Tecnológico Nacional de México Campus Hermosillo
Ingeniería en Sistemas Computacionales
Especialidad de Ciencia de Datos

Proyecto Final
Minería de Datos

Predicción de oportunidades de ataques al corazón

Mtro. Eduardo Antonio Hinojosa Palafox
Ei. Paul Isaac Torres Enriquez

Hermosillo, Sonora a día miércoles 14 de diciembre de 2022

Introducción. –

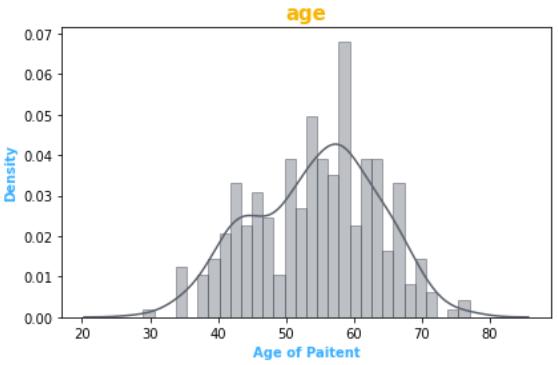
En el siguiente proyecto se llevará acabo la inspección y predicción de ataques al corazón (infarto de miocardio), esto pasa cuando una de las partes del musculo del corazón no bombea suficiente sangre. La mayor parte del tiempo esto pasa sin que la persona tenga un tratamiento de regulación de su flujo sanguíneo aumentando el daño hacia el musculo o en problemas cardiovasculares.

La razón común de un infarto de miocardio ocurre cuando el flujo de sangre del corazón se reduce gravemente o es bloqueada en su totalidad. El bloqueo se puede dar usualmente por el aumento de grasa o colesterol en las arterias del corazón.

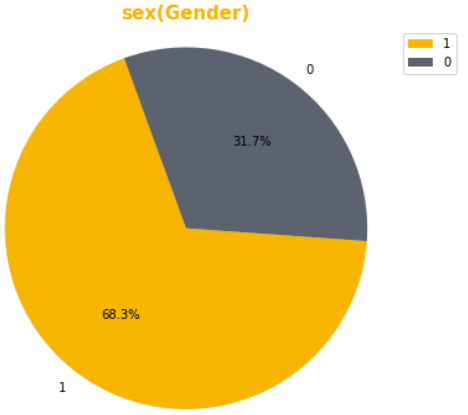
Descripción del problema a desarrollar. –

En base a los diferentes síntomas en hombres y mujeres se realizará la predicción del nivel de riesgo de ataque cardiaco.

Descripción del conjunto de datos. –



Como se puede ver en la siguiente gráfica la edad de población que está siendo analizada en este caso de estudio se encuentra en un rango de aproximadamente de 29 a 78 años, de los cuales el 68.3% de la población son hombres y el 31.7% son mujeres.

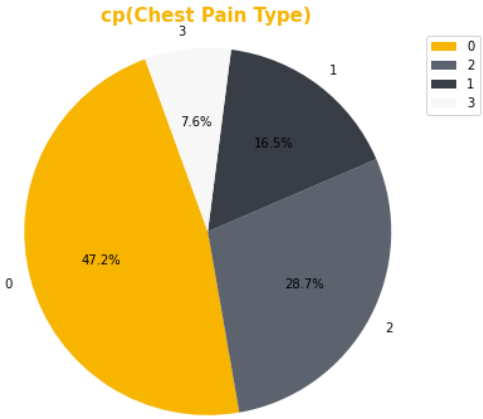


Por otra parte, tenemos el dolor torácico con el que los pacientes se presentan el cual consta de 4 valores

- 0 = Angina Típica
- 1 = Angina Atípica
- 2 = Dolor no Anginoso
- 3 = Asintomático

Una vez explicado esto, se puede presentar la siguiente gráfica de pastel la cual muestra la distribución de los datos en los 4 valores anteriores.

- 0 = 143 valores
- 1 = 87 valores
- 2 = 50 valores
- 3 = 23 valores



Una vez analizado esto, durante la exploración y análisis de los datos, nos mostró que el dataset se encuentra libre de datos nulos.

Descripción del código utilizado. –

Importamos las librerías necesarias para el trabajo del proyecto.

```
[3] import pickle
    import warnings
    import numpy as np #algebra linear
    import pandas as pd #manipulación y análisis de datos
    import seaborn as sns #Satistical Visualization
    import matplotlib.pyplot as plt #Visualización de datos

    warnings.filterwarnings('ignore')
```

Cargamos el dataset

```
[4] #cargamos el dataset en una variable
    df = pd.read_csv('/content/heart.csv')
    df
```

Cambiamos los nombres de las columnas del dataset

```
#cambio de nombre a columnas
new_columns = ["age","sex","cp","trtbps","chol","fbs","rest_ecg","thalach","exang","oldpeak","slope","ca","thal","target"]
df.columns = new_columns
df.tail()
```

	age	sex	cp	trtbps	chol	fbs	rest_ecg	thalach	exang	oldpeak	slope	ca	thal	target
298	57	0	0	140	241	0	1	123	1	0.2	1	0	3	0
299	45	1	3	110	264	0	1	132	0	1.2	1	0	3	0
300	68	1	0	144	193	1	1	141	0	3.4	1	2	3	0
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3	0
302	57	0	1	130	236	0	0	174	0	0.0	1	1	2	0

Visualizamos el tipo de objeto de nuestros datos

```
[9] #visualizamos la forma de tipo de objetos de nuestros datos
    df.info()

    <class 'pandas.core.frame.DataFrame'>
    RangeIndex: 303 entries, 0 to 302
    Data columns (total 14 columns):
     #   Column    Non-Null Count  Dtype
    ---  ------    --------------  -----
     0   age       303 non-null    int64
     1   sex       303 non-null    int64
     2   cp        303 non-null    int64
     3   trtbps    303 non-null    int64
     4   chol      303 non-null    int64
     5   fbs       303 non-null    int64
     6   rest_ecg  303 non-null    int64
     7   thalach   303 non-null    int64
     8   exang     303 non-null    int64
     9   oldpeak   303 non-null    float64
     10  slope     303 non-null    int64
     11  ca        303 non-null    int64
     12  thal      303 non-null    int64
     13  target    303 non-null    int64
    dtypes: float64(1), int64(13)
    memory usage: 33.3 KB
```

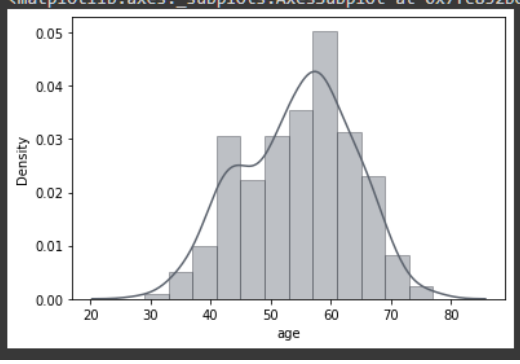
Separamos las variables categóricas de las numéricas para hacer una examinación estadística de los mismos.

```
[21] numerical = ["age", "trtbps", "chol", "thalach", "oldpeak"]
     categorical = ["sex", "cp", "fbs", "rest_ecg", "exang", "slope", "ca", "thal", "target"]

     df[numerical].describe()
```

	age	trtbps	chol	thalach	oldpeak
count	303.000000	303.000000	303.000000	303.000000	303.000000
mean	54.366337	131.623762	246.264026	149.646865	1.039604
std	9.082101	17.538143	51.830751	22.905161	1.161075
min	29.000000	94.000000	126.000000	71.000000	0.000000
25%	47.500000	120.000000	211.000000	133.500000	0.000000
50%	55.000000	130.000000	240.000000	153.000000	0.800000
75%	61.000000	140.000000	274.500000	166.000000	1.600000
max	77.000000	200.000000	564.000000	202.000000	6.200000

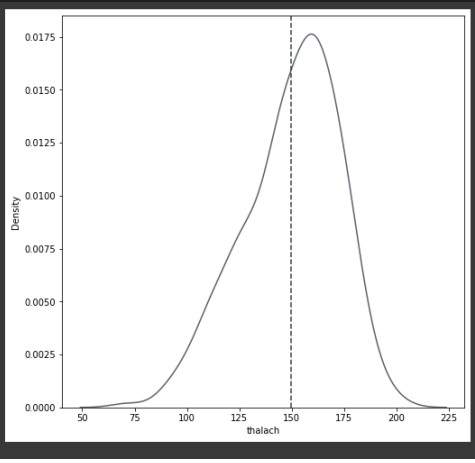
```
[22] sns.distplot(df["age"], hist_kws=dict(linewidth = 1, edgecolor = '#393E46'), color = '#5C636E')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc832be2a00>
```



Así mismo checamos el tipo de distribución que tenemos en el dataset

```
[25] x,y = plt.subplots(figsize = (8,8))
     sns.distplot(df['thalach'],hist = False, ax = y, color = '#5C636E')
     y.axvline(df['thalach'].mean(),color = "#393E46", ls = "--");
```



Checamos nuestras variables numéricas y las analizamos en conjunto al target del dataset

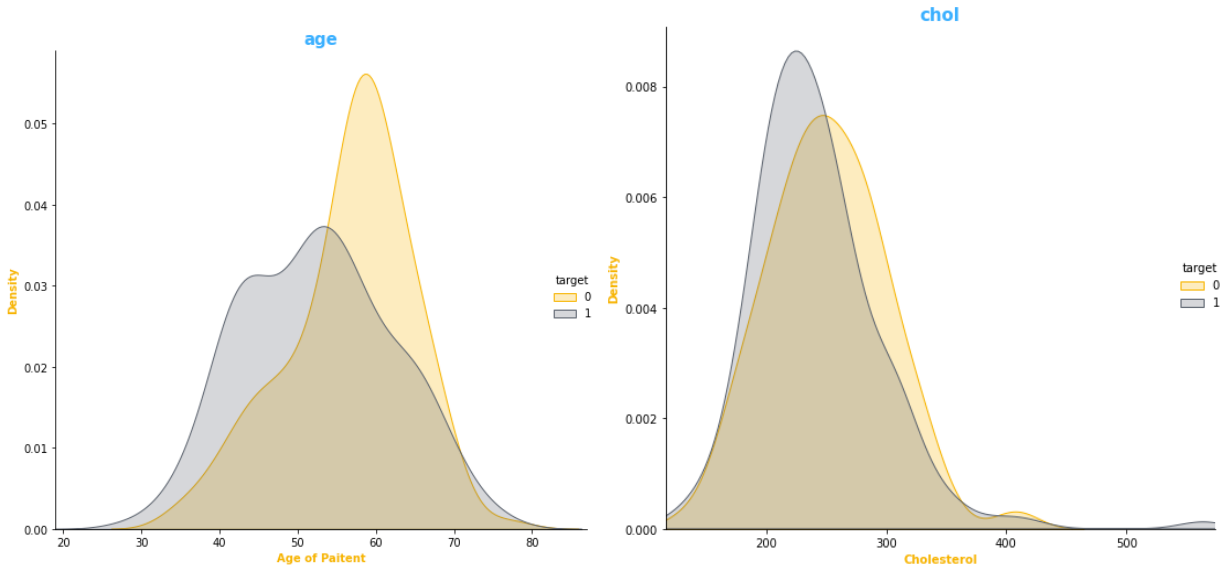
```
[39] numerical

['age', 'trtbps', 'chol', 'thalach', 'oldpeak']
```

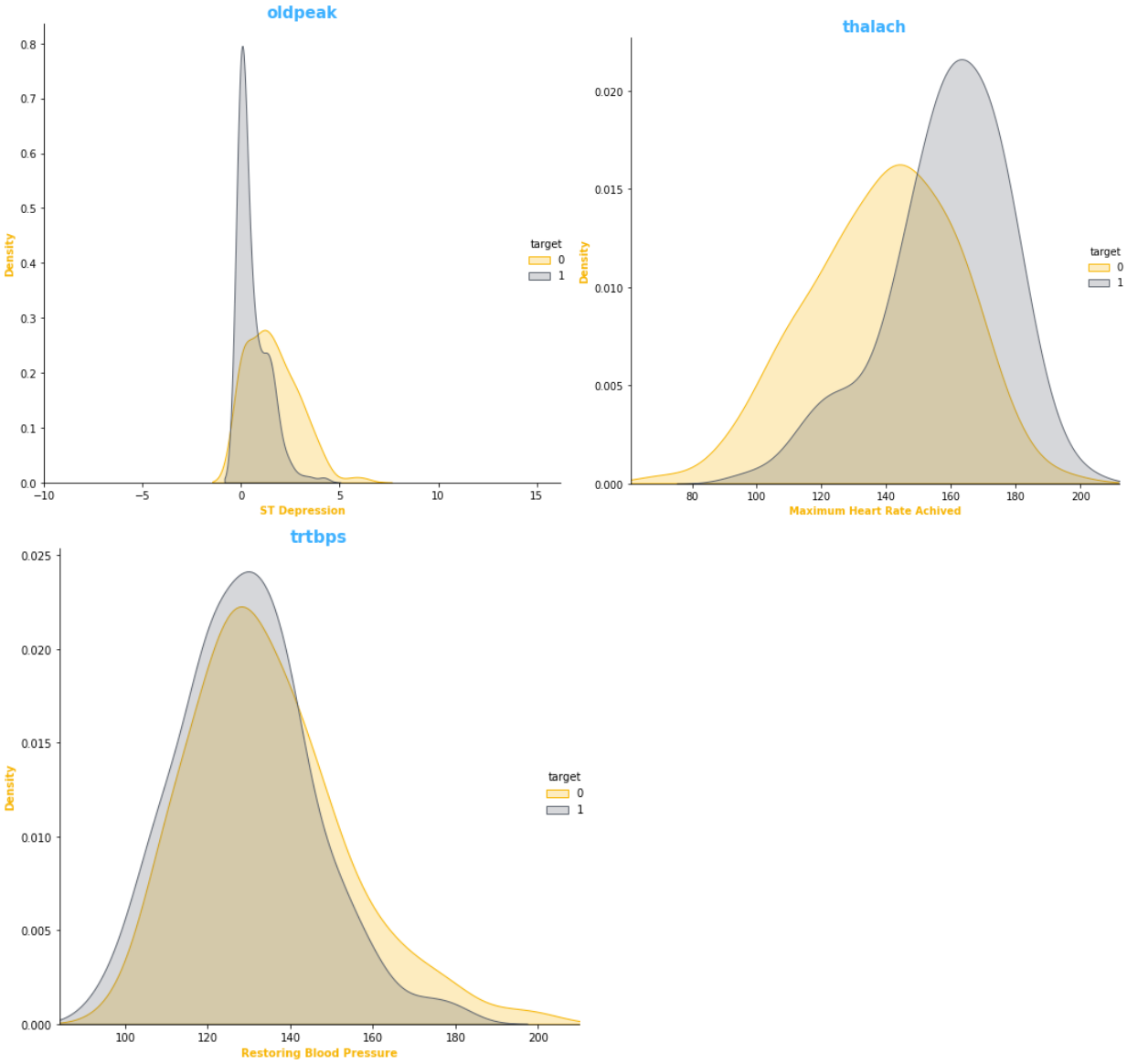
```
numerical.append("target") # Analizamos el target con todas las variables numericas
numerical
```

```
['age', 'trtbps', 'chol', 'thalach', 'oldpeak', 'target']
```

```
#Libreria Seaborn
title_font = {"family": "arial", "color":"#3AB0FF","weight":"bold","size":15}
axis_font = {"family": "arial", "color":"#F8B500","weight":"bold","size":10}
colors = ['#F8B500', '#5C636E', '#393E46', '#F7F7F7', '#3AB0FF']

for i,z in list(zip(numerical,numerical_axis)):
    graph = sns.FacetGrid(df[numerical], hue = "target", height =7,palette = colors ,xlim = (df[i].min()-10, (df[i].max() + 10)))
    graph.map(sns.kdeplot, i, shade = True)
    graph.add_legend()

    plt.title(i, fontdict = title_font)
    plt.xlabel(z, fontdict= axis_font)
    plt.ylabel("Density", fontdict = axis_font)

    plt.tight_layout()
    plt.show()
```

Como resultado del análisis obtenemos los siguientes gráficos





Separamos el dataframe en datos de prueba y entrenamiento

```
[48] from sklearn.model_selection import train_test_split

[49] X = dataframe_duplicate.drop(['target'], axis = 1)
     Y = dataframe_duplicate[['target']]

[50] X_Train, X_Test, Y_Train, Y_Test = train_test_split(X, Y, test_size = 0.2, random_state = 10)

[51] X_Test
```

	age	sex	cp	trtbps	chol	fbs	rest_ecg	thalach	exang	oldpeak	slope	ca	thal
246	56	0	0	134	409	0	0	150	1	1.9	1	2	3
183	58	1	2	112	230	0	0	165	0	2.5	1	1	3
229	64	1	2	125	309	0	1	131	1	1.8	1	0	3
126	47	1	0	112	204	0	1	143	0	0.1	2	0	2
184	50	1	0	150	243	0	0	128	0	2.6	1	0	3
...	...	...	...	...	...	...	...	...	...	...	...	...	...
277	57	1	1	124	261	0	1	141	0	0.3	2	0	3
121	59	1	0	138	271	0	0	182	0	0.0	2	0	2
187	54	1	0	124	266	0	0	109	1	2.2	1	1	3
301	57	1	0	130	131	0	1	115	1	1.2	1	1	3
283	40	1	0	152	223	0	1	181	0	0.0	2	0	3

61 rows × 13 columns

```
[52] X_Train.head()
```

	age	sex	cp	trtbps	chol	fbs	rest_ecg	thalach	exang	oldpeak	slope	ca	thal
286	59	1	3	134	204	0	1	162	0	0.8	2	2	2
102	63	0	1	140	195	0	1	179	0	0.0	2	2	2
242	64	1	0	145	212	0	0	132	0	2.0	1	2	1
65	35	0	0	138	183	0	1	182	0	1.4	2	0	2
35	46	0	2	142	177	0	0	160	1	1.4	0	0	2

```
[53] Y_Train.head()
```

	target
286	0
102	1
242	0
65	1
35	1

Realizamos el procesamiento de datos con técnicas utilizadas en clase como la técnica de regresión logística y árboles de decisión

```
[55] from sklearn.linear_model import LogisticRegression
     from sklearn.metrics import accuracy_score
```

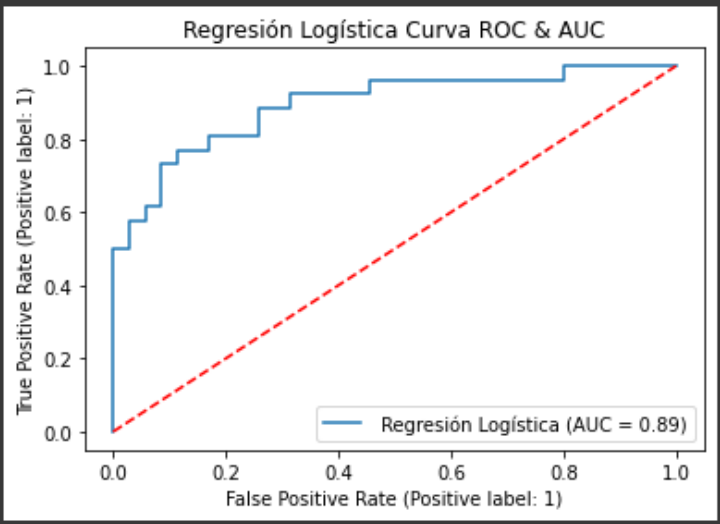
```
[56] Logistic_Regression = LogisticRegression()
     Logistic_Regression.fit(X_Train,Y_Train)
     Y_Predicted = Logistic_Regression.predict(X_Test)
     Y_Predicted
```

```
     array([0, 1, 0, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0, 1,
            1, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 1, 1,
            1, 1, 1, 1, 1, 1, 0, 1, 0, 0, 1, 1, 1, 1, 0, 0, 1])
```

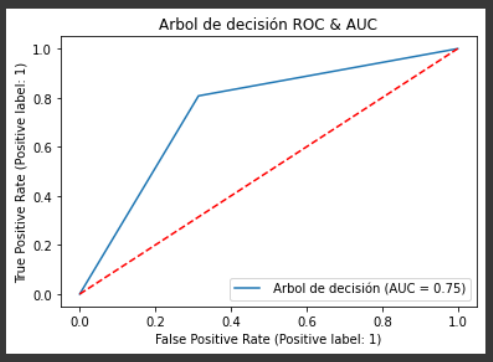
```
[57] Accuracy = accuracy_score(Y_Test,Y_Predicted)
     print("Precisión de la prueba:{}".format(Accuracy))
```

```
     Precisión de la prueba:0.7704918032786885
```

```
[58] from sklearn.metrics import plot_roc_curve
     plot_roc_curve(Logistic_Regression,X_Test,Y_Test, name =" Regresión Logística")
     plt.title("Regresión Logística Curva ROC & AUC")
     plt.plot([0,1],[0,1],"r--")
     plt.show()
```

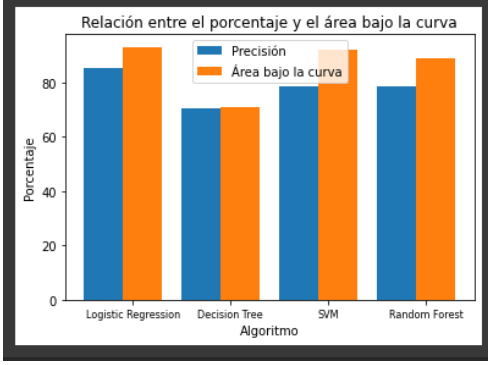


```
[59] from sklearn.tree import DecisionTreeClassifier
     Decision_Tree = DecisionTreeClassifier(random_state = 10)
     Decision_Tree.fit(X_Train,Y_Train)
     Y_Prediction = Decision_Tree.predict(X_Test)
     print("Precisión de prueba del arbol de decisión: ", accuracy_score(Y_Test,Y_Prediction))

     Precisión de prueba del arbol de decisión:  0.7377049180327869

[60] plot_roc_curve(Decision_Tree,X_Test,Y_Test, name =" Arbol de decisión")
     plt.title("Arbol de decisión ROC & AUC")
     plt.plot([0,1],[0,1],"r--")
     plt.show()
```



Por ultimo procedemos a comparar cada algoritmo usado y su porcentaje de precisión en la estimación de la probabilidad que tienen de tener un ataque cardiaco en base a la sintomatología presentada.

```
algorithm =['Logistic Regression', 'Decision Tree', 'SVM','Random Forest']
accuracy = [85.24, 70.49, 78.68,78.68]
auc = [93,71,92,89]
X_axis = np.arange(len(algorithm))

plt.bar(X_axis - 0.1, accuracy, 0.8, label = 'Precisión')
plt.bar(X_axis + 0.1, auc, 0.4, label = 'Área bajo la curva')


plt.xticks(X_axis, algorithm,size = 8)
plt.xlabel("Algoritmo")
plt.ylabel("Porcentaje")
plt.title("Relación entre el porcentaje y el área bajo la curva")
plt.legend()
plt.show()
```



Conclusión. –

Como conclusión, se pudo generar conocimiento en base a toda la información en bruto como procesada utilizando diferentes técnicas de minería de datos y como base de nuestro caso de estudio se puede concluir que el 54.5% de personas con sintomatología entra en el grupo de personas que más probabilidad tienen de sufrir un ataque cardiaco gracias a la sintomatología presentada. En su contraparte, el 45.5% se encuentra con menos probabilidad de sufrir un ataque cardiaco.

Referencias. –

- https://www.kaggle.com/code/sing05jatin/predection-and-analysis-of-heart-attack
- https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset

Liga de Github. –

https://github.com/PaulIsaacTorres/Mineria-de-Datos