MILESTONE 2

Whiteeth Group

[**Introduction 1**](#_5eubkey5conm)

[**Data preparation 2**](#_3ro7al1vqly7)

[**Materialising Minable View 3**](#_afyxuwq25xxf)

[**Model Building 4**](#_b0sg4oponbv5)

[**Evaluation 4**](#_j5kz8w3fnn1a)

[**Deployment Mockup 4**](#_bcjv3b9tnm6w)

[**Use of Technology 4**](#_y1z0p464uf4a)

[**References 4**](#_mhgf917krpaf)

# Introduction

YASS: ESPAÑOL (ya traduciremos con Chat GPT)

La temática de este proyecto gira en torno a dos aspectos principales de la odontología. Por un lado, la cirugía dentoalveolar, que es una rama de la cirugía oral y maxilofacial que se centra en procedimientos que involucran los dientes y el hueso alveolar circundante dentro de la boca, como extracciones de dientes y colocación de implantes dentales.

Por otro lado, la periimplantitis es una afección inflamatoria que afecta los tejidos que rodean un implante dental. Es similar a la enfermedad periodontal que afecta a los dientes naturales. Puede resultar en la pérdida del implante si no se trata adecuadamente.

Este proyecto tiene dos objetivos principales:

* El primero es identificar aquel perfil de paciente con mayor riesgo de padecer complicaciones tras la intervención, con el fin de prevenir estos daños y buscar alternativas.
* El segundo es predecir la duración de la intervención de cada paciente con el objetivo de reducir la cantidad de retrasos que se dan en las consultas clínicas.

**YASS: ENGLISH**

The focus of this project lies in two primary aspects of dentistry. Firstly, dentoalveolar surgery, a branch of oral and maxillofacial surgery concentrating on procedures involving teeth and the surrounding alveolar bone within the mouth, such as tooth extractions and dental implant placements. Secondly, peri-implantitis, an inflammatory condition impacting the tissues around a dental implant, akin to periodontal disease affecting natural teeth. Left untreated, it may lead to implant loss.

This project aims to achieve two main objectives:

* Firstly, to identify patient profiles at a higher risk of complications post-intervention, thereby preemptively preventing such issues and exploring alternative solutions.
* Secondly, to forecast the duration of each patient's intervention, aimed at minimizing delays in clinical consultations.

# Data preparation

YASS: ESPAÑOL (ya traduciremos con Chat GPT)

Los datos, brindados por el responsable de una clínica dental de Barcelona, consisten en tres conjuntos: uno de 2016, que tiene 1159 observaciones; otro de 2017, con 719 observaciones; y el último de 2018, con 228 observaciones. Todos ellos fueron el resultado de una encuesta realizada entre los dentistas de la clínica y cada conjunto de datos contiene 936 variables.

Dado que los conjuntos de datos tienen las mismas variables y no hay continuidad entre ellos, es decir, los pacientes de un año no se repiten en el mismo año o en los otros, se ha decidido considerar la totalidad de ellos como datos transversales y concatenarlos todos por fila. Esto resultó en un conjunto de datos con más de 2000 observaciones, más de 930 variables y sin distinción de tiempo.

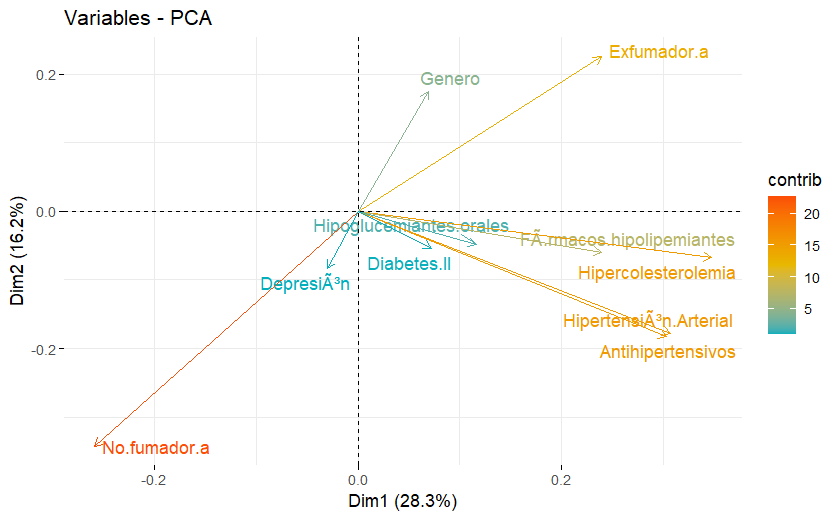
Respecto a las variables, la gran mayoría son binarias, y debido a que la cantidad era demasiado alta y se identificaba una distinción temática entre algunas de ellas, se han dividido en 4 bloques teniendo en cuenta lo que se desea hacer con ellas y lo que necesitan los objetivos. Estos bloques son:

* Las patologías sistémicas de los pacientes.
* La información sobre la intervención en sí misma.
* La medicación pre-intervención.
* La medicación post-intervención.

Además, a parte de la eliminación de errores o de columnas completamente vacías (como nombres, apellidos o correos electrónicos) y la binarización de variables categóricas, **se han realizado otras transformaciones enfocadas a cada uno de los objetivos**.

Para el **primer objetivo**, se han seleccionado las primeras 111 columnas binarias, en las cuales encontramos las características de los pacientes, qué patología sistémica tienen y qué medicamento están tomando. Además, hemos eliminado columnas irrelevantes, entre las cuales se encuentran medicación actual y patología sistémica, ya que esa información ya la tenemos en los distintos medicamentos y patologías individuales. Finalmente, después de eliminar el primer tipo de operación que no nos interesa (extracción del molar) obtenemos 32 datos de cirugía periimplantaria y 98 de cirugía dentoalveolar.

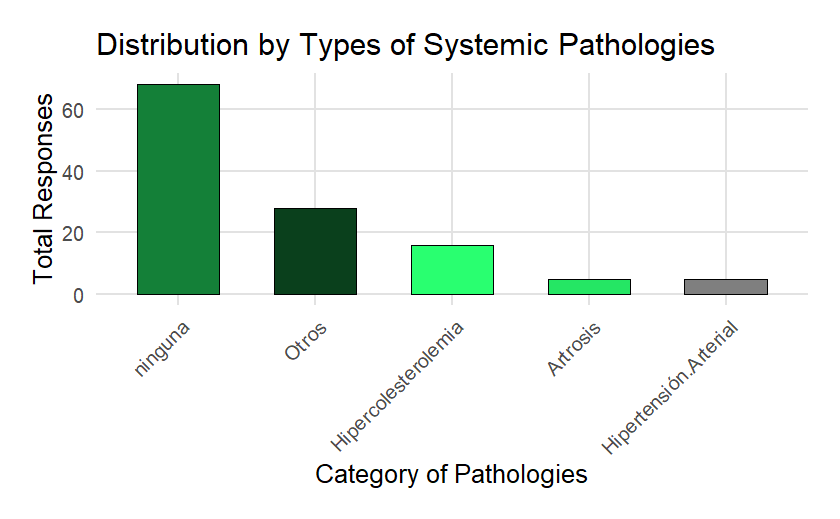
Para observar los datos de los pacientes y ver cómo se relacionan las variables, hemos realizado el PCA explicativo que vemos a continuación.

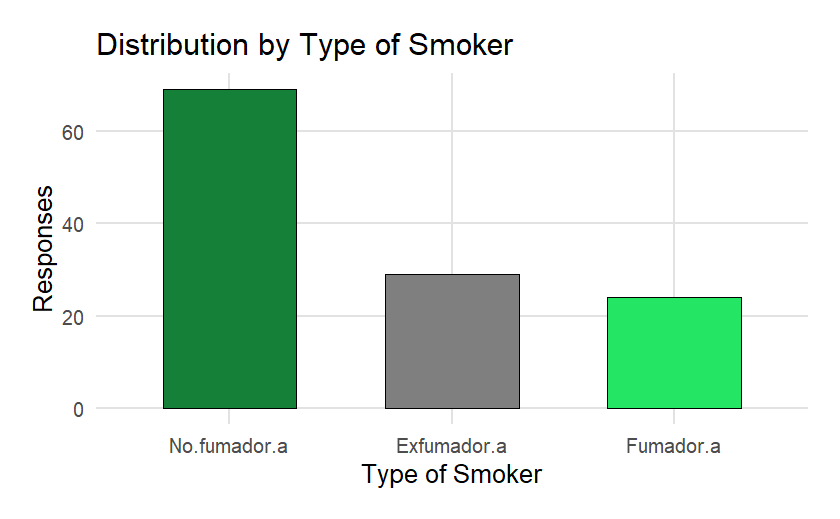
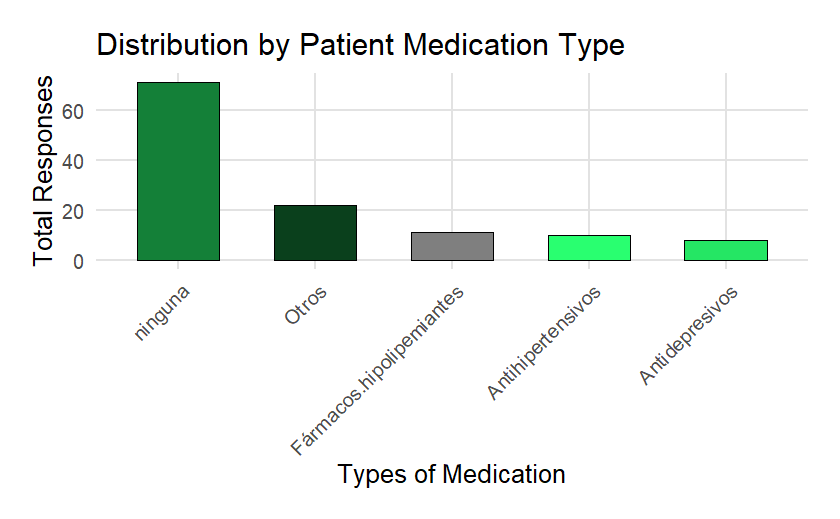
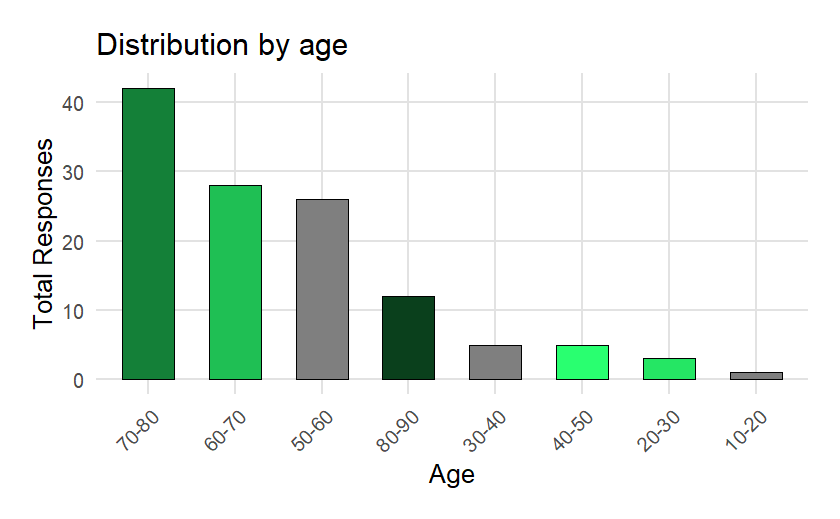


Este gráfico destaca las variables que ejercen mayor influencia en la varianza de los datos. Podemos confirmar la coherencia de nuestros datos, ya que se evidencia una fuerte correlación entre las enfermedades y sus medicamentos correspondientes. Asimismo, se observa una relación inversa entre los no fumadores y los exfumadores.

Además, para realizar un análisis univariante con gráficos más sencillos y preparar los datos para futuros análisis, hemos convertido las 111 columnas binarias a 8 columnas categóricas, donde hemos agrupado todos los medicamentos y patologías, los distintos tipos de consumidores de alcohol y drogas, grupos de edad, género y tipo de intervención. Finalmente, las medicaciones o patologías que aparecían 2 veces o menos las hemos renombrado como ‘otros’ (véase la limpieza más en profundidad en el anexo 1).

A continuación, mostramos el resumen de las variables más relevantes para explicar nuestros pacientes:





Para resolver la cuestión de las diferencias entre los pacientes, en el análisis se separarán los tipos de operaciones y se observará si existen diferencias significativas entre las distintas variables.

Para el **segundo objetivo**, predecir la duración de la operación, se ha realizado un Análisis Factorial de Correspondencias (AFC) múltiple con el fin de crear un subconjunto de datos con las variables más esenciales.

Una vez identificadas las 100 variables de mayor peso, se ha utilizado la librería *autoclean* con el fin de pulir los datos debido a la complejidad de las variables seleccionadas con el AFC, algunas contenían un gran número de categorías diferentes.

La librería "autoclean" es una herramienta que automatiza el proceso de limpieza de datos en Python. Facilita la eliminación de datos faltantes, duplicados y variables poco útiles, así como la conversión de tipos de datos, lo que agiliza y simplifica el preprocesado de datos en proyectos de análisis de datos y aprendizaje automático.

Y, por último, se transforma nuestra variable objetivo, duración de la intervención, de categórica por rangos de tiempo a numérica obteniendo la media de esos rangos. A continuación se muestra un breve análisis estadístico:

(INSERTE IMAGEN DE LO DE: Transformación numérica de la variable ‘Duracion\_intervencio’......)

APARTADO EN INGLÉS

The data, provided by the head of a dental clinic in Barcelona, consist of three sets: one from 2016, with 1159 observations; another from 2017, with 719 observations; and the last from 2018, with 228 observations. All of these were the result of a survey conducted among the dentists at the clinic and each data set contains 936 variables.

Since the data sets have the same variables and there is no continuity between them, meaning the patients from one year do not repeat in the same year or in the others, it has been decided to consider all of them as cross-sectional data and to concatenate them all by row. This resulted in a data set with more than 2000 observations, more than 930 variables, and no distinction of time.

Regarding the variables, the vast majority are binary, and because the number was too high and a thematic distinction was identified among some of them, they have been divided into 4 blocks considering what is intended to be done with them and what the objectives require. These blocks are:

1. The systemic pathologies of the patients.

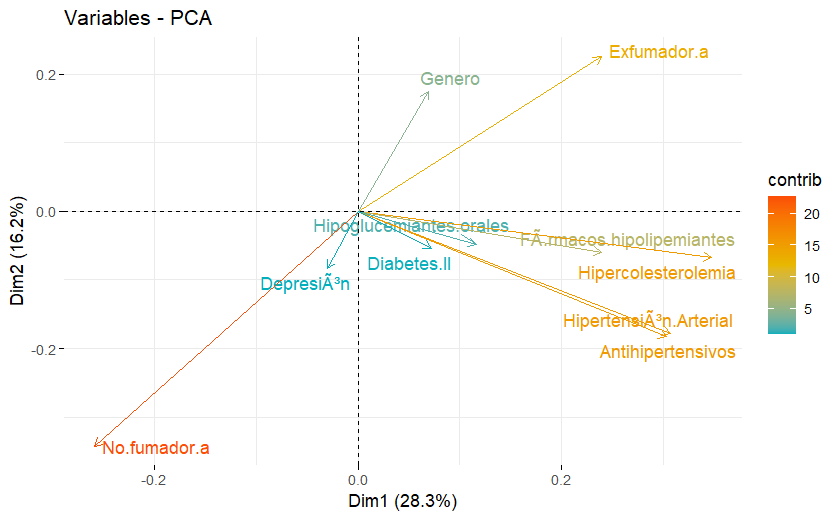
2. The information about the intervention itself.

3. The pre-intervention medication.

4. The post-intervention medication.

In addition to removing errors or completely empty columns (such as names, surnames, or emails) and binarizing categorical variables, other transformations focused on each of the objectives have been carried out.

For the first objective, the first 111 binary columns have been selected, in which we find the characteristics of the patients, what systemic pathology they have, and what medication they are taking. In addition, we have removed irrelevant columns, among which are current medication and systemic pathology, since we already have this information in the various individual medications and pathologies. Finally, after eliminating the first type of operation that does not interest us (molar extraction), we obtain 32 peri-implant surgery data and 98 dentoalveolar surgery data.



# 

# 

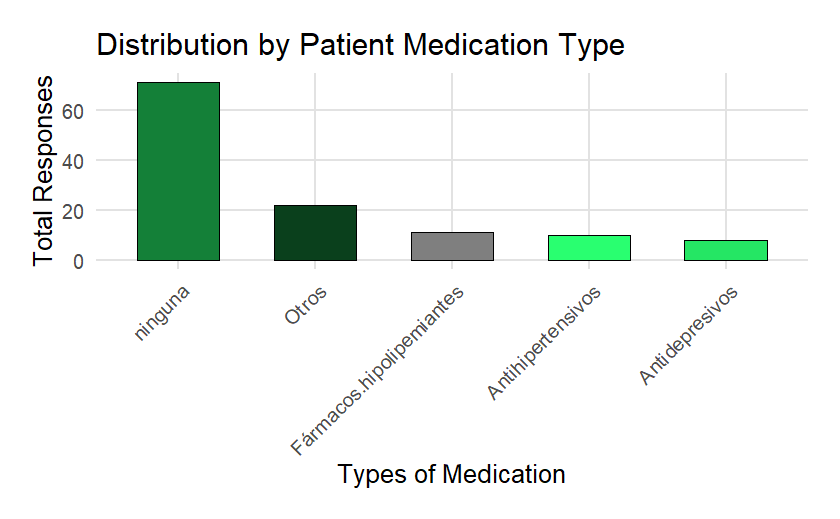
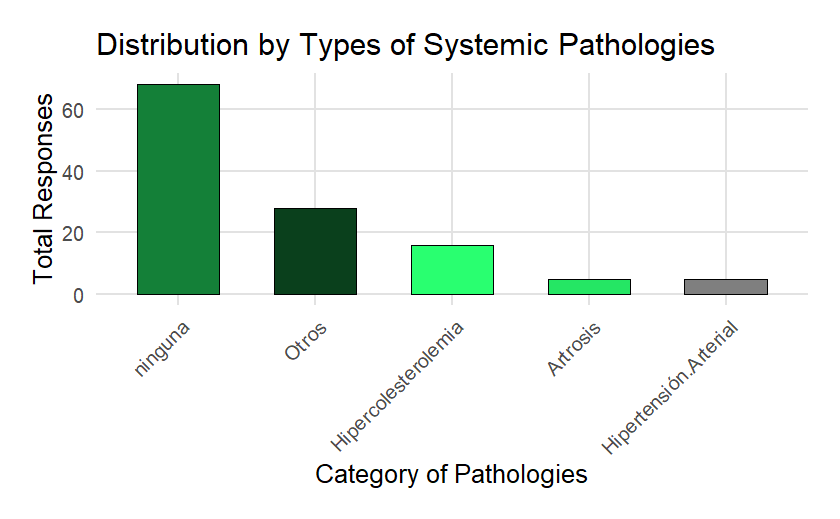
# 

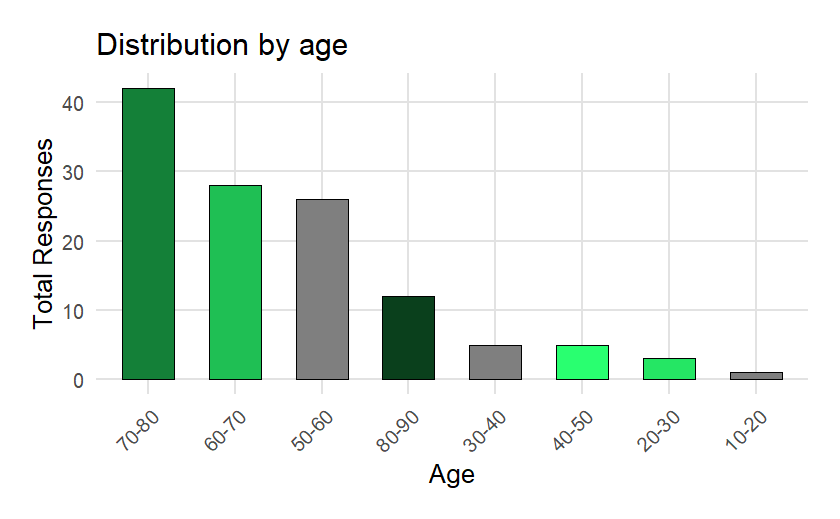
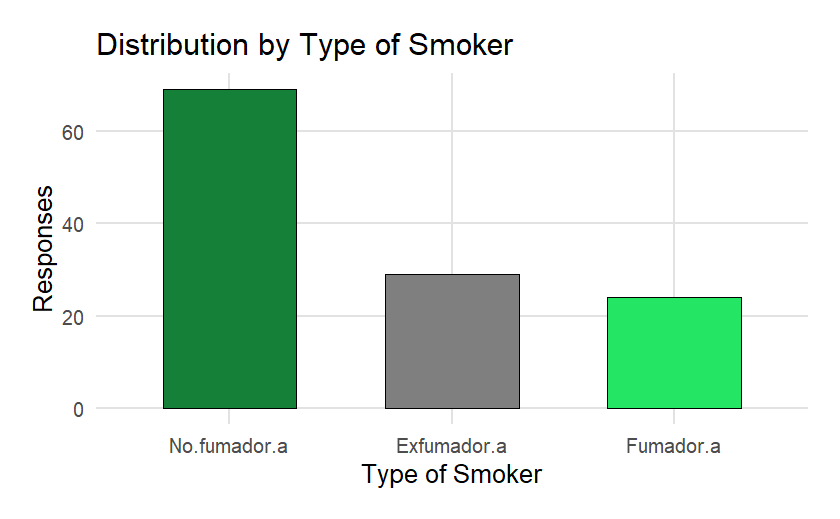
# 

This chart highlights the variables that exert the greatest influence on the variance of the data. We can confirm the coherence of our data, as there is a strong correlation observed between diseases and their corresponding medications. Additionally, an inverse relationship is observed between non-smokers and ex-smokers.

Moreover, to perform univariate analysis with simpler charts and to prepare the data for future analyses, we have converted the 111 binary columns into 8 categorical columns, where we grouped all the medications and pathologies, different types of alcohol and drug users, age groups, gender, and type of intervention. Finally, medications or pathologies that appeared 2 times or less were renamed as 'other' (see more detailed cleaning in Annex 1).

Below, we present a summary of the most relevant variables to explain our patients:





To address the issue of differences among patients, the analysis will separate the types of operations and observe if there are significant differences between the various variables.

For the second objective, predicting the duration of the operation, a Multiple Correspondence Analysis (MCA) has been conducted to create a subset of data with the most essential variables.

After identifying the 100 most significant variables, the "autoclean" library was used to refine the data due to the complexity of the variables selected with the MCA, some of which contained a large number of different categories.

The "autoclean" library is a tool that automates the data cleaning process in Python. It facilitates the removal of missing data, duplicates, and less useful variables, as well as the conversion of data types, which streamlines and simplifies the preprocessing of data in data analysis and machine learning projects.

Lastly, our target variable, the duration of the intervention, is transformed from a categorical by time ranges to a numeric one by obtaining the average of these ranges. Below is a brief statistical analysis:

(INSERTE IMAGEN DE LO DE: Transformación numérica de la variable ‘Duracion\_intervencio’......)

# SUMMARY

# Materialising Minable View

YASS: ESPAÑOL (ya traduciremos con Chat GPT)

Con el fin de completar la tarea de entrenar un modelo que prediga las duraciones de la intervención, definimos 2 partes principales:

* Las **variables de entrada:** serán todas aquellas variables de mayor influencia seleccionadas con un AFC y posteriormente con un análisis de importancia de características con Random Forest.

Estas variables son las mostradas en la Figura 1, las cuales son de especial relevancia porque están relacionadas con la naturaleza de la duración de las operaciones.

Con este paso tratamos de cuantificar cuánto ayuda cada característica para mejorar la precisión del modelo antes de su ajuste.

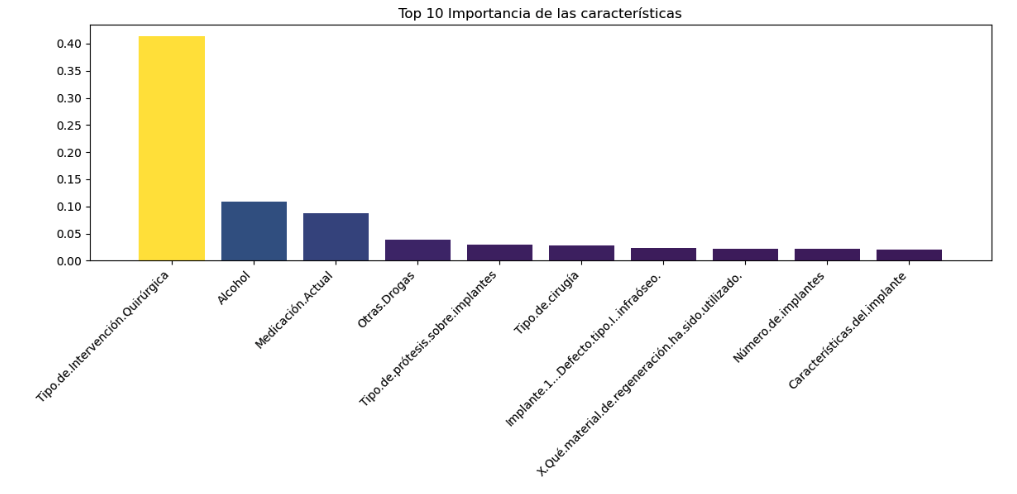
* La **variable de salida**: es el tiempo de duración de la intervención predicho. Dado que lo que se pretende predecir es una variable numérica, nuestro problema es de regresión.

**YASS: ENGLISH**

To accomplish the task of training a model that predicts intervention durations, we've outlined two main components:

* **Input variables**: These encompass all influential variables selected through an AFC and further refined via a feature importance analysis using Random Forest.

These variables, as depicted in Figure 6, hold particular relevance due to their association with the nature of operation durations. This step allows us to gauge each feature's contribution to enhancing model accuracy prior to adjustment.



* **Output variable**: This pertains to the predicted duration of the intervention. As our aim is to forecast a numerical variable, our challenge lies within the realm of regression.

# Model Building

El primer paso es el de dividir los datos en train y test para poder comenzar con el entrenamiento de los distintos modelos, así se separan con un split en 80% para train y 20% para test.

En segundo lugar, entrenamos un total de 10 modelos, los modelos lineales devolvieron valores muy altos, debido a la poca robustez para tratar con nuestro problema.

LinearRegression: 206665791346910.1875 (+/- 157415457083742.2188)

SGDRegressor: 2323862.8506 (+/- 2941995.8922)

SVR: 22.8684 (+/- 1.0679)

KNeighborsRegressor: 23.4226 (+/- 0.2583)

DecisionTreeRegressor: 21.3033 (+/- 0.7451)

**RandomForestRegressor: 21.1495 (+/- 0.5715)**

GradientBoostingRegressor: 21.1190 (+/- 0.6748)

AdaBoostRegressor: 21.2995 (+/- 0.3802)

MLPRegressor: 21.0263 (+/- 0.8421)

XGBRegressor: 21.2149 (+/- 0.8090)

El modelo que finalmente se ha escogido por su buen desempeño en este tipo de problemas ha sido el **RandomForestRegressor,** el cual nos devuelve un RMSE bastante aceptable e inferior al valor medio de los datos y de su desviación estándar, los presentamos a continuación:

**Media**: 30.01960085531005

Moda: 30.0

**Desviación Estándar**: 24.77176690577136

Varianza: 613.6404356338692

Mediana: 30.0

Mínimo: 5.0

Máximo: 180.0

IQR (Rango Inter-Cuartilico): 15.0

El siguiente paso es el de buscar los mejores hiperparámetros y poder reducir el RMSE, buscando un ajuste con menor error.

Donde se realiza una optimización de hiperparámetros para el modelo RandomForestRegressor usando GridSearchCV dentro de un Pipeline que incluye StandardScaler para la normalización de los datos.

Aquí se prueban combinaciones de hiperparámetros como el número de árboles, la profundidad máxima del árbol y el número mínimo de muestras por hoja, entre otros, para encontrar la configuración que minimiza el error cuadrático medio (MSE).

El proceso asegura la selección del modelo más eficaz basado en el RMSE obtenido a través de validación cruzada.

Los hiperparámetros escogidos son:

Mejor Modelo: Pipeline(steps=[('scaler', StandardScaler()),

('modelo',

RandomForestRegressor(max\_features='sqrt',

min\_samples\_split=10, n\_estimators=50,

random\_state=42))])

Mejor RMSE: **20.819899221181867**

Otras métricas calculadas:

Correlación de Pearson: **0.5643823802304564**

MAE: **14.81**

APARTADO EN INGLÉS

The first step is to split the data into train and test sets in order to start training the various models, using an 80% split for training and 20% for testing.

Secondly, we trained a total of 10 models. The linear models returned very high values, due to their lack of robustness in dealing with our problem.

LinearRegression: 206665791346910.1875 (+/- 157415457083742.2188)

SGDRegressor: 2323862.8506 (+/- 2941995.8922)

SVR: 22.8684 (+/- 1.0679)

KNeighborsRegressor: 23.4226 (+/- 0.2583)

DecisionTreeRegressor: 21.3033 (+/- 0.7451)

**RandomForestRegressor: 21.1495 (+/- 0.5715)**

GradientBoostingRegressor: 21.1190 (+/- 0.6748)

AdaBoostRegressor: 21.2995 (+/- 0.3802)

MLPRegressor: 21.0263 (+/- 0.8421)

XGBRegressor: 21.2149 (+/- 0.8090)

The model that has ultimately been chosen for its good performance in this type of problem is the RandomForestRegressor, which gives us a quite acceptable RMSE, lower than the mean value of the data and its standard deviation. We present them below:

**Media**: 30.01960085531005

Moda: 30.0

**Desviación Estándar**: 24.77176690577136

Varianza: 613.6404356338692

Mediana: 30.0

Mínimo: 5.0

Máximo: 180.0

IQR (Rango Inter-Cuartilico): 15.0

The next step is to search for the best hyperparameters and to reduce the RMSE, aiming for a fit with lower error.

This involves hyperparameter optimization for the RandomForestRegressor model using GridSearchCV within a Pipeline that includes StandardScaler for data normalization.

Here, combinations of hyperparameters such as the number of trees, maximum tree depth, and minimum number of samples per leaf are tested, among others, to find the configuration that minimizes the mean squared error (MSE).

The process ensures the selection of the most effective model based on the RMSE obtained through cross-validation.

The chosen hyperparameters are:

Mejor Modelo: Pipeline(steps=[('scaler', StandardScaler()),

('modelo',

RandomForestRegressor(max\_features='sqrt',

min\_samples\_split=10, n\_estimators=50,

random\_state=42))])

Mejor RMSE: **20.819899221181867**

Other calculated metrics:

Correlación de Pearson: **0.5643823802304564**

MAE: **14.81**

# 

# Evaluation

(HAY QUE DECIR QUÉ MÉTRICAS VAMOS A USAR) → Y tener en cuenta que igual nos conviene más sobreestimar que subestimar.

Las métricas seleccionadas para evaluar el modelo desde distintas perspectivas han sido RMSE, MAE y la correlación de Pearson. A continuación aclaramos por qué su uso:

* **RMSE**: destaca errores grandes, siendo crucial para contextos donde evitar grandes desviaciones es vital.
* **MAE**: proporciona una medida clara del error medio, útil cuando todos los errores tienen la misma importancia.
* **Correlación** **de** **Pearson**: evalúa la relación lineal entre predicciones y valores reales, indicando qué tan bien el modelo sigue las tendencias observadas.

Estas métricas permiten validar la confiabilidad del modelo en aplicaciones prácticas, ayudando a entender tanto la magnitud de los errores como la alineación de las predicciones con las tendencias reales.

Las métricas revelan que el modelo es moderadamente efectivo, captando tendencias generales pero con espacio significativo para mejorar la precisión y reducir los errores en las predicciones

El protocolo de evaluación usado para el ajuste de los valores ha sido tanto Cross-Validation como bootstrapping, siendo el mejor valor devuelto el del Cross-Validation de 5 iteraciones y RandomForest.

Los siguientes pasos serán bajar los 20.81 minutos de error de RMSE lo máximo posible, hasta donde nos dejen llegar los datos, e intentar mejorar la correlación de Pearson para mejorar la tendencia que sigue el modelo. Además, vamos a dar más peso a los errores que subestimen el tiempo de consulta.

APARTADO EN INGLÉS

The metrics selected to evaluate the model from different perspectives have been RMSE, MAE, and Pearson correlation. We clarify their use below:

* **RMSE**: highlights large errors, being crucial in contexts where avoiding large deviations is vital.
* **MAE**: provides a clear measure of average error, useful when all errors are equally important.
* **Pearson correlation:** assesses the linear relationship between predictions and actual values, indicating how well the model follows the observed trends.

These metrics allow for validating the model's reliability in practical applications, helping to understand both the magnitude of the errors and the alignment of the predictions with real trends.

The metrics reveal that the model is moderately effective, capturing general trends but with significant room for improvement in precision and reducing errors in predictions.

The evaluation protocol used for adjusting the values has been both Cross-Validation and bootstrapping, with the best value returned being from 5 iterations of Cross-Validation and RandomForest.

The next steps will be to reduce the 20.81 minutes of RMSE error as much as possible, to the extent allowed by the data, and to try to improve the Pearson correlation to enhance the trend that the model follows. Additionally, we will give more weight to errors that underestimate the consultation time.

# Deployment Mockup

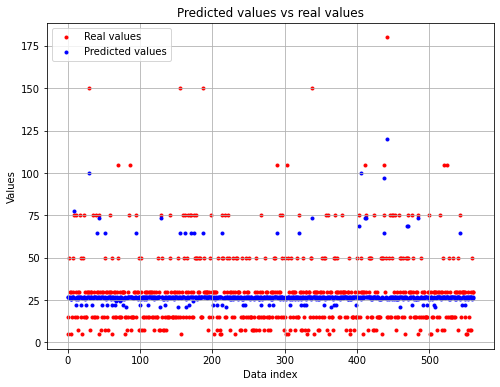
(testear el modelo y responder a la pregunta si puede funcionar → aquí creo que es lo de discusión/conclusión)

Para responder eficazmente a la pregunta “**¿puede funcionar?**”, hemos desarrollado un prototipo de aplicación que no solo implementa nuestro modelo sino que también muestra visualmente sus resultados. Este prototipo está diseñado con **dos componentes principales** para abordar de manera directa los objetivos originales propuestos.

El **primer componente** es un dashboard interactivo que permite a los usuarios, en este caso dentistas, visualizar y analizar el flujo de datos. Este panel de control es intuitivo y está diseñado para proporcionar una comprensión clara y precisa de cómo se mueven y se procesan los datos en tiempo real, lo cual es esencial para la toma de decisiones rápida y efectiva en un entorno clínico.

El **segundo componente** es un sistema de cuestionario compuesto por diez preguntas específicas que los dentistas deben responder antes de programar las intervenciones quirúrgicas. Estas preguntas están cuidadosamente diseñadas para capturar información crítica que alimenta al modelo predictivo. Basado en las respuestas proporcionadas, el modelo calcula una estimación del tiempo que llevará realizar cada procedimiento. Esta funcionalidad es clave para mejorar la precisión en la programación de las citas, optimizando así el tiempo y los recursos de la clínica.

El valor generado por este prototipo se puede evaluar en términos de la mejora de la eficiencia operativa y la satisfacción del paciente. Al proporcionar estimaciones de tiempo más precisas, el prototipo ayuda a reducir los tiempos de espera, permite atender a más pacientes y mejora la gestión general de las citas.



Por lo tanto, este prototipo no solo sirve como una herramienta para visualizar cómo el modelo puede ser aplicado en un escenario real, sino que también ofrece una oportunidad para evaluar preliminarmente el impacto potencial y la viabilidad del modelo. La respuesta a "¿puede funcionar?" se obtiene a través de la implementación práctica de este prototipo, que nos permite identificar fortalezas y áreas de mejora antes de su lanzamiento final.

# Use of Technology

(ESTO LO HACE EVGENY) Lo hago yo

A lo largo del desarrollo de este proyecto, diversos problemas de índole tecnológica fueron enfrentados debido a la complejidad que presenta la base de datos.

En primer lugar, durante la planificación del proyecto, se ha hecho uso de la aplicación The Gantt Project, para poder realizar el diagrama de Gantt con cierta facilidad.

Seguidamente, para llevar a cabo tareas tanto de predicción como de descripción de los datos, nos encontramos con la barrera de que pocos métodos diseñados para tales tareas presentan un buen desempeño en nuestro caso.

Para la limpieza y preparación de los datos se ha hecho uso principalmente el lenguaje R, resultando este el más fácil para dicha tarea. Los análisis siguientes se han desarrollado tanto en R como en Python, para poder abordar el problema con cierta versatilidad.

En cuanto a los distintos análisis se han empleado diversas librerías, con mención notable a scikit-learn, una librería de Python que contiene los diversos modelos que hemos probado para el desarrollo de sendas tareas.

A continuación, hemos de mencionar que hemos empleado la plataforma de Google Colab para computar los modelos, pues esta ofrece recursos de computación de forma gratuita, como por ejemplo GPU. Dado el tamaño de nuestra base, esto ha sido especialmente valioso.

Por último, pero no menos importante, para poder ofrecer un producto a partir del trabajo desarrollado, hemos explorado y empezado a emplear el paquete **Streamlit** de Python, pues este permite diseñar y desplegar aplicaciones de Machine Learning de manera bastante sencilla, puesto que el propio paquete ofrece recursos para el despliegue de aplicaciones.

**EVGENY: ENGLISH**

Throughout this project, we encountered several technological challenges stemming from the complexity of the database.

Initially, during project planning, The Gantt Project application was used to facilitate the creation of Gantt charts.

Subsequently, in executing tasks related to both data prediction and description, we faced the issue that few methods designed for such tasks performed well in our case.

For data cleaning and preparation, we primarily utilized the R language, finding it to be the most straightforward tool for this purpose. The following analyses were conducted in both R and Python to approach the issue with more flexibility

Various libraries were employed for diverse analyses, with notable mention to scikit-learn, a Python library containing diverse models tested for the respective tasks' development.

Furthermore, we utilized the Google Colab platform to compute models, benefiting from its provision of computing resources, including GPU, free of charge. Given the size of our database, this was particularly valuable.

Last but not least, in order to deliver a product based on our work, we explored the Streamlit package in Python. This tool enables the design and deployment of machine learning applications in a relatively intuitive manner, as it provides resources for application deployment within the package itself.

# References

(ESTO DICE QUE SE EXCLUYE DEL CONTADOR DE PALABRAS Y PÁGINAS)

ANEXO 1

data = read.csv('combianted\_Total.csv')

data <- data %>%

filter(data$Tipo\_De\_Intervencion\_Quirurgica == "Implantologia Bucal" | data$Tipo\_De\_Intervencion\_Quirurgica == "Cirugía Peri-implantaria" )

data <- data[, 1:111]

conteo <- table(data$Tipo\_De\_Intervencion\_Quirurgica)

#creamos columna del tipo de patología

for(i in 17:57) {

# Accedemos al nombre de la columna i-ésima

col\_name <- names(data)[i]

# Usamos un bucle para revisar cada fila

for(j in 1:nrow(data)) {

# Si encontramos un 1 en la columna i-ésima de la fila j-ésima

if(data[j, i] == 1) {

# Actualizamos 'Patologia\_1stemica' con el nombre de la columna actual

data[j, 'Patologia\_1stemica'] <- col\_name

}

}

}

data <- data %>%

mutate(Patologia\_1stemica = replace(Patologia\_1stemica, Patologia\_1stemica == "0", "ninguna"))

#creamos una columna del tipo de medicación

for(i in 60:108) {

# Accedemos al nombre de la columna i-ésima

col\_name <- names(data)[i]

# Usamos un bucle para revisar cada fila

for(j in 1:nrow(data)) {

# Si encontramos un 1 en la columna i-ésima de la fila j-ésima

if(data[j, i] == 1) {

# Actualizamos 'Patologia\_1stemica' con el nombre de la columna actual

data[j, 'Medicacion\_Actual2'] <- col\_name

}

}

}

data <- data %>%

mutate(Medicacion\_Actual2 = replace(Medicacion\_Actual2, Medicacion\_Actual2 == "0", "ninguna"))

data <- data %>%

select(-c(17:57, 60:108))

install.packages('lubridate')

library(lubridate)

# Primero, ajustamos los años en la fecha de nacimiento

data$Fecha\_Nacimiento <- as.character(data$Fecha\_Nacimiento)

data$Fecha\_Nacimiento <- sapply(data$Fecha\_Nacimiento, function(fecha) {

partes <- strsplit(fecha, "/")[[1]]

año <- as.numeric(partes[3])

if (año > 20 && año <= 99) { # Si el año está entre 21 y 99, se considera siglo XX

partes[3] <- paste0("19", partes[3])

} else if (año <= 20) { # Si el año es 20 o menor, se considera siglo XXI

partes[3] <- paste0("20", partes[3])

}

paste(partes[1], partes[2], partes[3], sep = "/")

})

data$Fecha\_Nacimiento <- dmy(data$Fecha\_Nacimiento)

# Ahora calculamos la edad

data$edad <- floor(interval(start = data$Fecha\_Nacimiento, end = Sys.Date()) / years(1))

# Finalmente, creamos la columna 'categoria\_edad' con las categorías especificadas

data <- data %>%

mutate(categoria\_edad = case\_when(

edad >= 10 & edad < 20 ~ '10-20',

edad >= 20 & edad < 30 ~ '20-30',

edad >= 30 & edad < 40 ~ '30-40',

edad >= 40 & edad < 50 ~ '40-50',

edad >= 50 & edad < 60 ~ '50-60',

edad >= 60 & edad < 70 ~ '60-70',

edad >= 70 & edad < 80 ~ '70-80',

edad >= 80 & edad < 90 ~ '80-90',

edad >= 90 & edad <= 100 ~ '90-100',

TRUE ~ 'Fuera de rango'

))

data <- data %>%

mutate(nivel\_de\_fumador = case\_when(

`Fumador.a` == 1 ~ "Fumador.a",

`No.fumador.a` == 1 & `Exfumador.a` == 1 ~ "Exfumador.a",

`Exfumador.a` == 1 & `No.fumador.a` == 0 ~ "Exfumador.a",

`No.fumador.a` == 1 & `Exfumador.a` == 0 ~ "No.fumador.a",

TRUE ~ NA\_character\_ # Para cualquier otro caso no especificado, asignamos NA

))

#eliminamos las que hemos transformado (agrupado en una columna)

data <- data %>%

select(-Fecha\_Nacimiento, -Date\_Create, -IP\_Adress, -Operador, -Auxiliar, -Jefe\_de\_dia, -Fecha\_intervencion, -Fecha\_Nacimiento, -No.fumador.a, -Exfumador.a, -Fumador.a, -Otras\_Drogas, -Medicacion\_actual, -Otro..especifique., -edad, -Tipo\_De\_cirugia, -Otro..especifique.\_17)

data <- data %>%

select(-Número\_de\_cigarrillos.día)

data <- data[!is.na(data$nivel\_de\_fumador), ]

#COmo patologia sistematica y medicacion actual contienen muchos valores con pocas muestras vamos a poner que se agrupen los que salgan menos de x veces en un valor llamado otros

# Obtener la frecuencia de cada valor en Patologia\_1stemica

freq\_patologia <- table(data$Patologia\_1stemica)

# Mostrar los valores que aparecen menos de 5 veces

print(freq\_patologia[freq\_patologia < 5])

# Identificar los valores que aparecen menos de 5 veces

infrequent\_values <- names(freq\_patologia[freq\_patologia < 5])

# Actualizar los valores a 'Otros'

data$Patologia\_1stemica[data$Patologia\_1stemica %in% infrequent\_values] <- 'Otros'

#y hacemos lo mismo con medicacion actual

# Obtener la frecuencia de cada valor en Patologia\_1stemica

freq\_medicacion <- table(data$Medicacion\_Actual2)

# Mostrar los valores que aparecen menos de 5 veces

print(freq\_medicacion[freq\_medicacion < 2])

# Identificar los valores que aparecen menos de 5 veces

infrequent\_values <- names(freq\_medicacion[freq\_medicacion < 5])

# Actualizar los valores a 'Otros'

data$Medicacion\_Actual2[data$Medicacion\_Actual2 %in% infrequent\_values] <- 'Otros'