Librerías en Python para ciencia de datos.

Ejercicio U10\_E02

MasterD

# Visualización y ajuste de modelos

Ahora vamos a comprobar qué tal te manejas generando algunas gráficas sencillas y construyendo algunos modelos de previsión en base a los hallazgos explorando los datos.

De nuevo, empezaremos leyendo datos de uno de los ficheros incluidos en el directorio de ejercicios de esta unidad. Antes que nada, conviene que cambiemos el directorio de trabajo de nuestra sesión en Python con la función os.chdir().

Ajusta la ruta del directorio a la que estés utilizando en tu ordenador:

**import** **os  
import** **numpy** **as** **np  
import** **pandas** **as** **pd**  
os.chdir("RUTA\_DIRECTORIO\_CUSO/10PythonLibreriasCienciaDatos/Ejercicios")

Ahora ya puedes leer el fichero Auto.csv Recuerda que Pandas ya incluye funciones para leer distintos tipos de fichero, y presta atención a los caracteres delimitadores y a los encabezados.

auto = ...

Comprobemos que los datos se han cargado bien

auto.head()

|  | **mpg** | **cylinders** | **displacement** | **horsepower** | **weight** | **acceleration** | **year** | **origin** | **name** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 18.0 | 8 | 307.0 | 130.0 | 3504 | 12.0 | 70 | 1 | chevrolet chevelle malibu |
| **1** | 15.0 | 8 | 350.0 | 165.0 | 3693 | 11.5 | 70 | 1 | buick skylark 320 |
| **2** | 18.0 | 8 | 318.0 | 150.0 | 3436 | 11.0 | 70 | 1 | plymouth satellite |
| **3** | 16.0 | 8 | 304.0 | 150.0 | 3433 | 12.0 | 70 | 1 | amc rebel sst |
| **4** | 17.0 | 8 | 302.0 | 140.0 | 3449 | 10.5 | 70 | 1 | ford torino |

## Visualización de datos

Vamos a comenzar con algunas gráficas para ver la distribución de las distintas variables que componen nuestro conjunto de datos, utilizando la librería seaborn.

**import** **seaborn** **as** **sns**

*# Si trabajas con Jupyter, recuerda incluir la siguiente directiva*

**%matplotlib** **inline**

Pinta un gráfico con la distribución de la variable "mpg" (el consumo, *miles per gallon*)

Ahora muestra la distribución de la variable weight (el peso de cada coche).

Y ahora con la variable 'horsepower' (la potencia en cv).

¿Qué ocurre al intentar pintar esta última variable? ¿Obtienes un error?

Los datos originales incluyen valores ausentes (NaN) y la librería no sabe cómo manejarlos al tratar de calcular y pintar el histograma.

Pero podemos utilizar el método dropna() para descartar todos los valores nulos antes de pintar la gráfica. Trata de dibujar la misma gráfica usando como datos el resultado de auto.dropna()

Muy bien. Ahora vamos a ver cómo influyen estas variables en el consumo.

Empieza pintando un gráfico de dispersión con la relación entre potencia y consumo.

Y ahora pinta la misma gráfica, pero con la distribución conjunta de peso y consumo.

¿Dirías que existe relación relevante con el consumo de combustible? ¿Dirías que es lineal?

## Ajustando modelos.

Tras explorar gráficamente los datos, vamos a intentar construir algunos modelos que nos permitan dar una estimación del consumo de un vehículo a partir de sus características.

Utilizaremos la librería statsmodels que hemos visto en la unidad 10.

**import** **statsmodels.api** **as** **sm  
import** **statsmodels.formula.api** **as** **smf**

Comienza creando un modelo de regresión lineal simple para predecir la variable consumo ('mpg') a partir de la variable peso ('weight'). Ajusta el modelo y muestra la información de resumen del ajuste. Puedes definir el modelo usando una fórmula.

model\_fn = ...

*# Ajustamos los parámetros del modelo (b0, b1)*

mfitted = ...

*# Veamos el resumen del modelo ajustado*

print(mfitted.summary())

¿Cuál es el coeficiente para el peso? ¿Y su intervalo de confianza?

¿Existe algún posible problema con el modelo y su ajuste?

Prueba ahora el modelo, prediciendo el valor con una serie de pesos (2000, 3000 y 6000). Pista en caso de errores: ¿es necesario añadir el término constante?

test\_auto = ...  
*# predice el consumo para los nuevos datos*

...

¿Qué te parece el consumo predicho para un peso de 6000?

En nuestra exploración gráfica anterior, ya pudimos distinguir una relación no lineal con el consumo.

Ajusta un nuevo modelo de regresión, pero esta vez aplicando una transformación logarítmica a la variable peso. Revisa la información de resumen del modelo ajustado.

model\_fn = …

*# Ajustamos los parámetros del modelo (b0, b1)*

mfitted = ...

*# Veamos el resumen del modelo ajustado*

print(mfitted.summary())

¿Dirías que este modelo es mejor o peor que el anterior?

Vuelve a predecir el consumo con este nuevo modelo para la serie de pesos (2000, 3000, 6000).

test\_auto = ...

*# predice el consumo para los nuevos datos*

...

¿Te parece más o menos razonable el consumo para un peso de 6000?

Para terminar, ajusta un nuevo modelo de regresión múltiple. Esta vez utilizando tanto el peso como la potencia como regresores. Aplica la transformación logarítmica a ambos términos. Revisa el resumen de resultados tras el ajuste.

model\_fn = ...

*# Ajustamos los parámetros del modelo (b0, b1)*

mfitted = ...

*# Veamos el resumen del modelo ajustado*

print(mfitted.summary())

¿Mejora este modelo a los anteriores?