Support Vector Machines - V

Prof. Dan A. Simovici

UMB

Data Set Description

Attribute Information:

- sepal length in cm
- sepal width in cm
- petal length in cm
- petal width in cm
- class:
  - Iris Setosa
  - Iris Versicolour
  - Iris Virginica

Data Presentation

Data contains 150 records: 50 records for each class value: *setosa*, *versicolor*, and *virginica*.

```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
.
.
.
7.0,3.2,4.7,1.4,Iris-versicolor
6.4,3.2,4.5,1.5,Iris-versicolor
6.9,3.1,4.9,1.5,Iris-versicolor
5.5,2.3,4.0,1.3,Iris-versicolor
.
.
.
6.3,2.5,5.0,1.9,Iris-virginica
6.5,3.0,5.2,2.0,Iris-virginica
6.2,3.4,5.4,2.3,Iris-virginica
5.9,3.0,5.1,1.8,Iris-virginica
```

Uniform Distribution Generation

The data set is already grouped on class values; this requires a random rearrangement of the record in order to extract the training set and the test set.
Uniform distribution generation
The function runif generates n values of a random variable uniformly distributed in the interval $[m, M]$.
It is called using

$$> \text{runif}(n, m, M)$$

```
> runif(10,12, 20)
[1] 14.81854 13.33863 17.58722 15.75252 17.11880 13.99228 19.8
12.95395 [9] 18.50042 12.46879
```

If called with one argument n it produces n random values in the interval $[0, 1]$.

Ordering Permutation

The function order returns a permutation which rearranges its first argument into ascending or descending order, breaking ties by further arguments.

```
> iris_rand <- iris[order(runif(150)), ]
```

Classifier Generation

```
> iris_train <- iris_rand[1:120,]
> iris_test <- iris_rand[121:150,]
> iris_classifier <- ksvm(class ~ .,
+ data = iris_train, kernel = "vanilladot")
> iris_prediction <- predict(iris_classifier,iris_test)
> table(iris_prediction,iris_test$class)
```

Kernels available in kernlab

- The linear `vanilladot` is the simplest and is given by $K(\boldsymbol{u}, \boldsymbol{v}) = \boldsymbol{u}'\boldsymbol{v}$; this is useful when dealing with large sparse data vectors (typically text cathegorization).
- the Gaussian radial basis kernel `rbfdot` is $K(\boldsymbol{u}, \boldsymbol{v}) = e^{-\sigma\|\boldsymbol{u}-\boldsymbol{v}\|^2}$; a typical invocation is

$$\text{rbf} \ < -\text{rbfdot}(\text{sigma} = 0.05)$$

  This is a general kernel and is used when no further prior knowledge exists about data.
- The polynomial kernel `polydot` $K(\boldsymbol{u}, \boldsymbol{v}) = (k\boldsymbol{u}'\boldsymbol{v} + c)^d$ frequently used in image classification.

- The hyperbolic tangent kernel `tanhdot` is

$$K(\boldsymbol{u}, \boldsymbol{v}) = \tanh(k\boldsymbol{u}'\boldsymbol{v} + c)$$

  mainly used as an alternative to neural networks.
- The Laplace radial basis kernel `laplacedot`

$$K(\boldsymbol{u}, \boldsymbol{v}) = e^{-\sigma\|\boldsymbol{u}-\boldsymbol{v}\|}$$

  is a general purpose kernel.
- the ANOVA radial basis kernel `anovadot`

$$K(\boldsymbol{u}, \boldsymbol{v}) = \left(\sum_{i=1}^{n} e^{-\sigma(u_i - v_i)^2}\right)^d$$

  used in multidimensional regression problems.

Example

```
> letter <−read.csv("letter-recognition.csv",header=TRUE,sep=",")
> letters_train <− letter[1:16000,]
> letters_test <− letter[16001:20000,]
> letter_classifier <− ksvm(lettr .,data = letters_train,kernel="rbfdot")
Using automatic sigma estimation (sigest) for RBF or laplace kernel
> letter-classifier
Error: object 'classifier' not found
> letter_classifier
Support Vector Machine object of class "ksvm"
SV type: C-svc (classification)
parameter : cost C = 1
Gaussian Radial Basis kernel function.
Hyperparameter : sigma = 0.0474609039404198
Number of Support Vectors : 8680
Objective Function Value : -43.1068 -33.8779 -59.0838 -27.2155 -34.6708 -46.8762 ....
Training error : 0.051625
```

Example

```
> letter_classifier <- ksvm(lettr  .,data = letters_train,kernel="polydot")
Setting default kernel parameters
> letter_classifier
Support Vector Machine object of class "ksvm"
SV type: C-svc (classification)
parameter : cost C = 1
Polynomial kernel function.
Hyperparameters : degree = 1 scale = 1 offset = 1
Number of Support Vectors : 7035
Objective Function Value : -14.1746 -20.0072 -23.5628 -6.2009 -7.5524 -32.7694 ....
Training error : 0.130125
```

Example

```
> letter_classifier <- ksvm(lettr  .,data = letters_train,kernel= "tanhdot")
Setting default kernel parameters
> letter_classifier
Support Vector Machine object of class "ksvm"
SV type: C-svc (classification)
parameter : cost C = 1
Hyperbolic Tangent kernel function.
Hyperparameters : scale = 1 offset = 1
Number of Support Vectors : 15696
Objective Function Value : -15157.29 -1786.306 -15642.6 -5531.012 -1218.474 -14029.91 ...
Training error : 0.910875
```

Example

```
> letter_classifier <- ksvm(lettr  .,data = letters_train,kernel="laplacedot")
Using automatic sigma estimation (sigest) for RBF or laplace kernel
> letter_classifier
Support Vector Machine object of class "ksvm"
SV type: C-svc (classification)
parameter : cost C = 1
Laplace kernel function.
Hyperparameter : sigma = 0.0477332265453678
Number of Support Vectors : 11331
Objective Function Value : -101.5121 -67.578 -131.9846 -70.7183 -77.3382 -109.682 ...
Training error : 0.084875
```

Example

```
> letter_classifier <- ksvm(lettr  .,data = letters_train,kernel="anovadot")
Setting default kernel parameters
> letter_classifier
Support Vector Machine object of class "ksvm"
SV type: C-svc (classification)
parameter : cost C = 1
Anova RBF kernel function.
Hyperparameter : sigma = 1 degree = 1
Number of Support Vectors : 6636
Objective Function Value : -8.7926 -9.3741 -12.0187 -6.6614 -5.8274 -16.8295 ...
Training error : 0.032687
```