Aprendizaje de Máquina Práctica 05

Pablo Díaz - 30343 | Kevin Huerta - 30502 | Diego Zuazo - 30046 | Gerardo Hernandez - 29902

***Abstract*— Comprender y utilizar las funciones básicas en el lenguaje de programación R y Python para realizar técnicas de remuestreo con énfasis en métodos de Cross-Validation y Bootstrap.**

I. INTRODUCCIÓN

Este reporte tiene como objetivo explicar las metodologías aplicadas en los ejercicios resueltos de la práctica número 5, a través del uso de dos lenguajes de programación: Python y R. El enfoque principal que tiene esta práctica es hacer uso de las diferentes técnicas de remuestreo que existen, haciendo énfasis en Cross-Validation y Bootstrap. La estructura de la practica es la siguiente: primero se explicaran los fundamentos necesarios para la practica, después la metodología a seguir y finalmente los resultados. Al final se presenta nuestra conclusión sobre los temas y la practica.

II. FUNDAMENTOS

Los fundamentos teóricos de la implementación elaborada en esta practica abordan principalmente los métodos de remuestreo. El propósito por el cual se construye la teoría de remuestreo es cuando se quiere obtener información adicional acerca del modelo entrenado que no esta disponible cuando entrenamos solamente una vez con un solo conjunto de entrenamiento. Por lo cual se opta a generar distintas conjuntos derivados de conjunto de entrenamiento. El cual para cada nueva muestra generada habrá un modelo que se entrene con dicha muestra. Esto con el fin de examinar que tanto difiere un modelo con otro. Actualmente dos de las mas comunes métodos de remuestreo son, 'cross-validation' y 'bootstrap'. Por la naturaleza de R, estos son mas sencillos de aplicar en dicho lenguaje. En el caso de Python, descubrimos una librería (rpy2) que permite realizar funciones nativas de R en Python, lo cual nos fue muy útil en el caso de la librería boot.

A. Cross-Validation

Cross-Validation es un procedimiento de remuestreo que se utiliza para evaluar modelos de aprendizaje con una muestra de datos limitada. Los métodos de Cross-Validation consisten en dejar afuera un subconjunto del conjunto de datos de tal forma que se obtenga un subconjunto de entrenamiento y otro para evaluar.

Hay varios métodos de este tipo. Uno de los mas conocidos, el K-Fold Cross-Validation tiene un único parámetro llamado k, que se refiere al número de grupos en los que se dividirá una muestra de datos determinada. Existe también el Leave-One-Out Cross-Validation, en el cual el conjunto de datos se separa en una sola observación para evaluar y todas las demás para entrenar. También existe el que hace las separaciones de tamaños similares.

B. Bootstrap

Bootstrap es cualquier prueba o métrica que utiliza muestreo aleatorio con reemplazo y se incluye en la clase más amplia de métodos de remuestreo. Bootstrap asigna medidas de precisión a estimaciones de muestra. Esta técnica permite la estimación de la distribución muestral de casi cualquier estadística utilizando métodos de muestreo aleatorio.

III. METODOLOGÍA

La metodología del equipo fue la misma que en veces anteriores, nos dividimos en dos grupos para la implementación del código, un grupo de R y un grupo de Python. Al igual que en practicas pasadas, los equipos se alternaron de lenguaje para practicar. Considerando que es un trabajo en equipo, al igual que en las practicas pasadas, tuvimos que trabajar en equipo para encontrar el enfoque adecuada para algunas actividades. De forma mas puntual, la metodología fue la siguiente:

A. Ejercicio número 1

En el primer ejercicio se usa el "Validation Set Approach" para estimar errores de prueba que resultan que resultan de ajustar varios modelos lineales en el conjunto de datos Auto. Se hace uso de la función sample () para dividir el conjunto de observaciones sample () en dos mitades, seleccionando un subconjunto aleatorio de 196 observaciones de las 392 observaciones originales. Nos referimos a estas observaciones como el "training set".

B. Ejercicio número 2

En el ejercicio 2 se aplico el método de "Leave-One-Cross-Validation". La estimación para LOOCV se puede calcular para cualquier modelo lineal usando las funciones glm() y cv.glm() usamos la función glm () para realizar la regresión logística pasando en el argumento familia = "binomio". porque si usamos glm () para ajustar un modelo sin pasar el argumento de la familia, entonces se realiza una regresión lineal, al igual que la función lm ().

C. Ejercicio número 3

Para el ejercicio 3 se aplico el método de k-fold cross-validation. La función de cv.glm() también se puede usar para implementar este método. Se uso k=10 (un valor de k muy común) en conjunto de datos Auto. Se establece una semilla aleatoria e inicializamos un vector en donde se almacenara los errores de CV que corresponden a los ajustes polinomiales de los ordenes una a diez.

D. Ejercicio número 4

Finalmente para el ejercicio 4 se implementa el método de Bootstrap de una manera simple en donde se estima la precisión de la línea modelo de regresión en el conjunto de datos Auto.

IV. RESULTADOS

Los resultados se encuentras plasmados de la misma forma que en practicas pasadas. Lo que se hace es que. en el caso de R, los resultados se muestran tal y como lo especifica la practica del libro. Ahora bien, en el caso de Python, estos se encuentran en un formato distinto a los presentados en R, pero abordan el mismo problema de la manera mas cercana que permite Python, teniendo en cuenta sus limitaciones al realizar problemas que R realiza con facilidad.

A. Ejercicios en R:

1) Actividad 1- Los resultados obtenidos para la Actividad 1 fueron los siguientes:

23.2660086465003

18.7164594933828
18.7940067973945

25.7265106448139
20.4303642741463
20.3853268638776

2) Actividad 2- Los resultados obtenidos para la Actividad 2 fueron los siguientes:

(Intercept): 39.9358610211705 horsepower: -0.157844733353654

(Intercept): 39.9358610211705 horsepower: -0.157844733353654

24.2315135179292 · 24.2311440937561

24.2315135179292 · 19.2482131244897 · 19.3349840640291 · 19.4244303104302 · 19.0332138547041

3) Actividad 3- Los resultados obtenidos para la Actividad 3 fueron los siguientes:

24.2720671232254 · 19.2690929085129 · 19.3480535605547 · 19.2949648229745 · 19.0319780002896 · 18.8978121059401 · 19.1290666690695 · 19.1496831054789 · 18.8701307442148 · 20.9592342280394

4) Actividad 4- Los resultados obtenidos para la Actividad 4 fueron los siguientes:

0.57583207459283

0.736837501928544

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Portfolio, statistic = alpha.fn, R = 1000)


Bootstrap Statistics :
     original       bias    std. error
t1* 0.5758321 -0.001695873  0.09366347
```

(Intercept): 39.9358610211705 horsepower: -0.157844733353654

(Intercept): 40.3404516830189 horsepower: -0.163486837689938
(Intercept): 40.1186906449022 horsepower: -0.157706320543503

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Auto, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
      original        bias    std. error
t1* 39.9358610  0.0544513229 0.841289790
t2* -0.1578447 -0.0006170901 0.007343073
```

A matrix: 2 × 4 of type dbl

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	39.9358610	0.717498656	55.65984	1.220362e-187
horsepower	-0.1578447	0.006445501	-24.48914	7.031989e-81

```
Call:
boot(data = Auto, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
        original        bias     std. error
t1* 56.900099702  3.511640e-02 2.0300222526
t2* -0.466189630 -7.080834e-04 0.0324241984
t3*  0.001230536  2.840324e-06 0.0001172164
```

A matrix: 3 × 4 of type dbl

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	56.900099702	1.8004268063	31.60367	1.740911e-109
horsepower	-0.466189630	0.0311246171	-14.97816	2.289429e-40
I(horsepower^2)	0.001230536	0.0001220759	10.08009	2.196340e-21

B. Ejercicios en Python:

1) Actividad 1- Los resultados obtenidos para la Actividad 1 fueron los siguientes:

```
[23.172322563688436, 19.450442202363618, 19.444359249527423]
```

```
[24.663732413967836, 19.515744461302223, 19.5135796153207]
```

2) Actividad 2- Los resultados obtenidos para la Actividad 2 fueron los siguientes:

```
INTERCEPT: 39.93586102117047 HORSEPOWER: [-0.15784473]
```

```
[1] 24.23151 24.23114
```

```
[1] 24.23151 19.24821 19.33498 19.42443 19.03321
```

3) Actividad 3- Los resultados obtenidos para la Actividad 3 fueron los siguientes:

```
[0.52614612 0.55012096 0.39495847 0.72501551 0.50305843 0.63314617
 0.58899073 0.68289297 0.53810027 0.65321772]
```

4) Actividad 4- Los resultados obtenidos para la Actividad 4 fueron los siguientes:

```
[1] 0.5758321
```

```
[1] 0.7368375
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Portfolio, statistic = alpha.fn, R = 1000)


Bootstrap Statistics :
     original       bias    std. error
t1* 0.5758321 -0.001695873  0.09366347
```

```
(Intercept)  horsepower
 39.9358610  -0.1578447
```

```
(Intercept)  horsepower
 40.3404517  -0.1634868
(Intercept)  horsepower
 40.1186906  -0.1577063
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Auto, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
      original        bias    std. error
t1* 39.9358610  0.0412152338 0.832986409
t2* -0.1578447 -0.0005105609 0.007228529
```

```
             Estimate  Std. Error   t value      Pr(>|t|)
(Intercept) 39.9358610 0.717498656  55.65984 1.220362e-187
horsepower  -0.1578447 0.006445501 -24.48914  7.031989e-81
```

```
ORDINARY NONPARAMETRIC BOOTSTRAP


Call:
boot(data = Auto, statistic = boot.fn, R = 1000)


Bootstrap Statistics :
       original        bias     std. error
t1* 56.900099702  3.511640e-02 2.0300222526
t2* -0.466189630 -7.080834e-04 0.0324241984
t3*  0.001230536  2.840324e-06 0.0001172164
                    Estimate   Std. Error   t value      Pr(>|t|)
(Intercept)     56.900099702 1.8004268063  31.60367 1.740911e-109
horsepower      -0.466189630 0.0311246171 -14.97816  2.289429e-40
I(horsepower^2)  0.001230536 0.0001220759  10.08009  2.196340e-21
```

V. CONCLUSIONES

En conclusión, consideramos que los temas vistos en esta practica son de gran utilidad. Nos parece interesante como métodos tan sencillos de aplicar en R, como lo son aquellos de Cross-Validation y Bootstrap, pueden llegar a ser tan útiles. El método que nos parece mas valioso es el de K-Fold CV, ya que consideramos que es sencillo de aplicar y arroja buenos resultados comparado con otros de CV. El realizar las actividades nos ayudó a entender mejor estos dos tenas, esto debido a que los ejercicios consisten en la aplicación de los mismos. El entender a profundidad estos temas nos ayudan a reforzar lo que sabemos de la materia, así como amplían nuestras capacidades al momento de evaluar modelos. Además, al realizar esta practica no solo estamos reforzando dichos temas, sino que también estamos reforzando nuestras habilidades de programar y desarrollar problemas de aprendizaje maquina en R y Python.