac{1}{2} \|w\|^2 + c \sum_i \xi_i \]

such that

\[ y_i(w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i \]

\[ \xi_i \geq 0, \text{ for all } i \]

• What is the optimal value of \( \xi \) for fixed \( w \) and \( b \)?
SVMs with Slack

\[
\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + c \sum_i \xi_i
\]

such that

\[
y_i(w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i
\]

\[
\xi_i \geq 0, \text{ for all } i
\]

• What is the optimal value of \(\xi\) for fixed \(w\) and \(b\)?

• If \(y_i(w^T x^{(i)} + b) \geq 1\), then \(\xi_i = 0\)

• If \(y_i(w^T x^{(i)} + b) < 1\), then \(\xi_i = 1 - y_i(w^T x^{(i)} + b)\)
SVMs with Slack

\[ \min_{w,b,\xi} \frac{1}{2} \|w\|^2 + c \sum_i \xi_i \]

such that

\[ y_i (w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i \]

\[ \xi_i \geq 0, \text{ for all } i \]

• We can formulate this slightly differently

  • \( \xi_i = \max\{0, 1 - y_i (w^T x^{(i)} + b) \} \)

• Does this look familiar?

• Hinge loss provides an upper bound on Hamming loss
Hinge Loss Formulation

• Obtain a new objective by substituting in for $\xi$

$$\min_{w,b} \frac{1}{2} \|w\|^2 + c \sum_i \max\{0, 1 - y_i (w^T x^{(i)} + b)\}$$

Can minimize with gradient descent!
Hinge Loss Formulation

- Obtain a new objective by substituting in for $\xi$

$$
\min_{w,b} \frac{1}{2} \|w\|^2 + c \sum_i \max\{0, 1 - y_i(w^T x^{(i)} + b)\}
$$

Penalty to prevent overfitting

Hinge loss
Imbalanced Data

- If the data is imbalanced (i.e., more positive examples than negative examples), may want to evenly distribute the error between the two classes

\[
\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + \frac{c}{N_+} \sum_{i:y_i=1} \xi_i + \frac{c}{N_-} \sum_{i:y_i=-1} \xi_i
\]

such that

\[
y_i (w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i
\]

\[
\xi_i \geq 0, \text{ for all } i
\]
Dual of Slack Formulation

\[
\min_{w, b, \xi} \frac{1}{2} \|w\|^2 + c \sum_i \xi_i
\]

such that

\[y_i(w^T x^{(i)} + b) \geq 1 - \xi_i, \text{ for all } i\]

\[\xi_i \geq 0, \text{ for all } i\]
Dual of Slack Formulation

\[ L(w, b, \xi, \lambda, \mu) = \frac{1}{2} w^T w + c \sum_i \xi_i + \sum_i \lambda_i (1 - \xi_i - y_i (w^T x^{(i)} + b)) + \sum_i -\mu_i \xi_i \]

Convex in \( w, b, \xi \), so take derivatives to form the dual

\[ \frac{\partial L}{\partial w_k} = w_k + \sum_i -\lambda_i y_i x_k^{(i)} = 0 \]

\[ \frac{\partial L}{\partial b} = \sum_i -\lambda_i y_i = 0 \]

\[ \frac{\partial L}{\partial \xi_k} = c - \lambda_k - \mu_k = 0 \]
Dual of Slack Formulation

\[
\max_{\lambda \geq 0} - \frac{1}{2} \sum_i \sum_j \lambda_i \lambda_j y_i y_j x^{(i)T} x^{(j)} + \sum_i \lambda_i
\]

such that

\[
\sum_i \lambda_i y_i = 0
\]

\[c \geq \lambda_i \geq 0, \text{ for all } i\]
Generalization

• We argued, intuitively, that SVMs generalize better than the perceptron algorithm

  • How can we make this precise?

  • Coming soon... but first...
Roadmap

• Where are we headed?
  • Other simple hypothesis spaces for supervised learning
    • $k$ nearest neighbor
    • Decision trees
  • Learning theory
    • Generalization and PAC bounds
    • VC dimension
    • Bias/variance tradeoff