Escuela Superior
de Ingeniería y Tecnología
Universidad de La Laguna

# Trabajo de Fin de Grado

Grado en Ingeniería Informática

---

# Una herramienta para el análisis preliminar de datos y el aprendizaje automático

*A tool for the preliminary data analysis and machine learning*

José Del Castillo González

---

La Laguna, *8* de *septiembre* de 2021

D. **Francisco Almeida Rodríguez**, con N.I.F. 42831571-M profesor Catedrático de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas Departamento de la Universidad de La Laguna, como tutor

D. **Alberto Cabrera Pérez** con N.I.F. 54061124-T contratado predoctoral adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutor

**C E R T I F I C A (N)**

Que la presente memoria titulada:

*"Una herramienta para el análisis preliminar de datos y el aprendizaje automático"*

y ha sido realizada bajo su dirección por:

D. **José Del Castillo González**, con N.I.F. 51168707-D

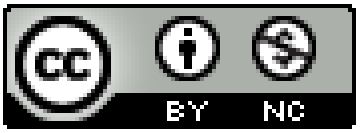
Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 8 de Septiembre de 2021

# Agradecimientos

Me gustaría empezar agradeciendo a D. **Francisco Almeida Rodríguez**, por todo su labor y empeño a lo largo de todas las fases del proyecto, aconsejándome, corrigiendo todos los innumerables e incontables fallos que he cometido y motivándome semana a semana.

También agradecer a mi familia su comprensión y apoyo incondicional

# Licencia



## Resumen

Dentro del análisis y representación de datos hay una amplia gama de herramientas disponibles, esto se debe a la importancia que ha cobrado en los últimos años la posibilidad de extraer información de los datos. Los procesos de digitalización en empresas y sector público así como la generación de datos desde múltiples fuentes, ha permitido producir ingentes cantidades de datos que requieren de las herramientas adecuadas si se desea extraer la información asociada.

Este proyecto plantea la aproximación al tratamiento de los datos mediante el desarrollo de un `framework` que permite una iniciación sencilla y automática al análisis de datos y a la generación de modelos de aprendizaje automático a usuarios no necesariamente especializados. El `framework` proporciona facilidades para el tratamiento preliminar de los datos y para la generación de modelos con los que poder realizar estimaciones .

El proyecto ha sido desarrollado empleando el lenguaje de programación Python, debido a la amplitud de librerías con las que cuenta este lenguaje. Para mejorar la calidad del código y agilizar el diseño se han empleado los patrones de diseño, en concreto el patrón de comportamiento estrategia. Esto facilita la implementación de nuevas representaciones o métodos para el aprendizaje automático, creando una herramienta flexible y fácilmente extensible.



**Palabras clave:** Tratamiento de datos, Visualización de Datos, Aprendizaje Automático, Python

**Abstract**

*Within the analysis and representation of data there is a wide range of tools available, this is due to the importance that the possibility of extracting information from data has gained in recent years. The digitization processes in companies and the public sector, as well as the generation of data from multiple sources, have made it possible to produce huge amounts of data that require the appropriate tools if the associated information is to be extracted.*

*This project proposes the approach to data processing through the development of a framework that allows a simple and automatic initiation to data analysis and the generation of machine learning models to users who are not necessarily specialized. The framework provides facilities for the preliminary treatment of data and for the generation of models with which to make estimates.*

*The project has been developed using the Python programming language, due to the wide range of libraries available in this language. To improve the quality of the code and speed up the design, design patterns have been used, specifically the strategy behavior pattern. This facilitates the implementation of new representations or methods for machine learning, creating a flexible and easily extensible tool.*

**Keywords:** Data processing, Data visualization, Machine learning, Python

# Índice general

# Índice de figuras

# Índice de tablas

# Capítulo 1 Introducción

## 1.1 Objetivos

### 1.1.1 Objetivos Generales

El fin de este proyecto es construir una herramienta genérica con la que el usuario pueda introducirse de manera sencilla y amigable en el proceso de generación de modelos para el Aprendizaje Automático. El paradigma de trabajo en esta herramienta sería que el usuario pudiera incorporar un conjunto de datos sobre el que quiere obtener un modelo de Aprendizaje Automático, se le permitiera hacer un análisis exploratorio de la información contenida en el conjunto de datos y, por último, que el usuario pueda generar modelos basados en el Aprendizaje Automático de manera automática y con escasos conocimientos de esta disciplina.

### 1.1.2 Objetivos específicos

- Estructuración de los datos:

Los datos son un eje central en este proyecto por tanto la herramienta debería permitir incorporar datos en diferentes formatos que se convertirían a una estructura de datos básica. La base de dicha estructura es el `dataframe` de la librería Pandas de Python, que permite almacenar la información en un sistema de dos dimensiones etiquetado. Esta estructura permitirá almacenar datos de diversos tipos sobre los que realizar operaciones de manera masiva.

- Visualización de los datos - Análisis exploratorio de los datos:

La representación y visualización de los datos constituye una parte protagonista de este proyecto, ya que permitirá al usuario realizar un análisis exploratorio gráfico de los datos de una manera sencilla y transparente.

Los datos pueden representar diferentes características, puede tratarse de datos cualitativos o cuantitativos, siendo esto un factor a considerar en el momento de la representación. Por tanto se intentará proporcionar al usuario la mayor cantidad de métodos de representación posibles para sobrepasar este obstáculo.

- Aprendizaje automático

Se trata en este caso de ofertar un conjunto amplio de métodos de aprendizaje automático que puedan aplicarse de manera “genérica” a los conjuntos de datos. El usuario selecciona la información sobre la que desea extraer un modelo y los diferentes métodos a aplicar. Los resultados se podrán mostrar gráfica y analíticamente lo que

permitirá seleccionar los métodos adecuados sobre los que se desea profundizar.

## 1.2 Antecedentes

Con el aumento de la generación de datos, a lo largo del tiempo ha crecido el mercado de las herramientas que permiten extraer la información de ellos. Cada herramienta del mercado sigue un punto de vista desde el cual orienta el trabajo con los datos, siendo dos de estos puntos de vista el de los diseñadores de elementos gráficos y el de desarrolladores de aplicaciones. Este proyecto sigue la línea de la representación de los datos para desarrolladores.

Describimos a continuación algunas de estas herramientas:

- Infogram

Si bien Infogram [1] es una herramienta para la visualización de los datos está centrada, como su nombre indica, en la creación de infografías y no tanto en la explotación de los datos. Permite mostrar de manera muy elegante gráficas y texto conjuntamente pero no está pensada para profundizar en el análisis de los datos, más bien está diseñada para permitir al usuario las diferentes opciones sobre cómo desea exponer los datos, no solo las propias gráficas sino la estructura y texto que acompañarán a las gráficas formando la infografía.

- Tableau

Tableau [2] es una herramienta que se puede utilizar on-line y como aplicación de escritorio. Permite analizar multitud de fuentes de datos aplicando múltiples tipos de gráficas con gran personalización por parte del usuario. Esto deja a los diseñadores un sin fin de formas de representación de los datos, desde las más simples hasta la aplicación de técnicas de machine learning. Esta herramienta cuenta con versiones gratuita y de pago, siendo la principal diferencia que en la gratuita no se permiten proyectos privados.

- D3.js:

D3.js [3] es una librería de Javascript para la manipulación de conjuntos de datos que permite programar la generación de distintas gráficas a partir de archivos de datos en diferentes formatos. Su punto fuerte reside en la creación de imágenes SVG personalizables. Las imágenes SVG son gráficos vectoriales escalables descritos en el lenguaje de programación XML. Es necesario para cada visualización crear el código de dicha visualización.

Hemos descrito tres herramientas, de muchas que pueden encontrarse en el mercado, una que permite hacer representaciones de los datos, otra que permite representar y realizar análisis de los datos y la última orientada a diseñadores dispuestos a crear gráficas y realizar sus propios análisis a partir de dicha herramienta.

La herramienta desarrollada en este proyecto se queda en un punto medio entre representar/analizar y desarrollar, el usuario podrá analizar los datos desde múltiples perspectivas pero si desea estudiar nuevos conjuntos de datos podría ser necesario incorporar nuevos analizadores o estrategias de visualización. En conclusión esta herramienta no está pensada para los que solo desean analizar datos, es más bien una herramienta de iniciación al mundo de la representación y análisis de los datos para nuevos desarrolladores, proporcionando una base sobre la pueden experimentar y crear distintos métodos de representación y análisis que ellos mismos pueden incorporar.

# Capítulo 2 Metodología

Los patrones de diseño constituyen una pieza muy importante en el diseño de software que permiten agilizar el diseño y mejoran la calidad del software desarrollado [4]. En este trabajo se ha hecho uso de un patrón de diseño del "tipo comportamiento" para facilitar el uso y creación de las distintas opciones de representación y análisis. El patrón de comportamiento implementado es el patrón estrategia que permite un diseño más limpio de las distintas opciones, pudiendo codificar cada una de estas de manera independiente sin afectar a la estructura cuando se incorporan nuevos métodos de representación o análisis.

Este proyecto se ha creado haciendo uso del lenguaje de programación Python, un lenguaje de programación interpretado con un amplio espectro de librerías en las que se apoya el proyecto para poder ejecutar todas los módulos que lo conforman. Dichos módulos hacen referencia, en primer lugar, a la lectura y limpieza de archivos para la creación de una estructura de datos estandarizada sobre la que los siguientes módulos puedan trabajar de manera eficiente. El segundo módulo se ocupa de las representaciones gráficas de los datos y un último módulo que aborda los métodos de aprendizaje automático con sus debidas representaciones gráficas para mostrar los resultados.

## 2.1 El patrón Estrategia

Un patrón de diseño es una solución general repetible a un problema común en el diseño de software. No se trata de un diseño terminado que se pueda transformar directamente en código. Un patrón de diseño es una descripción o plantilla, que indica cómo resolver un problema, y puede ser utilizada en muchas situaciones diferentes [4]. La ventaja de usar los patrones de diseño es que constituyen un juego de herramientas con soluciones comprobadas a problemas habituales en el diseño de software. Permiten resolver muchos problemas utilizando los principios básicos del diseño orientado a objetos [5].

Suelen venir agrupados en tres clases: los patrones de creación, los patrones de estructura y los patrones de comportamiento.

Los patrones de diseño creacional son patrones que se ocupan de los mecanismos de creación de objetos, tratando de crear objetos de una manera adecuada a la situación. La forma básica de creación de objetos podría dar lugar a problemas de diseño o una mayor complejidad al diseño cuando el software evoluciona. Los patrones de diseño de creación resuelven este problema controlando la creación de los objetos bajo unos principios básicos [6]. Los patrones de diseño estructural son patrones de diseño que facilitan el diseño al identificar una forma sencilla de crear relaciones entre entidades [7]. Por último los patrones de diseño de comportamiento son patrones de diseño orientados a mejorar la interacción y responsabilidades entre clases y objetos, así como los algoritmos que encapsulan [8].

En este proyecto se hace un uso intensivo del patrón estrategia. Se trata de un patrón de diseño de comportamiento que está pensado para situaciones en las que una acción que puede realizarse de diferente forma según el contexto en el que se encuentre.

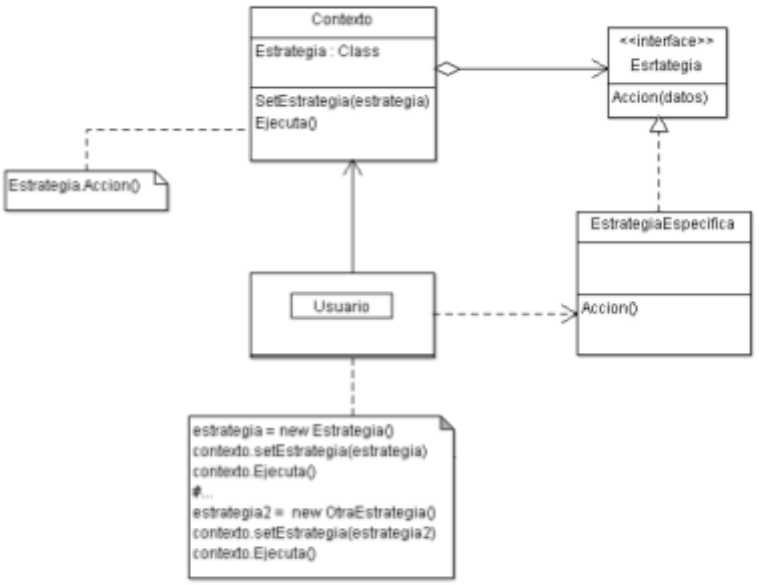



Figura 2.1. Diagrama uml para el Patrón Estrategia

Los elementos principales que intervienen en el diseño de este patrón son la acción que realizará la estrategía y el contexto. En la figura 2.1 se encuentra un diagrama del patrón estrategia que contiene los siguientes elementos; la clase `Estrategia` contiene el método llamado `Accion()` quel representa la actividad a realizar, la clase `EstrategiaEspecifica` hereda de la Clase Estrategia el método `Accion()` modificándolo de manera que realice una tarea distintiva pero manteniendo un mismo objetivo. El `Contexto` es la clase que contendrá la instancia de la `Estrategia` y desde la cual se ejecutará la estrategia. Para que la clase `Contexto` ejecute la acción usará el método `Ejecuta()` para efectuar la acción de la `EstrategiaEspecifica` que esté inicializada en el contexto. La clave del patrón estrategia es que dependiendo de los `inputs` o elecciones por parte del usuario en el contexto se establecerá una

estrategia u otra que permita obtener su `output` objetivo.

Una vez presentado el patrón estrategia sobre el que se sustenta el proyecto es necesario mencionar que se hizo uso de este patrón para la implementación de las distintas partes que conforman el proyecto: el análisis de la estructura del archivo de datos, visualizaciones básicas y modelos que aplican tecnologías de machine learning.

En lo referente al análisis de la estructura del archivo de datos, el patrón nos es de utilidad para poder cambiar de estrategia de análisis sin necesidad de modificar la estructura. En el módulo de visualización de los datos el patrón nos servirá para el cambio entre los diferentes métodos de visualización. La figura 2.2 muestra un diagrama de clases que representa el diseño de la aplicación, el usuario interactuará con la clase `interfazUsuario`, cuando seleccione un método para la limpieza y preparación de los datos la clase `intermedio` le proporciona el método seleccionado para poder crear el `dataset`. Una vez el usuario cuente con el `dataset` podrá realizar las representaciones llamando al método `show()` de la clase `intermedio`, que hará uso de la clase `Selector` para que establezca la estrategia seleccionada por el usuario en el `Contexto`, una vez tengamos el `Contexto` se invocará al método `show()` representando así la gráfica final.

En lo que se refiere al aprendizaje automático su principal objetivo es la construcción de un modelo con el que poder trabajar posteriormente. Para obtener un modelo de aprendizaje automático es necesario realizar una serie de pasos, primero es necesario una preparación, para cada modelo de aprendizaje automático declararemos el modelo, con los parámetros de configuración que se desee trabajar y prepararemos los datos de manera que obtengamos unos datos con los que poder entrenar el modelo y otros con los que poder validar el rendimiento del modelo. Una vez esté el modelo declarado y los datos preparados será el momento de llevar a cabo su entrenamiento, en el caso de que así lo requiera el modelo. Una vez tengamos el modelo correctamente preparado se realizará la validación del modelo. La validación proporciona información sobre cómo de bueno ha sido el entrenamiento y el rendimiento con los distintos datos de validación, hasta datos únicos dependiendo del área al que esté asociado el modelo. En el proyecto se aprovecharán los datos extraíbles a los modelos combinados con múltiples representaciones gráficas, para poder mostrar y analizar los diferentes modelos implementados.

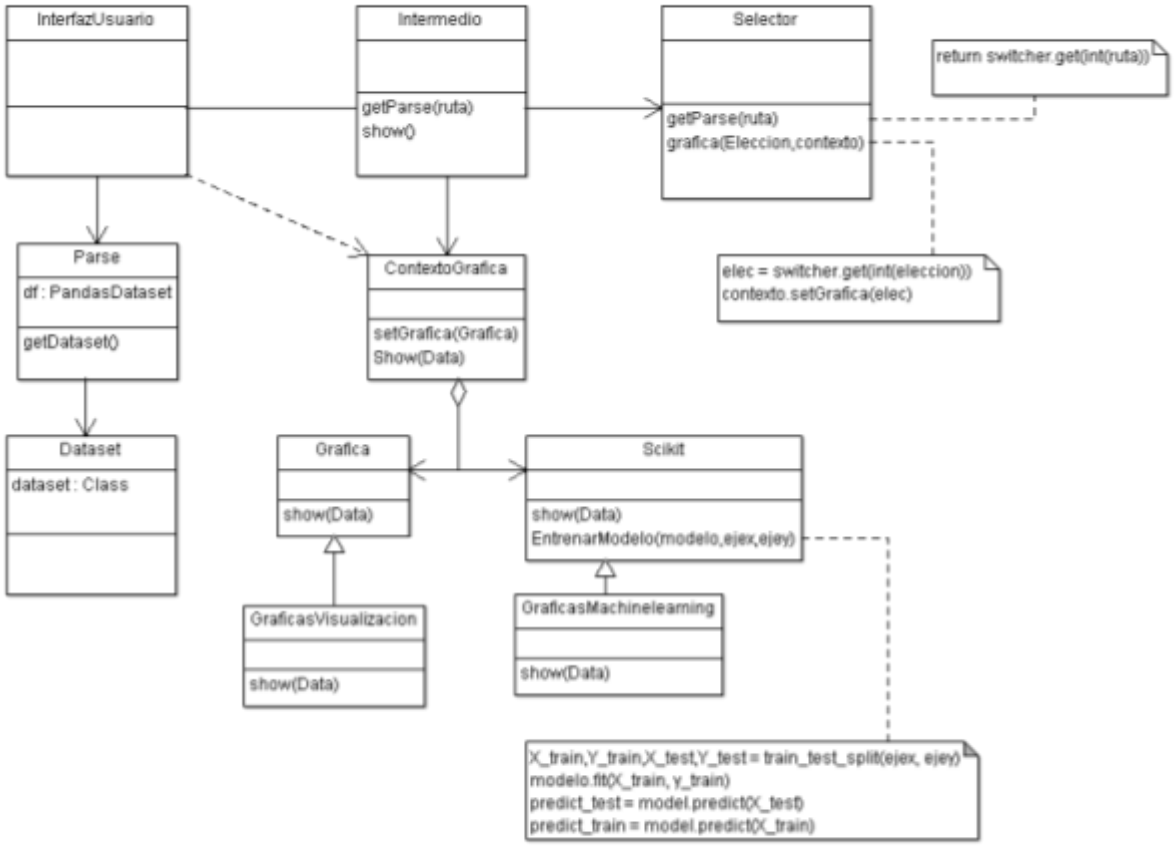



Figura 2.2. Diagrama clases del proyecto

## 2.2 El patrón estrategia en el Tratamiento de los Datos

Dado que los datos pueden provenir de distintas fuentes y por tanto en distintos formatos, se desarrolla un proceso preliminar en el que los datos se transforman en una estructura genérica a emplear por todos los procesos de la herramienta, en la práctica un `dataset` de Python.

Esto implica un proceso de análisis y limpieza del conjunto original a través de un método de análisis específico para cada conjunto de datos. El trabajo en común de todos los depuradores de archivos, llamados `Parse` en la figura 2.3, es el de crear el conjunto de datos genérico a partir de los datos de origen, por tanto esa es la acción y el contexto que varía según la fuente de datos viene dado por las diferentes clases encargadas de analizar los datos. Por ello el trabajo del patrón estrategia en este módulo consiste en asignar un método de análisis u otro en función del conjunto de datos para poder generar el `dataset`.

En la figura 2.3 se encuentra el diagrama UML referido al tratamiento de los datos. Arranca en la clase de `InterfazUsuario`, cuando el usuario ha seleccionado una fuente de datos se le indica a la clase intermedia que pida al Selector la estrategia `Parse` que se ha de emplear. Cuando ya se tenga definida cuál será el `Parse` a emplear se podrá generar el `dataset`.

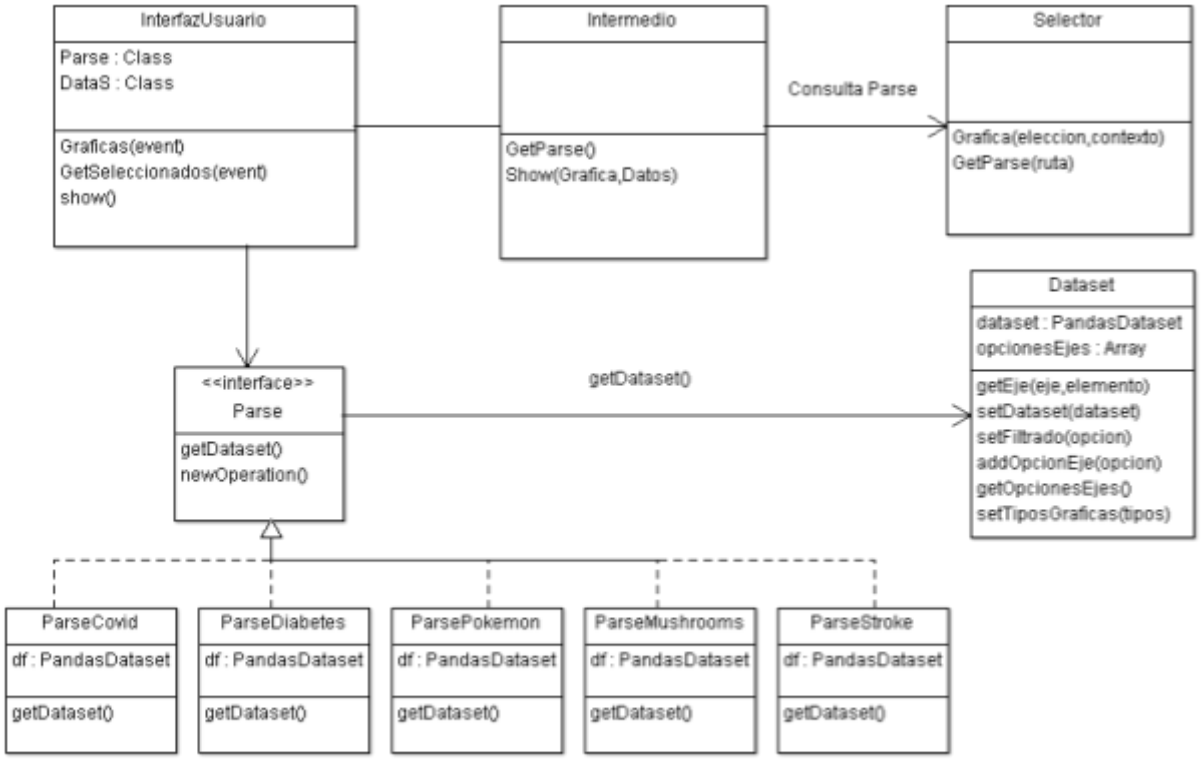



Figura 2.3. Diagrama UML para el análisis y formateo de datos

## 2.3 El patrón estrategia en la Visualización

El proceso de visualización de los datos se asemeja, desde el punto de vista del diseño, al procesamiento de los datos, las diferentes clases de representación cuentan con un método `show()`, que es el encargado de hacer la representación y constituye la acción común, el contexto viene dado por el método de representación que el usuario selecciona.

Una vez el usuario haya escogido el conjunto de datos con el que desea trabajar, elegirá el tipo de representación y las diferentes opciones de representación (ejes, …). Finalizado el proceso de selección se pasa el conjunto de de datos y el tipo de representación al objeto de representación para que lo muestre mediante el correspondiente método `show()`. En la figura 2.4 se puede observar un diagrama con las distintas clases que componen la estructura de trabajo descrito en esta sección.

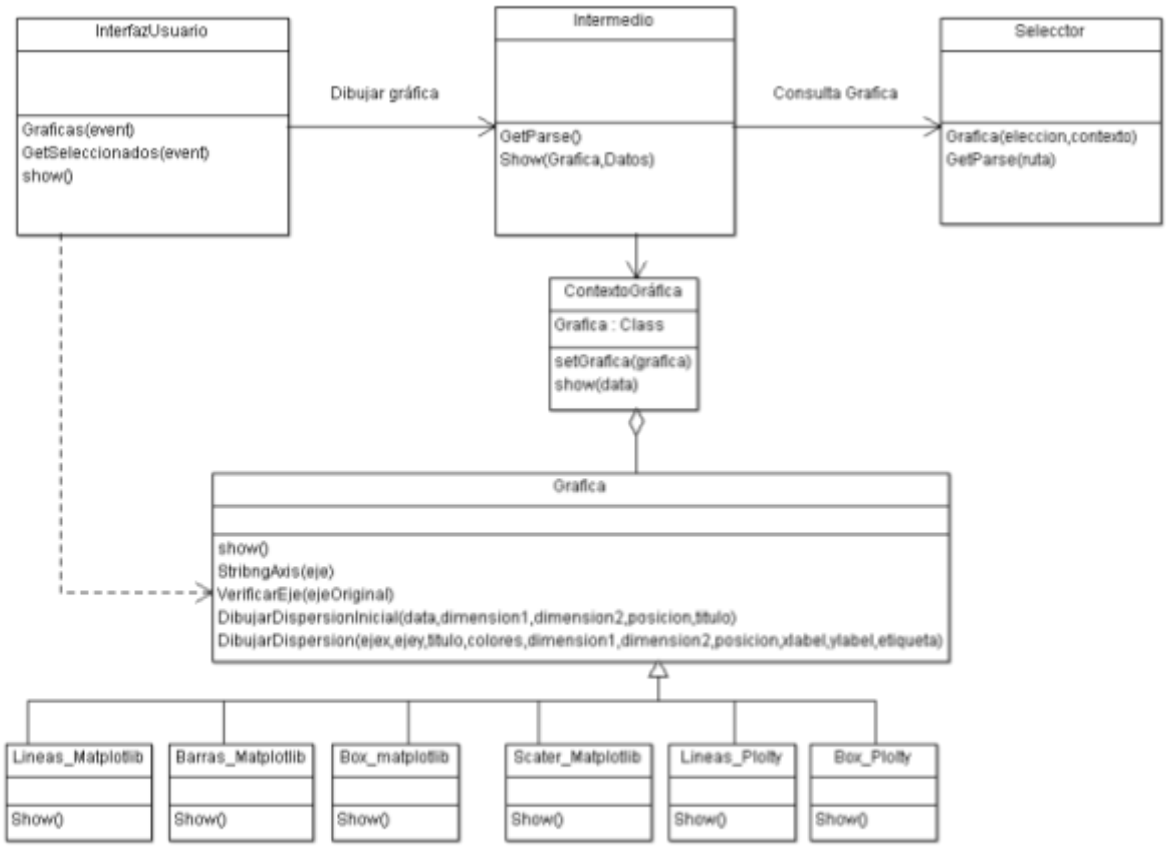



Figura 2.4. Diagrama UML del patrón estrategia en la visualización

## 2.4 El patrón estrategia en el Aprendizaje Automático

En el aprendizaje automático el tipo de “comportamiento” es distinto al presente en el caso de la visualización. En el aprendizaje automático, los métodos de la misma categoría heredarán el método de representación de la categoría dado que la forma de representarlo es la misma, solo varía el modelo entrenado y los datos resultantes proporcionados por el modelo, mientras que el flujo de trabajo se mantiene. Las categorías con las que se trabaja en este proyecto son Regresión, Clasificación y Clustering.

Cuando hablamos de aprendizaje automático es necesario hablar del flujo de trabajo necesario para poder desarrollarlo, dicho flujo es el mismo independientemente del área o técnica que se emplee. El flujo de trabajo consta principalmente de cuatro partes: preparación, entrenamiento, validación y estudio de resultados. Esencialmente el aprendizaje automático persigue entrenar un modelo y su posterior validación, para ello es necesario la preparación de los datos y el uso del modelo una vez entrenado.

En la preparación se encuentran los procesos asociados a limpiar, estructurar y adaptar los datos para que puedan utilizarse en las siguientes fases. Limpiar y estructurar en este proyecto se realiza en la fase de tratamiento de los datos visto en la sección 2.2,

pero la adaptación de los datos para el aprendizaje consiste en crear dos conjuntos de datos, los de entrenamiento para poder entrenar el modelo y los de validación para poner a prueba y validar el modelo. La última parte de la preparación es inicializar el modelo que vamos a emplear con los parámetros con los que se quiera trabajar. En la fase de entrenamiento, se ajustará el modelo con el conjunto de datos de entrenamiento, esto se consigue utilizando el método `fit()` que contienen todos los modelos de aprendizaje automático. Este método permite al modelo recoger los datos y realizar el aprendizaje siguiendo los algoritmos internos que tenga definidos. Para realizar la validación, en este proyecto se hace uso del método `train_test_split()` de la librería Sklearn. Del mismo modo que en el entrenamiento le pasaremos los datos al modelo pero esta vez para que aplique lo aprendido en la fase de entrenamiento y lleve a cabo todos los cálculos para obtener una validación.

Por último se encuentra la fase de estudio de resultados, cuando el modelo ya esté entrenado y se le haya puesto a prueba se puede extraer con los métodos pertinentes, que dependen de cada modelo, todos los datos resultantes y será el trabajo en esta fase de poder extraer la información de dichos resultados. Los resultados de los modelos pueden ser numéricos o analíticos, estos resultados se aprovecharán para llevar a cabo diversas representaciones dependiendo del modelo y gracias a las cuales obtendremos la información de cómo de eficiente ha sido el modelo con los datos de entrenamiento y el rendimiento con los datos de validación.

En la figura 2.5 se encuentra un diagrama que contiene la estructura del patrón estrategia aplicado al aprendizaje automático que se emplea en el proyecto. El usuario tiene la posibilidad de escoger el modelo que desee aplicar sobre los datos seleccionados, una vez seleccionado se asignará al Contexto la estrategia para la construcción del modelo de aprendizaje automático pertinente.

Cada área de aprendizaje automático cuenta con su propia clase para llevar a cabo la representación, pero antes de poder realizar la representación es necesario la pieza más importante dentro del aprendizaje automático, que es el modelo. El modelo será inicializado en su respectiva clase, que tendrá en cuenta los parámetros de configuración pertinentes. Una vez el modelo está declarado, en la clase padre Scikit se encuentra el método que lleva a cabo el entrenamiento y validación de los modelos, por último cada área cuenta con una clase que contiene un método para realizar la representación de los resultados experimentales obtenidos de los modelos.

En la figura 2.5 no está incluida en el módulo del tratamiento de los datos presentado en la sección 2.2.

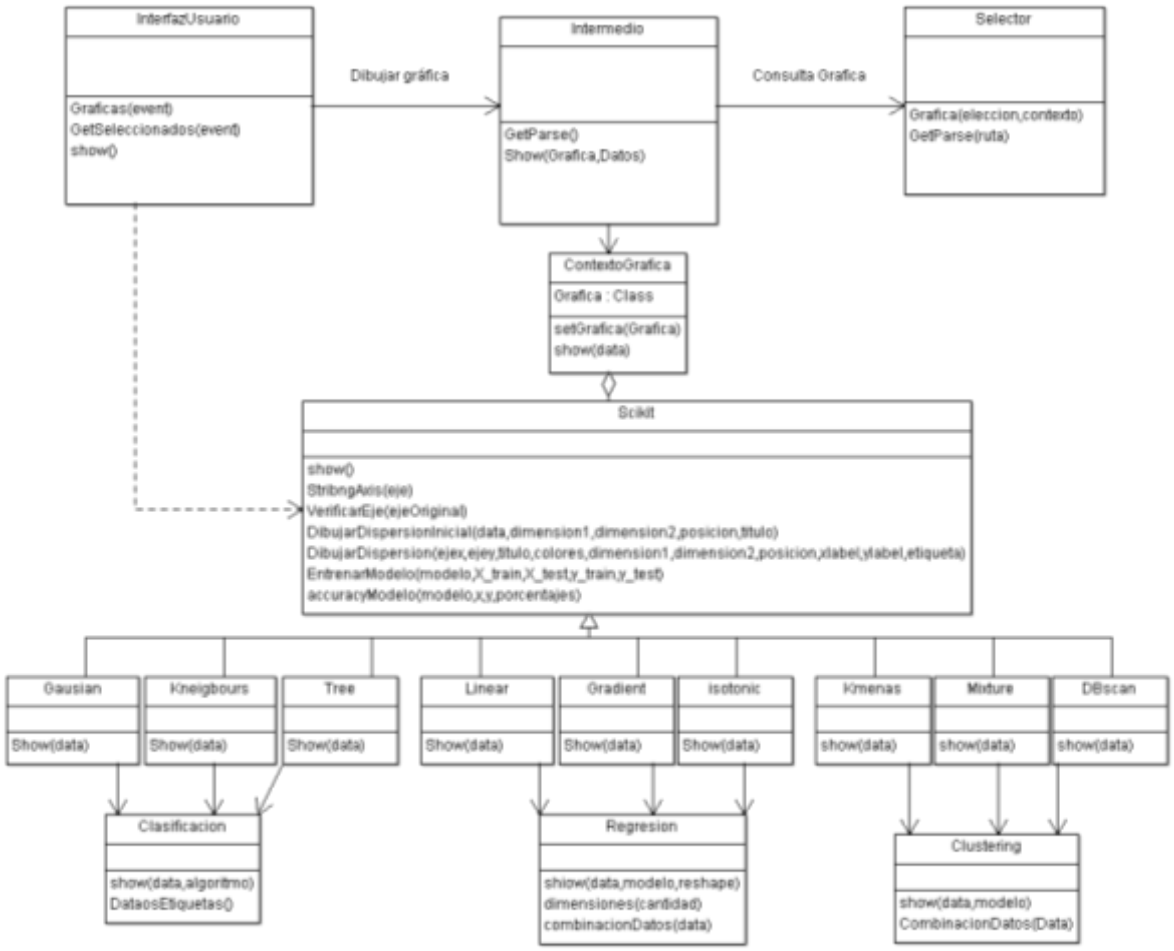



Figura 2.5. Diagrama UML del Patrón estrategia en el aprendizaje automático

# Capítulo 3 Implementación

Este proyecto ha sido desarrollado sobre el sistema operativo windows 10, por ello las herramientas que se van a mencionar a continuación y más específicamente su instalación están orientadas a este sistema operativo. Todo el desarrollo de este proyecto se ha realizado empleando el lenguaje de programación Python, este lenguaje, además de su gran potencia y flexibilidad, cuenta con una amplia colección de librerías para la representación de datos, para el cálculo científico y para el aprendizaje automático. A lo largo de este proyecto se han usado las siguientes librerías de este "ecosistema" asociado a Python: matplotlib, plotly, numpy, tkinter, pandas y scikit-learn.

## 3.1 Python

Python es un lenguaje de programación interpretado de alto nivel que permite realizar un gran número de scripts, funcionalidades y proyectos gracias a su flexibilidad y numeroso conjunto de librerías. La versión utilizada es la 3.7. Cuando se ha terminado la instalación de Python hay que asegurar que se ha creado correctamente la ruta para poder acceder a la consola de Python desde el terminal windows. En la figura 3.1 se puede observar a la izquierda el panel de propiedades del sistema desde el que podemos acceder a las variables de entorno y a la derecha el panel de las variables de entorno.

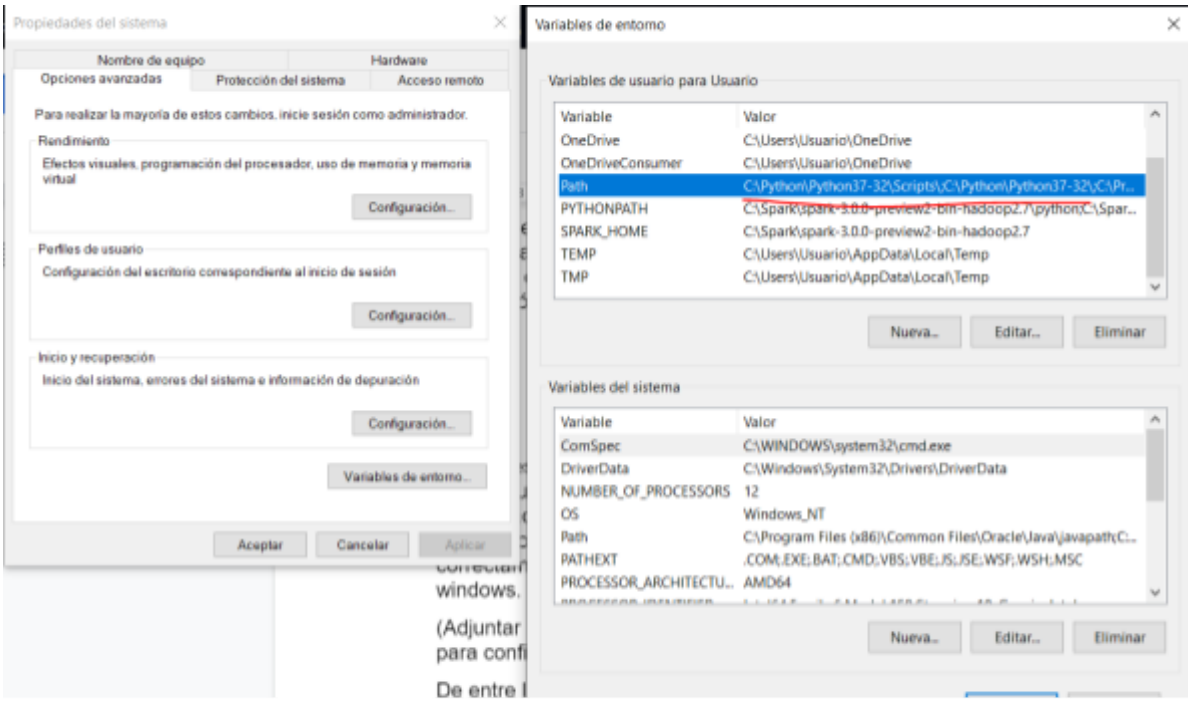



Figura 3.1. Configuración de la ruta de acceso para python en windows

A continuación mencionamos brevemente las principales librerías que han sido usadas para el desarrollo de este proyecto:

- Numpy: Numpy es una librería pensada para la creación de arrays multidimensionales y proporciona muchas facilidades y métodos para su manipulación, desde métodos matemáticos pasando por los algebraicos hasta los estadísticos, aplicables con amplia rapidez a estas estructuras. La versión instalada es la 1.18.4.

  Para este proyecto utilizaremos principalmente la estructura array de Numpy, que permite almacenar más datos y a mayor velocidad que la estructura básica de Numpy, y los métodos sobre dichos arrays para la correcta manipulación de los datos.

  El fragmento de código que puede verse en la figura 3.2 muestra algunos ejemplos de operaciones que pueden realizarse sobre la estructura de datos `numpy.array`

con la librería Numpy.

```
#////
#Redimensionando un array
X_train = np.reshape(X_train, (-1, 1))
#Ordenando los valores de un array usando el mergesort
X_test =  np.sort(X_test, kind = 'mergesort')
#Creando un rango de valores
porcentajes = np.arange(0.3, 0.9, 0.05)
#/////
```

Figura 3.2. Código librería Numpy

- Plotly: Plotly es una librería también open source que permite realizar gráficas interactivas, y dado que está construida sobre Javascript a dichos gráficos se accede principalmente desde navegadores web. Esta librería estaba pensada para una expansión del proyecto al entorno web, pero se ha mantenido como librería para hacer representaciones gráficas en web. La versión instalada es la 4.7.1.

  El fragmento de código de la figura 3.3 muestra una representación con gráfico de líneas usando Plotly. Primero es necesario declarar la figura que contendrá los gráficos, seguidamente se añadirán los datos a representar, posteriormente los datos informativos como los títulos de los ejes y por último representar la figura.

```
#////
import plotly.graph_objects as go
#Declaro la figura
fig = go.Figure()
for selec in seleccionados:
       ejeX = data.getEje(data.getSeleccionEjeX(),selec)
       ejeY = data.getEje(data.getSeleccionEjeY(),selec)
       #Añado la línea a la figura
       fig.add_trace(go.Scatter(x = ejeX, y = ejeY, name =
selec))
#Actualizo los datos del layout
axisx= data.getSeleccionEjeX()
axisy= data.getSeleccionEjeY()
fig.update_layout(title = data.getTitle(), xaxis=
dict(title=axisX), yaxis=dict(title=axisY))
#Muestro la figura
fig.show()
#////
```

Figura 3.3. Código libreria Plotly

- Pandas: Pandas es una extensión de la librería Numpy, pensada para el tratamiento de datos científicos y se apoya en las estructuras de datos que proporciona Numpy.

  Principalmente su uso en este proyecto ha sido la utilización de los `dataset`, estructuras de datos similares a las tablas en SQL. Los `dataset` nos permitirán almacenar todos los datos provenientes de la fuente de datos de forma que los podamos manejar con soltura. La versión instalada es la 1.0.3.

  El fragmento de código que se puede ver en la figura 3.4 contiene un ejemplo de tratamiento de un fichero `csv` con Pandas para poder extraer una columna de un `dataset`. Primero se abre el fichero csv que contiene los datos, las siguientes líneas de código sirven para limpiar los datos, continuamos con la selección de la columna que deseamos y por último con la columna seleccionada la pasamos a una estructura de datos más fácil de manipular como son las listas de Python haciendo uso de la propiedad `value`.

```
#////
# Para generar el dataset desde un csv
self.df = pd.read_csv("data/diabetes.csv")
#Limpiando el dataset de valores incorrectos
self.df["Outcome"].replace({0: "Negativo", 1: "Positivo"},
inplace=True)
self.df[i].replace({None: 0}, inplace=True)
#Escogiendo una columna del dataset
filtrado = self.df[self.df[self.filtrar].isin([elemento])]
#Convirtiendo la columna en un Numpy array para poder manejarlo
mejor
filtrado = filtrado[eje].values
#///
```

Figura 3.4. Código libreria Pandas

- Tkinter: El paquete de Tkinter es la interfaz Python estándar para el conjunto de herramientas `TK GUI`. Usado como herramienta para poder desarrollar la interfaz con la que interactúa el usuario.

  En el fragmento de código que se muestra en la figura 3.5 presenta un ejemplo de la creación de una ventana con Tkinter, la asignación de una `dropdown` con las opciones disponibles al usuario y un método asignado al `dropdown` que permita ejecutarlo una vez se seleccionada la opción correspondiente.

```
#////
ventana = tk.Tk()
ventana.geometry('430x600') # Asignamos unas dimensiones a la
ventana
#Creación del desplegable que contendrá opciones a seleccionar
dropDownSeleccion = ttk.Combobox(state = "readonly")
dropDownSeleccion.bind("<<ComboboxSelected>>", addSeleccion)
#Asignamos valores al desplegrable
dropDownSeleccion["values"] = [*opciones]
#La asignamos a la ventana
dropDownSeleccion.grid(column = 2, row = 1)
#///
```

Figura 3.5. Código libreria Tkinter

- Matplotlib: Librería cuya finalidad es la creación de gráficas en Python. Será el principal pilar para la representación de todos los datos. La versión instalada es la 3.2.1.

  En el fragmento de código de la figura 3.6 se encuentra un ejemplo de representación gráfica haciendo uso de la librería matplotlib. Se realizan dos representaciones gráficas sobre la misma figura, la primera es una gráfica de tipo línea y la segunda una gráfica de dispersión.

```
#/////
#Dibujando una linea de regresión con matplotlib
plt.plot(X_test, regresion_y,c = 'red')
#Declarando una figura de matplotlib
fig, ax = plt.subplots()
#Representando una gráfica de dispersión usando los axis
ax.scatter(data.getEje(data.getSeleccionEjeX(),selec),data.getEje(
data.getSeleccionEjeY(),selec),alpha=0.3,label=selec)
#Asignado datos a la gráfica
plt.xlabel(data.getSeleccionEjeX())
plt.ylabel(data.getSeleccionEjeY())
plt.title("Dispersión con regresión")
#Dibujando la gráfica, cuidao que funciona como un return asi que
solo usar 1 por método de representación
plt.show()
          #////
```

Figura 3.6. Código librería Matplotlib

- Scikit learning: Liberia diseñada para la aplicación de técnicas de aprendizaje automático y sobre la que estarán sustentados todos los métodos de aprendizaje que se aplicarán a lo largo del proyecto. Esta librería es compatible con librerías

matemáticas como son Numpy o scipy. La versión instalada es la 0.23.1.

En el fragmento de código de la figura 3.7 se muestra la generación de un modelo de regresión lineal (con su posterior representación) empleando scikit-learn, y haciendo uso del método `train_test_split` de la propia librería para la generación de los conjuntos de pruebas y testeo.

```
#////
from sklearn.linear_model import LinearRegression
#Declaración de un modelo
model = LinearRegression()
# Separación de los datos
X_train, X_test, y_train, y_test = train_test_split( EjeX, EjeY,
test_size=0.4, random_state=0)
X_train,ejex = Scikit.VerificarEje(X_train)
y_train,ejey = Scikit.VerificarEje(y_train)
#Entreno del modelo
model.fit(X_train,y_train)
#Realizando un predict
regresion_y = model.predict(X_test)
plt.scatter(ejex,ejey)
plt.plot(X_test, regresion_y,c = 'red')
#////
```

Figura 3.7. Código libreria Scikit

Para la utilización de una nueva librería es siempre necesario instalarla, además de de importarla desde el código. Un aspecto a tener en cuenta cuando se instalan nuevas librerías mientras se trabaja con visual studio es que se han de añadir también al entorno de desarrollo, para ello hay que seguir los siguientes pasos:

El primer paso es hacer uso del comando `pip` (>pip install “libreria”), comando que proporciona la propia instalación de Python. El siguiente paso es abrir la sección de paquetes del entorno de Python, para ello primero hay que seleccionar el entorno en el que estamos trabajando. En la figura 3.8 se encuentra la sección desde la que podemos observar los entornos de Python que están habilitados actualmente en el entorno de desarrollo y seleccionar con el que queremos trabajar.

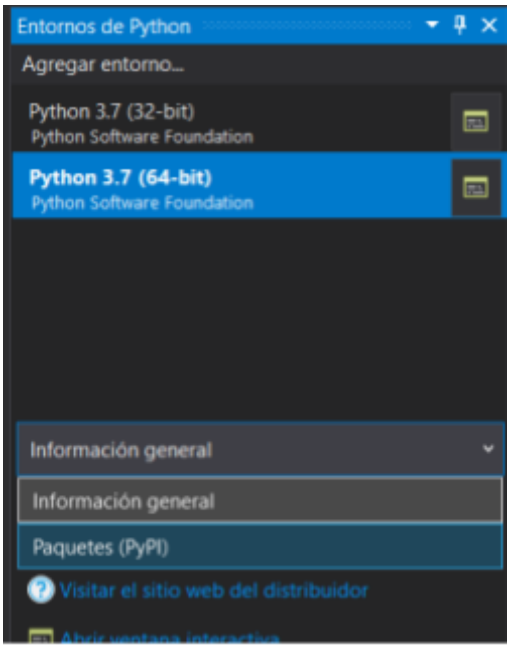



Figura 3.8. Entorno python

Posteriormente seleccionamos la opción de paquetes Python para obtener el listado de paquetes, como el que se encuentra en la figura 3.9, desde donde podremos añadir nuevos, observar los que ya están instalados y actualizar los que ya están instalados si así lo quisiéramos. Si no se realiza este paso las librerías pese que pueden estar instaladas en el ordenador desde visual no se podrán emplear ya que no las encontrará en el entorno que está trabajando.

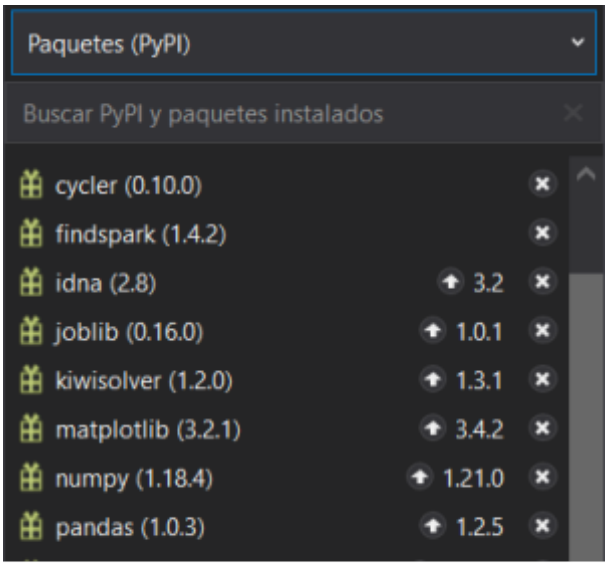



Figura 3.9. Paquetes Python

## 3.2 Otras herramientas

### Visual

Visual studio es un entorno de desarrollo integrado, que me permite desarrollar todo el código del proyecto, depurarlo y ejecutarlo.

La versión con la que se ha trabajado es la 16.4.5, como el trabajo está desarrollado íntegramente en Python también es necesario tener instalado el módulo de Python. La instalación del módulo de Python se realiza desde el panel de herramientas, dicho panel se abre desde la pestaña de herramientas ubicado en la parte superior, una vez abierto el desplegable hay que seleccionar la opción “Obtener herramientas y características” esta ruta está presente en la imagen 3.10. Una vez abierto el panel de herramientas podemos seleccionar el paquete que necesitemos para el proyecto, como se puede visualizar en la figura 3.11.

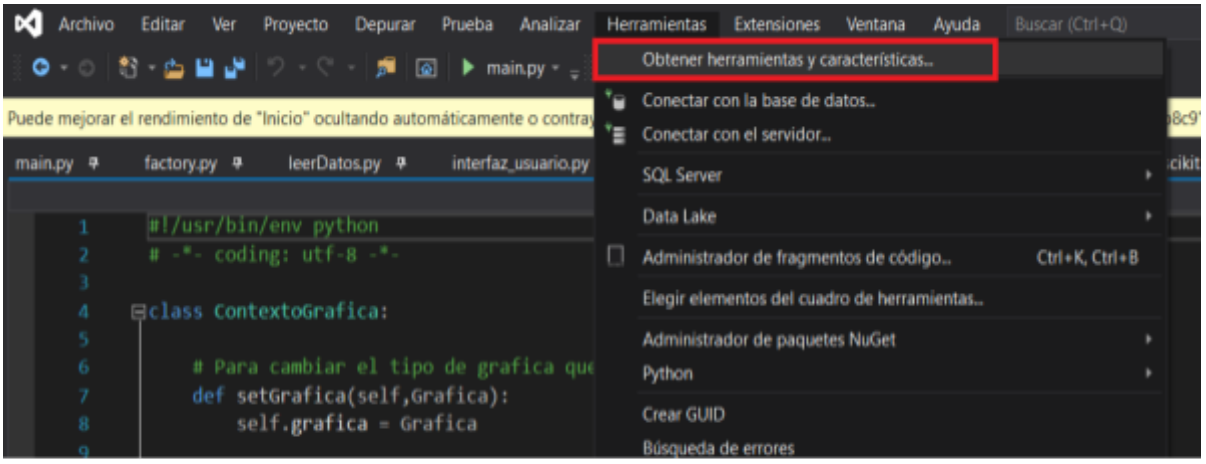



Figura 3.10. Ubicación características visual studio

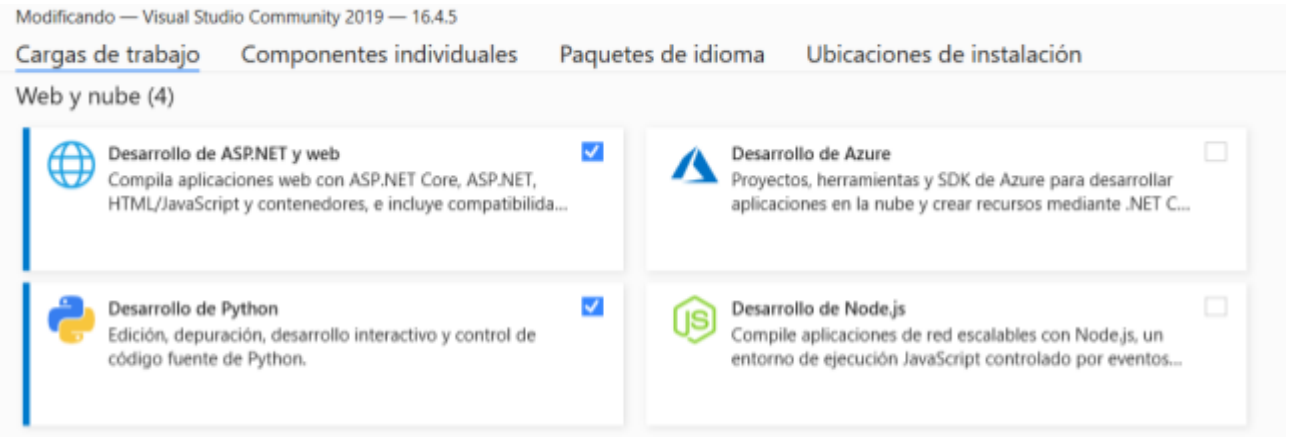



Figura 3.11. Paquetes python

## Github

Github es un sistema de control de versiones accesible desde internet que emplea el sistema git para controlar todos los cambios que se produzcan en el código. Al estar trabajando en windows hay dos opciones para llevar a cabo las nuevas actualizaciones al proyecto, usando el método tradicional con comandos desde el terminal o directamente usando la herramienta específica para windows. En este proyecto se ha optado por la versión de la herramienta desarrollada para windows. En dicha herramienta una vez que el repositorio esté vinculado y haya cambios en el código podremos realizar un `commit` de estas modificaciones, cuando tengamos los `commits` preparados realizaremos la acción `push` para que se actualice la rama indicada con los nuevos cambios. En la figura 3.12 se encuentra la interfaz de esta aplicación.

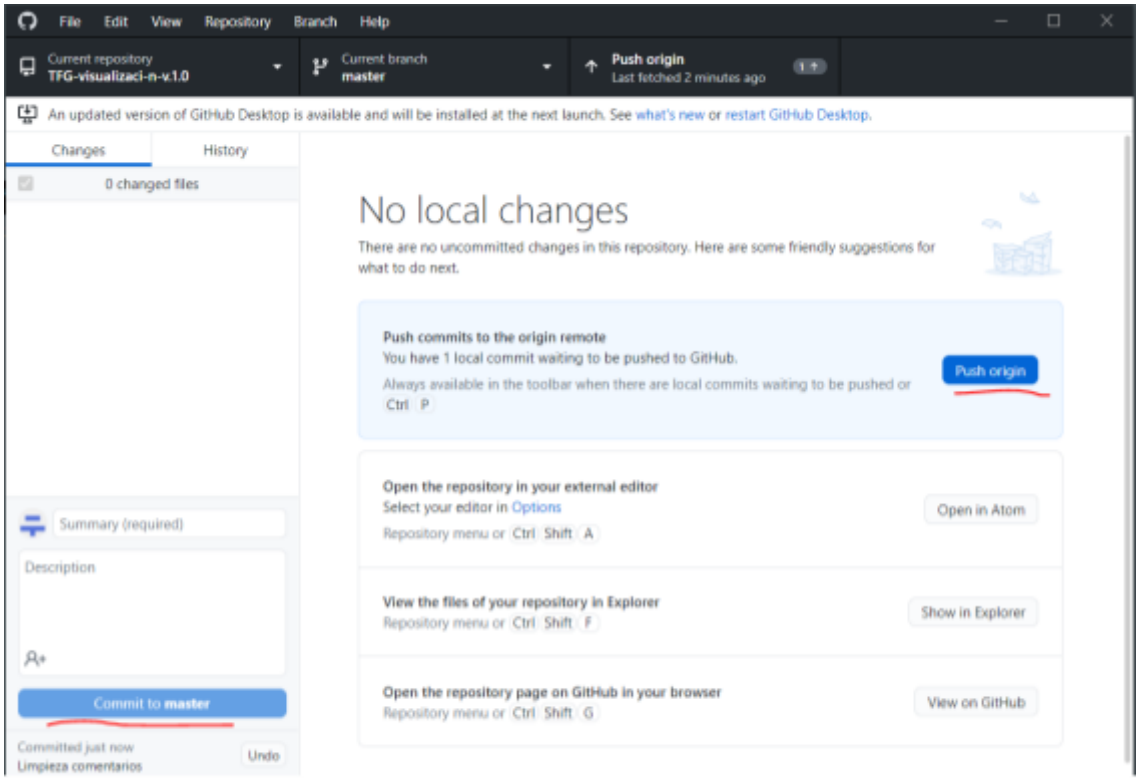



Figura 3.12. Herramienta github windows

Otro punto a tener en cuenta sobre github es que podremos enlazarlo con visual studio para poder ver el registro de los `commits` que se han realizado desde el propio IDE. En la figura 3.13 se encuentra el historial de `commits` de este proyecto.

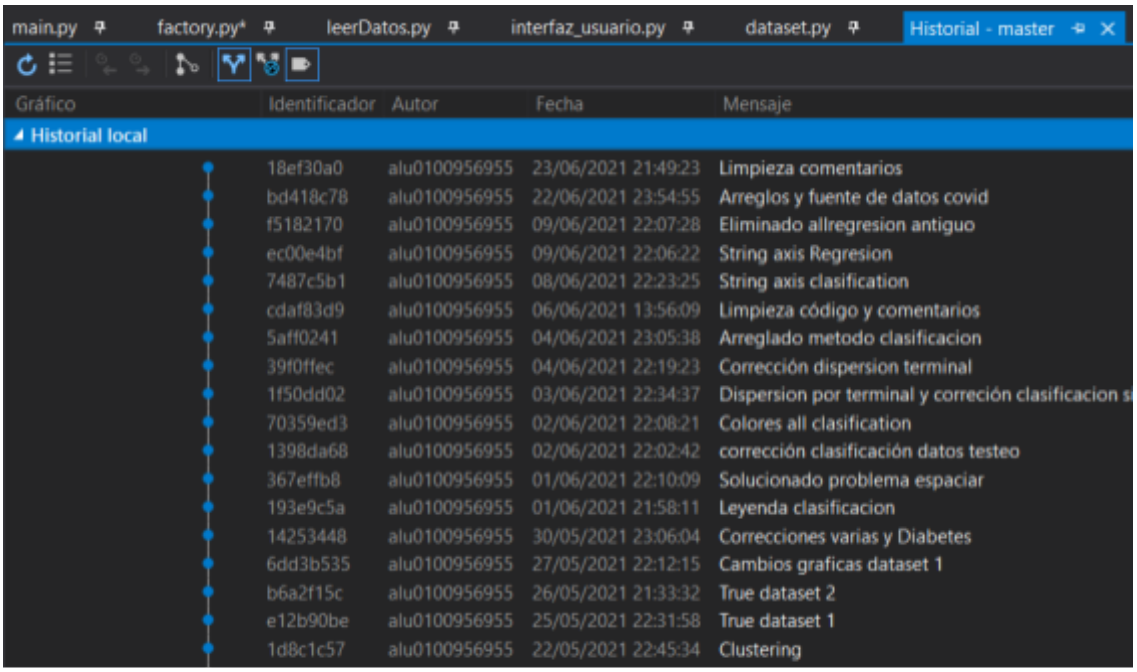



Figura 3.13. Commits github en visual studio

Todo el proyecto se encuentra ubicado en el siguiente repositorio de github:
https://github.com/alu0100956955/TFG-visualizaci-n-v.1.0

# Capítulo 4 La aplicación

## 4.1 La Aplicación

La aplicación final resultante del desarrollo del proyecto sigue el diagrama de actividades que se encuentra en la figura 4.1. La interacción del usuario con la aplicación se orienta a la elección de los valores y parámetros a mostrar en los gráficos. Una vez que el usuario elija la fuente de datos se generará el `dataset` con el que se trabajará en los procedimientos posteriores, la elección de los siguientes parámetros no se terminarán de aplicar sobre la gráfica resultante hasta que el usuario marque la opción de representar.

Dentro de cada opción de representación se llevan a cabo las operaciones pertinentes, siendo mayor la cantidad de operaciones en las estrategias de aprendizaje automático debido a que es necesario el proceso de construcción de los modelos con su posterior validación sumado a que su representación hace uso de una combinación de múltiples gráficas.

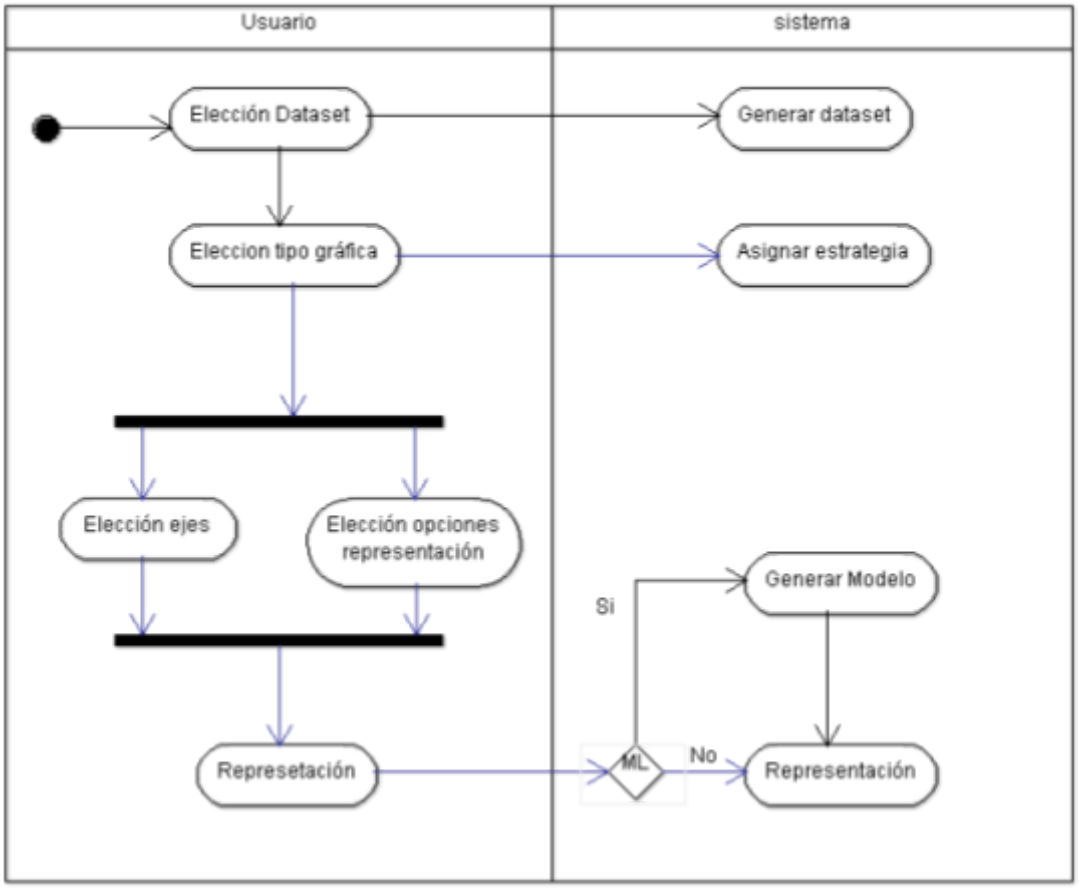



Figura 4.1. Diagrama actividades de la aplicación

El funcionamiento de la aplicación es la siguiente: Una vez la aplicación esté iniciada se abrirá la ventana principal, como se puede observar en la figura 4.2, con un desplegable pidiéndonos seleccionar una fuente de datos, con la que se creará el `dataset`.

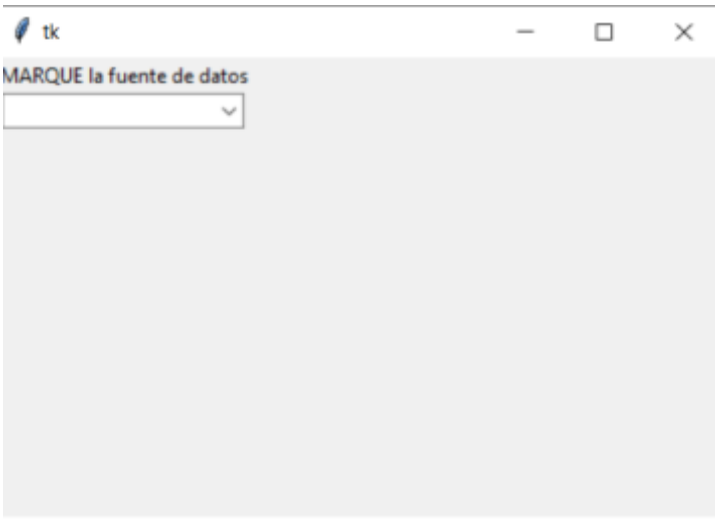



Figura 4.2. Inicio de la aplicación

Una vez que se haya seleccionado la fuente de datos se dispondrá un nuevo

desplegable con el que se debe elegir el tipo de gráfica que se desea representar sobre la fuente de datos que se ha escogido. En la figura 4.3 se muestra la ventana con el desplegable para selección del tipo de gráfica, siendo las opciones disponibles las recogidas en la figura 4.6.

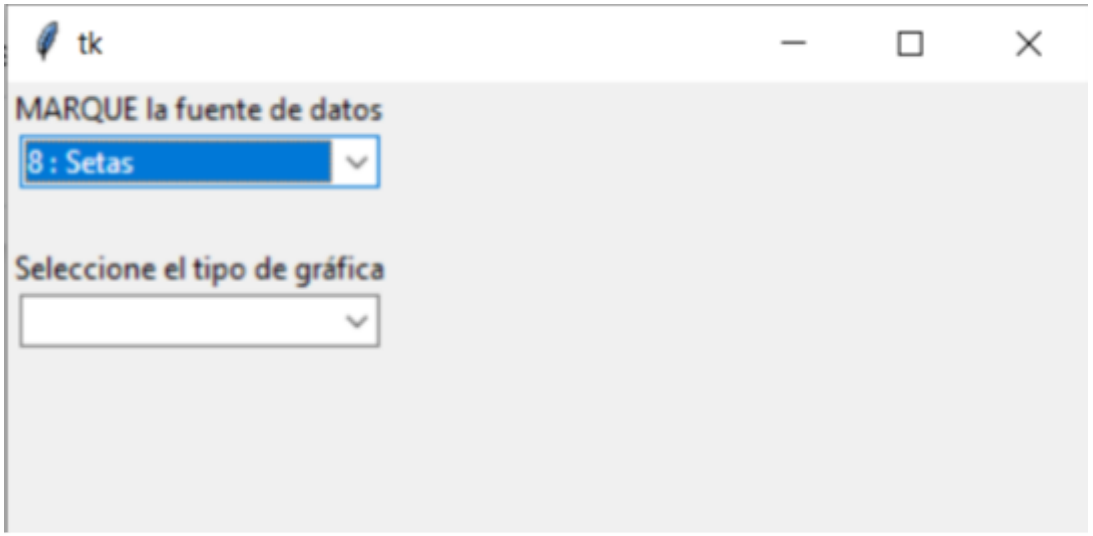



Figura 4.3. Selección de gráficas

Cuando se haya seleccionado tanto la fuente de datos como el tipo de gráfico a realizar, se debe indicar qué ejes componen la gráfica y qué opciones, de entre las que están disponibles por parte de la fuente de datos, queremos que estén representadas. La figura 4.4 contiene la ventana de la aplicación con los nuevos elementos necesarios para la representación.

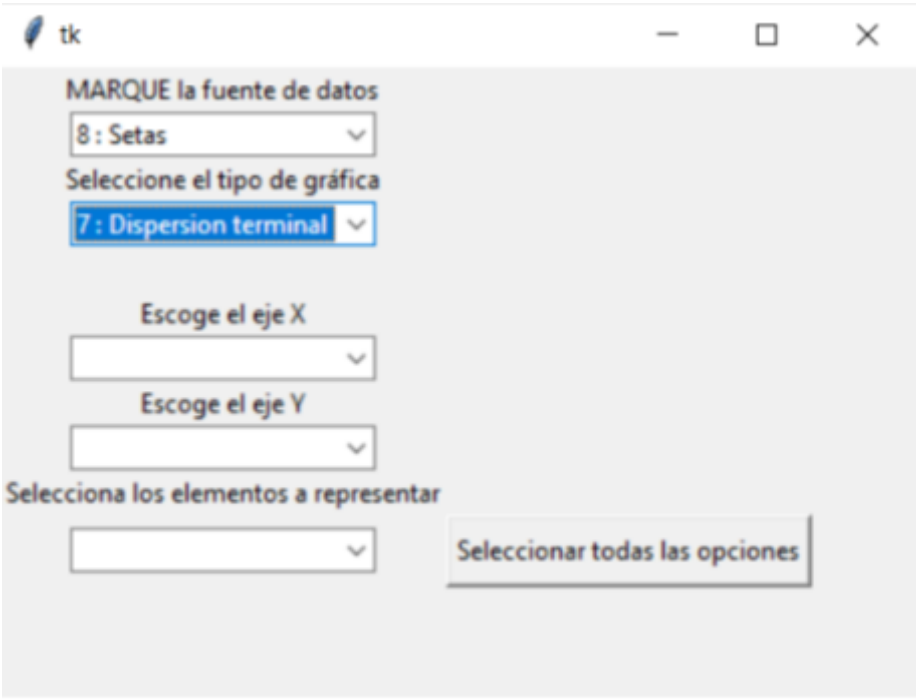



Figura 4.4. Selección Ejes y opciones gráficas

Una vez que el usuario haya seleccionado todos los elementos pertinentes para la representación se pondrá a su disposición un nuevo botón con el que podrá obtener la gráfica resultante con los parámetros indicados anteriormente. De entre las representaciones se encuentran la gráfica de líneas, barras, dispersión, de bigotes e

histogramas. En el punto 4.2 se tratan con mayor profundidad todas las representaciones disponibles. La figura 4.5 muestra la pantalla final una vez el usuario ha seleccionado todos los parámetros.

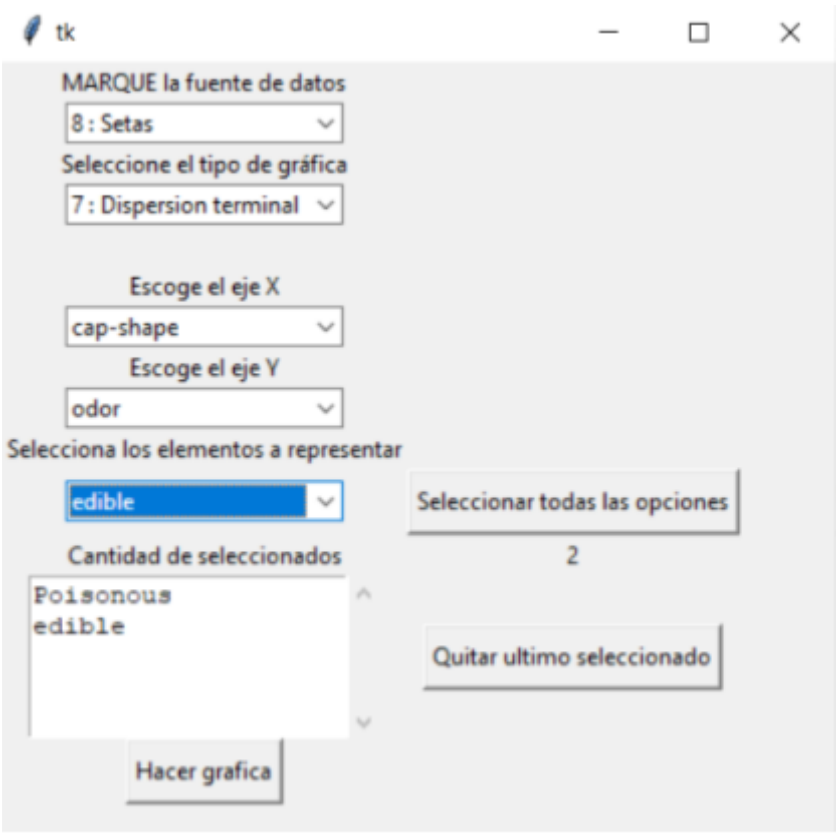



Figura 4.5. Ventana final

El hecho de que los elementos para la representación se vayan presentando por partes solo ocurrirá en el primer flujo de uso, de esta manera presentamos una explicación por partes al usuario de como usar la propia herramienta. Después del primer ciclo de ejecución todos los elementos dispuestos en la ventana se mantendrán pero en caso de modificar la fuente de datos escogida originalmente

## 4.2 Los métodos de visualización

La figura 4.6 muestra las opciones a disposición del usuario para realizar las distintas gráficas. Las representaciones gráficas implementadas se pueden separar en dos grupos, las de representación simple y las que hacen uso del aprendizaje automático para obtener los valores a representar, el primer grupo corresponde a las opciones comprendidas entre 1 y 10, las opciones que se encuentran entre 11 y 22 son las opciones para el grupo que emplea el aprendizaje automático.

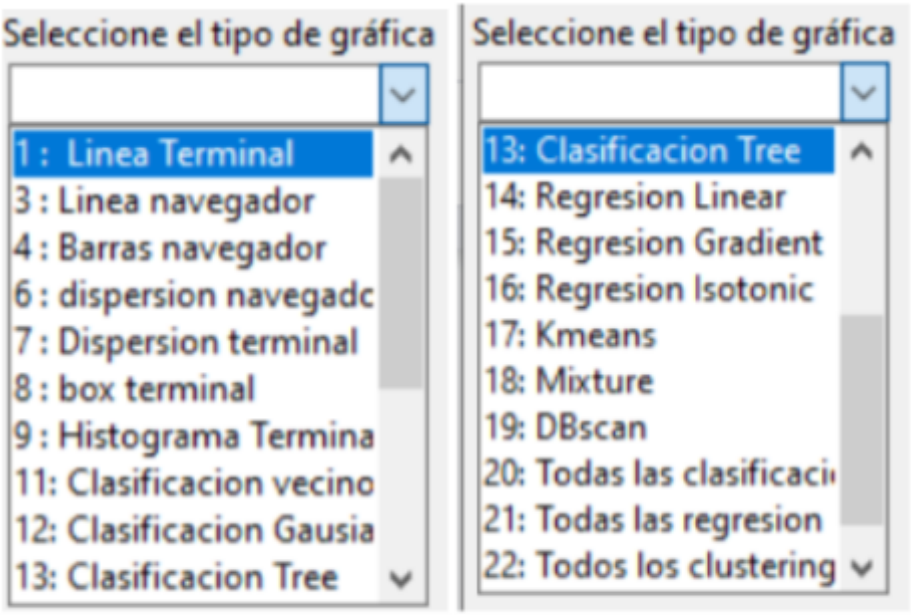



Figura 4.6. Opciones gráficas

Dentro de las gráficas básicas que se encuentran en el proyecto serían las siguientes:

❖ Gráfica de línea: Los gráficos de líneas muestran una serie como un conjunto de puntos conectados mediante una sola línea. Los gráficos de líneas se usan para representar datos a lo largo de un periodo continuo [9]. Para representar dicha línea primero se asignan todos los puntos al espacio de representación y luego conectando una línea entre los puntos previamente representados. De esta manera podemos ver la tendencia de los datos a lo largo del intervalo.

Como se ha mencionado anteriormente esta representación de los datos debe ser a los largo de un intervalo, por lo que un problema que nos podemos encontrar con los datos que trabajamos es que no sigan un intervalo creando una gráfica carente de sentido.

Las opciones que usan esta representación son:

- Línea terminal: Realiza la representación de una gráfica lineal para el escritorio, haciendo uso de la librería matplotlib. En la figura 4.7 puede verse un ejemplo de la gráfica de líneas con matplotlib donde se representa el total de muertes por covid en los Países Japón e Italia. La gráfica muestra cómo ha sido la evolución del total de muertes a lo largo del tiempo en estos dos países.

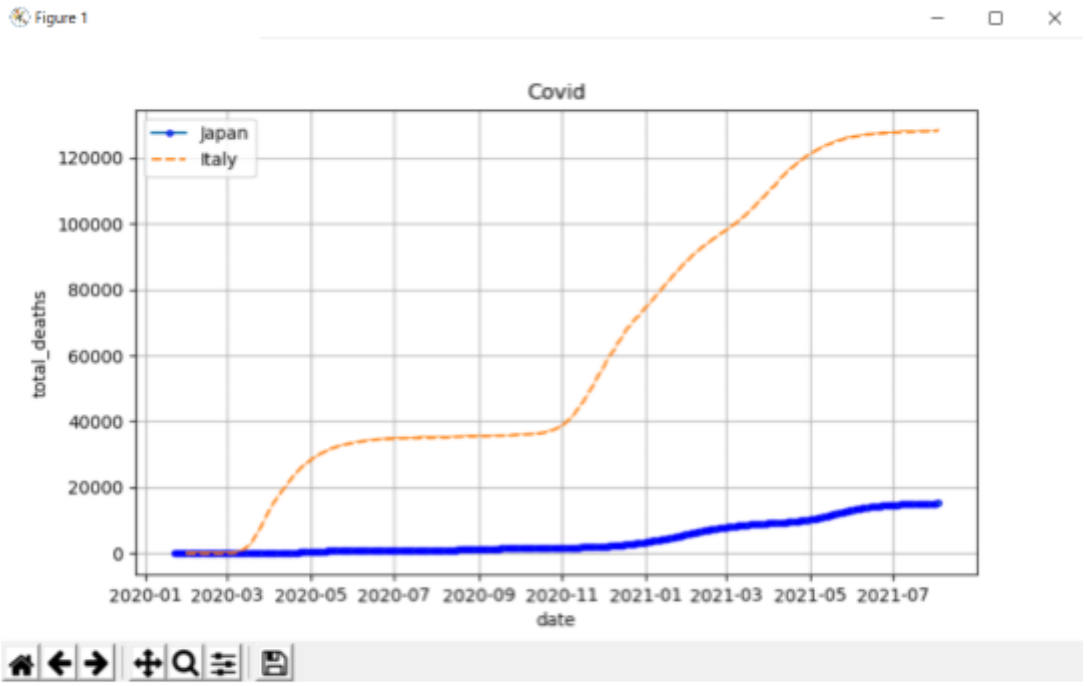



Figura 4.7. Representación lineal con matplotlib.

○ Línea navegador: Realiza la representación de una gráfica lineal presentándola sobre el navegador, haciendo uso de la librería Plotly. En la figura 4.8 se muestra un gráfico de líneas de la librería plotly, la suma total de la cantidad de casos Covid que han sido registrados a lo largo de los meses en los países Finlandia y Japón. Siendo el eje X la fecha y el eje Y los casos totales registrados.

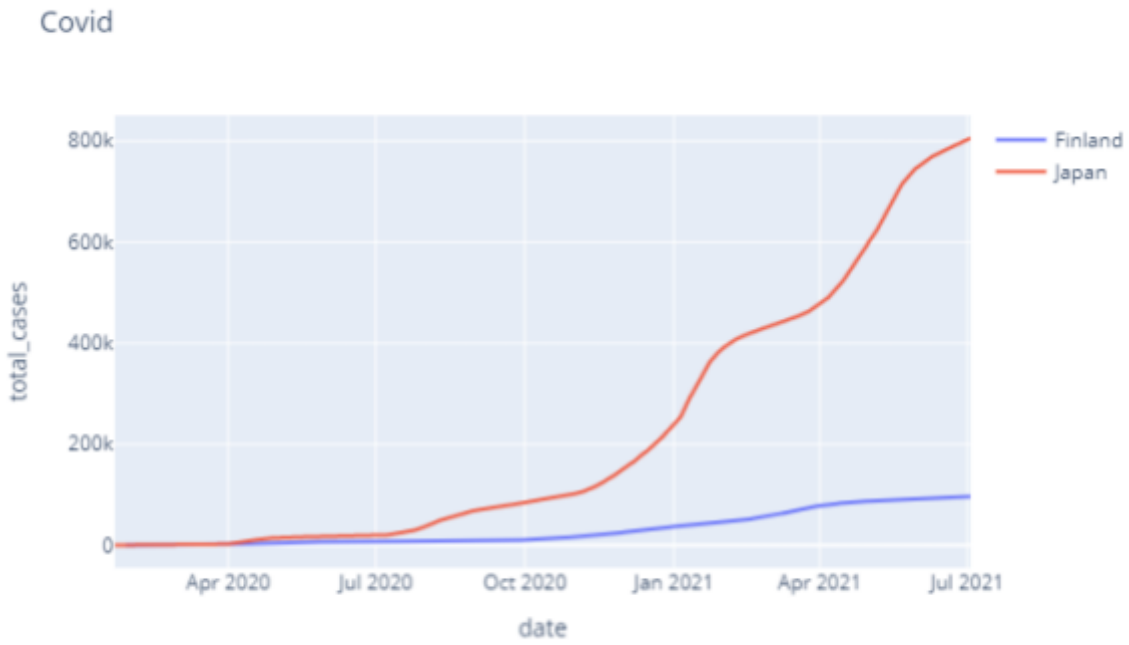



Figura 4.8. Representación lineal con Plotly

- Gráfica barras: Las gráficas de barras nos permiten representar la cantidad de elementos con las que cuentan las diferentes categorías. Esto quiere decir que en uno de los ejes representaremos distintas categorías y en otro eje las cantidades que esta presenta.

El mayor problema que se nos presenta al realizar esta representación nos la encontramos cuando intentamos representar un gran número de categorías, o si la diferencias entre las magnitudes es muy grande ya que esto dificulta la labor de evaluar los datos representados debido al escalado. Esto podría impedir mostrar correctamente los datos de algunas categorías resultando en una mayor dificultad para el estudio de los datos mostrados. El término escalado es mencionando la cantidad de `tickts` que presentan los ejes, a mayor el escalado más fraccionado estará el eje.

Las opciones que usan esta representación son:

- Barras navegador: Representación de gráficos de barras haciendo uso de la librería Plotly. En la figura 4.9 se encuentra la gráfica de barras, realizada con Plotly, en la que está la representación de los accidentes de tráfico en España a lo largo de los meses de los años 2010, 2014 y 2017, siendo 2010 el año en el que más accidentes por mes se observan.

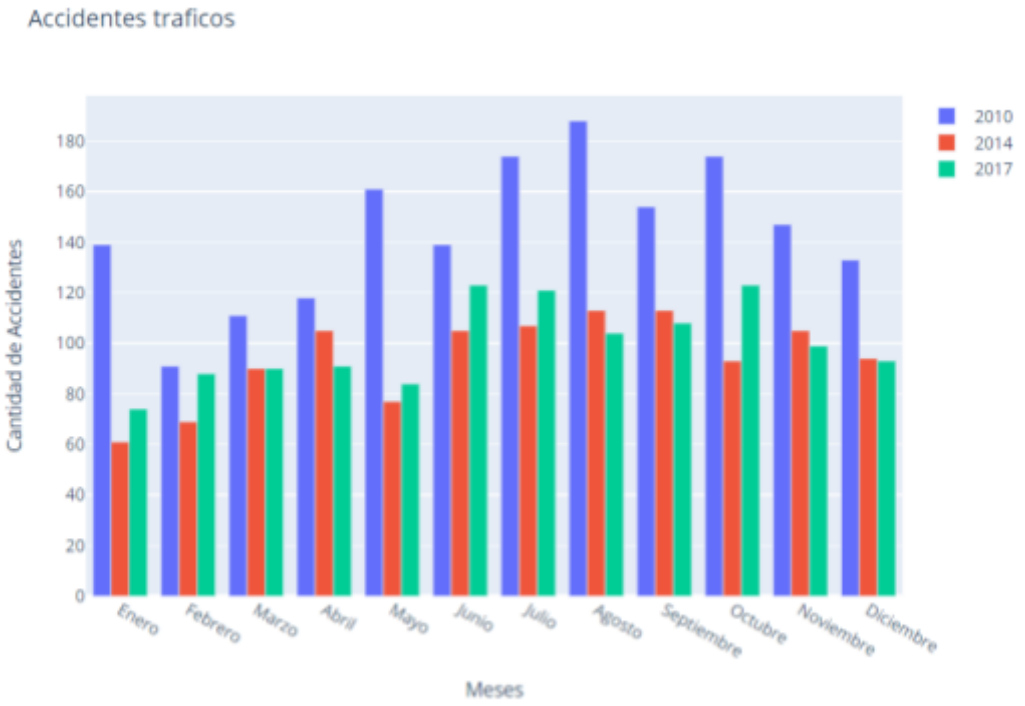



Figura 4.9. Gráfico barras con Plolty

- Gráfica dispersión:

Las gráficas de dispersión realizan la representación de los distintos datos convirtiéndolos en coordenadas para representarlos como puntos a lo largo de un mapa cartesiano.

Esta gráfica nos permite observar si hay alguna relación entre las variables dispuestas en los ejes, esto es así dado que en el mapa de puntos resultante podemos observar

concentraciones o tendencias en los datos. Normalmente se acompaña esta gráfica con una línea de regresión que nos sirve para ver la tendencia de los datos con mayor claridad.

Las opciones que usan esta representación son:

- Dispersión terminal: Representación de una gráfica de dispersión con la librería matplotlib. En la figura 4.10 están dispuestos las estadísticas de vida (HP) y defensa de los personajes de Agua y Roca del juego de pokemon. Indicando cómo progresan y donde se concentran las estadísticas. Los datos de los personajes de Agua a medida que aumenta su vida también presentan un aumento de su defensa por otro lado los personajes de roca a medida que aumenta su vida obtienen una disminución de la defensa.

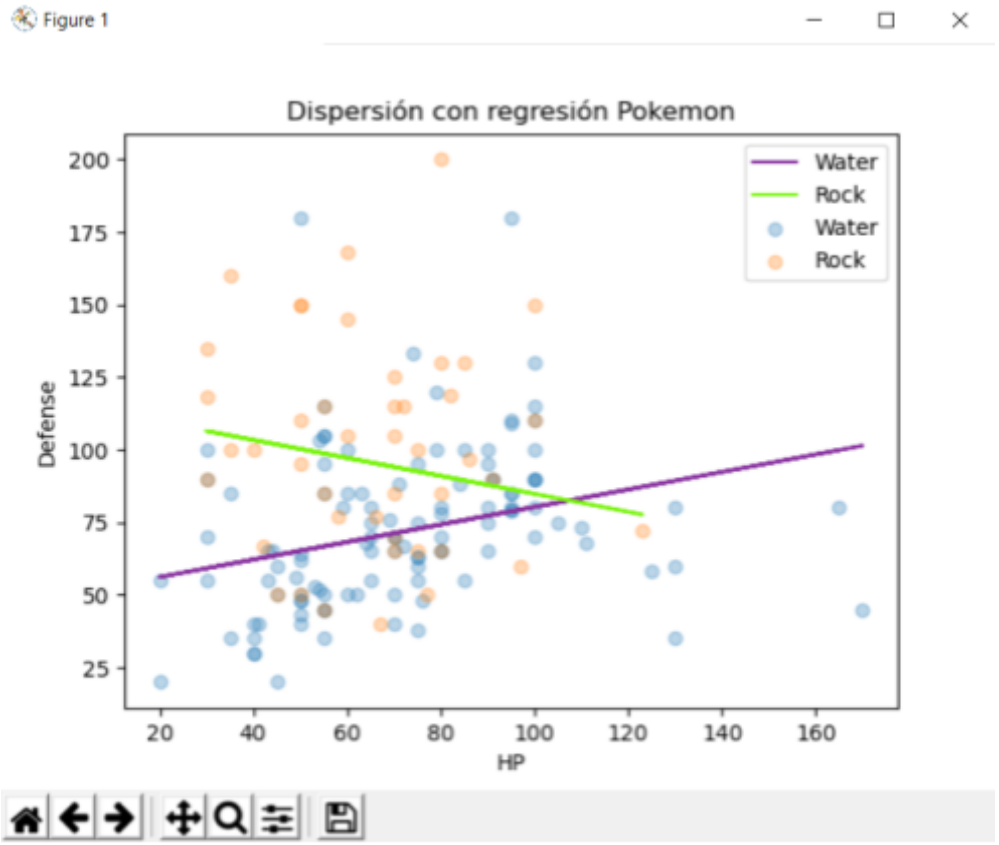



Figura 4.10. Dispersión con matplotlib

- Dispersión navegador: Gráfica de dispersión empleando la librería plotly. En la figura 4.11 está la relación entre presión sanguínea (en el eje y) y la edad (en el eje x) de personas que han sido diagnosticadas de diabetes. Esta gráfica nos permite ver que a medida que aumenta la edad en los casos positivos se presenta una mayor presión sanguínea respecto a los sujetos que dan negativo.


Figura 4.11. Dispersión navegador

- Gráfica de cajas o bigotes:

Los diagramas de bigotes permiten visualizar conjuntos de datos a través de cajas, en las que representaremos sus cuartiles y la mediana, y bigotes para indicar la variabilidad fuera de los cuartiles superior e inferior.

Los cuartiles representados son el primer cuartil, o cuartil inferior que representa el valor mayor al 25% de los datos, el segundo cuartil o mediana, indica dónde se encuentra el valor medio de la serie, y el tercer cuartil, o cuartil superior que nos indicar el valor superior al 75%. Los bigotes representan los datos que se encuentran fuera de los cuartiles y se extienden hasta los límites superiores e inferiores. También existe una última representación y son los datos atípicos que se encuentran fuera los límites superiores o inferiores.

Las opciones que usan esta representación son:

- Box terminal: Representación del gráfico de cajas haciendo uso de la librería matplotlib. La figura 4.12 contiene un diagrama de cajas realizado con matplotlib en el que el eje x representa los usuarios que han dado positivo o negativo en diabetes, mientras que el eje y representa la edad de los pacientes, siendo mayor la media de edad de los pacientes que han dado positivo en diabetes. Como podemos observar la mayoría de los casos positivos se encuentran comprendidos entre la edad de 28 y 45, mientras que los casos negativos entre 25 y 35 años.

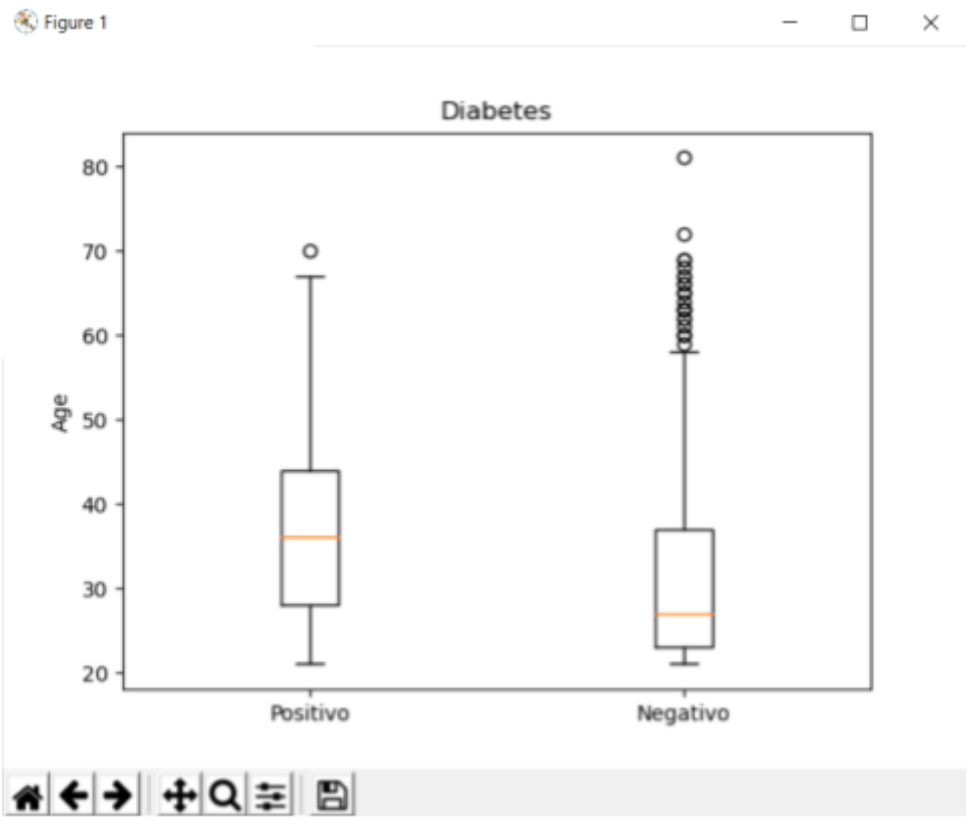



Figura 4.12. Diagrama de cajas con matplotlib

- Histograma:

  Los histogramas nos permiten representar la distribución de los datos a lo largo de un rango de intervalos continuo. Para poder llevar a cabo esta gráfica se separan en intervalos el rango de datos sobre en que se representará y después se indica cuántos datos se encuentran en cada intervalo para obtener la frecuencia y representarlo con una barra. Cada uno de estos intervalos no tiene por que ser del mismo tamaño.

  El histograma lo acompañaremos de una gráfica de tipo línea que nos indicará la densidad del histograma.

  Las opciones que usan esta representación son:

  - Histograma terminal: Representación gráfica del histograma haciendo uso de la librería matplotlib. La figura 4.13 muestra un histograma de pacientes que han dado positivo o negativo en diabetes para los datos mostrados en la figura 4.12. A diferencia en esta representación tenemos como están distribuidos los datos a lo largo de la edad, siendo la posición de cada barra un intervalo de 5 años y la altura de cada barra la concentración de datos. Con lo que concluimos que para los positivos en diabetes la mayoría de datos se encuentran entre 20 y 60 años y para los negativos la gran mayoría entre 20 y 25 años.

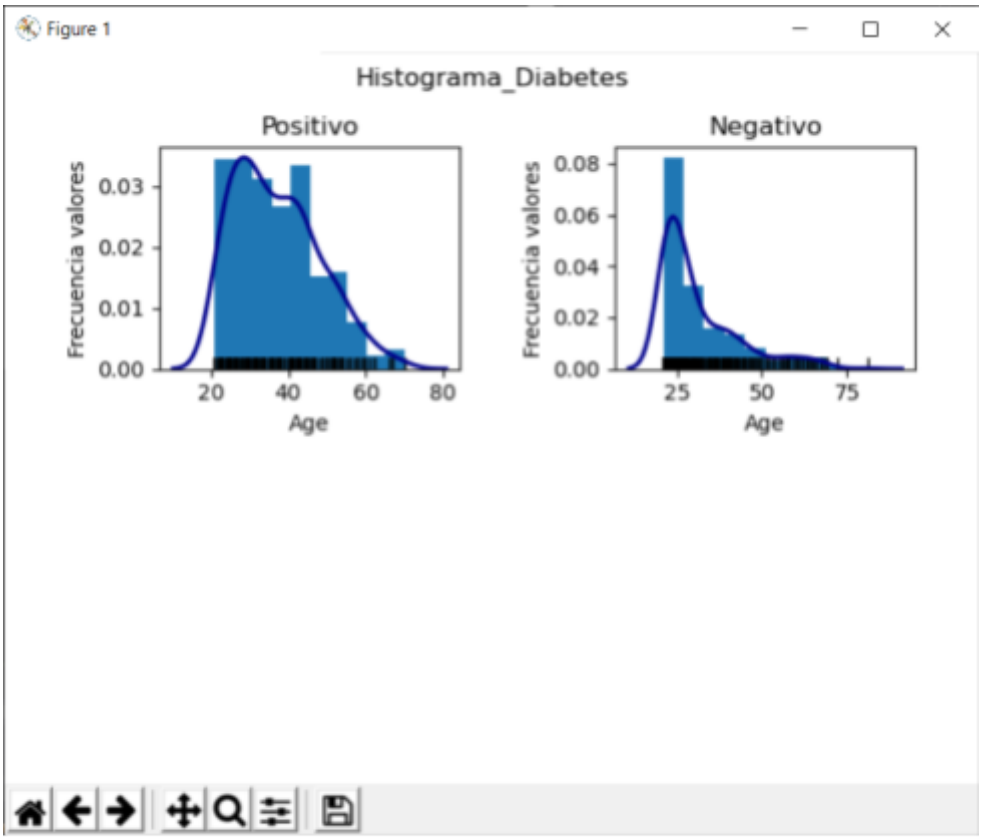



Figura 4.13. Histograma con matplotlib.

## 4.3 Los modelos de Aprendizaje Automático (Machine learning)

El aprendizaje automático es una rama de la inteligencia artificial que se centra en el uso de los datos y algoritmos que imitan la manera en la que los humanos aprenden [10]. Los algoritmos que soportan los modelos de aprendizaje automático están diseñados para emular la inteligencia humana mediante el aprendizaje del entorno. Son capaces de, automáticamente alterar o adaptar su arquitectura a través de la repetición (es decir, la experiencia) para que sean cada vez mejores en la tarea que desean conseguir. El proceso de adaptación se llama entrenamiento, en el que se proporcionan muestras de datos de entrada junto con los resultados deseados. A continuación, el algoritmo se configura de forma óptima para que no sólo pueda producir el resultado deseado cuando se le presentan las entradas de entrenamiento, sino que pueda generalizar para producir el resultado deseado a partir de nuevos datos no conocidos previamente [10].

El aprendizaje automático diferencia dos tipos de estrategias o métodos de aprendizaje: los supervisados y los no supervisados.

Los algoritmos de aprendizaje automático supervisados se definen por el uso de conjuntos de datos etiquetados para entrenar algoritmos que clasifiquen datos o predigan resultados con precisión. A medida que los datos de entrada se introducen en el modelo, éste ajusta sus ponderaciones hasta que el modelo se ha ajustado

adecuadamente, lo que ocurre como parte del proceso de validación cruzada [11].

Los modelos de aprendizaje automático no supervisados utilizan algoritmos para analizar y agrupar conjuntos de datos sin etiquetar. Estos algoritmos descubren patrones ocultos o agrupaciones de datos sin necesidad de intervención humana [12].

### 4.3.1 Regresión

La regresión consiste en la aplicación de métodos estadísticos para modelar la relación entre una variable dependiente y una independiente. Los modelos de regresión son útiles para predecir valores numéricos basados en los valores de entrenamiento [13]. Se trata de un método de aprendizaje automático supervisado. Dependiendo del modelo de regresión que se emplee la gráfica resultante variará, debido a que la forma de calcular la tendencia de datos es diferente en cada modelo. Una vez elegido el modelo será necesario su entrenamiento y posterior validación para obtener la información.

Para representar la información que se obtiene de los modelos de regresión una vez que están entrenados en este proyecto procedemos del siguiente modo: El modelo de regresión una vez entrenado contiene una ecuación con los parámetros asignados de manera que al aplicarlo a un conjunto de datos nos devolverá los valores que se obtienen para cada uno de los datos dentro del conjunto. El conjunto de datos que emplea para la validación del modelo serán los valores del eje X y el conjunto de datos resultantes será el eje Y, con ambos valores obtenemos un conjunto de puntos a representar. Para acompañar la representación de los puntos resultantes se realiza una gráfica de dispersión. Esta representación se realiza por cada una de las opciones escogidas por el usuario.

Adicionalmente para aportar más información de la que nos puede proporcionar el modelo, se adjuntan dos gráficas de tipo barras en las cuales se representa el error cuadrático y absoluto del modelo con los datos empleados.

Las primeras representaciones dispuestas en la ventana emergente a la hora de representar la regresión contienen la gráfica de regresión del método escogido aplicada a las distintas opciones seleccionadas. Las siguientes gráficas se corresponden a la representación del error cuadrático y el error absoluto calculado por el modelo en cada una de las opciones representadas.

La figura 4.18 muestra un ejemplo del resultado de seleccionar la regresión lineal de los países Japón y Finlandia entre los nuevos casos de contagios y las nuevas muertes por Covid. Presentándose una mayor cantidad de muertes por nuevos casos en Japón respecto a Finlandia.

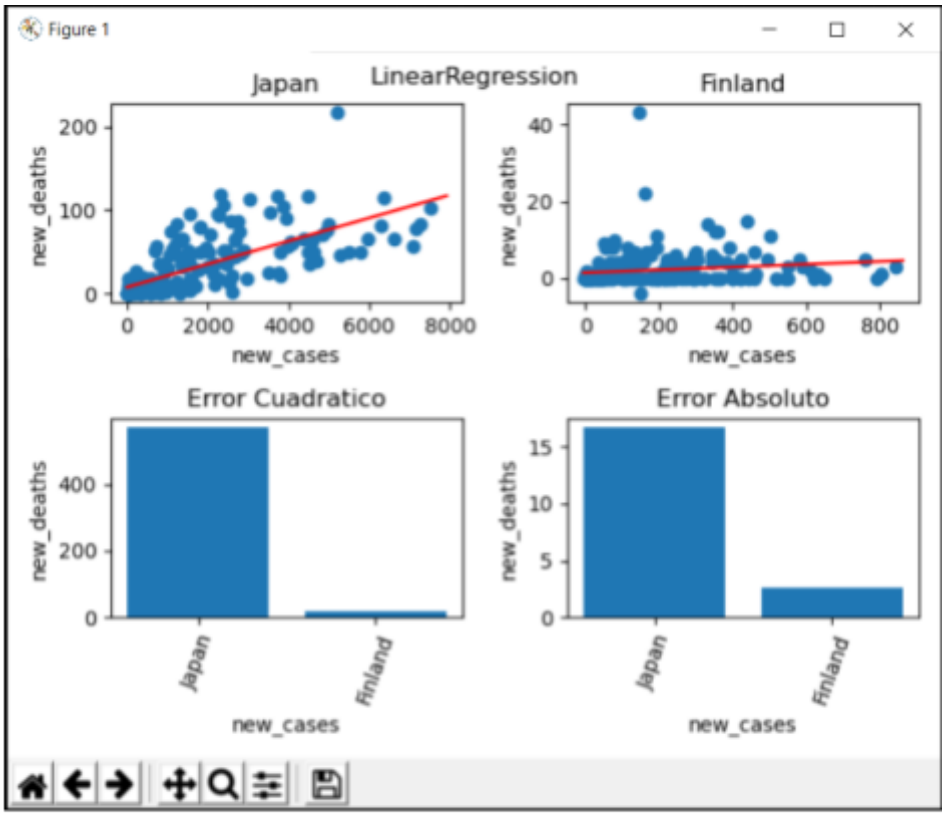



Figura 4.18. Representación regresión final

Los modelos utilizados en este proyecto son la regresión Lineal, la regresión Gradiente Boost y la regresión Isotonic.

- ***Regresión Lineal:***

  La regresión lineal consiste en encontrar la relación entre dos variables continuas obteniendo como resultado una gráfica lineal que sigue la ecuación `Y = mx + n + e`. Siendo `e` el error del modelo y la línea final la que presente el menor error de predicción [14].

  En la figura 4.14 hay un ejemplo de la regresión lineal aplicado a sujetos que presentan o no Diabetes. En los ejes se sitúan la edad y el porcentaje de masa corporal (BMI). Los casos negativos presentan una mayor concentración de sujetos entre 20 y 40 años con una BMI comprendida entre 20 y 40 siendo la progresión constante a lo largo de la edad. Por otro lado, en los casos positivos a medida que avanzan en edad decrece la BMI.

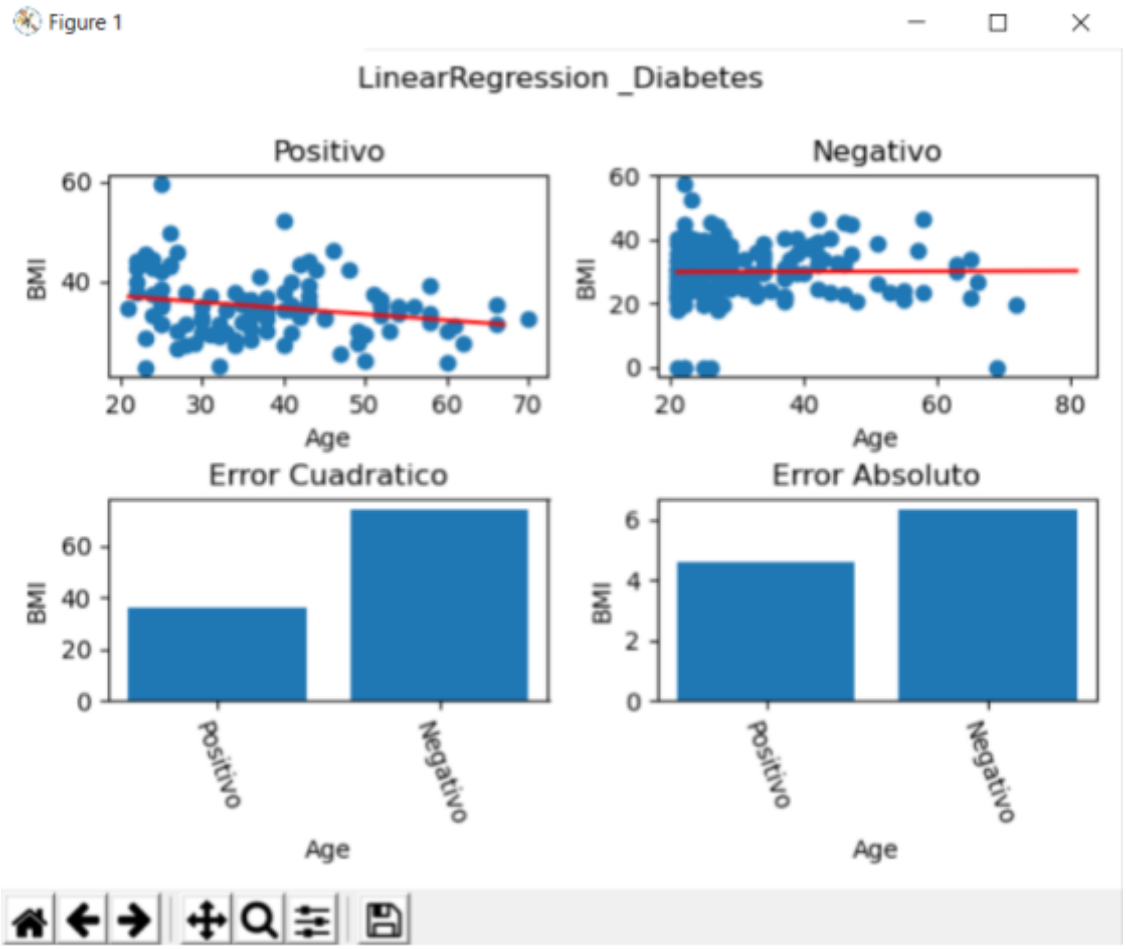



Figura 4.14. Regresión lineal

- ***Regresión Gradient Boost (GBM):***

  El método GBM puede verse como un algoritmo de optimización numérica que tiene como objetivo encontrar un modelo aditivo que minimice la función de pérdida. Por lo tanto, el algoritmo GBM agrega iterativamente en cada paso un nuevo árbol de decisión que reduce mejor la función de pérdida. Siendo más precisos, en la regresión, el algoritmo comienza inicializando el modelo mediante una primera conjetura, que suele ser un árbol de decisión que reduce al máximo la función de pérdida (que es para la regresión el error cuadrático medio), luego, en cada paso, se crea un nuevo árbol de decisión ajustado al residual actual y agregado al modelo anterior para actualizar el residual. El algoritmo continúa iterando hasta que se alcanza un número máximo de iteraciones, proporcionado por el usuario. Este proceso se denomina por etapas, lo que significa que en cada nuevo paso los árboles de decisión agregados al modelo en los pasos anteriores no se modifican. Al ajustar los árboles de decisión a los residuos, el modelo mejora en las regiones donde no funciona bien [15].

  La figura 4.15 muestra un ejemplo de la regresión empleando Gradient Boosting en los casos de Diabetes donde se relaciona la edad con la insulina. Los casos negativos presentan una mayor concentración de valores entre los 20 y 45 años.

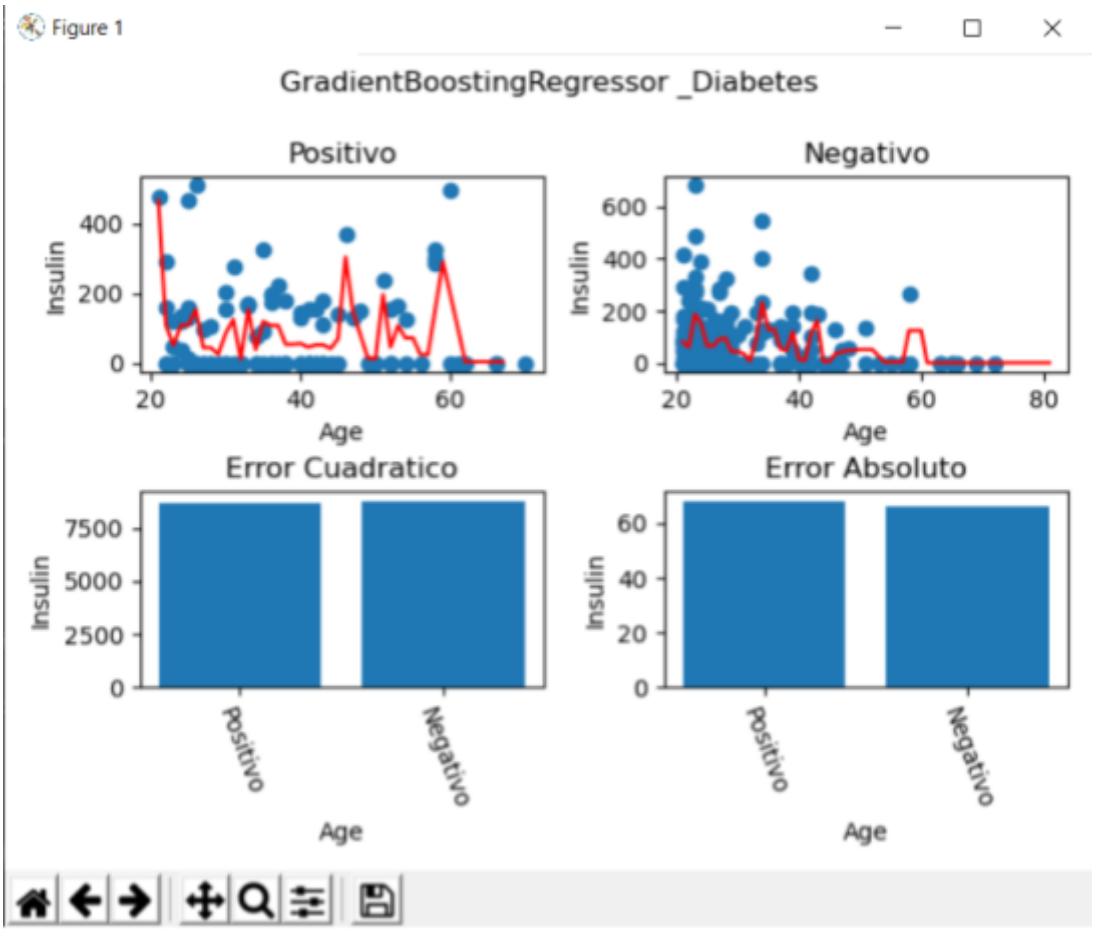



Figura 4.15. Regresión Gradiente Boost

- ***Regresión Isotonic:***

La regresión Isotonic consiste en una regresión lineal en la que la línea de ajuste se configura con una forma más libre. para ello sobre un conjunto de datos finitos Y = $y_1$, $y_2$, ...., $y_n$ representando las respuestas observadas y X = $x_1$, $x_2$, ..., $x_n$, los valores de respuesta desconocidos que se ajustarán para encontrar una función que minimice la fórmula de la figura 4.16 [16].

$$f(x) = \sum_{i=1}^{n} w_i (y_i - x_i)^2$$

Figura 4.16. Fórmula Isotonic

Contiene una restricción para que funcione correctamente y es que tanto la X como la Y han de ser superiores a las del punto anterior, esto es debido a que las gráficas isotónicas son gráficas monótonas. La figura 4.17 muestra un ejemplo de la regresión Isotonic donde se relaciona la edad frente a la glucosa en sangre en los pacientes con diabetes. Presentando una mayor concentración de valores de glucosa en sangre en los casos positivos con una leve elevación de la pendiente a partir de los 60 años, en los casos negativos la glucosa en sangre es menor pero la progresión de regresión presenta un mayor aumento a lo largo de la edad.

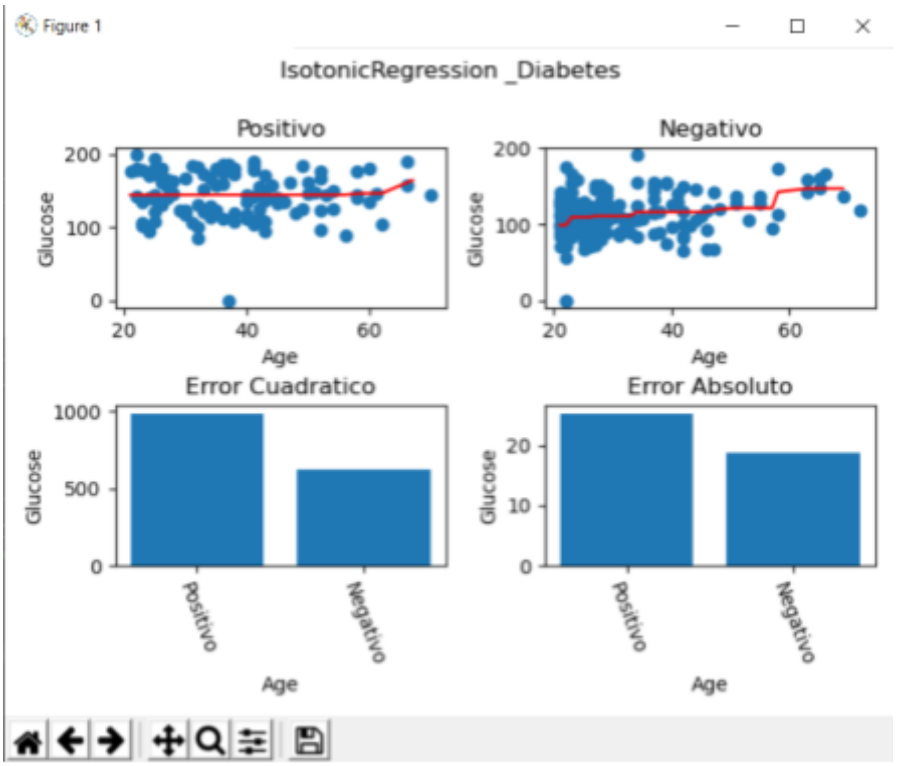



Figura 4.17: Regresión Isotonic

Para poder comparar los 3 métodos implementados se desarrolló un nuevo método que realiza la representación de la regresión de los 3 métodos y las dos gráficas de barras donde se representa el error cuadrático y absoluto en cada uno de los modelos. Por cada opción se abrirá una ventana con estas gráficas indicando en el título de la propia ventana a qué opción corresponde, se diseñó de esta manera para mantener una limpieza en las representaciones. En la figura 4.19 se encuentra la representación del método de comparación de los 3 modelos usando como opción de representación los casos positivos de diabetes y los ejes de representación la edad y la glucosa en sangre. Dentro de las regresiones la Isotonic se mantiene a lo largo de los datos, la lineal tiene una leve pendiente descendente y la GradientBoosting la línea presenta puntos de decrecimiento.

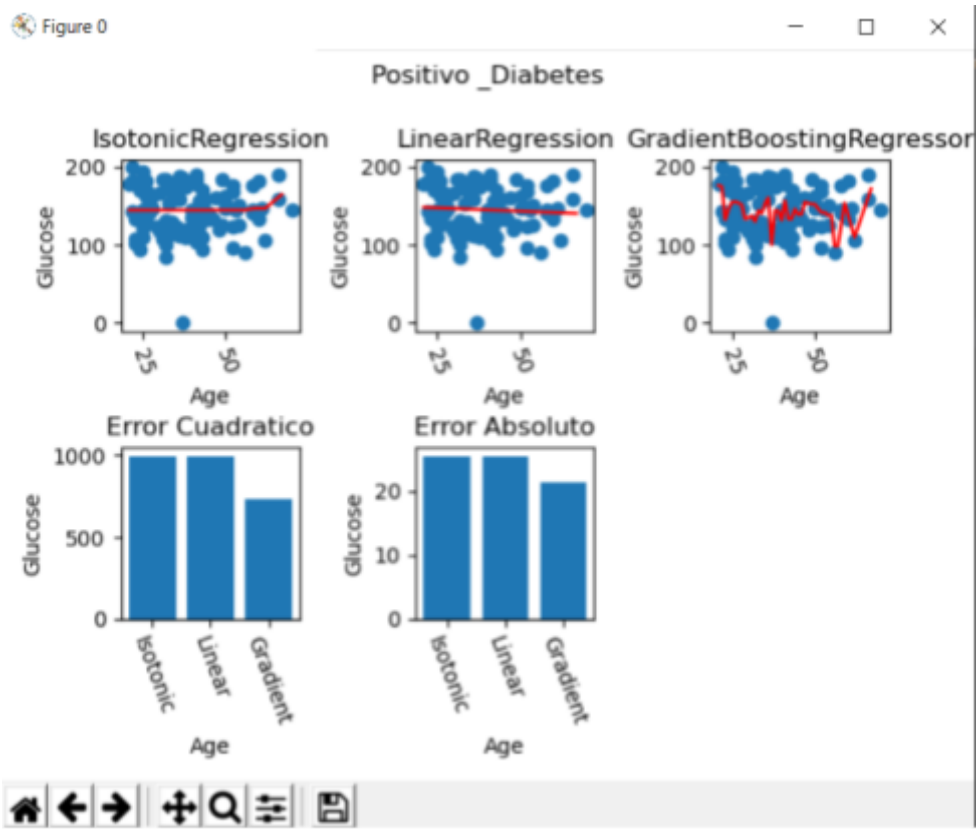



Figura 4.19. Representación de los todos los métodos de regresión

### 4.3.2 Clasificación

La clasificación hace uso de un algoritmo para que de la forma más acertada posible asigne las categorías a los datos. Reconoce entidades específicas dentro del conjunto de datos e intenta extraer conclusiones sobre cómo esas entidades deben etiquetarse o definirse [11]. Cada algoritmo reconoce las distintas categorías de manera distinta. La clasificación, así como la regresión, pertenece al conjunto de técnicas agrupadas dentro del aprendizaje supervisado en aprendizaje automático. Cada modelo de clasificación requiere de un entrenamiento previo antes de poder categorizar las distintas fuentes de datos.

Para poder mostrar los resultados de la clasificación haremos uso de diferentes gráficas, en la figura 4.20 se muestra un ejemplo. Dentro de la ventana de matplotlib las dos primeras gráficas son de dispersión para poder representar los datos que se emplearán para el entrenamiento y para la validación respectivamente. La tercera es una gráfica de líneas en la que se indica cómo de preciso es el modelo dependiendo de lo grande que sea el conjunto de datos de pruebas respecto al conjunto de datos original, los datos restantes se emplearán para la validación del modelo. La cuarta  muestra una gráfica de dispersión donde se representan los resultados del proceso de clasificación obtenido del modelo al llevar a cabo la predicción de las categorías del conjunto de validación.

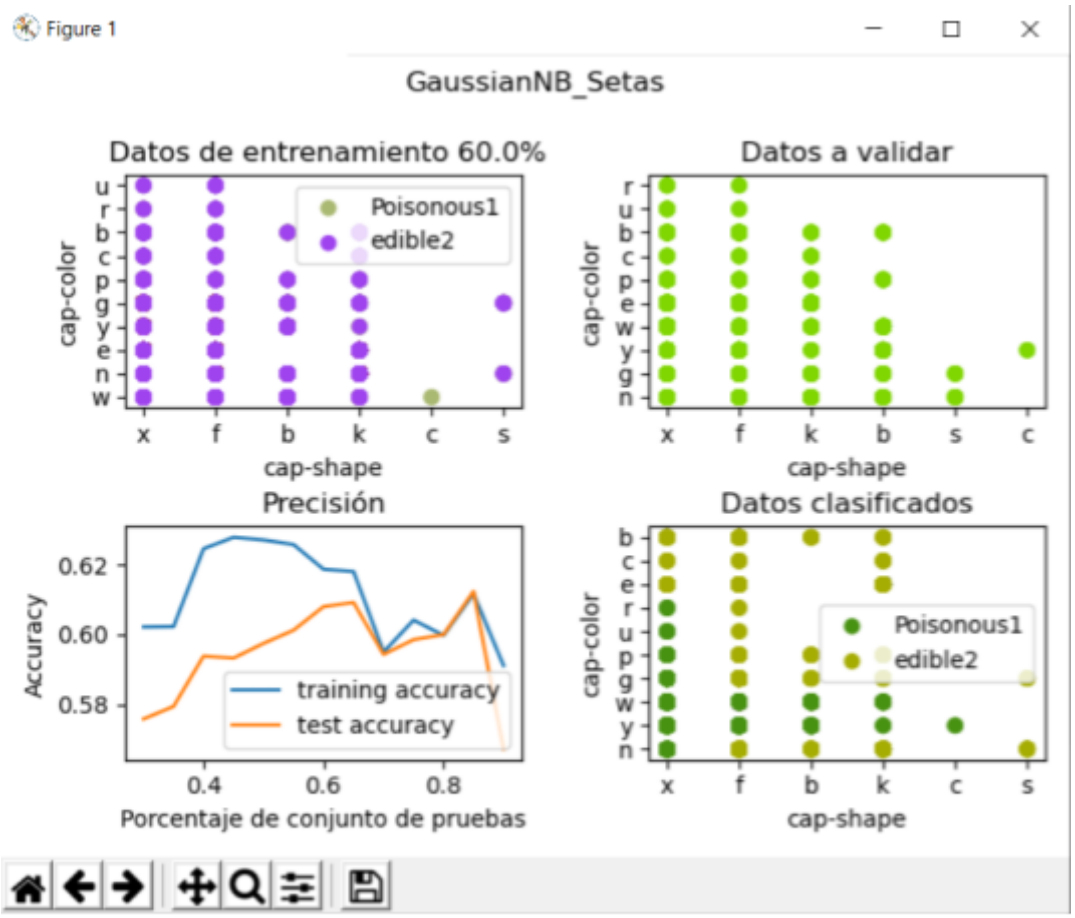



Figura 4.20. Representación clasificación

### *Matriz de confusión*

Un elemento adicional y muy importante a la hora de visualizar los resultados de clasificación y rendimiento de distintos modelos en aprendizaje automático será la matriz de confusión. La matriz de confusión proporciona una medida del desempeño de los modelos de clasificación en el aprendizaje automático.

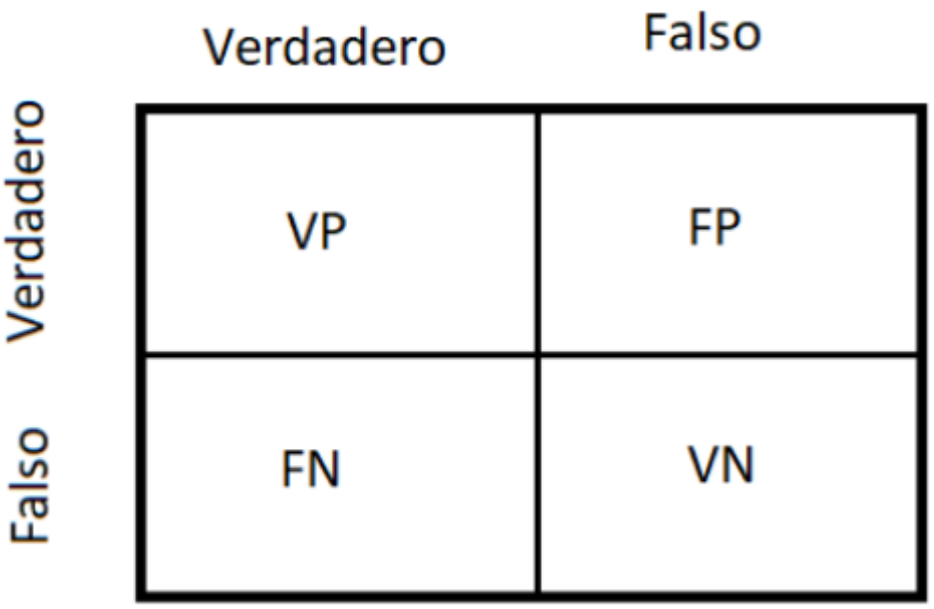



Figura 4.21. Esquema matriz de confusión

La figura 4.21. contiene un ejemplo de la estructura de la matriz de confusión de 2 categorías, el eje X representa las categorías actuales del elemento, el eje Y indica las categorías predichas. Dentro de la matriz de confusión encontramos los 4 casos que se pueden dar, Verdadero positivo, Falso positivo, Verdadero negativo, Falso negativo.

- VP: Los valores positivos que son clasificados correctamente.
- FP: Los valores positivos que son clasificados de forma incorrecta.
- VN: Los valores negativos que son clasificados correctamente.
- FN: Los valores negativos que son clasificados de forma incorrecta.

Los valores de la diagonal principal suelen ser indicadores para analizar las prestaciones de un modelo, ya que si presentan un alto valor significa que de un gran número de casos han sido predichas las categorías correctas de manera satisfactoria.

Una representación más ilustrativa suele combinar la matriz de confusión con un mapa de calor, donde se muestra la matriz coloreando cada celda en función de la cantidad de datos de dicha celda. Además en vez de utilizar la cantidad de casos presentes en cada categoría se emplean los porcentajes.

La librería empleada para llevar a cabo esta gráfica es la de scikit-learn. En la figura 4.22 se encuentra un ejemplo de matriz de confusión que indica cómo de precisa ha sido sobre la predicción de setas venenosas y comestibles, siendo más precisas en los casos que son venenosas.

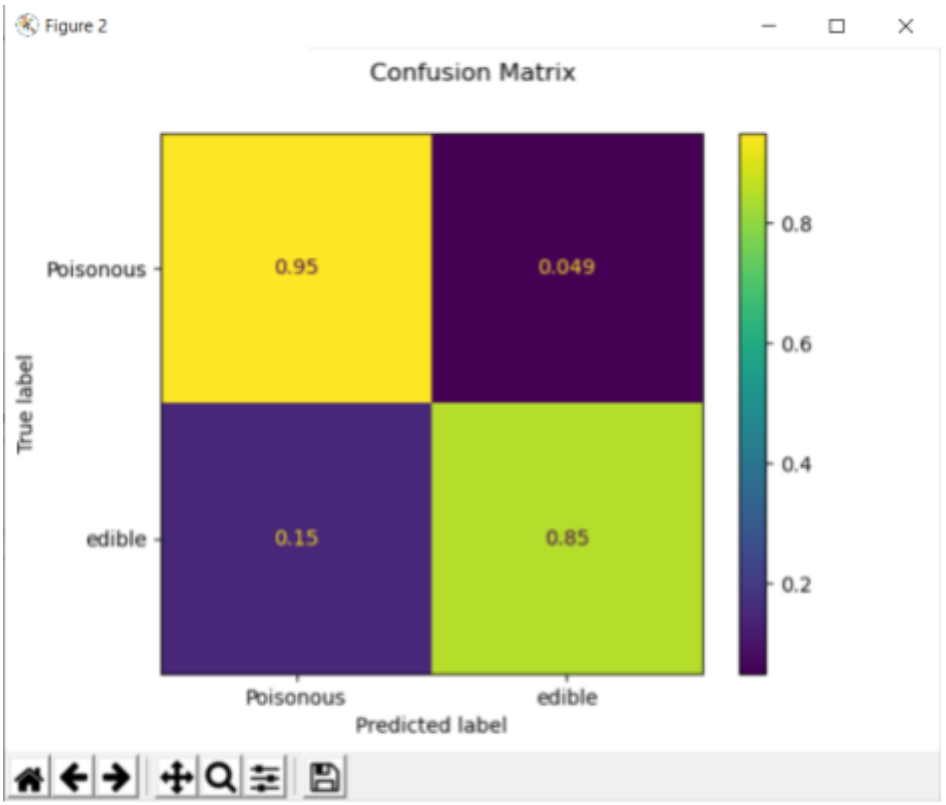



Figura 4.22. Matriz de confusión

La matriz de confusión de la figura 4.22 indica que el modelo ha clasificado de manera correcta la categoría de venenoso un 95%, mientras que la categoría comestible solo el 85% de los casos han sido clasificados correctamente. Por tanto el modelo clasifica de manera más satisfactoria los casos venenosos.

En este proyecto se han empleado tres modelos con los que abordar la clasificación, cada uno realiza la labor de clasificación de manera distinta por lo que para el mismo conjunto de datos podrían expresar resultados distintos, lo tres modelos en cuestión son los siguientes:

- ***K vecinos:***

  El modelo de K vecinos, K nearest neighbors (Knn) en inglés, es el modelo de clasificación más sencillo y conocido de entre estos modelos. La idea detrás de este modelo es que entre elementos similares existe proximidad. Durante la fase de entrenamiento el modelo determina qué características contiene cada categoría y en la fase de validación cada punto será evaluado de manera individual. El funcionamiento del modelo es el siguiente, primero se seleccionan los n vecinos más cercanos, después se comprobará qué categoría tiene asociado cada vecino, se sumarán las cantidades encontradas de cada categoría y la categoría que más vecinos tenga determinará la categoría del punto que estamos evaluando [17].

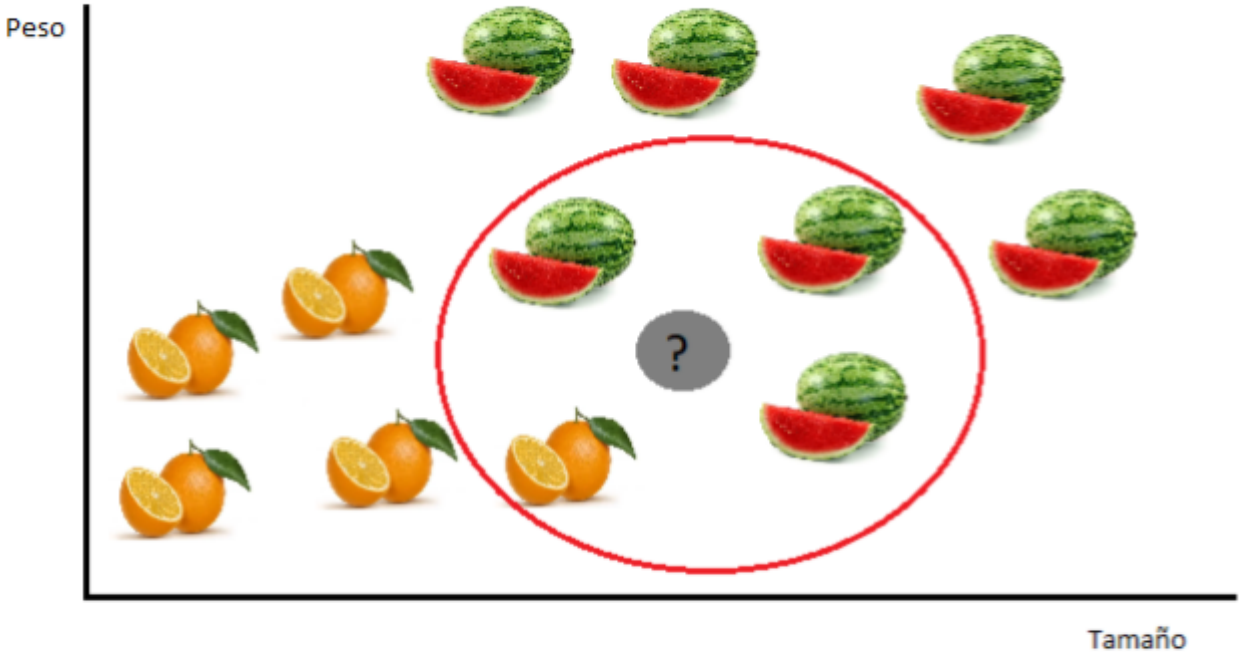



Figura 4.23. Representación ejemplo k vecinos

En la figura 4.23, se encuentra un ejemplo del funcionamiento del algoritmo de k vecinos más cercanos, si quisiéramos determinar la categoría del círculo gris entre sandías y naranjas, teniendo en cuenta los 4 vecinos más cercanos, observamos que 3 de ellos son sandías y 1 es naranja por ello el círculo gris se le asignaría la categoría de sandía.

En la figura 4.24 se muestra una representación gráfica de la aplicación del modelo Knn a datos sobre pacientes. El eje x muestra la presión sanguínea del sujeto y el eje y la edad del paciente. En la figura 4.25 se encuentra la matriz de confusión del modelo K vecinos, en dicha matriz se puede observar que en los casos de diabetes negativos el modelo entrenado tiene un 75% de aciertos y, por otro lado en los casos de positivos sólo el 18% de los casos son correctamente clasificados y el restante 82% son clasificados como positivo. Esto indica que cerca del 80% de los casos serán categorizados como negativos.

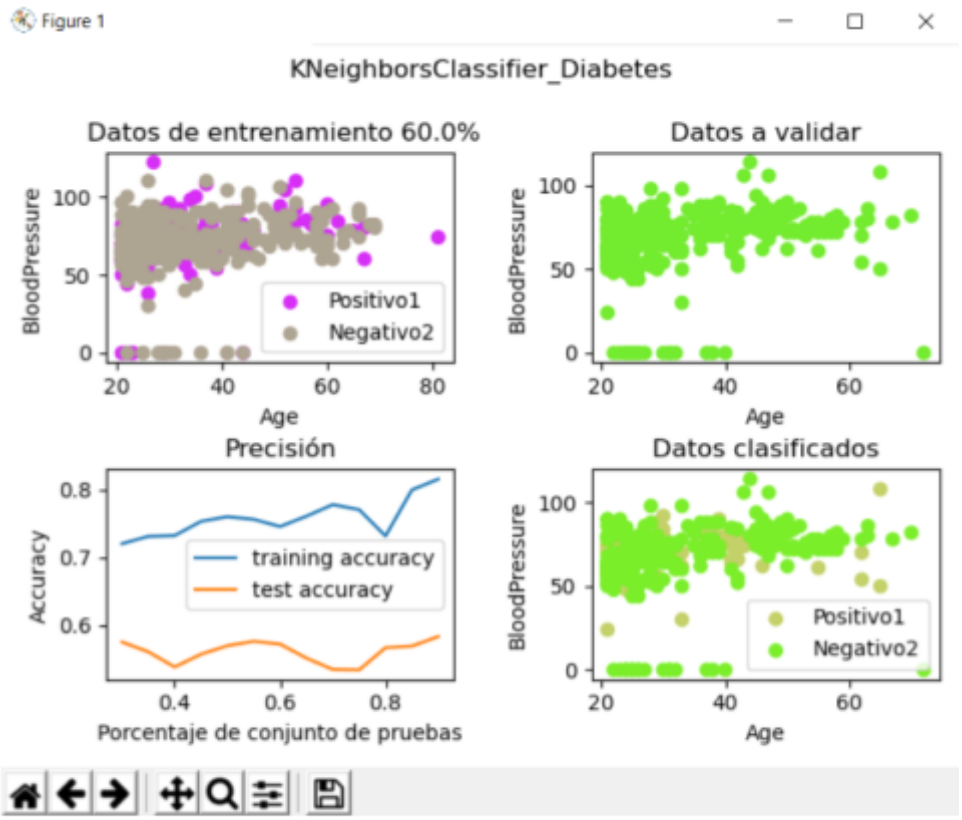



Figura 4.24. Representación knn

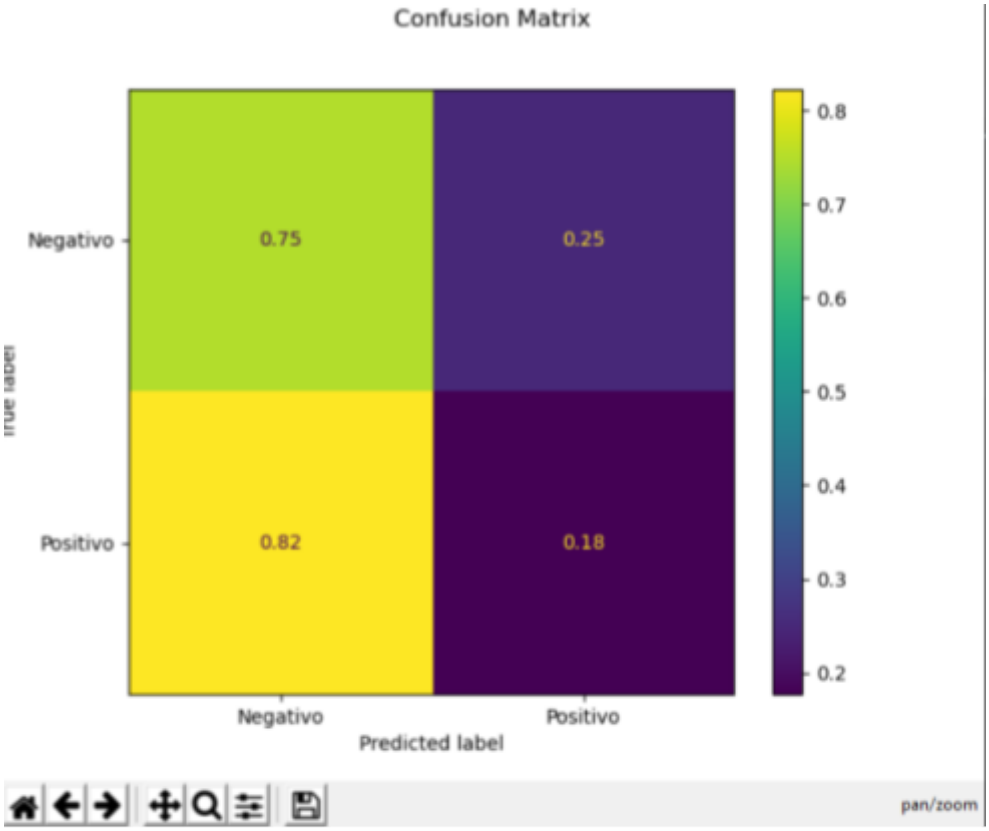



Figura 4.25. Matriz de confusión Knn

- ***Gaussian Naive Bayes:***

Pertenece a la familia de algoritmos de Naive Bayes, se trata de clasificadores de tipo probabilístico que siguen el teorema de Bayes (figura 4.26),

$$P(h|d) = (P(d|h) * P(h)) / P(d)$$

Figura 4.26. Teorema Bayes

Siendo las partes que lo componen el teorema las siguientes:

- ❖ h: la mejor hipótesis
- ❖ d: los datos analizados
- ❖ P(h|d): Es la probabilidad de la hipótesis dado el dato
- ❖ P(d|h): Es la probabilidad del dato siendo la hipótesis correcta.
- ❖ P(h): Es la probabilidad de la hipótesis de ser correcta sin tener en cuenta los datos de entrada.
- ❖ P(d): Es la probabilidad del dato sin contar con la hipótesis.

Como se puede observar lo que interesa es calcular la probabilidad P(h|d) [18].

Lo descrito con anterioridad contiene la referencia a Naive Bayes en el nombre del modelo, la parte de Gaussian hace referencia a que el algoritmo está configurado para trabajar mejor con distribuciones de tipo Gaussiana, para ello sigue la forma de las distribuciones Gaussianas [19].

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Figura 4.27. Fórmula Gaussian Naive Bayes

La fórmula de la figura 4.27 representa el cálculo de la distribución Gaussiana. Dicha fórmula sustituye a P(d|h) de la fórmula de Bayes de la figura 4.26. El proceso para el cálculo de la posible mejor clase sigue siendo igual que el expresado en la fórmula de Bayes.

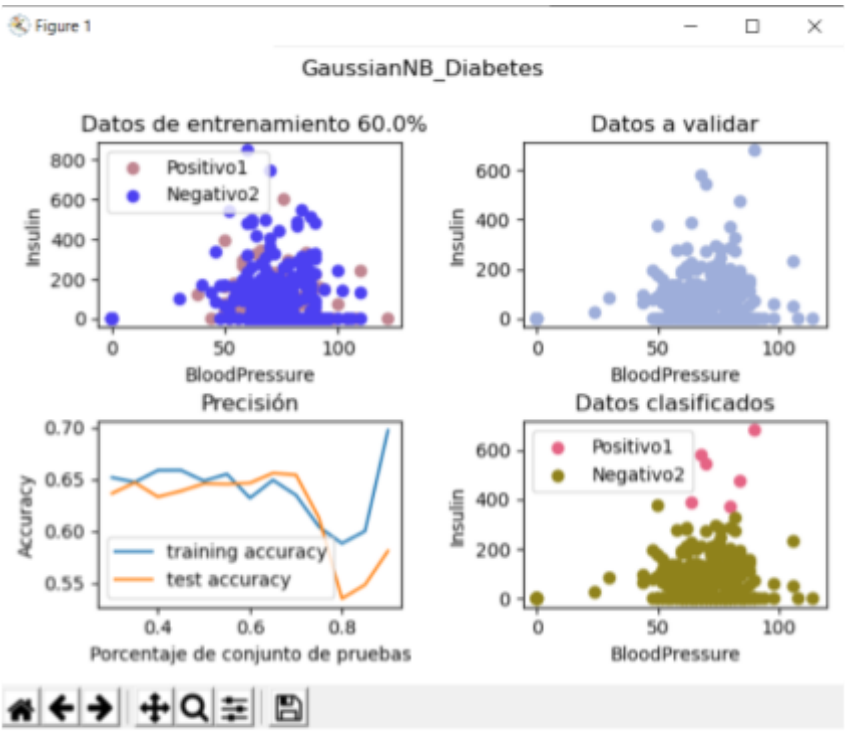



Figura 4.28. Representación Gaussian Naive Bayes

En la figura 4.28 se muestra una representación de la clasificación empleando el modelo de Gaussian Naive Bayes, sobre los datos de pacientes de diabetes, con presión sanguínea e insulina como ejes x e y respectivamente. En la figura 4.29 se encuentra la matriz de confusión sobre estos datos, donde se puede observar que para los datos seleccionados el modelo es muy inexacto a la hora de clasificar los pacientes negativos de diabetes.

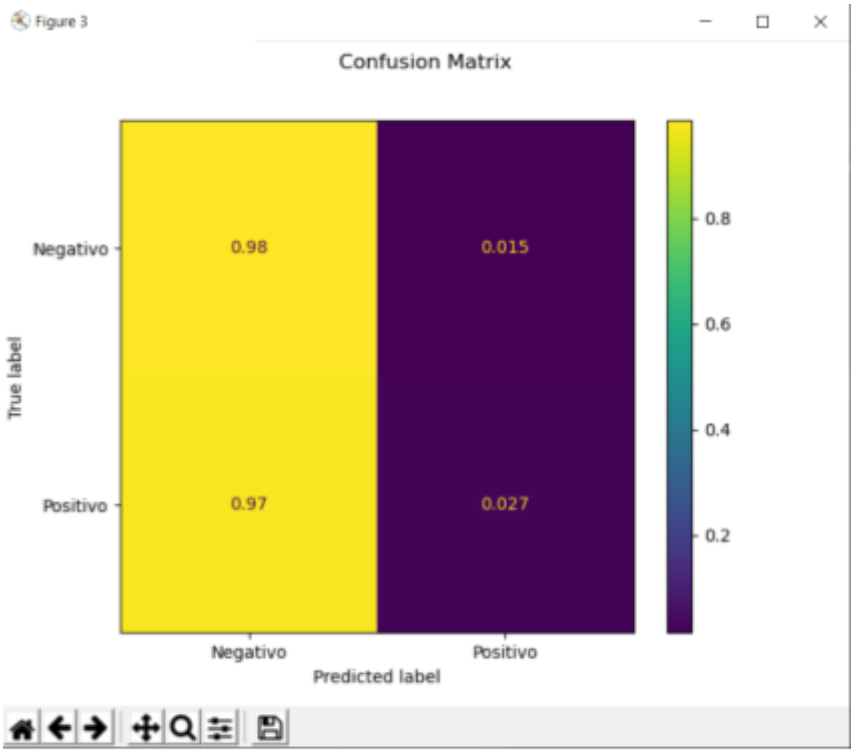



Figura 4.29. Matriz de confusión Gaussian

- ***Tree:***

  En la clasificación de tipo árbol de decisión los datos se van dividiendo siguiendo ciertos parámetros, evaluados en la fase de entrenamiento, para determinar la categoría [20].

  Los árboles de decisión están compuestos por 3 elementos:

  ❖ Nodos: Donde se evalúan los datos para saber a qué rama pasarán.
  ❖ Ramas: Corresponde al resultado de un nodo y conecta con el siguiente nodo o nodo hoja.
  ❖ Nodos hoja: Nodos terminales que determinan la categoría del dato.

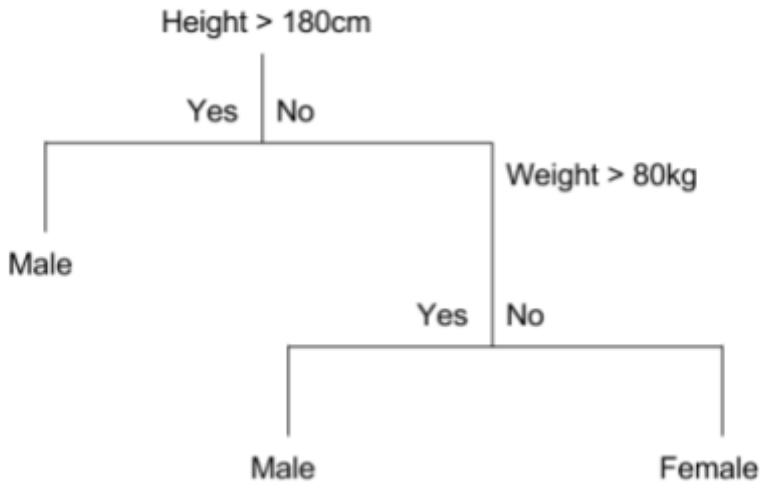



Figura 4.30. Esquema algoritmo árbol

En la figura 4.30, se encuentra un ejemplo de cómo funciona el algoritmo que emplea este modelo. Los datos se evalúan uno a uno en los nodos, dependiendo de la evaluación del nodo van a una u otra rama, así continuarán hasta que lleguen a un nodo hoja donde se concretará la categoría asignada.

En la figura 4.31. se muestra una representación del modelo de clasificación Tree, sobre los datos de pacientes con diabetes, con los ejes Presión sanguínea y edad como ejes x e y respectivamente.

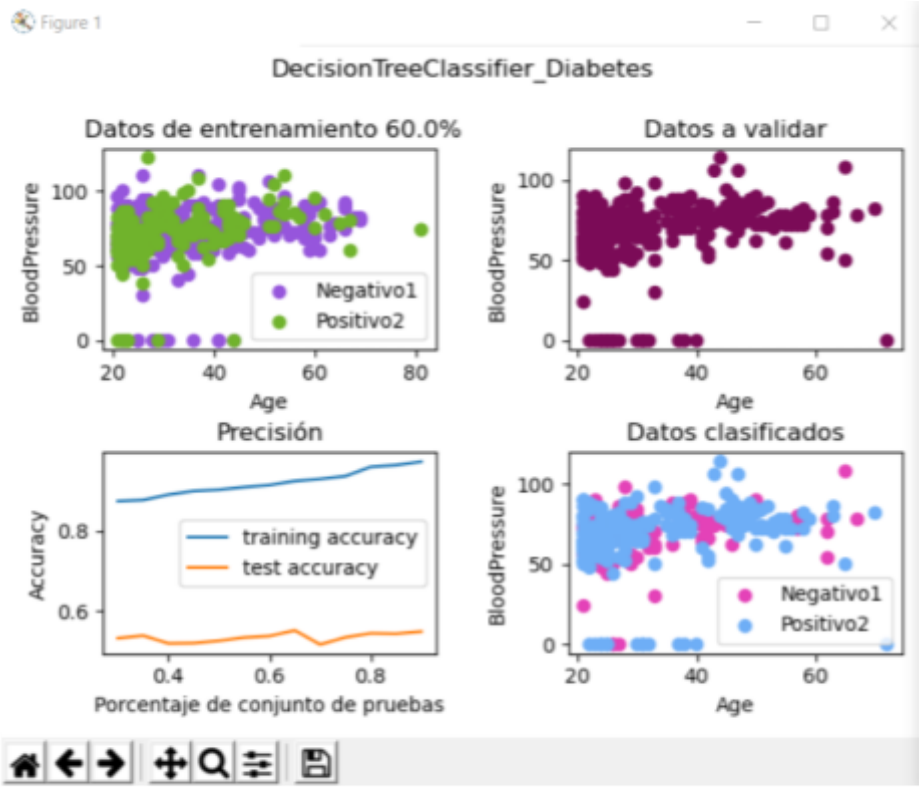



Figura 4.31. Representación modelo Tree

En la figura 4.32 se encuentra la matriz de confusión sobre estos datos. Presenta unos resultados similares a los del modelo K vecinos en la figura 4.24 y mejores que en el modelo anterior de la figura 4.29. Respecto a K vecinos es ligeramente peor clasificando los casos negativos pero levemente mejor en los casos positivos. Aun así los 3 modelos han presentado bajísimos porcentajes de aciertos sobre los casos positivos por tanto los datos seleccionados no son muy buenos para ser sometidos a clasificación.

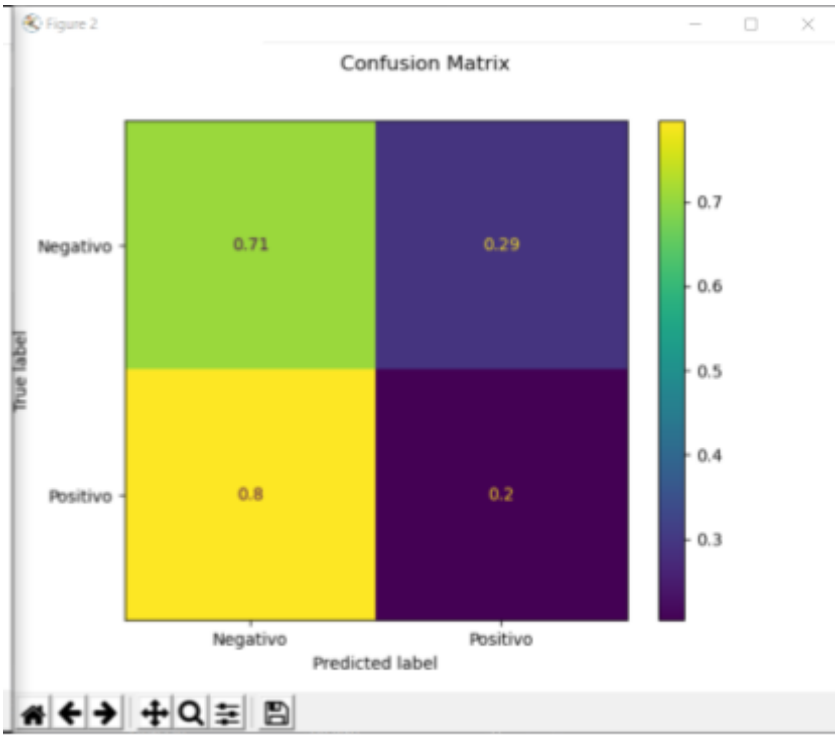



Figura 4.32. Matriz de confusión modelo tree

Del mismo modo que los modelos de Regresión, se ha implementado un método de representación donde es posible realizar la comparativa de los 3 modelos de clasificación mencionados previamente.

Dicho método consiste en representar los resultados de clasificación haciendo uso de una gráfica de dispersión para mostrar los elementos clasificados y una gráfica de líneas para mostrar la precisión de los datos dependiendo del tamaño del conjunto de datos empleado. Estas dos representaciones se aplicarán para los 3 modelos.

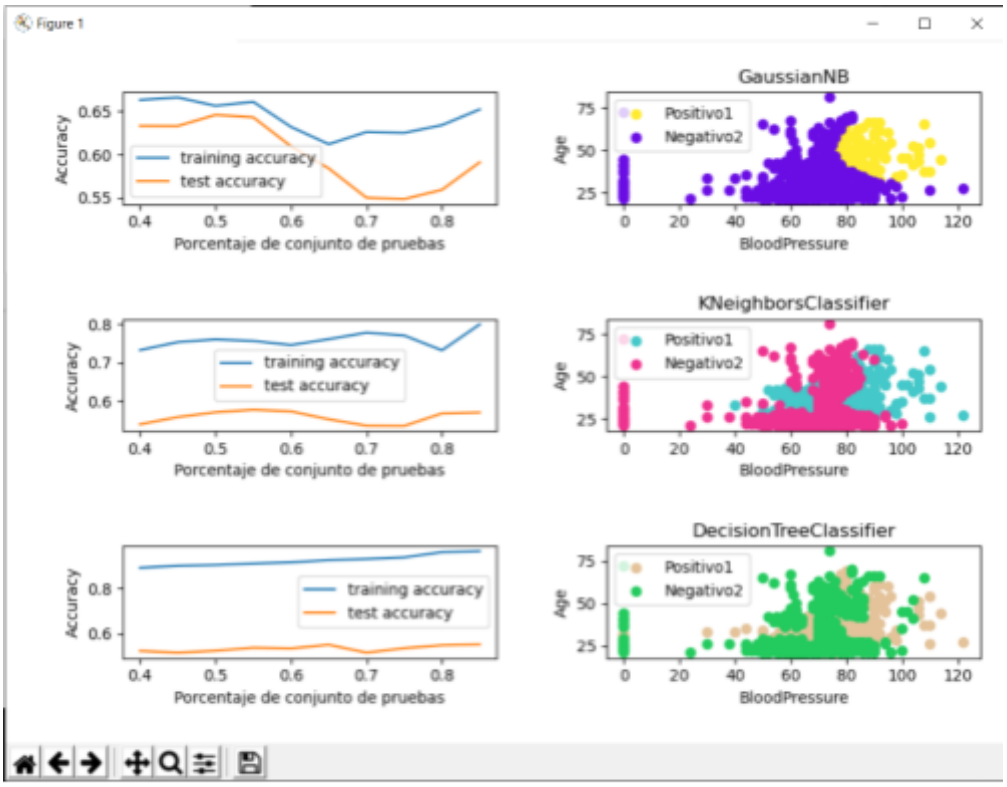



Figura 4.33. Método de representación de los 3 modelos de clasificación

En la figura 4.33 se muestran las gráficas del método de representación de los 3 modelos de clasificación implementados sobre los datos de pacientes de diabetes con presión sanguínea y edad como ejes x e y. En la comparativa se puede observar como el modelo de GaussianNB clasifica a los sujetos de todas las edades con presión sanguínea menor a 80 como casos negativos, y para los sujetos con presión superior a 80 y más de 40 años como positivo. En los otros modelos la clasificación de casos positivos para usuarios con presión superior a 80 se acerca bastante a la de GaussianNB pero además estos modelos clasifican como positivos sujetos con presión entre 50 y 70 y una franja de edad de entre 30 y 40.

### 4.3.3 Clustering

La agrupación en `clusters` puede considerarse uno de los más importantes problemas de aprendizaje sin supervisión; se trata de encontrar una estructura en una colección de datos sin etiquetar. Por tanto, un cluster es una colección de objetos que son "Similares" entre ellos y son "Diferentes" a los objetos que pertenecen a otros grupos.

Los algoritmos de agrupación de datos pueden ser jerárquicos o en partición. Los algoritmos jerárquicos encuentran `clusters` sucesivos utilizando `clusters` previamente establecidos, mientras que los algoritmos que particionan determinan todos los `clusters` a la vez. Los algoritmos jerárquicos pueden ser aglomerativos (ascendentes) o divisivos (descendentes). Los algoritmos aglomerativos comienzan con cada elemento como un `clúster` separado y los fusionan en grupos sucesivamente más grandes. Los algoritmos divisivos comienzan con todo el conjunto y proceden a dividirlo en sucesivamente grupos más pequeños [21]

Para representar los `clusters` resultantes del modelo de `clustering` se emplearán dos gráficas de dispersión, en la primera de las gráficas de dispersión se mostrará el conjunto de datos original, indicando mediante colores cuales son las categorías originales a las que está asociado cada dato, en la segunda gráfica de dispersión se representarán los mismos datos pero ahora los colores indicarán a qué `cluster` se encuentran asociados.

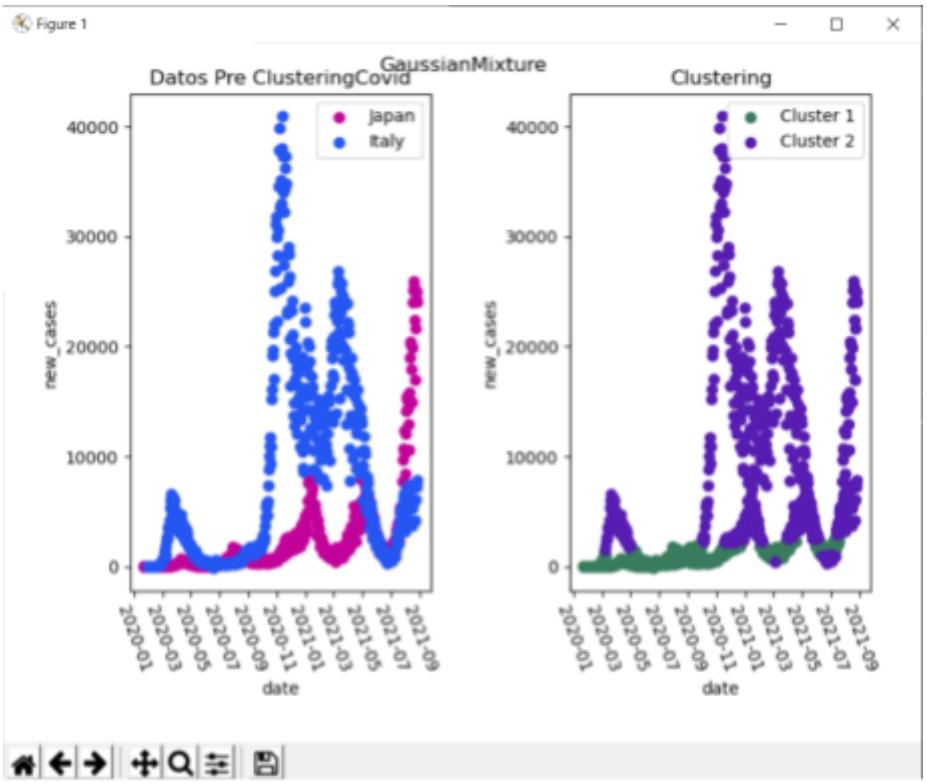



Figura 4.34. Dispersión en los métodos de clustering.

En la figura 4.34. Hay un ejemplo de la representación de los métodos `cluster`. Están representados los datos de casos covid, donde se comparan la cantidad de nuevos casos por fecha para los países Italia y Japón. El método empleado para el cálculo de los `clusters` fue Gaussian Mixture, que se describe posteriormente. En la gráfica está representada la progresión de los nuevos casos de covid, ambos países tienen picos que representan las distintas “olas” que han sufrido a lo largo de la pandemia, a la hora de `clusterizar` el modelo determina un `cluster` en la parte baja de los nuevos casos siendo todo lo restante atribuido al segundo `cluster`.

En lo que respecta a los diferentes modelos de `clustering`, los 3 modelos implementados en este proyecto son KMeans, Gaussian Mixture Models y DBScan.

- ***K means:***

  Es un método duro de `clustering`, esto quiere decir que asociará a cada punto uno y solo un `cluster`, sin tener en cuenta elementos como la probabilidad. Este modelo lo primero que calcula es el centro de cada uno de los `clusters` y posteriormente los datos que componen cada `cluster`. Esto lo realiza primero

calculando los `centroids` de los k `clusters` de manera aleatoria, y de manera iterativa optimizando la posición de los `centroids` hasta que ya no se pueda optimizar más [22].

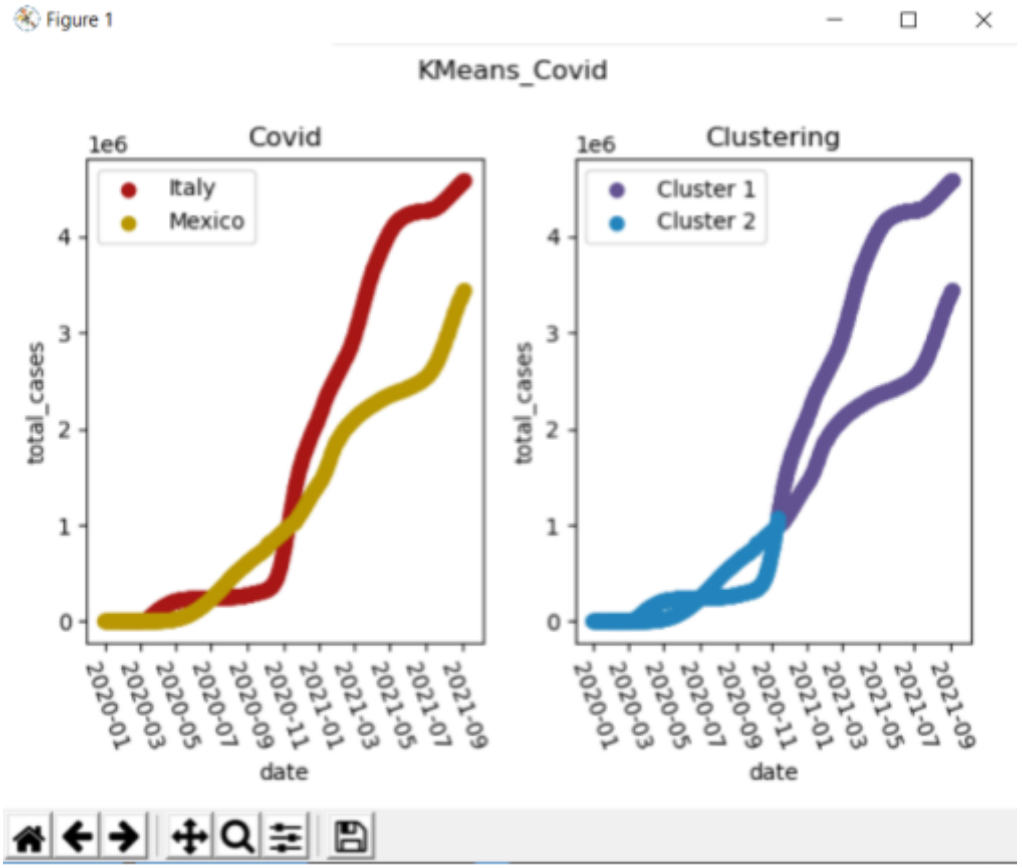



Figura 4.35. Representación k- Means

En la figura 4.35, se encuentra la representación de este modelo sobre los datos de covid, estando la fecha en el eje X y total de casos en el eje Y, por ello se representa el avance de casos a lo largo del tiempo en los países Italia y México. En este caso la gráfica crece de manera ascendente y similar para ambos países. En los resultados del `clustering`, el modelo separa cuando el total de casos no ha presentado el grán aumento en un `cluster` y la parte con un mayor total de casos el segundo `cluster`.

- ***Gaussian Mixture models (GMM)***

  Este modelo es un modelo probabilístico que asume que todos los puntos proceden de la mezcla de un conjunto finito de distribuciones Gaussianas (figura 4.36). Puede verse como una generalización del método de `clustering` k-means en el que se incorpora información sobre la estructura de la covarianza de los datos así como el centro de las Gaussianas.

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\sigma})^2}$$

Figura 4.36. Distribución gaussiana

El modelo de `Gaussian mixture` se parametriza mediante dos tipos de valores, los pesos de los componentes de la mezcla y las medias y varianzas / covarianzas de los componentes. Para un modelo de mezcla gaussiana con *K* componentes, cada *k* tiene una media $\mu_k$ y una varianza $\sigma_k$. Los pesos de los componentes de la mezcla se definen como $\phi$ con la restricción de que el sumatorio de todos los pesos ha de ser igual a 1 ($\sum_{i=1}^{k} \phi = 1$) de manera que el total de la probabilidad de la distribución se normaliza a 1. Sabiendo esto, se puede obtener la fórmula que emplea el método y se encuentra en la figura 4.37 [23].

$$p(x) = \sum_{i=1}^{K} \phi_i \mathcal{N}(x \mid \mu_i, \sigma_i)$$

$$\mathcal{N}(x \mid \mu_i, \sigma_i) = \frac{1}{\sigma_i\sqrt{2\pi}} \exp\left(-\frac{(x-\mu_i)^2}{2\sigma_i^2}\right)$$

$$\sum_{i=1}^{K} \phi_i = 1$$

Figura 4.37. Fórmula Gaussian mixture model

Este modelo emplea la técnica de `expectation maximization`, en adelante EM, para calcular los parámetros necesarios en la fórmula. EM es una técnica numérica iterativa para la estimación de máxima verosimilitud, y generalmente se usa cuando se pueden calcular expresiones de forma cerrada para actualizar los parámetros del modelo (que se mostrará a continuación).

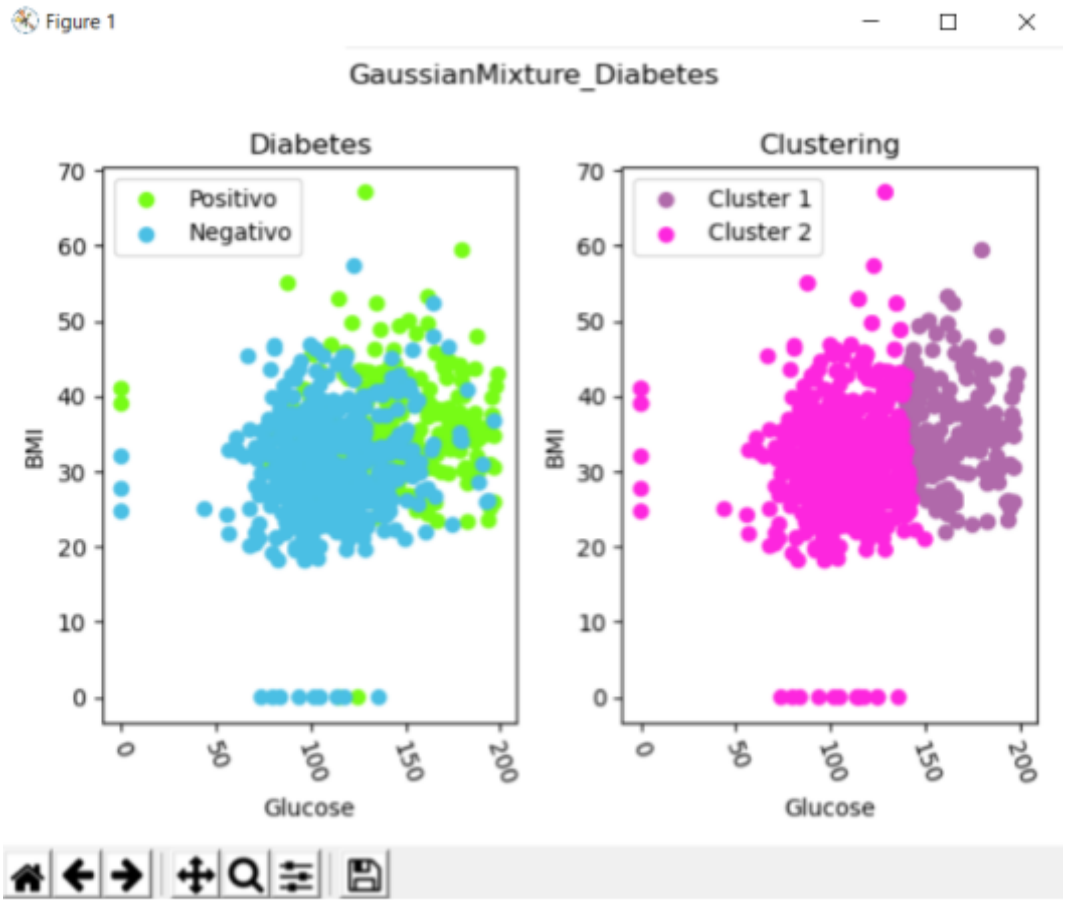



Figura 4.38. Representación Mixture

La figura 4.38 contiene la representación gráfica del modelo Gaussian mixture model haciendo uso de gráficas de dispersión, en las que se representan los datos de casos de diabetes con la glucosa y el BMI en los ejes X e Y respectivamente. La gráfica muestra las características de glucosa y BMI en los casos de diabetes siendo menor la cantidad de glucosa y levemente inferior el BMI para los sujetos negativos en diabetes. En el `clustering` se encuentran dos `clusters`, siendo el primero todos los sujetos con glucosa menor a 150 y el segundo los usuarios con glucosa superior a 150. Por tanto los clusters resultantes del modelo se acercan a la categorización original de los datos.

- ***DBSCAN***

  La primera parte del nombre de este modelo (`Density-based`) se refiere a un método de aprendizaje sin supervisión que distingue diferentes grupos en los datos siguiendo la idea de que los `clusters` son regiones de datos con alta densidad de puntos separados por espacios con baja densidad de puntos.

  DBSCAN es un modelo basado en la densidad que permite calcular agrupaciones de datos o `clusters` teniendo en cuenta el ruido y los valores `outliers`. Para ello precisa de dos parámetros:

  ❖ minPts: El mínimo de puntos agrupados en una región para ser considerado como de alta densidad y por tanto `cluster`.

  ❖ eps(ε): La función de distancia que se emplea para localizar la vecindad de

cualquier punto.

El algoritmo de este modelo funciona de la siguiente manera; Primero de manera arbitraria se selecciona un punto del `dataset`. Segundo, desde el punto seleccionado todo punto que se encuentre a una distancia igual o menor que $\varepsilon$ se registra y si se encuentra el mínimo de puntos requeridos significa que todos los puntos encontrados pertenecen al mismo `cluster`. Por último los `clusters` se expanden repitiendo recursivamente el cálculo de vecindad para cada punto vecino [24].

Algo a tener en cuenta en este modelo es la distancia de medida, si es muy grande podría darse el caso de que todos los puntos presentes en una representación pertenezcan al mismo `cluster`, por el contrario si la distancia es demasiado pequeña la gran mayoría de puntos serían considerados como ruido y por tanto no clasificados en un `cluster`.

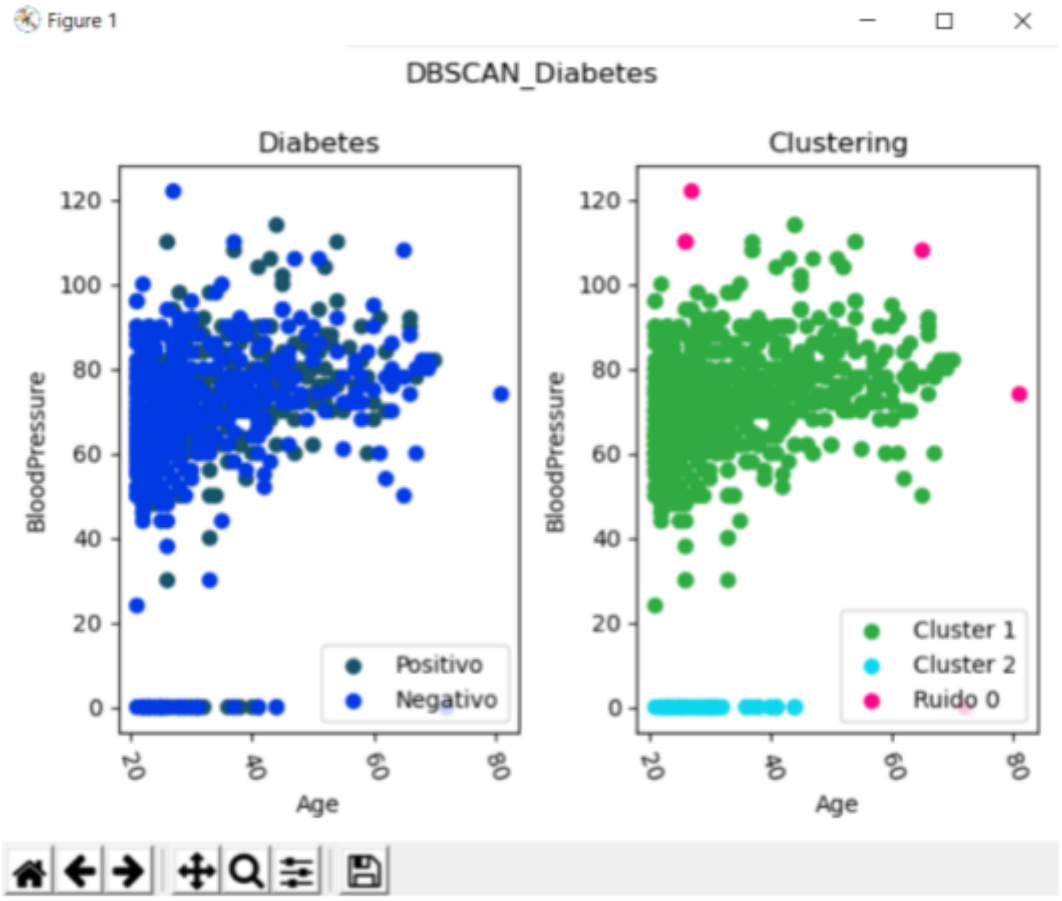



Figura 4.39. Representación DBSCAN

En la figura 4.39 se muestra una representación del modelo DBSCAN, de los datos de pacientes de diabetes, siendo el eje X la edad, y en el eje Y la presión sanguínea. En los resultados se observa como el modelo ha separado los datos en dos `clusters` altamente diferenciados, con unos pocos valores de ruido.

Del mismo modo que en la regresión y la clasificación se implementó un método para poder hacer la comparativa entre los métodos de clustering desarrollados en este caso

se comparan los resultados de aplicar `clustering` a un conjunto de datos representando el resultado de cada uno de los modelos con una gráfica de dispersión.

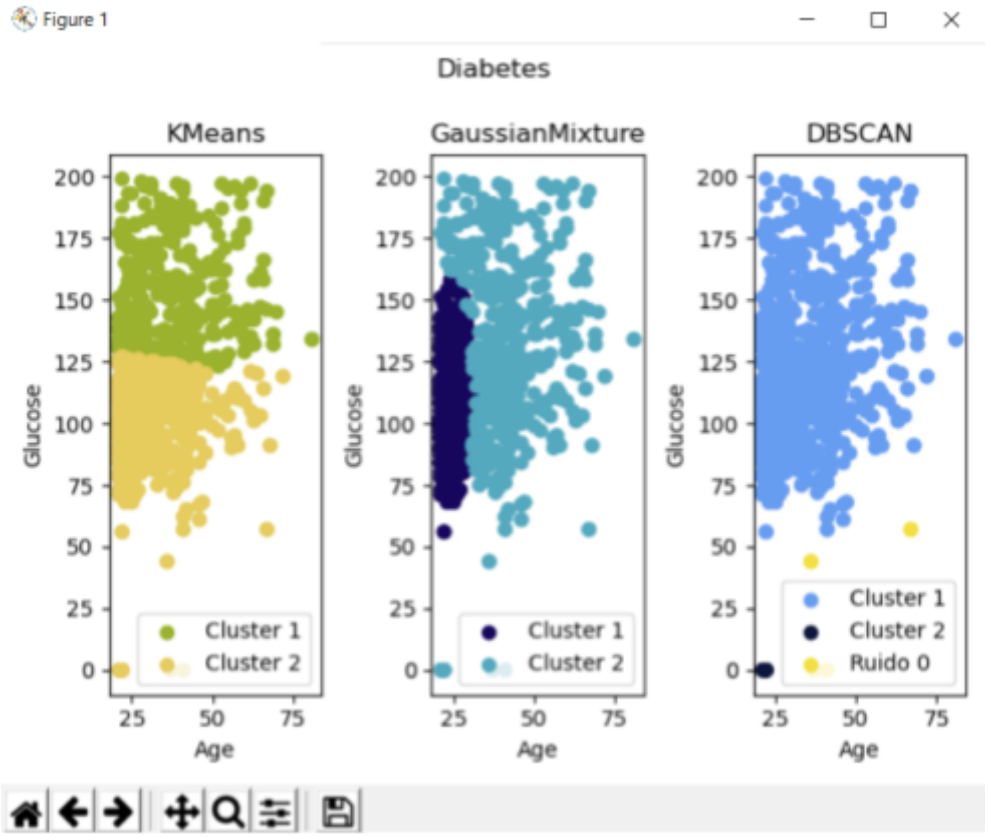



Figura 4.40. Representación todos los métodos de clustering

La figura 4.40 muestra la representación gráfica de los clusters generados a partir del mismo conjunto de datos. Se realizó sobre los datos de diabetes estando la edad en el eje X y la glucosa en el eje Y. En el caso de K Means separó los dos clusters por el rango de glucosa siendo en los casos superiores a 125 pertenecientes al primer cluster y los inferiores al segundo cluster, para Gaussian el primer cluster corresponde a los usuarios de edad inferior a 25 años y glucosa inferior a 150, el resto corresponde al segundo cluster. Por último para DBscan al tener tan poca separación entre los datos reconoce prácticamente a todo el conjunto de datos como un solo cluster.

# Capítulo 5 Planificación y presupuesto

La planificación original del desarrollo de este proyecto contaba con un desempeño de las actividades de manera paralela y a tiempo completo, pero debido a las circunstancias adversas que se presentaron con el covid y la incorporación al mundo laboral ha sido necesaria una replanificación de las actividades y de la dedicación del tiempo diario a la ejecución se disminuyó bastante. Se reagruparon y estructuraron algunas de las actividades establecidas durante el anteproyecto para poder ajustarnos a la nueva situación.

Las actividades finalmente planificadas para el desarrollo del proyecto fueron las siguientes:

- Actividad 1: Análisis y desarrollo bajo el ecosistema de herramientas de Python de un entorno de visualización en el contexto de grandes volúmenes de datos. 2 meses
    - Actividad 1.1: Creación de la interfaz del usuario
    - Actividad 1.2: Estructura para la limpieza y estructuración de los datos
    - Actividad 1.3: Métodos básicos de representación de los datos
- Actividad 2: Aplicación de técnicas de aprendizaje automático en el contexto especificado. 2 meses.
    - Actividad 2.1: Métodos de regresión
    - Actividad 2.2: Métodos de clasificación
    - Actividad 2.3: Métodos de clustering

Además de las actividades necesarias para el desarrollo del proyecto en la planificación también se incluye la redacción de esta memoria como Actividad 3

En lo referido al presupuesto se incluye el software empleado y licencias necesarias así como las horas de trabajo del desarrollador. El software utilizado en este proyecto ha sido Visual Studio, Python con sus múltiples librerías y github principalmente, aunque este software es libre y gratuito github cuenta con una versión pro, aunque no se utilizó en el desarrollo de este proyecto. También hacer mención al sistema operativo windows sobre el cual se trabajó pero no es un elemento obligatorio para el desarrollo aun así se indica su coste, junto con el de github pro en la tabla 5.1.

En lo que se refiere a las horas de trabajo del desarrollador se computa un total de horas

de dedicación al proyecto de 260, siendo el precio estimado por hora 1 de 5€. En la tabla 5.2 se encuentra una tabla con el desglose de horas junto con el coste, y el coste total del proyecto.

| Concepto | Horas | Coste | Total |
|---|---|---|---|
| Windows 10 | --- | 145 € | 145 € |
| Github Pro | --- | 4 €/mes | 60 € |
| Actividad 1.1 | 50 h | 5 €/h | 250 € |
| Actividad 1.2 | 20 h | 5 €/h | 100 € |
| Actividad 1.3 | 45 h | 5 €/h | 225 € |
| Actividad 2.1 | 45 h | 5 €/h | 225 € |
| Actividad 2.2 | 45 h | 5 €/h | 225 € |
| Actividad 2.3 | 45 h | 5 €/h | 225 € |
| Actividad 3 | 10 h | 5 €/h | 50 € |
| Total | 260 h | | 1505 € |

Tabla 5.2. Tabla costes horas

.

# Capítulo 6 Conclusiones y líneas futuras

En este proyecto se ha desarrollado una herramienta que facilita la iniciación en el análisis de datos y en el aprendizaje automático. La herramienta incorpora diversos métodos para la visualización de datos e información y también un conjunto de métodos para la aplicación de métodos de aprendizaje automático. Se trata de una herramienta que de manera automática permitiría, para conjuntos genéricos de datos, realizar un análisis preliminar de los datos así como extraer de una manera automática y sencilla modelos de aprendizaje automático.

El desarrollo de la herramienta se ha realizado siguiendo una metodología que proporciona gran flexibilidad para la incorporación de nuevos métodos tanto para la visualización como para el aprendizaje automático. Los métodos de visualización comprenden desde representaciones simples como las que aparecen en series temporales hasta métodos de aprendizaje automático supervisado y no supervisado (Regresión, Clasificación y Clustering). La metodología empleada permitirá extender sus funcionalidades mediante la incorporación de nuevos métodos.

El proyecto ha sido desarrollado utilizando el lenguaje Python y un amplio conjunto de librerías que proporciona su ecosistema, librerías para la representación de datos así como librerías para desarrollar aprendizaje automático. Asimismo, ha sido validado sobre un conjunto amplio y diverso de datos en el que han sido probados los distintos métodos de representación implementados así como los métodos de aprendizaje automático.

Como líneas de futuro, mencionar que actualmente este proyecto trabaja de manera secuencial con el conjunto de datos, por lo que dependiendo del tamaño conjunto de datos a analizar se podría generar una restricción a futuro. Por ello sería deseable explorar opciones que admitieran el procesamiento paralelo o distribuido de los datos. Apache Spark (Spark) es un motor de procesamiento de datos de código abierto para grandes conjuntos de datos y podría ser una de estas opciones de cara a la continuación del desarrollo del proyecto. Del mismo modo, este proyecto está desarrollado para trabajar desde el escritorio de un ordenador, pero sería interesante tener la opción de acceder y trabajar desde otras plataformas que permitieran el acceso remoto a la información y a la ejecución de los modelos.

# Capítulo 7 Conclusions and Future Work

This project presents a tool that eases the initiation in the data analysis and in the building of machine learning models. The tool integrates a diverse set of data analysis and visualization methods, and also a set of machine learning methods. The tool aims to develop a pre-analysis of data provided as generic data sets and also allows the generation of machine learning models in an automatic manner.

The tool has been developed following a methodology that provides a great flexibility to incorporate new methods as much in the visualization as for machine learning. The visualization methods comprehend from basic representation like the ones that appear in temporal series until supervised and unsupervised machine learning methods (Regression, Classification and Clustering). The methodology applied will allow to extend the functionalities by incorporating new methods.

The project has been developed using the Python language and a wide set of libraries provided by its ecosystem, libraries for data representation as well as libraries to develop machine learning. In addition, it has been validated on a large and diverse dataset on which the different representation methods implemented and the machine learning methods have been tested.

As future lines, it must be mentioned that currently this project performs a sequential processing of the dataset, so depending on the size of the dataset to be analyzed it could generate restrictions in the future. Therefore, it would desirable to explore options to support parallel or distributed data processing. Apache Spark (Spark) is an open source data processing engine for large datasets and could be one of these options for a future development of the project. Similarly, this project has been developed to run on a computer desktop, but it would be interesting to have options for the remote access to the information and for the execution of the models.

# Bibliografía


[1]. Infogram: https://infogram.com/

[2]. Tableau: https://www.tableau.com/es-es

[3]. D3.js: https://d3js.org/

[4]. Descripción patrones de diseño: https://sourcemaking.com/design_patterns

[5]. Ventajas patrones diseño: https://refactoring.guru/es/design-patterns/why-learn-patterns

[6]. Patrones creación: https://sourcemaking.com/design_patterns/creational_patterns

[7]. Patrones de estructura: https://sourcemaking.com/design_patterns/structural_patterns

[8].Patrones de comportamiento: https://es.wikipedia.org/wiki/Patr%C3%B3n_de_dise%C3%B1o

[9]. Gráfica tipo línea: https://docs.microsoft.com/es-es/sql/reporting-services/report-design/line-charts-report-builder-and-ssrs?view=sql-server-ver15

[10]. El Naqa, I., & Murphy, M. J. (2015). What is machine learning?. In machine learning in radiation oncology (pp. 3-11). Springer, Cham.

[11]. IBM Supervised learning: https://www.ibm.com/cloud/learn/supervised-learning

[12]. IBM unsupervised learning: https://www.ibm.com/cloud/learn/unsupervised-learning

[13]. IBM supervised vs unsupervised learning: https://www.ibm.com/cloud/blog/supervised-vs-unsupervised-learning

[14]. Linear regression: https://towardsdatascience.com/linear-regression-detailed-view-ea73175f6e86

[15]. Touzani, S., Granderson, J., & Fernandes, S. (2018). Gradient boosting machine for modeling the energy consumption of commercial buildings. Energy and Buildings, 158, 1533-1543.

[16]. Isotonic regression: https://spark.apache.org/docs/1.5.1/mllib-isotonic-regression.html

[17]. K vecinos: https://towardsdatascience.com/machine-learning-basics-with-the-k-nearest-neighbors-algorithm-6a6e71d01761

[18]. Naive bayes classification: https://machinelearningmastery.com/naive-bayes-for-machine-learning/

[19]. Formula Naive Bayes Classification: https://iq.opengenus.org/gaussian-naive-bayes/

[20]. Classification tree: https://medium.com/swlh/decision-tree-classification-de64fc4d5aac

[21]. Madhulatha, T. S. (2012). An overview on clustering methods. *arXiv preprint arXiv:1205.1117*.

[22]. K Means: https://towardsdatascience.com/understanding-k-means-clustering-in-machine-learning-6a6e67336aa1

[23]. Gaussian mixture model: https://brilliant.org/wiki/gaussian-mixture-model/

[24]. DBSCAN:

https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html

[25]. Ejemplo Gaussian Naive Bayes:
https://www.datasciencecentral.com/profiles/blogs/naive-bayes-for-dummies-a-simple-explanation