Preparación de datos

El primer paso en la construcción de un modelo de aprendizaje automático es recolectar datos, el segundo paso es prepararlos y limpiarlos. En la fase de preparación de datos se busca escalarlos, normalizarlos, procesar errores, identificar valores atípicos y seleccionar características relevantes para modelar la variable dependiente. Una correcta preparación de los datos maximiza el aprendizaje del modelo, por ello es la actividad que más tiempo invierten los científicos de datos.

La preparación de datos consta en 5 etapas:

Etapa 1: Análisis exploratorio de datos (AED)

La identificación de características relevantes, valores atípicos y errores se hace a través de métodos visuales, un AED se puede hacer mediante:

- Diagramas de cajas
- Histogramas
- Diagramas de dispersión (Scatter plots)
- Gráficos circulares (pie chart)

En Python existen varias librerías para la visualización de datos, por ejemplo, matplotlib. Revisen la documentación de la librería en https://matplotlib.org/ y sigan los siguientes tutoriales para aprender a usarla:

https://www.youtube.com/playlist?list=PLeo1K3hjS3uu4Lr8_kro2AqaO6CFYgKOl

Un acordeón útil para recordar las funcionalidades en matplotlib es:

https://s3.amazonaws.com/assets.datacamp.com/blog_assets/Python_Matplotlib_Cheat_Sheet.pdf

Opcionalmente, para ver el potencial de matplotlib pueden revisar el Jupyter *notebook* del libro *Introduction to scientific computation in Python* de Robert Johansson:

https://github.com/jrjohansson/scientific-python-lectures/blob/master/Lecture-4-Matplotlib.ipynb

Etapa 2: Mejorar la calidad de los datos

En la fase de recolección es posible capturar errores o ruidos, por ejemplo, un sensor registrando un valor incorrecto al tiempo de hacer una medición. Otro problema recurrente en el aprendizaje

automático son los datos incompletos, por ejemplo, un campo vacío en la entrada de una base de datos. Mejorar la calidad de los datos implica detectar sus impropiedades para arreglarlas antes de que sean introducidas al modelo. Efectúen el siguiente tutorial para lidiar con valores incompletos en una base de datos:

https://www.kaggle.com/rtatman/data-cleaning-challenge-handling-missing-values

Otra forma de mejorar la calidad de los datos es a partir de optimizaciones orientadas al hardware en el que se ejecutará el código. Un ejemplo de lo anterior podría ser utilizar técnicas de redondeo para reducir la cantidad de cifras significativas de los datos, con lo que se disminuye el número de bits requeridos para hacer las representaciones numéricas y habilita el uso de tipos de datos que ocupen menos memoria. Este tipo de optimizaciones son relevantes al implementar modelos en sistemas con recursos limitados como lo podrían ser sistemas embebidos o microcontroladores.

Etapa 3: Consistencia en los datos

Los atributos recolectados pueden estar en magnitudes diferentes, por ejemplo, si los valores de una columna estuvieran en el orden de decenas mientras que los de otra columna estuvieran en el orden de trillones. La desproporción en la escala de los atributos de entrada podría ocasionar problemas al combinar los valores como atributos del modelo. La técnica de normalización transforma los valores de las columnas de entrada a una escala en común sin distorsionar las diferencias en los intervalos de valores. Inspeccionen el método de MinMaxScaler de la librería scikit-learn para visualizar un método de normalización:

https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html

La consistencia de datos también alude a que los atributos deben estar en el mismo sistema de unidades (e.g. sistema internacional de unidades) y en un mismo formato (e.g. registrar fechas como DD/MM/AA). Adicionalmente, los métodos en Python requieren que sus argumentos sean de un tipo de dato en específico, por lo que en esta etapa también se hace la conversión de tipos de datos (e.g. transformar una lista a un ndarray).

Etapa 4: Ingeniería de atributos

En palabras de uno de los máximos referentes en educación de inteligencia artificial, Andrew Ng, "El proponer características es complicado, consume tiempo y requiere un conocimiento profundo del fenómeno a modelar. El aprendizaje automático es, en esencia, ingeniería de atributos." La ingeniería de atributos (*feature engineering)* consta en extraer características útiles

de los datos en bruto para entrenar un modelo. La base de *feature engineering* recae significativamente en la pericia de la científica de datos para saber cómo extraer la mayor cantidad de información de los datos. Basándose en su conocimiento del área, una científica puede combinar atributos para capturar interacciones de las variables independientes y así generar un mejor modelo.

Estapa 5: Dividir los datos en entrenamiento y en evaluación

Los datos que ya estén preparados y limpios se deben dividir en dos bloques, uno de entrenamiento y otro de evaluación. Tradicionalmente se hace una división de 80% para el de entrenamiento y 20% para el de evaluación. Es conveniente que no exista traslapo en los datos que se seleccionen, los datos que se escogieron para el de entrenamiento no deben repetirse en el de validación. Revisen el método de train_test_split de la librería sklean y reflexionen por qué es necesario dividir el set de datos en entrenamiento y validación.

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html