7.SPARSE KERNEL MACHINES

이전에 Non-linear kernels에 기반한 learning algorithms들을 배워봤다. 이런 알고리즘들의 한계는 kernel function이 트레이닝 셋 내의 모든 가능한 pair에 대해 계산을 해야한다는 것이다. 이는 새로운 데이터가 들어왔을 때 매우 비효율적이다.

여기선 sparse solution을 갖는 kernel-based algorithms을 배운다.

그래서 새로운 input에 대한 predictions이 train data의 subset에 의해서만 결정된다.

SVM으로 시작하자. Which is good for classification and novelty detection.

SVM is the determination of the model parameters corresponds to a convex optimization problem => local=global.

The SVM is a decision machine and so does not provide posterior probabilities.

To obtain the benefits of determining probabilities, an alternative sparse kernel technique, RVM, is based on a Bayesian formulation and provides posterior probabilistic outputs, much sparser solution than the SVM.

7.1 Maximum Margin Classifiers

To support vector machines, Let's go back to the binary classification using linear models $y(x) = w\top\phi(x) + b$

$\phi(x)$: A fixed feature-space transformation.

b: bias parameter

Let's see dual representation expressed in terms of kernel functions, which avoids having to work explicitly in feature space.

Input training vectors : $x_1, \ldots, x_N$,

Target values : $t_1, \ldots, t_N$ where $t_n \in \{-1, 1\}$

New data points x_{N+1} are classified according to the sign of $y(x_{N+1})$.

Assume that the training data set is linearly separable in feature space, So that by definition,

there exists "at least" one choice of the parameters w and b such that a function of the form satisfies

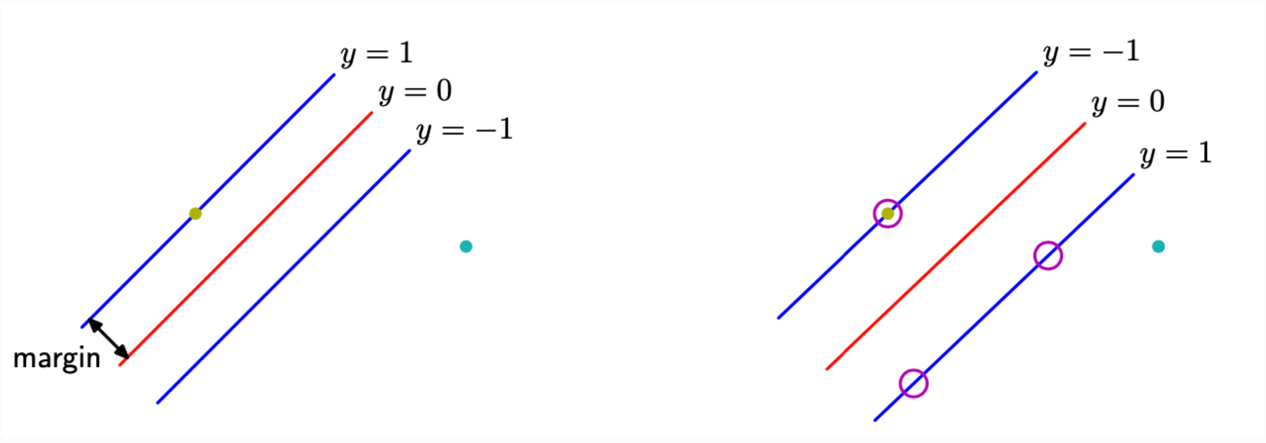
$y(x_n) > 0$ for points having $t_n = +1$, $y(x_n) < 0$ for points having $t_n = -1$

So that $t_n y(x_n) > 0$ for all training data points. There are another solutions.

In this case, we should try to find the one that will give the smallest generalization error.

The support vector machine approaches this problem through the concept of the margin,

Margin : the smallest distance between the decision boundaries.



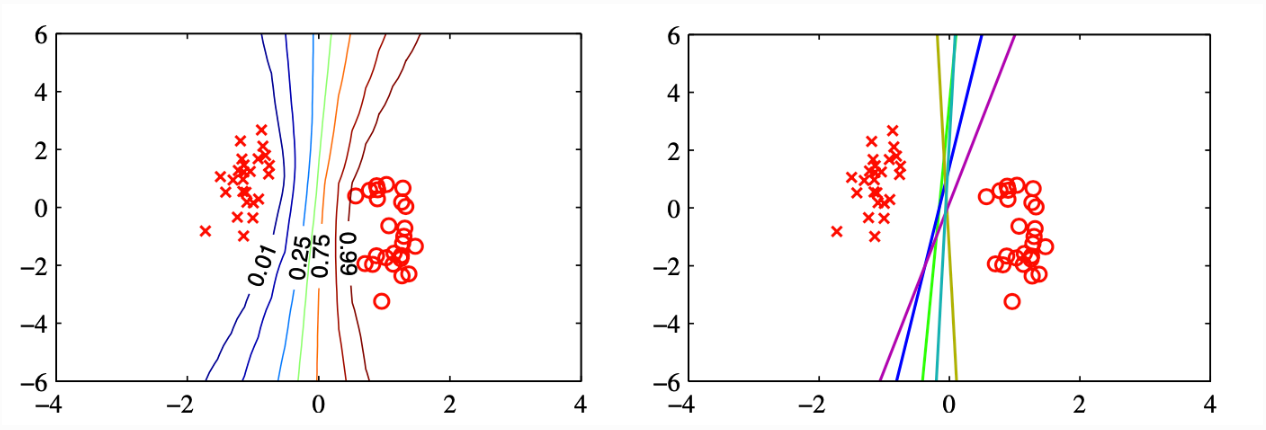
In SVM, the decision boundary is chosen to be the one for which the margin is maximized.

The maximum margin solution is motivated from computational learning theory(statistical learning theory).

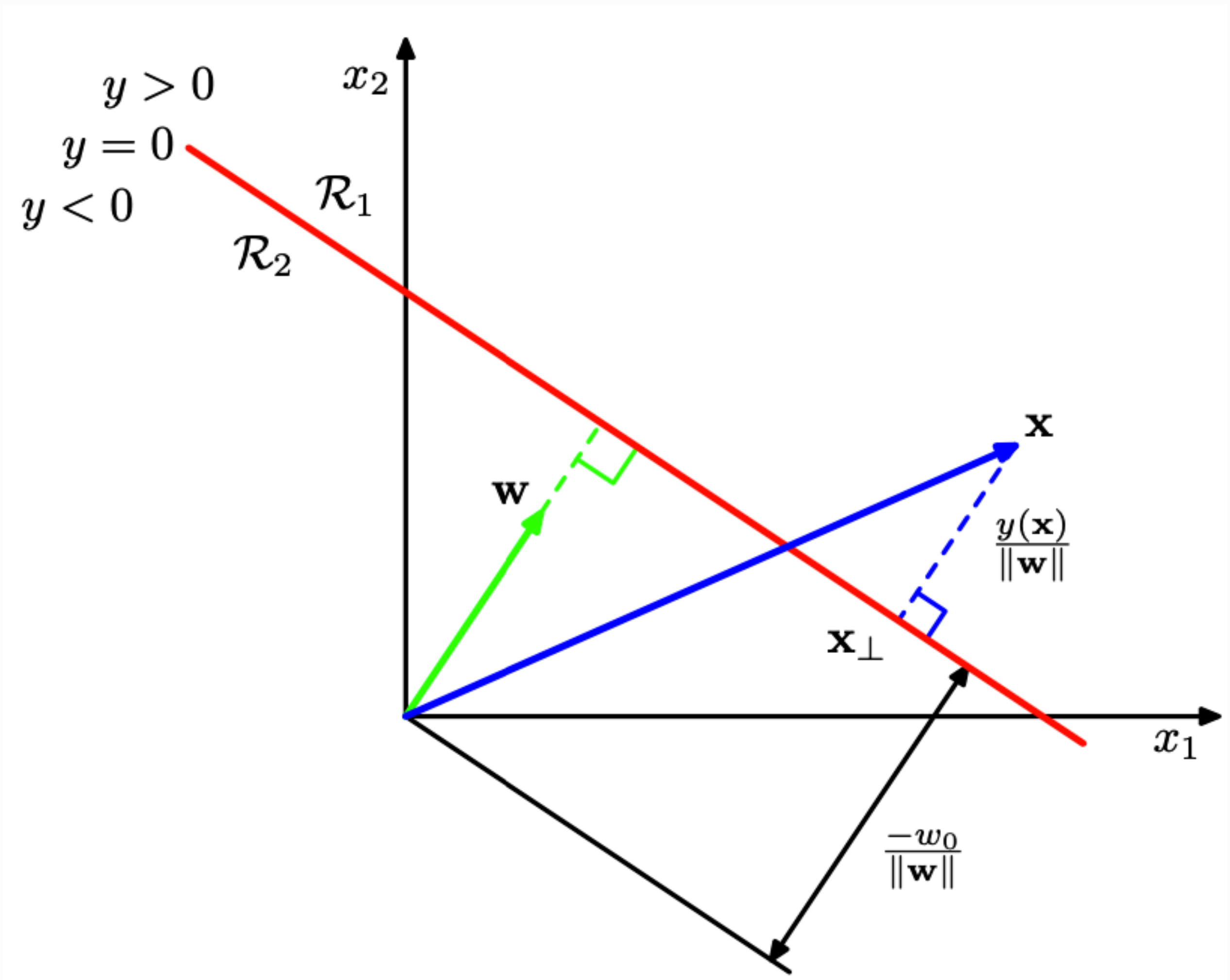
Original Insight of maximum margin is from 통 and 쿨러. They use Parzen density estimator with Gaussian kernels having a common parameter σ^2.

Together with the class priors, this defines an optimal misclassification-rate decision boundary as hyperplane.

Their approach had a limit that the hyperplane is independent of data points that are not support vectors.



That marginalization with respect to the prior distribution of the parameters in a Bayesian approach for a simple linearly separable data set leads to a decision boundary that lies in the middle of the region separating the data points. The large margin solution has similar behavior.



the Perpendicular distance of a point x from a hyperplane defined by $y(x) = 0$, is given by $|y(x)|/||w||$.

we are only interested in solutions for which all data points are correctly classified($t_n y(x_n) > 0$) for all n.

Thus the distance of a point x_n to the decision surface is given by

$$D(distance, margin) = \frac{t_n y(x_n)}{||w||} = \frac{t_n(w^\top \phi(x_n) + b)}{||w||}$$

Margin : the perpendicular distance to the closest point x_n from the data set to the decision surface,

Goal : Optimize the parameters w and b to maximize this distance(margin).

$$Goal \ : \ \arg\max_{w,b}\{\frac{1}{||w||}\min_n[t_n(w^\top \phi(x_n) + b)]\}$$

Direct solution of this optimization problem would be very complex.

So we shall convert it into an easier equivalent problem.

If we rescale $w \to \kappa w$ and $b \to \kappa b$,

$t_n((\kappa w)^\top \phi(x_n) + \kappa b)/||\kappa w|| \to t_n(\kappa w^\top \phi(x_n) + \kappa b)/||\kappa w|| \to \kappa t_n(w^\top \phi(x_n) + b)/|\kappa|||w||$

then the distance $t_n y(x_n)|||w||$ is unchanged.

We can use this freedom to set $t_n(w \top \phi(x_n) + b) = 1$.

In this case, all data points will satisfy the constraints $(7.5) \ \ t_n(w \top \phi(x_n) + b) \geq 1, \ \ n = 1, , \ldots, N$

(since margin is minimum distance)

This is known as "the canonical representation" of the decision hyperplane.

Data points that the equality holds, the constraints are said to be active,

Whereas for the remainder, the constraints are said to be inactive.

By definition, there will always be at least one active constraint,

because there will always be a closest point,

And once the margin has been maximized, there will be at least two active constraints.

The optimization problem then simply requires that we maximize $\frac{1}{||w||}$, which is equivalent to minimizing $||w||^2$

$$Equivalent \ Goal \ : \ \arg\min_{w,b} \frac{1}{2}||w||^2$$

subject to the constraints $t_n(w^\top \phi(x) + b) \geq 1$, $\frac{1}{2}$ is included for later convenience.

This is an example of a quadratic programming problem in which we are trying to

minimize a quadratic function subject to a set of linear inequality constraints.

It seems that the bias parameter b has disappeared from the optimization.

However, bias will be determined implicitly via the constraints,

because changes of $||w||$ compensated by changes of b .

We shall see how this works shortly.

In order to solve this constrained optimization problem,

we introduce Lagrange multipliers $a_n \geq 0$, with one Lagrange multiplier a_n ,as known as λ_n,

for each of the constraints in(7.5),

$L(w, b, a) = \frac{1}{2}||w||^2 - \sum_{n=1}^{N} a_n\{t_n(w\top\phi(x_n) + b) - 1\}$ Where $a = (a_1, \dots, a_N)^\top$.

The reason for calculating the difference of two term is that we are minimizing with respect to $w\, and\, b$, and maximizing with respect to a

Setting the derivatives of $L(w, b, a)$ with respect to w and b equal to zero,

we obtain the following two conditions.

$w = \sum_{n=1}^{N} a_n t_n \phi(x_n)$, $0 = \sum_{n=1}^{N} a_n t_n$.

Eliminating w and b from $L(w, b, a)$ using these conditions

then gives the dual representation of the maximum margin problem in which we maximize (since minus term)

$\tilde{L}(a) = \sum_{n=1}^{N} a_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{M=1}^{n} a_n a_m t_n t_m k(x_n, x_m)$

With respect to a subject to the constraints $a_n \geq 0, \;\; n = 1, \dots, N \sum_{n=1}^{N} a_n t_n = 0.$

Here the kernel function is defined by $k(x, x') = \phi(x)^\top \phi(x')$.

Again, this takes the form of a quadratic programming problem

in which we optimize a quadratic function of a subject to a set of inequality constraints. Details are in 7.1.1

There are some advantageous talk about this process and we will skip to classifying new data point

In order to classify new data points using the trained model, we evaluate the sign of $y(x) = w \top \phi(x) + b$

This can be expressed in terms of the parameters $\{a_n\}$ and the kernel function by substituting for$w = \sum_{n=1}^{N} a_n t_n \phi(x_n)$ $w^\top = \sum_{n=1}^{N} a_n t_n \phi^\top(x) \Rightarrow y(x) = \sum_{n=1}^{N} a_n t_n k(x, x_n) + b$.

In appendix E, we show that a constrained optimization of this form satisfies the KKT conditions,

which in this case require that the following three properties hold

$$
\begin{gathered}
a_N \geq 0 \\
t_n y(x_n) - 1 \geq 0 \\
a_n \{t_n y(x_n) - 1\} = 0
\end{gathered}
$$

Thus for every data point, either $a_n = 0$ "or" $t_n y(x_n) = 1$.

Any data point for which $a_n = 0$ will not appear in $y(x) = \sum_{n=1}^{N} a_n t_n k(x, x_n) + b$ so plays no role

in making predictions for new data points.

The remaining data points are called "support vectors", and because they satisfy $t_n y(x_n) = 1$, which means $y(x_n) = +1 \; or \; -1$ they correspond to points that lie on the maximum margin hyperplanes in feature space

"This property is central to the practical applicability of support vector machines"

Once the model is trained, amount of data points will be discarded

and only the support vectors will be remained.

Having solved the quadratic programming problem and found a value for a,

we can ten determine the value of the threshold parameter b

by noting that any support vector x_n satisfies $t_n y(x_n) = 1$.

$y(x) = \sum_{n=1}^{N} a_n t_n k(x, x_n) + b$ gives $t_n \left(\sum_{m \in S} a_m t_m k(x_n, x_m) + b\right) = 1$

S : the set of indices of the support vectors. Support vector 의 index set.

Just for the efficiency,

A numerically more stable solution is obtained by first multiplying through by t_n, making use of $t_n^2 = 1$,

and then averaging these equations over all support vectors and solving for b to give $b = \frac{1}{N_S} \sum_{m \in S}(t_n - \sum_{m \in S} a_m t_m k(x_n, x_m))$

N_S : the total number of support vectors.

For comparison between models, shortly see criterion of the maximum-margin classifier

in terms of the minimization of an error function, with a simple quadratic regularizer,

in the form $\sum_{n=1}^{N} E_\infty(y(x_n)t_n - 1) + \lambda||w||^2$

$E_\infty(z)$: 0 if $z \geq 0$, ∞ otherwise to satisfy $t_n(w\top\phi(x_n) + b) \geq 1$.

Note that as long as the regularization parameter satisfies $\lambda > 0$, its precise value plays no role.

Further part is a example of classification of synthetic data with $k(x, x') = exp(\frac{-||x-x'||^2}{2\sigma^2})$ kernel.

Let go to 7.1.1

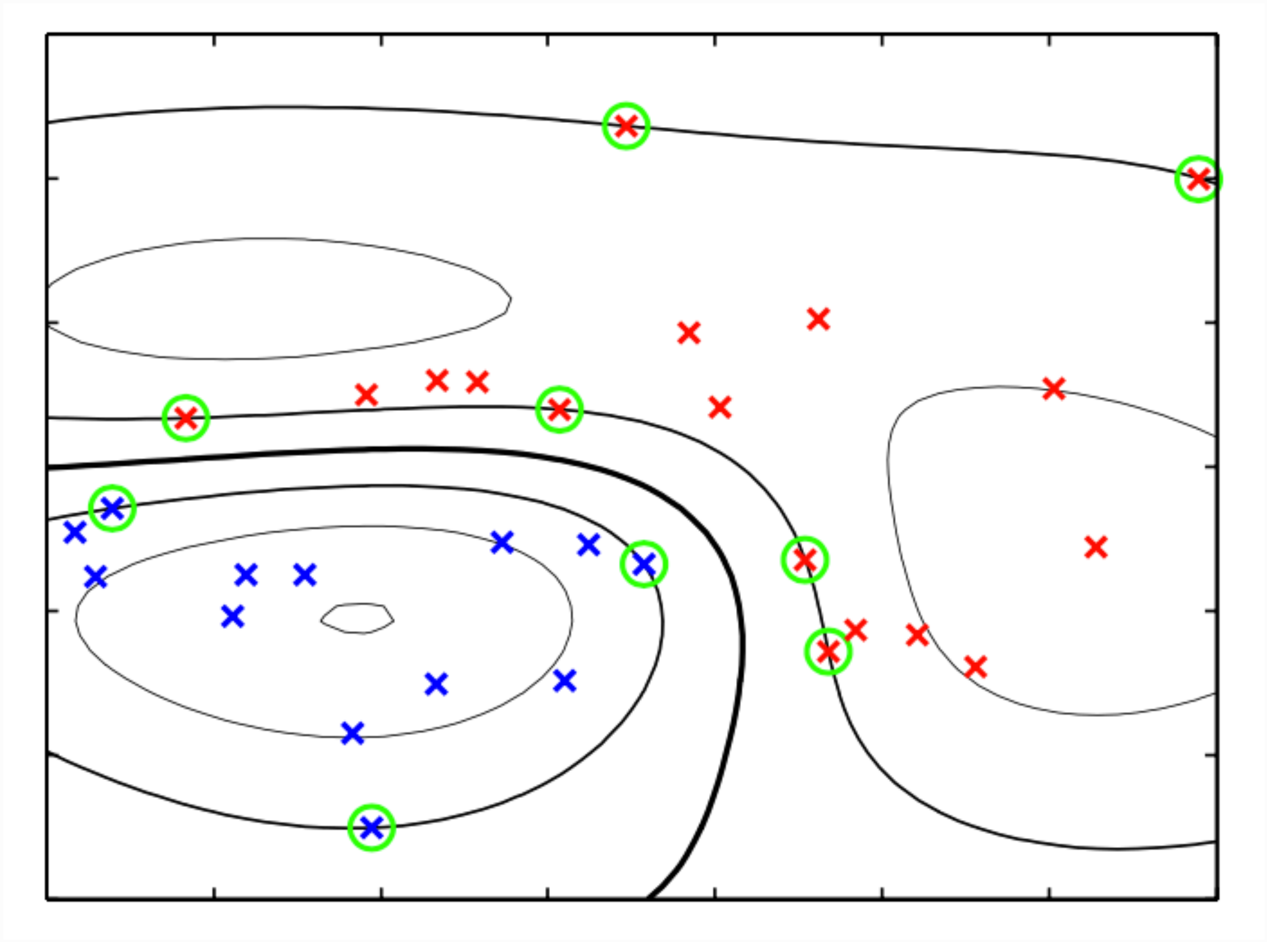


Figure shows an example of the classification resulting from training a support vector machine on a simple synthetic data set using a Gaussian kernel $k(x, x') = exp(\frac{-||x-x'||^2}{2\sigma^2})$. Although the data set is not linearly separable in the two-dimensional data space x, it is linearly separable in the nonlinear feature space defined implicitly by the nonlinear kernel function.

Thus the training data points are perfectly separated in the original data space.

This example also provides a geometrical insight into the origin of sparsity in the SVM.

The maximum margin hyperplane is defined by the location of the support vectors.

Other data points can be moved around freely (so long as they remain outside the margin region) without changing the decision boundary, and so the solution will be independent of such data points.