Proyecto 1 "Calcular las enfermedades cardiovasculares en Sudáfrica

Tuesday, April 28, 2020 12:40 PM

En este proyecto se busca el obtener un modelo de machine learning que pronostique la principal causa de enfermedades cardiovasculares en Sudáfrica.

Para comenzar, se debe contar con el archivo "phpgNaXZe.csv" el cual se obtiene de la página de Openml: https://www.openml.org/d/1498

El archivo cuenta con la siguiente información:

A	B	C	D	E	F	G	H	I	J
V1	V2	V3	V4	V5	V6	V7	V8	V9	Class
160	12	5.73	23.11	1	49	25.3	97.2	52	2
144	0.01	4.41	28.61	2	55	28.87	2.06	63	2
118	0.08	3.48	32.28	1	52	29.14	3.81	46	1
170	7.5	6.41	38.03	1	51	31.99	24.26	58	2
134	13.6	3.5	27.78	1	60	25.99	57.34	49	2
132	6.2	6.47	36.21	1	62	30.77	14.14	45	1
142	4.05	3.38	16.2	2	59	20.81	2.62	38	1
114	4.08	4.59	14.6	1	62	23.11	6.72	58	2
114	0	3.83	19.4	1	49	24.86	2.49	29	1
132	0	5.8	30.96	1	69	30.11	0	53	2

Como se puede observar el archivo cuenta con 10 columnas cuyos nombres no muestran de que trata dicha información, por lo cual se deberá cambiar la información por la siguiente:

- V1→"Sbp"
- V2→"Tabaco"
- V3→"Ldl"
- V4→"Adiposity"
- V5→"Familia"
- V6→"Tipo"
- V7→"Obesidad"
- V8→"Alcohol"
- V9→"Edad"
- Class→"Chd"

Está información se encuentra disponible en la descripción de la base de datos

- Title:

South Africa Heart Disease Dataset

- Description

A retrospective sample of males in a heart-disease high-risk region of the Western Cape, South Africa. There are roughly two controls per case of CHD. Many of the CHD positive men have undergone blood pressure reduction treatment and other programs to reduce their risk factors after their CHD event. In some cases the measurements were made after these treatments. These data are taken from a larger dataset, described in Rousseauw et al, 1983, South African Medical Journal.

- Attributes:

sbp systolic blood pressure
tobacco cumulative tobacco (kg)
ldl low densiity lipoprotein cholesterol
adiposity
famhist family history of heart disease (Present, Absent)
typea type-A behavior
obesity
alcohol current alcohol consumption
age age at onset
chd response, coronary heart disease

Para iniciar el proyecto, se hará uso del IDE Sublime Text 3. Se debe contar con las librerías Pandas y

Matplotlib instaladas.

Importamos ambas librerías de la siguiente forma:

```
import pandas as pd
import matplotlib.pyplot as plt
```

Se lee la información del archivo csv con el siguiente comando y se guarda en la variable data, seguidamente se pide una impresión de la cabecera de la data

```
data=pd.read_csv('phpgNaXZe.csv')
print(data.head())
```

La información de la cabecera que se muestra es la siguiente:

```
    V1     V2    V3     V4  V5  V6     V7     V8  V9  Class
0  160  12.00  5.73  23.11   1  49  25.30  97.20  52      2
1  144   0.01  4.41  28.61   2  55  28.87   2.06  63      2
2  118   0.08  3.48  32.28   1  52  29.14   3.81  46      1
3  170   7.50  6.41  38.03   1  51  31.99  24.26  58      2
4  134  13.60  3.50  27.78   1  60  25.99  57.34  49      2
```

Ahora se modificará los nombres de las columnas de nuestra data con la información mencionada al inicio del reporte y se vuelve a imprimir la cabecera de la data

```
Columnas=["Sbp","Tabaco","Ldl","Adiposity","Familia","Tipo","Obesidad","Alcohol","Edad","Chd"]
data.columns=Columnas #Se modifica el nombre de las columnas
print(data.head())
```

La información de la cabecera que se muestra ya con la modificación es la siguiente:

```
   Sbp  Tabaco   Ldl  Adiposity  Familia  Tipo  Obesidad  Alcohol  Edad  Chd
0  160   12.00  5.73      23.11        1    49     25.30    97.20    52    2
1  144    0.01  4.41      28.61        2    55     28.87     2.06    63    2
2  118    0.08  3.48      32.28        1    52     29.14     3.81    46    1
3  170    7.50  6.41      38.03        1    51     31.99    24.26    58    2
4  134   13.60  3.50      27.78        1    60     25.99    57.34    49    2
```

Ahora se debe verificar la información de los datos en las columnas con el comando dtypes

```
#Conocer el tipo de datos de la data
print(data.dtypes)
#La mayoria de los datos son entero y flotante, por lo cual no se requiere algun cambio
```

La información indicada son enteros y flotantes, por lo cual no se tendrá que hacer una modificación en el tipo de dato

```
Sbp           int64
Tabaco      float64
Ldl         float64
Adiposity   float64
Familia       int64
Tipo          int64
Obesidad    float64
Alcohol     float64
Edad          int64
Chd           int64
dtype: object
```

Ahora se debe verificar la cantidad de datos nulos en cada columna, para esto se ocupa el comando isnull().sum()

```
Sbp           0
Tabaco        0
Ldl           0
Adiposity     0
Familia       0
Tipo          0
Obesidad      0
Alcohol       0
Edad          0
Chd           0
dtype: int64
```

Los datos de Familia y Chd son valores numéricos 1 y 2, por lo cual se deberá cambiar la información por 0 y 1, para esto se usara la librería de sklearn.preprocessing usando la clase LabelEncoder

```
#Los datos de familia y chd son valores 1 y 2, se cambiaran por 0 y 1
#Cambiar los datos de las columnas familia y chd
from sklearn.preprocessing import LabelEncoder #Modifica los resultados de una etiqueta
```

Para hacer este cambio se usa la clase LabelEncoder para de ahí usar la función fit_transform y realizar los cambios en los valores en las columnas familia y chd

```
encoder=LabelEncoder()
data["Familia"]=encoder.fit_transform(data["Familia"])
data["Chd"]=encoder.fit_transform(data["Chd"])
print(data.head())
#Donde antes habia un 1 se cambio por 0 y el 2 cambio a 1
```

Se imprime la cabecera de la data, con estas modificaciones

```
   Sbp  Tabaco   Ldl  Adiposity  Familia  Tipo  Obesidad  Alcohol  Edad  Chd
0  160   12.00  5.73      23.11        0    49     25.30    97.20    52    1
1  144    0.01  4.41      28.61        1    55     28.87     2.06    63    1
2  118    0.08  3.48      32.28        0    52     29.14     3.81    46    0
3  170    7.50  6.41      38.03        0    51     31.99    24.26    58    1
4  134   13.60  3.50      27.78        0    60     25.99    57.34    49    1
```

Los valores de la columna Sbp son muy desproporcionados, por lo cual es momento de hacer un escalamiento de 0 a 100, para esto de la librería sklearn.preprocessing se importa la función MinMaxScaler, a la columna data que se modifica la escala se debe poner en una matriz 2D por lo cual se usa el reshape(-1,1)

```
#Con esta función se establecen un minimo y maximo de valores definidos y los transforma a este rango
scale=MinMaxScaler(feature_range=(0,100))
data["Sbp"]=scale.fit_transform(data["Sbp"].values.reshape(-1,1))
print(data.head())
```

Se imprime la cabecera de la data, ya con todas las modificaciones hechas

```
         Sbp  Tabaco   Ldl  Adiposity  ...  Obesidad  Alcohol  Edad  Chd
0  50.427350   12.00  5.73      23.11  ...     25.30    97.20    52    1
1  36.752137    0.01  4.41      28.61  ...     28.87     2.06    63    1
2  14.529915    0.08  3.48      32.28  ...     29.14     3.81    46    0
3  58.974359    7.50  6.41      38.03  ...     31.99    24.26    58    1
4  28.205128   13.60  3.50      27.78  ...     25.99    57.34    49    1
```

Comenzaremos a graficar la información del data con plot de la siguiente manera:
data.plot(x="Nombre de la columna en el eje x", y="Nombre de la columna en el eje y", title="Titulo de tu grafica", kind="Tipo de grafico", fig_size=("Tamaño en el eje x", "Tamaño en el eje y"))

Se crearan 3 graficos con la siguiente información

En el eje X se pondrá la edad
En el eje Y se pondrá Obesidad, Tabaco y Alcohol
Tipo de grafico será scatter
El tamaño de la figura será 10 en X y 5 en Y

```
#Visualizar la obesidad de acuerdo a la edad
data.plot(x="Edad",y="Obesidad",title="Edad vs Obesidad",kind="scatter",figsize=(10,5))
#Visualizar el tabaco consumido de acuerdo a la edad
data.plot(x="Edad",y="Tabaco",title="Edad vs Tabaco consumido",kind="scatter",figsize=(10,5))
#Visualizar el alcohol consumido de acuerdo a la edad
data.plot(x="Edad",y="Alcohol",title="Edad vs Alcohol consumido",kind="scatter",figsize=(10,5))
plt.show()
```

Gráfico 1

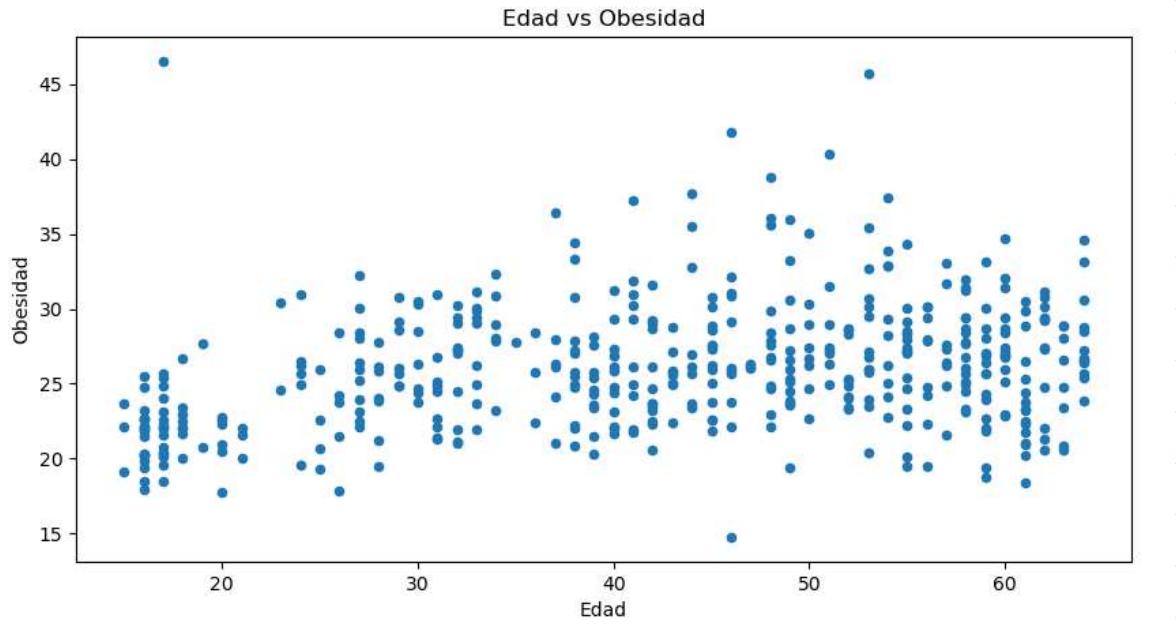


Gráfico 2

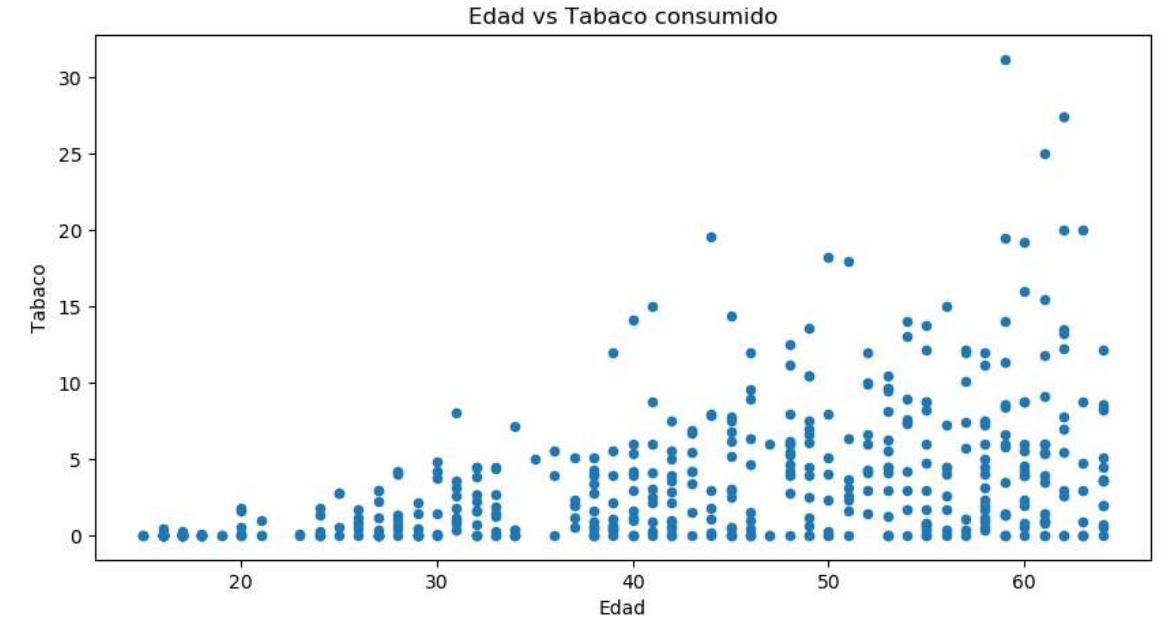
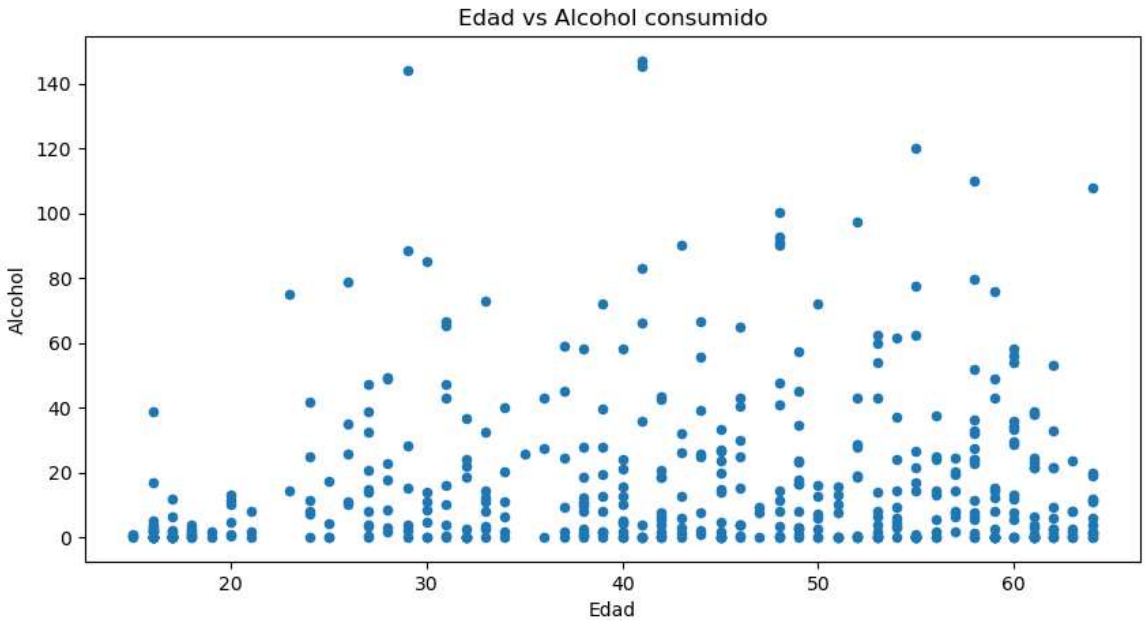


Gráfico 3



Se hará uso del modelo de Maquinas de vectores de soporte, así mismo se dividirá el data en prueba y entrenamiento y se hará uso de la matriz de confusión, además de la precisión y exactitud del modelo, por lo cual se deben importar las siguientes librerías

```
#Separar datos de entrenamiento y prueba
from sklearn.model_selection import train_test_split
#Importamos el modelo de maquinas de vectores de soporte, por que es un problema de clasificación
from sklearn import svm
#Importamos mas funciones para trabajar
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score
```

Se definen las variables X e Y, en donde Y sera la columna 10 "Chd" y X seran las columnas 1 a la 9, para esto último usaremos la función drop para eliminar la columna 10

```
#Definir variable dependiente e independiente
y=data["Chd"] #Y sera la columna Chd
X=data.drop("Chd",axis=1) #X el resto de valores del conjunto de datos
```

Vamos a separar los datos de X e Y en prueba y entrenamiento, para eso se usará un muestra del 80% en entrenamiento y del 20% en prueba

```
#Separar los datos de entrenamiento y prueba
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_state=1)
```

Creamos la variable algoritmo que usará el modelo de machine learning seleccionado, para lo cual se hará uso de un kernel lineal

```
#Definir el algoritmo a utilizar
algoritmo=svm.SVC(kernel="linear")
```

Se entrenará el algoritmo con los datos de X_trian e y_train

```
#Definir el algoritmo
algoritmo.fit(X_train,y_train)
```

Una vez entrenado nuestro algoritmo, se le brindara la información de X_test que son los datos de prueba para realizar una predicción del modelo

```
#Realizar la predicción
y_test_pred=algoritmo.predict(X_test)
```

Realizada la predicción, se hará uso de la matriz de confusión para ver la cantidad de datos correctos y

erróneos que tenemos en la predicción

```
#Se calcula la matriz de confusión
print(confusion_matrix(y_test,y_test_pred))
#Los datos correctos son [a11]+[a22], diaognal principal
#Los datos incorrectos son [a12]+[a21], diaonal secundaria
```

La suma de los valores en la diagonal principal es el número de datos correctos, en las diagonales secundarias se encuentran los datos incorrectos, en total se tienen 67 datos correctos y 26 datos incorrectos

```
[[57  9]
 [17 10]]
```

Por ultimo hay que imprimir el valor de la exactitud y precisión del modelo de Machine learning usado, por lo cual se usan los siguientes comandos

```
#Se calcula la exactitud del modelo
print(accuracy_score(y_test,y_test_pred))
#Se calcula la presición del modelo
print(precision_score(y_test,y_test_pred))
```

La precisión del modelo es de 52.63%
La exactitud del modelo es de 72.04%

```
0.7204301075268817
0.5263157894736842
```