KEEPCODING
Tech School

Propuesta de guión para la práctica

1. **Descarga y preprocesado de los datos**
   a. Descargar los datos, almacenar las imágenes y eliminar filas del CSV para los cuales no se haya descargado la imagen – Sesión 1.
   b. Split de datos en *train/val/test*.
   c. Normalización* y redimensionado de datos numéricos y categóricos – Módulo ML, Sesión 1 (nota 1).
   d. Normalización* y redimensionado de imágenes – Sesión 1.
   e. Empleo de exactamente el mismo split *train/val/test* que para los datos tabulares anteriores.

2. **Modelado**
   a. Para los datos tabulares, preferiblemente desarrollar de cero una red neuronal de capas *Dense*. Factible porque el número de *features* de entrada (número de variables seleccionadas) es constante, ergo conocemos siempre el tamaño del vector de entrada y éste es fijo – Sesión 3.
   b. Para las imágenes, diseñar y/o elegir una arquitectura de red neuronal (Dense, CNN, RNN, *Transformer*…). Preferible el empleo de modelos pre-entrenados, justificando la relación entre el pre-entrenamiento (qué datos vio el modelo antes, posible impacto de cara a nuestra tarea…) y su elección – Sesiones 4-7.
   c. Elección para cada modelo de una función de pérdidas acorde con el tipo de tarea (regresión/clasificación) – Sesión 3.
   d. Entrenamiento de los modelos: *from scratch* (entrenamiento desde cero) / *transfer-learning* / *fine-tuning* – Sesión 6.**
   e. Optimización de hiperparámetros, prevención de posible *overfit* – Sesiones 3, 5, 6.
   f. Post-procesado de las predicciones. Si elegimos hacer regresión y hemos transformado nuestras etiquetas al rango [0, 1], devolverlas (así como nuestras predicciones) a valores “reales”.

3. **Fusión de modelos**
   a. *Early-fusion*: Extraer *features* de las imágenes (usar partes de una red entrenada como *feature extractor* – Sesión 5) y concatenar ese vector con los datos numéricos. Emplear un modelo que vea ese súper-vector como *input* para realizar la predicción. Se denomina *early-fusion* porque la decisión de regresión/clasificación se realiza a partir de una combinación temprana de las *features*.
   b. *Late-fusion*: Operar normalmente con dos modelos (datos tabulares-numéricos, imágenes) y emitir predicciones independientes. Emplear las predicciones como inputs para un modelo que vea las 2 predicciones como input para el clasificador/regresión. Ejemplo: Para la muestra *x*, el uso de datos tabulares predice 0.7; usando imágenes, 0.84. El “nuevo” *input* será el vector [0.7, 0.84].
   c. Para la fusión de modelos podrá emplearse cualquier algoritmo de clasificación/regresión visto (*SVM, RandomForest*, Redes neuronales…).

* Preprocesado y normalización deben realizarse atendiendo a los datos de train. No podemos usar los datos de test para “preprocesar” la información, sino que a test aplicamos la transformación aprendida con train.
** Se valorará muy positivamente la obtención de un “resultado baseline” frente al que comparar sucesivas mejoras, entendiendo el flujo de trabajo en Deep Learning (confirmar aprendizaje básico, incluir mejoras de manera iterativa sobre la base de datos validation).

- Nota 1:

1. Seleccionar variables de interés de entre:

```
'Property Type', 'Room Type', 'Cancellation Policy', 'Accommodates', 'Bathrooms', 'Bedrooms',
'Beds','Guests Included', 'Extra People', 'Minimum Nights', 'Maximum Nights', 'Number of
Reviews', 'Host Total Listings Count'
```

2. Detección y procesado de missing values (NaN)
3. Procesado de las etiquetas: Si optamos por regresión, MinMaxScaler (facilitar regresión al rango [0, 1]). Si elegimos clasificación, definir cómo dividimos las muestras en [0, 1, 2, …, N] etiquetas representadas en números enteros.

* Preprocesado y normalización deben realizarse atendiendo a los datos de train. No podemos usar los datos de test para “preprocesar” la información, sino que a test aplicamos la transformación aprendida con train.
** Se valorará muy positivamente la obtención de un “resultado baseline” frente al que comparar sucesivas mejoras, entendiendo el flujo de trabajo en Deep Learning (confirmar aprendizaje básico, incluir mejoras de manera iterativa sobre la base de datos validation).