# Prueba de consolidación I

Entornos de data science con Python

## Parte 1

En esta primera parte trabajaremos con ficheros de la iniciativa de datos abiertos del Gobierno británico. Pueden consultarse aquí: <https://www.data.gov.uk/dataset/broadband-coverage>

Si descargamos el fichero etiquetado como csv/zip denominado "Broadband Coverage and speeds (postcode level data)" y lo descomprimimos tendremos el dataset en dos fragmentos y la descripción del mismo.

### 1.1. Lectura de datos

* Carga los datos como un DataFrame fusionando los dos ficheros en uno solo.
* Renombra la primera columna para hacerla más corta (investiga alguna función para renombrar en pandas).

### 1.2. Limpieza de datos

* Observa los valores faltantes (¿en qué columnas están?), cuantifícalos (¿cuántos hay en cada columna?). Pista: la función np.unique() puede darnos ayuda en esto.
* Cuenta cuántas filas quitamos si eliminamos todas las filas con algún valor faltante, o con al menos tres valores faltantes.

### 1.3. Procesamiento

* Extrae las variables que refieren a la velocidad media, mediana y máxima y el código postal a una matriz de NumPy.

El siguiente código permite convertir una de las columnas a numérico (suponemos que m es la matriz del punto anterior):

In [20]:

def isFloat(string):

try:

float(string)

return True

except ValueError:

return False

L = []

for i in np.asarray(m[:, 1]):

if isFloat(i):

L.append(float(i)) L[:10]

Out[20]:

[20.0, 23.3, 17.9, 13.1, 19.6, 21.4, 18.0, 18.5, 19.8, 17.0]

* Basándote en la idea del código anterior, calcula la diferencia en valor absoluto entre la media de la velocidad y el valor 3.0 por cada código postal, utilizando arrays de NumPy. Nota: No todos los valores son convertibles, por lo que no se aplicará a todas las filas.

## Y dibuja un boxplot de la velocidad media para los datos que pueden convertirse a float con la función isFloat (ignora el resto), previamente tomando los primeros 3000 valores y partiendo el array en tres trozos, formando una matriz con tres subseries de 1000 datos cada una.

## Parte 2

En esta segunda parte trabajaremos con un dataset de R que describe las características físicas de un conjunto de diamantes. Puede obtenerse de Internet.

* Calcula los valores máximo, mínimo y medio de la variable carat.
* Dibuja su histograma y sobre el mismo gráfico, el histograma de una muestra aleatoria del mismo tamaño con puntos obtenidos de una distribución gamma con la misma media y desviación típica. Haz que la visualización sea más adecuada haciendo transparente uno de los histogramas utilizando el parámetro *alpha*.
* Utilizando matplotlib, dibuja la nube de puntos de carat contra el precio, con ambos ejes en escala logarítmica.
* Ahora repite el gráfico pero utiliza solo los datos cuyo color sea "E" y su claridad sea "SI1"
* Investiga con un gráfico la relación entre el volumen (que puedes aproximar a partir de x, y, z) y el precio, de nuevo en escala logarítmica.
* Utilizando scipy.stats, comprueba si se la relación anterior ajusta bien a un modelo de regresión lineal, tomando solo los 1000 primeros valores del dataset.
* Utilizando seaborn, representa la nube de puntos del precio contra el carat y asociando al diagrama las distribuciones de las dos variables.
* Identifica con un comando los diferentes tipos de corte (cut).
* Comprueba si hay diferencias significativas en la media del precio para los de corte "Ideal" y "Premium" y el resto, tomando 1000 valores de cada (nótese que no se puede asumir que las muestras son de una distribución normal). Puedes investigar funciones lógicas vectorizadas en NumPy como logical\_or().
* Obtén la mediana del precio por cada valor de la claridad (clarity) y ordena los resultados de manera descendente.

## Parte 3

Podemos descargar datos de empresas del dataset Forbes2000 (paquete HSAUR) mediante get\_rdataset(). Encontraremos un ranking de empresas categorizadas por sector, y con datos de su país, ventas, beneficios, recursos y valor de mercado.

* Seleccionar del dataframe los nombres de empresa que incluyen "elect" (sin tener en cuenta mayúsculas). Intentamos con esto obtener empresas “del sector de la electricidad” de forma aproximada.
* Utilizando como índice el país y la categoría y ordenándolo (sort\_index), encontrar el valor de mercado de las empresas japonesas y estadounidenses.
* Sobre el mismo subconjunto de empresas eléctricas, obtener el país donde el sector de utilities tiene mayor valor de mercado medio en sus empresas.
* Volviendo al dataset original, obtener las empresas españolas.
* Indexar el dataframe anterior por beneficios (profits) y obtener los nombres y rangos de las que tengan un beneficio nulo o negativo. Nota: intenta indexar de manera mixta con números (filas) y etiquetas columnas), hay una operación específica para esto en pandas.