Lecture 12: Support Vector Regression, Kernel Trick and Optimization Algorithm

Instructor: Prof. Ganesh Ramakrishnan

Some observations

- $\alpha_i, \alpha_i^* \geq 0$, $\mu_i, \mu_i^* \geq 0$, $\alpha_i + \mu_i = C$ and $\alpha_i^* + \mu_i^* = C$
  Thus, $\alpha_i, \mu_i, \alpha_i^*, \mu_i^* \in [0, C]$, $\forall i$
- If $0 < \alpha_i < C$, then $0 < \mu_i < C$
  (as $\alpha_i + \mu_i = C$)
- $\mu_i \xi_i = 0$ and $\alpha_i(y_i - w^\top \phi(x_i) - b - \epsilon - \xi_i) = 0$ are complementary slackness conditions
  So $0 < \alpha_i < C \Rightarrow \xi_i = 0$ and $y_i - w^\top \phi(x_i) - b = \epsilon + \xi_i = \epsilon$
  - All such points lie on the boundary of the ϵ band
  - Using any point x_j (that is with $\alpha_j \in (0, C)$) on margin, we can recover b as:
    $b = y_j - w^\top \phi(x_j) - \epsilon$

Support Vector Regression

Dual Objective

Dual function

- Let $L^*(\alpha, \alpha^*, \mu, \mu^*) = \min_{w,b,\xi,\xi^*} L(w, b, \xi, \xi^*, \alpha, \alpha^*, \mu, \mu^*)$
- By weak duality theorem, we have:
  $\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \geq L^*(\alpha, \alpha^*, \mu, \mu^*)$
  s.t. $y_i - w^\top \phi(x_i) - b \leq \epsilon - \xi_i$, and
  $w^\top \phi(x_i) + b - y_i \leq \epsilon - \xi_i^*$, and
  $\xi_i, \xi^* \geq 0$, $\forall i = 1, \ldots, n$
- The above is true for any $\alpha_i, \alpha_i^* \geq 0$ and $\mu_i, \mu_i^* \geq 0$
- Thus,

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) \geq \max_{\alpha,\alpha^*,\mu,\mu^*} L^*(\alpha, \alpha^*, \mu, \mu^*)$$

s.t. $y_i - w^\top \phi(x_i) - b \leq \epsilon - \xi_i$, and
$w^\top \phi(x_i) + b - y_i \leq \epsilon - \xi_i^*$, and
$\xi_i, \xi^* \geq 0$, $\forall i = 1, \ldots, n$

Dual objective

- In case of Support Vector Regression, we have a strictly convex objective and linear constraints $\Rightarrow$ KKT conditions are necessary and sufficient and strong duality holds:

$$\min_{w,b,\xi,\xi^*} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) = \max_{\alpha,\alpha^*,\mu,\mu^*} L^*(\alpha, \alpha^*, \mu, \mu^*)$$

s.t. $y_i - w^\top \phi(x_i) - b \leq \epsilon - \xi_i$, and
$w^\top \phi(x_i) + b - y_i \leq \epsilon - \xi_i^*$, and
$\xi_i, \xi^* \geq 0$, $\forall i = 1, \ldots, n$

- This value is precisely obtained at the $(\mathbf{w}, \mathbf{b}, \xi, \xi^*, \alpha, \alpha^*, \mu, \mu^*)$ that satisfies the necessary (and sufficient) optimality conditions
- Given strong duality, we can equivalently solve

$$\max_{\alpha,\alpha^*,\mu,\mu^*} L^*(\alpha, \alpha^*, \mu, \mu^*)$$

- $L(\alpha, \alpha^*, \mu, \mu^*) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{n}(\xi_i + \xi_i^*) +$
  $\sum_{i=1}^{n}\left(\alpha_i(y_i - w^\top\phi(x_i) - b - \epsilon - \xi_i) + \alpha_i^*(w^\top\phi(x_i) + b - y_i - \epsilon - \xi_i^*)\right)$
  $\sum_{i=1}^{n}(\mu_i\xi_i + \mu_i^*\xi_i^*)$
- We obtain w, b, ξ_i, ξ_i^* in terms of α, α^*, μ and μ^* by using the KKT conditions derived earlier as $w = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\phi(x_i)$ and $\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0$ and $\alpha_i + \mu_i = C$ and $\alpha_i^* + \mu_i^* = C$
- Thus, we get:
  $L(w, b, \xi, \xi^*, \alpha, \alpha^*, \mu, \mu^*)$
  $= \frac{1}{2}\sum_i\sum_j(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\phi^\top(x_i)\phi(x_j) +$
  $\sum_i\left(\xi_i(C - \alpha_i - \mu_i) + \xi_i^*(C - \alpha_i^* - \mu_i^*)\right) - b\sum_i(\alpha_i - \alpha_i^*) -$
  $\epsilon\sum_i(\alpha_i + \alpha_i^*) + \sum_i y_i(\alpha_i - \alpha_i^*) - \sum_i\sum_j(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\phi^\top(x_i)\phi(x_j)$
  $= -\frac{1}{2}\sum_i\sum_j(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\phi^\top(x_i)\phi(x_j) - \epsilon\sum_i(\alpha_i + \alpha_i^*) +$
  $\sum_i y_i(\alpha_i - \alpha_i^*)$

Kernel function: $K(x_i, x_j) = \phi^T(x_i)\phi(x_j)$

- $w = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\phi(x_i) \Rightarrow$ the final decision function $f(x) = w^T\phi(x) + b =$ $\sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\phi^T(x_i)\phi(x) + y_j - \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\phi^T(x_i)\phi(x_j) - \epsilon$ x_j is any point with $\alpha_j \in (0, C)$
- The dual optimization problem to compute the α's for SVR is:

$$max_{\alpha_i,\alpha_i^*} - \frac{1}{2}\sum_i \sum_j (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\phi^\top(x_i)\phi(x_j)$$

$$-\epsilon \sum_i (\alpha_i + \alpha_i^*) + \sum_i y_i(\alpha_i - \alpha_i^*)$$

  s.t.
  - $\sum_i(\alpha_i - \alpha_i^*) = 0$
  - $\alpha_i, \alpha_i^* \in [0, C]$
- ***We notice that the only way these three expressions involve ϕ is through $\phi^\top(x_i)\phi(x_j) = K(x_i, x_j)$, for some i, j***

Kernelized form for SVR

- The *kernelized* dual optimization problem to compute the α's for SVR is:

$$max_{\alpha_i,\alpha_i^*} - \frac{1}{2}\sum_i \sum_j (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)K(x_i, x_j)$$

$$-\epsilon \sum_i (\alpha_i + \alpha_i^*) + \sum_i y_i(\alpha_i - \alpha_i^*)$$

s.t.

- $\sum_i (\alpha_i - \alpha_i^*) = 0$
- $\alpha_i, \alpha_i^* \in [0, C]$

The Kernel function in SVR

- Again, invoking the ***kernel* function**: $K(x_1, x_2) = \phi^\top(x_1)\phi(x_2)$
- The decision function becomes: $f(x) = \sum_i (\alpha_i - \alpha_i^*) K(x_i, x) + b$
- Using any point x_j (that is with $\alpha_j \in (0, C)$) on margin, we can recover b as: $b = y_j - w^\top \phi(x_j) - \epsilon = y_j - \sum_i (\alpha_i - \alpha_i^*) K(x_i, x_j)$
- Thus, the optimization problem as well as the final decision function are only in terms of the kernel function $K(x, x')$.
- We will see that, often, computing $K(x_1, x_2)$ does not even require computing $\phi(x_1)$ or $\phi(x_2)$ explicitly

How about Ridge Regression?

- Recall for Ridge Regression: $w = (\Phi^T\Phi + \lambda I)^{-1}\Phi^T y$, where,

$$\Phi = \begin{bmatrix} \phi_1(x_1) & \dots & \phi_p(x_1) \\ \dots & \dots & \dots \\ \phi_1(x_m) & \dots & \phi_p(x_m) \end{bmatrix}$$

  and

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \dots \\ y_m \end{bmatrix}$$

- $\left(\Phi^T\Phi\right)_{ij} = \sum_{k=1}^{m} \phi_i(x_k)\phi_j(x_k)$ whereas $\left(\Phi\Phi^T\right)_{ij} = \sum_{k=1}^{p} \phi_k(x_i)\phi_k(x_j) = K(x_i, x_j)$

Please note the difference between Φ and $\phi(x)$

- $$\Phi = \begin{bmatrix} \phi_1(x_1) & \dots & \phi_p(x_1) \\ \dots & \dots & \dots \\ \phi_1(x_m) & \dots & \phi_p(x_m) \end{bmatrix}$$

  and

  $$\phi(x_j) = \begin{bmatrix} \phi_1(x_j) \\ \dots \\ \phi_p(x_j) \end{bmatrix}$$

- $\phi^T(x_i)\phi(x_j) = K(x_i, x_j)$
- $\left(\Phi^T\Phi\right)_{ij} = \sum_{k=1}^{m} \phi_i(x_k)\phi_j(x_k)$
- $\left(\Phi\Phi^T\right)_{ij} = \sum_{k=1}^{p} \phi_k(x_i)\phi_k(x_j) = \phi^T(x_i)\phi(x_j) = K(x_i, x_j)$

Kernelizing Ridge Regression

- Given $w = (\Phi^T\Phi + \lambda I)^{-1}\Phi^T y$ and using the identity $(P^{-1} + B^T R^{-1} B)^{-1} B^T R^{-1} = PB^T(BPB^T + R)^{-1}$
  - $\Rightarrow$
  - $\Rightarrow$

How about Ridge Regression?

- Given $w = (\Phi^T\Phi + \lambda I)^{-1}\Phi^T y$ and using the identity $(P^{-1} + B^T R^{-1} B)^{-1} B^T R^{-1} = PB^T(BPB^T + R)^{-1}$
  - $\Rightarrow w = \Phi^T(\Phi\Phi^T + \lambda I)^{-1} y = \sum_{i=1}^{m} \alpha_i \phi(x_i)$ where $\alpha_i = \left((\Phi\Phi^T + \lambda I)^{-1} y\right)_i$
  - $\Rightarrow$ the final decision function $f(x) = \phi^T(x) w = \sum_{i=1}^{m} \alpha_i \phi^T(x)\phi(x_i)$
- Again, ***We notice that the only way the decision function*** $f(x)$ ***involves*** ϕ ***is through*** $\phi^\top(x_i)\phi(x_j)$***, for some*** i, j

The Kernel function in Ridge Regression

- We call $\phi^\top(x_1)\phi(x_2)$ a ***kernel* function**:
  $K(x_1, x_2) = \phi^\top(x_1)\phi(x_2)$
- The preceding expression for decision function becomes
  $f(x) = \sum_{i=1}^{m} \alpha_i K(x, x_i)$
  where $\alpha_i = \left(([K(x_i, x_j)] + \lambda I)^{-1} y\right)_i$

Back to the Kernelized version of SVR

- The kernelized dual problem:

$$max_{\alpha_i,\alpha_i^*} - \frac{1}{2}\sum_i\sum_j(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)K(x_i,x_j)$$

$$-\epsilon\sum_i(\alpha_i+\alpha_i^*)+\sum_i y_i(\alpha_i-\alpha_i^*)$$

  s.t.

  - $\sum_i(\alpha_i-\alpha_i^*)=0$
  - $\alpha_i,\alpha_i^*\in[0,C]$

- The kernelized decision function:
  $f(x)=\sum_i(\alpha_i-\alpha_i^*)K(x_i,x)+b$
- Using any point x_j with $\alpha_j\in(0,C)$:
  $b=y_j-\sum_i(\alpha_i-\alpha_i^*)K(x_i,x_j)$
- Computing $K(x_1,x_2)$ often does not even require computing $\phi(x_1)$ or $\phi(x_2)$ explicitly

An example

- Let $K(x_1, x_2) = (1 + x_1^\top x_2)^2$
- What $\phi(x)$ will give $\phi^\top(x_1)\phi(x_2) = K(x_1, x_2) = (1 + x_1^\top x_2)^2$
- Is such a ϕ guaranteed to exist?
- Is there a unique ϕ for given K?

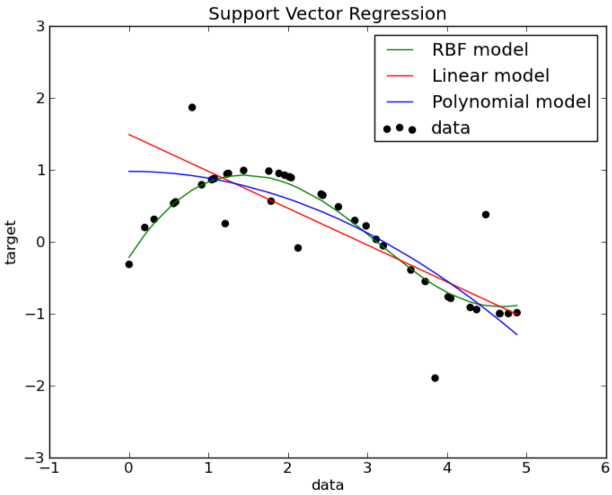
- We can prove that such a ϕ exists
- For example, for a 2-dimensional x_i:

$$\phi(x_i) = \begin{bmatrix} 1 \\ x_{i1}\sqrt{2} \\ x_{i2}\sqrt{2} \\ x_{i1}x_{i2}\sqrt{2} \\ x_{i1}^2 \\ x_{i2}^2 \end{bmatrix}$$

- $\phi(x_i)$ exists in a 5-dimensional space
- Thus, to compute $K(x_1, x_2)$, all we need is $x_1^\top x_2$, and there is no need to compute $\phi(x_i)$

Introduction to the Kernel Trick (more later)

- **Kernels** operate in a *high-dimensional*, *implicit* feature space without ever computing the coordinates of the data in that space, but rather by simply computing the Kernel function
- This approach is called the "*kernel trick*" and will talk about *valid kernels* a little later...
- This operation is often computationally cheaper than the explicit computation of the coordinates



Support Vector Regression
RBF model
Linear model
Polynomial model
data
target
data
3
2
1
0
−1
−2
−3
−1
0
1
2
3
4
5
6

Solving the SVR Dual Optimization Problem

- The SVR dual objective is:
  $max_{\alpha_i,\alpha_i^*} -\frac{1}{2}\sum_i\sum_j(\alpha_i-\alpha_i^*)(\alpha_j-\alpha_j^*)K(x_i,x_j)$
  $-\epsilon\sum_i(\alpha_i+\alpha_i^*)+\sum_i y_i(\alpha_i-\alpha_i^*)$
- This is a linearly constrained quadratic program (LCQP), just like the constrained version of Lasso
- There exists no closed form solution to this formulation
- Standard QP (LCQP) solvers[1] can be used
- Question: Are there more specific and efficient algorithms for solving SVR in this form?

[1] https://en.wikipedia.org/wiki/Quadratic_programming#Solvers_and_scripting_.28programming.29_languages

Solving the SVR Dual Optimization Problem

- It can be shown that the objective:
  $max_{\alpha_i,\alpha_i^*} - \frac{1}{2}\sum_i \sum_j (\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*) K(x_i, x_j)$
  $-\epsilon \sum_i (\alpha_i + \alpha_i^*) + \sum_i y_i(\alpha_i - \alpha_i^*)$
- can be written as:
  $max_{\beta_i} - \frac{1}{2}\sum_i \sum_j \beta_i \beta_j K(x_i, x_j) - \epsilon \sum_i |\beta_i| + \sum_i y_i \beta_i$
  s.t.
  - $\sum_i \beta_i = 0$
  - $\beta_i \in [-C, C], \forall i$
- Even for this form, standard QP (LCQP) solvers[2] can be used
- Question: How about (iteratively) solving for two β_i's at a time?
  - This is the idea of the Sequential Minimal Optimization (SMO) algorithm

[2] https://en.wikipedia.org/wiki/Quadratic_programming#Solvers_and_scripting_.28programming.29_languages

Sequential Minimal Optimization (SMO) for SVR

- Consider:
  $max_{\beta_i} - \frac{1}{2}\sum_i\sum_j \beta_i\beta_j K(x_i, x_j) - \epsilon\sum_i|\beta_i| + \sum_i y_i\beta_i$
  s.t.
  - $\sum_i \beta_i = 0$
  - $\beta_i \in [-C, C], \forall i$
- The SMO subroutine can be defined as:
  1. Initialise $\beta_1, \dots, \beta_n$ to some value $\in [-C, C]$
  2. Pick β_i, β_j to estimate closed form expression for next iterate (i.e. β_i^{new}, β_j^{new})
  3. Check if the KKT conditions are satisfied
     - If not, choose β_i and β_j that worst violate the KKT conditions and reiterate