Trabajo Práctico 6

Camussoni Francesco[1,*]

[1]Instituto Balseiro, Universidad Nacional de cuyo
[*]camussonif@gmail.com

ABSTRACT

Se implementaron métodos basados en árboles de decisión para resolver el problema de regresión y clasificación del dataset Carseats. Se utilizaron técnicas de Bagging, RandomForest y AdaBoost, además de un árbol convencional. En cada caso se obtuvo el error en los datos de test y los atributos más importantes. En los casos de clasificación se utilizó el valor del area bajo la curva ROC como métrica a minimizar y en los casos de regresión MSE

Ejercicio a

Se creó una variable 'High' de valor 1 cuando la variable 'Sales' es mayor o igual a 8 y 0 en caso contrario. Luego se separó el set en training set y test set con un test_size del 20% del set original. Luego se utilizó la funcion de panda get_dummies para realizar un one hot encoder sobre los atributos categóricos.

Ejercicio b

Se utilizo la clase DecisionTreeClassifier de sklearn para entrenar un árbol de clasificación de la variable 'High'. Para esto se excluyó la variable 'Sales' a fin de no tener una precisión falsa. Luego se realizó una matriz de correlaciones de las variables numéricas a fin de visualizar alguna relación directa entre las atributos y la clasificación final en donde se obtuvo la **Figura 1**.

Figure 1. Matriz de correlaciones.

En donde no parece haber demasiada correlacion entre las distintas atributos, sin embargo, se observa que en los atributos 'Price', 'Age' y 'Advertesing' hay una tendencia a obtener un precio alto o bajo según el valor de estos atributos, por lo que se espera se encuentren entrelos atributos más importantes dados por el árbol de clasificación.

En primer lugar se entrenó un arbol de clasificación sin hacer prunning, en donde se obtuvieron los resultados que se muestran en la **Figura 2**.

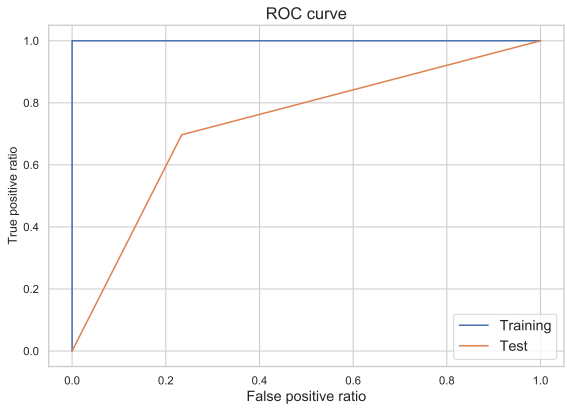

Figure 2. Curva ROC para los datos de training y de test sin realizar prunning ni controlar la profundidad del árbol.

En donde se observa un claro overfitting ya que el árbol realiza una clasificación en donde en cada hoja se realiza un split hasta tener 100% de pureza, obteniendo así un árbol muy complejo que debido al tamaño del mismo se presenta un enlace en el anexo de este documento.

Luego, se realizó la misma clasificación pero realizando una búsqueda con la funcion RandomizedSearchCV de sklearn en donde se consideraron los parámetros ccp_alpha, de prunning, y max_depth. Así se obtuvieron los resultados que se muestran en la **Figura 3** y el arbol que se muestra en la **Figura 4**.

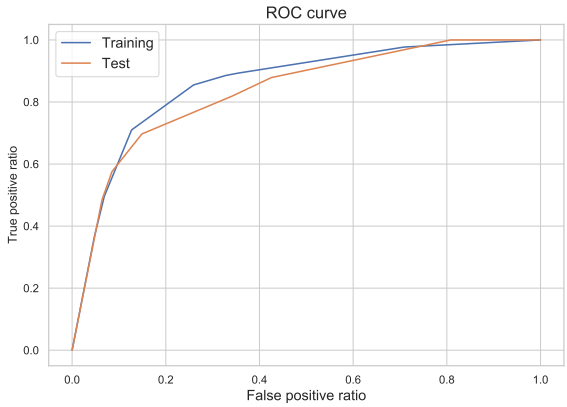

Figure 3. Curva ROC para los datos de training y de test con hiperpárametros adecuados.

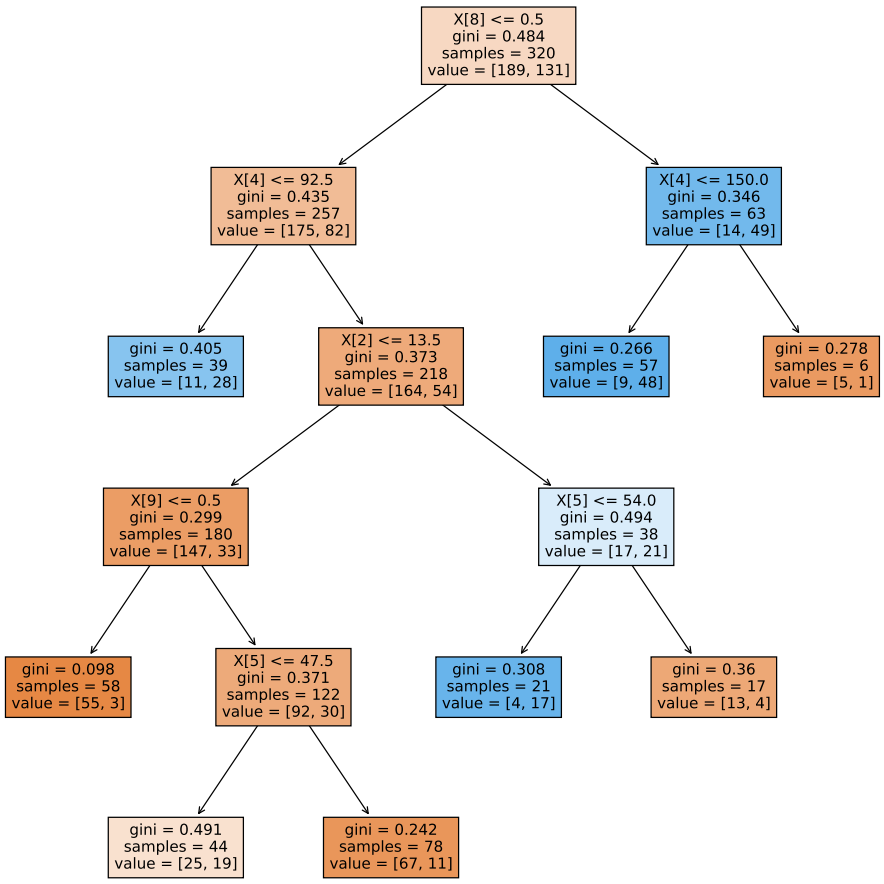

Figure 4. Arbol obtenido con hiperpárametros adecuados.

En donde se redujo el overfitting y se obtuvo un árbol mas sencillo.
En resumen los resultados fueron los siguientes:

- Sin busqueda de hiperparámetros
  - AUC training: 1
  - AUC test: 0.73
  - Atributos más importantes
    - 1: Price
    - 2: CompPrice
    - 3: SheveLoc_Good
    - 4: Age
    - 5: ADvertising
- Con hiperparámetros adecuados
  - ccp_alpha: 0.010
  - max_depth: 5
  - AUC training: 0.86

- AUC test: 0.84
- Atributos más importantes
    - 1: ShelveLoc_Good
    - 2: Price
    - 3: Age
    - 4: ADvertising
    - 5: ShelveLoc_Medium

En donde se encuentran entre los atributos más importantes los esperados según la **Figura 1**.

Ejercicio c y e

En primer lugar se obtuvo nuevamente una matriz de correlaciones entre variables que se muestra en la **Figura 5**.

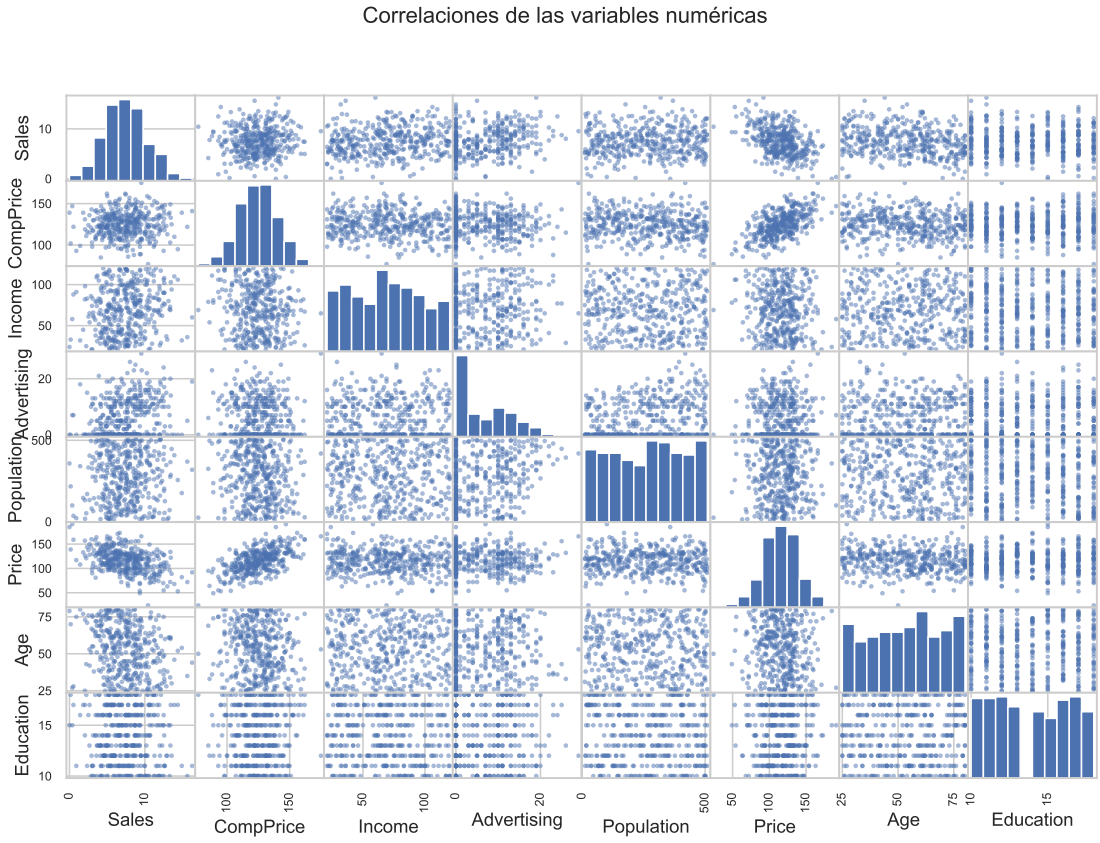

Figure 5. Matriz de correlaciones.

En donde las variables mas correlacionadas con 'Sales' fueron 'Price' con una correlación de 0.44, 'Advertising' con una correlación de 0.27 y 'Age' con una correlación de 0.23, siempre en módulo. Siendo estas las variables continuas que se esperan que aparezcan entre los atributos mas importantes de los arboles de regresión.

Se realizó el mismo procedimiento para obtener una regresión sobre la variable 'Sales'. Nuevamente, sin ajustar hiperparámetros se obtuvo un notorio overfitting, por las mismas razones descriptas anteriormente, mientras que al realizar una busqueda de hiperparámetros de prunning y max_depth se obvuto un mejor rendimiento. Así se obtuvieron los resultados que se muestran en las **Figuras 6** y **7**

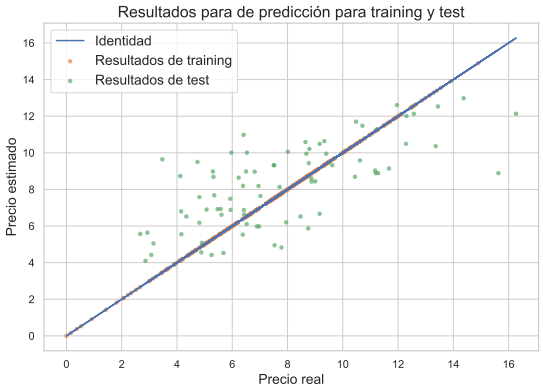

Figure 6. Resultados para los datos de training y test sin busqueda de hiperparámetros.

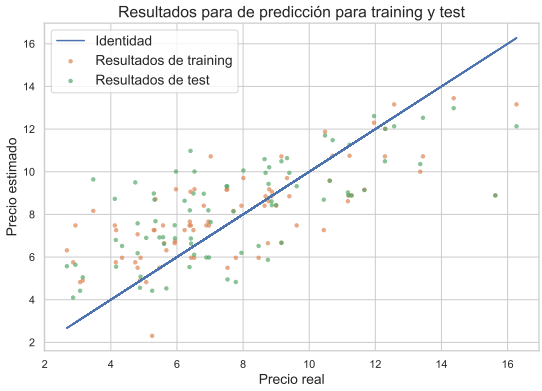

Figure 7. Resultados para los datos de training y test con hiperparámetros adecuados.

Los arboles obtenidos se encuentran en el anexo. Luego, en resumen, los resultaods fueron los siguientes:

- Sin busqueda de hiperparámetros
  - Maximo error training: 0
  - RMSE training: 0
  - Maximo error test: 6.74
  - RMSE test: 2.27
  - Atributos más importantes
    - 1: Price
    - 2: ShelveLoc_Good
    - 3: Age
    - 4: CompPrice
    - 5: ShelveBad_Good
- Con hiperparámetros adecuados
  - ccp_alpha: 0.0023
  - max_depth: 6
  - Maximo error training: 4.06
  - RMSE training: 1.14
  - Maximo error test: 6.67
  - RMSE test: 2.07
  - Atributos más importantes
    - 1: Price
    - 2: SheveLoc_Good
    - 3: Age
    - 4: CompPrice
    - 5: ADvertising

En donde nuevamente se verifica los atributos más importantes, sobre todo para el árbol correctamente entrenado.

Ejercicio f

Se implementó la metodología de Bagging con bootstrap en donde se utilizó el modelo final obtenido en el enunciado anterior como estimador de base. Se hizo una búsqueda azarosa de diversos hiperparámetros como el numero de estimadores, y la cantidad de ejemplos a utilizar. Así se obtuvieron los resultados que se muestran en la **Figura 8**

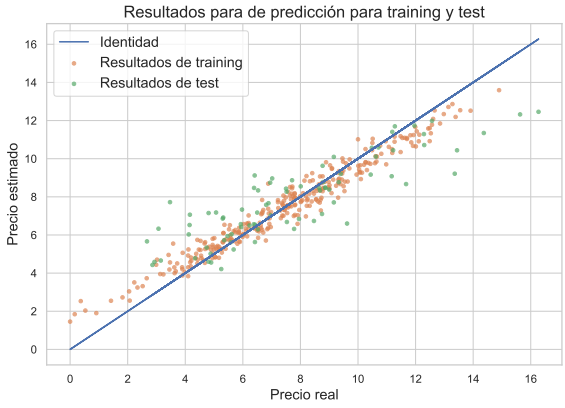

Figure 8. Resultados para los datos de training y de test con la metodología de Bagging.

En donde se observa que disminuyó el overfitting respecto al caso anterior.
Luego, los resultados mas importantes fueron los siguientes:

- max_samples: 0.84
- n_estimators: 70
- Maximo error training: 2.16
- RMSE training: 0.66
- Maximo error test: 4.25
- RMSE test: 1.63
- Atributos más importantes
  - 1: Price
  - 2: SheveLoc_Good
  - 3: CompPrice
  - 4: Age
  - 5: ADvertising

En donde para obtener los atributos mas importantes se realizó un promedio según cada estimador del método.

Ejerficio g

Se utilizó la metodología de Random Forest Regressor en donde se utilizó cross validation con una busqueda de varios hiperparámetros como ser cpp_alpha, n_estimators, max_features y max_depth. Así se obtuvieron los resultados que se muestran en la **Figura 9**.

Figure 9. Resultados para los datos de training y de test con la metodología de Random Forest.

En donde se observa un rendimiento similar a Bagging, sin embargo, el error de los datos de test disminuyo como se muestra a continuacion:

- max_features: 0.89
- n_estimators: 50
- cpp_alpha: 0.0018
- max_depth: 11
- Maximo error training: 1.88
- RMSE training: 0.61
- Maximo error test: 4.25
- RMSE test: 1.61
- Atributos más importantes
  - 1: Price
  - 2: SheveLoc_Good
  - 3: CompPrice
  - 4: Age
  - 5: ADvertising

En donde no se modificaron los atributos mas importantes, quizá debido a que el rendimiento es muy parecido en donde la reducción en el error es debido a que ambos métodos consideran como importante los atributos que realmente lo son.

Luego, se realizó predicciones para el mismo modelo anterior pero variando en un caso el max_features y en otro caso el max_depth. Así se obtuvieron los resultados que se muestran en las **Figuras 10** y **11**.

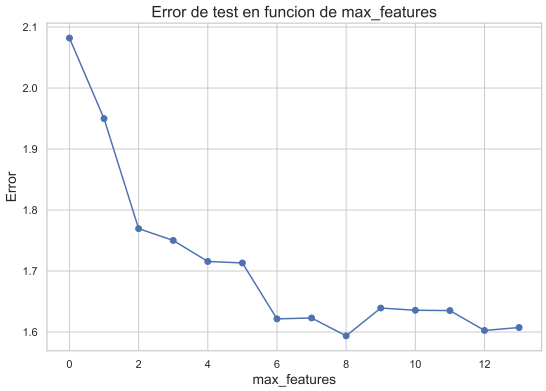

Figure 10. Error en los datos de test al variar la cantidad de atributos máxima.

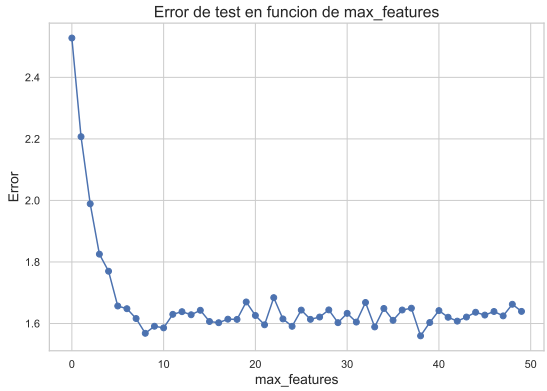

Figure 11. Error en los datos de test al variar la profundidad máxima.

En donde se observa que si se fija un hiperpárametro, el restante minimizaba el error de test con otro valor al obtenido en la búsqueda azarosa. Esto podría mejorarse si se utilizan mas iteraciones en la búsqueda de hiperparámetros o si se itera varias veces al observar cada rango óptimo de los mismos. Finalmente, se observa que para disminuir el error de test conviene tener un modelo complejo, aunque como solo se presenta el error de test, es probable que esta condición produzca overfitting.

Ejercicio h

A partir del modelo obtenido en el ejercicio anterior se obtuvo otro mediante la metodología de Ada Bost en donde se realizó una búsqueda del hiperparámetro de learning rate. Así se obtuvo el mejor rendimiento de todos con resultados que se muestran en la **Figura 12**.

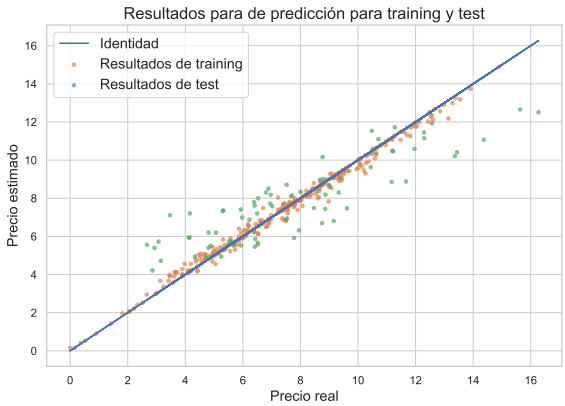

Figure 12. Resultados para los datos de training y de test con la metodología de Ada Boost.

Luego, los parámetros mas importantes fueron:

- learning rate: 1.7
- Maximo error training: 1.15
- RMSE training: 0.26
- Maximo error test: 3.76
- RMSE test: 1.54
- Atributos más importantes

- 1: Price
- 2: SheveLoc_Good
- 3: CompPrice
- 4: Age
- 5: ShelveLoc_Bad

Anexo

Los arboles obtenidos pueden encontrarse en el siguiente enlace: https://github.com/francescocamussoni/Deep_Learning_IB