Nicolas Segura - 201716989

Recomendaciones SaludAlpes

Link del repositorio:

La presentación se encuentra en los entregables como Presentación SaludAlpes.pptx

El dashboard se encuentra en el repositorio.

1. Tratamiento de los datos:
   - **Perfilamiento de datos**

   La base de datos cuenta con un total de 100.000 datos, cada uno con 27 variables diferentes.

```
# Cantidad de datos y número de variables
diagnostics.shape
```

```
(100000, 27)
```

Al revisar el tipo de datos nos damos cuenta de que las ultimas columnas no tienen datos ni nombre de variables

ivity	Fruits	Veggies	HvyAlcoholConsump	...	MentHlth	PhysHlth	DiffWalk	Sex	Age	Education	Income	Unnamed: 22	Unnamed: 23	Unnamed: 24	Unnamed: 25	Unnamed: 26
0	0	1	0	...	18	15	1	0	9	4	3	NaN	NaN	NaN	NaN	NaN
1	0	0	0	...	0	0	0	0	7	6	1	NaN	NaN	NaN	NaN	NaN
0	1	0	0	...	30	30	1	0	9	4	8	NaN	NaN	NaN	NaN	NaN
1	1	1	0	...	0	0	0	0	11	3	6	NaN	NaN	NaN	NaN	NaN
1	1	1	0	...	3	0	0	0	11	5	4	NaN	NaN	NaN	NaN	NaN

Aquí podemos ver una observar una descripción de los datos

```
# Y hacer una descripción de los datos
diagnostics.describe()
```

	Diabetes_012	HighBP	HighChol	CholCheck	BMI	Smoker	Stroke	HeartDiseaseorAttack	PhysActivity	Fruits	Veggies	HvyAlcoholConsump	...	Menti
count	99984	99982	99991	99984	99981	99979	99987	99987	99985	99976	99989	99988	...	99
unique	6	5	5	5	95	5	5	5	5	5	5	5	...	
top	0	0	0	1	27	0	0	0	1	1	1	0	...	
freq	84078	56544	56814	96330	9624	55426	95818	90617	76135	63639	81464	94248	...	68

Aquí podemos ver los datos nulos por columna, en las últimas cinco podemos ver 2ue todos son Nan.

```
diagnostics.isna().sum(axis = 0)
```

```
Diabetes_012                16
HighBP                      18
HighChol                     9
CholCheck                   16
BMI                         19
Smoker                      21
Stroke                      13
HeartDiseaseorAttack        13
PhysActivity                15
Fruits                      24
Veggies                     11
HvyAlcoholConsump           12
AnyHealthcare               16
NoDocbcCost                 22
GenHlth                     18
MentHlth                    14
PhysHlth                    22
DiffWalk                    13
Sex                          9
Age                         12
Education                   19
Income                       1
Unnamed: 22             100000
Unnamed: 23             100000
Unnamed: 24             100000
Unnamed: 25             100000
Unnamed: 26             100000
dtype: int64
```

- **Preparación de datos**

Para comenzar, se eliminaron las columnas “Unnamed” y “AnyHealthCare” ya que no brindan información de relevancia para el análisis.

```
# Es recomendable que todos los pasos de limpieza y preparación se realicen sobre otro archivo.
diagnostics_t = diagnostics.copy()
# Primero eliminamos las columnas que teniana  todos los valores en Nan, segundo eliminamos al atributo AnyHealthcare ya que tener un seguro medico no esta relacionado con tener diabetes
diagnostics_t = diagnostics_t.drop(["AnyHealthcare",'Unnamed: 22','Unnamed: 23','Unnamed: 24','Unnamed: 25','Unnamed: 26',], axis=1)
```

Se identificaron filas con valores nulos e inconsistentes en las columnas: “MentHlth” , “GenHlth”, “PhysHlth”, “Income”, “Education” y “Age”. Se tomó la anterior dado que resulta ser de suma importancia alimentar al modelo con datos consistentes que ayudará a obtener un buen diagnóstico de los pacientes.

```
# Eliminación registros con ausencias, en este caso se eliminaron los registros que registraban 0 en PsyHlth, BMI, GenHLth o MentHlth
diagnostics_t.replace('-', np.nan, inplace=True)
indices = diagnostics_t[(diagnostics_t['MentHlth'] == 0) | (diagnostics_t['GenHlth'] == 0) | (diagnostics_t['PhysHlth'] == 0) | (diagnostics_t['Income'] == 0) | (diagnostics_t['Education'] == 0) | (diagnostics_t['Age'] == 0) ].index
diagnostics_t.drop(indices, inplace=True)
diagnostics_t.dropna(axis=0, inplace=True)
```

Igualmente, se eliminaron las columnas con valores que contengan los caracteres ‘?’, ‘-’ y ‘Xx’.

```
diagnostics_t=diagnostics_t[~diagnostics_t.isin(['?','-','Xx'])]
diagnostics_t=diagnostics_t.dropna(axis=0)
diagnostics_t.describe()
```

Finalmente, revisamos los estadísticos de la variable de interés.

```
# Podemos ver los estadísticos de la variable "Diabetes_012"
pd.value_counts(diagnostics_t['Diabetes_012'])
```

```
0    83271
2    13826
1     1879
```

2. Construcción de los modelos:

   KNN (Nicolás Segura): Para la construcción de este modelo se realizó la previa normalización de los datos con el fin de que las distancias fueran equiparables. Además, se utilizaron los siguientes hiperparámetros validados con el algoritmo k-fold-cross validation: **Minkowski**, **p**=1, **vecinos** =11.

   Árbol de decisión (Jaime Torres): Debido a los cambios realizados con anterioridad al conjunto de datos, no hubo necesidad de hacer uno adicional. Los hiperparámetros utilizados para la construcción del modelo también se validaron a través de k-fold-cross validation y fueron: **Entropy**, **mm**=15, **profundidad**=20.

   Cabe resaltar que, para la validación de los hiperparámetros, se realizaron repetidas ejecuciones a partir del código para poder encontrar la mejor alternativa.

Regresión lineal (David Ruiz):

Se realizó una regresión lineal en donde se modeló la relación entre una variable dependiente con un conjunto dado de variables independientes con los datos de entrenamiento como argumentos. En este caso, la variable dependiente fue “Diabetes_012” y las variables independientes el resto de las columnas del conjunto. Una vez realizado el proceso se utilizó la siguiente función obtenida a partir de la función sigmoide para la clasificación de los datos:

$$\geq 0.65904607 = 1$$
$$< 0.65904607 = 0$$

3. Comparación y selección de los modelos:

   Una vez obtenidos los resultados de la precisión y el puntaje f1 de cada uno de los modelos decidimos seleccionar el modelo de **Árbol de decisión**, ya que a pesar de que contaba con un 2% menos de precisión que el de **Regresión lineal**, este contaba con un mejor puntaje f1, lo que en este caso resulta de mucha más importancia ya que dicho indicador permite saber qué tan bueno es el modelo para evitar los falsos negativos; los cuales resultan ser de vital importancia para SaludAlpes para evitar dar diagnósticos que puedan perjudicar la salud del paciente por la omisión de un tratamiento temprano.

Presentación:
https://www.canva.com/design/DAE48NCmu00/1pS38p4Rkk3GPJRCtBCuoA/edit?utm_content=DAE48NCmu00&utm_campaign=designshare&utm_medium=link2&utm_source=sharebutton