Week 5 Note

Machine Learning Linear Regression

- Evaluating Regression Models
    - Common metrics for evaluating regression models
        - Coefficient of determination or $R^2 = 1 - \frac{\sum_i (y_i - \hat{y}_i)^2}{\sum_i (y_i - \bar{y})^2}$; $\bar{y}$ is the mean of the observed targets
        - Mean absoute error(MAE) = $\frac{1}{N}\sum_i^N |y_i - \hat{y}_i|$
        - Mean squared error(MSE) = $\frac{1}{N}\sum_i^n (y_i - \hat{y}_i)^2$
        - Root mean squared error(RMSE) = $\sqrt{\frac{1}{N}\sum_i^N (y_i - \hat{y}_i)^2}$
- Formalization:
    - Input: $\vec{x}$
    - Output: y
    - Target function: $f : X \to Y$
    - Data: $(\vec{x}_1, y_1), (\vec{x}_2, y_2), ..., (\vec{x}_N, y_N)$
    - Hypothesis: $g : X \to Y$
- Cost function
    - in-sample error: $E_{in}(h) = \frac{1}{N}\sum_{n=1}^{N}(h(\vec{x}_n) - y_n)^2 = \frac{1}{N}||X\vec{w} - y||^2$

> - Linear regression with linear and non-linear basis function
>   - Polynomial basis functions:
>
> $$w_0 + w_1 x_1^2 + w_2 x_2^2 + ... + w_D x_D^2$$
>
>   - Gaussian basis functions/radial basis functions
>
> $$\phi_j(x) = e^{-\frac{1}{2\sigma^2}(x-\mu_j)^2}$$
>
>   - Sigmoidal basis functions
>
> $$g(\alpha) = \frac{1}{1 + e^{-\alpha}}$$
>
>   - tanh basis functions
>
> $$h(\alpha) = \frac{e^{2\alpha} - 1}{e^{2\alpha} + 1}$$

Summary

- Linear and non-linear basis functions may be used to formulate a linear regression function
- OLS used to estimate linear regresssion weights by minimising sum of squared residuals
- OLS solution boild down to computing pseudoinverse of the Design Matrix
- Linear regression models can be fit to data using gradient descent

Machine Learning SVM Regression

- Support Vector Regression
  - Find a function, $f(x)$ with at most ϵ-deviation from the target y
  - We don't care about errors as long as they are less than ϵ
  - Only the pint ouside the ϵ-region contribute to the final cost

$$
\begin{gathered}
\min \frac{1}{2}||w||^2 \\
s.t. y_i - w_1 x_i - b \leq \epsilon; \\
w_1 x_i + b - y_i \leq \epsilon;
\end{gathered}
$$

$$
\begin{gathered}
J(w) = \underbrace{\frac{1}{2} w' w}_{\text{正则化防过拟合}} + C \sum_1^N (\xi + \xi *); \\
y_i - (x_i w + b) \leq \epsilon + \xi_i \\
(x_i w + b) - y_i \leq \epsilon + \xi_i^* \\
\xi^* \leq 0 \\
\xi_i \leq 0
\end{gathered}
$$

- Hyperparameter C
  - As C increases, our tolerance for points outside of ϵ also increases
  - As C approaches 0, the tolerance approaches 0 and the quation collapes into the simplified(although sometimes infeasible) one

> 希望确保模型的预测值落在真实值的一个ε区域内，或者至少尽可能地靠近这个区域。slack variables ξ 和 ξ^* 允许我们有一些灵活性，即当预测值与真实值之间的差异大于ε时，它们会吸收这种差异。

- Optimizing the Lagrangian

$$
\begin{aligned}
L := \frac{1}{2}||w||^2 + C \sum_{i=1}^{l} (\xi_i + \xi_i^*) - \sum_{i=1}^{l} (\mu_i \xi_i + \mu_i^* \xi_i^*) \\
- \sum_{i=1}^{l} \alpha_i (\epsilon + \xi_i - y_i + \langle w, x_i \rangle + b) \\
- \sum_{i=1}^{l} \alpha_i (\epsilon + \xi_i^* + y_i - \langle w, x_i \rangle - b)
\end{aligned}
$$

Lagrange multipliers $\alpha_i^{(*)}, \mu_i^{(*)} \leq 0$

- Optimizing the Lagrangian
    - The partial derivatives of L with respect to the variables

$$\delta_b L = \sum_{i=1}^{l}(\alpha_i^* - \alpha_i) = 0$$

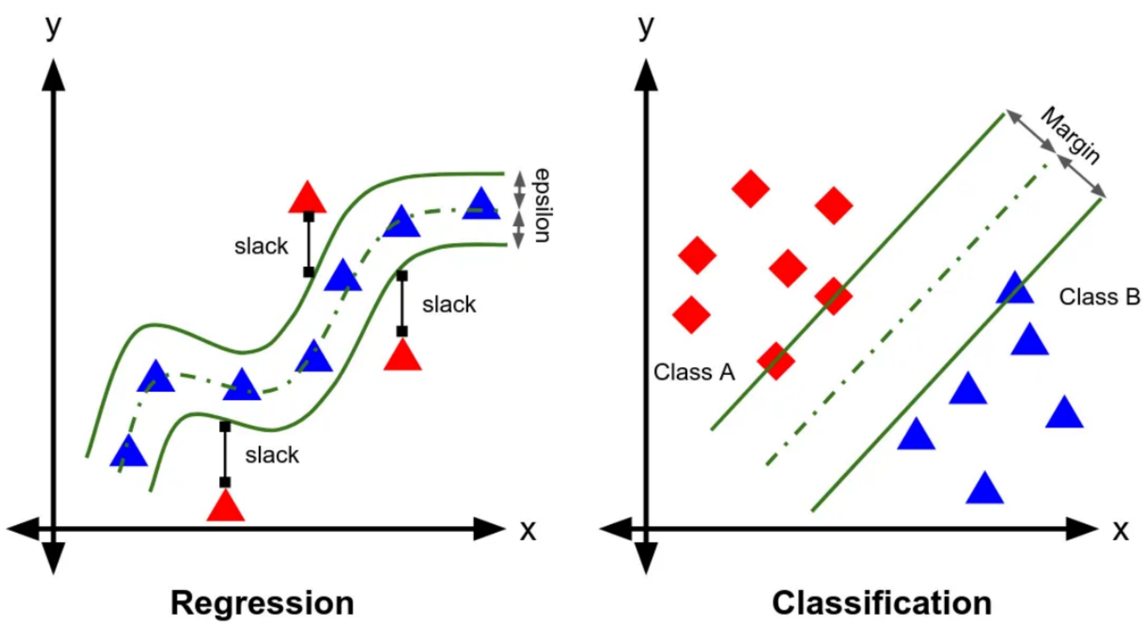
$$\delta_w L = w - \sum_{i=1}^{l}(\alpha_i - \alpha_i^*)x_i = 0$$

$$\delta_{\xi_i^{(*)}} L = C - \alpha_i^{(*)} - \mu_i^{(*)} = 0$$

$$maximize \begin{cases} -\frac{1}{2}\sum_{i,j=1}^{l}(\alpha_i - \alpha_i^*)(\alpha_j - \alpha_j^*)\langle x_i, x_j\rangle \\ -\epsilon\sum_{i=1}^{l}(\alpha_i + \alpha_i^*) + \alpha_{i=1}^{l} y_i(\alpha_i - \alpha_i^*) \end{cases}$$

Subject to $\sum_{i=1}^{l}(\alpha_i - \alpha_i^*) = 0$ and $\alpha_i, \alpha_i^* \in [0, C]$

- SVM: Regression vs Classification



Summary

- Linear regression tries to minimize the error between the real and predicted value
- SVR tries to fit the best line within a threshold value
- The threshold value is the distance between the hyperplane and boundary line
- Observations within the threshold of epsilon produce no error, only the observation outside the epsilon range produce error - sparse kernel machines