Programación de Inteligencia Artificial

Práctica 4

En esta práctica vamos a desarrollar una red multicapa que ayude a predecir si una solicitud de préstamo bancario se va a incumplir o no, es decir, si el cliente que la solicita va a poder afrontar el préstamo o no. La práctica constará de los siguientes ficheros:

- **modelo_train.py**: donde definiremos y entrenaremos al modelo según los apartados 2 a 4 (inclusive)
- **modelo_prueba.py**: donde probaremos el modelo seleccionado con nuevos datos, según lo que se indica en el punto 5

1. Analizando el dataset

Para el desarrollo de esta práctica se proporcionará un *dataset* en formato CSV que tiene las siguientes columnas:

- *LoanID*: identificador alfanumérico del préstamo (columna irrelevante para nuestro estudio)
- *Age*: edad del cliente solicitante
- *Income*: ingresos anuales del cliente solicitante
- *LoanAmount*: cantidad de dinero que se solicita
- *CreditScore*: puntuación crediticia del solicitante, que indica su idoneidad
- *MonthsEmployed*: número de meses que lleva trabajados el solicitante
- *NumCreditLines*: cuántos créditos simultáneos tiene el cliente actualmente
- *InterestRate*: tasa de interés del préstamo
- *LoanTerm*: período de tiempo (en meses) para solicitar el préstamo
- *DTIRatio*: relación entre la cantidad solicitada y los ingresos del solicitante. Cuanto más pequeño sea ese valor, más saludable se considera el préstamo a solicitar.
- *Education*: nivel de estudios del solicitante. Puede tomar los valores *PhD* (doctorado), *Master's*, *Bachelor's* (estudios universitarios) o *High School*
- *EmploymentType*: tipo de trabajo del solicitante: *Full-time*, *Part-time*, *Self-employed* (autónomo) o *Unemployed*
- *MaritalStatus*: estado civil del solicitante: *Single*, *Married* o *Divorced*
- *HasMortgage*: indica si el solicitante tiene actualmente alguna hipoteca
- *HasDependents*: indica si el solicitante tiene personas dependientes a su cargo (típicamente hijos menores).
- *LoanPurpose*: propósito del préstamo: *Home*, *Auto*, *Education*, *Business* u *Other*
- *HasCoSigner*: indica si el préstamo lo firma/solicita alguna otra persona junto con el solicitante (por ejemplo, su pareja)
- *Default*: indica si el préstamo se incumplirá (1) o se pagará correctamente (0). Ésta será nuestra columna objetivo del estudio.

2. Procesamiento inicial de datos (2 puntos)

Como primer paso vamos a cargar el CSV y realizar un procesamiento de los datos:

- Eliminar la columna *LoanID*
- Detectar y eliminar las filas que tengan algún valor nulo (si las hay)
- Codificar como *one hot* los valores de la columna *LoanPurpose*
- Codificar como *label encoding* los valores de las columnas *Education*, *EmploymentType* y *MaritalStatus*
- Codificar como 0/1 los valores de las columnas *HasMortgage*, *HasDependents* y *HasCoSigner*
- Procurar, por comodidad, que la columna objetivo *Default* sea la última del *dataframe*.
- Revisar que todas las columnas son de tipo numérico o booleano y, en caso contrario, convertir las que no lo sean

3. Definiendo un modelo inicial (2 puntos)

Ahora vamos a crear un modelo Keras/TensorFlow inicial para probar su funcionamiento. La red constará de:

- Una capa de entrada con tantas neuronas como columnas vayamos a tratar
- Dos capas ocultas de 4 neuronas cada una y activación ReLu.
- Una capa de salida con la configuración adecuada para el problema a resolver, en cuanto a número de neuronas y función de activación

Usaremos como función de coste la entropía cruzada binaria, y como optimizador SGD. Divide los datos en 80% para entrenamiento y 20% para test (configurando la aleatoriedad para que siempre se cojan los mismos datos). Lanza el entrenamiento durante 10 etapas y muestra después por pantalla la precisión final alcanzada (en entrenamiento y test).

4. Ajuste de hiperparámetros (4 puntos)

Vamos a tratar de probar distintas opciones sobre el modelo anterior para determinar cuál es la mejor de todas. Para ello puedes probar a variar algunas de estas cosas (a tu elección):

- Número de etapas de entrenamiento
- Número de capas intermedias
- Número de neuronas en las capas intermedias
- Elegir entre distintos optimizadores: SGD, Adam o RMSProp
- Elegir entre distintos tamaños de *batch*: 24, 32 o 48
- Indicar si escalar o no los datos de entrada. Usaríamos en este caso el `StandardScaler` de SciKit-Learn para todas las columnas de entrada
- Otras cosas que consideres relevantes (capas de *Dropout*, distintos *learning rates* para los optimizadores, etc)

Se pide definir un proceso automático para probar entre todas las configuraciones que se elijan, y se seleccione y guarde en fichero el mejor modelo. Si el modelo que se ha elegido incluye escalado, también se deben guardar los parámetros del escalador en otro fichero.

Se valorará con **1 punto** en este apartado el volcado a un fichero (de texto, por ejemplo) del resumen de cada configuración que se haya probado, **indicando también el modelo seleccionado**. Por ejemplo (se debe

respetar el siguiente formato en la medida de lo posible):

```
1. Modelo con 10 neuronas por capa, 2 capas, Dropout 0.1, optimizador SGD, batch 24,
- Coste final: 0.3854 (entrenamiento), 0.3901 (test)
- Precisión final: 0.8811 (entrenamiento), 0.8784 (test)

2. Modelo con 10 neuronas por capa, 3 capas, sin Dropout, optimizador Adam, batch 32,
- Coste final: 0.3255 (entrenamiento), 0.3248 (test)
- Precisión final: 0.8876 (entrenamiento), 0.8867 (test)

...
Se ha escogido el modelo 2.
```

5. Prueba del modelo seleccionado (2 puntos)

Una vez hayamos guardado el modelo, desde el fichero **modelo_prueba.py** lo cargaremos, y le pediremos al usuario los datos de entrada de un nuevo cliente que solicita el préstamo. Deberemos procesar esos datos adecuadamente (incluyendo escalado si nuestro modelo guardado lo tiene) y llamar al método `predict` del modelo recogiendo el resultado y mostrándolo con el formato apropiado por pantalla. Aquí vemos un ejemplo (que no tiene por qué dar la misma solución, pero sí se debe mostrar un formato de salida similar):

```
Introduce los datos del cliente solicitante:

- Edad: 35
- Sueldo anual: 40000
- Cantidad solicitada: 120000
- Puntuación crediticia: 450
- Número de meses empleado: 30
- Número de líneas de crédito abiertas: 3
- Tasa de interés: 12.15
- Período del préstamo en meses: 60
- Ratio DTI: 0.56
- Educación (0 = instituto, 1 = universidad,
  2 = master, 3 = doctorado): 1
- Tipo de empleo (0 = desempleado, 1 = autónomo,
  2 = tiempo parcial, 3 = tiempo completo): 1
- Estado civil (0 = divorciado, 1 = soltero, 2 = casado): 1
- Tiene hipoteca actualmente (0 = no, 1 = sí): 1
- Tiene personas dependientes a su cargo (0 = no, 1 = sí): 0
- Propósito del préstamo (0 = casa, 1 = estudios, 2 = coche,
  3 = negocios, 4 = otros)
- Tiene co-firmante (0 = no, 1 = sí): 0

Analizando solicitud...
El resultado final del análisis es de 0.687.
El cliente es probable que incumpla el préstamo.
```

Para entregar

Se debe entregar un fichero comprimido con:

- Los dos ficheros fuente
- El modelo guardado (incluyendo el fichero de escalado, si procede)
- El fichero de texto con el proceso de selección de modelo volcado

En el caso de que el fichero ocupe mucho tamaño se puede compartir a través de un enlace a Drive/Dropbox/OneDrive o similar.