Machine Learning (Semester 1 2024)

Support Vector Machines

Nina Hernitschek
Centro de Astronomía CITEVA
Universidad de Antofagasta

June 25, 2024

Motivation

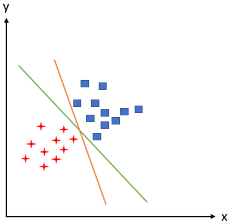
So far, we have seen a general overview about **Classification**.

Today we will learn about **Support Vector Machines**, a type of classifier.

Discriminative Classification

Discriminative classification consists of methods that seek to determine the **decision boundary in feature space**.

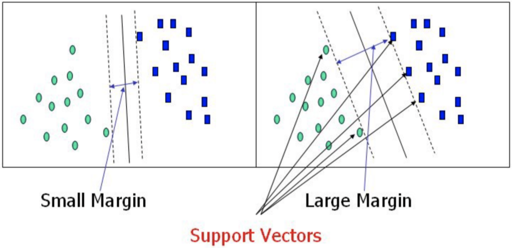
example:
We have the data as shown in the plot below. We could separate them by a line.
But: There are clearly lots of different lines that that would work. How do you do this optimally so it also works for the future target set? And what if the blobs are not perfectly well separated?



Discriminative Classification: Support Vector Machines

Support Vector Machines (SVM) solve this problem by defining a **hyperplane** in $N-1$ dimensions that maximizes the distance (the *margin*) of the closest point from each class. The points that touch the margin (or that are on the wrong side) are the **support vectors**.

There are lots of potential decision boundaries, but we want the one that maximize the distance of the support vectors from the decision hyperplane.



Discriminative Classification: Support Vector Machines

Motivation
Intro: Support Vector Machines
Maximal Margin Classifier
Support Vector Classifiers
Support Vector Machines
Summary & Outlook

For **realistic data sets** where the decision boundary is not obvious, we relax the assumption that the classes are linearly separable.

This changes the minimization condition and puts bounds on the number of misclassifications (which we would obviously like to minimize).

In the following, we will see how we can solve this problem mathematically, beginning by a simplified version of a classifier und problem.

The Maximal Margin Classifier

The Maximal Margin Classifier is a simple and intuitiv classifier.

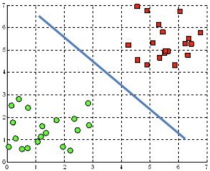
We first define a **hyperplane** and introduce the concept of an **optimal separating hyperplane**.

The Hyperplane

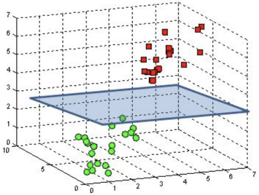
In a p-dimensional space, a **hyperplane** is a flat affine* subspace of dimension $p - 1$.

For instance, in 2D, a hyperplane is a flat 1D subspace: a line.
In 3D, a hyperplane is a flat 2D subspace: a plane.
In $p > 3$ dimensions, it can be hard to visualize a hyperplane, but the notion of a $(p-1)$-dimensional flat subspace still applies.

A hyperplane in $\mathbb{R}^2$ is a line



A hyperplane in $\mathbb{R}^3$ is a plane



*The word *affine* indicates that the subspace does not need to pass through the origin.

The Hyperplane

Mathematically, a hyperplane in 2D is defined by

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

for parameters β_0, β_1, β_2.

We say that the above equation defines the hyperplane: This means that any $X = (X_1, X_2)^T$ for which the equation holds is a point on the hyperplane.

The Hyperplane

Mathematically, a hyperplane in 2D is defined by

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0$$

for parameters β_0, β_1, β_2.

We say that the above equation defines the hyperplane: This means that any $X = (X_1, X_2)^T$ for which the equation holds is a point on the hyperplane.

We can easily extend this to p dimensions:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p = 0$$

defines a p-dimensional hyperplane.

The Hyperplane

Now, suppose that X does not satisfy the equation

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p = 0, \quad (*)$$

but rather

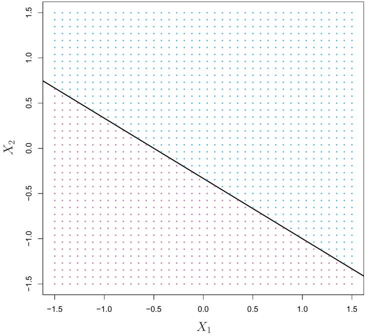
$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p > 0.$$

Then this tells us that X lies to one side of the hyperplane. On the other hand, if

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + ... + \beta_p X_p < 0$$

then X lies on the other side of the hyperplane.
With this, we can think of a hyperplane as dividing p-dimensional space into two halves. One can easily determine on which side of the hyperplane a point lies by simply calculating the sign of the left-hand side of Equ. $(*)$.

The Hyperplane



The hyperplane $1 + 2X_1 + 3X_2 = 0$.
The blue region gives the set of points for which $1 + 2X_1 + 3X_2 > 0$, and the purple region gives the set of points for which $1 + 2X_1 + 3X_2 < 0$.
Source: Fig. 9.1 from `https://www.statlearning.com/`

Classification Using a Hyperplane

Suppose we have an $n \times p$ data matrix $\mathbf{X}$ that consists of n **training observations** in p-dimensional space:

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

Suppose that these observations fall into two classes: $y_1, \ldots, y_n \in \{-1, 1\}$ where -1 represents one class and 1 the other class.

Classification Using a Hyperplane

Suppose we have an $n \times p$ data matrix $\mathbf{X}$ that consists of n **training observations** in p-dimensional space:

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

Suppose that these observations fall into two classes: $y_1, ..., y_n \in \{-1, 1\}$ where -1 represents one class and 1 the other class.

We also have a **test observation**, a p-vector of observed features $x^* = (x_1^*, ..., x_p^*)^T$.

Classification Using a Hyperplane

Suppose we have an $n \times p$ data matrix $\mathbf{X}$ that consists of n **training observations** in p-dimensional space:

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

Suppose that these observations fall into two classes: $y_1, \ldots, y_n \in \{-1, 1\}$ where -1 represents one class and 1 the other class.

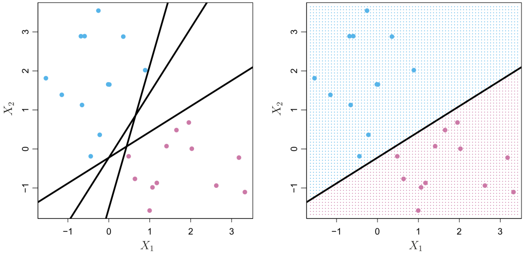
We also have a **test observation**, a p-vector of observed features $x^* = (x_1^*, \ldots, x_p^*)^T$.

Our **goal** is to develope a classifier based on the training data that will correctly classify the test observation using its feature measurements.

The Maximal Margin Classifier

Based on this, we now build a classifier that uses the concept of a **separating hyperplane**.

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



Left: Three separating hyperplanes, out of many possible ones.
Right: A separating hyperplane. The blue and purple grid indicates the decision rule made by a classifier based on this separating hyperplane.
Source: Fig. 9.2 from `https://www.statlearning.com/`

The Hyperplane

We now label the observations from the blue class as $y_i = 1$ and those from the purple class as $y_i = -1$.

With that, a separating hyperplane has the property that

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

and

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

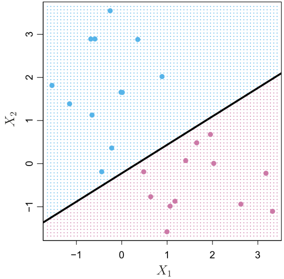
for all $i = 1, ..., n$.

If such 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.

The Hyperplane

The figure shows an example of such a classifier: We classify the test observation x^* based on the sign of $f(x^*) = \beta_0 + \beta_1 x_1^* + \beta_2 x_2^* + \ldots + \beta_p x_p^*$.



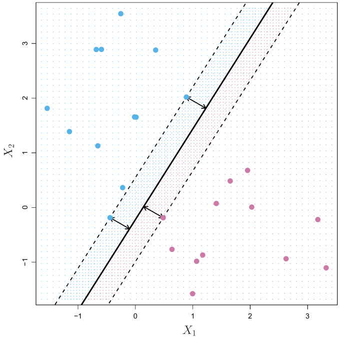
Source: Fig. 9.2 from `https://www.statlearning.com/`

If $f(x^*)$ is positive, then we assign the test observation to class 1, and if $f(x^*)$ is negative, then we assign it to class -1.

We can also make use of the value of $f(x^*)$: If $f(x^*)$ is far from zero, x^* lies far from the hyperplane, thus we can be confident about the classification of x^*. However, if $f(x^*)$ is close to zero, then x^* is located near the hyperplane, so we are less certain about the classification of x^*.

The Maximal Margin Classifier

If the data can be perfectly separated by a hyperplane, then there will in fact exist an infinite number of such hyperplanes: a separating hyperplane can usually be shifted or rotated a bit.
A reasonable is the **maximal margin hyperplane**, which is the separating hyperplane that is farthest from the training observations.
We can then classify a test observation based on which side of the maximal margin hyperplane it lies. This is known as the maximal margin classifier.



The maximal margin hyperplane is shown as a solid line. The margin is the distance from the solid line to either of the dashed lines. The blue and purple points on the dashed lines are the *support vectors*, and the distance from those points to the hyperplane is indicated by arrows. The purple and blue grid indicates the decision rule made by a classifier based on this separating hyperplane.
Source: Fig. 9.3 from `https://www.statlearning.com/`

The Maximal Margin Classifier

In summary, the task of constructing the maximal margin hyperplane based on a set of n training observations $x_1, ..., x_n \in \mathcal{R}^p$ and associated class labels $y_1, ..., y_n \in \{-1, 1\}$. is an **optimization problem**:

maximize margin M, $\beta_0, \beta_1, ..., \beta_p$

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

so that $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip}) \geq M \; \forall \; i = 1, ..., n.$

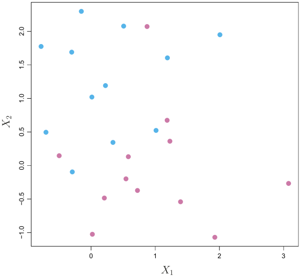
These **constraints** ensure that each observation is on the correct side of the hyperplane and at least a distance M from the hyperplane. Hence, M represents the margin of our hyperplane, and the optimization problem chooses $\beta_0, \beta_1, ..., \beta_p$ to maximize M.

This is exactly the definition of the maximal margin hyperplane.

The Maximal Margin Classifier

If a separating hyperplane exists, the maximal margin classifier is a very natural way to perform classification.

However: In many cases no separating hyperplane exists, and so there is no maximal margin classifier.



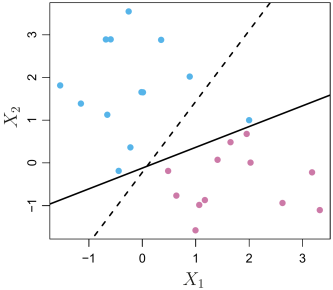
In this case, the two classes (blue and purple) are not separable by a hyperplane. The maximal margin classifier cannot be used. Source: Fig. 9.4 from `https://www.statlearning.com/`

Solution: Extend the concept of a separating hyperplane to a hyperplane that almost separates the classes, using a so-called **soft margin**.
The generalization of the maximal margin classifier to the non-separable case is known as the **support vector classifier**.

Support Vector Classifiers

A classifier based on a separating hyperplane will necessarily perfectly classify all of the training observations; this can lead to **sensitivity** to individual observations - also known as **overfitting**.

The figure below shows how the addition of a single observation in the leads to a dramatic change in the maximal margin hyperplane.



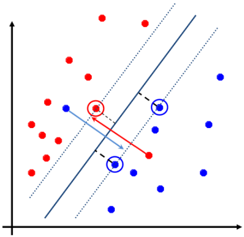
Two classes of observations are shown in blue and in purple, along with the maximal margin hyperplane. An additional blue observation has been added, leading to a dramatic shift in the maximal margin hyperplane shown as a solid line. The dashed line indicates the maximal margin hyperplane that was obtained in the absence of this additional point.
Source: Fig. 9.5 (right) from `https://www.statlearning.com/`

Support Vector Classifiers

To avoid overfitting, it might be better to consider a classifier based on a hyperplane that does not perfectly separate the two classes:

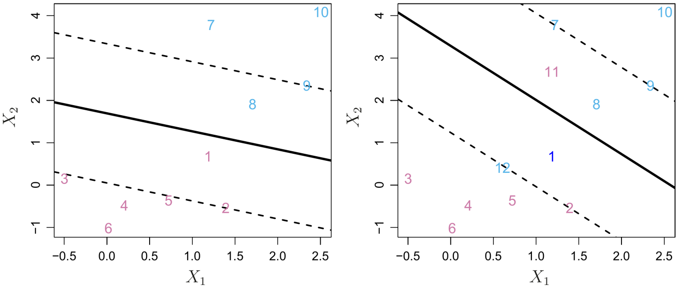
- Greater robustness to individual observations
- Better classification of most training observations: misclassify a few training observations in order to do a better job in classifying the remaining observations.

The **support vector classifier**, sometimes called a soft margin classifier, does exactly this:



Instead of seeking the largest possible margin so that every observation is on the correct side of the hyperplane and on the correct side of the margin, we allow some observations to be on the incorrect side of the margin or the hyperplance. This gives a *soft margin* that can be violated by some of the training observations.

Support Vector Classifiers



Left: A support vector classifier was fit to a small data set. The hyperplane is shown as a solid line and the margins are shown as dashed lines. Purple: Observations 3, 4, 5, and 6 are on the correct side of the margin, 2 is on the margin, 1 is on the wrong side of the margin. Blue: Observations 7 and 10 are on the correct side of the margin, 9 is on the margin, 8 is on the wrong side of the margin. None are on the wrong side of the hyperplane.

Right: Same as left panel with two additional points, 11 and 12. These two observations are on the wrong side of the hyperplane and the wrong side of the margin.

Support Vector Classifiers

The support vector classifier is the solution to the optimization problem:

maximize margin $M, \beta_0, \beta_1, ..., \beta_p, \epsilon_1, ..., \epsilon_n$

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

so that $y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + ... + \beta_p x_{ip} \geq M(1 - \epsilon_i)$

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

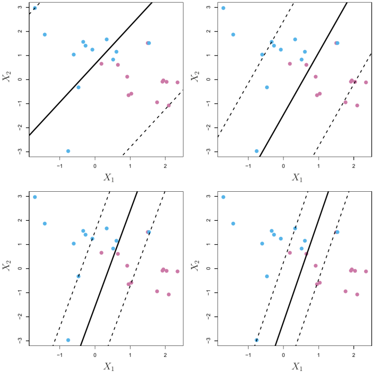
where C is a nonnegative tuning parameter.

M is the width of the margin; we make this quantity as large as possible. If $\epsilon_i = 0$, then the ih observtion is on the correct side of the margin. If $\epsilon_i > 0$, then the ith observtion is on the wrong side of the margin. If $\epsilon_i > 1$, then it is on the wrong side of the hyperplane.

Once whe have solved these equations, we classify a test observation x^* as before, by determining on which side of the hyperplane it lies, i.e. based on the sign of $f(x^*) = \beta_0 + \beta_1 x_1^* + ... + \beta_p x_p^*$.

Support Vector Classifiers

C controls the bias-variance trade-off of the statistical learning technique. When C is large, then there is a high tolerance for observations being on the wrong side of the margin, and so the margin will be large. As C decreases, the tolerance for observations being on the wrong side of the margin decreases, and the margin narrows.
The value of C is generally chosen via cross-validation.



A support vector classifier was fit using four different values of the tuning parameter C. The largest value of C was used in the top left panel, and smaller values were used in the top right, bottom left, and bottom right panels.
Source: Fig. 9.7 from `https://www.statlearning.com/`

Support Vector Classifiers

The optimization problem has a very interesting property:

It turns out that only observations that either lie on the margin or that violate the margin will affect the hyperplane, and hence the classifier obtained.

In other words, an observation that lies strictly on the correct side of the margin does not affect the support vector classifier!

Changing the position of that observation would not change the classifier at all, provided that its position remains on the correct side of the margin. Observations that lie directly on the margin, or on the wrong side of the margin for their class, are known as **support vectors**. These observations do affect the support vector classifier.

Non-Linear Decision Boundaries

For **realistic data sets** where the decision boundary is not obvious, we relax the assumption that the classes are linearly separable. This changes the minimization condition and puts bounds on the number of misclassifications (which we would obviously like to minimize).

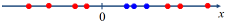
Non-Linear Decision Boundaries

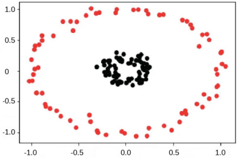
For **realistic data sets** where the decision boundary is not obvious, we relax the assumption that the classes are linearly separable. This changes the minimization condition and puts bounds on the number of misclassifications (which we would obviously like to minimize).

We first discuss a general mechanism for converting a linear classifier into one that produces **non-linear decision boundaries**. We then introduce the support vector machine, which does this in an automatic way.

Non-Linear Decision Boundaries

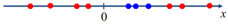
Datasets requiring **Non-Linear Decision Boundaries** can look very differently:

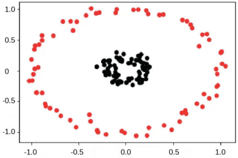




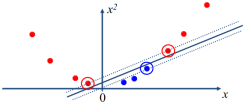
Non-Linear Decision Boundaries

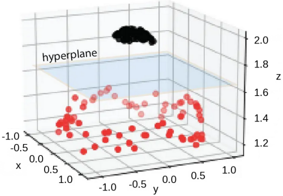
Datasets requiring **Non-Linear Decision Boundaries** can look very differently:





We could try mapping the original feature space into a higher-dimensional feature space:





Non-Linear Decision Boundaries

We enlarge the feature space using quadratic, cubic, and even higher-order polynomial functions of the predictors.

For instance, rather than fitting a support vector classifier using p features

$$X_1, X_2, ..., X_p$$

we could instead fit a proper support vector classifier using $2p$ features

$$X_1, X_1^2, X_2, X_2^2 ..., X_p, X_p^2.$$

Non-Linear Decision Boundaries

We enlarge the feature space using quadratic, cubic, and even higher-order polynomial functions of the predictors.

For instance, rather than fitting a support vector classifier using p features

$$X_1, X_2, ..., X_p$$

we could instead fit a proper support vector classifier using $2p$ features

$$X_1, X_1^2, X_2, X_2^2 ..., X_p, X_p^2.$$

Why does this lead to a non-linear decision boundary?

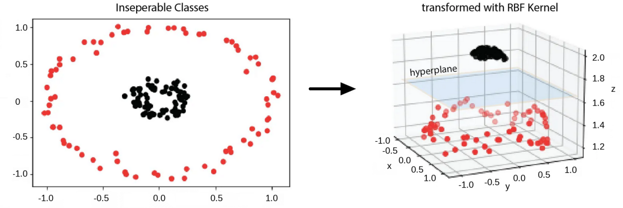
In the enlarged feature space, the decision boundary is in fact linear. But in original feature space, the decision boundary is polynomial. One might additionally want to enlarge the feature space with interaction terms of the form $X_j X_{j'}$ for $j \neq j'$.

Alternatively, other functions of the predictors could be considered rather than polynomials.

Support Vector Machines

The support vector machine (SVM) is an extension of the support vector classifier that results from enlarging the feature space in a specific way, using **kernels**.

The main idea is that we enlarge our feature space in order to accommodate a non-linear boundary between the classes. The kernel approach that we describe here is simply an efficient computational approach for carrying out this idea.



Support Vector Machines

It can be shown that

i) The linear support vector classifier can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle,$$

where the inner product of two r-vectors a and b is defined as $\langle a, b \rangle = \sum_{i=1}^{r} a_i b_i$, and there are n parameters α_i, $i = 1, ..., n$, one per training observation.

ii) To estimate the parameters α_i, all we need are $\binom{n}{2} = n(n-1)/2$ inner products $\langle x_i, x_{i'} \rangle$ between all pairs of training observations.
$\binom{n}{2}$ gives the number of pairs among a set of n items.

Support Vector Machines

Now suppose that every time the inner product appears in the representation $f(x)$, or in a calculation of the solution for the support vector classifier, we replace it with a generalization of the inner product of the form $K(x_i, x_{i'})$ where K is some function that we will refer to as a **kernel**.

A kernel is a function that quantifies the similarity of two observations. For instance, we could simply take

$$K(x_i, x_{i'}) = \sum_{j=1}^{p} x_{ij} x_{i'j}$$

which would just give us back the support vector classifier. The equation shown here is known as a linear kernel because the support vector classifier is linear in the features.

Support Vector Machines

Other kernels are possible, e.g. the **polynomial kernel**:

$$K(x_i, x_{i'}) = \left(1 + \sum_{j=1}^{p} x_{ij} x_{i'j}\right)^d$$

where d is a positive integer.
Using such a kernel with $d > 1$, instead of the standard linear kernel, in the support vector classifier algorithm leads to a much more flexible decision boundary.

Support Vector Machines

Motivation

Intro: Support Vector Machines

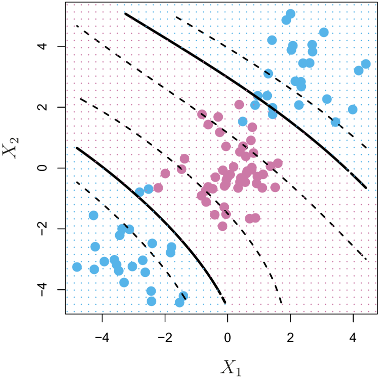
Maximal Margin Classifier

Support Vector Classifiers

Support Vector Machines

Summary & Outlook

A SVM with a polynomial kernel of degree 3 is applied, resulting in a far more appropriate decision rule.



Source: Left panel of Fig. 9.9 from `https://www.statlearning.com/`

Support Vector Machines

Another popular choice is the **radial kernel**:

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

where γ is a positive constant.

The idea behind the radial kernel:

If a given test observation $x^* = (x_1^*, ..., x_p^*)^T$ is far from a training observation x_i in terms of Euclidean distance, then $\sum_{j=1}^{p}(x_j^* - x_{ij})^2$ will be large, and so $K(x_i, x_{i'}) = \exp\left(-\gamma \sum_{j=1}^{p}(x_{ij} - x_{i'j})^2\right)$ wil lbe very small.
This means that x_i will then play virtually no role in $f(x^*)$. As the predicted class label for the test observation x^* is based on the sign of $f(x^*)$, training observations that are far from x^* will play essentially no role in the predicted class label for x^*.
This means that the radial kernel has **very local behavior**.

Support Vector Machines

Motivation

Intro: Support Vector Machines

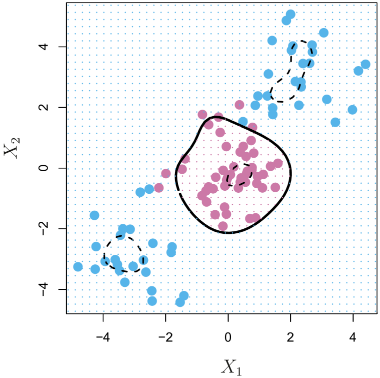
Maximal Margin Classifier

Support Vector Classifiers

Support Vector Machines

Summary & Outlook

A SVM with a radial kernel applied.



Source: Right panel of Fig. 9.9 from `https://www.statlearning.com/`

Multi-Class Support Vector Machines

In many cases when doing classification we want to discriminate between many classes.

For extending SVMs to the multi-class case have been made, the two most popular are the **one-versus-one** and **one-versus-all** approaches. We briefly discuss those two approaches here.

One-Versus-One Classification

Suppose that we would like to perform classification using SVMs, and there are $K > 2$ classes. A **one-versus-one** or **all-pairs** approach constructs $\binom{K}{2}$ SMVs, each of which compares a pair of classes.

For example, one such SMV might compare the kth class to the k'th class.

We classify a test observation using each of the $\binom{n}{2}$ classifiers, and we count the number of times that the test observation is assigned to each of the K classes.

The final classification is performed by assigning the test observation to the class to which is was most frequently assigned.

One-Versus-One Classification

For example, consider a multi-class classification problem with four classes: 'red', 'blue', 'green', 'yellow'.

This could be divided into six binary classification datasets as follows:

- Binary Classification Problem 1: red vs. blue
- Binary Classification Problem 2: red vs. green
- Binary Classification Problem 3: red vs. yellow
- Binary Classification Problem 4: blue vs. green
- Binary Classification Problem 5: blue vs. yellow
- Binary Classification Problem 6: green vs. yellow

One-Versus-All Classification

The **one-versus-all** approach, also referred to as **one-versus-rest**, is an alternative approach for applying SVMs in the case of $K > 2$ classes.

We fit K SVMs, each time comparing one of the K clases to the remaining $K - 1$ classes.

Let $\beta_{0k}, \beta_{1k}, ..., \beta_{pk}$ denote the parameters that result from fitting an SVM comparing the kth class to the others.

Let x^* denote a test observation. We assign the observation to the class for which $\beta_{0k} + \beta_{1k}x_1^* + ... + \beta_{pk}x_p^*$ is the largest, as this amounts to a high level of confidence that the test observation belongs to the kth class rather than to any of the other classes.

One-Versus-All Classification

For example, given a multi-class classification problem with examples for each class 'red', 'blue', 'green', 'yellow'. This could be divided into three binary classification datasets as follows:

- Binary Classification Problem 1: red vs [blue, green, yellow]
- Binary Classification Problem 2: blue vs [red, green, yellow]
- Binary Classification Problem 3: green vs [red, blue, yellow]
- Binary Classification Problem 3: yellow vs [red, blue, green]

This is significantly less datasets, and in turn, models than the one-versus-one approch. It requires only one model to be created for each class. For example, three classes requires three models.

However, this could be an issue for large datasets (e.g. millions of rows), slow models, or very large numbers of classes (e.g. hundreds of classes).

Summary

Some comments on SVM:

- SVM is not scale invariant so it is often worth rescaling the data to [0,1] or to whiten it to have a mean of 0 and variance 1 (remember to do this to the test data as well!).
- The data don't need to be separable (we can put a constraint in minimizing the number of 'failures').
- The median of a distribution is unaffected by large perturbations of outlying points, as long as those perturbations do not cross the boundary.

Summary

Some comments on SVM:

- In the same way, by maximizing the margin of support vectors rather than using all data points, SVM classification is similar to rank-based estimators. Once the support vectors are determined, changes to the positions or numbers of points beyond the margin will not change the decision boundary. For this reason, SVM can be a very powerful tool for discriminative classification.
- This is why there is a high completeness compared to the other methods: it does not matter that the background sources outnumber the RR Lyrae stars by a factor of $\sim$200 to 1. It simply determines the best boundary between the small RR Lyrae clump and the large background clump.
- This completeness, however, comes at the cost of a relatively large contamination level.

Summary

Today we have seen how to use Support Vector Machines.

Next time we will learn about **Tree-Based Classifiers**, another form of Discriminative Classification.