Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación

SOFTWARE PARA DIAGNÓSTICO EXPRESS DE TRASTORNOS DE LA CONDUCTA ALIMENTARIA (TCA) APLICANDO MINERÍA DE DATOS

Presenta:

JOSÉ MARÍA GUTIÉRREZ ONOFRE

Tesis para obtener el grado de:

LICENCIADO EN INGENIERÍA EN CIENCIAS DE LA COMPUTACIÓN

ENERO - 2023

Director de tesis:
DRA. BEATRIZ BELTRÁN MARTÍNEZ

Asesor de tesis:
DRA. BEATRIZ BELTRÁN MARTÍNEZ

Resumen

Trastornos de la Conducta Alimentaria (TCA) son un conjunto de enfermedades (entre ellas la Bulimia Nerviosa, la Anorexia, entre otras) que afectan la calidad de vida las personas. En una entrevista realizada a la Coordinadora de la Clínica Ángeles Trastornos de la Conducta Alimentaria S.C., explicó que causas de índole económica, tiempo a dedicar y falta de motivación, resulta en que posibles pacientes no acudan a una consulta y por ende no se identifiquen posibles casos de TCA.

En adición a lo anterior, al ser de las primeras Clínicas en México especializadas en TCA, cuenta desde el año 2004 hasta el año 2021, con una base de datos de sus pacientes que contiene datos de control y descriptivos, así como sus respectivos diagnósticos y seguimientos.

Por consecuencia, aplique mis conocimientos adquiridos durante la carrera, con el fin de solucionar un problema de la vida real. De este modo, con dicha base de datos, me apoye en diversas áreas de la computación como: la Ingeniería de Software, la Minería de datos, entre otras; para codificar un software que cumpla como objetivo realizar diagnósticos express de trastornos de la conducta alimentaria; respaldados en la experiencia de la Clínica y la Computación para detectar más casos e incluso salvar vidas.

En la presente tesis, explico mi propuesta de desarrollo de software codificado con el lenguaje de programación Python en su versión 3, así como documento el viaje realizado del cómo se logró el objetivo de alcanzar a diagnosticar con una precisión general del 91.69 % con base en los datos disponibles y las pruebas arrojadas con la técnica: Red Neuronal con validación cruzada con aleatoriedad, realizando un procesamiento previo a los datos y contrastando los resultados obtenidos.

Por último, aclaro que la privacidad de los pacientes se respetó en todo momento, y que el software no busca de ninguna manera reemplazar el trabajo de un profesional en el área de la salud; al contrario, al ser un coadyuvante, persigue animar a posibles pacientes de tomar el primer paso para su tratamiento y una vida mejor.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1

Introducción

1.1. Definición del problema

Un diagnóstico de trastornos de la conducta alimentaria (TCA); requiere entre siete y ocho horas, además el paciente debe acudir presencialmente a sus instalaciones en Interlomas, Estado de México. Conforme a la entrevista realizada a la coordinadora de la Clínica Ángeles Trastornos de la Conducta Alimentaria S.C. [Bustinzar, 2022]

Causas económicas, tiempo a dedicar y falta de motivación son razones o motivos para no acudir por un diagnóstico explicó la entrevistada.

1.2. Objetivo general y objetivos específicos

Objetivo general: Determinar rápidamente un diagnóstico a personas con sospecha de padecer un TCA, creando un software que aplique técnicas de minería de datos. En caso de riesgo, con tratamiento médico, lograr mayor calidad de vida e incluso salvar vidas.

Objetivos específicos:

1. Analizar y preprocesar la base de datos proporcionada, con Python 3, para aplicar técnicas de minería de datos.

2. Aplicar al menos un algoritmo de agrupamiento y uno de clasificación en los datos preprocesados para la recolección de métricas que sirvan de apoyo para determinar su rendimiento.

3. Comparar resultados y seleccionar la técnica más adecuada para obtener un diagnóstico express.

4. Generar diagramas de ingeniería de software para una comprensión mayor de la implementación del sistema.

1.3. Preguntas de investigación

Se plantearon las siguientes preguntas de investigación:

1. ¿Es posible crear un software que obtenga diagnósticos express preliminares con los datos recabados de la Clínica Ángeles?

2. ¿Los datos disponibles son suficientes para entrenar y validar los modelos de clasificación?

1.4. Hipótesis

Hipótesis: Los datos disponibles de la Clínica Ángeles de pacientes diagnosticados con algún trastorno de la conducta alimentaria acorde al DSM-V, son suficientes para aplicar algoritmos de agrupamiento y clasificación con validación cruzada, logrando con al menos uno, obtener un diagnóstico express respaldado, para ser un coadyuvante de inicio para posterior valoración de un experto.

1.5. Justificación

Hasta el momento de la escritura de la presente tesis, se investigó si ya existiera un software o programa que solucionara un problema similar, no obstante, solo se encontraron programas que buscan prevenir dichos trastornos mediante la elaboración de programas no computacionales, es decir, una metodología ordenada y clara a seguir desde el ámbito médico; en cuanto a software, se encontró un post en una la revista healthnine [Timmons, 2022],

donde se exponen sugerencias de aplicaciones para tratar el trastorno mediante monitoreo y registro de los alimentos consumidos, en términos simples hay soluciones de prevención y tratamiento más no de detección.

En adición a lo anterior, la mayor motivación para realizar este proyecto es aplicar el conocimiento adquirido a lo largo de mi carrera para solucionar un problema real.

1.6. Límites de la tesis

El presente trabajo de tesis no abarca ni documenta los resultados del software puesto en práctica con nuevos pacientes y por ende no corrobora ni estudia resultados post elaboración del software. Tampoco contempla la elaboración de un cuestionario corto que permita obtener el puntaje EAT-TRASTORNO en un tiempo menor.

1.7. Estructura de la tesis

El capítulo 1 (Introducción) es el preámbulo a la tesis con la definición del problema, el objetivo general así como los particulares, las preguntas de investigación que dieron origen a la hipótesis, la justificación y por último el límite de la tesis.

En el capítulo 2 (Marco Teórico), se comparte un resumen con los conceptos que deben comprenderse para lograr leer la presente tesis con efectividad.

El capítulo 3 (Implementación) propone los casos de uso y la abstracción de clases agrupadas en paquetes, que desempeñan una sola responsabilidad general y que en conjunto conforman a la denominada Suite Diagnóstico Express TCA.

En el capítulo 4 se explica el estado de los datos y su preprocesamiento para su posterior procesamiento y presentación de resultados en el capítulo 5.

Finalmente en el capítulo 6 (Conclusiones) se abordan la conclusión final y el trabajo a futuro propuesto.

Capítulo 2

Marco Teórico

A continuación resumo y agrupo los conceptos teóricos necesarios para que el lector tenga un preámbulo de los conocimientos necesarios para la lectura y comprensión de la presente tesis.

2.1. Ingeniería de Software

2.1.1. Definición

La ingeniería de software acorde al autor Pressamn [Pressman, 2010] es el proceso que conjunta un grupo de metodologías, en su mayoría prácticas y las herramientas necesarias con el fin de construir software de alta calidad.

La pregunta natural por consiguiente es: ¿Qué es software?. Sin entrar mucho en detalle para el objetivo del marco teórico, tomaré las definiciones 1 y 2 expuestas en el capítulo uno del libro escrito por Pressman [Pressman, 2010]: «Un conjunto de instrucciones que cuando se ponen en marcha proveen una o varias funcionalidades y rendimiento» y «Estructuras de datos que habilitan a los programas manipular información adecuadamente».

Asimismo, Pressman [Pressman, 2010] acierta con sus afirmaciones acerca del software:

1. Es desarrollado y conlleva un proceso de ingeniería, es decir, no es manufacturado desde un punto de vista clásico.

2. Si bien la industria cada vez avanza hacia desarrollo basado en componentes, la mayoría del software sigue siendo personalizado.

2.1.2. Principios básicos

La ingeniería de software es un ámbito muy amplio, con diversas técnicas, que debido al tamaño del problema no sería eficiente abarcar todas; para su elección me soporté en los siguientes principios de proceso y prácticos.

De proceso:

1. Enfoque en la calidad en cada paso.
2. Estar preparado para adaptarse.
3. Crear productos de trabajo que provean valor a otros.

Pressman [Pressman, 2010] establece más principios que los anteriores, no obstante, desde mi perspectiva son los más importantes.

Prácticos:

1. Divide y vencerás.
2. Entender el uso de la abstracción.
3. Consistencia.
4. Transformar la información (Interfaz Gráfica).
5. Software que exhiba una modularidad efectiva.

2.1.3. Técnicas escogidas

Sin duda, todas las técnicas expuestas en el libro son relevantes; para honrar los procesos anteriores y lograr el cuarto objetivo planteado, me centre en dos: los casos de uso y el modelado orientado a clases.

2.1.4. Casos de Uso

Los casos de uso resumen los requerimientos del software, es decir: que al final del día el usuario va a realizar, de este modo se centra uno en el problema y la solución. En esencia un caso de uso es una historia que delimita en términos simples, las tareas e interacciones a realizar desde el punto de vista de uno o más actores, que denotarán el comportamiento del software. Los casos de uso pueden ser una simple lista, o un diagrama UML como será el caso en esta tesis.

2.1.5. Modelado orientado a clases

En el análisis de requerimientos, Pressman [Pressman, 2010] hace énfasis que es el resultado de la especificación del software; en términos de funciones operacionales, indican su interacción con otros elementos de sistema.

Pressman propone diversos modelos como Modelos orientados al flujo; sin embargo, me enfocaré en el modelo orientado a clases; que gráfica las clases orientadas a objetos con atributos y comportamiento por dos motivos:

1. Mostrar como las clases colaboran para alcanzar los requisitos del sistema para su posterior programación.

2. Visualmente se podrá deducir el comportamiento del sistema aunque no se conozcan los detalles de su implementación.

2.2. Minería de datos

En términos simples, Minería de datos es una rama de la computación que estudia la recolección, limpieza, procesamiento y el análisis de datos con el fin de extraer información que no es deducible a simple vista.

2.2.1. Procedimiento

Acorde al autor Aggarwal [Aggarwal, 2015], el procedimiento de Minería de datos consiste en tres fases principales:

1. Recolección de datos.

2. Extracción de características y limpieza de datos (Preprocesamiento).

3. Procesamiento analítico.

La metodología en su primera fase contempla que se tienen las herramientas necesarias para la recolección de datos; desde computadoras hasta el mismo papel. Paso que es altamente dependiente de la aplicación y de suma importancia. La información recabada impactará en gran medida las fases siguientes, por lo tanto: la toma de buenas decisiones desde un inicio es crítica.

Los datos recolectados reciben un tratamiento previo (preprocesamiento): extracción de características, corrección de partes erróneas, datos incompletos, ordenamiento y transformación a un formato amigable; para finalmente avanzar a la tercera fase (procesamiento analítico) con algoritmos de minería de datos. A continuación se describen las técnicas ocupadas para la extracción de características.

2.2.2. Portabilidad de datos - Binarización

Los datos en su mayoría son heterogéneos, es decir: existe una variedad en el tipo de dato, que puede ser: numérico, texto, mixto, etc. Si bien lo ideal sería particularizar cada algoritmo a cada tipo de dato (con el fin de tener resultados optimizados), en la práctica no es factible al consumir más tiempo según Aggarwal [Aggarwal, 2015].

Para convertir datos categóricos (como lo son los diagnósticos) es necesario aplicar la técnica de Binarización que consiste en convertir las categorías en cadenas únicas de 0's y un solo 1, donde el 1 denota la categoría a la que pertenece, si se tienen por ejemplo cuatro categorías entonces se tendrían cuatro cadenas de longitud cuatro como se puede ver en la Tabla 2.1

Las categorías con binarización se podrán utilizar en algoritmos que requieran datos numéricos como entrada o salida; así se mantiene la paridad de relevancia sin perder la distinción entre ellas.

De otro modo, si los datos son convertidos a números naturales, por ejemplo, el número 1 para categoría A, el 2 para la B y así sucesivamente; al aplicarlo, la ponderación de las categorías no sería correcta, puesto que la categoría D tendría cuatro veces más peso que la categoría A e influiría altamente en los cálculos y por ende: en los resultados.

Tabla 2.1: Binarización de cuatro categorías

Categoría	Binarización
Categoría A	1000
Categoría B	0100
Categoría C	0010
Categoría D	0001

2.2.3. Escala y normalización

En la sección anterior, se comentó la importancia de binarizar los datos categóricos con el fin de hacer una distinción entre ellos y al mismo tiempo estar ponderados con el mismo peso; se presenta una situación similar pero ahora con datos numéricos.

En múltiples datos con diferentes escalas, por ejemplo si la estatura está en metros y el peso en kilogramos, claramente un valor tiene mayor magnitud afectando al resultado producido por los algoritmos de Minería de datos que al final del día efectúan sumas, multiplicaciones etc.; el dato con mayor escala tendrá una repercusión mayor (numéricamente hablando), al grado de incluso opacar a los otros datos, cuando en realidad ambos tienen la misma contribución al resultado real.

Para resolver dicho problema, se aplica la técnica de escala y normalización; que para esta tesis, no haré distinción entre ellos. Acorde al autor Aggarwal [Aggarwal, 2015] existen dos modos de lograr la estandarización de los datos: el primero, consiste en únicamente restar el mínimo al valor y dividirlo entre la sustracción del máximo y mínimo; el segundo es aplicando estadística más avanzada que se explica a continuación.

Definición 1. Sea x una variable aleatoria con n número de datos, para estandarizarlos entonces aplicaremos la ecuación (2.1) para cada cada valor de i.

$$Z = \frac{x_i - \overline{X}}{\sigma^2} \tag{2.1}$$

donde $\overline{X}$ es la media aritmética que se calcula

$$\overline{X} = \frac{1}{n} \sum_{i=1}^{n} x_i \tag{2.2}$$

y σ^2 es la varianza que se calcula de la siguiente manera:

$$\sigma^2 = \frac{1}{(n-1)} \sum_{i=1}^{n} (x_i - \overline{X})^2 \tag{2.3}$$

De este modo los valores oscilarán en su mayoría entre el intervalo $[-3, 3]$, siendo esta técnica la utilizada para el cálculo de probabilidad con distribuciones normales.

2.2.4. Agrupamiento

Al procesar datos, es común querer resumirlos y visualizar su comportamiento, como lo estipula el autor Aggarwal [Aggarwal, 2015]. Las técnicas de agrupamiento se aplican para particionar puntos de datos en grupos que comparten similitudes de manera intuitiva.

Algoritmo de K-Means

El algortimo de K-Means o K-medias, es la elección por excelencia, debido a su simpleza y aplicación de uso general, que logra particionar los puntos de datos al aproximar el punto de dato a su representación más cercana [Aggarwal, 2015].

Antes de pasar al algoritmo como tal Aggarwal [Aggarwal, 2015] define la función $Dist()$ como se muestra en la ecuación 2.4.

Definición 2. La suma de los cuadrados de las distancias euclidianas entre los puntos de datos X e Y con i dimensiones aplicaremos la función Dist (2.4)

$$Dist(\overline{X}, \overline{Y}) = ||\overline{X_i} - \overline{Y_i}||_2^2 \tag{2.4}$$

De entrada, la fórmula a pesar de ser aparentemente corta pudiere asustar un poco, además genera confusión la notación, en la definición anterior $\overline{X}$ coloque que es la media del vector X y aquí estamos hablando de K-medias. ¿Serán lo mismo? los conceptos aparentemente se mezclan y además ¿La fórmula es para una, dos o más dimensiones?.

Para entenderla mejor, desmenuzamos y reescribimos la ecuación (2.4). Primeramente X e Y son puntos de datos y para no generar confusión con la notación $\overline{X}$ se retira la barra superior que indicaría que es la media del vector X. El subíndice i representan la i-ésima dimensión.

$$Dist(X, Y) = ||X_i - Y_i||_2^2 \tag{2.5}$$

Donde $||X_i - Y_i||_p^2$ es la p norma de $X - Y$ y por ende

$$||X_i - Y_i||_2^2 = (|X_1 - Y_1|^2)^{\frac{1}{2}} \tag{2.6}$$

Notemos como la ecuación (2.6) ya describe mejor lo descrito en la definición 2, sin embargo, la ecuación solo es válida para una dimensión (una columna de datos). De modo que para n dimensiones

$$||X_i - X_j||_2^2 = \sqrt{(|X_1 - Y_1|^2 + |X_2 - Y_2|^2 + ... + |X_n - Y_n|^2} \tag{2.7}$$

Finalmente si simplificamos (2.7) y reemplazamos en (2.5) tenemos

$$Dist(X, Y) = \sqrt{\sum_{i=1}^{n} |X_i - Y_i|^2} \tag{2.8}$$

Pseudocódigo 1. Algortimo de K-Means

Parámetros: puntos_datos, k

```
Mientras no mejora
    representadas ← seleccionarAleatoriamente(puntos_datos, k)
    mejora ← Falso
    grupos ← iniciarMatriz(k)
    Para cada p en puntos_datos
        min_distancia ← infinito
        g ← 0
        Para cada r en representadas
            distancia_euclidiana ← Dist(p, r)
            Si distancia_euclidiana < min_distancia
            Entonces
                min_distancia ← distancia
                g ← r
        Fin Para
        grupos[g] ← p
    Fin Para
    X ← seleccionarAleatoriamente(puntos_datos, 1)
    Y ← seleccionarAleatoriamente(representada, 1)
    Si representadas[Y] ← X
    y presenta_mejora
    Entonces
        mejora ← Verdadero
Devolver grupos
```

2.2.5. Clasificación

El algoritmo de K-Means pretende determinar los mejores grupos cuando hay una mejora con los mismos datos de entrada, es decir, no ajusta observando los resultados reales; a esto le llamamos aprendizaje no supervisado. En cambio los algoritmos de clasificación apuestan por mejorar el rendimiento del algoritmo mediante ajustes con base en los resultados reales y lo aprendido con los datos de entrada; los siguientes algoritmos utilizan aprendizaje supervisado.

Árboles de decisión

En la rama de computación están los árboles como estructura de datos, cada uno con sus variantes, pero en general entre más niveles de herencia contenga el árbol mayor será la cantidad de datos sin complicar el acceso a estos. Los árboles de decisión acorde al autor Aggarwal [Aggarwal, 2015], recaen su resultado mediante la toma de decisiones jerarquizada con base en una o más variables.

A modo de ejemplo, se busca predecir si una persona física es elegible para un crédito; las variables a tomar en cuenta son su score crediticio, si es trabajador activo o tiene otros ingresos. Por sus características se opta por un árbol de decisión. En la Figura 2.1 se ilustra dependiendo del valor en cada condicionante, el camino a tomar (la decisión); eventualmente el árbol terminará con un nodo color verde o rojo delimitando si es elegible o no respectivamente. El árbol de decisión contempla más de una variable aunque no al mismo tiempo, además no es necesario recorrer todas las posibilidades para obtener una predicción.

Asimismo, no es necesario normalizar los datos, puesto que en la evaluación total al resultar verdadero o falso producto de otras operaciones parciales booleanas, es decir, al momento de convertirse en verdadero o falso se equilibra el peso los resultados; provocando que los datos a utilizar puedan ser numéricos, categóricos, o booleanos. El algoritmo va ajustando y creando más niveles con tal de satisfacer todos los resultados con los datos de entrenamiento.

Redes Neuronales

Las redes neuronales acorde al autor Aggarwal [Aggarwal, 2015], son un modelo que busca simular el sistema nervioso de los seres vivos, el cual compuesto por células (neuronas). El aprendizaje se logra cambiando la fuerza de conexión sináptica entre ellas. En computación, se abstrae como el grosor del cable (un número) que las une.

Aggarwal [Aggarwal, 2015] afirma que el perceptrón, es la unidad atómica en la arquitectura de una red neuronal; consistiendo en un nodo (la neurona) donde hay 1 o más entradas X_1^n las cuales están conectadas mediantes cables

Figura 2.1: Diagrama Ejemplo Árbol de decisión

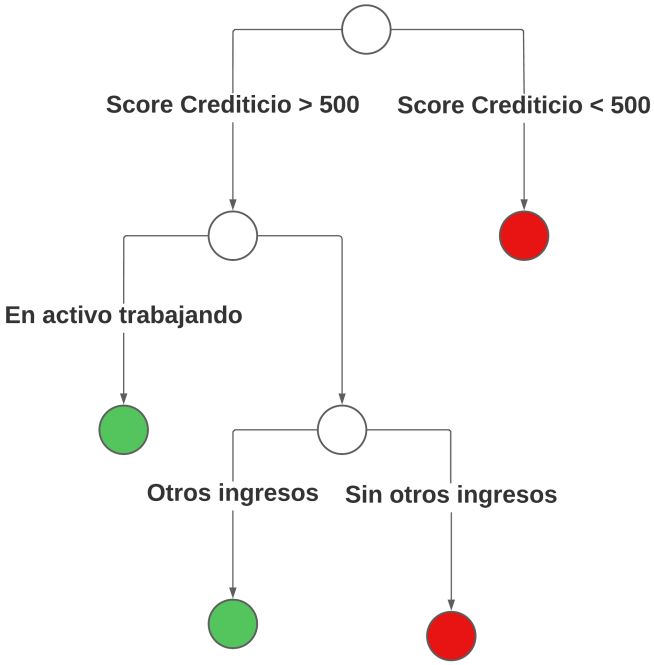


Figura elaborada por el autor

que son los pesos W_n como se puede observar en la Figura 2.2 (a). Asimismo, existe una variable b que corresponde al «bias», es decir, un valor independiente de las entradas y los pesos. La entrada es puesto en una función de activación produciendo así el resultado z_i.

Definición 3. Sea i el $i-ésimo$ dato (renglón) de entrenamiento y sean $X = \{x_i^j | x_i^j \in \mathbb{R}\}$ y $W = \{w_j | w_j \in \mathbb{R}\}$, b un número real y F una función de activación. El resultado z_i será determina por la siguiente ecuación.

$$z_i = F\left(\sum_{j=1}^{n} w_j x_i^j + b\right) \tag{2.9}$$

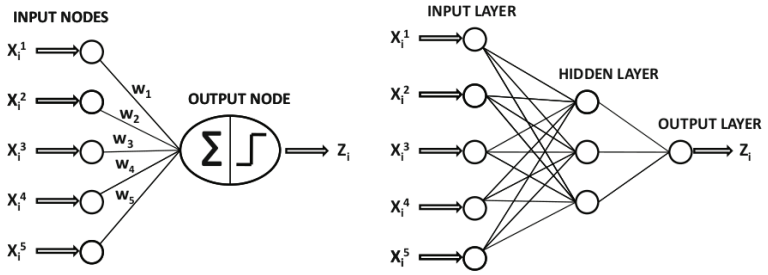
Una red neuronal multicapa no es otra cosa que múltiples neuronas interconectadas, donde no necesariamente cada capa tiene el mismo número de

neuronas, como podemos observar en la Figura 2.2 en el apartado b. Con lo anterior ya se tiene una red que ya puede obtener predicciones más no aprender de ellas. El proceso de aprendizaje esta dado por la ecuación (2.10) que va ajustando los pesos en función de error calculado entre la predicción y el valor real. Como nota al inicio los pesos serán aleatorios.

Definición 4. Sea t la $t-ésima$ iteración, W el vector de pesos, Y el vector de los resultados reales, Z el vector de predicciones y η la tasa de aprendizaje. El proceso de aprendizaje estará determinado por:

$$W^{t+1} = W^t + \eta\,(y_i - z_i)\,X_i \tag{2.10}$$

Figura 2.2: Diagrama Red Neuronal



(a) Perceptron (b) Multilayer

Figura elaborada por Aggarwal [Aggarwal, 2015]

Con una tasa de alto valor de aprendizaje la red aprenderá más rápido, no obstante la solución no será óptima para predecir resultados con nuevos datos. La función de activación dependerá de la aplicación, Tensorflow una biblioteca de Python [Abadi et al., 2015], explica en su documentación que el paquete **tf.keras.activations** contiene las funciones integradas a utilizar, así como su comportamiento y posibles modificaciones. Por ejemplo, si la aplicación requiere que el resultado sea 0 o 1, la función tanh no sería la ideal para la última capa; puesto que podría arrojar resultados menores a cero.

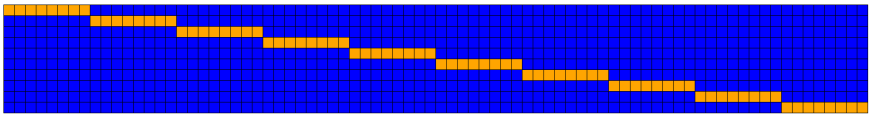
2.2.6. Validación Cruzada

Hasta ahora se ha hablado de datos de entrenamiento y reales; cómo aprenden los algoritmos y llegan a una predicción. La pregunta que resta es cómo medimos el rendimiento de estos si aún no se cuenta con los nuevos datos y sus resultados reales. Para ello Aggarwal [Aggarwal, 2015] define la validación cruzada como una técnica donde el conjunto de datos es dividido en m subconjuntos de igual tamaño. Donde uno de de los subconjuntos m es usado para pruebas y los demás $(m-1)$ para entrenamiento. Esto se repite hasta que todos los subconjuntos hayan sido probados. El que tenga mejor rendimiento será el seleccionado para producción.

La validación cruzada en 10 segmentos como lo sugiere Aggarwal [Aggarwal, 2015] para 80 datos, quedan bloques de 8 datos (en la figura 2.3 quedan representados con cuadrados color azul los datos de entrenamiento y de color naranja los datos de prueba); cada fila representa un segmento y notemos como los datos de prueba, aún aleatorizados, no se repiten. Por último, a partir de ahora, se hará referencia a los segmentos como particiones.

Figura 2.3: Diagrama Ejemplo Validación Cruzada

(a) Validación Cruzada sin aleatorizar



(b) Validación Cruzada con aleatorizar

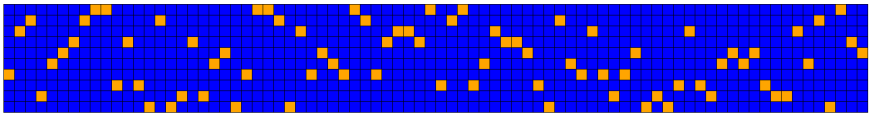


Figura elaborada por el autor

Capítulo 3

Implementación

Este capítulo explicará los diagramas de software elaborados que sirvieron de apoyo para una codificación más ordenada, clara y precisa; desde los casos de uso que ilustran como el software será utilizado por personas hasta los diagramas de clases que representan en una manera más técnica, la aproximación escogida para la programación del software.

3.1. Casos de Uso

El caso de uso principal es obtener un diagnóstico express. En un inicio el software será una herramienta interna, sin embargo, al hacer distinción del rol a ocupar, dependiendo del personal que utilice el software, se podrá disponer a público general si así se decidiera.

El usuario tendrá que iniciar de sesión para utilizar el software y poder identificar su rol. El actor **Usuario** (Por ejemplo. nutrióloga) únicamente tendrá la posibilidad de obtener un diagnóstico express, en cambio el actor **Administrador** podrá realizar lo mismo que el usuario más el permiso de cargar o actualizar el modelo de diagnóstico express TCA encargado de predecir el diagnóstico; más el poder obtener información relevante del modelo actual como su identificador y los nombres de los archivos ocupados. En la Figura 3.1 se puede observar el diagrama de casos de uso.

La implementación de estos casos uso serán responsabilidad de los paquetes DiagnósticoExpressTCA e InterfazWeb una vez se haya aplicado todo el

procedimiento de Minería de Datos.

Figura 3.1: Diagrama Casos de Uso

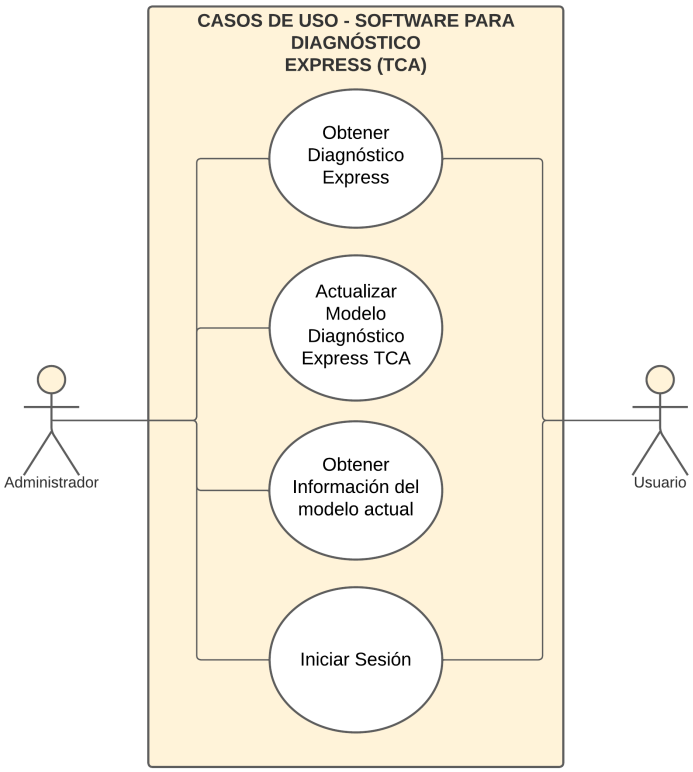


Figura elaborada por el autor

3.2. Suite Diagnóstico Express TCA

Abarcando el planteamiento del problema y con el fin de construir una solución adecuada, se planteó al hacer la revisión del marco teórico; para este apartado, más concretamente en la técnica de creación de algoritmos divide y vencerás, donde se partió el problema en dos. Primeramente construir los módulos necesarios para poder realizar el preprocesamiento, procesamiento y las pruebas que determinen la técnica de minería más adecuada y el modelo (el archivo producido finalmente a utilizar que arrojará las predicciones); la aplicación de minería de datos corresponde a esta primera parte. El segundo grupo de paquetes corresponde a la experiencia de usuario, siendo la lectura del modelo y la aplicación web donde recaen los casos de uso.

El conjunto de paquetes denominado **Suite Diagnóstico Express TCA**, donde cada uno contiene mayormente una clase o funciones que en suma realizan un trabajo en específico. En caso de necesitar mejoras en ciertas partes del software como: el modo de lectura o mejorar la experiencia de usuario; el código será más legible, no siendo necesaria la interrupción del servicio para su mantenimiento. Cuando se disponga de más datos, no será necesario montar la interfaz web para realizar nuevamente el proceso de Minería de datos y correr sus pruebas correspondientes.

Como se puede observar en la Figura 3.2, al primer grupo de paquetes se le identifica como **Pre-Diagnóstico Express**; siendo el paquete **Preprocesamiento** independiente. El paquete **AnálisisTCA** implementa las técnicas de minería de datos del marco teórico; no depende del paquete procesamiento (con agregación o composición) sino del archivo generado por este. El paquete **TcaPruebas** si depende de **AnálisisTCA** mediante composición. La segunda parte de la suite conjuga los paquetes **DiangósticoExpressTCA** e **InterfazWeb**. DignósticoExpressTCA lee el modelo escogido con base en los resultados de las pruebas del paquete **PruebasTCA** y alberga la implementación para los casos de uso. La interfaz web corresponde al proyecto Django.

Figura 3.2: Diagrama Suite Diagnóstico Express TCA

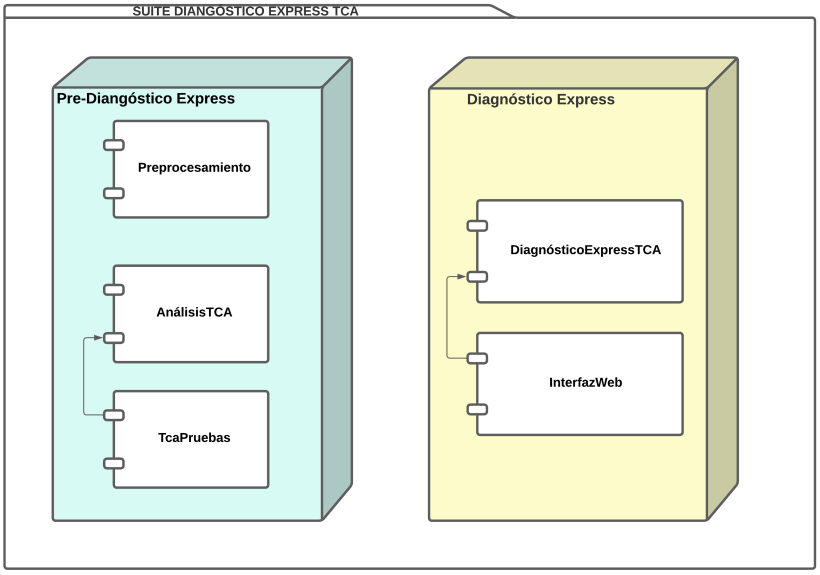


Figura elaborada por el autor

3.3. Diagramas de Clases

Como se habló en la sección anterior, el software contiene múltiples paquetes, donde cada paquete puede contener una o más clases además de una o más funciones. A continuación se describe cada paquete en conjunto con sus diagramas de clases, a excepción del paquete TcaPruebas, DiagnósticoExpressTCA e InterfazWeb.

3.3.1. Preprocesamiento

El diagrama de clases para el paquete Preprocesamiento como se muestra en la Figura 3.3, es muy sencillo, únicamente consta de una propiedad pública que es el la ruta del directorio de trabajo, es decir, donde se encuentran los archivos SPSS y donde se guarda el archivo resultante al preprocesar los datos, para lograrlo, se tienen dos métodos públicos, primeramente el método que convertirá los archivos SPSS a CSV para posteriormente el método preprocesar con pandas [pandas development team, 2020] lea y realice los pasos descritos en el apartado 4.4.

Figura 3.3: Diagrama de Clases - Preprocesamiento

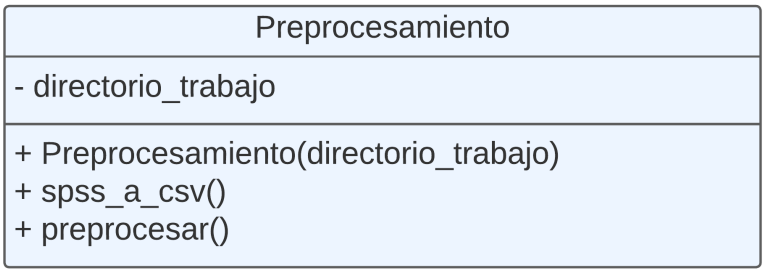


Figura elaborada por el autor

3.3.2. AnálisisTCA

El más extenso, se encarga de aplicar las diferentes técnicas de Minería de datos en forma ordenada, con o sin validación cruzada. Además realiza tareas adicionales como leer los datos preprocesados, proporcionar métricas que determinen la calificación, rendimiento o ambos de la técnica aplicada e implementarlo sin ser redundante en el código y pueda ser reutilizable en la mayor medida posible.

ParticiónValidaciónCruzada

Primeramente se resolvió la parte de validación cruzada, con la abstracción del concepto a una clase que contiene el número de segmento (particiones de aquí en adelante) al que pertenece. Cuatro estructuras que contienen los datos separados de entrenamiento y pruebas. El diagrama de clases se puede observar en la Figura 3.4 donde el constructor tiene tres puntos, porque la clase en sí es un dataclass, es decir, una clase que solo se pensó para almacenar propiedades y no métodos, así, el constructor recibe todas propiedades como parámetros y con el fin de evitar agrandar el diagrama innecesariamente, se optó por colocar los tres puntos para indicarlo.

Útil

Posteriormente, se abarcó como resolver aquellas funcionalidades adicionales que son necesarias; serán utilizadas por todas las técnicas de Minería

Figura 3.4: Diagrama de Clases - ParticiónValidaciónCruzada

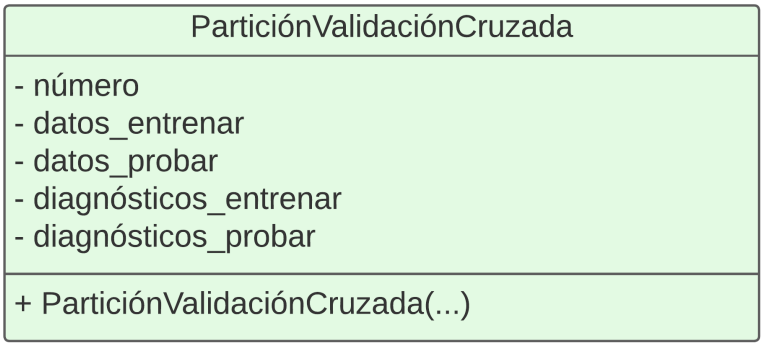


Figura elaborada por el autor

de datos y que no tienen mucha relación entre ellas. Para lograrlo, se creó una clase totalmente ESTÁTICA denomina Útil que solo realiza tres tareas:

1. Leer los datos preprocesados.
2. Normalizar datos.
3. Preparar las particiones para validación cruzada.

La primera tarea únicamente lee el archivo generado por el paquete Preprocesamiento y como se pueden tener múltiples versiones y diferentes directorios, se especifica su nombre y directorio. La segunda tarea normaliza los datos preprocesados, y la tercera, con esos datos ya preprocesados y normalizados se realiza el proceso de validación cruzada. De este modo, se logra ahorrar una gran cantidad de memoria al evitar instanciar la clase múltiples veces. En la Figura 3.5 se expone el diagrama de clases.

TécnicaMineríaDatos

En la Figura 3.6 se muestra la abstracción para la aplicación de múltiples técnicas de Minería de datos que comparten lógica; leer la base de datos preprocesada, normalizar los datos, inicializar variables de control como el directorio de trabajo son algunas de las funcionalidades a compartir.

Figura 3.5: Diagrama de Clases - Útil

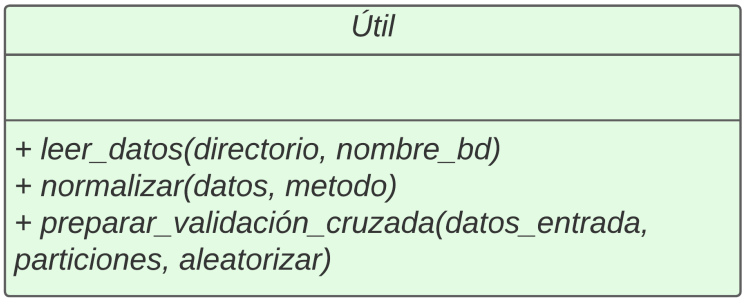


Figura elaborada por el autor

Primeramente se tienen propiedades como los datos en original y preprocesados, se mantuvieron ambos para la binarización más adelante; se necesitan los datos originales puesto que al normalizar únicamente se contemplan los datos de entrada, recortando los diagnósticos.

Asimismo, propiedades de alto interés como el id de procesamiento, se utiliza para identificar el momento de procesamiento y ubicar los archivos generados; como se puede observar se tienen tanto el modelo como el normalizador que son objetos finalmente.

Los métodos propuestos además del constructor y preparar, tienen como finalidad, leer el archivo separado por comas con los datos preprocesados para posteriormente normalizarlos. Los métodos abstractos son: procesar, guardar y generar reportes siendo suma importancia; de este modo se toma ventaja del paradigma orientado a objetos, concretamente con la herencia.

Cada técnica de datos implementa la lógica correspondiente y al mismo tiempo reutiliza el código compartido. Es decir, la clase TécnicaMineríaDatos Figura 3.6 es general con lógica común y las clases hereditarias particulares al procesar y reportar su rendimiento.

Por último, la definición posterior de métodos abstractos permite, tener orden en los nombres de las clases, forzando a tener la misma nomenclatura; en un futuro si se decidieran agregar más técnicas sean de minería de datos

o no, con heredar de esta clase y cumplir con el prototipo, no será necesario cambiar mucho o incluso nada en otra en otras clases que dependan ya sea con agregación o composición.

Figura 3.6: Diagrama de Clases - TécnicaMineríaDatos

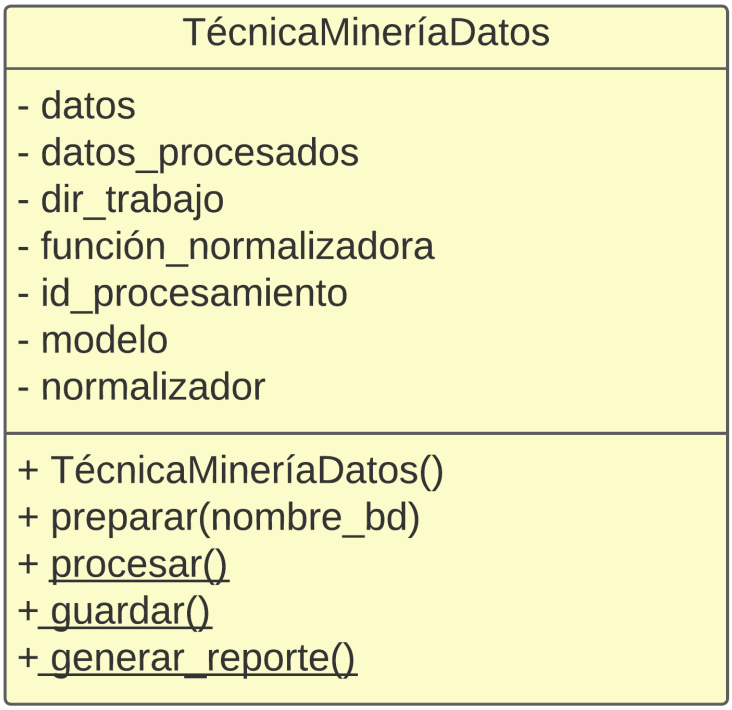


Figura elaborada por el autor

TécnicaAgrupamiento y TécnicaClasificación

Sin embargo, TécnicaMineríaDatos no es la única clase general o con métodos abstractos, debido a que el software implementará tanto agrupamiento como clasificación, se optó tener otra capa de abstracción, TécnicaAgrupamiento 3.7 y TécnicaClasificación 3.3.2 respectivamente.

TécnicaAgrupamiento como se observa en la Figura 3.7, no extiende sus propiedades, pero si agrega dos métodos adicionales; ambos relacionados con

la visualización. No obstante la clase TécnicaClasificación 3.3.2 si agrega una propiedad más, el binarizador; el objeto encargado de transformar el diagnóstico a una cadena binaria y viceversa. Por lo anterior el método binarización_diagnósticos, el encargado de crear dicho objeto; por otra parte, se incorpora el método abstracto determinante del rendimiento de la clasificación.

Figura 3.7: Diagrama de Clases - TécnicaAgrupamiento

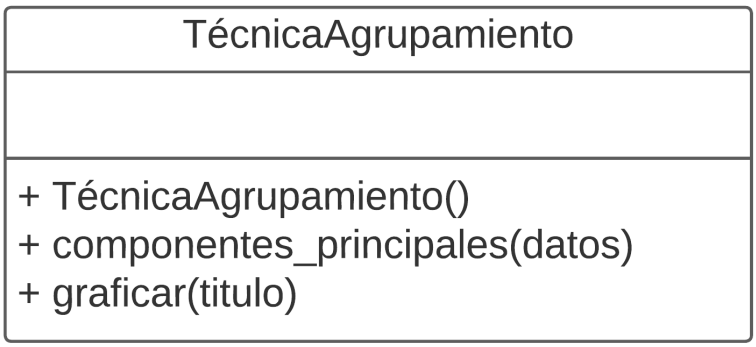


Figura elaborada por el autor

Figura 3.8: Diagrama de Clases - TécnicaClasificación

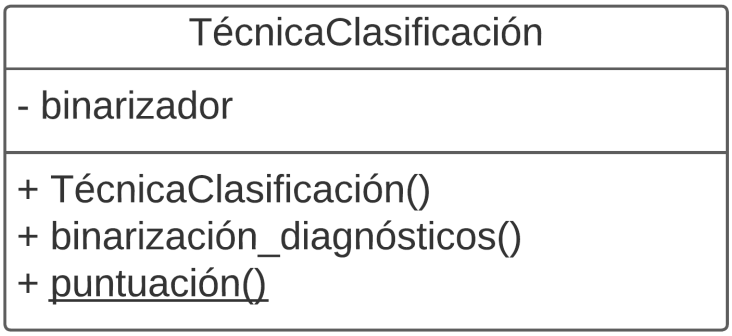


Figura elaborada por el autor

TécnicaClasificaciónValidaciónCruzada

Llegando al último nivel en cuanto a generalidad se trata, la clase TécnicaClasificaciónValidaciónCruzada como se visualiza en la Figura 3.9, ya no se extiende la clase con más métodos sean o no abstractos, al seguir diseñada como clase general, todavía no se sobrescriben dichos métodos abstractos, pero si se agregan propiedades de control necesarias para la validación cruzada.

Figura 3.9: Diagrama de Clases - TécnicaClasificaciónValidaciónCruzada

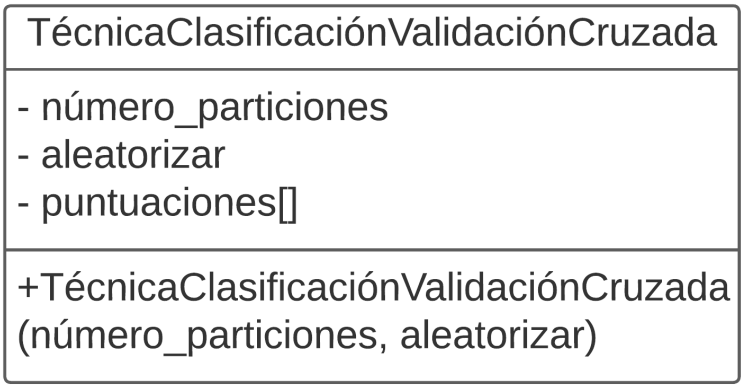


Figura elaborada por el autor

KMedias

Entrando en materia con las clases particulares, a partir de este punto no se colocan como tal los métodos abstractos en el diagrama a menos que se indique lo contrario; el lector sobreentiende que los métodos serán implementados. En la Figura 3.3.2 se muestra la clase KMedias, donde presenta una modificación al método abstracto; agrega un parámetro que determina el número de grupos a crear y un segundo método que prepara los metadatos para presentar una gráfica con mayor detalle.

ÁrbolDecisiónVC

Notemos como en el índice de las figuras, no se encuentra la figura individual de ÁrbolDecisionVC; la no integración de más propiedades o métodos a

Figura 3.10: Diagrama de Clases - KMedias

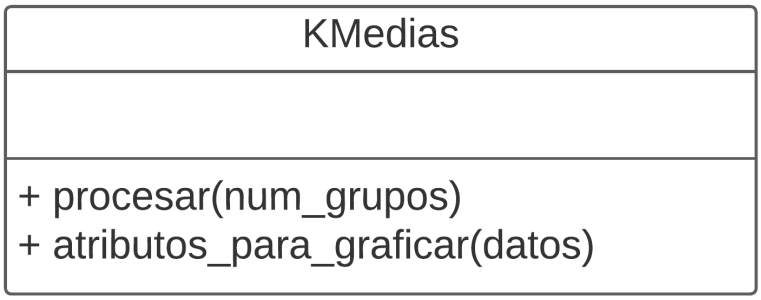


Figura elaborada por el autor

la clase TécnicaClasificaciónValidaciónCruzada lo determina. En la sección Clasificación 5.2.1 se argumenta el por qué no existe la clase ÁrbolDecisión; que correspondería a la aplicación del árbol de decisión pero sin validación cruzada.

Redes Neuronales

La Figura 3.11 muestra el diagrama para la red neuronal con o sin validación cruzada, ambas agregan la propiedad que determina la estructura de las capas ocultas; cada una hereda de una clase distinta, por este motivo se repite dicha propiedad.

Diagrama Completo

Finalmente, se tiene todo el árbol genealógico del paquete AnálisisTCA como se expone en la Figura 3.12 y se observa que clase hereda de cual, mostrando así el diseño alcanzado.

3.4. Paquetes Restantes

TcaPruebas: Se decidió atacar este módulo con un paradigma más funcional, puesto que fue pensado para usar como un programa de linea de comandos que se ejecutará varias veces e incluso concurremente. Construye

Figura 3.11: Diagrama de Clases - RedNeruonal

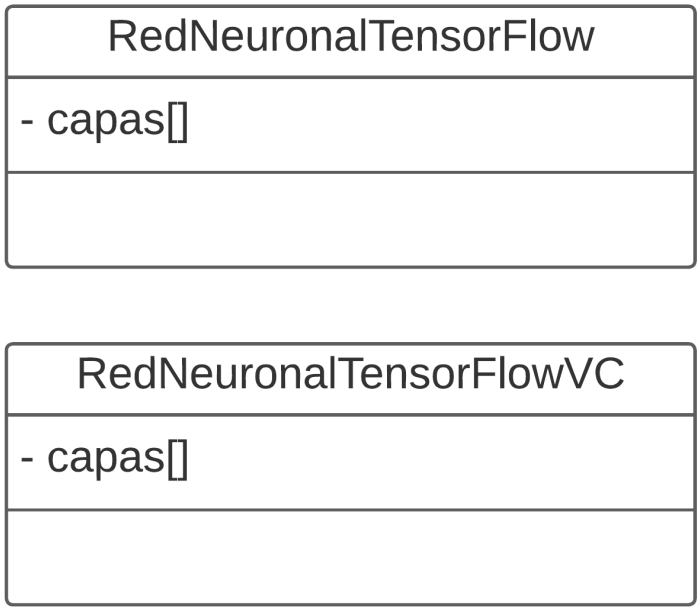


Figura elaborada por el autor

las pruebas a realizar con los argumentos enviados a través de la linea de comandos.

DiagnósticoExpressTCA: El paquete únicamente carga el modelo y el objeto normalizador que se tiene registrado como el de producción, además recibe los datos enviados por el usuario por medio de la interfaz web, normalizando la entrada y prediciendo el diagnóstico, realizando transformación inversa a la binarización y regresarlo.

InterfazWeb: Django [Foundation, 2021] es un marco de trabajo (Framework) formal que documenta perfectamente como se compone y se estructuran sus archivos aplicados a un proyecto, únicamente se importa el paquete DiagnósticoExpressTCA en el modelo (no confundir con el modelo que predice el diagnóstico de Minería de Datos) que se requiera.

Figura 3.12: Diagrama de Clases - AnálisisTCA - Completo

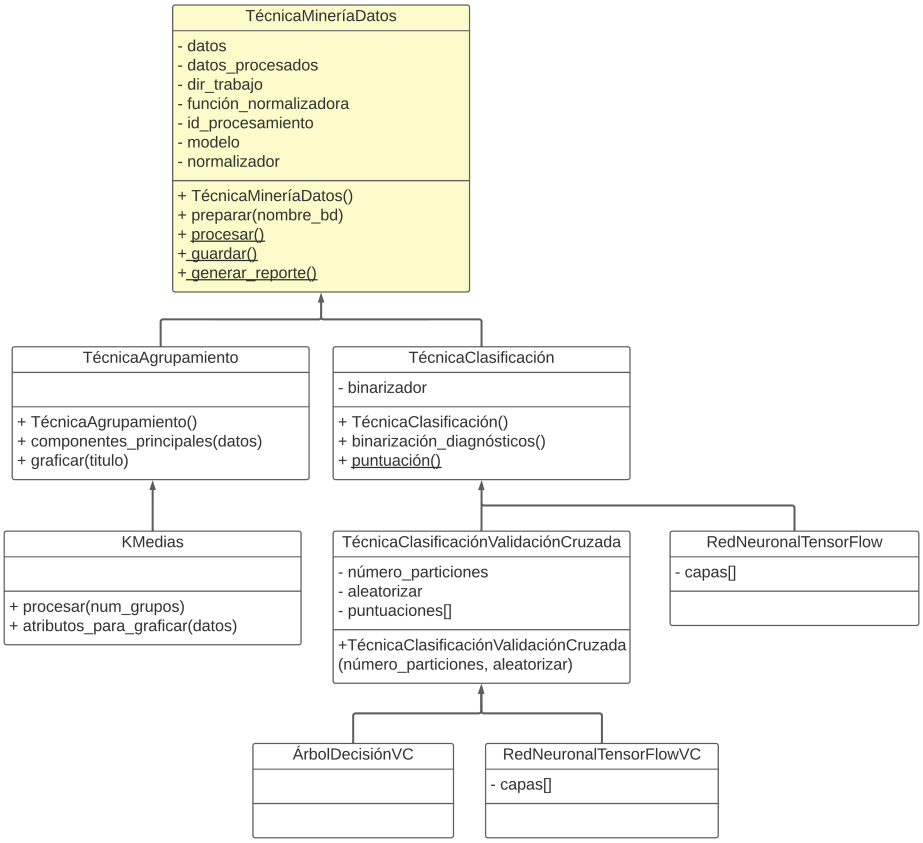


Figura elaborada por el autor

Capítulo 4

Datos y Preprocesamiento

4.1. Preámbulo

Los datos utilizados son propiedad de la Clínica Ángeles Trastornos de la Conducta Alimentaria, el tratamiento respeta en todo momento la privacidad de los pacientes.

Desde el año 2004 hasta el año 2021 se tienen registrado datos de control, genéricos, primera valoración, seguimientos de los pacientes, diagnóstico entre otros; para obtener un diagnóstico express, centré la atención en datos significativos, numéricos y pertenecientes a la primera valoración del paciente.

Clínica Ángeles registra los datos en el programa estadístico SPSS, no obstante, para el desarrollo de la presente tesis, se trabajó sobre archivos con extensión .sav cuyo nombre es acorde a la nomenclatura `BASE UTCAS {AÑO}.sav`

4.2. Lectura de los datos

La lectura se realizó con un programa elaborado con python 3 en conjunto con dos paquetes, en primer lugar con `pandas` [pandas development team, 2020] con el método `read_spss` que depende del segundo paquete pyreadstat para su lectura.

Con el método programado `spss_a_csv` recorro todos los archivos con extensión .sav dentro de un directorio establecido por el constructor, para posteriormente juntar los datos de los años disponibles en un solo dataframe, estructura de datos de pandas; generando un solo archivo con extensión .csv denominado DATOS.csv.

Posterior a la unificación de datos, se tienen un total de `1174 Registros (filas)` en un archivo csv, este paso es con el fin evitar releer y juntar nuevamente toda la base de datos cada vez que el programa entre en ejecución.

4.3. Descripción de los datos

De los datos agrupados, se tienen n columnas, no obstante, para el desarrollo de la presente tesis, únicamente contemplare las columnas más representativas, acorde a la primera valoración, puesto que las demás columnas contienen el seguimiento de los pacientes, valores psicológicos que ya no se registran entre otros factores. A continuación se expone una muestra de los datos a preprocesar y analizar.

Tabla 4.1: Dato Muestra

Registro	Peso	Estatura	Edad	IMC	EATTRASTORNO	Diagdsmv
79	51.5	1.57	31	20.89	9	SIN TCA

La columna Diagdsmv contiene el diagnóstico del paciente acorde al manual diagnóstico y estadístico de los trastornos mentales, quinta edición (DSM-V). En la tabla 4.2, se presentan las abreviaturas mostradas y diagnósticos a analizar, de la columna diagsmv; se excluyeron aquellos diagnósticos particulares que tienen muy poca frecuencia.

EATTRASTORNO corresponde al puntaje interno numérico de cada paciente, posterior a su valoración nutricional y psiquiátrica; a priori entre mayor sea el puntaje, mayor posibilidad de tener un trastorno de la conducta alimentaria.

Tabla 4.2: Abreviaturas y diagnósticos TCA

Abreviatura	Diagnóstico
AN	Anorexia Nerviosa
AN-P	Anorexia Purgativa
AN-R	Anorexia Restrictiva
BN	Bulimia Nerviosa
CAR	Conductas Alimentarias de Riesgo
OTAES	Otros Trastornos de Alimentación Especificados
TPA	Trastorno Por Atracción
TANE	Trastornos de Alimentación Especificados
SIN TCA	Sin Trastornos de la Conducta Alimentaria

4.4. Preprocesamiento

Para la preparación de los datos, se realizaron las siguientes acciones, selecciones, validaciones y transformaciones nuevamente con el paquete `pandas` [pandas development team, 2020]. A partir de este momento y durante toda esta sección, se hará referencia como etapa a cada paso del preprocesamiento.

1. Selección de datos con diagnóstico en Tabla 4.2.
2. Exclusión de datos nulos, indefinidos, sin valor o 0 en cualquier columna de la Tabla 4.3.
3. Revisión de datos incoherentes en mínimos y máximos.
4. Transformación de estaturas en centímetros a metros.
5. Validación de Índice de Masa Corporal (IMC).
6. Binarización de los diagnósticos.
7. Normalización de datos numéricos.

En la etapa número tres, se encontraron los datos incoherentes, al imprimir en salida estándar el resultado producido al llamar el método `describe` de pandas, perteneciente a la clase `DataFrame`, se observan datos estadísticos relevantes, entre ellos, los mínimos y máximos de cada columna; se observó en la columna PESO un valor mínimo de 1.5, peso que además de estar muy por debajo de la media, es incongruente al sentido común, se solucionó al esclarecer el dato con la clínica y se realizó el ajuste manual correspondiente; en la columna ESTATURA se encontró que el máximo dato era 173 cuando en su mayoría los datos están en metros, se corroboraron con la clínica y se transformaron a los datos correctos.

Posterior a ellos, los diagnósticos pasaron por un proceso de binarización y de normalización, aplicado del modo expuesto y debido a las razones planteadas en el marco teórico.

En la Tabla 4.3 resumo el número los datos tratados y perdidos, así como sus respectivos porcentajes por las etapas de la 1 a la 5.

Tabla 4.3: Etapas de Preprocesamiento

Etapa	Datos Iniciales	Datos Resultantes	Datos corregidos	Pérdida Relativa	Pérdida Absoluta
1	1174	1034	0	11.93 %	11.93 %
2	1034	985	0	4.74 %	16.09 %
3	985	985	2	0 %	16.09 %
4	985	985	8	0 %	16.09 %
5	985	819	0	16.85 %	30.23 %

Capítulo 5

Procesamiento y Análisis de Resultados

Con los datos preprocesados y listos para su procesamiento, se expone en el presente capítulo, la aplicación, así como el análisis de resultados producidos por el/los algoritmo(s) de agrupamiento y clasificación.

5.1. Agrupamiento - K-Means

Se procesaron los datos con el algoritmo de KMeans del paquete SKLearn [Pedregosa et al., 2011], módulo cluster, como se puede observar en la Tabla 4.2, se tienen nueve diagnósticos, de modo que se inicializó el algoritmo para nueve grupos. Idealmente se buscó que cada grupo contuviera solo registros con el mismo diagnóstico o en un porcentaje cercano al 100.

Previo a los resultados detallados, se graficaron los grupos en un plano bidimensional con el paquete Matplotlib [Hunter, 2007], aplicando Análisis de Componentes Principales para reducir la dimensionalidad a dos utilizando el paquete SKLearn [Pedregosa et al., 2011], módulo decomposition.

Como se puede observar en la Figura 5.1, los nueve grupos contienen diferentes diagnósticos en distintas proporciones y si bien en la gráfica 5.1 algunos registros de grupos están significativamente juntos y mezclados al de otros grupos, e incluso con registros muy cerca de estar sobrepuestos; no

deja de ser un plano bidimensional, lo que no significa que los resultados por clasificación vayan a ser poco óptimos.

Los grupos de las tablas 5.2 y 5.9 tienen un diagnóstico predominante. Sin embargo en los demás grupos, no se puede concluir con certeza un resultado, si un nuevo dato llegará a caer en estos grupos ¿Cuál sería su diagnóstico express?; más allá de mostrarle las probabilidades respecto a los porcentajes.

Tabla 5.1: Resultados - Agrupamiento - KMeans - Grupo 1

Diagnóstico	No. de Registros	Porcentaje	Acumulado
AN-R	62	37.8 %	62
CAR	30	18.29 %	92
SIN TCA	19	11.59 %	111
BN	15	9.15 %	126
OTAE1 AN-Atípica	12	7.32 %	138
AN-P	11	6.71 %	149
OTAE2 BN-bajafrecuencia	7	4.27 %	156
TANE	7	4.27 %	163
TPA	1	0.61 %	164

Tabla 5.2: Resultados - Agrupamiento - KMeans - Grupo 2

Diagnóstico	No. de Registros	Porcentaje	Acumulado
TPA	24	64.86 %	24
CAR	5	13.51 %	29
SIN TCA	3	8.11 %	32
TANE	2	5.41 %	34
AN-P	1	2.7 %	35
BN	1	2.7 %	36
OTAE2 BN-bajafrecuencia	1	2.7 %	37

Tabla 5.3: Resultados - Agrupamiento - KMeans - Grupo 3

Diagnóstico	No. de Registros	Porcentaje	Acumulado
AN-R	46	39.66 %	46
BN	27	23.28 %	73
AN-P	18	15.52 %	91
OTAE1 AN-Atípica	17	14.66 %	108
TANE	4	3.45 %	112
OTAE2 BN-bajafrecuencia	3	2.59 %	115
SIN TCA	1	0.86 %	116

Tabla 5.4: Resultados - Agrupamiento - KMeans - Grupo 4

Diagnóstico	No. de Registros	Porcentaje	Acumulado
BN	18	21.69 %	18
OTAE1 AN-Atípica	14	16.87 %	32
AN-R	12	14.46 %	44
CAR	10	12.05 %	54
SIN TCA	9	10.84 %	63
OTAE2 BN-bajafrecuencia	8	9.64 %	71
TANE	7	8.43 %	78
TPA	3	3.61 %	81
AN-P	2	2.41 %	83

Tabla 5.5: Resultados - Agrupamiento - KMeans - Grupo 5

Diagnóstico	No. de Registros	Porcentaje	Acumulado
BN	65	48.87 %	65
OTAE1 AN-Atípica	17	12.78 %	82
AN-R	15	11.28 %	97
AN-P	10	7.52 %	107
OTAE2 BN-bajafrecuencia	10	7.52 %	117
TPA	9	6.77 %	126
TANE	6	4.51 %	132
CAR	1	0.75 %	133

Figura 5.1: Gráfica - Agrupamiento por KMeans.

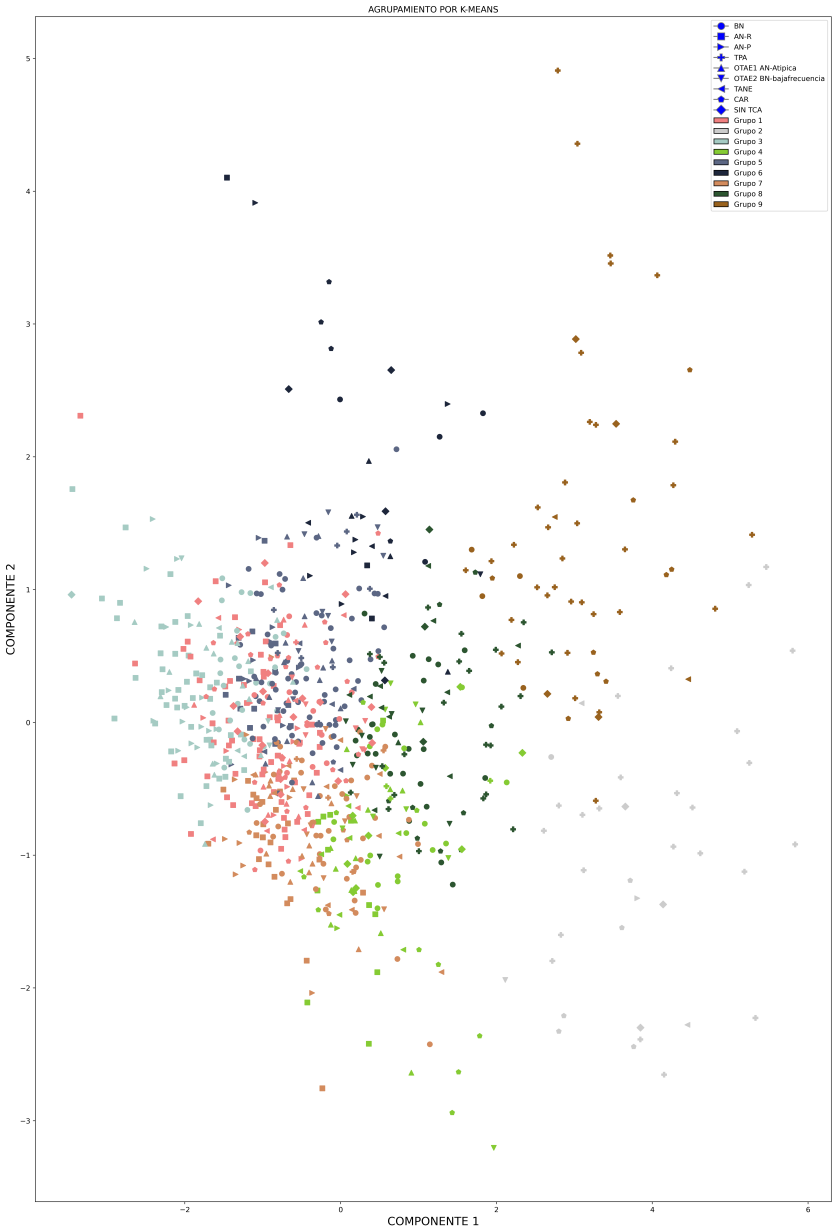


Tabla 5.6: Resultados - Agrupamiento - KMeans - Grupo 6

Diagnóstico	No. de Registros	Porcentaje	Acumulado
AN-P	7	23.33 %	7
BN	4	13.33 %	11
CAR	4	13.33 %	15
OTAE1 AN-Atípica	4	13.33 %	19
SIN TCA	4	13.33 %	23
AN-R	3	10.00 %	26
TANE	3	10.00 %	29
OTAE2 BN-bajafrecuencia	1	3.33 %	30

Tabla 5.7: Resultados - Agrupamiento - KMeans - Grupo 7

Diagnóstico	No. de Registros	Porcentaje	Acumulado
BN	43	36.13 %	43
AN-R	24	20.17 %	67
OTAE1 AN-Atípica	14	11.76 %	81
TANE	11	9.24 %	92
AN-P	10	8.4 %	102
OTAE2 BN-bajafrecuencia	8	6.72 %	110
CAR	5	4.2 %	115
TPA	4	3.36 %	119

Tabla 5.8: Resultados - Agrupamiento - KMeans - Grupo 8

Diagnóstico	No. de Registros	Porcentaje	Acumulado
BN	26	31.33 %	26
TPA	24	28.92 %	50
TANE	11	13.25 %	61
CAR	9	10.84 %	70
OTAE2 BN-bajafrecuencia	9	10.84 %	79
SIN TCA	3	3.61 %	82
OTAE1 AN-Atípica	1	1.2 %	83

Tabla 5.9: Resultados - Agrupamiento - KMeans - Grupo 9

Diagnóstico	No. de Registros	Porcentaje	Acumulado
TPA	35	64.81 %	35
CAR	9	16.67 %	44
BN	4	7.41 %	48
SIN TCA	4	7.41 %	52
TANE	2	3.7 %	54

5.2. Clasificación

5.2.1. Árbol de Decisión

La técnica: árboles de decisión, inaugura el análisis con algoritmos de clasificación. Los datos nuevamente fueron procesados con el paquete SKLearn [Pedregosa et al., 2011]; El módulo tree contiene la clase DecisionTreeClassifier que alberga dicho algoritmo.

En contraste del procesamiento con KMeans, únicamente se aplicó validación cruzada para el entrenamiento. Puesto que; si todos los datos se utilizaren para entrenar y posteriormente los mismos probar el modelo entrenado, se obtendría una precisión aparente del 100 %. En tal caso, no se podría validar y evaluar el rendimiento del modelo en nuevos casos; por los límites de esta tesis.

Asimismo, solo resta determinar el número de particiones (segmentos) más apropiados y si es conveniente aleatorizarlos, no obstante, la respuesta no es inmediata. Para obtener el mejor resultado posible se realizaron pruebas desde dos hasta nueve particiones.

Añadiendo, los resultados varían dependiendo del momento que se ejecute el algoritmo; debido a su aleatoriedad inicial. Se realizaron 500 pruebas por cada número de partición. En las Tablas 5.10 y 5.11 se resumen los resultados recolectados; el promedio, mínimo y máximo de cada una de las 500 pruebas se exponen tanto con validación cruzada con aleatoriedady sin aleatoriedad.

Analizando los resultados, se confirma que el realizar múltiples pruebas es altamente relevante; aunque pudiera ser exagerado realizar cuatro mil pruebas. Además, notemos como algunos de los mejores resultados, se arrojaron acercándose a la prueba 500. Como se puede observar, existe una diferencia de más de diez puntos porcentuales entre una ejecución y otra.

El mejor resultado absoluto obtenido exhibido en la Tabla 5.12, fue aplicando validación cruzada seleccionando aleatoriamente los datos para formar

Tabla 5.10: Resultados - Generales - Clasificación - Árbol de Decisión - CON aleatoriedad

No. de Particiones	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
2	31.70 %	36.82 %	43.03 %	2	435
3	33.33 %	38.18 %	43.58 %	3	487
4	33.65 %	39.35 %	47.80 %	2	253
5	34.14 %	40.27 %	46.62 %	5	193
6	35.03 %	41.45 %	53.28 %	1	128
7	36.75 %	42.14 %	51.28 %	5	341
8	36.89 %	43.38 %	54.90 %	7	218
9	37.36 %	43.90 %	53.84 %	7	332

Tabla 5.11: Resultados - Precisiones Generales - Clasificación - Árbol de Decisión - SIN aleatoriedad

No. de Particiones	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
2	33.17 %	35.89 %	39.02 %	1	36
3	32.60 %	35.11 %	38.09 %	2	223
4	34.31 %	37.47 %	40.68 %	4	444
5	34.75 %	39.54 %	44.17 %	5	4
6	35.03 %	38.06 %	41.60 %	2	451
7	35.04 %	39.77 %	47.00 %	6	193
8	36.89 %	42.36 %	47.57 %	3	500
9	36.26 %	41.46 %	49.45 %	5	66

los segmentos; con una precisión del 54.90 %, en 8 particiones. No son los resultados esperados, sin embargo, los diagnósticos AN-P, AN-R y TPA tienen una precisión aceptable.

Tabla 5.12: Resultados - Precisiones Particulares del Mejor Resultado General de la Tabla 5.10

Diagnóstico	Precisión	Número de Datos probados
AN-P	67 %	5
AN-R	79 %	22
BN	58 %	26
CAR	45 %	11
OTAE1 AN-Atípica	12 %	6
OTAE2 BN-bajafrecuencia	60 %	8
SIN TCA	22 %	4
TANE	29 %	8
TPA	89 %	12

Tabla 5.13: Resultados - Precisiones Particulares del Mejor Resultado General de la Tabla 5.11

Diagnóstico	Precisión	Número de Datos probados
AN-P	33 %	4
AN-R	67 %	20
BN	50 %	21
CAR	30 %	8
OTAE1 AN-Atípica	17 %	5
OTAE2 BN-bajafrecuencia	0 %	9
SIN TCA	100 %	4
TANE	0 %	8
TPA	73 %	14

5.2.2. Redes Neuronales

Si bien el paquete Scikit Learn [Pedregosa et al., 2011] contiene una clase para aplicar Redes Neuronales, se optó por utilizar el paquete especializado Tensorflow [Abadi et al., 2015].

La pregunta (parafraseada): ¿Cuál será su esqueleto? es recurrente al aplicar redes neuronales; cuantas neuronas habrá en la capa de entrada, de salida y en capas ocultas; además del número de éstas. En la capa de entrada, el número de entradas es análogo al número de datos (columnas); es decir aquellos en la Tabla 4.1, restando el registro. Para la capa de salida, se tendrán 9 neuronas que son el número de diagnósticos en su forma binarizada.

Para determinar el número de capas ocultas y número de neuronas por cada capa, propuse 16 estructuras que fueron puestas a prueba cada una, cinco veces, con validación cruzada de 2 a 9 particiones; tanto de manera aleatorizada como no aleatorizada. Debido al alta de carga de procesamiento requerida, no se realizaron 500 pruebas como en el Árbol de decisión; al ser 16 redes puestas a prueba 5 veces, resultando en 80 ejecuciones generales, donde cada una contiene 44 pruebas particulares obtenidas sumando las pruebas con particiones $(2 + 3 + ... + 9)$. Con un subtotal $3,520$ pruebas para validación cruzada aleatorizada; en total serían $7,040$ pruebas.

Ejecutar dichas pruebas de forma secuencial (aun siendo solo 5), tardarían más de una semana. Para acelerar la etapa de pruebas, se optó por correrlas de forma concurrente. De este modo, se obtiene un beneficio al utilizar los 6 núcleos y 12 hilos del procesador. Teóricamente se podrán realizarían 12 pruebas simultáneamente. Para lograrlo, me apoye del programa GNU Parallel [Tange, 2022]; se responsabiliza la ejecución concurrente, sin realizar muchos cambios adicionales al programa de pruebas y ninguno al paquete TCA.

El programa de pruebas versión 3.2 para redes neuronales, se llama tca-pruebas-v3.2. La versión 3 corresponde a que se probarán redes neuronales, y la subversión determina el modo de aplicación en la validación cruzada; siendo la número 2 con particiones aleatorias y se invoca de la siguiente manera:

python3 tca-pruebas-v3.2.py [número de pruebas] [número de particiones] [capa 1] .. [capa n]

Ejemplo: Realizar dos pruebas para una red neuronal con tres capas ocultas; cada una con 200 neuronas para validar con 8 particiones. El programa se invoca de la siguiente manera:

python3 tca-pruebas-v3.2.py 2 8 200 200 200

Con lo anterior se llama a parallel de la siguiente forma:

time parallel -k –bar python3 tca-pruebas-v3.2.py 5 :::: particiones :::: capas

Primeramente, se sumó time para conocer el tiempo de ejecución, posteriormente parallel para invocarlo con los argumentos: -k para ordenar la salida; puesto que al ser concurrente, el orden de ejecución de las pruebas es no determinista, –bar para mostrar una barra de progreso en la salida estándar. Después se indica el programa a correr en paralelo así como sus argumentos fijos (python3 tca-pruebas-v3.2.py 5) para con los dos puntos : por cuatro, indicar los argumentos variables de entrada para el programa a ejecutar; dichos argumentos están definidos en un archivo y al ser dos grupos de entradas, se realiza el producto cartesiano de ellos.

Finalmente, un total de 12 horas con 20 minutos y 20 segundos terminar las $3,520$ con validación cruzada con aleatoriedad y un total de 12 horas con 03 minutos y 11 segundos terminar las $3,520$ con validación cruzada sin aleatoriedad; fue el tiempo de ejecución. De la Tabla 5.14 a la Tabla 5.29, se exponen los resultados obtenidos generales. Debido a la ejecución únicamente de 5 pruebas, a diferencia de las tablas de Árbol de decisión, se omitió la columna de Mejor Prueba.

El mejor resultado que aplicó validación cruzada fue seleccionando aleatoriamente los datos para formar los segmentos; con una precisión del 91.69 % en 2 particiones. En la Tabla 5.31 se exhiben dichos resultados; de los 9 diagnósticos: 5 tienen una precisión superior al 90 % de los cuales 3 mayor al 95 % y solo uno por debajo del 80 % por un punto porcentual.

Tabla 5.14: Resultados - Generales - Clasificación - Red Neuronal - 2 Particiones - CON aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	34.47 %	35.50 %	37.32 %	1	1
100	34.96 %	37.05 %	38.88 %	2	2
350	38.14 %	39.70 %	40.73 %	1	4
50 50	35.21 %	38.97 %	41.81 %	2	5
50 100	36.83 %	40.04 %	43.66 %	1	5
100 50	36.19 %	39.39 %	42.3 %	2	2
100 100	39.36 %	41.77 %	43.9 %	1	3
350 350	48.29 %	51.39 %	55.01 %	2	1
50 50 50	39.27 %	41.41 %	43.77 %	2	1
50 100 50	45.85 %	47.44 %	49.02 %	1	1
50 50 150	45.37 %	48.78 %	52.68 %	1	4
150 150 100	53.17 %	55.25 %	56.34 %	1	4
150 50 150	50.00 %	52.76 %	54.39 %	1	1
200 200 200	49.02 %	62.68 %	86.31 %	2	1
350 350 350	53.66 %	69.85 %	89.76 %	1	3
350 200 150 100	53.41 %	69.22 %	91.69 %	2	1

Si se optará por no aplicar validación cruzada, apoyarse de todos los datos tanto para el entrenamiento del modelo como de su evaluación; se obtendría un efecto similar. Con base en los resultados recopilados en la Tabla 5.30, el mejor resultado como se puede observar es una precisión general del 92.8 %. Aquella precisión se traduce en ligeras mejoras para algunos diagnósticos; siendo SIN TCA el más beneficiado con un 100 % de aciertos sobre 43 casos. Por otra parte, TANE bajó cinco puntos porcentuales.

Por último, cabe aclarar, que pudiera haber ligeras discrepancias entre el resultado general y si se calculara manualmente el resultado general de los resultados particulares, esto porque para los resultados particulares, se documentó lo arrojado por el método classification_report del módulo metrics del paquete SKLearn. En cambio para el resultado general, se documentó el resultado arrojado por el método accuracy_score del mismo módulo y paquete. Además de por supuesto, la pérdida de precisión al redondear a dos decimales.

Tabla 5.15: Resultados - Generales - Clasificación - Red Neuronal - 3 Particiones - CON aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	35.90 %	37.88 %	40.29 %	2	3
100	37.00 %	38.75 %	40.66 %	2	1
350	35.16 %	39.19 %	41.39 %	2	2
50 50	38.46 %	40.73 %	42.12 %	2	3
50 100	39.93 %	42.12 %	44.69 %	2	4
100 50	41.76 %	43.08 %	44.32 %	3	3
100 100	42.49 %	44.18 %	46.52 %	3	5
350 350	46.89 %	49.38 %	53.85 %	1	1
50 50 50	42.12 %	44.25 %	45.42 %	2	1
50 100 50	47.62 %	50.11 %	54.21 %	3	5
50 50 150	47.25 %	51.43 %	54.58 %	1	2
150 150 100	52.01 %	53.41 %	55.68 %	1	2
150 50 150	48.35 %	52.09 %	54.58 %	1	2
200 200 200	52.38 %	54.73 %	57.88 %	1	1
350 350 350	54.58 %	56.92 %	59.34 %	3	4
350 200 150 100	46.89 %	52.45 %	58.24 %	3	1

Tabla 5.16: Resultados - Generales - Clasificación - Red Neuronal - 4 Particiones - CON aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	37.56 %	39.32 %	42.93 %	3	3
100	38.05 %	38.38 %	39.02 %	2	2
350	37.25 %	39.69 %	41.95 %	1	2
50 50	38.24 %	41.89 %	46.34 %	3	4
50 100	39.02 %	42.44 %	47.32 %	1	1
100 50	41.67 %	43.89 %	49.76 %	1	4
100 100	43.14 %	46.23 %	53.17 %	3	2
350 350	52.68 %	53.81 %	56.59 %	3	2
50 50 50	43.90 %	47.12 %	50.98 %	4	1
50 100 50	47.80 %	49.22 %	50.73 %	3	5
50 50 150	50.24 %	52.39 %	55.12 %	1	1
150 150 100	57.35 %	59.08 %	61.46 %	2	5
150 50 150	48.78 %	53.27 %	55.61 %	1	1
200 200 200	56.10 %	61.00 %	64.88 %	2	4
350 350 350	59.80 %	62.07 %	64.39 %	1	5
350 200 150 100	54.63 %	57.33 %	60.29 %	4	1

Tabla 5.17: Resultados - Generales - Clasificación - Red Neuronal - 5 Particiones - CON aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	35.98 %	39.00 %	42.68 %	1	2
100	38.65 %	41.39 %	45.73 %	1	5
350	40.24 %	44.56 %	48.17 %	1	5
50 50	39.63 %	43.16 %	46.63 %	5	1
50 100	42.68 %	45.61 %	49.39 %	1	4
100 50	42.07 %	44.86 %	48.78 %	2	1
100 100	41.72 %	46.21 %	49.39 %	2	3
350 350	53.66 %	57.15 %	60.74 %	5	4
50 50 50	43.90 %	46.15 %	47.85 %	5	4
50 100 50	50.61 %	52.93 %	54.27 %	1	3
50 50 150	51.83 %	55.32 %	60.12 %	5	5
150 150 100	59.15 %	60.93 %	62.8 %	1	5
150 50 150	53.66 %	57.02 %	59.76 %	3	4
200 200 200	57.67 %	60.92 %	64.02 %	1	3
350 350 350	59.76 %	62.56 %	68.29 %	3	1
350 200 150 100	58.54 %	60.76 %	65.03 %	5	2

Tabla 5.18: Resultados - Generales - Clasificación - Red Neuronal - 6 Particiones - CON aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	36.76 %	39.45 %	43.38 %	6	2
100	38.69 %	40.26 %	41.18 %	5	2
350	42.65 %	44.34 %	47.45 %	1	4
50 50	41.18 %	44.29 %	47.45 %	3	2
50 100	43.80 %	45.68 %	48.18 %	2	5
100 50	43.80 %	45.97 %	48.91 %	2	4
100 100	44.53 %	47.88 %	55.15 %	4	4
350 350	54.74 %	58.86 %	61.03 %	4	1
50 50 50	47.06 %	51.47 %	57.35 %	6	4
50 100 50	51.09 %	51.90 %	53.28 %	2	3
50 50 150	54.74 %	55.56 %	57.35 %	6	2
150 150 100	58.39 %	61.11 %	64.23 %	2	4
150 50 150	55.88 %	58.85 %	59.85 %	2	1
200 200 200	58.82 %	63.10 %	67.65 %	4	1
350 350 350	58.39 %	59.59 %	61.31 %	1	5
350 200 150 100	56.62 %	58.80 %	62.04 %	1	2

Tabla 5.19: Resultados - Generales - Clasificación - Red Neuronal - 7 Particiones - CON aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	37.61 %	41.54 %	46.15 %	7	5
100	38.46 %	42.74 %	46.15 %	3	3
350	42.74 %	44.79 %	47.86 %	1	2
50 50	41.88 %	43.93 %	45.3 %	1	3
50 100	42.74 %	45.98 %	48.72 %	3	4
100 50	43.59 %	47.01 %	49.57 %	6	1
100 100	43.59 %	48.38 %	52.99 %	5	4
350 350	58.12 %	59.49 %	61.54 %	4	4
50 50 50	47.86 %	49.06 %	50.43 %	4	3
50 100 50	52.99 %	54.53 %	58.97 %	2	5
50 50 150	54.70 %	56.07 %	58.12 %	5	4
150 150 100	58.97 %	62.05 %	67.52 %	3	2
150 50 150	56.41 %	59.83 %	63.25 %	1	3
200 200 200	61.54 %	62.74 %	63.25 %	1	2
350 350 350	62.39 %	65.64 %	71.79 %	6	2
350 200 150 100	58.12 %	64.96 %	74.36 %	6	2

Tabla 5.20: Resultados - Generales - Clasificación - Red Neuronal - 8 Particiones - CON aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	41.18 %	43.08 %	43.69 %	2	2
100	42.72 %	43.55 %	45.63 %	3	2
350	42.16 %	43.75 %	47.06 %	7	2
50 50	43.14 %	46.28 %	50.49 %	2	1
50 100	43.14 %	49.02 %	51.96 %	8	4
100 50	42.16 %	46.00 %	48.54 %	2	5
100 100	45.63 %	50.21 %	55.88 %	8	1
350 350	54.37 %	57.52 %	63.73 %	5	4
50 50 50	46.08 %	48.82 %	52.43 %	3	1
50 100 50	53.40 %	55.86 %	57.84 %	5	1
50 50 150	49.02 %	54.10 %	56.86 %	5	2
150 150 100	60.78 %	63.08 %	66.02 %	2	5
150 50 150	58.25 %	61.14 %	65.69 %	8	4
200 200 200	61.17 %	63.41 %	66.67 %	7	4
350 350 350	63.11 %	66.41 %	69.61 %	8	2
350 200 150 100	57.84 %	63.68 %	70.59 %	6	1

Tabla 5.21: Resultados - Generales - Clasificación - Red Neuronal - 9 Particiones - CON aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	39.56 %	41.32 %	43.96 %	6	4
100	39.56 %	45.05 %	48.35 %	1	2
350	42.86 %	46.15 %	48.35 %	7	3
50 50	43.96 %	48.57 %	51.65 %	5	3
50 100	46.15 %	48.79 %	50.55 %	7	2
100 50	43.96 %	46.59 %	51.65 %	5	3
100 100	49.45 %	52.75 %	58.24 %	6	4
350 350	56.04 %	58.90 %	60.44 %	6	2
50 50 50	47.25 %	51.21 %	53.85 %	6	2
50 100 50	51.65 %	55.60 %	60.44 %	7	2
50 50 150	53.85 %	55.82 %	57.14 %	1	2
150 150 100	61.54 %	65.27 %	68.13 %	6	3
150 50 150	56.04 %	63.74 %	69.23 %	7	3
200 200 200	63.74 %	65.49 %	69.23 %	9	3
350 350 350	62.64 %	64.18 %	64.84 %	4	1
350 200 150 100	59.34 %	64.40 %	68.13 %	6	2

Tabla 5.22: Resultados - Generales - Clasificación - Red Neuronal - 2 Particiones - SIN aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	34.96 %	36.04 %	37.41 %	2	3
100	35.45 %	36.77 %	37.9 %	2	4
350	37.90 %	38.66 %	39.61 %	2	5
50 50	35.37 %	37.44 %	38.63 %	2	2
50 100	37.90 %	39.74 %	41.81 %	2	5
100 50	37.80 %	38.80 %	39.51 %	1	3
100 100	37.90 %	40.87 %	44.39 %	1	3
350 350	47.80 %	49.63 %	51.83 %	2	2
50 50 50	41.81 %	43.45 %	45.97 %	2	2
50 100 50	44.01 %	47.73 %	50.86 %	2	1
50 50 150	47.43 %	49.00 %	50.37 %	2	1
150 150 100	51.10 %	54.69 %	59.66 %	2	3
150 50 150	48.66 %	52.20 %	54.03 %	2	3
200 200 200	55.26 %	56.74 %	57.95 %	2	1
350 350 350	53.06 %	56.33 %	59.17 %	2	5
350 200 150 100	53.90 %	59.72 %	63.66 %	1	1

Tabla 5.23: Resultados - Generales - Clasificación - Red Neuronal - 3 Particiones - SIN aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	34.43 %	35.82 %	36.63 %	3	1
100	35.16 %	36.41 %	38.46 %	2	1
350	37.36 %	38.61 %	40.66 %	2	1
50 50	36.26 %	39.12 %	40.66 %	2	5
50 100	39.19 %	42.12 %	45.42 %	2	3
100 50	38.83 %	40.81 %	42.49 %	2	3
100 100	39.56 %	42.86 %	46.15 %	2	2
350 350	46.15 %	48.50 %	50.18 %	2	5
50 50 50	43.22 %	44.47 %	45.42 %	3	3
50 100 50	45.42 %	46.81 %	49.08 %	3	2
50 50 150	47.62 %	49.38 %	51.28 %	2	3
150 150 100	51.65 %	53.92 %	56.78 %	2	2
150 50 150	50.55 %	52.97 %	58.24 %	3	4
200 200 200	50.92 %	54.29 %	58.97 %	2	5
350 350 350	49.45 %	53.77 %	58.97 %	3	1
350 200 150 100	50.55 %	53.26 %	57.51 %	3	1

Tabla 5.24: Resultados - Generales - Clasificación - Red Neuronal - 4 Particiones - SIN aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	37.25 %	38.87 %	40.0 %	3	2
100	38.54 %	39.26 %	40.0 %	3	1
350	39.22 %	40.04 %	40.49 %	3	2
50 50	41.46 %	42.15 %	42.93 %	2	4
50 100	40.98 %	43.26 %	45.59 %	4	5
100 50	41.46 %	43.41 %	48.78 %	2	5
100 100	43.14 %	44.92 %	47.32 %	2	2
350 350	53.17 %	54.40 %	56.10 %	3	3
50 50 50	41.95 %	45.50 %	48.29 %	3	4
50 100 50	46.34 %	50.30 %	55.39 %	4	3
50 50 150	50.73 %	51.86 %	53.92 %	4	1
150 150 100	56.37 %	57.71 %	60.49 %	2	2
150 50 150	50.98 %	54.05 %	58.54 %	2	3
200 200 200	56.37 %	59.49 %	63.41 %	2	3
350 350 350	55.12 %	57.91 %	59.51 %	3	2
350 200 150 100	53.66 %	56.21 %	60.00 %	2	5

Tabla 5.25: Resultados - Generales - Clasificación - Red Neuronal - 5 Particiones - SIN aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	37.20 %	39.73 %	43.56 %	5	5
100	38.04 %	40.14 %	41.46 %	3	1
350	39.63 %	41.76 %	45.12 %	3	2
50 50	42.07 %	44.57 %	46.01 %	5	3
50 100	44.51 %	45.97 %	47.85 %	5	4
100 50	41.72 %	44.01 %	46.01 %	5	3
100 100	43.29 %	46.46 %	48.78 %	2	1
350 350	51.22 %	54.23 %	57.06 %	5	1
50 50 50	46.95 %	49.76 %	51.83 %	3	2
50 100 50	47.24 %	51.04 %	54.6 %	5	2
50 50 150	49.39 %	53.43 %	56.44 %	5	5
150 150 100	55.21 %	59.07 %	61.35 %	5	2
150 50 150	53.66 %	56.66 %	60.37 %	3	2
200 200 200	57.06 %	59.83 %	64.02 %	3	2
350 350 350	57.93 %	61.46 %	65.24 %	3	2
350 200 150 100	57.32 %	58.31 %	59.51 %	5	3

Tabla 5.26: Resultados - Generales - Clasificación - Red Neuronal - 6 Particiones - SIN aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	37.50 %	40.74 %	43.38 %	4	2
100	39.71 %	40.74 %	41.91 %	4	4
350	40.44 %	41.85 %	43.38 %	4	1
50 50	41.18 %	43.76 %	46.32 %	6	3
50 100	42.34 %	43.70 %	44.85 %	6	1
100 50	42.34 %	43.84 %	45.99 %	3	4
100 100	46.72 %	47.96 %	50.00 %	6	4
350 350	51.47 %	55.66 %	61.03 %	5	3
50 50 50	45.99 %	48.83 %	50.00 %	6	1
50 100 50	50.00 %	52.49 %	56.93 %	3	2
50 50 150	53.68 %	55.41 %	56.93 %	3	1
150 150 100	57.35 %	59.09 %	62.50 %	5	2
150 50 150	54.41 %	55.95 %	58.82 %	5	5
200 200 200	55.88 %	60.29 %	62.50 %	6	3
350 350 350	60.29 %	63.39 %	67.15 %	3	1
350 200 150 100	56.62 %	59.18 %	60.58 %	3	1

Tabla 5.27: Resultados - Generales - Clasificación - Red Neuronal - 7 Particiones - SIN aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	40.17 %	42.22 %	45.30 %	4	2
100	41.03 %	42.74 %	43.59 %	6	3
350	41.03 %	44.27 %	47.01 %	6	5
50 50	39.32 %	42.39 %	44.44 %	4	4
50 100	43.59 %	46.50 %	47.86 %	4	1
100 50	42.74 %	45.81 %	48.72 %	4	3
100 100	42.74 %	48.21 %	51.28 %	4	5
350 350	54.70 %	57.95 %	61.54 %	3	3
50 50 50	47.86 %	50.60 %	53.85 %	4	3
50 100 50	52.99 %	55.73 %	58.97 %	4	2
50 50 150	55.56 %	57.44 %	59.83 %	3	3
150 150 100	58.97 %	63.59 %	70.94 %	4	3
150 50 150	53.85 %	59.66 %	64.96 %	4	1
200 200 200	63.25 %	64.79 %	65.81 %	4	3
350 350 350	64.96 %	66.84 %	68.38 %	3	3
350 200 150 100	58.97 %	61.03 %	63.25 %	4	5

Tabla 5.28: Resultados - Generales - Clasificación - Red Neuronal - 8 Particiones - SIN aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	43.14 %	44.51 %	47.06 %	5	1
100	41.18 %	43.14 %	44.12 %	5	2
350	44.12 %	45.69 %	49.02 %	5	5
50 50	45.10 %	47.85 %	50.98 %	5	3
50 100	44.66 %	47.18 %	50.49 %	3	3
100 50	44.12 %	47.66 %	52.43 %	3	3
100 100	45.63 %	49.13 %	51.46 %	3	1
350 350	56.86 %	58.39 %	62.14 %	3	2
50 50 50	48.04 %	50.98 %	53.92 %	4	5
50 100 50	52.43 %	53.99 %	58.25 %	3	3
50 50 150	54.37 %	57.67 %	62.14 %	3	2
150 150 100	58.82 %	62.88 %	68.93 %	3	5
150 50 150	56.86 %	60.23 %	63.11 %	3	2
200 200 200	62.75 %	65.49 %	69.90 %	3	4
350 350 350	59.80 %	61.13 %	65.05 %	3	2
350 200 150 100	59.22 %	63.28 %	67.96 %	3	4

Tabla 5.29: Resultados - Generales - Clasificación - Red Neuronal - 9 Particiones - SIN aleatoriedad

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Partición	Mejor Prueba
50	43.96 %	47.03 %	50.55 %	5	4
100	43.96 %	46.15 %	47.25 %	8	3
350	48.35 %	50.55 %	52.75 %	5	1
50 50	43.96 %	45.71 %	48.35 %	8	5
50 100	45.05 %	49.45 %	53.85 %	5	1
100 50	47.25 %	48.79 %	50.55 %	8	3
100 100	48.35 %	49.23 %	50.55 %	8	1
350 350	59.34 %	61.76 %	64.84 %	5	1
50 50 50	49.45 %	51.21 %	52.75 %	8	3
50 100 50	50.55 %	53.63 %	56.04 %	8	4
50 50 150	52.75 %	58.68 %	65.93 %	5	4
150 150 100	62.64 %	64.40 %	65.93 %	5	2
150 50 150	58.24 %	60.22 %	62.64 %	8	5
200 200 200	67.03 %	69.23 %	70.33 %	5	1
350 350 350	63.74 %	66.59 %	70.33 %	8	1
350 200 150 100	64.84 %	68.13 %	71.43 %	5	5

Tabla 5.30: Resultados - Generales - Clasificación - Red Neuronal - SIN Validación Cruzada

Capas Ocultas	Mínimo	Promedio	Máximo	Mejor Prueba
50	37.48 %	38.19 %	38.71 %	2
100	41.64 %	42.25 %	42.86 %	1
350	47.50 %	50.26 %	52.14 %	3
50 50	52.63 %	55.53 %	58.61 %	2
50 100	61.05 %	62.74 %	65.08 %	2
100 50	54.82 %	59.15 %	62.15 %	5
100 100	64.10 %	66.86 %	69.23 %	3
350 350	84.37 %	86.15 %	87.18 %	4
50 50 50	63.13 %	70.11 %	73.87 %	1
50 100 50	74.48 %	77.85 %	80.59 %	3
50 50 150	81.07 %	82.83 %	83.88 %	4
150 150 100	87.55 %	88.89 %	91.33 %	5
150 50 150	82.17 %	85.05 %	87.30 %	1
200 200 200	87.67 %	89.06 %	90.72 %	4
350 350 350	69.60 %	85.42 %	92.06 %	5
350 200 150 100	88.16 %	90.26 %	92.80 %	2

Tabla 5.31: Resultados - Precisiones Particulares del Mejor Resultado Tabla 5.14

Diagnóstico	Precisión	Número de Datos probados
AN-P	79 %	55
AN-R	96 %	157
BN	98 %	199
CAR	90 %	63
OTAE1 AN-Atípica	92 %	74
OTAE2 BN-bajafrecuencia	89 %	44
SIN TCA	80 %	43
TANE	81 %	40
TPA	98 %	89

Tabla 5.32: Resultados - Precisiones Particulares del Mejor Resultado Tabla 5.30

Diagnóstico	Precisión	Número de Datos probados
AN-P	92 %	27
AN-R	96 %	78
BN	97 %	102
CAR	86 %	28
OTAE1 AN-Atípica	94 %	36
OTAE2 BN-bajafrecuencia	94 %	24
SIN TCA	100 %	18
TANE	75 %	22
TPA	89 %	48

Capítulo 6

Conclusiones

El desenlace de la presente tesis converge en el cumplimiento de los objetivos e hipótesis planteados. Primeramente, con el preprocesamiento puesto a marcha de manera exitosa; resaltando las flaquezas en la captura de los datos. En segundo lugar, la aplicación de diversos algoritmos de procesamiento, que si bien dos de ellos no reportaron rendimientos arriba del 90 %, si trazaron el camino para lograrlo, mediante el entendimiento de sus resultados. En tercer lugar, la estrategia para la generación de diagramas y casos de uso derivó en un orden claro y preciso al momento de codificar el software y facilitar el proceso de pruebas y evaluación. En cuarto término, con base en los resultados, en su mayoría incrementar el número de capas y neuronas presentó mejora de resultados significativos; sin embargo, no se puede afirmar lo mismo al momento de agregar más particiones. Por último optar por contemplar e incluir validación cruzada con aleatoridad al momento de crear los segmentos, fue sumamente acertado; al alcanzar una precisión general arriba del 90 %, de otro modo, solo se hubiera alcanzado una precisión del 71 %.

6.1. Aportaciones

Al término de esta investigación se preservarán las siguientes aportaciones:

- El hallazgo de carencias en la captura de datos que ayudará a robustecer el mecanismo de inscripción de nuevos datos y corrección de datos antiguos.

- Un Software capaz de realizar diagnósticos express respaldados, extensible, ágil y amigable con el usuario.
- Una bitácora documentada con el procedimiento y detalles de una aplicación real en Minería de datos.

6.2. Trabajo a futuro propuesto

Finalmente, propongo los pasos a seguir para que tanto el software como las intenciones detrás de este se consoliden.

- Simplificar los cuestionarios aplicados para la obtención de la métrica EATTRASTORNO para una rápida determinación.
- Extender la interfaz de usuario para incluir dicho cuestionario.
- Validar y fortalecer el modelo de predicción con nuevos datos.
- Probar más técnicas de Inteligencia Artificial como algoritmos genéticos.

Bibliografía

[Abadi et al., 2015] Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from https://www.tensorflow.org/.

[Aggarwal, 2015] Aggarwal, C. C. (2015). *Data Mining The Textbook*. Springer International Publishing., Switzerland.

[Bustinzar, 2022] Bustinzar, M. (2022). Entrevista Privada.

[Foundation, 2021] Foundation, D. S. (2021). Django (version 4.0.4) [computer software]. Retrieved from https://www.djangoproject.com.

[Hunter, 2007] Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95.

[pandas development team, 2020] pandas development team, T. (2020). pandas-dev/pandas: Pandas.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

[Pressman, 2010] Pressman, R. S. (2010). *Software Engineering: a practioner's approach.* McGraw-Hill Companies, Inc.

[Tange, 2022] Tange, O. (2022). Gnu parallel 20220622 ('bongbong'). GNU Parallel is a general parallelizer to run multiple serial command line programs in parallel without changing them.

[Timmons, 2022] Timmons, J. (2022). Healthline's picks for the best eating disorder apps.