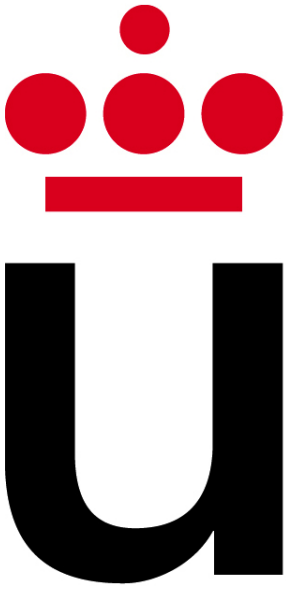
Caso Práctico III

Minería de datos

Estimación y predicción con máquinas de vectores soporte

José Ignacio Escribano



Móstoles, 12 de diciembre de 2015

Índice de figuras

Índice de tablas

Índice

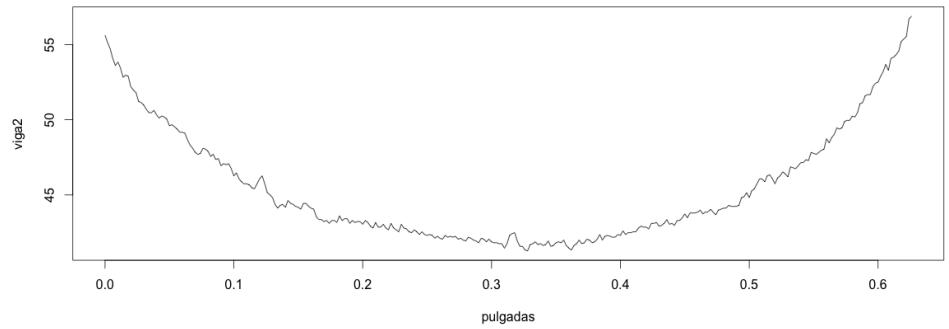


Figura 1: Función de la distancia frente a la densidad

1. Introducción

En este caso práctico veremos cómo aplicar las máquinas de vectores soporte para predecir la densidad de una viga. Aplicaremos distintos tipos de kernel para mejorar el ajuste. Por último, calcularemos el número mínimo de densidades para conseguir que la función de ajuste siga las especificaciones marcadas por los ingenieros.

2. Resolución de las cuestiones de evaluación

A continuación, resolveremos las cuestiones de evaluación planteadas.

2.1. Cuestión 1

Comenzando representando la función de la distancia (en pulgadas) frente a la densidad de la viga2 (Figura 1).

Seleccionamos 25 puntos al azar para intentar predecir a partir de éstos los restantes. Los puntos elegidos al azar son los siguientes:

```
53.44 50.18 48.67 46.94 45.15 44.44 43.55 43.30 43.61
42.55 42.31 42.37 42.19 42.06 41.76 41.70 41.64 42.31
43.11 43.11 46.34 46.27 47.28 52.20 56.71
```

Representamos los datos de entrenamiento en un diagrama de dispersión (Figura 2)

Con estos datos entrenaremos el SVM, y los restantes harán de test para comprobar la precisión del ajuste calculado.

Probamos el SVM con un kernel lineal y comprobamos la bondad del ajuste (Figura 3).

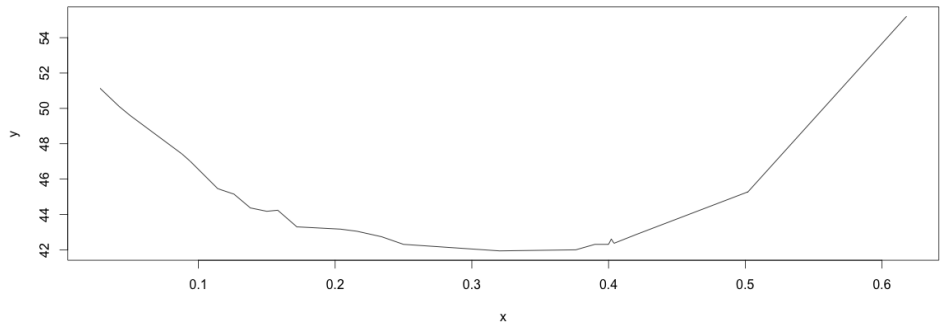


Figura 2: Función de la distancia frente a la densidad con los datos de entrenamiento

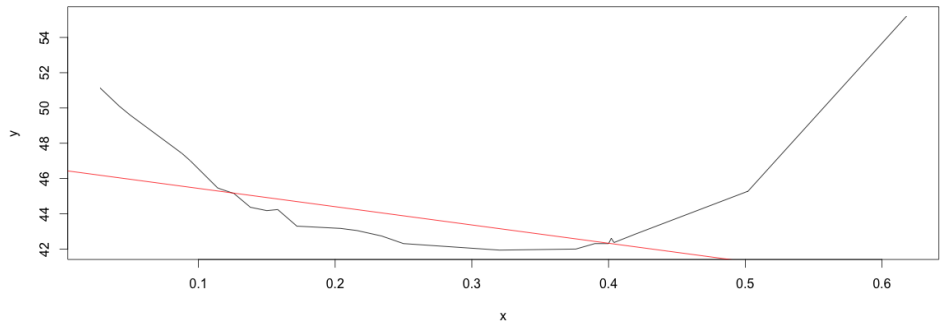


Figura 3: Predicción con un kernel lineal

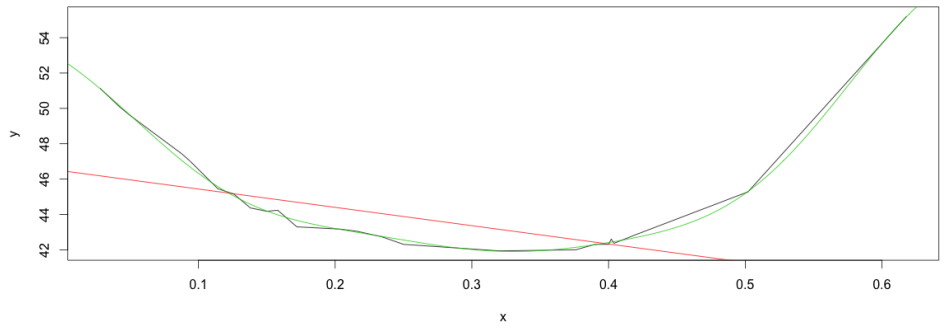


Figura 4: Predicción con un kernel no lineal

Vemos que este kernel da un muy mal ajuste, por lo que probamos con un kernel no lineal con parámetros épsilon = 2, C = 100 y kernel radial con gamma = 40. La predicción de este nuevo modelo se puede ver en la Figura 4.

Observamos que este modelo, sí ajusta con bastante exactitud la función de la densidad.

Calculamos la media y la desviación típica del error del modelo. La media es 0.2701867 y la desviación típica, 0.2947849, valores muy bajos, por debajo de los valores especificados por los ingenieros.

2.2. Cuestión 2

De acuerdo a los datos de media y desviación típica del error, tenemos que el error se encuentra bastante por debajo de 5, el valor especificado por los ingenieros como el límite de la calidad de la viga. Por tanto, tomando sólo 25 muestras, tenemos un excelente modelo que permitirá ahorrar costes.

Los datos proporcionados al fabricante serán los 25 datos de entrenamientos juntos con los 289 predichos por el SVM. Ésto es, los datos de entrenamiento son:

```
53.44 50.18 48.67 46.94 45.15 44.44 43.55 43.30 43.61
42.55 42.31 42.37 42.19 42.06 41.76 41.70 41.64 42.31
43.11 43.11 46.34 46.27 47.28 52.20 56.71
```

y los datos predichos

```
53.28458 53.17671 53.06670 52.95461 52.84048 52.72439 52.60639
52.48655 52.36492 52.24157 51.86188 51.73232 51.60138 51.46912
51.33563 51.20096 51.06520 50.92842 50.79069 50.51270 50.37258
50.23183 50.09051 49.94870 49.80649 49.66395 49.52117 49.37822
```

```
49.23519 49.09215 48.94918 48.80638 48.66381 48.52157 48.37973
48.23837 48.09757 47.95742 47.81800 47.54165 47.40488 47.13453
47.00112 46.73816 46.60877 46.48087 46.35453 46.22980 46.10677
45.98550 45.86603 45.74845 45.63280 45.51914 45.40752 45.29800
45.19061 44.98244 44.88172 44.78331 44.68723 44.59351 44.50217
44.41324 44.32673 44.24266 44.16104 44.08187 44.00515 43.93090
43.85909 43.78972 43.72278 43.65825 43.59610 43.53632 43.47888
43.42374 43.27176 43.22543 43.18119 43.13898 43.09875 43.06044
43.02398 42.98932 42.95638 42.92510 42.89540 42.86721 42.84046
42.81507 42.79096 42.76807 42.74630 42.72558 42.70584 42.68699
42.66895 42.65165 42.63501 42.61895 42.60340 42.58829 42.57355
42.55911 42.54489 42.53085 42.51691 42.50302 42.48913 42.47517
42.46112 42.44691 42.41788 42.40300 42.38783 42.37234 42.35653
42.34037 42.32386 42.30698 42.27213 42.25418 42.23588 42.21725
42.15964 42.13995 42.12008 42.07994 42.05976 42.03958 42.01944
41.99939 41.97951 41.95984 41.94044 41.92138 41.90273 41.88453
41.86687 41.84980 41.83339 41.81771 41.80281 41.77565 41.76350
41.75239 41.74237 41.73351 41.72585 41.71944 41.71433 41.71057
41.70820 41.70724 41.70975 41.71326 41.71831 41.72491 41.73308
41.74283 41.75416 41.76708 41.78157 41.79764 41.83443 41.85512
41.87730 41.90095 41.92604 41.95253 41.98038 42.00954 42.07164
42.10448 42.13843 42.17345 42.20947 42.24645 42.28432 42.32301
42.36248 42.40267 42.48492 42.52689 42.56932 42.61219 42.65542
42.69897 42.74279 42.78684 42.83107 42.87546 42.96453 43.00916
43.05384 43.09853 43.14323 43.18794 43.23266 43.27738 43.32213
43.36691 43.41176 43.45670 43.50176 43.54699 43.59243 43.63814
43.68417 43.73059 43.77747 43.82488 43.87291 43.92163 43.97114
44.02153 44.07290 44.12536 44.17901 44.23396 44.34820 44.40773
44.46902 44.53220 44.59738 44.66469 44.73424 44.80617 44.88058
44.95760 45.03734 45.11993 45.20546 45.29405 45.38581 45.48084
45.57922 45.68106 45.78643 45.89543 46.00811 46.12455 46.24480
46.36893 46.49697 46.62896 46.76492 46.90489 47.04886 47.19684
47.34883 47.50480 47.66473 47.82859 47.99632 48.34320 48.52220
48.70480 48.89090 49.08041 49.27321 49.46918 50.07474 50.28199
50.49166 50.70360 50.91761 51.13351 51.35111 51.79060 52.01208
52.23442 52.45741 52.68083 52.90444 53.12802 53.35134 53.57415
53.79622 54.01732 54.23721 54.45565 54.67240 54.88723 55.09990
55.31018 55.51784
```

La Figura 5 muestra estos datos en forma de gráfica.

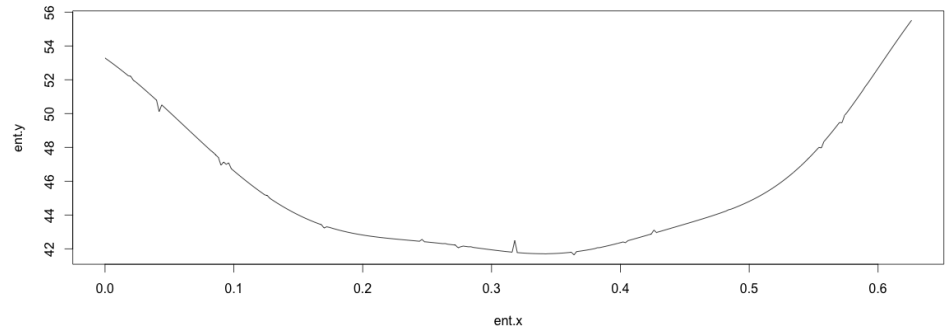


Figura 5: Datos proporcionados al fabricante

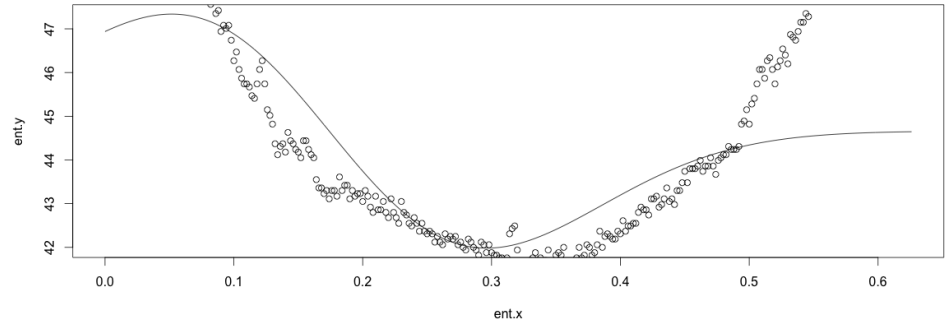


Figura 6: Función predicha por con dos vectores soporte

2.3. Cuestión 3

Para obtener el número mínimo de densidades para que el modelo siendo válido, es decir, el error absoluto en las predicciones sea mínimo, tomamos una muestra de 2 densidades aleatoriamente, entrenamos el modelo, y lo verificamos con los datos restantes. Si se cumple que el error absoluto es menor que 5, nos quedamos con él, en caso contrario aumentamos en una unidad el número de densidades.

Implementando el algoritmo anterior en R, tenemos que este número mínimo de densidades es 2. En la Figura 6 se puede ver la función predicha por este SVM junto a la función original.

Vemos que en algunos lugares el ajuste es bastante bueno, mientras que en otros es bastante malo, aunque el ajuste cumple con las especificaciones fijadas por los ingenieros (Tabla 1), aunque tiene una desviación típica del error muy alta.

Según vamos aumentando el número de vectores soportes, se va mejorando el ajuste de

Tabla 1: Media y desviación típica del error absoluto según el número de densidades

Número de densidades	Media	Desviación típica
2	1.851380	2.507192
3	1.957093	2.747603
4	2.324786	3.189393
5	1.077422	2.006100
6	1.008247	1.986431
7	1.097909	1.799587
8	0.877396	1.361815
9	0.363463	0.341001

la función (Figura 7).

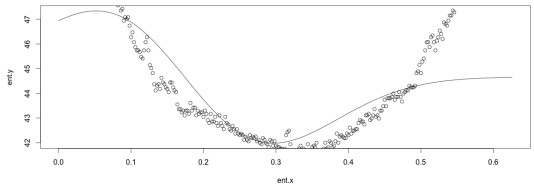
A tenor de la Tabla 1 y la Figura 7, todos los SVM entre 2 y 9 vectores soporte cumplen con las especificaciones de los ingenieros, aunque algunos debidos a la variabilidad de los errores no parecen óptimos.

En el caso de 9 vectores soporte, se observa un acusado descenso del promedio y de la desviación típica del error, por lo que parece un número de densidades asequible para ser utilizado, con su reducción de coste asociado, aunque se pueden elegir cualquiera de los demás modelos desde 2 hasta 8 vectores soporte ya que cumplen todos con las especificaciones de los ingenieros.

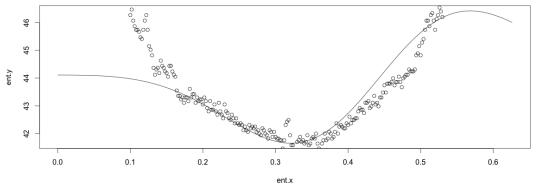
3. Conclusiones

En este caso práctico, hemos puesto en práctica lo aprendido sobre máquinas de vectores soporte, aplicándolo a una aplicación real como puede ser la predicción de la densidad de una viga. También, hemos visto cómo mejorar la predicción de una función, usando el "kernel trick".

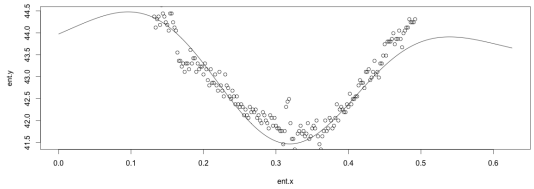
El software científico R nos ha ayudado mucho debido a las librerías que implementan los SVM, incluyendo distintos tipos de kernel, que nos han ahorrado mucho tiempo gracias a su potencia de cálculo.



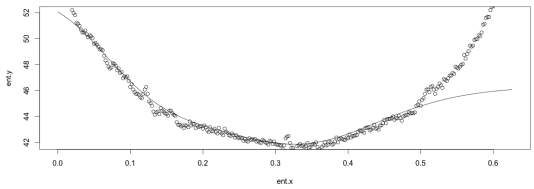
(a) 2 vectores soporte



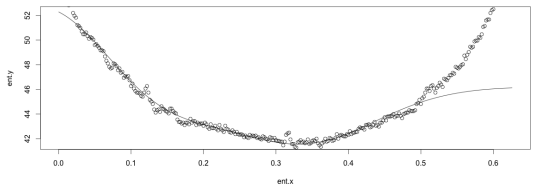
(b) 3 vectores soporte



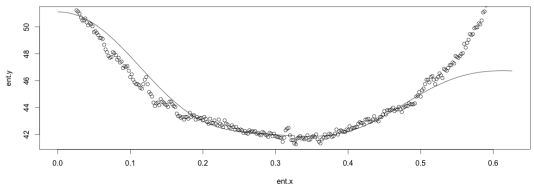
(c) 4 vectores soporte



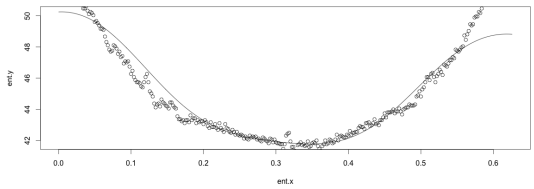
(d) 5 vectores soporte



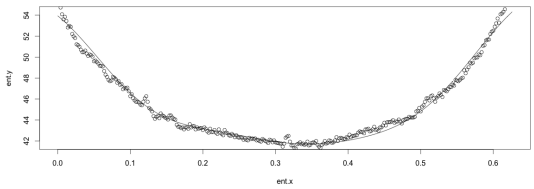
(e) 6 vectores soporte



(f) 7 vectores soporte



(g) 8 vectores soporte



(h) 9 vectores soporte

Figura 7: Función predicha por con distintos número de vectores soporte

4. Código R utilizado

```
#-----------------------------------------------------------
# Cuestiones de evaluación
#-----------------------------------------------------------

#-----------------------------------------------------------
# Cuestión 1
#-----------------------------------------------------------

# Cargamos la librería e1071
library(e1071)

# Cargamos el archivo de datos adamantasa.txt
adamantasa <- read.table(file.choose())

# Creamos variables propias para cada viga
pulgadas <- adamantasa[,1]
viga1 <- adamantasa[,2]
viga2 <- adamantasa[,3]

# Representamos la curva definida por los datos de densidades
plot(pulgadas, viga2, type='l')

# Entrenamos el SVM con 25 puntos elegidos al azar
ind.train <- sample(1:314, 25)
ind.train <- sort(ind.train)

# Variable con los cálculos de densidad de los 25 puntos
viga.train <- viga2[ind.train]

# Variable con los cálculos de densidad de los 289 puntos
viga.test <- viga2[-ind.train]

# Variable que será el punto del diámetro de la viga en la
# que se calcula la densidad
x <- pulgadas[ind.train]

y <- viga.train

# Representamos la curva con los 25 datos de entrenamiento
plot(x,y, type='l')
```

```
# Kernel lineal
x.svm <- svm(x, y, type="eps-regression", kernel="linear",
    epsilon=0.02, cost=100, scale=F)

x.test <- pulgadas[-ind.train]

# Predecimos los valores de la densidad
new <- predict(x.svm, x.test)

# Gráfico con la predicción
plot(x, y, type='l')
points(x.test, new, type='l', col=2)

# Kernel no lineal con epsilon = 2, C = 100, gamma = 40
x.svm2 <- svm(x,y, type="eps-regression", kernel="radial",
    epsilon=0.02, gamma=40, cost=100, scale=F)

# Nueva predicción
new <- predict(x.svm2, x.test)
points(x.test, new, type='l', col=3)

# Media y desviación típica del modelo
mean(abs(viga.test-new))
sd(abs(viga.test-new))

# Datos para presentar al fabricante
ord <- sort(c(x.test,x), index.return=T)
ent.x <- ord$x
ent.ind <- ord$ix
ent.y <- c(new, viga.train)[ent.ind]
plot(ent.x, ent.y, pch=".", type="l")


#-----------------------------------------------------------------
# Cuestión 2
#-----------------------------------------------------------------

# Calculamos la media y la desviación típica del modelo
mean(abs(viga.test-new))
sd(abs(viga.test-new))


#-----------------------------------------------------------------
```

```
# Cuestión 3
#------------------------------------------------------------

# Calculamos el número mínimo de densidades para el que el
# modelo siga siendo válido

densidades <- 2
densidades_totales <- 314
ESPECIFICACION <- 5 # Especificación de los ingenieros
esp_actual <- 1000
encontrado <- FALSE
media <- 100
while(esp_actual > ESPECIFICACION &&
    densidades <= densidades_totales && !encontrado){

  # Seleccionamos una muestra como entrenamiento
  ind.train <- sample(1:densidades_totales, densidades)
  ind.train <- sort(ind.train)

  # Entrenamiento viga
  viga.train <- viga2[ind.train]

  # Test de viga
  viga.test <- viga2[-ind.train]

  # Variable que será el punto del diámetro de la viga en la
  #  que se calcula la densidad
  x <- pulgadas[ind.train]

  y <- viga.train

  x.svm2 <- svm(x,y, type="eps-regression", kernel="radial",
      epsilon=0.02, gamma=40, cost=100, scale=F)

  x.test <- pulgadas[-ind.train]

  # Nueva predicción
  new <- predict(x.svm2, x.test)

  # Media y desviación típica del error
  media <- mean(abs(viga.test-new))
  desv  <- sd(abs(viga.test-new))
```

```
  # Objeto vacío que contendrá toda la información
  mejor <- NULL

  if(media < ESPECIFICACION){
    mejor$densidades <- densidades
    mejor$svm <- x.svm2
    mejor$media <- media
    mejor$desv <- desv
    mejor$train <- viga.train
    mejor$test <- x.test
    mejor$new <- new
    mejor$pulgadas <- x
    encontrado <- TRUE
    esp_actual <- media
  }

  densidades <- densidades + 1
}

ord <- sort(c(mejor$test,mejor$pulgadas), index.return=T)
ent.x <- ord$x
ent.ind <- ord$ix
ent.y <- c(new, viga.train)[ent.ind]
plot(ent.x, ent.y, pch=".", type="l")
points(pulgadas, viga2)
```