Chapter 9: Support Vector Machines

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.

categorical response y

SVMs perform well in variety of settings

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

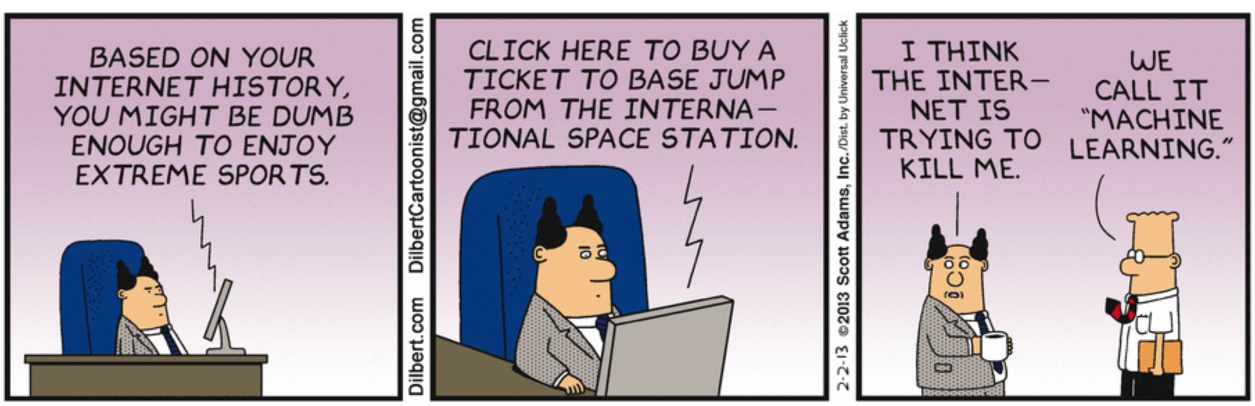
The support vector machine is a generalization of a simple and intuitive classifier called the *maximal margin classifier*.

Maximal margin classifier → generalization: allow for imperfect separation of training data → Support vector classifier → generalization: allow for nonlinear decision boundary → Support Vector machine

sometimes all three are referred to as SVM.

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

↳ categorical response w/ 2 levels



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

1 Maximal Margin Classifier

based on a hyperplane separator

extension of euclidean space.

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

eg. in 2D, a hyperplane is a flat 1 dim subspace – a line.

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

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

The mathematical definition of a hyperplane is quite simple,

In 2D, a hyperplane is by $\underbrace{\beta_0}_{} + \underbrace{\beta_1}_{\text{parameters}} X_1 + \underbrace{\beta_2}_{} X_2 = 0$.

i.e., any $X = (x_1, x_2)$ for which this equation holds, lies on the hyperplane.

Note this is just equation for a line.

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

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

i.e., any $X = (x_1, \ldots, x_p)$ for which this equation holds lies 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 and

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

You 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.

$$\underset{\sim}{x}_1 = \begin{pmatrix} x_{11} \\ \vdots \\ x_{1p} \end{pmatrix}, \dots, \underset{\sim}{x}_n = \begin{pmatrix} x_{n1} \\ \vdots \\ x_{np} \end{pmatrix}$$

training observations.

and that these observations fall into two classes.

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

where -1 represents one class
1 other class.

We also have a test observation.

p - vector of observed features

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

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

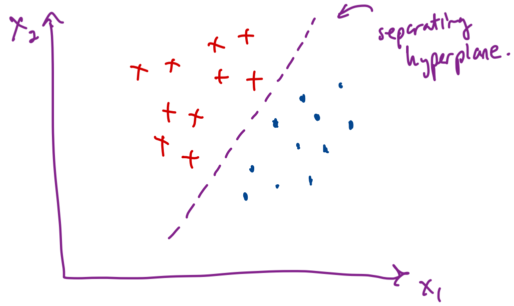
We have already used many approaches:

- Logistic regression (penalized version, i.e. LASSO or ridge)
- LDA
- classification trees
- Bagging
- Random forests
- Boosting, etc.

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

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} > 0 \quad \text{if } y_i = 1 \quad \text{and}$$

$$\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \dots + \beta_p x_{ip} < 0 \quad \text{if } y_i = -1$$

$$\Longleftrightarrow$$

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

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

A test observation is assigned to 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 $f(x^*) = \beta_0 + \beta_1 x_1^* + \cdots + \beta_p x_p^*$.

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

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

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

If $f(x^*)$ is far from zero (large magnitude) 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 (small magnitude) it is located near the hyperplane

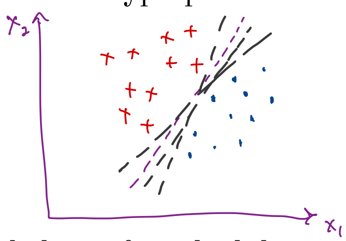
$\Rightarrow$ we are less confident about the class assignment for x^*

<u>Note</u>: a classifier based on separating hyperplane leads to a <u>linear</u> 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.

$p=2$



A given separating hyperplane can be shifted a tiny bit or rotated without coming into contact w/ any training observations

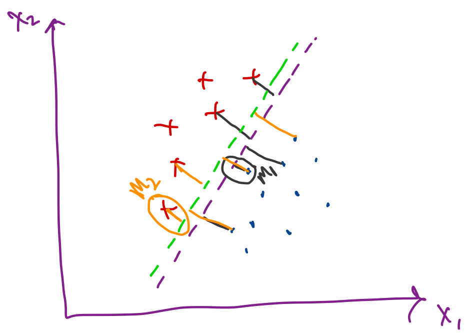
$\Rightarrow$ Which one to use for our classifier?

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.

$p=2$



$M_2 > M_1$

$\Rightarrow$ larger margin

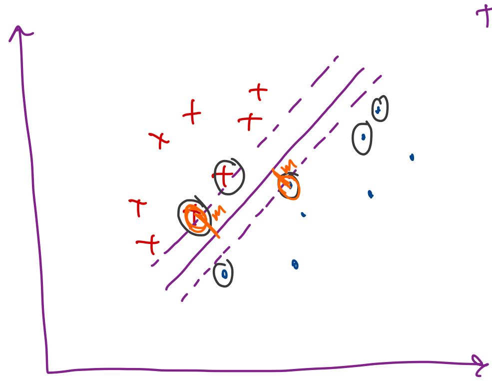
$\Rightarrow$ 2nd hyperplane is preferred.

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 test data

$\Rightarrow$ classify test data correctly

When p is large, we can see over fitting.



The two equidistant points from the maximal margin hyper plane are known as support vectors because they are p-dim vectors that "support" the hyperplane.

i.e. if the points move, the maximal margin hyperplane would move as well.

a small # of points.

NOTE: The maximal margin hyperplane only depends on the support vectors!
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.

$\underline{x}_1, \dots, \underline{x}_n \in \mathbb{R}^p$ $y_1, \dots, y_n \in \{-1, 1\}$.

The maximal margin hyperplane is the solution to the optimization problem

(1) $\underset{\beta_0, \beta_1, \dots, \beta_p, M}{\text{maximize}} \; M$ ← margin

subject to $\sum_{j=1}^{p} \beta_j^2 = 1$, (2)

(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 (if $M > 0$).

(2) ensures $y_i(\beta_0 + \beta_1 x_{i1} + \dots + \beta_p x_{ip})$ is perp. distance to hyperplane, and (3) means the point is at least M distance away $\Rightarrow$ M is the margin.

(1) chooses $\beta_0, \beta_1, \dots, \beta_p, M$ to maximize the margin

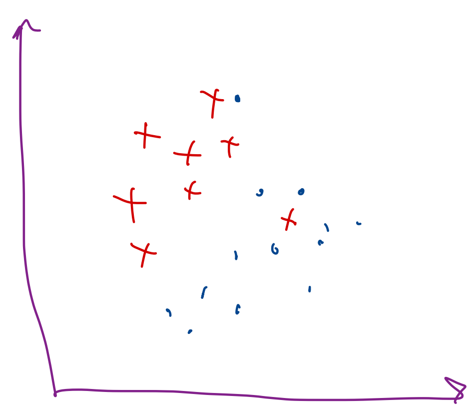
$\Rightarrow$ maximal margin hyperplane!

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

↳ we'll talk a little bit more later.

What happens when no separating hyperplane exists?

$\Rightarrow$ no maximal margin hyperplane!



We can develop a hyperplane that almost separates the classes

↑

a "soft margin"

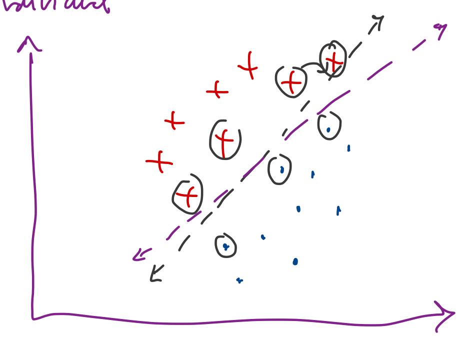
2 Support Vector Classifiers

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 oversensitivity to individual observations.

shifting / adding one data point can result in dramatic change in hyperplane (and 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

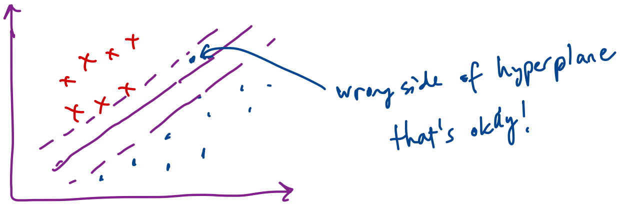
- better classification of most of observations.

i.e. could be worthwhile to misclassify a few observations to do a better job classifying the rest (also test observations).

→ sometimes called "soft margin classifier"

The *support vector 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:

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

subject to

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

$$y_i(\beta_0 + \beta_1 x_{i1} + \ldots + \beta_p x_{ip}) \geq M(1 - \varepsilon_i)$$

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

non negative tuning parameter

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

Once we have solved this optimization problem, we classify x^* 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 – tells us where observations lie relative to hyperplane & margin.

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

if $\varepsilon_i > 0 \Rightarrow$ obs on wrong side of margin (violated margin)

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

choose C by CV

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 amount of violations

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

If $C > 0 \Rightarrow$ no more than C observations can be on the wrong side of the hyperplane.

because $\varepsilon_i > 1$ and $\sum_{i=1}^{n} \varepsilon_i \leq C$.

small C $\Rightarrow$ narrow margin, large C $\Rightarrow$ wider margins and allows for more violations

C controls bias-variance trade-off.

The optimization problem has a very interesting property.

Only observations on the margin or violate the margin (or hyperplane) affect the hyperplane ⇒ the classifier!

i.e. observations that lie strictly on the correct side of the margin don't affect the 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 classifer.

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 violate the margin ⇒ many support vectors
i.e. many observations used to determine the hyperplane
⇒ low variance but potentially high bias.

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

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.

distinct from behavior of other classifier methods.

e.g. LDA depends on the mean of all observations within each class + within class covariance matrix.

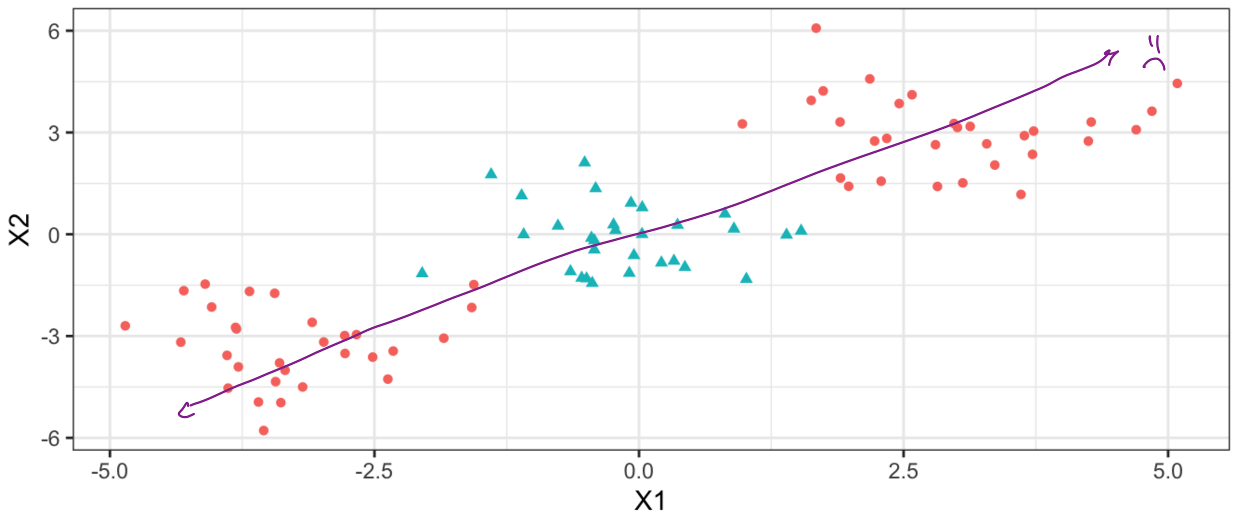
3 Support Vector Machines

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 boundaries:

How to draw a line separating?

Won't work well.



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

QDA, bagging, RF, boosting trees

logistic regression w/ polynomial features, etc.

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. adding quadratic or cubic terms

instead of fitting SV classifier w/ $X_1, \dots, X_p$

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

Then our optimization problem would become

Maximize M
$\beta_0, \beta_{11}, \beta_{12}, \dots, \beta_{1p}, \beta_{21}, \dots, \beta_{2p}, \varepsilon_1, \dots, \varepsilon_n, M$

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

quadratic polynomial leads to nonlinear boundary.

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

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

could consider higher order polynomials or other functions.

→ using "Kernels"

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

Want to enlarge feature space to have non-linear boundary.

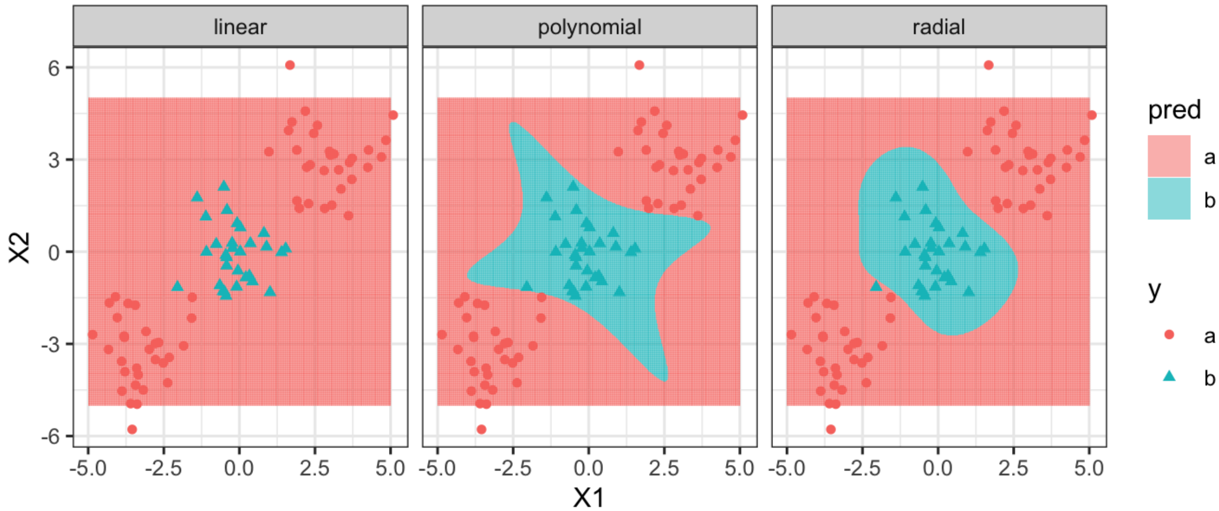
Computation of SV classifier... idea.

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).

It can be shown that

-
-
-

Now suppose every time the inner product shows up in the SVM representation above, we replaced it with a generalization.


linear
polynomial
radial
6
3
0
-3
-6
X2
-5.0
-2.5
0.0
2.5
5.0
X1
pred
a
b
y
a
b

4 SVMs with More than Two Classes

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

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

One-Versus-One

One-Versus-All