UNIVERSIDAD DE ANTIOQUIA

Desarrollo de un sistema de inteligencia artificial para la detección de apendicitis sobre ecografía de abdomen.

Rafael Reyes Velasquez

Resumen

Este proyecto de investigación se enfoca en abordar la detección oportuna de apendicitis, una urgencia médica común, a través de la aplicación de inteligencia artificial a la interpretación de imágenes de ecografías de abdomen. La iniciativa busca proporcionar al personal médico una herramienta precisa y eficiente que permita acelerar el proceso de diagnóstico, reducir errores humanos y, en última instancia, mejorar la calidad de la atención médica. La propuesta se basa en la revisión de la literatura existente, la recopilación y preparación de datos basados en imágenes, el desarrollo de un modelo de inteligencia artificial y su integración en un sistema de apoyo a la decisión clínica. El alcance del proyecto se limita a la detección de apendicitis y su interpretación en ecografías de abdomen, con un enfoque claro en la validación clínica y la mejora continua del sistema. El impacto potencial de esta innovación en el ámbito de la atención médica es significativo, ya que puede acelerar el proceso de diagnóstico y mejorar la precisión, lo que beneficia tanto a los profesionales de la salud encargados de la atención a pacientes, como a los mismos pacientes, previniendo sea el caso complicaciones de esta patología que puede ocasionar la muerte u otros agravamientos médicos.

Objetivo general

Desarrollar un sistema basado en inteligencia artificial que sirva como herramienta de apoyo para el personal médico en la detección precisa de apendicitis mediante el análisis de imágenes de ecografías de abdomen.

Objetivos específicos

- Desarrollar algoritmos de inteligencia artificial que mejoren la precisión en la detección de casos de apendicitis en ecografías pediátricas.
- Establecer métricas específicas, como la sensibilidad y especificidad, para evaluar y ajustar continuamente el rendimiento del sistema en términos de precisión diagnóstica.
- Realizar sesiones de capacitación con el personal médico para asegurar una adopción efectiva del sistema, promoviendo su utilización como una herramienta de apoyo integral en el proceso de diagnóstico de apendicitis.
- Implementar estrategias de optimización de procesos para reducir significativamente el tiempo necesario para realizar el diagnóstico de apendicitis utilizando el sistema de inteligencia artificial.

Origen de los datos

Se hará uso de una base de datos pública proporcionada por el Hospital St. Hedwig en Regensburg, Alemania. Esta base de datos fue creada a partir de un estudio que recopiló y organizó información sobre una cohorte de pacientes pediátricos que presentaban dolor abdominal entre 2016 a 2021.

Es importante aclarar que la información fue validada por el personal del hospital alemán anteriormente mencionado; asimismo, el estudio que publica los datos fue aprobado por el comité de ética de la Universidad de Regensburg, se llevó a cabo según las regulaciones y pautas de dicha universidad y fue finalmente publicado a través del repositorio de datos de acceso abierto respaldado por el CERN, Zenodo, bajo la licencia de Creative Commons para usos no comerciales.

En este dataset se encuentran varios archivos en diferentes formatos que según sus creadores, Marcinkevičs et al. (2023), corresponden a imágenes de ecografía abdominal en modo B que "muestran diversas regiones de interés, como el cuadrante inferior derecho del abdomen, el apéndice, los intestinos, los ganglios linfáticos y los órganos reproductores", además de conjuntos de datos que "incluyen información sobre pruebas de laboratorio, resultados de la exploración física, puntuaciones clínicas, como las puntuaciones de Alvarado y de apendicitis pediátrica, y resultados ecográficos elaborados por expertos".

Nuevamente, citando directamente a Marcinkevičs et al. (2023), el dataset posee la siguiente estructura:

"US_Pictures/: carpeta con las imágenes originales de ecografía en modo B en formato BMP; las imágenes se denominan como <subject #>.<view #> *.bmp". Se poseen 2092 sonogramas de 780 pacientes.

"app_data.xlsx: Archivo MS Excel con datos tabulares (la pestaña 'Resumen de datos' contiene una explicación de las variables); los números de asunto correspondientes de la carpeta US_Pictures/ se encuentran en la columna US_Number"

"multiple_in_one_: una lista de nombres de imágenes de ultrasonido que contienen varias instantáneas"

"test_set_codes_csv: una lista de puntos de datos del conjunto de prueba"

Según lo anterior, se estima que los datos son confiables, además de ser la única base de datos pública de este tipo que se pudo encontrar; sin embargo, esto no descarta una exploración más exhaustiva sobre el dataset para encontrar posibles fallas o falencias.

Datos Estructurados

Estos corresponden al archivo app_data.xlsx, en el repositorio de GitHub.

https://github.com/mamut1485/ProyectoS.git

Importar librerias de python

```
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
```

```
# Ruta del archivo
file_path ='https://raw.githubusercontent.com/mamut1485/ProyectoS/main/structured_data/app_data.xlsx'

# Cargar la primera hoja del archivo Excel en un DataFrame
df = pd.read_excel(file_path, sheet_name=0)
```

```
# Mostrar las primeras filas del DataFrame
df.head()
```

```
##      Age   BMI     Sex  ...  Meteorism  Enteritis  Gynecological_Findings
## 0  12.68  16.9  female  ...        NaN        NaN                     NaN
## 1  14.10  31.9    male  ...        yes        NaN                     NaN
## 2  14.14  23.3  female  ...        yes        yes                     NaN
## 3  16.37  20.6  female  ...        NaN        yes                     NaN
## 4  11.08  16.9  female  ...        NaN        yes                     NaN
##
## [5 rows x 58 columns]
```

```
# Visualizamos las variables contenidas en el conjunto de datos
df.info()
```

```
## <class 'pandas.core.frame.DataFrame'>
## RangeIndex: 782 entries, 0 to 781
## Data columns (total 58 columns):
##  #   Column                                Non-Null Count  Dtype
## ---  ------                                --------------  -----
##  0   Age                                   781 non-null    float64
##  1   BMI                                   755 non-null    float64
##  2   Sex                                   780 non-null    object
##  3   Height                                756 non-null    float64
##  4   Weight                                779 non-null    float64
##  5   Length_of_Stay                        778 non-null    float64
##  6   Management                            781 non-null    object
##  7   Severity                              781 non-null    object
##  8   Diagnosis_Presumptive                 780 non-null    object
##  9   Diagnosis                             780 non-null    object
##  10  Alvarado_Score                        730 non-null    float64
##  11  Paedriatic_Appendicitis_Score         730 non-null    float64
##  12  Appendix_on_US                        777 non-null    object
##  13  Appendix_Diameter                     498 non-null    float64
##  14  Migratory_Pain                        773 non-null    object
##  15  Lower_Right_Abd_Pain                  774 non-null    object
##  16  Contralateral_Rebound_Tenderness      767 non-null    object
##  17  Coughing_Pain                         766 non-null    object
##  18  Nausea                                774 non-null    object
##  19  Loss_of_Appetite                      772 non-null    object
##  20  Body_Temperature                      775 non-null    float64
##  21  WBC_Count                             776 non-null    float64
##  22  Neutrophil_Percentage                 679 non-null    float64
##  23  Segmented_Neutrophils                 54 non-null     float64
##  24  Neutrophilia                          732 non-null    object
##  25  RBC_Count                             764 non-null    float64
##  26  Hemoglobin                            764 non-null    float64
##  27  RDW                                   756 non-null    float64
##  28  Thrombocyte_Count                     764 non-null    float64
##  29  Ketones_in_Urine                      582 non-null    object
##  30  RBC_in_Urine                          576 non-null    object
##  31  WBC_in_Urine                          583 non-null    object
##  32  CRP                                   771 non-null    float64
```

```
##  33  Dysuria                          753 non-null    object
##  34  Stool                            765 non-null    object
##  35  Peritonitis                      773 non-null    object
##  36  Psoas_Sign                       745 non-null    object
##  37  Ipsilateral_Rebound_Tenderness   619 non-null    object
##  38  US_Performed                     778 non-null    object
##  39  US_Number                        760 non-null    float64
##  40  Free_Fluids                      719 non-null    object
##  41  Appendix_Wall_Layers             218 non-null    object
##  42  Target_Sign                      138 non-null    object
##  43  Appendicolith                    69 non-null     object
##  44  Perfusion                        63 non-null     object
##  45  Perforation                      81 non-null     object
##  46  Surrounding_Tissue_Reaction      252 non-null    object
##  47  Appendicular_Abscess             85 non-null     object
##  48  Abscess_Location                 13 non-null     object
##  49  Pathological_Lymph_Nodes         203 non-null    object
##  50  Lymph_Nodes_Location             121 non-null    object
##  51  Bowel_Wall_Thickening            99 non-null     object
##  52  Conglomerate_of_Bowel_Loops      43 non-null     object
##  53  Ileus                            60 non-null     object
##  54  Coprostasis                      71 non-null     object
##  55  Meteorism                        140 non-null    object
##  56  Enteritis                        66 non-null     object
##  57  Gynecological_Findings           26 non-null     object
## dtypes: float64(18), object(40)
## memory usage: 354.5+ KB
```

Exploración de datos perdidos

```
"""
Se identifican los valores vacíos para cada variable en las instancias.
Cantidad de Datos perdidos por variables, uasamos la funcion head para visualizar
los primeros 5 valores y ahorar espacio:
"""
```

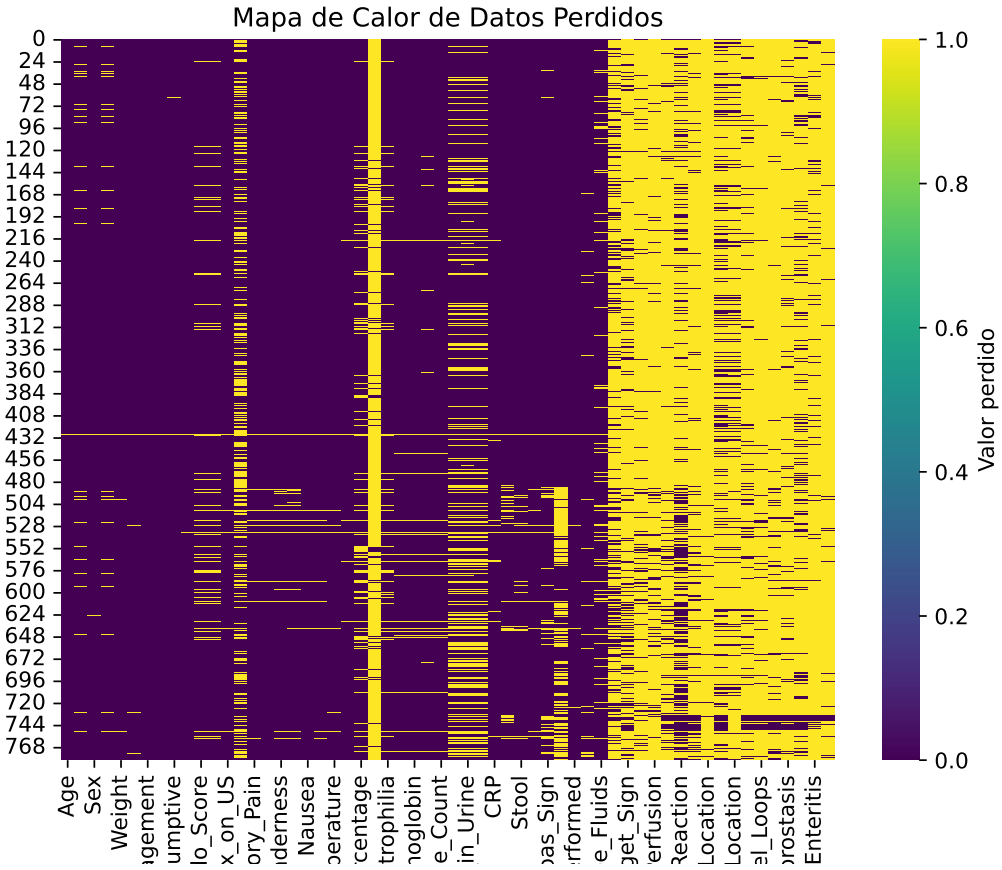
```
df.isnull().sum().head()
```

```
## Age        1
## BMI       27
## Sex        2
## Height    26
## Weight     3
## dtype: int64
```

Mapa de Calor de datos perdidos

```
"""
Utilizamos un mapa de calor para visualizar los datos perdidos de las variables
contenidas en el conjunto de datos.
"""
```

```
# Mapa de calor para visualizar datos perdidos
plt.figure(figsize=(8, 6))
sns.heatmap(df.isnull(), cmap='viridis', cbar=True,
cbar_kws={'label': 'Valor perdido'})
plt.title('Mapa de Calor de Datos Perdidos')
plt.show()
```



Variables Categóricas

```
# Identificación de las variables categóricas
categoricas = df.select_dtypes(include = ["object"]).columns.tolist()
df[categoricas]
```

```
##         Sex          Management  ... Enteritis Gynecological_Findings
## 0    female        conservative  ...       NaN                    NaN
## 1      male        conservative  ...       NaN                    NaN
## 2    female        conservative  ...       yes                    NaN
## 3    female        conservative  ...       yes                    NaN
## 4    female        conservative  ...       yes                    NaN
## ..      ...                 ...  ...       ...                    ...
```

```
## 777  female    primary surgical  ...       NaN                   NaN
## 778  female  secondary surgical  ...       NaN                   NaN
## 779  female    primary surgical  ...       NaN           unauffällig
## 780    male    primary surgical  ...       NaN                   NaN
## 781    male    primary surgical  ...       NaN                   NaN
##
## [782 rows x 40 columns]
```

Variables Numéricas

```
# Identificación de las varibles numéricas
numericas = df.select_dtypes(include = ['float64','int64']).columns.tolist()
df[numericas]
```

```
##            Age        BMI  Height  ...  Thrombocyte_Count    CRP  US_Number
## 0    12.680000  16.900000   148.0  ...              254.0    0.0      882.0
## 1    14.100000  31.900000   147.0  ...              151.0    3.0      883.0
## 2    14.140000  23.300000   163.0  ...              300.0    3.0      884.0
## 3    16.370000  20.600000   165.0  ...              258.0    0.0      886.0
## 4    11.080000  16.900000   163.0  ...              311.0    0.0      887.0
## ..         ...        ...     ...  ...                ...    ...        ...
## 777  12.413415  25.250476   166.5  ...              243.0   71.0      126.0
## 778  17.092402  20.429418   158.0  ...              310.0  245.0        NaN
## 779  14.992471  19.909972   152.0  ...              328.0    2.0      127.0
## 780   7.195072  14.295549   129.3  ...              345.0    8.0      128.0
## 781  11.509925  18.171441   146.5  ...              291.0    1.0      129.0
##
## [782 rows x 18 columns]
```

Normalizacion de datos perdidos

```
# Usando la funcion fillna(0), realizamos una normalizacion de los datos perdidos.

df['Age']= df ['Age'].fillna(0)
df['BMI']= df ['BMI'].fillna(0)
df['Sex']= df ['Sex'].fillna(0)
df['Height']= df ['Height'].fillna(0)
df['Weight']= df ['Weight'].fillna(0)
df['Length_of_Stay']= df ['Length_of_Stay'].fillna(0)
df['Management']= df ['Management'].fillna(0)
df['Severity']= df ['Severity'].fillna(0)
df['Diagnosis_Presumptive']= df ['Diagnosis_Presumptive'].fillna(0)
df['Diagnosis']= df ['Diagnosis'].fillna(0)
df['Alvarado_Score']= df ['Alvarado_Score'].fillna(0)
df['Paedriatic_Appendicitis_Score']= df ['Paedriatic_Appendicitis_Score'].fillna(0)
df['Appendix_on_US']= df ['Appendix_on_US'].fillna(0)
df['Appendix_Diameter']= df ['Appendix_Diameter'].fillna(0)
df['Migratory_Pain']= df ['Migratory_Pain'].fillna(0)
df['Lower_Right_Abd_Pain']= df ['Lower_Right_Abd_Pain'].fillna(0)
df['Contralateral_Rebound_Tenderness']= df ['Contralateral_Rebound_Tenderness'].fillna(0)
```

```
df['Coughing_Pain']= df ['Coughing_Pain'].fillna(0)
df['Nausea']= df ['Nausea'].fillna(0)
df['Loss_of_Appetite']= df ['Loss_of_Appetite'].fillna(0)
df['Body_Temperature']= df ['Body_Temperature'].fillna(0)
df['WBC_Count']= df ['WBC_Count'].fillna(0)
df['Neutrophil_Percentage']= df ['Neutrophil_Percentage'].fillna(0)
df['Segmented_Neutrophils']= df ['Segmented_Neutrophils'].fillna(0)
df['Neutrophilia']= df ['Neutrophilia'].fillna(0)
df['RBC_Count']= df ['RBC_Count'].fillna(0)
df['Hemoglobin']= df ['Hemoglobin'].fillna(0)
df['RDW']= df ['RDW'].fillna(0)
df['Thrombocyte_Count']= df ['Thrombocyte_Count'].fillna(0)
df['Ketones_in_Urine']= df ['Ketones_in_Urine'].fillna(0)
df['RBC_in_Urine']= df ['RBC_in_Urine'].fillna(0)
df['WBC_in_Urine']= df ['WBC_in_Urine'].fillna(0)
df['CRP']= df ['CRP'].fillna(0)
df['Dysuria']= df ['Dysuria'].fillna(0)
df['Stool']= df ['Stool'].fillna(0)
df['Peritonitis']= df ['Peritonitis'].fillna(0)
df['Psoas_Sign']= df ['Psoas_Sign'].fillna(0)
df['Ipsilateral_Rebound_Tenderness']= df ['Ipsilateral_Rebound_Tenderness'].fillna(0)
df['US_Performed']= df ['US_Performed'].fillna(0)
df['US_Number']= df ['US_Number'].fillna(0)
df['Free_Fluids']= df ['Free_Fluids'].fillna(0)
df['Appendix_Wall_Layers']= df ['Appendix_Wall_Layers'].fillna(0)
df['Target_Sign']= df ['Target_Sign'].fillna(0)
df['Appendicolith']= df ['Appendicolith'].fillna(0)
df['Perfusion']= df ['Perfusion'].fillna(0)
df['Perforation']= df ['Perforation'].fillna(0)
df['Surrounding_Tissue_Reaction']= df ['Surrounding_Tissue_Reaction'].fillna(0)
df['Appendicular_Abscess']= df ['Appendicular_Abscess'].fillna(0)
df['Abscess_Location']= df ['Abscess_Location'].fillna(0)
df['Pathological_Lymph_Nodes']= df ['Pathological_Lymph_Nodes'].fillna(0)
df['Lymph_Nodes_Location']= df ['Lymph_Nodes_Location'].fillna(0)
df['Bowel_Wall_Thickening']= df ['Bowel_Wall_Thickening'].fillna(0)
df['Conglomerate_of_Bowel_Loops']= df ['Conglomerate_of_Bowel_Loops'].fillna(0)
df['Ileus']= df ['Ileus'].fillna(0)
df['Coprostasis']= df ['Coprostasis'].fillna(0)
df['Meteorism']= df ['Meteorism'].fillna(0)
df['Enteritis']= df ['Enteritis'].fillna(0)
df['Gynecological_Findings']= df ['Gynecological_Findings'].fillna(0)

df
```

```
##            Age        BMI     Sex  ...  Meteorism  Enteritis  Gynecological_Findings
## 0    12.680000  16.900000  female  ...          0          0                       0
## 1    14.100000  31.900000    male  ...        yes          0                       0
## 2    14.140000  23.300000  female  ...        yes        yes                       0
## 3    16.370000  20.600000  female  ...          0        yes                       0
## 4    11.080000  16.900000  female  ...          0        yes                       0
## ..         ...        ...     ...  ...        ...        ...                     ...
## 777  12.413415  25.250476  female  ...          0          0                       0
## 778  17.092402  20.429418  female  ...          0          0                       0
```

```
## 779  14.992471  19.909972  female  ...          0          0             unauffällig
## 780   7.195072  14.295549    male  ...          0          0                       0
## 781  11.509925  18.171441    male  ...          0          0                       0
##
## [782 rows x 58 columns]
```

```
#Rastreamos nuevamente los datos perdidos, No existen datos perdidos.
df.isnull().sum().head()
```

```
## Age       0
## BMI       0
## Sex       0
## Height    0
## Weight    0
## dtype: int64
```

Conversion a enteros bulianos las variables categoricas

```
"""
Dentro del conjunto de datos se encuentran variables de tipo categoricas, que
cuenta con datos yes, no y 0, se procede a convertirlas en numericas remplazando
los valores por 0 y 1 , respectivamente.
"""
```

```
df['Migratory_Pain'] = df['Migratory_Pain'].replace("yes", '1').replace("no", '0')
df['Lower_Right_Abd_Pain']= df ['Lower_Right_Abd_Pain'].replace("yes",'1').replace("no",'0')
df['Contralateral_Rebound_Tenderness']= df ['Contralateral_Rebound_Tenderness'].replace("yes",'1').repla
df['Coughing_Pain']= df ['Coughing_Pain'].replace("yes",'1').replace("no",'0')
df['Nausea']= df ['Nausea'].replace("yes",'1').replace("no",'0')
df['Loss_of_Appetite']= df ['Loss_of_Appetite'].replace("yes",'1').replace("no",'0')
df['Neutrophilia']= df ['Neutrophilia'].replace("yes",'1').replace("no",'0')
df['Dysuria']= df ['Dysuria'].replace("yes",'1').replace("no",'0')
df['Psoas_Sign']= df ['Psoas_Sign'].replace("yes",'1').replace("no",'0')
df['Ipsilateral_Rebound_Tenderness']= df ['Ipsilateral_Rebound_Tenderness'].replace("yes",'1').replace(
df['US_Performed']= df ['US_Performed'].replace("yes",'1').replace("no",'0')
df['Free_Fluids']= df ['Free_Fluids'].replace("yes",'1').replace("no",'0')
df['Target_Sign']= df ['Target_Sign'].replace("yes",'1').replace("no",'0')
df['Surrounding_Tissue_Reaction']= df ['Surrounding_Tissue_Reaction'].replace("yes",'1').replace("no",'
df['Bowel_Wall_Thickening']= df ['Bowel_Wall_Thickening'].replace("yes",'1').replace("no",'0')
df['Conglomerate_of_Bowel_Loops']= df ['Conglomerate_of_Bowel_Loops'].replace("yes",'1').replace("no",'
df['Ileus']= df ['Ileus'].replace("yes",'1').replace("no",'0')
df['Coprostasis']= df ['Coprostasis'].replace("yes",'1').replace("no",'0')
df['Meteorism']= df ['Meteorism'].replace("yes",'1').replace("no",'0')
df['Enteritis']= df ['Enteritis'].replace("yes",'1').replace("no",'0')
df['Pathological_Lymph_Nodes']= df ['Pathological_Lymph_Nodes'].replace("yes",'1').replace("no",'0')

df
```

```
##           Age        BMI     Sex  ...  Meteorism  Enteritis  Gynecological_Findings
## 0    12.680000  16.900000  female  ...          0          0                       0
## 1    14.100000  31.900000    male  ...          1          0                       0
```

```
## 2    14.140000  23.300000  female  ...          1          1                      0
## 3    16.370000  20.600000  female  ...          0          1                      0
## 4    11.080000  16.900000  female  ...          0          1                      0
## ..         ...        ...     ...  ...        ...        ...                    ...
## 777  12.413415  25.250476  female  ...          0          0                      0
## 778  17.092402  20.429418  female  ...          0          0                      0
## 779  14.992471  19.909972  female  ...          0          0             unauffällig
## 780   7.195072  14.295549    male  ...          0          0                      0
## 781  11.509925  18.171441    male  ...          0          0                      0
##
## [782 rows x 58 columns]
```

Depuracion de registros sin ID

```
"""
Se borran los datos con los cuales no se tiene una relacion con imagenes
diagnosticas tomadas, estos se reflejan en cero en la columna US_Number,
registros que no cuentan con imagenes relacionadas con el estudio.
"""
```

```
df = df[df['US_Number'] != 0]
```

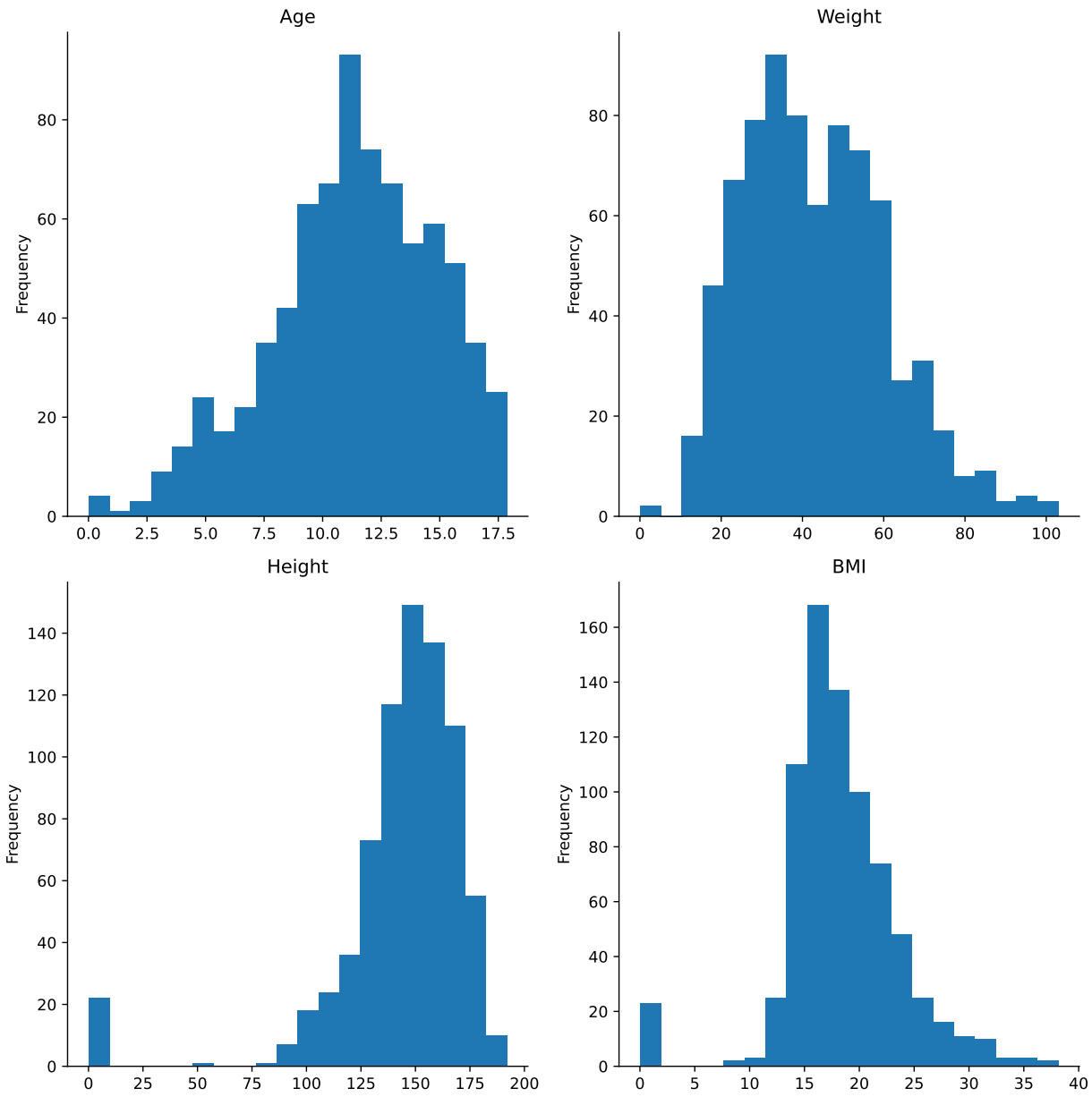
```
"""
Procedemos a limpiar el conjunto de datos, para quedarlos con las variables
definitivas del conjunto de datos, creamos un nuevo dataset con el conjunto
de datos definitico.
"""
```

```
df2 = df[['Age','BMI','Sex','Height','Weight','Length_of_Stay','Management',
          'Severity','Diagnosis_Presumptive','Diagnosis',
          'Appendix_on_US','Migratory_Pain','US_Number',]]

df2
```

```
##           Age        BMI     Sex  ...  Appendix_on_US  Migratory_Pain  US_Number
## 0    12.680000  16.900000  female  ...             yes               0      882.0
## 1    14.100000  31.900000    male  ...              no               1      883.0
## 2    14.140000  23.300000  female  ...              no               0      884.0
## 3    16.370000  20.600000  female  ...              no               1      886.0
## 4    11.080000  16.900000  female  ...             yes               0      887.0
## ..         ...        ...     ...  ...             ...             ...        ...
## 775   7.882272  18.698225    male  ...             yes               0      244.0
## 777  12.413415  25.250476  female  ...             yes               1      126.0
## 779  14.992471  19.909972  female  ...              no               0      127.0
## 780   7.195072  14.295549    male  ...             yes               1      128.0
## 781  11.509925  18.171441    male  ...             yes               0      129.0
##
## [760 rows x 13 columns]
```

Visualizacion de Distribuciones

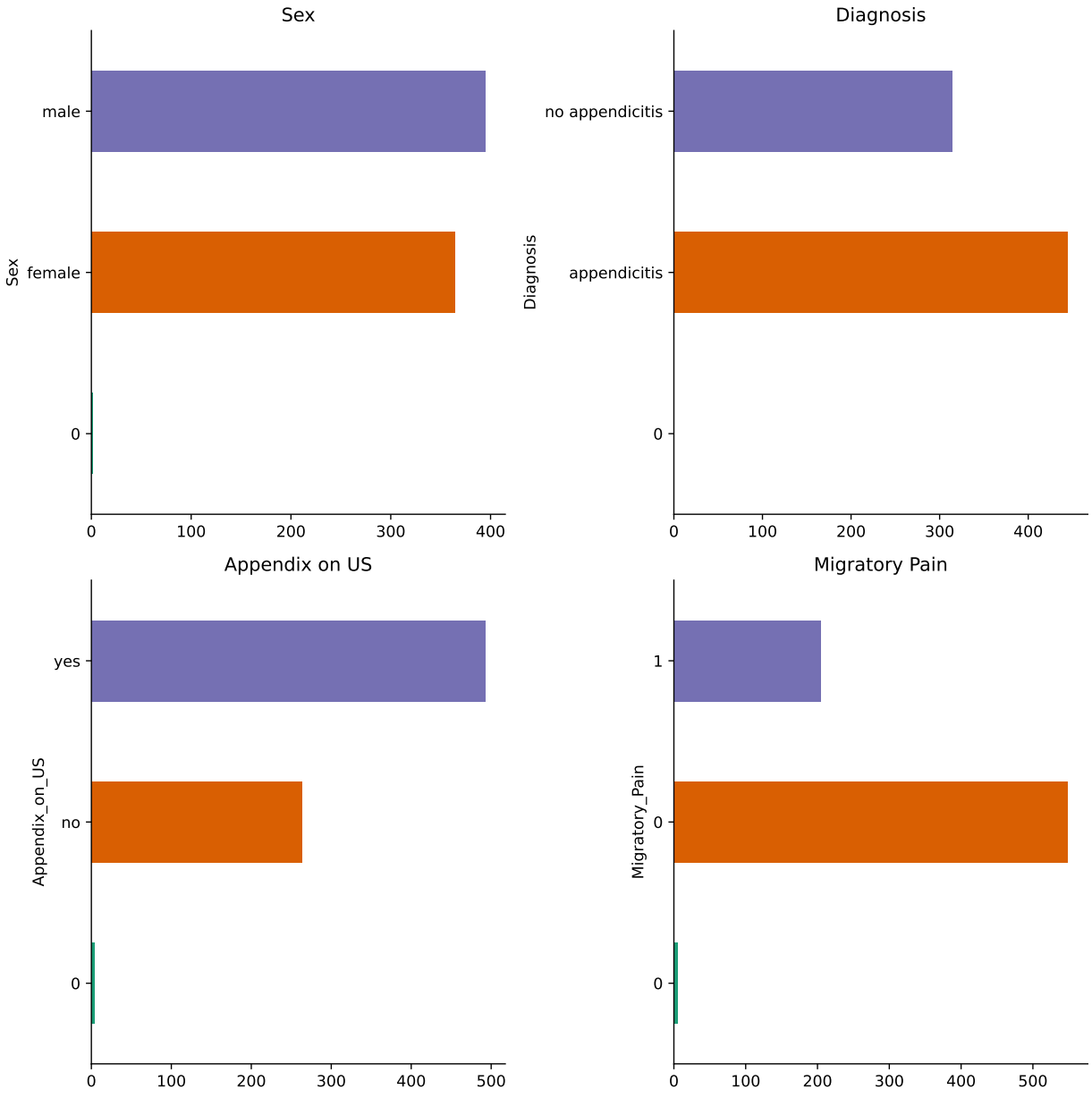
```
# Distribuciones
fig, axs = plt.subplots(2, 2, figsize=(10, 10))

df2['Age'].plot(kind='hist', bins=20, ax=axs[0, 0], title='Age')
axs[0, 0].spines[['top', 'right']].set_visible(False)

df2['Weight'].plot(kind='hist', bins=20, ax=axs[0, 1], title='Weight')
axs[0, 1].spines[['top', 'right']].set_visible(False)

df2['Height'].plot(kind='hist', bins=20, ax=axs[1, 0], title='Height')
axs[1, 0].spines[['top', 'right']].set_visible(False)

df2['BMI'].plot(kind='hist', bins=20, ax=axs[1, 1], title='BMI')
axs[1, 1].spines[['top', 'right']].set_visible(False)

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



Visializacion Variables Categoricas

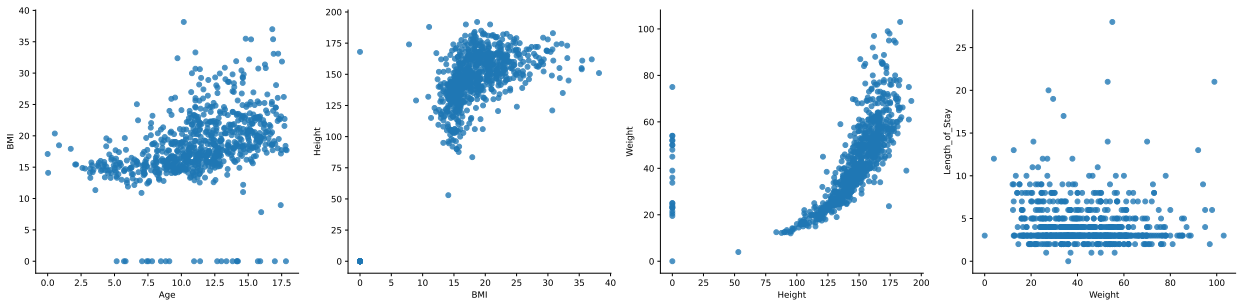
```
# Variables Categoricas
fig, axs = plt.subplots(2, 2, figsize=(10, 10))

df2.groupby('Sex').size().plot(kind='barh', color=sns.color_palette('Dark2'), ax=axs[0, 0])
axs[0, 0].spines[['top', 'right']].set_visible(False)
axs[0, 0].set_title('Sex')

df2.groupby('Diagnosis').size().plot(kind='barh', color=sns.color_palette('Dark2'), ax=axs[0, 1])
axs[0, 1].spines[['top', 'right']].set_visible(False)
axs[0, 1].set_title('Diagnosis')
```

```
df2.groupby('Appendix_on_US').size().plot(kind='barh', color=sns.color_palette('Dark2'), ax=axs[1, 0])
axs[1, 0].spines[['top', 'right']].set_visible(False)
axs[1, 0].set_title('Appendix on US')

df2.groupby('Migratory_Pain').size().plot(kind='barh', color=sns.color_palette('Dark2'), ax=axs[1, 1])
axs[1, 1].spines[['top', 'right']].set_visible(False)
axs[1, 1].set_title('Migratory Pain')

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



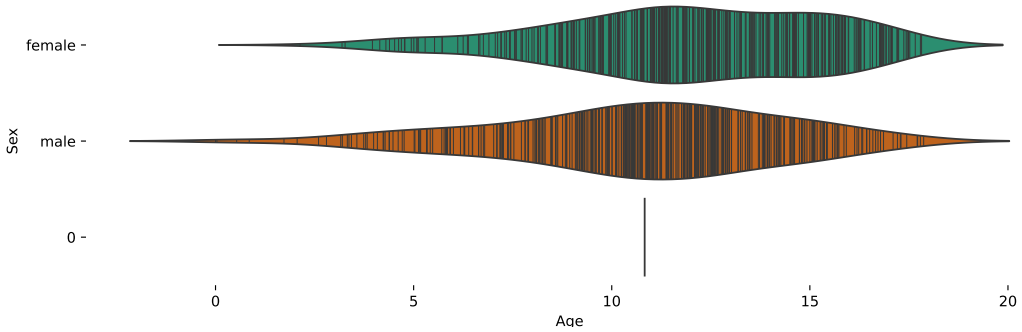
Distribuciones en 2D

```
# Distribuciones en 2D
# número de subgráficos a 1x4
fig, axs = plt.subplots(1, 4, figsize=(20, 5))

# axs[0] para especificar la primera subgráfica
df2.plot(kind='scatter', x='Age', y='BMI', s=32, alpha=.8, ax=axs[0])
axs[0].spines[['top', 'right']].set_visible(False)

 # axs[1] para la segunda subgráfica
df2.plot(kind='scatter', x='BMI', y='Height', s=32, alpha=.8, ax=axs[1])
axs[1].spines[['top', 'right']].set_visible(False)

 # axs[2] para la tercera subgráfica
df2.plot(kind='scatter', x='Height', y='Weight', s=32, alpha=.8, ax=axs[2])
axs[2].spines[['top', 'right']].set_visible(False)

# axs[3] para la cuarta subgráfica
df2.plot(kind='scatter', x='Weight', y='Length_of_Stay', s=32, alpha=.8, ax=axs[3])
axs[3].spines[['top', 'right']].set_visible(False)

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

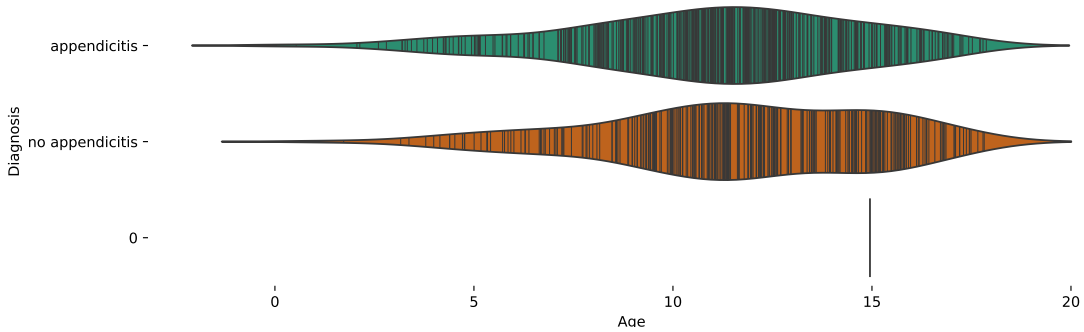


Distribuciones Facetadas

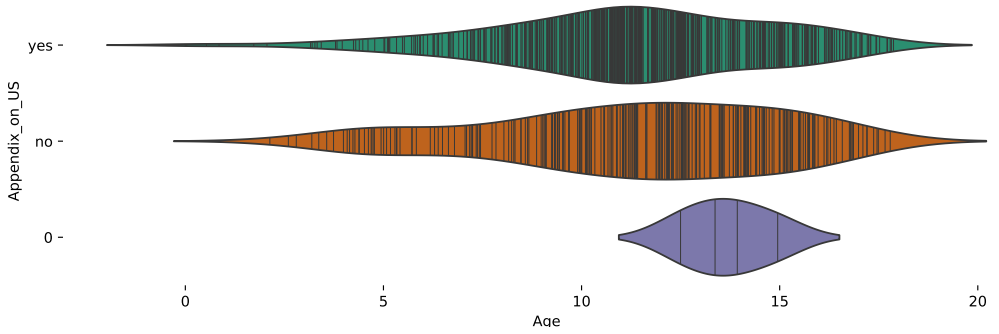
```
figsize = (12, 1.2 * len(df['Sex'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(df2, x='Age', y='Sex', inner='stick', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```



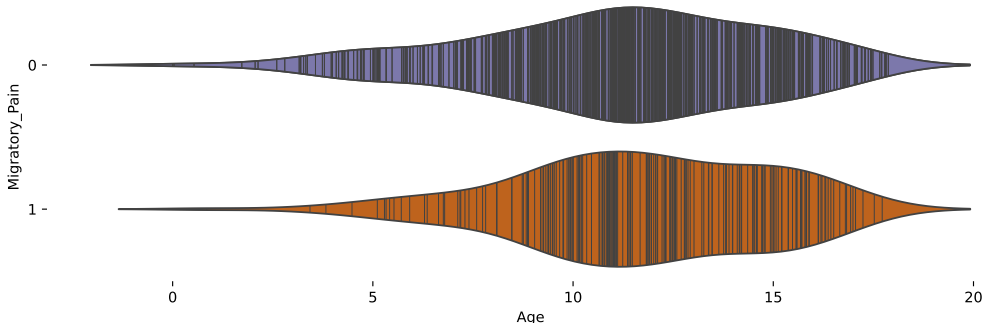

```
figsize = (12, 1.2 * len(df['Diagnosis'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(df2, x='Age', y='Diagnosis', inner='stick', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```




```
figsize = (12, 1.2 * len(df['Appendix_on_US'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(df2, x='Age', y='Appendix_on_US', inner='stick', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```




```
figsize = (12, 1.2 * len(df['Migratory_Pain'].unique()))
plt.figure(figsize=figsize)
sns.violinplot(df2, x='Age', y='Migratory_Pain', inner='stick', palette='Dark2')
sns.despine(top=True, right=True, bottom=True, left=True)
```


Migratory_Pain
0
1
0
5
10
15
20
Age