
**Desarrollo de una aplicación de medición instantánea
para la prevención de conductas autolesivas en
adolescentes**

**Development of an instant measurement application for
the prevention of self-injurious behavior in adolescents**



**Trabajo de Fin de Máster
Curso 2023–2024**

Autor
Aldair Fredy Maldonado Honores

Director
Gonzalo Méndez
Pablo Gervás Gómez-Navarro

Desarrollo de una aplicación de medición instantánea para la prevención de conductas autolesivas en adolescentes

Development of an instant measurement application for the prevention of self-injurious behavior in adolescents

Trabajo de Fin de Máster en Ingeniería Informática
Departamento de Ingeniería del Software e Inteligencia Artificial

Autor
Aldair Fredy Maldonado Honores

Director
Gonzalo Méndez
Pablo Gervás Gómez-Navarro

Convocatoria: Junio 2024

Máster en Ingeniería Informática
Facultad de Informática
Universidad Complutense de Madrid

9 de junio de 2024

Dedicatoria

A mi familia, quienes me han brindado su apoyo en todo momento y me han animado a seguir con el proyecto y el máster durante estos años.

Agradecimientos

Quisiera agradecer a Enrique Martín Martín, profesor y director del máster de Ingeniería Informática, por ayudarme en los momentos dónde me había atascado y que se salían del tema principal del TFM.

También quisiera agradecer a mis directores del TFM, Gonzalo Méndez y Pablo Gervás, por intentar ayudarme con mis problemas del TFM, a pesar de las dificultades externas que se han presentado en el proyecto y que nos han afectado negativamente a todos.

Resumen

Desarrollo de una aplicación de medición instantánea para la prevención de conductas autolesivas en adolescentes

El problema del suicidio y las conductas autolesivas entre los jóvenes es una preocupación de salud que ha ganado creciente atención en los últimos años, gracias a que progresivamente ha habido mayor conciencia social sobre este tema.

Es por eso que su prevención se ha vuelto un verdadero reto entre los profesionales de salud.

Bajo esta premisa surge el proyecto SIVARIA, que busca detectar de cuanto antes posibles conductas autolesivas suicidas y no suicidas entre jóvenes adolescentes de entre 12 y 21 años.

Para ello, se propone desarrollar una aplicación que tratará de predecir posibles conductas autolesivas y que los profesionales tomen las medidas pertinentes para su tratamiento. Para realizar esta función, la aplicación se conectará internamente a un Sistema Experto que recibirá conjuntos de datos como datos de entrada, y devolverá una predicción de una posible conducta autolesiva que pueda tener el usuario. Estos datos estarían a disposición de profesionales (psicólogos, médicos, etc.), y, de forma más general, para los familiares.

Palabras clave

Sistema Experto, Python, entrenamiento, testeo, red de Bayes, dataset, modelo, métrica, variable, predicción

Abstract

Development of an instant measurement application for the prevention of self-injurious behavior in adolescents

The issue of suicide and self-harming behaviors among young people is a growing health concern in recent years, thanks to an increasing social awareness of this issue. This is why its prevention has become a significant challenge for healthcare professionals.

Under this premise, the SIVARIA project emerges, aiming to detect as early as possible both suicidal and non-suicidal self-harming behaviors among adolescents aged 12 to 21.

To achieve this goal, the project proposes the development of a algorithm that will predict potential self-harming behaviors, enabling professionals to take relevant measures for intervention. To perform this function, the application will internally connect to an Expert System that will receive datasets as input and return a prediction of a possible self-harming behavior the user may exhibit. These data would be accessible to professionals (psychologists, doctors, etc.) and, more broadly, to family members.

Keywords

Expert System, Python, training, testing, Bayes network, dataset, model, metric, variable, prediction

Índice

1. Introducción	1
1.1. Motivación	1
1.2. Objetivos	3
2. Introduction	5
2.1. Motivation	5
2.2. Goals	7
3. Estado del Arte	9
3.1. Aplicaciones similares	9
3.1.1. Prevensuic	9
3.1.2. Suicide Safety Plan	9
3.1.3. ReMinder	10
3.1.4. ISNISS	10
3.1.5. PAPAGENO	11
3.1.6. Tabla comparativa	11
4. Herramientas de desarrollo y tecnologías	15
4.1. Herramientas	15
4.1.1. Visual Studio Code	15
4.1.2. Github	16
4.1.3. Latex	16
4.1.4. Draw.io	17
4.1.5. Anaconda	17
4.2. Tecnologías y lenguajes	17
4.2.1. Python	17
5. Descripción del Trabajo	21
5.1. Investigación	21
5.1.1. Sistema Experto	21
5.1.2. Redes de Bayes	22

5.1.3.	Distribuciones de probabilidad condicional	24
5.2.	Diseño	25
5.2.1.	Diseño del modelo	25
5.3.	Desarrollo de los modelos	29
5.3.1.	Datasets	29
5.3.2.	Clasificación del modelo	30
5.3.3.	Entrenamiento	30
5.3.4.	Testeo del entrenamiento del modelo	30
5.4.	Ciclos de entrenamiento	32
5.4.1.	Primer ciclo de entrenamiento	32
5.4.2.	Segundo ciclo de entrenamiento	35
5.4.3.	Tercer ciclo de entrenamiento	37
5.4.4.	Cuarto ciclo de entrenamiento	40
5.4.5.	Quinto ciclo de entrenamiento	42
5.4.6.	Sexto ciclo de entrenamiento	45
5.4.7.	Análisis de los resultados	47
6.	Funcionamiento de la aplicación	49
6.1.	Actores de la aplicación	49
6.2.	Funcionalidades del código	50
6.2.1.	Establecimiento del tipo de modelo	50
6.2.2.	Establecimiento del tipo de puntuación	50
6.2.3.	Listado de archivos de guardado	51
6.2.4.	Información del archivo de guardado	51
6.2.5.	Eliminación de un archivo de guardado	51
6.2.6.	Entrenamiento del modelo	52
6.2.7.	Predicción	52
7.	Conclusiones y Trabajo Futuro	57
7.1.	Conclusiones	57
7.2.	Trabajo a futuro	57
8.	Conclusions and Future Work	59
8.1.	Conclusions	59
8.2.	Future work	59
Bibliografía		61
A. Guía de instalación		63
A.1.	Contenido del repositorio	63
A.1.1.	Scripts	63
A.2.	Instrucciones para ejecutar el script	64
A.2.1.	Entornos de Python con Anaconda	65

A.2.2. Conexión del entorno virtual con Visual Studio Code	65
A.2.3. Instalación de los paquetes de Python	66

Índice de figuras

1.1. Causas de muertes externas en España 2008-2022. Fuente: INE (2024)	1
1.2. Causas de muertes externas en España 2013-2022 entre los jóvenes 15-19 años. Fuente: INE (2024)	2
1.3. Causas de muertes externas en España 2013-2022 entre los jóvenes 20-24 años. Fuente: INE (2024)	3
1.4. Número de suicidios en España entre los años 2000 y 2022. Fuente: Fundación española para la prevención del suicidio (2022), Estadísticas de Defunción por Causa de Muerte 2022. INE (2024)	4
2.1. Causes of external death in Spain from 2008 to 2022. Source: INE (2024)	5
2.2. Causes of external death in Spain from 2013 to 2022 among young people aged 15 and 19 years old. Source: INE (2024)	6
2.3. Causes of external death in Spain from 2013 to 2022 among young people aged 20 and 24 years old. Source: INE (2024)	7
2.4. Number of suicides in Spain between 2000 and 2022. Source: Fundación española para la prevención del suicidio (2022), Statistics of Causes of Death in 2022 of people of all ages. INE (2024)	8
3.1. Interfaces de Prevensuic	10
3.2. Interfaces de Suicide Safety Plan.	10
3.3. Interfaz web de ReMinder.	11
3.4. Interfaz web de ISNISS.	11
3.5. Interfaz web de PAPAGENO.	12
4.1. Interfaz web de Draw.io	17
4.2. Interfaz web de Anaconda	18
5.1. Estructura del Sistema Experto	22
5.2. Página inicial del sitio web de SIVARIA	22
5.3. Encuesta a jóvenes de más de 16 años de SIVARIA	23
5.4. Ejemplo de un Diagrama de Bayes. Fuente, Javatpoint	23
5.5. Red de Bayes del modelo del Autoinforme (I)	28
5.6. Red de Bayes del modelo del Autoinforme (II)	28

5.7.	Red de Bayes del modelo del Autoinforme (III)	29
5.8.	Red de Bayes del modelo de las familias	29
5.9.	Red de Bayes del modelo de los profesionales	29
5.10.	Ejemplo del directorio de archivos de guardado con el modelo del Autoinforme	30
5.11.	Matriz de confusión del Autoinforme al final del primer ciclo	33
5.12.	Matriz de confusión de las familias al final del primer ciclo	34
5.13.	Matriz de confusión de los profesionales al final del primer ciclo	34
5.14.	Matriz de confusión del Autoinforme al final del tercer ciclo	37
5.15.	Matriz de confusión de las familias al final del tercer ciclo	38
5.16.	Matriz de confusión de los profesionales al final del tercer ciclo	39
5.17.	Matriz de confusión del Autoinforme al final del quinto ciclo	42
5.18.	Matriz de confusión de las familias al final del quinto ciclo	43
5.19.	Matriz de confusión de los profesionales al final del quinto ciclo	44
6.1.	Estructura del Controlador con el Sistema Experto	50
6.2.	Diagrama de secuencia del comando para establecer el tipo de modelo . . .	51
6.3.	Diagrama de secuencia del comando para establecer el tipo de puntuación .	52
6.4.	Diagrama de secuencia del comando para listar los archivos de guardado .	53
6.5.	Diagrama de secuencia del comando para mostrar la información del archivo de guardado	53
6.6.	Diagrama de secuencia del comando para eliminar un archivo de guardado .	54
6.7.	Diagrama de secuencia del comando para entrenar al modelo	54
6.8.	Diagrama de secuencia del comando para realizar una predicción	55
A.1.	Entorno por defecto de Anaconda	65
A.2.	Página de Environment de Anaconda y botón de creación de entornos . .	65
A.3.	Ventana de creación de nuevo entorno de Python en Anaconda	65
A.4.	Panel de selección de entorno de Python en Visual Studio Code	66
A.5.	Comando Help del Sistema Experto	67

Índice de tablas

3.1. Tabla comparativa de aplicaciones de prevención contra el suicidio. Última actualización: mayo de 2024	12
3.2. Continuación de la Tabla comparativa 3.1	13
4.1. Popularidad de los lenguajes de programación, Fuente: PYPL (2023)	15
5.1. CPD de la variable <i>Alarm</i>	25
5.2. Parámetros de entrenamiento del primer ciclo	32
5.3. Comparativa de las puntuaciones del modelo del Autoinforme. Primer ciclo	33
5.4. Comparativa de las puntuaciones del modelo de las familias. Primer ciclo	33
5.5. Comparativa de las puntuaciones del modelo de los profesionales. Primer ciclo	33
5.6. Parámetros de entrenamiento del segundo ciclo	35
5.7. Comparativa de las puntuaciones del modelo del Autoinforme. Segundo ciclo	35
5.8. Comparativa de las puntuaciones del modelo de las familias. Segundo ciclo	35
5.9. Comparativa de las puntuaciones del modelo de los profesionales. Segundo ciclo	36
5.10. Parámetros de entrenamiento del tercer ciclo	37
5.11. Comparativa de las puntuaciones del modelo del Autoinforme. Tercer ciclo	37
5.12. Comparativa de las puntuaciones del modelo de las familias. Tercer ciclo	38
5.13. Comparativa de las puntuaciones del modelo de los profesionales. Tercer ciclo	38
5.14. Parámetros de entrenamiento del cuarto ciclo	40
5.15. Comparativa de las puntuaciones del modelo del Autoinforme. Cuarto ciclo	40
5.16. Comparativa de las puntuaciones del modelo de las familias. Cuarto ciclo	40
5.17. Comparativa de las puntuaciones del modelo de los profesionales. Cuarto ciclo	41
5.18. Parámetros de entrenamiento del quinto ciclo	42
5.19. Comparativa de las puntuaciones del modelo del Autoinforme. Quinto ciclo	42
5.20. Comparativa de las puntuaciones del modelo de las familias. Quinto ciclo	43
5.21. Comparativa de las puntuaciones del modelo de los profesionales. Quinto ciclo	43
5.22. Parámetros de entrenamiento del sexto ciclo	45
5.23. Comparativa de las puntuaciones del modelo del Autoinforme. Sexto ciclo	45
5.24. Comparativa de las puntuaciones del modelo de las familias. Sexto ciclo	45

5.25. Comparativa de las puntuaciones del modelo de los profesionales. Sexto ciclo	46
5.26. Tabla comparativa de las métricas del Autoinforme	47
5.27. Tabla comparativa de las métricas del modelo de las familias	47
5.28. Tabla comparativa de las métricas del modelo de los profesionales	48

Capítulo 1

Introducción

1.1. Motivación

El suicidio es un problema de salud a nivel global (wor (2014)), siendo la principal causa de muerte no natural entre jóvenes y adultos (Al-Halabí y Fonseca-Pedrero (2021), Gore et al. (2011)). Este tema del suicidio ha ido adquiriendo cada vez más atención dentro del ámbito de la salud, y su prevención ha presentado un reto para todos los profesionales, ya que dicha problemática afecta tanto a los propios individuos, como también a los familiares y amigos cercanos.

Desgraciadamente, España no ha sido la excepción. Según las estadísticas del INE, el suicidio sigue siendo la principal causa de muerte externa en España desde 2008 (1.1) y una de las principales de los jóvenes de entre 15 y 24 años (Figuras 1.2 y 1.3)(Eterovic).

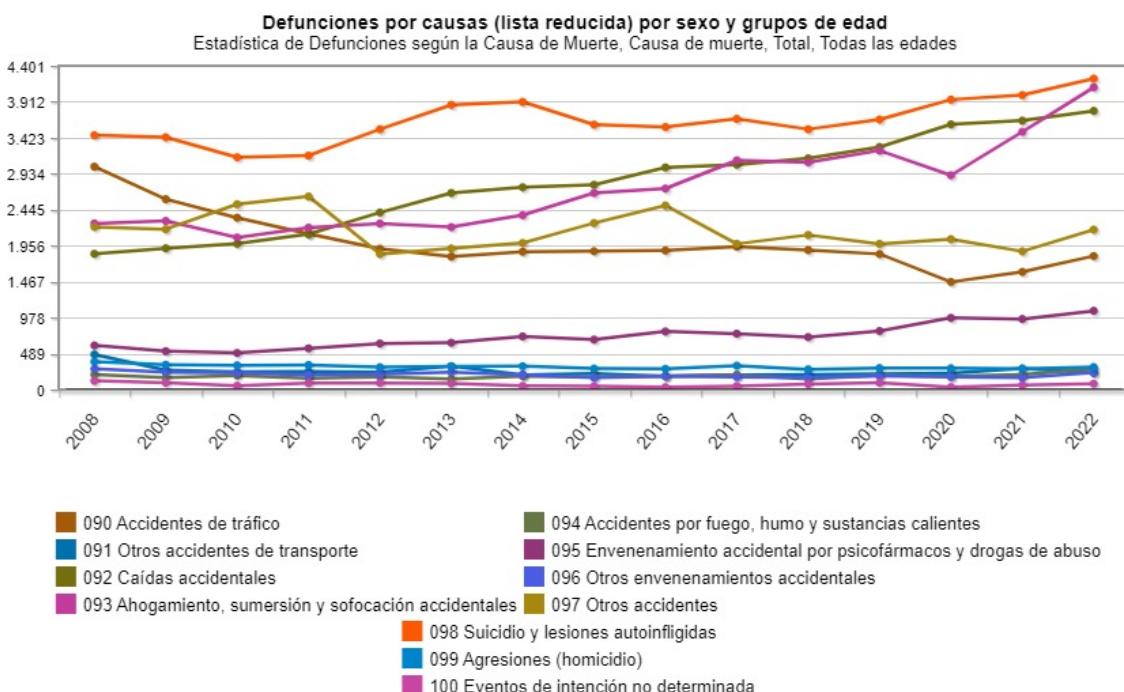


Figura 1.1: Causas de muertes externas en España 2008-2022. Fuente: INE (2024)

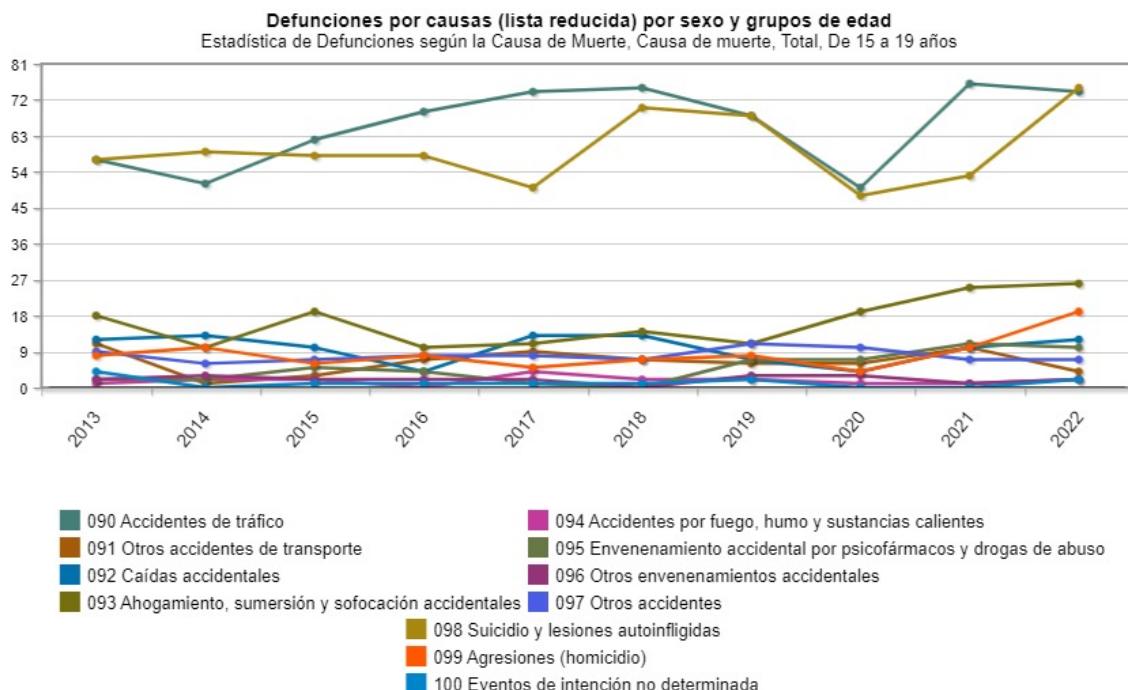


Figura 1.2: Causas de muertes externas en España 2013-2022 entre los jóvenes 15-19 años.
Fuente: INE (2024)

No sólo eso, sino que además el número de suicidios en España ha aumentado desde 2018, marcando cifras históricas (1.4).

A estas estadísticas también se le acompañan encuestas realizadas a grupos de jóvenes españoles, como el que se llevó a cabo por Muela et al. (2024).

Esto evidencia un claro aumento de suicidios y de las conductas autolesivas en los adolescentes en España.

Es por ello que en los últimos años, se están centrando los esfuerzos y los recursos económicos en la creación de programas de prevención del suicidio. En estos planes, es fundamental realizar diagnósticos lo más pronto posible para detectar a tiempo síntomas mentales en la persona, y por lo tanto, poder ofrecerle un tratamiento. El problema es que muchas veces, las personas con riesgo de adoptar estas conductas suicidas y autolesivas no son conscientes de que los padecen o que pueden padecerlos, por lo que no acuden a los planes de prevención. Esto trae, como consecuencia, el agravamiento del problema, hasta acabar en un desenlace fatal. Es importante el poder informar a la persona a tiempo sobre su estado de salud mental de forma automática, rápida y eficaz, para que así sea consciente del riesgo al que está sometido, y por lo tanto, pueda pedir ayuda con anticipación.

Ante este panorama, se busca desarrollar herramientas para prevenir este tipo de conductas en la juventud española. Es aquí donde comienza a jugar un papel fundamental la inteligencia artificial.

De acuerdo a al artículo de Fonseka et al. (2019), la inteligencia artificial se está utilizando cada vez más en el campo de la salud mental, siendo de gran utilidad para la prevención del suicidio y dar soporte a los profesionales para conseguir una evaluación exacta del riesgo de suicidio. De esta manera, se pueden tratar a los potenciales pacientes que posean estas conductas autolesivas o no autolesivas en etapas muy tempranas del desarrollo de estos comportamientos.

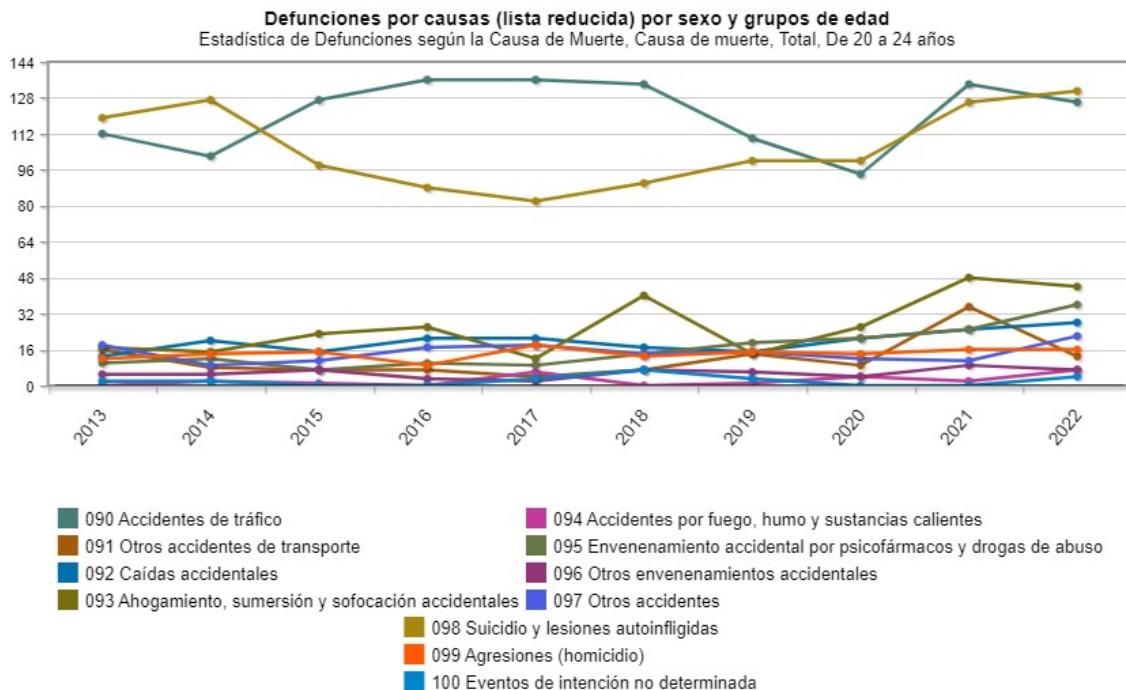


Figura 1.3: Causas de muertes externas en España 2013-2022 entre los jóvenes 20-24 años.
Fuente: INE (2024)

1.2. Objetivos

Es precisamente por este aumento del uso de la IA en el campo de la salud en estos últimos años, se ha propuesto este Trabajo de Fin de Máster. El objetivo es desarrollar un algoritmo para medir y prevenir las conductas autolesivas suicidas y no suicidas en jóvenes adolescentes.

Para ello, se deben realizar una serie de formularios hechos por profesionales a los jóvenes. Estas respuestas serán tratadas de forma anónima y se enviarán al modelo que se encargará de predecir posibles conductas autolesivas y no autolesivas que podría tener el usuario en base a las respuestas obtenidas del formulario.

Para ello, se propone la implementación de un Sistema Experto que se encargue del proceso de predicción, diagnóstico y evaluación, simulando así las decisiones que tomaría un profesional real.

Con estas premisas, se intentan fijar esta serie de objetivos:

- Diseñar un modelo inicial de Sistema Experto.
- Pasar el diseño del modelo a código.
- Entrenar al modelo con una serie de datasets proporcionados por parte de los profesionales involucrados en este proyecto.
- Testear el modelo con otros datos de prueba. Se cuantifica el rendimiento del modelo, se mide y en base a los resultados obtenidos, se calibra el modelo. Más adelante se explicará las distintas formas de medir el rendimiento de un modelo probabilístico.

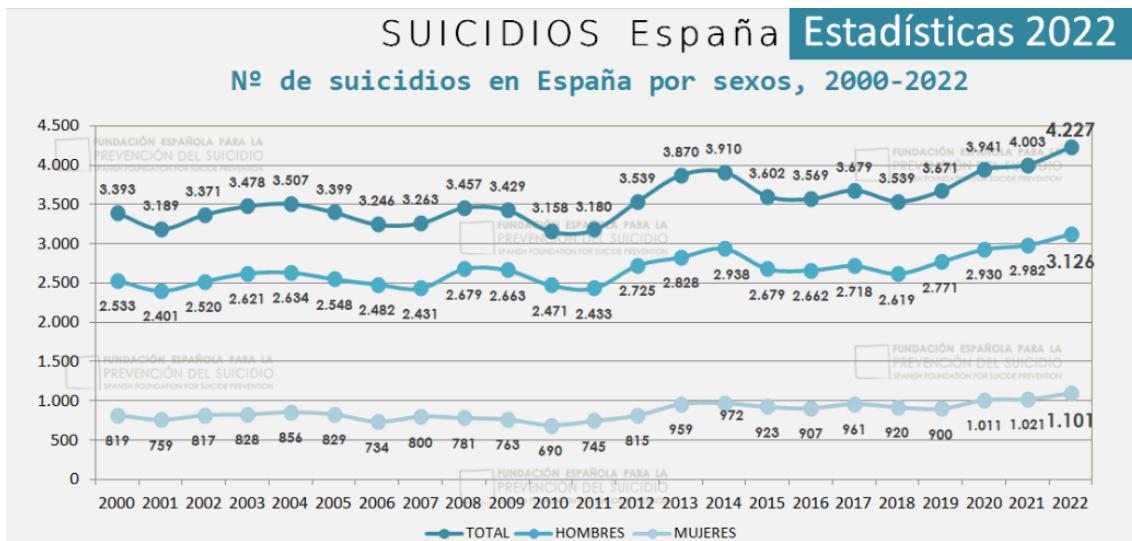


Figura 1.4: Número de suicidios en España entre los años 2000 y 2022. Fuente: Fundación española para la prevención del suicidio (2022), Estadísticas de Defunción por Causa de Muerte 2022. INE (2024)

- Se repite el proceso de entrenamiento y testeo del modelo hasta obtener un resultado lo suficientemente óptimo para que el modelo pueda realizar predicciones de casos reales.

Capítulo 2

Introduction

2.1. Motivation

Suicide is a global health problem (wor (2014)), being the leading cause of unnatural death among young people and adults (Al-Halabí y Fonseca-Pedrero (2021), Gore et al. (2011)). The issue of suicide has been gaining increasing attention within the health field, and its prevention has been a challenge for health professionals, as the problem affects both the individuals themselves, as well as their family members and close friends.

Unfortunately, Spain has been no exception. According to INE statistics, suicide continues to be the leading cause of external death in Spain since 2008 (2.1) and one of the main ones among young people aged 15 and 24 years old (Figures 2.2 and 2.3)(Eterovic).

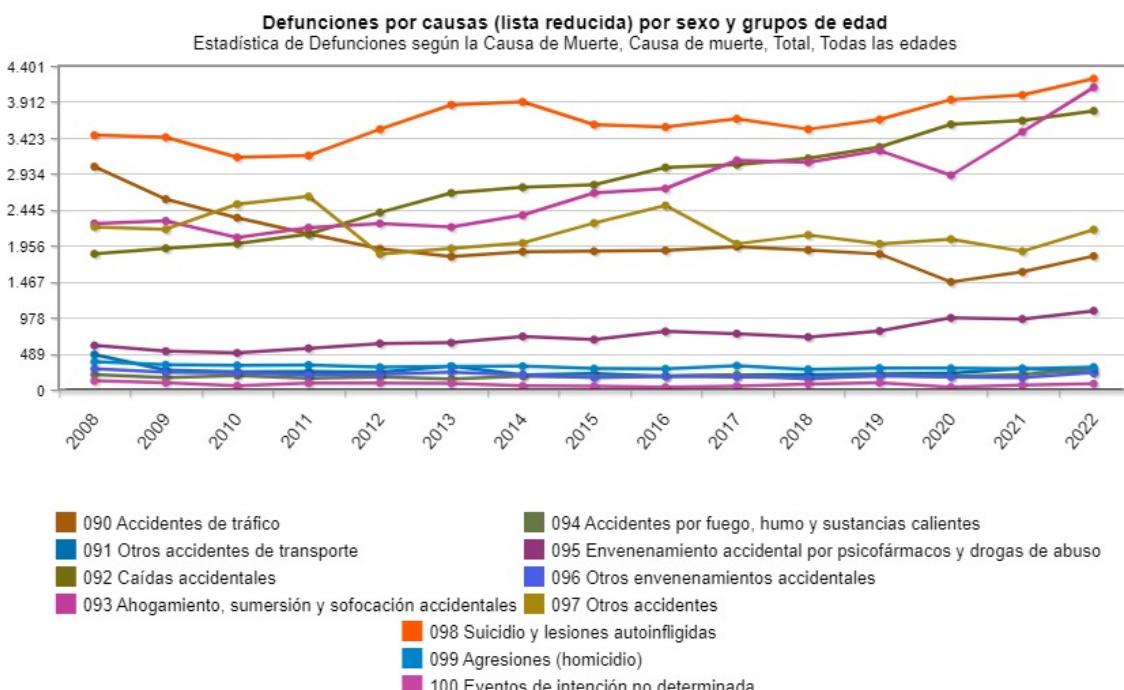


Figura 2.1: Causes of external death in Spain from 2008 to 2022. Source: INE (2024)

Furthermore, the number of suicides in Spain has increased since 2018, marking histo-

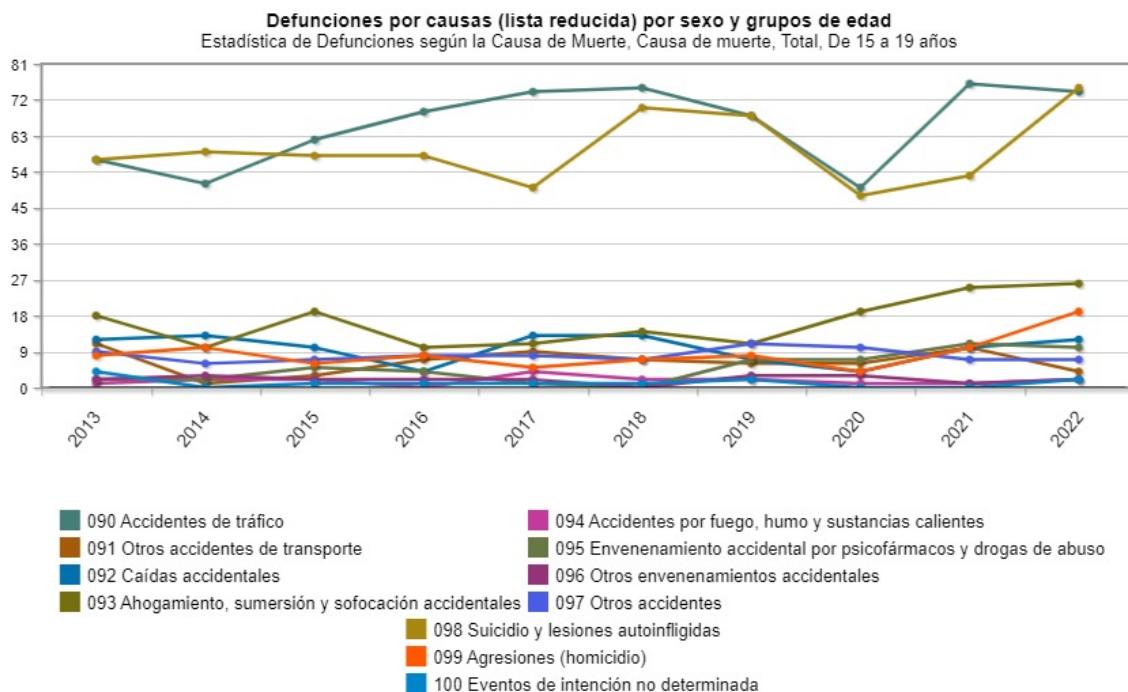


Figura 2.2: Causes of external death in Spain from 2013 to 2022 among young people aged 15 and 19 years old. Source: INE (2024)

rical figures (2.4).

These statistics are also accompanied by surveys conducted among groups of young Spaniards, such as the one conducted by Muela et al. (2024).

This provides evidence of a clear increase in suicides and self-injurious behaviors among adolescents in Spain.

For this reason, in recent years, efforts and economic resources have been focused on the creation of suicide prevention programs.

In these plans, it is essential to make diagnoses as early as possible to detect mental symptoms in the person, and therefore, to be able to offer treatment. The problem is that people at risk of adopting these suicidal and self-injurious behaviors are often unaware that they suffer from them or that they may suffer from them, so they do not attend prevention plans. As a consequence, this leads to a worsening of the problem, until it ends in a fatal outcome. It is important to be able to inform the person in time about his or her mental health condition automatically, quickly and effectively, so that he or she is aware of the risk to which he or she is subjected, and therefore can ask for help in advance.

Against this background, it is essential to develop strategies, tools and methods to prevent this type of behaviour among Spanish youth. This is where artificial intelligence begins to play a fundamental role.

According to the following article (Fonseka et al. (2019)), artificial intelligence is being used progressively in the field of mental health, being useful in suicide prevention and providing support to professionals to achieve an effective and efficient diagnosis and assessment. In this way, potential patients with these self-injurious and non-self-injurious behaviours can be treated in early stages.

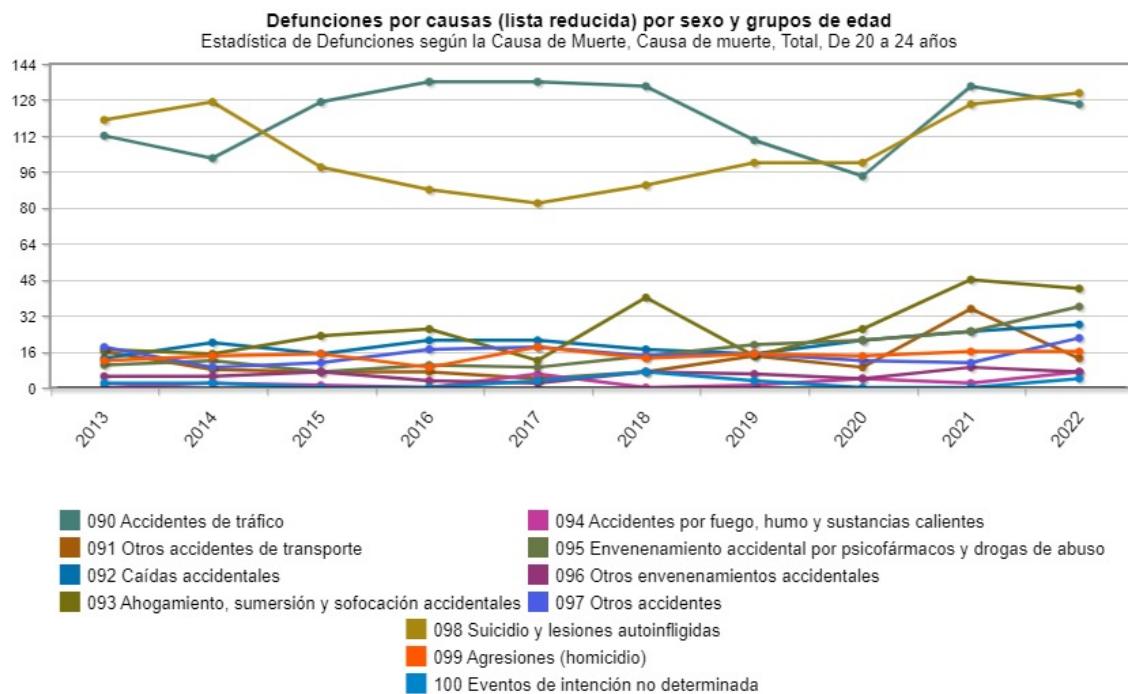


Figura 2.3: Causes of external death in Spain from 2013 to 2022 among young people aged 20 and 24 years old. Source: INE (2024)

2.2. Goals

It is precisely because of this increase in the use of AI that it has been proposed to develop an algorithm to measure and prevent suicidal and non-suicidal self-injurious behaviour in young adolescents.

To do this, a series of forms are to be filled out by professionals with young people. These answers will be treated anonymously and will be sent to the model that predicts the possible self-harming and non-self-harming behaviours that the user could have based on the answers obtained from the form.

In order to create this model, the implementation of an Expert System was proposed. This Expert System will be in charge of the prediction, diagnosis and evaluation process, thus simulating the decisions that a real professional would take.

Based on these premises, the following initial individual goals were set:

- To design an initial model of the Expert System. The types of systems that exist and the one selected for this project will be explained later on.
- To translate the design to code. A programming language and a set of libraries will be selected to implement the model.
- To train the model with datasets provided by the experts that are involved in this project.
- To test the model with part of the provided. The testing will be quantified in a score that will be based on the results obtained from the testing. This score will be interpreted in order to apply possible improvements to the model. The different methods to obtain this score will be explained later on.



Figura 2.4: Number of suicides in Spain between 2000 and 2022. Source: Fundación española para la prevención del suicidio (2022), Statistics of Causes of Death in 2022 of people of all ages. INE (2024)

- To Repeat training and testing process until getting an acceptable score from the model. That means that it is good enough to make predictions in real cases.

Capítulo 3

Estado del Arte

En este capítulo se expondrá la importancia del uso de la inteligencia artificial en el campo de la salud mental, así como ejemplos y referencias relacionados con el proyecto.

3.1. Aplicaciones similares

Antes de desarrollar la aplicación, se entregaron un documento con una lista de aplicaciones analizadas por los directores que se están desarrollando en este campo.

De la lista completa de aplicaciones, se han desatulado cinco de ellas, ya que eran las que estaban más relacionadas con las intenciones de nuestra aplicación, además de que eran las que proporcionaban una descripción más completa de sus funcionalidades y objetivos.

3.1.1. Prevensuic

Prevensuic¹ es un programa creado por la *Fundación Española para la Prevención del Suicidio*, destinado a la prevención, divulgación y la formación acerca del suicidio. Dicho programa cuenta con un sitio web y una aplicación móvil (disponible tanto en Android como en iOS) para ser accesible a un mayor número de personas, tanto adultos como jóvenes.

3.1.2. Suicide Safety Plan

Suicide Safety Plan² es una aplicación móvil desarrollada por Inquiry Health LLC y está disponible tanto en Android como en iOS. La aplicación es una especie de aplicación recordatorio para que el usuario pueda establecer qué cosas pueden afectar a su salud mental, e incluye información para ayudar a la persona a evaluar su propio estado mental. Además, incluye una serie de teléfonos de contacto en caso de necesitar ayuda.

¹<https://www.prevensuic.org/>

²https://play.google.com/store/apps/details?id=com.moodtools.crisis.app&hl=en_US



Figura 3.1: Interfaces de Prevensuic

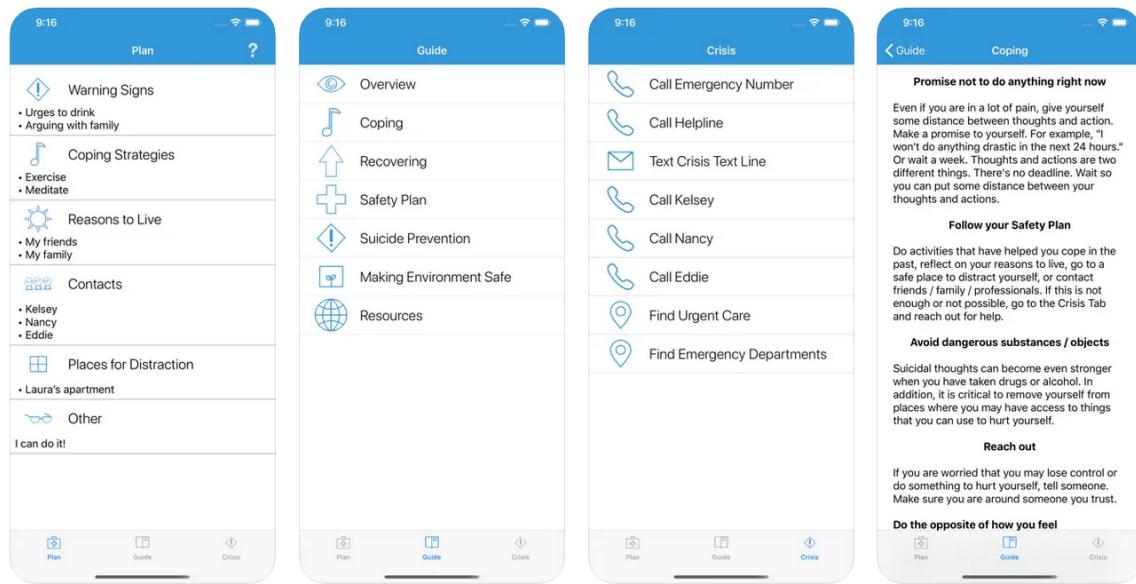


Figura 3.2: Interfaces de Suicide Safety Plan.

3.1.3. ReMinder

ReMinder³ es un programa inglés dirigido a gente adulta y que consta de un cuestionario de 10 preguntas.

3.1.4. ISNISS

En este caso, ISNISS⁴ es un sitio web que se encarga de dar información sobre cómo realizar una evaluación correcta del riesgo de suicidio en un paciente, aunque no la realiza el propio sitio web, aparte de otra serie de recomendaciones.

³<https://www.emhprac.org.au/directory/reminder-suicide-safety-plan/>

⁴<https://www.isniss.es/>



About ReMinder Suicide Safety Plan app

The ReMinder Suicide Safety Plan app was developed by Suicide Call Back Service to provide a step-by-step guide to creating a suicide safety plan which users can edit and share with their supporters at any time. Designed as a self-managed resource that can be adopted as part of a coping strategy, providing a reminder for users of reasons to live, and connect them with people and services that can help during tough times.

The ReMinder app includes:

- Access to helplines and emergency service numbers
- Create a 'team' of personal support contacts
- Store favourite images
- Change the ReMinder theme for a calming influence
- K10 questionnaire to determine to what extent you have experienced depression or anxiety over the past month
- Latest tweets from the Suicide Call Back Service for further information and advice on suicide safety

Figura 3.3: Interfaz web de ReMinder.

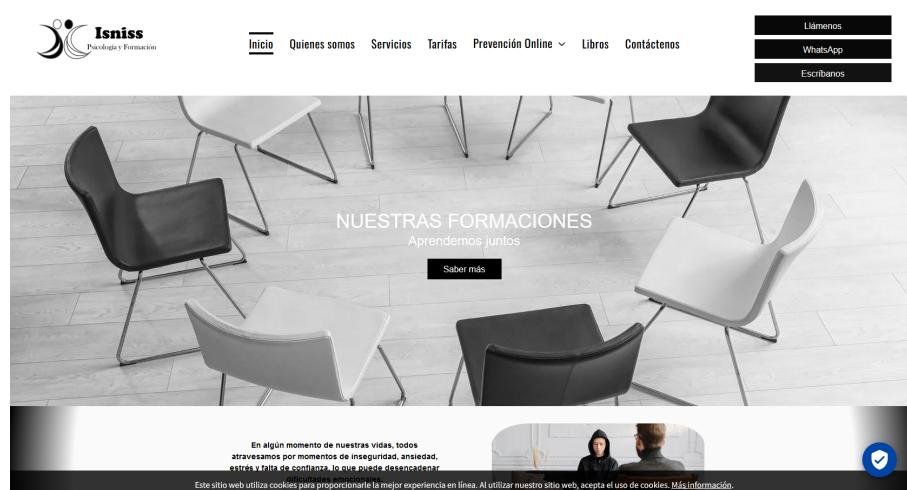


Figura 3.4: Interfaz web de ISNISS.

3.1.5. PAPAGENO

Similar a ISNISS, PAPAGENO⁵ es un blog que ofrece información sobre la postvención y recomendaciones para la evaluación por los sanitarios.

3.1.6. Tabla comparativa

Haciendo una comparación entre todas ellas, se llegó a las siguientes tablas 3.1 y 3.2.

De todas las aplicaciones de las Tablas 3.1 y 3.2, uno de los más destacados es la aplicación de Prevensuic.

Una de las aplicaciones que más fueron analizadas fue Prevensuic. Se le considera una aplicación o programa muy completa y psicoeducativo, es decir, que es capaz de brindar información a las personas que están sufriendo de un trastorno psicológico.

Sin embargo, Prevensuic presenta dos grandes inconvenientes:

⁵<https://papageno.es/about/quienes-somos>



Figura 3.5: Interfaz web de PAPAGENO.

	Prevensuic	Suicide Safety Plan	ReMinder
Tipo de aplicación	Web y Móvil	Móvil	Móvil
Población objetivo	Adultos	Adultos	Adolescentes
Idioma	Español	Inglés	Inglés
Evaluación del riesgo	X	X	X
Estrategias de afrontamiento	✓	✓	X
Psicoeducación	✓	✓	✓
Afectados	✓	✓	✓
Profesionales	✓	X	X
Familiares	✓	X	X

Tabla 3.1: Tabla comparativa de aplicaciones de prevención contra el suicidio. Última actualización: mayo de 2024

- La aplicación no es capaz de realizar una evaluación del riesgo de suicidio.
- La versión de móvil de Prevensuic no está disponible para las últimas versiones de (Android 14, por ejemplo). Esto se debe a que no se actualiza de manera recurrente. Su última actualización data del 11 de abril de 2019. Esto supone una disminución considerable de la accesibilidad de la aplicación y, por lo tanto, del programa en general.

Desde SIVARIA, se busca proporcionar una herramienta web y móvil que pueda proporcionar una evaluación automática del riesgo de suicidio en base a las respuestas de un cuestionario. Dicha aplicación se espera que reciba actualizaciones frecuentemente, además de que estará dirigida principalmente a jóvenes, pero también pueden acceder profesionales y los familiares y conocidos del paciente.

	ISNISS	PAPAGENO
Tipo de aplicación	Web	Web
Población objetivo	Adolescentes	Adultos
Idioma	Español	Español
Evaluación del riesgo	✗	✗
Estrategias de afrontamiento	-	-
Psicoeducación	✗	-
Afectados	✗	✓
Profesionales	✗	✓
Familiares	✗	✓

Notas: "–": no se sabe si la característica está en la aplicación.

Tabla 3.2: Continuación de la Tabla comparativa 3.1

Capítulo 4

Herramientas de desarrollo y tecnologías

4.1. Herramientas

4.1.1. Visual Studio Code

Visual Studio Code es un editor de código fuente, gratuito, de código abierto y multiplataforma, ya que fue creado por Microsoft para Windows, Linux, MacOS y Web. Cuenta con herramientas internas de depuración, resaltado de sintaxis, control de versiones de Git, autocompletado de código, entre otros.

Se ha propuesto que se utilizará esta herramienta para desarrollar el prototipo de la aplicación en React.

Cabe resaltar que, en un principio, se planeaba utilizar Android Studio como entorno de desarrollo, y programar la aplicación con Java, por su popularidad en el desarrollo de aplicaciones web y móviles, como se puede observar en la Tabla 4.1.

Sin embargo, Android Studio facilitaba exclusivamente el desarrollo de aplicaciones compatibles con dispositivos móviles Android en Java. Por ende, si se hubiera persistido con esta propuesta, habría existido el riesgo de que la aplicación presentara errores durante la ejecución o, incluso, que no pudiera ser ejecutada en otros sistemas operativos distintos, como iOS.

Otra de las razones por las que finalmente se ha optado por el uso de VS Code fue para

Rank	Lenguaje de programación	Cuota de mercado
1	Python	28.43 %
2	Java	16.04 %
3	JavaScript	8.72 %
4	C/C++	6.65 %
5	C#	6.63 %
6	R	4.63 %
7	PHP	4.45 %
8	TypeScript	2.96 %
9	Swift	2.71 %
10	Rust	2.53 %

Tabla 4.1: Popularidad de los lenguajes de programación, Fuente: PYPL (2023)

poder desarrollar scripts en Python. Al ser de código abierto, ha permitido que se puedan desarrollar diferentes plugins open-source para facilitar el desarrollo de Python y React en este editor.

4.1.2. Github

Todas las novedades y cambios en el código del Sistema Experto, aplicación o en la documentación se añaden a un repositorio de Github. Github es una plataforma en línea para alojar una gran diversidad de proyectos. El enlace al repositorio de SIVARIA es el siguiente:

<https://github.com/NILGroup/TFM-2324-SIVARIA>

Entre las características más notables de Github, destaca su sistema de control de versiones basado en Git. Git facilita el seguimiento de todos los cambios realizados en el código fuente, abarcando todas las ramas del repositorio. Esta capacidad resulta sumamente útil para administrar el desarrollo del software de manera eficiente y efectiva (Wikipedia (2024)).

Con respecto al flujo de trabajo, se tenía pensado seguir el flujo de trabajo *GitFlow* que ya está implementado en Github, en donde se parte de una *main*, que simulará ser una rama de producción. A partir de esta rama, deriva otra nueva rama *develop*. Finalmente, a partir de esta rama derivan una serie de subramas llamadas *feature*, una por cada tarea o corrección que surja.

De esta manera, cada tarea que se vaya a desarrollar se hará en una rama nueva *feature*, y cuando dicha tarea se termine, se mergea con la rama *develop*, y posteriormente con la rama *main*, liberando las tareas realizadas para que los directores o cualquier usuario pueda testear las nuevas implementaciones.

Siguiendo esta metodología, podemos tener el trabajo bien diferenciado por cada tarea a realizar, además de que evita el riesgo de presentar a los usuarios finales código defectuoso o con errores.

Por otro lado, en local se utilizará Github Desktop para subir todos los cambios en local al repositorio remoto de Github.

Github Desktop es una herramienta local que proporciona una interfaz de usuario que permite a usuarios convencionales poder gestionar los repositorios remotos sin necesidad de conocer en profundidad los comandos Git necesarios.

4.1.3. Latex

LATEX(Urban, 1986), creado y lanzado inicialmente por Leslie Lamport en los años 1980, es un sistema open-source de composición de texto ampliamente utilizado en la redacción de documentos, textos y artículos académicos y científicos. Una de las características más destacadas de Latex es que se basan en comandos de texto, que a su vez provienen del lenguaje de comandos original TEX, a diferencia de un procesador de textos convencional, que utilizan un enfoque visual. Esto nos permite escribir caracteres especiales y funciones matemáticas y, además de enumerar más fácilmente imágenes y tablas.

Esta herramienta es la que se ha utilizado para redactar esta memoria en su totalidad.

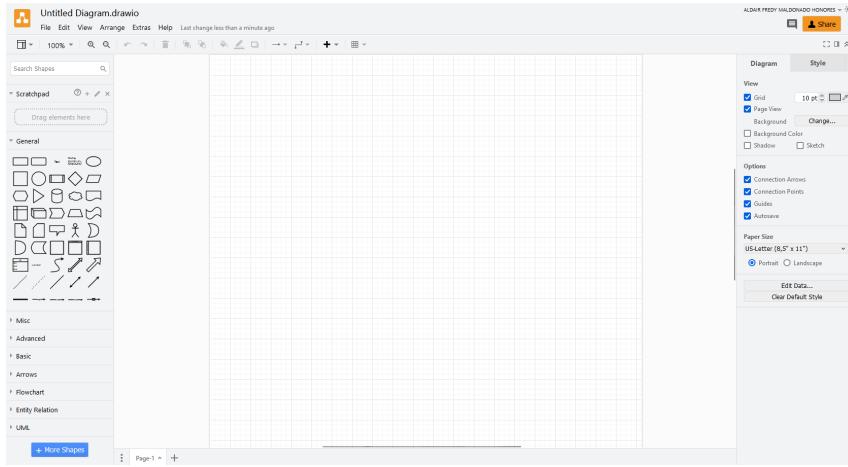


Figura 4.1: Interfaz web de Draw.io

4.1.4. Draw.io

Draw.io es una herramienta online de dibujos gráfico que es utilizado para crear diagramas de flujo, organigramas, diagramas UML o de red de forma rápida y sencilla. Es una herramienta gratuita que se puede vincular con Google Drive, de tal forma que podemos guardar el archivo draw.io en la nube. Se ha utilizado para diseñar y exportar los diagramas de Bayes en su totalidad.

4.1.5. Anaconda

Anaconda es una plataforma de distribución libre y abierta para los lenguajes de Python y R. Se utiliza en la ciencia de datos y en el aprendizaje automático de modelos y algoritmos.

Desde esta herramienta, se pueden crear entornos virtuales de Python para el desarrollo de scripts y la administración de paquetes mediante el sistema de gestión de paquetes *conda*.

Además, también desde su interfaz se puede acceder a herramientas internas de Anaconda, como Jupyter Notebook, entre otros, como se puede observar en la Figura 4.2. Es precisamente dicha herramienta la que se ha utilizado para la creación de cuadernos donde se ejemplifica el entrenamiento, predicción y testeо del modelo del Sistema Experto.

4.2. Tecnologías y lenaguajes

4.2.1. Python

Se decidió utilizar el lenguaje de programación Python para desarrollar el Sistema Experto de la aplicación, concretamente, la versión de Python utilizada es la 3.9 (Python (2020)).

Los motivos de su elección se deben a la gran variedad de bibliotecas de código abierto que ofrece Python para el aprendizaje automático, además de la manipulación y visualización de datos.

Por otro lado, Python es fácilmente escalable, es decir, se puede utilizar tanto en proyectos pequeños como grandes. Esto se debe a que Python da soporte a tecnologías que permiten el procesamiento distribuido, como por ejemplo, Apache Spark.

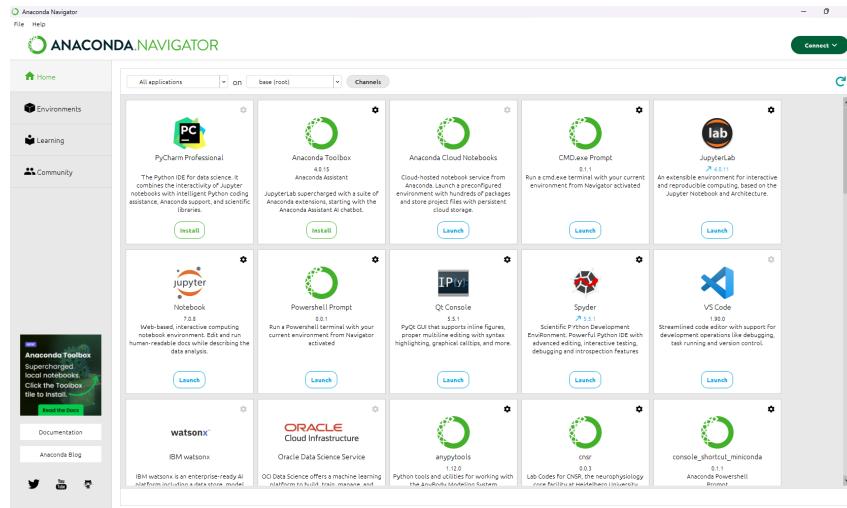


Figura 4.2: Interfaz web de Anaconda

4.2.1.1. Bibliotecas utilizadas

Se han utilizado las siguientes bibliotecas de código libre de Python:

- **Pandas**: ofrece estructuras de datos para poder almacenar el contenido de los datasets. En este caso, los datos se almacenarán en dataframes, que son estructuras de datos similares a las tablas de las bases de datos SQL. Por lo tanto, nos permitirán realizar consultas, inserciones, borrados y modificaciones sobre ellas de forma bastante intuitiva.
- **Scikit-learn**: incluye algoritmos de clasificación, regresión y análisis de grupos útiles para el desarrollo y evaluación de modelos de aprendizaje automático, aunque también se utilizan en la minería y análisis de datos.
- **Matplotlib**: permite crear gráficos estáticos, interactivos y animaciones de alta calidad. Es una biblioteca muy utilizada principalmente por su capacidad para generar una variedad de tipos de gráficos y visualizaciones, además de que permite exportarlos en formato PNG, JPEG, PDF y SVG. Se utilizará para mostrar la matriz de confusión de los entrenamientos realizados.
- **Sys**: librería instalada por defecto en el entorno virtual que se va a usar para obtener los parámetros enviados a través del comando de Python.
- **Os**: librería instalada previamente que permite ejecutar comandos del sistema desde un programa de Python. Se va a utilizar para revisar si existen directorios, en caso contrario, se crean, además de la eliminación de los archivos de guardado de los modelos.
- **Pickle**: librería que implementa una serie de herramientas que permiten serializar y deserializar una estructura de Python. Destaca un método *dump*, empleado para volcar la serialización sobre un archivo cuyo tipo debe especificar el programador. Se va a utilizar para guardar y cargar las variables en un archivo de configuración en el controlador, así se puede saber qué tipo de modelo se tiene que cargar y qué tipo de puntuación se ha escogido para el testeo del entrenamiento. Además, también se empleará en el guardado del modelo entrenado en un archivo de guardado dentro de

la carpeta `configFiles/<tipo_del_modelo>`. De esta manera, podemos almacenar la versión más reciente del modelo y cargarlo cuando se necesite.

- **Datetime**: librería que permite obtener la fecha y hora del sistema. Se utiliza durante la creación el entrenamiento del modelo, para obtener la fecha y hora del sistema en ese momento y se pondrá junto al nombre del nuevo archivo de guardado del modelo. Esto nos permitirá distinguir entre varias versiones del modelo.
- **Math**: librería que permite acceder a operaciones matemáticas definidas en el estándar de C. Se utiliza para la conversión del tamaño del archivo de guardado del modelo a KiB, MiB, GiB, etc.
- **NumPy**: paquete que proporciona una serie de herramientas para manejar matrices de forma eficiente. Se va a utilizar principalmente para calcular el número de verdaderos/falsos positivos y negativos, y para el entrenamiento y testeo de los modelos dentro del Sistema Experto.
- **Copy**: librería que permite realizar copias profundas (*deep copy*) de objetos de Python. Se utiliza para crear una copia profunda del modelo antes de entrenar para el testeo.

Cabe destacar que antes de Scikit-learn, se estudiaron la posibilidad de utilizar otros paquetes para la gestión de la red de Bayes. Estos son pgmpy(Ankan y Panda (2015)), pybnesian (Atienza et al. (2022)) y pomegranate(Schreiber (2014)).

En el caso de los pybnesian y pomegranate, se descartaron por la complejidad de su sintaxis. Con respecto a pgmpy, se llegó a desarrollar una implementación alternativa del modelo en pgmpy. Sin embargo, presentaba severos problemas con la capacidad de memoria local para crear el modelo. Esto provocaba que el proceso de entrenamiento tardase demasiado tiempo, y los procesos de testeo y predicción directamente no devolvía ningún tipo de respuesta.

Capítulo 5

Descripción del Trabajo

5.1. Investigación

Teniendo en cuenta los objetivos del proyecto, el primer paso fue diseñar la infraestructura del Sistema Experto. Para lograr dicho propósito, era necesario saber qué es un Sistema Experto, los distintos tipos que había, cuál era el tipo de sistema elegido, y finalmente, listar las herramientas y tecnologías que se requerían para su desarrollo.

5.1.1. Sistema Experto

Un sistema experto es un programa informático diseñado para simular el conocimiento y las habilidades analíticas de un especialista en un campo concreto. Su función es simular la forma en la que un especialista toma una decisión.

Dependiendo del tipo del sistema experto, pueden estar basados en:

- Reglas previamente establecidas (RBR, Rule Based Reasoning) o encadenamiento hacia atrás.
- Basados en casos (CBR, Case Based Reasoning) o encadenamiento hacia adelante.
- **Basados en redes bayesianas.**

En nuestro caso, se ha decidido implementar el Sistema Experto mediante el uso de redes bayesianas, ya que es el tipo más comúnmente utilizado para este tipo de aplicaciones con sistemas expertos o variantes basadas en la inteligencia artificial.

Durante esta fase de investigación, se realizó un proceso de aprendizaje sobre los sistemas expertos y las redes bayesianas. Se leyó documentación para aprender y entender el concepto de redes bayesianas y sus componentes.

Con respecto al proyecto, el objetivo de este sistema es:

- Identificar y, por lo tanto, predecir, aquellos jóvenes que cumplen con las condiciones para ser considerados de alto riesgo (pronóstico).
- Monitorizar sus conductas de riesgo mediante evaluación momentánea (Ecological Momentary Assessment) (monitorización).

- Prevenir que se produzcan conductas autolesivas futuras (prevención), mediante la provisión de indicaciones y actuaciones personalizadas y ajustadas a los niveles de riesgo bajo la supervisión de profesionales.

Por otro lado, se ha propuesto que la estructura del sistema sea de tal forma que reciba una serie de datos de entrada, como se muestra en la Figura 5.1.

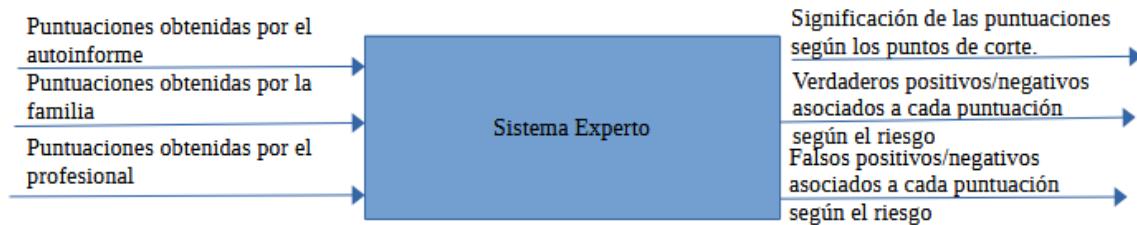


Figura 5.1: Estructura del Sistema Experto

El sistema recibirá una serie de puntuaciones obtenidas de las encuestas realizadas a los usuarios por el autoinforme, a las familias de los usuarios y a los profesionales de la salud (médicos, psicólogos, etc.).

Dichas encuestas se recogen en el sitio web de SIVARIA (Figuras 5.2 y 5.3).



Figura 5.2: Página inicial del sitio web de SIVARIA

En base a estos datos y al entrenamiento realizado, el sistema los procesará y devolverá una serie de positivos y negativos, tanto verdaderos como falsos. Esto será interpretado por la aplicación para determinar si existen conductas autolesivas en el usuario o los potenciales comportamientos que podría llegar a adoptar.

5.1.2. Redes de Bayes

Como se ha explicado anteriormente, se ha optado por implementar el Sistema Experto mediante una red de Bayes (Sucar y Tonantzintla (2006)).

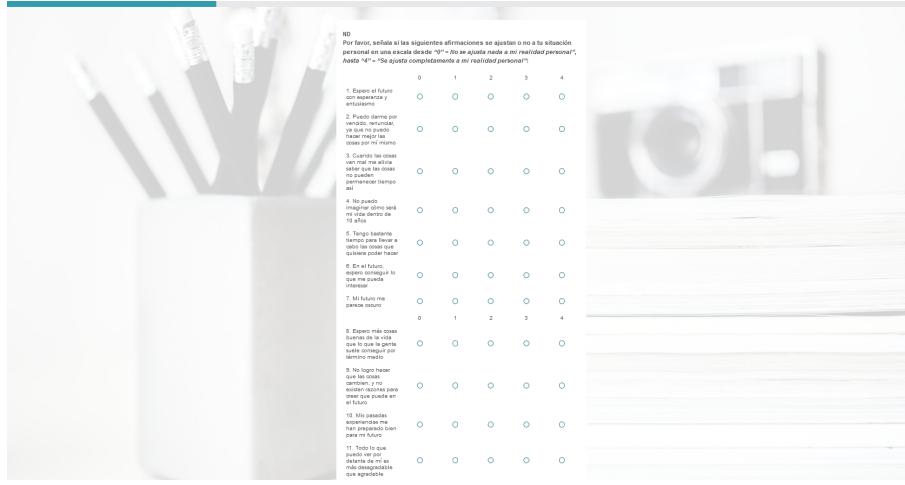


Figura 5.3: Encuesta a jóvenes de más de 16 años de SIVARIA

Las redes bayesianas, propuesto inicialmente por Judea Pearl en 1988, son un modelo probabilístico en el que se representan las relaciones probabilísticas entre las variables (nodos) y las dependencias condicionales que hay entre ellas, formando de esta manera un grafo acíclico dirigido (DAG, *Directed Acyclic Graph*).

Para que un grafo pueda ser considerado un DAG, es necesario que todos los arcos (*edges*) del grafo sean dirigidos, es decir, que indiquen una y sólo una dirección. Además, el grafo no debe tener ciclos, es decir, que dado un vértice v , no hay un camino directo que empiece y termine en v .

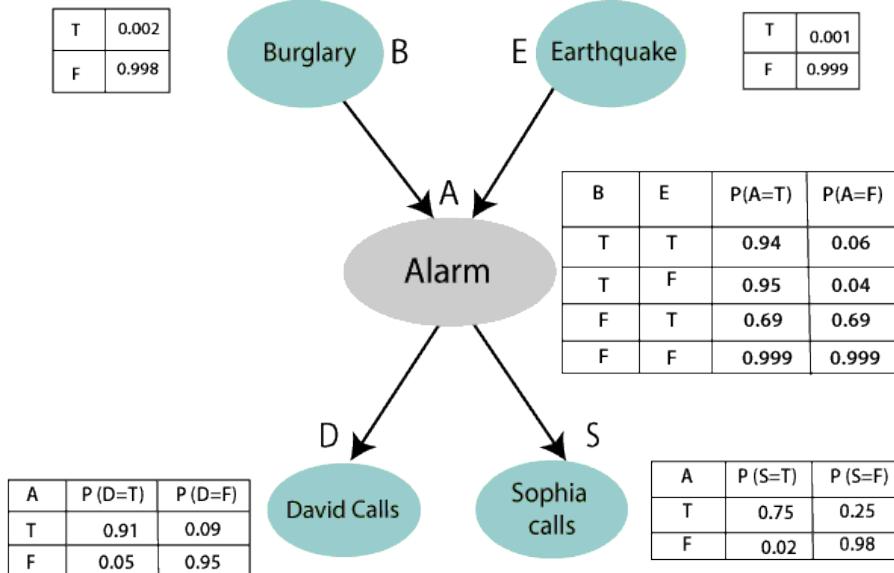


Figura 5.4: Ejemplo de un Diagrama de Bayes. Fuente, Javatpoint

Dentro de una red bayesiana, es necesario entender los conceptos de probabilidad conjunta y probabilidad condicional.

5.1.2.1. Probabilidad conjunta

La probabilidad conjunta es la probabilidad de que dos o más eventos ocurran al mismo tiempo. En una red de Bayes, la probabilidad conjunta de n variables se puede representar como:

$$P(X_1, X_2, \dots, X_n) = \prod_{i=1}^n P(X_i | X_{p(i)}) \quad (5.1)$$

Donde $X_{p(i)}$ es el padre de X_i .

Por ejemplo, dado:

- D = *David Calls.*
- S = *Sophie Calls.*
- A = *Alarm.*
- B = *Burglary.*
- E = *Earthquake.*

La probabilidad conjunta de la Figura 5.4 sería:

$$\begin{aligned} P(D, S, A, B, E) &= P(D|A)P(S, A, B, E) \\ &= P(D|A)P(S|A)P(A, B, E) \\ &= P(D|A)P(S|A)P(A|B, E)P(B, E) \\ &= P(D|A)P(S|A)P(A|B, E)P(B)P(E) \end{aligned} \quad (5.2)$$

5.1.2.2. Probabilidad condicional

La probabilidad condicional es la probabilidad de que un evento A ocurra dado otra evento B, $P(A|B)$.

Gracias a estos dos conceptos, se puede explicar uno de los elementos imprescindibles de las redes bayesianas: las tablas de distribución de probabilidad condicional.

5.1.3. Distribuciones de probabilidad condicional

Como se puede observar en la red de la Figura 5.4, una de las características de una red de Bayes son las tablas de distribución de probabilidad condicional (*CPD, Conditional Probability Distribution*).

Una CPD es una tabla donde se especifica la probabilidad de cada posible valor de una variable dada una combinación específica de los valores de sus variables padres. Tomando como ejemplo la Figura 5.4, la ecuación de la probabilidad condicional de que la alarma suene (*Alarm*) sería la siguiente:

$$P(\text{Alarm}|\text{SophiaCalls}, \text{DavidCalls}, \text{Burglary}, \text{Earthquake}) = P(\text{Alarm}|\text{Burglary}, \text{Earthquake}) \quad (5.3)$$

La variable *Alarm* es dependiente de *Burglary* y *Earthquake*, por lo que su probabilidad queda condicionada por la probabilidad conjunta de estas dos variables.

Burglary	Earthquake	P(Alarm=T)	P(Alarm=F)
T	T	0.94	0.06
T	F	0.95	0.04
F	T	0.69	0.69
F	F	0.999	0.999

Tabla 5.1: CPD de la variable *Alarm*

5.1.3.1. Inferencias

Una inferencia es un razonamiento probabilístico que consiste en asignar valores a ciertas variables (evidencia), y se obtiene la probabilidad posterior de las demás variables dadas las variables conocidas.

5.2. Diseño

En este sección se va a introducir la propuesta de los modelos para el Sistema Experto.

Se explicará al detalle los tipos de modelos propuestos, las variables que lo componen, los valores de cada variable, así como las relaciones entre ellos.

5.2.1. Diseño del modelo

Se propusieron tres modelos de redes de Bayes, uno por cada tipo de dataset recibido. Es decir, habrá un modelo para los datasets del autoinforme, un modelo de familia y otro de profesionales.

Esta propuesta de diseño se debe a que, en el entrenamiento posterior de los modelos, se requiere que todos los nodos sean entrenados en una sola ejecución, y en los datasets creados hay algunas variables que sólo están presentes en uno de los datasets. Por lo tanto, si se agrupan todos los nodos en una sola red de Bayes, sería imposible poder entrenar a todos los nodos con datos, lo que daría lugar a errores de ejecución en el código.

A continuación, se muestra los diagramas que hay, así como las variables y valores de cada uno. Cabe resaltar que todas las variables y los valores de los modelos se han basado en la documentación inicial proporcionado por los directores del proyecto y en las preguntas que se encuentran en los cuestionarios de los jóvenes, familias y profesionales del sitio web de SIVARIA.

5.2.1.1. Modelo del Autoinforme

El modelo va a constar de los siguientes grupos de variables:

- Variables de identificación
 - Curso: (COLEGIO, ESO, BACHILLERATO, UNIVERSIDAD, FORMACION PROFESIONAL)
- Variables sociodemográficas
 - Sexo asignado: (HOMBRE, MUJER, OTRO)

- Transgénero: (SÍ, NO, NO ESTOY SEGURO DE SER TRANS, NO ESTOY SEGURO DE LO QUE SE PREGUNTA)
 - Edad: (<12, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, >21)
 - Situación laboral padre: (NO TRABAJA, TRABAJA, PENSIONADO)
 - Situación laboral madre: (NO TRABAJA, TRABAJA, PENSIONADO)
 - Nivel profesional padre: (BAJO, MEDIO, ALTO)
 - Nivel profesional madre: (BAJO, MEDIO, ALTO)
 - Nivel promedio del rendimiento académico: (INSUFICIENTE, SUFICIENTE, NOTABLE, SOBRESALIENTE, EXTRAORDINARIO)
 - Nivel de autopercepción masculina: (0, 1, 2, 3, 4, 5, 6)
 - Nivel de autopercepción femenina: (0, 1, 2, 3, 4, 5, 6)
 - Nivel de heteropercepción masculina: (0, 1, 2, 3, 4, 5, 6)
 - Nivel de heteropercepción femenina: (0, 1, 2, 3, 4, 5, 6)
- Variables de salud
 - Altura: (<=149, 150-159, 160-169, 170-179, 180-189, >=190)
 - Peso: (<=49, 50-59, 60-69, 70-79, 80-89, >=90)
 - Tratamiento psiquiátrico previo: (SÍ, NO)
 - Presenta enfermedad crónica: (SÍ, NO)
 - Variables de bullying
 - Bullying víctima: (SÍ, NO)
 - Bullying perpetrador: (SÍ, NO)
 - Cyberbullying perpetrador: (SÍ, NO)
 - Cyberbullying perpetrador: (SÍ, NO)
 - Variables de adicción/abuso
 - Adicción/abuso alcohol: (SÍ, NO)
 - Adicción/abuso sustancias: (SÍ, NO)
 - Adicción/abuso Internet: (SÍ, NO)
 - Psicopatología
 - Problemas interiorizados: (SÍ, NO)
 - Problemas exteriorizados: (SÍ, NO)
 - Problemas de contexto: (SÍ, NO)
 - Problemas recursos psicológicos: (SÍ, NO)
 - Variables del constructo del comportamiento suicida
 - Problemas de discriminación: (SÍ, NO)
 - Fuente de discriminación: (EDAD, RAZA, DISCAPACIDAD, GÉNERO, ORIENTACIÓN SEXUAL)

- Variables de regulación y resistencia
 - Nivel de resistencia/resiliencia: (1, 2, 3, 4, 5)
 - Nivel de regulación positiva: (1, 2, 3, 4, 5)
 - Nivel de regulación negativa: (1, 2, 3, 4, 5)
- Variables volitivo-motivacionales para el suicidio
 - Atrapamiento interno: (SÍ, NO)
 - Atrapamiento externo: (SÍ, NO)
 - Nivel percibido de derrota o fracaso: (BAJO, MEDIO, ALTO)
 - Sentimiento de pertenencia frustada: (SÍ, NO)
 - Percepción de ser una carga: (SÍ, NO)
 - Autoeficiencia para el suicidio: (SÍ, NO)
- Perfil de riesgo psicosocial
 - Madre adolescente: (SÍ, NO)
 - Padre adolescente: (SÍ, NO)
 - Padres divorciados: (SÍ, NO)
 - Familia monoparental: (SÍ, NO)
 - Tratamiento psicológico padre/madre: (SÍ, NO)
 - Adicción padre/madre: (SÍ, NO)
 - Relaciones conflictivas hijo-padre/madre: (SÍ, NO)
 - Familia reconstruida: (SÍ, NO)
- Tecnologías y RRSS
 - Búsqueda información autolesión: (SÍ, NO)
 - Compartir en RRSS pensamiento autolesión: (SÍ, NO)
 - Petición ayuda en Internet: (SÍ, NO)
 - Contagio: (SÍ, NO)
 - Tener conocidos que comparten autolesión en Internet: (SÍ, NO)
 - Vía de contacto para RRSS: (SÍ, NO)
 - Contacto información autolesión: (SÍ, NO)
 - Denuncia autolesión Internet: (SÍ, NO)
- Desenlace: (COMUNICACIÓN, DESEO, IDEACIÓN, PLANIFICACIÓN, INTENCIÓN, FINALIDAD)

Debido al elevado número de variables que hay en el diagrama del Autoinforme, se ha dividido en dos partes para hacer fácilmente visible en el documento. De todas formas, la imagen completa de este diagrama, al igual que los otros dos diagramas se encuentran en el repositorio de Github, dentro de la carpeta *DiagramBayesSketches*.

Los diagramas se encuentran en las Figuras 5.5, 5.6 y 5.7.

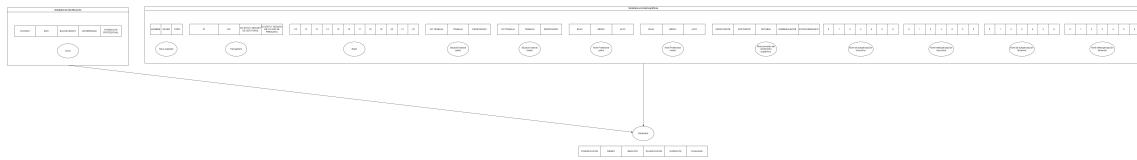


Figura 5.5: Red de Bayes del modelo del Autoinforme (I)

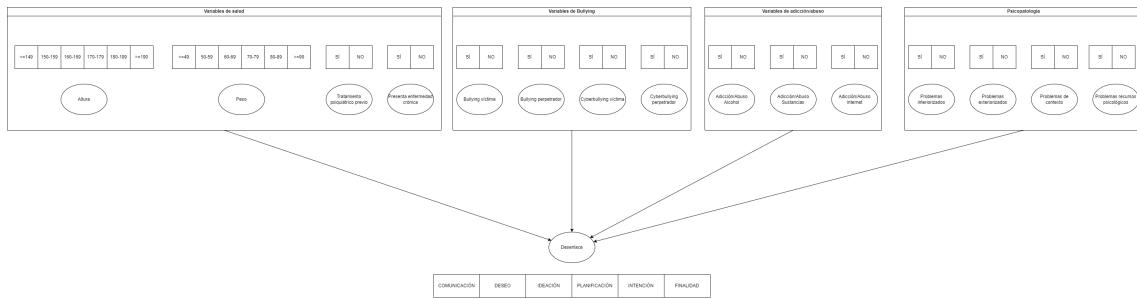


Figura 5.6: Red de Bayes del modelo del Autoinforme (II)

5.2.1.2. Modelo de las familias

El modelo va a constar de los siguientes grupos de variables con sus posibles valores:

- Perfil de riesgo psicosocial
- Psicopatología
- Variables de las familias
 - Aceptación/Rechazo parental: (ACEPTACIÓN, RECHAZO)
 - Control parental: (SÍ , NO).
- Desenlace

El diagrama se encuentra representado en la Figura 5.8.

5.2.1.3. Modelo de los profesionales

El modelo va a constar de los siguientes grupos de variables

- Perfil de riesgo psicosocial
- Variables de los profesionales
 - Situación económica familiar precaria: (SÍ, NO)
 - Estudios de la madre: (SÍ, NO)
 - Estudios del padre: (SÍ, NO)
 - Supervisión parental insuficiente: (SÍ, NO)
 - Maltrato al adolescente: (SÍ, NO)
 - Duelo: (SÍ, NO)
 - Ingreso familiar mensual: (<=499, 500-999, 1000-1499, 1500-1999, 2000-2499, >=2500)

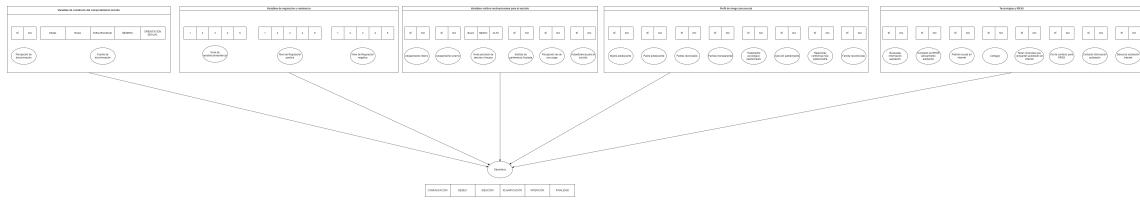


Figura 5.7: Red de Bayes del modelo del Autoinforme (III)

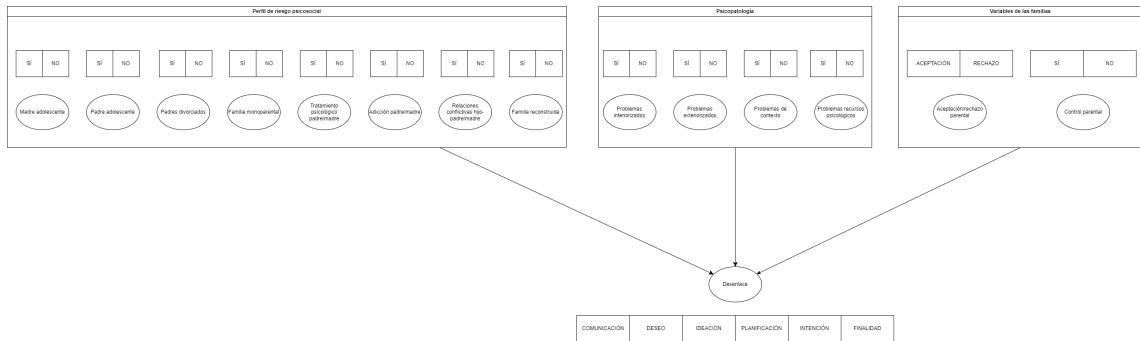


Figura 5.8: Red de Bayes del modelo de las familias

■ Desenlace

El diagrama se encuentra representado en la Figura 5.9.

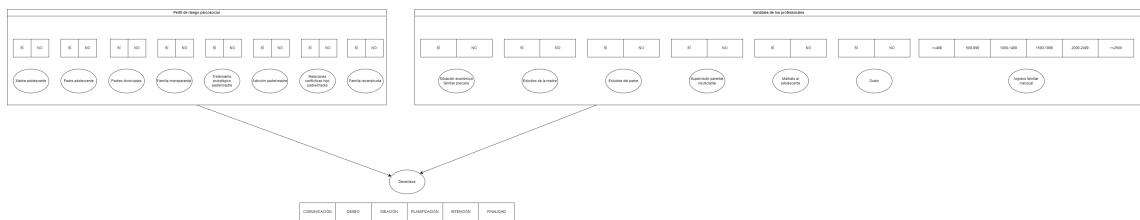


Figura 5.9: Red de Bayes del modelo de los profesionales

5.3. Desarrollo de los modelos

En esta sección se va explicar la el desarrollo de los modelos propuestos en el apartado anterior en un código de Python. La implementación del entrenamiento, predicción y testeo de los modelos se encuentra implementado tanto en un script de Python (*expertSystem.py*), como en el cuaderno de Jupyter *model_training_scikit_learn.ipnyb*, ambos ubicados en la carpeta *scripts* del repositorio de Github.

En primer lugar, se debe establecer qué tipo de modelo queremos entrenar: modelo del autoinforme, familia o profesional. Esto se hace para saber qué dataset se quiere cargar para entrenar al modelo.

5.3.1. Datasets

Antes de comenzar con el desarrollo de los scripts, hubo un inconveniente con respecto a los datasets que se tenían que usar para el entrenamiento.

Los datasets finalmente no fueron proporcionados por los profesionales, por lo que el entrenamiento del clasificador no se podía hacer.

Por lo tanto, a falta de datasets para tomarlos como referencia, se tuvo que usar, como alternativa, datasets generados de forma artificial y automática a través de un script de Python, en lugar de los datasets que supuestamente se debían de haber proporcionado.

Este obstáculo ha traído como desventaja principal, que el entrenamiento no sea tan bueno, y que las predicciones sean relativamente inexactas. Esto debido a que los datos generados por el script son aleatorios, por lo que puede dar lugar a datos poco realistas y posibles incoherencias.

5.3.2. Clasificación del modelo

En un principio, se ha decidido utilizar un clasificador Naive-Bayes para entrenar al modelo. El motivo es porque, a pesar de que es un algoritmo de clasificación simple, es bastante poderoso y que asume independencia entre características, permitiendo que sea rápido y eficiente. Es por ello que es bastante empleado para grandes conjuntos de datos, que es y será nuestro caso.

Toda la información del dataset se extrae y se guarda en un dataframe de la librería *pandas*. Posteriormente, dicho dataframe es decodificado y tratado para que pueda ser utilizado posteriormente en el proceso de entrenamiento del modelo.

Después, se divide los datos en un 75 % para entrenar al modelo y el 25 % restante se usará para testeo.

5.3.3. Entrenamiento

El entrenamiento se realiza con el 75 % de los datos del dataset y se hará de forma persistente. Es decir, en el caso de que haya otro modelo ya creado previamente, el entrenamiento se hará sobre dicho modelo.

Cabe resaltar que, cada vez que se finaliza este proceso, se procede al guardado del modelo en un fichero .sav que se ubicará en la carpeta local configFiles. De esta manera, se consigue que haya un histórico varias de todos los entrenamientos que ha recibido un modelo. Esto nos permite realizar un control de versiones de cada modelo, y pudiendo borrar versiones del modelo donde se considere que el entrenamiento no se ha hecho correctamente (Figura 5.10).

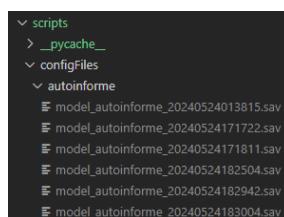


Figura 5.10: Ejemplo del directorio de archivos de guardado con el modelo del Autoinforme

5.3.4. Testeo del entrenamiento del modelo

Para medir el rendimiento de nuestro modelo, se debe comparar las respuestas predichas con las respuestas reales del conjunto de testeo.

Esto nos permitirá obtener la cantidad de verdaderos/falsos positivos y verdaderos/-falsos negativos que ha habido, que se usarán para calcular una puntuación que indicará el ratio de acierto de las predicciones del modelo.

Para calcular esta puntuación, existen cuatro tipos de métricas:

- **Exactitud (Accuracy)**: mide la proporción de predicciones correctas (tanto verdaderos positivos como verdaderos negativos) entre el total de predicciones realizadas.

$$\text{Exactitud} = \frac{TP + TN}{TP + TN + FP + FN} \quad (5.4)$$

Donde:

- TP: Verdaderos Positivos (*True Positives*)
- TN: Verdaderos Negativos (*True Negatives*)
- FP: Falsos Positivos (*False Positives*)
- FN: Falsos Negativos (*False Negatives*)

- **Precisión**: mide la proporción de verdaderos positivos entre todas las predicciones positivas hechas por el modelo. Es decir, de todas las veces que el modelo predijo que una instancia es positiva, cuántas veces acertó.

$$\text{Precision} = \frac{TP}{TP + FP} \quad (5.5)$$

- **Sensibilidad (Recall)**: mide la proporción de verdaderos positivos entre todas las instancias que son realmente positivas. Es decir, de todas las veces que una instancia es realmente positiva, cuántas veces el modelo la detectó correctamente.

$$\text{Sensibilidad} = \frac{TP}{TP + FN} \quad (5.6)$$

- **F1 Score**: es la media armónica entre la precisión y la sensibilidad. Es bastante útil en contextos donde se necesita un equilibrio entre la precisión y la sensibilidad, o cuando hay una distribución desequilibrada de clases.

$$F_{\text{score}} = 2 \cdot \frac{\text{Precision} \cdot \text{Sensibilidad}}{\text{Precision} + \text{Sensibilidad}} \quad (5.7)$$

El cálculo de todas estas métricas ya vienen incluido dentro del módulo scikit-learn (*sklearn.metrics*), así que no hace falta calcularlos todos de forma manual.

Cada métrica devolverá un ratio comprendido entre 0 y 1, y no tienen porqué devolver el mismo resultado. Dependiendo del contexto, es el desarrollador el que tendrá que escoger la métrica .

Sin embargo, hay un inconveniente con el cálculo de estas métricas. Sólo miden la puntuación de las predicciones de un conjunto concreto de valores de un dataset. Es posible que para otro conjunto aleatorio de datos de nuestro conjunto original, se obtenga una puntuación mucho más alta, lo que descartaría nuestra medición inicial.

Para asegurar la fiabilidad de nuestra medición, se recurre al cálculo del p-valor.

5.3.4.1. Cálculo del p-valor y la hipótesis nula

Antes de explicar el p-valor, es necesario introducir brevemente el concepto de la hipótesis nula.

La hipótesis nula es una afirmación inicial, normalmente llevada al absurdo, de un problema específico que se busca probar su falsedad, y por lo tanto rechazarla, en base a una serie de pruebas y mediciones. En este caso, nuestra hipótesis nula sería afirmar que la predicción de nuestro clasificador ha sido fruto de la suerte.

Por otro lado, el p-valor (o *valor de p*) es una medida estadística que indica la probabilidad de que, dado un conjunto nuevo D' derivado del original D , se produzca un resultado, $S_{muestra}$, aún más alto que el resultado inicial, $S_{inicial}$ ($S_{muestra} > S_{inicial}$). Se utiliza para decidir si una hipótesis nula se rechaza o no. Su fórmula es la siguiente:

$$p_valor = \frac{Cont + 1}{M + 1} \quad (5.8)$$

Donde:

- Cont: número de veces que la puntuación inicial es superior a la original ($Score_{muestra} > Score_{inicial}$).
- M: El número de nuevos conjuntos creados a partir del conjunto original.

Para calcular el valor de p, se utiliza un test de permutación, que consiste en medir la Exactitud de M permutaciones o muestras, y compararlas con la original.

Si el valor de p devuelto es superior a un nivel de significancia, se puede afirmar que el clasificador apenas ha devuelto predicciones mejores que la inicial, y por lo tanto, se rechaza la hipótesis nula.

5.4. Ciclos de entrenamiento

En esta sección se van a llevar a cabo los distintos tipos de entrenamientos del modelo, se compararán sus resultados y se determinará cuáles son los mejores parámetros para el entrenamiento de futuros datasets. En cada ciclo se va a resetear el entrenamiento del modelo y se van a proponer cambios en el entrenamiento y testeо.

5.4.1. Primer ciclo de entrenamiento

El primer ciclo de entrenamiento se realizó con los siguientes.

Clasificador	Naive-Bayes
Ubicación datasets	<i>scripts/datasets</i>
División de los datos Entrenamiento - Testeo	75 % - 25 %
Parámetro <i>average</i> de las métricas	micro

Tabla 5.2: Parámetros de entrenamiento del primer ciclo

El parámetro *average* se emplea en el cálculo de las métricas *precision*, *recall* y *f1_score*. En este caso, lo que hará será calcular a nivel global el número total de verdaderos positivos, falsos positivos y falsos negativos.

En base a estos parámetros, se obtuvieron los siguientes resultados en las métricas durante el proceso de testeо:

Métrica	Autoinforme					Media
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.182	0.160	0.166	0.178	0.174	0.172
Precision	0.182	0.160	0.166	0.178	0.174	0.172
Recall	0.182	0.160	0.166	0.178	0.174	0.172
F1 Score	0.182	0.160	0.166	0.178	0.174	0.172

Tabla 5.3: Comparativa de las puntuaciones del modelo del Autoinforme. Primer ciclo

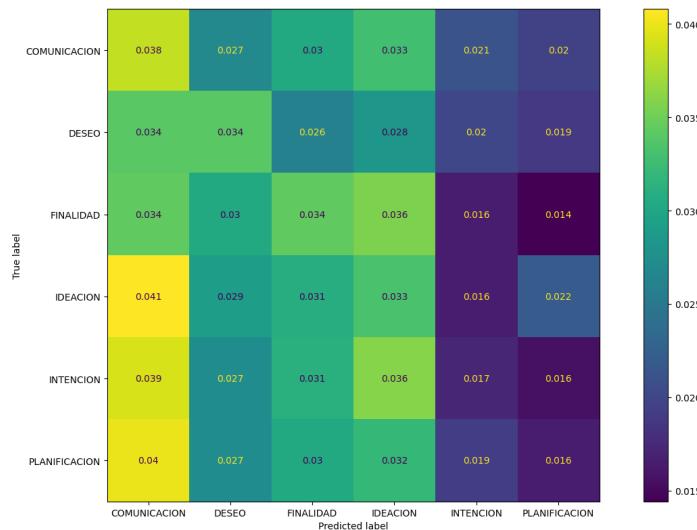


Figura 5.11: Matriz de confusión del Autoinforme al final del primer ciclo

Métrica	Familia					Media
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.174	0.168	0.175	0.176	0.182	0.175
Precision	0.174	0.168	0.175	0.176	0.182	0.175
Recall	0.174	0.168	0.175	0.176	0.182	0.175
F1 Score	0.174	0.168	0.175	0.176	0.182	0.175

Tabla 5.4: Comparativa de las puntuaciones del modelo de las familias. Primer ciclo

Métrica	Profesional					Media
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.166	0.176	0.159	0.153	0.171	0.165
Precision	0.166	0.176	0.159	0.153	0.171	0.165
Recall	0.166	0.176	0.159	0.153	0.171	0.165
F1 Score	0.166	0.176	0.159	0.153	0.171	0.165

Tabla 5.5: Comparativa de las puntuaciones del modelo de los profesionales. Primer ciclo



Figura 5.12: Matriz de confusión de las familias al final del primer ciclo

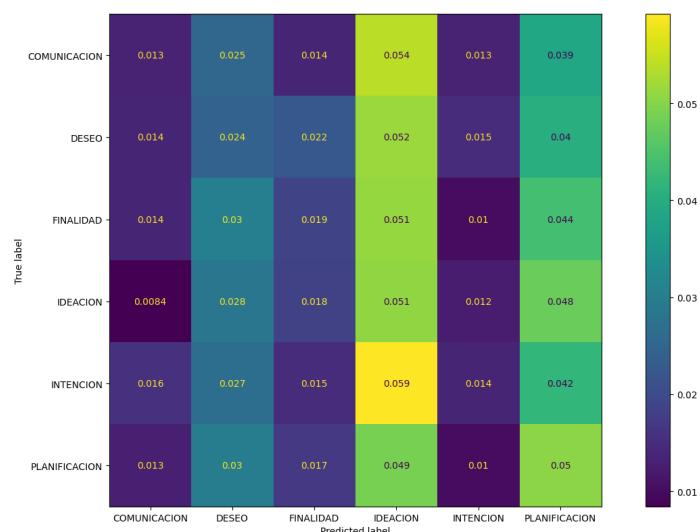


Figura 5.13: Matriz de confusión de los profesionales al final del primer ciclo

5.4.2. Segundo ciclo de entrenamiento

El segundo ciclo de entrenamiento se realizó un cambio en el cálculo del average de las métricas. En vez de usar la opción micro, se va a probar con la opción *weighted*⁶. A diferencia de *micro*, esta opción calcula la métrica de cada etiqueta o clase de manera separada y posteriormente, averigua su media ponderada por soporte, es decir, el número de instancias verdaderas para cada etiqueta.

Clasificador	Naive-Bayes
Ubicación datasets	<i>scripts/datasets</i>
División de los datos Entrenamiento - Testeo	75 % - 25 %
Parámetro <i>average</i> de las métricas	weighted

Tabla 5.6: Parámetros de entrenamiento del segundo ciclo

Los resultados fueron los siguientes:

Métrica	Autoinforme					Media
	Naive-Bayes					
Accuracy	0.182	0.160	0.166	0.178	0.174	0.172
Precision	0.182	0.160	0.163	0.178	0.171	0.171
Recall	0.182	0.160	0.166	0.178	0.174	0.172
F1 Score	0.182	0.157	0.160	0.174	0.170	0.169

Tabla 5.7: Comparativa de las puntuaciones del modelo del Autoinforme. Segundo ciclo

Métrica	Familia					Media
	Naive-Bayes					
Accuracy	0.174	0.168	0.175	0.176	0.182	0.175
Precision	0.175	0.172	0.173	0.175	0.181	0.175
Recall	0.174	0.168	0.175	0.176	0.182	0.175
F1 Score	0.173	0.159	0.168	0.172	0.179	0.170

Tabla 5.8: Comparativa de las puntuaciones del modelo de las familias. Segundo ciclo

⁶Cabe resaltar que este parámetro influye sobre las métricas durante el testeo, pero no en el entrenamiento. Esto implica que los resultados del entrenamiento serán los mismos si sólo se modifica este parámetro, por lo que las matrices de confusión serán iguales.

Métrica	Profesional					Media
	Naive-Bayes					
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.166	0.176	0.159	0.153	0.171	0.165
Precision	0.153	0.179	0.161	0.159	0.173	0.165
Recall	0.166	0.176	0.159	0.153	0.171	0.165
F1 Score	0.130	0.163	0.147	0.148	0.159	0.149

Tabla 5.9: Comparativa de las puntuaciones del modelo de los profesionales. Segundo ciclo

5.4.3. Tercer ciclo de entrenamiento

En este tercer ciclo de entrenamiento, se ha decidido modificar el porcentaje de división del dataset. En este escenario, el 70 % del dataset será utilizado en el entrenamiento, y el 30 % restante para testeo.

Por otro lado, el valor del parámetro *average* para el cálculo de las métricas vuelve a ser *micro*.

Clasificador	Naive-Bayes
Ubicación datasets	<i>scripts/datasets</i>
División de los datos Entrenamiento - Testeo	70 % - 30 %
Parámetro <i>average</i> de las métricas	micro

Tabla 5.10: Parámetros de entrenamiento del tercer ciclo

Los resultados fueron los siguientes:

Métrica	Autoinforme						Media
	Naive-Bayes						
Accuracy	0.183	0.168	0.160	0.178	0.169	0.169	0.172
Precision	0.183	0.168	0.160	0.178	0.169	0.169	0.172
Recall	0.183	0.168	0.160	0.178	0.169	0.169	0.172
F1 Score	0.183	0.168	0.160	0.174	0.169	0.169	0.172

Tabla 5.11: Comparativa de las puntuaciones del modelo del Autoinforme. Tercer ciclo



Figura 5.14: Matriz de confusión del Autoinforme al final del tercer ciclo

Métrica	Familia					Media
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.172	0.165	0.166	0.175	0.183	0.172
Precision	0.172	0.165	0.166	0.175	0.183	0.172
Recall	0.172	0.165	0.166	0.175	0.183	0.172
F1 Score	0.172	0.165	0.166	0.175	0.183	0.172

Tabla 5.12: Comparativa de las puntuaciones del modelo de las familias. Tercer ciclo

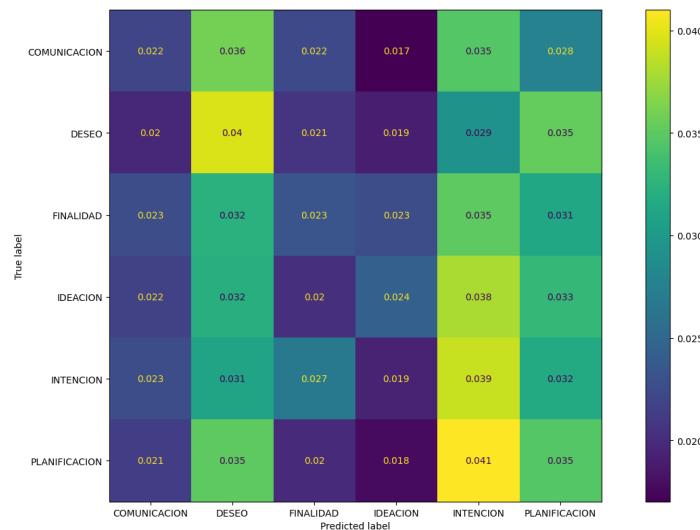


Figura 5.15: Matriz de confusión de las familias al final del tercer ciclo

Métrica	Profesional					Media
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.164	0.169	0.169	0.159	0.174	0.167
Precision	0.164	0.169	0.169	0.159	0.174	0.167
Recall	0.164	0.169	0.169	0.159	0.174	0.167
F1 Score	0.164	0.169	0.169	0.159	0.174	0.167

Tabla 5.13: Comparativa de las puntuaciones del modelo de los profesionales. Tercer ciclo

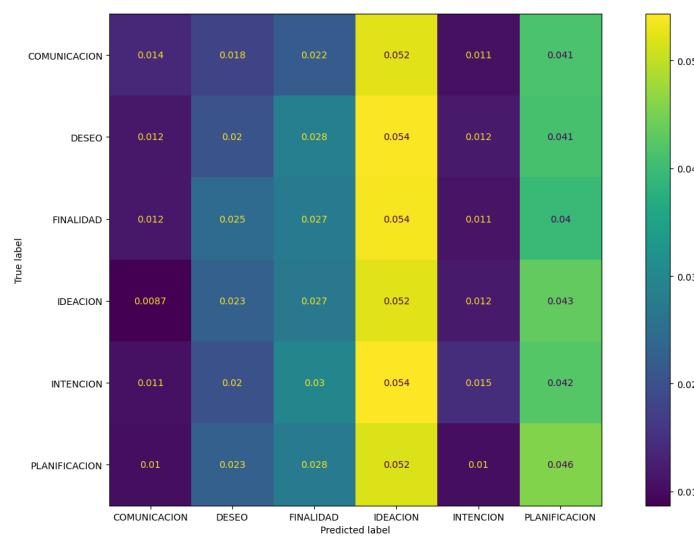


Figura 5.16: Matriz de confusión de los profesionales al final del tercer ciclo

5.4.4. Cuarto ciclo de entrenamiento

Clasificador	Naive-Bayes
Ubicación datasets	<i>scripts/datasets</i>
División de los datos Entrenamiento - Testeo	70 % - 30 %
Parámetro <i>average</i> de las métricas	weighted

Tabla 5.14: Parámetros de entrenamiento del cuarto ciclo

Los resultados fueron los siguientes:

Métrica	Autoinforme					Media
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.183	0.168	0.160	0.178	0.169	0.172
Precision	0.181	0.167	0.156	0.180	0.166	0.170
Recall	0.183	0.168	0.160	0.178	0.169	0.172
F1 Score	0.181	0.164	0.154	0.176	0.166	0.168

Tabla 5.15: Comparativa de las puntuaciones del modelo del Autoinforme. Cuarto ciclo

Métrica	Familia					Media
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.172	0.165	0.166	0.175	0.183	0.172
Precision	0.172	0.164	0.164	0.176	0.183	0.172
Recall	0.172	0.165	0.166	0.175	0.183	0.172
F1 Score	0.171	0.155	0.159	0.171	0.181	0.167

Tabla 5.16: Comparativa de las puntuaciones del modelo de las familias. Cuarto ciclo

Métrica	Profesional					Media
	Naive-Bayes					
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.164	0.166	0.169	0.159	0.174	0.166
Precision	0.144	0.176	0.172	0.156	0.181	0.166
Recall	0.164	0.166	0.169	0.159	0.174	0.166
F1 Score	0.132	0.155	0.159	0.154	0.164	0.153

Tabla 5.17: Comparativa de las puntuaciones del modelo de los profesionales. Cuarto ciclo

5.4.5. Quinto ciclo de entrenamiento

En este ciclo, se prueba una diferente división del dataset. Esta vez la división será un 80 % para entrenamiento y un 20 % para testeо. De esta manera, al haber un cambio en el porcentaje de división, los resultados del entrenamiento serán distintos, y por lo tanto, la matriz de confusión será distinta.

Clasificador	Naive-Bayes
Ubicación datasets	<i>scripts/datasets</i>
División de los datos Entrenamiento - Testeo	80 % - 20 %
Parámetro <i>average</i> de las métricas	micro

Tabla 5.18: Parámetros de entrenamiento del quinto ciclo

Los resultados fueron los siguientes:

Métrica	Autoinforme						Media
	Naive-Bayes						
Accuracy	0.182	0.164	0.168	0.180	0.177	0.174	0.174
Precision	0.182	0.164	0.168	0.180	0.177	0.174	0.174
Recall	0.182	0.164	0.168	0.180	0.177	0.174	0.174
F1 Score	0.182	0.164	0.168	0.180	0.177	0.174	0.174

Tabla 5.19: Comparativa de las puntuaciones del modelo del Autoinforme. Quinto ciclo



Figura 5.17: Matriz de confusión del Autoinforme al final del quinto ciclo

Métrica	Familia					Media
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.161	0.168	0.171	0.172	0.188	0.172
Precision	0.161	0.168	0.171	0.172	0.188	0.172
Recall	0.161	0.168	0.171	0.172	0.188	0.172
F1 Score	0.161	0.168	0.171	0.172	0.188	0.172

Tabla 5.20: Comparativa de las puntuaciones del modelo de las familias. Quinto ciclo

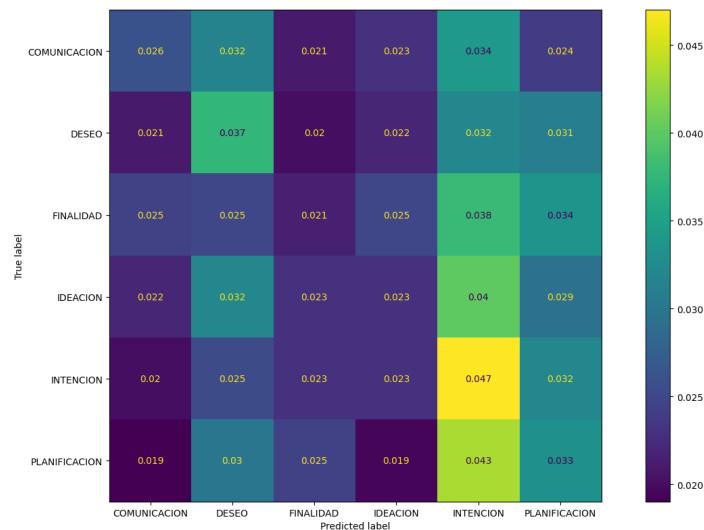


Figura 5.18: Matriz de confusión de las familias al final del quinto ciclo

Métrica	Profesional					Media
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.168	0.186	0.163	0.159	0.174	0.170
Precision	0.168	0.186	0.163	0.159	0.174	0.170
Recall	0.168	0.186	0.163	0.159	0.174	0.170
F1 Score	0.168	0.186	0.163	0.159	0.174	0.170

Tabla 5.21: Comparativa de las puntuaciones del modelo de los profesionales. Quinto ciclo

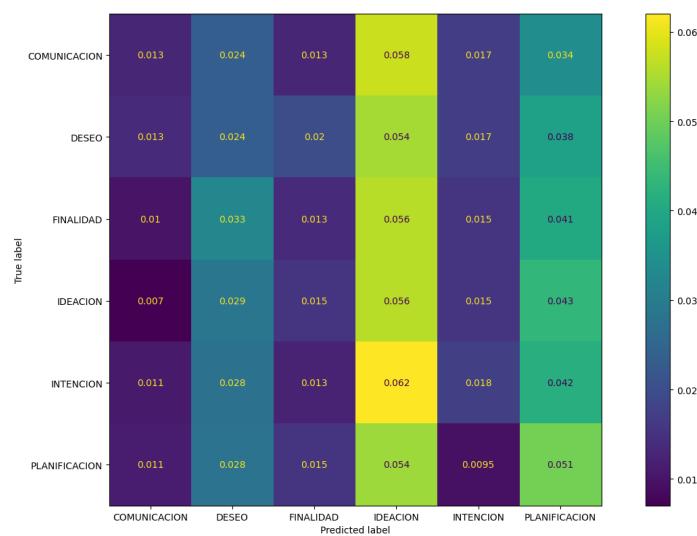


Figura 5.19: Matriz de confusión de los profesionales al final del quinto ciclo

5.4.6. Sexto ciclo de entrenamiento

En este ciclo, el balance se mantiene con respecto al ciclo anterior, con la diferencia de que el parámetro cambia de *micro* a *weighted*.

Clasificador	Naive-Bayes
Ubicación datasets	<i>scripts/datasets</i>
División de los datos Entrenamiento - Testeo	80 % - 20 %
Parámetro <i>average</i> de las métricas	weighted

Tabla 5.22: Parámetros de entrenamiento del sexto ciclo

Los resultados fueron los siguientes:

Métrica	Autoinforme					Media	
	Naive-Bayes						
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5		
Accuracy	0.182	0.164	0.168	0.180	0.177	0.174	
Precision	0.180	0.165	0.168	0.179	0.174	0.173	
Recall	0.182	0.164	0.168	0.180	0.177	0.174	
F1 Score	0.179	0.162	0.162	0.177	0.172	0.170	

Tabla 5.23: Comparativa de las puntuaciones del modelo del Autoinforme. Sexto ciclo

Métrica	Familia					Media	
	Naive-Bayes						
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5		
Accuracy	0.159	0.168	0.175	0.175	0.185	0.172	
Precision	0.157	0.171	0.178	0.175	0.179	0.172	
Recall	0.159	0.168	0.175	0.175	0.185	0.172	
F1 Score	0.155	0.159	0.168	0.170	0.177	0.166	

Tabla 5.24: Comparativa de las puntuaciones del modelo de las familias. Sexto ciclo

Métrica	Profesional					Media
	Naive-Bayes					
	Dataset1	Dataset2	Dataset3	Dataset4	Dataset5	
Accuracy	0.168	0.186	0.163	0.159	0.174	0.170
Precision	0.152	0.185	0.168	0.160	0.175	0.168
Recall	0.168	0.186	0.163	0.159	0.174	0.170
F1 Score	0.148	0.174	0.151	0.150	0.160	0.157

Tabla 5.25: Comparativa de las puntuaciones del modelo de los profesionales. Sexto ciclo

5.4.7. Análisis de los resultados

En esta sección, analizamos los resultados obtenidos de todos los ciclos, y se decide qué tipo de métrica se va a utilizar para futuros entrenamientos del modelo, el tipo de parámetro *average* (*micro* o *weighted*) y el porcentaje en la división del dataset.

Una vez realizados los distintos tipos de entrenamientos, se realizó una tabla comparativa de las medias de las métricas en cada ciclo:

Ciclos	Accuracy	Precision	Recall	F1 Score
Ciclo 1	0.172	0.172	0.172	0.172
Ciclo 2	0.172	0.171	0.172	0.169
Ciclo 3	0.172	0.172	0.172	0.172
Ciclo 4	0.172	0.170	0.172	0.168
Ciclo 5	0.174	0.174	0.174	0.174
Ciclo 6	0.172	0.172	0.172	0.166

Tabla 5.26: Tabla comparativa de las métricas del Autoinforme

En este caso, la tabla muestra que el ciclo que ofrece mejores resultados en todas las métricas es el quinto ciclo. Con respecto a la métrica *accuracy* y *recall*, la mejora es de un 0.002 con respecto al resto de ciclos. En la *precision*, la diferencia está entre un 0.002 y un 0.004. Finalmente, con la métrica *F1 Score*, la mejora es de entre un 0.002 y 0.008 con respecto al resto.

Ciclos	Accuracy	Precision	Recall	F1 Score
Ciclo 1	0.175	0.175	0.175	0.175
Ciclo 2	0.175	0.175	0.175	0.170
Ciclo 3	0.172	0.172	0.172	0.172
Ciclo 4	0.172	0.172	0.172	0.167
Ciclo 5	0.172	0.172	0.172	0.172
Ciclo 6	0.172	0.172	0.172	0.166

Tabla 5.27: Tabla comparativa de las métricas del modelo de las familias

En este caso, la tabla muestra que el ciclo que ofrece mejores resultados es el primer ciclo, ya que ofrece un ratio de 0.175 en todas las métricas, con una diferencia del 0.003 con respecto al resto de ciclos. Además, logra una diferencia del 0.005 con respecto al ciclo 2 en el *F1 Score*, haciendo que sea el mejor escenario para los modelos de las familias. En el ciclo 1, las características fueron una división del dataset del 75 (Entrenamiento) - 25 (Testeo), y el uso del valor *micro* en el parámetro *average* en el cálculo de las métricas.

En este caso, la tabla muestra que el ciclo que ofrece mejores resultados en todas las métricas es el quinto ciclo, ya que ofrece un ratio de 0.170 en todas las métricas, con una diferencia de entre 0.002 y 0.005 con respecto al resto de ciclos. Además, si se compara con el segundo ciclo más beneficioso, el ciclo 6, vemos que logra una diferencia del 0.002 con respecto al ciclo 6 en la métrica *precision* y un 0.13 con el *F1 Score*.

5.4.7.1. Elección de la métrica y los parámetros

Con respecto a la elección de la métrica, en general, la que ofrece mejores resultados y más consistentes, en todos los ciclos es el *accuracy*, a diferencia de las otras, que varían

Ciclos	Accuracy	Precision	Recall	F1 Score
Ciclo 1	0.165	0.165	0.165	0.165
Ciclo 2	0.165	0.165	0.165	0.149
Ciclo 3	0.167	0.167	0.167	0.167
Ciclo 4	0.166	0.166	0.166	0.153
Ciclo 5	0.170	0.170	0.170	0.170
Ciclo 6	0.170	0.168	0.170	0.157

Tabla 5.28: Tabla comparativa de las métricas del modelo de los profesionales

dependiendo de la división del dataset o por la forma en la se calcula el promedio de los verdaderos/falsos positivos y verdaderos/falsos negativos.

Por lo tanto, el *accuracy* será la métrica utilizada para entrenar al modelo en el futuro cuando se proporcionen nuevos datasets.

Por otro lado, se puede observar que el ciclo 5 es el que ofrece, en general, el *accuracy* más elevado en las tablas del Autoinforme y el de los profesionales. En el caso del Autoinforme, la mejora de la métrica del ciclo 5 con respecto al resto es del 0.002 de manera uniforme. Mientras que en el caso de los profesionales, la mejora del ciclo 5 suele ser de entre un 0.003 y 0.005 con respecto al resto de ciclos. Considero que estas diferencias de mejora compensan la ligera pérdida de mejora en los modelos de la familia, ya que en este escenario, el ciclo 1 ofrece mejores resultados que el ciclo 5.

Recordando las características del ciclo 5, se utiliza el valor *micro* en el parámetro *average* y se utiliza una división del 80 % (Entrenamiento) - 20 %(Testeo).

Capítulo 6

Funcionamiento de la aplicación

En esta sección se va explicar el funcionamiento del código desarrollado y de las opciones presentes en el Sistema Experto.

6.1. Actores de la aplicación

Dentro del código de Python se distinguen lo siguientes agentes internos del Sistema Experto:

- **Controller:** recibe el comando introducido por el usuario y ejecuta la correspondiente opción y devuelve el resultado realizado por el Sistema Experto.
- **Expert System:** Sistema Experto de la aplicación. Se encarga internamente de entrenar un modelo seleccionado por el usuario, testear, crear la matriz de confusión y realizar predicciones dado un CSV.
- **Checker:** realiza revisiones en el código sobre el formato de los valores recibidos. Si el formato no se cumple, se lanza una excepción.
- **File Manager:** se encarga de administrar los archivos de guardado de las versiones de los modelos. Su trabajo es crear, mostrar y eliminar los archivos de guardado de un modelo. De esta manera, podemos garantizar el control de versiones para todos los modelos y la persistencia de los entrenamientos realizados.
- **Configurator:** se encarga de administrar el archivo de configuración del Sistema Experto. Es capaz de añadir y sustituir valores de la configuración, como el tipo de modelo o puntuación seleccionado.
- **Decoder:** se encarga de recibir el dataframe del CSV, y lo prepara para el entrenamiento del modelo, sustituyendo las variables continuas del CSV por intervalos de valores, y de esta manera, discretizándolos.

Para poder desarrollar todos los actores, se ha seguido la programación orientada a objetos en Python (*Object Oriented Programming*). Gracias a este paradigma, se pueden dividir los objetos en clases u objetos, lo que permite crear scripts de Python específicos para cada objeto.

En el Apéndice A, se explican los scripts que componen el código y qué clase es cada uno. Además, también incluirá una guía para poder ejecutar cada una de las funcionalidades que se mostrarán en el siguiente apartado.

Por otro lado, es de importancia mencionar que las peticiones del usuario no se realizan de manera directa al Sistema Experto, sino que al principio van dirigidos al Controlador, quien es el que recibe las órdenes del usuario y llama al Sistema Experto para realizar las correspondientes acciones.

Entonces, la estructura sería similar a lo que se puede ver en la Figura 6.1.

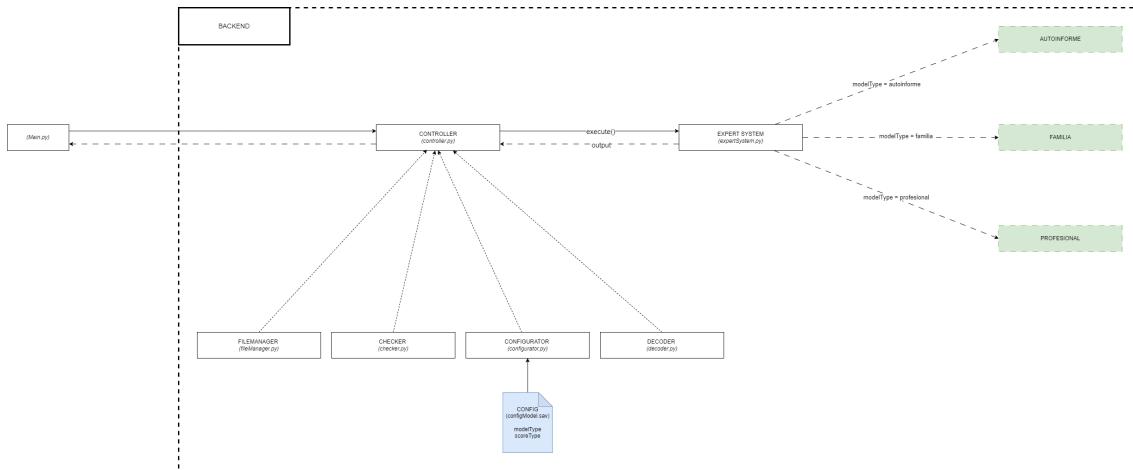


Figura 6.1: Estructura del Controlador con el Sistema Experto

6.2. Funcionalidades del código

En esta sección se va mostrar las funcionalidades que hay en el código.

6.2.1. Establecimiento del tipo de modelo

Esta funcionalidad permite mostrar y establecer el tipo de modelo que se va a utilizar para el entrenamiento del modelo. Los valores permitidos son: *autoinforme*, *familia* y *profesional*.

Posteriormente, el valor establecido se guarda en el archivo de configuración en la carpeta *configFiles* del repositorio.

En la Figura 6.2 se puede ver un diagrama de secuencia del comando en cuestión.

6.2.2. Establecimiento del tipo de puntuación

Esta funcionalidad es similar al anterior, la diferencia es que en vez de establecer el tipo de modelo, se establece el tipo de puntuación. Los posibles valores pueden ser: *accuracy*, *precision*, *recall* y *f1_score*.

Posteriormente, el valor establecido se guarda en el archivo de configuración en la carpeta *configFiles* del repositorio.

En la Figura 6.3 se puede ver un diagrama de secuencia del comando en cuestión.

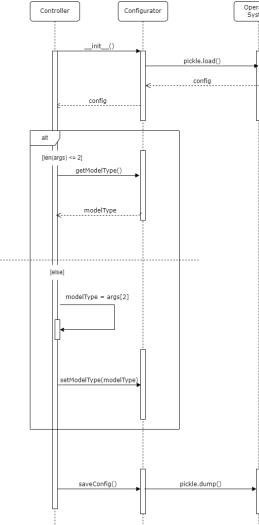


Figura 6.2: Diagrama de secuencia del comando para establecer el tipo de modelo

6.2.3. Listado de archivos de guardado

Devuelve una lista de los archivos de guardado que hay dentro de la carpeta `configFiles/<tipo del modelo>`. Se puede especificar el modelo para ver los archivos de guardado de ese modelo, o no se introduce ningún parámetro extra, y como resultado, se devuelven los archivos de guardado de todos los modelos.

En la Figura 6.4 se puede observar el diagrama de secuencia del comando que muestra la lista de archivos de guardado de los modelos.

6.2.4. Información del archivo de guardado

El Controlador recibe el nombre de un archivo de guardado y devuelve la información básica de ese archivo. Esta información consta de:

- El nombre del archivo.
- La fecha y hora del último acceso al archivo.
- La fecha y hora de la última modificación.
- La fecha hora de la última vez que ha habido un cambio en los metadatos en Unix y la hora de creación en Windows.
- El tamaño del archivo
- La unidad del tamaño del archivo (KiB, MiB, GiB, TiB, etc.).

En la Figura 6.5 se puede observar el diagrama de secuencia del comando.

6.2.5. Eliminación de un archivo de guardado

Elimina un archivo de guardado dado el nombre del archivo. En la Figura 6.6 se puede observar el diagrama de secuencia del comando del borrado del archivo de guardado.

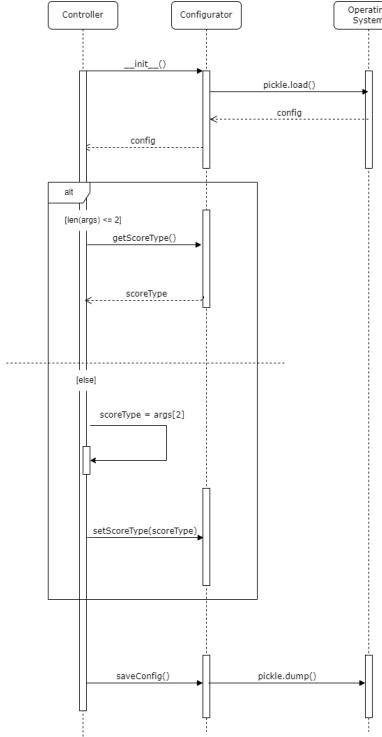


Figura 6.3: Diagrama de secuencia del comando para establecer el tipo de puntuación

6.2.6. Entrenamiento del modelo

Dado un dataset por parte del usuario (se debe especificar el path del archivo), divide el dataset en base al balance seleccionado en la Sección 5.4.7.

Finalmente, se entrena al modelo con la parte correspondiente del dataset, y la parte restante se usará para crear la matriz de confusión y testear el modelo con el cálculo del valor de p.

Finalmente, el modelo entrenado se guarda en un archivo SAV dentro de la carpeta `configFiles/<tipo de modelo>`, y se devuelve un diccionario de Python, que contiene el número de verdaderos/falsos negativos de cada valor de la variable *Desenlace*, especificado en la Sección 5.2.1.1, y el valor final del valor de p, junto con las veces en donde que se obtuvo un valor superior a la puntuación original.

Es necesario que previamente a la ejecución de esta opción, es necesario que se haya especificado previamente el tipo de modelo y el tipo de puntuación.

Para visualizarlo mejor, en la Figura 6.7 se puede observar el diagrama de secuencia donde se muestra los pasos internos del entrenamiento.

6.2.7. Predicción

Dado un dataset enviado a través de los parámetros del comando, se realiza una predicción de las filas almacenadas dentro de dicho dataset.

Se devuelve un dataframe con los resultados de esa predicción.

En la Figura 6.8 se puede observar el diagrama de secuencia del comando.

Todas las funcionalidades previamente explicadas se podrán acceder a ella y ejecutarlas

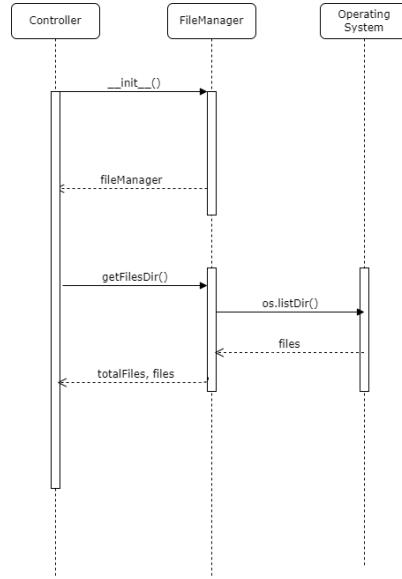


Figura 6.4: Diagrama de secuencia del comando para listar los archivos de guardado

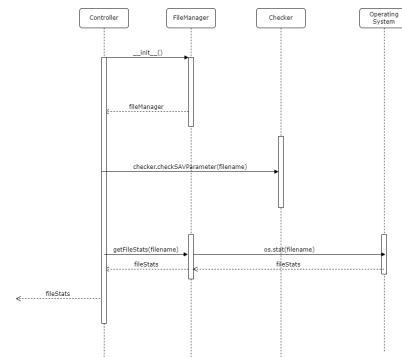


Figura 6.5: Diagrama de secuencia del comando para mostrar la información del archivo de guardado

a través del Controlador mediante una serie de comandos creados específicamente para la aplicación. Estos comandos están explicados en el Apéndice A.

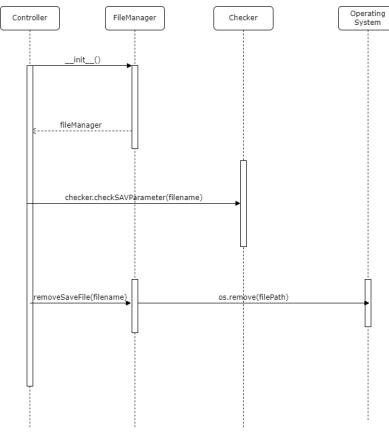


Figura 6.6: Diagrama de secuencia del comando para eliminar un archivo de guardado

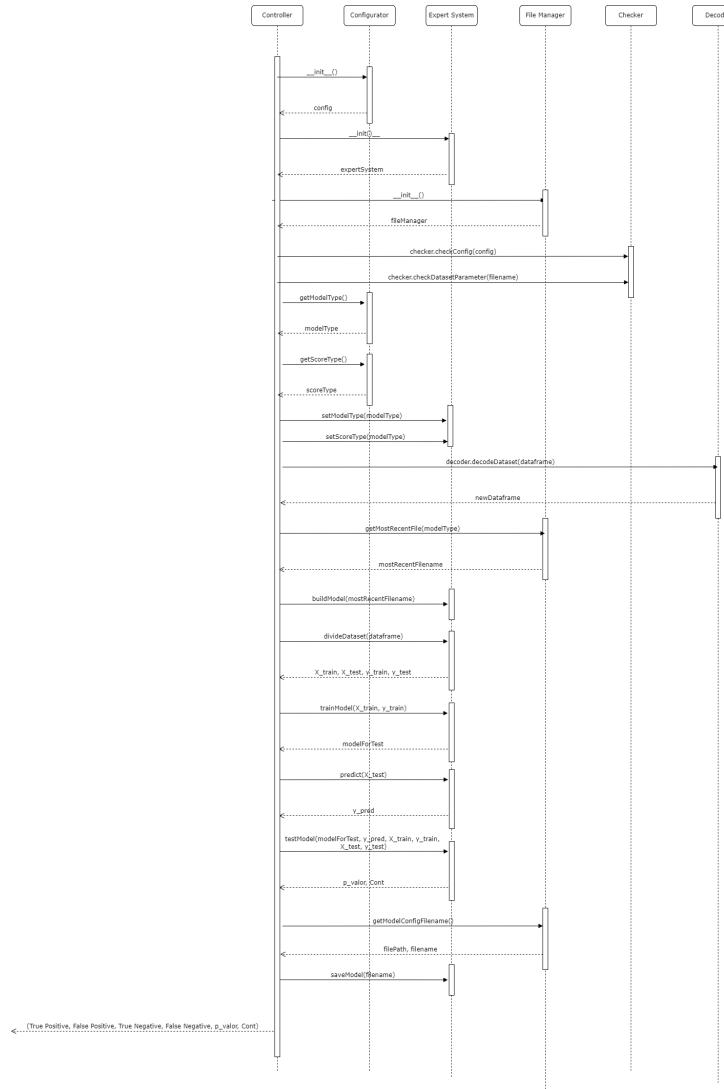


Figura 6.7: Diagrama de secuencia del comando para entrenar al modelo

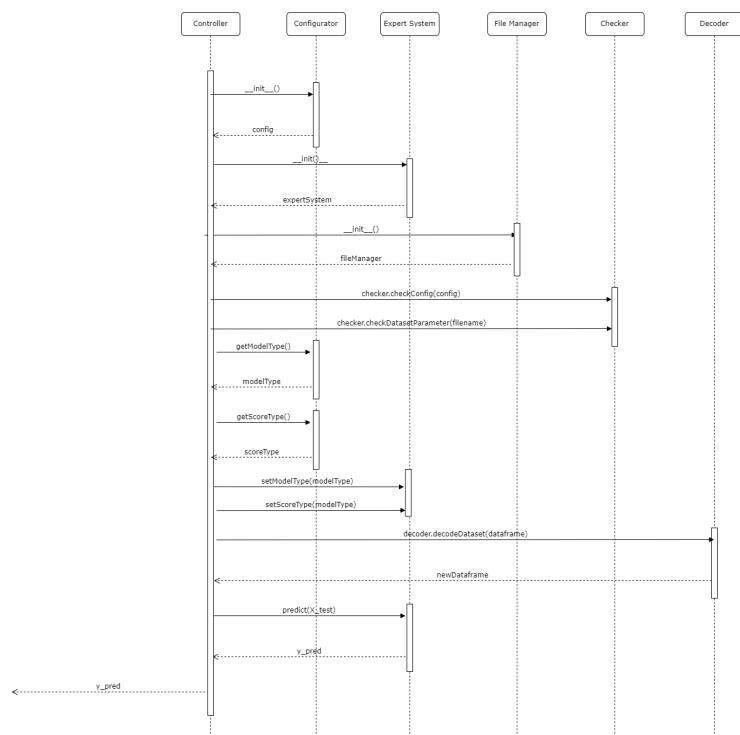


Figura 6.8: Diagrama de secuencia del comando para realizar una predicción

Capítulo 7

Conclusiones y Trabajo Futuro

7.1. Conclusiones

En conclusión, el objetivo primordial del proyecto es reducir y prevenir el suicidio en adolescentes, una problemática creciente en estos últimos años. Con esa finalidad, se busca poder identificar y predecir con antelación las conductas autolesivas en los jóvenes, para avisar a los profesionales de la salud y que puedan atender a la potencial víctima lo más pronto posible.

Con el desarrollo de este Sistema Experto, se busca cumplir con dicho objetivo mediante la aplicación de la inteligencia artificial, más concretamente, de las redes bayesianas, al campo de la salud, método que se está comenzando a utilizar más en este ámbito.

Finalmente, el incremento de este tipo de proyecto que implementan la técnica de las redes bayesianas ayudarán a contribuir al campo de la salud mental, ayudando a los profesionales realizar diagnósticos con mayor precisión, y permitiendo a los jóvenes y familiares pueden recibir una predicción del modelo, que se hará con gran exactitud ya que estará basado en datos de otros pacientes anteriores, que se habrán usado durante el entrenamiento y el testeo.

7.2. Trabajo a futuro

Con respecto al trabajo a futuro, se ha planteado la posibilidad de llevar el trabajo del Sistema Experto a la nube. Desafortunadamente, la velocidad ha sido uno de los principales inconvenientes a la hora de escoger una librería de Python para desarrollar el Sistema Experto, ya que la gran mayoría necesitaban demasiado tiempo para realizar los procesos de creación, entrenamiento, testeo y predicción del modelo en una máquina local. En algunos casos, ha provocado fallos en la ejecución. Con esta solución, se podría distribuir la carga de trabajo entre varios servidores que tengan mayor capacidad que una máquina local, agilizando el trabajo y obteniendo respuestas más rápidas.

Por otro lado, se espera que en el futuro se pueda materializar el desarrollo de la aplicación móvil, para que los jóvenes puedan acceder más fácilmente a los cuestionarios del sitio web de SIVARIA y puedan obtener resultados más rápidos del Sistema Experto. Además, también podría ser de utilidad para los profesionales de la salud para poder recoger y observar las estadísticas de los resultados obtenidos del Sistema Experto. La idea de la aplicación es que sea gratuita y multiplataforma, es decir, que esté disponible tanto

en Android como en IOS, además de su versión de escritorio.

También está abierta la posibilidad de poder mejorar algunos campos del diagrama de la red de Bayes, añadiendo más variables, más tipos de resultados para una variable en concreto o añadiendo más dependencias entre ellas. Todo esto se realizará en concordancia con los profesionales de la salud involucrados en el proyecto. De esta manera, el modelo puede refinar cada vez más, obteniendo predicciones más exactas.

Capítulo 8

Conclusions and Future Work

8.1. Conclusions

In conclusion, the main objective of the project is to reduce and prevent suicide in adolescents, a growing problem in recent years. To this end, the aim is to be able to identify and predict self-harming behaviour in young people in advance, in order to warn health professionals so that they can attend to the potential victim as soon as possible.

With the development of this Expert System, the aim is to fulfil this objective by applying artificial intelligence, more specifically Bayesian networks, to the field of health, a method that is beginning to be used more in this field.

Finally, the increase of these type of projects that implement the Bayesian Networks technique will contribute in the mental health field, helping professionals in their diagnoses, and allowing young people and relatives receive a highly accurate prediction from the model, based on training data coming from previous patients that had similar pathologies.

8.2. Future work

Regarding the future work, the possibility has been raised of taking the work of the Expert System to the cloud. Unfortunately, the speed is one of the main inconveniences of a model design, since the python libraries generally take too much time to do all the creating, training, testing and predicting processes in a local machine. By this solution, the workload could be distributed among several servers, speeding up the work and obtaining faster responses.

On the other hand, it is expected that in the future the development of the mobile application will materialise, so that young people can more easily access the questionnaires on the SIVARIA website and obtain faster results from the Expert System. get quicker results from the Expert System. In addition, it could also be useful for health professionals to be able to collect and observe the statistics of the results obtained from the Expert System. The idea of the application is that it should be free and multiplatform, i.e. available on both Android and IOS, in addition to its desktop version.

It is also open the possibility of improving some fields of the Bayes network diagram, adding more variables, more possible values for the variables or adding more dependencies between them. However, these previous changes need to be done in agreement with the

health professionals involved in the project. In this way, the model can be further refined, resulting in more accurate predictions.

Bibliografía

- Preventing suicide: A global imperative.* World Health Organization and others, 2014.
- Fundación española para la prevención del suicidio. observatorio del suicidio en españa 2022. datos definitivos diciembre 2023. 2022. Disponible en <https://www.fsme.es/observatorio-del-suicidio-2022-definitivo/> (último acceso, Abril, 2024).
- AL-HALABÍ, S. y FONSECA-PEDRERO, E. Suicidal behavior prevention: The time to act is now. *Clínica y Salud*, vol. 32(2), páginas 89–92, 2021.
- ANKAN, A. y PANDA, A. pgmpy: Probabilistic graphical models using python. En *Proceedings of the Python in Science Conference*, SciPy. SciPy, 2015. ISSN 2575-9752.
- ATIENZA, D., BIELZA, C. y LARRAÑAGA, P. Pybnesian: An extensible python package for bayesian networks. *Neurocomputing*, vol. 504, páginas 204–209, 2022.
- ETEROVIC, N. K. H. El observatorio del suicidio, impulsor de la red de prevención del suicidio de las islas baleares. ????
- FONSEKA, T. M., BHAT, V. y KENNEDY, S. H. The utility of artificial intelligence in suicide risk prediction and the management of suicidal behaviors. *Australian & New Zealand Journal of Psychiatry*, vol. 53(10), páginas 954–964, 2019.
- GORE, F. M., BLOEM, P. J., PATTON, G. C., FERGUSON, J., JOSEPH, V., COFFEY, C., SAWYER, S. M. y MATHERS, C. D. Global burden of disease in young people aged 10–24 years: a systematic analysis. *The Lancet*, vol. 377(9783), páginas 2093–2102, 2011.
- INE. Causas de muertes externas en españa. 2024. Disponible en <https://www.ine.es/jaxiT3/Tabla.htm?t=7947> (último acceso, Abril, 2024).
- MUELA, A., GARCÍA-ORMAZA, J. y SANSINENA, E. Suicidal behavior and deliberate self-harm: A major challenge for youth residential care in spain. *Children and Youth Services Review*, vol. 158, página 107465, 2024. ISSN 0190-7409.
- PYPL. Pypl. 2023. Disponible en <https://pypl.github.io/PYPL.html> (último acceso, Abril, 2024).
- PYTHON. *Documentación de Python*. Versión electrónica, 2020. Disponible en <https://docs.python.org/3.9/> (último acceso, Febrero, 2024).
- SCHREIBER, J. pomegranate. *Software download. URL https://github.com/jmschrei/pomegranate*, 2014.

- SUCAR, L. E. y TONANTZINTLA, M. Redes bayesianas. *Aprendizaje Automático: conceptos básicos y avanzados*, vol. 77, página 100, 2006.
- URBAN, M. *An introduction to LATEX*. TEX users group, 1986.
- WIKIPEDIA. Github — wikipedia, la enciclopedia libre. 2024. [Internet; descargado 2-abril-2024].

Apéndice A

Guía de instalación

Todo el desarrollo del TFM, tanto código como bocetos de los diagramas, se encuentran en el repositorio online de Github.

A.1. Contenido del repositorio

Al descargarse el repositorio, se pueden observar tres carpetas principales:

- **DiagramBayesSketches**: esta carpeta contiene los bocetos de los diagramas de Bayes de todos los modelos: autoinforme, familia y profesional. Además, aparte hay una subcarpeta con el boceto de cada grupo de variables, para poder ver con mayor claridad cómo está compuesto cada grupo.
- **Memoria**: en ella se encuentra la presente memoria.
- **scripts**: en esta carpeta se encuentran los scripts necesarios para la creación del Sistema Experto y el Controlador.

A.1.1. Scripts

Constan de los siguientes archivos:

- **exceptions**: carpeta que contiene una serie de excepciones personalizadas creadas para el Sistema Experto.
- **configFiles**: carpeta donde se almacenará los archivos de guardado de los modelos del Autoinforme, Familia y Profesional.
- **configFilesJupyter**: carpeta similar a *configFiles*, con la diferencia de que en esta se guardan los modelos del cuaderno de Jupyter *model_training_scikit_learn.py*.
- **datasets**: carpeta donde se encuentran los datasets de prueba creados de forma manual, aleatoria y artificial para entrenar a los modelos, y así, probar el rendimiento del Sistema Experto.
- *expertSystem.py*: contiene la clase Sistema Experto, que utiliza el módulo scikit learn para realizar las funciones de predicción, entrenamiento y testeo. Se encarga de construir el modelo inicial a partir del tipo de modelo especificado por el usuario, que

puede ser el modelo del autoinforme, el de la familia o el de los profesionales. Además, también posee una función para guardar el modelo entrenado en un archivo .sav y que lo puede colocar en las carpetas *scripts/configFiles/autoinforme*, *scripts/configFiles/familia* o *scripts/configFiles/profesional*, también dependiendo del tipo de modelo.

- *controller.py*: ejerce de Controlador que es capaz de utilizar métodos del Sistema Experto a través de una serie de comandos:
 - -h: muestra un mensaje de ayuda que explica los comandos disponibles del controller.py. Tiene un formato similar a las páginas *man* de Linux.
 - -mt: muestra y establece el tipo de modelo seleccionado por el usuario (autoinforme, familia, profesional).
 - -st: muestra y establece el tipo de puntuación seleccionado por el usuario (accuracy, precision, recall, f1_score).
 - -lst: devuelve una lista de archivos de guardado que hay según el tipo de modelo.
 - -fs: devuelve la información del archivo dado su nombre.
 - -rm: elimina un archivo de guardado dado su nombre.
 - -t: entrena al modelo. Es necesario que previamente se establezca el tipo de modelo (-mt) y el tipo de puntuación (-st). Además, debemos añadir en los parámetros el dataset que se va a usar para el entrenamiento. Devuelve una lista de verdaderos/falsos positivos y verdaderos/falsos negativos de cada tipo de desenlace.
 - -p: realiza la predicción de un dataset dado por parámetros.
- *constants.py*: contiene las constantes que se usa en los dos scripts principales.
- *checker.py*: objeto estático Checker que contiene métodos para revisar el formato del dato de una variable.
- *decoder.py*: objeto estático Decoder con métodos que decodifica un dataframe recibido.
- *fileManager.py*: objeto FileManager con métodos para administrar la creación y eliminación de archivos de guardado de los modelos.
- *model_training_scikit_learn.py*: cuaderno de Jupyter que muestra un ejemplo básico del entrenamiento que realiza el Sistema Experto.
- *requirements.txt*: fichero de texto con los paquetes de Python necesarios para ejecutar los scripts correctamente. Se debe ejecutar antes de probar los scripts.
- *main.py*: script principal que crea el objeto Controlador y ejecuta el comando.

A.2. Instrucciones para ejecutar el script

Una vez descargado el repositorio, se necesita crear un entorno virtual para ejecutar los scripts de Python. Se recomienda instalar Anaconda, ya que permite crear entornos virtuales que incluyen una serie de paquetes que son utilizados en el código desarrollado, por ejemplo, Numpy, que ya se encuentra instalado en el entorno de Anaconda.

All applications base (root)

Figura A.1: Entorno por defecto de Anaconda

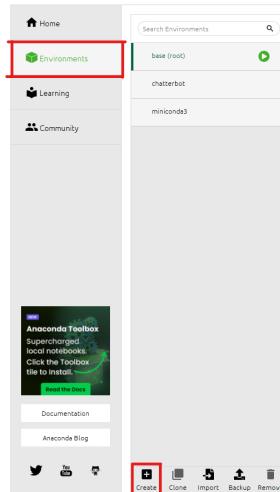


Figura A.2: Página de Environment de Anaconda y botón de creación de entornos

A.2.1. Entornos de Python con Anaconda

Anaconda de por sí ya tiene creado un entorno llamado *base (root)*.

Sin embargo, también existe la opción de crear entornos nuevos aparte del principal. Para ello, siguiendo la interfaz de la Figura 4.2, se debe ir al apartado *Environment*. Una vez dentro, aparecerá una lista de los entornos creados. Se debe dar al botón *Create* situado debajo de la lista.

Esto abre una pequeña ventana, donde se debe escribir el nombre del entorno y escoger la versión de Python. En nuestro caso, es la versión de Python 3.19.9. De todas formas, se pueden mostrar nuevas versiones de Python actualizando el Anaconda Navigator o descargando de forma manual la versión requerida desde la página oficial de Python.

A.2.2. Conexión del entorno virtual con Visual Studio Code

Una vez finalizado este paso, el siguiente es conectar el nuevo entorno, o el *base* si no se ha creado ninguno, a Visual Studio Code.

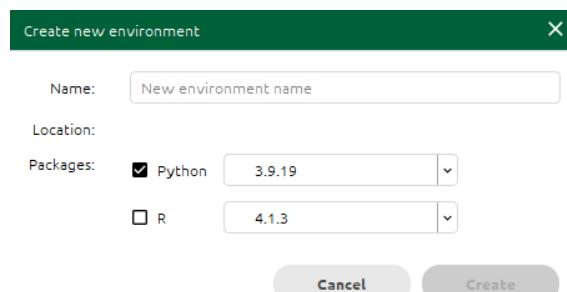


Figura A.3: Ventana de creación de nuevo entorno de Python en Anaconda

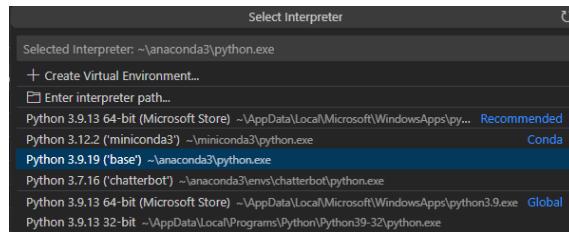


Figura A.4: Panel de selección de entorno de Python en Visual Studio Code

Para ello, se debe abrir el panel de comandos (CTRL+Shift+P) y seleccionar la opción **Python:Select Interpreter**.

Una vez ahí, seleccionamos el entorno que queremos. Finalmente, se abre un terminal desde dónde se podrán descargar los paquetes del proyecto y ejecutar los scripts.

A.2.3. Instalación de los paquetes de Python

Posteriormente, se tiene que ejecutar el archivo requirements.txt, donde se encuentran los paquetes necesarios para instalar en el entorno. Como se mencionó anteriormente, este archivo se encuentra ubicado también en la carpeta *scripts*.

```
pip install -r requirements.txt
```

Es importante mencionar que si no se dispone de La línea de comandos pip en el entorno o en el sistema (Windows o Linux), por lo que, si no dispone de dicho comando, es necesario instalarlo.

Como alternativa, si la ejecución de este archivo falla, se pueden instalar los paquetes de forma manual directamente desde la consola de Anaconda.

Finalmente, se accede a la carpeta *scripts* del repositorio, y una vez dentro se debe ejecutar el script *main.py*, junto con los parámetros necesarios, dependiendo de la opción que se quiere ejecutar.

Por ejemplo, si queremos consultar el tipo de modelo seleccionado, se debe ejecutar el siguiente comando:

```
python main.py -mt
```

De todas formas, todos los posibles comandos aparecen explicados y con ejemplos en un comando *help* implementado directamente en el Controlador. Se ejecuta de la siguiente forma:

```
python main.py -h
```

Esto devolverá el siguiente resultado:

```
(base) C:\Users\alda1\Downloads\Universidad\Curso23_24\TFM\Proyecto\TFM-2324-SIV\AlTA\scripts>python main.py -h
main.py [OPTION] [DATASET][MODEL TYPE][MODEL SAVE FILENAME]

OPTION
-h      Prints the help message.
-mt     Print the current model type selected.
        If the following parameter [MODEL TYPE] is a model type,
        the configuration will be modified to this new model type.
        The possible values are: autoinforme, familia, profesional
        Example: python main.py -mt autoinforme
-st     Print the current score type selected.
        If the following parameter [MODEL TYPE] is a score type,
        the configuration will be modified to this new score type.
        The possible values are: accuracy, precision, recall, f1_score
        Example: python main.py -st accuracy
-lst    Return an object of list of save files. If a model type [MODEL TYPE] is provided,
        it returns all the save files of that model.
        Otherwise, it returns an object with all the save files of all model types.
        Example: python main.py -lst autoinforme
-fs    Return an object with the stats of a given save file name in [MODEL SAVE FILENAME].
        Example: python main.py -fs model_autoinforme_20240607181100.sav
-rm    Remove a save filename of a model given its name in [MODEL SAVE FILENAME].
        Example: python main.py -rm model_autoinforme_20240607181100.sav
-t      Train a model, that can be "autoinforme", "familia" and "profesional" given a dataset [DATASET].
        In order to train the model correctly, a model type and a score type previously.
        If not, the training will not work.
        Example: python main.py -t datasets/autoinforme/dataset1.csv
-p      Return a list of predicted results given a dataset [DATASET].
        Example: python main.py -p datasets/autoinforme/dataset5.csv
```

Figura A.5: Comando Help del Sistema Experto

