Escuela Politécnica Superior de Jaén

Grado en Ingeniería Informática

UNIVERSIDAD DE JAÉN
Escuela Politécnica Superior de Jaén

Trabajo Fin de Grado

CLASIFICACIÓN AUTOMÁTICA DE NIVEL DE ESTRÉS EN PERSONAL SANITARIO

Alumno: William David Guerra Robledo

Tutor: María Teresa Martín Valdivia
Dpto: Departamento de Informática

Junio, 2023

(Página intencionalmente en blanco)

Universidad de Jaén

Departamento de Informática

Doña María Teresa Martín Valdivia, tutora del Trabajo Fin de Grado titulado: '**Clasificación automática de nivel de estrés en personal sanitario**', que presenta Don William David Guerra Robledo, otorga el visto bueno para su entrega y defensa en la Escuela Politécnica Superior de Jaén.

Jaén, Junio de 2023

El alumno:

GUERRA ROBLEDO WILIAM DAVID - 77647047J

Firmado digitalmente por GUERRA ROBLEDO WILIAM DAVID - 77647047J
Fecha: 2023.06.29 18:41:59 +02'00'

William David Guerra Robledo

La tutora:

MARTIN VALDIVIA MARIA TERESA - 26006595N

Firmado digitalmente por MARTIN VALDIVIA MARIA TERESA - 26006595N
Nombre de reconocimiento (DN): c=ES, serialNumber=IDCES-26006595N, givenName=MARIA TERESA, sn=MARTIN VALDIVIA, cn=MARTIN VALDIVIA MARIA TERESA - 26006595N
Fecha: 2023.06.29 13:03:56 +02'00'

María Teresa Martín Valdivia

(Página intencionalmente en blanco)

Agradecimientos

Lo primero de todo es agradecer a todas las personas que me han apoyado en el tiempo transcurrido durante la realización de este proyecto. A mis padres por estar cada día presentes en mi vida y apoyándome en los buenos y malos momentos. A mi hermana por darme consejos para este proyecto y a mis amigos y compañeros de clase por las ideas que me dieron y por los momentos de descanso que hemos compartido juntos.

Por otra parte, me gustaría agradecer en especial a dos personas que han hecho posible que pudiese llevar a cabo este trabajo. En primer lugar a mi tutora María Teresa Martín Valdivia por ofrecerme la oportunidad de trabajar junto a ella en el grupo de investigación y en segundo lugar a Flor Miriam Plaza del Arco por proporcionarme el conocimiento y recursos necesarios para hacer este proyecto. Gracias a estas dos personas he podido conocer un campo de la investigación que desconocía y que, mirándolo en retrospectiva, es bastante potente y bello.

Y por último a todos aquellos profesores y profesoras que me han hecho descubrir el campo de la informática y el cual me ha enseñado un conocimiento que jamás pensé que llegaría a aprender

Sinceramente, muchísimas gracias por vuestra colaboración y apoyo.

Ficha del Trabajo Fin de Título	
Titulación	Grado en Ingeniería Informática
Modalidad	Trabajo teórico/Experimental
Especialidad (solo TFG)	Sistemas de Información
Mención (solo TFG)	Tratamiento Inteligente de la Información
Idioma	Español
Tipo	Específico
TFT en equipo	No
Autor/a	William David Guerra Robledo
Fecha de asignación	21/03/2023

Descripción corta	El nivel de estrés en el personal sanitario es un problema que va en aumento, más aún, tras la pandemia. Desde el sindicato de enfermería SATSE se ha realizado una encuesta durante los años 2012, 2017 y 2021 para comprobar precisamente algunos temas de salud mental por parte de los profesionales de la salud. Este TFG propone el desarrollo software para la implementación de diferentes modelos de clasificación automática utilizando herramientas de aprendizaje automático y técnicas de Procesamiento de Lenguaje Natural (PLN). Concretamente se pretende desarrollar un prototipo que a partir de los datos de encuestas realizados a profesionales clínicos sea capaz de determinar si el profesional se encuentra o no estresado.

NORMAS APLICADAS EN ESTE DOCUMENTO	
LOCALES	
TFT-UJA:2017	Normativa de Trabajos Fin de Grado, Fin de Máster y otros Trabajos Fin de Título de la Universidad de Jaén (Normativa marco UJA aprobada en Consejo de Gobierno)
TFT-EPSJ:2017	Normativa sobre Trabajos Fin de Grado y Fin de Máster en la Escuela Politécnica Superior de Jaén (Normativa EPSJ aprobada en Junta de Escuela)
TFT-EPSJ	Criterios de evaluación y normas de estilo para TFG y TFM de la Escuela Politécnica Superior de Jaén
NACIONALES E INTERNACIONALES	
ISO 2145:1978	Documentación - Numeración de divisiones y subdivisiones en documentos escritos
UNE 50132:1994	Traducción de la ISO 2145
APA 6ª edición	Estilo de referencias y citas de APA (American Psychological Association)

NORMAS UTILIZADAS COMO BASE O REFERENCIA	
NACIONALES	
UNE 157001:2014	Criterios generales para la elaboración formal de los documentos que constituyen un proyecto técnico
UNE 157801:2007	Criterios generales para la elaboración de proyectos de sistemas de información
Estas normas se han utilizado como base o referencia para la inclusión de algunos contenidos y definiciones sobre elaboración de proyectos, entendiendo como proyecto la documentación consensuada entre una empresa y un cliente, que da lugar al perfeccionamiento de un contrato para la elaboración de una obra o la prestación de un servicio. Por consiguiente, no debe esperarse la aplicación de estas normas en cuanto a la completitud de los contenidos ni a la organización de los mismos.	

Contenido

1 Especificación del trabajo 13
1.1 Introducción 13
1.2 Objetivos del trabajo 15
1.3 Antecedentes y estado del arte 16
1.4 Descripción de la situación de partida 18
1.4.1 Preguntas sobre datos generales 20
1.4.2 Preguntas subjetivas sobre el entorno laboral antes de la pandemia 20
1.4.3 Preguntas subjetivas sobre el entorno laboral desde el inicio de la pandemia 21
1.4.4 Preguntas objetivas sobre síntomas de estrés 21
1.4.5 Descripción del entorno actual 22
1.5 Alcance 22
1.6 Hipótesis y restricciones 23
1.6.1 Hipótesis 23
1.6.2 Restricciones 23
1.7 Estudio de alternativas y viabilidad 25
1.8 Descripción de la solución propuesta 26
1.9 Material y métodos 26
1.9.1 Pregunta Q9. “¿Crees que estás estresado/a?” 28
1.9.2 Pregunta Q11. “¿Dirías que sientes agotamiento emocional?” 29
1.9.3 Encuesta Q13. “¿Dirías que estás quemado en tu trabajo?” 30
1.10 Tecnologías utilizadas 31
1.11 Metodología de desarrollo de software 33
1.12 Estimación del tamaño y esfuerzo 35
1.13 Planificación temporal 37
1.14 Presupuesto 40
1.14.1 Elementos software 41
1.14.2 Elementos hardware 41
1.14.3 Personal y servicios de las instalaciones 42
1.14.4 Coste total del proyecto 44
2 Diseño inicial 45
2.1 Análisis y diseño del sistema 45
2.1.1 Análisis 45
2.1.2 Diseño 46
2.1.2.1 Arquitectura base de un sistema de clasificación automática 46
2.1.2.2 Arquitectura Transformers 49
3 Desarrollo 53
3.1 Primera Iteración: investigación y toma de contacto con el problema 53
3.2 Segunda Iteración: análisis y estudio del conjunto de datos 54
3.2.1 Balanceo de los datos 54
3.2.2 Análisis de las observaciones 58
3.2.3 Nubes de palabras de la clase “estresada” 61
3.3 Tercera iteración: Entrenamiento de los modelos 64

3.4 Pruebas finales 66
4 Experimentación, resultados y discusión 66
4.1 Experimentaciones y pruebas 66
4.1.1 Regresión logística de Scikit Learn 68
4.1.2 Uso del lexicón iSOL 68
4.1.3 Modelos del lenguaje Transformer sin ajuste de hiper-parámetros 70
4.1.4 Modelos del lenguaje con optimización de hiper-parámetros 73
4.2 Resultados y discusión 75
4.3 Análisis de errores 77
4.3.1 Falsos negativos 79
4.3.2 Falsos positivos 80
4.4 Realización de la interfaz gráfica 80
5 Conclusiones y trabajos futuros 82
5.1 Trabajos futuros 83
6 Bibliografía 84

Índice de ilustraciones

Ilustración 1.1 Gráfica sobre el porcentaje de personal sanitario estresado/a 14
Ilustración 1.3 Tabla comparativa del modelo GPT-4 con los demás modelos del lenguaje [14] 18
Ilustración 1.4 Esquema del modelo incremental 34
Ilustración 1.5 y 1.6 Diagrama de Gantt basado en la planificación temporal 40
Ilustración 2.1 Diagrama de flujo sobre la estructura básica del desarrollo de un sistema de Machine Learning 47
Ilustración 2.2 Gráfica sobre la búsqueda de parámetros 49
Ilustración 2.3 Resultados de la traducción del inglés al alemán en diferentes modelos neuronales junto al modelo Transformer 50
Ilustración 2.4 Configuración básica de una red neuronal 51
Ilustración 1.2 Esquema sobre la arquitectura Transformers [43] 52
Ilustración 3.1 Balanceo de los datos en Q9 55
Ilustración 3.2 Balanceo de los datos en Q11 56
Ilustración 3.3 Balanceo de los datos en Q13 57
Ilustración 3.4 Nube de palabras de adjetivos exclusivos de la clase “estresada” 62
Ilustración 3.5 Nube de palabras de verbos exclusivos de la clase “estresada” 63
Ilustración 3.6 Nube de palabras de adverbios exclusivos de la clase “estresada” 63
Ilustración 3.7 Nube de palabras de nombres exclusivos de la clase “estresada” 64
Ilustración 4.1 Flujo de trabajo de los modelos Transformers de Hugging Face 71
Ilustración 4.2 Matriz de confusión del experimento 78
Ilustración 4.3 Interfaz gráfica de la aplicación desarrollada 81
Ilustración 4.4 Ejemplo de predicción con la aplicación 82

Índice de tablas

Tabla 1.1 Ejemplo de una instancia del conjunto de datos 19
Tabla 1.2 Ejemplo de las preguntas realizadas en la encuesta 27
Tabla 1.3 Formato del subconjunto de datos de la Q9 28
Tabla 1.4 Formato del subconjunto de datos de la Q11 29
Tabla 1.5 Formato del subconjunto de datos de la Q13 30
Tabla 1.6 Tabla de parámetros para el modelo COCOMO con sus distintos modos 37
Tabla 1.7 Planificación temporal del Trabajo de Fin de Grado 40
Tabla 1.8 Coste de los elementos software 41
Tabla 1.9 Coste de los elementos hardware 42
Tabla 1.10 Coste de los servicios auxiliares 44
Tabla 1.11 Presupuesto final 44
Tabla 3.1 Estadística descriptiva de las observaciones 58
Tabla 3.2 Estadística descriptiva de las observaciones con el umbral aplicado 59
Tabla 3.3 Categoría gramatical de las palabras de las observaciones 60
Tabla 3.3 Frecuencia de las emociones a nivel de palabra 61
Tabla 4.1 Resultados de clasificación con Regresión Logística 68
Tabla 4.2 Resultados de clasificación con lexicón iSOL 70
Tabla 4.3 Resultados de clasificación con Transformers sin hiper-parámetros 73
Tabla 4.4 Resultados de clasificación con Transformers con hiper-parámetros 75
Tabla 4.5 Resultados de toda la experimentación 76
Tabla 4.6 Ejemplos de falsos negativos 79
Tabla 4.7 Ejemplos de falsos positivos 80

1 ESPECIFICACIÓN DEL TRABAJO

1.1 Introducción

A día de hoy, una gran parte de la población tiene constancia de la desafortunada epidemia que se vivió a nivel mundial y que empezó a principios del año 2020. El Coronavirus es una enfermedad infecciosa que cambió el mundo drásticamente y repercutió en la forma de vivir de la mayoría de personas. Ahora mismo la epidemia fue declarada como finalizada el 5 de mayo de 2023 por la OMS [1] y su impacto en la población mundial va disminuyendo de manera progresiva si comparamos los datos de personas infectadas que había a mediados del 2020 con los de ahora. Algo que cabe destacar es el papel importante que tuvo que desempeñar el personal sanitario a la hora de atender, cuidar y controlar a la gente enferma de este virus. Sin embargo, esto ha tenido consecuencias, empezando por el estrés y tensión que tuvieron que sufrir, entre otros profesionales sanitarios, los enfermeros y enfermeras durante esos meses de hospitales completos de enfermos, pocas horas de sueño y en una constante exposición a un virus que perjudica a la salud de las personas. Un estudio realizado por el Sindicato de enfermería SATSE reveló que del 84% del personal sanitario que sufría de estrés en 2017, en 2021 ese porcentaje aumentó al 85,71% [2]. Este porcentaje tan elevado repercute negativamente tanto en la salud mental de los diferentes profesionales sanitarios como en la atención hacia el paciente y su seguridad. En este mismo estudio también se observa un aumento del agotamiento emocional hasta un 77,56% y un 63% de enfermeros y enfermeras se sienten quemados de su trabajo. Todo esto hace que incluso sufran síntomas de diferentes enfermedades como ansiedad, dolor en la espalda, depresión, insomnio, entre otros muchos.

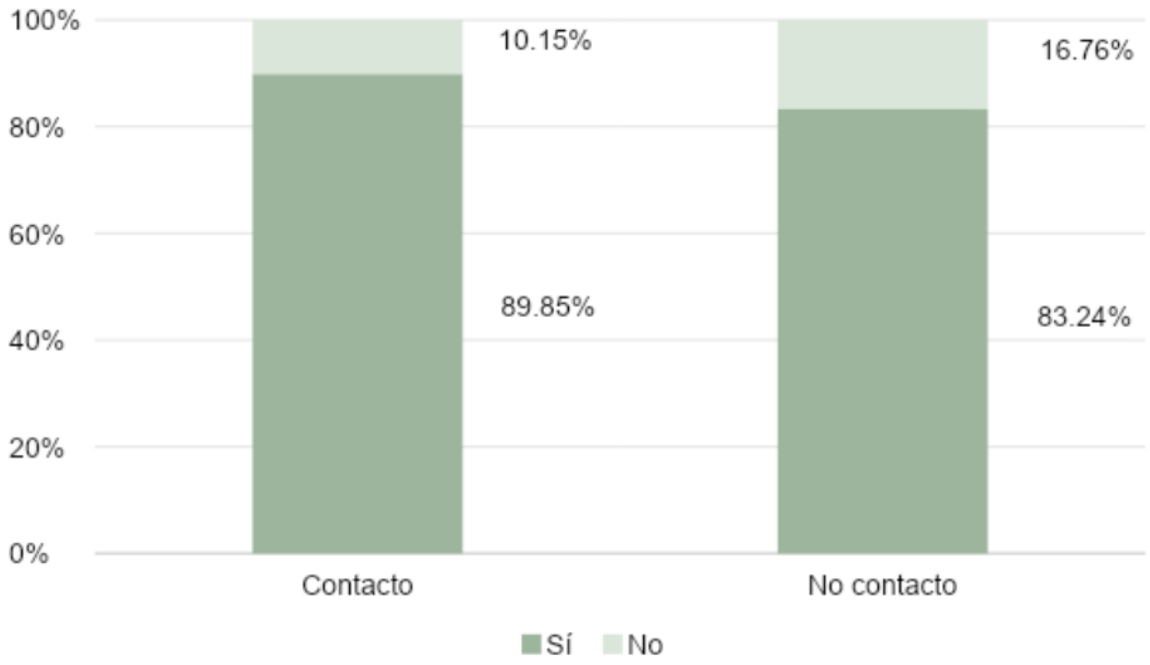


Ilustración 1.1 Gráfica sobre el porcentaje de personal sanitario estresado/a

Los porcentajes mencionados anteriormente son un grave indicio de que el sistema sanitario español ha pasado por un periodo complicado y el grado de satisfacción y realización de la gente que trabaja en este sector ha disminuido. Esto no se puede ignorar ya que el sistema sanitario de España se considera el octavo mejor sistema de atención sanitaria del mundo [3], además de la importancia que ha adquirido la salud mental en las personas en estos últimos años. Por ello, la propuesta de este trabajo consiste en detectar con mayor rapidez y eficacia estos signos de cansancio y estrés en el personal a través de herramientas como la Inteligencia Artificial y, para ser más precisos, mediante diferentes métodos basados en **Procesamiento del Lenguaje Natural (PLN)**.

En esta memoria de trabajo nos centraremos en la implementación, estudio y comparación de diferentes modelos de clasificación binaria de etiquetas/clases para determinar si el personal sanitario está estresado o no utilizando herramientas de aprendizaje automático y técnicas de PLN. Para ello, se utilizó como conjunto de datos una encuesta realizada por el SATSE a los enfermeros y enfermeras de España. Dicha encuesta consiste en una serie de preguntas de Sí/No (“¿Crees que se ha empeorado la atención que se presta en tu unidad/centro?”) o preguntas de

valoración (“Del siguiente material de protección por favor, indícanos qué concepto se adecua más a la cantidad de material de protección del que disponías…”).

Con estos datos, se utilizaron varios modelos de clasificación de etiquetas implementados en librerías de Python como Scikit-Learn a través de un pipeline en donde primero se aplica un TF-IDF y luego esos valores numéricos son introducidos al modelo para entrenarlo. Después, se usaron los modelos del lenguaje con el fin de mejorar las predicciones.

Como un pequeño acercamiento al PLN, se podría definir como una parte de la Inteligencia Artificial que investiga y formula mecanismos computacionales y efectivos los cuales tienen como objetivo facilitar la interacción entre las personas y las máquinas mediante el empleo de un lenguaje más natural y menos rígido y restrictivo. Este campo del PLN ha ido evolucionando y su interés ha ido en aumento debido al gran abanico de aplicaciones que tiene: traducción automática, recuperación de información, sistemas de búsqueda de respuestas, extracción de conocimiento, reconocimiento y generación del habla, etc. Por todas estas aplicaciones, diversas entidades gubernamentales y negocios han puesto su atención en esta rama de estudio con el objetivo de optimizar tanto tiempo como costes.

1.2 Objetivos del trabajo

Los diferentes objetivos que tiene esta propuesta son los siguientes:

- Realizar un estudio de los diferentes modelos de clasificación y de las diferentes técnicas de PLN y aprendizaje automático aplicables a este problema.
- Diseñar y desarrollar un sistema para la clasificación automática del nivel de estrés en el personal sanitario a través de la documentación proporcionada por el SATSE con el objetivo de poder predecir y clasificar si una persona está o no estresada en su trabajo.
- Diseñar e implementar una aplicación que permita la visualización de los resultados en una interfaz web de forma intuitiva y sencilla.

- Redactar una memoria que recoja todo el trabajo desarrollado así como los manuales de instalación y de usuario.

1.3 Antecedentes y estado del arte

Para realizar un sistema de clasificación automática del nivel de estrés, nos apoyaremos sobre todo en los denominados **modelos de lenguaje**. El modelado del lenguaje consiste principalmente en la aplicación de diferentes técnicas estadísticas y probabilísticas con el objetivo de, dada una frase o secuencia de palabras, determinar la probabilidad de aparición de alguna palabra en concreto o proporcionar una predicción de las palabras que pueden aparecer en dicha frase [4]. Hay diferentes tipos o enfoques a la hora de realizar un modelo del lenguaje, desde los que adoptan una perspectiva probabilística a través del uso de N-gramas hasta enfoques basados en redes neuronales mediante *Word Embeddings* o incluso híbridos de los dos enfoques.

A lo largo de estos últimos años, el desarrollo y optimización de diferentes modelos del lenguaje ha crecido de manera abrupta. Desde sus inicios en 2017 con la propuesta de Google de la arquitectura de los Transformers [5][6], pasando por el debut del *Generative Pre-Trained Transformer* (popularmente conocido por sus siglas GPT) con 117 millones de parámetros y basándose en la arquitectura de los Transformers [7]. Siguiendo a este modelo surgió en 2018 el modelo del lenguaje desarrollado por investigadores de Google el cual se llamaba Bidirectional Encoder Representations from Transformers (simplificando por sus siglas, BERT [8]) con 110 millones de parámetros. En este trabajo se han utilizado diversos modelos los cuales se basaron en esta arquitectura de Google como el *Spanish BERT* o *Spanish RoBERTa* [9][10].

Pasando por numerosos modelos destacando GPT-2, LaMDA (Language Model for Dialogue Applications) y el BigScience Large Open-science Open-access Multilingual Language Model (BLOOM), llegamos al día de hoy en donde los parámetros llegan a la escala de cientos de miles de millones (por ejemplo BLOOM

con 176 mil millones de parámetros) y la mayoría de estos sistemas y modelos están al alcance de cualquier usuario a través de diversos portales como Hugging Face o spaCy para poder utilizarlos como modelos pre-entrenados para un posterior *fine tuning*, proceso el cual es muy importante ya que la mayoría de los lenguajes de modelos pre-entrenados han hecho uso de un corpus enorme en donde hay multitud de palabras relacionadas con varios campos o materias. Si nuestro corpus es de un campo muy específico (en este caso, palabras relacionadas al ámbito sanitario), va a tratar esas palabras como tokens raros y eso va a hacer que su rendimiento sea menor. Para evitar esto, antes de entrenar el modelo con nuestros datos, se realiza un *fine tuning* para que así el modelo se ajuste mejor al dominio que va a tratar [11]. También se conoce como *domain adaptation* y fue popularizado en 2018 por **ULMFiT** [12], uno de las primeras arquitecturas neuronales en donde se aplico el aprendizaje transferido en el PLN.

Y por último, hay que hablar sobre el modelo de lenguaje que ha sido una revolución y ha dado a conocer al mundo entero el potencial del PLN y de los modelos de lenguaje: el modelo GPT-4. Un modelo de lenguaje multimodal el cual, aparte de poder recibir *prompts* de texto, también puede recibir imágenes, capturas de pantalla o incluso bocetos a mano alzada. Cuenta con una mayor capacidad de recibir inputs más complejos y tiene un mejor rendimiento en comparación a otros modelos del lenguaje considerados como estado del arte [13].

	GPT-4 Evaluated few-shot	GPT-3.5 Evaluated few-shot	LM SOTA Best external LM evaluated few-shot	SOTA Best external model (incl. benchmark-specific tuning)
MMLU [49] Multiple-choice questions in 57 subjects (professional & academic)	**86.4%** 5-shot	70.0% 5-shot	70.7% 5-shot U-PaLM [50]	75.2% 5-shot Flan-PaLM [51]
HellaSwag [52] Commonsense reasoning around everyday events	**95.3%** 10-shot	85.5% 10-shot	84.2% LLaMA (validation set) [28]	85.6 ALUM [53]
AI2 Reasoning Challenge (ARC) [54] Grade-school multiple choice science questions. Challenge-set.	**96.3%** 25-shot	85.2% 25-shot	85.2% 8-shot PaLM [55]	86.5% ST-MOE [18]
WinoGrande [56] Commonsense reasoning around pronoun resolution	**87.5%** 5-shot	81.6% 5-shot	85.1% 5-shot PaLM [3]	85.1% 5-shot PaLM [3]
HumanEval [43] Python coding tasks	**67.0%** 0-shot	48.1% 0-shot	26.2% 0-shot PaLM [3]	65.8% CodeT + GPT-3.5 [57]
DROP [58] (F1 score) Reading comprehension & arithmetic.	80.9 3-shot	64.1 3-shot	70.8 1-shot PaLM [3]	**88.4** QDGAT [59]
GSM-8K [60] Grade-school mathematics questions	**92.0%*** 5-shot chain-of-thought	57.1% 5-shot	58.8% 8-shot Minerva [61]	87.3% Chinchilla + SFT+ORM-RL, ORM reranking [62]

Ilustración 1.3 Tabla comparativa del modelo GPT-4 con los demás modelos del lenguaje [14]

1.4 Descripción de la situación de partida

Para el desarrollo de nuestro sistema de clasificación automática de estrés, primeramente se ha contado con un conjunto de datos para el entrenamiento de nuestro modelo o sistema. En nuestro caso, consiste en una hoja de cálculo en donde cada fila corresponde a un/a enfermero/a que ha respondido a una encuesta realizada por el SATSE. Dicha encuesta se realiza cada 5 años y las preguntas en cuestión no suelen variar, a excepción de la encuesta de 2021, en donde hay ciertas preguntas con relación al COVID-19 (Abreviatura comúnmente utilizada para referirse al virus del Coronavirus, proveniente del inglés *COronaVIrus Disease*). En resumen, las preguntas se dividen en 4 bloques:

- **Datos generales**: preguntas de aspecto general como el sexo, la edad, provincia y años de experiencia laboral.

- **Preguntas de carácter subjetivo sobre el entorno laboral antes de la pandemia**: orientadas a obtener información sobre la percepción de esa persona sobre su entorno laboral.
- **Preguntas de carácter subjetivo sobre el entorno laboral durante la pandemia.**
- **Preguntas objetivas sobre el padecimiento de sintomatología asociada al estrés laboral.**

Nosotros, a la hora de entrenar el modelo, nos centraremos especialmente en 3 preguntas que pertenecen al segundo bloque. Estas preguntas son:

- ¿Crees que estás estresado/a? → Q9
- ¿Dirías que sientes agotamiento emocional? → Q11
- ¿Dirías que estás quemado en tu trabajo? → Q13

Más adelante se especificará de qué manera hemos trabajado con las respuestas a estas preguntas pero hay que tener en cuenta que esas respuestas actuarán como etiqueta o clase del conjunto de datos (en este caso, una etiqueta de tipo binaria con dos únicas respuestas de sí/no). Y por último, los valores que se utilizarán como input al entrenamiento será la columna de **"Observaciones"** en donde la gente expresa mediante lenguaje natural su pensamiento y postura frente a la situación que han vivido.

Observaciones	Q9 (Estrés)	Q11 (Agotamiento Emocional)	Q13 (Quemado del trabajo)
"Nefasta gestión de la pandemia…"	No	No	Sí

Tabla 1.1 Ejemplo de una instancia del conjunto de datos

1.4.1 Preguntas sobre datos generales

Las preguntas de esta categoría tienen el propósito de recoger datos relacionados con la persona que está realizando el cuestionario como por ejemplo el sexo, la edad, provincia y años de experiencia laboral.

- **Sexo:** esta variable se debe tener en cuenta ya que más del 80% de los profesionales de enfermería son mujeres y este factor en españa tiene consecuencias como que un gran porcentaje de compagina tanto una jornada laboral como familiar y que se presenta diferentes expectativas de exito profesional con respecto a sus compañeros varones.
- **Edad y experiencia profesional:** estos aspectos suelen ir directamente relacionados y podemos deducir cierto grado de estabilidad laboral de la persona gracias a estos factores. También cabe destacar que una persona con una gran experiencia laboral posiblemente tenga más responsabilidades y preocupaciones que otra persona que sea más novel en su trabajo
- **Provincia:** España es un país que está compuesto de diversas comunidades autónomas y cada una de ellas tiene una gestión del sistema sanitario diferente. Por tanto, cada grupo tendrá unas condiciones distintas, en especial cuando ocurrió la pandemia ya que el impacto de esta era mayor o menor en distintos puntos del país.

1.4.2 Preguntas subjetivas sobre el entorno laboral antes de la pandemia

Este grupo de preguntas se preguntaba al personal sobre las condiciones laborales y su entorno en el intervalo de tiempo entre 2017 (cuando se realizó el estudio anterior) y febrero de 2020. Las preguntas son las mismas que las del grupo siguiente pero con la diferencia de que estas se quería obtener datos pre-pandemia.

1.4.3 Preguntas subjetivas sobre el entorno laboral desde el inicio de la pandemia

Esta categoría cuenta con 13 preguntas, las cuales se dividen en dos subgrupos en función de lo que se preguntaba:

En las 8 primeras preguntas se quería obtener la opinión y perspectiva de los entrevistados sobre el sistema sanitario español y las actuaciones directas de los distintos profesionales de sus respectivas unidades.

De la pregunta 9 a la 13 se quiere consultar sobre la satisfacción personal y el estado físico y mental de los distintos enfermeros y enfermeras.

Después de estas preguntas que están presentes tanto en las preguntas subjetivas antes de la pandemia y durante la pandemia, hay preguntas más específicas haciendo referencia a la situación de la pandemia del Coronavirus. Un ejemplo es el siguiente “¿Has sentido miedo a contagiar a tus familiares y/o amigos?” o “¿Crees que tu estado de estrés ha aumentado al trabajar con pacientes COVID?”, entre otros. También hay un apartado dedicado a preguntar sobre si están satisfechos con la cantidad de material de protección como mascarillas, guantes o gel hidroalcohólico.

1.4.4 Preguntas objetivas sobre síntomas de estrés

Al ser un cuestionario en donde el estado de estrés está presente en la mayoría de las personas encuestadas, es importante profundizar más en el problema y determinar qué síntomas son los más frecuentes y a cuáles hay que prestar más atención. Dentro de estas preguntas se divide en una sección donde se pregunta sobre síntomas que afectan directamente al **estado físico** como tensión muscular o problemas de estómago y otras donde se interroga sobre la percepción de la persona de **síntomas emocionales y de conducta** como un retraso en la toma de decisiones o cambios de humor.

1.4.5 Descripción del entorno actual

Una de las muchas consecuencias que tuvo la pandemia fue el aumento del porcentaje de abandono por parte del personal de enfermería en muchos hospitales debido al continuo estrés, a las condiciones precarias que tenían que vivir y, sobre todo, a los pocos recursos que se les administraba a veces. Según un estudio en el que participaron alrededor de 67.000 trabajadores sanitarios de 21 países, se demostró que el porcentaje de trabajadores que mostraban síntomas de estrés fue del 23,2%, la depresión del 22,8%, la ansiedad del 21,4% y los trastornos del sueño del 38% [39].

Uno de los objetivos que pretende cumplir este proyecto es el de proporcionar al sistema sanitario, en especial al sector administrativo de los diferentes hospitales y centros, una herramienta que permita detectar el estrés y agotamiento del personal sanitario con mayor rapidez y precisión con las consecuencias que tendría como por ejemplo hacer una mejor toma de decisiones con respecto al entorno laboral, prestar más atención a las necesidades del personal sanitario y fomentar una mejor relación entre los compañeros de trabajo.

1.5 Alcance

Los diferentes entregables que se incluirán en este proyecto son los siguientes:

- **Memoria**: este documento contendrá toda la información relativa al proyecto, esto incluye los diferentes resultados, bibliografía, experimentación, manuales de uso del sistema resultante, entre otros.
- **Código fuente**: se incluirá el código del sistema con los que se llevaron a cabo la experimentación del proyecto.

- **Resultados**: tanto en la memoria como en el código fuente debido a la naturaleza de los ficheros con formato .ipynb (Jupyter Notebook) se incluirá los diferentes resultados obtenidos.

1.6 Hipótesis y restricciones

1.6.1 Hipótesis

Las principales hipótesis de las que se parte a la hora de realizar este TFG son las siguientes:

- **Uso de modelos pre-entrenados del lenguaje:** Cuando se trata de experimentar con diversas técnicas y herramientas para desarrollar un sistema de clasificación binaria, se anticipa que los modelos que hayan sido ajustados a partir de un modelo pre-entrenado con nuestro conjunto de datos arrojarán mejores resultados en comparación con el uso de clasificadores binarios simples.
- **Análisis lingüístico del corpus:** Después de realizar un análisis al campo de 'Observaciones' mediante herramientas de PLN, dicho análisis nos dará una mayor comprensión de este corpus y, por tanto, se podrá ajustar parámetros a la hora de entrenar el modelo con un mayor criterio.
- **Tamaño del corpus:** Se hace la suposición de que a medida que aumente la cantidad de instancias en el conjunto de entrenamiento, el modelo tendrá un mejor rendimiento en términos de predicción y mejores resultados.

1.6.2 Restricciones

A la hora de desarrollar este TFG nos hemos encontrado con una serie de restricciones que debemos abordar para finalizar el trabajo.

La primera restricción a tener en cuenta es el límite de tiempo, a la hora de realizar el Trabajo de Fin de Grado. Esta es una asignatura de 12 créditos lo que supone una duración teórica total de 300 horas. Por lo tanto, como restricción de partida tenemos que considerar la duración del trabajo.

La segunda restricción a la que se enfrenta este proyecto es a los modelos de lenguaje que hay publicados actualmente y los que se han utilizado. Más adelante se dará una descripción con mayor profundidad de estos modelos pero el principal inconveniente es en relación a los datos con los que se ha entrenado estos modelos. Por ejemplo, el modelo de lenguaje **BETO: A Spanish BERT**, es un modelo basado en BERT y entrenado en un gran corpus en español [15]. Este corpus se compone de 300 millones de líneas de texto y unos 3 mil millones de tokens. Este texto proviene de diversas fuentes como charlas TED, Wikipedia, etc [16]. Al ser una gran corpus pero no estar especializado en el campo del estrés o de la sanidad, es de suponer que cuando se entrene el modelo y se realice sus pertinentes predicciones sobre el subconjunto de datos de test, las diferentes métricas de evaluación serán bajas. Esto difiere de otros problemas como por ejemplo la detección de la ofensividad en redes sociales puesto que para este dominio ya hay modelos del lenguaje entrenados específicamente sobre corpus centrados en este problema y, por lo tanto, los resultados suelen ser mejores.

La tercera restricción tiene relación con el coste computacional que se requiere para entrenar los modelos y ejecutarlos. Para su entrenamiento se hace uso de la GPU para que así el tiempo de entrenamiento sea más corto. Sin embargo, a la hora de llevar a cabo el entrenamiento en Google Collaboratory, tanto el tamaño de la RAM como de la GPU es limitada, además de tener un tiempo de uso de la GPU limitado debido a que se usa el plan gratuito.

Y por último, la siguiente restricción está relacionada con los datos con los que contamos para realizar el *fine-tuning* a nuestro sistema. La mayor parte de estos datos necesitan un pre-procesamiento, además de tener la problemática del balanceo del conjunto de datos, ya que la mayoría de las instancias están

etiquetadas con una clase predominante y la otra clase cuenta con un número mucho menor de ejemplos.

1.7 Estudio de alternativas y viabilidad

En un primer acercamiento al problema, se propusieron diferentes alternativas, no solo para comparar resultados y tener una experimentación más abierta, sino también para ver posibles soluciones que se nos iba planteando a medida que se trataba el problema. Entre las diferentes alternativas se encuentran:

- **Lexicón iSOL** [17][18]: este lexicón realizado por el grupo de investigación SINAI consiste en una lista de palabras que indican la opinión de cada palabra, sin tener en cuenta el contexto. Se compone de un conjunto de palabras pertenecientes a un lexicón que mantiene el profesor Bing Liu [19]. Este lexicón, traducido automáticamente, consta de 2.509 palabras positivas y por 5.626 negativas. Este lexicón se ha usado como punto de partida, entrenando modelos de clasificación básicos en función de si una observación tenía un mayor número de palabras positivas o palabras negativas, es decir, la polaridad de la observación.
- **Modelos de clasificación en Scikit-Learn** [20]**:** antes de emplear directamente los modelos de lenguaje pre-entrenados, se decidió usar modelos básicos de aprendizaje automático que se encuentran en la librería de Python llamada “Scikit-Learn”, la cual está especializada y provee de numerosos algoritmos relacionados con el *machine learning* (cuya abreviatura es ML) y modelos de clasificación o clustering. Obviamente, para el uso de dichos modelos, previamente se aplica un modelo TF-IDF al texto incluido en las observaciones. El TF-IDF trata de cuantificar la importancia o el valor de un término en función a su frecuencia en un documento/observación (TF son las siglas en inglés de *Term Frequency*) y también tiene en cuenta la frecuencia inversa en el documento (IDF viene de *Inverse Document Frequency*) [21] ya que, si un término se repite mucho a lo largo de todas las

observaciones, se considera que dicho término es **poco discriminatorio** y, por tanto, no tiene mucho valor para diferenciar una observación de otra.

1.8 Descripción de la solución propuesta

La solución escogida para presentar es aplicar en una primera parte un análisis de las distintas observaciones mediante Procesamiento de Lenguaje Natural con el objetivo de encontrar palabras significativas u obtener la estructura general de las observaciones. La segunda parte de la solución presentada consiste en utilizar los modelos de lenguaje que se encuentran en la plataforma Hugging Face que tiene como objetivo desarrollar recursos de código abierto que permita al usuario integrar herramientas de Inteligencia Artificial en diversos productos y flujos de trabajo. Para ello, la IA (sigla de Inteligencia Artificial) debe ser accesible, optimizada y responsable y en esos principios se basa Hugging face [22]. Entre los distintos recursos que ofrece el portal, el que se usa principalmente es la librería ***Transformer*** que permite descargar, cargar y usar diversos modelos pre-entrenados, entrenarlos y ajustarlos al problema en concreto debido a que todos los modelos son clases de librerías de aprendizaje automático como PyTorch o TensorFlow. Estos modelos se centran en diferentes tareas, ya sea para detección de objetos, clasificación de imágenes, generación de texto, traducción automática y muchos más [23]. También cuenta con más de 35.000 conjuntos de datos para que los usuarios puedan entrenar modelos haciendo uso del fine-tuning a dichos datasets [24]. Y por último, se usarán herramientas para la búsqueda o ajuste de hiper-parámetros con el propósito de mejorar los resultados obtenidos en la segunda parte.

1.9 Material y métodos

El primer medio que se ha usado para el proyecto ha sido el conjunto de datos empleado para el entrenamiento de los diferentes modelos de lenguaje. Como

punto de partida, el conjunto de datos consta de 11.650 instancias que representan a cada persona del equipo sanitario que ha realizado la encuesta diseñada por el SATSE. Como se ha explicado anteriormente en el apartado 1.4 “Descripción de la situación de partida”, dicha encuesta se realiza cada 5 años y, en esta ocasión, a razón de la pandemia del Covid-19, hubo ciertas preguntas enfocadas a la situación que se vivió.

En general, y una vez declarada la pandemia:	**¿Han empeorado tus condiciones de trabajo en los últimos meses a consecuencia de la COVID 19?**	**¿Se ha deteriorado el ambiente laboral de tu unidad/centro en los últimos meses?**	**¿Crees que ha empeorado la atención que se presta en tu unidad/centro?**
...	Sí/No	Sí/No	Sí/No

Tabla 1.2 Ejemplo de las preguntas realizadas en la encuesta

Como se indicó, nos fijaremos en las preguntas relacionadas con el estrés y el agotamiento emocional que sufrió el personal. Concretamente se trata de 3 preguntas (Q9, Q11 y Q13). Una desventaja de este conjunto de datos es que, de las 11.650 filas con las que se cuenta en el estudio, muchas de ellas no pueden ser utilizadas para la experimentación que vamos a realizar debido a que los requisitos para que una fila sea válida para su posterior uso son:

- Que la persona haya respondido a **alguna** pregunta correspondiente de las 3 preguntas objetivo que hemos planteado, es decir, que no estén en blanco.
- Que dicha persona haya escrito alguna observación.

Dichas restricciones hacen que 8.305 filas hayan sido eliminadas del conjunto de datos debido a que no cumplían con las dos restricciones.

1.9.1 Pregunta Q9. "¿Crees que estás estresado/a?"

Columnas	Propósito	Valores
Respondent_id	ID del encuestado para mantener el anonimato.	XXXXXXXXXX
Collector_id	ID de la encuesta.	XXXXXXXXX
Date_created	Fecha en la que se realizó la encuesta	MM/DD/AAAA HH:MM:SS
Estresado	Etiqueta/Variable objetivo del dataset	Sí/No
Observaciones	Entrada de texto introducido por el/la enfermero/a	"No han contratado personal suficiente para afrontar esta pandemia…"
Personas estresadas		2617
Personas no estresadas		603

Tabla 1.3 Formato del subconjunto de datos de la Q9

Como se puede observar, en este subconjunto tenemos un total de 3.220 instancias, cada una de ellas con una etiqueta (en este caso, si está estresado o no) y su correspondiente casilla de la observación. A la hora de mirar el balance de los datos, vemos como hay un total de 2.617 personas que están clasificadas como estresadas (han contestado Sí a la pregunta Q9) y 603 que no lo están (han

contestado No a la pregunta Q9). Este desbalance en los datos va a suponer un gran problema que más adelante se describirá con mayor detalle.

1.9.2 Pregunta Q11. "¿Dirías que sientes agotamiento emocional?"

Columnas	Propósito	Valores
Respondent_id	ID del encuestado para mantener el anonimato.	XXXXXXXXXX
Collector_id	ID de la encuesta.	XXXXXXXXX
Date_created	Fecha en la que se realizó la encuesta	MM/DD/AAAA HH:MM:SS
Agotado emocionalmente	Etiqueta/Variable objetivo del dataset	Sí/No
Observaciones	Entrada de texto introducido por el enfermero/a	"No me importaría dejar el trabajo si tuviera otro"
Personas agotadas		2548
Personas no agotadas		689

Tabla 1.4 Formato del subconjunto de datos de la Q11

1.9.3 Encuesta Q13. "¿Dirías que estás quemado en tu trabajo?"

Columnas	Propósito	Valores
Respondent_id	ID del encuestado para mantener el anonimato.	XXXXXXXXXX
Collector_id	ID de la encuesta.	XXXXXXXXX
Date_created	Fecha en la que se realizó la encuesta	MM/DD/AAAA HH:MM:SS
Quemado del trabajo	Etiqueta/Variable objetivo del dataset	Sí/No
Observaciones	Entrada de texto introducido por el enfermero/a	"Deberían cambiar las condiciones laborales y hacernos contratos dignos."
Personas quemadas del trabajo		2059
Personas no quemadas del trabajo		1250

Tabla 1.5 Formato del subconjunto de datos de la Q13

Como comentario general de los distintos subconjuntos de datos que hemos seleccionado, podemos ver como los tres rondan el número de 3.200 instancias y en general la distribución de las etiquetas es la misma, habiendo una mayor proporción de instancias de la clase positiva (aproximadamente 2/3) que de la clase negativa.

Esto es debido a que una gran parte de las personas que responden a alguna de las tres anteriores preguntas, responden también a las otras dos preguntas, por lo que la variedad de respuestas no es grande entre los distintos subconjuntos.

En cuanto a los métodos para realizar el trabajo, la experimentación se ha ido realizando en notebooks de Jupyter con Python para mayor eficiencia a la hora de entrenar los diferentes modelos y utilizar los métodos que se incluyen en las librerías de Python a utilizar como Numpy o Pandas, entre otros.

1.10 Tecnologías utilizadas

Entre las distintas tecnologías que se han usado para la realización de este proyecto, se destacan las siguientes:

- **Python 3.10** [25]: el lenguaje de programación escogido ha sido Python, ya que es un lenguaje conocido por su gran variedad de librerías que desarrolla tanto la comunidad como grandes empresas tecnológicas. Aparte de ser código abierto, también destaca debido a que, al ser un lenguaje interpretable, sea más rápido de configurar y de poner en marcha, haciendo que los usuarios que quieran empezar a programar en este lenguaje no tenga que hacer una instalación exhaustiva en comparación a C/C++ (es decir, a lenguajes de programación compilados). En este contexto, este lenguaje cuenta con diversas librerías para el PLN y entrenamiento de modelos de clasificación.

- **Pandas** [26]: esta librería implementada en Python proporciona diversas herramientas para leer diversos tipos de datos en formatos distintos: Ya sea en un

CSV o Excel, esta librería ha sido usada ya que permite leer dicho datos y agruparlos en un objeto de dicha clase el cual cuenta con una gran variedad de métodos y atributos para poder realizar un análisis de dichos datos e incluso manipular y añadir más datos .

- **Numpy** [27]: librería fundamental que cuenta con numerosas herramientas relacionadas con la ciencia computacional en Python. Para ser más exactos, Numpy provee de diversos objetos de vectores multidimensionales con el objetivo de guardar datos en estos vectores y aplicar operaciones matemáticas, lógicas o de ordenación de manera rápida y eficiente.

- **Transformers** [28]: librería clave en la realización de este proyecto. La librería Transformers de Hugging Face cuenta con miles de modelos pre-entrenados para realizar tareas en diferentes modalidades como audio, visión o texto. Consiste en una API (Abreviatura de *Application Programming Interfaces*) que integra 3 librerías populares de deep learning como son **Jax, PyTorch y TensorFlow** y permite descargar y usar los diferentes modelos y datasets que hay en el portal de Hugging Face.

- **NLTK** [29]: abreviatura de ***Natural Language Toolkit***, consiste en una plataforma que provee de herramientas de fácil uso para trabajar con datos en formato de lenguaje natural y que se basan sobre más de 50 corpus y recursos léxicos como WordNet. Dentro de los diversos recursos que hay en NLTK, los que se han usado para el desarrollo de este proyecto son la lista de palabras vacías (también conocido por su correspondientes palabras en inglés *Stop Words*), la división de las observaciones en tokens y la omisión de caracteres como signos de puntuación y símbolos presentes en algunas observaciones.

- **Matplotlib y Seaborn** [30] [31]: estas dos librerías se encargan de la visualización de los diferentes datos que tenemos en diversas gráficas; ya sea una gráfica de barras, un gráfico circular o, en nuestro caso, de un mapa de calor para mostrar una matriz de confusión. También cuenta con diversas interfaces para dibujar llamativas gráficas que cuenten con cierto aspecto estadístico para informar .
- **Scikit-Learn** : librería open-source que proporciona métodos para aprendizaje supervisado y no supervisado. El uso que se le ha dado ha sido para emplear las diferentes herramientas de evaluación de modelos como la precisión, recall y F1-Score, entre otras.
- **Google Collaboratory** [32]: consiste en un producto de Google Research en donde se puede escribir y ejecutar código de Python en el navegador. Es especialmente adecuado para tareas de aprendizaje automático y análisis de datos. Se basa en la estructura que tienen los cuadernos de jupyter para la ejecución de código en celdas y cuenta con el acceso (de manera limitada y restringida) de ejecutar código con recursos hardware adicionales como memoria RAM o de GPUs. Esto último es lo que más nos llama la atención ya que, a la hora de entrenar los distintos modelos, el proceso de entrenamiento se ayudará principalmente de la arquitectura paralela con la que cuentan las últimas GPUs.
- **Optuna** [33]: esquema de trabajo que permite automatizar la búsqueda de hiper-parámetros con el objetivo de encontrar los parámetros que nos den las mejores métricas .

1.11 Metodología de desarrollo de software

Las distintas metodologías de desarrollo de software son sistemas que empezaron a aparecer alrededor de 1960, cuyo principal objetivo era poder desarrollar grandes sistemas de información que cumpliesen con cierto requisitos tales como que fuesen funcionales, una estructura adecuada y reiterativa para cada etapa de su ciclo de vida [34]. Como este proyecto es a gran escala. Es de vital importancia establecer una metodología apropiada.

Después de barajar varias posibilidades y de determinar la naturaleza del problema y de sistema a desarrollar, nos decantamos por una metodología **incremental** debido a que a medida que se iba desarrollando las distintas partes del sistema, teníamos reuniones semanales con el objetivo de ver las funcionalidades implementadas, ver qué estaba bien y qué estaba mal e ir añadiendo nuevas características para obtener un producto final adecuado. Como se puede ver en la ilustración 1.4, a medida que se iba avanzando en el número de incrementos, el sistema iba ganando mayor funcionalidad y cada uno de estos incrementos viene con una fase de comunicación para ir informando sobre el estado del sistema.

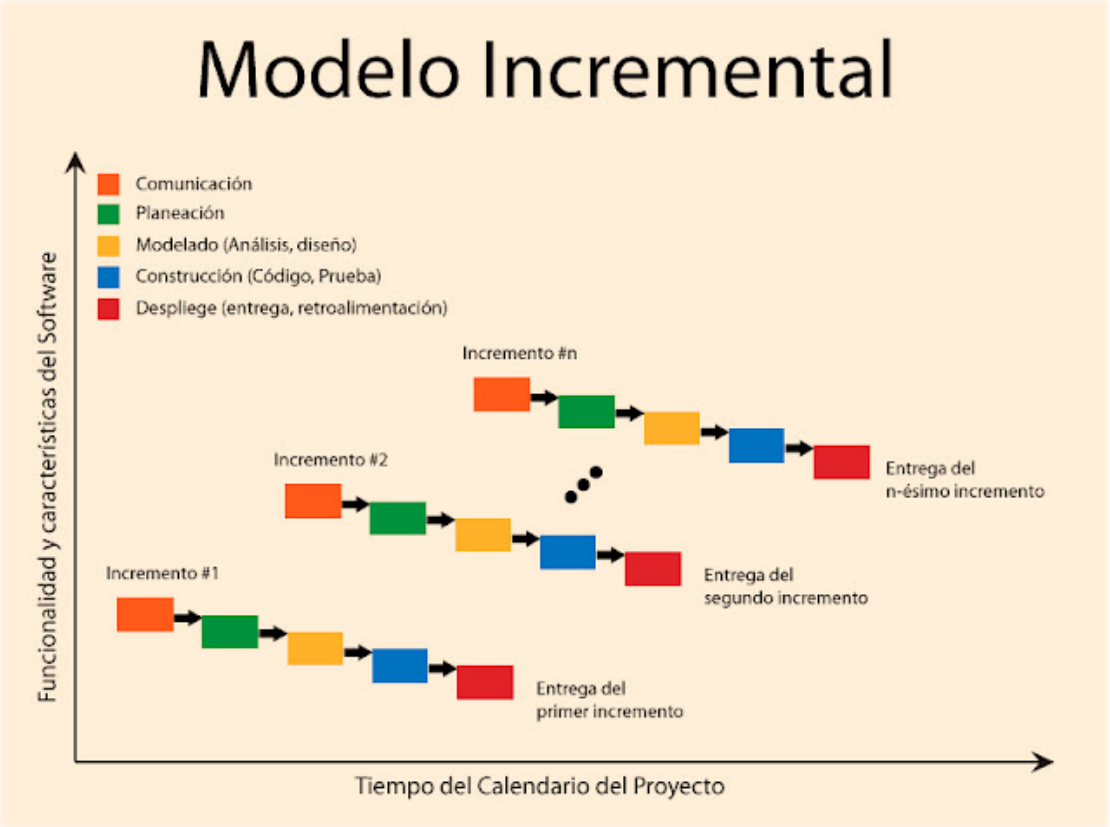


Ilustración 1.4 Esquema del modelo incremental

Entre las diferentes ventajas que ofrece este tipo de metodología de desarrollo, nos encontramos con [35]:

- Se va generando software **operativo** de manera temprana, produciendo resultados visibles más rápido.
- Modelo **más flexible**, dando la posibilidad de cambiar tanto de alcance/propósitos como de requisitos con un coste mucho más reducido en tiempo y dinero.
- Se puede probar y depurar el producto en desarrollo **más fácil**.
- Gestión de riesgos con mayor facilidad.
- Cada iteración corresponde a un hito.

1.12 Estimación del tamaño y esfuerzo

Ya que el presente proyecto es un TFG, no existen restricciones de tipo económico, sino de tipo temporal, con un número aproximado a 300 horas. Por consiguiente, los cálculos de tamaño del proyecto están supeditados al tiempo disponible. En cuanto al esfuerzo, se dispone de tan solo un efectivo por lo que el esfuerzo será de aproximadamente 300 horas por persona. A la hora de estimar el tamaño y esfuerzo requerido para este proyecto, el modelo a utilizar será el modelo COCOMO.

Este modelo fue desarrollado por B. W. Boehm a finales de 1970 y comienzo de 1980. Permite estimar el coste, esfuerzo y planificación que puede conllevar la realización de un proyecto y está dividido en 3 sub-modelos los cuales funcionan mejor de acuerdo con la estructura del proyecto y equipo de trabajo. En nuestro caso, al ser un TFG hecho por una persona, se va a optar por el modelo básico que abarca la mayoría de proyectos pequeños y medianos. Dentro de este sub-modelo hay 3 modos de desarrollo: orgánico, semiencajado y empotrado [36].

- **Modo orgánico**: se caracteriza por haber un grupo pequeño de programadores, el tamaño del software varía de unos **pocos miles** de líneas de código a unas decenas de miles de líneas.
- **Modo empotrado**: el proyecto cuenta con unas fuertes restricciones, posiblemente relacionadas al procesador o, en general, al hardware. Y el problema es único y es difícil basarse en la experiencia **puesto que no puede haberla.**
- **Modo semiencajado**: es un modo el cual actúa como categoría intermedia entre los dos anteriores

Nuestro proyecto se caracteriza tanto por tener un grupo muy pequeño de programadores (una persona en este caso), un tamaño relativamente pequeño rondando aproximadamente 1.000 o 2.000 líneas de código a la vez que cuenta con una gran restricción a la hora de entrenar los modelos de clasificación ya que este entrenamiento requiere de un uso notoriamente alto de la GPU y al estar trabajando en entornos de ejecución de la nube, el hardware está muy restringido. También algo a tener en cuenta es la experiencia en este tipo de proyecto debido a que es un proyecto de gran innovación técnica y requiere de bastante experiencia y formación. En consecuencia, el modo escogido será el semiencajado.

La ecuación que determina la estimación de esfuerzo consiste en:

$$E = a_i S^{b_i} m(X)$$

Donde S indica el número de miles de líneas de código, m(X) es un multiplicador y tanto a como b adoptan un valor en función a una tabla que tiene la siguiente forma:

	Básico		Intermedio	
Modo	a_i	b_i	a_i	b_i
Orgánico	2,4	1,05	3,2	1,05
Semiencajado	3,0	1,12	3,0	1,12
Empotrado	3,6	1,2	2,8	1,2

Tabla 1.6 Tabla de parámetros para el modelo COCOMO con sus distintos modos

Como nuestro proyecto se sitúa dentro del modo semiencajado, tendremos dos ecuaciones para determinar el esfuerzo del personal y el tiempo de desarrollo respectivamente las cuales son:

$$K_m = 3 * S^{1,12} \qquad t_d = 2,5 * K_m^{0,32}$$

Donde K indica el esfuerzo de cada persona al mes y t es el tiempo de desarrollo en meses. Como se mencionó anteriormente que nuestro proyecto tenía una extensión de aproximadamente 2.000 líneas de código, el esfuerzo K sería igual a 6,52 persona-mes y el tiempo t sería igual a 4,55 meses. Por lo que el resultado final **sería 6,52 / 4,55 = 1,43 personas**, que redondeando serían dos personas.

1.13 Planificación temporal

La planificación va a abarcar desde que se inició el proyecto hasta el comienzo de la redacción de la memoria. Para ello nos vamos a apoyar en una tabla en donde se irán especificando los diferentes hitos que se han ido consiguiendo.

Hito	Fecha de comienzo	Descripción
Toma de contacto con el problema	2 de mayo de 2022	En este primer periodo recibí el conjunto de datos con los que iba a tratar y estuvimos investigando el contexto que engloba al problema
Análisis del campo de Observaciones	9 de mayo de 2022	Antes de empezar con la clasificación, se procedió a analizar y obtener un análisis sobre el campo “Observaciones” debido a la importancia que este atributo tenía a la hora de clasificar el estrés.
Experimentación con clasificadores básicos	23 de mayo de 2022	Antes de recurrir directamente al uso de modelos de lenguaje, se realizó una experimentación previa con clasificadores binarios básicos para ver su rendimiento con el corpus.
Investigación de los modelos de lenguaje	20 de junio de 2022	Punto de inflexión en el desarrollo del proyecto. Se inicia la indagación de información sobre los modelos del lenguaje y sobre su funcionamiento.

Implementación y entrenamiento de los modelos	11 de julio de 2022	Puesta en marcha del entrenamiento de los diferentes modelos de lenguaje escogidos
Estudio de hiper-parámetros	12 de septiembre de 2022	Con el objetivo de mejorar los resultados previamente conseguidos. Se comienza a investigar sobre el uso de hiper-parámetros y aplicar una búsqueda de estos para los distintos modelos
Análisis y presentación de resultados	7 de noviembre de 2022	Una vez obtenidos todos los posibles resultados y métricas que se han ido recogiendo, se han llevado a cabo un análisis y se han presentado.
Inicio de la redacción de memoria	27 de febrero de 2023	Por último, se comienza a reunir todo lo realizado en un informe o memoria.

Tabla 1.7 Planificación temporal del Trabajo de Fin de Grado

ETAPAS	MAYO				JUNIO				JULIO			
	S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4
Toma de contacto												
Ánalisis Observaciones												
Experimentación Clasificadores Básicos												
Investigación Modelos Lenguaje												
Implementación Modelos Lenguaje												

ETAPAS	SEPTIEMBRE				OCTUBRE				NOVIEMBRE			
	S1	S2	S3	S4	S1	S2	S3	S4	S1	S2	S3	S4
Estudio Hiper-Parámetros												
Análisis y presentación de resultados												

Ilustración 1.5 y 1.6 Diagrama de Gantt basado en la planificación temporal

1.14 Presupuesto

Para confeccionar el presupuesto estimado, el susodicho se desglosará en 3 apartados distintos: primeramente se detallarán los elementos de software involucrados, acto seguido vendrán los elementos hardware los cuales juegan un papel importante en este proyecto y por último un apartado de elementos auxiliares como personal, electricidad y internet.

1.14.1 Elementos software

En este apartado se incluyen los diferentes productos software que se pueden adquirir para realizar el proyecto. Como se ha descrito anteriormente en el apartado **"Tecnologías utilizadas"**, la mayoría de los componentes que se han utilizado son Open Source, incluso el entorno de desarrollo como es Google Collaboratory tiene un plan gratuito. Lo único que entraría dentro de este punto sería el sistema operativo [37] con el que trabaja el ordenador de trabajo cuyo precio se detalla en la siguiente tabla (vamos a tener en cuenta que los bienes se utilizan un **total de 9 meses**):

Elemento	Vida útil estimada	Precio inicial	Amortización	Coste final
Windows 10 Versión Pro	3 años	259 €	259 € / 36 meses = 7,19€ amortizados al mes	64,80 €

Tabla 1.8 Coste de los elementos software

1.14.2 Elementos hardware

Este proyecto se basa principalmente en el desarrollo y entrenamiento de diferentes modelos de clasificación basados en modelos matemáticos que contienen una gran multitud de operaciones matemáticas con matrices de altas dimensiones. Por esta razón, el proceso de entrenar un modelo y hacerle un fine-tuning consume una gran capacidad computacional.

Elemento	Vida útil estimada	Precio inicial	Amortización	Coste final
Apple MacBook Pro Intel Core i5 1.4GHz/8GB/ 256GB SSD	5 años	1.170,70 €	19,51 € por mes	175,60 €
NVIDIA T4 70W LOW PROFILE PCIe GPU ACCELERATOR	4 años	5.138,05 € [38]	107,05 € por mes	963,38 €
Logitech G604 Lightspeed, Inalámbrico, 25.600 ppp	3 años	110 €	3,05 € por mes	27,5 €
Suma total				**1.166,48 €**

Tabla 1.9 Coste de los elementos hardware

1.14.3 Personal y servicios de las instalaciones

Por último, se tendrá en cuenta el coste que se tendría que invertir en el proyecto a través de contratar diferentes personas que se encargarían de diferentes aspectos del proyecto, a la vez que se calculará los gastos que conlleva el uso de aspectos básicos como la electricidad del edificio, servicio de internet y agua. En este caso, estas personas no estarían contratadas los 9 meses, sino un total de **3 meses**, ya que lo demás consistiría en la formulación de la memoria.

Elemento	Rol en el proyecto	Sueldo/coste mensual	Coste final
Data Scientist	Persona que realiza el pre-procesado de los datos, limpieza y entrenamiento de los modelos.	2.708 €	8.124 €
Jefe de proyecto informático	Individuo encargado de organizar, gestionar y dirigir el proyecto, organizando al personal de la plantilla	3.675 €	11.025 €
Servicio de luz	Servicio necesario para alimentar de electricidad al ordenador	52,29 €	156,87 €
Servicio de agua	Servicio para proveer de agua al edificio y necesario para el personal	33,10 €	99,3 €
Servicio de internet	Indispensable para conectar el ordenador a internet y llevar a cabo la investigación	30,95 €	92,85 €

Suma total	**19.498,02 €**

Tabla 1.10 Coste de los servicios auxiliares

1.14.4 Coste total del proyecto

Se vuelve a mencionar que el presupuesto se ha calculado en función a una duración aproximada de 9 meses. Por lo que el coste final de todo sumado sería el siguiente

Elemento	Coste
Elementos software	64,80 €
Elementos hardware	1.166,48 €
Servicios auxiliares	19.498,02 €
Suma total	**20.729,3 €**

Tabla 1.11 Presupuesto final

2 DISEÑO INICIAL

2.1 Análisis y diseño del sistema

2.1.1 Análisis

Para comprender el proyecto es fundamental tener conocimiento sobre su diseño. Esto implica entender el diseño inicial del sistema, las integraciones de diversas herramientas y tecnologías utilizadas, así como las modificaciones realizadas en los modelos para llevar a cabo diferentes mejoras.

Respondiendo a esto, el sistema a desarrollar tiene el objetivo principal de, dado una observación hecha por algún/a enfermero/a de un hospital expresando su estado o cómo se siente, clasificar si dicha persona se encuentra bajo un estado de estrés o no. De esta manera, se puede detectar el cansancio y agotamiento emocional de los sanitarios y sanitarias para evitar que renuncien a su puesto o mejorar sus condiciones laborales.

Para conseguir esto se necesita cumplir una serie de requisitos iniciales los cuales son:

- **Tiempos de respuesta tolerables**: a la hora de llevar a cabo la predicción/clasificación del estrés en función de la observación, el modelo debe dar una respuesta en un tiempo adecuado, incluso de manera instantánea. Esto no será mucho problema ya que el modelo al estar ya entrenado, el proceso de predecir será rápido.
- **Facilidad para usarlo**: uno de los pilares fundamentales a la hora de que un sistema informático tenga éxito reside en la sencillez del sistema en el momento de usarlo puesto que hay una gran probabilidad de que estos clientes objetivos no tengan un conocimiento informático elevado.

2.1.2 Diseño

En cuanto al diseño del sistema, hay que explicar la arquitectura de este. Para ello se explicará en qué consiste la estructura básica de un sistema de clasificación y, como el desarrollo se basó en el empleo de modelos de lenguaje. Concretamente, se explicará en mayor detalle la arquitectura Transformers.

2.1.2.1 Arquitectura base de un sistema de clasificación automática

Como pequeña definición, un sistema de clasificación automática consiste en un software en donde se le introduce un conjunto de datos con el propósito de entrenarlo en base a modelos matemáticos para que así, en el momento de introducir nuevos datos o ejemplos, sea posible predecir la variable objetivo o clase.

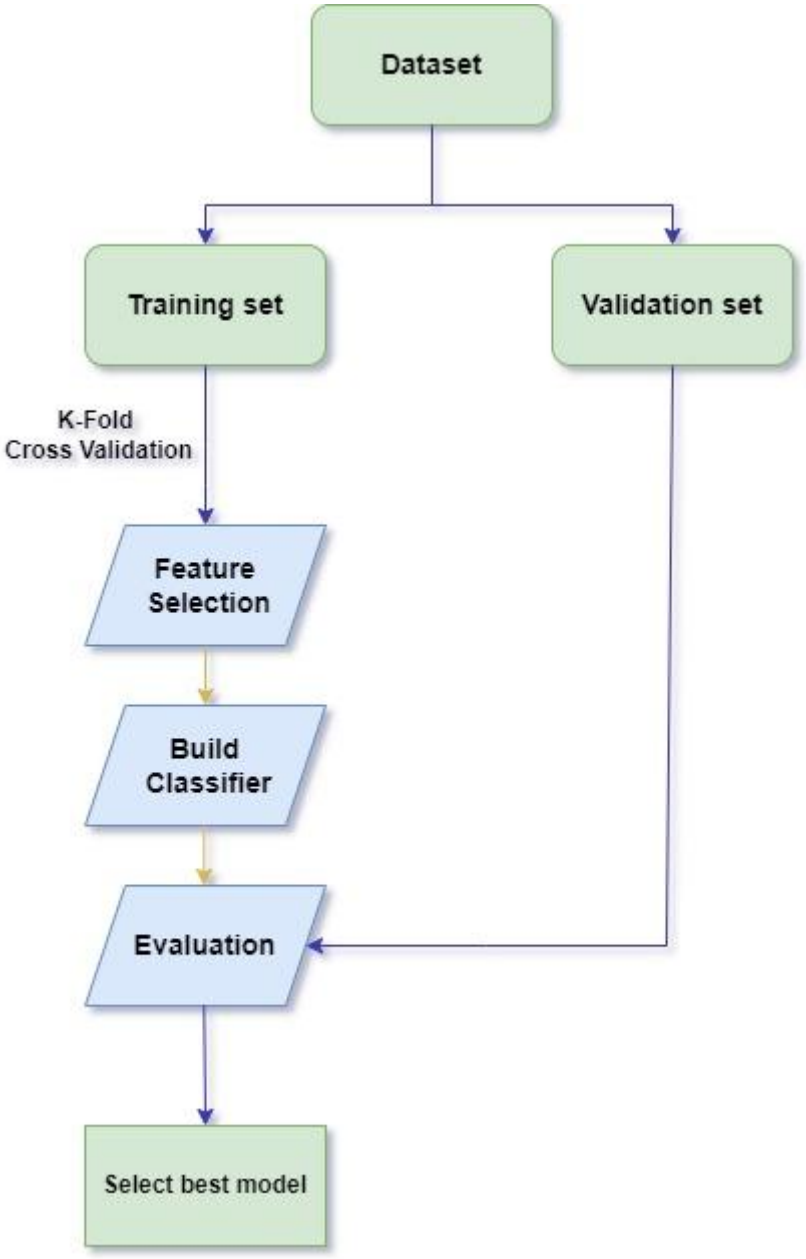


Ilustración 2.1 Diagrama de flujo sobre la estructura básica del desarrollo de un sistema de Machine Learning

Como se puede ver en la Ilustración 2.1, lo primero que se necesita para desarrollar un software de este calibre es un conjunto de datos. En función de cómo se encuentren estos datos pueden ser:

- **Datos estructurados**: son aquellos datos que se encuentran en tablas relacionales de bases de datos o en hojas de cálculo en donde hay una serie de columnas o **atributos** los cuales adoptan un valor específico.
- **Datos no estructurados**: estos datos contienen información importante para el problema, sin embargo, no están organizados o se encuentran mezclados en ficheros de texto plano, en formato PDF o incluso en imágenes.

Al mismo tiempo, este conjunto de datos contiene una serie de atributos los cuales serán esenciales para que el sistema decida, en función del valor de dichos atributos, una variable objetivo/etiqueta/clase**.** Se pueden distinguir entre dos tipos de datasets: Aquellos en donde todas las instancias (o ejemplos del conjunto de datos) tiene asignado una clase, conocido como **aprendizaje supervisado** y aquellos en donde no se conoce el número de clases o las instancias no vienen etiquetadas, **aprendizaje no supervisado**.

Para trasladarlo a nuestra situación, en un inicio, los datos se recogieron de las encuestas realizadas por el SATSE a los enfermeros y enfermeras, por lo que no estaban estructurados pero luego se procesó los datos recogidos y se reunió en una hoja de cálculo, por lo que los datos pasaban a estar estructurados y como hay una columna específicamente para la clase, entonces tratamos con un problema de aprendizaje supervisado. Sin embargo, solamente tendríamos un atributo que sería las observaciones y la clase que sería el estar estresado o no. Este conjunto de datos se divide en 3 subconjuntos: uno de entrenamiento el cual será el usado por el modelo para aprender, otro de validación con el objetivo de examinar cuán bien o mal clasifica el modelo, y por último estará el subconjunto de test que se usa para determinar el rendimiento del modelo y obtener resultados más verídicos.

Un apartado a tener en cuenta que no aparece en la ilustración anterior es la optimización de hiper-parámetros. Como se ha comentado anteriormente, a la hora de entrenar estos sistemas, se emplean diversos modelos que contienen numerosos parámetros: Tasa de aprendizaje, número de épocas, pesos de clases, tamaño del lote… No hay un valor óptimo de estos parámetros donde funcione en todos los problemas. Por lo tanto, una manera de solventar este problema es realizar una

búsqueda de parámetros en donde se van probando diversos valores para ciertos parámetros y quedarnos con aquellos valores que hayan dado la mejor métrica de evaluación.

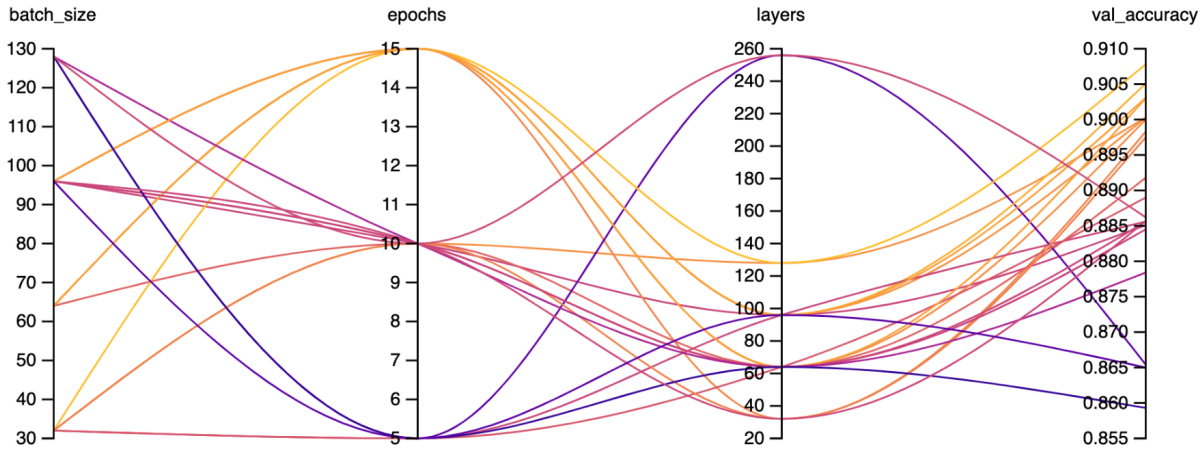


Ilustración 2.2 Gráfica sobre la búsqueda de parámetros

En la ilustración 2.2 podemos ver una una representación gráfica de cómo es esa búsqueda de hiper-parámetros en donde en el ejemplo de la imagen, se prueban con un intervalo de valores tanto para el tamaño del lote, el número de épocas y las capas de la red neuronal y, al final, se van recogiendo diversos resultados.

2.1.2.2 Arquitectura Transformers

Como se ha citado anteriormente, la arquitectura Transformer fue una propuesta realizada por un equipo de investigadores del Google Research en colaboración con investigadores de la Universidad de Toronto. Esta arquitectura revolucionó el campo del procesamiento del lenguaje natural ya que se obtenían mejores resultados en tareas de traducción automática o de búsqueda de respuestas. Lo novedoso que trajo esta arquitectura es que da la posibilidad al modelo de ser consciente de su contexto, lo cual hace que obtenga un mejor rendimiento en tareas en donde el contexto de la frase es importante.

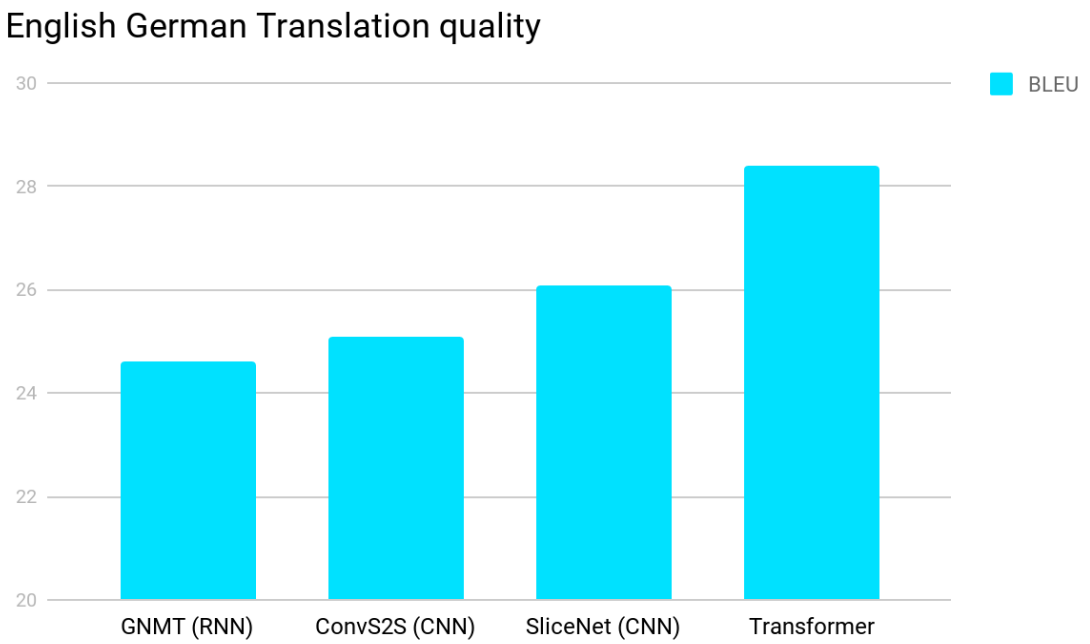


Ilustración 2.3 Resultados de la traducción del inglés al alemán en diferentes modelos neuronales junto al modelo Transformer

Por ejemplo, para abordar un problema de desambiguación de una palabra en una frase “He llegado al **banco** después de cruzar el río”, mientras que las redes neuronales convolucionales y las recurrentes (conocidas por sus siglas CNN y RNN respectivamente) van comprobando paso por paso y comparando por pares de palabras para determinar el contexto de la oración hasta cuando se lee la palabra ‘río’, con los Transformers, se realiza un número constante de pasos en donde se forman modelos relacionales entre todas las palabras de una frase independientemente de sus posiciones en la oración. Para ser más específicos, para cada palabra se compara con las demás palabras de la oración y se asigna una puntuación de atención que determina cuánto contribuye a la representación de la palabra en el contexto característico [40].

La arquitectura Transformer se asentó sobre las redes neuronales las cuales son un conjunto de modelos computacionales formados por una serie de capas compuestas de diferentes nodos o neuronas, intentando simular la estructura del cerebro humano.

En estas redes neuronales nos encontramos con 3 tipos distintos de capas de nodos o *layers* [41]:

- **Capa de entrada (*Input layer*):** se caracteriza por su función de aceptar y representar los datos que se utilizarán en la red.
- **Capa de salida (*Output layer*):** corresponde a la última capa de la red en donde se muestra el resultado del problema
- **Capas ocultas (*Hidden layers*):** siendo una o varias capas, estás capas son las que van procesando los datos de entrada llevando a cabo diversas funciones matemáticas: transformaciones, creación automática de atributos, regresión lineal, etc.

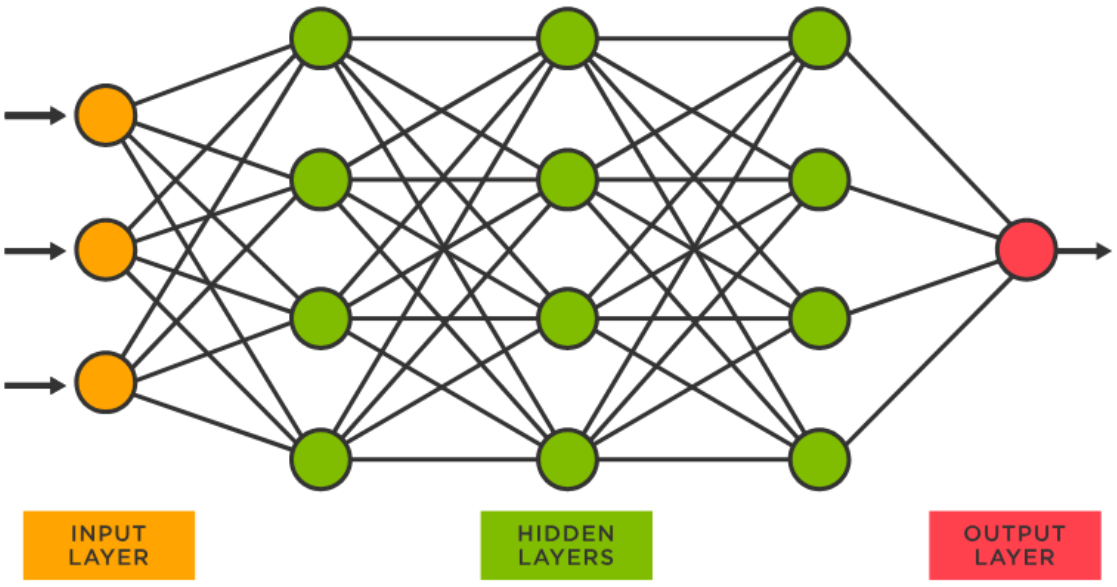


Ilustración 2.4 Configuración básica de una red neuronal

Dentro de estas neuronas hay una función matemática (por ejemplo, un modelo de regresión lineal) en donde en función de los diversos parámetros de entrada que reciba la función, se obtendrá un resultado y este servirá como entrada

de otro nodo de alguna capa oculta, sea resultado de la capa de salida o incluso sirva como entrada de ese mismo nodo como pasa con las RNN [42].

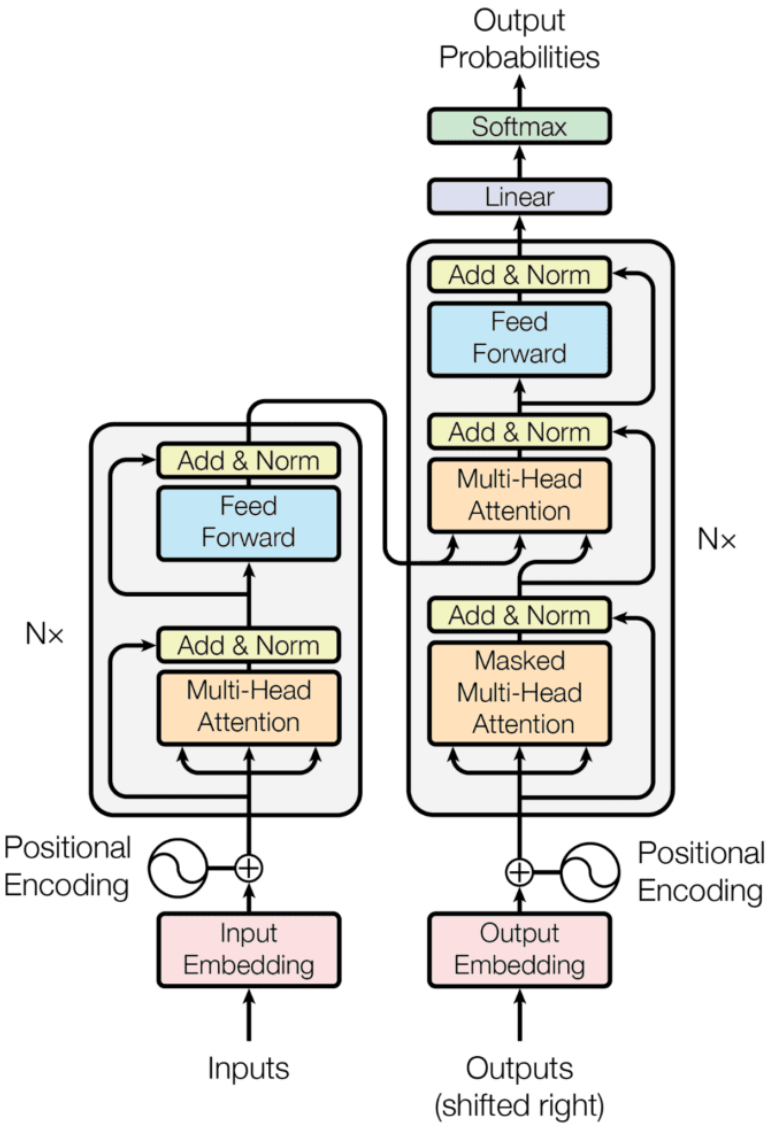


***Ilustración 1.2 Esquema sobre la arquitectura Transformers* [43]**

Los Transformers tienen una arquitectura como se muestra en la Ilustración 1.2 en donde se pueden distinguir dos elementos fundamentales: un codificador y un decodificador.

- **Codificador**: este componente consta de una pila de N capas (siendo N = 6), cada una de estas capas se compone de dos subcapas:
    - Un mecanismo de **auto-atención multicapa**.
    - Una **red prealimentada** que consiste en una red de nodos los cuales sus conexiones no forman un ciclo.
- **Decodificador**: también consiste en una pila de 6 capas. Sin embargo, cuenta con 3 subcapas en donde, además de las dos subcapas previamente mencionadas, hay una tercera capa de atención que se aplica sobre la salida de la pila del codificador.

Aparte de estos elementos, también tiene herramientas como un tokenizador el cual tiene en cuenta las posiciones de las palabras del texto ya que este aspecto es muy importante en tareas como traducción automática o búsqueda de respuestas.

3 DESARROLLO

3.1 Primera Iteración: investigación y toma de contacto con el problema

El primer paso de este proyecto era investigar sobre el dominio del problema. Se realizó una lectura del documento del Estudio SATSE “Percepción de estrés en los profesionales de Enfermería en España. Comparativa 2012-2017-2021” en el cual se encuentra una introducción explicando la situación y problemas a los que se enfrenta el sistema sanitario, la metodología para recoger los datos y opiniones de los distintos profesionales y unas gráficas relacionadas con los resultados obtenidos de las encuestas realizadas. Después de esto se realizó también una investigación para aprender la arquitectura base de los modelos de lenguaje y de algunos modelos de clasificación que se usaron como los SVM (siglas provenientes del

inglés *Support Vector Machine*: Máquina de soporte vectorial) y la regresión logística.

Para aprender más sobre los modelos de lenguaje y su implementación, se recurrió a un curso de Hugging Face [44] el cual tiene como objetivo el enseñar nociones básicas sobre el Procesamiento de Lenguaje Natural usando librerías del ecosistema de Hugging Face. Del contenido de este curso se destaca la explicación de los modelos Transformers y su guía de como hacer *fine-tuning* a los modelos pre-entrenados disponibles en este portal. Por último, también se tuvo que adquirir nociones de uso del entorno de desarrollo que se ha utilizado, es decir, de Google Collaboratory [45] por las ventajas que este ofrecía a la hora de entrenar los diversos modelos de clasificación en la nube.

3.2 Segunda Iteración: análisis y estudio del conjunto de datos

Una vez realizada la búsqueda de información relacionada con el problema y las herramientas y tecnologías a utilizar, se continuó con un estudio sobre el conjunto de datos al que nos enfrentábamos. Ya se comentó sobre el formato del conjunto de datos tanto en el Apartado 1.4 como en el Apartado 1.9. Teniendo en cuenta las observaciones como única variable de entrada para entrenar el modelo, tenemos la variable objetivo que solo puede adoptar dos únicos valores: “estresado” y “no estresado”. A la hora de realizar una experimentación con las otras dos preguntas objetivo, para ser exactos, con las preguntas relacionadas con el agotamiento emocional y el estar quemado del trabajo (las preguntas Q11 y la Q13, respectivamente), la clase cambiaría pero en los tres casos trataremos con un problema de clasificación binaria.

3.2.1 Balanceo de los datos

Para cada pregunta objetivo se tiene un balanceo de los datos distinto.

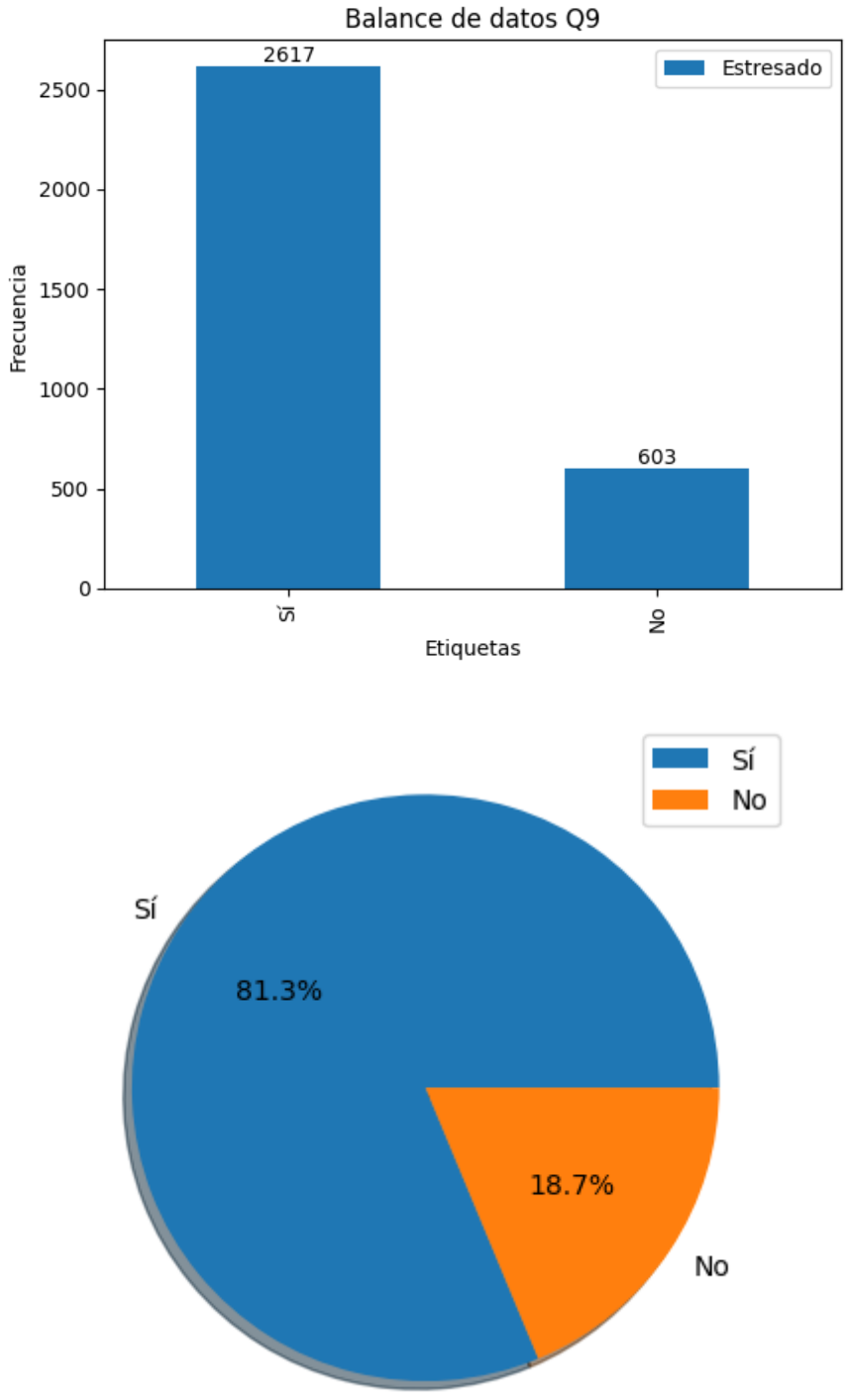
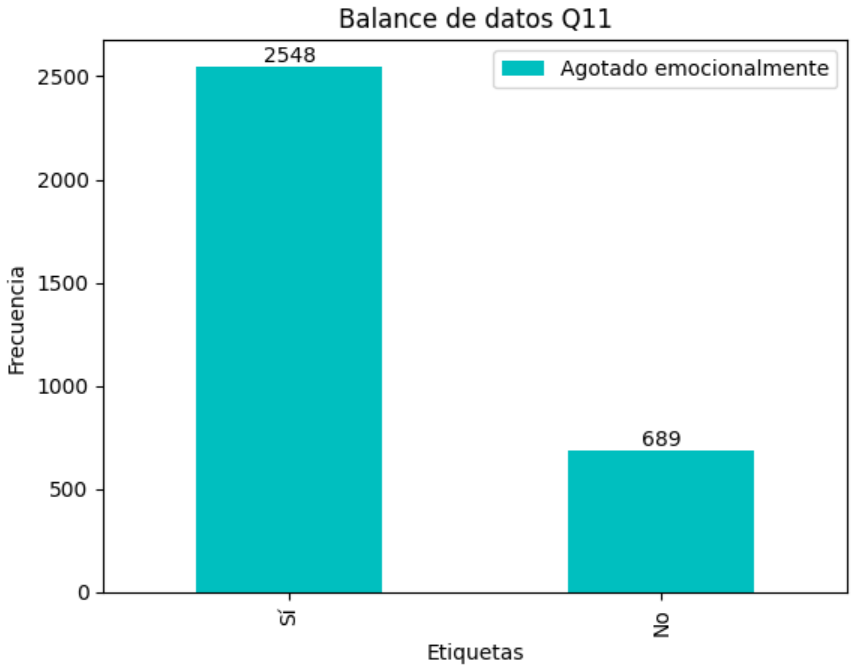


Ilustración 3.1 Balanceo de los datos en Q9



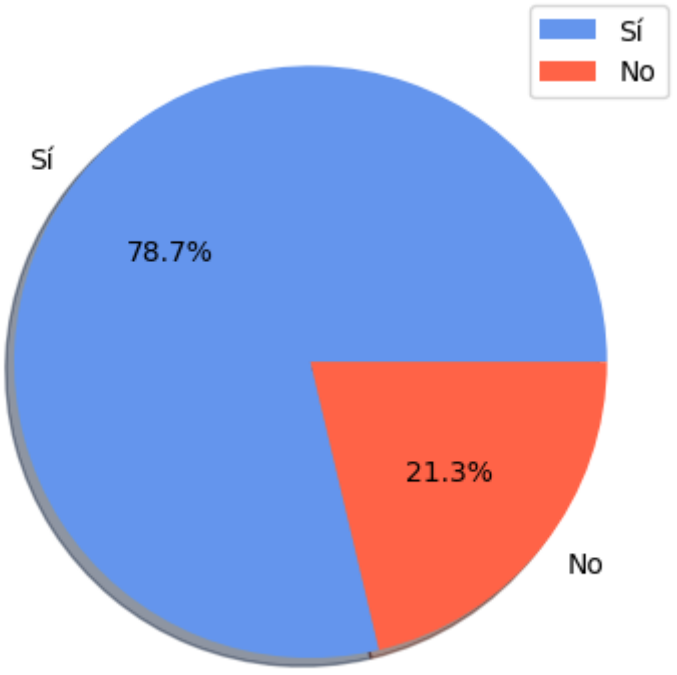


Ilustración 3.2 Balanceo de los datos en Q11

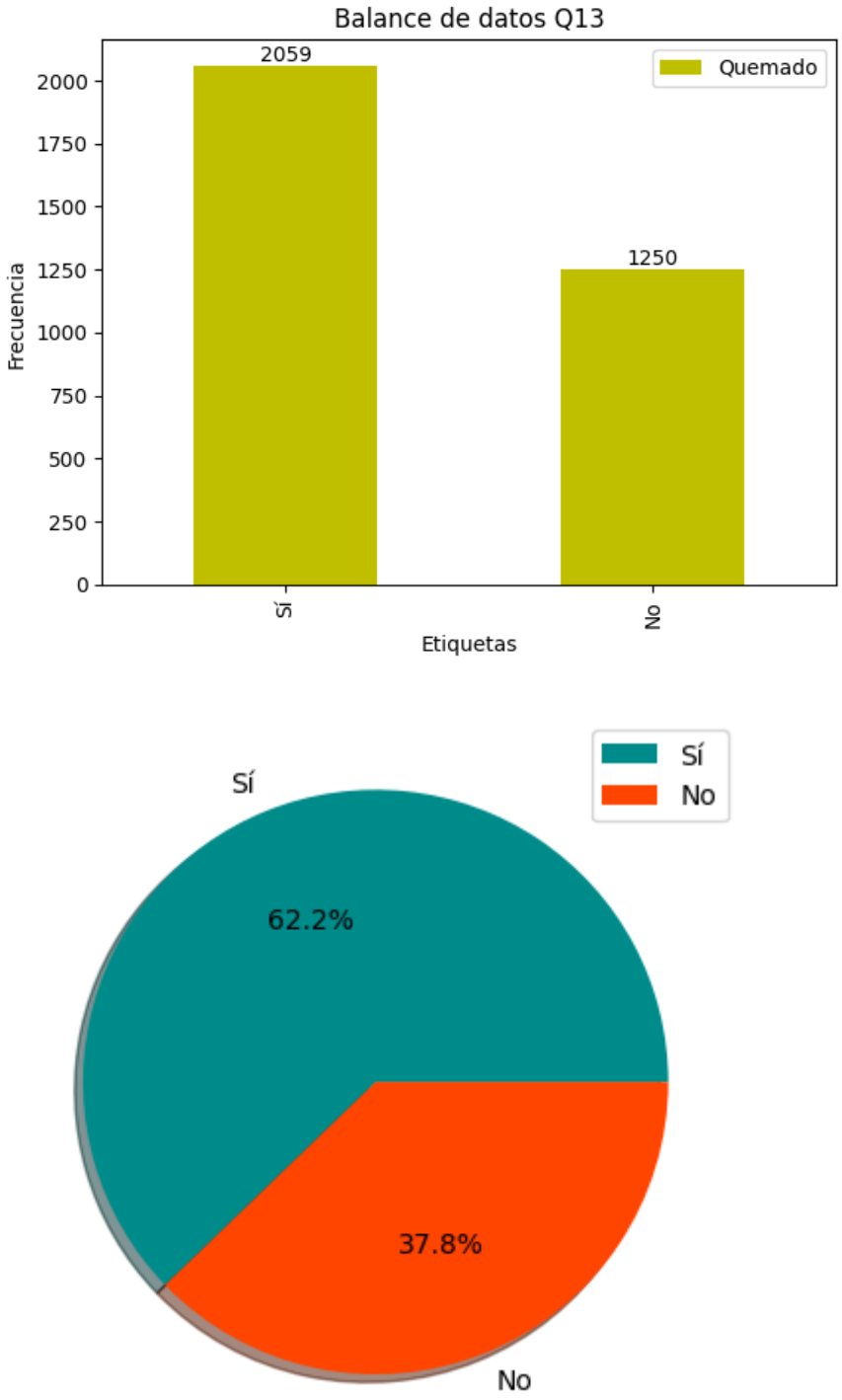


Ilustración 3.3 Balanceo de los datos en Q13

Como se puede observar en las ilustraciones 3.1, 3.2 y 3.3, hay un mayor número de instancias con la clase positiva asociada (ya sea de estrés, agotamiento

o del síndrome del trabajador quemado) que la clase negativa. Visto este balance, una de las modificaciones que se plantearon a la hora de entrenar el modelo es balancear el dataset igualando el número de instancias etiquetadas positivamente como negativamente.

3.2.2 Análisis de las observaciones

Se hizo un estudio en relación a las observaciones relacionadas con la pregunta de si la persona estaba estresada o no y se sacaron los siguientes datos:

Estadísticas descriptivas	Cifra
Número de valores	3.220
Número de valores únicos	2.546
Observación más frecuente	“No”
Frecuencia de la observación más frecuente	245

Tabla 3.1 Estadística descriptiva de las observaciones

Según los datos recogidos en la tabla, hay un total de 3.220 observaciones pero muchas de estas se repiten, teniendo como observación más frecuente el simple adverbio negativo “No”, con una frecuencia de 245 observaciones. Viendo que muchos de estos registros eran una simple palabra como “No”, “.”, “Nada”, “Ninguna”, se decidió eliminar este tipo de registros porque no aportaban

información. Así, se introdujo una umbral de 7 caracteres para determinar cuando una observación era válida o no.

Estadísticas descriptivas	Cifra
Número de valores	2.513
Número de valores únicos	2.479
Observación más frecuente	"Falta personal"
Frecuencia de la observación más frecuente	7

Tabla 3.2 Estadística descriptiva de las observaciones con el umbral aplicado

También se llevó a cabo un estudio sobre la categoría gramatical de las palabras que componen las observaciones. En la tabla 3.3 se puede ver la cuantas palabras de las diferentes categorías hay tanto en aquellas observaciones de las personas que han respondido "Sí" a la pregunta de si estaban estresadas como en las personas que han respondido "No" a dicha pregunta

Categoría gramatical	Número de palabras pertenecientes a las personas que no están estresadas	Número de palabras pertenecientes a las personas que están estresadas
Adjetivos	1.257	5.219
Adverbios	1.076	4.718
Nombres	3.515	14.345
Verbos	1.735	7.419

Tabla 3.3 Categoría gramatical de las palabras de las observaciones

Para este problema, cada observación puede denotar un sentimiento o emoción: ya sea miedo, sorpresa, enfado, entre otros. Por ello se intentó detectar las emociones que denotaba las distintas observaciones en función a un lexicón realizado por el concejo nacional de investigación de Canadá, conocido por sus siglas en inglés NRC (*National Research Council*) [46].

Emoción	Número de palabras pertenecientes a las personas que no están estresadas	Número de palabras pertenecientes a las personas que están estresadas
Miedo	1.023	4.357
Negativismo	1.919	8.254
Tristeza	816	3.315
Enfado	551	2.380

Tabla 3.3 Frecuencia de las emociones a nivel de palabra

Por último, se realizó una nube de palabras de los distintos adjetivos, verbos, adverbios y nombres de las observaciones de la clase “estresada”

3.2.3 Nubes de palabras de la clase “estresada”

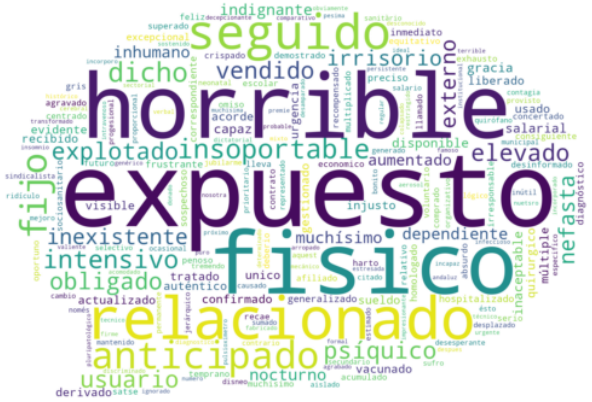


Ilustración 3.4 Nube de palabras de adjetivos exclusivos de la clase “estresada”

- Podemos ver como los adjetivos que más se repiten son “horrible”, “expuesto”, “físico” como “explotado” e “insoportable". Todos estos adjetivos denotan cierta desesperación y agobio que sufría el personal sanitario durante la pandemia

Ilustración 3.5 Nube de palabras de verbos exclusivos de la clase "estresada"

- Con relación a los verbos, los que más se repiten son aquellos como "permitir", "agravar", "resultar, incluso el verbo "quemar", el cual hace referencia a la variable objetivo que se tiene en la pregunta Q13

Ilustración 3.6 Nube de palabras de adverbios exclusivos de la clase "estresada"

- Mucha gente mencionó su condición mental y eso se refleja en como un adverbio frecuente en la clase "estresada" fue "psicológicamente", haciendo referencia a cómo la situación actual los estaba afectando a su salud mental.

Ilustración 3.7 Nube de palabras de nombres exclusivos de la clase "estresada"

- Por último, se ven como los nombres que más se repiten es "estabilidad", "gerencia", "jubilación", entre otros. Podemos ver como muchas de estas observaciones iban dirigidas a llamar la atención a la gerencia del sistema sanitario o como el personal anhelaba una vuelta de la estabilidad o incluso de la jubilación, demostrando un cansancio y agotamiento emocional en su trabajo.

3.3 Tercera iteración: Entrenamiento de los modelos

Una vez hecho el análisis del campo de observaciones, se procedió al entrenamiento de los distintos modelos de clasificación. En una primera aproximación se realizó una clasificación sencilla en la cual se tenía en cuenta la polaridad de cada observación y, en función de si tenía más palabras positivas que negativas, se clasificaba dicha instancia como "estresada" o "no estresada".

Después de este primer experimento, la segunda estrategia consistió en utilizar modelos de clasificación que vienen implementados en Scikit Learn. Para conseguir esto, se procesó el texto mediante la clase *TF-IDFVectorizer*, una clase de la misma librería de Scikit Learn la cual convierte una colección de documentos de texto sin procesar en una matriz de valores TF-IDF [47]. En este mismo experimento se probó a entrenar el modelo mediante un balanceo manual del corpus y con un parámetro del propio modelo llamado *class_weight* el cual se asigna un peso a cada clase en función del número de instancias relacionada a dicha clase entre el número total de instancias.

Seguidamente empezaría la experimentación con los modelos de lenguaje en donde se realiza un pre-procesado del texto diferente en el cual se explicará en el apartado de “Experimentación, resultados y discusión”. Y por último, con el objetivo de mejorar los resultados que se estaban obteniendo, se pasará a la optimización de hiper-parámetros de los modelos. Para ello, se fijó aquellos parámetros a optimizar y que influyeron significativamente en una mejora de los resultados. Estos son:

- ***Learning rate***: o tasa de aprendizaje, es el parámetro que controla la actualización de los pesos del modelo en cada iteración. Establecer una buena tasa de aprendizaje es difícil y varía en función de la naturaleza del problema. Mientras que una tasa de aprendizaje alta puede hacer que a la hora de actualizar los pasos, dicho cambio sea muy drástico, omitiendo las ponderaciones que permite la solución óptima, una tasa muy pequeña hace que los cambios de estos pesos sean tan pequeños que conseguir la configuración de pesos óptima podría llevar mucho tiempo. Por ello, situamos el espacio de búsqueda en el intervalo [1e-5, 0.01].
- ***Weight decay***: o decaimiento de los pesos, sirve para evitar el sobreajuste de una red neuronal a través de la aplicación de una penalización a la función de coste. El intervalo se define entre 0.01 y 0.1.
- **Número de épocas**: Se dice que el algoritmo ha completado una época cuando el dataset ha pasado por toda la red neuronal tanto

hacia delante como hacia atrás [48]. Los valores a probar son 5, 7 y 10 épocas.

3.4 Pruebas finales

A la hora de probar el sistema y ver su rendimiento, el dataset se divide en dos subconjuntos: El subconjunto de entrenamiento el cual se usa para entrenar al modelo y el subconjunto de test para, una vez se tiene el modelo entrenado y ajustado, se realiza una predicción sobre las observaciones del subconjunto de testeo y se compara con las etiquetas reales, obteniendo las diversas métricas de evaluación como la precisión, recall, F1-score, entre otros.

4 Experimentación, resultados y discusión

4.1 Experimentaciones y pruebas

Para cada experimento realizado con los distintos métodos y herramientas, se utilizó por separado tanto el conjunto de datos con la pregunta Q9, Q11 y Q13. También se realizó una unión de las 3 preguntas en donde, se etiquetaba como "Positivo" a aquella persona que sufría de estrés, agotamiento y *burnout* (término en inglés que hace referencia al síndrome del trabajador quemado). Esta combinación de datos se denominará **Q9-Q11-Q13**.

Se probó cada uno de estos conjuntos de datos tanto con el lexicón iSOL, con modelos del lenguaje sin ajustar hiper-parámetros y con modelos del lenguaje habiendo llevado a cabo una optimización de hiper-parámetros. Además, para probar más configuraciones, se probó a entrenar los modelos tanto con los datos desbalanceados como balanceados (es decir, teniendo el mismo número de instancias positivas como negativas en el conjunto de entrenamiento).

Por último, cabe destacar que para la fase de experimentación, se han recogido diversas métricas de evaluación las cuales son [49]:

- Precisión: mide la capacidad de que un clasificador etiquete una instancia positiva como positiva (lo que se conoce como *True Positive*, o su abreviatura TP) o a la inversa (*True Negative,* o su abreviatura TN). $Precision_{Positive} = TP/(TP + FP)$.
- Recall: se encarga de estimar la capacidad del modelo de encontrar todos los ejemplos positivos (o negativos) de una clase. Consistiría en la fracción de predicciones de una clase las cuales son identificadas como correctas. $Recall_{Positive} = TP/(TP + FN)$.
- F1-Score: consiste en la media armónica de la precisión y el recall. Este valor es muy útil para determinar el rendimiento de un modelo de clasificación en conjuntos de datos muy desbalanceados en donde no solo interesa clasificar correctamente la clase positiva, sino también predecir correctamente la clase negativa. $F1 = 2 * Recall * Precision/(Recall + Precision)$
- Para cada una de estas métricas se puede realizar diferentes métodos para realizar las medias sobre estas como por ejemplo:
  - *Macro average*: es calculada usando la media aritmética de todas los valores que hay dentro de una métricas.
  - *Micro average*: calcula una media global a través del conteo de la suma de los verdaderos positivos, falsos negativos y falsos positivos.
- Nosotros vamos a centrarnos en la **Macro F1-Score** la cual es una media aritmética de todos las F1-Score (en este caso, solo hay 2 valores, una para cada clase) ya que como hemos podido ver en el apartado de "Segunda Iteración: análisis y estudio del conjunto de datos" (para ser más específicos, en la ilustración 3.1, 3.2 y 3.3), el dataset está desbalanceado y si nos guiamos solamente por la precisión, veríamos como la precisión es altísima y por tanto pensaríamos que el modelo funciona adecuadamente. Sin embargo, esto solo es una falsa sensación de que el modelo funciona bien porque dicho modelo estaría etiquetando siempre a la clase mayoritaria. Por esta razón,

como la F1-Score tiene en cuenta tanto la precisión como el recall, pues es la métrica idónea.

4.1.1 Regresión logística de Scikit Learn

Con la intención de establecer un punto de partida en el proyecto, realizamos la clasificación binaria con modelos básicos de clasificación binaria como la regresión logística. Para ello, simplemente implementamos una *pipeline* el cual conste de un *TfidfVectorizer* el cual se encarga de obtener la importancia de un término en función a todas las palabras que hay en las diversas observaciones y, una vez obtenido el TF-IDF, se usa un modelo de regresión logística para que el modelo aprenda del corpus empleado.

Pregunta	Macro-F1	Micro-F1	Macro-precision	Macro-recall
Q9	0,44	0,81	0,40	0,50
Q11	0,44	0,78	0,39	0,50
Q13	0,44	0,44	0,53	0,51

Tabla 4.1 Resultados de clasificación con Regresión Logística

4.1.2 Uso del lexicón iSOL

Como se comentó en la sección “Estudio de alternativas y viabilidad”, un recurso que utilizamos fue el lexicón iSOL que consiste en un conjunto de palabras o términos que se utilizan en un idioma particular. Es una representación estructurada

de palabras donde se almacena información lingüística relevante para su procesamiento. En este caso, esta información relevante sería la opinión.

La experimentación con este lexicón consiste en revisar cada palabra de cada observación y determinar si dicha palabra estaba catalogada como positiva o negativa según el lexicón. Se lleva la suma total de palabras negativas y positivas de esa observación y si al final tiene más palabras negativas, esa observación se clasificaba como “estresada”. En caso contrario, se clasificaba como “no estresada”

Balance del dataset	Pregunta	Macro-F1	Micro-F1	Macro-precision	Macro-recall
Datos no balanceados	**Q9**	0,49	0,62	0,50	0,50
	Q11	0,51	0,62	0,51	0,52
	Q13	0,52	0,57	0,52	0,52
	Q9-Q11-Q13	0,51	0,55	0,52	0,52
Datos Balanceados	**Q9**	0,49	0,51	0,50	0,50
	Q11	0,50	0,52	0,52	0,52
	Q13	0,50	0,53	0,52	0,52
	Q9-Q11-Q13	0,50	0,52	0,52	0,52

Tabla 4.2 Resultados de clasificación con lexicón iSOL

4.1.3 Modelos del lenguaje Transformer sin ajuste de hiper-parámetros

Antes de mostrar los resultados, se explicará cómo se ha procesado la información a través de las distintas herramientas que proporciona la librería de Hugging Face: **Transformers**.

1. **Tokenización:** La primera parte consiste en tokenizar el texto. Este proceso se puede realizar de diversas maneras, ya sea dividiendo las palabras de una frase solamente en espacios o también teniendo en cuenta los símbolos de puntuación como un apóstrofe, coma, punto, etc. Una tercera forma de tokenizar una oración es mediante los **tokenizadores basados en sub-palabras**, en donde se divide la frase en palabras sencillas y, aquellas palabras raras o complejas/compuestas, dividirlas en sub-palabras. Por ejemplo, la palabra “raramente” se dividiría en “rara” y “mente” [50].
2. **Entrenamiento:** habiendo procesado y tokenizado el texto, se especifican los parámetros de la clase *Trainer* de la librería Transformers. Entre los distintos parámetros de entrada está el tipo de modelo que se va a usar, el número de épocas, la tasa de aprendizaje, el tamaño del lote, entre otros.
3. **Evaluación:** una vez entrenado el modelo y habiendo hecho un fine-tuning con nuestro conjunto de datos, viene la evaluación de este modelo de clasificación a través de la medición de distintas métricas de evaluación como el F1-Score, recall o la precisión.

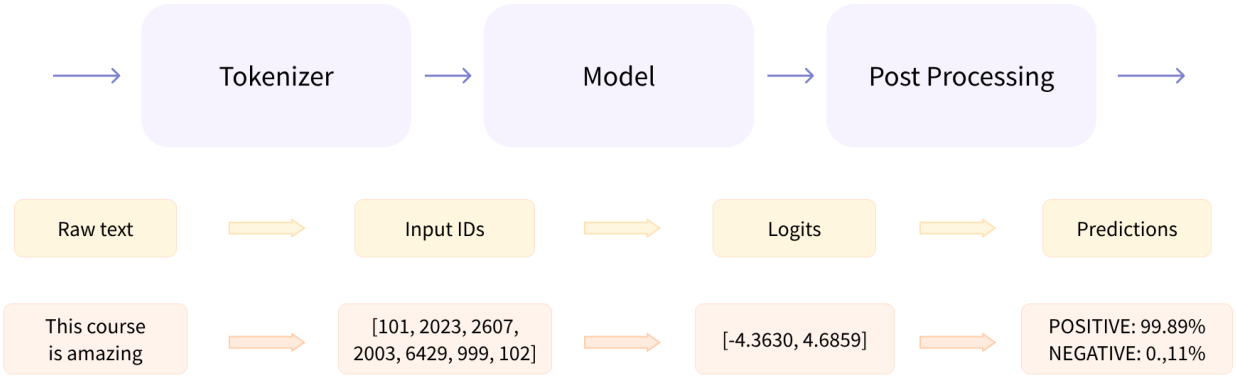


Ilustración 4.1 Flujo de trabajo de los modelos Transformers de Hugging Face

Los resultados mostrados a continuación son de aquellos modelos que devolvieron las mejores métricas. Para ser exactos, nos fijamos en la métrica **Macro F1-Score** debido a que estamos trabajando con un conjunto de datos desbalanceado y fijarnos solamente en la precisión sería un error debido a que al haber una mayor proporción de instancias de una clase, si el modelo etiqueta todas los ejemplos con la clase mayoritaria, obtendría una buena precisión pero tanto la recall de la clase minoritaria como el F1-Score serían muy bajos.

También se probaron distintos modelos pre-entrenados que se encontraban en Hugging Face y al final nos quedamos con dos modelos que daban los mejores resultados de entre todos los modelos que se probaron. Los dos modelos en cuestión fueron:

- **BETO: Spanish BERT**: este modelo BERT fue entrenado con un corpus grande en español en el que se incluye varios artículos de Wikipedia y otras wikis, charlas de TED, comentarios de noticias, etc.
- **RoBERTa base trained with data from the National Library of Spain (BNE):** Este modelo está basado en el modelo RoBERTa y fue pre-entrenado con un total de 570 GB de texto procesado que se extrajo de la Biblioteca Nacional de España, de material entre 2009 y 2019.

Balance del dataset	Modelo	Pregunta	Macro-F1	Micro-F1	Macro-precision	Macro-recall
Datos no balanceados	**BETO**	**Q9**	0,50	0,78	0,52	0,51
		Q11	0,56	0,73	0,57	0,56
		Q13	0,56	0,61	0,57	0,56
		Q9-Q11-Q13	0,55	0,58	0,56	0,55
	RoBERTa	**Q9**	0,53	0,79	0,56	0,53
		Q11	0,49	0,76	0,50	0,50
		Q13	0,58	0,66	0,61	0,58
		Q9-Q11-Q13	0,56	0,58	0,56	0,56
Datos balanceados	**BETO**	**Q9**	0,53	0,53	0,53	0,53
		Q11	0,51	0,52	0,52	0,52

		Q13	0,55	0,55	0,55	0,55
		Q9-Q11-Q13	0,58	0,58	0,55	0,58
	RoBERTa	**Q9**	0,55	0,56	0,56	0,56
		Q11	0,54	0,55	0,55	0,54
		Q13	0,54	0,55	0,55	0,55
		Q9-Q11-Q13	0,62	0,62	0,62	0,62

Tabla 4.3 Resultados de clasificación con Transformers sin hiper-parámetros

4.1.4 Modelos del lenguaje con optimización de hiper-parámetros

Para este apartado se ha realizado una búsqueda de parámetros con el objetivo de encontrar aquellos parámetros cuyos valores permitan obtener el resultado óptimo. Los parámetros a optimizar y su función se detallaron en el apartado sobre la "Tercera iteración: Entrenamiento de los modelos". El proceso realizado para la búsqueda de estos hiper-parámetros consistió primero en definir unos intervalos de valores que los parámetros iban a adoptar. Luego se establece un estudio con la librería Optuna el cual se encarga de ir realizando los diferentes entrenamientos del modelo y mostrando la macro F1-Score y, por último, después de que el estudio se haya ejecutado apróximadamente 2 horas, se revisan los

resultados y nos quedamos con los hiper-parámetros que mejor macro F1-Score hayan dado.

Balance del dataset	Modelo	Pregunta	Macro-F1	Micro-F1	Macro-precision	Macro-recall
Datos no balanceados	**BETO**	**Q9**	0,52	0,75	0,52	0,52
		Q11	0,56	0,77	0,59	0,55
		Q13	0,58	0,64	0,59	0,58
		Q9-Q11-Q13	0,59	0,61	0,59	0,59
	RoBERTa	**Q9**	0,55	0,77	0,56	0,55
		Q11	0,56	0,76	0,58	0,56
		Q13	0,59	0,65	0,60	0,59
		Q9-Q11-Q13	0,60	0,63	0,61	0,60

Datos balanceados	**BETO**	**Q9**	0,52	0,52	0,52	0,52
		Q11	0,52	0,53	0,53	0,53
		Q13	0,52	0,52	0,52	0,52
		Q9-Q11-Q13	0,57	0,57	0,57	0,57
	RoBERTa	**Q9**	0,57	0,58	0,58	0,57
		Q11	0,52	0,53	0,53	0,53
		Q13	0,57	0,57	0,57	0,57
		Q9-Q11-Q13	0,55	0,77	0,56	0,55

Tabla 4.4 Resultados de clasificación con Transformers con hiper-parámetros

4.2 Resultados y discusión

Para discutir mejor los resultados obtenidos en los diferentes experimentos se agruparán los mejores resultados de cada combinación.

Estrategia empleada	Balance del dataset	Pregunta	Macro-F1	Micro-F1	Modelo utilizado
Regresión Logística	–	Q11	0,44	0,78	–
Lexicón iSOL	Datos no balanceados	Q13	0,52	0,57	–
Transformers sin hiper-parámetros	Datos balanceados	Q9-Q11-Q13	0,62	0,62	RoBERTa
Transformers con hiper-parámetros	Datos no balanceados	Q9-Q11-Q13	0,60	0,63	RoBERTa

Tabla 4.5 Resultados de toda la experimentación

- El modelo que mejor ha funcionado ha sido el **RoBERTa**. Se obtenían resultados similares con el modelo BETO, sin embargo, cuando se entrenaba con el dataset correspondiente a la unión de las 3 preguntas (es decir, la pregunta Q9-Q11-Q13), se obtenía un mejor Macro F1-Score con el modelo RoBERTa que el BETO.
- En todas las preguntas, se han obtenido mejores resultados con los modelos de lenguaje aplicándose un fine-tuning de nuestro conjunto de datos.
- La búsqueda de hiper-parámetros ha permitido mejorar de manera conjunta los resultados, a pesar de que el mejor resultado obtenido sin hiper-parámetros es mejor que aquel con hiper-parámetros.

- Hay que tener en cuenta que para este trabajo se ha escogido como métrica objetivo el macro F1-Score pero en función del propósito o del uso que se le vaya a dar al sistema de clasificación, a veces interesará tener una recall alta en la clase positiva (es decir, en la etiqueta de estresada/agotada/quemada). Como el F1-Score combina las métricas de precisión y recall y es la más apropiada para conjuntos de datos desbalanceados pues se ha priorizado pero esto puede cambiar en función de si se aumentase y se pudiese recopilar más datos para enriquecer al dataset.
- En aquellos experimentos con los datasets balanceados, se puede observar como se tiene unas métricas más uniformes e iguales entre sí. Esto puede deberse al hecho de que al balancear los datos, nos quedamos con un número escaso de instancias, no solo en el conjunto de entrenamiento, sino que también en el de test.

4.3 Análisis de errores

Como bien hemos podido ver en las tablas de los apartados anteriores, los diferentes modelos de clasificación que se han desarrollado no son perfectos y a la hora de hacer una predicción sobre el conjunto de test se equivoca dando lugar a falsos positivos (es decir, el modelo ha predicho que la instancia era positiva, siendo realmente negativa) o falsos negativos (el modelo ha predicho que la instancia era negativa, siendo realmente positiva). Por ello, vamos a analizar los errores de predicción que tiene el modelo con mejor resultado el cual es el RoBERTa con el dataset compuesto por la unión de las 3 preguntas.

Lo primero que se mostrará es la matriz de confusión que es una herramienta utilizada en la clasificación de problemas dentro del aprendizaje automático. Es una tabla que permite visualizar el desempeño de un modelo de clasificación al comparar las predicciones realizadas por el modelo con los valores reales de las etiquetas de clase.

Dentro de esta matriz, las filas representan las clases reales y las columnas representan las clases que ha predicho el modelo. Cada celda de la matriz muestra el número o la frecuencia de las instancias que pertenecen a una clase específica según la clasificación realizada por el modelo.

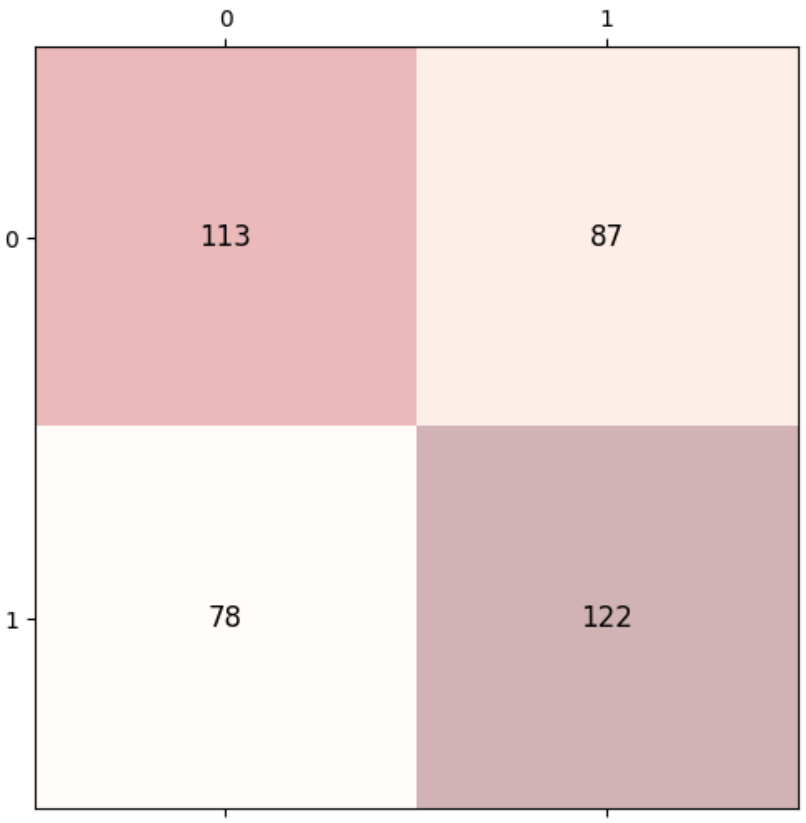


Ilustración 4.2 Matriz de confusión del experimento

En este conjunto de testeo hay un total de 400 instancias, 200 de ellas son positivas y las otras 200 negativas. Gracias a esta matriz podemos ver que, en general, el modelo ha predicho casi el mismo número de instancias positivas como negativas. Viendo las diferentes métricas vemos como la precisión tanto de la clase negativa como positiva son de 0,59 y 0,58 respectivamente, algo que se puede ver reflejado en la matriz de confusión. Vamos a entrar más en detalle a ver aquellas observaciones en las cuales el modelo se ha equivocado prediciendo.

4.3.1 Falsos negativos

Los falsos negativos podrían considerarse como los más peligrosos o aquellos a los cuales hay que darles más importancia a la hora de reducir su número ya que en un caso práctico real, el modelo estaría diciendo que una persona está bien y sana cuando en realidad está estresada o pasa por un mal momento.

Observación o instancia
“Trabajo en una uci”
“No necesito”
“'Con más personal y más medios sería mejor'”

Tabla 4.6 Ejemplos de falsos negativos

En la tabla 4.6 podemos ver algunos ejemplos de instancias cuya clase real es positiva pero el modelo la ha predicho como negativa. Las posibles razones por las cuales ese tipo de observaciones han sido predichas erróneamente son por la corta longitud que tiene estas observaciones y, sobre todo, por el escaso significado léxico. También en algunos casos se puede ver cómo se usan palabras como “mejor”, término el cual el modelo tiene tendencia a verlo en instancias con la otra clase.

4.3.2 Falsos positivos

Observación o instancia
“Trabajo en Salud mental, no estoy en hospital”
“Es una vergüenza en las condiciones en las que hemos trabajado durante estos meses”
“Yo trabajo en consulta , antes estuve en una planta, pero cuando 1 pandemia COVID me desplazaron de centro y llevaron a una planta COVID , por eso las respuestas son un poco dispares”

Tabla 4.6 Ejemplos de falsos positivos

En el caso de los falsos positivos, la mayoría de estas observaciones son clasificadas como positivas debido a que contienen términos que son predominantes en la clase positiva como “COVID”, “mental”, entre otros. Sin embargo, esas palabras en ese contexto de la oración no denotan que la persona está estresada o agotada, sino que simplemente explica de manera informativa e imparcial la situación en la que se encuentra o el cargo que tiene asignado.

4.4 Realización de la interfaz gráfica

Una vez obtenidos los diferentes modelos de clasificación con los mejores parámetros, viene la última parte relacionada con el desarrollo de una demostración

de cómo funcionan estos modelos. Para ello la mejor forma es mediante una aplicación con una interfaz gráfica.

Como esta aplicación debe de tener cargado un modelo de clasificación junto con sus pesos y tokenizador, se buscó una forma de desarrollar una especie de demo que hiciese uso de esos modelos y la herramienta más adecuada fue Gradio [51]. Esta librería de Python permite crear interfaces de usuario rápidas y personalizables para modelos de aprendizaje automático. Proporciona una forma sencilla de compartir y visualizar modelos de aprendizaje automático a través de una interfaz web interactiva.

Esta librería cuenta con una serie de métodos que permiten crear diferentes elementos gráficos como cajas de texto, botones, etc. En nuestro caso, al ser una interfaz gráfica sencilla necesitamos una caja de texto en donde la persona introduzca el texto y otra caja en donde salga los resultados. Como en la experimentación obtuvimos diferentes modelos entrenados sobre diferentes preguntas, la interfaz también cuenta con un sistema de pestañas en donde cada una está asociada a un modelo en específico en función de si la pregunta es si la persona está estresada, agotada emocionalmente, quemada del trabajo o la unión de estas 3.

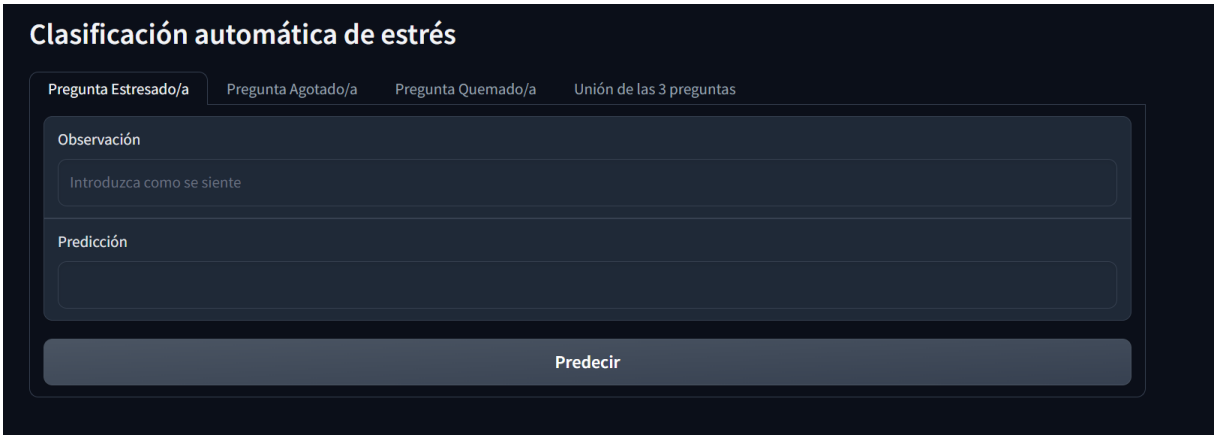


Ilustración 4.3 Interfaz gráfica de la aplicación desarrollada

Por lo tanto, dada una observación introducida en la caja de texto "Observación", la caja de texto "Predicción" muestra la clase que ha sido predicha con el modelo junto a un porcentaje de probabilidad de acierto.

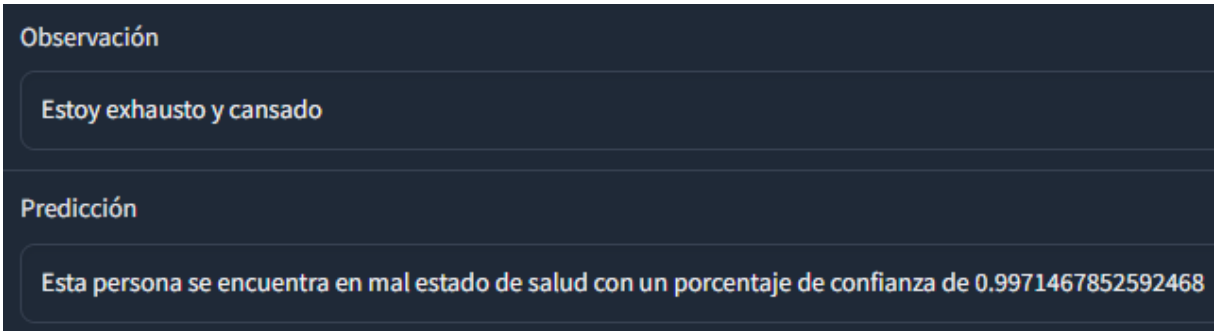


Ilustración 4.4 Ejemplo de predicción con la aplicación

5 Conclusiones y trabajos futuros

Como conclusiones, se ha podido verificar cómo, a través de un buen preprocesamiento del texto a través de técnicas de PLN, tareas complejas como la clasificación de estrés en las personas es posible gracias a un buen tratamiento del texto. Dichas técnicas se emplean y están presentes en la forma que tienen los distintos modelos del lenguaje en procesar el texto. Esta tecnología permite que problemas complejos, como la detección de sentimientos o de estrés en personas, sea posible y se realice de manera rápida y sencilla para la persona, la cual solo tendría que expresar con sus palabras cómo se encuentra.

A pesar de todos los beneficios que presentan los diferentes tipos de modelos que se apoyan en arquitecturas innovadoras, estos modelos de clasificación no son perfectos y hay errores que ocurren debido a un mal procesamiento del texto o por la naturaleza intrínseca del problema. También hay que señalar lo complicado que ya es nuestro lenguaje y que las técnicas para su procesamiento darán mejor o peor resultado en función del contexto y contenido con el que se esté trabajando. Una manera de mejorar los modelos es mediante un aumento en la cantidad de los datos de entrenamiento.

Por último, gracias a este trabajo he podido descubrir y profundizar más en el mundo del PLN, sus novedosas técnicas para permitir que los sistemas informáticos aprendan a procesar el lenguaje como el ser humano lo hace, o al menos de forma prácticamente similar y, sobre todo, tener mi primera toma de contacto con el mundo del aprendizaje automático y machine learning, el cual aúna una colección de estrategias, arquitecturas, modelos matemáticos e informáticos que propician la capacidad de resolver problemas complejos y abstractos gracias al desarrollo de diversas herramientas informáticas. En este caso hemos podido comprobar como, con un corpus limitado de texto, es posible mejorar la salud mental y física de las personas mediante la aplicación de la Inteligencia Artificial y, para ser más precisos, mediante el Procesamiento de Lenguaje Natural

5.1 Trabajos futuros

A pesar de haber finalizado este trabajo, hay multitud de maneras de poder seguir desarrollando este modelo y mejorándolo. Lo primero de todo es re-entrenar el modelo con una base de conocimientos más amplia. Esto podría lograrse a través una ampliación del conjunto de datos original, añadir más recursos léxicos que sean más acertados para el contexto de este problema o aplicar técnicas más innovadoras de sobremuestreo o submuestreo con el objetivo de equilibrar de mejor manera el conjunto de datos.

Otra cuestión importante es la parte de elegir el modelo de lenguaje sobre el cual realizar un fine-tuning de nuestro conjunto de datos. Un aspecto que ha influido ha sido el hecho de que los modelos que se han probado están basados en un corpus muy grande pero cabe la posibilidad de que si se encontrase un modelo cuyo corpus fuese más específico al ámbito médico, se pudieran conseguir mejores resultados.

Para concluir, otro trabajo a realizar posteriormente sería el hecho de profundizar sobre la optimización de hiper-parámetros. Debido a la restricción de tiempo y de capacidad computacional, no se pudo experimentar libremente con las diferentes combinaciones que ofrecía la librería de Optuna, o incluso haber probado con otras librerías enfocadas en la optimización de hiper-parámetros.

6 BIBLIOGRAFÍA

[1] Se acaba la emergencia por la pandemia. https://www.paho.org/es/noticias/6-5-2023-se-acaba-emergencia-por-pandemia-pero-covid-19-continua . Visitado el 12-06-2023

[2] Estudio SATSE “Percepción de estrés en los profesionales de Enfermería en España. Comparativa 2012-2017-2021: Percepción y síntomas de estrés en el colectivo enfermero español” (Fecha de redacción: 01/01/2021)

[3] Revealed: Countries With The Best Health Care Systems, 2021. https://ceoworld.biz/2021/04/27/revealed-countries-with-the-best-health-care-systems-2021/ .Visitado el 13-06-2023

[4] Ben Lutkevich. (2020). Definition of language modelling https://www.techtarget.com/searchenterpriseai/definition/language-modeling. Visitado el 14-04-2023

[5] Alan D. Thompson (2023) Timeline of AI and language models https://lifearchitect.ai/timeline/. Visitado el 14-04-2023

[6] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin. Attention Is All You Need. arXiv:1706.03762, 2017

[7] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever. Improving Language Understanding by Generative Pre-Training

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805, 2018

[9] Modelo BETO: Spanish BERT. https://huggingface.co/dccuchile/bert-base-spanish-wwm-uncased . Visitado el 20-06-2023

[10] Modelo RoBERTa. https://huggingface.co/PlanTL-GOB-ES/roberta-base-bne#model-description . Visitado el 20-06-2023

[11] Fine-tuning a masked language model. https://huggingface.co/learn/nlp-course/chapter7/3?fw=pt . Visitado el 27-06-2023

[12] Jeremy Howard, Sebastian Ruder. Universal Language Model Fine-tuning for Text Classification. arXiv:1801.06146, 2018

[13] Yadullah Abidi. The 5 Best New GPT-4 Features Explained https://www.makeuseof.com/best-new-gpt4-features-explained/. Visitado el 14-04-2023

[14] OpenAI. GPT-4 Technical Report

[15] Hugging Face: modelo BETO dccuchile/bert-base-spanish-wwm-cased. https://huggingface.co/dccuchile/bert-base-spanish-wwm-cased. Visitado el 04-05-2023

[16] GitHub: Corpus utilizado para entrenar BETO josecannete/spanish-corpora. https://github.com/josecannete/spanish-corpora. Visitado el 04-05-2023

[17] Lexicón iSOL SINAI. https://sinai.ujaen.es/investigacion/recursos/isol. Visitado el 04-05-2023

[18] Molina-González, M. D., Martínez-Cámara, E., Martín-Valdivia, M. T., & Perea-Ortega, J. M. (2013). Semantic orientation for polarity classification in Spanish reviews. Expert Systems with Applications, 40(18), 7250-7257.

[19] Bing Liu (2015). Opinion Mining, Sentiment Analysis, and Opinion Spam Detection. https://www.cs.uic.edu/~liub/FBS/sentiment-analysis.html. Visitado el 04-05-2023

[20] Página oficial de Scikit-Learn. https://scikit-learn.org/stable/getting_started.html . Visitado el 04-06-2023

[21] Anirudha Simha (2021). Understanding TF-IDF for Machine Learning. https://www.capitalone.com/tech/machine-learning/understanding-tf-idf/. Visitado el 04-06-2023

[22] About Hugging Face. https://apply.workable.com/huggingface/. Visitado el 15-06-2023

[23] Hugging Face. Transformers. https://huggingface.co/docs/transformers/index#transformers. Visitado el 15-06-2023

[24] Hugging Face. https://huggingface.co/. Visitado el 04-06-2023

[25] Python. https://www.python.org/. Visitado el 04-06-2023

[26] Pandas. https://pandas.pydata.org/about/. Visitado el 04-06-2023

[27] Numpy. https://numpy.org/doc/stable/. Visitado el 04-06-2023

[28] Transformers,Hugging Face. https://github.com/huggingface/transformers. Visitado el 04-06-2023

[29] Natural Language Toolkit. https://www.nltk.org/. Visitado el 04-06-2023

[30] Matplotlib: Visualization with Python. https://matplotlib.org/. Visitado el 04-06-2023

[31] Seaborn: Statistical data visualization. https://seaborn.pydata.org/. Visitado el 04-06-2023

[32] Google Collaboratory. Preguntas frecuentes. https://research.google.com/colaboratory/intl/es/faq.html. Visitado el 04-06-2023

[33] Optuna: A hyperparameter optimization framework. https://optuna.org/ .Visitado el 04-06-2023

[34] Las 5 Mejores Metodologías de Desarrollo de Software. https://gooapps.es/2022/10/27/las-5-mejores-metodologias-de-desarrollo-de-software/ . Visitado el 05-06-2023

[35] Modelo incremental - Ingeniería de Software. http://isw-udistrital.blogspot.com/2012/09/ingenieria-de-software-i.html . Visitado el 05-06-2023

[36] El modelo COCOMO. http://www.sc.ehu.es/jiwdocoj/mmis/cocomo.htm . Visitado el 05-06-2023

[37] Licencias de Windows 10 y 11: tipos, ediciones, precios y dónde comprar. https://www.lavanguardia.com/andro4all/compras/licencias-windows#:~:text=En%20cuanto%20a%20las%20versiones,la%20versi%C3%B3n%20Enterprise%20para%20empresas. Visitado el 06-06-2023

[38] Dell 16GB NVIDIA® Tesla® T4 GPU Tarjeta grafica https://www.dell.com/es-es/shop/dell-16gb-nvidia-tesla-t4-gpu-tarjeta-grafica/apd/490-beym/tarjetas-gr%C3%A1ficas. Visitado el 07-06-2023

[39] Teresa Arora, Ian Grey, Linda Östlundh, Kin Bong Hubert Lam, Omar M Omar, and Danilo Arnone. The prevalence of psychological consequences of COVID-19: A

systematic review and meta-analysis of observational studies. https://doi.org/10.1177/1359105320966639

[40] Transformer: A Novel Neural Network Architecture for Language Understanding. https://ai.googleblog.com/2017/08/transformer-novel-neural-network.html. Visitado el 10-06-2023

[41] Deep Learning 101: Beginners Guide to Neural Networks. https://www.analyticsvidhya.com/blog/2021/03/basics-of-neural-network/ . Visitado el 11-06-2023

[42] ¿Qué son las redes neuronales? https://www.ibm.com/es-es/topics/neural-networks . Visitado el 11-06-2023

[43] Stefania Cristina. (2022). The Transformer Model https://machinelearningmastery.com/the-transformer-model/. Visitado el 21-04-2023

[44] Hugging Face NLP course. https://huggingface.co/learn/nlp-course/chapter1/1. Visitado el 27-06-2023

[45] Google Colab. https://colab.research.google.com/?hl=es. Visitado el 27-06-2023

[46] NRC Word-Emotion Association Lexicon. https://saifmohammad.com/WebPages/NRC-Emotion-Lexicon.htm. Visitado el 19-06-2023

[47] Scikit Learn. TfidfVectorizer. https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html. Visitado el 19-06-2023

[48] Epoch vs Batch Size vs Iterations https://towardsdatascience.com/epoch-vs-iterations-vs-batch-size-4dfb9c7ce9c9 . Visitado el 19-06-2023

[49] Micro, Macro & Weighted Averages of F1 Score, Clearly Explained. https://towardsdatascience.com/micro-macro-weighted-averages-of-f1-score-clearly-explained-b603420b292f#989c . Visitado el 20-06-2023

[50] Hugging Face Course. Tokenizers. https://huggingface.co/learn/nlp-course/chapter2/4?fw=pt Visitado el 20-06-2023

[51] Gradio. https://gradio.app/. Visitado el 27-06-2023