**Semana 5 día 3**

Modificadores

Conceptos que van a permitir a una variable tipificarla como publica o privada.

Han de estar dentro de una clase

\_x # es una variable protegida. Se puede llamar a la función dentro del propio fichero/carpeta en el que se creó, aunque se solicite fuera de la clase. Si están fuera del fichero/carpeta no se puede acceder

\_\_X # es una variable privada. Privada encapsulada. Sólo se puede acceder a ella desde dentro de la propia clase en la que se ha creado. Funciones a las que tengan acceso otras personas o no. Se evita que algunos parámetros obligatorios para que funcione la clase se puedan cambiar por una persona externa. Puede aparecer dentro del fichero/carpeta donde se ha creado pero no permite el acceso. Desde otro fichero/carpeta ni siquiera aparecería esta función

If \_\_name\_\_ == ‘\_\_main\_\_’

se ejecuta lo de abajo

**class properties**

# se pueden crear atributos de una clase fuera de la clase al crear una instancia pero no es una buena práctica

Puedo acceder a una función privada de una clase creando otra función en es misma clase que me llame a la función privada

# @ es el decorador de la función

@property (es un decorador ya creado por defecto)

Para llamar a la función no hace falta que coloque los “()”, puedo acceder a una función como un atributo, aunque sea una función

def \_\_init\_\_(self, x):

self.x=\_\_x

@property

def x(self):

Return self.\_\_x

p.x # llamo a la función como un atributo(sin parentésis), en vez de como una función

@x.setter # se llama a la propiedad x para poder modificar la variable. Hay que crear la función con el mismo nombre de la propiedad (x)

def x(self,x):

if x< 0:

self.\_\_x = 0

elif x> 0:

self.\_\_x = 1000

else::

self.\_\_x = x

p.x # estoy llamando a la propiedad

p.x = -9999 # si asignamos un valor a x llamamos a setter

@classmethod # es una función que permite construir una nueva instancia utilizando unos atributos diferentes a los especificados en el constructor de la clase

Class person:

Def \_\_init\_\_ (self, name, age, is\_alive):

self,\_name = name

self.\_age = age

@classmethod

def from\_birthyear(cls, name, birthyear):

return cls(name=name, age=(date.today().year - birthyear), is\_alive = True)

Lucia = person,from\_birth\_year(cls, name, birthyear):

@staticmethod # va a utilizar algún concepto de la clase. Sin tener que crear una instancia de la clase pero son funciones que utilizan algún atributo de la clase

def is\_adult(age):

return age > 18

person.is\_adult()

# función método de la clase tablero

@staticmethod

def check\_boats(number\_of\_boats):

if number\_of\_boats ¡= 10:

print (“no tiene 10 barcos)

else:

print(‘tiene los barcos necesarios’)

# función de la instancia tablero

def check\_boats2(self, number\_of\_boats):

if number\_of\_boats ¡= 10:

print (“no tiene 10 barcos)

else:

print(‘tiene los barcos necesarios’)

tablero().check\_boats(number\_of\_boats=10)

poner al abrir archivos siempre ‘low\_memory=False’ para que lea todo correctamente. Para ficheros muy grandes

leer en Python con diccionarios, en caso de ficheros muy grandes. Panda tiende a ser eficiente e interpreta los datos de forma no muy rigurosa

users2=users.copy()

users2.drop(columns =[‘occupation’], inplace=True) # para que me haga el cambio directamente en la variable. Atribuir los cambios en la nueva base a la primera como aparece arriba, hay que ponerlo con users.copy para que no me cambie la base original

El None para numpy es NaN

Para el proyecto: 1000 filas y más o igual a 5 columnas

**13-04-2021**

Comprobar siempre que la versión de Python de jupyter y de Python es la misma. Aparecen, la de jupyter en la parte de arriba de visual studio y la de Python en la barra de abajo

**Correlaciones**

# pearson

Es un valor numérico entre -1 < r < 1

0 -> no correlación

1 -> las dos variables aumentan de igual forma. Correlación positiva alta entre 0.9 y baja: 0,5

-1 -> correlación negativa (mientras una variable aumenta la otra disminuye).

Import seaborn

Import matplotlib

Tamaño del cuadro de colores

plt.figure(figsize =(15,10))

c=df.corr()

sns.heatmap(c, cmap = ‘BrBg’, annot = True)

Gama de colores del cuadro de colores

La última línea de código me da un mapa de calor

Fijarnos en donde las correlaciones positivas son > 0.5 y las negativas

< - 0.45

sns.pairplot(data=df)

sns.pairplot(data=df, kind=’reg’)

dos columnas

sns.regplot (x =’nombre\_columna\_1’, y = ‘nombre\_columna2’ , data= df)

**detectar outliers**

sns.boxplot(x=df[‘Number of Doors’])

mediana = np.median(nombre del archivo)

media = nombre del archivo.mean()

std = nombre del archivo.std()

los outliers quedan fuera del máximo y el mínimo del boxplot

**detectar anomalías en los datos**

Puede cambiar con el tiempo porque a medida que se obtienen más datos puede dejara de ser una anomalía.

Siempre en la detección de anomalías hay que estudiar las propias anomalías

Quartiles

Q1 = df[‘Engine’].quantile(0.25) # que me de el cuartil 0.25 de la variable ‘Engine’

Q3 = df[‘Engine’].quantile(0.75)

IQR = Q3-Q1

Límite superior antes del outlier

Limite superior = Q3 +1.5\*IQR

# el valor 1.5 lo puedo establecer yo para detectar anomalías. Este valor es el establecido de forma estándar

Límite inferior antes del outlier

Limite inferior = Q1 -1.5\*IQR

Los valores entre el Q3 y el límite superior

df2 = df[((df<limite inferior)|( df>limite superior).any(axis=1))]

sns.boxplot(x=df2) # boxplot de las anomalías

representación de las anomalías con boxplot para conocer

df4 = df[df> limite superior]

sns.boxplot(x=df4) #boxplot de las anomalías del límite superior

lo que no es anomalía

df3 = df[ ((df<limite inferior)|( df>limite superior).any(axis=1))]

Símbolo de línea ondulada horizontal. Agua de ejercicio de barcos

Indica lo contrario de lo que está por encima y debajo de los límites establecidos

pueden aparecer valores que estén dentro del rango que teníamos como válidos porque está cogiendo todos los outliers de todas las columnas por lo que puede que esté recogiendo algún valor de alguna columna que esté dentro de los rangos válidos. Se ha cogido como referencia para cacular los limites los cuartiles de Engine pero al dar esta orden estamos calculando los outliers de toda la base, independientemente de la columna de la que se ha partido.

Sns.boxplot(x=df\_hp\_with\_outliers[‘Engine\_HP’]

df\_Engine\_HP = df[ ((df[‘Engine\_Hp’]<limite inferior)|( df[‘Engine\_Hp’]>limite superior))]

sns.boxplot(x=df\_Engine\_HP) # boxplot de los outlier de Engine\_HP

**Semana del 17 de mayo**

Detectar valores duplicados

Data[data.duplicated()]

Pasar a numérica una columna que, aunque tiene números aparecen como string

def try\_convert\_to int(x):

try:

return int(x)

except:

print(‘rare value:’, x)

return x

Admiración invertida

len[data[‘horsepower’] == ‘?’]

Hay que tomar una decisión sobre cómo actuar con esos valores raros

Una posibilidad es asignarles la media

data[data[‘horsepower’] ¡= ‘?’].horsepower.astype(int).mean()

data[‘horsepower’] .value\_counts (normalize = True)

asignar la media con Python a toda la columna

horspower\_mean = 0

count = 0

for pos, value in

import argpase

para ejecutar un archivo de Python pasándole los parámetros que queramos

librería que recoge cada uno de los valores que le vamos dando

copiamos la ruta absoluta que aparece.

Vamos a la terminal de Windows (pegamos con botón derecho). Se ejecuta el archivo pero por la terminal. Mientras se va ejecutando podemos seguir trabajando con visual studio code.

Ctrl+ C interrumpimos la ejecución

from my\_module import exponent

función exponent (x,y, v=None):

result = x\*\*y

return result \*v if v else result

import sys

aquí no hace falta especificar si es x, y, v. Siempre se recogen todos los valores como un string

**Martes 18 de mayo**

Instalar la librería streamlit y altair

Ploty.- librería para crear gráficas interactivas

Copiamos la ruta al fichero que aparece en el archivo app.py de la semana 6

St. Button lo coloco encima o debajo de la tabla según coloque el if button antes o después del if write

Importar la librería pandas\_profile