Statistical Machine Learning
Logistic Regression and SVM

Horacio Gómez-Acevedo
Department of Biomedical Informatics

February 12, 2024

UAMS

Logistic Regression

Consider that we have an experiment with covariate(s) X and response Y with either two possibilities *yes* or *no.* For convenience, we use the binary codes for the response variable, say 1 for yes and 0 for no. We use **logistic regression** to model the probability that Y is 1 or 0.
We use the following notation to denote the probability that given

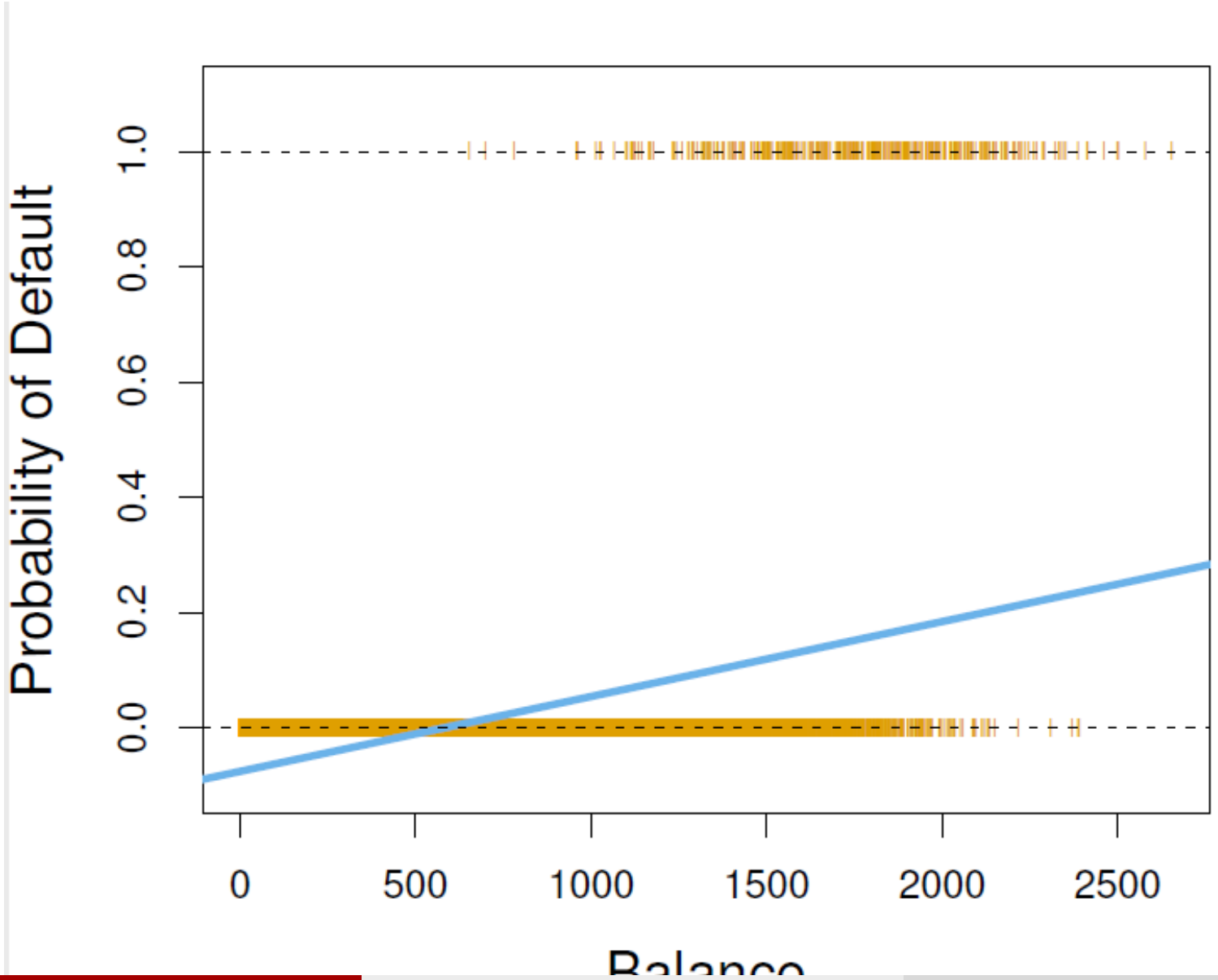
$$p(X = x) = Pr(Y = 1|X = x)$$

Thus, if $p(X = x) > 0.5$ we can say that $X = x$ belongs to the class 1.
Our first attempt may be to use linear regression as follows:

$$p(X) = \theta_0 + \theta_1 X$$

However, when we use such a model we may encounter negative probabilities!

Logistic Regression (cont)



Logistic Function

We apply the so-called **logistic function**

$$p(X) = \frac{\exp(\theta_0 + \theta_1 X)}{1 + \exp(\theta_0 + \theta_1 X)}$$

After some algebraic manipulation, we can show that

$$\log\left(\frac{p(X)}{1 - p(X)}\right) = \theta_0 + \theta_1 X \tag{1}$$

The left-hand of (1) is called the **log-odds** or **logit**.
For prediction, in the logistic regression model we use $\hat{p}(X)$ as follows

$$\hat{Y} = \begin{cases} 0 & \text{if } \hat{p}(X) \leq 0.5 \\ 1 & \text{if } \hat{p}(X) > 0.5 \end{cases}$$

Logistic Regression Coefficients Interpretation

In the normal linear regression model θ_1 give the average chance in Y associated with a one-unit increase in X. Say for instance we have a linear model

$$\text{sales} = \theta_0 + \theta_1(\text{TV units sold})$$

Thus, the $E(\text{sales})$ when the TV units sold is increased by 1 unit is θ_1. In a logistic regression model increasing X by one unit changes the log odds by θ_1, or equivalently it multiplies the odds by $\exp(\theta_1)$.

Estimating the coefficients

Whereas we can use our training data to estimate the coefficients θ_0 and θ_1 using a least squares approach, this is not used in logistic regression. Instead the method of **maximum likelihood** is applied.
The likelihood function is defined as the join probability of the observed data values, given the model parameters. Formally, it is defined as

$$\ell(\theta_0, \theta_1) = \prod_{i:y_i=0} p(x_i) \prod_{i':y_i'=1} (1 - p(x_{i'}))$$

The **deviance** is defined as

$$\text{deviance} = -2 \log \text{likelihood}$$

The estimates $\hat{\theta}_0$ and $\hat{\theta}_1$ are chosen to *maximize* this likelihood function. Equivalently, we can *minimize* the deviance.

Estimating the coefficients (cont)

The terminology for logistic regression and multiple linear regression is sometimes confusing, but they are describing the same basic principles.

Regression	Logistic Regression
Degrees of Freedom	Degrees of Freedom
Sum of squares (SS)	Deviance (D)
Mean sum of squares (divide by DF)	Mean deviance (divide by DF)
Fitting by minimizing residual SS	Fitting by minimizing the deviance

Multiple Logistic Regression

As with multi linear regression, we can expand the model to accommodate more predictors

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \theta_0 + \theta_1 X_1 + \cdots + \theta_p X_p,$$

where $X = (X_1, \ldots, X_p)$ are p predictors. And the corresponding logistic function is given by

$$p(X) = \frac{\exp(\theta_0 + \theta_1 X_1 + \cdots + \theta_p X_p)}{1 + exp(\theta_0 + \theta_1 X_1 + \cdots + \theta_p X_p)}$$

Maximal Margin Classifiers

A **hyperplane of dimension** p consists of the points (vectors) in $\mathbb{R}^p$ that satisfy the condition

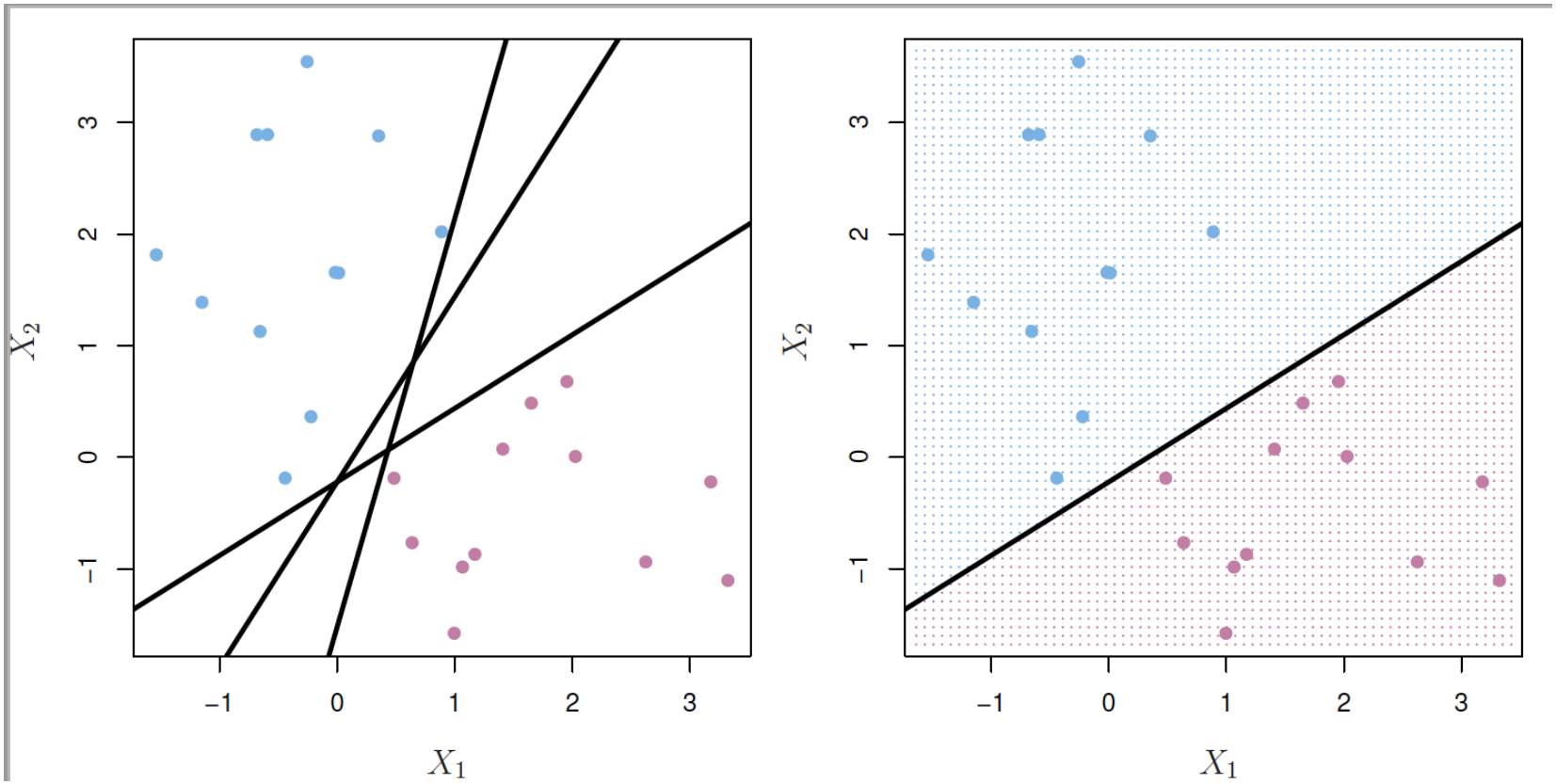
$$\beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p = 0, \tag{2}$$

where $X = (X_1, \ldots, X_p)^t$ is an element of $\mathbb{R}^p$.
The hyperplane gives us a natural separation of $\mathbb{R}^p$ into two regions

$$A^- = \{X \in \mathbb{R}^p : \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p < 0\}$$
$$A^+ = \{X \in \mathbb{R}^p : \beta_0 + \beta_1 X_1 + \cdots + \beta_p X_p > 0\}$$

Separating Hyperplanes (Visual Idea)

The problem of separating by a hyperplane (line in $\mathbb{R}^2$, plane in $\mathbb{R}^3$, and hyperplane for higher dimensions)



Classification by separating hyperplanes

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

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

and that these observations belong into either of the classes w_1 or w_2. For simplicity sake, we label the classes as a form of a vector $Y = (y_1, \ldots, y_p)$ where $y_i \in \{-1, 1\}$ for each i.

Classification by separating hyperplanes

If there is a hyperplane that separates the training observations perfectly according to their labels, we call it **separating hyperplane**. Then it satisfies the conditions

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

Alternatively, a separating hyperplane satisfies the condition

$$y_i \left(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}\right) > 0 \quad \text{for all } i \in \{1, \ldots, n\}$$

Maximal Margin Classifier

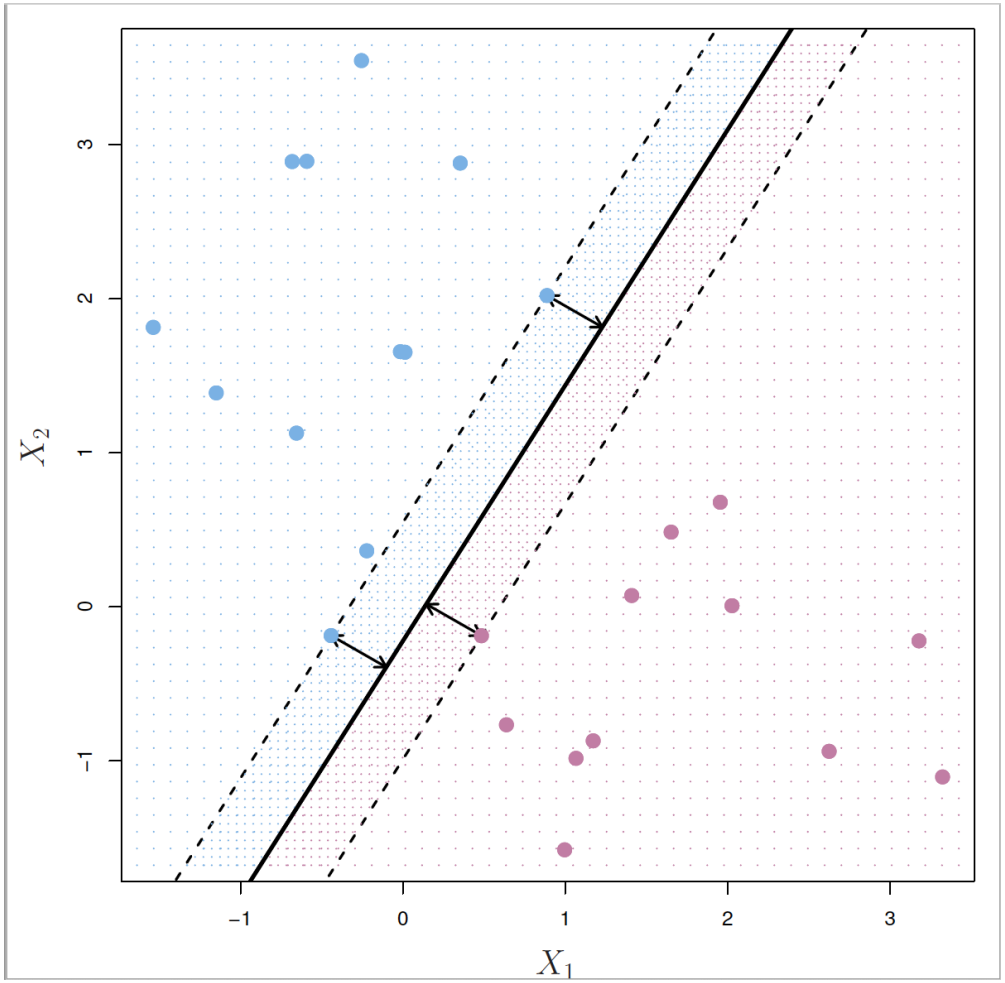
Normally, if we have a separating hyperplane, there are an infinite number of alternative hyperplanes. The question is how do we select one?
We will select an "optimal" plane by performing the following optimization process on the training data

- Compute the (perpendicular) distance from each training point to the given separating hyperplane.
- The minimum distance from the hyperplane defines the **margin**.
- Select the hyperplane with the maximal margin M.

The classification is carried out in test data by using the **maximal margin classifier**. That is the sign of $f(x^*) = \beta_0 + \beta_1 x_1^* + \cdots + \beta_p x_p^*$.
The points that are equidistant from the maximal margin hyperplane are refer to as **support vectors**.

Support Vectors

An important property is that the maximal margin hyperplane depends directly on only a small subset of the observations (the support vectors).



Construction of the Maximal Margin Classifier

Finding the maximal margin hyperplane corresponds to finding a solution to the optimization problem

$$\begin{aligned}
&\text{maximize}_{\ \beta_0,\ldots,\beta_p} M \\
&\text{subject to} \sum_{j=1}^{p} \beta_j^2 = 1, \\
&y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}) \geq M \quad \forall i \in \{1,\ldots,n\}
\end{aligned}$$

where our n training observations are x_i and Y is the vector with the class labels.

Non-separable Case

There are instances in which we cannot find the separating plane.
One way to get around this problem is to introduce **soft margins**, which means that we allow observations to be on the incorrect side of the margin, or even the incorrect side of the hyperplane.

$$\text{maximize}_{\beta_0,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n} M$$

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

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

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

where C is a hyperparameter that will be tuned.

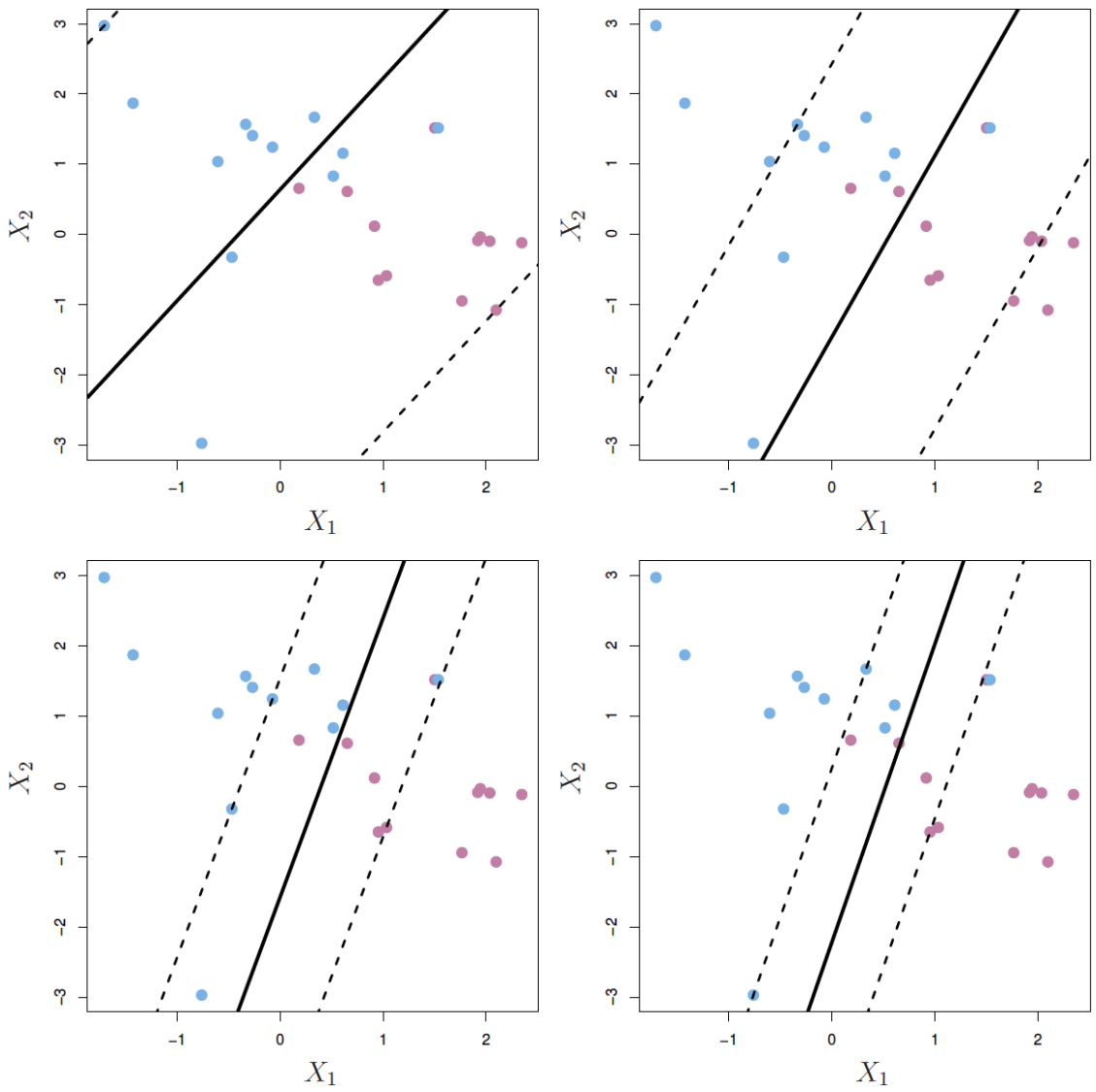
Support Vector Classifier

The variables ϵ_i are called **slack variables** that allow individual observations to be on the wrong side of the margin or the hyperplane. More precisely, if $\epsilon_i = 0$ the ith observation is on the right side of the margin. If $0 < \epsilon_i \leq 1$, the ith observation is on the wrong side of the margin. If $\epsilon_i > 1$ the observation ith is on the wrong side of the hyperplane.
The hyperparameter C limits the tolerance that we allow for the observations to be either on the wrong side of the margin or hyperplane. It controls the bias-variance trade-off.

Support Vector Classifier (cont)

- If C is large, the margin is wider and we allow more violations; this amounts to fitting the data less hard and increasing bias but decreasing variance.
- If C is small, we fit very tightly our model, so a low bias but high variance.

Variable C



Support Vector Machines

We generalize the previous problem by expanding the number of features. Say, instead of of fitting a support vector classifier using p features

$$X_1, \ldots, X_p,$$

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

$$X_1, X_1^2, X_2, X_2^2, \ldots, X_p, X_p^2$$

In this case, the optimization process becomes

$$\text{maximize}_{\beta_0, \beta_{11}, \beta_{12}, \ldots, \beta_{1p}\beta_{2p}, \epsilon_1, \ldots, \epsilon_n} M$$

$$\text{subject to } y_i(\beta_0 + \sum_{j=1}^{p} \beta_{j1} x_{ij} + \sum_{j=1}^{n} \beta_{j2} x_{ij}^2) \geq M(1 - \epsilon_i)$$

$$\sum_{j=1}^{p} \sum_{j=1}^{2} \beta_{ij}^2 = 1, \epsilon_i \geq 0, \sum_{i=1}^{n} \epsilon_i \leq C$$

Support Vector Machines

As we saw in the example above, one way to extend the support vector classifier is enlarging the feature space by the so-called **kernels**
Instead of using the observations, we will use the **inner product** of them.
The dot product is a form of an inner product

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

In this framework, the linear support classifier can be represented as

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

Note that for the estimation of β_0 and α_i we need the $\binom{n}{2}$ inner products $\langle x_i, x_{i'} \rangle$ between all pairs of training observations. However, this expression simplifies because the if x_i is not a support vector $\alpha_i = 0$.

Other kernels

We can replace the inner product with other kernels:

- Polynomial kernel of degree d

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

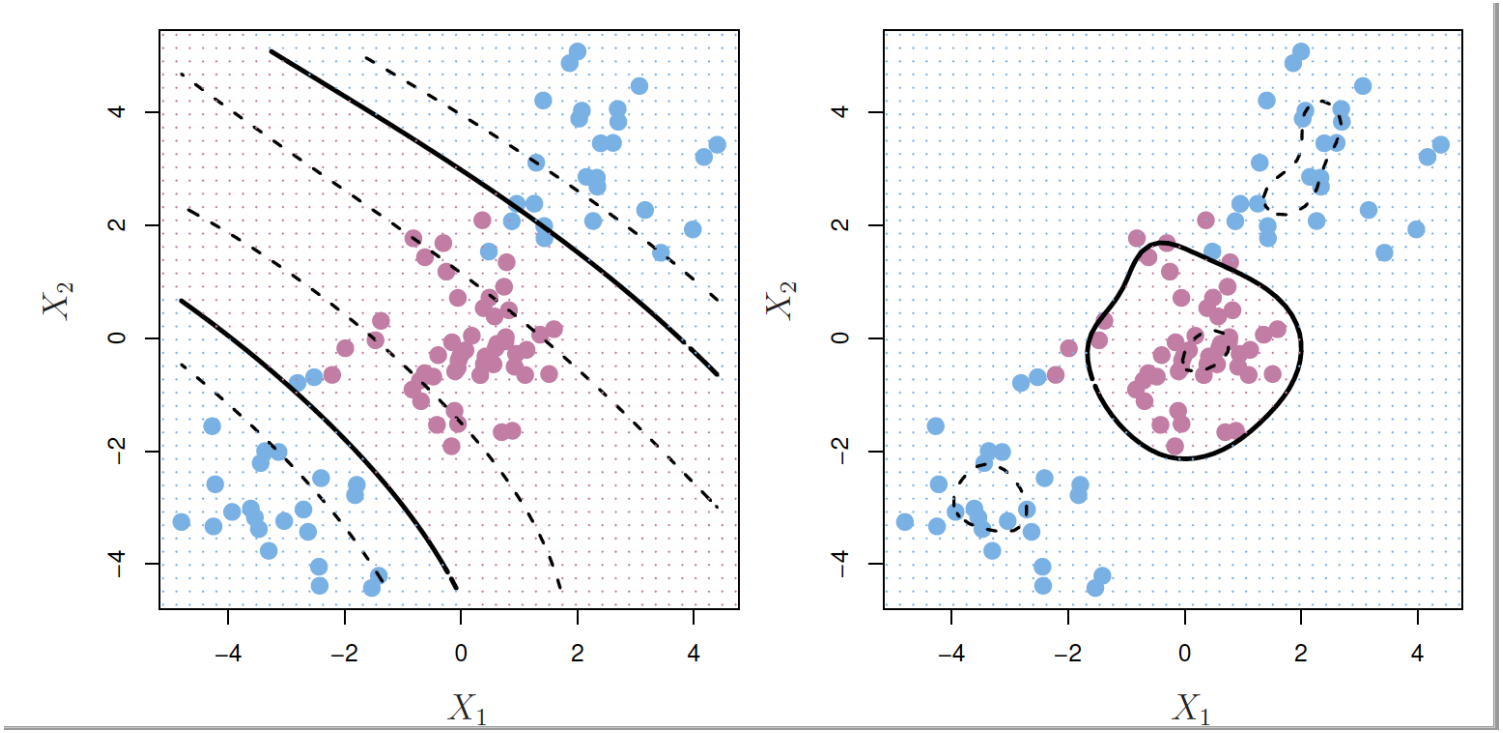
- Radial kernel

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

- Neural network

$$K(x_i, x_{i'}) = \tanh(\kappa_1 \langle x, x' \rangle + \kappa_2)$$

SVMs with different kernels



In the left, we have a polynomial kernel of degree 3, on the right we have a radial kernel.

Rough Optimization Ideas

Let's start with an equivalence

$$\begin{aligned}
&\text{maximize}_{\beta_0,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n} M \\
&\text{subject to } \sum_{j=1}^{p} \beta_j^2 = 1, \\
&y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}) \geq M(1-\epsilon_i) \\
&\epsilon_i \geq 0, \sum_{i=1}^{n} \epsilon_i \leq C
\end{aligned}$$

With the following (dropping the norm constrain in β and define $C = 1/\|\beta\|$)

$$\begin{aligned}
&\text{minimize}_{\beta_0,\ldots,\beta_p,\epsilon_1,\ldots,\epsilon_n} \|\beta\| \\
&\text{subject to } y_i(\beta_0 + \beta_1 x_{i1} + \cdots + \beta_p x_{ip}) \geq (1-\epsilon_i) \\
&\epsilon_i \geq 0, \sum_{i=1}^{n} \epsilon_i \leq \text{constant}
\end{aligned} \tag{3}$$

Optimization Problem

The problem (3) is quadratic with linear inequality constrains, hence it is a convex optimization problem. We rewrite this problem as

$$\min_{\beta,\beta_0} \frac{1}{2}\|\beta\|^2 + \gamma \sum_{i=1}^{N} \epsilon_i$$
$$\text{subject to } \epsilon_i \geq 0, y_i(x_i^T \beta + \beta_0) \geq 1 - \epsilon_i \forall i$$

The Lagrange (primal) function is

$$\begin{aligned} L_p = & \frac{1}{2}\|\beta\|^2 + \gamma \sum_{i=1}^{N} \epsilon_i - \\ & \sum_{i=1}^{N} \alpha_i [y_i(x_i^T \beta + \beta_0) - (1 - \epsilon_i)] - \sum_{i=1}^{N} \mu_i \epsilon_i \end{aligned} \tag{4}$$

which is minimized with respect to β, β_0 and ϵ_i. Using calculus we take partial derivatives and make them zero.

Optimization Problem

$$\begin{aligned} \beta_j &= \sum_{i=1}^{N} \alpha_i y_i x_{ij} \\ 0 &= \sum_{i=1}^{N} \alpha_i y_i, \\ \alpha_i \gamma - \mu_i &\quad \forall i \end{aligned} \tag{5}$$

as well as the positivity constrains $\alpha_i, \mu_i, \epsilon_i \geq 0$ for all i. If we substitute (5) into (4), we obtain the Lagrangian(Wolfe) dual objective function

$$L_D = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i'=1}^{N} \alpha_i \alpha_{i'} y_i y_{i'} x_i^T x_{i'}, \tag{6}$$

We maximize L_D subject to $0 \leq \alpha_i \leq \gamma$ and $\sum_{i=1}^{N} \alpha_i y_i = 0$.

What we have learned?

- Logistic regression as a classification process
- Maximal margin classifier as a method for classification
- Support Vector classifier uses a hyperparameter to control soft limits
- Kernels allow us to generalize the SVM to take care of the non-linearity.

References

Materials and some of the pictures are from [1], [3], [2], [4] and [5].

[1] J. Gareth et al. *An Introduction to Statistical Learning*. 1st edition. Springer, 2015. ISBN: 978-1-4614-7137-0.

[2] A. Géron. *Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow. Concepts, Tools, and Techniques to Build Intelligent Systems*. 2nd. Edition. O'Reilly, 2019. ISBN: 978-1-492-03264-9.

[3] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. 1st Edition. Springer Series in Statistics. Springer, 2001. ISBN: 978-0-387-95284-0.

[4] J. Maindonald and J. Braun. *Data Analysis and Graphics Using R. An example-based approach*. Second Edition. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2008. ISBN: 9780521861168.

[5] W. R. Pestman. *Mathematical Statistics. An Introduction*. De Gruyter, 1998. ISBN: 3-11.015356-4.