Práctica 5
Prevención del sobreaprendizaje

TÉCNICAS ESTADÍSTICAS PARA EL APRENDIZAJE II

Máster Universitario en Estadística Computacional
y Ciencia de Datos para la Toma de Decisiones

Prevención del sobreaprendizaje

Como sabemos, se denomina **sobreaprendizaje** [*overfitting*] al efecto producido al sobreentrenar (o sobreajustar) un algoritmo de aprendizaje a unos ciertos datos para los que se conoce el resultado deseado.

Cuando un sistema se sobreentrena el algoritmo puede quedar ajustado a unas características muy específicas de los datos de entrenamiento que no tienen relación causal con la función objetivo y **el algoritmo no generaliza**.

En esta práctica veremos cómo aplicar diferentes **técnicas de regularización con Keras,** que nos ayudarán a reducir el error de generalización de nuestros modelos.

Prevención del sobreaprendizaje

En esta práctica también vamos a ver cómo guardar un modelo durante el entrenamiento a través de **puntos de control**.

El *callback ModelCheckpoint, que* se utiliza junto con el entrenamiento mediante la función *fit(),* permite definir dónde guardar el modelo, cómo se debe nombrar el archivo y bajo qué circunstancias hacer un punto de control del modelo.

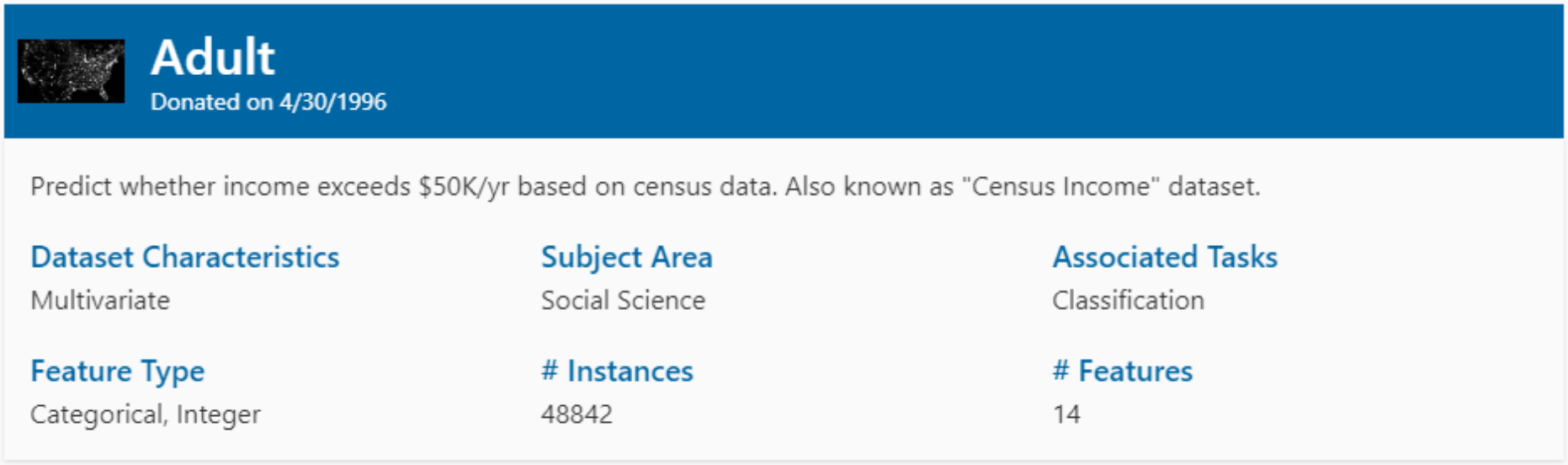
https://keras.io/api/callbacks/model_checkpoint/

Nosotros vamos a configurar el punto de control para guardar el modelo al final de cada época (opción por defecto en *ModelCheckpoint*) siempre que haya una mejoría en el *accuracy* del conjunto de validación.

Prevención del sobreaprendizaje

En esta práctica trabajaremos con la **base de datos "Adult"** del repositorio **UCI Machine Learning**, que incluye un censo de adultos realizado en 1994, y cuya tarea a resolver es determinar si una persona gana mas de 50000$ al año o no.

https://archive.ics.uci.edu/dataset/2/adult



Prevención del sobreaprendizaje

Descripción de la base de datos "Adult"

age: variable numérica
workclass: variable categórica, 7 categorías
fnlwgt: variable numérica
education: variable categórica, 16 categorías
education-num: variable numérica
marital-status: variable categórica, 7 categorías
occupation: variable categórica, 14 categorías
relationship: variable categórica, 6 categorías
race: variable categórica, 5 categorías
sex: variable categórica, 2 categorías
capital-gain: variable numérica
capital-loss: variable numérica
hours-per-week: variable numérica
native-country: variable categórica, 41 categorías
income: variable categórica, 2 categorías: >50K, <=50 → **clase a predecir**

Prevención del sobreaprendizaje

En una base de datos como esta debemos prestar especial atención al **preprocesamiento de datos**.

Tenemos **datos faltantes** y hay numerosas **variables categóricas** que debemos tratar.

También podemos ver que algunas de las variables categóricas toman **muchos valores**. Por ejemplo, *native-country* tiene 41 posibles valores. En el preprocesamiento de datos podríamos decidir, por ejemplo, agrupar por continentes para simplificar el análisis.

Además de todo lo anterior, como siempre, tendremos que **dividir en train/test**, **normalizar** las variables numéricas …

También debemos prestar atención a no tratar como numérica una variable categórica que este codificada, por ejemplo, con enteros.

Prevención del sobreaprendizaje

Regularización L1 y L2: con el **parámetro** *kernel_regularizer* podemos realizar una regularización L1, L2 o una combinación de ambas:

L1: *regularizer.L1(l1=0.01)*

$$L = L_E + \lambda \sum |w_k|$$

L2: *regularizer.L2(l2=0.01)*

$$L = L_E + \frac{1}{2} \lambda \sum w_k^2$$

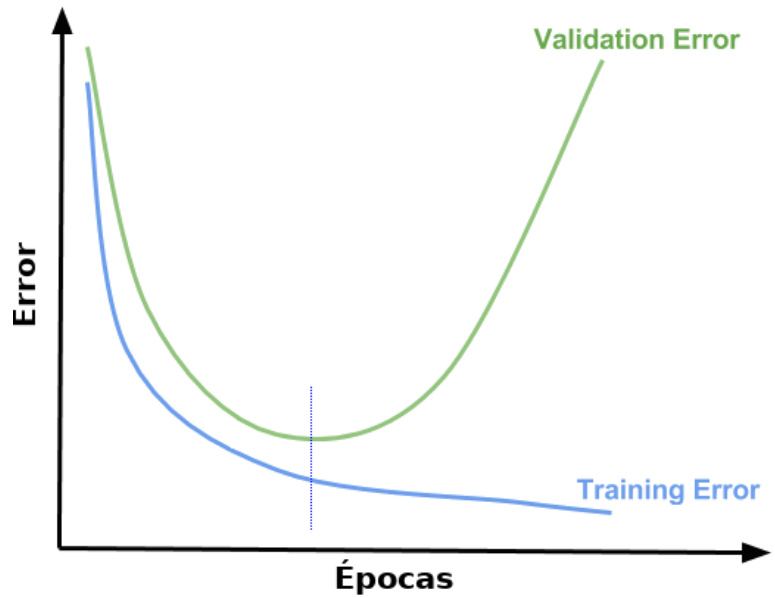
L1-L2: *regularizer.L1L2(l1=0.0, l2=0.0)*

$$L = L_E + \lambda_1 \sum |w_k| + \frac{1}{2} \lambda_2 \sum w_k^2$$

Prevención del sobreaprendizaje

Early stopping:

Detener el entrenamiento antes de que suceda el *overfitting* (en la línea de puntos de la gráfica).



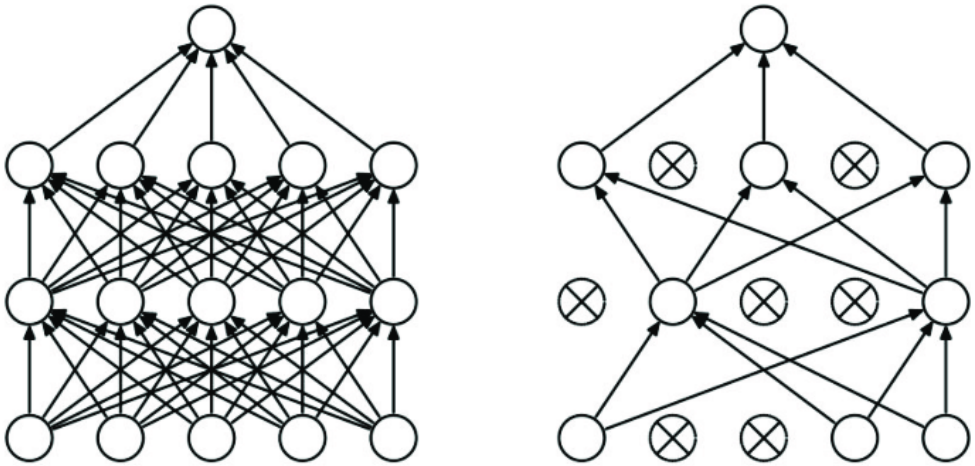
Con *callbacks.EarlyStopping* podemos hacer que la red deje de entrenar cuando una cantidad monitoreada haya dejado de mejorar.

Además de la métrica a monitorear, debemos indicarle nuestra "paciencia", es decir, el número de épocas sin mejora después de las cuales se detendrá el entrenamiento.

Prevención del sobreaprendizaje

Dropout[1]: consiste en desconectar aleatoriamente un porcentaje de las neuronas de la red en cada iteración del entrenamiento.

El efecto es que **la red se vuelve menos sensible a los pesos específicos de las neuronas**. Esto, a su vez, da como resultado una red que es capaz de una mejor generalización y es menos probable que se adapte demasiado a los datos de entrenamiento.



[1]Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, y Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. Journal of Machine Learning Research, 15:1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html

Prevención del sobreaprendizaje

El documento original sobre *dropout* proporciona algunas **heurísticas útiles** en su apéndice:

- **Utilizar una red más grande**. Es de esperar que la eliminación de unidades reduzca la capacidad de una red neuronal, por lo que es probable obtener un mejor rendimiento cuando se utiliza *dropuot* en una red más grande, lo que le da al modelo una mayor oportunidad para aprender representaciones independientes.

- **Utilizar una gran tasa de aprendizaje y un gran momentum** aceleran significativamente el aprendizaje. Aumentar la tasa de aprendizaje en un factor de 10 a 100 y utilizar un momentum alto de 0.95 o 0.99.

- **Restringir el tamaño de los pesos de la red**. Una gran tasa de aprendizaje puede resultar en pesos muy grandes. Para evitarlo, se puede usar una restricción en el tamaño de los pesos de la red, como la regularización max-norm $||w||_2 < k$, con un valor de k de 3 ó 4.

- **Utilizar un valor de *dropuot* del 20% al 50% de neuronas**. Una probabilidad demasiado baja tiene un efecto mínimo y un valor demasiado alto da como resultado un bajo aprendizaje por parte de la red.

Prevención del sobreaprendizaje

Dropout puede aplicarse en la capa de entrada y en las capas ocultas de la red, añadiendo capas *Dropout()*.

https://keras.io/api/layers/regularization_layers/dropout/

Para aplicarlo a las neuronas de entrada, añadimos una capa de *dropout* entre la entrada y la primera capa oculta. Por ejemplo, si la tasa de *dropuot* se establece en 0.2, una de cada cinco entradas se excluirá al azar en cada iteración del entrenamiento.

Además, como se recomienda en el documento original sobre *dropout*, podemos imponer una restricción en los pesos para cada capa oculta, lo que garantiza que la norma de los pesos no exceda un valor de, por ejemplo, 3 (**regularización max-norm**).

Esto se hace con ***kernel_constraint=max_norm(3)***

Prevención del sobreaprendizaje

Ejercicio 1.

Combina alguna de las técnicas de regularización vistas en esta práctica, por ejemplo, *dropout* con *early stopping*, y comprueba si el comportamiento del algoritmo de entrenamiento mejora.

Ejercicio 2.

Aplica alguna de las técnicas de regularización vistas en esta práctica a los ejercicios de prácticas anteriores, y comprueba si el comportamiento del algoritmo de entrenamiento mejora.