UNIVERSIDAD CATÓLICA SANTO TORIBIO DE MOGROVEJO

FACULTAD DE INGENIERÍA

ESCUELA DE INGENIERÍA DE SISTEMAS Y COMPUTACIÓN

Aplicación móvil basada en técnicas de clasificación de machine learning como apoyo en el reconocimiento de emociones en textos de estudiantes universitarios

TESIS PARA OPTAR EL TÍTULO DE

INGENIERO DE SISTEMAS Y COMPUTACIÓN

AUTOR

Sara Maria Benel Ramirez

ASESOR

Mariana Chavarry Chankay

https://orcid.org/0000-0001-5136-7177

Chiclayo, 2023

Aplicación móvil basada en técnicas de clasificación de machine learning como apoyo en el reconocimiento de emociones en textos de estudiantes universitarios

PRESENTADA POR

Sara Maria Benel Ramirez

A la Facultad de Ingeniería de la
Universidad Católica Santo Toribio de Mogrovejo
para optar el título de

INGENIERO DE SISTEMAS Y COMPUTACIÓN

APROBADA POR

Segundo José Castillo Zumarán
PRESIDENTE

William Alfredo Noblecilla Vinces
SECRETARIO

Mariana Chavarry Chankay
VOCAL

Dedicatoria

A mis padres que me dieron la oportunidad de estudiar y su apoyo incondicional.
A toda mi familia que me ha servido de inspiración, ha creído en mí y me ha dado la fortaleza para no rendirme.

Agradecimientos

A mis amistades por su inagotable soporte, afecto, y conocimiento que me proporcionan día a día.
Al personal docente por sus enseñanzas, dedicación y paciencia que brindaron durante toda la carrera.
A los profesionales que me brindaron su tiempo, consejos y ayuda para realizar este trabajo.

Tesis

INFORME DE ORIGINALIDAD

11%	11%	0%	%
INDICE DE SIMILITUD	FUENTES DE INTERNET	PUBLICACIONES	TRABAJOS DEL ESTUDIANTE

FUENTES PRIMARIAS

1	hdl.handle.net Fuente de Internet	6%
2	tesis.usat.edu.pe Fuente de Internet	1%
3	repositorio.unan.edu.ni Fuente de Internet	<1%
4	sedici.unlp.edu.ar Fuente de Internet	<1%
5	gestion.pe Fuente de Internet	<1%
6	qdoc.tips Fuente de Internet	<1%
7	core.ac.uk Fuente de Internet	<1%
8	repositorio.uladech.edu.pe Fuente de Internet	<1%
9	zaguan.unizar.es Fuente de Internet	<1%

Índice

Lista de tablas

Lista de figuras

Resumen

Los estudiantes universitarios están expuestos a distintos factores económicos, sociales y académicos que afectan su estado emocional, adicionalmente estos tienden a ignorar su salud mental lo que es perjudicial a largo plazo. Debido a esto, en la presente investigación se pretende construir una aplicación móvil a través de la cual los escolares puedan llevar un control de su estado anímico con tan solo responder unas simples preguntas. Para poder identificar estas emociones en las respuestas se desarrolló un modelo con una técnica de Machine Learning denominada redes neuronales de tipo Transformer y se desplegó en un servicio web. Este modelo tiene la capacidad de clasificar texto en 6 emociones diferentes como son: tristeza, alegría, enojo, miedo, amor y sorpresa. En la validación se alcanzó una exactitud de 93%, un promedio en la precisión de 89% y en el promedio del puntaje F1 un 88%. Así mismo se creó una aplicación móvil para los estudiantes y una plataforma web de administración en donde se pueda observar el historial de las emociones registradas.

Palabras clave: Machine Learning, Deep Learning, Transformers, Clasificación Multiclase de Texto.

Abstract

University students are exposed to different economic, social and academic factors that affect their emotional state, additionally they tend to ignore their mental health which is detrimental in the long term. Because of this, this research aims to provide a mobile application through which students can keep track of their mood by answering a few simple questions. To identify these emotions in the answers, a model was developed with a Machine Learning technique called Transformer neural networks and deployed in a web service. This model has the ability to classify text in 6 different emotions such as: sadness, joy, anger, fear, love and surprise. The validation achieved an accuracy of 93%, an average accuracy of 89% and an average F1 score of 88%. Likewise, a mobile application was created for students and a web administration platform where the history of recorded emotions can be observed.

Keywords: Machine Learning, Deep Learning, Transformers, Multiclass Text Classification.

Introducción

Las emociones juegan un importante papel en la persona. Se ha demostrado que las emociones son capaces de influir en la toma de decisiones [1], en la regulación de la actividad física [2], en la salud mental y comportamiento [3]. La salud mental, por su parte, cuenta con factores de riesgo como situaciones estresantes, antecedentes de enfermedad mental, consumo de alcohol o drogas, relaciones poco saludables y experiencias traumáticas [4], [5] que van desgastando nuestra salud mental y aumentando el riesgo de sufrir una enfermedad mental [6].

Una población que comúnmente vive bajo factores de riesgo y tiene descuidada su salud mental es la comunidad universitaria. A nivel internacional podemos observar esto en la encuesta del 2021 realizada por la Asociación Americana de Salud Universitaria en donde se aprecia que el 27.3% de estudiantes fueron diagnosticados de ansiedad y el 21.5% de depresión, además, a través un cuestionario, se detectó un comportamiento suicida en el 28% de los escolares, de los cuales el 3.1% tuvo un intento de suicido en los últimos 12 meses [7]. A nivel nacional, en el artículo "Determinantes de la salud mental en estudiantes universitarios de Lima y Huánuco" se encontró como resultado que los factores determinantes en la salud eran: estrés, estilo de afrontamiento evitativo, centro de estudios y área interpersonal. También se menciona que una de las causas que hace de la vida universitaria estresante en el Perú es la mezcla de factores como el nivel académico, financiero y social del estudiante universitario [8]. Por otro lado, el Ministerio de Educación (Minedu) ha alertado que el 85% de la comunidad educativa peruana presenta problemas de salud mental y los casos más recurrentes son de ansiedad, estrés y violencia [9]. Adicionalmente en el Perú, en la publicación del Consorcio de Universidades, se reportaron presencia de síntomas extremadamente severos de ansiedad (28%), depresión (25%) y estrés (12%) junto con una idealización suicida del 30.8% [10].

Por esta razón es necesario prestar atención al estado emocional de los estudiantes. La OMS menciona que entre los síntomas de un trastorno emocional pueden estar las reacciones excesivas de irritabilidad o cambios inesperados en el estado de ánimo [11]. Además, de acuerdo con la Clínica Mayo, cuando estos los síntomas se vuelven permanentes es probable que evidencien un problema de salud mental y así afecten nuestro desempeño en la sociedad y nuestras relaciones interpersonales. En concordancia con esto se encuentran los resultados de la encuesta realizada en esta investigación a estudiantes universitarios, en dónde se encontró que el 54.3% afirma haber vivido situaciones de intensa tristeza y estrés por causas universitarias, de entre las cuales las más señaladas fueron: exámenes (68.6%), calificaciones (62.9%) y tareas (62.9%) [12]. Asimismo, el 87.1% considera que su estado anímico afecta su

rendimiento académico lo cual indica una relación directa de causalidad entre estos factores, que se traduce en mejor rendimiento si se tiene un mejor estado de ánimo [12].

Ahora bien, aunque las emociones tengan un papel clave en la persona, los estudiantes parecen no tomar conciencia de su propio estado emocional. Es el estilo de afrontamiento evitativo el más popular entre los jóvenes que normalmente consideran otras actividades como más importantes, descuidando así su bienestar emocional [8]. Este estilo de afrontamiento aumenta el riesgo de desarrollar trastornos o enfermedades psicológicas más adelante [13]. Es por esto por lo que el problema que se presenta es el poco control y monitoreo que los estudiantes universitarios tienen de sus emociones.

Para abordar esta problemática se planteó como objetivo general implementar una aplicación móvil basada en técnicas de clasificación de Machine Learning como apoyo en el reconocimiento de emociones en textos de estudiantes universitarios. Y como objetivos específicos: determinar la técnica de clasificación de Machine Learning adecuada para el reconocimiento de emociones basadas en textos, desarrollar el modelo predictivo basado en la técnica de clasificación de Machine Learning previamente identificada para reconocer emociones en textos, validar el modelo de clasificación para verificar su precisión en el reconocimiento de emociones en textos y desplegar el modelo de clasificación validado en una aplicación móvil como apoyo en el reconocimiento de emociones en textos de estudiantes universitarios.

Esta investigación encuentra justificaciones desde diversas perspectivas. De manera científica porque se ha realizado una investigación de las diferentes técnicas de clasificación de texto en Machine Learning desarrollándose así un modelo basado en la arquitectura de Transformers capaz de clasificar texto, además que se propone un nuevo uso para estos modelos que comúnmente son usados por empresas para analizar los comentarios de sus productos. Socialmente debido a que el estado anímico juega un papel importante en el desenvolvimiento de una persona en la sociedad y el tomar conciencia sobre este genera un mayor autocuidado. Económicamente debido a que se buscó minimizar los costos de esta investigación usando recursos de código abierto y herramientas gratuitas.

Revisión de literatura

En esta sección se identifican trabajos de investigación con una problemática relacionada en los que también se realizó la tarea de clasificar texto. Adicionalmente se revisan conceptos fundamentales para el entendimiento del problema y el desarrollo de la solución.

Antecedentes

Se han considerado para esta investigación antecedentes nacionales, internacionales y locales.

Antecedentes Internacionales

Sailunaz y Alhajj [14] en su investigación se proponen analizar tweets con el objetivo de generar recomendaciones a partir de la detección de sentimientos y emociones de estos. la clasificación tanto en el análisis de sentimientos como en el de emociones se utilizó el algoritmo Naive Bayes (NB) para clasificar los textos dentro de las emociones definidas, las cuales fueron: enojo, disgusto, alegría, miedo, tristeza, sorpresa y neutral. También se comparó la exactitud de este algoritmo con el de Support Vector Machine (SVM) y Random Forest (RF) dando como resultado una mayor precisión por parte del algoritmo de Naive Bayes. Esta investigación contribuye con una perspectiva diferente de cómo abordar el problema de la clasificación de textos, específicamente aplicando el algoritmo probabilístico de NB.

Cao et al. [15] realiza una investigación con el objetivo de analizar el texto de publicaciones en la red social Weibo para identificar emociones en los habitantes de Wuhan en China durante la cuarentena por COVID-19. En este trabajo se utilizó una red neuronal Bi-LSTM (Bidireccional Long Short Term Memory) que clasificó la data en siete emociones: admiración, esperanza, alegría, neutro, miedo, reprobación y angustia. También se comparó este método con otros como: SVM, RNN, CNN y Single-LSTM dando como resultado que este modelo tenía una mejor performance. Las métricas obtenidas fueron una precisión de 0.714, un recall de 0.704 y un F1-score 0.706. El valor agregado que se distingue en este trabajo es como se propone una solución diferente que da mejores resultados.

En el trabajo de Graterol et al. [16] se plantea implementar la detección de emociones en robots sociales basándose en Transformers. Para esta implementación debido al trabajo que realizan los robots se ve necesario que puedan procesar distintos tipos de fuentes como imágenes, textos, voz. La cantidad de emociones a detectar se definieron en: enojo, anticipación, disgusto, miedo, alegría, amor, optimismo, pesimismo, tristeza, sorpresa y confianza. Con el objetivo de detectar emociones en el texto, primero realizaron una tarea de clasificación identificando si el texto poseía una emoción o no. Como segunda tarea, los textos que sí poseían una emoción se procesaron con Transformers preentrenados y posteriormente se procesan con un perceptrón multicapa. Este modelo presenta una precisión de 0.535. En este caso el valor agregado que lo diferencia es la manera de aproximarse a una solución tomando

en consideración la existencia de textos neutros y su previa filtración para que no interfieran con la clasificación en emociones.

Antecedentes Nacionales

En la tesis de Morzán [17] se logra la elaboración de una aplicación capaz de detectar el estado de ánimo del usuario a través de una conversación en español. Aquí a través del procesamiento del lenguaje natural (NLP, por sus siglas en inglés) se emplea una técnica llamada Reconocimiento de Palabras Clave y como resultado de un total de 49 pruebas se alcanzó el éxito de 70% de aciertos en el estado de ánimo. La contribución de este trabajo es que se puede apreciar la utilización de esta técnica clásica de NLP.

En otro trabajo de la misma universidad de Reyes-Paredes [18] se encarga de clasificar textos de titulares periodísticos en sentimientos positivos o negativos con el fin de determinar si las buenas noticias tenían un impacto beneficioso en el estado de ánimo. Para lograr su objetivo usó una red neuronal Long Short Term Memory (LSTM) para la clasificación de texto. Para la implementación de la red neuronal se comparó su "accuracy" con el método Naive Bayes, dando como resultado que la primera mencionada presentaba una mejor performance con 87,98%. Lo que se puede apreciar de este trabajo es que se demuestra una superioridad de la precisión que pueden ofrecer las redes neuronales LSTM sobre el método Naive Bayes si estas son implementadas correctamente.

En la investigación de Cuzcano [19] se requiere comparar modelos de clasificación con el objetivo de detectar ciberbullying en tweets. Se entrenaron y compararon modelos de Multinomial Logistic Regression, SVM, NB y Random Forest. Se obtuvo que el clasificador de mejor performance fue el Support Vector Machine con una precisión de 0.687 con el uso de Tri-grams y TFIDF. El valor que aporta esta investigación es que se considera el entrenamiento de Multinomial Logistic Regression que no es considerado en las investigaciones anteriores.

Antecedentes locales

En el trabajo de grado de Segura [20], se afronta la tarea de evaluar distintos mecanismos de clasificación para el minado de opinión en la plataforma web de Twitter. Se realizó la comparación de algoritmos como Naive Bayes, Support Vector Machine y Regression Tree. Estos se compararon usando la métrica de precisión usando diferentes escenarios. De lo cual se concluyó que la técnica de SVM fue más precisa con 74%. El valor agregado que se rescata de

esta investigación es la diferente propuesta de solución que se plantea con estos algoritmos de ML.

Bases teórico-científicas

En esta sección se conceptualizan los términos de los componentes que forman parte de la solución propuesta. Para la realización de esta investigación se utilizaron conocimientos del campo de NLP y ML.

Procesamiento del Lenguaje Natural

El procesamiento del lenguaje natural (NLP) es una rama de la Inteligencia Artificial (IA) que tiene el objetivo de dotar a las computadoras con la capacidad de comunicarse con las personas en un lenguaje natural [21]. Por lenguaje natural se hace referencia a un lenguaje humano y no a lenguajes formales como los lenguajes de programación [22]. Este campo alcanza su objetivo a través de la investigación y desarrollo de técnicas computacionales [23] que utiliza para el entendimiento del lenguaje natural (NLU, Natural Language Understanding) y la generación del lenguaje natural (NLG, Natural Language Generation) [24].

Para entender el lenguaje el NLP hace 6 distintos tipos de análisis del lenguaje. El análisis morfológico que se centra en buscar la palabra origen de la palabra ingresada; también encuentra su género, número y persona [24]. El análisis léxico que realiza un análisis a nivel de palabra; en este proceso se fragmenta el texto ingresado en palabras y se omiten los espacios en blanco para poder tokenizar estas palabras [24]. El análisis semántico que se encarga del significado de cada palabra; acá nos aseguramos de que el significado que tienen sea claro y coherente [22]. El análisis sintáctico que es el encargado de evaluar si la estructura sintáctica de la frase es correcta [23]. El análisis pragmático que es en dónde buscamos extraer algún tipo de información del texto ingresado [24]. Y finalmente el análisis de discurso que se centra en la colección de oraciones y en la coherencia que tienen entre sí [22].

Dentro de este campo encontramos diversas tareas de acuerdo a [25]. De estas podemos mencionar al modelamiento del lenguaje que consiste en predecir la siguiente palabra de una oración; también a la clasificación de texto que busca clasificar a un texto dentro de un set de categorías previamente conocidas; a la extracción de información que se basa en extraer información relevante de un texto; a la creación de agentes conversacionales que consiste en construir sistemas de diálogo que puedan comunicarse en lenguaje humano, ejemplo de esto tenemos a los asistentes virtuales como Siri, Asistente de voz, Alexa, etc. ; a la recuperación de

información en donde se pretende devolver como resultado de una consulta un documento relevante dentro una gran colección, esto se materializa en los buscadores como Google o Bing; y a la traducción automática que su objetivo es traducir de un idioma humano a otro, producto de esta tarea son las herramientas de traducción como DeepL, Traductor de Google.

Para poder llevar a cabo las tareas de procesamiento de lenguaje natural es necesario que el texto pueda ser entendido por estas soluciones. Este es el primer reto para poder crear cualquier modelo que implique procesar texto, y actualmente existen diversos métodos para poder representar el texto en números o estructuras numéricas [26]. Entre las representaciones más famosas tenemos a label encoding, one hot encoding y word embeddings. Esta última es la utilizada en esta investigación Aquí se usan redes neuronales que buscan encontrar similitudes entre palabras por el contexto en que son ingresadas [25]. Inicialmente se definen los vectores de una dimensionalidad determinada (10,500,800) con pesos al azar para cada palabra, estos pesos irán variando conforme la red neuronal mejore en encontrar similitudes [27]. Finalmente, estos pesos pueden ser guardados para ser utilizados posteriormente [27]. Actualmente contamos con herramientas como Word2Vect o GloVe que son embeddings preentrenados que podemos utilizar en distintas soluciones [28].

Machine Learning

El Machine Learning (ML) es un campo de la IA que puede ser descrito o explicado de diversas maneras. Una de las maneras más típicas y de fácil entendimiento es describirla como el campo que busca que un sistema aprenda a hacer determinada tarea sin ser explícitamente programado para hacerla, es decir dotar a este sistema de la capacidad de aprender por sí solo [29]. Otra forma más detallada explica que es una disciplina que usa algoritmos capaces de mejorar su performance basándose en encontrar patrones en los datos o aprender de experiencias previas mediante prueba y error [30].

Dentro del Machine Learning existen 3 formas populares de entrenar a un modelo para que aprenda automáticamente: la manera supervisada, no supervisada y reforzada. En esta investigación se utilizó el aprendizaje supervisado que consiste en brindarle tanto los datos de entrada como de salida al modelo, es decir le darás los datos que quieres ingresar y también los datos que deseas recibir como resultado [31]. Esto con el objetivo de que en la fase de entrenamiento el modelo descubra la relación entre estos datos y la aprenda [30]. Así al aprender la relación será capaz de dar resultados cuando se le ingresen datos nuevos.

Las redes neuronales son un método dentro del Deep Learning (DL) que tienen la capacidad de aprender automáticamente. La arquitectura del modelo utilizado en este trabajo es la arquitectura que se describe a continuación:

- Transformers

Los Transformers es la arquitectura del modelo preentrenado que se utiliza en el presente trabajo. Estos constan de dos grandes secciones: los codificadores y decodificadores [32] y se caracterizan porque se basan en la utilización del mecanismo de atención resultando en un entrenamiento más rápido y de mejor resultado [32]. Este proceso de atención se lleva a cabo en la capa de atención y lo que se busca específicamente es que cada palabra calcule su relación con las otras palabras de la oración. Ahora, debido a que esta arquitectura posee 8 cabezas de atención este proceso se realiza ocho veces paralelamente obteniendo 8 matrices resultantes [32]. No obstante, la siguiente capa espera solo una matriz por lo cual se concatena todas las matrices y se multiplican por otra matriz de pesos que es entrenada juntamente con el modelo [32]. Un detalle en la arquitectura de los Transformers es que cada capa de atención y prealimentada es seguida por una capa de normalización. Esta capa ayuda a reducir el tiempo de entrenamiento y a normalizar las salidas ayudando así a reducir pesos muy elevados que pueden ocasionar un sesgo en la red neuronal [33].

Machine Learning, Deep Learning y NLP

Estos son 3 campos que pertenecen a la Inteligencia Artificial y se relacionan entre ellos para afrontar distintos problemas. Esta relación se observa de manera gráfica en *Fig. 4*.

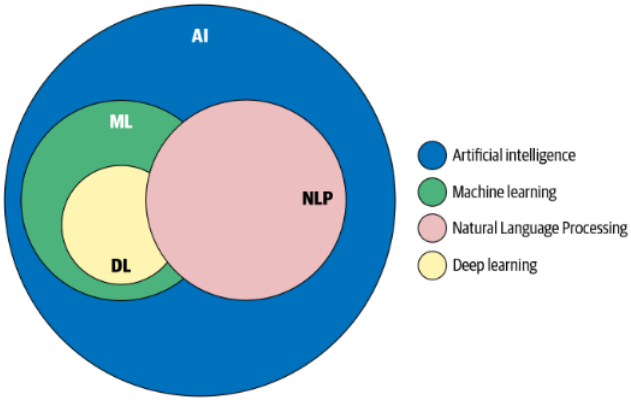


Fig. 1 Relación entre ML, DL y NLP [34]

Esta relación se evidencia de manera práctica en que ciertas técnicas del ML y DL son utilizadas para resolver tareas del NLP. Dentro del campo del ML se observan técnicas como Naive Bayes, Support Vector Machine, Hidden Markv Model, Conditional Random Fields [25].

Y en el campo del DL específicamente se hace uso de los diferentes tipos de redes neuronales que existen, entre ellas: Recurrent Neural Networks, Long Short-Term Memory, Convolutional Neural Networks y Transformers [25].

Materiales y métodos

En esta sección se indica el tipo de investigación, los métodos, técnicas y procedimientos que se llevaron a cabo en la realización de este proyecto.

Tipo de investigación

Esta investigación es de tipo aplicada y cuasiexperimental. Aplicada debido a que está dirigida explícitamente a cumplir con un objetivo práctico [35] y cuasiexperimental se cuenta con un solo grupo de control [36].

Método de investigación

Los métodos de investigación empleados son los mencionados en la *TABLA I*.

TABLA I
MÉTODOS DE INVESTIGACIÓN

Método	Descripción
Analítico [37]	Estudio y análisis del problema que se presenta
Deductivo [37]	Estrategia para el planteamiento de la propuesta de solución al problema
Implementación [38]	Despliegue del modelo de clasificación de texto en producción

Técnicas e instrumentos de recolección de datos

En la *TABLA II* se muestran las técnicas e instrumentos que fueron útiles para la recolección de datos con el objetivo de obtener datos confiables para su procesamiento y análisis.

TABLA II
TÉCNICAS E INSTRUMENTOS DE RECOLECCIÓN DE DATOS

Técnicas	Instrumentos	Elementos de la población	Propósito
Análisis documental	Ficha de registro	Informes estadísticos, artículos científicos	Identificar el problema y sus características
Juicio por expertos	Plantilla Juicio de expertos	Preguntas elaboradas	Validar el formulario elaborado para los estudiantes
Encuesta	Plantilla de encuesta	Estudiantes	Comprensión del problema

Procedimientos

Metodología de desarrollo

Para el desarrollo del modelo se considerará a ‘Machine Learning Project Checklist’ de Aurélien Géron [34] y para la aplicación móvil a ‘Scrum’ [39] de Schwaber y Sutherland. Las

actividades que se realizaron en la primera metodología son: definir el problema; obtener la data; descubrir y visualizar los datos para obtener información; preparación de los datos; elegir y entrenar un modelo; afinar el modelo; y finalmente desplegar el modelo. Después de tener el modelo listo se procedió con la construcción de la aplicación móvil. Para esto se realizó inicialmente la creación de un product backlog [40] que se observa en la *TABLA III*, tal como indica la metodología Scrum.

TABLA III
PRODUCT BACKLOG

Ítem	Product backlog
1	Autenticar al usuario
2	Implementar agente chatbot
3	Enviar mensajes al agente chatbot
4	Identificar y registrar emociones
5	Mostrar estadísticas de los registros de emociones
6	Editar datos del usuario
7	Iniciar sesión en aplicación web
8	Visualizar datos de los estudiantes en aplicación web
9	Crear, eliminar y actualizar estado de usuarios para la aplicación web
10	Cambiar contraseña del usuario en la aplicación web

Las actividades designadas dentro de cada Sprint son:

- Planeamiento del Sprint: Esta es una etapa clave en la realización de un sprint. Es aquí en donde se definirá el 'Objetivo del Sprint' que es el indicador que nos avisa cuando la iteración ha terminado, también se seleccionarán elementos del Product Backlog para incluirlos en el Sprint actual [41].
- Reuniones diarias: Estas reuniones son una actividad común pero importante. Son de corta duración y sirven para revisar el progreso realizado y adaptar el Sprint Backlog [41].
- Revisión del Sprint: Esta etapa se dará con el objetivo de revisar el resultado del Sprint y determinar si cumplió con el objetivo o no.
- Retrospectiva del Sprint: Esta etapa final se utilizará para determinar o planificar formas de mejorar la calidad o efectividad.

Consideraciones éticas

En esta sección se listan las atenciones éticas que se tuvieron para la realización de la solución, desde la recolección de información hasta el desarrollo de la aplicación:

- Protección de los derechos de autor: se respetó la propiedad intelectual y no se omitió ninguna cita.

- Confidencialidad de los datos: se respetó la privacidad de los estudiantes que participaron de la encuesta y se elaboró una política de privacidad en la aplicación detallando que la aplicación no podrá compartir los datos con terceros.
- Seguridad de la información: se tomó en cuenta la seguridad de los datos almacenados en la base de datos a través de reglas para la lectura y escritura de los datos.
- Protección de contraseñas y datos registrados: las contraseñas son encriptadas al momento de guardarse en la base de datos.

Resultados y discusión

En esta sección se presentarán los resultados obtenidos de acuerdo con los objetivos planteados, además se mencionan los impactos que se espera tener. Se apoyarán los resultados con evidencias como imágenes, métricas y descripciones.

OE1: Determinar la técnica de clasificación de Machine Learning adecuada para el reconocimiento de emociones basadas en textos.

Este primer objetivo se estableció con la finalidad de contar con el mejor método de clasificación que se adapte a los objetivos y necesidades de esta investigación. Esto debido a que se requiere un modelo que logre una gran performance, pero al mismo tiempo no requiera mucha capacidad computacional debido a que se aloja en un servidor de pocos recursos y será la base de un servicio web.

Para cumplir con este fin se seleccionaron técnicas de clasificación de texto, se compararon y se eligió la más adecuada. Así, después de una extensa investigación y tomando en consideración los antecedentes, se seleccionaron 4 técnicas de clasificación para ser comparadas. Las técnicas consideradas fueron:

- Naive Bayes: Es un clasificador probabilístico basado en el teorema de Bayes que se entrena de manera supervisada. Este algoritmo de Machine Learning durante el entrenamiento estima la probabilidad de que cada característica pertenezca a cada clase [42]. Al momento de analizar reconoce características en el texto y multiplica su probabilidad de pertenencia para todas las clases y finalmente elige la mayor [25]. Este es un modelo generativo, lo que indica que se fija en la distribución del dataset para retornar la probabilidad.
- Support Vector Machine: Este al igual que el Naive Bayes es un algoritmo de Machine Learning, sin embargo se diferencia porque es un clasificador discriminativo [43]. Su objetivo es encontrar un hiperplano que separe las clases con el mayor margen posible,

cabe resaltar que estas separaciones pueden ser no lineales [25]. Además añadir que algo característico de esta técnica es que toma como soportes a los vectores de los casos que se encuentran más cerca de la frontera entre las clases y a partir de estos halla la dirección y la posición del hiperplano [42].

- LSTM: Este modelo es un tipo de red neuronal recurrente las cuales pertenecen a los modelos discriminativos [43]. Esta arquitectura pertenece específicamente al campo del Deep Learning y al ser un tipo de modelo recurrente analiza los datos de manera secuencial, algo que es muy conveniente cuando trabajamos con texto ya que este es naturalmente una secuencia de palabras ordenadas [44]. Al mismo tiempo algo que las caracteriza dentro de las redes recurrentes es que poseen un tipo especial de neuronal llamada "memory cell" la cual ayuda a mantener cierta información pasada debido a que se encarga de manejar que información será recordada y cuál será olvidada [45].
- Transformer: Este modelo también pertenece específicamente al campo del Deep Learning y, a diferencia de la arquitectura LSTM, no es un tipo de red neuronal recurrente. En este modelo lo característico es que analiza las oraciones de manera paralela, posibilitando que el entrenamiento sea más rápido, además está basado en el mecanismo de atención, el cual se describió a detalle en la sección de bases científicas, que en síntesis busca relaciones entre las palabras que componen la sentencia y analiza de diferentes maneras esta relación [32]. Gracias a este tipo de análisis se logra una mejor comprensión del texto y por lo tanto performance.

A continuación, se describirán los criterios que se establecieron para realizar la comparación:

- Tipo de modelo: Dentro de este criterio nos encontramos con modelos discriminativos y generativos, que se diferencian por la manera en que resuelven la tarea de la clasificación [43]. Los discriminativos, o también conocidos como condicionales, se centran en crear límites que separen correctamente a las clases. Una de sus ventajas es que frente a los valores atípicos estos son mejores que los generativos, y una de sus desventajas es que este tipo de modelos no puede generar nuevos puntos de datos [46]. Por otro lado, los modelos generativos se centran más en la distribución de las clases, y comúnmente usan el teorema de Bayes para hallar la probabilidad conjunta [47]. Cabe resaltar que estos últimos si son capaces de generar nuevos datos por lo que son usados para una variedad de tareas, mientras que los discriminativos solo para clasificación. Se valorará en este caso a los discriminativos debido a que han demostrado una mejor performance al momento de clasificar tanto en los antecedentes como en otras investigaciones comparativas [43], [47].

TABLA IV
VALORACIÓN DE TIPO DE MODELO

Categoría	Valoración numérica
Generativo	1
Discriminativo	2

- Tipo de aprendizaje: En este criterio se hace referencia al Machine Learning y Deep Learning. Algo que aclarar en este punto es que todos los algoritmos presentados pertenecen al campo del Machine Learning, por lo tanto lo que se pretende aquí es discernir cuáles técnicas pertenecen al subcampo del aprendizaje profundo y cuáles otras solo al aprendizaje automático [25]. El Deep Learning es un campo dentro del ML que se caracteriza porque hace uso de las redes neuronales y es así capaz de resolver problemas de clasificación más complejos que los algoritmos solo pertenecientes al ML [48], [49]. Una desventaja de este subcampo es que necesita un gran dataset para ser entrenado y mayor poder computacional. Sin embargo, al momento de trabajar con imágenes, texto o audio es capaz de realizar un mejor trabajo [25], [50]. En consecuencia, y tomando en cuenta los antecedentes, se valorará a las técnicas de DL debido a que este es un problema complejo ya que se trabaja con texto y múltiples clases.

TABLA V
VALORACIÓN DE TIPO DE APRENDIZAJE

Categoría	Valoración numérica
Machine Learning	1
Deep Learning	2

- Tipo de procesamiento: Este criterio es para analizar si el procesamiento de los datos se da de manera secuencial o paralela. El análisis secuencial va procesando en orden elemento por elemento lo que ocasiona que se demore mucho más al momento del entrenamiento. Por otro lado, el análisis paralelo lidia con todos los elementos de una oración al mismo tiempo [51]. Se tendrá en cuenta a los modelos que ofrecen un procesamiento paralelo debido a que es más rápido y favorece a los tiempos de la investigación.

TABLA VI
VALORACIÓN DE TIPO DE PROCESAMIENTO

Categoría	Valoración numérica
Secuencial	1
Paralelo	2

El cuadro de comparación se muestra en la *TABLA VII*.

TABLA VII
CUADRO COMPARATIVO DE TÉCNICAS DE CLASIFICACIÓN

Criterio	NB	SVM	LSTM	Transformer
Tipo de modelo	Generativo (1)	Discriminativo (2)	Discriminativo (2)	Discriminativo (2)
Tipo de aprendizaje	Machine Learning (1)	Machine Learning (1)	Deep Learning (2)	Deep Learning (2)
Tipo de procesamiento	Secuencial (1)	Secuencial (1)	Secuencial (1)	Paralelo (2)
TOTAL	**3**	**4**	**5**	**6**

Como resultado a la elección de los modelos se ha determinado implementar un modelo Transformer puesto que es el más prometedor. Ahora bien, debido al alto costo de entrenar un Transformer se previó utilizar la técnica de Transfer Learning que consiste en el uso de modelos preentrenados para posteriormente afinarlos en una sola tarea y así agilizar la creación de un modelo que cumpla las expectativas del proyecto. En esta investigación se consideraron los siguientes:

- BERT: Sus siglas corresponden a "Bidirectional Encoder Representations from Transformers". Es un modelo basado en la arquitectura Transformers, la cual se explicó a detalle en la sección de bases científicas. Un Transformer está compuesto por un codificador y decodificador, BERT sin embargo solo hace uso del codificador para aprender a representar el texto [51]. Este modelo en su versión base cuenta con 12 bloques de Transformers que contienen 12 capas de atención y 768 capas ocultas, resultando en 110 millones de parámetros [51].
- DistilBERT: Es una versión destilada de BERT que conserva el 97% de su rendimiento tan solo utilizando 66 millones de parámetros y es 60% más rápida [52]. Su tiempo de entrenamiento se reduce 4 veces al de BERT y fue entrenado con la misma cantidad de datos (16 GB) [52]. Los beneficios que ofrece es que mantiene una gran performance ahorrando tiempo de entrenamiento y recursos [53].
- RoBERTa: Por sus siglas en inglés Robustly optimized BERT approach. Esta es una propuesta más robusta y optimizada de BERT que presenta una mejora de hasta 20% en la performance. Este modelo está entrenado con 160 GB de información y se hicieron cambios en su método de entrenamiento para alcanzar una mejor propuesta [54]. Es bueno resaltar que este modelo aunque tiene mejores métricas que BERT es también más grande y costoso de entrenar [53], [55].

Así pues, para establecer los criterios con los que serán comparados estos modelos se tomó en cuenta las consideraciones iniciales, donde afirmamos que este modelo se alojará en un servidor y será parte de un servicio por lo que es más conveniente contar con un modelo ligero

y rápido que pueda atender peticiones. Por lo tanto, los criterios a comparar serán performance, tamaño, y rapidez.

TABLA VIII
CUADRO COMPARATIVO DE MODELOS PREENTRENADOS

Criterio	BERT	DistilBERT	RoBERTa
Tamaño	110 millones parámetros	66 millones parámetros	110 millones parámetros
Performance (GLUE benchmark)	79.6 %	3% menor a BERT	2-20% mejor a BERT
Tiempo entrenamiento	8 GPU x V100 x 12 días	4 veces menos que BERT	4 veces más que BERT

De esta manera se concluyó que el modelo a utilizar es el modelo DistilBERT debido a la gran performance que nos puede ofrecer sin tener que contar con grandes recursos para poder obtener buenos resultados.

OE2: Desarrollar el modelo predictivo basado en la técnica de clasificación de Machine Learning previamente identificada para reconocer emociones en textos.

Lo que se buscó como resultado de este objetivo es contar con un modelo, capaz de alcanzar un alto rendimiento en el entrenamiento, que se pueda considerar apto para poder pasar a la etapa de validación. En este objetivo se llevaron a cabo las 6 primeras fases de la metodología escogida; la validación y la última etapa de despliegue se verán en los objetivos correspondientes. Una consideración importante es que, para mantener un control de la performance, durante el entrenamiento, se utilizó la métrica "Accuracy" que se calculaba al final de cada ciclo, esto es importante debido a que nos indicó que tan afinado estaba el modelo con respecto a la data de entrenamiento y con esta confianza dar el siguiente paso a una data nueva que es la de prueba. A continuación, el desarrollo de las etapas:

Definir el problema

En esta etapa se tuvo que adoptar una visión global de lo que nuestro modelo iba a solucionar en el mundo real. Antes de definir un objetivo para el modelo, primero se tuvo en cuenta cuál será su utilidad en la práctica: lo que se busca es poder tener un servicio capaz de recibir texto, identificar una emoción en él y devolver una respuesta; este estará en un servidor en la nube atendiendo peticiones. Por lo tanto, el objetivo a nivel técnico fue desarrollar un modelo que sea capaz de clasificar texto en distintas emociones, lo que se corresponde con el problema de NLP 'clasificación de texto multiclase'. Además, debido a que estará en un servidor, por temas de recursos, será preferible que este sea ligero y rápido.

Con todo lo anteriormente definido se procedió a decidir por la elección de una métrica para problemas de clasificación multiclase para el objetivo de validación. Se tuvieron en cuenta las siguientes métricas: precisión, recall, puntaje F1 [56]. Estas métricas fueron obtenidas de una matriz de confusión previamente elaborada y se explica un poco de ellas a continuación:

- La precisión nos responde a la pregunta '¿Qué proporción de positivos predichos son realmente positivos?' la cual se tiene que preguntar por cada clase con respecto a las demás y su fórmula es:

$$precision = \frac{TP}{TP + FP}$$

- El recall toma en cuenta a los verdaderos positivos (TP) y los falsos negativos. Su fórmula es:

$$recall = \frac{TP}{TP + FN}$$

- El puntaje F1 toma en cuenta tanto a la precisión como al recall. Una de sus ventajas es que toma en cuenta a los falsos tanto negativos como positivos. Esto se ve a continuación:

$$F1\ score = \frac{2\ * precision\ * recall}{precision + recall}$$

Se decidió tomar en cuenta la medida puntaje-F1 porque es importante no solo tener en cuenta la cantidad de veces que se aciertan sino también la cantidad de veces que el modelo se equivocó para poder reducir estos errores. Además, el F1es fundamental cuando se cuenta con clases desbalanceadas en el dataset. Sin embargo, la métrica F1 se calcula por clase, es decir en nuestro caso calcularemos esta medida cinco veces; para esto existen técnicas para poder obtener un promedio de este puntaje:

- **Macro promedio F1 (Macro-F1):** Este es el promedio simple, se calcula sumando todas las métricas F1 y dividiendo el resultado entre el número de clases.
- **Peso promedio F1 (Weighted-F1):** Este es el promedio con pesos, estos se sacan de acuerdo con las proporciones de cada clase. Se calcula multiplicando cada F1 con el porcentaje que representa de la data de prueba y luego sumando estos resultados.
- **Micro promedio F1 (Micro-F1):** Esta métrica se calcula sumando todos los verdaderos positivos, falsos positivos y falsos negativos de todas las clases para con estos datos calcular un F1 global.

Finalmente se consideró utilizar la métrica Macro-F1 debido a que al no usar pesos, considera a todas las clases por igual, y nos da un promedio más realista si contamos con clases desbalanceadas [57].

Obtener la data

En esta etapa estableció que datos serán necesarios para el entrenamiento del modelo y se consiguió un conjunto de datos de calidad. Esta etapa, al igual que la anterior, es clave en todo el proceso debido a que de esta depende el entendimiento que tenga nuestro modelo. Por ejemplo, si la data está mal etiquetada nuestro modelo no predecirá correctamente. Así mismo su calidad también depende del tamaño, suciedad o errores en el texto y un desbalance entre sus clases.

Primero se procedió a determinar qué datos se necesitan para un entrenamiento supervisado. En el entrenamiento supervisado es necesario poseer una entrada y una salida para que nuestro modelo pueda aprender a producirla por sí mismo. Como entrada se vio correspondiente que se ingrese el texto y como salida la emoción que se contiene en él.

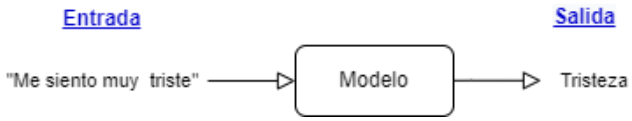


Fig. 2 Entrada y salida del modelo

Se procedió a buscar conjuntos de datos con estas características y se obtuvieron los siguientes:

- Dataset “GoEmotions”: Este conjunto de post de Reddit es resultado de la investigación “GoEmotions: A Dataset of Fine-Grained Emotions” [58] que se ha hecho de dominio público. Este dataset cuenta con 27 emociones más una neutral y 58.000 registros. En la investigación advierten que el dataset contiene biases porque está muy desbalanceado. Dentro de las pruebas que realizaron sus métricas de puntaje-F1 no superaron el 70%.
- Dataset “CARER”: Es un conjunto de tweets resultado de una investigación [59] “CARER: Contextualized Affect Representations for Emotion Recognition” que puede ser utilizado para fines educativos y de investigación. Este dataset cuenta con 6 emociones básicas: enojo, miedo, amor, tristeza y sorpresa, además cuenta con 20.000 registros. Dentro de las pruebas que realizaron con distintos modelos sus puntaje-F1 superaron al 70%.

Posteriormente se evaluaron mediante criterios para establecer cuál era el más conveniente para esta investigación. Los criterios establecidos fueron:

- Tamaño del dataset: Importante debido a que mientras más data tenemos podemos entrenar más a nuestro modelo.

TABLA IX
VALORACIÓN DE TAMAÑO DEL DATASET

Tamaño	Valoración
10000+	4
5000-10000	3
1000-5000	2
0 – 1000	1

- Limpieza: Se valorará que los datos estén limpios eso agilizará y facilitará el preprocesamiento de la data. Se toman en consideración los signos, las comas, emojis, palabras mal escritas, etc.

TABLA X
VALORACIÓN DE LIMPIEZA DEL DATASET

Limpieza	Valoración
Muy limpio	4
Regular limpio	3
Poco limpio	2
Sin limpiar	1

- Desbalance: Importante para no causar que nuestro modelo presente sesgos por la clase predominante del dataset. Se tomó en cuenta la diferencia que existe entre la clase predominante y la clase menor.

TABLA XI
VALORACIÓN DE DESBALANCE DEL DATASET

Diferencia entre la clase mayoritaria y minoritaria	Desbalance	Valoración
0-1000	No desbalanceada	4
1000-5000	Poco desbalance	3
5000-10000	Regular desbalance	2
10000+	Muy desbalanceada	1

- Cantidad de emociones: Se valorará que el modelo presente mínimamente las emociones básicas.

TABLA XII
VALORACIÓN DE CANTIDAD DE EMOCIONES DEL DATASET

Cantidad de emociones	Valoración
10+	4
7-10	3
4-7	2
0-4	1

Teniendo en cuenta los criterios se procedió a valorar y comparar los conjuntos de datos en la *TABLA XIII*.

TABLA XIII
COMPARACIÓN ENTRE DATASETS

Criterios	GoEmotions	CARER
Tamaño	58.000 (4)	20.000 (4)
Limpieza	Sin limpiar (1)	Muy limpio (4)
Balance	Muy desbalanceado (1)	Poco desbalanceado (3)
Cantidad de emociones	27 (4)	6 (2)
TOTAL	**10**	**13**

Con esta comparación, y teniendo en cuenta que en el artículo de GoEmotions se reconoce que su dataset posee sesgos debido a su desbalance, es que se concluye en esta iteración que la data a utilizar será la de CARER debido a que la información se encuentra mucho más limpia y balanceada, lo cual se puede traducir en mejores resultados para el modelo.

Después se procedió a descargar el dataset de HuggingFace con el uso de la librería "datasets" y se revisó ligeramente su estructura de acuerdo con la metodología. A primera impresión notamos que la data ya está separada, y se decidió respetar y mantener esta división para trabajar en adelante. También se comprobó que consiste satisfactoriamente con lo requerido: una columna con el texto y otra con la etiqueta del sentimiento en formato de número entero.

Descubrir y visualizar los datos para obtener información

En esta etapa se exploró la data con el objetivo de conocerla a fondo. Solo se tomará consideración la data de entrenamiento como indica la metodología debido a que se quiere evitar el sesgo de data snooping. Primero revisamos las características de la data.

```
{
  'label': ClassLabel(
  num_classes=6,
  names=['sadness', 'joy', 'love', 'anger', 'fear', 'surprise'],
  id=None),
  'text': Value(
    dtype='string',
    id=None)
}
```

Fig. 3 Características del conjunto de datos

Se observa que se cuenta efectivamente con 6 clases que están ordenadas en un array. Para una mejor exploración se creó un Dataframe con la data de entrenamiento y se imprimieron los datos.

	text	label
0	i didnt feel humiliated	0
1	i can go from feeling so hopeless to so damned...	0
2	im grabbing a minute to post i feel greedy wrong	3
3	i am ever feeling nostalgic about the fireplac...	2
4	i am feeling grouchy	3

Fig. 4 Vista general del conjunto de datos

Así mismo se revisó los tipos de datos de cada columna. La columna "label" tiene un formato entero, el cual es adecuado para que pueda ser procesada por la red neuronal, sin embargo, para la exploración se consideró usar el nombre de la emoción a la que corresponde. La columna "text" cuenta con el tipo de dato "object" que es indicado para la exploración. Se creó una nueva columna con los nombres de las emociones para que los gráficos sean más entendibles.

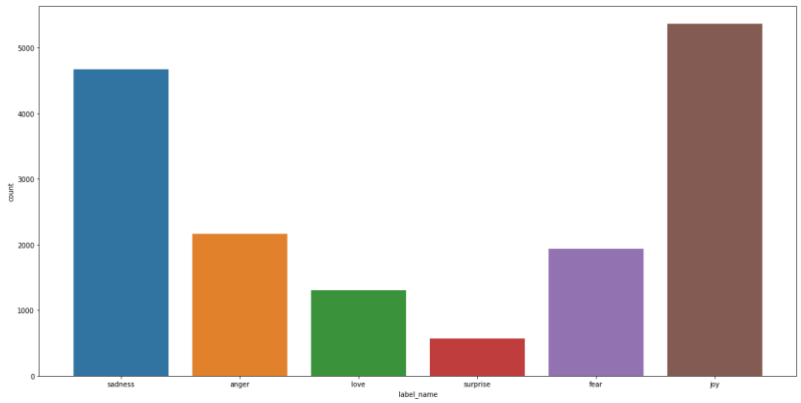


Fig. 5 Distribución de las emociones en el conjunto de datos

La emoción que tiene menos ejemplos es la emoción de sorpresa seguida por la emoción de amor, y la que tiene más es alegría y tristeza. También para extraer las palabras más comunes por cada emoción se creó una variable que contenga todo el texto de los registros de una emoción en particular y se utilizará la librería Wordcloud para mostrar las palabras más comunes en una nube. Este proceso se realizará para todas las emociones.

```
from wordcloud import WordCloud

emotion_list = ds[ds['label'] == 0]['text'].tolist()
emotion_docx = ' '.join(emotion_list)

def plot_wordcloud(docx):
  mywordcloud = WordCloud().generate(docx)
  plt.figure(figsize=(20,10))
  plt.imshow(mywordcloud,interpolation='bilinear')
  plt.axis('off')
  plt.show()

plot_wordcloud(emotion_docx)
```

Fig. 6 Palabras más comunes por cada emoción

Fig. 7 Wordcloud de felicidad

Después de la exploración se revisó la calidad de los datos. Es necesario tener certeza de que no existen datos nulos tanto en la columna de texto como en las etiquetas.

```
1  ds.isnull().sum()
```

```
text          0
label         0
label_name    0
dtype: int64
```

Fig. 8 Verificando la calidad de los datos

Preparación de la data

En esta etapa se limpió la data, específicamente la columna de texto debido a que es común encontrar textos con puntuaciones, signos, números, errores ortográficos, etc. y posteriormente se la tokenizó para que pueda ingresar en la red neuronal. En la exploración de datos se ha observado que el texto de los ejemplos luce limpio sin embargo se volverá a ejecutar una limpieza para garantizar que todos los registros cumplan con las condiciones de limpieza necesarias. Para limpiarla se vio necesario excluir todos los caracteres que no sean letras (a-z), convertir el texto a minúsculas y extraer las palabras vacías, mejor conocidas como "stopwords". Este proceso se realizó para la data de entrenamiento, validación y test.

```
import re
import nltk
from nltk.corpus import stopowords

nltk.download('stopwords')
sw = stopwords.words('english')

def text_clean(data, column):
  corpus = []
  for text in data[column]:
    text = re.sub("[^a-zA-Z]", " ", text)
    text = text.lower()
    text = text.split()
    text = [word for word in text if word not in sw]
    text = " ".join(text)
    corpus.append(text)
  return corpus
```

Fig. 9 Función para limpiar los datos

Después de la limpieza se procedió a tokenizar al texto en embeddings. En este caso debido a que usaremos como base el modelo Distilbert y este cuenta con su propio tokenizador, se hará uso de este. El tokenizador, tanto como el modelo se pueden encontrar en la librería Transformers.

```
from transformers import DistilBertTokenizerFast

def tokenize_text(ds):
    return tokenizer(ds["text"], truncation=True)

columns = dataset["train"].column_names
columns.remove("label")
encoded_dataset = dataset.map(tokenize_text, batched=True,
remove_columns=columns)
encoded_dataset
```

Fig. 10 Función para tokenizar los datos

Elegir un modelo

En esta etapa lo que se busca es elegir el tipo de técnica de ML que más se corresponda con el tipo de tarea que debemos realizar, y hacer que logre un desempeño básico para posteriormente afinarlo. Parte del desarrollo de esta etapa se corresponde con el primer objetivo, en donde se buscó seleccionar a la técnica más adecuada, es así que para no redundar en la misma información se tomaron las conclusiones del primer objetivo y se prosiguió con la obtención del modelo base DistilBERT el cual se obtuvo de la librería “transformers”.

```
from transformers import DistilBertForSequenceClassification
from datasets import load_metric
from transformers import TrainingArguments
from transformers import Trainer
from datetime import datetime
import numpy as np
import json

num_labels = 6
model_name = 'distilbert-base-uncased'
model = DistilBertForSequenceClassification.from_pretrained(model_name,
num_labels=num_labels)
```

Fig. 11 Importación de DistilBERT

Este modelo cuenta con 6 capaz de codificadores de Transformer, con una última capa linear de decodificación y además con 12 cabezas de atención. Su configuración inicial es la que se muestra en la *TABLA XIV* y es la que viene por defecto en el modelo. Se consideró mantener inicialmente estos parámetros debido a que son producto de una previa investigación en la que se concluyó que estos eran los más adecuados [52].

TABLA XIV
CONFIGURACIÓN INICIAL DEL MODELO

Parámetros	**Valor**
vocab_size	30522

max_position_embeddings	512
n_layers	6
n_heads	12
Hidden_dim	3072
dropout	0.1
activation	gelu

Los parámetros de configuración se describen a continuación:

- Vocab_size: El número máximo de tokens únicos que pueden ser ingresados en el modelo.
- Max_position_embeddings: La longitud máxima de la oración con la que se puede utilizar este modelo.
- N_layers: Número de capas ocultas en el codificador del Transformer.
- Hidden_dim: El número de capas intermedias que tiene la red neuronal "feed-forward" de cada codificador.
- N_heads: Número de cabezas de atención por cada capa de atención en el Transformer.
- Dropout: Indica la probabilidad total de dropout para todas las capas conectadas. Asignar un dropout es un método utilizado para evitar que las redes neuronales se sobreajusten. Consiste en ignorar los resultados de unas neuronas en un ciclo de entrenamiento de un cierto conjunto de unidades que se elige al azar.
- Activation: Se indica la función de activación que se va a utilizar. En este caso "gelu" hace referencia a "Gaussian Error Linear Unit". Esta función es la utilizada en todos los modelos BERT por ser un poco más eficiente que Relu [60].

Después de la obtención del modelo base se elaboró la función de evaluación que se ejecutará después de cada ciclo de entrenamiento. Las métricas por utilizar se importaron con el módulo de "load_metric" desde la librería "datasets". De igual manera las métricas fueron guardadas para su futura evaluación.

```
def compute_metrics(eval_preds):
    accuracy = load_metric("accuracy")
    f1 = load_metric("f1")
    recall = load_metric("recall")

    logits, labels = eval_preds
    predictions = np.argmax(logits, axis=-1)

    metrics = {
        "accuracy": accuracy.compute(predictions=predictions, references=labels)["accuracy"],
        "f1": f1.compute(predictions=predictions, references=labels, average='micro')["f1"],
        "recall": recall.compute(predictions=predictions, references=labels, average='micro')["recall"]
    }

    with open(f"metrics/metrics-{str(datetime.now()).replace(':', '.')}.json", 'w') as fp:
        json.dump(metrics, fp)

    return metrics
```

Fig. 12 Función para calcular las métricas

Posteriormente se importó la clase “Trainer” y “TrainingArguments”. En Trainer se agrupan los elementos involucrados en el entrenamiento como son el tokenizador, el modelo a utilizar, la función que calculará las métricas, los parámetros del entrenamiento, el dataset de entrenamiento y el de prueba. En TrainingArguments se encuentran los parámetros propios del entrenamiento y que son los que se modifican para alcanzar métricas más elevadas.

Inicialmente en la clase TrainingArguments solo se especificaron los parámetros que no consideramos modificar posteriormente, los cuales son:

- **Evaluation_strategy**: Con este parámetro indicamos en que etapa dentro del entrenamiento se va a realizar la evaluación. En este caso deseamos que se haga una evaluación después de completar un ciclo de entrenamiento.
- **Output_dir**: Especifica el directorio en que queremos que se guarden los resultados del entrenamiento.
- **Load_best_model_at_end**: Especificamos si queremos que al finalizar el entrenamiento nos cargue el modelo con mejor métrica.
- **Metric_for_best_model**: Especificamos que métrica usaremos para la evaluación del modelo que se hace al final de cada ciclo.
- **Save_strategy**: Especificamos la estrategia con la que guardaremos los avances del entrenamiento del modelo. En este caso se decidió guardar el modelo después de cada ciclo.

Afinar el modelo

En esta etapa el objetivo es afinar nuestro modelo base para que aprenda a hacer la tarea de clasificación multiclase y lograr el performance esperado. Se realizaron una serie de cambios en los siguientes parámetros de entrenamiento:

- batch_size: Es la cantidad de muestras que se usan para el entrenamiento por lotes.
- num_train_epochs: Es el número de ciclos que se va a entrenar a un modelo.
- weight_decay: Es el decaimiento que le aplicaremos a los pesos del modelo. Este parámetro nos ayuda a que nuestro modelo pueda generalizar y no se sobre ajuste.
- learning_rate: Es la taza de aprendizaje de nuestro modelo que indica que tanto cambiaremos el modelo con el objetivo de minimizar el error.

```python
from transformers import TrainingArguments
from transformers import Trainer

batch_size = 16
num_train_epochs=40


train_dataset = encoded_dataset["train"].shuffle(seed=42)

training_args = TrainingArguments(
    output_dir="results",
    num_train_epochs=num_train_epochs,
    learning_rate=2e-5,
    per_device_train_batch_size=batch_size,
    per_device_eval_batch_size=batch_size,
    load_best_model_at_end=True,
    metric_for_best_model="f1",
    weight_decay=0.01,
    evaluation_strategy="epoch",
    save_strategy="epoch",
    save_total_limit=1,
)

trainer = Trainer(
    model=model,
    args=training_args,
    compute_metrics=compute_metrics,
    train_dataset=train_dataset,
    eval_dataset=encoded_dataset["validation"],
    tokenizer=tokenizer
)
```

Fig. 13 Clase Trainer y TrainingArguments

Para asignar los parámetros primero se consideraron los argumentos por defecto de la clase Trainer y luego se fueron modificando de acuerdo con los resultados obtenidos. Inicialmente obtenemos resultados satisfactorios con un accuracy de 94.15% en el último ciclo de entrenamiento. Con intención de mejorar esta métrica se aumentó el tamaño del batch y los ciclos, además se redujo la ratio de aprendizaje. Los resultados los observamos en la *TABLA XV*.

TABLA XV
COMPARACIÓN DE VALORES DE PARÁMETROS DEL ENTRENAMIENTO

Pruebas	Batch Size	Epochs	Learning Rate	Weight Decay	Accuracy	Best epoch
1	8	3	5e-5	0	0.9415	3
2	16	40	2e-5	0.01	0.944	15
3	32	40	2e-5	0.01	0.943	11
4	16	20	5e-5	0.01	0.941	14

En el cuadro notamos que los resultados son muy parecidos y que el mejor modelo es el que cuenta con un batch de 16 y una ratio de aprendizaje de 2e-5 debido a que es con el que se consiguió una mejor exactitud. Otro dato importante es que se evidenció que el modelo no necesita muchos ciclos de entrenamiento debido a que sus mejores resultados los logró obtener antes de llegar a la mitad de los ciclos programados y de ahí en adelante su performance fue decayendo. Concluimos así que el modelo a utilizar será el modelo que obtuvimos de la prueba 2.

OE3: Validar el modelo de clasificación para verificar su precisión en el reconocimiento de emociones en textos.

La finalidad de este objetivo es brindarle al modelo desarrollado una nueva data que no conozca previamente y así comprobar su desempeño frente a nuevos datos. Para medir su performance se usó la métrica Macro-f1 que es un promedio simple de entre las métricas F1 obtenidas por cada clase.

$$Macro\ F1 = \frac{1}{Q}\sum_{k=1}^{Q}\frac{2 \times Pk \times Rk}{Pk + Rk}$$

Donde:

- Q = Cantidad de categorías de clasificación
- K = Representa a una categoría, en este caso a una emoción
- Pk = Precisión obtenido en la categoría k
- Rk = Recall obtenido en la categoría k

Para calcular Macro-F1 inicialmente importamos el modelo entrenado que se encuentra en el directorio 'custom-model'. Con el modelo que se obtuvo de la fase entrenamiento se predijo una emoción para cada entrada de la data test y cada predicción obtenida se comparó con el resultado verdadero.

```
import torch
import torch.nn.functional as F
from sklearn import metrics
import matplotlib.pyplot as plt
import numpy as np

y_preds = []
y_trues = []
for index,test_text in enumerate(dataset['test']):
    tokenized_test_text = tokenizer(test_text['text'],truncation=True,
padding=True, return_tensors="pt")
    labels = torch.tensor([1]).unsqueeze(0)
    outputs = model(**tokenized_test_text,labels=labels )
    logits = outputs.logits
    prediction = F.softmax(logits, dim=1)
    y_pred = torch.argmax(prediction).numpy()
    y_true = test_text['label']
    y_preds.append(y_pred)
    y_trues.append(y_true)
```

Fig. 14 Código para la validación del modelo entrenado

Como se puede observar en *Fig. 14*, después de importar el modelo entrenado, contamos con dos arreglos importantes:

- Y_preds: Aquí guardaremos las predicciones que hizo nuestro modelo.
- Y_trues: Aquí se encuentran los resultados correctos que vienen en el dataset.

Con estos datos ya se puede realizar la matriz de confusión y representarla a través de un mapa de calor que se muestra en *Fig. 15*. En esta figura podemos observar que contamos con un eje Y en donde están las etiquetas verdaderas, en el eje X las predichas y sus coincidencias numéricas. Este cuadro responde a la pregunta '¿Cuántas etiquetas predichas de predijeron correctamente por emoción?'.

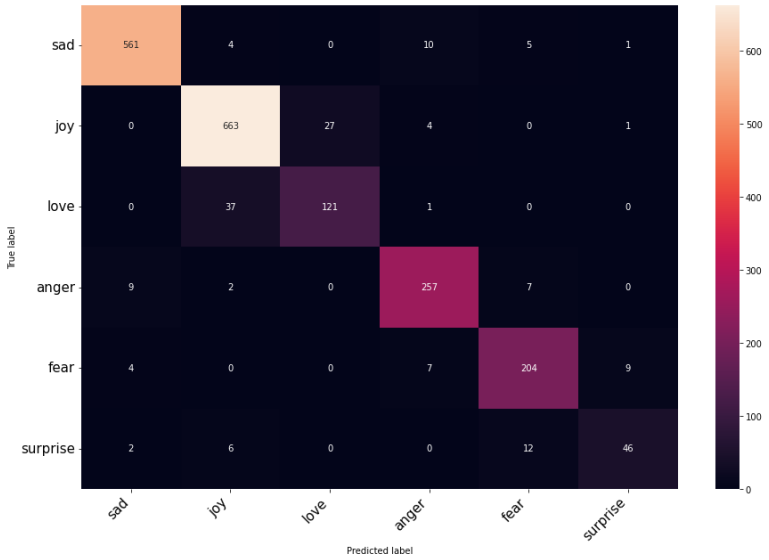


Fig. 15 Matriz de confusión

Ahora que ya tenemos una vista gráfica proseguiremos con el reporte de clasificación (*Fig. 16*) que nos dará las métricas para determinar si cumplimos o no con el objetivo. En este caso utilizamos el módulo "metrics" de la librería "Sklearn".

	precision	recall	f1-score	support
0	0.97	0.97	0.97	581
1	0.93	0.95	0.94	695
2	0.82	0.76	0.79	159
3	0.92	0.93	0.93	275
4	0.89	0.91	0.90	224
5	0.81	0.70	0.75	66
accuracy			0.93	2000
macro avg	0.89	0.87	0.88	2000
weighted avg	0.93	0.93	0.93	2000

Fig. 16 Reporte de clasificación del modelo

Finalmente podemos observar que los puntajes F1 para cada clase van desde 75% hasta el 97%, esto probablemente debido al desbalance de las clases que se observó durante la exploración de datos. Pese a esto el macro-F1 es 88% lo cual nos indica que se ha superado satisfactoriamente el umbral establecido y cumplimos así el objetivo establecido.

OE4: Desplegar el modelo de clasificación validado en una aplicación móvil como apoyo en el reconocimiento de emociones en textos de estudiantes universitarios.

Lo que se buscó en este objetivo es desplegar el modelo en un servicio web y construir una aplicación móvil que se comunique con este para que los usuarios lo puedan usar. Aquí se culminó el despliegue del modelo como un servicio y para la construcción de la aplicación se utilizó la metodología SCRUM.

Para desarrollar el servicio se utilizó un Framework de Python llamado Flask conocido por su simplicidad al momento de crear servicios API Rest. La app se alojó en un droplet de Digital Ocean con la intención de que esté siempre disponible. El servicio está activo con el nombre 'emotion-analysis-api.service' como se observa en *Fig. 17*.

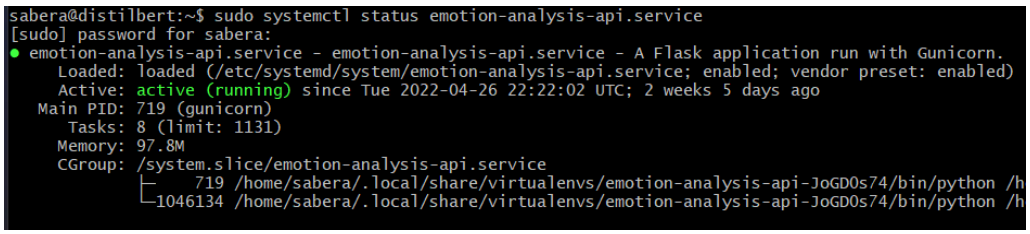


Fig. 17 Estado del servicio desplegado

Con el servicio activo se procedió a verificar su correcto funcionamiento con pruebas desde la aplicación Postman. Los criterios con los que se evaluó se encuentran en *TABLA XVI*.

TABLA XVI
CRITERIOS DE FUNCIONALIDAD

Criterio	Descripción	Valores
El código HTTP de la respuesta debe ser 200	Recibir el código 200 significa que la petición se realizó exitosamente	0: No cumplido 1: Cumplido
La respuesta debe tener la estructura JSON esperada	El JSON esperado debe tener como key a los nombres de las emociones y en sus valores la probabilidad calculada	0: No cumplido 1: Cumplido
El tiempo de respuesta debe ser menor a 1s	El tiempo de respuesta debe ser menor a 1s para que no se pierda la atención del usuario	0: No cumplido 1: Cumplido

Para que se considere que se ha logrado el objetivo se deberán cumplir satisfactoriamente todos los criterios de la *TABLA XVI*, es decir se deberá obtener un puntaje total de 3. Después de realizar las pruebas como se evidencia en *Fig. 18* se procedió a calificar en *TABLA XVII*.

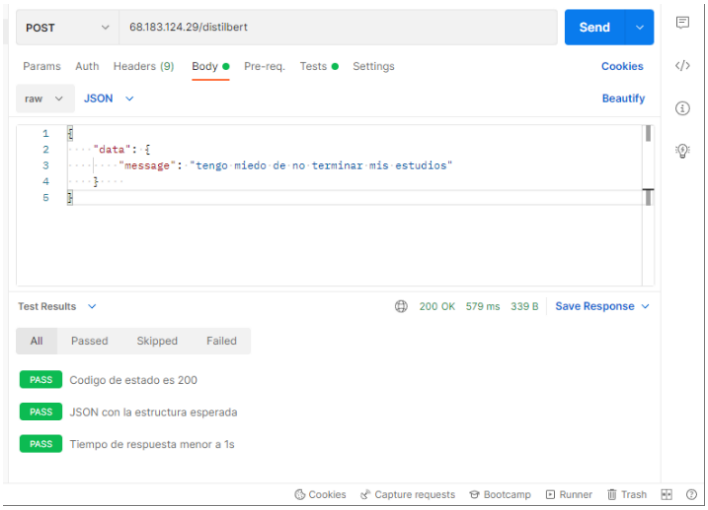


Fig. 18 Test realizados al servicio desplegado

De acuerdo a las consideraciones anteriores se asignaron los puntajes y se obtuvo un total de 3 puntos lo cual indicó que se cumplió el objetivo satisfactoriamente.

TABLA XVII
CUMPLIMIENTO DE CRITERIOS DE FUNCIONALIDAD

Criterio	Cumplimiento	Calificación	Total
El código HTTP de la respuesta debe ser 200	Cumplido	1	
La respuesta debe tener la estructura JSON esperada	Cumplido	1	3 de 3
El tiempo de respuesta debe ser menor a 1s	Cumplido	1	

Después de demostrar que el servicio funciona correctamente, se procedió a desarrollar la aplicación móvil con la metodología planteada. Las evidencias del desarrollo se encuentran en el *Anexo 4*. Adicionalmente, como complemento, al logro de este objetivo, se realizó una validación funcional por parte de un profesional de la carrera que se puede observar en el *Anexo 1*.

Discusión

Esta investigación se planteó con el objetivo de apoyar a los estudiantes en el control y monitoreo de sus emociones debido a que estos están expuestos a distintos factores sociales, económicos y académicos que llegan a mermar su salud mental; además regularmente los universitarios tienen a no prestar atención de sus estados emocionales y evitan afrontar este tipo de problemas. Por lo antes mencionado es que se buscó crear una aplicación que esté al alcance de ellos de manera inmediata y que así pudieran, a través de unas simples preguntas, llevar un monitoreo de sus emociones. Consecuentemente a este planteamiento se buscó entrenar un modelo de entendimiento del lenguaje que pudiera detectar emociones en las respuestas que nos sean brindadas por los escolares y con los resultados poder brindarles estadísticas para su fácil monitoreo. A continuación, se discutirán los resultados obtenidos y se evaluarán en referencia a los antecedentes.

Con respecto al primer objetivo se buscó encontrar una técnica adecuada para esta investigación. Y como resultado se obtuvo que la más acertada era las redes neuronales de tipo Transformer. De entre los tipos de mecanismos a comparar se tomó en consideración a los utilizados en los antecedentes Naive Bayes [14], Support Vector Machine [19], [20], LSTM [18], Bi-LSTM [15] y Transformers [16], [61]. Comparaciones entre estas técnicas podemos encontrar en investigaciones como [49] en donde se comparan modelos como Naive Bayes, Support Vector Machine, entre otros clásicos de ML, junto con Deep Neural Networks (DNN); esta comparativa se da mediante 7 métricas, siendo F1-score de nuestro interés, en exactamente 8 conjuntos de datos diferentes; resultando que son las DNN las que obtienen mejores resultados, seguidas de SVM. Además en [62] se comparó para la tarea de clasificación de texto a los modelos SVM, LSTM, CNN, NB con modelos Transformer como Bert y XLM-RoBERTa

utilizando las métricas de Accuracy y F1; y teniendo como resultado que los Transformers superan a los otros modelos. Adicionalmente esto se demuestra también el artículo [51] en donde es BERT quien sobrepasa a técnicas como Bi-LSTM en pruebas de entendimiento del lenguaje. Por todo lo mencionado se infiere que la utilización de modelos de tipo Transformer es una de las mejores de opciones.

Con respecto al segundo objetivo, lo que se busca es el desarrollo del modelo clasificador. En este caso su desarrollo durante el entrenamiento se verá reflejado en la métrica de exactitud, mejor conocida como "accuracy", que es la que indica cuantas predicciones son correctas con respecto al total. Es necesario aclarar que esta métrica no toma en cuenta si contamos con desbalance en las clases del conjunto de datos por lo que no debe ser considerada como una métrica para validación del modelo sino simplemente como un indicador de que contamos con un modelo lo suficiente ajustado a los datos y que ya contando con un valor alto puede pasar a la fase de validación. Este también es el motivo por el cual se propuso un umbral tan alto ya que al ser más propenso a verse sesgado por el desbalance es fácil alcanzar valores altos.

Con respecto al tercer objetivo, la finalidad fue validar el modelo entrenado con una métrica confiable. El indicador elegido fue macro-F1, este como promedio simple a la métrica F1 que será calculada para cada emoción. El cálculo de F1 se considera más confiable cuando estamos tratando con clases desbalanceadas, así también se afirma en [56]. En [14], [17], [18] sin embargo se utiliza la métrica "accuracy" para evaluar su modelo dando como resultado 0.473, 0.70 y 0.879 correspondientemente; aquí nuestro modelo supera a los tres ya que obtuvo un 0.944 en esta métrica. En [15] se obtiene 0.706 en F1 y 0.714 en la precisión para la clasificación de 6 emociones y un estado neutral. En [16] se obtiene un macro-F1 de 0.481 para 11 emociones con el uso de Transformers. En [19] se obtuvo 0.750 de F1 con la utilización de SVM en un conjunto de datos desbalanceado para una clasificación binaria. En [62] se utilizó el Transformer XLMRoBERTa y se obtuvo 0.706 en la métrica F1 para clasificar comentarios de Youtube en 5 emociones. Teniendo esto en consideración y como base se pudo establecer el umbral de 0.80, el cual se superó satisfactoriamente con 0.88. También se cree necesario resaltar que este resultado también depende de la calidad y cantidad de los datos con los que entrenamos al modelo, debido a que, si no brindamos suficientes datos de entrenamiento a un modelo de Redes Neuronales Profundas como son los Transformers o LSTM, estos no serán capaces de alcanzar su potencial. En caso se cuente con data reducida sería mejor optar por un modelo clásico de ML [43], [48], [63].

Con respecto al cuarto objetivo, se planteó la creación del servicio y de las plataformas que serán utilizadas por los usuarios que quieran hacer uso del modelo desarrollado. Se trabajaron

dos servicios en la nube, una aplicación móvil y una plataforma de administración. Para los servicios en la nube se esperó que el tiempo de respuesta sea menor a 1s de acuerdo con Jakob Nielsen en [64] en donde establece 3 tiempos, siendo 1s el tiempo intermedio; los dos servicios cumplen con responder en menos de un segundo lo demostrado previamente. Adicionalmente se hicieron pruebas para verificar que el servicio responda a las solicitudes correctamente. También es bueno mencionar que se realizaron pruebas de carga a la API que atenderá las consultas de los usuarios llegando a poder responder hasta 10.000 solicitudes simultáneas. Para la validación de las aplicaciones se realizaron pruebas de caja negra y blanca. Otra consideración es que la mayoría de los antecedentes de esta investigación no llevaron a cabo un despliegue de su modelo; y las que sí cuentan con una aplicación no brindaron más detalle de esta.

Conclusiones

Se describirán lo que se logró y concluyó por cada objetivo establecido del proyecto.

1. Se identificó a la técnica de redes neuronales de tipo Transformer, específicamente al modelo preentrenado DistilBERT. La evaluación se realizó estableciendo criterios que garantizaran que se adecue a los objetivos del proyecto.
2. Se desarrolló un modelo predictivo de acuerdo con la técnica previamente identificada. Para su correcto desarrollo se siguió una metodología de Machine Learning y se entrenó al modelo hasta alcanzar o superar el 90% en la métrica de entrenamiento “accuracy”, la cual se sobrepasó con el 94.4%.
3. Se validó el modelo de clasificación de textos entrenado utilizando la métrica macro-F1 en la cual se estableció el umbral de 80% y se superó alcanzando el 88%.
4. Se desplegó el modelo validado en un servicio web y se desarrolló una aplicación móvil para que los usuarios puedan hacer uso del servicio. Se validó un correcto funcionamiento mediante pruebas de carga y de caja negra y blanca.

Recomendaciones

Al igual que las conclusiones, las recomendaciones deben ser claras, breves y concisas sin profundizar en mayores detalles.

1. Se recomienda evaluar los nuevos conjuntos de datos con mayor diversidad de emociones para poder abarcar la gran gama que se presentan en los estudiantes.
2. Analizar las nuevas técnicas de clasificación de Machine Learning que ofrezcan buena performance en entendimiento del lenguaje natural con el fin de comparar los resultados obtenidos y alcanzar mejoras.

3. Considerar los nuevos métodos de selección de datos para reducir el consumo de recursos utilizados en el entrenamiento de los modelos de Machine Learning.

Referencias

[1] E. B. Andrade y D. Ariely, "The enduring impact of transient emotions on decision making", *Organ. Behav. Hum. Decis. Process.*, vol. 109, núm. 1, pp. 1–8, may 2009, doi: 10.1016/j.obhdp.2009.02.003.

[2] D. Jekauc y R. Brand, "Editorial: How do Emotions and Feelings Regulate Physical Activity?", *Front. Psychol.*, vol. 8, 2017, Consultado: el 9 de abril de 2022. [En línea]. Disponible en: https://www.frontiersin.org/article/10.3389/fpsyg.2017.01145

[3] C. T. Gloria y M. A. Steinhardt, "Relationships Among Positive Emotions, Coping, Resilience and Mental Health: Positive Emotions, Resilience and Health", *Stress Health*, vol. 32, núm. 2, pp. 145–156, abr. 2016, doi: 10.1002/smi.2589.

[4] Personal de Mayo Clinic, "Enfermedad mental - Síntomas y causas - Mayo Clinic". https://www.mayoclinic.org/es-es/diseases-conditions/mental-illness/symptoms-causes/syc-20374968 (consultado el 10 de septiembre de 2021).

[5] Organización Mundial de la Salud (OMS), *Prevención de los trastornos mentales: intervenciones efectivas y opciones de políticas, informe compendiado.* Ginebra.: OMS., 2004.

[6] World Health Organization, *Doing what matters in times of stress: an illustrated guide*. Geneva: World Health Organization, 2020. [En línea]. Disponible en: https://apps.who.int/iris/handle/10665/331901

[7] American College Health Association, "American College Health Association-National College Health Assessment III: Undergraduate Student Executive Summary Fall 2021", NCHA, USA, 2021. [En línea]. Disponible en: https://www.acha.org/documents/ncha/NCHA-III_FALL_2021_UNDERGRADUATE_REFERENCE_GROUP_EXECUTIVE_SUMMARY.pdf

[8] C. Chau y P. Vilela, "Determinantes de la salud mental en estudiantes universitarios de Lima y Huánuco", *Rev. Psicol.*, vol. 35, núm. 2, pp. 387–422, jul. 2017, doi: 10.18800/psico.201702.001.

[9] E. P. de S. E. S. A. E. PERÚ, "Salud mental: Minedu y Minsa trabajan con 21 universidades públicas". https://andina.pe/agencia/noticia-salud-mental-minedu-y-minsa-trabajan-21-universidades-publicas-769880.aspx (consultado el 10 de septiembre de 2021).

[10] M. Cassaretto Bardales, C. Chau Pérez Aranibar, M. del C. Espinoza Reyes, F. Otiniano Campos, L. Rodríguez Cuadros, y M. Rubina Espinosa, *SALUD MENTAL EN UNIVERSITARIOS DEL CONSORCIO DE UNIVERSIDADES DURANTE LA PANDEMIA*, 1a ed. Lima - Perú: CONSORCIO DE UNIVERSIDADES, 2021. [En línea]. Disponible en: https://www.consorcio.edu.pe/wp-content/uploads/2021/10/SALUD-MENTAL-CONSORCIO-DE-UNIVERSIDADES.pdf

[11] World Health Organization, "Salud mental del adolescente". https://www.who.int/es/news-room/fact-sheets/detail/adolescent-mental-health (consultado el 10 de septiembre de 2021).

[12] S. M. Benel Ramírez, "Encuesta para medir la salud mental de los estudiantes universitarios", Lambayeque, 2020.

[13] M. J. Bahamón Muñetón *et al.*, "Estilos de afrontamiento como predictores del riesgo suicida en estudiantes adolescentes", *Psicol. Desde El Caribe*, vol. 36, núm. 1, pp. 120–132, abr. 2019, doi: 10.14482/psdc.36.1.616.8.
[14] K. Sailunaz y R. Alhajj, "Emotion and sentiment analysis from Twitter text", *J. Comput. Sci.*, vol. 36, p. 101003, sep. 2019, doi: 10.1016/j.jocs.2019.05.009.
[15] G. Cao *et al.*, "Analysis of social media data for public emotion on the Wuhan lockdown event during the COVID-19 pandemic", *Comput. Methods Programs Biomed.*, vol. 212, p. 106468, nov. 2021, doi: 10.1016/j.cmpb.2021.106468.
[16] G. W, D.-A. J, C. Y, D. I, L.-S. E, y S.-L. C, "Emotion detection for social robots based on nlp transformers and an emotion ontology", 2021, doi: 10.3390/s21041322.
[17] S. A. Morzán Fuentes, "Detección de estados de ánimo mediante sentiment analysis en hispanohablantes", *Repos. Inst. - Ulima*, 2019, doi: 10.26439/ulima.tesis/11318.
[18] G. A. Reyes-Paredes, "Análisis de sentimientos de noticias escritas usando un modelo basado en la red neuronal long short-term memory para determinar si las noticias positivas mejoran el estado de ánimo de las personas", *Repos. Inst. - Ulima*, 2020, Consultado: el 13 de septiembre de 2021. [En línea]. Disponible en: https://repositorio.ulima.edu.pe/handle/20.500.12724/11171
[19] X. M. Cuzcano Chavez, "A comparison of classification models to detect cyberbullying in the peruvian spanish language on Twitter", *Repos. Inst. - Ulima*, 2020, Consultado: el 18 de abril de 2022. [En línea]. Disponible en: https://repositorio.ulima.edu.pe/handle/20.500.12724/12718
[20] L. Y. Segura Vásquez, "EVALUACIÓN DE ALGORITMOS DE CLASIFICACIÓN PARA EL MINADO DE OPINIÓN EN TWITTER", *Repos. Inst. - USS*, 2019, Consultado: el 9 de abril de 2022. [En línea]. Disponible en: http://repositorio.uss.edu.pe//handle/20.500.12802/6253
[21] S. Raj, *Building Chatbots with Python: Using Natural Language Processing and Machine Learning*. 2018.
[22] E. Kumar, *Natural Language Processing*. I. K. International Pvt Ltd, 2011.
[23] N. Indurkhya y F. J. Damerau, *Handbook of Natural Language Processing*. CRC Press, 2010.
[24] B. K. Mishra y R. Kumar, *Natural Language Processing in Artificial Intelligence*. Apple Academic Press, 2020.
[25] S. Vajjala, B. Majumder, A. Gupta, y H. Surana, *Practical Natural Language Processing: A Comprehensive Guide to Building Real-world Nlp Systems*. Oreilly & Associates Inc, 2020.
[26] S. Chandran, "Introduction to Text Representations for Language Processing — Part 1", *Medium*, el 16 de noviembre de 2021. https://towardsdatascience.com/introduction-to-text-representations-for-language-processing-part-1-dc6e8068b8a4 (consultado el 17 de abril de 2022).
[27] S. Mokhtarani, "Embeddings in Machine Learning: Everything You Need to Know | FeatureForm", el 26 de agosto de 2021. https://www.featureform.com/post/the-definitive-guide-to-embeddings (consultado el 17 de abril de 2022).
[28] J. Alammar, "The Illustrated Word2vec", el 27 de marzo de 2019. http://jalammar.github.io/illustrated-word2vec/ (consultado el 17 de abril de 2022).
[29] IBM, "Machine Learning", el 24 de marzo de 2021. https://www.ibm.com/pe-es/analytics/machine-learning (consultado el 8 de octubre de 2021).
[30] O. Theobald, *Machine learning for absolute beginners*. Independently Published, 2017.
[31] J. D. Kelleher, B. M. Namee, y A. D'Arcy, *Fundamentals of Machine Learning for Predictive Data Analytics: Algorithms, Worked Examples, and Case Studies*. MIT Press, 2015.

[32] A. Vaswani *et al.*, “Attention Is All You Need”, *ArXiv170603762 Cs*, dic. 2017, Consultado: el 29 de abril de 2022. [En línea]. Disponible en: http://arxiv.org/abs/1706.03762
[33] J. L. Ba, J. R. Kiros, y G. E. Hinton, “Layer Normalization”, *ArXiv160706450 Cs Stat*, jul. 2016, Consultado: el 17 de abril de 2022. [En línea]. Disponible en: http://arxiv.org/abs/1607.06450
[34] A. Géron, *Hands-On Machine Learning with Scikit-Learn and TensorFlow Concepts, Tools, and Techniques to Build Intelligent Systems*, 1a ed. O’Reilly Media, 2017.
[35] OCDE, “Manual de Frascati 2015”, *FECYT*, el 27 de septiembre de 2018. https://www.fecyt.es/es/publicacion/manual-de-frascati-2015 (consultado el 15 de abril de 2022).
[36] U. O. of R.- Innocenti, “Diseño y métodos cuasiexperimentales”, *UNICEF-IRC*. https://www.unicef-irc.org/publications/817-diseño-y-métodos-cuasiexperimentales.html (consultado el 10 de mayo de 2022).
[37] C. A. Bernal, *Metodología de la investigación: administación, Economia, humanidades y ciencias sociales*. Colombia, Bogota: Pearson Educación., 2010.
[38] M. Berndtsson, J. Hansson, B. Olsson, y B. Lundell, *Thesis Projects*. London: Springer, 2008. doi: 10.1007/978-1-84800-009-4.
[39] “El Product Backlog debe estar Ordenado no Priorizado”, *Scrum.org*. https://www.scrum.org/resources/blog/el-product-backlog-debe-estar-ordenado-no-priorizado (consultado el 24 de junio de 2021).
[40] K. Schwaber y J. Sutherland, “The Scrum Guide”, *Scrum.org*. https://www.scrum.org/resources/scrum-guide (consultado el 18 de abril de 2022).
[41] K. Schwaber y J. Sutherland, “La Guía de Scrum”. Ken Schwaber and Jeff Sutherland, noviembre de 2020. [En línea]. Disponible en: https://www.scrumguides.org/docs/scrumguide/v2020/2020-Scrum-Guide-Spanish-Latin-South-American.pdf
[42] H. I. Rhys, *Machine Learning with R, the tidyverse, and mlr*. Shelter Island, NY, 2020.
[43] Y. Pu, D. B. Apel, y C. Wei, “Applying Machine Learning Approaches to Evaluating Rockburst Liability: A Comparation of Generative and Discriminative Models”, *Pure Appl. Geophys.*, vol. 176, núm. 10, pp. 4503–4517, oct. 2019, doi: 10.1007/s00024-019-02197-1.
[44] J. P. Mueller y L. Massaron, *Deep Learning for Dummies*. For Dummies, 2019. Consultado: el 29 de abril de 2022. [En línea]. Disponible en: http://gen.lib.rus.ec/book/index.php?md5=CB2584FA0563E26207F9B021CAD75B2F
[45] A. Sherstinsky, “Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network”, *Phys. Nonlinear Phenom.*, vol. 404, p. 132306, mar. 2020, doi: 10.1016/j.physd.2019.132306.
[46] S. Wang y C. D. Manning, “Baselines and bigrams: simple, good sentiment and topic classification”, en *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, USA, jul. 2012, pp. 90–94.
[47] A. Y. Ng y M. I. Jordan, “On discriminative vs. generative classifiers: a comparison of logistic regression and naive Bayes”, en *Proceedings of the 14th International Conference on Neural Information Processing Systems: Natural and Synthetic*, Cambridge, MA, USA, ene. 2001, pp. 841–848.
[48] A. Baldominos, A. Cervantes, Y. Saez, y P. Isasi, “A Comparison of Machine Learning and Deep Learning Techniques for Activity Recognition using Mobile Devices”, *Sensors*, vol. 19, núm. 3, Art. núm. 3, ene. 2019, doi: 10.3390/s19030521.
[49] A. Korotcov, V. Tkachenko, D. P. Russo, y S. Ekins, “Comparison of Deep Learning With Multiple Machine Learning Methods and Metrics Using Diverse Drug Discovery Data

Sets", *Mol. Pharm.*, vol. 14, núm. 12, pp. 4462–4475, dic. 2017, doi: 10.1021/acs.molpharmaceut.7b00578.

[50] S. Bouktif, A. Fiaz, A. Ouni, y M. A. Serhani, "Optimal Deep Learning LSTM Model for Electric Load Forecasting using Feature Selection and Genetic Algorithm: Comparison with Machine Learning Approaches †", *Energies*, vol. 11, núm. 7, Art. núm. 7, jul. 2018, doi: 10.3390/en11071636.

[51] J. Devlin, M.-W. Chang, K. Lee, y K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding", *ArXiv181004805 Cs*, may 2019, Consultado: el 1 de mayo de 2022. [En línea]. Disponible en: http://arxiv.org/abs/1810.04805

[52] V. Sanh, L. Debut, J. Chaumond, y T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter", *ArXiv191001108 Cs*, feb. 2020, Consultado: el 1 de mayo de 2022. [En línea]. Disponible en: http://arxiv.org/abs/1910.01108

[53] A. F. Adoma, N.-M. Henry, y W. Chen, "Comparative Analyses of Bert, Roberta, Distilbert, and Xlnet for Text-Based Emotion Recognition", en *2020 17th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP)*, dic. 2020, pp. 117–121. doi: 10.1109/ICCWAMTIP51612.2020.9317379.

[54] Y. Liu *et al.*, "RoBERTa: A Robustly Optimized BERT Pretraining Approach", *ArXiv190711692 Cs*, jul. 2019, Consultado: el 1 de mayo de 2022. [En línea]. Disponible en: http://arxiv.org/abs/1907.11692

[55] P. Gupta, S. Gandhi, y B. R. Chakravarthi, "Leveraging Transfer learning techniques-BERT, RoBERTa, ALBERT and DistilBERT for Fake Review Detection", en *Forum for Information Retrieval Evaluation*, New York, NY, USA, dic. 2021, pp. 75–82. doi: 10.1145/3503162.3503169.

[56] M. Grandini, E. Bagli, y G. Visani, "Metrics for Multi-Class Classification: an Overview", *ArXiv200805756 Cs Stat*, ago. 2020, Consultado: el 23 de abril de 2022. [En línea]. Disponible en: http://arxiv.org/abs/2008.05756

[57] P. E. Kafrawy, A. Mausad, y H. Esmail, "Experimental Comparison of Methods for Multi-label Classification in different Application Domains", 2016, doi: 10.5120/20083-1666.

[58] D. Demszky, D. Movshovitz-Attias, J. Ko, A. Cowen, G. Nemade, y S. Ravi, "GoEmotions: A Dataset of Fine-Grained Emotions", *ArXiv200500547 Cs*, jun. 2020, Consultado: el 23 de abril de 2022. [En línea]. Disponible en: http://arxiv.org/abs/2005.00547

[59] E. Saravia, H.-C. T. Liu, Y.-H. Huang, J. Wu, y Y.-S. Chen, "CARER: Contextualized Affect Representations for Emotion Recognition", en *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, oct. 2018, pp. 3687–3697. doi: 10.18653/v1/D18-1404.

[60] D. Hendrycks y K. Gimpel, "Gaussian Error Linear Units (GELUs)", arXiv, arXiv:1606.08415, jul. 2020. doi: 10.48550/arXiv.1606.08415.

[61] M. Hasan, E. Rundensteiner, y E. Agu, "Automatic emotion detection in text streams by analyzing Twitter data", *Int. J. Data Sci. Anal.*, vol. 7, núm. 1, pp. 35–51, feb. 2019, doi: 10.1007/s41060-018-0096-z.

[62] T. Alam, A. Khan, y F. Alam, "Bangla Text Classification using Transformers", arXiv, arXiv:2011.04446, nov. 2020. doi: 10.48550/arXiv.2011.04446.

[63] A. Ezen-Can, "A Comparison of LSTM and BERT for Small Corpus", *ArXiv200905451 Cs*, sep. 2020, Consultado: el 29 de abril de 2022. [En línea]. Disponible en: http://arxiv.org/abs/2009.05451

[64] J. Nielsen, *Usability Engineering*. Boston, 1993.

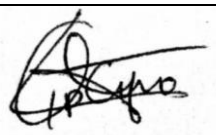
Anexos

Anexo 01: Informe de validación del producto acreditable por experto

Documento para medir la funcionalidad	
Elaborado por:	Sara María Benel Ramírez
Revisado por:	Ing. Chavarry Chankay Mariana
Aprobado por:	Ing. Chavarry Chankay Mariana
Objetivo de la Ficha	
Calificar la funcionalidad de la aplicaciones móvil y web desarrolladas y el modelo clasificador.	
Procedimiento	
El experto debe evaluar la funcionalidad de la aplicación móvil desarrollada.	
Nombre del Experto:	Ing. Guadalupe Teresa Lip Curo
Indicación:	Calificación de la funcionalidad de la aplicación móvil.
Criterios de evaluación:	Si: Se marca cuando el requerimiento indicado se esté cumpliendo en un 100% No: Se marca cuando el requerimiento indicado no se está cumpliendo en su totalidad.

Para el módulo de autenticación responda la siguiente pregunta			
Nº	**Pregunta**	**Criterio**	
		Si	**No**
1	¿Se validó el inicio de sesión en la aplicación móvil?	X	
2	¿Se validó el inicio de sesión en la aplicación web de administración?	X	
3	¿Se validó el registro de usuarios?	X	
4	¿Se validó que se guarden correctamente los datos del usuario?	X	
5	¿Se validó que no se pueda registrar un correo previamente guardado?	X	
6	¿Se validó que se llenen obligatoriamente todos los campos en el registro?	X	
7	¿Se validó que se cierre sesión correctamente?	X	
8	¿Se validó que se encriptan las contraseñas de los usuarios?	X	
Para el módulo de Chat responda la siguiente pregunta			
Nº	**Pregunta**	**Criterio**	
		Si	**No**
9	¿Se validó que el chatbot responda correctamente a los saludos?	X	

10	¿Se validó que el chatbot realice las preguntas en el orden correcto?	X	
11	¿Se validó que el chatbot realice las preguntas que se validaron por juicio de expertos?	X	
12	¿Se validó que el chatbot esté preparado para recibir mensajes imprevistos?	X	
Para el módulo de Clasificación de texto y estadísticas responda la siguiente pregunta			
N°	**Pregunta**	**Criterio**	
		Si	**No**
13	¿Se visualizan las estadísticas de emociones diarias?	X	
14	En caso no haya registrado emociones ¿Se validó que se muestre un aviso en lugar de los gráficos estadísticos?	X	
15	¿Se validó que se visualicen las estadísticas de emociones de los últimos 7 días?	X	
16	¿Se clasifica el texto en una de las 6 emociones planteadas?	X	
17	¿Se registra correctamente el resultado del modelo en la base de datos?	X	
18	¿Se lista correctamente el historial de emociones por fecha?	X	
Para el módulo de Perfil responda la siguiente pregunta			
N°	**Pregunta**	**Criterio**	
		Si	**No**
19	¿Se visualizan correctamente los datos del usuario?	X	
20	¿Se cambia correctamente el nombre del usuario?	X	
21	¿Se cambia correctamente la contraseña del usuario?	X	
Para el módulo de Administrador responda la siguiente pregunta			
N°	**Pregunta**	**Criterio**	
		Si	**No**
21	¿Se listan correctamente los usuarios registrados?	X	
22	¿Se listan correctamente el historial de emociones de cada usuario?	X	

23	¿Se crean correctamente nuevas cuentas de administrador?	X	
24	¿Se crean correctamente nuevas cuentas de usuario?	X	
25	¿Se elimina correctamente una cuenta de administrador?	X	
26	¿Se actualiza correctamente la contraseña de una cuenta?	X	
27	¿Se cambia correctamente el estado de un usuario?	X	
Observaciones			
Se recomienda que la pantalla de inicio muestre un mensaje para que el usuario comprenda la dinámica del chatbot, además sería conveniente que las listas con muchos datos se paginen en lugar de solo usar el scroll.			
Conclusión			
Yo como experto valido que se ha revisado todo el sistema y se concluye que cumple con las funcionalidades esperadas			
Firma del experto:		Firma del tesista:	

Anexo 02: Encuesta para medir la salud mental de los estudiantes universitarios

¿Cuánto tiempo llevas en la universidad? *En Años *

Elegir

¿Cómo considerarías tu experiencia en la universidad desde que ingresaste?

- Estresante
- Incómoda
- Tranquila
- Emocionante
- Igual que antes de entrar

¿Con qué frecuencia tu vida universitaria afecta tu estado de ánimo?

- Siempre
- A menudo
- Solo a veces
- Rara vez
- No la afecta en absoluto

¿Qué factor en la universidad consideras que es más probable que afecte tu estado anímico? *

- [ ] Tareas
- [ ] Exámenes
- [ ] Calificaciones
- [ ] Exposiciones
- [ ] Amistades
- [ ] Profesores
- [ ] Otros: _____

¿Has vivido situaciones de intensa tristeza o estrés a causa de actividades universitarias? *

- ◯ Sí
- ◯ No

¿Consideras que tu estado anímico afecta tu rendimiento universitario?

- ◯ Sí
- ◯ No

¿Alguna vez consideraste en ir a un psicólogo o psiquiatra?

- ◯ Sí
- ◯ No

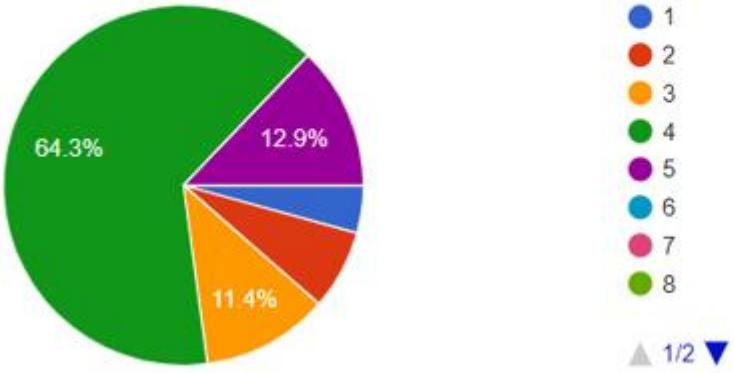
¿Asistes a un psicólogo o psiquiatra?

- ◯ Sí
- ◯ No

Anexo 03: Recolección de datos

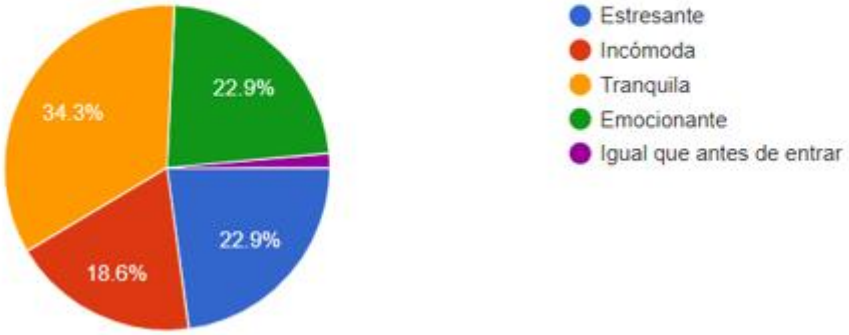
¿Cuánto tiempo llevas en la universidad? *En Años

70 respuestas



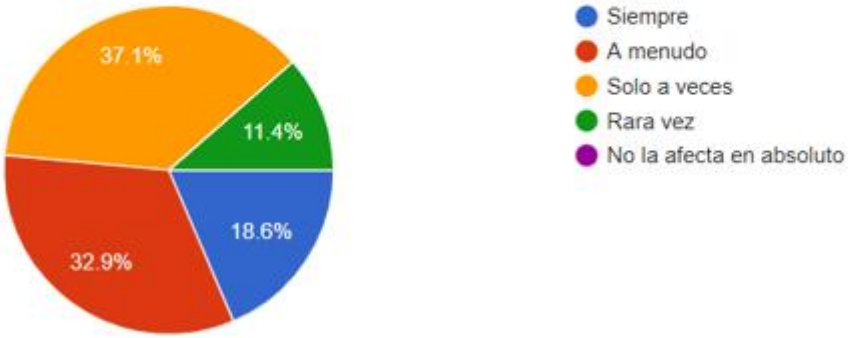
¿Cómo considerarías tu experiencia en la universidad desde que ingresaste?

70 respuestas



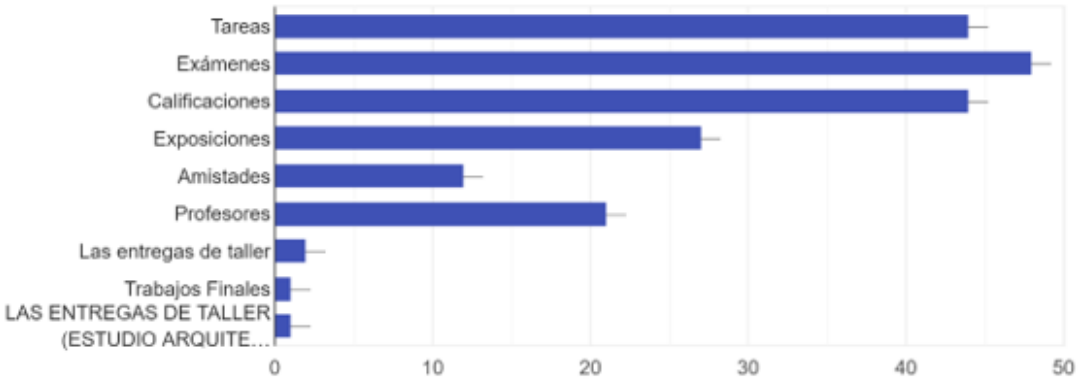
¿Con qué frecuencia tu vida universitaria afecta tu estado de ánimo?

70 respuestas



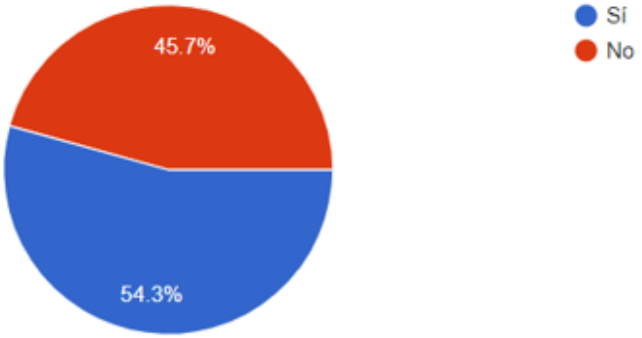
¿Qué factor en la universidad consideras que es más probable que afecte tu estado anímico?

70 respuestas



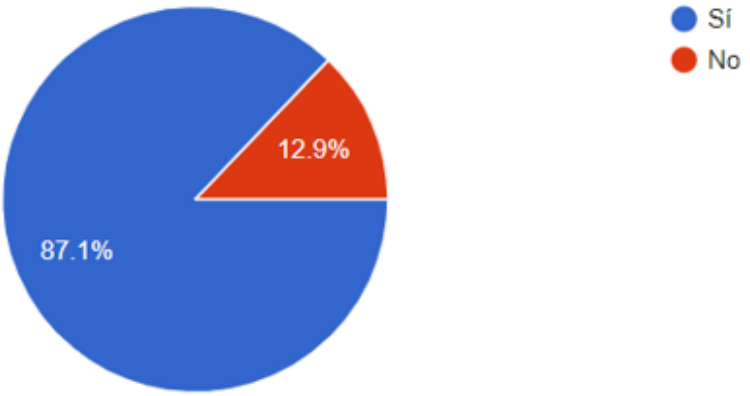
¿Has vivido situaciones de intensa tristeza o estrés a causa de actividades universitarias?

70 respuestas



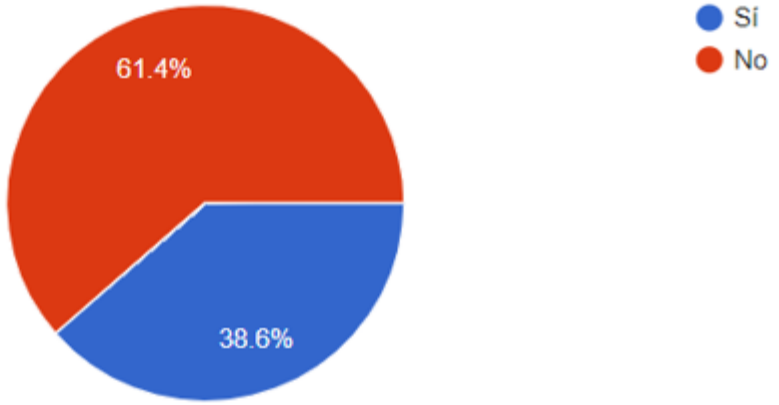
¿Consideras que tu estado anímico afecta tu rendimiento universitario?

70 respuestas



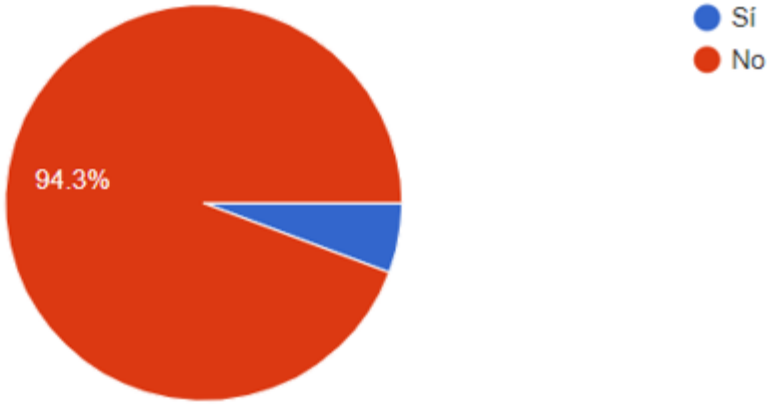
¿Alguna vez consideraste en ir a un psicólogo o psiquiatra?

70 respuestas



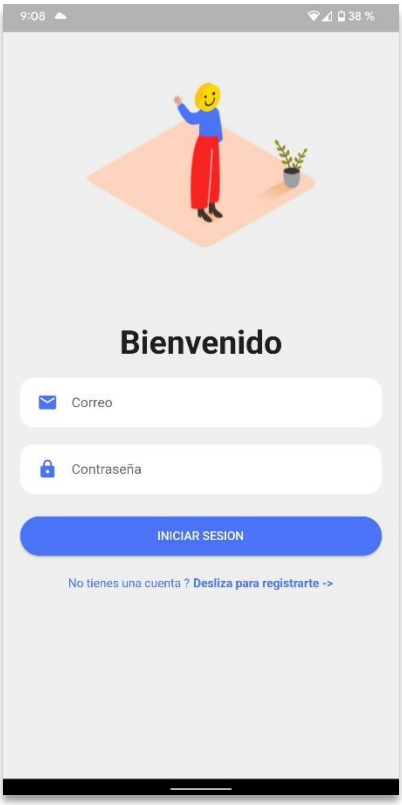
¿Asistes a un psicólogo o psiquiatra?

70 respuestas

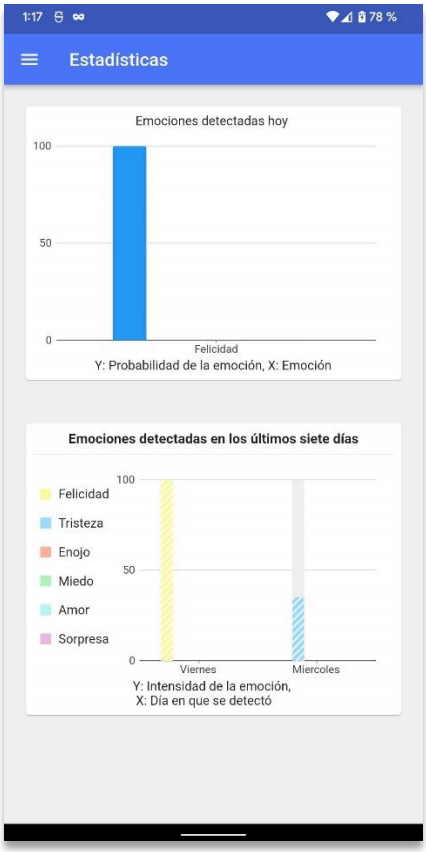


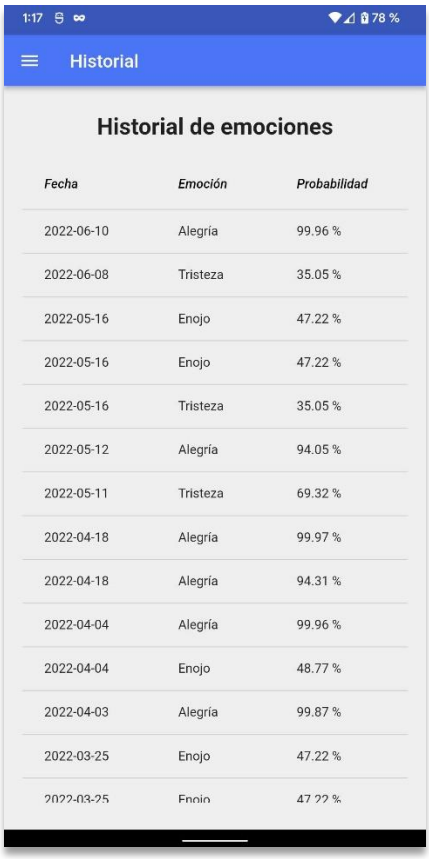
Anexo 04: Evidencias de la aplicación desarrollada

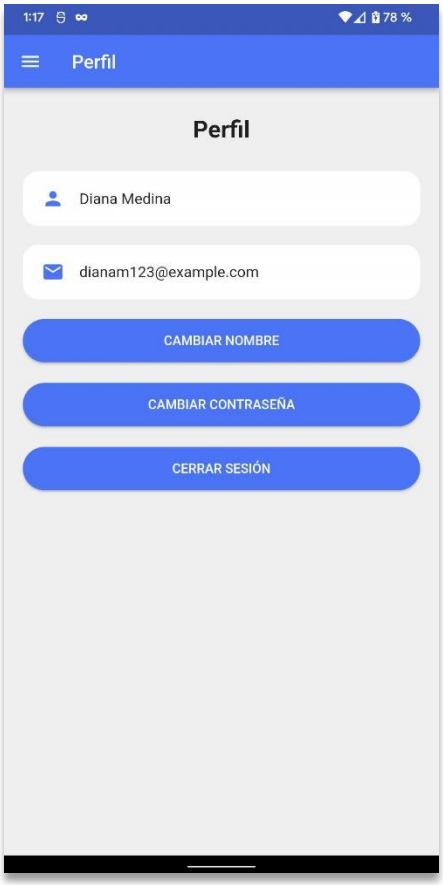
Aplicación móvil

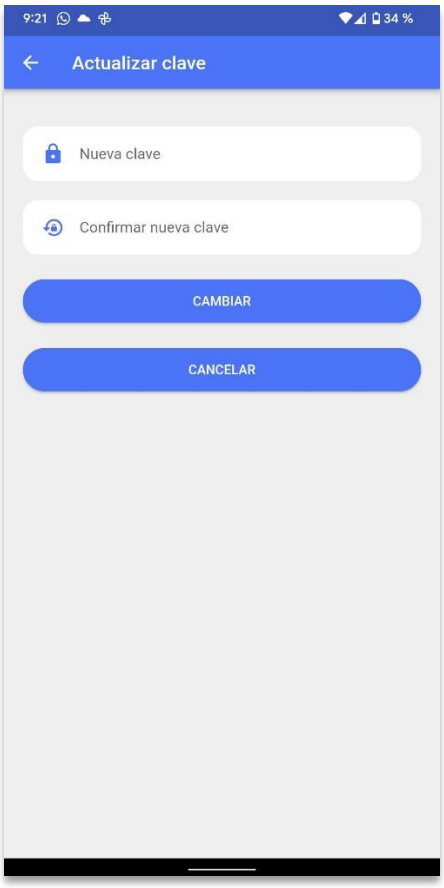


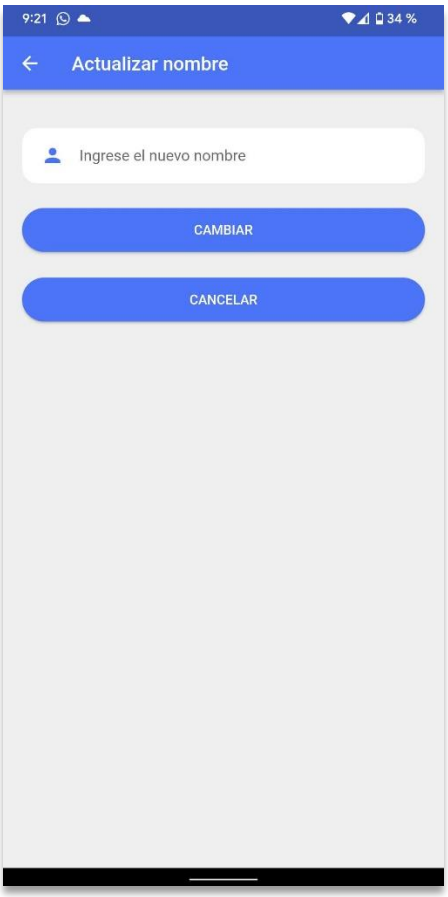

1:16
78 %
EmCare Bot
Hoy, 01:16
hola
Hola, ¿Qué tal te va?
Por favor sé claro y ten buena ortografía en tus respuestas.
me va bien
¿Qué te hace sentir feliz?
estoy más relajada porque terminé mis tareas
Muchas gracias por responder, tus resultados estarán listos en unos minutos.
¡Ha sido un gusto conversar contigo, hasta la próxima!
Escribe un mensaje...


1:17
78 %
Estadísticas
Emociones detectadas hoy
100
50
0
Felicidad
Y: Probabilidad de la emoción, X: Emoción
Emociones detectadas en los últimos siete días
Felicidad
Tristeza
Enojo
Miedo
Amor
Sorpresa
100
50
0
Viernes
Miercoles
Y: Intensidad de la emoción,
X: Día en que se detectó

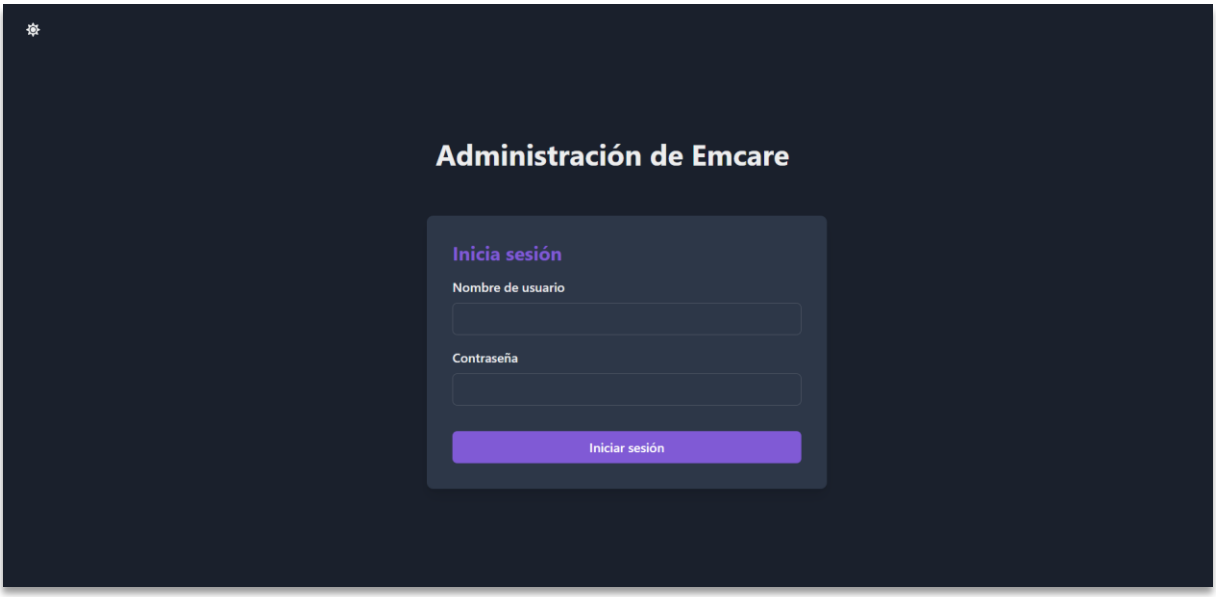

1:17
78 %
Historial
Historial de emociones
Fecha
Emoción
Probabilidad
2022-06-10 Alegría 99.96 %
2022-06-08 Tristeza 35.05 %
2022-05-16 Enojo 47.22 %
2022-05-16 Enojo 47.22 %
2022-05-16 Tristeza 35.05 %
2022-05-12 Alegría 94.05 %
2022-05-11 Tristeza 69.32 %
2022-04-18 Alegría 99.97 %
2022-04-18 Alegría 94.31 %
2022-04-04 Alegría 99.96 %
2022-04-04 Enojo 48.77 %
2022-04-03 Alegría 99.87 %
2022-03-25 Enojo 47.22 %

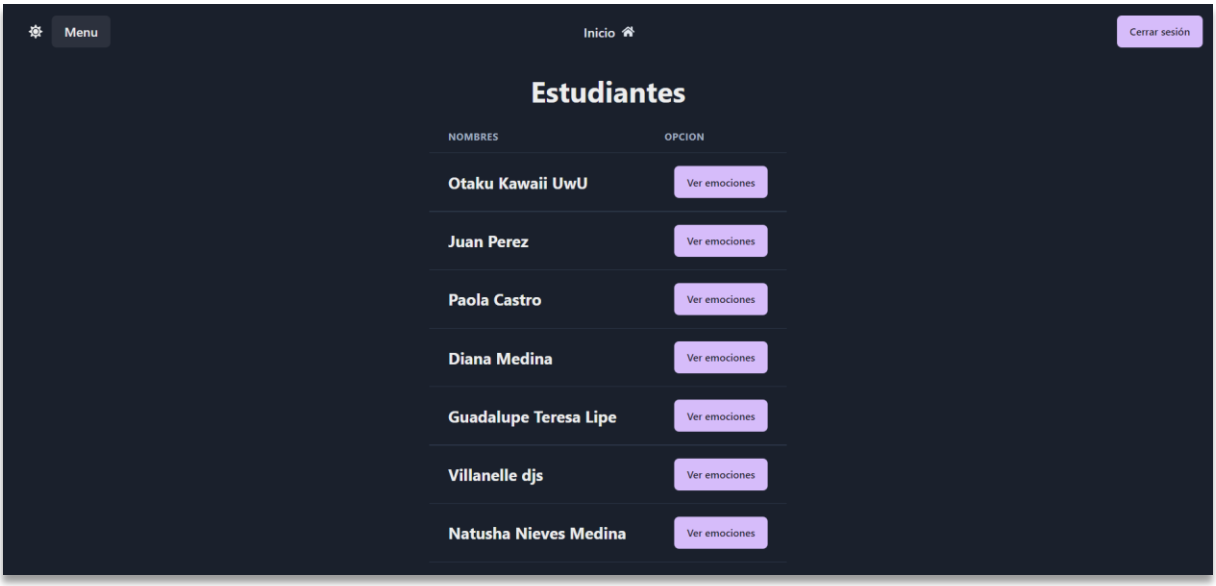

1:17
78 %
Perfil
Perfil
Diana Medina
dianam123@example.com
CAMBIAR NOMBRE
CAMBIAR CONTRASEÑA
CERRAR SESIÓN

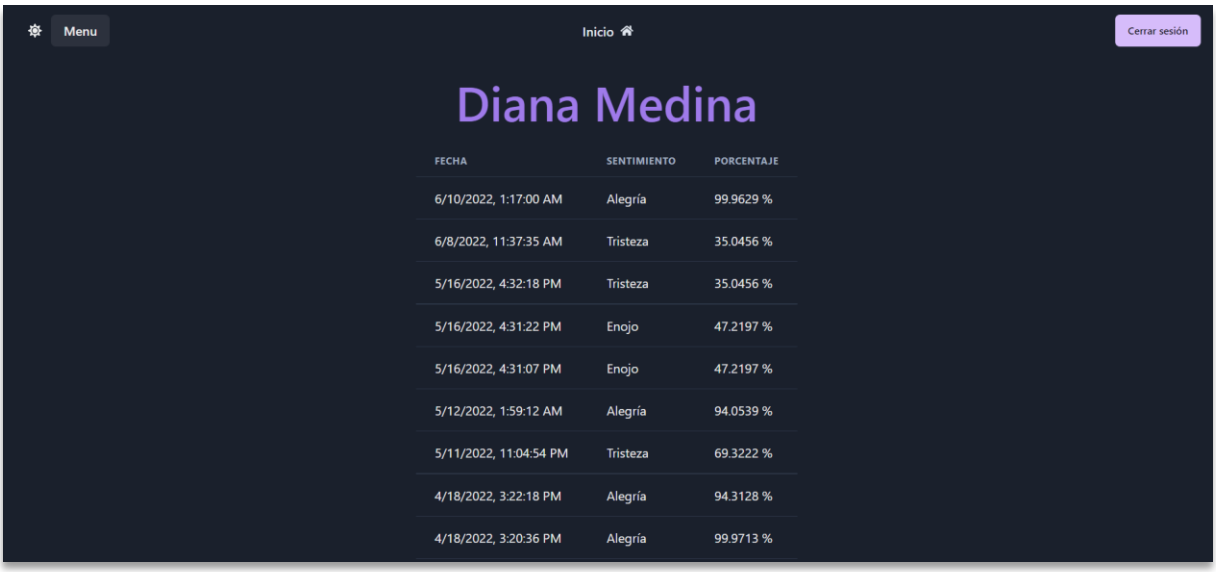

9:21
34 %
Actualizar clave
Nueva clave
Confirmar nueva clave
CAMBIAR
CANCELAR


9:21
34 %
Actualizar nombre
Ingrese el nuevo nombre
CAMBIAR
CANCELAR

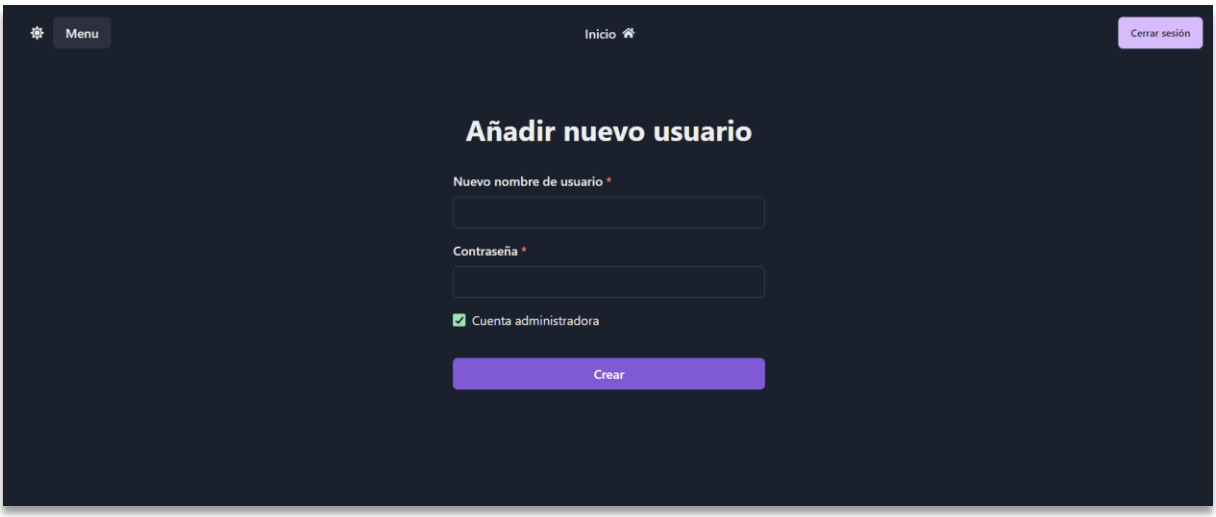
Aplicación web






Menu
Inicio
Cerrar sesión
Diana Medina
FECHA
SENTIMIENTO
PORCENTAJE
6/10/2022, 1:17:00 AM
Alegría
99.9629 %
6/8/2022, 11:37:35 AM
Tristeza
35.0456 %
5/16/2022, 4:32:18 PM
Tristeza
35.0456 %
5/16/2022, 4:31:22 PM
Enojo
47.2197 %
5/16/2022, 4:31:07 PM
Enojo
47.2197 %
5/12/2022, 1:59:12 AM
Alegría
94.0539 %
5/11/2022, 11:04:54 PM
Tristeza
69.3222 %
4/18/2022, 3:22:18 PM
Alegría
94.3128 %
4/18/2022, 3:20:36 PM
Alegría
99.9713 %


Menu
Inicio
Cerrar sesión
Cambiar contraseña
Nueva contraseña *
Nueva contraseña
Confirmar nueva contraseña *
Nueva contraseña
Cambiar


Menu
Inicio
Cerrar sesión
Añadir nuevo usuario
Nuevo nombre de usuario *
Contraseña *
Cuenta administradora
Crear


Menu
Inicio
Cerrar sesión
Lista de usuarios
NOMBRES DE USUARIO
TIPO DE USUARIO
ESTADO
ELIMINAR
ESTADO
abaldarrago
Normal
Activo
Eliminar
Banear
admin
Administrador
Activo
Eliminar
Banear
josefigueroa
Administrador
Activo
Eliminar
Banear
jvilchez
Normal
Desactivado
Eliminar
Activar
mvallejos
Normal
Desactivado
Eliminar
Activar
pcastro
Normal
Activo
Eliminar
Banear
sbenel
Administrador
Activo
Eliminar
Banear