4_SVM

2023-02-03

Support Vector Machines

Tunning de support vector machines o SVM

SVM Lineal Para el SVM lineal se necesita la constante de regularización C. A continuación se construyen un par de modelos buscando un valor C optimo utilizando las variables seleccionadas con *stepwise*:

Como en los modelos anteriores, se utiliza el archivo dengue.RDS que contiene los datos preprocesados

```
# Carga de datos
dengue <- readRDS(file = "./data/processed/dengue.rds")

# 1. SVM -------------
# 1.1 SVM lineal -----

SVMgrid <- expand.grid(C = c(0.01, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10))
set.seed(12345)
control_svm <- trainControl(method = "cv", number = 4, savePredictions = "all")

svm_1 <-train(
    data = dengue,
    factor(varObjBin) ~ h1Opix + temp90 + trees + trees90 + Ymin +
    Ymax + temp + humid + humid90,
    method = "svmLinear",
    trControl = control_svm,
    tuneGrid = SVMgrid,
    verbose = FALSE)
```

Se muestran los resultados y el gráfico correspondiente

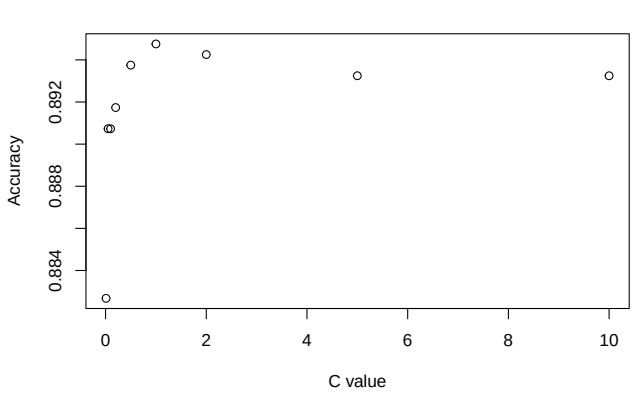
```
#svm_1
knitr::kable(svm_1$results, "pipe")
```

C	Accuracy	Kappa	AccuracySD	KappaSD
0.01	0.8826781	0.7639678	0.0019727	0.0047681
0.05	0.8907305	0.7804037	0.0061257	0.0123735
0.10	0.8907295	0.7801867	0.0073525	0.0145912
0.20	0.8917365	0.7820591	0.0071642	0.0142584
0.50	0.8937506	0.7860458	0.0069711	0.0137572
1.00	0.8947566	0.7880580	0.0082336	0.0165615
2.00	0.8942526	0.7869114	0.0090433	0.0185537
5.00	0.8932476	0.7848042	0.0076308	0.0159467
10.00	0.8932466	0.7847956	0.0084889	0.0177490

```
svm_1$bestTune
```

```
##   C
## 6 1
```

```
plot(svm_1$results$C, svm_1$results$Accuracy, xlab = "C value", ylab = "Accuracy",
     main = "Resultados svm lineal 1")
```



Observando bestTune y el gráfico, para este primer modelado de svm el valor C mas adecuado estaría entre 0.20 y 2 (donde se observa mejor *accuracy* en tabla), caret en bestTune indica un valor de C de 1 que es también lo que se observa en el gráfico. Con este nuevo valor de C se construye otro modelo donde se busca si es que hay un mejor valor de C en un grid de 0.1 hasta 1

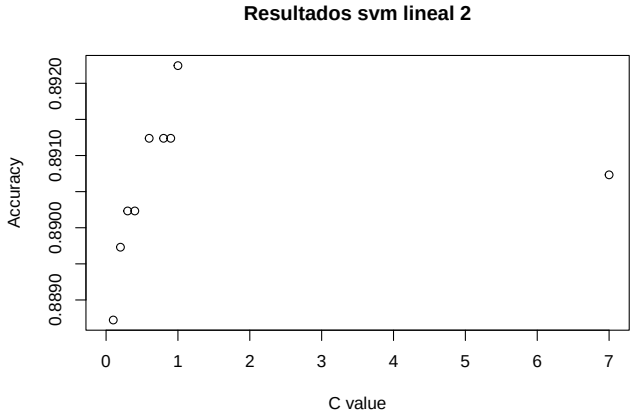
```
SVMgrid_1 <- expand.grid(C = c(0.1, 0.2, 0.3, 0.4, 0.6, 0,7, 0.8, 0.9, 1))

svm_2 <- train(
    data = dengue,
    factor(varObjBin) ~ h10pix + temp90 + trees + trees90 + Ymin +
    Ymax + temp + humid + humid90,
    method = "svmLinear",
    trControl = control_svm,
    tuneGrid = SVMgrid_1,
    verbose = FALSE)
```

Se despliegan los resultados

C	Accuracy	Kappa	AccuracySD	KappaSD
0.0	NaN	NaN	NA	NA
0.1	0.8887225	0.7759217	0.0149168	0.0299117
0.2	0.8897305	0.7778107	0.0160410	0.0321021
0.3	0.8902325	0.7787794	0.0151992	0.0304367
0.4	0.8902325	0.7787794	0.0151992	0.0304367
0.6	0.8912386	0.7807338	0.0139547	0.0278815
0.8	0.8912386	0.7807338	0.0139547	0.0278815
0.9	0.8912386	0.7807338	0.0139547	0.0278815
1.0	0.8922446	0.7828480	0.0129108	0.0256217
7.0	0.8907325	0.7797644	0.0103568	0.0203075

```
##   C
## 9 1
```



Para este dataset con SVM lineal el valor de C mas adecuado seria C=1.

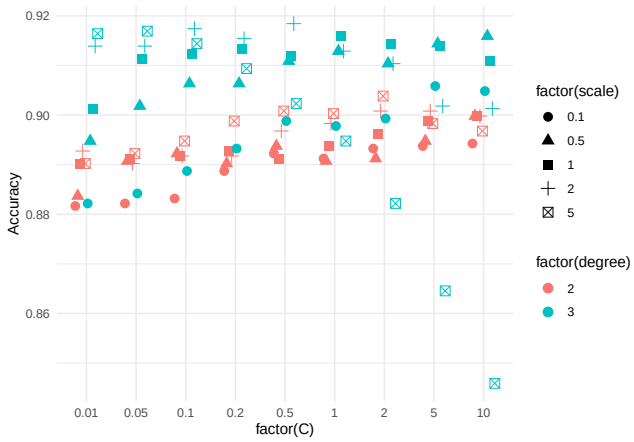
SVM Polinomial En este segundo apartado se utiliza un kernel polinomial buscando el valor de C, el grado del polinomio y la escala (C, degree y scale respectivamente), se entrena el modelo:

```
# 1.2 SVM Polinomial ----
SVMgrid_p <- expand.grid(
    C = c(0.01, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10),
    degree = c(2, 3),
    scale = c(0.1, 0.5, 1, 2, 5))

svm_3 <- train(
    data = dengue,
    factor(varObjBin) ~ h10pix + temp90 + trees + trees90 +
    Ymin + Ymax + temp + humid + humid90,
    method = "svmPoly",
    trControl = control_svm,
    tuneGrid = SVMgrid_p,
    verbose = FALSE)
```

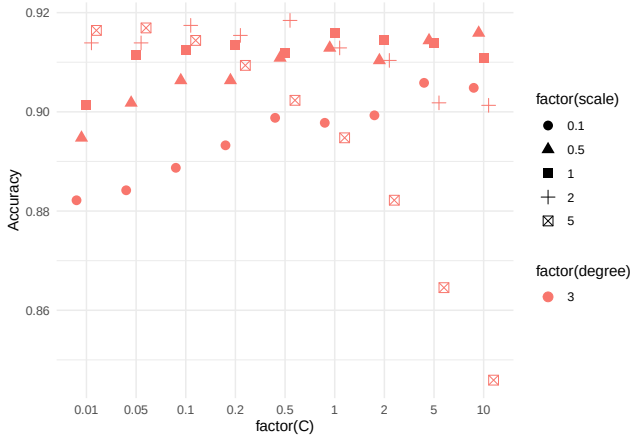
Se despliegan los resultados

	degree	scale	C
49	3	2	0.5



En los resultados con bestTune se sugiere un degree de 3, scale 2 y un c value de 0.5. En el gráfico los mejores resultados en *accuracy* los muestra con degree 3, para c value muestra buenos resultados de 0.5 en adelante.

Como los mejores resultados los da el degree 3, se muestra un nuevo gráfico con ese grado y realizar observaciones:



En general scale tiende a generar una curva hasta el valor C=0.5 y después descender para volver a alcanzar un accuracy alto en C=10. Como un C muy grande puede tender al sobreajuste para este dataset es mejor un C 0,5 o 1 y escala 2

SVM RBF En este tercer apartado se agrega al c value el parámetro sigma de varianza-escala buscando el valor gamma/sigma mas adecuado para este modelo. Gamma: a mayor gamma se puede presentar sobreajuste.

```
# 1.3 SVM RBF --------
#Se agrega el parametro sigma
SVMgrid_rbf <- expand.grid(
    C = c(0.01, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10, 30),
    sigma = c(0.01, 0.05, 0.1, 0.2, 0.5, 1, 2, 5, 10, 30)
)

svm_4 <- train(
    data = dengue, factor(varObjBin) ~ h10pix + temp90 + trees + trees90 +
    Ymin + Ymax + temp + humid + humid90,
    method = "svmRadial",
    trControl = control_svm,
    tuneGrid = SVMgrid_rbf,
    verbose = FALSE)
```

Se despliegan los resultados

	sigma	C
95	0.5	30

\begin{center}\includegraphics[width=%50%]{4_SVM_files/figure-latex/chunk-svmRBF2-1} \end{center}

- A mayor sigma (10, 30) no se observa que los resultados mejoren en términos de *accuracy.*
- bestTune recomienda un sigma de 0.5 y un C de 30, pero desde c 1 hasta c 30 los resultados se observan relativamente parejos por lo que se mantiene el dejar el valor c en 1 y un sigma de 0.5

Validacion cruzada para support vector machines

Con los valores obtenidos de svm lineal, polinomial y RBF se realiza la validación cruzada para su posterior ploteo y comparación de medias para este modelo versus los generados con los algoritmos anteriores

```
# 2. Validacion cruzada rep SVM ------------------------------------------
#cv para lineal
medias14 <- cruzadaSVMbin(
    data = dengue,
    vardep = "varObjBin",
    listconti = c("h10pix", "temp90", "trees", "trees90",
    "Ymin", "Ymax", "temp", "humid", "humid90"),
    listclass = c(""),
    grupos = 4,
    sinicio = 1234,
    repe = 5,
    C = 1
)

medias14$modelo <- "svmLineal"

# cv para poli
medias15 <- cruzadaSVMbinPoly(
    data = dengue,
    vardep = "varObjBin",
    listconti = c("h10pix", "temp90", "trees", "trees90",
    "Ymin", "Ymax", "temp", "humid", "humid90"),
    listclass = c(""),
    grupos = 4,
    sinicio = 1234,
    repe = 5,
    C = 0.5,
    degree = 3,
    scale = 2
)

medias15$modelo <- "svmPoly"

# cv RBF
medias16 <- cruzadaSVMbinRBF(
    data = dengue,
    vardep = "varObjBin",
    listconti = c("h10pix", "temp90", "trees", "trees90",
    "Ymin", "Ymax", "temp", "humid", "humid90"),
    listclass = c(""),
    grupos = 4,
    sinicio = 1234,
    repe = 5,
    C = 1,
    sigma = 0.5
)

medias16$modelo <- "svmRBF"
```

Comparacion de medias y boxplot

Utilizando la función genera_graficos() se despliega el gráfico de comparación de medias

Observaciones

- De los tres modelos el que presenta mayor varianza es el modelo polinomial, los resultados en accuracy sobre todo en svmRBF son buenos pero sigue siendo hasta ahora un randomForest o un xgbm (solo observando el gráfico) los que mejores resultados entregan.