Chapter 9: Support Vector Machines

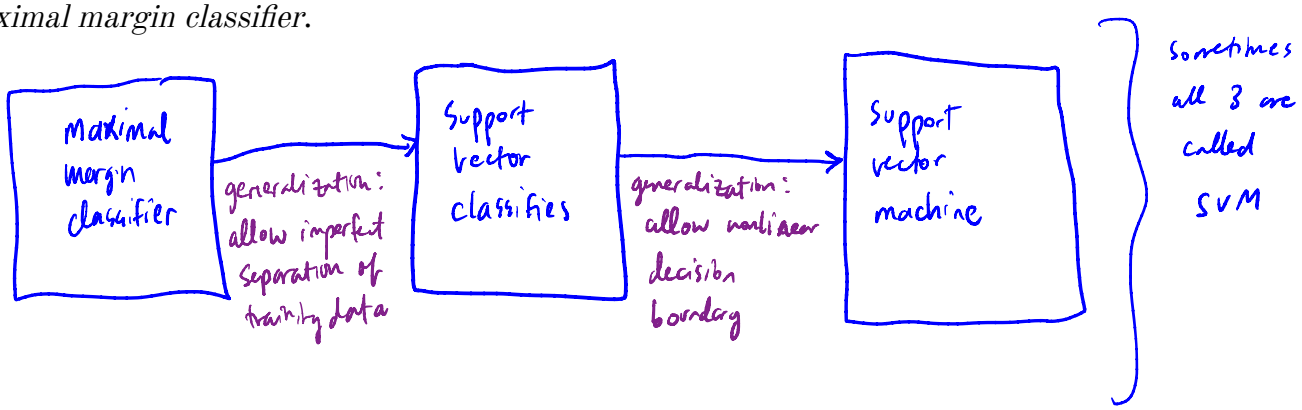
categorical response Y

The *support vector machine* is an approach for classification that was developed in the computer science community in the 1990s and has grown in popularity.

SVMs perform well in a variety of settings

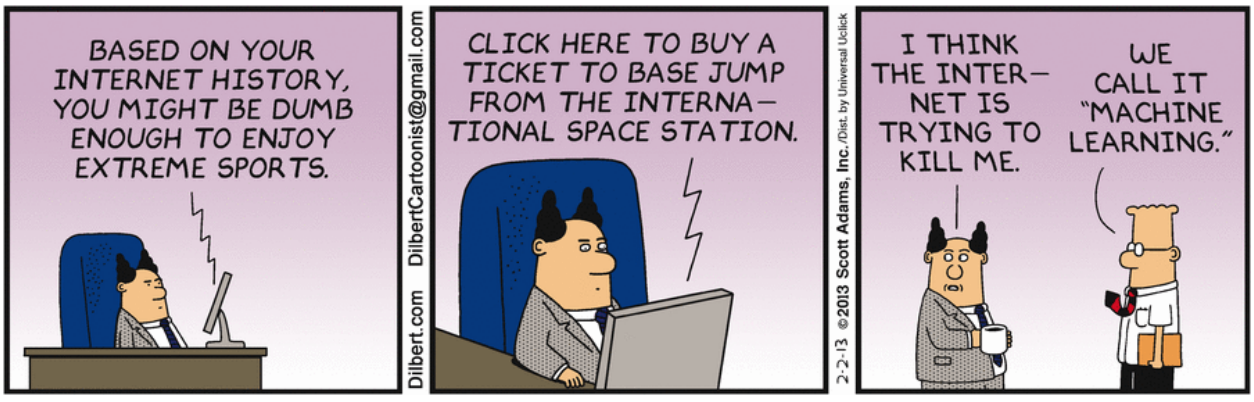
often considered one of the best "out of the box" classifiers.

The support vector machine is a generalization of a simple and intuitive classifier called the *maximal margin classifier*.



Support vector machines are intended for binary classification, but there are extensions for more than two classes.

categorical response Y with only 2 classes



Credit: https://dilbert.com/strip/2013-02-02

1 Maximal Margin Classifier

based on a hyperplane separator.

In p-dimensional space, a *hyperplane* is a flat affine subspace of dimension $p-1$.

extension of Euclidean space.

e.g. In 2 dim., a hyperplane is a flat 1 dim. subspace – a line.

In 3 dim., a hyperplane is a flat 2 dim. subspace – a plane

⋮

In $p>3$ dim., a hyperplane is harder to conceptualize, is still a flat $p-1$ dim. subspace.

The mathematical definition of a hyperplane is quite simple,

In 2 dim. a hyperplane is defined by $\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$

parameters

i.e. any $X = (X_1, X_2)$ for which this equation holds lies on the hyperplane.

NOTE: this is just the equation for a line. $\beta_0 + \beta_1 X_1 = -\beta_2 X_2$

$-\frac{\beta_0}{\beta_2} - \frac{\beta_1}{\beta_2} X_1 = X_2$

"b" "a" "x" "y"

This can be easily extended to the p-dimensional setting.

$\beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p = 0$ defines a p-dim. hyperplane.

i.e. any $X = (X_1, \ldots, X_p)$ for which this equation holds will lie on the hyperplane.

We can think of a hyperplane as dividing p-dimensional space into two halves.

if $\beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p > 0$ then $X = (X_1, \ldots, X_p)$ lies on one side of the hyperplane,

If $\beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p < 0$ then X lies on the other side of the hyperplane.

We can determine which side of the hyperplane by just determining the sign of $\beta_0 + \beta_1 X_1 + \ldots + \beta_p X_p$

1.1 Classificaton Using a Separating Hyperplane

Suppose that we have a $n \times p$ data matrix $\boldsymbol{X}$ that consists of n training observations in p-dimensional space.

training observations

$$x_1 = \begin{pmatrix} x_{11} \\ \vdots \\ x_{1p} \end{pmatrix}, \ldots, x_n = \begin{pmatrix} x_{n1} \\ \vdots \\ x_{np} \end{pmatrix}$$

and that these observations fall into two classes.

$$y_1, \ldots, y_n \in \{-1, 1\}$$

where -1 represents one class
and 1 represents the other.

We also have a test observation.

p-vector of observed features

$$x^* = (x_1^*, \ldots, x_p^*)^T$$

Our Goal: Develop a classifier based on training data that will correctly classify the test observation base on feature measurements.

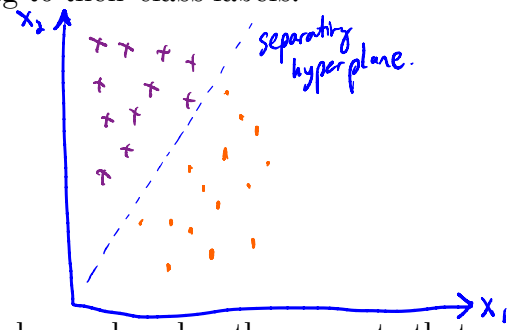
We have already used many approaches:

- classification trees
- random forest, bagging, boosting.
- logistic regression
- LDA

We will see a new approach using a separating hyperplane.

Suppose it is possible to construct a hyperplane that separates the training observations perfectly according to their class labels.

$p=2$



Then a separating hyperplane has the property that

$\beta_0+\beta_1 x_{i1}+\dots+\beta_p x_{ip} > 0$ if $y_i = 1$ and

$\beta_0+\beta_1 x_{i1}+\dots+\beta_p x_{ip} < 0$ if $y_i = -1$ for $i=1,\dots,n$.

$\Leftrightarrow$

$y_i(\beta_0+\beta_1 x_{i1}+\dots+\beta_p x_{ip}) > 0 \quad \forall\ i=1,\dots,n.$

If a separating hyperplane exists, we can use it to construct a very natural classifier:

A test observation is assigned a class depending on which side of the hyperplane it is located.

That is, we classify the test observation x^* based on the sign of $f(x^*) = \beta_0 + \beta_1 x_1^* + \dots + \beta_p x_p^*$.

if $f(x^*) > 0$ assign x^* to class 1

if $f(x^*) < 0$ assign x^* to class -1.

We can also use the magnitude of $f(x^*)$.

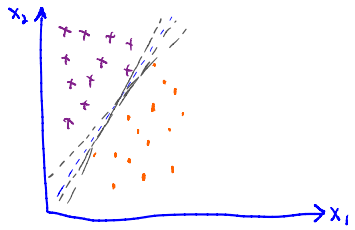
If $f(x^*)$ is far from zero, this means x^* lies far from the hyperplane
$\Rightarrow$ we can be confident about our class assignment for x^*.

If $f(x^*)$ is close to zero, this means x^* lies close to the hyperplane.
$\Rightarrow$ we are less confident about the class assignment for x^*

NOTE a classifier based on a separating hyperplane leads to a linear decision boundary.

1.2 Maximal Margin Classifier

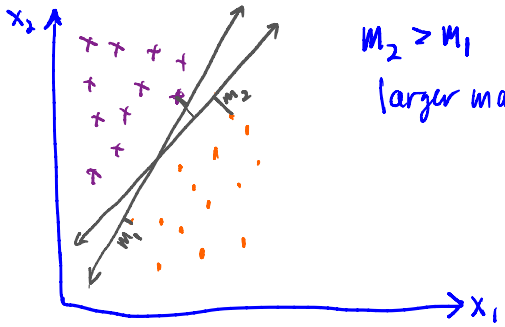
If our data can be perfectly separated using a hyperplane, then there will exist an infinite number of such hyperplanes.



A given hyperplane can usually be shifted up/down or rotated slightly without coming into contact with any observations. => Which one should we base our classifier on?

A natural choice for which hyperplane to use is the *maximal margin hyperplane* (aka the *optimal separating hyperplane*), which is the hyperplane that is farthest from the training observations.

- We compute the perpendicular distance from each observation to a given separating hyperplane.
- the smallest distance is known as the "margin"
- the maximal margin hyperplane is the one w/ the largest margin, i.e. farthest from all training points

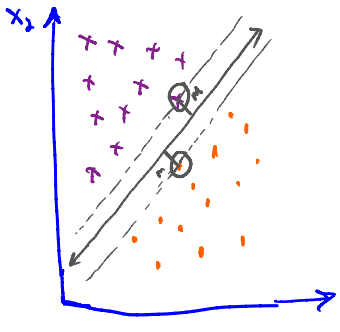


$m_2 > m_1$
larger margin => preferred hyperplane.

We can then classify a test observation based on which side of the maximal margin hyperplane it lies – this is the *maximal margin classifier*.

Hopefully a large margin on training data will lead to a large margin on the test data, => classify the test data correctly.

WARNING: when p is large can lead to overfitting.



These 2 points are equidistant from the maximal margin hyperplane.
These are known as "support vectors" because they are p-dim vectors that "support" the hyperplane.
i.e. if the 2 points moved, the maximal margin hyperplane would move as well.

NOTE: the maximal margin hyperplane only depends on the support vectors (a small # of points). The rest of the points can move and it doesn't matter.

We now need to consider the task of constructing the maximal margin hyperplane based on a set of n training observations and associated class labels.

$x_1, \dots, x_n \in \mathbb{R}^p$ $y_1, \dots, y_n \in \{-1, 1\}$.

The maximal margin hyperplane is the solution to the optimization problem

① Maximize M ← margin

$\beta_0, \beta_1, \dots, \beta_p, M$

Subject to

② $\sum_{j=1}^{p} \beta_j^2 = 1$

③ $y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip}) \geq M \quad \forall\, i = 1, \dots, n$

③ means each observation will be on the correct side of the hyperplane ($M \geq 0$) with some cushion ($M > 0$).

② ensures $y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip})$ is perp. distance to hyperplane and ③ means each point is atleast M distance away from the hyperplane ⇒ M is margin.

① choose $\beta_0, \beta_1, \dots, \beta_p, M$ to maximize margin.

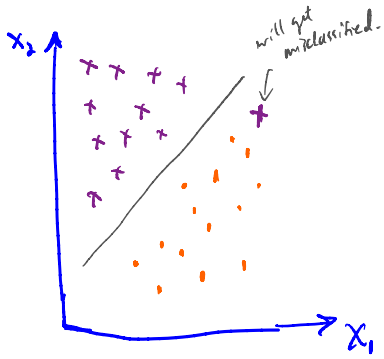
⇒ Maximal margin hyperplane!

This problem can be solved efficiently, but the details are outside the scope of this course.

We'll talk a little more about this later.

What happens when no separating hyperplane exists?

⇒ no maximal margin hyperplane!



We can't draw a hyperplane to separate these perfectly.

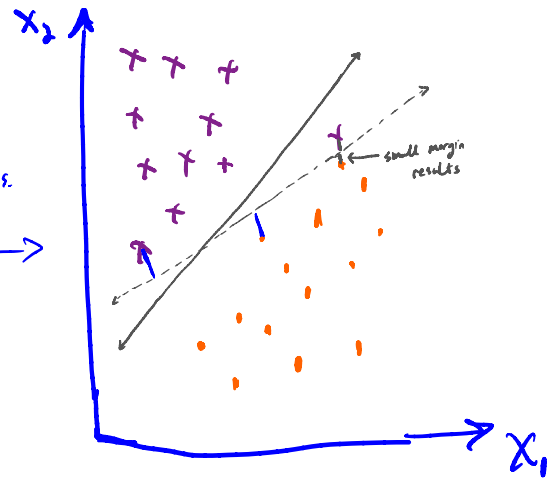
We can develop a hyperplane that almost separates the classes

= a "soft margin"

2 Support Vector Classifiers

It's not always possible to separate training observations by a hyperplane. In fact, even if we can use a hyperplane to perfectly separate our training observations, it may not be desirable.

- A classifier based on a separating hyperplane will necessarily perfectly classify all training observations.
- this can lead to sensitivity to individual observations.



Shifting one data point, results in a dramatic change in the hyperplane
& resulting hyperplane has very small margin.

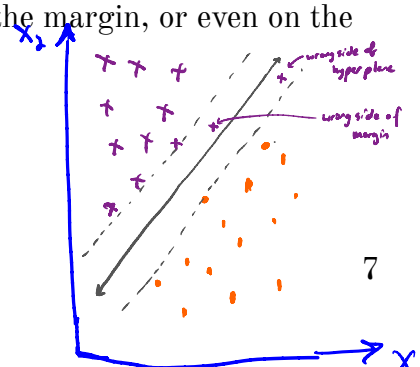
We might be willing to consider a classifier based on a hyperplane that does *not perfectly* separate the two classes in the interest of

- greater robustness to individual observations
- better classification of most of the training observations.

i.e. could be worthwile to misclassify a few observations to do a better job classifying the rest.

The *support vector classifier* (→ sometimes called "soft margin classifier") does this by finding the largest possible margin between classes, but allowing some points to be on the "wrong" side of the margin, or even on the "wrong" side of the hyperplane.

→ when there is no separating hyperplane this is inevitable.



The support vector classifier classifies a test observation depending on which side of the hyperplane it lies. The hyperplane is chosen to correctly separate **most** of the training observations.

Solution to the following optimization problem:

$$\text{maximize}_{\beta_0, \beta_1, \ldots, \beta_p, \varepsilon_1, \ldots, \varepsilon_n, M} \; M \quad \leftarrow \text{margin width}$$

subject to

$$\sum_{j=1}^{p} \beta_j^2 = 1$$

$$y_i(\beta_0 + \beta_1 x_{1i} + \ldots + \beta_p x_{pi}) \geq M(1 - \varepsilon_i)$$

$$\varepsilon_i \geq 0, \quad \sum_{i=1}^{n} \varepsilon_i \leq C$$

(ε_i: "slack variables" allow observations to be on the wrong side of the margin or the hyperplane.)

(C: "cost" nonnegative tuning parameter)

Once we have solved this optimization problem, we classify x^* (test observation) as before by determining which side of the hyperplane it lies.

classify x^* based on sign of $f(x^*) = \beta_0 + \beta_1 x_1^* + \ldots + \beta_p x_p^*$.

ϵ_i - tell us where each observation lies relative to the hyperplane and margin.

if $\varepsilon_i = 0 \Rightarrow$ obs. on correct side of margin.

$0 < \varepsilon_i \leq 1 \Rightarrow$ obs on wrong side of margin ("violated margin")

$\varepsilon_i > 1 \Rightarrow$ obs on wrong side of hyperplane.

C - tuning parameter, bounds sum of ε_i's $\Rightarrow$ determines # and severity of violations we will allow.

Think of C as a budget for the violations.

If $C = 0 \Rightarrow$ no budget for violations $\Rightarrow \varepsilon_1 = \ldots = \varepsilon_n = 0 \Rightarrow$ SV classifier = maximal margin classifier (if exists).

If $C > 0$ no more than C observations can be on wrong side of the hyperplane because wrong side of hyperplane $\Rightarrow \varepsilon_i > 1$ and $\sum_{i=1}^{n} \varepsilon_i \leq C$

Small $C \Rightarrow$ narrow margins.

large $C \Rightarrow$ wide margins allow for more violations.

controls bias-variance trade-off

*Choose C by CV.

The optimization problem has a very interesting property.

only observations on the margin or violate the margin or hyperplane affect the hyperplane ⇒ classifier!

i.e. observations that lie strictly on the correct side of the margin do not affect the support vector classifier!

Observations that lie directly on the margin or on the wrong side of the margin or hyperplane are called *support vectors*.

These observations do affect the classifier.

The fact that only support vectors affect the classifier is in line with our assertion that C controls the bias-variance tradeoff.

When C large ⇒ margin is wide, many observations violating margin or hyperplane.
⇒ many support vectors, i.e. many observations to determine hyperplane.
⇒ low variance but potentially high bias.

When C small ⇒ fewer support vectors
⇒ high variance but potentially low bias.

Because the support vector classifier's decision rule is based only on a potentially small subset of the training observations means that it is robust to the behavior of observations far away from the hyperplane.

different from behavior of other classifiers:

e.g. LDA depends on mean of all observations in a class & within class covariance.

3 Support Vector Machines

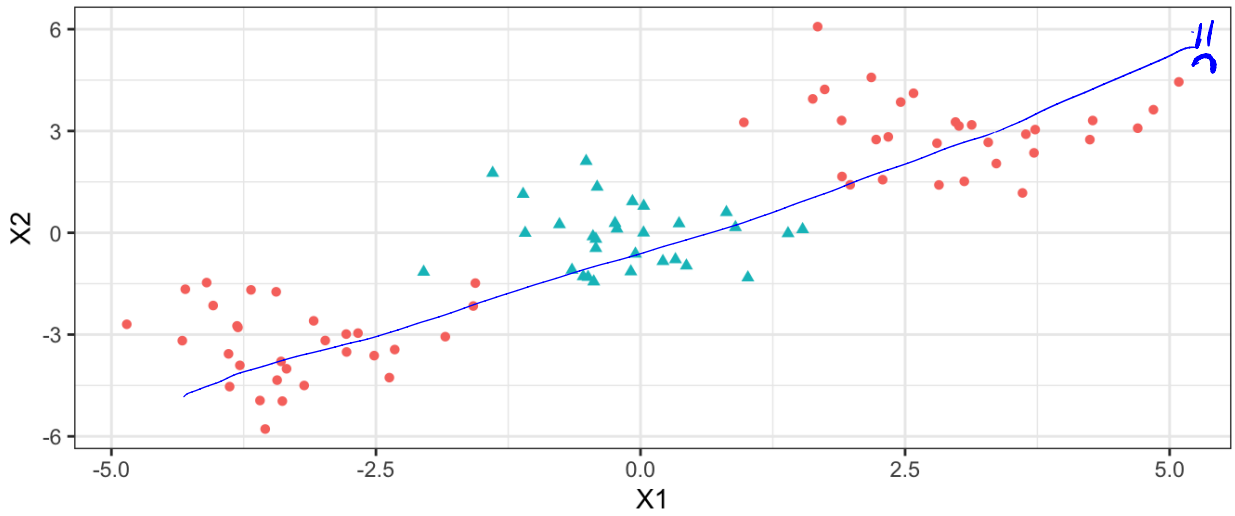
The support vector classifier is a natural approach for classification in the two-class setting...

If the decision boundary is linear!

Sometimes we have nonlinear decision boundaries!

How to draw a line separating?

won't work well.



We've seen ways to handle non-linear classification boundaries before.

KNN, QDA, RF, boosted trees.

logistic regression w/ polynomial features.

In the case of the support vector classifier, we could address the problem of possible non-linear boundaries between classes by enlarging the feature space.

e.g. quadratic or cubic terms

instead of fitting SV classifier on $X_1, \dots, X_p$

could use $X_1, X_1^2, \dots, X_p, X_p^2$.

Then our optimization problem would become

$$\text{maximize}_{\beta_0, \beta_{11}, \beta_{12}, \dots, \beta_{p1}, \beta_{p2}, \varepsilon_1, \dots, \varepsilon_n, M} \; M$$

Subject to

$$\sum_{j=1}^{p} \sum_{k=1}^{2} \beta_{jk}^2 = 1$$

$$y_i \left(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{p} \beta_{j2} x_{ij}^2 \right) \geq M(1 - \varepsilon_i)$$

$$\sum_{i=1}^{n} \varepsilon \leq C$$

hyperplane in extended feature space will lead to nonlinear boundary in feature space.

could consider higher order polynomials, or other functions

The *support vector machine* allows us to enlarge the feature space used by the support classifier in a way that leads to efficient computation. using "kernels"

Want to enlarge feature space to result in nonlinear boundary.

computation of support vector classifier

It turns out that the solution to the support vector classification optimization problem involves only *inner products* of the observations (instead of the observations themselves).

inner product, $a, b \in \mathbb{R}^r$ $\langle a, b \rangle = \sum_{i=1}^{r} a_i b_i$

two observations $x_i, x_{i'}$

inner product: $\langle x_i, x_{i'} \rangle = \sum_{j=1}^{p} x_{ij} x_{i'j}$

It can be shown that

$f(x) = \beta_0 + \beta_1 x_1 + \dots + \beta_p x_p$

- The linear support vector classifier can be represented as
  $f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle$ α_i, $i = 1, \dots, n$ parameters
- To estimate α_i and β_0 need $\binom{n}{2}$ inner products between all pairs of observations.
- α_i nonzero only for support vectors in the solution! → typically have less than n.

⇒ rewrite $f(x) = \beta_0 + \sum_{i \in \mathcal{S}} \alpha_i \langle x, x_i \rangle$, $\mathcal{S}$ = indices of support vectors. ← we need inner products, not observations themselves.

Kernel trick

Now suppose every time the inner product shows up in the SVM representation above, we replaced it with a generalization. ↑ $\langle x, x_i \rangle$

Kernel: $K(x_i, x_{i'})$ some function that quantifies similarity of two observations.

e.g. $K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$ results in support vector classifier "linear kernel" b/c linear boundary.

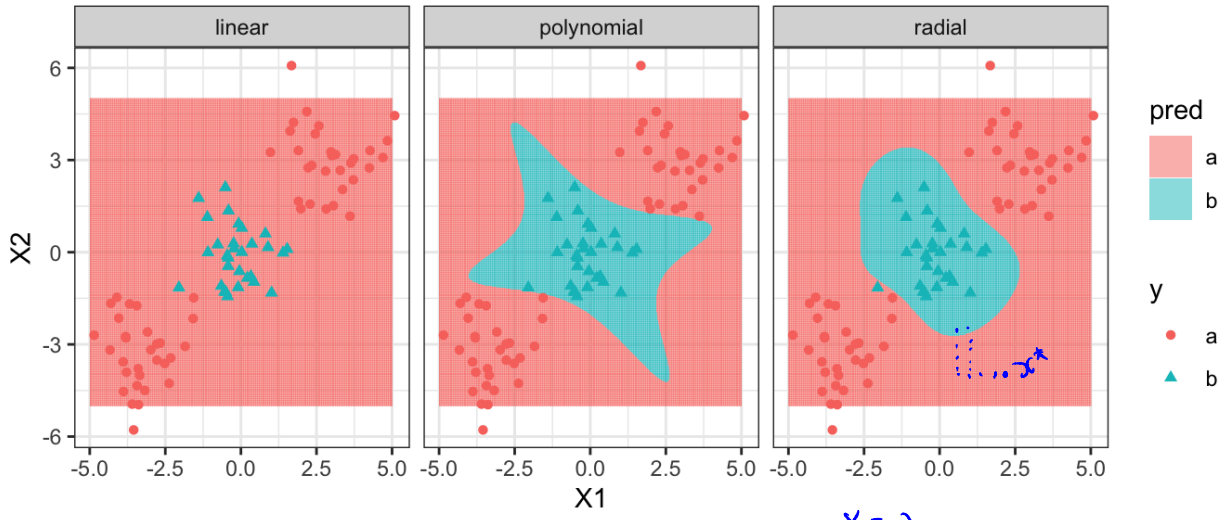
"support vector machine"

$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^d$ ← positive integer "polynomial kernel"

$K(x_i, x_{i'}) = \exp\left(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2\right)$ "radial kernel"

↑ pos constant

$$f(x) = \beta_0 + \sum_{i \in S} \alpha_i K(x, x_i)$$



$d = 4$

= fit a support vector classifier in a higher dimensional feature space w/ polynomials of degree 4.

$\gamma = 2$

$$K(x_i, x_{i'}) = \exp\left(-\gamma \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2\right)$$

if test observation x^* is far from a training observation x_i, then $\sum_{j=1}^{p} (x_j^* - x_{ij})^2$ will be large.

$\Rightarrow K(x^*, x_i)$ will be small.

$\Rightarrow$ will play a small/no role in $f(x^*)$.

$\Rightarrow$ training observations far from x^* will play very little role in prediction of class of x^*.

Why use a kernel instead of enlarging the feature space using functions of features?

Computational

- only need to compute $K(x_i, x_{i'})$ $\forall \binom{n}{2}$ distinct pairs i, i'

$\Rightarrow$ don't have to explicitly work in enlarged space (may be too large to compute hyperplane)

e.g. radial kernel - feature space is infinite dimensional.

4 SVMs with More than Two Classes

So far we have been limited to the case of binary classification. How can we extend SVMs to the more general case with some arbitrary number of classes?

This is actually not clear. There no one obvious way to extend hyperplanes to K classes.

Two popular options:

Suppose we would like to perform classification using SVMs and there are $K > 2$ classes.

One-Versus-One

(1) Construct $\binom{K}{2}$ SVMs each comparing a pair of classes.

(2) Classify a test observation using each of the $\binom{K}{2}$ SVMs

(3) Assign test observation to class if was most frequently assigned.

One-Versus-All Let x^* be a test observation

(1) Fit K SVMs compare each class to remaining k-1 classes.

(2) assign x^* to the class for which $\beta_{0k} + \beta_{1k}x_1^* + \dots + \beta_{pk}x_K^*$ is largest.

results high level of confidence test observation belongs to k^{th} class over any other.