Anexo I

September 24, 2023

0.1 Importar pandas

[4]:
```
import pandas as pd
```

0.2 Cargar el dataset stroke

[5]:
```
df = pd.read_csv('/home/guincho/CEMP/stroke.csv')
```

0.3 Entendiendo los datos. "Data understanding"

- Estructura del dataset
- head & columns
- dtypes
- describe

[6]:
```
df.shape
```

[6]:
```
(5110, 12)
```

El primer valor corresponde al numero de filas en el conjunto de datos. Es el numero de observaciones o muestras El segundo valor corresponde al numero de columnas, tambien conocidas como caracteristicas variables o atributos

[7]:
```
df.head()
```

[7]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural	202.21	NaN	never smoked
2	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked
3	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked

```
   stroke
0       1
1       1
2       1
3       1
4       1
```

[8]:
```
print(df.columns)

print(df["hypertension"].unique())

print(df["heart_disease"].unique())

print(df["ever_married"].unique())

print(df["work_type"].unique())

print(df["Residence_type"].unique())

print(df["smoking_status"].unique())

print(df["stroke"].unique())
```

```
Index(['id', 'gender', 'age', 'hypertension', 'heart_disease', 'ever_married',
       'work_type', 'Residence_type', 'avg_glucose_level', 'bmi',
       'smoking_status', 'stroke'],
      dtype='object')
[0 1]
[1 0]
['Yes' 'No']
['Private' 'Self-employed' 'Govt_job' 'children' 'Never_worked']
['Urban' 'Rural']
['formerly smoked' 'never smoked' 'smokes' 'Unknown']
[1 0]
```

Esto nos devuelve una lista con el nombre de las columnas, así obtenemos una información más amplia. Podemos ver si hay alguna columna que a priori podamos eliminar.

De momento no eliminamos ninguna, pero si haremos una descripcion de que es cada una de ellas. Ademas hemos visto con unique si podría haber errores de mayúsculas o fallos de escritura que pudieran hacernos ver como diferentes observaciones iguales. Como “yes” /= “Yes”

- id - Numero de identidad
- gender - genero
- age - edad
- hypertension - 1 es paciente hipertenso/ 0 no hipertenso
- heart_disease - 1 es paciente con enfermedad coronaria/ 0 sin enfermedad
- ever_married - “Yes” si el paciente ha estado casado o está/ “No” si no.

- work_type - Tipo de trabajo
- ibm - Indice de masa corporal
- Residence_type - Si trabaja en el campo o en ciudad
- smoking_status - Fumador, exfumador, nunca fumó o no se sabe
- stroke - Si ha sufrido un accidente cerebrovascular

[9]:
```
df.dtypes
```

[9]:
```
id                   int64
gender              object
age                float64
hypertension         int64
heart_disease        int64
ever_married        object
work_type           object
Residence_type      object
avg_glucose_level  float64
bmi                float64
smoking_status      object
stroke               int64
dtype: object
```

Podemos ver que tipo tiene cada una de las columnas

0.4 Preparando los datos. "Data preparation"

- Comprobar correccion de los datos
- Cambiar datos y tipos de datos
- Crear nuevas columnas

[10]:
```
df.rename(columns={'Residence_type' : 'residence_type'}, inplace = True)
```

Queremos cambiar el 'yes', 'no' de 'ever_married' a 1 y 0

[11]:
```
df['ever_married'] = df['ever_married'].map({'Yes':1, 'No':0})
```

Ahora veamos si hay ID repetidos

[12]:
```
id_repetidos = df['id'].duplicated()

valores_repetidos = df['id'][id_repetidos].unique()

print(valores_repetidos)
```

```
[]
```

No hay ninguno.

Ahora vamos a identificar los NaN que hay por columnas. Antes habíamos comprobado los diferentes resultados de cada columna y no hemos encontrado ningún valor que se hubiese podido

introducir en vez de NaN. Algún string del estilo “No encontrado”. Pero hay columnas en las que faltarán datos

[13]:
```
nan_counts = df.isna().sum()

print(nan_counts)
```

```
id                   0
gender               0
age                  0
hypertension         0
heart_disease        0
ever_married         0
work_type            0
residence_type       0
avg_glucose_level    0
bmi                201
smoking_status       0
stroke               0
dtype: int64
```

La única columna con NaN es bmi o indice de masa corporal. Podemos no eliminar estas filas. Lo que si que haremos será una nueva columna con la clasificación de imc, llamada imc_str + bmi $<$ 18.5 - Bajo peso + 18.5 $<$ bmi $<$ 24.9 - Saludable + 25 $<$ bmi $<$ 29.9 - Sobrepeso + 30 $<$ bmi $<$ 34.9 - Obesidad de grado I + 35 $<$ bmi $<$ 39.9 - Obesidad de grado II + bmi $>$ 40 - Obeso morbido o grado III

Este proceso recibe el nombre de binning

[14]:
```
intervalos = [float('-inf'), 18.5, 24.9,29.9, 34.9, 39.9, float('inf')]

etiquetas = ['Bajo peso', 'Peso saludable', 'Sobrepeso', 'Obeso g I', 'Obeso g␣
 ↪II', 'Obeso g III']

df['imc_str'] = pd.cut(df['bmi'], bins=intervalos, labels=etiquetas,␣
 ↪right=False)

df.head()
```

[14]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type	residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0	0	1	1	Private	Urban	228.69	36.6	formerly smoked	1
1	51676	Female	61.0	0	0	1	Self-employed	Rural	202.21	NaN	never smoked	1
2	31112	Male	80.0	0	1	1						
3	60182	Female	49.0	0	0	1						
4	1665	Female	79.0	1	0	1						

```
2        Private          Rural             105.92  32.5     never smoked
3        Private          Urban             171.23  34.4           smokes
4  Self-employed          Rural             174.12  24.0     never smoked

   stroke         imc_str
0       1      Obeso g II
1       1             NaN
2       1       Obeso g I
3       1       Obeso g I
4       1  Peso saludable
```

Por último vamos a ver si hay datos que no tengan sentido como un niño mayor de 18 años o un niño de menos de 16 trabajando para el gobierno. Primero vemos que considera exactamente como niños este dataset

[15]:
```
mayor_edad_children = df[df['work_type'] == 'children']['age'].max()
print(mayor_edad_children)


print(" \n RESULTADO 2 \n")

result2 = df[(df['work_type'] != 'children') & (df['age'] < 16)]

print(result2)

print(" \n RESULTADO 3 \n")

result3 = df[(df['work_type'] == 'children') & (df['age'] < 16)]

print(result3)

print(" \n RESULTADO 4 \n")

result4 = (df[(df['work_type'] != 'children') & (df['age'] < 16) &␣
 ↪(df['smoking_status']== 'formerly smoked')]).shape[0]

result5 = (df[(df['work_type'] != 'children') & (df['age'] < 16) &␣
 ↪(df['smoking_status']== 'smokes')]).shape[0]

result6 = (df[(df['work_type'] == 'children') & (df['age'] < 16) &␣
 ↪(df['smoking_status']== 'formerly smoked')]).shape[0]

result7 = (df[(df['work_type'] == 'children') & (df['age'] < 16) &␣
 ↪(df['smoking_status']== 'smokes')]).shape[0]

print("Hay {} niños que trabajaron y dejaron de fumar \n {} niños que␣
 ↪trabajaron y fuman hoy \n". format(result4, result5))
```

```
print("Hay {} niños que nunca trabajaron y dejaron de fumar \n{} niños que␣
 ↪nunca trabajaron y fuman".format(result6,result7))
```

```
16.0
```

```
 RESULTADO 2
```

```
         id  gender   age  hypertension  heart_disease  ever_married  \
251   16523  Female   8.0             0              0             0
253   46136    Male  14.0             0              0             0
284   26325    Male  14.0             0              0             0
410   54975    Male   7.0             0              0             0
455    7351    Male  13.0             0              0             0
…      …      …   …            …              …             …
4709  41930    Male  15.0             0              0             0
4806  69723    Male  15.0             0              0             0
4903  56629  Female  14.0             0              0             0
4923  72186  Female  15.0             0              0             0
4981  61801    Male  15.0             0              0             0

          work_type residence_type  avg_glucose_level   bmi smoking_status  \
251         Private          Urban             110.89  17.6        Unknown
253    Never_worked          Rural             161.28  19.1        Unknown
284        Govt_job          Urban              82.34  31.6        Unknown
410   Self-employed          Rural              64.06  18.9        Unknown
455         Private          Urban              92.14  23.2   never smoked
…             …             …                 …   …            …
4709        Private          Rural             144.15  24.1   never smoked
4806        Private          Urban             137.27  19.3   never smoked
4903        Private          Rural              83.56  33.1        Unknown
4923        Private          Rural              82.19  40.5   never smoked
4981        Private          Urban              65.05  24.6        Unknown

      stroke         imc_str
251        0       Bajo peso
253        0  Peso saludable
284        0       Obeso g I
410        0  Peso saludable
455        0  Peso saludable
…      …             …
4709       0  Peso saludable
4806       0  Peso saludable
4903       0       Obeso g I
4923       0     Obeso g III
4981       0  Peso saludable
```

```
[68 rows x 13 columns]
```

RESULTADO 3

```
         id  gender    age  hypertension  heart_disease  ever_married  \
162   69768  Female   1.32             0              0             0
245   49669  Female  14.00             0              0             0
249   30669    Male   3.00             0              0             0
282   33759  Female   3.00             0              0             0
290   55680    Male  13.00             0              0             0
…     …      …    …            …              …             …
5089  56714  Female   0.72             0              0             0
5094  28048    Male  13.00             0              0             0
5095  68598    Male   1.08             0              0             0
5098    579    Male   9.00             0              0             0
5104  14180  Female  13.00             0              0             0

     work_type residence_type  avg_glucose_level   bmi smoking_status  stroke  \
162   children          Urban              70.37   NaN        Unknown       1
245   children          Rural              57.93  30.9        Unknown       1
249   children          Rural              95.12  18.0        Unknown       0
282   children          Urban              73.74  16.0        Unknown       0
290   children          Urban             114.84  18.3        Unknown       0
…        …             …                 …   …           …      …
5089  children          Rural              62.13  16.8        Unknown       0
5094  children          Urban              82.38  24.3        Unknown       0
5095  children          Rural              79.15  17.4        Unknown       0
5098  children          Urban              71.88  17.5        Unknown       0
5104  children          Rural             103.08  18.6        Unknown       0

             imc_str
162              NaN
245        Obeso g I
249        Bajo peso
282        Bajo peso
290        Bajo peso
…              …
5089       Bajo peso
5094  Peso saludable
5095       Bajo peso
5098       Bajo peso
5104  Peso saludable

[676 rows x 13 columns]
```

RESULTADO 4

Hay 6 niños que trabajaron y dejaron de fumar

```
 1 niños que trabajaron y fuman hoy
```

```
Hay 13 niños que nunca trabajaron y dejaron de fumar
2 niños que nunca trabajaron y fuman
```

La mayor edad que considera para un niño son 16 años.

La edad minima para trabajar en la mayoria de paises desarrollados es de 16 años y no tenemos forma de saber exactamente de qué país son estos datos. Aun asi en algunos paises como Estados Unidos es legal trabajar a partir de 14 años con ciertas restricciones. Por ello vamos a cambiar el work_type de aquellos individuos menores de 14 años a 'children'. Cambiaremos a children los 'Never_worked' menores de 16, ya que definen lo mismo. Tambien cambiaremos los 'Govt_job' de menores de 16 a children.

Hay 22 niños que fumen o hayan fumado. Eso es verosímil

[16]:
```
df.loc[df['age'] <= 14, 'work_type'] = 'children'

condition = (df['age'] <= 16) & ((df['work_type'] == 'Never_worked') |␣
 ↪(df['work_type'] == 'Govt_job'))

df.loc[condition, 'work_type'] = 'children'
```

[17]:
```
result = df[(df['work_type'] != 'children') & (df['age'] <= 16)]
```

Ya tenemos los datos corregidos. Vamos a hacer por último un binning con diferentes edades.

[18]:
```
intervalos = [0, 18, 35, 50, 65, float('inf')]

etiquetas = ['0-17', '18-34', '35-49', '50-64', '65+']

df['grupo_edad'] = pd.cut(df['age'], bins = intervalos, labels = etiquetas,␣
 ↪right=False)
```

[19]:
```
df.head()
```

[19]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type	residence_type	avg_glucose_level	bmi	smoking_status
0	9046	Male	67.0	0	1	1	Private	Urban	228.69	36.6	formerly smoked
1	51676	Female	61.0	0	0	1	Self-employed	Rural	202.21	NaN	never smoked
2	31112	Male	80.0	0	1	1	Private	Rural	105.92	32.5	never smoked
3	60182	Female	49.0	0	0	1	Private	Urban	171.23	34.4	smokes
4	1665	Female	79.0	1	0	1	Self-employed	Rural	174.12	24.0	never smoked

```
   stroke         imc_str grupo_edad
0       1      Obeso g II        65+
1       1             NaN      50-64
2       1       Obeso g I        65+
3       1       Obeso g I      35-49
4       1  Peso saludable        65+
```

Nos habiamos quedado con la columna id, aunque la información que da no es relevante para el análisis, por ver si había alguna repetición. Asi que creamos un nuevo dataset sin esa columna

[20]:
```
df1 = df.drop('id', axis = 1)
```

Hemos acabado el data preparation y understanding. Creamos un nuevo archivo csv corregido que vamos a estudiar en R

[21]:
```
df1.to_csv('stroke_corregido.csv', index = False)
```

[ ]:
```

```