Support Vector Machine (метод опорных векторов)

В 1990-е люди научились, например, довольно хорошо распознавать и оцифровывать тексты (в том числе рукописные) или очищать почту от спама. Половина из этих методов работала на знаменитом SVM, придуманном в начале 1990-х Червоненкисом и Вапником. В середине 2000-х в любой известной конторе работали на SVM-е — и у нас, и в Яху, и в Гугле, и в Амазоне, и в Нетфликсе.

Аркадий Волож, сооснователь и генеральный директор Яндекса

Описание метода

Приведём описание метода, основанное на выдержках из книги James G., Witten D., Hastie T., Tibshirani R. «An Introduction to Statistical Learning: With Applications in R», 2013.

In a p-dimensional space, a *hyperplane* is a flat affine subspace of dimension $p-1$.[1] For instance, in two dimensions, a hyperplane is a flat one-dimensional subspace—in other words, a line. In three dimensions, a hyperplane is a flat two-dimensional subspace—that is, 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.

The mathematical definition of a hyperplane is quite simple. In two dimensions, a hyperplane is defined by the equation

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 = 0 \tag{9.1}$$

for parameters β_0, β_1, and β_2. When we say that (9.1) "defines" the hyperplane, we mean that any $X = (X_1, X_2)^T$ for which (9.1) holds is a point on the hyperplane.

[1] The word *affine* indicates that the subspace need not pass through the origin.

Что такое гиперплоскость

Equation 9.1 can be easily extended to the p-dimensional setting:

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p = 0 \tag{9.2}$$

defines a p-dimensional hyperplane, again in the sense that if a point $X = (X_1, X_2, \ldots, X_p)^T$ in p-dimensional space (i.e. a vector of length p) satisfies (9.2), then X lies on the hyperplane.

Now, suppose that X does not satisfy (9.2); rather,

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p > 0. \tag{9.3}$$

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

$$\beta_0 + \beta_1 X_1 + \beta_2 X_2 + \ldots + \beta_p X_p < 0, \tag{9.4}$$

then X lies on the other side of the hyperplane. So we can think of the 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 (9.2).

Разделяющие гиперплоскости

Suppose that it is possible to construct a hyperplane that separates the training observations perfectly according to their class labels. Examples of three such *separating hyperplanes* are shown in Figure 9.2.
We can label the observations from the blue class as $y_i = 1$ and those from the purple class as $y_i = -1$.

Then a separating hyperplane has the property that

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

and

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

Equivalently, a separating hyperplane has the property that

$$y_i(\beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \ldots + \beta_p x_{ip}) > 0$$

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

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.

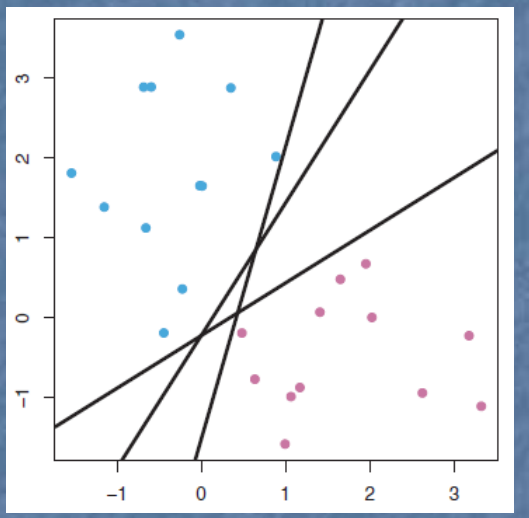

FIGURE 9.2.

The Maximal Margin Classifier

In general, if our data can be perfectly separated using a hyperplane, then there will in fact exist an infinite number of such hyperplanes. This is because a given separating hyperplane can usually be shifted a tiny bit up or down, or rotated, without coming into contact with any of the observations. Three possible separating hyperplanes are shown in the left-hand panel of Figure 9.2. In order to construct a classifier based upon a separating hyperplane, we must have a reasonable way to decide which of the infinite possible separating hyperplanes to use.

A natural choice is the *maximal margin hyperplane* (also known as the *optimal separating hyperplane*), which is the separating hyperplane that is farthest from the training observations. That is, we can compute the (perpendicular) distance from each training observation to a given separating hyperplane; the smallest such distance is the minimal distance from the observations to the hyperplane, and is known as the *margin*. The maximal margin hyperplane is the separating hyperplane for which the margin is largest—that is, it is the hyperplane that has the farthest minimum distance to 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*.

Опорные векторы

Examining Figure 9.3, we see that three training observations are equidistant from the maximal margin hyperplane and lie along the dashed lines indicating the width of the margin. These three observations are known as

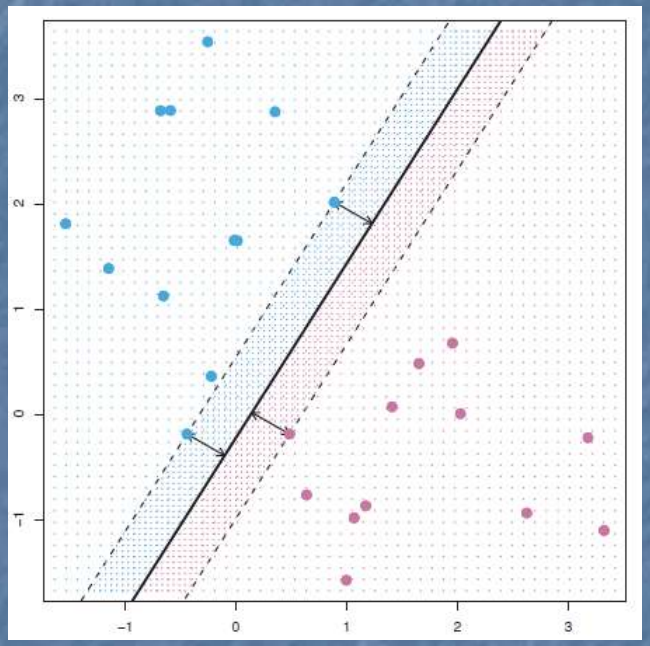

FIGURE 9.3.

support vectors, since they are vectors in p-dimensional space (in Figure 9.3, $p = 2$) and they "support" the maximal margin hyperplane in the sense that if these points were moved slightly then the maximal margin hyperplane would move as well.

Случай, когда не существует разделяющей гиперплоскости

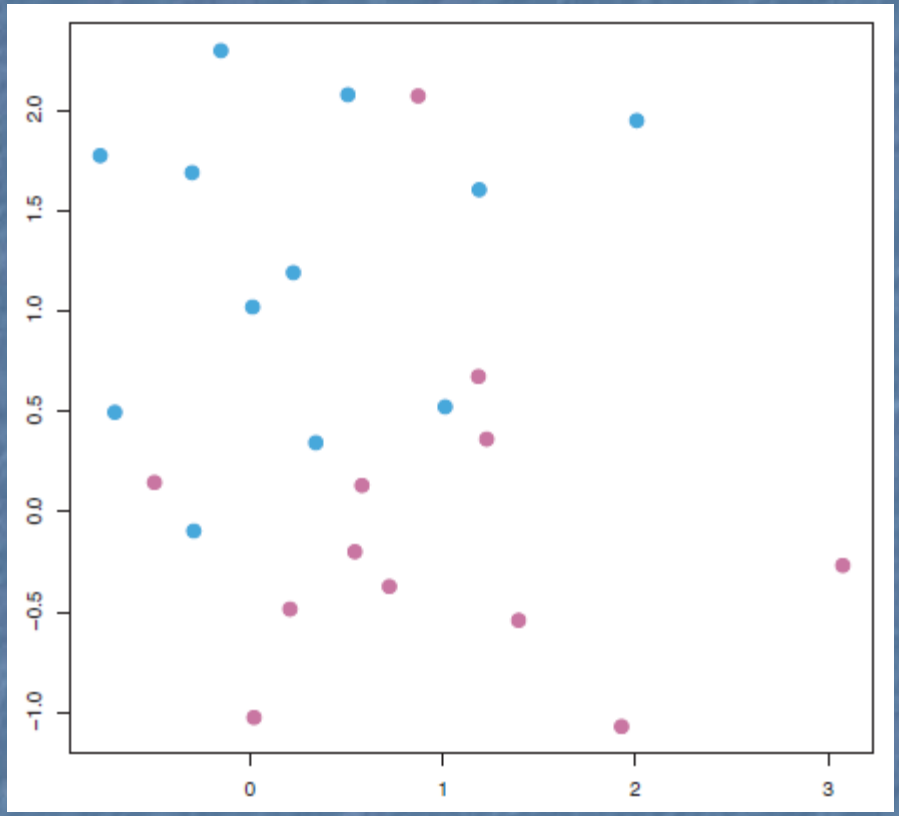

FIGURE 9.4. *There are two classes of observations, shown in blue and in purple. In this case, the two classes are not separable by a hyperplane, and so the maximal margin classifier cannot be used.*

Наглядная интерпретация

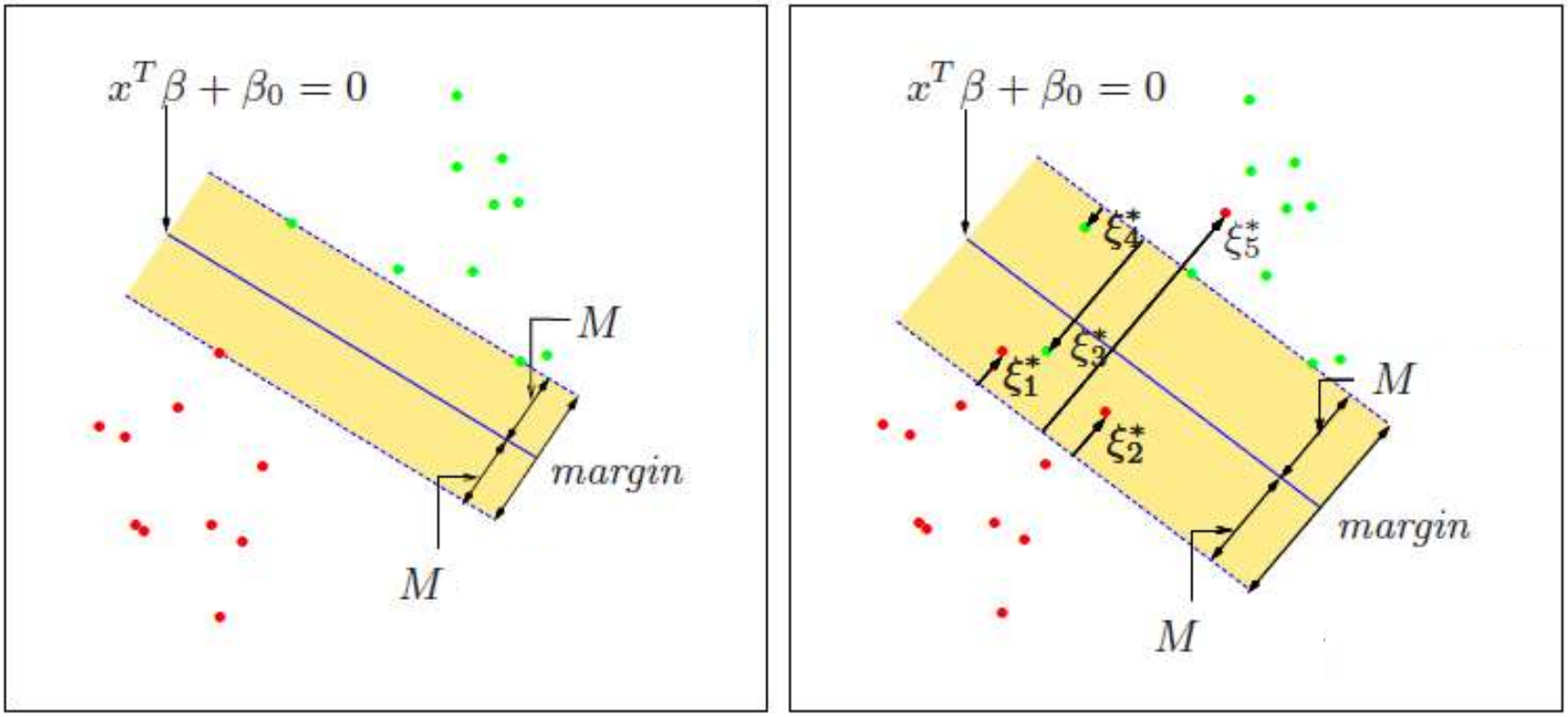

FIGURE 12.1. *Support vector classifiers. The left panel shows the separable case. The decision boundary is the solid line, while broken lines bound the shaded maximal margin of width* $2M$. *The right panel shows the nonseparable (overlap) case. The points labeled* ξ_j^* *are on the wrong side of their margin by an amount* $\xi_j^* = M\xi_j$; *points on the correct side have* $\xi_j^* = 0$. *The margin is maximized subject to a total budget* $\sum \xi_i \leq$ *constant. Hence* $\sum \xi_j^*$ *is the total distance of points on the wrong side of their margin.*

Математическая формулировка

Our training data consists of N pairs $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$, with $x_i \in \mathbb{R}^p$ and $y_i \in \{-1, 1\}$. Define a hyperplane by

$$\{x : f(x) = x^T\beta + \beta_0 = 0\}, \tag{12.1}$$

where β is a unit vector: $\|\beta\| = 1$. A classification rule induced by $f(x)$ is

$$G(x) = \text{sign}[x^T\beta + \beta_0]. \tag{12.2}$$

The geometry of hyperplanes is reviewed in Section 4.5, where we show that $f(x)$ in (12.1) gives the signed distance from a point x to the hyperplane $f(x) = x^T\beta + \beta_0 = 0$. Since the classes are separable, we can find a function $f(x) = x^T\beta + \beta_0$ with $y_i f(x_i) > 0 \ \forall i$. Hence we are able to find the hyperplane that creates the biggest *margin* between the training points for class 1 and -1 (see Figure 12.1). The optimization problem

$$\begin{aligned} &\max_{\beta, \beta_0, \|\beta\|=1} M \\ &\text{subject to } y_i(x_i^T\beta + \beta_0) \geq M, \ i = 1, \ldots, N, \end{aligned} \tag{12.3}$$

captures this concept. The band in the figure is M units away from the hyperplane on either side, and hence $2M$ units wide. It is called the *margin*.

Suppose now that the classes overlap in feature space. One way to deal with the overlap is to still maximize M, but allow for some points to be on the wrong side of the margin. Define the slack variables $\xi = (\xi_1, \xi_2, \ldots, \xi_N)$.

Let's modify the constraint in (12.3):

$$y_i(x_i^T\beta + \beta_0) \geq M(1 - \xi_i), \qquad \forall i, \ \xi_i \geq 0, \ \sum_{i=1}^N \xi_i \leq \text{constant}.$$

Support Vector Machines

The support vector classifier is a natural approach for classification in the two-class setting, if the boundary between the two classes is linear. However, in practice we are sometimes faced with non-linear class boundaries.

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*. We may want to 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 enacting this idea.

We have not discussed exactly how the support vector classifier is computed because the details become somewhat technical. However, it turns out that the solution to the support vector classifier problem (слайд 19) involves only the *inner products* of the observations (as opposed to the observations themselves). The inner product of two r-vectors a and b is defined as $\langle a, b \rangle = \sum_{i=1}^{r} a_i b_i$.

Представление SVM через скалярные произведения

It can be shown that

- The linear support vector classifier can be represented as

$$f(x) = \beta_0 + \sum_{i=1}^{n} \alpha_i \langle x, x_i \rangle, \qquad (9.18)$$

where there are n parameters α_i, $i = 1, \ldots, n$, one per training observation.

- To estimate the parameters $\alpha_1, \ldots, \alpha_n$ and β_0, all we need are the $\binom{n}{2}$ inner products $\langle x_i, x_{i'} \rangle$ between all pairs of training observations.

(The notation $\binom{n}{2}$ means $n(n-1)/2$, and gives the number of pairs among a set of n items.)

Notice that in (9.18), in order to evaluate the function $f(x)$, we need to compute the inner product between the new point x and each of the training points x_i. However, it turns out that α_i is nonzero only for the support vectors in the solution—that is, if a training observation is not a support vector, then its α_i equals zero.

Ядра SVM

Now suppose that every time the inner product $\langle x, x_i \rangle$ appears in the representation (9.18), 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

$$K(x, x_i),$$

where K is some function that we will refer to as a *kernel*.
For instance, one could replace $\langle x, x_i \rangle$ with the quantity

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

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

Another popular choice is the *radial kernel*, which takes the form

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

Различие между Support Vector Classifier и дискриминантным анализом

Support vector machines are a relatively recent development in classification, and their performance is often excellent. A support vector machine for a binary classification problem tries to find a hyperplane in multidimensional space such that ideally all elements of a given class are on one side of that hyperplane, and all the other elements are on the other side. Furthermore, it allocates a margin around that hyperplane, and points that are less than margin distance away from the hyperplane are called its support vectors. In other words, whereas discriminant analysis tries to separate groups by focusing on the group means, support vector machines target the border area where the groups meet, and seeks to set up a boundary there.

In summary, support vector machines are excellent classifiers and probably our best choice if the goal is to achieve optimal classification performance for an application. Their disadvantage is that they are difficult to interpret and provide little insight into what factors drive the classification.

Французская средневековая проза и поэзия

Ernestus *et al.* (2007) studied register variation and diachronic variation in the use of syntactic constructions in Medieval French. For 29 authors (some of whom are anonymous), and often for several manuscript versions of the same text, the counts of the 35 most frequent tag trigrams (tag triplets) were calculated. Texts with more than 2000 words were subdivided into chunks of 2000 words.

The data of this study are available in the form of two data frames. The `oldFrench` data frame contains the counts of tag trigrams (columns) for 342 texts (rows). The `oldFrench Meta` data frame provides metadata on these texts, including information on author, region of origin, date of composition, register, and topic.

The columns of `oldFrench` represent the frequencies of the tag trigrams in the text fragments. What we would like to know is whether there are systematic differences in the frequencies of these tag trigrams as a function of author, topic, genre, region, and time.

Практическое задание

1) Установите и подключите, поставив «галку», пакет e1071
2) Убедитесь, что подключён пакет languageR
3) Примените метод опорных векторов и выведите результаты:

```
s=svm(oldFrench, oldFrenchMeta$Genre); s
```

4) Постройте диаграмму рассеяния, используя многомерное шкалирование:

```
plot(cmdscale(dist(oldFrench)),
col=c("black", "darkgrey")[as.integer(oldFrenchMeta$Genre)],
pch=c("o", "+")[1:nrow(oldFrenchMeta) %in% s$index + 1])
```

(смысл параметров команд объясняется на след. двух слайдах)

5) Узнайте качество прогноза:

```
xtabs(~oldFrenchMeta$Genre + predict(s))
```

6) Выполните 10-кратную перепроверку адекватности модели:

```
s=svm(oldFrench, oldFrenchMeta$Genre, cross = 10);  summary(s)
```

Смысл параметров функции plot

The second and third lines of this plot command illustrate a feature of subscripting that has not yet been explained, namely, that a vector can be subscripted for more elements as it is long, provided that these elements refer to legitimate indices in the vector:

```
> c("black", "darkgrey")[c(1, 2, 1, 2, 2, 1)]
[1] "black"    "darkgrey" "black"    "darkgrey" "darkgrey" "black"
```

In the second line of the plot command, `as.integer(oldFrenchMeta$Genre)` is a vector with ones and twos, corresponding to the levels `poetry` and `prose`. This vector is mapped onto a vector with `black` representing `poetry` and `darkgrey` representing `prose`. The same mechanism is at work for the third line. The vector between the square brackets is dissected as follows. The index extracted from the model object,

```
> genre.svm$index
[1]   2   3   6  13  14  15  16  17
```

refers to the row numbers in `oldFrench` of the support vectors.

Продолжение описания параметров

The vector

```
1:nrow(oldFrenchMeta)
```

is the vector of all row numbers. The `%in%` operator checks for set membership. The result is a vector that is `TRUE` for the support vectors and `FALSE` for all other rows. When 1 is added to this vector, `TRUE` first converts to 1 and `FALSE` to zero, so the result is a vector with ones and twos, which are in turn mapped onto the `o` and `+` symbols.

Точность прогноза и перекрёстная проверка

```
oldFrenchMeta$Genre poetry prose
             poetry    198     0
             prose       1   143
```

```
10-fold cross-validation on training data:
Total Accuracy: 96.78363
Single Accuracies:
  97.05882 97.05882 97.05882 94.11765 97.14286 97.05882
  97.05882 97.05882 100 94.28571
```

Диаграмма рассеяния с изображением опорных векторов

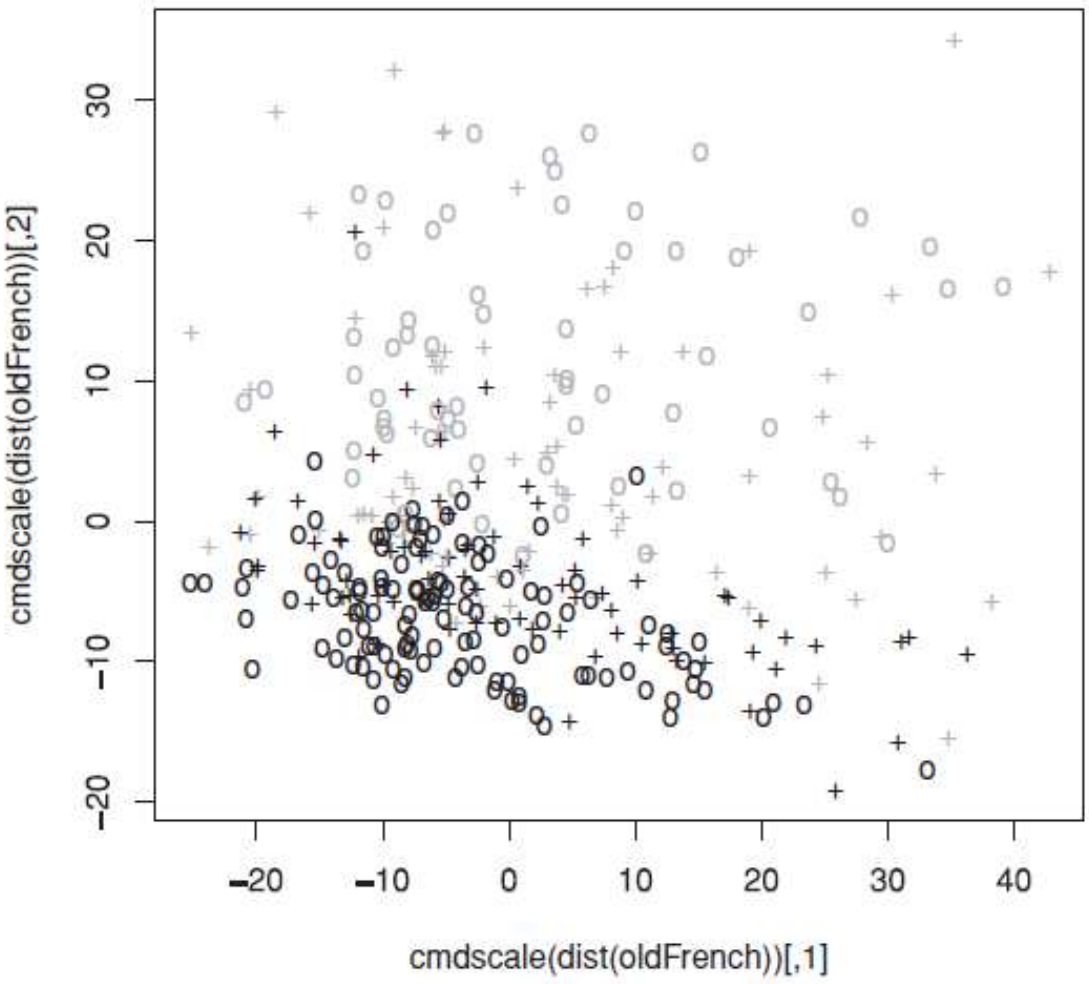

Figure 5.19. *Multidimensional scaling for registers in Medieval French on the basis of tag trigram frequencies, with support vectors highlighted by the plus symbol. Black points represent poetry, grey points represent prose.*

Подходы OVO и OVA

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 $\binom{K}{2}$ SVMs, each of which compares a pair of classes. For example, one such SVM might compare the kth class, coded as $+1$, to the k'th class, coded as -1. We classify a test observation using each of the $\binom{K}{2}$ classifiers, and we tally 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 it was most frequently assigned in these $\binom{K}{2}$ pairwise classifications.

The *one-versus-all* approach is an alternative procedure for applying SVMs in the case of $K > 2$ classes. We fit K SVMs, each time comparing one of the K classes to the remaining $K - 1$ classes. Let $\beta_{0k}, \beta_{1k}, \ldots, \beta_{pk}$ denote the parameters that result from fitting an SVM comparing the kth class (coded as $+1$) to the others (coded as -1). Let x^* denote a test observation. We assign the observation to the class for which $\beta_{0k} + \beta_{1k}x_1^* + \beta_{2k}x_2^* + \ldots + \beta_{pk}x_p^*$ is 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.