14/12/2023

INTELIGENCIA ARTIFICIAL

Práctica - 4

G.1311 - P4

Joaquín Abad Díaz - Carlos García Santa

Práctica de aprendizaje automático

Parte 1

Construcción de clasificadores en bases de datos sintéticas

ESTO ES PARA EL FINAL

1. Describe las características de los datos:

- Tipo de datos
- Número de ejemplos de cada una de las clases.
- Número y tipo (nominales no ordenados, nominales ordenados, numéricos) de atributos.

Los datos son una mezcla de atributos categóricos (nominales) y numéricos, con un atributo de clase binario, el conjunto de datos contiene 700 ejemplos, con 300 ejemplos en cada clase y hay 20 atributos, 7 numéricos y 13 categóricos.

2. Detalla la metodología utilizada:

- Partición de los datos: tamaño de los conjuntos de entrenamiento y test, uso de estratificación en el muestreo.
- Preprocesamiento: codificación de los atributos, construcción y selección de características, normalización, etc. (¡solo se debe utilizar la información del conjunto de entrenamiento!)
- Determinación de los hiperparámetros; por ejemplo, mediante búsqueda en rejilla y validación cruzada.
- Estimación del error de generalización y su incertidumbre.

A PARTIR DE AQUI LO OTRO

Prueba a cambiar los siguientes parámetros (al menos 3-5 variantes en cada caso) y observa las consecuencias en la frontera de clasificación construida:

- **Número de vecinos en k-nn.**
    - **¿Debería ser impar cuando hay dos clases? Justifica la respuesta.**

En el algoritmo k-nn, cuando tenemos dos clases, es recomendable optar por un número impar de vecinos, k. Esto se basa en la posibilidad de un empate perfecto que puede ocurrir con un número par de vecinos. Por ejemplo, si k es 4, podríamos tener dos votos para cada clase. En tal caso, el algoritmo tendería a elegir el vecino que aparece primero en el conjunto de datos, introduciendo así un sesgo arbitrario. En contraste, al utilizar un número impar, nos aseguramos de que siempre existirá una mayoría, aunque sea por un solo voto. Por lo tanto, para situaciones con dos clases, un número impar para k es la mejor opción.

- **¿Debería ser impar cuando hay más de dos clases? Justifica la respuesta.**

Con más de dos clases, es correcto optar también por un número impar de vecinos. Aunque la probabilidad de empates exactos disminuye con un mayor número de clases, un número impar de vecinos sigue siendo beneficioso para reducir aún más dicha probabilidad. Por ejemplo, si k es par en un escenario de tres clases, podría darse un empate entre las clases. Utilizando un número impar, se incrementa la posibilidad de obtener una mayoría más definida para alguna de las clases.

Nº de vecinos	Score input 1	Score input 2	Score input 3	Score input 4
1	0,93	0,97	1,00	0,97
9	0,96	0,98	1,00	0,98
21	0,95	0,94	0,99	0,98
51	0,94	0,83	0,95	0,92
101	0,95	0,69	0,87	0,59
201	0,95	0,60	0,81	0,53

Tabla 1: puntuaciones para 4 entradas con diferentes números de vecinos (impares)

- **Profundidad máxima de los árboles de decisión.**

Al aumentar la profundidad máxima del árbol de decisión de 1 a 5, se observan diferencias bastante notables. Esto es especialmente evidente en los conjuntos de datos de separación lineal y en los de forma circular. A pesar de esto, cuando la profundidad máxima supera el valor de 5, no se observan incrementos adicionales en las puntuaciones, ya que esto implica adaptar los conjuntos de datos a un árbol de decisión excesivamente complejo, resultando en un sobreajuste.

Profundidad	Score input 1	Score input 2	Score input 3	Score input 4
1	0,88	0,59	0,79	0,61
2	0,92	0,59	0,88	0,73
3	0,93	0,76	0,88	0,82
5	0,92	0,91	0,95	0,93
10	0,91	0,93	0,95	0,93
20	0,91	0,93	0,95	0,94
50	0,91	0,93	0,95	0,93

Tabla 2: puntuaciones para 4 entradas con diferentes profundidades

- **Número de neuronas en la red neuronal y máximo número de épocas de entrenamiento. Nota: (50,) indica una única capa oculta con 50 neuronas. (50,10,) indica dos capas ocultas con 50 y 10 neuronas respectivamente. (50,10,20,) indica tres capas ocultas con 50, 10 y 20 neuronas respectivamente, etc**

Se han probado diferentes configuraciones: primero se ha probado con una única capa a aumentar el número de neuronas, esto ha proporcionado mejores resultados pero aumentando considerablemente el coste computacional; posteriormente se ha introducido una nueva capa y se ha mantenido el número de neuronas de la primera a 50, hemos observado que el mejor punto entre coste computacional y resultados en este caso ha sido (50, 10), aumentando progresivamente el número de neuronas en la segunda capa los resultados demostraron un sobreajuste; por último se introducimos una última capa los resultados también parecen indicar sobreajuste.

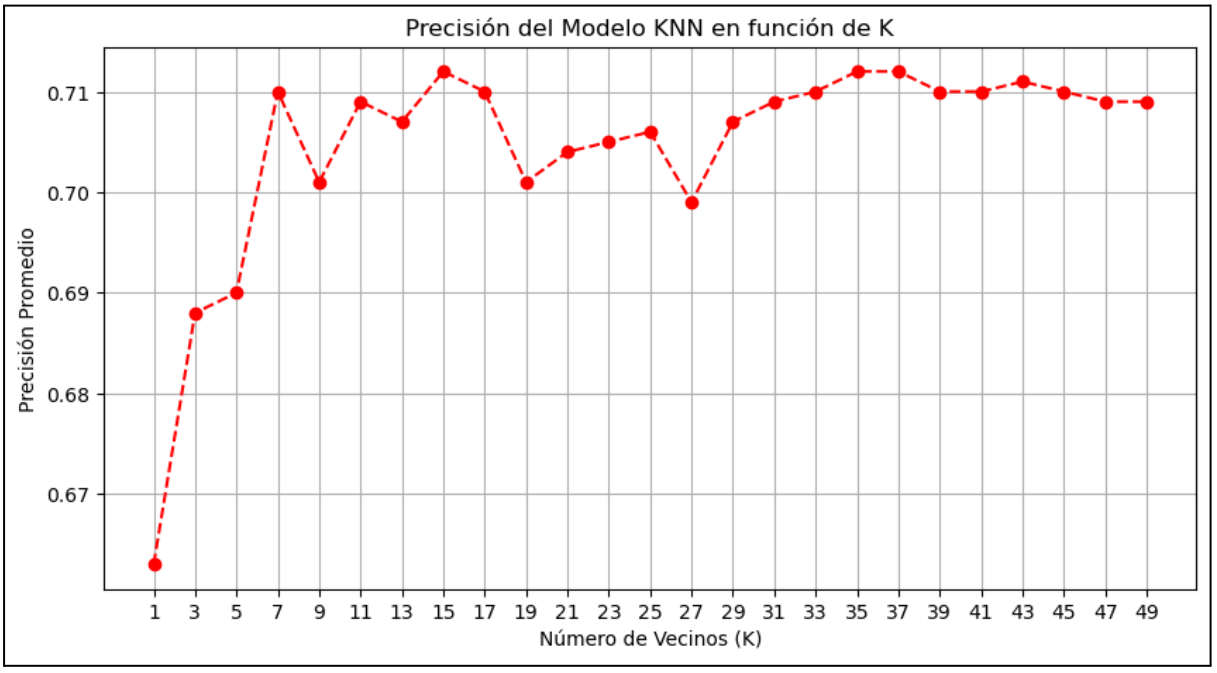
En cuanto a el número de épocas, se ha analizado con las configuración más óptimas encontradas (50, 10) que con un número de épocas mayor a 1000 los resultados no varían y con número menor a 1000 varían significativamente los resultados de los input 2 y 4.

Nº neuronas y capas	Score input 1	Score input 2	Score input 3	Score input 4
(50,)	0,95	0,99	0,98	0,99
(100,)	0,95	0,99	0,99	0,99
(1000,)	0,95	0,99	1,00	0,99
(50, 10)	0,95	0,99	1,00	0,99
(50, 100)	0,95	0,99	1,00	0,98
(50, 10, 20)	0,95	0,99	1,00	0,98

(50, 10, 50)	0,95	0,99	1,00	0,98

Tabla 3: puntuaciones para 4 entradas con número de neuronas y capas

Haz una gráfica que muestre la dependencia de la precisión de un clasificador de vecinos próximos con el número de vecinos. Si es más conveniente, utiliza gráficas en escala logarítmica para alguno de los ejes (semilogx, semilogy, loglog).



- **Utilizando los conceptos de sub- y sobreajuste:**
  - **Comenta los resultados cuando el número de vecinos es pequeño**

En la gráfica, podemos observar que la precisión aumenta rápidamente a medida que el número de vecinos crece desde 1, lo que indica que el modelo con K=1, que es un número de vecinos muy pequeño, está sobreajustado y es demasiado sensible al ruido en el conjunto de entrenamiento.

  - **Comenta los resultados cuando el número de vecinos es grande.**

Para un número de vecinos grande, se observa que la precisión disminuye gradualmente a medida que K se hace más grande, lo que sugiere que el modelo con un gran número de vecinos puede estar subajustado y no es capaz de capturar las fronteras de decisión entre las diferentes clases.

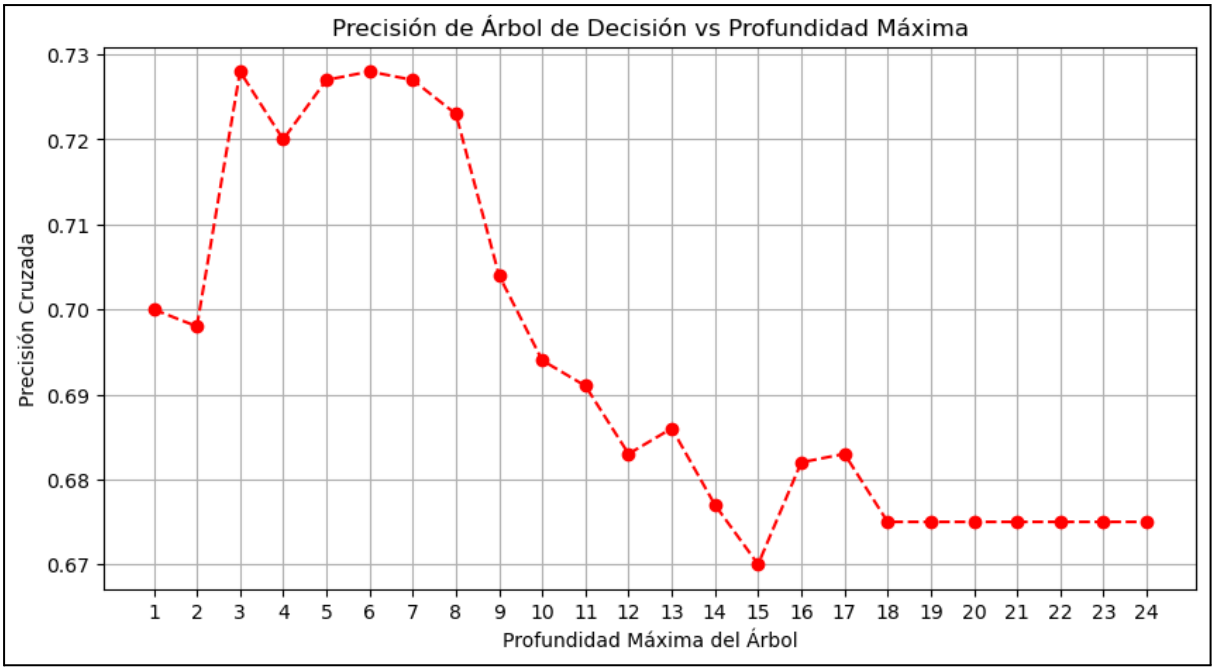
- **Explica el significado del valor de la precisión cuando el número de vecinos toma el valor mayor posible**

Si el número de vecinos es el máximo posible, la precisión tiende a bajar en relación al valor máximo porque el modelo es excesivamente generalizado, considerando demasiada información que no contribuye a decisiones precisas de clasificación.

¿Cuál es la mejor precisión que se alcanza con k-nn y para qué k (valor de n_neighbours)?

El número óptimo de vecinos (n_neighbours) que hemos encontrado es 15 con una precision: 0.712 es decir del 71% aproximadamente.

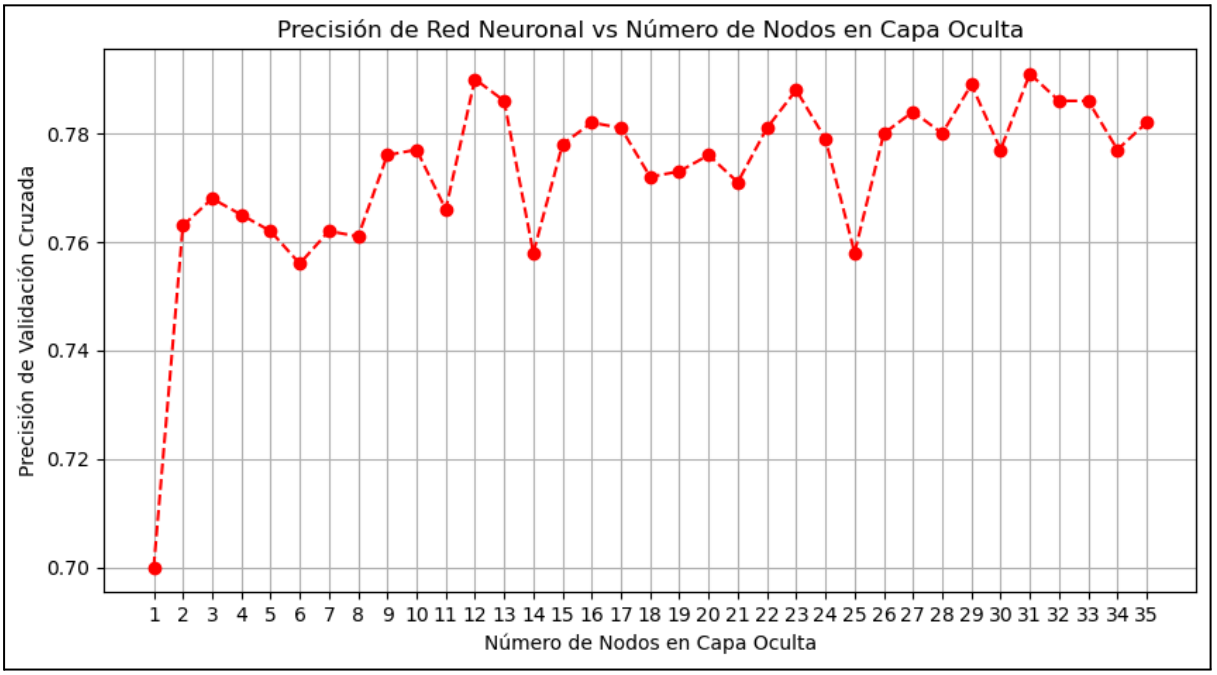
¿Cuál es la mejor precisión que se alcanza con un árbol de decisión y con qué profundidad máxima (valor de max_depth)? Para ello, haz una gráfica que muestre la dependencia de la precisión con la profundidad máxima del árbol. Comenta los resultados



La profundidad máxima con la que alcanza la mejor precisión es 3, con una precisión de 0,728 o 73% aproximadamente. Después de este punto de profundidad óptimo vemos una disminicuión sustancial de la precisión lo que sugiere que esta profundidad específica es la más adecuada para evitar subajuste o sobreajuste. Antes de eso en profundidades menores, el árbol es muy simple y no captura bien las relaciones (subajuste), mientras que en

profundidades mayores (posteriores a 17), los valores se estabilizan y el árbol parece volverse muy complejo y puede aprender demasiado del ruido de los datos (sobreajuste).

¿Cuál es la mejor precisión que se alcanza con una red neuronal con una sola capa oculta y con qué configuración (valor de `hidden_layer_sizes`)? Para ello, haz una gráfica que muestre la dependencia de la precisión con el número de nodos en la capa oculta. Comenta los resultados.



La mejor precisión la alcanzamos con 31 neuronas en la capa oculta, concretamente con una precisión de 0,790 aproximadamente un 80%. Se ha utilizado un máximo de 1000 épocas. La gráfica muestra fluctuaciones en la precisión de la red neuronal a medida que varía el número de nodos en la capa oculta, no se observa una tendencia clara que sugiera que más nodos siempre resulten en mayor precisión, lo cual puede indicar que tanto el subajuste como el sobreajuste pueden estar ocurriendo a lo largo del rango probado. Una conclusión que se puede extraer es que previamente a 29 neuronas los valores fluctúan más y tienden a ser menores, además de que claramente con pocas neuronas la precisión es considerablemente menor, lo cual sí que podría indicar un subajuste.

Resume los resultados y conclusiones del estudio realizado.

Según los resultados obtenidos el clasificador de redes neuronales es claramente el mejor dada su precisión, pero antes de llegar a unas conclusiones finales conviene como vamos a hacer en el siguiente apartado si manipular los datos antes de aplicar los distintos métodos de clasificación.

Procesamiento

El procesamiento de datos es una etapa crítica en cualquier tarea de aprendizaje automático y puede tener un impacto sustancial en el rendimiento del modelo. Hemos considerado varias posibilidades de procesamiento de datos:

Construcción de atributos y eliminación de atributos: La construcción de nuevos atributos puede revelar relaciones que no eran evidentes con los atributos originales. Esto puede resultar de enorme utilidad ya que si tenemos dos atributos cuya relación tiene una fuerte correlación con lo que queremos predecir nuestros modelos serán más eficientes y serán entrenados con un menor número de atributos. Así mismo podemos eliminar atributos que no afecten a nuestro objetivo a predecir al no tener una relación. Esto no se ha implementado por falta de tiempo.

Detección de Outliers: Los valores atípicos pueden distorsionar los resultados y afectar negativamente a la precisión de los modelos, especialmente a aquellos sensibles a variaciones extremas de los datos. Esto es sencillo de implementar y puede resultarnos útil con lo cual se ha implementado.

Missing Values: La presencia de valores faltantes puede generar problemas, pero en nuestro caso no hay missing values, por lo tanto no se ha implementado.

Centrado y Escalado : Algunos algoritmos, como las redes neuronales, son sensibles a la escala de las características. El escalado nos asegura que todas las características contribuyan equitativamente al aprendizaje del modelo. Esto de extrema utilidad, por lo tanto hemos implementado un estandarizado y un escalado: el primero lo usaremos cuando los outliers no sean muy significativos ya que el estandarizado ajusta los datos para que tengan una media de cero (centrado) y la media es un valor muy sensible a los extremos; Por el contrario cuando

los outliers sean significativos, usaremos el escalado robusto que es menos sensible a los outliers.

En conclusión, si ejecutamos nuestros modelos de clasificación después de aplicar un procesamiento previo a los datos, es de esperar observar una mayor precisión en ellos.

Parte 3

Predicción de fugas de clientes en una compañía telefónica

Tenemos por objetivo el desarrollar un modelo predictivo para identificar los 100 clientes actuales de una compañía telefónica que tienen mayor probabilidad de abandonar la empresa, para ello hacemos uso de dos datasets. El primero, "fuga_clientes_empresa_telefonica_construccion.csv", tiene información de clientes pasados con múltiples características y una columna Churn Status que indica si abandonó (1) o no (0) la compañía. Por otra parte tenemos otro, "fuga_clientes_empresa_telefonica_explotación.csv" que contiene los mismos datos pero de clientes actuales y sin la columna Churn Status.

Para la creación del modelo se separan las características en "x" y la variable objetivo Churn Status en "y" en el set de construcción y se eliminan las columnas Customer ID y Churn Status de las características al no ser estas relevantes para la predicción.

Los 5 modelos propuestos que testeamos para elegir uno que tenga un rendimiento correcto con la predicción que queremos hacer son:

- Red Neuronal (MLPClassifier)
- K-NN (KNeighborsClassifier)
- Naive BAyes GAussiano (GaussianNB)
- Regresión Logística (LogisticRegression)
- Árbol de decisión (DecisionTreeClassifier)

Para evaluar los modelos, hemos utilizado la técnica de validación cruzada para evaluar su rendimiento general en el dataset de construcción. La validación cruzada fue crucial para comprender la efectividad y la estabilidad de cada modelo, ofreciendo un balance entre sesgo y varianza y ayudando a identificar si algún modelo estaba sobreajustado.

Para afinar los modelos y obtener el máximo rendimiento se utilizó GridSearchCV, lo cual fue clave para la elección de parámetros en la red neuronal, el algoritmo K-NN y el árbol de decisión.

Las puntuaciones resultantes fueron las siguientes:

Score global del modelo neural network: 0.58 +/- 0.05

Score global del modelo k-nn: 0.63 +/- 0.08

Score global del modelo gaussian: 0.62 +/- 0.02

Score global del modelo logistic regression: 0.65 +/- 0.04

Score global del modelo decision tree: 0.75 +/- 0.05

Siendo claramente decision tree el que mejor rendimiento genera basado en su puntuación media y la desviación estándar obtenida en la validación cruzada.

Una vez hecho esto se hacen con este modelo las predicciones sobre el dataset de explotación obteniendo las posibilidades de que cada cliente actual abandone la compañía.

Con esto se eligen a los 100 con mayor probabilidad de fuga y se genera un archivo csv (top_100_clientes_fuga.csv) que contiene los IDs de estos.