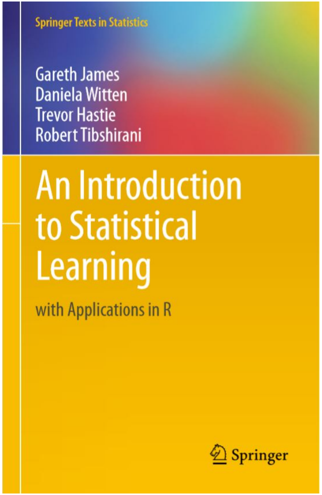
깊게 배우는 머신러닝

Ch.9 Support Vector Machines

훈 러닝 (Hun Learning)

January 22, 2020

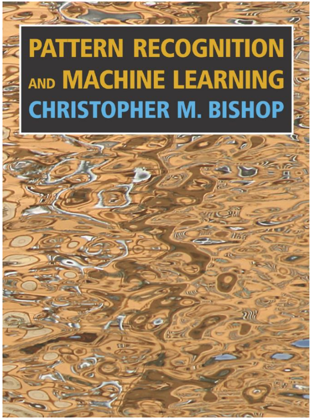
TEXTBOOK

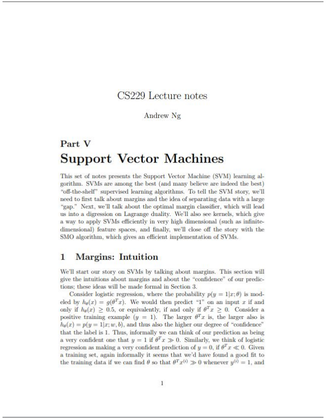


- **An Introduction to Statistical Learning : with Applications in R**
- 목차:
  1. Intro
  2. Statistical Learning
  3. Linear Regression
  4. Classification
  5. Resampling Methods
  6. Linear Model Selection and Regularization
  7. Moving Beyond Linearity
  8. Tree-based Methods
  9. Support Vector Machines
  10. Unsupervised Learning

TEXTBOOK

PRML 책과 Stanford CS229 Machine Learning (http://cs229.stanford.edu/) 강의안 참조.





CS229 Lecture notes

Andrew Ng

Part V

Support Vector Machines

This set of notes presents the Support Vector Machine (SVM) learning algorithm. SVMs are among the best (and many believe are indeed the best) "off-the-shelf" supervised learning algorithms. To tell the SVM story, we'll need to first talk about margins and the idea of separating data with a large "gap." Next, we'll talk about the optimal margin classifier, which will lead us into a digression on Lagrange duality. We'll also see kernels, which give a way to apply SVMs efficiently in very high dimensional (such as infinite-dimensional) feature spaces, and finally, we'll close off the story with the SMO algorithm, which gives an efficient implementation of SVMs.

1 Margins: Intuition

We'll start our story on SVMs by talking about margins. This section will give the intuitions about margins and about the "confidence" of our predictions; these ideas will be made formal in Section 3.

Consider logistic regression, where the probability $p(y = 1|x; \theta)$ is modeled by $h_\theta(x) = g(\theta^T x)$. We would then predict "1" on an input x if and only if $h_\theta(x) \geq 0.5$, or equivalently, if and only if $\theta^T x \geq 0$. Consider a positive training example ($y = 1$). The larger $\theta^T x$ is, the larger also is $h_\theta(x) = p(y = 1|x; w, b)$, and thus also the higher our degree of "confidence" that the label is 1. Thus, informally we can think of our prediction as being a very confident one that $y = 1$ if $\theta^T x \gg 0$. Similarly, we think of logistic regression as making a very confident prediction of $y = 0$, if $\theta^T x \ll 0$. Given a training set, again informally it seems that we'd have found a good fit to the training data if we can find θ so that $\theta^T x^{(i)} \gg 0$ whenever $y^{(i)} = 1$, and

1

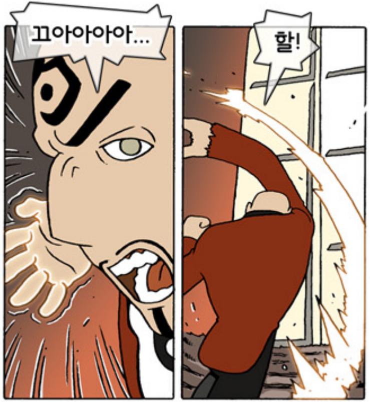
HYPERPLANE

Hyperplane이란?

- p차원 공간($X \in \mathbb{R}^{p\times 1}$)에서 Hyperplane은 전체 공간을 두 부분으로 나누는 평면(할!).
- 수학적으로 다음과 같이 정의한다.

$$y(X) = W^T X + b = 0$$

- SVM은 feature space를 이러한 hyperplane으로 두 공간으로 나누는 Classifier.
- feature mapping을 어떻게 정의하냐에 따라 직선은 물론 곡선, 원 등 다양한 boundary가 가능하다!



Hyperplane의 특성

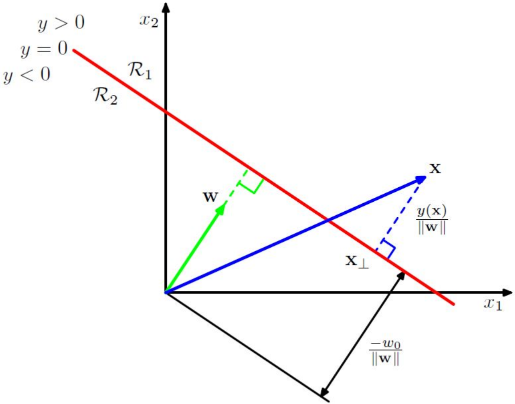
1. W: Hyperplane의 방향을 결정
   - **pf)** $y(X_A) = y(X_B) \to W^T(X_A - X_B) = 0$
2. b: 원점과 Hyperplane 간의 거리를 결정
   - **pf)** $proj_W(X) = \frac{W^T W}{\|W\|} = -\frac{b}{\|W\|}$
3. 임의의 벡터 A와 Hyperplane 간의 거리는 $\frac{y(A)}{\|W\|}$
   - **pf)**

$$A = A_\perp + \gamma \frac{W}{\|W\|}$$

$$W^T A + b = W^T A_\perp + b + \gamma \frac{W^T W}{\|W\|}$$

$$y(A) = 0 + \gamma \|W\|$$

$$\gamma = \frac{y(A)}{\|W\|}$$



Hyperplane과 Margin

$y_i \in \{-1, 1\}$ 의 예측에서, X의 Hyperplane을 이용한 다음의 classifier를 생각해보자.

$$h_{W,b}(X) = g(W^T X + b) \textbf{ where } g(z) = ifelse(1, -1, z \geq 0)$$

- **Margin:** γ_i를 임의의 관측치 X_i와 Hyperplane 사이의 거리는

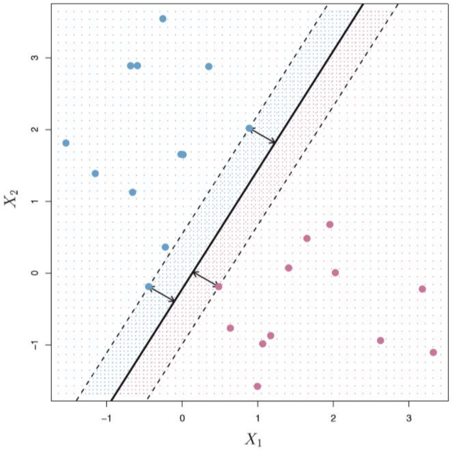
$$\gamma_i := y_i(\frac{W^T X_i + b}{\|W\|}) = y_i(\frac{W^T}{\|W\|} X_i + \frac{b}{\|W\|})$$

- X_i를 포함한 전체 데이터 셋과 Hyperplane 사이의 거리는 다음과 같이 정의하자. 즉 가장 가까운 놈을 기준으로 잰다는 것.

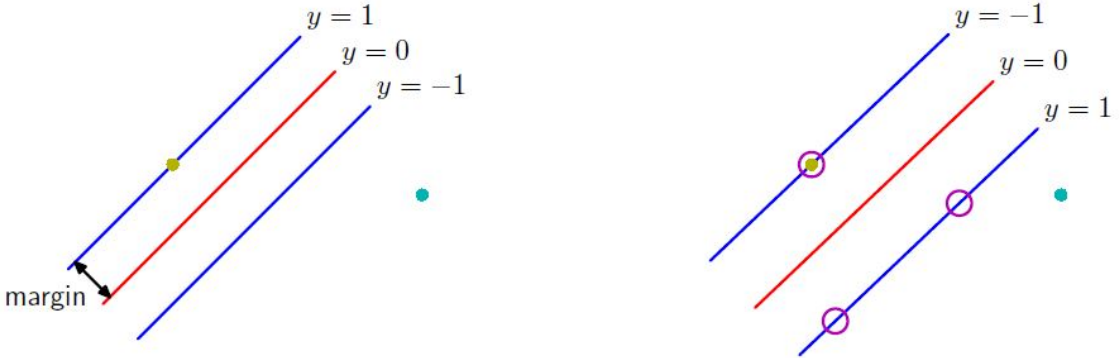
$$\gamma := \min_i y_i(\frac{W^T}{\|W\|} X_i + \frac{b}{\|W\|})$$

이렇게 정의한 거리를 이용해 Optimal Margin Classifier를 구해보자.

Optimal Margin Classifier



Optimal Margin Classifier



Optimal Margin Classifier

- 전체 데이터 셋과 거리가 가장 먼 Hyperplane을 구하는 것은 다음과 같다.

$$\arg\max_{W,b} \frac{1}{\|W\|} \min_i y_i(W^T X_i + b)$$

- 이때 W와 b의 길이를 쭉쭉 늘려도 $\gamma := \min_i y_i(\frac{W^T}{\|W\|}X_i + \frac{b}{\|W\|})$의 값은 바뀌지 않는다. 때문에 $\min_i y_i(W^T X_i + b) = 1$로 아예 고정해버리자. 그럼 위 식은 $\|W\|$의 값을 최소화하는 것이므로, 아래와 같이 다시 쓸 수 있다.

$$\begin{cases} \min_{W,b} & \frac{1}{2}\|W\|^2 \\ s.t. & y_i(W^T X_i + b) \geq 1 \quad {}^\forall i \in [N] \end{cases}$$

- Quadratic Programming을 이용해 이걸 만족하는 W^*, b^*를 구할 수는 있지만 시간이 많이 걸린다. 좀 더 빠른 방법을 찾기 위해서는 **Lagrange Duality**에 대해 알아야 한다.

LANGRAGE DUALITY

핵심은 최대화하고 최소화(Primal)하는 값이 최소화하고 최대화(Dual)하는 값과 같다는 것!

- 우리가 친숙한 Lagrangian 식은 다음과 같이 생겨먹었다. $(W \in \mathbb{R}^{N\times 1})$

$$\begin{cases} \min_W & f(W) \\ s.t. & h_i(W) = 0 \ \ {}^\forall i \in [l] \end{cases}$$

Lagrangian: $L(W, \beta) = f(W) + \sum \beta_i h_i(W)$

Find: $\dfrac{\partial L}{\partial W_i} = 0, \quad \dfrac{\partial L}{\partial \beta_i} = 0$

Generalized Lagrangian

- 부등식 조건이 추가된 다음과 같은 Primal Optimization Problem을 생각해보자.

$$\begin{cases} \min_W & f(W) \\ s.t. & g_i(W) \leq 0 \quad {}^\forall i \in [k] \\ & h_i(W) = 0 \quad {}^\forall i \in [l] \end{cases}$$

General Lagrangian: $L(W, \alpha, \beta) = f(W) + \sum \alpha_i g_i(W) + \sum \beta_i h_i(W)$

- $\Theta_p(W) := \max_{\alpha \geq 0, \beta} f(W) + \sum \alpha_i g_i(W) + \sum \beta_i h_i(W)$ 라고 정의하자. 그러면

$$\Theta_p(W) = \begin{cases} f(W) & \text{if W satisfies all the primal constraints} \\ \infty & \text{o.w.} \end{cases}$$

Primal Optimization Problem: $\min_W \Theta_p(W) = \min_W \max_{\alpha \geq 0, \beta} L(W, \alpha, \beta) = \min_W f(W)$

Generalized Lagrangian

- $\Theta_D(W) := \min_W f(W) + \sum \alpha_i g_i(W) + \sum \beta_i h_i(W)$ 라고 정의하자. 그러면

Dual Optimization Problem: $\max_{\alpha \geq 0, \beta} \Theta_D(W) = \max_{\alpha \geq 0, \beta} \min_W L(W, \alpha, \beta)$

- Primal이 먼저 조건식 라그랑지 승수로 라그랑지 식을 최대화한 후 그걸 최소화하는 W를 찾는거라면, Dual은 먼저 라그링지 식을 W로 최소화한 후 그걸 최대화하는 조건식 라그랑지 승수를 찾는 것! 직관적으로 생각해보면

$$\max_{\alpha \geq 0, \beta} \min_W L(W, \alpha, \beta) \leq \min_W \max_{\alpha \geq 0, \beta} L(W, \alpha, \beta)$$

- 그러나 일정한 조건이 만족되면 위 식은 똑같다! 즉 Dual을 만족하는 (W^*, α^*, β^*)이 라그랑지 식의 해가 되는 것![1]

[1] Convex Optimization 강의 참조 http://www.stat.cmu.edu/ ryantibs/convexopt/

Generalized Lagrangian

- **KKT 조건을 만족하는 경우 Dual과 Primal의 해는 같다.**
  (이는 f, g_i가 convex, h_i가 affine 함수, $g_i \leq 0$가 feasible인 조건과 동치)

$$\begin{cases} \min_W & f(W) \\ s.t. & g_i(W) \leq 0 \;\; {}^{\forall} i \in [k] \\ & h_i(W) = 0 \;\; {}^{\forall} i \in [l] \end{cases}$$

General Lagrangian: $L(W, \alpha, \beta) = f(W) + \sum^{k} \alpha_i g_i(W) + \sum^{l} \beta_i h_i(W)$

- Karush-Kuhn-Tucker(KKT) 조건은 다음과 같다.

$$\begin{cases} \textbf{Stationarity:} & \frac{\partial L}{\partial W_i} = 0 \\ \textbf{Complementary slackness:} & \alpha_i g_i(W) = 0 \;\; {}^{\forall} i \in [k] \\ \textbf{Primal feasibility:} & g_i(W) \leq 0, \; h_j(W) = 0 \;\; {}^{\forall} i \in [k], \; {}^{\forall} j \in [l] \\ \textbf{Dual feasibility:} & \alpha \geq 0 \;\; {}^{\forall} i \in [k] \end{cases}$$

Optimal Margin Classifiers

- Optimal Margin Classifiers의 조건은 다음과 같다. (Primal의 해와 똑같음)

$$\begin{cases} \min_{W,b} & \frac{1}{2}\|W\|^2 \\ s.t. & y_i(W^T X_i + b) \geq 1 \ \ ^\forall i \in [n] \\ (i.e. & 1 - y_i(W^T X_i + b) \leq 0) \end{cases}$$

Lagrangian: $L(W, b, \alpha, \beta) = \frac{1}{2}\|W\|^2 + \sum^n \alpha_i[1 - y_i(W^T X_i + b)]$

- 이 경우 KKT 조건은 다음과 같다.

$$\begin{cases} \textbf{Stationarity:} & \frac{\partial L}{\partial W_i} = \frac{\partial L}{\partial b} = 0 \\ \textbf{Complementary slackness:} & \alpha_i[1 - y_i(W^T X_i + b)] = 0 \ \ ^\forall i \in [n] \\ \textbf{Primal feasibility:} & 1 - y_i(W^T X_i + b) \leq 0 \ \ ^\forall i \in [n] \\ \textbf{Dual feasibility:} & \alpha \geq 0 \ \ ^\forall i \in [n] \end{cases}$$

OPTIMAL MARGIN CLASSIFIERS

Optimal Margin Classifiers

- KKT 조건을 만족한다고 가정하면 먼저 Dual $\Theta_D(W, b) = \min_{W,b} L(W, b, \alpha, \beta)$을 구한 후, $\Theta_D(W, b)$ 를 최소화하는 α 를 찾는 문제로 바뀐다.

$$\begin{cases} \max_\alpha & \sum^n \alpha_i - \frac{1}{2}\sum^n_{i,j} y_i y_j \alpha_i \alpha_j \langle X_i, X_j \rangle \\ s.t. & \alpha_i \geq 0 \ \ {}^\forall i \in [n] \\ & \sum^n \alpha_i y_i = 0 \end{cases}$$

- 이렇게 α^*를 구하고 나면 W^*는 $\frac{\partial L}{\partial W_i} = 0$ 인 조건을 이용해 구할 수 있으며, $b*$는 Hyperplane의 위치를 생각해보면 아래처럼 쉽게 구할 수 있다.

$$W^* = \sum^n \alpha_i y_i X_i$$

$$b* = -\frac{1}{2}\left(\max_{i|y_i=-1} W^{*T} X_i + \min_{i|y_i=1} W^{*T} X_i \right)$$

Optimal Margin Classifiers

- 때문에 α_i를 구했다면 임의의 점 X_h이 1 혹은 -1로 분류되는지는 오직 **새로운 데이터와 기존 데이터 셋 간의 내적에 의해** 전적으로 결정되는 것!

$$
\begin{aligned}
W^T X_h + b &= \left(\sum^n \alpha_i y_i X_i\right)^T + b \\
&= \sum^n \alpha_i y_i \langle X_i, X_h \rangle + b
\end{aligned}
$$

- 이 얘기를 하려고 Lagrange Duality까지 들먹이며 이 고생을 한거다. 이게 왜 중요하냐면 나중에 Kernel을 도입하면 저 내적 자리에 Kernel만 쏙 넣으면 논 리니어한 바운더리가 생겨나기 때문!

KERNEL FOR NON-LINEARITY

때로는 Hyperplane이 직선이 아닐 경우도 있다. 이 경우는 어떻게 하나?

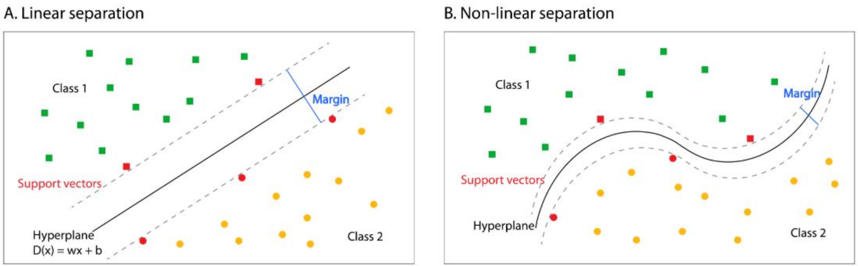


Illustration of separating two classes using SVMs. Linear (A.) and non-linear (B.) perfect separation of two classes (green and orange) with a hyperplane (black) and maximal margin (blue and dotted gray lines). Support vectors defining the hyperplane are in red. No misclassifiactions or margin violations are included. [2]

[2] http://www.coxdocs.org/doku.php?id=perseus:user:activities:matrixprocessing:learning:classificationparameteroptimization

KERNEL FOR NON-LINEARITY

- LDA에서도 $log(\frac{p}{1-p}) = \beta_0 + \beta_1 X_1 + +\beta_2 X_2 + \beta_3 X_1^2 + \beta_4 X_2^2$처럼 차수를 더해주면 (X_1, X_2) 평면에서 곡선인 boundary를 그릴 수 있다고 했다. SVM에서도 마찬가지이다. feature를 확장하면 곡선, 원형 등 다양한 boundary를 그릴 수 있다.
- Original input인 X를 **"input attributes"**, 이를 $X, X^2, X^3, ...$ 등으로 확장한 것을 **"input features"** 라고 하자. 그러면 attribute에서 feature로 이어주는 **"Feature Mapping"**은:

$$\phi(X) = [X, X^2, X^3, ...]^T$$

- Nonlinear한 SVM을 위해서는 X를 그냥 $\phi(X)$로 바꿔주기만 하면 끝이다. 사실 이것도 필요없이

$$W^T X_h + b = \sum^n \alpha_i y_i \langle X_i, X_h \rangle + b$$

$$\rightarrow W^T \phi(X_h) + b = \sum^n \alpha_i y_i \langle \phi(X_i), \phi(X_h) \rangle + b$$

즉, $\langle \phi(X_i), \phi(X_h) \rangle$만 알면 되는 것이다.

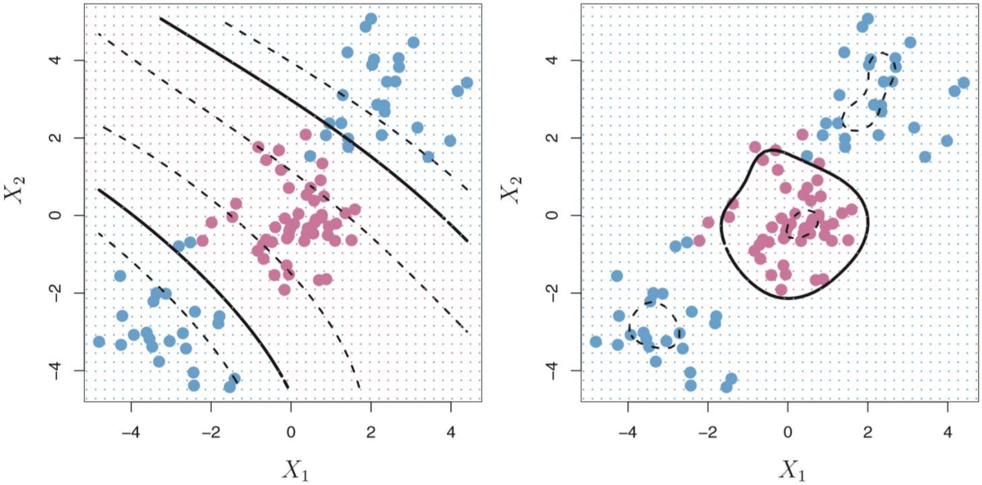
KERNEL FOR NON-LINEARITY

- Feature Mapping $\phi(X) : \mathbb{R}^p \rightarrow \mathbb{R}^P$이 주어졌을 때 **Kernel**은:

$$K(X,Z) = \phi(X)^T \phi(Z)$$

- Non-linear SVM에서 $\langle \phi(X_i), \phi(X_h) \rangle$를 계산할 때는 그냥 $K(X,Z)$만 계산하면 된다. 즉 $\phi(X)$의 정확한 형태를 몰라도 된다는 것!
  - 예를 들어 $K(X,Z) = (X^T Z)^2$로 정의했을때 $K(X,Z) = (\sum X_i Z_i)^2 = \sum_{i,j} (x_i x_j)(z_i z_j)$이므로 이에 대응되는 feature는 $\phi(X) = [x_1x_1, x_1x_2, x_2x_1, x_2x_2]^F$이다(n=2일때). feature를 구하고 계산하려면 $\mathcal{O}(n^2)$의 연산이 소요되나 Kernel로 바로 구하면 $\mathcal{O}(n)$의 연산으로 끝.
- 이걸 알면 반대로 생각해서 1) 먼저 Kernel을 정의하고 2) 이 Kernel이 말이 되는지 안 되는지 (여기에 대응하는 feature가 있는지 없는지)를 생각할 수 있다. 이렇게 하면 좋은 점은 먼저 feature 에 대해 고민할 필요 없이 **Kernel을 어떻게 잡냐에 따라 다양한 boundary가 직접적으로 결정할 수 있는 것이다.**
  - **Polynomial Kernel:** 곡선 형태의 바운더리, $K(X,Z) = (1 + \sum_i^p X_i Z_i)^d$
  - **Radial Kernel:** 원 모양의 바운더리, $K(X,Z) = exp(-\gamma \sum_i^p (X_i - Z_i))^2)$

KERNEL FOR NON-LINEARITY

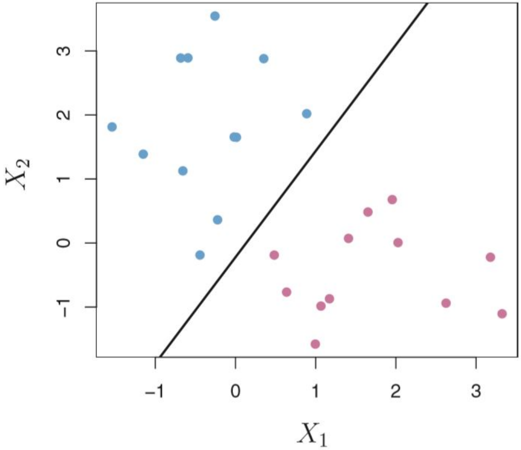


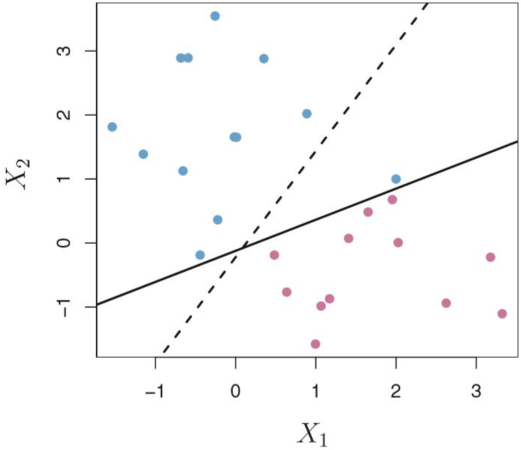
Mercer Theorem

- 사족이지만 Kernel은 원래의 정의를 생각하면 두 벡터 간의 각도 쯤으로 직관적으로 생각할 수 있다. 직교하면 값이 작아지고, 가까우면 값이 커지는 느낌.
- **Mercer Theorem:**
  어떤 $m(<\infty)$개의 점으로 이뤄진 (training set말고도 아무거나 상관 없음) $\{X_1, X_2, ...X_m\}$에 대해 $K_{ij} = K(X_i, X_j)$로 정의한 행렬을 Kernel Matrix이라고 한다. **Kernel에 대응하는 feature가 존재하기 위해서는 Kernel Matrix이 1) Symmetric 2) Positive semi definite** 이어야 한다. 그 반대도 성립한다(필요충분).
  1. $K(X_i, X_j) = K(X_j, X_i)$이어야 하므로 당연
  2. 임의의 벡터 Z에 대해
     $Z^T KZ = \sum_i \sum_j Z_i K_{ij} Z_j$
     $= \sum_i \sum_j Z_i \sum_k \phi_k(X_i)\phi_k(X_j)Z_j$
     $= \sum_k \sum_i \sum_j Z_i\phi_k(X_i)\phi_k(X_j)Z_j = \sum_k (\sum_i z_i\phi_k(x_i))^2 \geq 0$
- Kernel의 의의는 **feature를 무지막지하게 크게 확장시켜도 연산이 가능하다는 것!**

REGULARIZATION FOR ERROR TOLERANCE

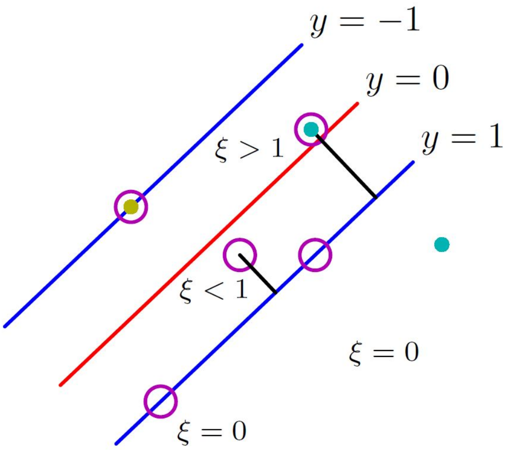
에러를 하나도 허용하지 않으면 (즉 hyperplane으로 완벽하게 나누려면) Outlier에 크게 영향을 받으며, 데이터마다 fitting이 많이 달라진다.





REGULARIZATION FOR ERROR TOLERANCE

각 관측치마다 약간의 오차 ϵ를 허용하지만 전체 오차를 제한하는 "soft-penalizing errors"을 해보자.



REGULARIZATION FOR ERROR TOLERANCE

- soft-penalizing을 하는 문제는 다음과 같다. (일종의 l^1 *Regularization*)

$$\begin{cases} \min_{W,b,\epsilon} & \frac{1}{2}\|W\|^2 + C\sum_i^N \epsilon_i \\ s.t. & y_i(W^T X_i + b) \geq 1 - \epsilon_i \\ & \epsilon_i \geq 0 \end{cases}$$

Lagrangian: $L(W, b, \epsilon, \alpha, \gamma) = \frac{1}{2}\|W\|^2 + C\sum^N \epsilon_i + \sum^N \alpha_i(1 - \epsilon_i - y_i(W_i^T + b)) + \sum^N \gamma_i(-\epsilon_i)$

- 이 경우 KKT 조건은 다음과 같다.

$$\begin{cases} \textbf{Stationarity:} & \frac{\partial L}{\partial W_i} = \frac{\partial L}{\partial b} = \frac{\partial L}{\partial \epsilon_i} = 0 \\ \textbf{Complementary slackness:} & \alpha_i[1 - y_i(W^T X_i + b)] = 0,\ \gamma_i \epsilon_i = 0\ \ {}^\forall i \in [n] \\ \textbf{Primal feasibility:} & 1 - y_i(W^T X_i + b) \leq 0,\ -\epsilon_i \leq 0\ \ {}^\forall i \in [n] \\ \textbf{Dual feasibility:} & \alpha \geq 0,\ \gamma_i \geq 0\ \ {}^\forall i \in [n] \end{cases}$$

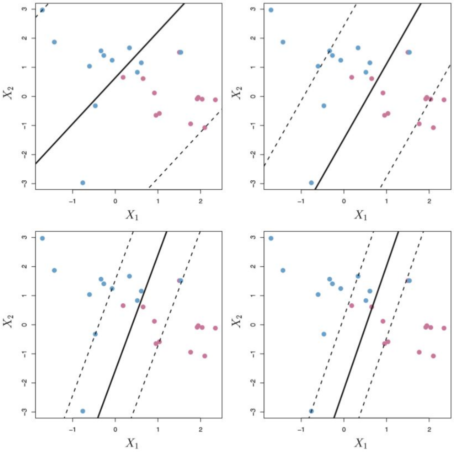
REGULARIZATION FOR ERROR TOLERANCE

- 먼저 Stationarity 조건을 $L(W, b, \epsilon, \alpha, \gamma)$에 대입해 Dual $\min_{W,b,\epsilon} L$을 구한 후 나머지 KKT 조건을 이용하면 다음의 문제로 바뀐다.

$$\begin{cases} \max_\alpha & \sum^N \alpha_i - \frac{1}{2}\sum_{i,j}^N y_i y_j \alpha_i \alpha_j \langle X_i, X_j \rangle \\ s.t. & 0 \le \alpha_i \le C \\ & \sum^N \alpha_i y_i = 0 \end{cases}$$

- $W^* = \sum^n \alpha_i y_i X_i$로 똑같이 구하고, b^*는 조금 다르지만 결국 비슷하다.
- (Slackness) $\alpha_i[1 - \epsilon_i - y_i(W^T X_i + b)] = 0, \ \gamma_i \epsilon_i = 0 \ \ ^\forall i \in [n]$를 생각하면
  1. if $\alpha_i > 0$, then $y_i(W^T X_i + b) = 1 - \epsilon_i$ (SVM에 기여하는 Support Vectors)
     - ★ if $\alpha_i < C$, since $\frac{\partial L}{\partial \epsilon_i} \to \alpha_i = C - \gamma_i$, then $\gamma_i > 0$ which in turn means $\epsilon_i = 0$ since $\gamma_i \epsilon_i = 0$.
     - ★ if $\alpha_i = C$, since $\alpha_i[1 - \epsilon_i - y_i(W^T X_i + b)] = 0, \to y_i(W^T X_i + b) = 1 - \epsilon_i$, which is correctly classified if $\epsilon_i \le 1$ and misclassifed if $\epsilon_i > 1$.
  2. if $\alpha_i = 0$, then these points do not contribute to the model since $W^* = \sum^n \alpha_i y_i X_i$.

C는 tuning parameter. C가 클수록 에러에 대한 패널티를 세게 주는 것이므로 마진이 점점 더 좁아져 어떤 에러도 허용하지 않게 된다. 적정 C는 당연히 CV-error로 구한다.



SVM VS LOGIT REGRESSION

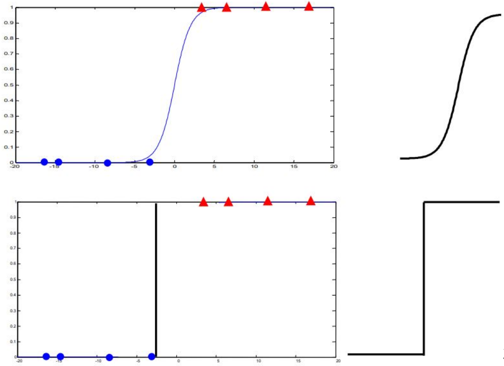
Logistic Regression Revisited

- $y_i \in \{-1, +1\}$에서 LR Classifier $W^T X + b$는

$$\sigma(W^T X + b) \begin{cases} \geq 0.5 \rightarrow y_i = +1 \\ < 0.5 \rightarrow y_i = -1 \end{cases}$$

$$where \quad \sigma(f(X)) = \frac{1}{1 + \exp - f(X)}$$

- $\sigma(0) = 0.5$
- Let $Z = W^T X + b$, and we have $\| \frac{\partial \sigma(Z)}{\partial (X)} \|_{Z=0} = \frac{1}{4} \|W\|$



[3]

[3] http://www.robots.ox.ac.uk/ az/lectures/ml/2011/lect4.pdf

SVM VS LOGIT REGRESSION

- LR Classifier의 fitting은 MLE

$$\begin{cases} P(y=1|X) & = \sigma(f(X)) = \frac{1}{1+e^{-f(X)}} \\ P(y=-1|X) & = 1-\sigma(f(X)) = \frac{1}{1+e^{+f(X)}} \end{cases}$$

$$\rightarrow P(y_i|X_i) = \frac{1}{1+e^{-y_i f(X_i)}}$$

- iid sampling을 가정하면 전체 모델의 Joint Likelihood는

$$Joint\ Likelihood \quad \prod_{i}^{N} \frac{1}{1+e^{-y_i f(X_i)}}$$

$$Negative\ Log\ Likelihood \quad \sum_{i}^{N} \log\left(1+e^{-y_i f(X_i)}\right)$$

이 식이 바로 **LR Classifier의 Loss Function!**

SVM VS LOGIT REGRESSION

- 즉 OLS를 Loss + Penalty로 재해석해 Lasso, Ridge 등 Regularization을 한 것처럼 LR Classifier도 다음의 문제로 다시 쓸 수 있다.

$$\min_{W \in \mathbb{R}^d} \underbrace{\sum_{i}^{N} log(1 + e^{-y_i f(X_i)})}_{\text{loss function}} + \underbrace{\lambda \|W\|^2}_{\text{regularization}}$$

- 분류가 맞으면 $y_i f(X_i) > 0 \rightarrow log(1 + e^{-y_i f(X_i)})$가 낮음
- 분류가 틀리면 $y_i f(X_i) < 0 \rightarrow log(1 + e^{-y_i f(X_i)})$가 큼
- 이런 식으로 틀린 분류에 Loss를 주며, 모델의 복잡도를 l^2 $regularization$으로 조정하는 일종의 **Regularized Logit Regression을 생각해볼 수 있다.**

SVM VS LOGIT REGRESSION

- 에러가 허용된 SVM의 조건식을 다시 생각해보면

$$\begin{cases} \min_{W,\epsilon} & \frac{1}{2}\|W\|^2 + C\sum_i^N \epsilon_i \\ s.t. & y_i(f(X_i)) \geq 1 - \epsilon_i \\ & \epsilon_i \geq 0 \end{cases}$$

- 제대로 분류된 애들은 $\epsilon_i = 0$이므로 $\epsilon_i = 0 \geq 1 - y_i f(X_i)$일 것이다.
- 마진의 안쪽에 있거나 아예 분류가 틀린 애들은 $\epsilon_i > 1 - f(X_i) > 0$일 것이다. 따라서 위 조건식은 아래처럼 다시 쓸 수 있다.

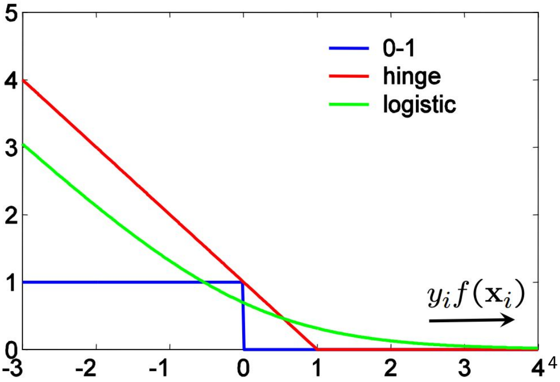
$$\min_W C\sum_i^N max[0, 1 - y_i f(X_i)] + \frac{1}{2}\|W\|^2$$

0-1 loss를 추정하는 서로 다른 loss function!

$$\min_{W \in \mathbb{R}^d} \underbrace{\sum_{i}^{N} log(1 + e^{-y_i f(X_i)})}_{} + \underbrace{\lambda \|W\|^2}_{}$$

$$\min_{W} \underbrace{C \sum_{i}^{N} max[0, 1 - y_i f(X_i)]}_{} + \underbrace{\frac{1}{2}\|W\|^2}_{}$$

- LR는 모든 데이터를 고려하지만, SVM은 마진 밖에 있으면 그냥 무시함. Loss f 형태 자체가 그렇게 생겨먹었다.
- 또한 SVM은 Kernel이 Mercer 조건만 만족하면 맘껏 그림을 그릴 수 있는 장점.



[4]

[4] http://www.robots.ox.ac.uk/ az/lectures/ml/2011/lect4.pdf