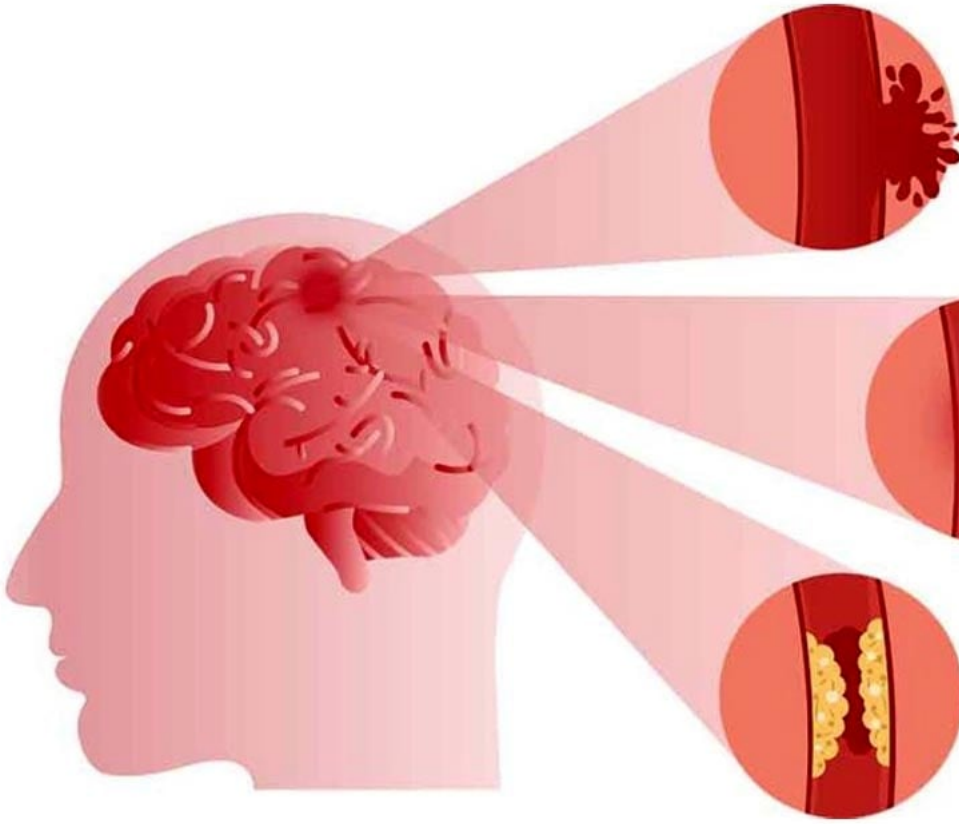
Modulo 2: Implementacion de una tecnica de aprendizaje máquina sin el uso de un framework

Juan Carlos Varela Téllez A01367002
Fecha de inicio: Fecha de inicio: 24/08/2022
Fecha de finalizacion: 31/08/2022

En caso de no tener las bibliotecas necesarias, utilizar los siguientes comandos:
python -m pip install numpy
python -m pip install pandas
python -m pip install scikit-learn

Las apoplejias son un evento cuando el suministro de sangre al cerebro se ve interrumpida, causando en falta de oxigeno, daño cerebral y perdida de funciones tanto motoras como mentales.
Globalmente, 1 de cada 4 adultos mayores de 25 años va a tener una apoplejia en su vida.
12,2 millones de personas tendra su primer apoplejia en este año, y 6.5 millones mas
moriran como resultado de esta. Mas de 110 millones de personas han tenido una apoplejia.[1]
Este codigo tiene como objetivo analizar datos para poder predecir que personas son mas propensas a tener una apoplejia y asi poder evitar secuelas y bajar estas estadisticas.
[1] https://www.world-stroke.org/world-stroke-day-campaign/why-stroke-matters/learn-about-stroke#:~:text=Globally%201%20in%204%20adults,the%20world%20have%20experienced%20stroke.



Para poder leer, procesar y analizar los datos e información que sacaremos de dichos datos es necesario importar ciertas bibliotecas que nos ayudaran de forma importante:

- Pandas: esta biblioteca nos ayuda a leer nuestros datos, al igual que modificar nuestros datos a traves de un data-frame para manipularlos y analizarlos. Para más información haz click aquí.
- Numpy: esta biblioteca nos da diferentes herramientas matemáticas vectorizadas para acelerar nuestros cálculos. Para más información haz click aquí.

- Scikit-learn: esta biblioteca es de las más importantes que se utiliza ya que contiene la gran mayoría de herramientas de machine learning que se van a utilizar en este reto, desde regresiones hasta bosques aleatorios. En este caso solamente vamos a utilizar su funcion para modularizar los datos en bloques de entrenamiento, validacion y pruebas. Para más información haz click aquí.

Ahora vamos a importar nuestro data-set para poder trabajar. El dat-set se puede encontrar en este link.

Para este analisis y creacion de modelo de regresion logistica, se van a tomar en cuenta distintas caracteristicas que son importantes, como la edad, si eran fumadores, enfermedades del corazon, etc para crear los modelos.
Asimismo se van a utilizar caracteristicas que al momento no se consideran relevantes, como residencia, genero, estado civil, etc para comparar y ver que tan fuerte es la conexion entre caracteristicas que generalmente se consideran relevantes contra caracteristicas que normalmente no se considerar relevantes.

Creacion del modelo sin framework :computer:

El modelo que se creo para esta actividad no fue desarrollado con librerias ni frameworks. Se trata de un modelo de regresion logistica cuyo algoritmo de optimizacion es gradiente descendiente. Asimismo, al momento de utilizar mas caracteristicas se tuvi que crear modelos que pudieron utilizar mas variables. Para ver los modelos y como se crearon haga click aqui.

Data-set :chart_with_upwards_trend:

Para poder entender mejor nuestros datos, es necesario saber con que columnas cuenta, asi que para eso vamos a la documentacion del mismo data-set para saber los metadatos.

1) gender: "Male", "Female" or "Other"
2) age: age of the patient
3) hypertension: 0 if the patient doesn't have hypertension, 1 if the patient has hypertension
4) heart*disease: 0 if the patient doesn't have any heart diseases, 1 if the patient has a heart disease 5) ever*married
6) work*type: "children", "Govt*jov", "Never*worked", "Private" or "Self-employed" 7) Residence*type: "Rural" or "Urban"
8) avg*glucose*level: average glucose level in blood
9) bmi: body mass index
10) smoking_status: "formerly smoked", "never smoked", "smokes" or "Unknown"*
11) stroke: 1 if the patient had a stroke or 0 if not

*Note: "Unknown" in smoking_status means that the information is unavailable for this patient

Primera caracteristica: edad :walking:

Primero utilizaremos una caracteristica que es considerada importante, al menos superficialmente: la edad.
Nuestras funciones de hipotesis y costo se encuentran en el modulo de [modelos.py]
((https://github.com/JuanVaTe/RetoModulo2/blob/main/modelos.py).
Se utilizaron los siguientes:

```
h1 = lambda x, theta: 1 / (1 + m.exp(theta[0] + theta[1] * x))
j_1 = lambda x, y, theta: (y * m.log(h1(x, theta))) + ((1 - y) * (m.log(1 - h1(x, theta))))
```

Ahora lo unico que falta es implementarlo. Con una *theta0* de 1, *theta1* de 1, 1000 iteraciones y un alfa de 0.00001, el resultado de nuestro modelo fue el siguiente:

```
==========================================
Valores de theta para modelo usando 'edad'
[1.000448246366269, 1.0323902543753165]
==========================================
Valor de costo promedio
3.9573028374719694
==========================================
Matriz de confusion
[[0, 68], [0, 1178]]
==========================================
Metricas de rendimiento para caracteristica 'edad'
Exactitud     : 0.9454253611556982
Precision     : 0
Exhaustividad : 0.0
Puntaje F1    : 0
==========================================
```

Aunque la exactitud nos diga que nuestro modelo puede llegar a ser bueno, todas las demas metricas se encuentran en el suelo. Nuestro modelo predice que absolutamente todas las personas no tienen riesgo de apoplejia.

Segunda caracteristica: fumadores :smoking:

Esta es otra caracteristica que se considera importante en cuanto a la salud.
Antes de poder empezar a implementar un modelo de regresion logistica, debemos convertir nuestros datos cualitativos a datos numericos binarios.
Para esto, pandas cuenta con una funcion que nos puede ayudar llamada `get_dummies()`.

Primero vemos los datos unicos con los que cuenta nuestro data-frame:

```
Valores unicos en caracteristca de 'fumador'
['formerly smoked' 'never smoked' 'smokes' 'Unknown']
```

Ya que tenemos datos incompletos, vamos a quitarlos para no contaminar nuestro modelo de regresion
Tambien tendremos que quitar estos datos de nuestra variable dependiente para que se mantengan con la misma cantidad de datos.
Asimismo, debido a que la variable cualitativa *smoking_status* tiene un dominio mayor de 2, es necesario crear una columna para cada valor unico.
Lo ultimo que falta hacer es modularizar nuevamente los datos para obtener nuestros 3 bloques de entrenamiento, validacion y pruebas.
Con todo esto completado podemos implementar nuevamente nuestro modelo de regresion logistica.
Al igual que el modelo anterior, se empieza con todos los valores *theta* en 1, un alfa de 0.00001 y 1000 iteraciones.
El resultado fue el siguiente:

```
==========================================
Valores de theta para modelo usando 'smoking'
[0.9995173436401023, 0.9998120716237856, 0.9999347102283542]
==========================================
Valor de costo promedio
0.25922562464679916
==========================================
Matriz de confusion
[[0, 45], [0, 826]]
==========================================
Metricas de rendimiento para caracteristica 'smoking'
Exactitud     : 0.9483352468427095
Precision     : 0
Exhaustividad : 0.0
Puntaje F1    : 0
==========================================
```

Desafortunadamente, este modelo tiene el mismo rendimiento que el anterior.
Lo que podemos inferir viendo el data-set con el cual estamos trabajando es que la mayoria de los datos indican que la persona no tuvo apoplejia, entonces nuestros modelos aprenden eso. Esto es ya que nuestros modelos son simples y se necesitaria un mejor metodo o modelo para poder empezar a predecir de forma correcta.

Tercera caracteristica: tipo de residencia :house_with_garden:

La intuicion nos indica que esta caracteristica no deberia tener tanto peso como la edad o si la persona fumaba. Para poder asegurarnos de esto, vamos a hacer un modelo de regresion logistica para comparar su rendimiento.
Aunque esta es una variable cualitativa y necesita cuantificarse, debido a que su rango es de 2, podemos cuantificar con un valor binario (0, 1).
Por ultimo, no podemos olvidarnos de la modularizacion de los datos.
Con todo el proceso completado podemos implementar el mismo modelo con los mismos hiperparametros (*theta* con valor de 1, alfa de 0.00001 y 1000 iteraciones).
El resultado es el siguiente:

```
============================================
Valores de theta para modelo usando 'residence'
[0.9992738165638939, 0.9998342896968835]
============================================
Valor de costo promedio
0.2984727040766804
============================================
Matriz de confusion
[[0, 68], [0, 1178]]
============================================
Metricas de rendimiento para caracteristica 'residence'
Exactitud     : 0.9454253611556982
Precision     : 0
Exhaustividad : 0.0
Puntaje F1    : 0
============================================
```

Podemos observar el mismo fenomeno donde nuestro modelo predice que no hay peligro alguno de apoplejia. Aunque estamos utilizando una variable poco relevante, esto nos indica que podriamos necesitar un modelo mas complejo.

Modelo combinado: edad y fumadores :walking: :smoking:

Para poder crear un modelo mas complejo, necesitamos contemplar mas caracteristicas, es por eso que vamos a utilizar las caracteristicas que consideramos mas relevantes y vamos a usarlas al mismo tiempo en un nuevo modelo.
Como se hizo anteriormente, empezaremos por quitar las filas con el valor de `Unknown`, cuantificaremos la variable de fumadores representandola con varias columnas para cada valor que pueda obtener y modularizaremos nuestros datos.
Implementaremos el mismo modelo con los mismo hiperparametros (*theta* con valor de 1, alfa de 0.00001 y 1000 iteraciones).
El resultado es el siguiente:

```
============================================
Valores de theta para modelo usando 'edad' y 'smoking'
[1.000298850346251, 1.0001302680905189, 1.0000651340981292, 1.0203333308862053]
============================================
Valor de costo promedio
3.7293613399136745
============================================
Matriz de confusion
[[0, 45], [0, 826]]
============================================
Metricas de rendimiento para caracteristica 'residence'
Exactitud     : 0.9483352468427095
Precision     : 0
Exhaustividad : 0.0
Puntaje F1    : 0
============================================
```

Incluso utilizando diferentes caracteristicas, si el modelo sigue siendo simple, vemos que tiene sus limitaciones. Para poder resolver y predecir complicaciones como estas es necesario utilizar metodos mas optimos y poderosos.

Predicciones :stars:

Ya que el modelo mas complejo fue este ultimo, vamos a hacer unas predicciones para verificar estos resultados

```
Prediccion numero 1 ======================
Datos: never smoked     0.0
smokes           0.0
age             40.0
Name: 3311, dtype: float64
Prediccion: 0.0
Realidad: 0
==========================================
Prediccion numero 2 ======================
Datos: never smoked     1.0
smokes           0.0
age             81.0
Name: 2173, dtype: float64
Prediccion: 0.0
Realidad: 0
==========================================
Prediccion numero 3 ======================
Datos: never smoked     0.0
smokes           0.0
age             51.0
Name: 136, dtype: float64
Prediccion: 0.0
Realidad: 1
==========================================
Prediccion numero 4 ======================
Datos: never smoked     1.0
smokes           0.0
age             51.0
Name: 364, dtype: float64
Prediccion: 0.0
Realidad: 0
==========================================
Prediccion numero 5 ======================
Datos: never smoked     0.0
smokes           0.0
age             29.0
Name: 368, dtype: float64
Prediccion: 0.0
Realidad: 0
==========================================
Prediccion numero 6 ======================
Datos: never smoked     1.0
smokes           0.0
age             17.0
Name: 3197, dtype: float64
Prediccion: 0.0
Realidad: 0
==========================================
```

Conclusion :heavy_check_mark:

Es importante no casarse con solamente un tipo de modelo para resolver distintos problemas. Aunque en este codigo se utilizaron modelos muy basicos, si se intenta resolver todos los problemas con la misma herramienta, no importa lo robusta o poderosa, no siempre va a dar el mejor resultado.

Para poder obtener una vista mas detallada del proceso detras de escenas, puedes revisar el codigo aqui.

Mejoras a partir de la retroalimentacion

- Creacion de documentacion en README.md
- Creacion de documentacion en .pdf
- Implementacion de predicciones explicitas
- Conculsion del codigo