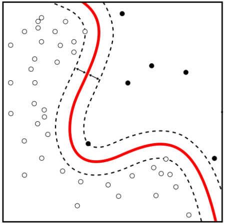
Support Vector Machines and Kernel Methods

Shrey Gupta

Applied Machine Learning (HOUSECS 59-01), Duke University

October 10, 2018

Support Vector Machines



- Suppose we plotted all our relevant data for a classification problem–there should be a dividing "line" (or hyperplane) that classifies the data into classes.
  - Obviously, there might not be a perfect classification hyperplane (and more features might be needed).

Image source: Wikipedia

Margin

- The *margin* of a data point is it's distance to the classification boundary.
  - *Positive* if on the correct side of the boundary, and *negative* if not.
- It would be preferred to have all data points as far from the boundary as possible (i.e. large margin).
  - *Why?* Small shifts in the boundary won't affect the classification output.

Margin

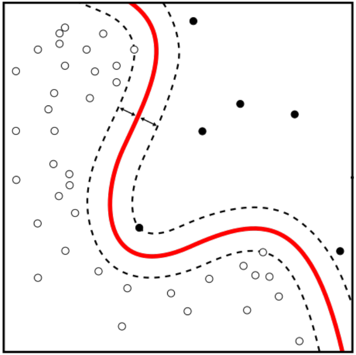


Image source: Wikipedia

Margin

- Support vector machines (SVMs) maximize the minimum margin over the training set.
- Many other machine learning algorithms are poor at this.
  - Hence, test data points near the boundary can easily be misclassified.

Support Vectors

- Support vectors “define” the classification boundary: they are the data points nearest to the boundary.
  - The other data points are “irrelevant” and do not have an effect on the boundary.

Optional: Lagrangian Formulation

$$\min_{w,b,\xi} \frac{1}{2}|w|^2 + C\sum_{i=0}^{n} \xi_i$$

$$\text{s.t. } y_i(w^T\phi(x_i) + b) \geq 1 - \xi_i, \xi_i \geq 0 \ \forall \ i$$

- This is our goal: the first term relates to maximizing the minimum margin (we want $1/|w|^2$ to be large).
- The second term allows some "slack" for incorrect classifications: we allow them, but with some penalty (C).
- ϕ is a kernel transformation, and will be introduced soon.

Optional: Lagrangian Formulation

$$\min_{w,b,\xi} \frac{1}{2}|w|^2 + C\sum_{i=0}^{n} \xi_i$$

$$\text{s.t. } y_i(w^T\phi(x_i) + b) \geq 1 - \xi_i, \xi_i \geq 0 \ \forall \ i$$

- We have some constraints; the first is that the scaled margin, plus slack, must be greater than one.
- The second is that the “slack” must be positive (which makes sense intuitively).

Optional: Lagrangian Formulation

$$L = \frac{1}{2}|w|^2 + C\sum_{i=0}^{n}\xi_i - \sum_{i=0}^{n}\alpha_i[y_i(w^T\phi(x_i)+b) - 1 + \xi_i] - \sum_{i=0}^{n} r_i\xi_i$$

$$\max_{\alpha}\sum_{i=0}^{n}\alpha_i - \frac{1}{2}\sum_{i,k=0}^{n}\alpha_i\alpha_k y_i y_k \phi(x_i)^T\phi(x_k)$$

$$\text{s.t. } 0 \leq \alpha_i \leq C \ \forall\ i, \sum_{i=0}^{n}\alpha_i y_i = 0$$

- We can formulate the Lagrangian and dual problem, after a little bit of work.

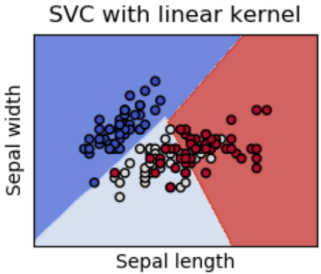
Optional: Lagrangian Formulation

- We can solve the dual problem using an algorithm such as sequential minimal optimization.
- Note: while this is outside the scope of the course, if you find it interesting, take a deeper look!

Kernels

- It's very likely that the dividing hyperplane is not enough to separate the data well.
- Let's use a "trick" similar to what we did with regression: *transform* our features using a *kernel*.
  - A lot of the mathematics behind kernels is out of the scope of the course, but may be interesting (and insightful) to you.

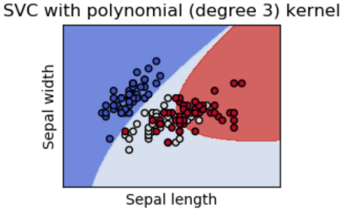
Linear Kernels



- Kernel: $\langle x, x' \rangle$ is the "basic" kernel, and does not map to a higher dimensional space.

Image source: scikit-learn

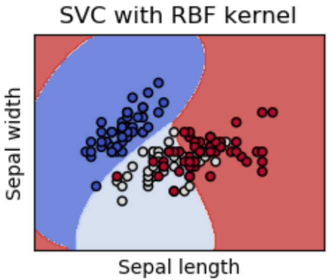
Polynomial Kernels



- Kernel: $(\langle x, x' \rangle + c)^d$ maps to a d-dimensional space, with hyperparameters c and d.

Image source: scikit-learn

RBF Kernels



- Kernel: $\exp(-\gamma|x - x'|^2)$ maps to an *infinite* dimensional space, with hyperparameter γ.

Image source: scikit-learn

Practicalities

- Distance is an important metric for SVMs, so it is crucial to normalize features! (Some packages do this automatically.)
- Start with simpler kernels first, and work your way up to more complex kernels *if* they perform better.
  - *Very large dataset:* algorithm can become infeasible.
  - *Small dataset but large number of features:* be careful using a kernel (e.g. RBF) that easily over-fits the training data.

Notebook

- Today's notebook will work through an example of support vector machines.