**UNIVERSIDAD NACIONAL AGRARIA DE LA MOLINA**



**Curso: Estadística Computacional.**

**Profesor: Jaime Porras Cerrón.**

**Tema: Validación Cruzada**

**Integrantes:**

* Flores Cáceres, Fiorella
* Huarcaya Quinteros, Katja

2017- 0

**Paquete DAAG**

**Función CVbinary: Validación cruzada para la regresión con una respuesta binaria**

Objetivo

Proporciona medidas de entrenamiento (interna) y validación cruzada de precisión predictiva para la regresión con una respuesta binaria. Los datos se dividen aleatoriamente entre un número de "pliegues".

Cada pliegue se retira, a su vez, mientras que los datos restantes se utilizan para volver a ajustar el modelo de regresión y predecir las observaciones omitidas.

CVbinary(obj, rand=NULL, nfolds=10, print.details=TRUE)

cv.binary(obj, rand=NULL, nfolds=10, print.details=TRUE)

Argumentos

|  |  |
| --- | --- |
| obj | Un objeto glm |
| rand | Un vector que asigna cada observación a un pliegue |
| nfolds | El número de pliegues |
| print.details | Variable lógica (TRUE = imprimir salida detallada, el valor predeterminado) |

Atributos

|  |  |
| --- | --- |
| cvhat | Valores predichos de la validación cruzada |
| internal | Valores predichos internos o de entrenamiento |
| training | Valores predichos internos o de entrenamiento |
| acc.cv | Estimación de precisión de la validación cruzada |
| acc.internal | Estimación de la precisión interna o de entrenamiento |
| acc.training | Estimación de la precisión interna o de entrenamiento |

El término "entrenamiento" parece preferible al término "interno" en relación con los valores predichos, y la medida de exactitud, que se basa en las observaciones utilizadas para derivar el modelo.

Aplicación

library(DAAG)

**#Utilizando la data frogs**

frogs.glm <- glm(pres.abs ~ log(distance) + log(NoOfPools),family=binomial,data=frogs)

CVbinary(frogs.glm)

Fold: 1 2 10 4 5 9 6 8 3 7

Internal estimate of accuracy = 0.759

Cross-validation estimate of accuracy = 0.755

**#Utilizando la data mifem**

mifem.glm <- glm(outcome ~ ., family=binomial, data=mifem)

CVbinary(mifem.glm)

Fold: 10 4 6 8 7 1 3 9 2 5

Internal estimate of accuracy = 0.807

Cross-validation estimate of accuracy = 0.805

**Paquete cvTools**

**Función repCV: Validación cruzada para modelos lineales**

Estimar el error de predicción de un modelo lineal mediante la validación cruzada (repetida) de K-pliegues. Las funciones de validación cruzada están disponibles para los ajustes de mínimos cuadrados calculados con *lm*, así como para las siguientes alternativas robustas: Los modelos de tipo MM calculados con *lmrob* y los ajustes de mínimos cuadrados recortados calculados con *ltsReg*.

repCV(object, ...)

|  |
| --- |
| ## S3 method for class 'lm'  repCV(object, cost = rmspe, K = 5, R = 1, foldType = c("random", "consecutive", "interleaved"), folds = NULL, seed = NULL, ...) |
| ## S3 method for class 'lmrob'  repCV(object, cost = rtmspe, K = 5, R = 1, foldType = c("random", "consecutive", "interleaved"),  folds = NULL, seed = NULL, ...) |
| ## S3 method for class 'lts'  repCV(object, cost = rtmspe, K = 5, R = 1, foldType = c("random", "consecutive", "interleaved"),  folds = NULL, fit = c("reweighted", "raw", "both"), seed = NULL, ...) |

Argumentos

|  |  |
| --- | --- |
| object | Un objeto devuelto por una función de ajuste del modelo. Se implementan métodos para objetos de clase "lm" computados con lm, objetos de clase "lmrob" computados con lmrob y objeto de clase "lts" computados con ltsReg. |
| cost | Una función de coste que mide la pérdida de predicción. |
| K | Un número entero que da el número de grupos en los que se deben dividir los datos (el valor predeterminado es cinco). Tenga en cuenta que esto debe ser elegido de tal manera que todos los grupos sean de aproximadamente el mismo tamaño. El establecimiento de K igual a n produce una validación cruzada leave-one-out. |
| R | Un número entero que da el número de repeticiones. Esto se ignora para la validación cruzada leave-one-out y otras divisiones no aleatorias de los datos. |
| foldType | Especifica el tipo de pliegues a generar. Los valores posibles son "random" (aleatorio) el predeterminado, "consecutive" (consecutivo) o "interleaved" (entrelazado). |
| folds | Un objeto de la clase "cvFolds" que da los pliegues de los datos para la validación cruzada (como devuelto por cvFolds). Si se suministra, se prefiere sobre K y R. |
| fit | Especifica para qué ajuste estimar el error de predicción. Los valores posibles son "reweighted” (“reponderados”) (el valor predeterminado) para el error de predicción del ajuste reponderado; "raw” (“crudo") para el error de predicción del ajuste “bruto”, o "both" para el error de predicción de ambos ajustes. |
| seed | Semilla inicial opcional para el generador de números aleatorios. |
| ... | Argumentos adicionales que deben pasar al costo de la función de pérdida de predicción. |

Aplicación

library(cvTools)

library(robustbase)

data(coleman)

head(coleman)

salaryP fatherWc sstatus teacherSc motherLev Y

1 3.83 28.87 7.20 26.6 6.19 37.01

2 2.89 20.10 -11.71 24.4 5.17 26.51

3 2.86 69.05 12.32 25.7 7.04 36.51

4 2.92 65.40 14.28 25.7 7.10 40.70

5 3.06 29.59 6.31 25.4 6.15 37.10

6 2.07 44.82 6.16 21.6 6.41 33.90

set.seed(1234)

**# Establecer pliegues para la validación cruzada**

folds <- cvFolds(nrow(coleman), K = 5, R = 10)

**# Realizar validación cruzada para un modelo de regresión LS**

fitLm <- lm(Y ~ ., data = coleman)

repCV(fitLm, cost = rtmspe, folds = folds, trim = 0.1)

5-fold CV results:

CV

1.650149

**# Realizar validación cruzada para un modelo de regresión LTS**

fitLts <- ltsReg(Y ~ ., data = coleman)

repCV(fitLts, cost = rtmspe, folds = folds, trim = 0.1)

5-fold CV results:

CV

1.140772

repCV(fitLts, cost = rtmspe, folds = folds, fit = "both", trim = 0.1)

5-fold CV results:

reweighted raw

1.140772 1.511817

**Paquete cvTools**

**Función cvSelect: Selección de modelos basada en la validación cruzada**

Combina los resultados de la validación cruzada para varios modelos en un objeto y selecciona el modelo con el mejor rendimiento de predicción.

cvSelect(..., .reshape = FALSE, .selectBest = c("min", "hastie"), .seFactor = 1)

Argumentos

|  |  |
| --- | --- |
| ... | Objetos que heredan de la clase "cv" o "cvSelect" que contienen resultados de validación cruzada. |
| .reshape | Indica si los objetos con más de una columna de resultados de validación cruzada deberían ser remodelados para tener sólo una columna. |
| .selectBest | Especifica un criterio para seleccionar el mejor modelo. Los valores posibles son "min" (el valor predeterminado) o "hastie". El primero selecciona el modelo con el menor error de predicción. El último es útil para modelos anidados o para modelos con un parámetro de ajuste que controla la complejidad del modelo (por ejemplo, una regresión penalizada). |
| .seFactor | Un valor numérico que da un factor de multiplicación del error estándar para la selección del mejor modelo. Esto se ignora si .selectBest es "min". |

Aplicación

library(cvTools)

library(robustbase)

data(coleman)

set.seed(1234)

**# Establecer pliegues para la validación cruzada**

folds <- cvFolds(nrow(coleman), K = 5, R = 10)

**## Comparar la regresión LS y LTS**

**# Realizar validación cruzada para un modelo de regresión LS**

fitLm <- lm(Y ~ ., data = coleman)

cvFitLm <- cvLm(fitLm, cost = rtmspe, folds = folds, trim = 0.1)

**# Realizar validación cruzada para un modelo de regresión LTS**

fitLts <- ltsReg(Y ~ ., data = coleman)

cvFitLts <- cvLts(fitLts, cost = rtmspe, folds = folds, trim = 0.1)

**# Comparar resultados de validación cruzada**

cvSelect(LS = cvFitLm, LTS = cvFitLts)

5-fold CV results:

Fit CV

1 LS 1.650149

2 LTS 1.140772

Best model:

CV

"LTS"

**## Comparar estimadores de LTS raw y reweighted para**

**## subsets 50% y 75%**

**# subsets 50%**

fitLts50 <- ltsReg(Y ~ ., data = coleman, alpha = 0.5)

cvFitLts50 <- cvLts(fitLts50, cost = rtmspe, folds = folds,

fit = "both", trim = 0.1)

**# subsets 75%**

fitLts75 <- ltsReg(Y ~ ., data = coleman, alpha = 0.75)

cvFitLts75 <- cvLts(fitLts75, cost = rtmspe, folds = folds,

fit = "both", trim = 0.1)

**# Combinar y mostrar resultados**

cvSelect("0.5" = cvFitLts50, "0.75" = cvFitLts75)

5-fold CV results:

Fit reweighted raw

1 0.5 1.140772 1.511817

2 0.75 0.963192 1.165930

Best model:

reweighted raw

"0.75" "0.75"

**Paquete mlr**

**Función makeResampleDesc**

En **mlr** la estrategia de remuestreo puede ser elegido a través de la función **makeResampleDesc**. Las estrategias de remuestreo soportados son:

* La validación cruzada ( "CV")
* Deja fuera de una validación cruzada ( "LOO"")
* Repetida de validación cruzada ( "RepCV")
* Fuera de la bolsa de arranque y otras variantes ( "Bootstrap")
* Submuestreo, también llamado Monte-Carlo de validación cruzada ("Subsample")
* Retención (entrenamiento / prueba) ( "Holdout").

makeResampleDesc(method, predict = "test", ..., stratify = FALSE,

stratify.cols = NULL)

Argumentos

|  |  |
| --- | --- |
| method | “CV”,”LOO”, "RepCV",…"Holdout" |
| predict | Numero de iteraciones, proporcion de casos de entrenamiento, |
| stratify | Para las tareas de clasificación, esto significa que la estrategia de remuestreo se aplica a todas las clases individualmente y los conjuntos de índices resultantes se unen para asegurarse de que la proporción de observaciones en cada conjunto de entrenamiento es como en el conjunto de datos original. |
| stratify.cols | Estratificar en columnas específicas referenciadas por nombre. |

**Función setAggregation**

Establecer el modo en que se agregará esta medida después del remuestreo.

|  |  |
| --- | --- |
| measure | Medida de rendimiento. |
| aggr | Función de agregación. |

Aplicación

|  |  |  |
| --- | --- | --- |
| > library(mlr)  > library("ESKNN") #Data hepatitis  > data("hepatitis")  > hepa<-data.frame(hepatitis)  > dim(hepa)  [1] 155 20  > hepa<-na.omit(hepa)  > dim(hepa)  [1] 80 20  > head(hepa)  V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18  6 1 34 1 2 2 2 2 2 2 2 2 2 2 2 0.9 95 28 4.0  11 1 39 1 1 1 2 2 2 1 1 2 2 2 2 1.3 78 30 4.4  12 1 32 1 2 1 1 2 2 2 1 2 1 2 2 1.0 59 249 3.7  13 1 41 1 2 1 1 2 2 2 1 2 2 2 2 0.9 81 60 3.9  14 1 30 1 2 2 1 2 2 2 1 2 2 2 2 2.2 57 144 4.9  16 1 38 1 1 2 1 1 1 2 2 2 2 1 2 2.0 72 89 2.9  V19 V20  6 75 2  11 85 2  12 54 2  13 52 2  14 78 2  16 46 2  >x<-hepa[,-20]  >x<-data.frame(x)  >V20<-as.factor(hepa[,20])  >task = makeClassifTask(data = hepa, target = "V20")  >rdesc1 = makeResampleDesc("CV", iters = 10)  >rcv = resample(makeLearner("classif.lda"), task, rdesc1)  [Resample] cross-validation iter: 1  [Resample] cross-validation iter: 2  [Resample] cross-validation iter: 3  [Resample] cross-validation iter: 4  [Resample] cross-validation iter: 5  [Resample] cross-validation iter: 6  [Resample] cross-validation iter: 7  [Resample] cross-validation iter: 8  [Resample] cross-validation iter: 9  [Resample] cross-validation iter: 10  **[Resample] Result: mmce.test.mean=0.15** | | |
| > rdesc2 = makeResampleDesc("LOO")  > rcv = resample(makeLearner("classif.lda"), task, rdesc2)  [Resample] LOO iter: 1  [Resample] LOO iter: 2  [Resample] LOO iter: 3  [Resample] LOO iter: 4  [Resample] LOO iter: 5  [Resample] LOO iter: 6  …  …  [Resample] LOO iter: 76  [Resample] LOO iter: 77  [Resample] LOO iter: 78  [Resample] LOO iter: 79  [Resample] LOO iter: 80  **[Resample] Result: mmce.test.mean=0.175** |
| > set.seed(10)  > rdesc = makeResampleDesc("Bootstrap", predict = "both", iters = 10)  > b632.mmce = setAggregation(mmce, b632)  > rdiab = resample("classif.lda", task, rdesc,measures = b632.mmce, show.info = FALSE)  > rdiab$aggr  mmce.b632  **0.1232565** |

**Paquete boot**

**Función cv.glm:** **Validación cruzada para modelos lineales generalizados**

Esta función calcula el error estimado de predicción de validación cruzada de K-fold para modelos lineales generalizados.

cv.glm(data, glmfit, cost, K)

Argumentos

|  |  |
| --- | --- |
| data | Una matriz o data.frame que contiene los datos. Las filas deben ser casos y las columnas corresponden a variables, una de las cuales es la respuesta. |
| glmfit | Objeto de la clase "glm" que contiene los resultados de un modelo lineal generalizado ajustado a los datos. |
| cost | Una función de dos argumentos vectoriales que especifican la función de coste para la validación cruzada. El primer argumento al coste debe corresponder a las respuestas observadas y el segundo argumento debe corresponder a las respuestas predichas o ajustadas del modelo lineal generalizado. El costo debe devolver un valor escalar no negativo. El valor predeterminado es la función de error cuadrático medio. |
| K | El número de grupos en los que los datos deben dividirse para estimar el error de predicción de la validación cruzada. El valor de K debe ser tal que todos los grupos tengan aproximadamente el mismo tamaño. Si el valor suministrado de K no satisface este criterio, entonces se establecerá en el entero más cercano que lo hace y se genera una advertencia especificando el valor de K utilizado. El valor predeterminado es establecer K igual al número de observaciones en los datos el cual da la validación cruzada de leave-one-out habitual. |

Aplicación

|  |
| --- |
| > library(ISLR)  > library(caTools)  > attach(Auto)  > head(Auto)  mpg cylinders displacement horsepower weight acceleration year  1 18 8 307 130 3504 12.0 70  2 15 8 350 165 3693 11.5 70  3 18 8 318 150 3436 11.0 70  4 16 8 304 150 3433 12.0 70  5 17 8 302 140 3449 10.5 70  6 15 8 429 198 4341 10.0 70  origin name  1 1 chevrolet chevelle malibu  2 1 buick skylark 320  3 1 plymouth satellite  4 1 amc rebel sst  5 1 ford torino  6 1 ford galaxie 500  > dim(Auto)  [1] 392 9  > Auto<-na.omit(Auto)  > dim(Auto)  [1] 392 9  > Auto<-Auto[,-c(7,8,9)]  > set.seed(1)  > muestra <- sample.split(Auto, SplitRatio = 0.75)  > Auto.train <- subset(Auto, muestra == TRUE)  > Auto.test <- subset(Auto, muestra == FALSE)  > lm.fit <- lm(mpg ~ horsepower+weight, data = Auto.train)  > mean((Auto.test$mpg ‐ predict(lm.fit, Auto.test))^2)  [1] **16.81581**  > glm.fit <- glm(mpg ~ horsepower, data = Auto)  > library(boot) #Aplicar una validación Cruzada  > cv.err <- cv.glm(Auto, glm.fit)  > attributes(cv.err)  $names  [1] "call" "K" "delta" "seed"  > cv.err$delta[1]  [1] **18.11295** |