UNIVERSIDAD DE GRANADA

Departamento de Ciencias de la
Computación e Inteligencia Artificial

Inteligencia de Negocio

Práctica 1:
Resolución de problemas de clasificación y análisis experimental

Curso 2020-2021

Cuarto curso del grado de Ingeniería Informática

Luis Miguel Aguilar González

Índice

1.- Introducción **3**

2.- Procesado de datos **3**

2.1.- Eliminación de Filas 3

2.2.- Reemplazo simple 3

2.3.- Reemplazo Knn 4

3.- Configuración de Algoritmos y sus resultados **4**

3.1.- Naive Bayes 7

3.2.- Multi-layer Perceptron 9

3.3.- K-nn Algorithm (Vecinos más próximos) 11

3.4.- CART Algorithm 13

3.5.- Random Forest 15

4.- Análisis de resultados **18**

5.- Interpretación de los datos **21**

6.- Bibliografía **21**

1.- Introducción

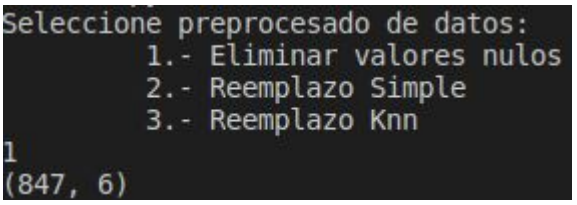
En esta práctica nos enfrentamos a un problema de aprendizaje supervisado en la predicción de datos científicos sobre el cáncer de mama. Se trata en general de un problema de clasificación cuya resolución basaremos en 5 algoritmos detallando posteriormente cuál es el que mejor se adapta a este teniendo en cuenta cómo se comportan cada uno de ellos. Dichos algoritmos son probabilísticos como Naive Bayes, de redes neuronales como Multi-layer Perceptron, de distancias como el algoritmo K-nn, de árboles de decisión como CART o Random Forest. No cabe olvidar el preprocesado de los datos pues tras leerlos hay que adaptarlos convenientemente al problema. Para todo nos ayudaremos de la extensa colección de librerías que posee python.

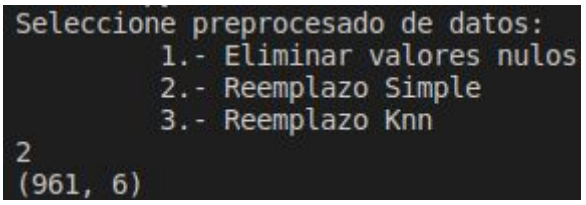
2.- Procesado de datos

El preprocesamiento será común a todos los algoritmos para que estén en igualdad de condiciones a la hora de hacer las predicciones:

2.1.- Eliminación de Filas

El conjunto de datos tiene un total de 961 entradas, este método elimina aquellas filas que contienen valores nulos reduciendo el número de muestras a solo 847. En este caso la precisión no es una buena medida pues cuando tenemos menor número de entradas la precisión aumenta, lo que ocurre es que al tener menor número de ejemplos el modelo no tiene que clasificar tantas entradas y por ello tiene mayor tasa de acierto.





2.2.- Reemplazo simple

En este caso haciendo uso de la función SimpleImputer se nos permite sustituir los valores nulos según varias estrategias disponibles. Por defecto, hace uso de la estrategia “mean”, esta estrategia reemplaza los valores nulos con la media de valores de la columna. Para que se pueda usar este método como es comprensible es necesario que la columna sea de tipo numérico, en nuestro caso al existir columnas no numéricas usaremos la estrategia most_frequent que reemplaza los valores nulos por el valor más frecuente de la columna. Manteniendo de esa forma el número total de muestras. Y reduciendo en este caso la precisión de los modelos por lo que ya se ha dicho anteriormente, al aumentar el número de muestras que clasificar el modelo falla en más ocasiones.

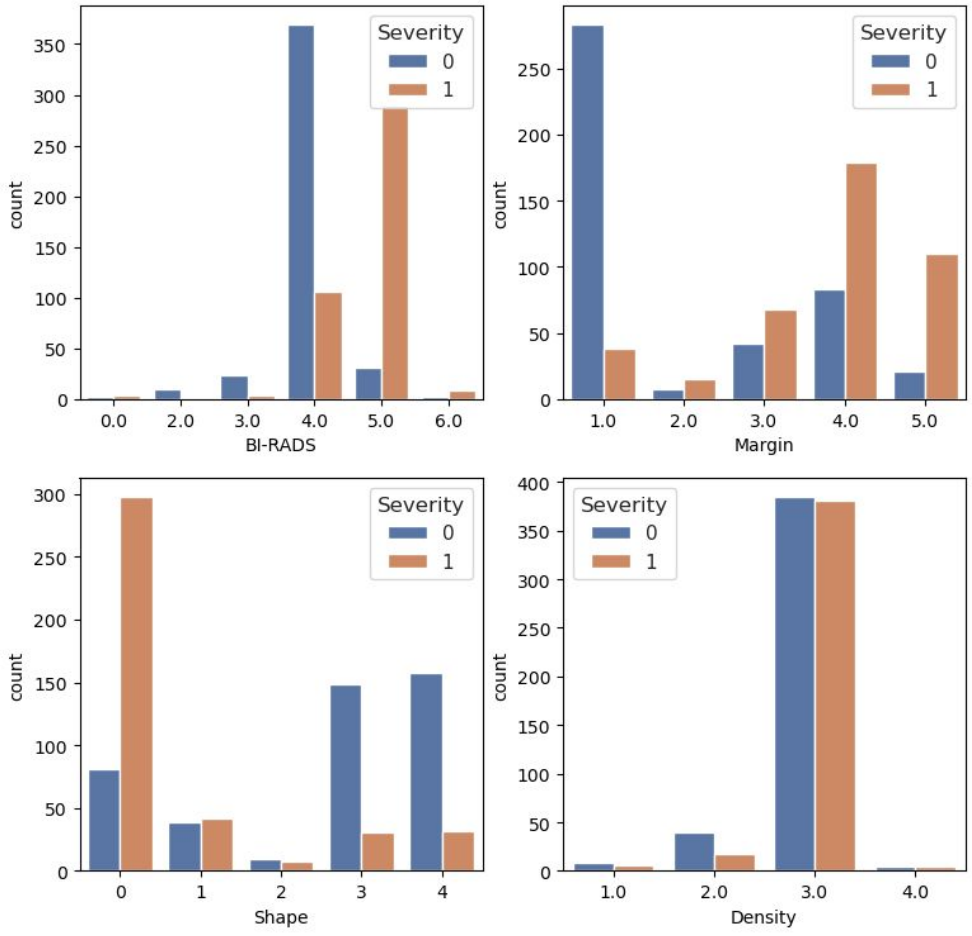
2.3.- Reemplazo Knn

En este caso se predicen los valores nulos usando el modelo K-nearest donde las muestras se imputan encontrando las muestras en el conjunto de entrenamiento "más cercano" a ella y promedia estos puntos cercanos para completar el valor.

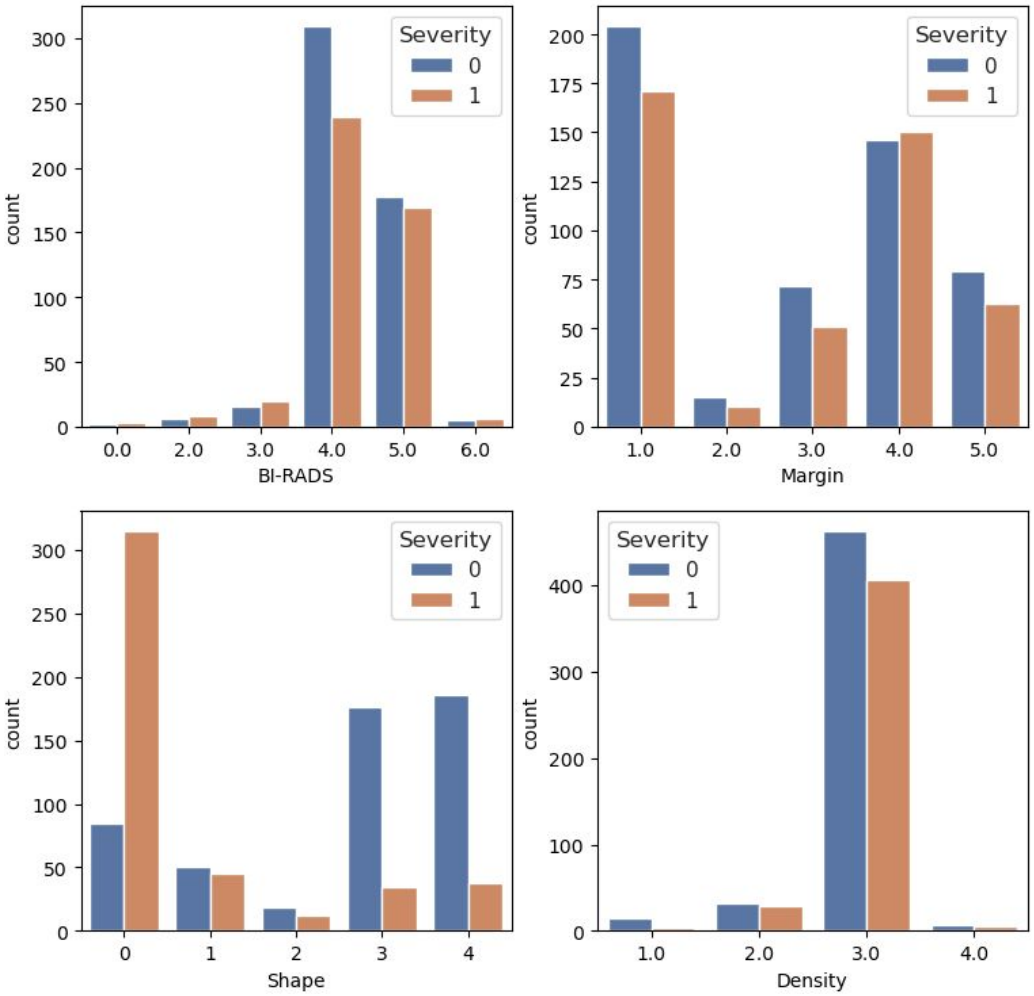
3.- Configuración de Algoritmos y sus resultados

En primer lugar, antes de proceder con el desarrollo de los algoritmos veremos cómo el preprocesado de datos influye en el valor de cada variable dando lugar a un diagnóstico u otro.

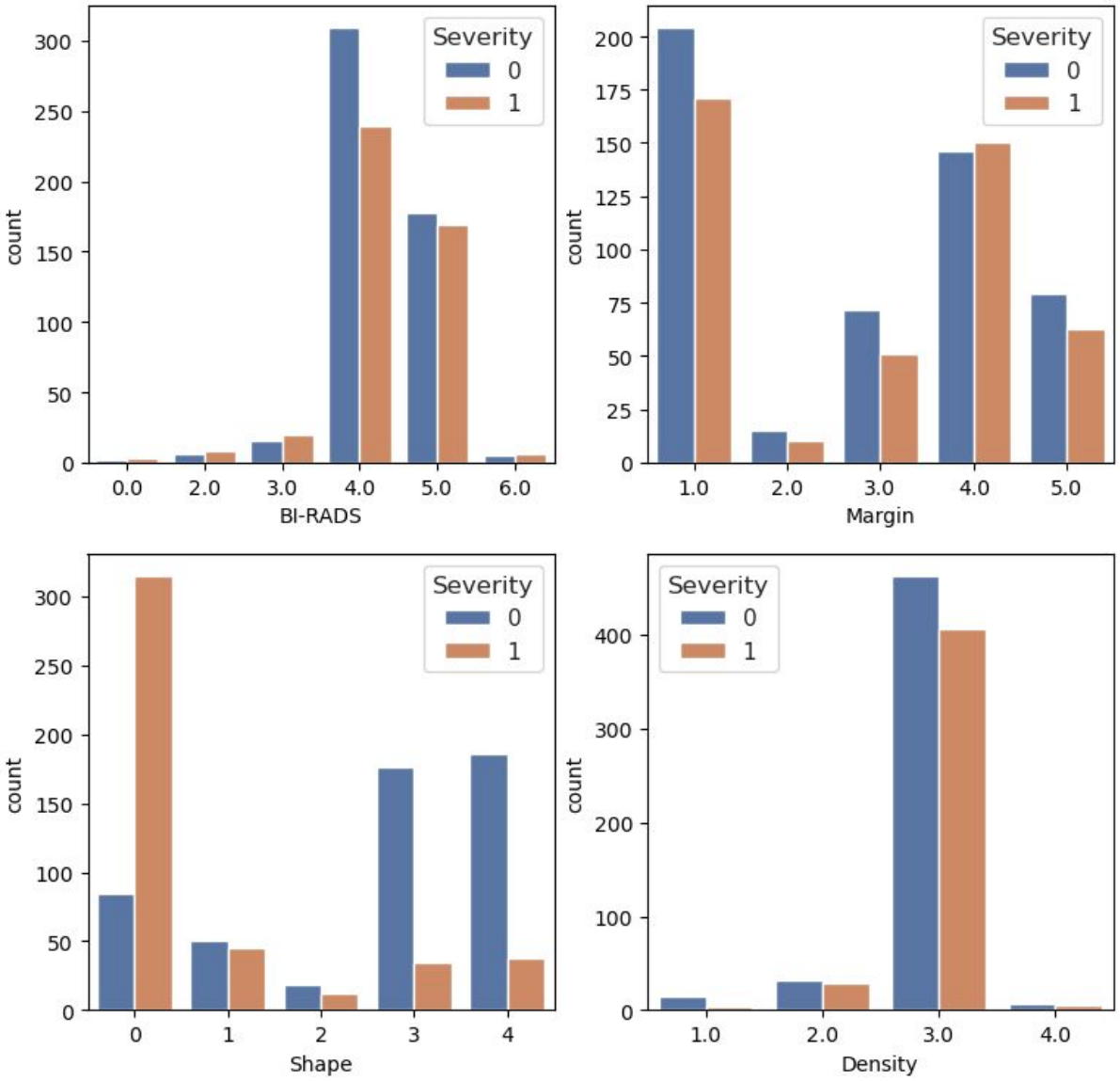
Eliminar Valores



Reemplazo Simple



Reemplazo K-nearest



En el primer caso se ve que las variables que se contabilizan alcanzan valores inferiores porque las filas directamente se eliminan al contener valores nulos.

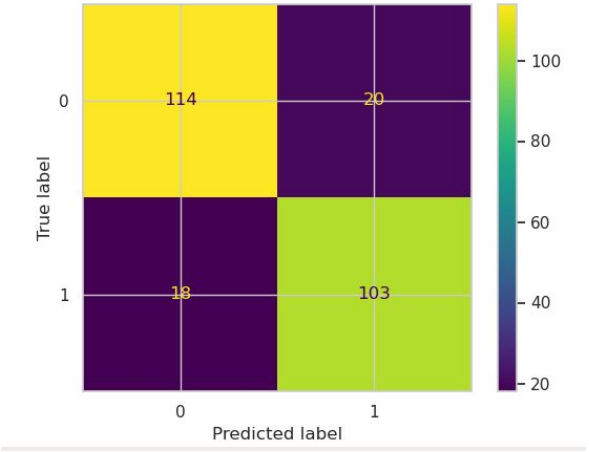
En el segundo caso al usar la estrategia most_frequent se puede observar que aquellos valores que tenían una alta tasa de contabilización en el primer caso, es aún mayor en el segundo caso.

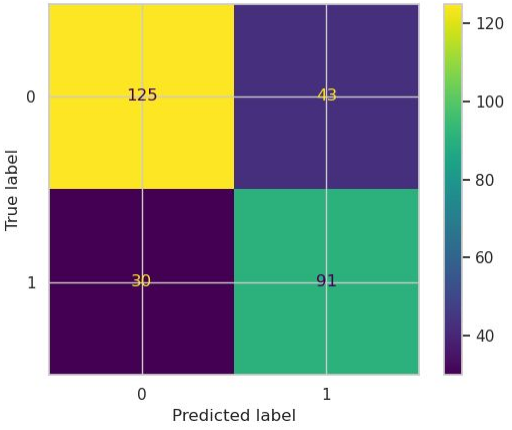
En el tercer caso al tener una muestra relativamente pequeña el uso de K-nn da resultados muy próximos a la estrategia most_frequent pues siguiendo esta estrategia toma los valores cercanos e inserta el promedio, si el conjunto de valores que se toma para hacer este cálculo es cercano al total de las muestras la estrategia es prácticamente idéntica a la segunda, por ello es importante remarcar que al ser estrategias tan parecidas y tener resultados tan próximos la segunda y tercera estratégia de procesado, esta última no se tendrá en cuenta en los análisis aunque se hará referencia a ella.

Para procesar los algoritmos dividiremos los conjuntos de entrenamiento y de test de forma aleatoria tras haber codificado aquellas variables cualitativas.

3.1.- Naive Bayes

Cómo se ha visto en teoría el modelo de Naïve Bayes es un modelo de red bayesiana orientada a la clasificación más simple.
Dependiendo del método de preprocesado usado obtendremos un nivel de precisión u otro, en el caso del primer método de eliminación de filas con valores nulos obtenemos 85’09% de precisión, por otro lado usando el segundo método de preprocesado obtenemos un 74’74% de precisión, esto se debe principalmente a que al tener una muestra mayor el porcentaje de acierto es menor pues hay que predecir sobre una mayor cantidad de ejemplos. En el tercer método de preprocesado el resultado de precisión es idéntico por las razones ya expresadas anteriormente.
Vistos los valores de precisión será mejor hacer uso de otra métrica para ver cómo de bueno es este modelo, se puede hacer uso de la validación cruzada para comprobar que de alguna manera si la bondad del modelo depende del conjunto de entrenamiento, en el caso de eliminar las filas con valores nulos la precisión media es del 80’99%, en el segundo caso (y el tercero) se obtiene una precisión media mínimamente superior a la precisión obtenida por defecto con los conjuntos de entrenamiento y test aleatorios, un 75’75%. De nuevo la validación cruzada no nos permite conocer realmente que preprocesado de datos hace al modelo mejor.
Probamos ahora con una matriz de confusión:

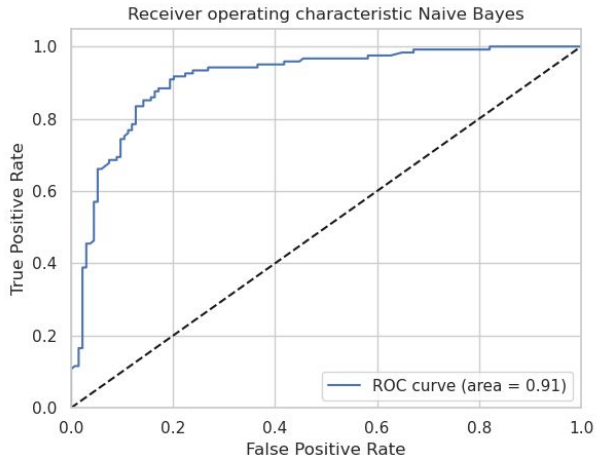


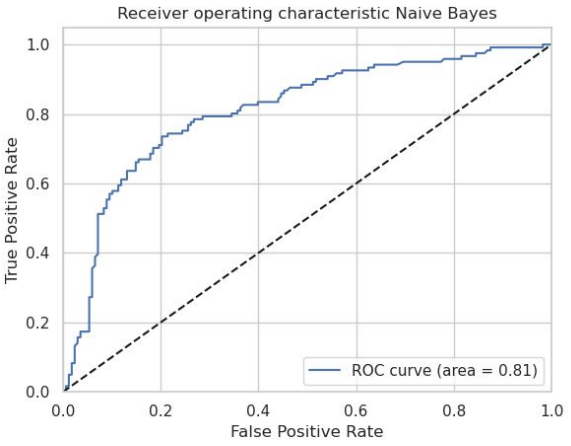


		Predicción	
		Positivos	**Negativos**
Observación	**Positivos**	Verdaderos Positivos (VP)	Falsos Negativos (FN)
	Negativos	Falsos Positivos (FP)	Verdaderos Negativos (VN)

En el primer caso vemos que el número de falsos positivos y falsos negativos es bastante menor que en el segundo caso cuando el número de aciertos en el primer y segundo caso son mínimamente diferentes. Aunque parezca que esta métrica nos salva del problema que teníamos no es así debido a que el número de veces que la predicción ha sido errónea ha aumentado porque el número de muestras ha sido también mayor.

Todo esto nos conduce a la mejor forma de medir la bondad del modelo, haciendo uso de la curva ROC (Receiver Operating Characteristic), se busca con ella un modelo donde los VP (como expresa la tabla de la página anterior) aumenten a un ritmo superior a los FP. Dicha curva tiene la siguiente forma para el primer preprocesado y el segundo (~ tercero) respectivamente:





A la hora de leer una curva ROC cuanto más cerca estén los valores de esta de la esquina superior izquierda se puede decir que mejor es el modelo, por lo que se puede observar que independientemente del tamaño de la muestra, el modelo de datos entrenado con la muestra menor es mejor que aquel que ha sido entrenado con la inserción simple (o inserción Knn).
Otras medidas de interés pueden ser la media geométrica de TPR y TNR, la F1-score o media armónica de PPV y TPR y la media geométrica de PPV y TPR como quedan representadas en la siguiente tabla:

	Eliminar filas	Reemplazo simple
G-mean	0.8509929332802403	0.748046123414072
F1-Score	0.8442622950819672	0.7137254901960784
G-measure	0.5910492488321798	0.56413287395073

La media geométrica trata de maximizar el acierto en ambas clases con un buen balance.
La medida geométrica por su lado, de forma similar a F1-score, penalizan más los errores al clasificar ejemplos positivos que los errores al clasificar los ejemplos negativos.
En cualquiera de las medidas se puede observar que el modelo entrenado con el conjunto de datos menos consigue mejores resultados que el segundo.

3.2.- Multi-layer Perceptron

En este caso Sklearn hace uso de una red neuronal de la clase de feedforward artificial neural network (ANN), se denominan así cuando las conexiones entre los nodos no forman un ciclo. MLP tiene una serie de limitaciones importantes a conocer al aplicar esta resolución a un problema, la primera es que no extrapola bien, es decir, si la red se entrena mal, las salidas pueden ser imprecisas. La segunda es que la existencia de mínimos locales en la función de error dificulta el entrenamiento.
Cada nodo posee un peso que minimiza el error en toda la salida, usando un descenso de gradiente (algoritmo de minimización para cualquier función), encuentra el cambio en cada peso representado por la siguiente función:

$$\Delta w_{ji}(n) = -\eta \frac{\partial \mathcal{E}(n)}{\partial v_j(n)} y_i(n)$$

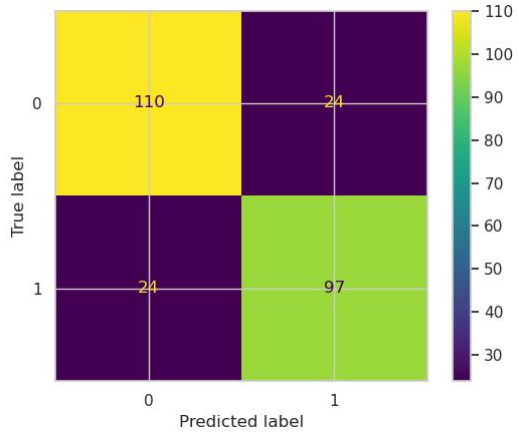
donde y es la salida de la neurona anterior y η es el ritmo de aprendizaje, que se selecciona cuidadosamente para que los pesos converjan a una respuesta rápidamente, sin producir oscilaciones.

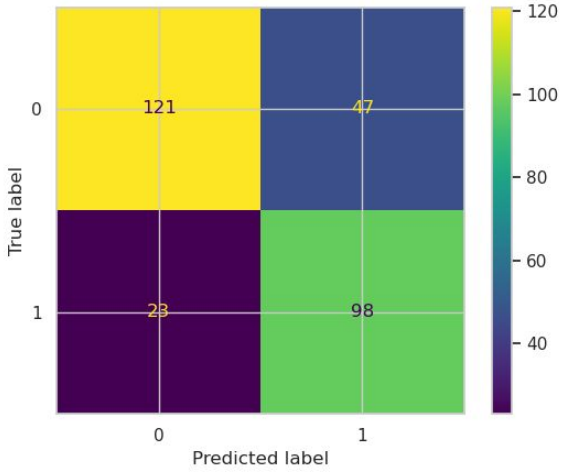
Dadas las premisas anteriores puede parecer que no es el mejor modelo para el conjunto de datos que tenemos, sin embargo consigue valores de acierto bastante buenos aunque peores que el modelo bayesiano seguramente esto se deba al tamaño del conjunto de entrenamiento pues en ejemplos como MNIST (base de datos con números escritos a mano) con 70000 imágenes consigue valores de precisión del 97% .

	Eliminar filas	Reemplazo simple
Precisión	0.8117647058823529	0.7577854671280276
Validación Cruzada	0.7993038635572572	0.773159542314335

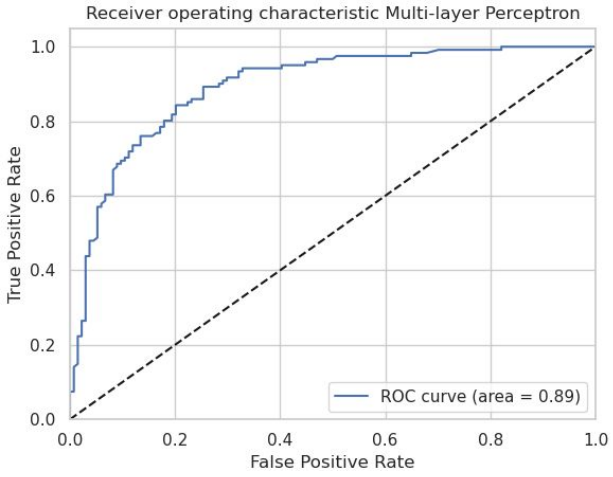
Como ya he expresado antes, cuantos más datos tengamos en el conjunto de entrenamiento, mejor modelo tendremos como resultado, es por eso que a pesar de que Naïve bayes consigue una mayor precisión al eliminar todos los datos, es en el reemplazo simple donde MLP consigue una mayor precisión pues el tamaño de la muestra, aumenta.

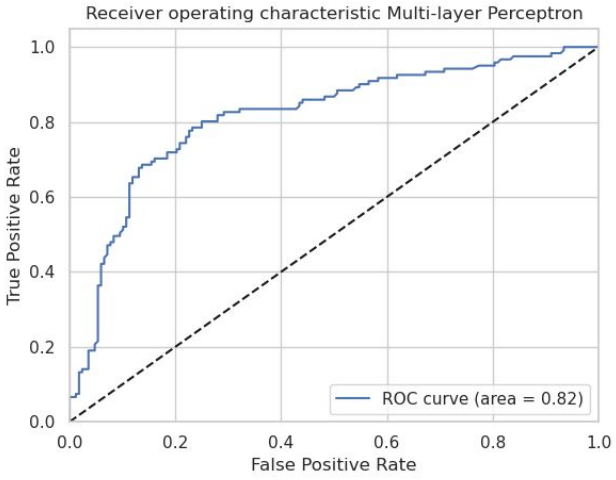
Al mirar las matrices de confusión(eliminando filas y reemplazo respectivamente) :





podemos ver que el número de acierto es mayor en el segundo preprocesado, además la tasa de falsos positivos y negativos es menor que el primer modelo con el mismo preprocesado.





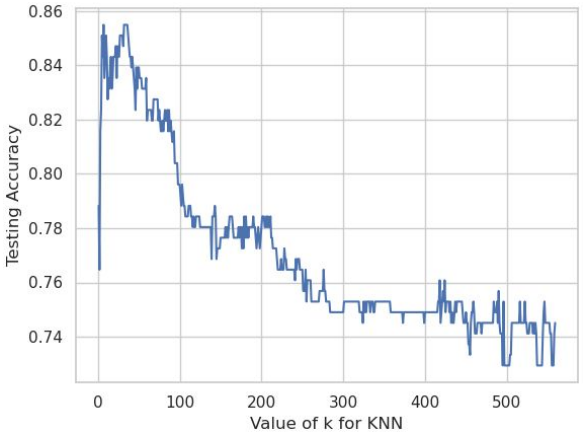
Efectivamente la curva de ROC reafirma que este modelo consigue mejores valores con el reemplazo simple que el modelo anterior, sin embargo, eliminando las filas que contienen valores nulos Naïve Bayes es mejor modelo.
Otro datos de importancia para determinar la calidad del modelo:

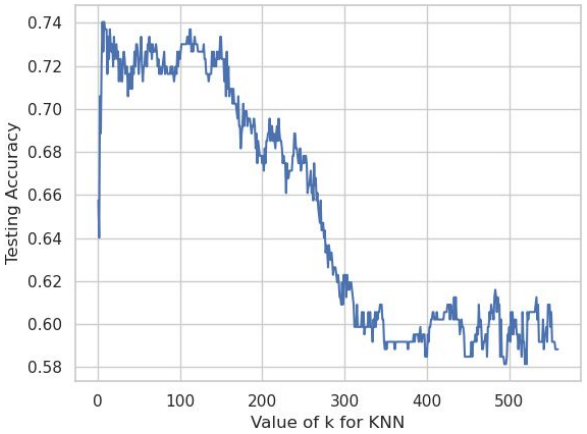
	Eliminar filas	Reemplazo simple
G-mean	0.8112171534266078	0.7637626158259733
F1-Score	0.8016528925619834	0.736842105263158
G-measure	0.5822460564703765	0.567899463877536

Estos valores reafirma lo ya dicho, de nuevo MLP consigue mejores valores que Naïve Bayes con el reemplazo simple, sin embargo, Naïve Bayes es mejor en el primer modelo de preprocesado de los datos.

3.3.- K-nn Algorithm (Vecinos más próximos)

El algoritmo K-nn o vecinos más cercanos es un método de clasificación supervisada al igual que el resto. En el reconocimiento de patrones, el algoritmo k-nn es usado como método de clasificación de objetos basado en un entrenamiento mediante ejemplos cercanos en el espacio de los elementos. Es el mismo algoritmo del que hace uso el proceso de inserción que usa el tercer modelo de preprocesado. Este algoritmo de clasificación depende muy fuertemente de una variable, K, es el número de ejemplos más cercanos seleccionados a partir de los cuales toma la clase que más se repite para dar una estimación. Generalmente, para valores grandes de k reducen el efecto de ruido en la clasificación, pero crean límites entre clases parecidas.
Para seleccionar el mejor valor de k he tomado cada uno de los valores posibles de esta variable y he hecho una estimación, tomando aquella k que obtiene la mayor precisión, como se muestra a continuación en las gráficas que expresan el valor de k respecto el nivel de precisión que esta consigue eliminando las filas y haciendo un reemplazo simple respectivamente:

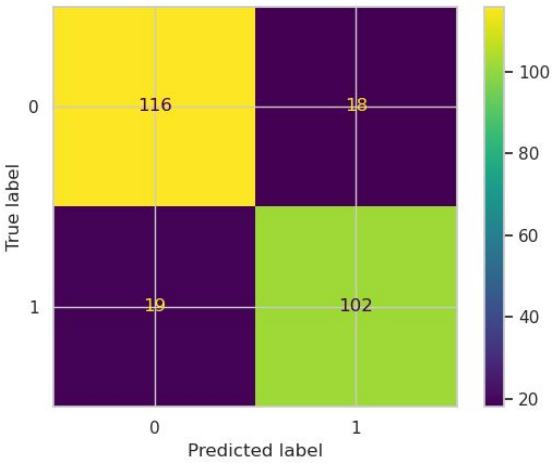


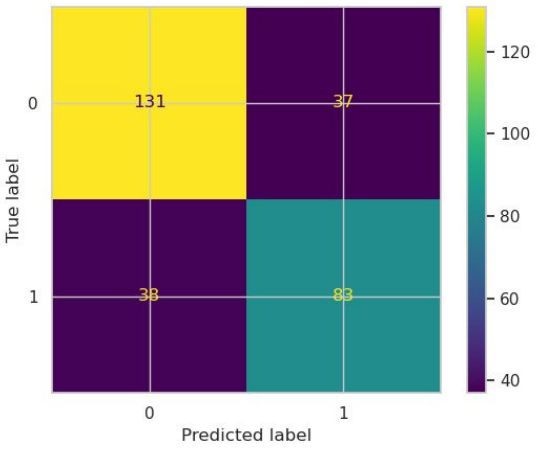


	Eliminar filas	Reemplazo simple
Precisión	0.8352941176470589	0.726643598615917
Validación Cruzada	0.7981273929690219	0.7283894645941278
K	7	6

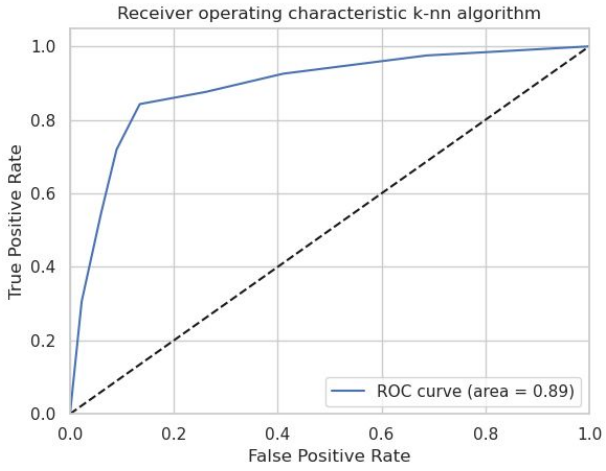
Aunque en el primer caso la precisión del conjunto de entrenamiento aleatorio es superior que la del modelo MLP, al hacer la validación cruzada por muy poco es el peor modelo hasta el momento. Cuando se produce el reemplazo simple es por mucho el peor de los tres. Esto se debe a que el set de entrenamiento contiene muestras que son cercanas entre sí y el modelo no puede aprender a clasificar

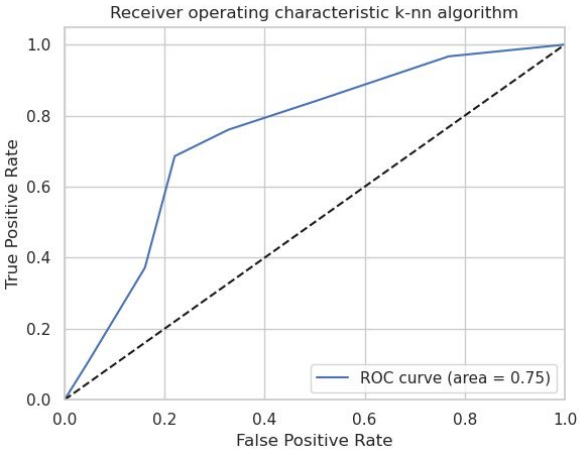
de forma adecuada. Además en la inserción por reemplazo simple al usar la técnica del más frecuente esto hace que empeore aún más el set de entrenamiento.
Al mirar las matrices de confusión, de nuevo, eliminando filas con valores nulos y reemplazo simple respectivamente:





La tasa de aciertos es bastante baja y aunque la de error es menor que en los otros casos es en la segunda matriz donde se puede observar un mayor error y en la celda [4,4] una menor tasa de acierto además.





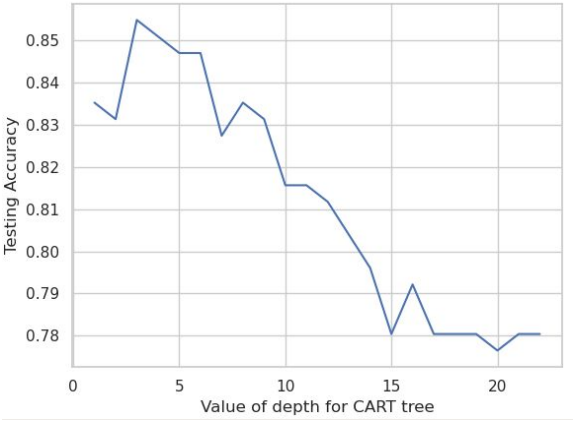
En la curva de ROC, el índice AUC es el mismo eliminando las filas con valores nulos que en el modelo MLP (peor que Naïve Bayes), sin embargo, cuando se hace uso del preprocesado de reemplazo simple el valor AUC obtenido es el peor de los tres.
Otro datos de importancia para determinar la calidad del modelo:

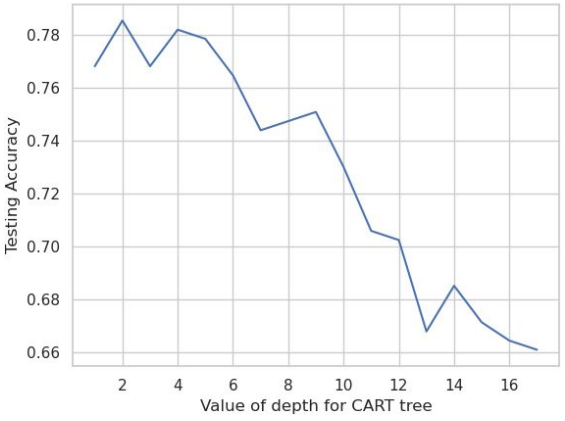
	Eliminar filas	Reemplazo simple
G-mean	0.8542480500982037	0.7313535402164207
F1-Score	0.846473029045643	0.6887966804979254
G-measure	0.7417261495035429	0.5737848214028444

Como se puede observar en estos últimos datos, se obtienen por lo general mejores valores que en los otros modelos, esto sucede porque a la hora de mirar las matrices de confusión este modelo consigue clasificar con éxito mayor casos positivos y se podría valorar como mejor modelo hasta el momento si lo que queremos es saber si el cáncer es benigno.

3.4.- CART Algorithm

Este algoritmo basado en árboles de decisión hace uso del índice estadístico de GINI e intenta minimizarlo en cada división del árbol hasta llegar a cero, con el conjunto de etiquetas vacío. En dicho momento se podrá decir que el conjunto está perfectamente clasificado, sin embargo, en nuestro caso con un conjunto de datos tan amplio dicho índice estadístico no se hace 0, hasta llegar a una profundidad 23 en el caso de eliminar las filas con valores nulos y una profundidad 18 si reemplazamos los valores nulos. Un árbol con tal profundidad tiene un número proporcional de hojas haciendo la clasificación más costosa computacionalmente e incluso más imprecisa, por ello puede ser interesante estudiar con qué profundidad nuestro modelo consigue una mayor precisión en la clasificación. Como hicimos en el algoritmo k-nn podemos ver a continuación el nivel de precisión del algoritmo en función de la profundidad para el preprocesado en el que se eliminan las filas y aquel en el que se realiza un reemplazo simple respectivamente:

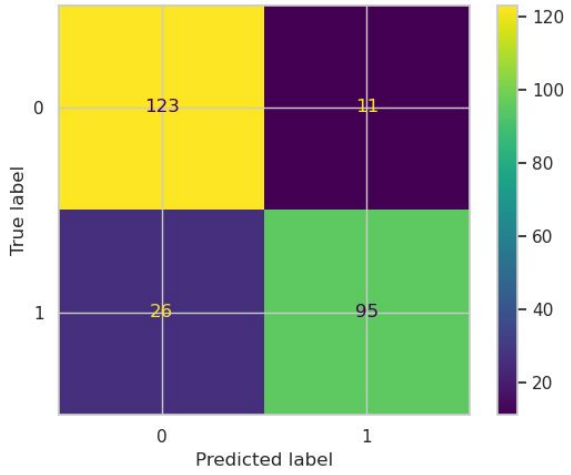


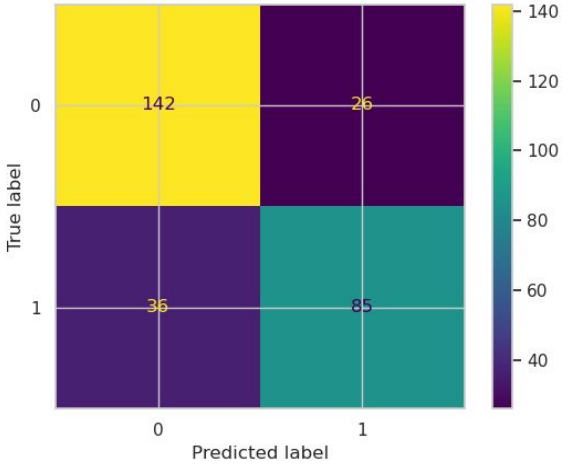


El algoritmo toma aquellos atributos que considera más importantes y aunque GINI no llega a ser 0, consigue niveles de precisión bastante altos para el conjunto de muestras que tenemos.

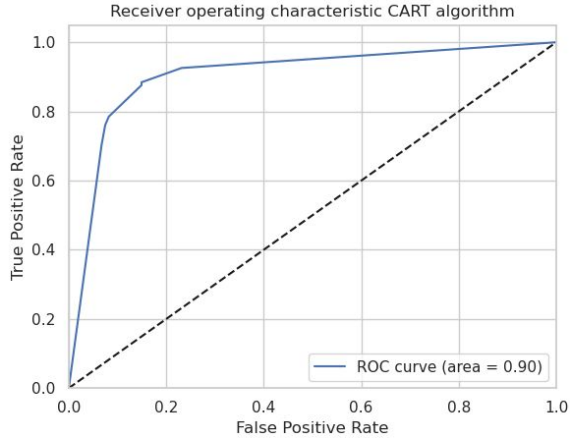
	Eliminar filas	Reemplazo simple
Precisión	0.8549019607843137	0.7854671280276817
Validación Cruzada	0.8394082840236686	0.765878670120898
Depth	3	2
Max_depth	23	18

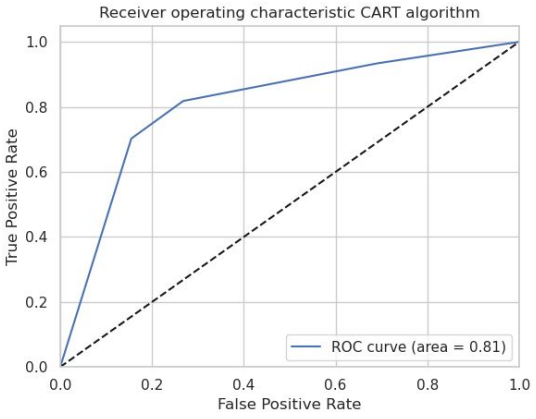
Puedo adelantar que este algoritmo es el que mejores valores de precisión da en general excepto al hacer la validación cruzada pues en media al hacer el preprocesado por inserción simple es el modelo MLP el que mejor valor da.
Como ya hemos hecho en los otros modelos veremos las matrices de confusión al eliminar filas con valores nulos y al reemplazar los mismos con la inserción simple:





En las matrices de confusión se puede observar que el número de aciertos positivos es mayor en ambos casos manteniendo un perfil de error bastante bajo.



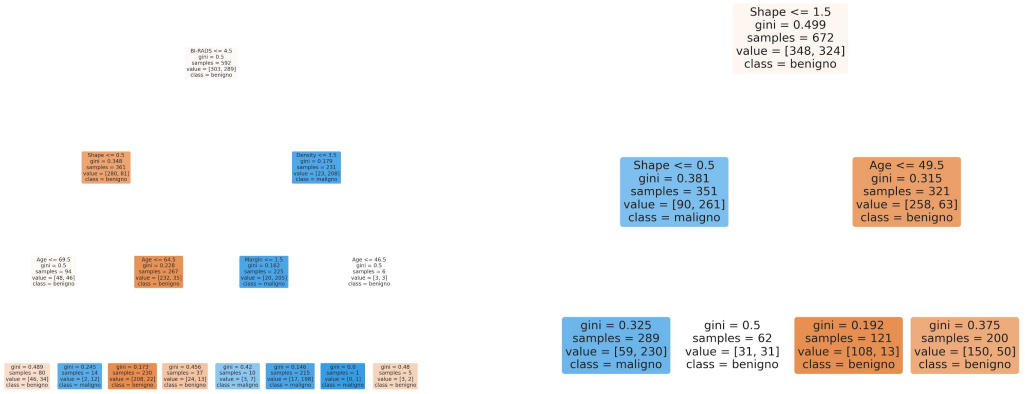


En el primer preprocesado de datos podemos ver por el valor AUC que este algoritmo es el que mejor funciona para la muestra de datos que tenemos después de Naïve Bayes, en el segundo preprocesado que se propone nuestro modelo de clasificación se queda por detrás de otros que hemos presentado como MLP y con el mismo AUC que Naïve Bayes.

Otro datos de importancia para determinar la calidad del modelo:

	Eliminar filas	Reemplazo simple
G-mean	0.8489249036539762	0.7705597305256439
F1-Score	0.8370044052863437	0.7327586206896551
G-measure	0.8488245712457232	0.8370944835803463

Por lo general se consiguen mejores valores incluso que en el modelo anterior, esto nos lleva deducir que como ya se ha visto en las matrices de confusión la tasa de aciertos positivos es superior a otros casos.



Los árboles de decisión con profundidad 3 y 2 respectivamente.

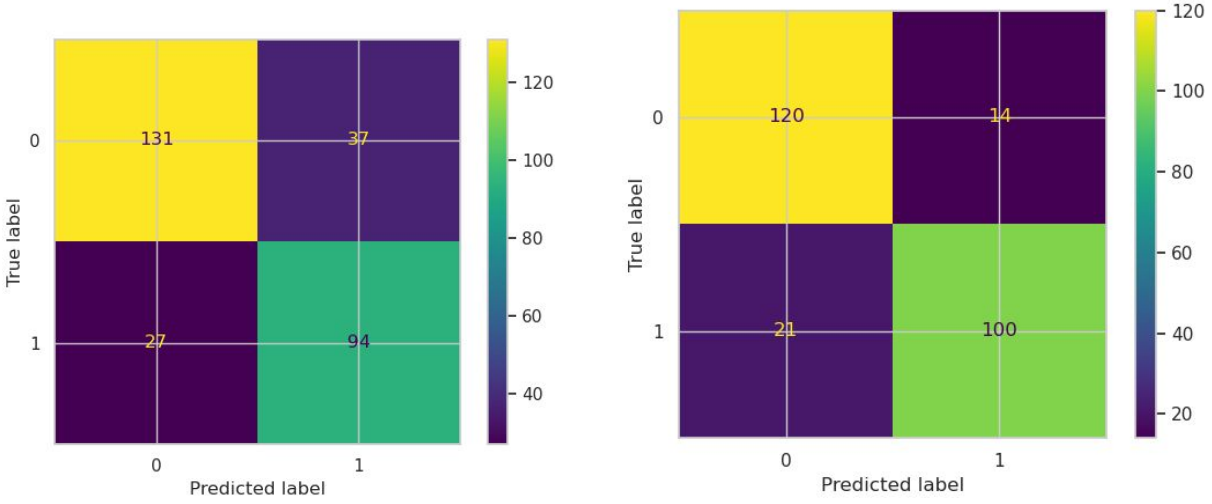
3.5.- Random Forest

Repetimos en este caso un nuevo modelo basado en árboles de decisión, el método genera el árbol de clasificación como un conjunto de árboles de decisión generados a partir de tomar varias muestras de forma aleatoria del conjunto total de datos, con el fin de mejorar la precisión y controlar el sobreajuste. La fortaleza de este método reside en que cada árbol protege al otro de sus errores individuales, además este modelo funciona, al igual que MLP, muy bien con grandes cantidades de datos, puede parecer que la muestra que tenemos es reducida pero para nuestro modelo a diferencia del MLP, en este caso es suficiente.

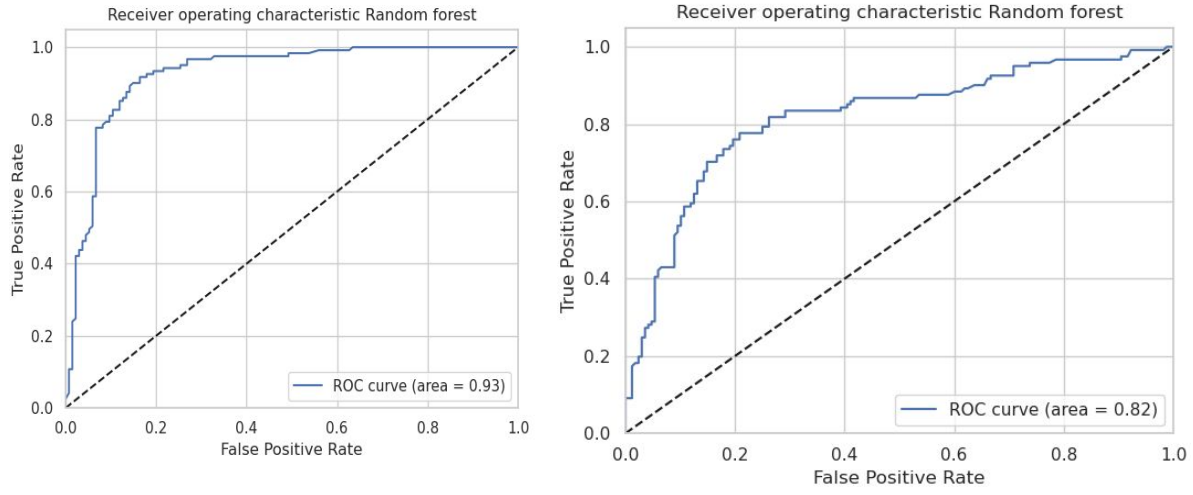
Al igual que en el modelo anterior la profundidad es un dato relevante a la hora de mejorar la precisión, en este caso la calcularemos haciendo uso de funciones propias de sklearn en lugar de hacerlo con un bucle for repitiendo las estimaciones con distintas profundidades en cada ciclo aunque es lo que hace el método usado de manera interna.

	Eliminar filas	Reemplazo simple
Precisión	0.8627450980392157	0.7785467128027682
Validación Cruzada	0.8287782805429863	0.7721232728842833
Depth	3	3

La precisión del estimador es la mejor que hemos tenido hasta el momento, sin embargo al hacer la validación cruzada podemos ver el modelo de CART consigue mejores valores, por otro lado al realizar el reemplazo simple como en todos los casos se pierde precisión pero aún así es el mejor de los estimadores porque aunque fijándonos en los números no los supere con unos ajustes el valor podría cambiar y ser superior.



Por las matrices de confusión se puede intuir que es el mejor modelo que hemos visto hasta el momento y las curvas de abajo lo confirman pues el número de aciertos es bastante alto manteniendo una tasa de fallos bastante baja.



Otros datos que pueden reforzar la hipótesis de que es el mejor modelo:

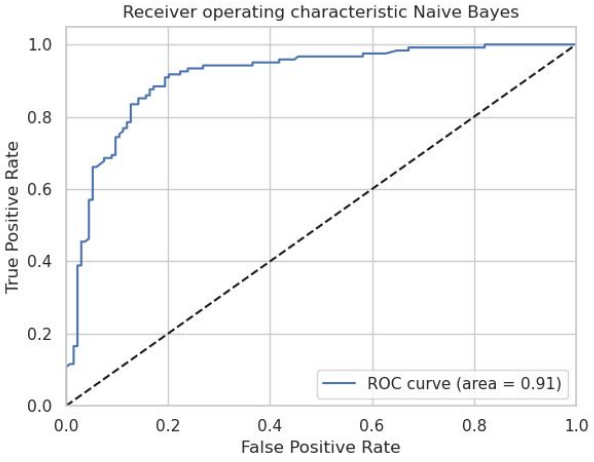
	Eliminar filas	Reemplazo simple
G-mean	0.8516449519686256	0.7783093515271017
F1-Score	0.8412017167381973	0.746031746031746
G-measure	0.6260141029113568	0.5541771353257202

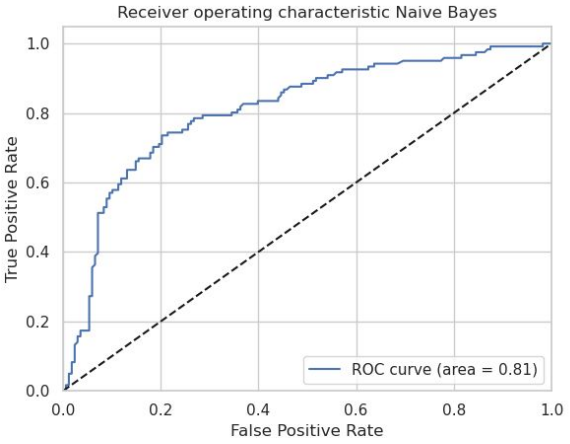
En general, se puede observar que se consiguen mejores valores que en resto de modelos, en G-measure pueden ser menores porque se alcanzan valores tan cercanos a cero al hacer la media geométrica que para que el cálculo se pueda realizar adecuadamente he decidido eliminarlos.

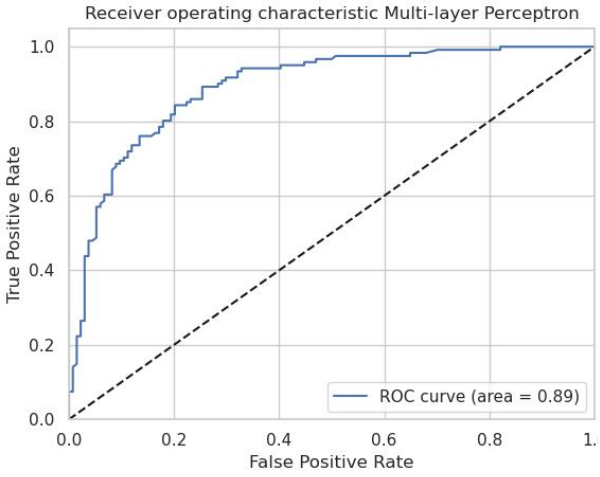
4.- Análisis de resultados

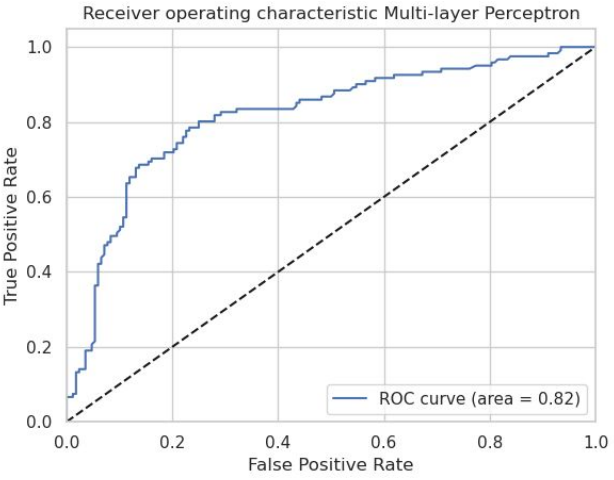
		Precisión	Validación Cruzada	AUC	G-mean	F1-score	G-measure
Naïre Bayes	Eliminar Repetidos	0.8509	0.8099	0.91	0.8509	0.8442	0.5910
	Reemplazo Simple	0.7474	0.7575	0.81	0.74804	0.7137	0.5641
	Reemplazo K-nn	0.7474	0.7575	0.81	0.74804	0.7137	0.5641
Multi-layer Perceptron	Eliminar Repetidos	0.8117	0.7993	0.89	0.8112	0.8016	0.5822
	Reemplazo Simple	0.7577	0.7731	0.82	0.7637	0.7368	0.5678
	Reemplazo K-nn	0.7577	0.7731	0.82	0.7637	0.7368	0.5678
k-nn algorithm	Eliminar Repetidos	0.8352	0.7981	0.89	0.8542	0.8464	0.7417
	Reemplazo Simple	0.7266	0.7283	0.75	0.7313	0.6887	0.5737
	Reemplazo K-nn	0.7266	0.7283	0.75	0.7313	0.6887	0.5737
CART algorithm	Eliminar Repetidos	0.8549	0.8394	0.90	0.8489	0.8370	0.8488
	Reemplazo Simple	0.7854	0.7658	0.81	0.7705	0.7327	0.8370
	Reemplazo K-nn	0.7854	0.7658	0.81	0.7705	0.7327	0.8370
Random Forest	Eliminar Repetidos	0.8588	0.8381	0.93	0.8559	0.8461	0.5622
	Reemplazo Simple	0.7785	0.7700	0.82	0.7783	0.7460	0.5899
	Reemplazo K-nn	0.7785	0.7700	0.82	0.7783	0.7460	0.5899

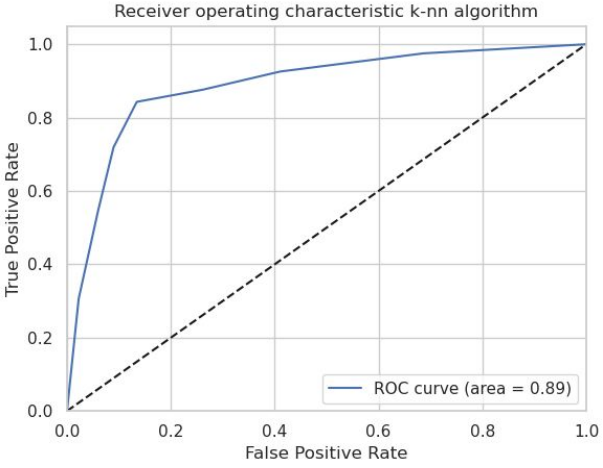
Tanto la curva de ROC como el valor de AUC me parecen valores altamente representativos de la bondad de los modelos por ello me gustaría hacer una recopilación de los gráficos obtenidos para los dos primeros tipos de preprocesado presentados:

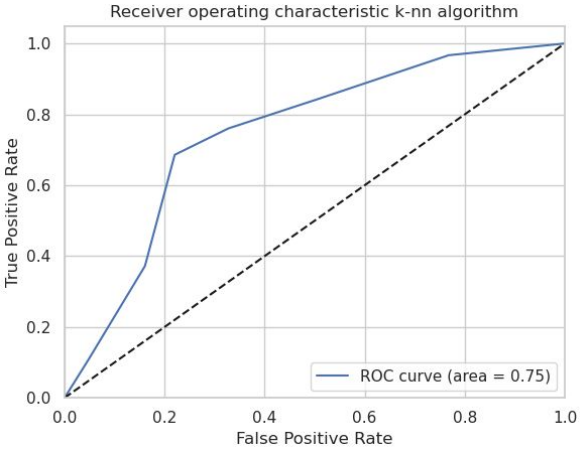


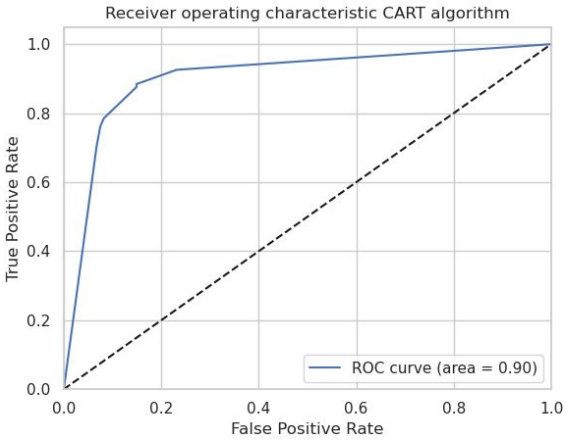


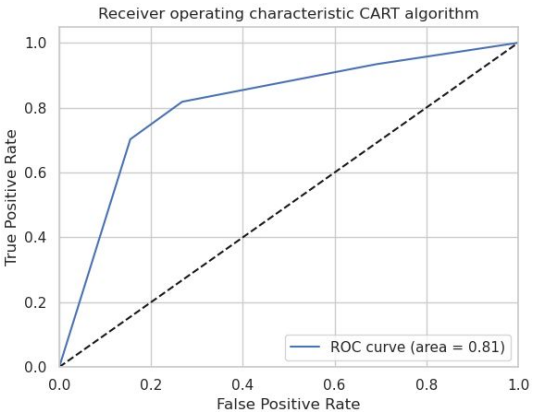


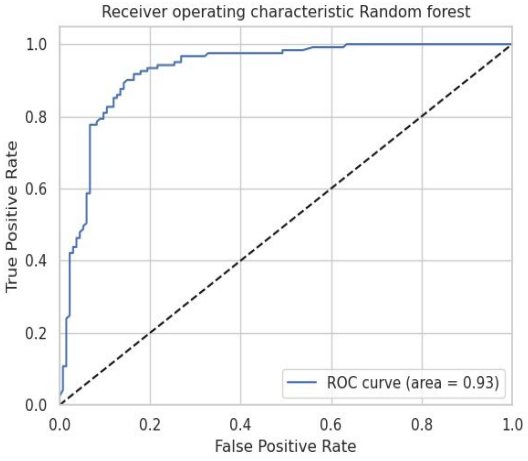


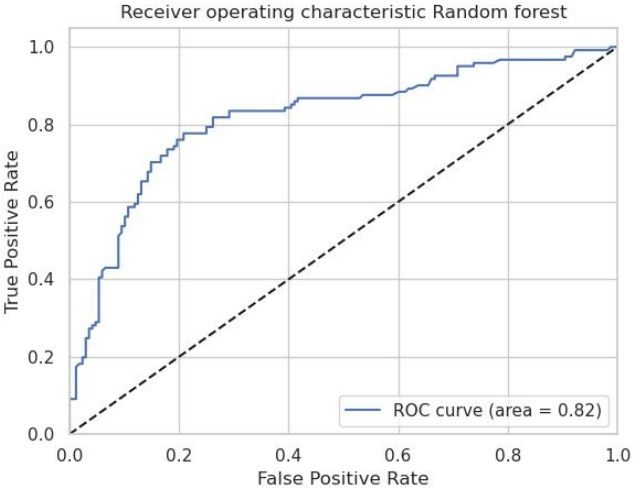












Existen distintas mejoras que se pueden aplicar a los modelos a través de los parámetros que se pueden pasar a cada una de las funciones, sin embargo, es el modelo K-nn el que puede tener una mayor capacidad de mejora usando una serie de pesos para obtener así un modelo k-nn ponderado, que otorgue mayor importancia a unas etiquetas u otras de forma que se consiga una mayor precisión en la predicción.

5.- Interpretación de los datos

En este punto haré fundamentalmente dos cosas, ordenar los modelos propuestos en función de distintos criterios y fines y en segundo lugar, justificar cuál elegiría personalmente y por qué. Ya se ha ido haciendo una clasificación progresiva de los modelos justificando la misma con distintos valores estadísticos y gráficos:

	Eliminar filas	Reemplazo simple
1	Random Forest	Multi-layer perceptron
2	Naïve Bayes	Random Forest
3	CART	CART
4	K-nn	Naïve Bayes
5	Multi-layer perceptron	K-nn

El fruto de dicha clasificación es principalmente el valor de AUC, cuando este coincide se hace uso de la precisión media al realizar la validación cruzada que consigue el modelo. En el primer caso Random Forest es por mucho el mejor, esto se debe a la principal fortaleza del mismo que suple las debilidades individuales de cada uno de los árboles sobre el otro a una profundidad determinada. Multi-layer perceptron obtiene mejores valores en la segunda ocasión debido a que este algoritmo se nutre principalmente de la cantidad de datos que tiene nuestra muestra. Knn ocupa posiciones tan bajas por la cercanía de los valores de la muestra que no permiten al modelo entrenarse para hacer una clasificación adecuada, además al usar una estrategia del más frecuente para la inserción simple empeora la muestra para dicho modelo.
Personalmente seleccionaría Random Forest pues consigue el mejor valor de AUC y mejores valores de precisión en el primer preprocesado y en el segundo, aunque no ocupa la primera posición consigue también valores muy buenos, dando a entender que para este problema el modelo es bastante bueno por lo general.

6.- Bibliografía

https://sci2s.ugr.es/node/125
https://scikit-learn.org/stable/user_guide.html
https://pradogrado2021.ugr.es/pluginfile.php/299497/mod_resource/content/1/IN_evaluacion_de_clasificadores.pdf
https://towardsdatascience.com/understanding-random-forest-58381e0602d2
https://en.wikipedia.org/wiki/Decision_tree_learning
https://es.wikipedia.org/wiki/K_vecinos_m%C3%A1s_pr%C3%B3ximos
https://en.wikipedia.org/wiki/Multilayer_perceptron
https://es.wikipedia.org/wiki/Clasificador_bayesiano_ingenuo
https://en.wikipedia.org/wiki/Random_forest
https://scikit-learn.org/stable/modules/impute.html