ESCUELA INTERNACIONAL DE POSGRADO

Trabajo Final de Máster

Máster en Ingeniería del Software: Cloud, Datos y Gestión TI

MODELOS PREDICTIVOS PARA LA DETECCIÓN DE RIESGOS EN EL EMBARAZO

Realizado por

Lola Gómez Jemes

Dirigido por

DRA. MARÍA DEL CARMEN ROMERO TERNERO

ANDREEA MADALINA OPRESCU

Departamento Tecnología Electrónica

Sevilla, 10 diciembre de 2021

Considerando que la presentación de un trabajo hecho por otra persona o la copia de textos, fotos y gráficos sin citar su procedencia se considera plagio,

Yo, Dña. Lola Gómez Jemes, con NIF número 32733632D, estudiante del Máster Ingeniería del Software: Cloud, Datos y Gestión TI en la Escuela Internacional de Posgrado de la Universidad de Sevilla,

ASUMO LA AUTORÍA RESPONSABLE Y DECLARO que el Trabajo de Fin de Máster que presento para su exposición y defensa titulado:

MODELOS PREDICTIVOS PARA LA DETECCIÓN DE RIESGOS EN EL EMBARAZO y cuyas tutoras son Dña. María del Carmen Romero Ternero y Dña. Andreea Madalina Oprescu

ES ORIGINAL Y QUE TODAS LAS FUENTES UTILIZADAS PARA SU REALIZACIÓN HAN SIDO DEBIDAMENTE CITADAS EN EL MISMO.

Así mismo, acepto que se utilicen las herramientas de control del plagio que garanticen la autoría de este Trabajo de Fin de Máster.

En Sevilla, a 10 de diciembre de 2021

Fdo.: Lola Gómez Jemes

RESUMEN

El uso de la inteligencia artificial en ámbito de la salud en general, y en ámbito obstétrico y ginecológico en concreto, tiene un gran potencial. La IA podría ayudar a mejorar la salud universal de las embarazadas, monitorizar de forma más cercana la salud durante el embarazo, o disminuir la morbilidad y mortalidad materna y perinatal con la detección precoz de enfermedades.

En este Trabajo Final de Máster, se propone el desarrollo de un modelo predictivo de riesgos en embarazadas, en concreto de predección de la preeclampsia y el retraso del crecimiento intrauterino. Para ello, se va a utilzar un dataset perteneciente a un estudio llevado a cabo por el Centro Médico Universitario de Liubliana, en el que se recogen los datos de 95 mujeres embarazadas con preeclampsia y retraso de crecimiento intrauterino.

PALABRAS CLAVE

Inteligencia artificial, embarazo, aprendizaje automático, modelos predictivos, riesgos, preeclampsia, retraso de crecimiento intrauterino

ABSTRACT

The use of artificial intelligence in healthcare in general, and in obstetrics and gynecology in particular, has great potential. AI could help to improve the universal health of pregnant women, monitoring their health parameters during pregnancy closely, or reducing maternal and perinatal morbidity and mortality with early detection of pathologies.

In this Master Thesis, we propose the development of a predictive model of risk in pregnancy, in particular the prediction of preeclampsia and intrauterine growth restriction. For this purpose, we will use a dataset from a study carried out by the University Medical Center of Ljubljana, in which data from 95 pregnant women with preeclampsia and intrauterine growth restrition were collected.

KEY WORDS

Artificial Intelligence, pregnancy, machine learning, predictive models, risk, preeclampsia, intrauterine growth restrition

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

1 DESCRIPCIÓN DEL PROYECTO

1.1 INTRODUCCIÓN Y MOTIVACIÓN

El uso de la Inteligencia Artificial (IA) en el ámbito de la salud tiene un gran potencial para mejorar la atención sanitaria en todo el mundo. Entre sus posibles aplicaciones se pueden destacar las siguientes: ofrecer apoyo a los profesionales sanitarios mediante información científica actualizada y de confianza, prácticas médicas novedosas o mejoras en prácticas ya existentes que pueden ayudar a la atención del paciente, reducir los errores de diagnóstico y encontrar información relevante en grandes volúmenes de datos para reducir los riesgos de salud y poder realizar predicciones diagnósticas en tiempo real (Jiang *et al.*, 2017). Además, podría facilitar el acceso a los servicios de salud en los países en vías de desarrollo con recursos más escasos, o zonas rurales en las que los pacientes a menudo tienen dificultades para recibir atención sanitaria frecuente y de calidad. Sin embargo, la IA también podría suponer un riesgo si no se usa de forma debida. En ese sentido, la Organización Mundial de la Salud (OMS) publicó en junio de 2021 una guía para garantizar una aplicación ética de la inteligencia artificial en temas relacionados con la salud (WHO, 2021). Uno de los principios que destacan en la guía es que se debe garantizar la transparencia, la claridad y la inteligibilidad de tecnología de IA utilizada. La transparencia del modelo permite a los usuarios entender, auditar e incluso corregir las decisiones tomadas por el modelo. Algunos incluso sostienen que se debería anteponer la transparencia/explicabilidad del modelo a la precisión (WHO, 2021).

En el ámbito obstétrico y ginecológico en concreto, el uso de IA también tiene un gran potencial. Oprescu et al. (A.M. Oprescu *et al.*, 2020) hace una revisión sobre la literatura científica publicada entre 2008 y 2020, en la que queda patente el creciente interés de la comunidad científica en este ámbito. Algunas de las conclusiones obtenidas en este estudio son que la IA podría mejorar la salud universal de las embarazadas, al disminuir las barreras impuestas por factores socioeconómicos y demográficos (países en vías de desarrollo, o zonas rurales con acceso limitado a atención sanitaria), permitiría una monitorización más cercana durante el embarazo, una disminución de la morbididad materna a través de análisis de datos automáticos, y la detección temprana de cambios en la patología de la embarazada.

En este Trabajo Fin de Máster (TFM) se va a llevar a cabo un modelo predictivo de riesgos en embarazadas. En concreto, se va a desarrollar un modelo para predecir si una embarazada padece preeclampsia, retraso del crecimiento intrauterino o ambas afecciones.

La preeclampsia (PE) y el retraso de crecimiento intrauterino (Intrauterine Growth Restriction, IGR) son complicaciones del embarazo relacionadas con una disfunción de la placenta.

Por un lado, la preeclampsia es un trastorno hipertensivo gestacional con una incidencia global del alrededor del 3-5%. Se presenta después de la semana 20 de gestación, y a veces lo hace junto con otras disfunciones orgánicas maternas como la insuficiencia renal, la afectación hepática, complicaciones neurológicas o hematológicas, disfunción uteroplacentaria, o restricción del crecimiento intrauterino (Mol *et al.*, 2016).

Por otro, el retraso del crecimiento intrauterino se define como el crecimiento fetal inferior al crecimiento normal potencial debido a factores genéticos o ambientales (Murki and Sharma, 2014).

Tanto la PE como la IGR son consideradas una causa importante de morbilidad y mortalidad fetal, neonatal y materna (Friedman and Cleary, 2014). La mejor manera de reducir los riesgos de la preeclampsia es detectarla de forma temprana, así el médico puede adoptar todas las medidas necesarias para frenar sus consecuencias. De esta forma, la predicción temprana de ambas condiciones clínicas es crucial para mejorar los resultados maternos y perinatales. Así, el desarrollo de un modelo de aprendizaje automático para la predicción de estas enfermedades podría ser una herramienta de gran valor para dar soporte al personal clínico.

La PE y la IGR se caracterizan por una formación anormal de la placenta que provoca que el flujo sanguíneo uteroplacentario no sea el adecuado. La ecografía Doppler de la arteria uterina es un método diagnóstico no invasivo que utiliza el sonido de alta frecuencia que permite evaluar la circulación uteroplacentaria. El empleo del Doppler de arterias uterinas no se ha conseguido imponer en la práctica habitual, pero en combinación con los marcadores angiogénicos sFlt-1 (*fms-like tyrosine kinase 1*, en español Tirosina quinasa 1 soluble tipo fms) y PlGF (*placental growth factor*, en español factor de crecimiento placentario) se podría convertir en una herramienta con gran potencial para la predicción y el diagnóstico temprano de la preeclampsia (García *et al.*, 2011).

Para llevar a cabo el modelo, se van a utilizar un conjunto de datos públicos disponibles en *Data Mendeley* (*Mendeley Data - Uterine arteries Doppler and sFlt-1/PlGF ratio in hypertensive disorders during pregnancy*, no date). Estos datos pertenecen a un estudio de cohorte prospectivo realizado por el Centro Médico Universitario de Liubliana, en el que se recogieron datos desde 2012 hasta 2015 de un total de 95 mujeres (V *et al.*, 2019). En este *dataset* recoge datos de embarazadas entre la semana 24 y 37 de gestación, entre los que encuentran los valores obtenidos de la ecografía Doppler de la arteria uterina, y los valores de la tirosina quinasa 1 soluble tipo fms (sFlt-1, también conocida como receptor 1 del factor de crecimiento vascular) y factor de crecimiento placentario (PlGF).

Se realizará un análisis exploratorio sobre los datos, se aplicarán distintas técnicas de preprocesamiento, y se entrenará un modelo multietiqueta con dos posibles salidas: preeclampsia (PE) y retraso del crecimiento intrauterino (IUGR). Se hará una discusión sobre los distintos modelos obtenidos y se hará una elección final del modelo, teniendo en cuenta métricas como *recall, Hamming loss, acuracy* frente a la explicabilidad del modelo.

Además, se realizará una revisión de la literatura científica sobre modelos de detección de riesgos en embarazadas publicados entre 2020 y 2021.

1.2 OBJETIVOS

Los objetivos establecidos en este Trabajo Fin de Máster son:

1. Estudiar y aplicar las técnicas explicables (white box) de Machine Learning para la predicción de riesgo en el embarazo, en concreto preeclampsia y retraso del crecimiento intrauterino.
2. Estudiar distintas librerías y frameworks de Python para Machine Learning.
3. Desarrollar un modelo predictivo siguiendo la guía establecida en Guidelines for Developing and Reporting Machine Learning Predictive Models in Biomedical Research: A Multidisciplinary View (Wei *et al.*, 2016).
4. Estudiar y aplicar técnicas de explicabilidad de Machine Learning (*XAI)*.

1.3 METODOLOGÍA

La metodología seguida para realizar el modelo de predictivo viene descrita en Guidelines for Developing and Reporting Machine Learning Predictive Models in Biomedical Research: A Multidisciplinary View (Wei *et al.*, 2016).

En los últimos años, se ha visto aumentado el uso de modelos de Machine Learning en la investigación biomédica. Sin embargo, no siempre se reportan los resultados de forma correcta, lo que dificulta la evaluación de la validez del modelo y la interpretación de sus resultados (Wei *et al.*, 2016).

Esta guía surge de la necesidad de proporcionar una serie de directrices que permitan comprobar que los modelos se aplican correctamente, y que se comunican de forma suficiente para poder distinguir entre descubrimientos reales y coincidencias aleatorias.

De esta forma, se proporciona una guía especificando la información que debería incluirse en cada una de las secciones del artículo de investigación en el que se describe el modelo predictivo desarrollado.

A continuación, se muestra un cuadro resumen con la información que debería incluirse en cada sección según esta metodología.

Sección	Tema	Información para incluir
Introducción	Justificación	- Identificar el objetivo clínico. - Revisar los modelos ya existentes.
	Objetivos	- Definir el objeto de predicción. - Identificar cómo se beneficiaría el objetivo clínico de la predicción obtenida por el modelo.
Métodos	Describir el contexto	- Identificar el entorno clínico para el modelo predictivo objetivo. - Identificar el contexto de modelización en términos de tamaño, volumen y duración de los datos disponibles.
	Definir el problema de clasificación	- Determinar si el estudio es un pronóstico o diagnóstico. - Determinar el tipo de problema: clasificación o regresión. - Definir métrica de evaluación del modelo - Definir el criterio de éxito para la predicción
	Preparación de los datos para el desarrollo del modelo	- Identificar fuente de datos relevantes - Describir lo criterios de inclusión y exclusión de los datos - Describir el tamaño de la muestra de los datos y tiempo total de recolección de datos. - Definir variables del conjunto de datos y sus unidades. - Describir todos los procesos de preprocesamiento aplicados a los datos (limpieza de datos y transformaciones). - Eliminar valores imposibles (*outliers* de tipo atípico que surgen de un error de procedimiento).

Sección	Tema	Información para incluir
		- Explicar cómo se gestionan los valores ausentes. - Describir las estadísticas básicas del conjunto de datos, en particular de la variable de respuesta. Para el caso de clasificación, incluir la relación de clases positivas y negativas, y para uno de regresión la distribución de la variable de respuesta. - Definir las estrategias de validación del modelo, siendo requisito mínimo la validación interna, y deseable la validación externa. - Definir las métricas usadas para evaluar los modelos. Para regresión, se usará la raíz del error cuadrático medio. Para clasificación, se deberá incluir los valores de sensibilidad, especificidad, el valor predictivo positivo y negativo, y el área bajo la curva ROC.
	Desarrollo del modelo predictivo	- Eliminar variables independientes redundantes. - Evaluar si se dispone de suficientes datos para un buen ajuste del modelo. - Determinar el conjunto de técnicas de modelización que se van a evaluar (regresión logística, *random forest…).* - Definir las métricas de rendimiento para elegir el mejor modelo. - Evaluar el equilibrio entre la precisión del modelo y su interpretabilidad.
Resultados	Informar sobre el modelo final	- Informar del modelo final y su rendimiento en función de las métricas establecidas en la sección anterior. - Si es posible, informar sobre las estimaciones de los parámetros del modelo y sus intervalos de confianza. - Interpretación del modelo final. Comentar qué variables han demostrado ser predictivas de la variable respuesta.

Tabla 1. Cuadro elaborado a partir de la guía descrita en Guidelines for Developing and Reporting Machine Learning Predictive Models in Biomedical Research: A Multidisciplinary View

Esta guía se ha seguido a la hora de documentar el modelo desarrollado en este Trabajo Fin de Máster.

2 PLANIFICACIÓN

2.1 PLANIFICACIÓN TEMPORAL

2.1.1 Metodología

En el desarrollo de este TFM, se ha seguido una metodología de trabajo ágil. El término *Metodología Ágil* fue acuñado por primera vez en el año 2001, cuando un grupo de profesionales de la Ingeniería de Software se reunieron con el objetivo de proponer nuevas ideas para mejorar el proceso de desarrollo software. Este grupo, *"The Agile Alliance"*, fueron los responsables de dar inicio al Movimiento Ágil que conocemos hoy en día. Este término

surge para nombrar a los nuevos métodos, más ligeros y flexibles, que comenzaban a emerger en contraposición a los métodos tradicionales (*History: The Agile Manifesto*, no date). Algunos ejemplos de metodologías ágiles son *Scrum, Extreme Programming, Lean Development* y *Agile Modeling.*

En este proyecto, se ha seguido la metodología SCRUM. SCRUM es una de las metodologías ágiles más representativas y utilizadas en la actualidad. Su enfoque inicial fue para utilizarlo en proyectos de desarrollo de Software, aunque sus principios son fácilmente adaptables en cualquier contexto o área de conocimiento (Schwaber, 1997).

Se ha utilizado Jira (*Jira | Software de seguimiento de proyectos e incidencias*, no date) como herramienta de gestión de proyectos. Jira es una herramienta en línea desarrollada por la empresa Atlassian para facilitar la planificación y supervisión de tareas en un proyecto. Además, tiene disponible distintas plantillas para llevar a cabo la planificación, entre las cuáles se encuentra disponible una plantilla para SCRUM.

En primer lugar, se definió las tareas que se debían cubrir para llevar a cabo el Trabajo Final de Máster. Esto constituyó el *Product Backlog*. *En la Figura 1. Captura de pantalla de fragmento del backlog de tareas en Jira* se puede observar un ejemplo de las tareas que constituían el *Backlog.* De ahí se seleccionaban las tareas que iban a ser desarrolladas durante los *sprints* (bloques de tiempo de trabajo).

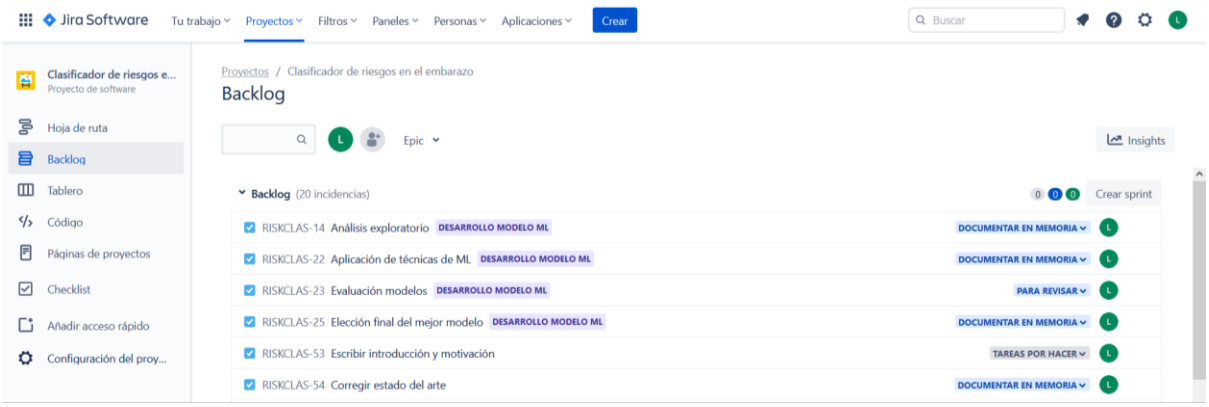

Figura 1. Captura de pantalla de fragmento del backlog de tareas en Jira

Los *sprints* son el centro de la metodología SCRUM. Corresponden al proceso de desarrollo de las necesidades del cliente divididas en módulos funcionales, de tal manera que al final de cada Sprint se obtiene un producto incremental (Schwaber, 1997).

Se establecieron *sprints* de dos semanas. Al final del *sprint,* se llevaba a cabo una reunión de seguimiento con las tutoras del TFM, en las que se trataban los siguientes puntos:

- Se comentaban las tareas completadas.
- Se revisaban las tareas que necesitaran algún tipo de validación.
- Se resolvían dudas surgidas durante el *sprint*.
- Se comprobaba si había quedado pendiente alguna tarea y, en caso afirmativo, se pasaba al siguiente *sprint.*
- Se planificaban las tareas del siguiente *sprint*.

En Jira se llevaban a acabo la planificación de los *sprints,* incluyendo las tareas que debían completarse durante esas dos semanas. La *Figura 2. Ejemplo de programación de sprint en Jira* muestra uno de los *sprints* completados durante el desarrollo del TFM.

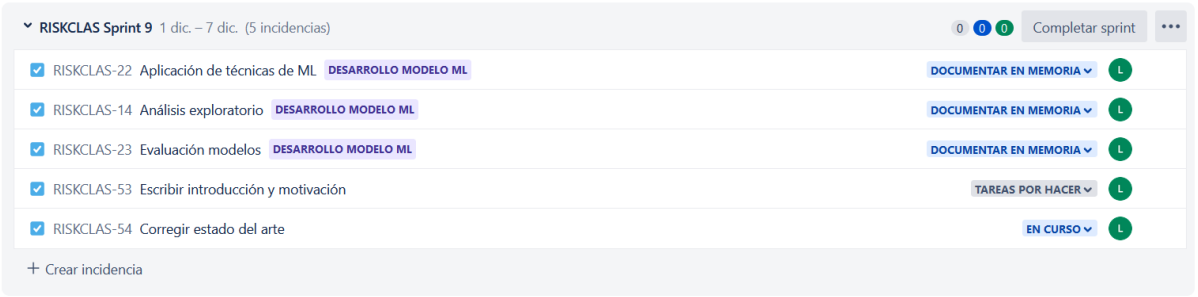

Figura 2. Ejemplo de programación de sprint en Jira

Se utilizó un **tablero de Kanban** para llevar un seguimiento de las tareas. Este tablero consta de un número variable de columnas (habitualmente entre 3 y 5), a través del cual se observa el avance de las tareas planificadas durante un Sprint. En este caso, se han diferenciado cuatro columnas distintas:

- **Por hacer:** backlog del sprint. Contiene todas las tareas del *sprint* que aún no han comenzado.
- **En curso:** tareas en las que se estaban trabajando.
- **Para revisar:** tareas completadas que necesitaban ser consultadas con las tutoras al final del *sprint*.
- **Documentar en memoria:** tareas listas para ser documentadas en la memoria del TFM.
- **Listo:** tareas finalizadas. Se considera una tarea finalizada si ya ha sido documentada en la memoria (en caso de requerirlo), o ya ha sido implementada (si está relacionada con el desarrollo del código).

La *Figura 3. Captura de pantalla tablero de Jira* muestra una captura real del tablero Kanban del proyecto. Como se puede observar, tablero permite conocer a simple vista el estado de las tareas que se están desarrollando durante el *sprint*, favoreciendo la transparencia al conocer en qué se está trabajando en cada momento.

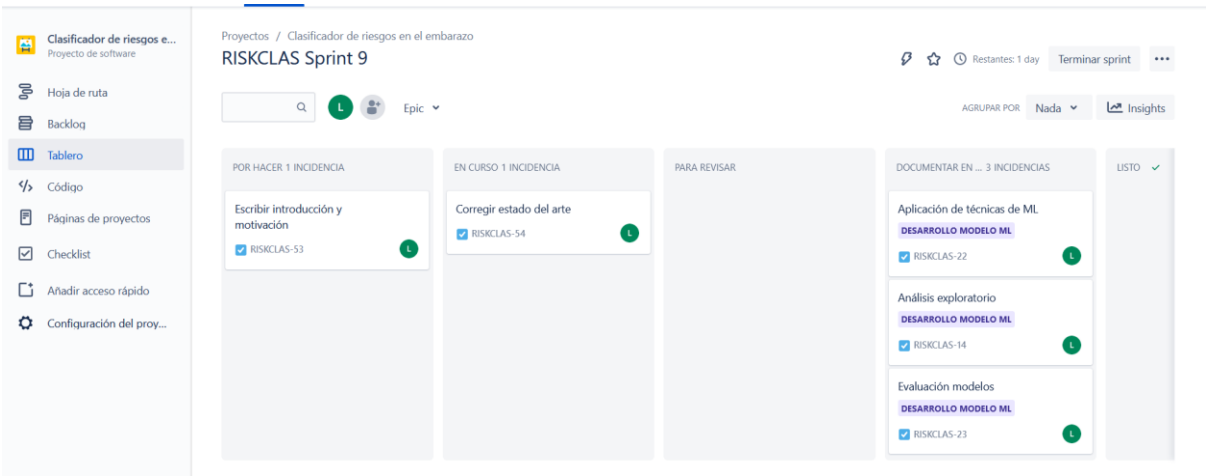

Figura 3. Captura de pantalla tablero de Jira

2.1.2 Planificación temporal inicial

Para la planificación temporal del proyecto, se ha decidido utilizar la estructura de descomposición de trabajo (EDT) (*Guía de los fundamentos para la dirección de proyectos : (Guía del PMBOK®)*, 2017). La EDT es una descomposición jerárquica del trabajo que debe ser llevado a cabo en el proyecto. La EDT subdivide el trabajo en porciones más pequeñas y fáciles de manejar. Existen dos tipos de enfoque para llevarla a cabo: una basada en entregables y otra basada en fases. Se ha elegido el enfoque basado en fases para este trabajo, identificando las siguientes fases:

1. **Inicial**

   Fase inicial del proyecto. En esta fase se definirá el alcance del proyecto y los objetivos. Además, se hará una búsqueda de *datasets* para llevar a cabo el modelo predictivo de riesgos en embarazadas.

2. **Revisión de la literatura científica**

   En esta fase se va a plantear una pregunta de investigación en relación al tema del TFM. Para darle respuesta, se consultará una base de datos bibliográfica (*Scopus*), y se revisará la literatura disponible. A partir de ello, se llevará a cabo el Estado del Arte.

3. **Análisis exploratorio**

   Se realizará la carga y lectura del *dataset* elegido durante la fase inicial. Esta fase tendrá como finalidad conocer más en profundidad los datos del problema de clasificación, para poder definir qué clase de preprocesamiento se necesitará aplicar a los datos antes de llevar a cabo el entrenamiento del modelo..

4. **Preprocesamiento de los datos**

   A partir de la información recabada en la fase 3, se determinará qué transformaciones hay que aplicar a los datos. En un principio, se establece 4 procesos de preprocesamiento: selección de atributos, imputación de valores ausentes, normalización de las variables numéricas, y codificación de las variables categóricas.

5. **Entrenamiento del modelo de machine learning**

   En la fase 5 se llevará a cabo el entrenamiento del modelo. Se ha establecido que el modelo que se va a desarrollar va a ser del tipo multietiqueta. En esta fase también se contempla un tiempo de formación en este tipo de problemas de clasificación.

6. **Validación del modelo de machine learning**

   Una vez entrenados, se procederá a su evaluación. Se tendrán en cuenta múltiples métricas para llevar a cabo la comparación de los modelos.

7. **Documentación del trabajo**

   Fase final del proyecto. Se contempla dos tipos de documentación diferentes: la memoria del TFM y el documento de *jupyter notebook*, que será el documento que contenga todo el código desarrollado para el TFM.

En la *Figura 4. Estructura de desglose de trabajo* (elaboración propia)se muestra la estructura de desglose de trabajo, indicando para cada fase sus correspondientes subtareas.

Una vez identificadas las tareas, se ha utilizado la herramienta de Microsoft Project para llevar a cabo la planificación temporal. En la *Figura 5. Estimación temporal de subtareas del proyecto* (elaboración propia) se muestra la estimación inicial para cada una de las tareas identificadas en la EDT

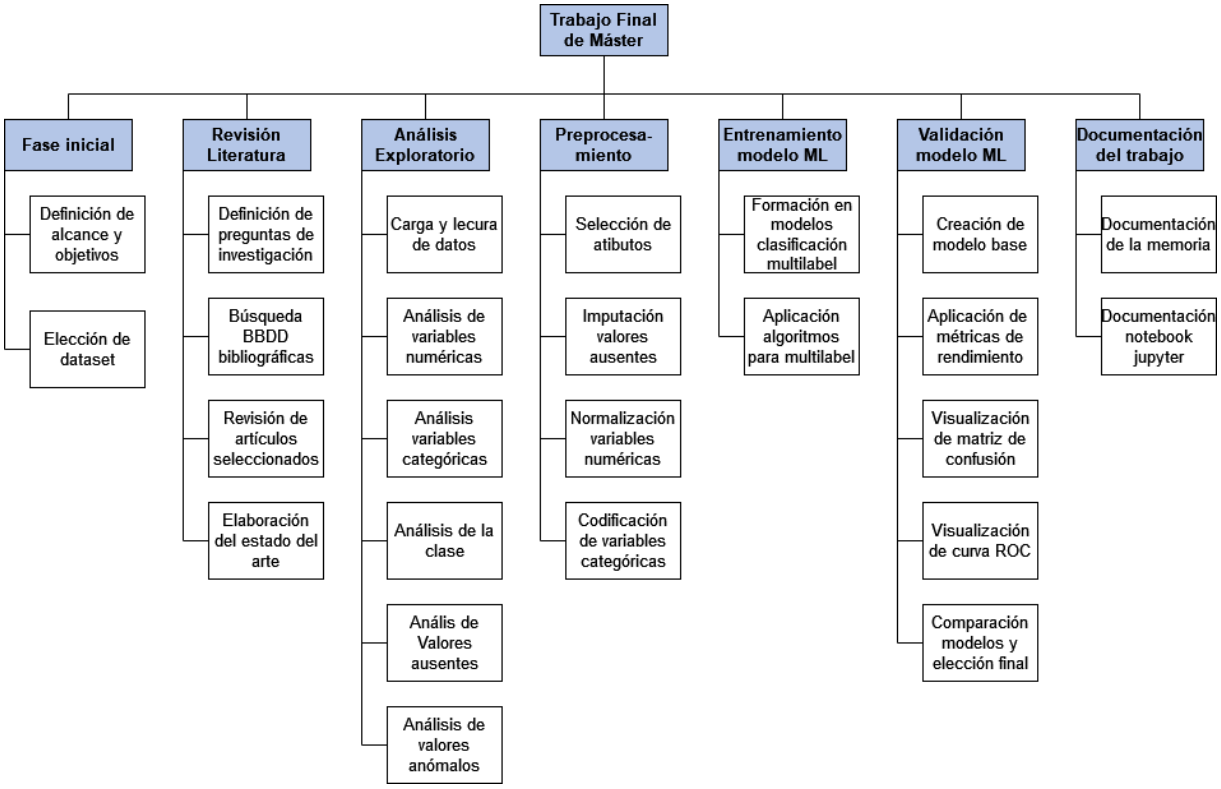

Figura 4. Estructura de desglose de trabajo (elaboración propia)

Se ha establecido como fecha de inicio el 1 de septiembre 2021, y como fecha final el 26 de noviembre de 2021, alcanzando un total de 285 horas de trabajo. En la *Figura 6. Diagrama de Gantt de planificación inicial* (elaboración propia) se muesta el diagrama de Gantt de la planificación inicial.

2.1.1 Planificación temporal final

Se ha producido algunos retasos en las tareas, lo que ha impactado en la planificación final del Trabajo Final de Máster. Esto se ha debido a que se ha tenido que revisar más literatura de la esperada al principio, y a varias correcciones que se le han tenido que aplicar al modelo de *machine learning* desarrollado. El número total de hora trabajados han sido 315 horas.

En la *Figura 7. Diagrama de Gantt de planificación final* (elaboración propia)Figura 7. Diagrama de Gantt de planificación final (elaboración propia)Figura 7. Diagrama de Gantt de planificación final (elaboración propia) se muesta el diagrama de Gantt de seguimiento, en el que se compara la planificación inicial (línea base) con la planificación final del trabajo.

Nombre de tarea	Duración	Comienzo	Fin	Predecesoras
Fase Inicial	**2 días**	**mié 01/09/21**	**jue 02/09/21**	
Reunión inicial y definición de objetivos y alcance TFM	1 día	mié 01/09/21	mié 01/09/21	
Búsqueda y elección de dataset	2 días	mié 01/09/21	jue 02/09/21	
Revisión de la literatura	**19 días**	**vie 03/09/21**	**mié 29/09/21**	
Definición de preguntas de investigación	1 día	vie 03/09/21	vie 03/09/21	3
Búsqueda en BBDD bibliográficas	3 días	lun 06/09/21	mié 08/09/21	5
Revisión de artículos seleccionados	10 días	jue 09/09/21	mié 22/09/21	6
Elaboración del estado del arte	5 días	jue 23/09/21	mié 29/09/21	7
Análisis Exploratorio de los datos	**10 días**	**jue 30/09/21**	**mié 13/10/21**	
Carga y lectura de los datos	1 día	jue 30/09/21	jue 30/09/21	8
Análisis de las variables numéricas	3 días	vie 01/10/21	mar 05/10/21	10
Análisis de las variables categóricas	2 días	mié 06/10/21	jue 07/10/21	11
Análisis de la clase	2 días	vie 08/10/21	lun 11/10/21	12
Análisis de los valores ausentes	1 día	mar 12/10/21	mar 12/10/21	13
Análisis de valores anómalos	1 día	mié 13/10/21	mié 13/10/21	14
Preprocesamiento	**8 días**	**jue 14/10/21**	**lun 25/10/21**	
Selección de atributos	1 día	jue 14/10/21	jue 14/10/21	15
Imputación de valores ausentes	2 días	vie 15/10/21	lun 18/10/21	17
Normalización de variables numéricas	3 días	mar 19/10/21	jue 21/10/21	18
Codificación de variables categóricas	2 días	vie 22/10/21	lun 25/10/21	19
Entrenamiento del modelo de ML	**18 días**	**mar 26/10/21**	**jue 18/11/21**	
Formación en modelos clasificación multilabel	6 días	mar 26/10/21	mar 02/11/21	20
Aplicación algoritmos para clasificación multilabel	12 días	mié 03/11/21	jue 18/11/21	22
Validación del modelo ML	**6 días**	**vie 19/11/21**	**vie 26/11/21**	
Creación de modelo base	1 día	vie 19/11/21	vie 19/11/21	23
Aplicación de métricas de evaluación de rendimiento a los modelos entrenados	3 días	lun 22/11/21	mié 24/11/21	25
Visualización matriz de confusión de modelos evaluados	1 día	lun 22/11/21	lun 22/11/21	26CC
Visualización de gráfica de curva ROC de modelos evaluados	2 días	lun 22/11/21	mar 23/11/21	26CC
Comparación de modelos y elección final	2 días	jue 25/11/21	vie 26/11/21	26
Documentación del trabajo	**15 días**	**lun 08/11/21**	**vie 26/11/21**	
Documentación de la memoria	15 días	lun 08/11/21	vie 26/11/21	29FF
Documentación de notebook de jupyter	5 días	lun 22/11/21	vie 26/11/21	29FF

Figura 5. Estimación temporal de subtareas del proyecto (elaboración propia)

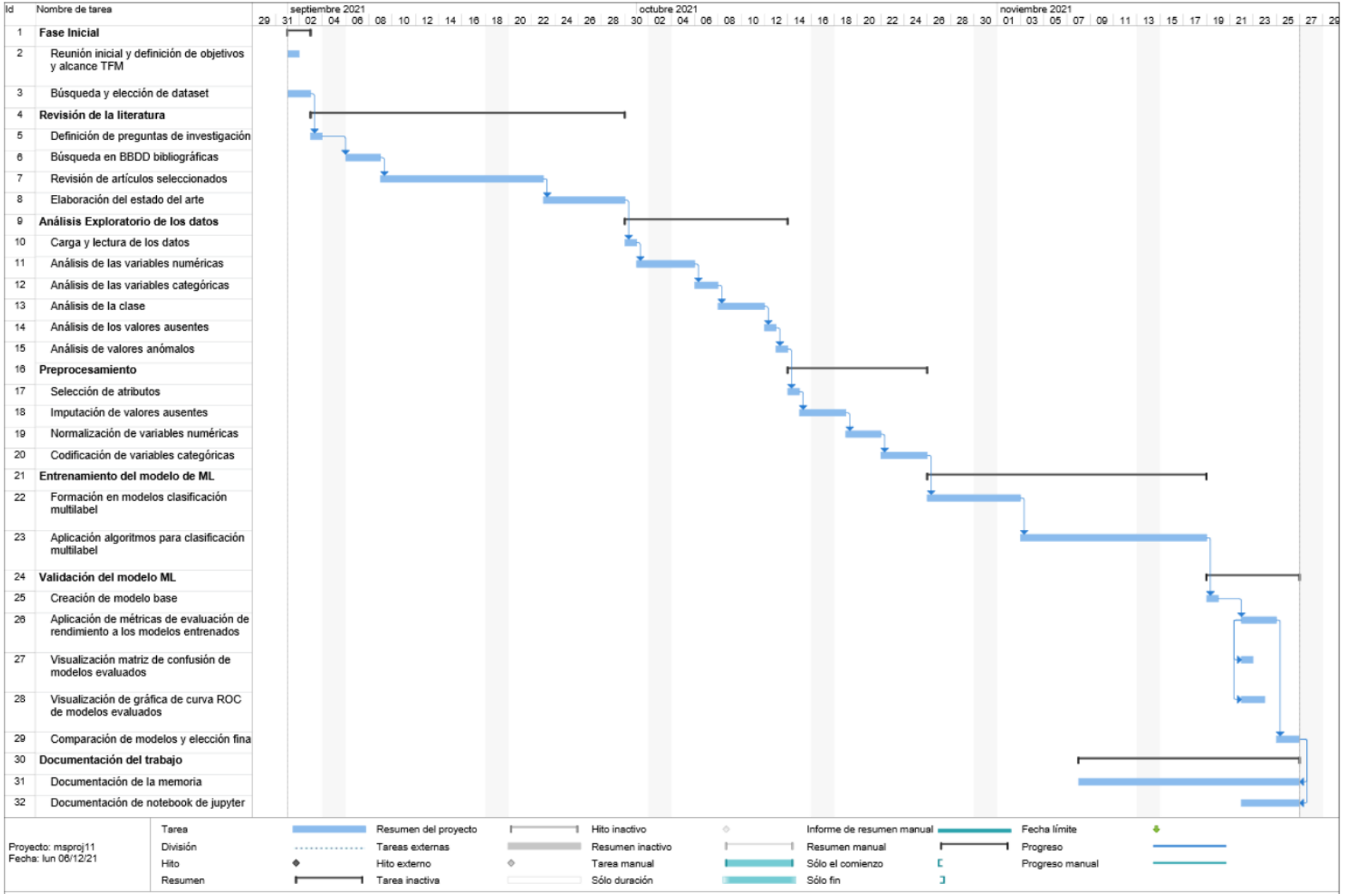

Figura 6. Diagrama de Gantt de planificación inicial (elaboración propia)

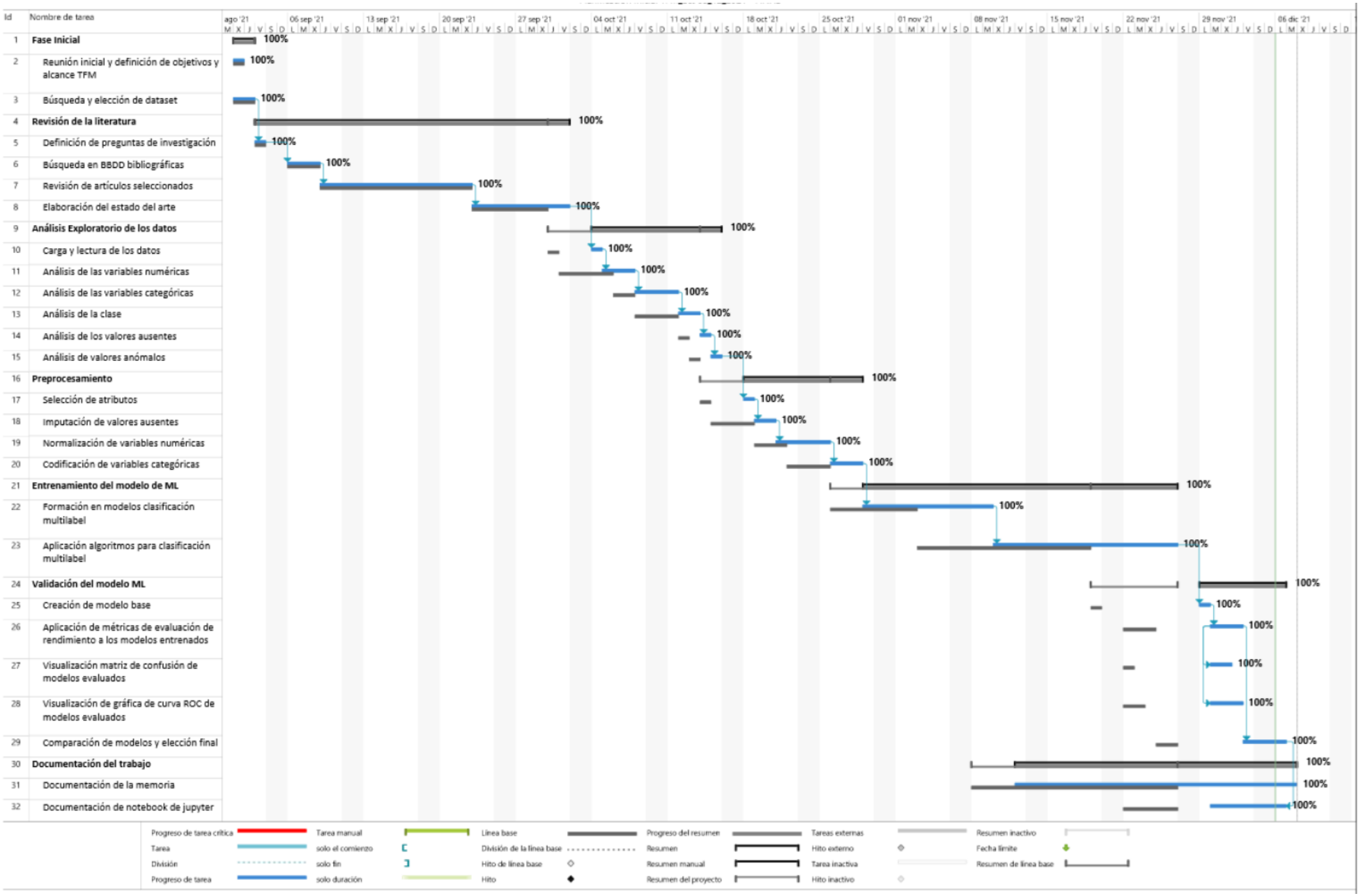

Figura 7. Diagrama de Gantt de planificación final (elaboración propia)

3 ESTADO DEL ARTE

En este capítulo, se va a tratar el estado de la cuestión sobre el uso de modelos predictivos para la detección de riesgos durante el embarazo. Para ello, se ha partido de la revisión sistemática exploratoria realizada en *Artificial Intelligence in Pregnancy: A Scoping Review* (Andreea M. Oprescu *et al.*, 2020). En ella, se hace una revisión de la literatura científica publicada entre 2008 y 2020 para identificar qué metodologías, técnicas, algoritmos y *frameworks* de IA han sido aplicados para mejorar la salud y bienestar de las embarazadas. Asimismo, el estudio busca evaluar el uso de la computación afectiva en este ámbito, ya que el estado emocional también supone un factor de riesgo durante el embarazo.

Oprescu et al. presentan una visión general de todo el material revisado en función de distintas características como aspectos generales (por ejemplo, autores, palabras claves y país de publicación), aspectos médicos (procesos asociados con el embarazo tratado, características del embarazo, información sobre el parto) o aspectos sobre *machine learning* (tipo de aprendizaje utilizado, técnicas de preprocesamiento aplicadas, medidas de rendimiento, etc.). De esta forma, se presenta una revisión muy completa que da una visión global de las tendencias actuales en la investigación del uso de IA para mejorar la salud y el bienestar de las embarazadas. Además, pone de manifiesto el creciente interés de la comunidad científica en este tema y su gran potencial. Algunas de las conclusiones extraídas por los autores es que la IA permitiría mejorar la salud universal de las embarazadas al disminuir las dificultades impuestas por los factores socioeconómicos; facilitaría una atención más cercana a las pacientes, lo que generaría un sentimiento de mayor seguridad y disminuiría su preocupación o ansiedad; y la posibilidad de poder monitorizar factores de riesgos a través de dispositivos *wearables.* Igualmente, se ha identificado que, de momento, no existe una cantidad significativa de estudios en los que se haya explorado el uso de computación afectiva para dar soporte a la salud mental de las embarazadas y, en consecuencia, a su salud física.

En este Trabajo Final de Máster, se va a continuar con este mismo hilo conductor, pero revisando los documentos publicados entre enero de 2020 y septiembre 2021, ya que este periodo no está contemplado en la *scoping review*. No obstante, la búsqueda se va a limitar un poco más, centrándose especialmente en el uso de la Inteligencia Artificial en la detección de **riesgos** durante el embarazo. De esta forma, se parte de la siguiente pregunta de investigación de estado del arte:

¿Qué metodologías, técnicas, algoritmos y marcos de trabajo son utilizados en inteligencia artificial para la detección de riesgos durante el embarazo?

Para responderla, se va a consultar la base de datos bibliográfica de *Scopus.* Scopus es una base de datos de referencias bibliográficas y citas que incorpora un gran número de registros de revistas científicas, libros y actas de congresos, entre otros. Además, posee una serie de herramientas que permiten realizar búsquedas más avanzadas.

Tomando como referencia a Oprescu et al., se van a utilizar las siguientes palabras clave en la búsqueda:

- **Términos relacionados con IA:** *artificial intelligence*, *machine learning* y *affective computing.*
- **Términos relacionados con la población objetivo del trabajo**: *pregnancy* y *pregnant.*
- **Términos relacionados con la salud y bienestar**: *health* y *well being*

Asimismo, se han añadido un término adicional que no estaba contemplado por los autores: *risk* (en español, riesgo).

Se han utilizado los operadores *OR* y *AND* para relacionar los términos de búsqueda. Por un lado, *OR* ha sido usado para relacionar los términos dentro de la misma categoría para indicar que al menos alguna de esas palabras debe aparecer. Por otro lado, *AND* ha sido el nexo entre las distintas categorías, para asegurar que todas las categorías de palabras estén presentes en los artículos obtenidos como resultado de la búsqueda. La consulta se muestra a continuación:

(("artificial intelligence" OR "machine learning" OR "affective computing") AND ("health" OR "well being") AND "risk" AND ("pregnancy" OR "pregnant"))

Se ha especificado que estos términos de interés deben aparecer en los campos de título, resumen y palabras clave de los documentos (*TITLE-ABS-KEY).* Además, la búsqueda se ha limitado sólo a artículos (*article*) y actas de conferencia (*article paper*) escritos en inglés y publicados entre los años 2020 y 2021. De esta forma, la búsqueda final es:

TITLE-ABS-KEY (("artificial intelligence" OR "machine learning" OR "affective computing") AND (health OR "well being") AND RISK AND (pregnancy OR pregnant)) AND (LIMIT-TO (DOCTYPE, "ar") OR LIMIT-TO (DOCTYPE , "cp")) AND (LIMIT-TO (PUBYEAR , 2021) OR LIMIT-TO (PUBYEAR, 2020)) AND (LIMIT-TO (LANGUAGE, "English"))

Se han obtenido 95 documentos que cumplen con los criterios especificados. Como se muestra en la *Figura 8. Documentos obtenidos en la búsqueda clasificados por área temática*. Elaboración propia a partir de los resultados obtenidos en Scopus. estos documentos pertenecen a distintas áreas temáticas como ingeniería, matemáticas, medicina o informática. En el presente trabajo, se ha limitado el alcance de la revisión a los documentos pertenecientes al área de informática (*Computer Science*), que suponen un total de 27 documentos.

Existe un gran número de estudios científicos publicados este último año que muestran cómo la IA puede contribuir a la detección de factores de riesgos que pueden afectar a la salud y al bienestar de las embarazadas.

De estos 27 artículos iniciales, se han descartado ocho: tres porque eran modelos predictivos de patologías concretas enfocados a la población en general (anemía falciforme (Alfaleh and Gollapalli, 2020), obesidad (Chatterjee, Gerdes and Martinez, 2020) y cáncer de cuello de útero y ovario (Asaduzzaman *et al.*, 2021)); tres por tratarse de revisiones de la literatura en los que no se desarrollan un modelo de *machine learning* ((A.M. Oprescu *et al.*, 2020; Damaraji, Permanasari and Hidayah, 2020; Chinnaiyan and Alex, 2021); uno por tratar temas de seguridad en el uso de datos clínicos ((Manasrah *et al.*, 2021); y por último se ha descartado un artículo por centrarse en el desarrollo de un modelo de predicción de depresión posparto en los padres (Shatte *et al.*, 2020).

En el resto de artículos, se llevan a cabo modelos de clasificaciónde de tipo supervisado. Entre las ténicas más utilizadas se encuentran *Random Forest, Support Vector Machine* y *KNeighbors*, y como métricas de evaluación se suelen utilizar los valores de *accuracy, recall, precision, AUROC* y *F1-score.*

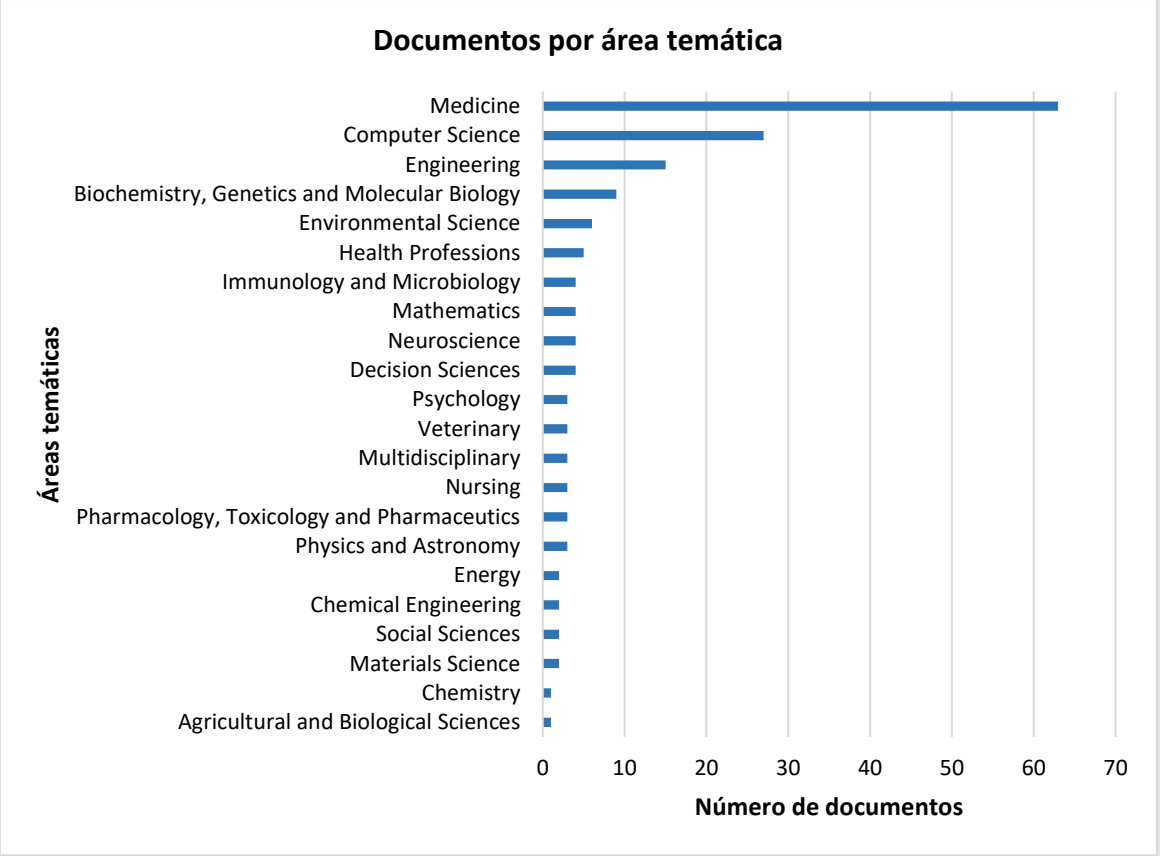

Figura 8. Documentos obtenidos en la búsqueda clasificados por área temática. Elaboración propia a partir de los resultados obtenidos en Scopus.

La mayoría de artículos utilizan datos disponibles en repositorios, seguidos de datos obtenidos a partir de registros médicos (por ejemplo, historiales clínicos).

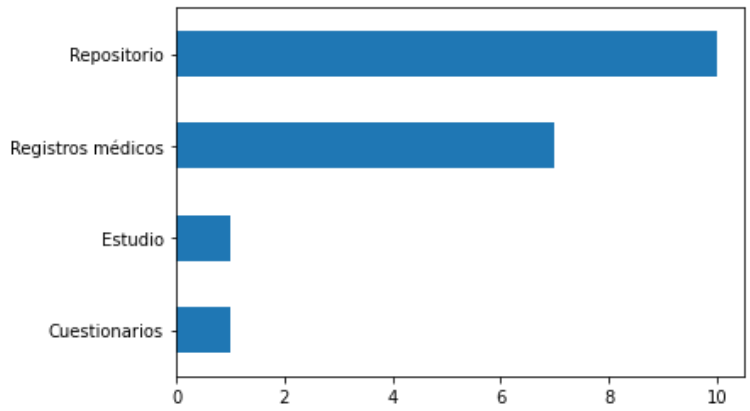

Figura 9. Fuentes de adquisición de datos de los artículos revisados. Elaboración propia a partir de los artículos revisados de Scopus.

La finalidad de los modelos desarrollados es muy variada. Se ha podido comprobar que existen múltiples temas de interés, entre los que destacan el desarrollo de sistemas de monitorización en tiempo real para detectar cambios en el estado de salud de la madre y el feto.

Figura 10. Finalidad de los modelos desarrollados en los artículos revisados. Elaboración propia a partir de los artículos revisados de Scopus.

En cuanto a temas de explicabilidad, sólo cinco evaluaron la importancia de los atributos en la predicción y estudiaron cómo se llevaban a cabo las predicciones (F. Hou *et al.*, 2020; Sterckx *et al.*, 2020; Baker, Xiang and Atkinson, 2021; Irfan, Basuki and Azhar, 2021; Y. Zhang *et al.*, 2021). Se ha podido comprobar que existe una necesidad de desarrollar modelos explicables.

A continuación, se va a comentar algunos de los artículos en más profundidad.

En el artículo presentado por Moreira et al. (Moreira *et al.*, 2021) , se propone utilizar técnicas neuro difusas (*neuro-fuzzy*) para predecir el síndrome de HELLP (del inglés ***H****emolytic anemia,* ***E****levated* ***L****iver enzyme,* ***L****ow* ***P****latelet count,* cuya traducción es “anemia hemolítica”, “elevación de enzimas hepáticas” y “trombocitopenia” respectivamente). El síndrome de HELLP es un trastorno hipertensivo grave poco frecuente, poco conocido y difícil de diagnosticar que aparece durante el embarazo o el posparto y que puede provocar la muerte de la embarazada.

Otro trastorno que puede aparecer durante el embarazo es la diabetes mellitus gestacional (en adelante, DMG). Entre las complicaciones perinatales asociadas a la DMG se incluyen trastornos hipertensivos (como preeclampsia y eclampsia), partos prematuros, distocia de hombros, muerte durante del parto del neonato, hipoglucemia neonatal, hipoglucemia, hiperbilirrubinemia y partos por cesárea. Las complicaciones posparto incluyen obesidad, diabetes y enfermedades cardiovasculares en las madres, además de una alteración en la tolerancia a la glucosa en la descendencia (Kim, 2010). Para apoyar el diagnóstico de este trastorno, Hou et al. (Fan Hou *et al.*, 2020) proponen el uso de un modelo de *LigthGBM* (*Light Gradient Boosting Machine*) para predecir si una embarazada puede padecer o no DMG.

En la literatura revisada, también se proponen varios sistemas de monitorización de embarazadas a través del uso de sensores y *wereables.* El internet de las cosas (*Internet of Things, IoT)* ofrece soluciones para una amplia gama de aplicaciones (ciudades inteligentes, la gestión residuos, seguridad, logísticas, etc.), pero la atención médica y la asistencia sanitaria representan uno de los ámbitos de aplicación más atractivas. La IoT tiene el potencial para dar lugar a muchas aplicaciones médicas, como monitorización remota de la salud, programas de fitness, enfermedades crónicas y el cuidado de ancianos. Veena y Aravindhar (Veena and Aravindhar, 2021) proponen un sistema no invasivo para monitorizar en tiempo real de la embarazada y el feto a través de sensores inalámbricos. Este sistema permitiría determinar cualquier riesgo relacionado con el embarazo, como el parto prematuro,

y poder informar a los profesionales sanitarios para que puedan tomar las medidas necesarias en caso de emergencia. Los sensores utilizados monitorizan la frecuencia cardíaca, el nivel de glucosa en sangre, la temperatura corporal y las contracciones uterinas. Estos datos son enviados a través de una aplicación móvil, y analizados posteriormente por un modelo de *machine learning*.

Igualmente, Ahmed y Kashem (Ahmed and Kashem, 2020) proponen un sistema de este estilo en el contexto de Bangladés. Bangladés es un país en vías desarrollo en el que muchas mujeres mueren por complicaciones durante el embarazo por no tener suficiente atención sanitaria, unido a otros factores como la imposibilidad de transporte o la desinformación sobre los cuidados durante este periodo. En este contexto, un sistema de monitorización continua permitiría reducir los problemas durante el embarazo ya que permitiría una monitorización continua a distancia. Este sistema contaría con una serie de sensores para recolectar datos, que serán evaluados por un algoritmo basado en un árbol de decisión para determinar el nivel de riesgo de la paciente.

Marques et al. (Marques *et al.*, 2020) también proponen un sistema similar a los comentados anteriormente, pero incluyendo además una monitorización del estado de salud del feto. El sistema propuesto posee un sensor de ultrasonido Doppler que permite recolectar la frecuencia cardiaca del feto. Así, los autores han desarrollado un sistema capaz de predecir a través de una red neuronal convolucional si la madre o el feto se encuentran en situación de riesgo.

Por otro lado, se proponen sistemas para predecir el riesgo de sufrir una hemorragia postparto (Yawei Zhang *et al.*, 2021), para identificar si va a ocurrir o no un parto prematuro (Begum, Redoy and das Anty, 2021), o incluso si la embarazada pudiera sufrir un aborto espontáneo (Tissot and Pedebos, 2021).

Por último, caben destacar un estudio que buscan evaluar los riesgos relacionados con la salud mental. Yiye Zhang et al. (Yiye Zhang *et al.*, 2021) proponen un modelo para predecir el riesgo de sufrir depresión posparto a partir del historial de salud mental, las complicaciones obstétricas sufridas, las órdenes de prescripción de medicamentos y las características demográficas de la paciente.

La IA es una herramienta para desarrollar sistemas de detección de riesgos en embarazadas con gran potencial y muy versátil. Además, es un tema de creciente interés en la comunidad científica. En la *Figura 11. Evolución en el tiempo de documentos publicados en Scopus que cumplen con los criterios de búsqueda especificados*. Elaboración propia a partir de los resultados obtenidos en Scopus se muestra la evolución en el tiempo del número de artículos publicados que cumplan con los criterios de búsqueda especificados anteriormente.

Como se puede observar, en los últimos años ha ido en aumento la publicación de artículos sobre esta temática.

Igualmente, la combinación de IA e IoT es muy interesante ya que podría permitir implementar sistemas de telemedicina para monitorizar en tiempo real el estado de salud de embarazadas en zonas rurales o en países en vías de desarrollo en los que no tienen acceso a la atención sanitaria suficiente.

Figura 11. Evolución en el tiempo de documentos publicados en Scopus que cumplen con los criterios de búsqueda especificados. Elaboración propia a partir de los resultados obtenidos en Scopus

Por último, se ha detectado que la mayoría de los artículos revisados proponen modelos que son solo evaluados a partir de datos obtenidos a partir de repositorios, y no son llevados a un entorno de producción. En ese sentido, se detecta una necesidad de comenzar a desplegar estos modelos como herramientas de apoyo para el personal clínico para que puedan ser pilotados en entornos hospitalarios reales, y así poder evaluar su redimiento real.

4 TECNOLOGÍAS UTILIZADAS

Para llevar a cabo este proyecto, se ha usado el lenguaje de programación Python 3. Python es un lenguaje de programación interpretado, orientado a objetos y de alto nivel con semántica dinámica (What is Python? Executive Summary | Python.org).

Python es ampliamente utilizado en el ámbito de *Machine Learning*. Ofrece múltiples librerías y frameworks muy potentes para facilitar el desarrollo de modelos de aprendizaje automático. Además, es un lenguaje de código abierto, y está respaldado por una gran cantidad de recursos y documentación de calidad. También posee una comunidad muy activa y amplia.

Para el desarrollo del modelo de este TFM, se han utilizado las siguientes librerías:

- **Pandas**: es una herramienta de análisis y manipulación de datos de código abierto rápida, potente, flexible y fácil de usar, construida sobre el lenguaje de programación Python (*pandas - Python Data Analysis Library*, no date). Pandas se ha utilizado para llevar a cabo la manipulación del dataset elegido. Se han cargado los datos en un *dataframe*, y se ha utilizado las funciones que ofrece Pandas para visualizar el tipo de variables, el tamaño del *dataset*, aplicar transformaciones a las columnas, etc.
- **Numpy:** es una librería que ofrece numerosas funciones matemáticas (*NumPy*, no date). Esta librería se ha utilizado para implementar funciones matemáticas como la media, el cálculo de percentiles, entre otras.

- **Scikit Learn:** es una librería de python de código abierto pata el aprendizaje automático (*scikit-learn: machine learning in Python — scikit-learn 1.0.1 documentation*, no date). Proporciona múltiples algortimos de aprendizaje supervisado y no supervisado, así como múltiples módulos para llevar a cabo el preprocesamiento de los datos y la evaluación de los modelos. Todos los algoritmos utilizados en el TFM han sido importados de Scikit Learn. Además, se ha importado *metrics,* clase que contiene múltiples métricas de evaluación; *preprocessing*, que contiene múltiples herramientas de preprocesamiento de datos; y *pipeline*, que permite automatizar el flujo de los pasos a seguir en el desarrollo del modelo de *Machine Learning.*
- **Matplotlib:** es una librería para crear visualizaciones estáticas, animadas e interactivas en Python (*Matplotlib — Visualization with Python*, no date). Se ha utilizado para realizar las gráficas del TFM.
- **Seaborn:** es una biblioteca de visualización de datos en Python basada en matplotlib. Proporciona una interfaz de alto nivel para dibujar gráficos estadísticos atractivos e informativos (Waskom, 2021). Se ha usado junto a *matplotlib* para llevar a cabo las gráficas mostradas en el TFM.
- **Scikit multilearn:** es una biblioteca con licencia distribución de software Berkeley (*Berkeley Software Distribution,* BSD) para la clasificación multietiqueta que está construida sobre el ecosistema de scikit-learn (*scikit-multilearn: Multi-Label Classification in Python — Multi-Label Classification for Python*, no date). Scikit multilearn ha sido utilizado para implementar algoritmos específicos de la clasificación *multilabel.*
- **Feature engine:** es una biblioteca de Python con múltiples transformadores para preprocesar los atributos del conjunto de datos antes de ser usados en modelos de aprendizaje automático (*Feature-engine: A Python library for Feature Engineering for Machine Learning — 1.1.2*, no date). Se ha utilizado transformadores para imputar los valores ausentes, codificar las variables categóricas yseleccionar atributos, entre otros.

Para la implementación del código, se ha utilizado *Jupyter notebook*, una aplicación web para crear y compartir documentos que contienen código, visualizaciones y texto. Es ampliamente utilizado en el ámbito de la ciencia de dato, del modelado estadístico y del aprendizaje automático (*Project Jupyter | Home*, no date).

El código desarrollado en el *notebook* se ha subido a un repositorio de Github. GitHub es una interfaz basada en la web que utiliza *git*, un software de control de versiones de código abierto(*An Introduction to GitHub – Digital.gov*, no date). Esta herramienta ha permitido por un lado llevar un control de las versiones desarrolladas del código, y por otro el acceso y visualización del código por parte de las tutoras del Trabajo Final de Máster.

5 DESARROLLO MODELO PREDICTIVO

Todo el código desarrollado en este apartado se encuentra publicado en GitHub en el siguiente enlace: https://github.com/lolagj/TFM-Modelo-Riesgos-Embarazadas.

5.1 DISEÑO DEL MODELO

Como ya se ha comentado, los datos utilizados pertenecen a un *dataset* público disponible en Mendeley Data (*Mendeley Data - Uterine arteries Doppler and sFlt-1/PlGF ratio in hypertensive disorders during pregnancy*, no date). Pertenecen a un estudio de cohorte prospectivo realizado por el Centro Médico Universitario de Liubliana, en el que se recogieron datos desde 2012 hasta 2015 de un total de 95 mujeres (V *et al.*, 2019).

Los criterios de inclusión en el estudio fueron mujeres con embarazo único con 24 semanas o más de gestación en el momento de la recogida de datos. Participaron 22 embarazadas con PE, 32 con PE y IGR, 12 con IGR y 29 con un embarazo normal (sin ningún signo de desórdenes hipertensivos durante el embarazo, hipertensión antes del embarazo, diabetes antes del embarazo o diabetes gestacional).

El modelo que se va a desarrollar tiene como finalidad pronosticar el estado de salud de la embarazada en función de los datos observados.

La aproximación elegida para el desarrollo del modelo ha sido llevar a cabo una clasificación *multilabel* en el que vamos a tener salidas para dos etiquetas: PE y IUGR. Así, vamos a contemplar que una embarazada pueda tener a la vez ambas afecciones, solo una de ellas, o no tener ninguna. Las posibles salidas de nuestro modelo serían las indicadas en al Tabla 2.

PE	IUGR	Significado
0	**0**	Población control
1	**0**	Preeclampsia
0	**1**	Retraso de crecimiento intrauterino
1	**1**	Preeclampsia con retraso de crecimiento intrauterino

Tabla 2. Salidas posibles del modelo de clasificación multilabel

Para evaluar el modelo desarrollado, vamos a utilizar las siguientes métricas:

- **Precision** (precisión*):* número de aciertos en las instancias propuestas como positivas. Como tenemos dos etiquetas distintas, vamos a indicar en el parámetro *average* de la métrica que vamos a usar la opción de *macro* (opción disponible en Scikit-Learn). Con esta opción, indicamos que queremos que se calcule el valor de *precision* para las dos etiquetas, y que se devuelva el valor medio.
- **Recall** (Exhaustividad*):* proporción de la clase positiva que fue clasificada de forma correcta. En el parámetro *average* indicaremos *macro*.
- **F1-Score** (Valor F*):* media armónica entre *precision* y *recall.* En el parámetro *average* indicaremos *macro*.
- **Accuracy** (Exactitud*):* porcentaje de casos que el modelo ha clasificado de forma correcta.
- **AUC-ROC** (Área bajo la curva ROC): La curva AUC-ROC es una medida de rendimiento que indica en qué medida el modelo es capaz de distinguir entre clases. Cuanto mayor sea el AUC, mejor será el modelo para predecir las clases 0 como 0 y las clases 1 como 1. En el parámetro *average* indicaremos *macro*.

- **Hamming Loss** (error de Hamming)*:* es una medida de rendimiento para los problemas *multilabel*. Mide la fracción de etiquetas que han sido clasificadas de forma incorrecta. Cuanto menor sea este número, mejor. Un valor ideal sería 0 (todos los datos han sido clasificados de forma correcta), frente a 1 que sería el peor valor posible (todos los datos han sido clasificados de forma incorrecta).

Además, se va a mostrar la gráfica ROC y la matriz de confusión para cada etiqueta. Se estudiará también la varianza de los modelos entre entrenamiento y validación.

Por otra parte, vamos a definir un modelo base que nos servirá para compararlo cualquier otro algoritmo de aprendizaje automático. El objetivo será conseguir mejores resultados que el modelo base definido. El modelo que vamos a utilizar es un *Dummy Classifier,* un clasificador que realiza predicciones en función de reglas simples. En nuestro caso, la estrategia de clasificación elegida va a ser *most_frenquent*, es decir, siempre se va a clasificar los datos con la etiqueta más frecuente del *dataset*. Podemos implementar un modelo *dummy* con la librería de Scikit-learn como se muestra a continuación:

```
from sklearn.dummy import DummyClassifier

clf_dummy = DummyClassifier(strategy: "most_frequent", random_state=8)
```

Figura 12. Creación de modelo base con DummyClassifier

El modelo final deberá tener un rendimiento mejor que el clasificador *dummy.*

5.2 DATOS UTILIZADOS

El *dataset* recoge datos para 21 atributos distintos. A continuación, se muestra una tabla con la información relativa a cada uno de ellos.

Nombre variable	Descripción	Tipo
Patient number	Número identificador de la paciente	Cadena de caracteres
Maternal age	Edad de la embarazada	Número entero
Parity	Número de veces que ha dado a luz a un feto con 24 semanas o más de gestación, independientemente de si nació vivo o muerto	Número entero
Pre-pregnancy weight	Peso de la embarazada antes del embarazo	Número decimal. Unidad: kilogramos
Maternal height	Altura de la embarazada	Número decimal. Unidad: metros
BMI before pregnanciy	Índice de masa corporal (IMC) antes del embarazo. Se calcula dividiendo el peso entre la altura al cuadrado	Número decimal. Unidad: kg/m^2
Gestational age at delivery	Edad gestacional en el momento del parto	Número decimal. Unidad: semanas
Weight	Peso del neonato	Número decimal. Unidad: gramos
S-Flt1	Niveles séricos de la tirosina quinasa 1 soluble similar a fms	Número decimal. Unidad: µg/L

Nombre variable		Descripción	Tipo
S-PLGF		Factor de crecimiento placentario	Número decimal. Unidad: µg/L
sFLT/PLGF		Ratio sFlt-1 y PLGF	Número decimal
Class		Clase del conjunto de datos. Representa el estado de salud de la embarazada en el momento de la recogida de datos	4 posibles valores: control, PE, IUGR, PE+IUGR
UtADoppler measures (Medidas Doppler)	Art ut. D-resistance index [RI]	Índice de resistencia de la arteria uterina derecha	Número decimal
	Art ut. L-resistance index [RI]	Índice de resistencia de la arteria uterina izquierda	Número decimal
	Mean RI	Índice de resistencia medio	Número decimal
	Art ut. D-pulsatility index [PI]	Índice de pulsatilidad de la arteria uterina derecha	Número decimal
	Art ut. L-pulsatility index [PI]	Índice de pulsatilidad de la arteria uterina izquierda	Número decimal
	Mean PI	Índice de pulsatilidad medio	Número decimal
	Art ut. D-Peak Systolic Velocity [PSV]	Pico sistólico máximo de la arteria uterina derecha	Número decimal
	Art ut. L-Peak Systolic Velocity [PSV]	Pico sistólico máximo de la arteria uterina izquierda	Número decimal
	Mean PSV	Pico sistólico máximo medio	Número decimal
	Bilateral notch	Presencia de muescas o incisuras diastólicas (*notch)*	3 posibles valores: 2= *notch* en ambas arterias, 1= *notch* en una sola, 0: no se ha detectado ningún *notch*

Tabla 3. Atributos del dataset. Cuadro elaborado a partir de los datos del dataset Mendeley Data - Uterine arteries Doppler and sFlt-1/PlGF ratio in hypertensive disorders during pregnancy

5.3 LECTURA Y CARGA DE LOS DATOS

El conjunto de datos en crudo (*raw data)* descargado desde Mendeley no puede ser cargado directamente con la librería Pandas. Si visualizamos el *raw data*, podemos comprobar que tiene una triple cabecera (*Figura 13. Captura de pantalla de los datos en crudo*).

Vamos a saltarnos las dos primeras filas del *dataset* al cargar los datos en pandas, quedándonos solo con una cabecera y evitando así el error en la lectura del archivo. De esta forma, no tendremos ningún problema durante la carga de los datos en el *notebook* de *Jupyter* (*Figura 14. Carga del conjunto de datos con Pandas*). Además, vamos a renombrar las columnas para facilitar el tratamiento del *dataframe.*

																		Infant	Cobas		
	Patient nube	l age at s	ternal wei	ternal hei	BMI											ional age at de	Parity	th weight		S-PLGF	sFLT/PLGF
	Patient nub	Materna	gnancy we	ernal high	MI [kg/m	Art ut. D-r	Art ut. D-p	Art ut. D-P	index [RI]	index [PI]	city [PSV]	Mean RI	Mean PI	Mean PSV	eral notch	l age at delive	Parity	weight [g]	S-Flt1 [µg/L]	S-PLGF [µg	sFLT/PLGF
Control	2 DM	37	58	1,70	20,07	0,43	0,56	65,3	0,49	0,6	53,9	0,46	0,58	59,6	0	37 2/7	1	3060	11511,0	102,80	111,97
Control	650387	35	46	1,64	17,10	0,4	0,54	62,5	0,67	0,73	72,9	0,535	0,635	67,7	0	36 5/7	3	2780	3146,0	752,80	4,18
Control	650394	34	64	1,75	20,90	0,41	0,6	59,7	0,53	0,62	62,3	0,47	0,61	61	0	35 1/7	1	2450	3858,0	203,20	18,99
Control	678474	31	49	1,52	21,21	0,65	0,97	61,5	0,59	0,99	68,1	0,62	0,98	64,8	0	36 3/7	2	2490	995,0	1051,00	0,95
Control	653633	31	66	1,60	25,78	0,62	0,88	58,1	0,59	0,76	59,1	0,605	0,82	58,6	0	32 1/7	1	1905	1667,0	1440,00	1,16
Control	688875	33	55	1,67	19,72	0,68	0,68	55,3	0,66	0,57	61,3	0,67	0,625	58,3	0	34	1	2180	5444,0	736,10	7,40
Control	698789	27	76	1,73	25,39	0,43	0,6	59,7	0,51	0,68	64,2	0,47	0,64	61,95	0	34 3/7	1	1910	3412,0	1410,00	2,42
Control	698795	35	53	1,63	19,95	0,71	0,83	60,5	0,77	1,38	59,1	0,74	1,105	59,8	0	37 1/7	1	2600	3403,0	723,80	4,70
Control	698794	26	55	1,73	18,38	0,54	0,65	64,7	0,55	0,67	60,2	0,545	0,66	62,45	0	39 2/7	1	3110	2989,0	1111,00	2,69
Control	698798	32	49	1,64	18,22	0,51	0,67	57,1	0,57	0,71	59,4	0,54	0,69	58,25	0	31 6/7	2	1510	2441,0	361,30	6,76
Control	711346	31	54	1,66	19,60	0,59	0,62	54,9	0,61	0,75	64,2	0,6	0,685	59,55	0	40 4/7	1	3550	9247,0	222,80	41,50
Control	711341	36	63	1,69	22,06	0,49	0,57	62,4	0,54	0,68	58,9	0,515	0,625	60,65	0	39 3/7	2	3310	1583,0	416,00	3,81
Control	721284	25	65	1,70	22,49	0,57	0,62	56,7	0,64	0,72	64,9	0,605	0,67	60,8	0	37 1/7	1	3600	3326,0	1607,00	2,07
Control	730020	22	50	1,52	21,64	0,59	0,69	54,3	0,62	0,7	66,1	0,605	0,695	60,2	0	38 4/7	1	2860	1897,0	681,80	2,78
Control	732260	38	55	1,63	20,70	0,67	0,76	57,1	0,7	0,81	59,4	0,685	0,785	58,25	0	38 5/7	3	2650	1852,0	568,00	3,26
Control	732275	34	72	1,69	25,21	0,55	0,64	52,8	0,59	0,69	59,7	0,57	0,665	56,25	0	40 6/7	1	3575	4252,0	281,30	15,12
Control	740746	26	65	1,70	22,49	0,68	0,77	54,9	0,69	0,85	60,2	0,685	0,81	57,55	0	33 6/7	1	1890	4173,0	722,30	5,78
Control	740930	32	72	1,76	23,24	0,54	0,65	53,7	0,66	0,75	59,3	0,6	0,7	56,5	0	35 5/7	2	3090	1486,0	663,30	2,24
Control	744636	26	112	1,68	39,68	0,57	0,71	64,1	0,66	0,77	64,1	0,615	0,74	64,1	0	41 1/7	1	3700	1647,0	627,40	2,63

Figura 13. Captura de pantalla de los datos en crudo

```python
data= pd.read_excel("data/Raw_data.xlsx", skiprows= lambda x: x in [0, 1])
data.columns= ["class","ID","age", "weight", "height", "BMI", "R-RI", "R-PI", "R-PSV", "L-RI",
                        "L-PI", "L-PSV", "meanRI", "meanPI", "meanPSV", "notch", "age_delivery",
                        "parity", "birth_weight", "S-Flt1", "S-PLGF", "ratio_F_P"]
```

Figura 14. Carga del conjunto de datos con Pandas

Verificamos que los datos se han cargado correctamente mostrando las 5 primeras filas del *dataset*.

```python
df.head()
```

	class	ID	age	weight	height	BMI	R-RI	R-PI	R-PSV	L-RI	...	meanRI	meanPI	meanPSV	notch	age_delivery	parity	birth_weight	S-Flt1
0	Control	2 DM	37.0	58.0	1.70	20.069204	0.43	0.56	65.3	0.49	...	0.460	0.580	59.6	0.0	37.285714	1.0	3060	11511.0
1	Control	650387	35.0	46.0	1.64	17.102915	0.40	0.54	62.5	0.67	...	0.535	0.635	67.7	0.0	36.714286	3.0	2780	3146.0
2	Control	650394	34.0	64.0	1.75	20.897959	0.41	0.60	59.7	0.53	...	0.470	0.610	61.0	0.0	35.142857	1.0	2450	3858.0
3	Control	678474	31.0	49.0	1.52	21.208449	0.65	0.97	61.5	0.59	...	0.620	0.980	64.8	0.0	36.428571	2.0	2490	995.0
4	Control	653633	31.0	66.0	1.60	25.781250	0.62	0.88	58.1	0.59	...	0.605	0.820	58.6	0.0	32.142857	1.0	1905	1667.0

Figura 15. Cinco primeras filas del dataframe

Antes de continuar con el análisis exploratorio, vamos a eliminar una serie de columnas que no vamos a utilizar en el problema de clasificación:

- *ID*: número identificador de la paciente. Es un valor de tipo cadena que no aporta información al problema de clasificación.
- *Age_delivery*: representa la edad gestacional, es decir, la semana a la que la embarazada ha dado a luz. Este atributo es un dato que conocemos a término del embarazo, por lo que no se podría utilizar para detectar riesgos **durante el embarazo**.
- *Birth_weight*: peso del bebe al nacer. Igual que con *age_delivery*, es un dato que no tenemos disponible durante el embarazo.

5.4 ANÁLISIS EXPLORATORIO

El análisis exploratorio es un proceso crítico en el que se realiza una exploración inicial de los datos con el fin de ver qué tipo de información contiene, descubrir patrones, detectar anomalías, probar hipótesis y comprobar suposiciones con la ayuda de estadísticas resumidas y gráficos (Tukey, 1977).

5.4.1 Tipo de variables

Comenzamos estudiando el tipo de variables que contiene el dataset, verificando que el tipo es el esperado. Para ver el tipo de variables usamos la función *info()* de pandas.

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 99 entries, 0 to 98
Data columns (total 19 columns):
 #   Column     Non-Null Count  Dtype  
---  ------     --------------  -----  
 0   class      95 non-null     object 
 1   age        95 non-null     float64
 2   weight     94 non-null     float64
 3   height     94 non-null     float64
 4   BMI        92 non-null     float64
 5   R-RI       95 non-null     float64
 6   R-PI       95 non-null     float64
 7   R-PSV      95 non-null     float64
 8   L-RI       95 non-null     float64
 9   L-PI       95 non-null     float64
 10  L-PSV      95 non-null     float64
 11  meanRI     99 non-null     float64
 12  meanPI     99 non-null     float64
 13  meanPSV    93 non-null     float64
 14  notch      95 non-null     float64
 15  parity     95 non-null     float64
 16  S-Flt1     99 non-null     float64
 17  S-PLGF     99 non-null     float64
 18  ratio_F_P  99 non-null     float64
dtypes: float64(18), object(1)
memory usage: 14.8+ KB
```

Figura 16. Tipo de las variables del dataset

Todas las variables tienen el tipo esperado. Tenemos en total 18 variables numéricas y una de tipo de object (correspondiente a la clase).

5.4.2 Número de observaciones y valores ausentes

El dataset contiene 99 filas y 19 columnas. Hemos podido comprobar que ninguna de estas filas se encuentra duplicadas. En cuanto a valores ausentes, hemos obtenido 63 valores ausentes en total repartidos entre varias columnas.

En la *Figura 17. Número de valores nulos por columna*, podemos observar que la columna *class* tenemos 4 valores ausentes. En la documentación del dataset, viene indicado que el número total de instancias debería ser 95 (22 embarazadas con PE, 32 con PE y IGR, 12 con IGR y 29 con un embarazo normal).

```
class          4
age            4
weight         5
height         5
BMI            7
R-RI           4
R-PI           4
R-PSV          4
L-RI           4
L-PI           4
L-PSV          4
meanRI         0
meanPI         0
meanPSV        6
notch          4
parity         4
S-Flt1         0
S-PLGF         0
ratio_F_P      0
dtype: int64
```

Figura 17. Número de valores nulos por columna

Visualizamos las filas con el valor de la clase ausente, y obtenemos lo siguiente:

```
data[data['class'].isnull()]
```

	class	age	weight	height	BMI	R-RI	R-PI	R-PSV	L-RI	L-PI	L-PSV	meanRI	meanPI	meanPSV	notch	parity	S-Flt1	S-PLGF	ratio_F_P
29	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.568103	0.698448	NaN	NaN	NaN	3498.620690	648.940690	13.197129
62	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.708125	1.371719	NaN	NaN	NaN	19948.625000	124.384375	407.334090
75	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.724583	1.395833	NaN	NaN	NaN	8872.083333	120.595000	230.745479
98	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	0.635000	0.999318	NaN	NaN	NaN	18343.545455	205.533182	314.132643

Figura 18. Filas con valores nulos en el atibuto 'class'

Estas filas contienen la mayoría de los valores nulos detectados. En estas filas, se han recogido los valores medios de las columnas meanRI, meanPI, S-Flt1, S-PLGF y ratio_F_P para cada una de las clases del dataset (la fila 29 corresponde a los valores medios de la clase *control,* la 62 a los de *IUGR+PE*, la 74 a los de *IUGR* y la 98 a los de *PE)*. No aportan ninguna información nueva a nuestro problema de clasificación, ya que se trata de una combinación de valores de otras filas. Por lo tanto, vamos a eliminarlas del *dataset*.

El resto de los valores nulos se encuentran repartidos en las columnas *BMI, weight, height* y *meanPSV*. Vamos a visualizar las filas que los contine:

age	weight	height	BMI	R-RI	R-PI	R-PSV	L-RI	L-PI	L-PSV	meanRI	meanPI	meanPSV	bilateralNotch	parity	S-Flt1	S-PLGF	ratio_F_P	class
43	66.0	1.52	NaN	0.63	0.60	67.4	0.59	0.67	62.9	0.610	0.635	65.15	0	1	27390.0	153.60	178.320312	IUGR_PE
36	52.0	1.64	NaN	0.67	0.81	57.9	0.69	0.89	61.4	0.680	0.850	59.65	1	1	29805.0	79.73	373.824157	IUGR_PE
26	NaN	NaN	NaN	0.84	2.02	68.8	0.83	2.28	59.4	0.835	2.150	64.10	1	1	10015.0	42.64	234.873358	IUGR
30	53.0	1.57	21.501886	0.49	0.52	60.4	0.51	0.56	58.1	0.500	0.540	NaN	0	1	10535.0	34.97	301.258221	PE
26	74.0	1.65	27.180900	0.69	0.62	54.9	0.73	0.71	60.5	0.710	0.665	NaN	2	1	12154.0	15.84	767.297980	PE

Figura 19. Filas con valores ausentes

En la fila 1º y 2º, nos encontramos con que falta el valor de BMI. El BMI representa el índice de masa corporal, y se calcula con la siguiente fórmula:

$$\boldsymbol{BMI} = \frac{peso\ (kg)}{altura\ (m)^2}$$

En esa fila conocemos tanto el valor del peso (*weight*) como el de la altura (*height*), podríamos imputar este valor ausente a través de la fórmula descrita.

En cuanto a la fila 3º, el valor de peso y altura también está ausente. En este caso, podríamos optar por usar la media o la mediana de la columna correspondiente para imputar el valor ausente.

Por último, en la fila 4º y 5º vemos que falta el valor de *meanPSV*. Como en ambas filas tenemos los valores de L-PSV y R-PSV, podríamos sustituir el valor ausente por la media de estos dos valores.

5.4.3 Clase

La variable *class* puede tener 4 valores distintos: IUGR_PE, Control, PE, IUGR.

Valor	Descripción	Número de pacientes
IUGR_PE	Preeclampsia con retraso de crecimiento intrauterino	32
Control	No padece ninguna de las dos enfermedades	29
PE	Preeclampsia	22
IUGR	Retraso de crecimiento intrauterino	12

Figura 20. Valores de la clase del dataset.

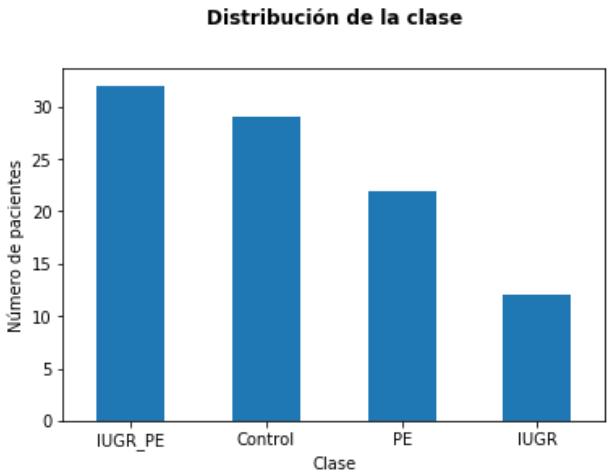

Figura 21. Distribución inicial de la clase

Las distintas clases no se presentan en la misma proporción, sino que están desbalanceadas.

Como se ha comentado anteriormente, vamos a considerar el problema como una clasificación *multilabel*, teniendo dos posibles etiquetas: PE e IUGR. La codificación de cada clase muestra a continuación:

	class	PE	IUGR
1	Control	0	0
40	IUGR+PE	1	1
63	IUGR	0	1
76	PE	1	0

Figura 22. Transformación de la columna class en dos columnas: PE e IUGR

Así, si por ejemplo obtuviéramos a la salida del clasificador el valor [0,0], sabríamos que la embarazada no padece ni preeclampsia ni retraso del crecimiento intrauterino.

La nueva distribución de la clase sería la siguiente:

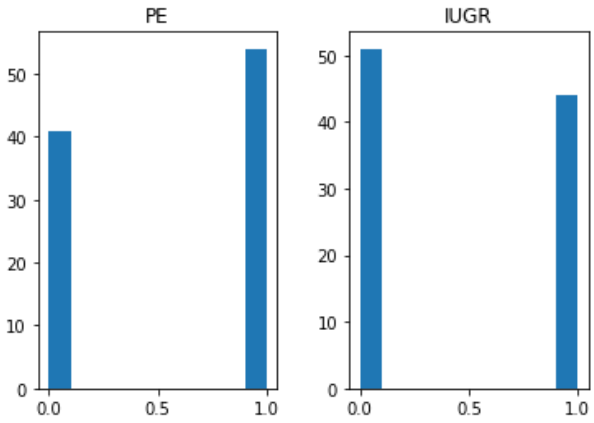

Figura 23. Distribución de las etiquetas

5.4.4 Variables numéricas

Las variables numéricas del *dataset* presentan distintos rangos y magnitudes. Por ejemplo, en la *Figura 24. Estadísticas descriptivas de un subconjunto de variables del dataset*, podemos observar que Figura 24. Estadísticas descriptivas de un subconjunto de variables del datasetla columna L-RI tiene un valor medio de 0,655895, mientras que el de la columna de S-Plt1 es 13156,2.

Deberemos estandarizar los datos en la fase de preprocesamiento para que estén en una escala similar.

	L-RI	height	S-Flt1	S-PLGF	ratio_F_P
count	95.000000	94.000000	95.000000	95.000000	95.000000
mean	0.655895	1.655638	13156.200000	302.825789	243.129174
std	0.108437	0.062554	13824.935398	383.390465	351.790952
min	0.430000	1.500000	995.000000	10.050000	0.946717
25%	0.590000	1.620000	3623.500000	54.020000	7.367625
50%	0.670000	1.660000	9530.000000	105.900000	119.728305
75%	0.755000	1.700000	17947.000000	448.550000	362.884861
max	0.830000	1.800000	74283.000000	1607.000000	2297.421203

Figura 24. Estadísticas descriptivas de un subconjunto de variables del dataset

Pasemos a estudiar la distribución de las variables.

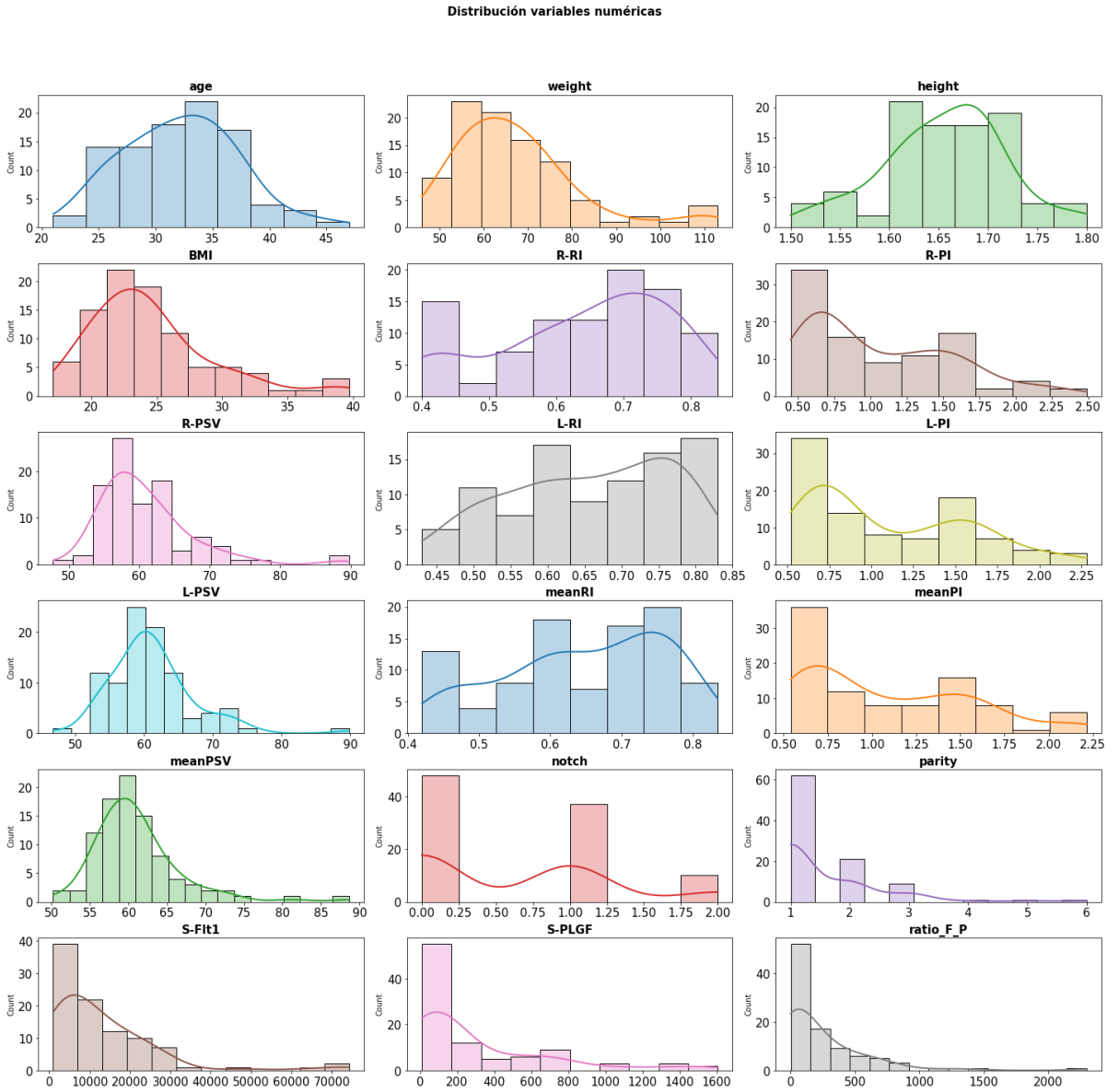

Figura 25. Ditribución de las variables numéricas del conjunto de datos

Si nos fijamos en las variables *parity* y *notch*, podemos observar que toman muy pocos valores y además se concentran en torno a uno o dos valores. A partir de ahora, vamos a considerar estas dos variables como categóricas.

Respecto al resto de distribuciones numéricas, vamos a ver de qué tipo son: normales o asimétricas (*skewed)*. Podríamos determinar cuáles no presentan una forma normal mirando las gráficas (por ejemplo, *ratio_F_P* es claramente asimétrica positiva), pero no sería una medida objetiva. Por eso, vamos a utilizar la prueba de Shapiro-Wilks para determinar qué variables no tienen una distribución normal. La prueba de Shapiro-Wilks plantea la hipótesis nula que los datos evaluados son una muestra de una distribución normal. Un p-valor inferior a 0,05 indicaría que la hipótesis nula no se cumple, y por tanto la distribución no es normal.

Aplicamos la prueba de Shapiro-Wilks a todas las variables numéricas del dataset, y nos quedamos solo con los p-valor inferiores a 0,05.

	Feature	p-value
15	ratio_F_P	7.573866e-13
13	S-Flt1	6.484138e-12
14	S-PLGF	8.077696e-12
6	R-PSV	5.040247e-08
8	L-PI	4.421298e-07
11	meanPI	9.603907e-07
5	R-PI	1.621203e-06
9	L-PSV	9.295508e-06
4	R-RI	3.834041e-05
10	meanRI	2.771516e-04
7	L-RI	5.187430e-04

Figura 26. Variables con p-valor inferior a 0,05 en la prueba de Shapiro-Wilks

Las distribuciones quedarían agrupadas así:

- **Distribución normal:** age, weight, height, BMI y meanPSV.
- **Distribución asimétrica**: R-RI, R-PI, R-PSV, L-RI, L-PI, L-PSV, meanRI, meanPI, S-Flt1, S-PLGF y ratio_F_P'.

En la fase de preprocesamiento, vamos a aplicar una transformación logarítmica en las variables numéricas con una distribución asimétrica. Reemplazar los datos con su transformación logarítmica puede ayudar a disminuir su asimetría (*skewedness*)(Kuhn and Johnson, 2013).

Según la literatura consultada, las medidas obtenidas a partir de la ecografía Doppler y los valores de los biomarcadores sFlt-1 y PlGF eran relevantes en la detección precoz de la PE y IRG. Veamos como se distribuyen los valores de estos atributos en función de las clases.

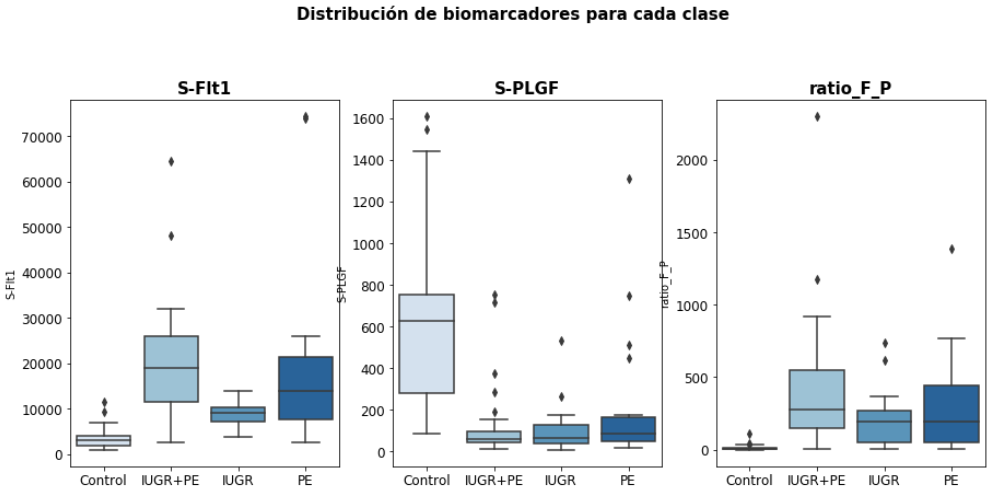

Figura 27. Distribuci de biomarcadores para cada clase (Control, IUGR+PE, IUGR, PE)

Para los biomarcadores, vemos una clara diferencia entre el grupo de control y los grupos con alguna disfunción de la placenta. En concreto, vemos que, para el grupo de control, el valor de la sFlt-1 es mucho más bajo que en el resto de grupo, mientras que el valor PlGF es mucho mayor. Entre los otros grupos que padecen alguna enfermedad, también podemos encontrar diferencias reseñables en el valor de sFlt-1 (en el caso de PlGF si tienen una distribución similar). A simple vista, parece que van a ser unos parámetros influyentes en la clasificación de los datos.

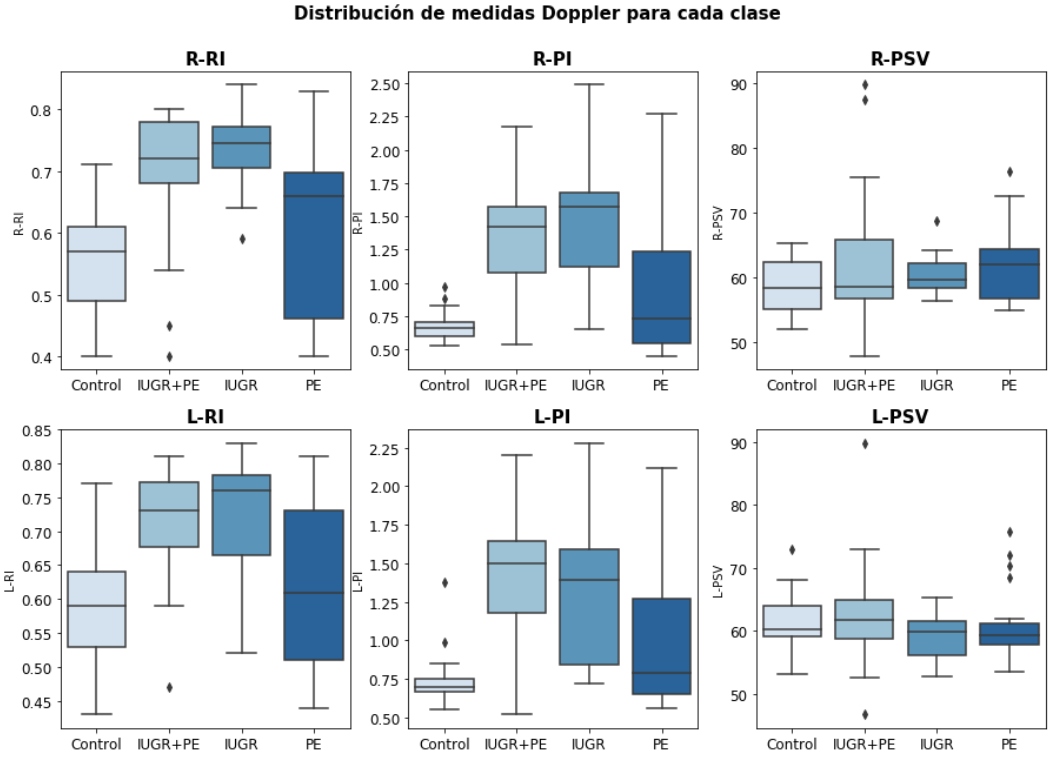

Figura 28. Distribuciones de las medidas Doppler para cada clase (Control, IUGR+PE, IUGR, PE)

En las medidas Doppler, también podemos apreciar diferencias considerables entre las distintas clases para los valores correspondientes al índice de pulsatilidad de la arteria (derecha e izquierda) e índice de resistencia de la arteria (derecha e izquierda). Para el valor de PSV no se aprecian tantas diferencias.

Sin embargo, también cabe destacar que entre la clase *Control* y *PE*, existe una proporción de datos importante que comparten valores similares para PI y RI, y lo mismo ocurre con IUGR e IUGR+PE. A simple vista, podríamos pensar que estos valores serán influyentes para diferenciar las clases IUGR e IUGR+PE de PE y Control.

El atributo *notch* también se obtiene a partir de la ecografía Doppler. Al considerar esta variable categórica, estudiaremos su relación con la clase en el apartado 5.4.7. Variables categóricas.

5.4.5 Correlación entre las variables numéricas

Algunos modelos de *machine learning* se pueden ver perjudicados por la presencia de variables altamente correlacionadas. Por eso, vamos a estudiar la correlación entre las variables numéricas del dataset. Vamos a utilizar el coeficiente de Pearson como medida de correlación. A continuación, se muestra la matriz de correlación de los atributos numéricos:

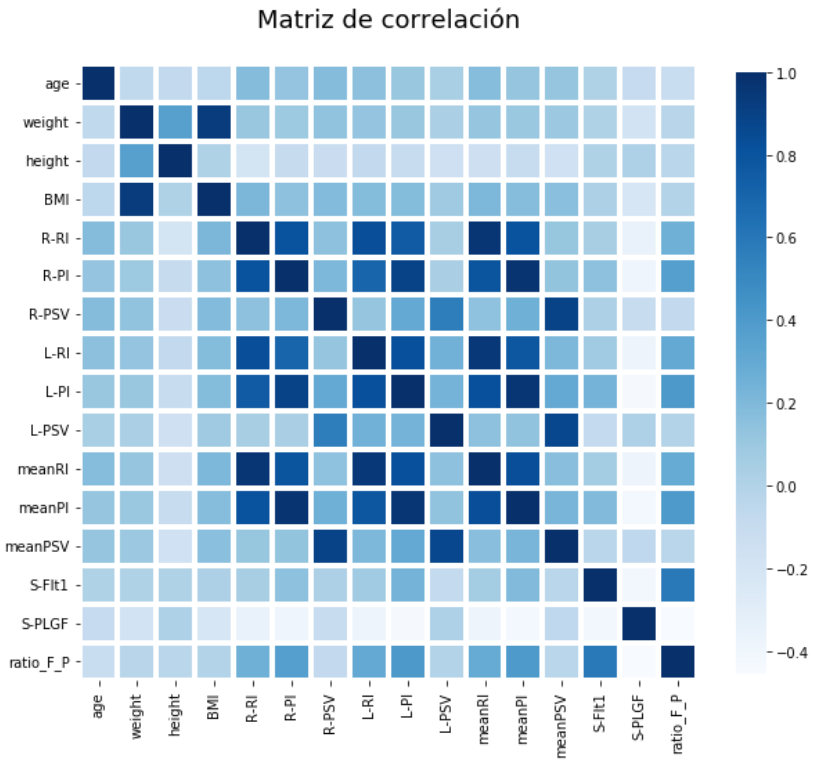

Figura 29. Matriz de correlación de variables numéricas

Existen variables altamente correladas en el dataset. Vamos a crear un *dataframe* con las variables relacionadas y el valor de correlación de *Pearson* correspondiente, para poder estudiar de forma más cómoda qué variables tienen un valor de correlación mayor.

		Correlación
R-PI	**meanPI**	0.974681
L-PI	**meanPI**	0.971184
R-RI	**meanRI**	0.963732
L-RI	**meanRI**	0.950348
weight	**BMI**	0.931485
R-PSV	**meanPSV**	0.896657
R-PI	**L-PI**	0.893303
L-PSV	**meanPSV**	0.873123

Figura 30. Variables con valor de correlación de Pearson superior a 0,85

Las variables *meanPI, meanRI, meanPSV* y *BMI* contiene información sobre otras columnas (por ejemplo, *meanPI* contiene el valor medio de *L-PI* y *R-PI).* Es por eso que están tan relacionadas con esas columnas.

Sería conveniente eliminarlas en el entrenamiento del modelo ya que son columnas que no aportan información nueva, y que incluso podrían introducir ruido.

5.4.6 Detección de valores anómalos

Vamos a visualizar si existe algún valor anómalo (*outliers)* en las distribuciones numéricas. Lo vamos a hacer de forma gráfica usando el diagrama de caja.

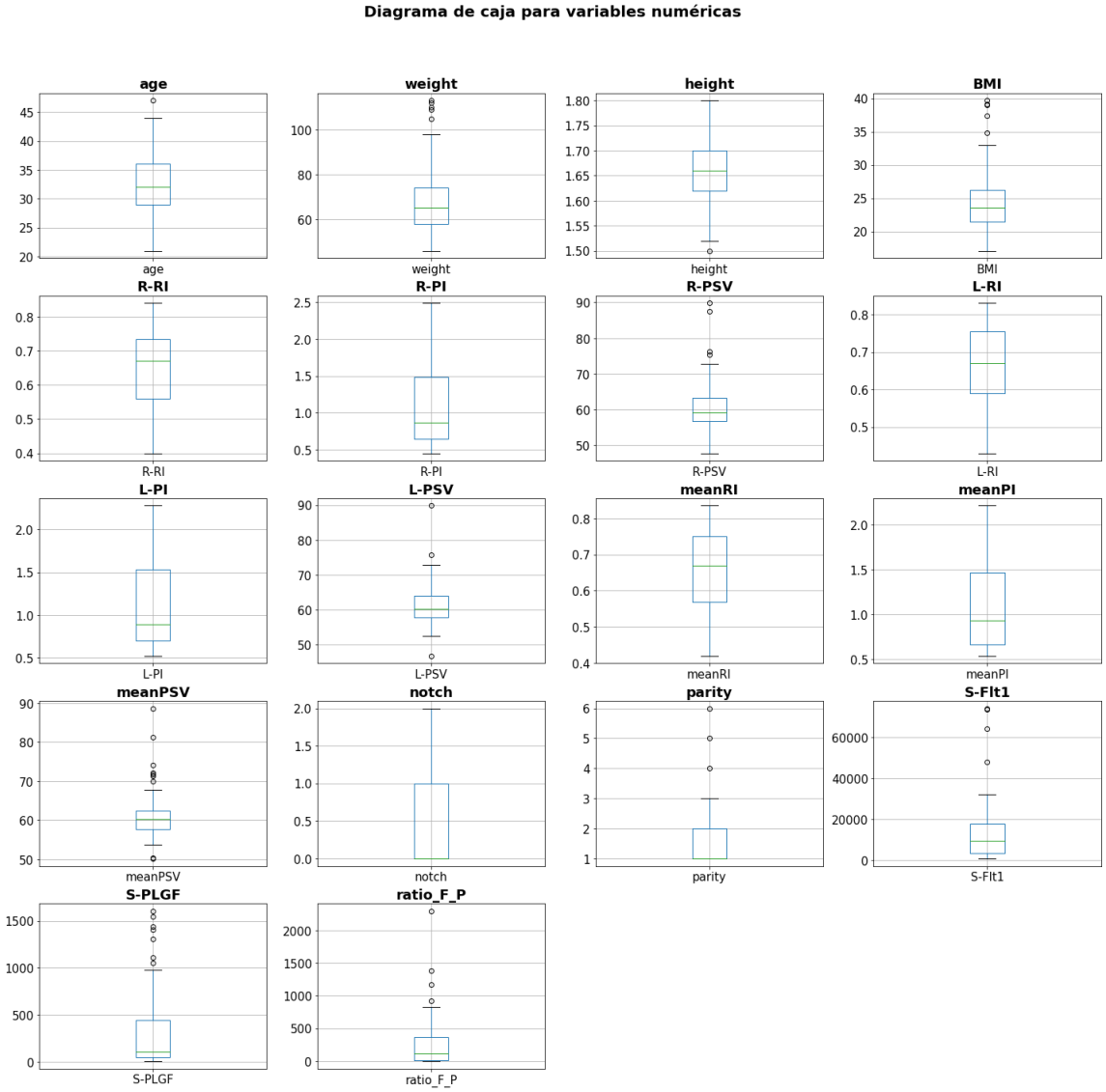

Figura 31. Gráfica de caja. Visualización de valores anómalos de variables numéricas

Vemos que algunas variables poseen valores anómalos que se salen del rango de la distribución. No vamos a eliminar estos valores ya que estos valores podrían ayudarnos a detectar la presencia o ausencia de PE e IUGR.

5.4.7 Variables categóricas

En el apartado 5.4.7 Variables numéricas hemos indicado que vamos a considerar *notch* y *parity* como categóricas. Por ello, vamos a analizar en este apartado.

	bilateralNotch	parity
count	95	95
unique	3	6
top	0	1
freq	48	62

Figura 32. Estadísticas descriptivas para variables categóricas

La variable *parity* puede tomar seis posibles valores: 1, 2, 3, 4, 5 y 6. Esta variable representa el número de veces que se ha dado a luz.

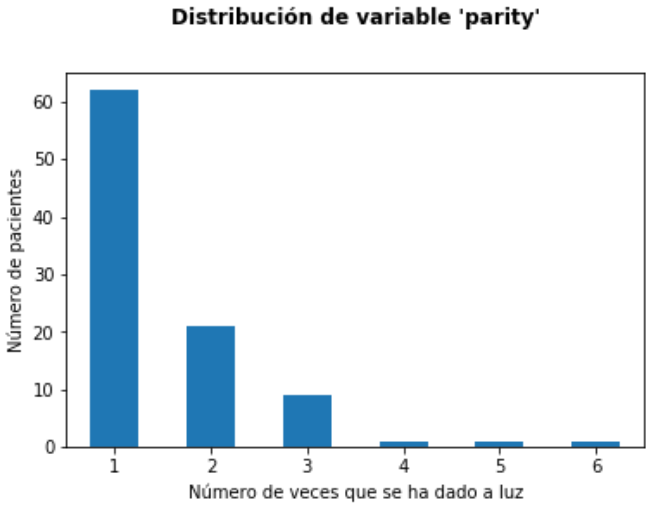

Figura 33. Distribución de la variable parity

Como hemos indicado anteriormente, el valor 1 representa a las madres primerizas, es decir, las embarazadas que dieron a luz durante el desarrollo del estudio. Esta variable se encuentra muy desbalanceada. La gran mayoría de las embarazadas del estudio son primerizas, contando con muy poca representación de embarazadas que han tenido más de un parto. Por ejemplo, para los valores de *parity* 4, 5 y 6 solo poseemos una instancia.

	parity
1.0	62
2.0	21
3.0	9
4.0	1
5.0	1
6.0	1

Figura 34. Frecuencia absoluta de la variable parity

Los valores de *parity* 4.0,5.0 y 6.0 son los tres *outliers* detectados en el apartado 6.5.6. Detección de outliers. Al tener un solo ejemplo de estas etiquetas, podríamos tener problema al realizar la codificación de la clase, ya que habría datos que estarían presentes en el conjunto de *train* pero en el de *test* no, y viceversa.

En este caso, sí vamos a tratar los valores anómalos. Vamos a sustituir los valores *outliers* por el valor del del percentil 90, que para *parity* sería 3.0.

De esta forma, la distribución de *parity* quedaría así:

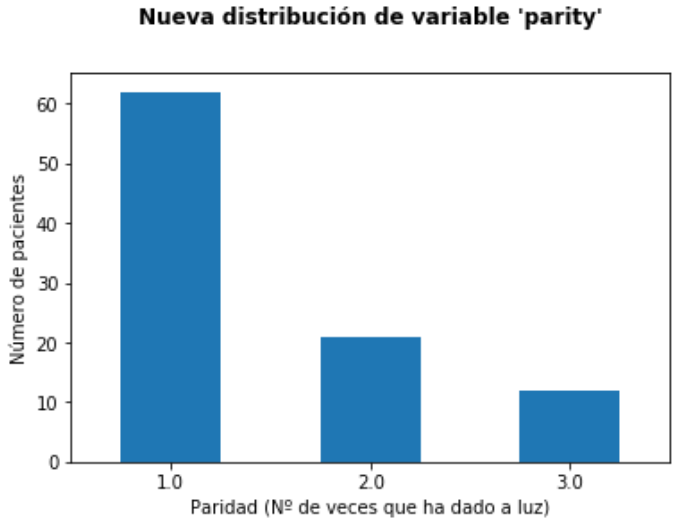

Figura 35. Nueva distribución de parity tras tratar valores anómalos

En cuanto a *notc,* puede tomar tres posibles valores: 0 si no existe una muesca diastólica, 1 si aparece en una arteria, y 2 si aparece en ambas.

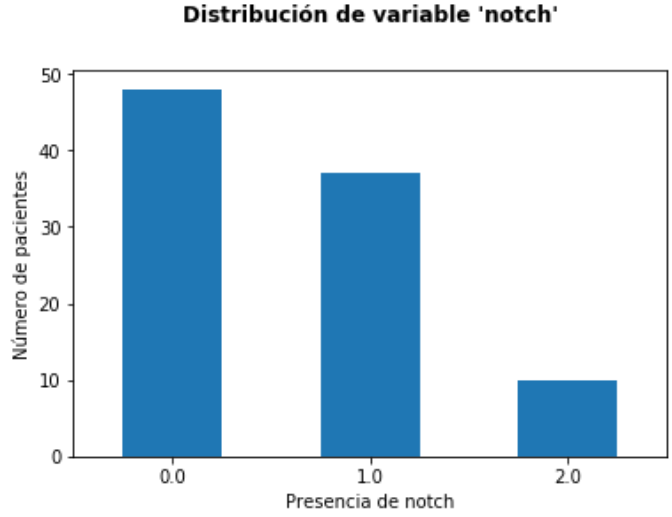

Figura 36. Distribución de variable Notch

Vemos que una minoría de embarazadas tiene un *notch* bilateral. La *Figura 37. Tipos de notch según clase* muestra la distribución por clases de los tres posibles valores de *notch.* Podemos ver que en la población de control no se encontró ningún *notch* en las arterias. Además, podemos observar que el *notch* unilateral se presenta en mayor proporción en las embarazadas con IUGR. Podemos intuir que este parámetro va a ser influyente en la predicción de las clases.

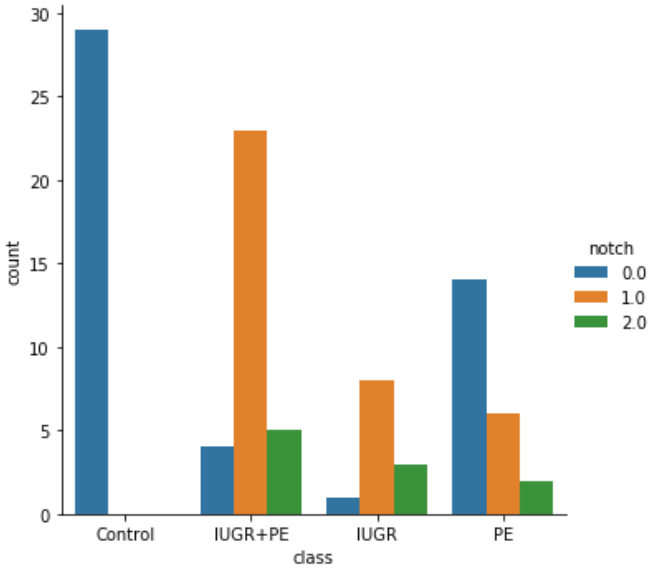

Figura 37. Tipos de notch según clase

5.5 PREPROCESAMIENTO

Antes de comenzar la fase de preprocesamiento, dividimos el *dataset* en conjunto de *train* y *test*, asignando un 80% del conjunto a entrenamiento y un 20% a validación.

Todas las transformaciones que se van a describir en los próximos apartados se realizarán únicamente sobre el conjunto de *train*, y la agruparemos en un mismo *pipeline.*

Los pasos que vamos a seguir en el preprocesamiento son:

Figura 38. Pipeline de preprocesamiento. Elaboración propia.

5.5.1 Selección de atributos

El primer paso del pipeline es eliminar las variables con una alta correlación, que corresponden con las columnas que son transformaciones de otras: *BMI, meanRI, meanPI* y *meanPSV.*

```
p0=('drop_features', DropFeatures(features_to_drop=["BMI",'meanRI','meanPI','meanPSV']))
```

Figura 39. Primer paso del pipeline de preprocesamiento: eliminación de atributos con correlación alta

5.5.2 Imputación de valores ausentes

Tras eliminar las columnas de *meanPSV* y *BMI,* tan sólo nos quedan 2 valores ausentes: uno en la columna de *height*, y el otro en *weight.* Vamos a imputar estos valores ausentes con el valor medio de la columna correspondiente.

```
p1=('impute_nan_values', MeanMedianImputer(
        imputation_method='mean', variables=['weight', 'height']))
```

Figura 40. Segundo paso del pipeline preprocesamiento: Imputación valores ausentes con valor medio

5.5.3 Transformación logarítmica

Vamos a aplicar la transformación logarítmica las *skewed_variables.*

```
sk =['R-RI', 'R-PI', 'R-PSV', 'L-RI', 'L-PI', 'L-PSV', 'S-Flt1', 'S-PLGF', 'ratio_F_P']
p2= ('log_transf',  LogTransformer(variables=sk))
```

Figura 41. Paso 3 de pipeline preprocesamiento: Transformación logarítmica de variables con distribución asimétrica

5.5.4 Escalado de variables numéricas

La estrategia de normalización elegida es la transformación z-score. Esta transformación consiste en dividir cada predictor entre su desviación típica después de haber sido centrado

```
num_vars=['age', 'weight','height','R-RI','R-PI','R-PSV','L-RI','L-PI','L-PSV',
          'S-Flt1', 'S-PLGF','ratio_F_P']
p3=('normalization', SklearnTransformerWrapper(transformer=StandardScaler(), variables=num_vars))
```

Figura 42. Paso 4 de pipeline de preprocesamiento: escalado de variables numéricas

5.5.5 Codificación de variables categóricas

Vamos a utilizar la librería de *scik-learn* OneHotEncoder. Esta es una técnica de codificación que consiste en crear tantas columnas como valores distintos contenga la columna, y atribuir el valor 1 a la categoría a la que corresponda el dato, y el 0 al resto de ellas.

Antes de hacer la codificación de *parity*, imputamos los valores anómalos con el valor de *parity* 3.0 (valor correspondiente al percentil 90).

```
X_train['parity'] = np.where(X_train['parity']>3.0, 3.0, X_train['parity'])
X_train['parity'].unique()
```

Figura 43. Imputación de valores outliers de atributo parity

```
p4= ('one_hot_encoding_parity_notch',OneHotEncoder(variables=['notch']))
p5= ('one_hot_encoding_parity_parity',OneHotEncoder(variables=['parity']))
```

Figura 44. Paso 4 y 5 del pipeline de preprocesamiento: Codificación de variables categóricas (parity y notch)

5.5.6 Pipeline de preprocesamiento final

Finalmente, agrupamos todas las transformaciones en un único *pipeline* llamado *prec_pipe.* Este *pipeline* lo utilizaremos para preparar el conjunto de datos *train* para entrenar el modelo, y posteriormente al conjunto de *test* también para poder evaluar los modelo desarrollados.

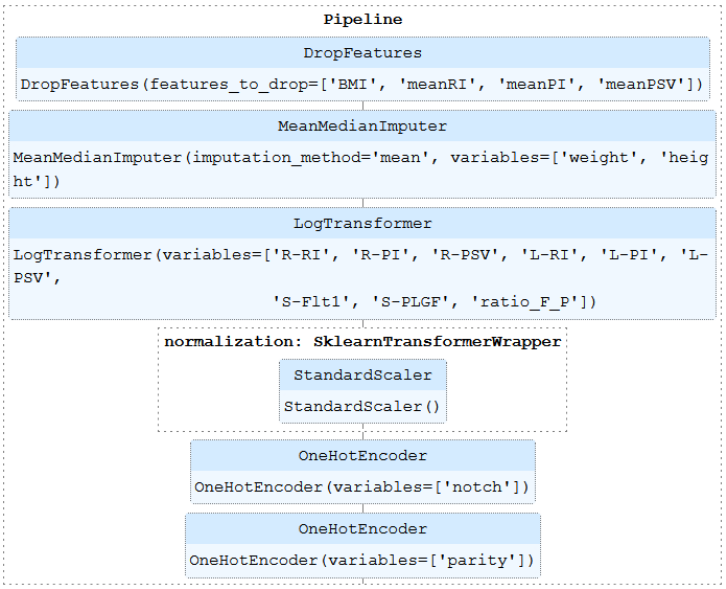

Figura 45. Diagrama del pipeline final de preprocesamiento

Aplicamos el *pipeline* al conjunto de entrenamiento *X_train*, obteniendo el siguiente resultado:

age	weight	height	R-RI	R-PI	R-PSV	L-RI	L-PI	L-PSV	S-Flt1	S-PLGF	ratio_F_P	notch_0.0	notch_1.0	notch_2.0	parity_1.0	parity_3.0	parity_2.0
2.255062	-0.161080	-2.317489	0.023143	-1.020398	1.019268	-0.499070	-0.996980	0.326403	1.157708	0.172504	0.436888	1	0	0	1	0	0
0.843659	-0.689212	-1.805776	0.688062	0.487213	3.526655	1.036215	1.034865	4.067687	-0.062667	0.049716	-0.060650	0	1	0	1	0	0
0.238771	-0.689212	0.752786	0.562685	1.142014	-0.812464	0.820523	0.946845	-0.065036	0.746317	-0.437984	0.625723	0	1	0	0	1	0
-1.172632	-0.227097	0.752786	0.367836	-0.496479	-0.951447	0.361972	-0.445493	-0.134626	-0.793046	1.378411	-1.236930	1	0	0	1	0	0
-0.769374	-0.755229	0.752786	-0.589770	-0.820227	-0.374170	-0.499070	-0.962645	0.590356	-1.515424	0.509032	-1.032282	1	0	0	1	0	0
0.037142	-0.557179	-0.952922	-1.495455	-1.091599	-2.281909	-1.749586	-1.584366	-1.552757	1.180952	-0.494031	0.865434	1	0	0	1	0	0
-0.366116	-0.821245	0.752786	1.101332	1.679527	-0.624586	0.893365	1.549136	-1.296209	1.150110	-0.127526	0.621289	0	1	0	1	0	0
0.238771	-0.227097	0.411644	1.101332	1.679527	0.294141	1.243766	1.758465	0.225686	0.660022	-1.668165	1.355848	0	1	0	0	0	1
-1.172632	-0.689212	0.752786	-2.027045	-1.241677	0.294141	0.200212	0.592080	1.876791	1.741849	-0.257683	0.981386	0	0	1	1	0	0
-0.567745	-0.029047	-0.952922	1.044560	0.945590	-0.194593	0.893365	1.077652	0.242539	0.063140	-1.197771	0.780161	0	1	0	1	0	0

Figura 46. Conjunto de X_train preprocesado

5.5.7 Codificación de la clase

Hemos definido dos funciones distintas: una para codificar la clase (*class_encode(y_train)),* y otra para decodificarla y obtener los valores originales (*class_decode(y_pred)*). A continuación, se muestra cómo quedaría clase:

```
y_train_enc= class_encode(y_train)
y_train_enc.head()
```

	PE	IUGR
59	1	1
35	1	1
40	1	1
16	0	0
19	0	0

Figura 47. Codificación de la clase del conjunto de entrenamiento

5.6 ENTRENAMIENTO DE MODELOS DE CLASIFICACIÓN MULTIETIQUETA

Se han seguido dos enfoques principales para llevar a cabo el aprendizaje de datos multietiqueta: uno basado en la transformación de los datos y otro en la adaptación de los modelos. El primero se basa en técnicas de transformación que, aplicadas a los datos multietiqueta, son capaces de producir uno o más conjuntos de datos binarios o multiclase, que ya sí podrían ser procesados con clasificadores tradicionales. El segundo tiene como objetivo adaptar los algoritmos de clasificación existentes para que puedan trabajar directamente con datos multietiqueta, produciendo varias salidas en lugar de una sola. (Herrera *et al.*, 2016). Existe otra aproximación basada en *ensembles*, que consiste en utilizar conjuntos de clasificadores. Un ejemplo de *ensemble* aplicado a clasificación multietiqueta es el Conjunto de Cadena de clasificadores (*Ensemble of Classifier Chains, ECC).*

En este TFM, solo aplicaremos el enfoque basado en la transformación de los datos, y el de adaptación de modelos. A modo resumen, se van a aplicar las siguientes técnicas:

- Adaptación de modelos: vamos a utilizar técnicas que pueden adaptarse a problemas de clasificación *multilabel (adaptation approach).* En Scikit-Learn tenemos disponibles los siguientes modelos que soportan clasificación multietiqueta:
    - Decision Tree Classifier
    - Extra Tree Classifier
    - Kneighbors Classifier
    - Random Forest Classifier

- Transformación del problema de clasificación: vamos a seguir algunos de los métodos propuesto en la literatura:
    - *Binary Relevance:* realiza una clasificación por etiqueta. Se transforma un problema de n etiquetas en n problemas de clasificación binaria separados por etiquetas, y se aplica sobre ellos el mismo clasificador especificado. El resultado de la predicción es la unión de todos los clasificadores. En nuestro caso, como tenemos dos etiquetas, tendríamos dos clasificadores: uno para PE y otro para IUGR. La desventaja que presenta esta técnica es que no tiene en cuenta la posible correlación entre las etiquetas. En la librería Scikit-multilearn tenemos disponible una función para aplicar esta técnica.

- *Label Powerset:* se transforma el problema multietiqueta en un problema multiclase con un clasificador multiclase que se entrena en todas las combinaciones de etiquetas únicas encontradas en los datos de entrenamiento. Esto sería como volver a tener el problema multiclase original.

5.6.1 Clasificadores basados en la adaptación del modelo

Vamos a utilizar un método disponible en Scikit-learn llamado *GridSearchCV* para ajustar con los mejores hiperparámetros los modelos de los algoritmos adaptados. Vamos a crear una función llamada *evalua_modelo.* Esta función va a ajustar los modelos con los hiperparámetros especificados con *GridSearchCV,* y va a ir guardando en un *dataframe* el mejor resultado obtenido para la métrica indicada, y con qué valores se ha conseguido. Como métrica de evaluación, hemos indicado el valor ROC_AUC. Esta decisión se ha tomado en base a la necesidad de obtener el modelo que clasifique mejor cada una de las clases.

A continuación, se recoge los modelos y parámetros evaluados.

Modelo	Parámetros	Valores
Decision Tree Classifier	criterion	["gini", 'entropy']
	splitter	['best', 'random']
Extra trees	criterion	["gini", 'entropy']
	n_estimators	[100,200,300]
KNeighbotsClassifier	n_neighbors	[1,2,3,4,5,6,7,8,9,10]
	weights	['uniform', 'distance']
	algorithm	['auto', 'ball_tree', 'kd_tree', 'brute']
RandomForestClassifier	criterion	["gini", 'entropy']
	max_features	['auto', 'sqrt', 'log2']

Tabla 4. Algoritmos e hiperparámetros evaluados. Elaboración propia

GridSearchCV utiliza la validación cruzada para seleccionar al mejor modelo.

Tras evaluarlos, estos son los resultados preliminares que obtenemos sobre el conjunto de *train*.

	Parametros	modelo	Roc_auc
DecisionTreeClassifier	{'criterion': 'entropy', 'random_state': 10, '...	DecisionTreeClassifier(criterion='entropy', ra...	0.750476
DecisionTreeClassifier_sin_ajustar	{'random_state': 10}	DecisionTreeClassifier(random_state=10)	0.728690
ExtraTreesClassifier	{'criterion': 'entropy', 'n_estimators': 100, ...	(ExtraTreeClassifier(criterion='entropy', rand...	0.825302
ExtraTreesClassifier_Sin_ajustar	{'random_state': 10}	(ExtraTreeClassifier(random_state=1165313289),...	0.811677
KNeighborsClassifier	{'algorithm': 'auto', 'n_neighbors': 7, 'weigh...	KNeighborsClassifier(n_neighbors=7, weights='d...	0.794977
KNeighborsClassifier_sin_ajustar	{}	KNeighborsClassifier()	0.786009
RandomForestClassifier	{'criterion': 'entropy', 'max_features': 'auto...	(DecisionTreeClassifier(criterion='entropy', m...	0.833569
RandomForestClassifier_Sin_ajustar	{'random_state': 10}	(DecisionTreeClassifier(max_features='auto', r...	0.808469

Figura 48. Mejor combinación de hiperparámetros obtenida, junto al valor de ROC-AUC

Vemos que, con el ajuste, obtenemos resultados ligeramente mejores que sin el ajuste.

5.6.2 Clasificadores basados en la transformación del problema

La relevancia binaria (*binary relevance)* es una de las soluciones más simples al problema de clasificación multietiqueta. Funciona descomponiendo la tarea de clasificación en una serie de tareas de clasificación binaria independiente. El inconveniente de esta técnica es que no tiene en cuenta las posibles correlaciones entre las distintas etiquetas, al tratar el problema con clasificadores independientes. Debido a esto, han ido surgiendo extensiones como *Classifiers Chain* (Herrera *et al.*, 2016).

En este TFM, sólo vamos a probar la versión más simple. En Scikit-multilearn, tenemos disponible una clase para aplicar la técnica de *Binary relevance*. Tenemos que pasarle como parámetro de entrada el estimador que queramos utilizar para llevar a cabo la clasificación binaria. Este estimador será utilizado para toda la predicción de todas las etiquetas.

Los algoritmos que vamos a probar van a ser:

- Gaussian Naive Bayes.
- Randomn Forest Classifier
- Support Vector Machine
- KNeighbors Classifier
- Decision Tree Classifier

```
BR_GNB = BinaryRelevance(GaussianNB())
BR_GNB.fit(X_train_prec, y_train_enc)

BR_RF = BinaryRelevance(RandomForestClassifier(random_state=8))
BR_RF.fit(X_train_prec, y_train_enc)

BR_SVC = BinaryRelevance(SVC(random_state=8))
BR_SVC.fit(X_train_prec, y_train_enc)


BR_KN = BinaryRelevance(KNeighborsClassifier())
BR_KN.fit(X_train_prec, y_train_enc)

BR_DT = BinaryRelevance(DecisionTreeClassifier(random_state=8))
BR_DT.fit(X_train_prec, y_train_enc)

BR=[BR_GNB, BR_RF, BR_SVC,BR_LR, BR_KN, BR_DT]
BR_label=["BR_GaussianNB", "BR_RandomForest", "BR_SVC",  "BR_KNeighborsClassifier", "BR_DecisionTreeClassifier"]
```

Figura 49. Entrenamiento de modelos Binary Relevance

En *Label Powerset,* el problema de varias etiquetas se transforma en un problema de una única etiqueta, es decir, pasamos a tener un problema de clasificación multiclase. Tendríamos tantas clases como posibles combinaciones distintas entre las etiquetas. En Scikit-multilearn, tenemos disponible una clase para aplicar la técnica de *Label Powerset.*

Vamos a entrenar los siguientes algoritmos:

- Randomn Forest Classifier
- Support Vector Machine
- KNeighbors Classifier
- Decision Tree Classifier

```
LP_RF = LabelPowerset(RandomForestClassifier(random_state=8))
LP_RF.fit(X_train_prec, y_train_enc)

LP_SVC = LabelPowerset(SVC(random_state=8))
LP_SVC.fit(X_train_prec, y_train_enc)

LP_KN = LabelPowerset(KNeighborsClassifier())
LP_KN.fit(X_train_prec, y_train_enc)

LP_DT = LabelPowerset(DecisionTreeClassifier(random_state=8))
LP_DT.fit(X_train_prec, y_train_enc)

LP=[LP_RF,LP_SVC,LP_KN,LP_DT]
LP_label=[ "LP_RandomForest", "LP_SVC", "LP_KNeighborsClassifier", "LP_DecisionTreeClassifier"]
```

Figura 50. Entrenamiento de modelos con Label Powerset

5.7 EVALUACIÓN DE LOS MODELOS

Para la evaluación de los modelos, hemos creado una función llamada test_modelo, en la que se ve evaluando los modelos con el conjunto de *test,* y se va creando un *dataframe* de resultados en el que se recogen todas las métricas indicadas en el apartado de diseño. Además, se muestra la matriz de confusión y la curva ROC para cada modelo evaluado.

Antes de evaluar los modelos, aplicamos el pipeline de preprocesamiento al conjunto de *X_test*, para que tenga la misma forma que el conjunto de training, y codificamos la clase del conjunto test (y_test).

```
X_test_prec =X_test.copy()
X_test_prec['parity'] = np.where(X_test['parity']>3.0, 3.0, X_test['parity'])
y_test_enc= class_encode(pd.DataFrame(y_test))
X_test_prec= prec_pipe.fit_transform(X_test_prec)
```

Figura 51. Codificación del conjunto de test

- **Dummy Classifier**

Comenzamos evaluando el modelo *dummy* que hemos establecido como línea base. Este modelo clasifica en función de la etiqueta más frecuente del conjunto.

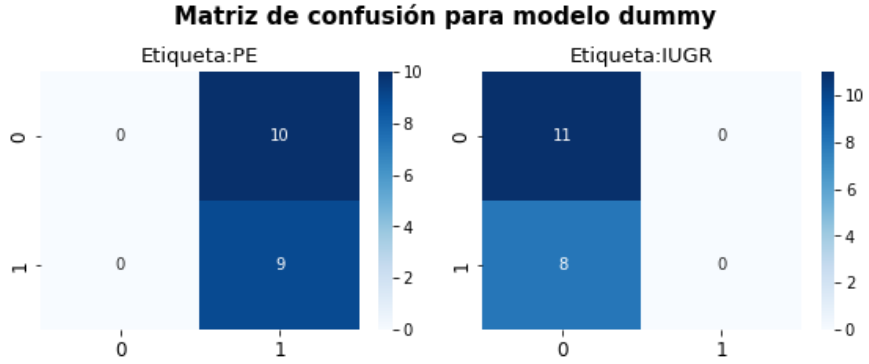

Figura 52. Matriz de confusión del modelo dummy

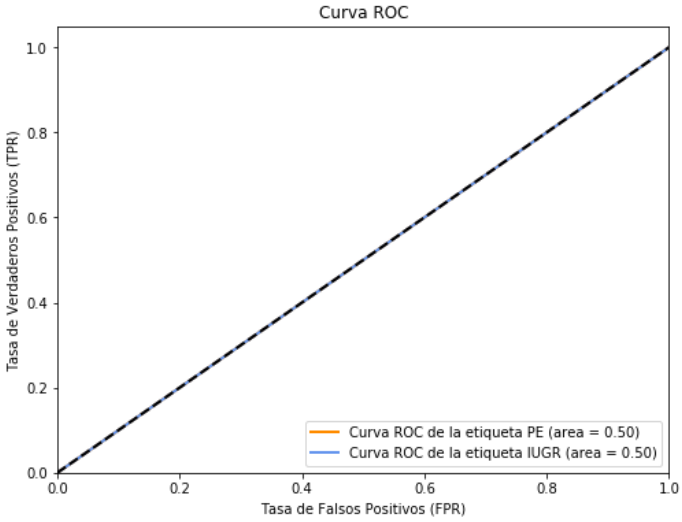

Figura 53. Curva ROC para modelo Dummy

El clasificador ficticio ha clasificado todos en una clase, dándonos un valor de AUC (*area under ROC)* de 0,5. Las métricas de este modelo nos servirá como referencia para comparar las métricas del resto de modelos evaluados.

- **Decision Tree Classifier sin ajustar**

Vamos a evaluar el modelo sin ajustar hiperparámetros, dejando los valores por defecto.

Para este modelo no hemos obtenido una buena clasificación para ninguna de las etiquetas. El modelo clasifica peor el valor 0 en la etiqueta PE, y el valor 1 en IUGR. De esta forma, tiene un alto porcentaje de falsos positivos para la preeclampsia y de falsos negativos para IUGR.

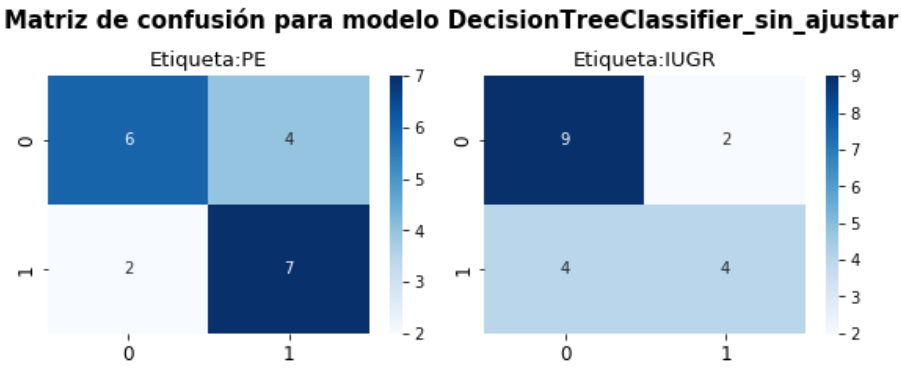

Figura 54. Matriz de confusión Decision Tree sin ajustar

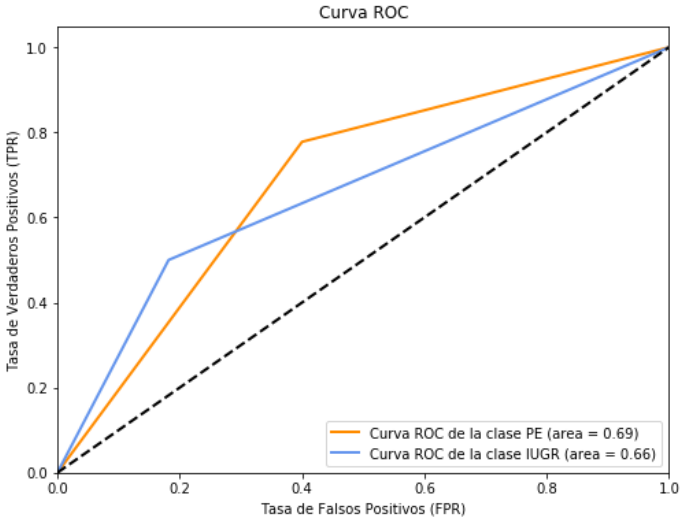

Figura 55. Curva ROC de Decision Tree sin ajustar

- **Decision Tree Classifier ajustado**

Vamos a evaluar el modelo con los hiperparámetros obtenidos en la búsqueda de *gridSearchCV.* Vemos que se obtiene un resultado levemente mejor en la etiqueta IUGR, aunque el efecto es mínimo.

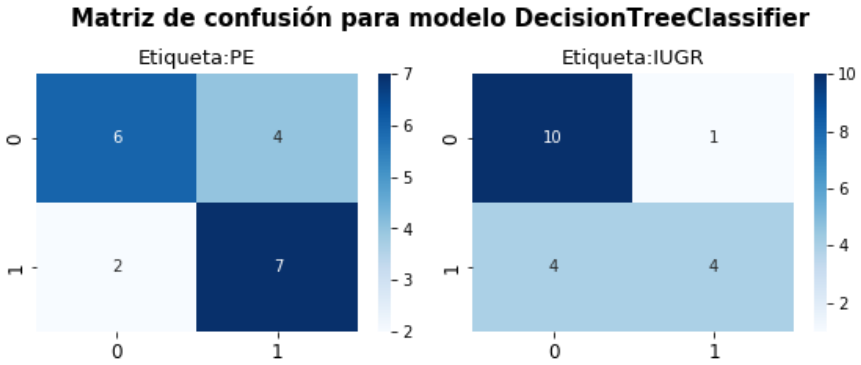

Figura 56. Matriz de confunsión de Desicion Tree Classifier ajustado

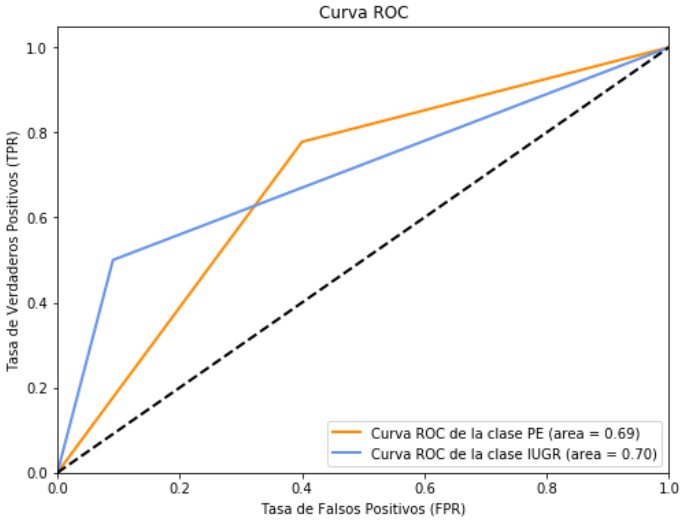

Figura 57. Curva ROC Decision Tree Classifier ajustado

- **Extra Trees Classifier sin ajustar**

Con este modelo obtenemos muy buena clasificación. Para la etiqueta IUGR, tenemos una predicción casi perfecta. En el caso de la preeclampsia, vemos que en algunos casos tiene dificultad para distinguir entre tener preeclampsia o no.

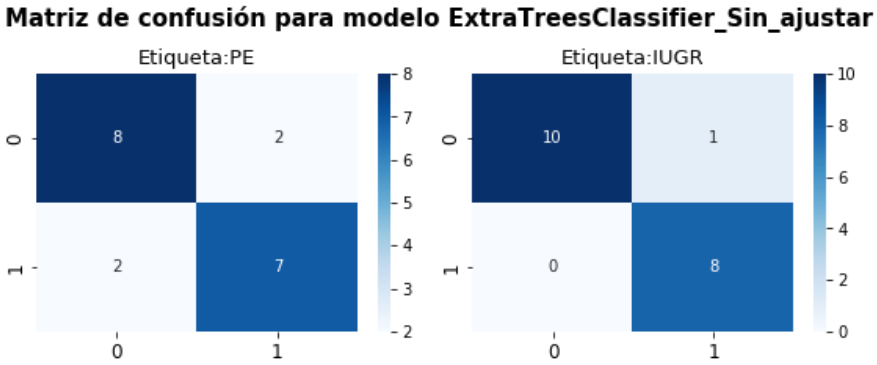

Figura 58. Matriz de confusión para Extra Trees Classifier sin ajustar

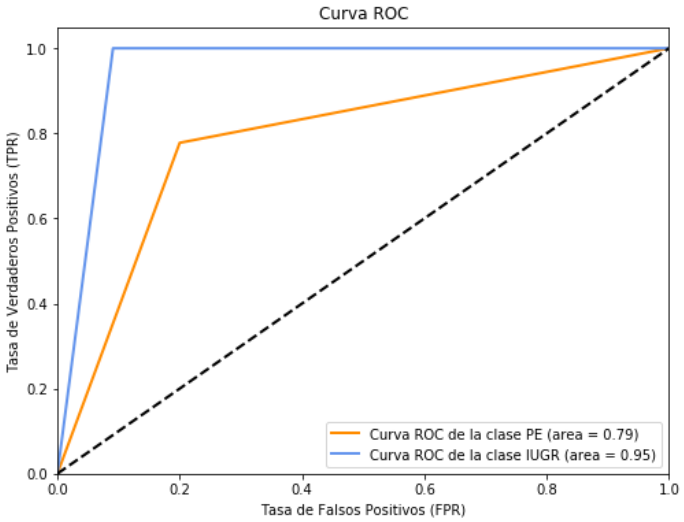

Figura 59. Curva ROC para Extra Trees Classifier sin ajustar

En la gráfica de la curva ROC podemos ver que el valor AUC-ROC para IUGR es muy alto. Para PE es más bajo, aunque sigue siendo bueno.

- **Extra Trees Classifier ajustado**

En el caso de este modelo, no hemos obtenido mejores resultados al ajustar los hiperparámetros. Los resultados son levemente peores para IUGR.

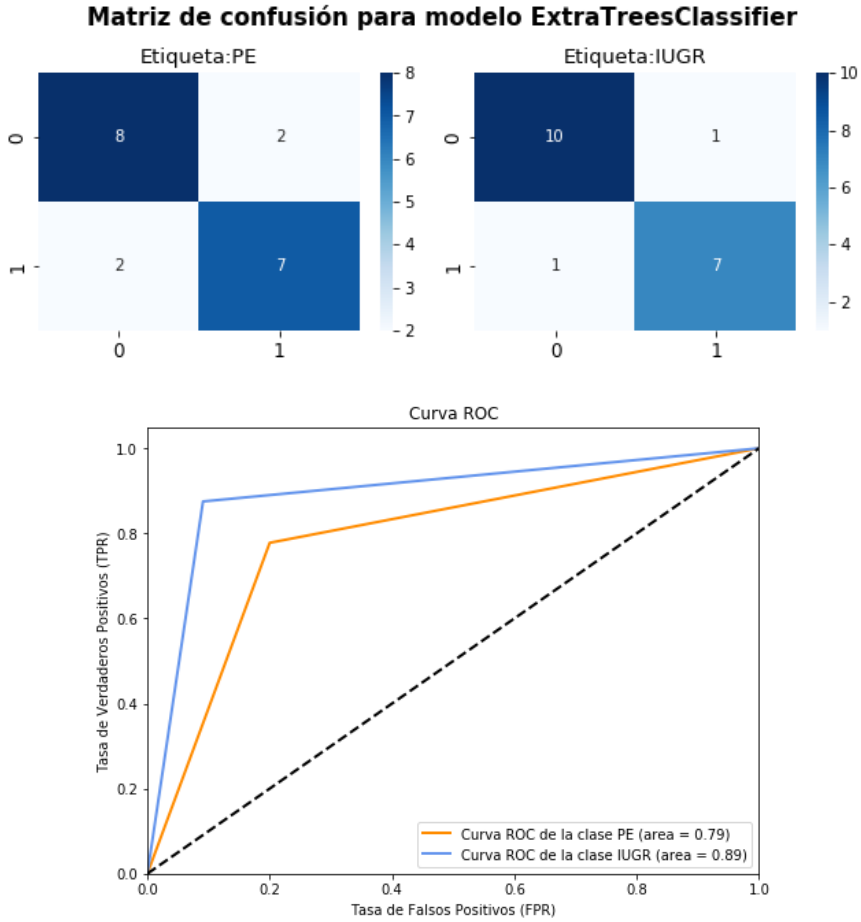

Figura 60. Curva ROC para Extra Trees ajustado

- **Kneighbors Classifier sin ajustar**

Con este algoritmo obtenemos un buen clasificador para la etiqueta IUGR. Sin embargo, el rendimiento para PE no es tan bueno, teniendo un porcentaje alto de falsos positivos.

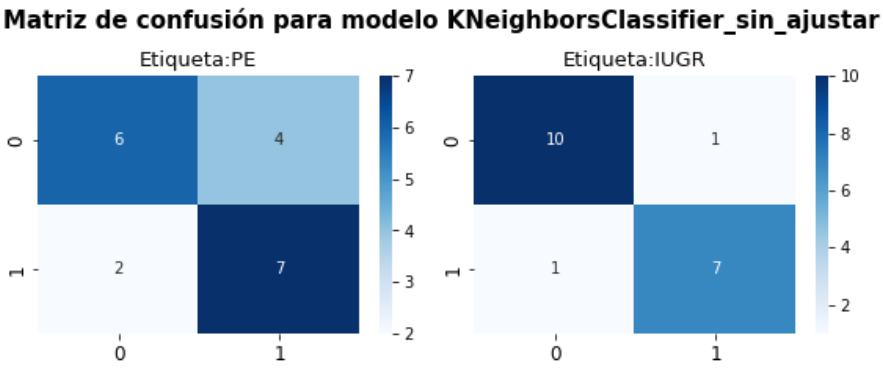

Figura 61. Matriz de confusión para Kneighbors Classifier sin ajustar

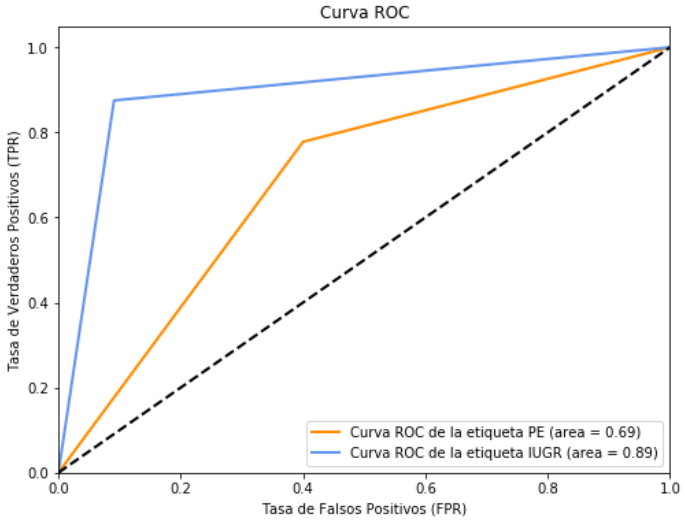

Figura 62. Curva ROC de Kneighbors Classifier sin ajustar

- **Kneighbors Classifier ajustado**

En el modelo ajustado, obtenemos una leve mejoría en el rendimiento del clasificador de PE, pero la clasificación de IUGR empeora levemente. Cabe destacar, que para IUGR ha aumentado el número de falsos negativos y para PE ha disminuido los falsos positivos. Teniendo en cuenta que estamos prediciendo dos enfermedades, este clasificador sería peor que el anterior ya que es preferible tener algún falso positivo más, que pasar por alto la presencia de la enferma (falsos negativos).

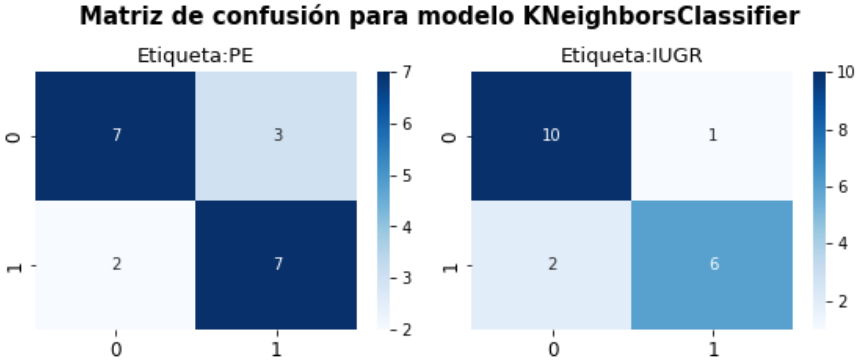

Figura 63. Matriz de confunsión para Kneighbors Classifier ajustado

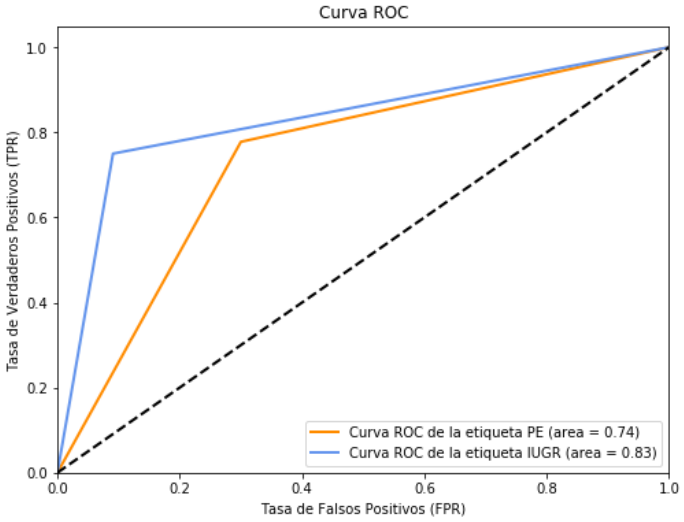

Figura 64. Curva ROC para Kneighbors Classifier ajustado

- **Random Forest Classifier sin ajustar**

Con este algoritmo obtenemos un buen resultado en la clasificación de ambas etiquetas.

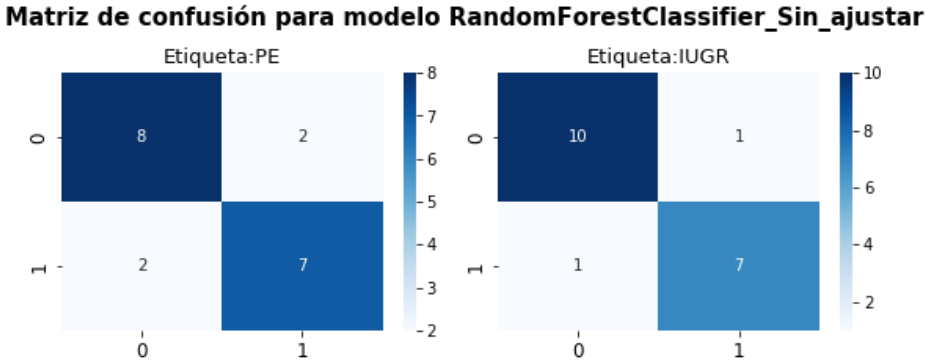

Figura 65. Matriz de confusión para Random Forest Classifier sin ajustar

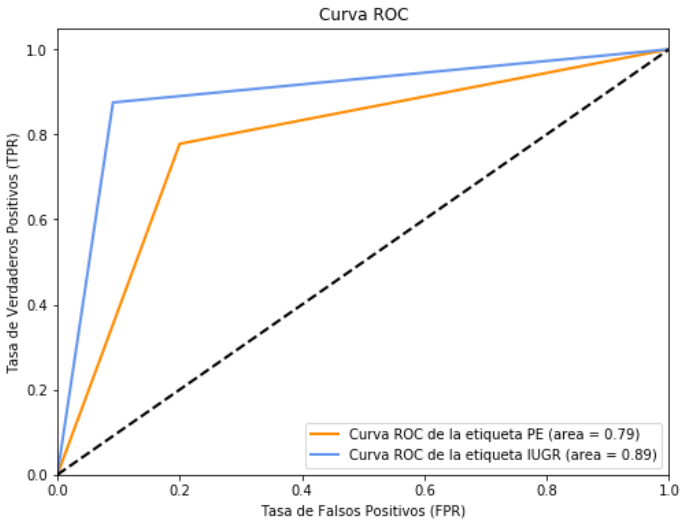

Figura 66. Curva ROC para Random Forest Classifier sin ajustar

A continuación, estudiaremos si se obtiene un resultado mejor ajustando los hiperparámetros.

- **Random Forest Classifier ajustado**

Con el ajuste, hemos obtenido los mismos resultados que el modelo sin ajustas en el conjunto de *test*.

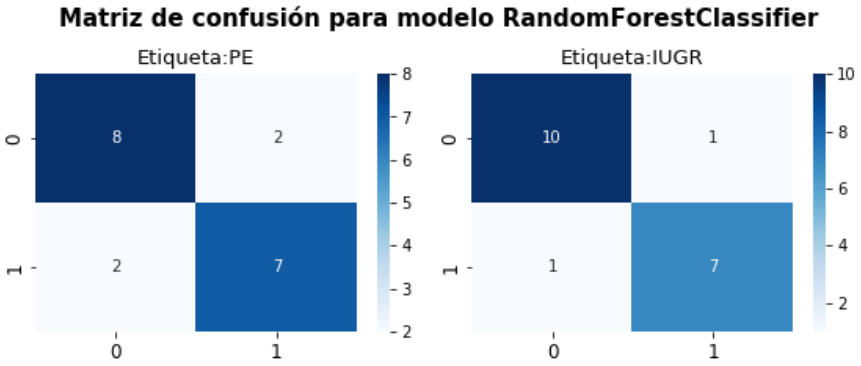

Figura 67. Matriz de Confusión Random Forest Classifier ajustado

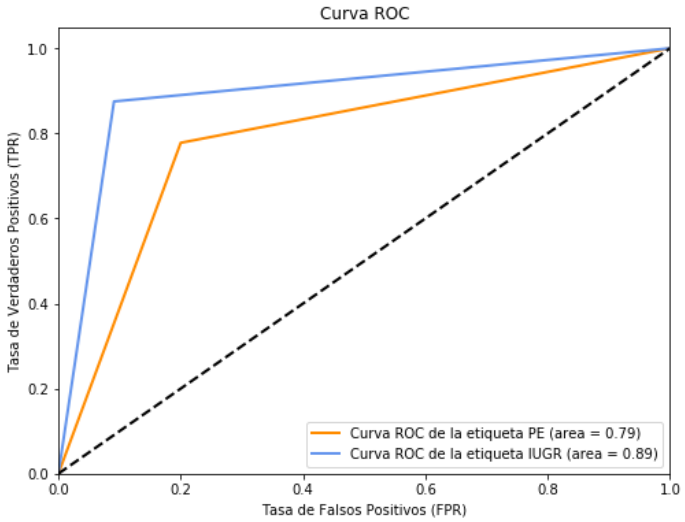

Figura 68. Curva ROC para Random Forest Classifier ajustado

- **Binary Research con GaussianNB**

Este modelo nos devuelve un buen rendimiento para IUGR. Para PE, está por encima del valor de referencia, pero no es de los mejores rendimientos obtenidos.

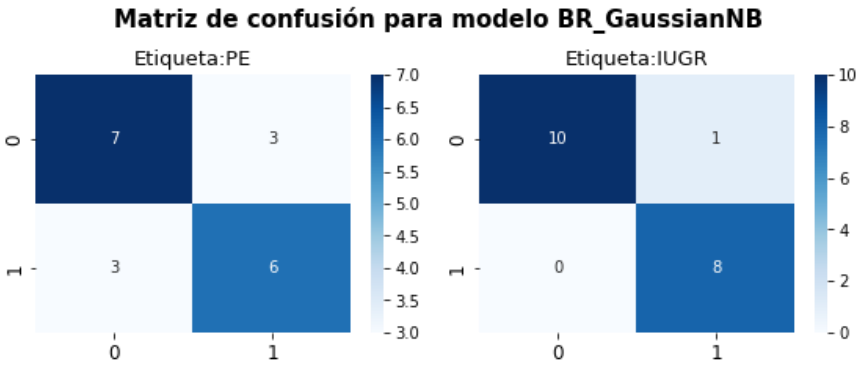

Figura 69. Matriz de confusión para Binary Research con GaussianNB

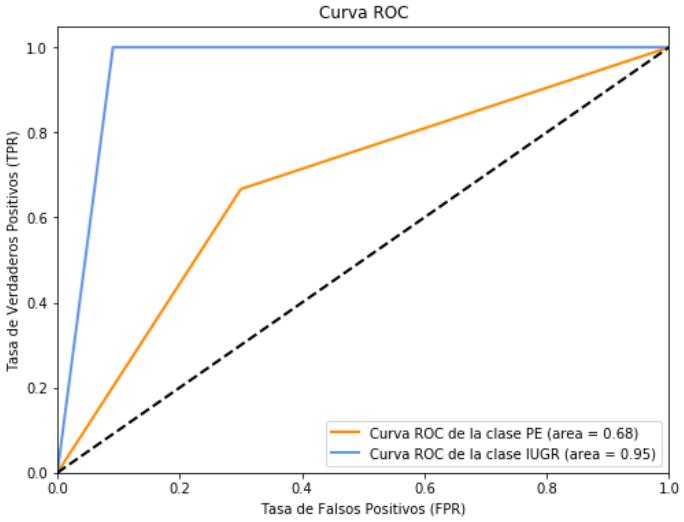

Figura 70. Curva ROC Binary Research con GaussianNB

- **Binary Research Random Forest**

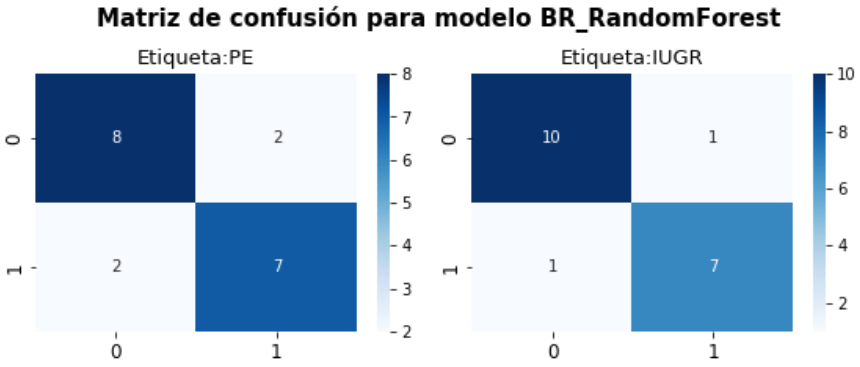

Figura 71. Matriz de confusión para Binary Research con Random Forest

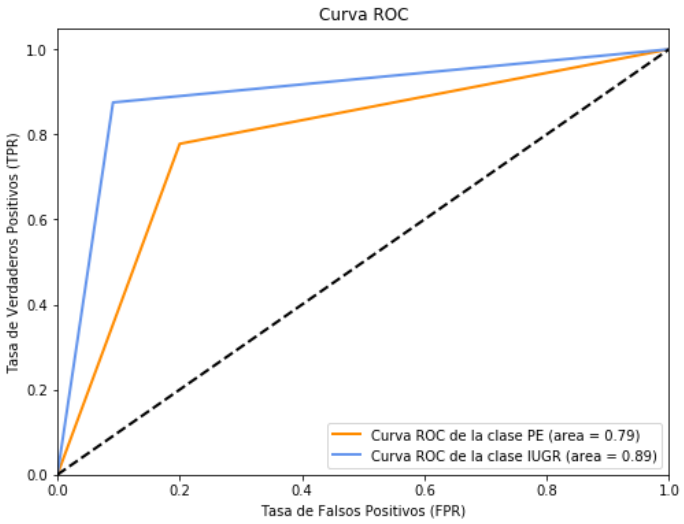

Figura 72. Curva ROC para Binary Research con Random Forest

- **Binary Research SVC**

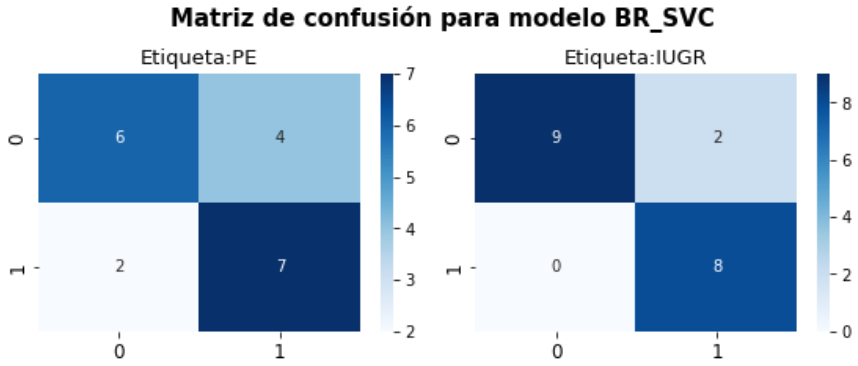

Figura 73. Matriz de confusión de Binary Research con SVC

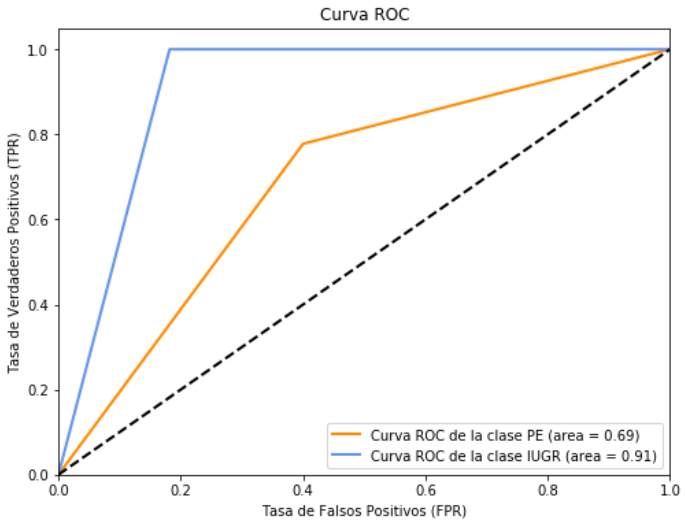

Figura 74. Curva ROC para Binary Research con SVC

- **Binary Research KNeighbors Classifier**

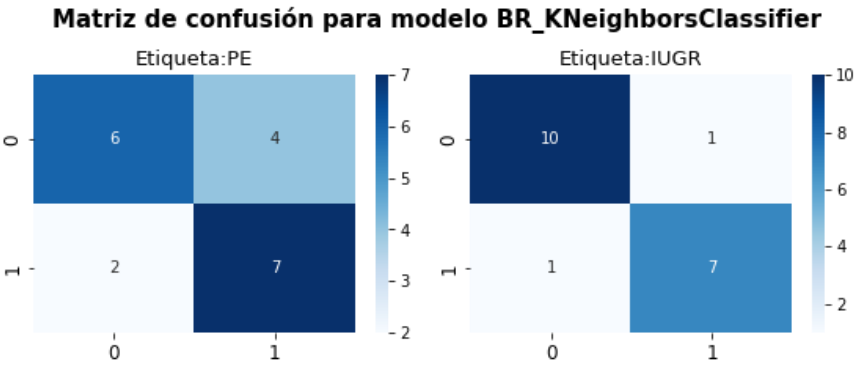

Figura 75. Matriz de confusión para Binary Research con KNeighbors Classifier

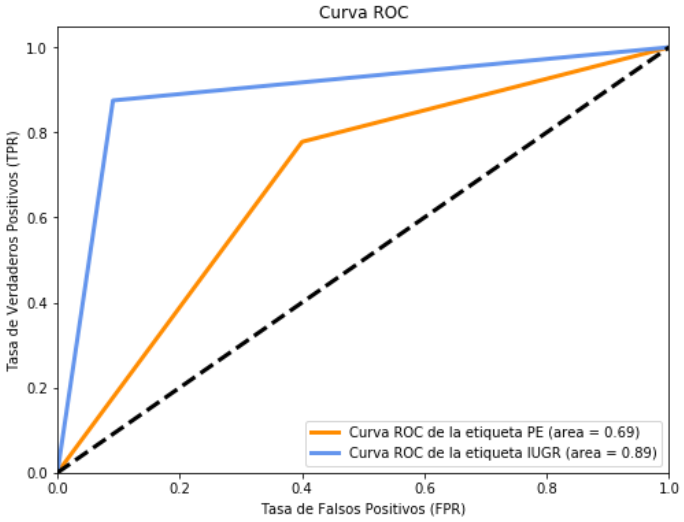

Figura 76. Curva ROC para Binary Research con KNeighbors Classifier

- **Binary Research Decision Tree Classifier**

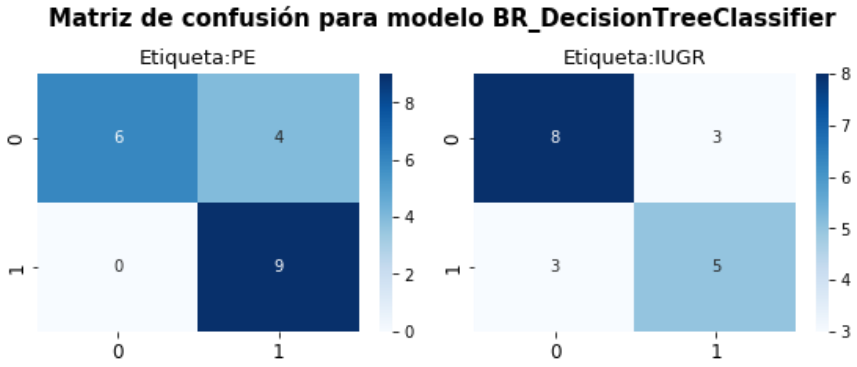

Figura 77. Matriz de confusión para Binary Research con Decision Tree Classifier

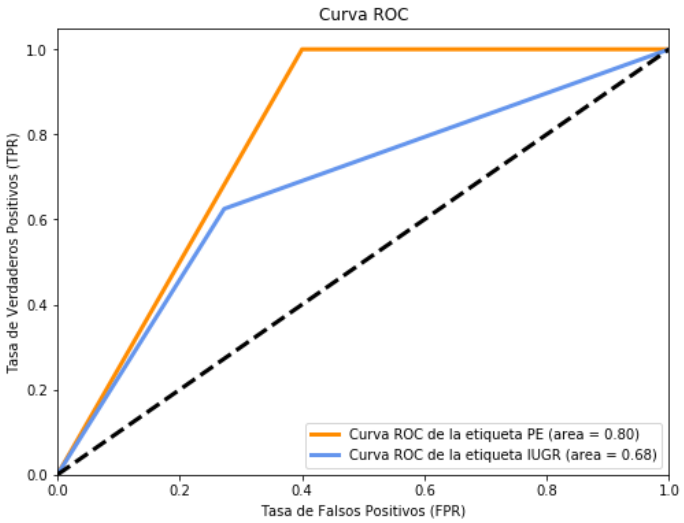

Figura 78. Curva ROC para Binary Research con Decision Tree Classifier

- **Label Powerset con Random Forest**

La salida de este modelo para IUGR, aunque tenga un valor AUC-ROC alto, es peor que las salidas de otros modelos, ya que ha detectado más falsos negativos. Como se ha comentado anteriormente, teniendo en cuenta el contexto clínico en el que nos encontramos, es preferible tener más falsos positivos (dentro de un límite razonable, por supuesto) que falsos negativos.

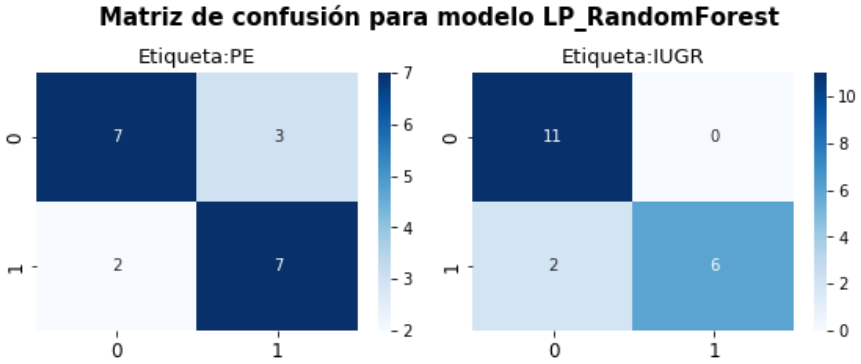

Figura 79. Matriz de Confusión para Label Powerset Random Forest Classifier

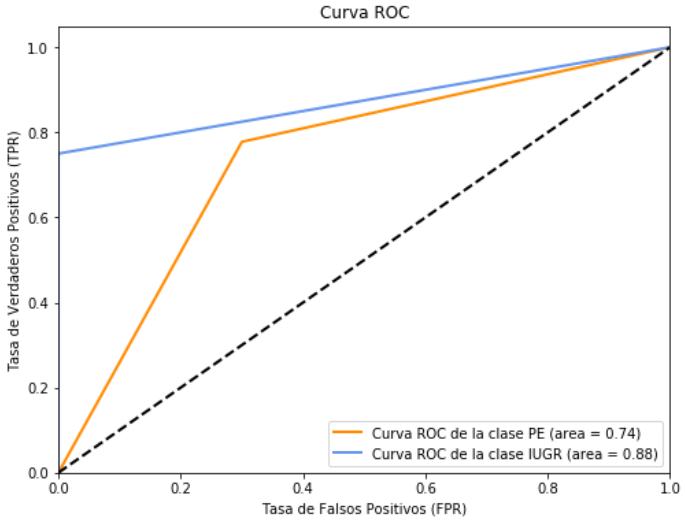

Figura 80. Curva ROC para Label Powerset Random Forest Classifier

- **Label Powerset con SVC**

Con este clasificador, obtenemos un mal rendimiento para PE, mientras que para IUGR sí son bastantes buenos.

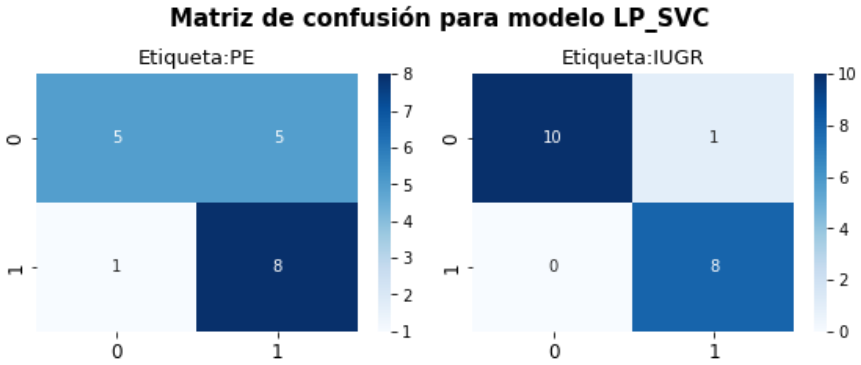

Figura 81. Matriz de confusión para Label Powerset SVC

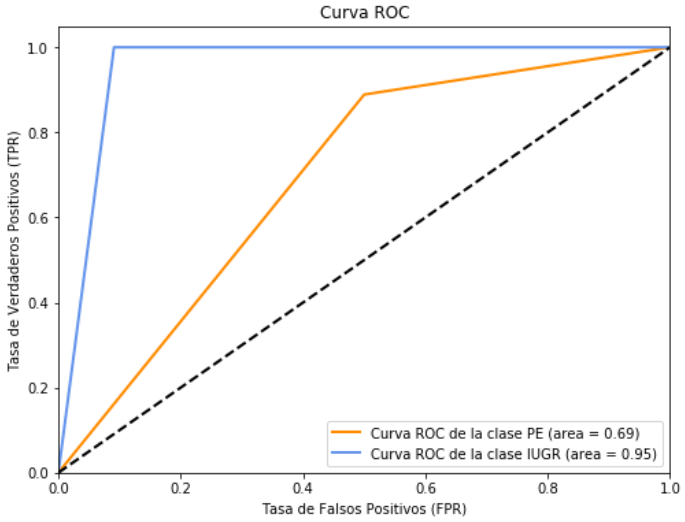

Figura 82. Curva ROC para Label Powerset SVC

- **Label Powerset KNeighbors Classifier**

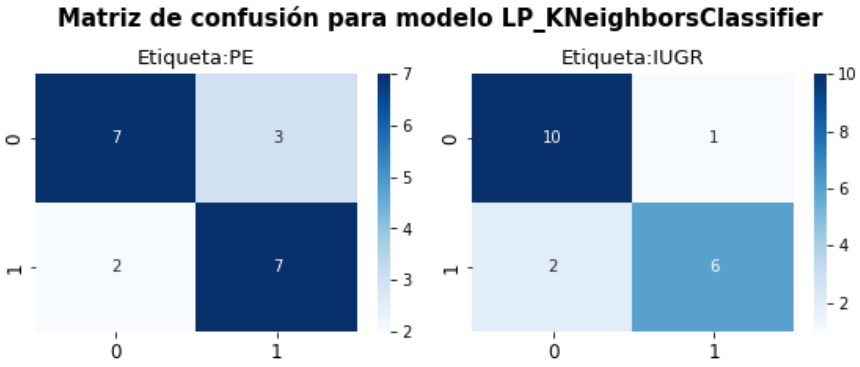

Figura 83. Matriz de confusión para Label Powerset con KNeighbors Classifier

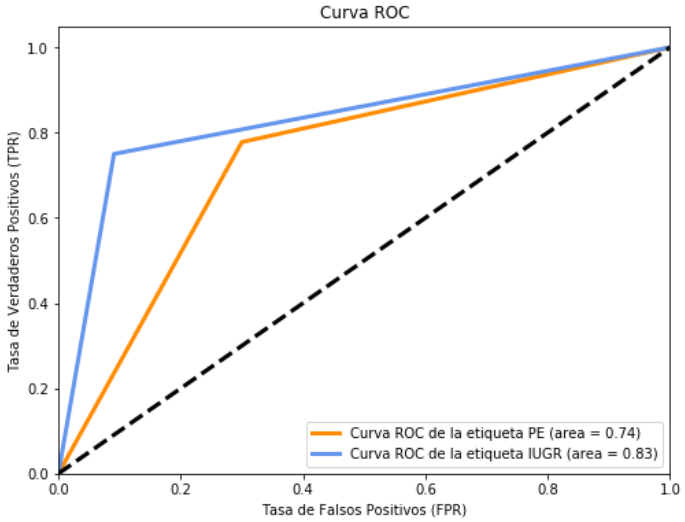

Figura 84. Curva ROC para Label Powerset Kneighbors Classifier

- **Label Powerset Decision Tree Classifier**

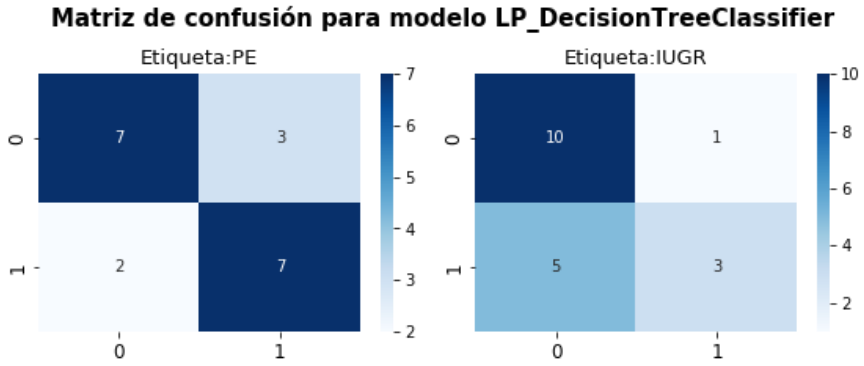

Figura 85. Matriz de confusión para Label Powerset con Decision Tree Classifier

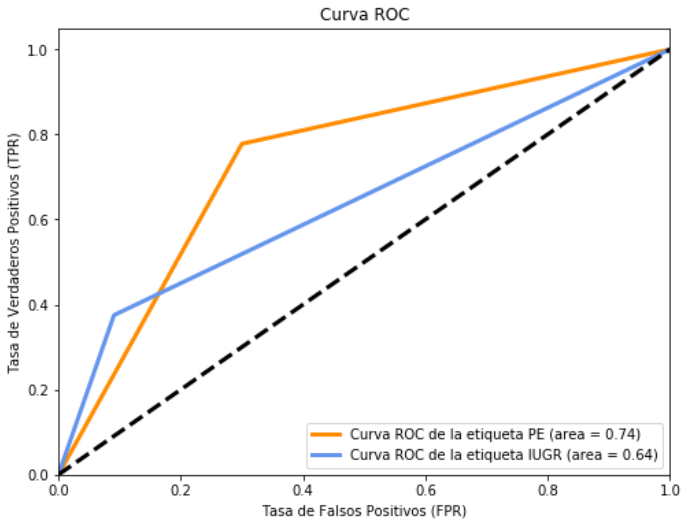

Figura 86. Curva ROC para Label Powerset con Decision Tree Classifier

De forma general, hemos podido observar que los modelos desarrollados tienen un rendimiento mejor para la etiqueta IUGR.

Las métricas de rendimiento de todos estos modelos se han ido almacenando en *dataframe* de resultados para poder hacer una comparación de los modelos más fácilmente. Las columnas del *dataframe* son: *accuracy, precision, recall, F1-score, ROC-AUC score* y el valor de *Hamming Loss.*

La *Figura 87.* Métricas de rendimiento de los modelos validadosmuestra la tabla de los resultados obtenidos, ordenada en función del valor del área bajo la curva ROC en orden descendente.

Todos los modelos han superado el modelo de referencia *dummy*. Así, podemos concluir que todos los modelos desarrollados hacen al menos una clasificación mejor que un modelo que utiliza reglas simples.

En general, el modelo **Extra Tree Classifier** ha sido el que ha obtenido un mejor rendimiento en casi todas las métricas.

	Accuracy	Precision	Recall	F1	roc auc scores	Hamming Loss
ExtraTreesClassifier_Sin_ajustar	0.789474	0.833333	0.888889	0.859477	0.871717	0.131579
RandomForestClassifier_Sin_ajustar	0.736842	0.826389	0.826389	0.826389	0.840467	0.157895
BR_RandomForest	0.736842	0.826389	0.826389	0.826389	0.840467	0.157895
RandomForestClassifier	0.736842	0.826389	0.826389	0.826389	0.840467	0.157895
ExtraTreesClassifier	0.736842	0.826389	0.826389	0.826389	0.840467	0.157895
LP_SVC	0.631579	0.752137	0.944444	0.834225	0.824495	0.184211
BR_GaussianNB	0.631579	0.777778	0.833333	0.803922	0.818939	0.184211
LP_RandomForest	0.631579	0.850000	0.763889	0.796992	0.806944	0.184211
BR_SVC	0.631579	0.718182	0.888889	0.794444	0.798990	0.210526
BR_KNeighborsClassifier	0.578947	0.755682	0.826389	0.787500	0.790467	0.210526
KNeighborsClassifier_sin_ajustar	0.578947	0.755682	0.826389	0.787500	0.790467	0.210526
LP_KNeighborsClassifier	0.578947	0.778571	0.763889	0.768421	0.784217	0.210526
KNeighborsClassifier	0.578947	0.778571	0.763889	0.768421	0.784217	0.210526
BR_DecisionTreeClassifier	0.526316	0.658654	0.812500	0.721591	0.738068	0.263158
DecisionTreeClassifier	0.421053	0.718182	0.638889	0.657692	0.696717	0.289474
LP_DecisionTreeClassifier	0.526316	0.725000	0.576389	0.618421	0.690467	0.289474
DecisionTreeClassifier_sin_ajustar	0.421053	0.651515	0.638889	0.635714	0.673990	0.315789
dummy	0.210526	0.236842	0.500000	0.321429	0.500000	0.473684

Figura 87. Métricas de rendimiento de los modelos validados

5.8 ELECCIÓN Y EXPLICACIÓN DEL MODELO FINAL

El modelo elegido es el **Extra Tree Classifier.** Con este modelo, hemos conseguido las siguientes métricas de rendimiento: 0,789474 en *accuracy,* 0.83333 en *precision*, 0.888889 en *recall,* 0.859477 en *F1-score*, 0.871717 en *AUC-ROC* y 0.131579 en *Hamming Loss*. Las métricas han superado las del modelo base establecido. Además, consideramos que se han obtenido unos buenos resultados teniendo en cuenta la limitación del tamaño del dataset y el uso de unos datos reales pertenecientes a un estudio.

Como se puede observar, se ha obtenido muy buen resultado en la medida de *Hamming Loss* (valor ideal de 0). Eso significa que la tasa de error del modelo en la predicción de las etiquetas es baja.

A continuación, se va estudiar la importancia de cada variable sobre la predicción del modelo.

Como se puede observar en la *Figura 88.* Importancia de cada variable sobre el modelo., los atributos con mayor relevancia en el modelo son los relacionados con las medidas Doppler y los valores de los biomarcadores. Este resultado es coherente con lo visto en la bibliografía

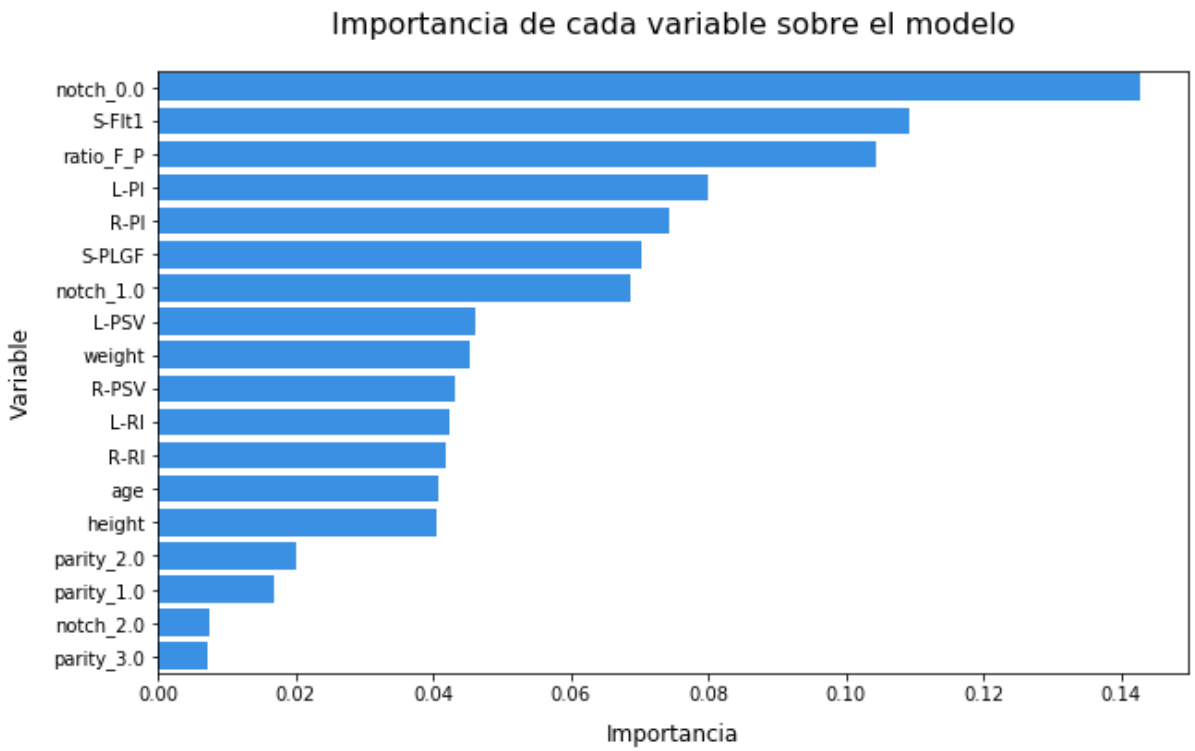

Figura 88. Importancia de cada variable sobre el modelo.

Podemos ver que un gran indicador es la ausencia de una muesca diastólica en la ecografía Doppler. Como vimos en el análisis exploratorio, el grupo de control no presentaba *notch*, y en el resto de grupos se podía apreciar que los que presentaban retraso del crecimiento intrauterino tenían una mayor de proporción de *notch* tanto unilateral (notch_1.0) como bilateral (notch_2.0).

La sFlt-1 también es un indicador muy relevante. Como pudimos observar en el análisis exploratorio, el grupo de control presentaba valores bajos de esta proteína, mientras que los grupos con alguna patología presentaban valores mucho mayores, sobre todo para los grupos con preeclampsia. Al contrario ocurría con el PlGF: para el grupo de control teníamos valores altos, mientras que para el resto teníamos valores bajos (en este caso, la diferencia entre preeclampsia y retraso de crecimiento intrauterino no es tan notable).

Vemos que de las medidas que no pertenecen a la ecografía Doppler o a los biomarcadores, el peso es la que tiene mayor relevancia.

Tras esto, vamos a decodificar la salida del modelo y vamos a visualizar una matriz multiclase, en el que se compara la clase obtenida a la salida del modelo tras decodificarla (por ejemplo, [1,0] sería decodificado como preeclampsia), y la salida real.

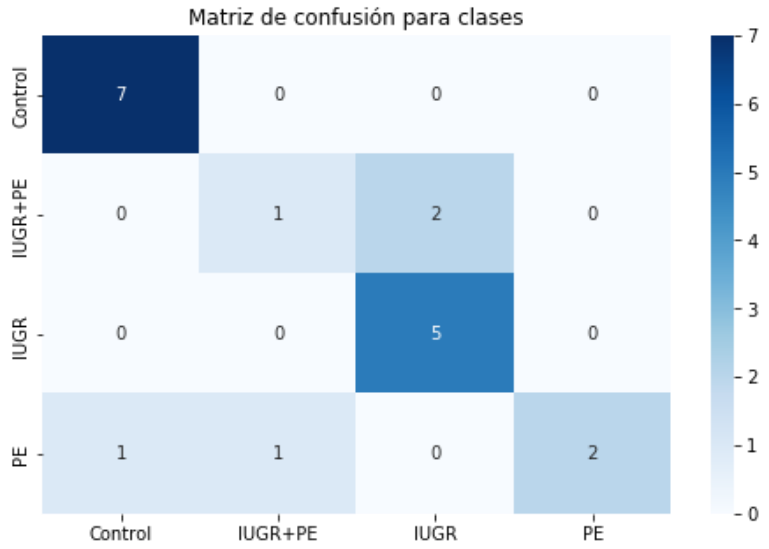

Figura 89. Matriz de confusión multiclase

Vemos que en la mayoría de los casos el modelo se ha equivocado entre IUGR, PE e IUGR+PE. Esto se debe a que detecta sólo una de las enfermedades, a pesar de padecer las dos. En ese sentido, necesitaríamos más datos de embarazadas que padezcan estas enfermedades para mejorar la clasificación.

6 CONCLUSIONES

En este Trabajo Fin de Máster se ha desarrollado un modelo predictivo para detectar riesgos en embarazadas, en concreto preeclampsia y retraso del crecimiento uterino. El *dataset* utilizado para ello es un *dataset* real sacado de un estudio. Esto ha supuesto varios retos en el desarrollo de este TFM. El principal reto ha sido el tamaño del *dataset*. Se cuenta con muy pocos datos, y el hecho de tener cuatro clases distintas (en proporción desbalanceada), ha dificultado el desarrollo y abordaje del modelo. Otro de los retos encontrado ha estado en el preprocesamiento de los datos. Al tratarse de datos reales, se ha invertido bastante tiempo en tratarlos.

De cualquier forma, ha sido una experiencia muy enriquecedora a nivel tanto académico como personal. La elaboración de esta memoria me ha permitido profundizar más en el ámbito del aprendizaje automático, y adquirir muchos nuevos conocimientos. Además, me ha permitido comenzar a estudiar temas relacionados con la explicabilidad de la Inteligencia Artificial, concepto muy importante en el ámbito sanitario para comenzar a adoptar herramientas de Inteligencia Artificial.

El TFM también me ha permitido conocer más de cerca los avances de la comunidad científica en el desarrollo de modelos predictivos para riesgos en embarazadas. Creo en el gran potencial de la IA en ámbito de la salud en general, y en el de obstétrica y ginecología en particular. Muchas embarazadas podrían beneficiarse de este tipo de sistemas, por ejemplo, con tratamientos más personalizados, a través de la monitorización de su estado de salud, o con un diagnóstico precoz de alguna patología. Este caso último correspondería con el modelo desarrollado en este trabajo. La detección temprana de la preeclampsia y el retraso de crecimiento intrauterino es crítica para comenzar a tratarla, y para disminuir la mortalidad

y morbilidad materna y perinatal. Además, también hemos podido comprobar que las medidas Doppler y los biomarcadores sFlt-1 y PlGF son datos importantes para la predicción de estas enfermedades.

Por último, se han cumplido con los tres primeros objetivos del TFM. En cuanto al cuarto objetivo, finalmente no ha podido abordarse y no se ha podido profundizar tanto como se esperaba en un principio en temas de explicabilidad.

7 TRABAJOS FUTUROS

Algunos de los posibles trabajos futuros identificados en este trabajo son:

- Desplegar el modelo de *machine learning* para que pueda ser consultado por otros sistemas.
- Desarrollar una aplicación web clínica que integre el modelo de *machine learning*. Esta aplicación serviría como cuadro de mandos para el médico, para poder visualizar el histórico de los pacientes, crear nuevos registros y obtener una predicción, y visualizar gráficas sobre la evaluación de los parámetros de las pacientes.
- Evaluar el modelo con nuevos datos para que continue aprendiendo y mejorar la predicción. Como se ha comentado, uno de los principales problemas encontrados durante el desarrollo del modelo ha sido el tamaño del *dataset.*
- Incluir nuevos atributos de otras fuentes de información, y evaluar el impacto en las métricas de la clasificación (cuestionarios de reporte de síntomas, datos de historia clínica, etc).

8 BIBLIOGRAFÍA

Ahmed, M. and Kashem, M.A. (2020) “IoT Based Risk Level Prediction Model for Maternal Health Care in the Context of Bangladesh,” *2020 2nd International Conference on Sustainable Technologies for Industry 4.0, STI 2020* [Preprint]. doi:10.1109/STI50764.2020.9350320.

Alfaleh, A. and Gollapalli, M. (2020) “A Critical Review of Data Mining Techniques Used for the Management of Sickle Cell Disease,” in *PervasiveHealth: Pervasive Computing Technologies for Healthcare*, pp. 147–152. doi:10.1145/3408066.3408105.

An Introduction to GitHub – Digital.gov (no date). Available at: https://digital.gov/resources/an-introduction-github/ (Accessed: December 10, 2021).

Asaduzzaman, S. *et al.* (2021) “Machine learning to reveal an astute risk predictive framework for Gynecologic Cancer and its impact on women psychology: Bangladeshi perspective,” *BMC Bioinformatics*, 22(1). doi:10.1186/s12859-021-04131-6.

Baker, S., Xiang, W. and Atkinson, I. (2021) “Hybridized neural networks for non-invasive and continuous mortality risk assessment in neonates,” *Computers in Biology and Medicine*, 134. doi:10.1016/j.compbiomed.2021.104521.

Begum, M., Redoy, R.M. and das Anty, A. (2021) “Preterm Baby Birth Prediction using Machine Learning Techniques,” *2021 International Conference on Information and*

Communication Technology for Sustainable Development, ICICT4SD 2021 - Proceedings, pp. 50–54. doi:10.1109/ICICT4SD50815.2021.9396933.

Chatterjee, A., Gerdes, M.W. and Martinez, S.G. (2020) “Identification of risk factors associated with obesity and overweight—a machine learning overview,” *Sensors (Switzerland)*, 20(9). doi:10.3390/s20092734.

Chinnaiyan, R. and Alex, S. (2021) “Machine Learning Approaches for Early Diagnosis and Prediction of Fetal Abnormalities,” in *2021 International Conference on Computer Communication and Informatics, ICCCI 2021*. doi:10.1109/ICCCI50826.2021.9402317.

Damaraji, G.M., Permanasari, A.E. and Hidayah, I. (2020) “A Review of Expert System for Identification Various Risk in Pregnancy,” in *2020 3rd International Conference on Information and Communications Technology, ICOIACT 2020*, pp. 99–104. doi:10.1109/ICOIACT50329.2020.9332003.

Feature-engine: A Python library for Feature Engineering for Machine Learning — 1.1.2 (no date). Available at: https://feature-engine.readthedocs.io/en/1.1.x/ (Accessed: December 10, 2021).

Friedman, A.M. and Cleary, K.L. (2014) “Prediction and prevention of ischemic placental disease,” *Seminars in Perinatology*, 38(3), pp. 177–182. doi:10.1053/J.SEMPERI.2014.03.002.

García, I.H. *et al.* (2011) “Doppler de arterias uterinas y marcadores angiogénicos (sFlt-1/PlGF): futuras implicaciones para la predicción y el diagnóstico de la preeclampsia,” *Diagnóstico Prenatal*, 22(2), pp. 32–40. doi:10.1016/J.DIAPRE.2010.01.001.

Guía de los fundamentos para la dirección de proyectos : (Guía del PMBOK®) (2017) *Project Management Institute*.

Herrera, F. *et al.* (2016) “Multilabel Classification,” *Multilabel Classification*, pp. 17–31. doi:10.1007/978-3-319-41111-8_2.

History: The Agile Manifesto (no date). Available at: http://agilemanifesto.org/history.html (Accessed: December 10, 2021).

Hou, F. *et al.* (2020) “Prediction of Gestational Diabetes Based on LightGBM,” in *PervasiveHealth: Pervasive Computing Technologies for Healthcare*, pp. 161–165. doi:10.1145/3433996.3434025.

Hou, Fan *et al.* (2020) “Prediction of Gestational Diabetes Based on LightGBM,” *ACM International Conference Proceeding Series*, pp. 161–165. doi:10.1145/3433996.3434025.

Irfan, M., Basuki, S. and Azhar, Y. (2021) “Giving more insight for automatic risk prediction during pregnancy with interpretable machine learning,” *Bulletin of Electrical Engineering and Informatics*, 10(3), pp. 1621–1633. doi:10.11591/eei.v10i3.2344.

Jiang, F. *et al.* (2017) “Artificial intelligence in healthcare: past, present and future,” *Stroke and vascular neurology*, 2(4), pp. 230–243. doi:10.1136/SVN-2017-000101.

Jira | Software de seguimiento de proyectos e incidencias (no date). Available at: https://www.atlassian.com/es/software/jira (Accessed: December 10, 2021).

Kim, C. (2010) “Gestational diabetes: risks, management, and treatment options,” *International Journal of Women’s Health*, 2(1), p. 339. doi:10.2147/IJWH.S13333.

Kuhn, M. and Johnson, K. (2013) “Applied predictive modeling,” *Applied Predictive Modeling*, pp. 1–600. doi:10.1007/978-1-4614-6849-3.

Manasrah, A. *et al.* (2021) “Assessment of machine learning security: The case of healthcare data,” in *ACM International Conference Proceeding Series*, pp. 91–98. doi:10.1145/3460620.3460738.

Marques, J.A.L. *et al.* (2020) “IoT-based Smart Health System for Ambulatory Maternal and Fetal Monitoring,” *IEEE Internet of Things Journal* [Preprint]. doi:10.1109/JIOT.2020.3037759.

Matplotlib — Visualization with Python (no date). Available at: https://matplotlib.org/ (Accessed: December 10, 2021).

Mendeley Data - Uterine arteries Doppler and sFlt-1/PlGF ratio in hypertensive disorders during pregnancy (no date). Available at: https://data.mendeley.com/datasets/zsjhvy9ytx/1 (Accessed: October 22, 2021).

Mol, B.W.J. *et al.* (2016) “Pre-eclampsia,” *The Lancet*, 387(10022), pp. 999–1011. doi:10.1016/S0140-6736(15)00070-7.

Moreira, M.W.L. *et al.* (2021) “Neuro-fuzzy model for HELLP syndrome prediction in mobile cloud computing environments,” *Concurrency and Computation: Practice and Experience*, 33(7), p. 1. doi:10.1002/CPE.4651.

Murki, S. and Sharma, D. (2014) “Intrauterine Growth Retardation - A Review Article,” *Journal of Neonatal Biology*, 3(3), pp. 1–13. doi:10.4172/2167-0897.1000135.

NumPy (no date). Available at: https://numpy.org/ (Accessed: December 10, 2021).

Oprescu, A.M. *et al.* (2020) “Artificial intelligence in pregnancy: A scoping review,” *IEEE Access*, 8, pp. 181450–181484. doi:10.1109/ACCESS.2020.3028333.

Oprescu, Andreea M. *et al.* (2020) “Artificial intelligence in pregnancy: A scoping review,” *IEEE Access*, 8, pp. 181450–181484. doi:10.1109/ACCESS.2020.3028333.

pandas - Python Data Analysis Library (no date). Available at: https://pandas.pydata.org/ (Accessed: December 10, 2021).

Project Jupyter | Home (no date). Available at: https://jupyter.org/ (Accessed: December 10, 2021).

Schwaber, K. (1997) “SCRUM Development Process,” *Business Object Design and Implementation*, pp. 117–134. doi:10.1007/978-1-4471-0947-1_11.

scikit-learn: machine learning in Python — scikit-learn 1.0.1 documentation (no date). Available at: https://scikit-learn.org/stable/ (Accessed: December 10, 2021).

scikit-multilearn: Multi-Label Classification in Python — Multi-Label Classification for Python (no date). Available at: http://scikit.ml/ (Accessed: December 10, 2021).

Shatte, A.B.R. *et al.* (2020) “Social media markers to identify fathers at risk of postpartum depression: A machine learning approach,” *Cyberpsychology, Behavior, and Social Networking*, 23(9), pp. 611–618. doi:10.1089/cyber.2019.0746.

Sterckx, L. *et al.* (2020) “Clinical information extraction for preterm birth risk prediction,” *Journal of Biomedical Informatics*, 110. doi:10.1016/j.jbi.2020.103544.

Tissot, H.C. and Pedebos, L.A. (2021) “Improving Risk Assessment of Miscarriage During Pregnancy with Knowledge Graph Embeddings,” *Journal of Healthcare Informatics Research* [Preprint]. doi:10.1007/S41666-021-00096-6.

Tukey, J.W. (1977) *Exploratory data analysis*, *Exploratory Data Analysis, Volume 2*. Reading Mass.: Addison-Wesley Pub. Co.

V, F.-V. *et al.* (2019) “Correlation between uterine artery Doppler and the sFlt-1/PlGF ratio in different phenotypes of placental dysfunction,” *Hypertension in pregnancy*, 38(1), pp. 32–40. doi:10.1080/10641955.2018.1550579.

Veena, S. and Aravindhar, D.J. (2021) “Remote Monitoring System for the Detection of Prenatal Risk in a Pregnant Woman,” *Wireless Personal Communications 2021 119:2*, 119(2), pp. 1051–1064. doi:10.1007/S11277-021-08249-X.

Waskom, M. (2021) “seaborn: statistical data visualization,” *Journal of Open Source Software*, 6(60), p. 3021. doi:10.21105/JOSS.03021.

Wei *et al.* (2016) “Guidelines for Developing and Reporting Machine Learning Predictive Models in Biomedical Research: A Multidisciplinary View,” *J Med Internet Res 2016;18(12):e323 https://www.jmir.org/2016/12/e323*, 18(12), p. e5870. doi:10.2196/JMIR.5870.

What is Python? Executive Summary | Python.org (no date). Available at: https://www.python.org/doc/essays/blurb/ (Accessed: December 10, 2021).

WHO (2021) “Ethics and Governance of Artificial Intelligence for Health: WHO guidance,” *World Health Organization*, pp. 1–148. Available at: http://apps.who.int/bookorders. (Accessed: December 1, 2021).

Zhang, Yiye *et al.* (2021) “Development and validation of a machine learning algorithm for predicting the risk of postpartum depression among pregnant women,” *Journal of Affective Disorders*, 279, pp. 1–8. doi:10.1016/J.JAD.2020.09.113.

Zhang, Y. *et al.* (2021) “Ensemble Learning Based Postpartum Hemorrhage Diagnosis for 5G Remote Healthcare,” *IEEE Access*, 9, pp. 18538–18548. doi:10.1109/ACCESS.2021.3051215.

Zhang, Yawei *et al.* (2021) “Ensemble Learning Based Postpartum Hemorrhage Diagnosis for 5G Remote Healthcare,” *IEEE Access*, 9, pp. 18538–18548. doi:10.1109/ACCESS.2021.3051215.