Supervised, Unsupervised and Reinforcement Learning in Finance

Week 1: Supervised Learning

Support Vector Machines part III: the kernel trick

Igor Halperin

NYU Tandon School of Engineering, 2017

Constrained optimization for SVM

Solution for the SVM Regression:

$$f(x)=\sum_{i=1}^{N}\left(\alpha_i-\alpha_i^*\right)\left\langle x_i,x\right\rangle+b$$

The dual optimization problem for Lagrange multipliers $\alpha_i,\ \alpha_i^*\geq 0$:

$$\min_{\alpha_i,\alpha_i^*}\ \frac{1}{2}\sum_{i,j=1}^{N}\left(\alpha_i-\alpha_i^*\right)\left(\alpha_j-\alpha_j^*\right)\left\langle x_i,x_j\right\rangle+\varepsilon\sum_{i=1}^{N}\left(\alpha_i+\alpha_i^*\right)-\sum_{i=1}^{N}y_i\left(\alpha_i-\alpha_i^*\right)$$

$$s.t.\ \sum_{i=1}^{N}\left(\alpha_i-\alpha_i^*\right)=0,\quad 0\leq\alpha_i,\ \alpha_i^*\leq C$$

The solution depends only on a dot product of x rather than on x itself!

Constrained optimization for SVM

Solution for the SVM Regression:

$$f(x)=\sum_{i=1}^{N}\left(\alpha_i-\alpha_i^*\right)\left\langle x_i,x\right\rangle+b$$

The dual optimization problem for Lagrange multipliers $\alpha_i,\ \alpha_i^*\geq 0$:

$$\min_{\alpha_i,\alpha_i^*}\ \frac{1}{2}\sum_{i,j=1}^{N}\left(\alpha_i-\alpha_i^*\right)\left(\alpha_j-\alpha_j^*\right)\left\langle x_i,x_j\right\rangle+\varepsilon\sum_{i=1}^{N}\left(\alpha_i+\alpha_i^*\right)-\sum_{i=1}^{N}y_i\left(\alpha_i-\alpha_i^*\right)$$

$$s.t.\ \sum_{i=1}^{N}\left(\alpha_i-\alpha_i^*\right)=0,\quad 0\leq\alpha_i,\ \alpha_i^*\leq C$$

The solution depends only on a dot product of x rather than on x itself!

If we used features $\Phi(x)$ instead of raw inputs, the solution is obtained in the same way:

$$f(x)=\sum_{i=1}^{N}\left(\alpha_i-\alpha_i^*\right)\left\langle\Phi(x_i),\Phi(x)\right\rangle+b$$

SVM: introducing non-linearity by preprocessing

How to make features that handles non-linearities by converting a non-linear problem into a linear problem in a feature space

Example: we want to capture quadratic features for points on a plane $x \in R^2$

$\Phi(x) = \left(x_1^2, \sqrt{2} x_1 x_2, x_2^2 \right)$ can be viewed as a vector in R^3. The key observation is that (check it!)

$$\left\langle \Phi(x), \Phi(x') \right\rangle = \left\langle \left(x_1^2, \sqrt{2} x_1 x_2, x_2^2 \right), \left(x'^2_1, \sqrt{2} x'_1 x'_2, x'^2_2 \right) \right\rangle = \left\langle x, x' \right\rangle^2$$

This means that by using features constructed from raw inputs, we can capture quadratic and higher order effects in the SVM, as long as all results depend only on dot products

Questions:
1. How to construct features?
2. Can we produce MANY features, or even infinite number of features? Can it help?

SVM: the kernel trick

If we do linear SVM in a feature space (e.g. R^3, in our example), the results is

$$f(x) = \sum_{i=1}^{N} \left(\alpha_i - \alpha_i^*\right) \left\langle \Phi(x_i), \Phi(x) \right\rangle + b$$

$$= \sum_{i=1}^{N} \left(\alpha_i - \alpha_i^*\right) k(x_i, x) + b$$

Kernel

The kernel trick: replace the feature engineering by a kernel engineering, i.e. model a kernel instead of modeling features!

Why may be a good idea:

1. A kernel should be interpretable as a dot product in some high-dimensional (or even infinite-dimensional) feature space
2. A number of theorems provide conditions for admissible kernels (Mercer's theorem, etc., see Smola and Scholkopf, or Geron)
3. A number of popular kernels is available in both the literature and software

Examples of kernels for SMV

1. A linear kernel $$k(x,x') = \langle x,x' \rangle$$

Does not handle non-linearities, but is faster to work with than using non-linear kernels

2. Polynomial kernel $$k(x,x') = \left(\gamma \langle x,x' \rangle + r\right)^d$$

3. Sigmoid kernel $$k(x,x') = \tanh\left(\gamma \langle x,x' \rangle + r\right)$$

4. Gaussian Radial Basis Function (RBF) kernel

$$k(x,x') = \exp\left(-\gamma \| x - x' \|^2\right)$$

Features corresponding to a Gaussian RBF kernel are actually infinite-dimensional!

Kernel parameters such as γ are model hyper-parameters (alongside C, ε). Can be tuned using a validation dataset or cross-validation. A version called ν-SVR tunes ε

More advanced methods: kernel learning

Control question

Select all correct answers

1. The kernel trick used by the SVM is based on the observation that the final result depends only on dot products of inputs x rather than directly on their values.

2. Instead of constructing features $\Phi(x)$, we can directly model kernels $k(x,x')$.

3.Kernels $k(x,x') = \langle x,x' \rangle$, $k(x,x') = \left(\langle x,x' \rangle + r\right)^d$ are both valid kernels.

4.A Gaussian RBF kernel is defined as $\exp\left(-\gamma k^2(x,x')\right)$

5.A Gaussian RBF kernel is defined as $\exp\left(-\gamma k^2 \left\|x - x'\right\|^2\right)$

Correct answers: 1,2, 3, 5