# [NÚMERO NO SE QUE] Predicciones

Tras realizar la limpieza de datos inicial en el apartado anterior, pasamos a realizar los entrenamientos necesarios para poder generar los distintos modelos, y así, predecir con los algoritmos que se mencionarán más adelante.

La idea es poder ser capaces de predecir si una persona padecerá de recidiva bioquímica (RBQ) de cáncer de próstata, a partir de unos datos. La recidiva bioquímica consiste en la repetición de una enfermedad después de haberse recuperado de dicha enfermedad.

Para ello, utilizamos el notebook (adjuntos en el anexo de esta memoria, y con el proyecto como fichero aparte) con la limpieza de datos para realizar el análisis de los datos y las distintas pruebas; para así, posteriormente, copiar y pegar el código necesario del notebook en el código que contiene el *backend* de nuestra aplicación.

Para realizar el análisis de los datos utilizamos la librería Pandas, y para generar los modelos y realizar las predicciones utilizamos scikit-learn, una librería de *Machine Learning* muy sencilla de utilizar. MENCIONAR QUE LOS DATOS CATEGÓRICOS SON NUMÉRICOS… QUE MIREN EL WORD CON EL *MAPEO. QUIZAS SE MENCIONE EN LA PARTE DE “LIMPIEZA”*

## Descripción inicial de los datos

En primer lugar, visualizamos los datos para tener una idea inicial de lo que tenemos en mano. Haciendo df.shape obtuvimos el número de filas y de columnas del dataset tras la primera limpieza previa.

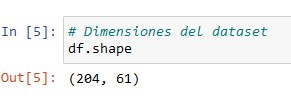


Figura 1. Forma del dataset tras la primera limpieza

Vimos que el dataset tiene 204 filas (es decir, 204 casos de seguimiento de cáncer de próstata) y 61 columnas (o variables). Nada más ver esto descubrimos que tenemos pocos datos para realizar el entrenamiento, pero al ser casos muy extraños, no existe ninguna forma de obtener más datos.

También utilizamos df.describe(include="all",datetime\_is\_numeric=True) para ver un resumen estadístico.

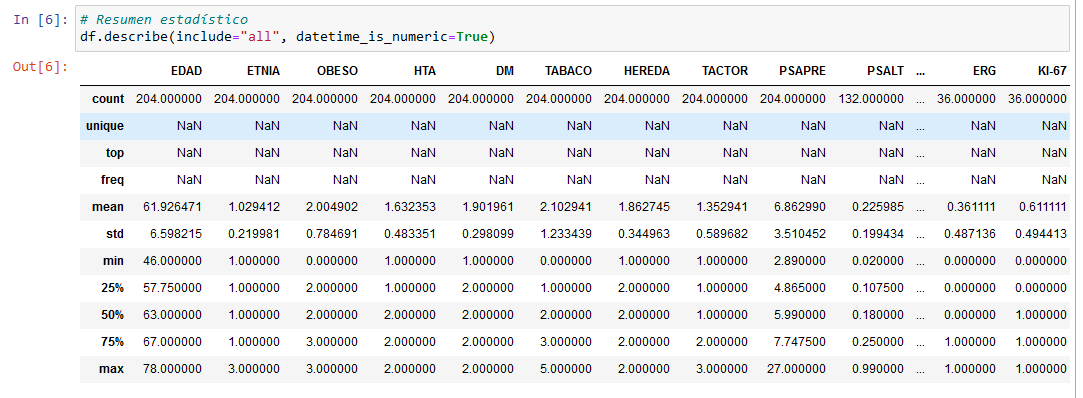


Figura 2. Resumen estadístico

Con este resumen estadístico pudimos ver, por ejemplo, que la edad media de los pacientes es de unos 62 años, que el más joven tiene 46 años, y que el más mayor tiene 78 años. También aparecen las estadísticas de otras columnas como PSAPRE, PSALT, etc. Explicar e interpretar estos tecnicismos médicos se alejan del ámbito de este Trabajo Final de Grado.

## Limpieza de columnas con filas vacías

Una cosa que tuvimos muy en cuenta (ya que es necesario para poder realizar los entrenamientos), es no tener ninguna columna con datos nulos. Para detectar estos nulos utilizamos df.info().

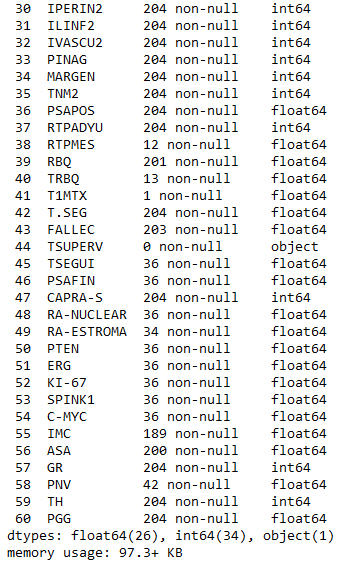
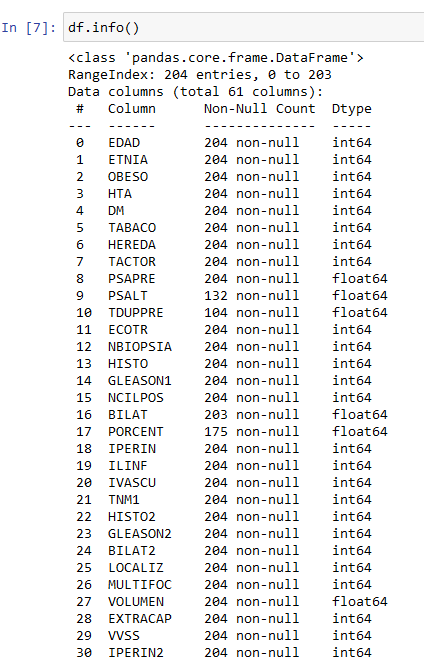


Figura 3. Visualización para localizar columnas vacías

Como se puede observar en las capturas anteriores, hay algunas columnas con entradas nulas. Algunas ni siquiera llegan a las 100 entradas no nulas, por lo que tuvimos que hacer más limpieza para no tener ninguna columna con datos vacíos, y que scikit-learn pueda hacer sus entrenamientos correctamente. Para hacer esta limpieza, decidimos rellenar aquellos nulos con la mediana de los datos de cada columna.

Antes de hacer el rellenado, eliminamos aquellas columnas con más del 50% de filas vacías, ya que consideramos que rellenar con la mediana una columna prácticamente vacía sería falsear demasiado nuestro dataset, haciendo que nuestras predicciones no sean certeras y reflejen una realidad que no es. A continuación, mostramos la función para eliminar las columnas casi vacías.

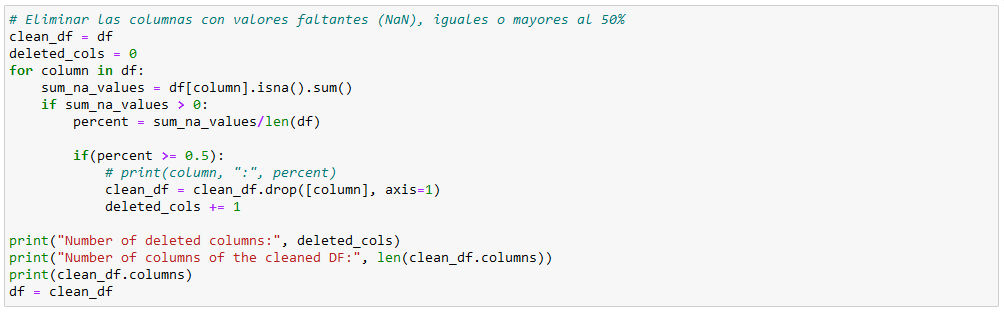


Figura 4. Función para eliminar columnas con valores faltantes mayores o iguales al 50%

Una vez eliminadas las columnas casi vacías, rellenamos las demás columnas con la mediana a través de la siguiente función:

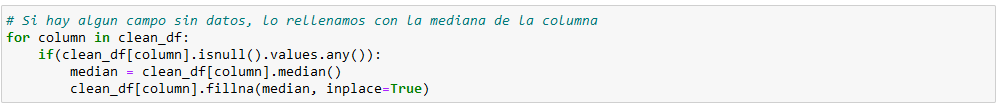


Figura 5. Función para rellenar columnas con filas vacías con la mediana

Una vez rellenadas las columnas confirmamos que, efectivamente, no tenemos ninguna columna con filas vacías.

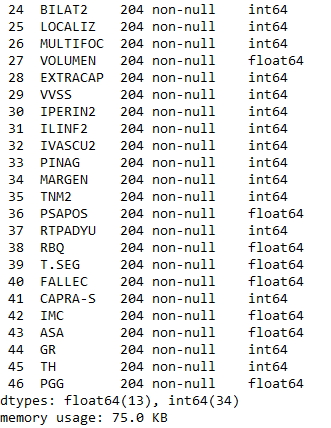
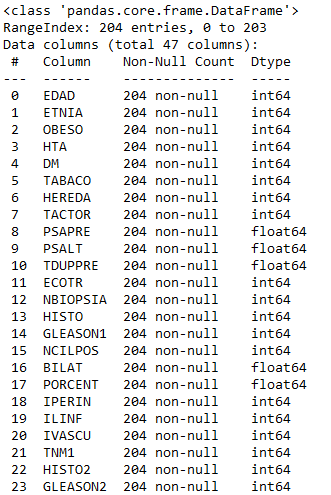


Figura 6. Confirmación de que todas las columnas están totalmente rellenas

Tras esta limpieza, observamos que tenemos todas las columnas rellenas, pero pasamos de tener 61 columnas a tener 47 columnas. 14 columnas tenían más del 50% de filas vacías.

## Visualización general y estudio de los datos

Después de detectar y limpiar aquellas columnas con datos vacíos, visualizamos los datos a través de histogramas para entender un poco mejor nuestro dataset.

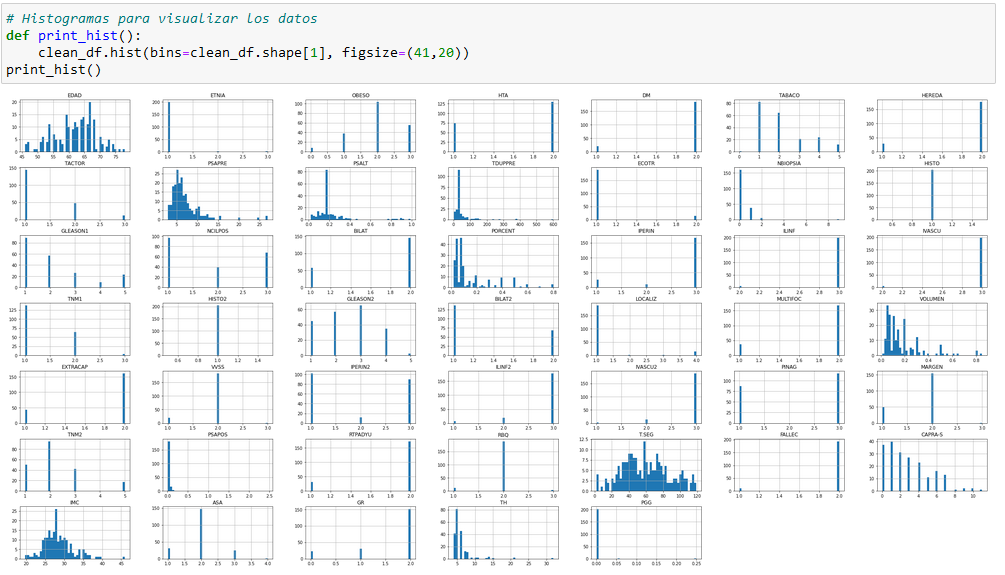


Figura 7. Histogramas del dataset

Tras visualizar estos datos, podemos estudiar cada columna de manera más sencilla. A continuación, se detallará el análisis realizado sobre algunas columnas cuyo estudio es interesante:

* **ETNIA.** Esta columna no es muy relevante a la hora de predecir la recidiva bioquímica. Además, estos estudios se realizaron en España, por lo que la mayoría de la población es caucásica, y no aportaría mucha información. Puede producir *overfitting* si dejamos esta columna, ya que el hecho de ser caucásico influenciaría demasiado sobre la predicción.
* **IPERIN.** Hay tres categorías (“si”, “no”, “nc”). La mayoría de los casos no se conocen (“nc”). Al tener menos de 30 casos conocidos (sobre los más de 200), no tiene mucho sentido utilizar esta variable, ya que solo añadiría ruido a nuestros modelos.
* **LINF, LINF2, IVASCU e IVASCU2.** Hay tres categorías (“si”, “no”, “nc”). La mayoría de los casos no se conocen (“nc”). No tiene mucho sentido utilizar estas variables, ya que, al igual que el caso anterior, solo añadirían ruido a nuestros modelos.
* **FALLEC.** Esta variable se tiene que eliminar, ya que el fallecimiento no influye en nada en la predicción de la recidiva bioquímica. Además, el fallecimiento es una consecuencia de la recidiva, no una causa. Usarlo en nuestros modelos no tiene ningún sentido.
* **IPERIN2.** Hay tres variables (“si”, “no”, “nc”). La mayoría de la población está repartida entre el “sí” y el “nc”, siendo el “sí” el que tiene mayor peso. Para este caso hemos decidido reasignar a todos aquellos “nc” al “si” o al “no” de manera aleatoria.

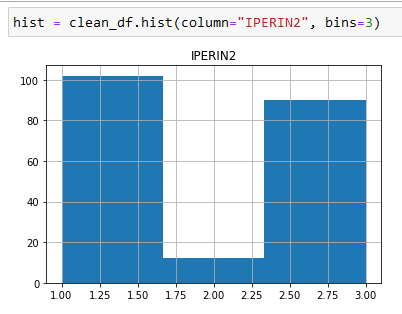
****

Figura 8. Casos de IPERIN en el dataset

Para realizar la reasignación aleatoria de los “nc”, utilizamos la función descrita a continuación:



Figura 9. Función para reasignar los casos de "nc"

Una vez realizadas las reasignaciones, obtenemos un nuevo IPERIN2 sin los casos “nc”, tal y como se muestra en el histograma de la *Figura 10*. Nótese que, al ser aleatorio, el histograma no será el mismo en todas las reasignaciones, pero en la mayoría de casos habrá un mayor balance entre el “si” y el “no”.

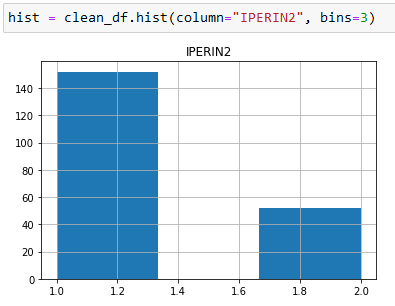


Figura 10. IPERIN2 tras las reasignaciones aleatorias

* **RBQ.** Esta es nuestra variable a predecir. Tiene tres categorías: “SI (CASOS)”, “NO (CONTROLES)” y “PERSISTENCIA PSA”. Después de realizar el análisis de esta columna, preguntamos a nuestro codirector de proyecto sobre la relevancia de “PERSISTENCIA PSA”, ya que son casos muy escasos (menos de 10 casos). La respuesta fue que la “PERSISTENCIA PSA” no tiene mucho impacto sobre la RBQ y que se puede descartar (y así hicimos). Nuestro predictor se convierte entonces en un clasificador binario, el cual se limita a predecir “SI” o “NO”.

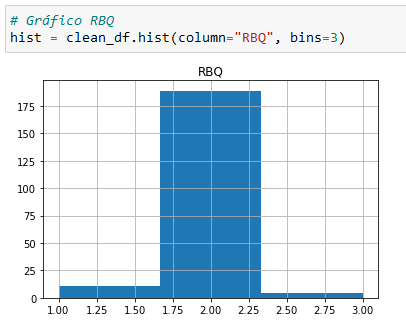


Figura 11. Histograma de RBQ (con PERSISTENCIA PSA)

La función para realizar eliminar los casos de “PERSISTENCIA PSA” mediante reasignaciones a “SI” o “NO” de manera aleatoria de describe en la captura siguiente:

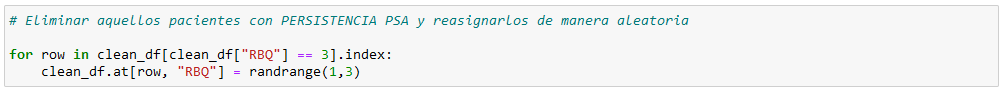


Figura 12. Función para reasignar los casos de PERSISTENCIA PSA

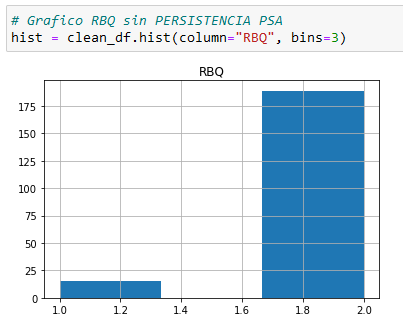


Figura 13. Histograma de RBQ (sin PERSISTENCIA PSA)

Si nos fijamos en el histograma, podemos observar que esta variable está muy desbalanceada, dando mayor peso al “NO”. Este desbalanceo y la escasez de datos limita bastante la capacidad predictiva de nuestros modelos, y no podemos hacer mucho para mejorar dicha capacidad, ya que solo disponemos de los más de 200 (y escasos) casos de cáncer de próstata en seguimiento durante más de 10 años.

Además, a través de estos histogramas sabemos de antemano que nuestros modelos estarán muy sesgados y que en la mayoría de las predicciones el resultado será “NO”. Existe una probabilidad muy alta de que existan muchos falsos negativos de casos con recidiva bioquímica.

En el apartado siguiente se mostrarán los pasos utilizados para poder obtener modelos de predicción con la mayor efectividad y certeza posible.

## Generación de los modelos de predicción

// TODO