

prog_datasci_6_preproc_EjerciciosResueltos

May 6, 2020

1 Programación para *Data Science*

1.1 Unidad 6: Preprocesamiento de datos en Python

Ejercicios para practicar

2 Introducción

En este Notebook encontraréis un primer conjunto de **ejercicios para practicar**, éstos no puntúan para la PEC, pero os recomendamos que los intentéis resolver como parte del proceso de aprendizaje. Encontraréis ejemplos de posibles soluciones a los ejercicios al propio notebook, pero es importante que intentéis resolverlos vosotros antes de consultar las soluciones. Las soluciones os permitirán validar vuestras respuestas, así como ver alternativas de resolución de las actividades. También os animamos a preguntar cualquier duda que surja sobre la resolución de los **ejercicios para practicar** en el foro del aula.

2.1 Ejercicios para practicar

Los siguientes 3 ejercicios no puntúan para la PEC, pero os recomendamos que los intentéis resolver antes de pasar a los ejercicios propios de la PEC. También encontraréis las soluciones a estos ejercicios al final del Notebook.

2.2 Ejercicio 1

Cargue los datos del fichero `bank_edited.csv` en un dataframe. Este conjunto de datos recoge información respecto a una campaña de marketing de un banco portugués. El conjunto original se puede encontrar en el [repositorio de datos de Machine Learning de la UC Irvine] (<http://archive.ics.uci.edu/ml/datasets/Bank+Marketing>), pero el conjunto que utilizaremos tiene alguna modificación .

Observación: revise la documentación de la función `read_csv` para ver qué parámetro disponemos para ajustar el proceso de cargar de datos.

Los valores del estado civil (atributo `marital`) contienen errores tipográficos y incluyen el uso de diferentes nomenclaturas. En este ejercicio unificaremos la nomenclatura de los valores de esta variable. NM

a) ¿Cuántos valores diferentes tiene el atributo `marital` en el conjunto de datos? Mostrad estos valores.

```
[10]: # Respuesta
```

b) Unificad los atributos `marital` en los valores: “single”, “married” o “divorced”.

```
[11]: # Respuesta
```

c) ¿Qué columnas contienen valores perdidos?

```
[12]: # Respuesta
```

d) Calculad el primer y el tercer cuartil del atributo “balance”.

```
[13]: # Respuesta
```

2.3 Ejercicio 2

El atributo `poutcome` contiene información sobre si el cliente del banco contrató un depósito. Calcula la correlación entre el atributo `poutcome` y el resto de atributos (usa la función ‘`corr`’). ¿Qué variable presenta mayor correlación con `poutcome`? NM

```
[14]: # Respuesta
```

2.4 Ejercicio 3

El módulo `sklearn` incluye varios datasets de ejemplo, dentro del módulo `sklearn.datasets`. Estos datasets se almacenan en formato **Bunch**, propio de `sklearn`. Un **Bunch** es un objeto tipo diccionario, los atributos interesantes son: `data`, con los datos en crudo, `target`, con generalmente las etiquetas de clasificación o etiquetas objetivo, `target_names`, el significado de las etiquetas, `feature_names`, el significado de las características o atributos, `DESCR`, la descripción completa del conjunto de datos.

Importa el dataset `iris` de `sklearn`. Almacena los datos este dataset como un objeto `pandas`, con los correspondientes nombres de variables. Añade la variable `target` en el dataframe con el nombre de atributo `Species` y los valores con el tipo de especie de cada muestra. EG

```
[15]: # Respuesta
```

A continuación, los **ejercicios y preguntas teóricas que debe completar en esta PEC** y que forman parte de la evaluación de esta unidad.

2.5 Soluciones a los ejercicios para practicar

2.6 Ejercicio 1

- a) ¿Cuántos valores diferentes tiene el atributo `marital` en el conjunto de datos? Mostrad estos valores.

```
[30]: import pandas as pd
import numpy as np

data = pd.read_csv("data/bank_edited.csv", sep=";", dtype={"balance":np.float})

import numpy as np

# Con unique podemos encontrar los valores únicos

v = data.marital.unique()

print("There are {} different values in marital:\n{}".format(len(v), v))
```

There are 11 different values in marital:

```
['married' 'single' 'married' 'divorced' 'married' 'sing' 'Married'
 'MARRIED' 'DIVORCED' 'Single' 'SINGLE']
```

- b) Unificad los atributos `marital` en los valores: “single”, “married” o “divorced”.

```
[31]: data.loc[(data.marital == "Married") | (data.marital == "married") | (data.
    ↳ marital == "MARRIED") |
        (data.marital == "married"), "marital"] = "married"
data.loc[(data.marital == "Single") | (data.marital == "SINGLE") | (data.
    ↳ marital == "sing"), "marital"] = "single"
data.loc[(data.marital == "DIVORCED"), "marital"] = "divorced"

# Comprobación

v = data.marital.unique()
print("There are {} different values in marital:\n{}".format(len(v), v))
```

There are 3 different values in marital:

```
['married' 'single' 'divorced']
```

- c) Qué columnas contienen valores perdidos?

```
[32]: # Para esto usamos la función any_isna a cada columna del dataframe
print(data.isna().any())
```

```
age          False
job          False
marital      False
```

```

education    False
default      False
balance      True
housing      False
loan         False
contact      False
day          True
month        False
duration     True
campaign     False
pdays       False
previous     False
poutcome     False
y            False
dtype: bool

```

d) Calculad el primer y el tercer cuartil del atributo **balance**.

```
[33]: print((data['balance'].quantile(0.25)))
      print(data['balance'].quantile(0.75))
```

```

68.0
1476.0

```

2.7 Ejercicio 2

El atributo **poutcome** contiene información sobre si el cliente del banco contrató un depósito. Calula la correlación entre el atributo **poutcome** y el resto de atributos (usa la función `corr`). Qué variable presenta mayor correlación con `poutcome`?

```
[34]: # Visualizamos los valores de la columna poutcome
import pandas as pd
import numpy as np
import warnings

warnings.filterwarnings('ignore')

data.poutcome.unique()
```

```
[34]: array(['unknown', 'failure', 'other', 'success'], dtype=object)
```

```
[35]: # Seleccionamos únicamente la muestras que contienen información precisa
      ↪ sobre el cliente contrató o no el depósito

data_pout = data[data.poutcome.isin(["failure", "success"])]

# Discretizamos la columna poutcome para poder calcular la correlación
```

```
data_pout['poutcome_cat'] = data_pout.poutcome.astype("category").cat.codes

# Calculamos la correlación con el resto de columnas

data_pout.corr()["poutcome_cat"]
```

```
[35]: age           0.090540
      balance       0.039791
      day           0.009252
      duration      0.142385
      campaign      -0.059986
      pdays        -0.276853
      previous      0.023411
      poutcome_cat  1.000000
      Name: poutcome_cat, dtype: float64
```

Observad que la columna `pdays` muestra tener mayor correlación en valor absoluto con `poutcome_cat`. Tened en cuenta que hemos asignado un 0/1 a “failure”/“success” de manera arbitraria, así que el signo no es significativo en este caso.

2.8 Ejercicio 3

El módulo `sklearn` incluye varios datasets de ejemplo, dentro del módulo `sklearn.datasets`. Estos datasets se almacenan en formato `Bunch`, propio de `sklearn`. Un `Bunch` es un objeto tipo diccionario, los atributos interesantes son: `data`, con los datos en crudo, `target`, con generalmente las etiquetas de clasificación o etiquetas objetivo, `target_names`, el significado de las etiquetas, `feature_names`, el significado de las características o atributos, `DESCR`, la descripción completa del conjunto de datos.

Importa el dataset `iris` de `sklearn`. Almacena los datos este dataset como un objeto `pandas`, con los correspondientes nombres de variables. Añade la variable `target` en el dataframe con el nombre de atributo `Species` y los valores con el tipo de especie de cada muestra.

```
[36]: from sklearn import datasets
      import pandas as pd
      iris_bunch = datasets.load_iris()
      iris_pandas = pd.
        ↳ DataFrame(iris_bunch['data'], columns=iris_bunch['feature_names'])
      iris_pandas['Species'] = iris_bunch['target_names'][iris_bunch['target']]
```

```
[37]: iris_pandas.head()
```

```
[37]:   sepal length (cm)  sepal width (cm)  petal length (cm)  petal width (cm)  \
0                5.1                3.5                1.4                0.2
1                4.9                3.0                1.4                0.2
2                4.7                3.2                1.3                0.2
3                4.6                3.1                1.5                0.2
```

4	5.0	3.6	1.4	0.2
---	-----	-----	-----	-----

```

Species
0  setosa
1  setosa
2  setosa
3  setosa
4  setosa

```

```
[38]: iris_pandas.tail()
```

```
[38]:
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	petal width (cm)	\
145	6.7	3.0	5.2	2.3	
146	6.3	2.5	5.0	1.9	
147	6.5	3.0	5.2	2.0	
148	6.2	3.4	5.4	2.3	
149	5.9	3.0	5.1	1.8	

```

Species
145  virginica
146  virginica
147  virginica
148  virginica
149  virginica

```

```
[39]: iris_pandas.describe()
```

```
[39]:
```

	sepal length (cm)	sepal width (cm)	petal length (cm)	\
count	150.000000	150.000000	150.000000	
mean	5.843333	3.057333	3.758000	
std	0.828066	0.435866	1.765298	
min	4.300000	2.000000	1.000000	
25%	5.100000	2.800000	1.600000	
50%	5.800000	3.000000	4.350000	
75%	6.400000	3.300000	5.100000	
max	7.900000	4.400000	6.900000	

```

petal width (cm)
count      150.000000
mean        1.199333
std         0.762238
min         0.100000
25%         0.300000
50%         1.300000
75%         1.800000
max         2.500000

```