Unidad 6

Diego Vallarino

1/5/2021

Contents

1 ANALISIS TEORICO DEL MODELO SVM **1**

1.1 Fortalezas y Debilidades del SVM 1

2 DESARROLLO DEL MODELO **2**

2.1 Paso UNO - Levantar los Datos, Transformarlos y Analizarlos 2

2.2 Paso DOS - Desarrollo y Entrenamiento del Modelo 2

2.3 Paso TRES: Evaluacion del Modelo 2

2.4 Paso CUATRO: Performance de cada modelo 3

2.5 Paso CINCO: Mejorar la performance de cada modelo 4

REFERENCE **5**

1 ANALISIS TEORICO DEL MODELO SVM

Una máquina de vectores de soporte (SVM) se puede imaginar como una superficie que define un límite entre varios puntos de datos que representan ejemplos trazados en espacio multidimensional de acuerdo con sus valores de características.

El objetivo de una SVM es para crear un límite plano, llamado hiperplano, que conduce a particiones bastante homogéneas de datos en ambos lados. De esta manera, el aprendizaje de SVM combina aspectos de "del vecino más cercano"" basado en instancias presentado en el Capítulo 3, *Lazy Learning* -Clasificación mediante vecinos más cercanos y el modelo de regresión lineal descrito en el Capítulo 6, *Pronóstico de datos numéricos: métodos de regresión*. La combinación es extremadamente potente, lo que permite a las SVM modelar relaciones muy complejas.. Lantz (2015)

1.1 Fortalezas y Debilidades del SVM

Las fortalezas y debilidades de este algoritmo son de la siguiente maneras:

Fortalezas	**Debilidades**
* Poco propenso a sobre ajustar * No muy influenciado por datos ruidosos	* Encontrar el mejor modelo requiere prueba de varias combinaciones * Da como resultado una caja negra compleja

Fortalezas	Debilidades
* Es más simple de usar que el modelo de redes neuronales	* Requiere una gran cantidad de memoria
* Alta precision en su modelamiento	* Puede ser lento para entrenar

2 DESARROLLO DEL MODELO

2.1 Paso UNO - Levantar los Datos, Transformarlos y Analizarlos

Tenemos un conjunto de datos con **2001** variables y **62** observaciones. Estos son dinamicas por lo que se pueden modificar en funcion de los parametros elegidos.

```
library(readr)
colon2 <- read_csv("D:/Master en BioEstadistica/Materias/2.Machine Learning/6/colon2.csv")
data<-colon2
data$y<-as.factor(data$y)
```

2.2 Paso DOS - Desarrollo y Entrenamiento del Modelo

```
# creamos training & test data automatizados y parametrizados
set.seed(1234)
data_train <- data[1:41, ]
data_test  <- data[42:62, ]

#Creamos el modelo
library(kernlab)
tejido_classifier <- ksvm(y ~ ., data = data_train)

tejido_classifier
```

```
## Support Vector Machine object of class "ksvm"
##
## SV type: C-svc  (classification)
##  parameter : cost C = 1
##
## Gaussian Radial Basis kernel function.
##  Hyperparameter : sigma =  0.000372948272766689
##
## Number of Support Vectors : 35
##
## Objective Function Value : -18.0475
## Training error : 0.073171
```

2.3 Paso TRES: Evaluacion del Modelo

```
tejido_predictions <- predict(tejido_classifier, data_test)

head(tejido_predictions)
```

```
## [1] t t t t t t
## Levels: n t
```

```
table(tejido_predictions, data_test$y)
```

```
##
## tejido_predictions  n  t
##                  n  1  1
##                  t  8 11
```

2.4 Paso CUATRO: Performance de cada modelo

```
agreement <- tejido_predictions == data_test$y
table(agreement)
```

```
## agreement
## FALSE  TRUE
##     9    12
```

```
prop.table(table(agreement))
```

```
## agreement
##     FALSE      TRUE
## 0.4285714 0.5714286
```

```
library(caret)
matriz<-confusionMatrix(tejido_predictions, data_test$y)
matriz
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  n  t
##          n  1  1
##          t  8 11
##
##                Accuracy : 0.5714
##                  95% CI : (0.3402, 0.7818)
##     No Information Rate : 0.5714
##     P-Value [Acc > NIR] : 0.5909
##
##                   Kappa : 0.0308
##
##  Mcnemar's Test P-Value : 0.0455
```

```
##
##             Sensitivity : 0.11111
##             Specificity : 0.91667
##          Pos Pred Value : 0.50000
##          Neg Pred Value : 0.57895
##              Prevalence : 0.42857
##          Detection Rate : 0.04762
##    Detection Prevalence : 0.09524
##       Balanced Accuracy : 0.51389
##
##        'Positive' Class : n
##
```

2.5 Paso CINCO: Mejorar la performance de cada modelo

```
set.seed(12345)
tejido_classifier_rbf <- ksvm(y ~ ., data = data_train, kernel = "rbfdot")
tejido_predictions_rbf <- predict(tejido_classifier_rbf, data_test)

agreement_rbf <- tejido_predictions_rbf == data_test$y
table(agreement_rbf)
```

```
## agreement_rbf
## FALSE  TRUE
##     8    13
```

```
prop.table(table(agreement_rbf))
```

```
## agreement_rbf
##     FALSE      TRUE
## 0.3809524 0.6190476
```

```
matriz2<-confusionMatrix(tejido_predictions_rbf, data_test$y)
matriz2
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  n  t
##          n  1  0
##          t  8 12
##
##                Accuracy : 0.619
##                  95% CI : (0.3844, 0.8189)
##     No Information Rate : 0.5714
##     P-Value [Acc > NIR] : 0.41709
##
##                   Kappa : 0.125
##
##  Mcnemar's Test P-Value : 0.01333
```

```
##
##             Sensitivity : 0.11111
##             Specificity : 1.00000
##          Pos Pred Value : 1.00000
##          Neg Pred Value : 0.60000
##              Prevalence : 0.42857
##          Detection Rate : 0.04762
##    Detection Prevalence : 0.04762
##       Balanced Accuracy : 0.55556
##
##        'Positive' Class : n
##
```

REFERENCE

Lantz, Brett. 2015. *Machine Learning with r*. Packt Publishing Ltd.