BENEMÉRITA UNIVERSIDAD AUTÓNOMA DE PUEBLA

FACULTAD DE CIENCIAS DE LA COMPUTACIÓN

Estrategias basadas en aprendizaje profundo para la toma de decisiones en aplicaciones de COVID19

TESIS

Presentada para obtener el grado de

Licenciatura en Ingeniería en Ciencias de la Computación

P R E S E N T A

Abdel Bustamante Domínguez

DIRECTOR(A) DE TESIS O TESINA

Abraham Sanchez López

ASESOR(A) DE TESIS O TESINA

Abraham Sanchez López

Puebla, México. Agosto del 2022

Agradecimientos.

Me gustaría empezar agradeciendo a mis padres, que desde un principio son los primeros que siempre confiaron en mí, me otorgaron las herramientas y los valores que para mí como profesional y persona fueron indispensables para mi crecimiento dentro de mi trayectoria, así como dejar en claro que nada de lo logrado hasta el momento hubiera sido sin ellos.

A mi familia por siempre brindarme el apoyo y cariño para cumplir mis metas, por sus consejos y ejemplo para poder tomar las mejores decisiones para mi vida.

A mi asesor por su especial orientación y guía a lo largo de mi carrera universitaria, la paciencia y los conocimientos otorgados son de gran valor e importancia para que yo terminará logrando este gran proyecto y que sin duda haber creado una gran amistad.

A mis amigos y todas las personas que me acompañaron durante esta travesía, explorando el camino que implica, el equivocarse y el lograr grandes cosas mediante el trabajo en equipo, por apoyarme cuando las situaciones se volvían complicadas y el disfrutar de grandes momentos alrededor de nuestra carrera universitaria.

A los maestros, doctores de la facultad de ciencias de la computación de esta universidad, que fueron de vital importancia y fueron parte de mi formación como profesionista.

Resumen

En mi servicio social tuve la oportunidad de aprender técnicas avanzadas de aprendizaje automático (Machine Learning), entre las cuales estudié el campo del aprendizaje profundo (Deep learning). Incluso, aprendí a cómo realizar el tratamiento de la información antes de poder aplicar alguno de los métodos de aprendizaje máquina. De entre la variedad de datasets que utilicé, obtuve varios relacionados con la pandemia del Covid19, de ahí mi interés en profundizar sobre estos aspectos.

Podría afirmar que las aplicaciones varían de acuerdo con la disponibilidad de los datos y cómo estos se procesan para las tareas de aprendizaje. Afortunadamente en esta situación pandémica existen muchos repositorios donde se pueden extraer datos relacionados al Covid19. Las aplicaciones que se pueden desarrollar y muchas de ellas se han reportado en la literatura en los últimos meses, son amplias, por ejemplo: detección de información errónea, análisis del sentimiento público, en el ámbito de la visión artificial, se tienen análisis de imágenes médicas, inteligencia ambiental y robótica basada en visión.

Al detallar cada aplicación de aprendizaje profundo en el tema de Covid19, pondremos énfasis en la representación de los datos y la tarea. Las tareas describirán principalmente cómo se construye una aplicación Covid19 cómo un problema de aprendizaje. Nos dedicaremos únicamente en aplicaciones de aprendizaje profundo, por lo tanto, nos referiremos al aprendizaje de representación de datos brutos o de alta dimensión.

Índice general

Índice de figuras

Capítulo 1. Introducción

Desde tiempos anteriores, la ciencia y la tecnología ha avanzado con gran rapidez a medida que las necesidades por la automatización y administración en la mayoría de los sectores de la industria es requerida. Esto vienen sucediendo desde la revolución industrial, donde ocurrió un gran desarrollo en el campo de la tecnología, donde ya bastantes trabajos que requerían mano de obra humana eran duros y tardados habían sido reemplazados por tecnología, lo que ayudó mucho a la humanidad. Es qui donde surge la inteligencia artificial (IA), la cual es una de las innovaciones tecnológicas que surgieron para reemplazar el trabajo físico, en diversos campos. Esta es una [1] rama de la ciencia y tecnología que desarrolla máquinas inteligentes y programas informáticos para realizar diversas tareas que requieren inteligencia humana, en su mayoría de los casos, esta funciona imitando varias funciones que la sociedad actual puede realizar.

Así como lo mencionamos, la inteligencia artificial es cada vez más utilizada en los últimos años, una tecnología que anteriormente se consideraba cómo “futurista” o “imposible” de lograr hoy en día se vuelve una realidad. Este éxito se le debe a diversos factores, uno de los cuales puede ser la flexibilidad que esta tiene de participación en diversas áreas, como medicina, educación, leyes, finanzas, transporte, tecnologías actuales, comercio, marketing, manufactureras, entre muchas otras.

AI utilizar datos externos como big data en conjunto con la inteligencia artificial se logra [2] un excelente rendimiento para tareas que involucran analizar una gran cantidad de información mediante la automatización de diversos algoritmos para rescatar la información relevante del mismo. Anteriormente, la IA era solo un concepto que se veía en la ciencia ficción y los debates que discutían sobre el efecto de la tecnología en el mundo moderno. Pero ahora, se ha implementado e inculcado en casi todas nuestras herramientas que resulta ser algo común en nuestra vida cotidiana. Eso se ha convertido en la función clave de muchos sectores técnicos y varios otros. Su capacidad para hacer tareas que el ser humano no puede realizar fácilmente mejora significativamente diversos atributos, tales como su rendimiento y la productividad.

Así es como llegamos a este documento donde podremos utilizar la inteligencia artificial en una de sus más tradicionales e importantes aplicaciones, las cuales son aprendizaje automático y aprendizaje profundo.

Es por ese motivo que este documento está dividido en cinco importantes capítulos, los cuales tratan de lo siguiente:

Empezando con el capítulo uno, el cual, nos entrega un breve resumen de los objetivos e información que se abordaran a lo largo de este documento e introduciendo un poco de los capítulos a conocer, tal como lo hacemos en este punto de la lectura.

El capítulo dos describe de manera simplificada le definición de aprendizaje profundo “deep learning”, así como sus principales características y uso de este en el mundo actual.

Posteriormente seguimos con el capítulo tres, el cual se explica de manera breve las propuestas que se abordaran en el documento, se hablará del concepto de aprendizaje automático “machine learning” y al igual que con el capítulo uno, hablaremos de sus características principales, tales como la limpieza y preparación de los datos, los diferentes algoritmos, la interpretación de los resultados, relacionándolo con los anteriores y posteriores capítulos.

Continuando el capítulo cuatro, se detallan los resultados obtenidos, explicando todas las representaciones y graficas obtenidas, hablando brevemente del código empleado para el análisis e implementación de los algoritmos, además de hablar sobre un panorama general de la computación en la nube, una pequeña implementación en nuestro proyecto de tesis, el cómo este está teniendo uno de los más grandes auges de la actualidad, sus diferentes servicios y proveedores, así como el uso de la inteligencia artificial en la misma.

Por último, contemplamos los resultados en el capítulo cinco, le asignamos una interpretación, y visualizamos gráficas representando la comparación de los algoritmos de aprendizaje automático y profundo usando un conjunto de datos relacionado con la pandemia del virus COVID-19 así como algunas futuras aplicaciones del mismo documento.

Capítulo 2. Deep learning

Existen muchas definiciones acerca de estos temas, uno de los más comunes lo define cómo que [3] el aprendizaje profundo da luchar a que diversos modelos computacionales tengan una estructura multi capas donde el procesamiento se aprende de representaciones de datos de diversas pruebas o información para de estos abstraer la información valiosa. Aplicado en diversas tecnológicas de la sociedad, tales como el reconocimiento de voz, el reconocimiento de objetos visuales, la detección de objetos y muchos otros dominios, como el descubrimiento de fármacos y la genómica.

Además, se sabe que, en el pasado, [4] los inventores siempre han soñado con crear máquinas que piensen, se dice que los mitos de la antigua Grecia hablan de objetos inteligentes, como estatuas animadas de seres humanos. mesas que llegan llenas de comida y bebidas cuando se les llama. Cuando se concibieron por primera vez las computadoras programables, la gente se preguntaba si podrían volverse inteligentes, más de cien años antes de que se construyera una (Lovelace, 1842).

Hoy en día, el aprendizaje profundo está constituido de muchas implementaciones y temas de investigación. Se busca crear software inteligente para automatizar el trabajo de rutina, comprender el habla o las imágenes, hacer diagnósticos en medicina y apoyar la investigación científica básica.

Además, en los grandes conjuntos de información usualmente se hace gran uso de diversos algoritmos de propagación hacia atrás donde indica cómo una máquina debe cambiar sus parámetros internos para mediar la interpretación de cada capa a partir de la representación en la capa anterior. Un ejemplo de esto son las famosas “redes convolucionales profundas” que se aplican principalmente para el procesamiento de imágenes, video, voz y audio, mientras otro tipo común de redes trabajan sobre datos secuenciales como texto y voz, son llamadas “recurrentes”.

Además, permite que la computadora abstraiga definiciones complejas a partir de definiciones más simples. Por ejemplo, un [5] sistema de aprendizaje profundo puede representar el concepto de una imagen de una persona combinando conceptos más simples, como esquinas y contornos, que a su vez se definen en términos de bordes.

El método por excelencia de un modelo de aprendizaje profundo es una red profunda automática o de múltiples capas (MLP). Tal como hablamos anteriormente, esto utiliza de una función matemática que asigna un conjunto de valores de entrada a los valores de salida que a su vez son interpretados para una mejor toma decisiones.

Las funciones se conforman de funciones más simples. Podemos pensar que cada implementación de una función matemática diferente proporciona una nueva representación de la entrada. La idea de aprender una representación precisa de los datos proporciona información sobre el aprendizaje profundo. Otra visión del aprendizaje profundo es que permite que una computadora aprenda un programa de computadora en varios pasos. Cada capa de representación se puede considerar cómo el estado de la memoria de la computadora después de que se ejecuta otro conjunto de instrucciones en paralelo.

Además, el Deep learning permite la capacidad de ejecutar instrucciones secuencialmente, lo que conlleva un mejor control de eventos que puedan suceder y provee de mejores interpretaciones de eventos anteriores mediante predicciones. Estas interpretaciones pueden ser usadas para almacenar información de su estado, lo que hace posible ejecutar un programa que comprende los datos entrada de mejor manera. Esta información puede ser similar a un contador o puntero en un programa de computadora tradicional. No se trata específicamente del contenido de los datos de entrada, pero ayuda a organizar en gran medida su procesamiento.

2.1 Limpieza de datos

En la actualidad la limpieza de datos [6] se comprende de un conjunto de técnicas especializadas y que, además, podrían llegar a considerarse totalmente una ciencia. Es mucho más que actualizar un registro con datos “correctos”. La limpieza de datos comúnmente incluye los procesos de “Descomposición” y “Re-ensamblamiento” de los datos. Se podría separar esta metodología en 6 principios básicos:

1. Elementarizar.
2. Estandarizar.
3. Verificar.

4. Relacionar.
5. Domiciliar.
6. Documentar.

Originalmente, esto fue visto cómo una forma simple y natural de almacenar toda esta información. Comienza [7] almacenando la información e indexando de manera organizada, aunque esto todavía no es suficiente, porque a medida que la información continúa llegando, el almacenamiento se vuelve cada vez más complejo.

Justo después del auge de los sistemas de almacenamiento de información, tales como las bases de datos, los administradores de bases de datos y todas las aplicaciones relacionadas, los empresarios se preguntaban cada vez más cómo extraer toda esta información de manera rápida, segura y eficaz manteniendo la consistencia y transparencia de los datos. Por lo que encontraron en sus análisis que uno de los frecuentes problemas que se obtenían al no realizar una limpieza de datos adecuada eran los siguientes.

- Información deficiente (faltante, incorrecta o con formato inadecuado)
- Información que no es necesaria para los procesos de negocio involucrados.
- Datos desactualizados, etc.

Un análisis que comienza con datos sin limpieza y procesamiento adecuado [8] indudablemente conducirá a decisiones equivocadas y, por lo tanto, a problemas mayores para la organización dada la velocidad de actualización de los datos y la gran variedad de datos en aumento. Con fuentes potenciales, existe un gran desafío en la limpieza de los datos y el control de calidad que se debe realizar en ellos.

2.2 Algoritmos de aprendizaje profundo

Como hemos hablado, [9] esta forma de aprendizaje automático se basa en modelo de patrones de datos como redes complejas y de múltiples capas. Debido a que es la manera más general de modelar un problema, tiene el potencial de resolver problemas difíciles como la visión por computadora y el procesamiento del lenguaje natural que mejoran en diversos aspecto tanto a la programación convencional como a otras técnicas IA.

El aprendizaje profundo no solo puede producir resultados útiles cuando fallan otros métodos, sino que también puede generar modelos más precisos que otros métodos y puede reducir el tiempo que lleva crear un modelo. Sin embargo, entrenar un modelo de aprendizaje profundo requiere una gran cantidad de poder de cómputo. Una desventaja del aprendizaje profundo además de la anterior consiste en la dificultad de interpretar los modelos de aprendizaje profundo. El sello distintivo del aprendizaje profundo es que el modelo que se entrena tiene más de una capa oculta entre las entradas y las salidas. Se ha relacionado, que el aprendizaje profundo significa el uso de redes neuronales profundas. Sin embargo, algunos algoritmos implementan el aprendizaje profundo utilizando otros tipos de capas ocultas distintas de las redes neuronales.

2.2.1 Aprendizaje profundo vs aprendizaje automático

Se mencionó que el aprendizaje profundo es una forma de aprendizaje automático, para un mejor entendimiento, me referiré al aprendizaje no profundo de la máquina como aprendizaje clásico de la máquina, para ajustarse al uso común.

De acuerdo a varios estudios según las necesidades de las situaciones en cuestión, los algoritmos clásicos de aprendizaje automático se ejecutan mucho más rápido que los algoritmos de aprendizaje profundo; una o más CPUs a menudo serán suficientes para entrenar un modelo clásico. Los modelos de aprendizaje profundo a menudo necesitan aceleradores de hardware como GPUs, TPUs o FPGAs para su procesamiento, y también para su despliegue en varias herramientas de análisis. Sin ellos, los modelos tardarían meses en entrenarse.

Sin embargo, existen situaciones en las que los algoritmos clásicos de aprendizaje de máquinas producirán un modelo que es lo “suficientemente bueno” para emplearse. En otros caso, los algoritmos clásicos de aprendizaje automático no han funcionado muy bien en el pasado o no son completamente compatibles.

2.2.2 Aplicaciones de aprendizaje profundo

Hay muchos ejemplos de problemas que actualmente requieren estos tipos de algoritmos, que en general, suelen producir los mejores modelos y resultados para diversas aplicaciones. Uno de ellos, se trata del procesamiento del lenguaje natural (PNL), por sus

siglas en inglés, es una muy buena aplicación que utilizamos frecuentemente en la actualidad y utiliza en mayor medida modelos de aprendizaje profundo.

Un excelente ejemplo, sucedió en el otoño de 2016, la calidad de Google Translate para las traducciones del inglés, francés, inglés, chino, inglés y japonés mejoró significativamente, desde la creación de una combinación de palabras hasta la creación de oraciones. Lo que sucedió detrás de escena es que el equipo de Google Brain y Google Translate renovó Google Translate, pasando de usar algoritmos antiguos de traducción estadística automática basados en oraciones (un tipo clásico de aprendizaje automático) al usar una red neuronal profunda basada en la incrustación de palabras usando el marco TensorFlow de Google. Además del problema de traducción de idiomas resuelto por Google Translate, las tareas principales de PNL incluyen el resumen automático, la resolución de referencias cruzadas, el análisis del discurso, la segmentación morfológica, el reconocimiento de entidades nombradas, la generación de lenguaje natural, la comprensión del lenguaje natural y el marcado de partes de discurso.

Otro buen ejemplo de una aplicación de aprendizaje profundo es la clasificación de imágenes. Dado que los organismos vivos procesan imágenes utilizando su corteza visual, muchos investigadores han tomado la arquitectura de la corteza visual de los mamíferos como modelo para las redes neuronales diseñadas para realizar el reconocimiento de imágenes comenzando en la década de 1950.

Otra excelente aplicación basada en Vision fue LeNet-5 de 1998 de Yann LeCun. Esta es una red neuronal convolucional (CNN) de 7 niveles para reconocer dígitos escritos a mano digitalizados en imágenes de 32x32 píxeles. Para analizar imágenes a resoluciones más altas, la red LeNet 5 debe extenderse a más neuronas y capas. Además, los mejores modelos de clasificación de imágenes de profundidad de la actualidad pueden identificar diferentes catálogos de objetos con resolución de color HD. Además de las redes neuronales puras profundas (DNN), a veces se utilizan modelos de visión híbridos que combinan el aprendizaje profundo con los algoritmos tradicionales de aprendizaje automático para realizar subtareas específicas.

La clasificación de imágenes se puede extender a la clasificación de video extrayendo cuadros individuales del video y clasificando cada cuadro. Los objetos detectados en un videoclip se pueden rastrear de fotograma a fotograma. Ha sido mejorado e implementado gracias a algoritmos de aprendizaje profundo, entre muchas otras aplicaciones.

2.2.3 Redes neuronales de aprendizaje profundo

La idea de las redes neuronales artificiales se remonta a la década de 1940. La idea básica es que una red de neuronas artificiales construida a partir de interruptores de umbral interconectados puede aprender a reconocer patrones de la misma manera que lo hacen los cerebros y los sistemas nerviosos de los animales (incluida la retina).

- *Retropropagación:* El aprendizaje ocurre en una red neuronal profunda al fortalecer la conexión entre dos neuronas cuando ambas están activas al mismo tiempo durante el entrenamiento. En los programas modernos de redes neuronales, suele implicar aumentar los valores de peso de las conexiones entre neuronas mediante una regla conocida como propagación de errores, backprop o BP.
- *Neuronas:* Para entender esto, nos hacemos la pregunta, ¿cómo se simulan las neuronas? En donde, cada uno tiene una función de difusión que cambia la salida de las neuronas conectadas, generalmente en una suma ponderada. La salida de la función de propagación se pasa a la función de activación, que se activa cuando su entrada supera un valor umbral.
- *Funciones de activación:* En las décadas de 1940 y 1950, las neuronas artificiales usaban una función de activación gradual llamada perceptiva. Las redes neuronales modernas pueden decir que usan información, pero en realidad tienen funciones de activación flexibles, como funciones logísticas o sigmoideas, tangentes hiperbólicas y unidades lineales corregidas (ReLU). ReLU es generalmente la mejor opción para una convergencia rápida, aunque existe el problema de que las neuronas "mueren" durante el entrenamiento, si la tasa de aprendizaje es demasiado alta. La salida de la función de habilitación se puede pasar a la función de salida para una mayor modulación. Sin embargo, la función de salida suele ser la función de reconocimiento, lo que significa que la salida de la función de activación se propaga a las neuronas conectadas aguas abajo.
- *Topología de redes neuronales:* Ahora que conocemos las neuronas, necesitamos saber más sobre la topología de redes neuronales comunes. En una red de retroalimentación, las neuronas se organizan en diferentes capas: una capa de entrada, cualquier cantidad de capas de procesamiento ocultas y una capa de salida, donde las salidas de cada capa se refieren a la siguiente. En una red de retransmisión con conexiones de acceso directo, algunas conexiones pueden saltar

a través de una o más capas intermedias. En una red neuronal en anillo, las neuronas pueden influirse entre sí directa o indirectamente a través de la siguiente capa.

- *Entrenamiento:* El aprendizaje supervisado de una red neuronal se realiza como cualquier otro aprendizaje automático, con un conjunto de datos de entrenamiento, donde, se compara la salida de la red con la salida deseada, genera vectores de error y aplica correcciones a la red en función del vector de error. Los conjuntos de datos de entrenamiento que se ejecutan juntos antes de que se aplique la corrección se denominan épocas. Para aquellos interesados en los detalles, la retropropagación usa la pendiente de la función de error (o costo) relativa a los pesos y sesgos del modelo para encontrar la dirección correcta para reducir el error.
- *Optimizadores:* Los optimizadores de redes neuronales suelen utilizar algún tipo de algoritmo de descenso gradiente para impulsar la retropropagación, a menudo con un mecanismo que ayuda a evitar atascarse en los mínimos locales, como la optimización de pequeños trabajos seleccionados al azar (Descenso de Gradiente Estocástico), y la aplicación de correcciones de momento de aplicar el gradiente. Se observa que ciertos algoritmos de optimización también adaptan los parámetros del modelo, observando el historial de gradientes (AdaGrad, RMSProp y Adam).
- *DNNs: Se refiere este concepto a que* una red neural profunda para un problema real puede tener más de 10 capas ocultas. Su topología puede ser simple o bastante compleja.

Al igual que con todo el aprendizaje automático, las predicciones de redes neuronales deben compararse con otro conjunto de datos de validación. De lo contrario, corre el riesgo de crear una red neuronal que solo memorice la entrada en lugar de aprender a convertirse en un predictor generalizado.

Cuantas más capas haya en la red, más características se pueden detectar. Desafortunadamente, cuantas más capas hay en la red, más tiempo lleva calcular y más difícil es entrenar.

2.2.4 Principales usos del aprendizaje profundo

Como se mencionó anteriormente, la mayor parte del aprendizaje profundo tiene lugar en redes neuronales profundas. Las redes neuronales convolucionales (CNN) se utilizan comúnmente en la visión artificial. Las redes neuronales recurrentes (RNN) se utilizan ampliamente para procesar el lenguaje natural y otras secuencias, al igual que las redes de memoria a corto plazo (LSTM) y las redes neuronales basadas en la atención. Los bosques aleatorios (también llamados bosques de decisiones aleatorias) son útiles para muchos problemas de clasificación y regresión en lugar de redes neuronales. Las redes neuronales de CNN legítimas suelen utilizar capas complejas, agregadas, totalmente conectadas y con pérdidas para simular la corteza visual.

Las capas convolucionales esencialmente fusionan muchas pequeñas regiones superpuestas. Una capa de agrupación implementa una forma de remuestreo no lineal. Hay algunas capas llamadas "ReLU" que implementan la función de activación no saturada $f(x) = \max(0,x)$. En capas totalmente conectadas, las neuronas se conectan a todas las actividades de la capa anterior. Esta clase calcula cómo entrenar una red para penalizar la diferencia entre las etiquetas esperadas y correctas mediante la función softmax, la función de pérdida de clasificación cruzada o la función de pérdida de regresión euclidiana. RNN, LSTM son redes neuronales basadas en la atención. En una red neuronal de retroalimentación, la información fluye desde la entrada a través de capas ocultas hasta la salida. Esto limita la red a procesar solo un caso a la vez. En una red neuronal periódica, la información fluye en un bucle, lo que permite que la red recuerde salidas pasadas recientes. Esto permite la secuenciación y el análisis de series temporales.

Los RNN tienen dos problemas comunes: explosión de gradiente (que se puede solucionar fácilmente conservando los gradientes) y desvanecimiento de gradiente (no es fácil de solucionar).

En LSTM, la red puede olvidar o recordar la información anterior, en ambos casos ajustando los pesos. Esto le da al LSTM memoria tanto a corto como a largo plazo y resuelve el problema de abandono. LSTM puede procesar cientos de entradas en el pasado.

Las unidades de atención son puertas generalizadas que aplican pesos al vector de entrada. El codificador neuronal jerárquico utiliza varias capas de unidades de atención para procesar decenas de miles de entradas anteriores.

- Random Forest: otro tipo de algoritmo de aprendizaje profundo que no es una red neuronal profunda es un bosque aleatorio o un bosque de decisiones aleatorias. Un bosque aleatorio se construye a partir de muchas capas. Sin embargo, se construye a partir de árboles de decisión en lugar de neuronas y produce una media estadística (modo de clasificación o media de regresión) de las predicciones de cada árbol. El aspecto aleatorio de los bosques aleatorios es el uso de la agregación de arranque (también conocido como empaquetado) en árboles individuales y subconjuntos aleatorios de características.

Entre muchos otros algoritmos de aprendizaje profundo, se puede adaptar a cualquier situación o problema que se presente.

2.2.5 Aprendizaje de transferencia profunda

Es el proceso de ajustar un modelo entrenado en un conjunto de datos a otro conjunto de datos. Transferir el aprendizaje es mucho más rápido que entrenar un modelo desde cero y requiere muchos menos datos para el entrenamiento. Una de las herramientas más utilizadas para este tipo de aprendizaje es Google Cloud AutoML, que implementa deep transfer learning para visión, traducción y lenguaje natural. Azure Machine Learning Service proporciona servicios de aprendizaje de transferencia profunda similares a visión personalizada, idioma personalizado, traducción y búsqueda personalizada. TensorFlow tiene su propia forma de coordinar el entrenamiento distribuido con un servidor de parámetros, pero un enfoque más general usa Open MPI (Message Passing Interface). Horovod, un marco de entrenamiento distribuido para TensorFlow, Keras y PyTorch desarrollado en Uber, usa Open MPI y Nvidia NCCL. Según el modelo entrenado, Horovod logra una eficiencia de escalado del 68 % al 90 %.

2.3 Machine learning

Esta es la tecnología más demandada en el mercado actual. Sus aplicaciones van desde automóviles autónomos hasta la predicción de enfermedades mortales como la ELA (esclerosis lateral amiotrófica). Esta sección proporciona una breve introducción al aprendizaje automático, comprendiendo todos los conceptos básicos del aprendizaje automático y luego comprendiendo las implementaciones prácticas del aprendizaje automático utilizando algunos algoritmos en el lenguaje R. El término aprendizaje automático fue acuñado [10] por Arthur Samuel, un pionero estadounidense. en los campos de los juegos de computadora y la inteligencia artificial, en 1959 lo llamó "la capacidad de darle a una computadora la capacidad de aprender sin ser programada explícitamente". Una vez que se definió el concepto, toda la historia comenzó cuando Tom Mitchell presentó una definición matemática y relacional en 1997 que definía "un programa de computadora aprende de la experiencia E en una tarea T y mide su desempeño P". Si su aplicación está en la tarea T, la tarea T se mide en el rendimiento P y tiende a mejorar en la experiencia E.

El aprendizaje automático es una palabra de moda en estos días. Bien vale la pena. Porque es una de las áreas más apasionantes de la informática. Entonces, ¿qué significa realmente el aprendizaje automático? Echemos un vistazo rápido al aprendizaje automático. Intenta tirar el papel a la basura. Después del primer intento, te das cuenta de que le pusiste mucho esfuerzo. Después del segundo intento, podemos ver que estamos más cerca del objetivo, pero necesitamos aumentar el ángulo de lanzamiento. Lo que sucede aquí es que con cada lanzamiento aprendes algo y mejoras tus resultados. Estamos programados para aprender de la experiencia. Esto significa que las tareas manejadas por el aprendizaje automático brindan una definición operativa más fundamental que la definición de dominios en términos cognitivos. Esto es después de que Alan Turing hiciera la pregunta "¿Pueden pensar las máquinas?" en su artículo "Computadoras e inteligencia". Propuesto. Reemplácela con la pregunta "¿Pueden las máquinas hacer lo que nosotros (como entidades pensantes) podemos hacer?"

En el campo del análisis de datos, el aprendizaje automático se utiliza para diseñar modelos y algoritmos complejos adecuados para hacer predicciones. En uso comercial, esto se denomina análisis predictivo. Estos modelos analíticos permiten a los

investigadores, científicos de datos, ingenieros y analistas aprender relaciones históricas y tendencias en conjuntos de datos (entradas) para "brindar decisiones y resultados confiables y repetibles" y "revelar conocimientos ocultos".

2.3.1 Definiciones de aprendizaje automático

Algoritmo: Un algoritmo de aprendizaje automático es un conjunto de reglas y técnicas estadísticas que se utilizan para aprender patrones a partir de datos y extraer información significativa de ellos. Es la lógica detrás de un modelo de machine learning. Un ejemplo de un algoritmo de aprendizaje automático es el algoritmo de regresión lineal.

Modelo: Un modelo es el componente principal de machine learning. Se entrena mediante el uso de un algoritmo de aprendizaje automático. Un algoritmo mapea todas las decisiones que se supone que debe tomar un modelo en función de la entrada dada, para obtener la salida correcta.

Variable predictora: Es una característica o características de los datos que se pueden usar para predecir la salida.

Variable de respuesta: Es la característica o la variable de salida que debe predecirse mediante el uso de la(s) variable(s) predictora(s).

Datos de entrenamiento: El modelo de aprendizaje automático se crea utilizando los datos de entrenamiento. Los datos de entrenamiento ayudan al modelo a identificar tendencias y patrones clave esenciales para predecir el resultado.

Prueba de datos: Después de entrenar el modelo, debe probarse para evaluar con qué precisión puede predecir un resultado. Esto se hace mediante el conjunto de datos de prueba.

2.3.2 Proceso de aprendizaje automático

El aprendizaje automático implica la construcción de un modelo predictivo que se puede usar para encontrar una solución a la declaración de un problema. Para comprender el proceso de aprendizaje automático de una manera sencilla, supongamos que queremos predecir si lloverá o no mediante el aprendizaje automático, para los cuales seguiremos los siguientes pasos.

Paso 1: Determinar el objetivo del enunciado del problema: En este paso, debemos entender exactamente qué esperar. En nuestro caso, el objetivo era predecir la probabilidad de precipitación mediante el estudio de las condiciones meteorológicas. En este punto, es necesario tener en cuenta qué tipo de datos se pueden usar para resolver este problema o qué tipo de enfoque debe tomar para encontrar una solución.

Paso 2: Recopilación de datos: En esta etapa, debe estar haciendo preguntas como,

- ¿Qué tipo de datos se necesitan para resolver este problema?
- ¿Están disponibles los datos?
- ¿Cómo puedo obtener los datos?

Una vez que conozca los tipos de datos que se requieren, debe comprender cómo puede obtener estos datos. La recopilación de datos se puede realizar manualmente o mediante web scraping. Sin embargo, si es un principiante y solo está buscando aprender Machine Learning, no tiene que preocuparse por obtener los datos. Hay miles de recursos de datos en la web, simplemente puede descargar el conjunto de datos y comenzar.

Paso 3: Preparación de datos: Los datos que recopiló casi nunca están en el formato correcto. Encontrará muchas inconsistencias en el conjunto de datos, como valores perdidos, variables redundantes, valores duplicados, etc. Eliminar tales inconsistencias es muy importante porque pueden conducir a cálculos y predicciones erróneos. Por lo tanto, en esta etapa, escanea el conjunto de datos en busca de inconsistencias y las corrige en ese momento.

Paso 4: Análisis de datos exploratorios: Este paso es analizar los datos y revelar todos los acertijos ocultos en los datos o análisis exploratorio de datos es la etapa de lluvia de ideas del aprendizaje automático. El descubrimiento de datos implica comprender patrones y tendencias en los datos. En esta etapa, se extrae toda la información útil y se comprenden las asociaciones entre las variables. Por ejemplo, en el caso de pronosticar lluvia, sabemos que es probable que llueva si baja la temperatura. Estas asociaciones deben ser entendidas y mapeadas en este punto.

Paso 5: Creación de un modelo de aprendizaje automático: Toda la información y los patrones obtenidos durante la exploración de datos se utilizan para crear modelos de aprendizaje automático. Este paso siempre comienza dividiendo el conjunto de datos en dos partes, los datos de entrenamiento y los datos de prueba. Los datos de entrenamiento

se utilizarán para construir y analizar el modelo. La lógica del modelo se basa en el algoritmo de aprendizaje automático en curso. En el caso de pronosticar lluvia, dado que la salida será verdadera (lloverá mañana) o falsa (no lloverá mañana), podemos usar un algoritmo de clasificación como la regresión logística. Elegir el algoritmo correcto depende del tipo de problema que está tratando de resolver, el conjunto de datos y la complejidad del problema. En las siguientes secciones, discutiremos los diferentes tipos de problemas que se pueden resolver utilizando el aprendizaje automático.

Paso 6: Evaluación y optimización del modelo: Después de construir un modelo utilizando el conjunto de datos de entrenamiento, finalmente es hora de poner a prueba el modelo. El conjunto de datos de prueba se utiliza para verificar la eficiencia del modelo y la precisión con la que puede predecir el resultado. Una vez que se calcula la precisión, se pueden implementar mejoras adicionales en el modelo en esta etapa. Se pueden utilizar métodos como el ajuste de parámetros y la validación cruzada para mejorar el rendimiento del modelo.

Paso 7: Predicciones: Una vez que el modelo se evalúa y mejora, finalmente se utiliza para hacer predicciones. El resultado final puede ser una variable categórica (p. ej., verdadero o falso) o puede ser una cantidad continua (p. ej., el valor previsto de una acción).

Así que ese fue todo el proceso de manera general para algoritmos machine learning. Ahora es el momento de conocer las diferentes formas en las que estos mismos pasos los podemos trasladar al propósito de este documento que es utilizar en un conjunto de datos covid19.

2.4 Computación de la nube

La computación en la nube [11] es una evolución de la tecnología de la información y un negocio dominante modelo para la entrega de recursos de TI (Tecnologías de la información). Con la computación en la nube, las personas y las organizaciones pueden obtener acceso a la red bajo demanda a un grupo compartido de recursos administrados y recursos de TI escalables, como servidores, almacenamiento y aplicaciones.

Recientemente, académicos, así como los profesionales han prestado mucha atención a la computación en la nube.

La demanda de esta tecnología depende en gran medida de los servicios que usamos regularmente en nuestra vida diaria, como el almacenamiento de datos, el procesamiento de textos, la gestión comercial y los juegos en línea. La computación en la nube también proporciona la infraestructura que impulsa las principales tendencias digitales, como la computación móvil, Internet de las cosas, big data e inteligencia artificial, acelerando el dinamismo de la industria, revolucionando las industrias y transformando los modelos comerciales existentes para revolucionar e impulsar la transformación digital. Sin embargo, la computación en la nube ofrece más que solo muchos beneficios y oportunidades. También conlleva varios desafíos y preocupaciones, como la protección de los datos de los clientes. En 2004, la computación en la nube se hizo popular. Este es esencialmente el modelo utilizado por las empresas que administran su propia infraestructura para mantener las cosas en funcionamiento (Salesforce.com, Amazon, Google, Facebook, etc.), donde se realizan cambios y actualizaciones en todo el sistema y el sistema es La situación cambia dependiendo de la situación. Expandido. Desde entonces, la computación en la nube se ha convertido en la forma principal de implementar y consumir infraestructura informática (computación y almacenamiento), middleware y aplicaciones. Este es uno de los cambios fundamentales, con la llegada de Internet. Internet está cambiando la forma en que nos comunicamos, hacemos negocios y brindamos servicios.

2.4.1 Microsoft Azure

Azure es un conjunto de [12] servicios en la nube en constante expansión que ayuda a su organización a enfrentar los desafíos comerciales actuales y futuros. Azure brinda la libertad de crear, administrar e implementar aplicaciones en una red global masiva utilizando sus herramientas y frameworks..

Azure ofrece más de 100 servicios que le permiten hacer de todo, desde ejecutar aplicaciones existentes en máquinas virtuales hasta explorar nuevos modelos de software, como bots inteligentes y realidad mixta. Varios equipos están comenzando a explorar la nube al mover sus aplicaciones existentes a máquinas virtuales que se ejecutan en Azure. Migrar aplicaciones existentes a máquinas virtuales es un buen comienzo, pero la nube no es solo otro lugar para ejecutar sus máquinas virtuales.

Por ejemplo, Azure proporciona servicios de aprendizaje automático e inteligencia artificial que pueden comunicarse localmente con los usuarios a través de la vista, el sonido y la voz. También proporciona soluciones de almacenamiento dinámicamente escalables para almacenar grandes cantidades de datos. Los servicios de Azure permiten soluciones que no serían posibles sin el poder de la nube.

2.4.2 Azure cognitive services

Cognitive Services proporciona [13] capacidades de aprendizaje automático para resolver problemas comunes como: B. Análisis de texto para opinión emocional o análisis de imagen para reconocimiento de objetos o rostros. No se requieren habilidades de aprendizaje automático o ciencia de datos para usar estos servicios. Es un grupo de servicios, cada uno de los cuales admite diferentes capacidades predictivas generalizadas. Los servicios cognitivos proporcionan algunos o todos los componentes (datos, algoritmos y modelos entrenados) de una solución de aprendizaje automático. Estos servicios están diseñados para requerir habilidades generales de datos sin experiencia en aprendizaje automático o ciencia de datos. Estos servicios proporcionan SDK y API REST basados en lenguaje. Por lo tanto, se requiere el conocimiento de un lenguaje de programación para utilizar el servicio. Este proporciona un modelo preentrenado. Es una combinación de datos y algoritmos, disponible a través de SDK o REST API. Dependiendo de su escenario, puede implementar este servicio en minutos. Los servicios cognitivos brindan respuestas a preguntas comunes como: B. Frases clave en texto o identificadores en imágenes.

Capítulo 3. Estrategia propuesta

Siguiendo con este escrito, contamos con la propuesta de solución a la situación presentada. Con la cual, se basa en un descargar un repositorio público directamente de la página oficial de la universidad de medicina “John Hopkins”. Se procede a descargar el conjunto de datos y programar algunos de los algoritmos tanto machine learning como deep learning para mostrar que ventajas y que utilidades ofrecen cada uno de estos al

momento de analizarlos mediante gráficas proporcionadas por el entorno de desarrollo integrado R Studio y la herramienta de análisis de inteligencia artificial Azure Cognitive Services.

Empezando con el proyecto, primero es necesario ubicar la página de la Universidad de "John Hopkins" en la cual específica un conjunto de datos mediante su apartado web "How to use our data", donde nosotros podemos ver un repositorio virtual, para el cual está dividida en dos secciones, una es un repositorio refiriéndose a toda la información relacionada con la pandemia de COVID-19 de manera Global en todos los países y la otra sección se trata de un repositorio de datos COVID-19 pero específico para el país de Estados Unidos. Nosotros, de acuerdo con el objetivo de esta tesis, nos enfocaremos en el ambiente global porque estamos enfocados en los datos que ocurren en nuestro país (México). Se logra apreciar que ingresando a este repositorio nos ofrece una gran cantidad de información con detalles específicos presentados en formato CSV, y si ingresamos a uno de ellos podemos ver qué nos dan características tales como el país/estado de donde pertenece, el municipio, la última fecha de actualización, cuánto duró, los casos confirmados, los decesos, cuántos casos se han recuperado, los casos activos. el incidente de crecimiento, todos estos pequeños detalles que no serán de vital importancia para poder realizar nuestro trabajo de tesis.

Empezando con el análisis de repositorio primero procederemos a descargar los conjuntos de datos en formato CSV provenientes de la página John Hopkins en nuestro ordenador, Esos vienen ordenados cómo habíamos comentado mediante fechas y nosotros tomaremos los relacionados con la fecha del mes de febrero los cuales fueron los últimos realizados y más recientes hasta el momento, una vez descargado en el ordenador, nosotros procederemos a instalar un IDE la cual en español es un "entorno de desarrollo integrado", el cual nos permitirá analizar la información proveniente de fuentes externas tales como empresas, bases de datos, repositorios privadas/públicas, entre muchas otras fuentes para su posible análisis con nuestros algoritmos a implementar, el software que utilizaremos para aplicar nuestros algoritmos de machine learning y deep learning es llamado R Studio para el cual nosotros procederemos primero instalar el lenguaje de R para el sistema operativo utilizado en el momento (Windows 10) en este caso y posteriormente empezar a codificar todo nuestro nuestros algoritmos.

Una vez con el conjunto datos en nuestro ordenador con nuestro IDE instalado y listo para poder programar, necesitamos saber qué algoritmos vamos a utilizar para analizar esta información, cómo ya habíamos comentado previamente, nosotros estamos listos para utilizar algoritmos machine learning y en el capítulo anterior explicamos brevemente sobre en qué consisten, sus algoritmos y sus más comunes aplicaciones.

Cabe recalcar que estos algoritmos utilizan en su mayoría datos estáticos, donde tenemos la certeza de que son confiables. Con esto en mente, empezaremos primero analizando los algoritmos de regresión lineal, de modelos de clasificación, de regresión logística y por último de K-nearest neighbors que en este caso son los más comunes y que sin problema los podemos aplicar en este en este conjunto de datos, ya que cómo recordamos en machine learning usualmente hay dos formas de estimar los datos, mediante una función F(x), nos referíamos a las predicciones y las inferencias. Las predicciones se basan solamente en obtener la relación de los valores individuales en el conjunto de datos, no se concentra en investigar las relaciones que se tiene con sus otros datos cercanos o de mayor relación. En cambio, la inferencia se trata de encontrar los patrones o la relación entre todas las variables de salida de entrada, estos van a permitir una optimización en las predicciones en un futuro.

Además, sabemos que la predicción utiliza como herramienta principal la precisión la cual simplemente es una función que nos permite deducir qué tan impactante es nuestro algoritmo en un conjunto de datos, qué tan certero y fiable son los resultados que estamos obteniendo. Y para la inferencia nosotros tenemos como herramienta principal la interpretación, la cual simplemente nos permite observar qué tan fuerte son las relaciones entre los datos presentados en un conjunto, si son muy fuertes permitirán resultados e inferencias más acertadas en el futuro próximo, y si no lo son o si no tienen relación, concluimos no pude haber una relación existente

3.1 R – Studio

Esta será la herramienta que utilizaremos para compilar nuestros programas y generar reportes visuales que permitan el análisis y comparación de los datos con diferentes algoritmos.

Cómo podemos observar en los próximos resultados de nuestro estudio, R-Studio [14] es todo un completo IDE de desarrollo, pero embutido en una aplicación web, que permite además integrarse con una serie de herramientas enfocadas en la gestión de proyectos.

R es un lenguaje de programación de código abierto orientado al trabajo con datos y su análisis estadístico, usado principalmente en el ámbito de la investigación matemática y machine learning, minería de datos, etc… Es multiplataforma, por lo que se puede usar en cualquier sistema operativo de escritorio.

Por su parte, R-Studio es un entorno de desarrollo remoto, que se instala comúnmente en un servidor Linux/Windows, que permite manejar y ejecutar proyectos en R de manera remota, sin tener que instalar nada en el ordenador del usuario.

R-Studio ofrece todas las herramientas que podemos esperar de un IDE moderno, como coloreado de sintaxis, ayudas para completado y formateado de código. Ofrece además una plataforma de ejecución para los programas escritos en R, de modo que se pueden poner en marcha de manera cómoda, online y sin salir de la propia aplicación.

El entorno de desarrollo integra diversas herramientas adicionales dentro del espacio de trabajo, como la documentación del lenguaje R, sistemas de control de versiones (Git y otros), la gestión de proyectos y visualización de datos, así como un depurador que permite localizar y corregir errores en el código fácilmente. Además, se puede extender por medio de paquetes (packages) adicionales en función de las necesidades de los profesionales. Todo ello funciona en el navegador y por tanto es accesible desde cualquier lugar, simplemente disponiendo de un acceso a Internet, lo que permite el trabajo en remoto y la disponibilidad de las herramientas de análisis de datos, así como cualquiera de los archivos usados, desde cualquier lugar.

Dentro de la categoría del machine learning, ofrece un entorno de desarrollo completamente accesible desde el navegador desde el que puedes fácilmente desarrollar y depurar código y organizar tus documentos en proyectos. La interfaz es una plataforma para análisis y cálculo para proyectos con grandes cantidades de datos o con funciones matemáticas complejas.

3.2 Conjunto de datos Johns Hopkins

El Centro de Recursos de Coronavirus (CRC) de Johns Hopkins es una fuente actualizada continuamente de datos de COVID-19 y orientación de expertos. Recopilan y analizan los mejores datos disponibles sobre casos, muertes, pruebas, hospitalizaciones y vacunas para ayudar al público, en los que los encargados se encargan principalemnte de formular políticas y los profesionales de la salud de todo el mundo a responder a la pandemia. TIME reconoció al CRC como la "fuente de datos de referencia" para COVID-19 y lo nombró entre los 100 mejores inventos de 2020. En 2021, Research America nombró al CRC como receptor de su premio "Meeting the Moment for Public Health" . El CRC cuenta con el apoyo de Bloomberg Philanthropies y la Fundación Stavros Niarchos.

Además, los expertos realizan los últimos análisis todos los lunes sobre vacunas, casos, muertes, hospitalizaciones, entre otros. Cuentan con un boletín electrónico The Week in COVID-19 del Centro de Recursos de Coronavirus.

Estos cuentan con su propia página web en la que muestran a todo el mundo un seguimiento detallado y en tiempo real acerca del avance del virus COVID-19. Este cuenta con datos tales como casos confirmados, decesos, falsos positivos mostrados de manera simplificada, graficada e incluso cuenta con un repositorio en la plataforma Git-Hub donde incluye esta información para la mayoría de los países del mundo en formato CSV, donde usaremos este último para poder implementarlo y expórtalos en nuestros algoritmos de inteligencia artificial

3.3 Algoritmos machine learning usados

Los datos de 2021 COVID-19 de México ya están almacenados en su computadora. Luego implementa algoritmos de aprendizaje automático para procesarlo y analizarlo, de modo que pueda tomar mejores decisiones en el futuro. Tenga en cuenta que estos se dividen en tres tipos principales, aprendizaje y procesamiento de datos, según el contexto.

1. Aprendizaje supervisado

El aprendizaje supervisado usa ejemplos para entrenar una máquina. Por lo tanto, el operador proporciona al algoritmo de aprendizaje automático un conjunto de datos conocido que contiene las entradas y salidas deseadas. Los algoritmos tienen que encontrar una manera de determinar cómo obtener sus entradas y salidas. Los operadores saben la respuesta correcta a un problema, pero los algoritmos reconocen patrones en los datos, aprenden de las observaciones y hacen predicciones. Los algoritmos hacen predicciones y son corregidos por el operador. Este proceso continúa hasta que el algoritmo alcanza un alto nivel de precisión y rendimiento. Los operadores conocen la respuesta correcta a un problema, pero los algoritmos identifican patrones en los datos, aprenden de las observaciones y hacen predicciones. Los algoritmos hacen predicciones y son corregidos por el operador. Este proceso continúa hasta que el algoritmo alcanza un alto nivel de precisión y rendimiento

2. Aprendizaje sin supervisión

Aquí, los algoritmos de aprendizaje automático examinan los datos para identificar patrones. No hay claves de respuesta ni operadores humanos para dar instrucciones. En cambio, las máquinas descubren conexiones y relaciones analizando los datos disponibles. El aprendizaje no supervisado requiere algoritmos de aprendizaje automático para interpretar grandes conjuntos de datos y enrutar esos datos en consecuencia. Entonces, el algoritmo intenta organizar estos datos de alguna manera para explicar su estructura. Esto significa agrupar datos en grupos u organizarlos de una manera que los haga parecer más organizados. A medida que analiza más datos, su capacidad para tomar decisiones al respecto se vuelve progresivamente mejor y más sofisticada.

3. Aprendizaje por refuerzo

1. El aprendizaje por refuerzo se centra en un proceso de aprendizaje controlado en el que se proporciona a los algoritmos de aprendizaje automático un conjunto de acciones, parámetros y valores finales. Al definir reglas, el algoritmo de aprendizaje automático explora diferentes opciones y posibilidades, observando y evaluando el resultado de cada una, tratando de determinar el mejor resultado.
   Como resultado, el sistema aprende la máquina a través de un proceso de prueba y error. Aprenda de la experiencia pasada y adapte su enfoque a la situación para obtener los mejores resultados. Como ya se mencionó, estos algoritmos [15] permiten que las computadoras reconozcan patrones en grandes cantidades de

datos y hagan predicciones (análisis predictivo). Este aprendizaje permite que la computadora realice ciertas tareas de manera autónoma. En otras palabras, no tienes que programarlo. Se centra en el aprendizaje supervisado y no supervisado utilizando algoritmos de aprendizaje automático. A continuación, describimos algunos de los algoritmos que implementamos para esta tarea. Algoritmos de regresión

Para las tareas de regresión [16], los programas de aprendizaje automático necesitan estimar y comprender las relaciones entre las variables. El análisis de regresión es particularmente útil para pronosticar y pronosticar porque se enfoca en una variable dependiente y muchas otras variables que cambian.

Para resolver el problema de la regresión lineal es necesario [17] representar matemáticamente las ideas de “línea recta” y “mejor ajuste”, discutidas a continuación.
La relación lineal entre dos variables x (variable independiente) y y (variable dependiente) se representa matemáticamente a través de la ecuación:

$$y = wx + b$$

Figura 3.3.1 Relación lineal entre dos variables

1. Donde w es la pendiente (inclinación) de la línea y b es la intersección con el eje y. Por lo tanto, el objetivo de la regresión lineal es encontrar los valores más apropiados de w, b que mejor describan la relación entre las variables x, y. Para encontrar este "mejor", debe definir correctamente alguna métrica que pueda cuantificar la precisión de la regresión lineal que está haciendo.

Esta métrica se conoce como función de costo y se conoce comúnmente como pérdida de aprendizaje automático. Lo explicaré a continuación. Nos permite medir la diferencia entre los datos reales (y) y los datos obtenidos después de ejecutar la regresión lineal (que llamaremos a continuación). Hay varias formas de definir matemáticamente esta pérdida, pero la más utilizada es el error cuadrático medio (ECM), definido de la siguiente manera:

$$\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

Figura 3.3.2 Función de costo

Algoritmos lineales- Donde N es el número total de datos originales, y_i y 〖y^^〗_i son los datos originales y los datos obtenidos de la regresión respectivamente, y la resta (y_i-〖y^^〗_i) es todo en las diferencias totales son positivos y no cancelados.

Por lo tanto, ECM mide el error promedio que existe entre los datos originales y los datos obtenidos de la regresión lineal. Ahora que hemos definido la pérdida, hablemos de a qué se refiere el término "mejor ajuste".

2. Algoritmos bayesianos

   Este tipo de algoritmo de clasificación se basa en el teorema de Bayes clasifica cada valor como independiente entre sí. Esto nos permite usar probabilidades para predecir una clase o categoría en función de un conjunto determinado de características. A pesar de su simplicidad, los clasificadores funcionan sorprendentemente bien y superan a los métodos de clasificación más sofisticados, razón por la cual se usan con frecuencia.

3. Algoritmo de agrupación

   Se utilizan en el aprendizaje no supervisado para clasificar datos no etiquetados, es decir, datos sin categorías o grupos definidos.

   El algoritmo usa el número de grupos representados por la variable K para encontrar grupos en los datos. Luego mostramos cómo funciona este tipo de algoritmo iterativamente, asignando cada punto de datos a uno de los K grupos de acuerdo con la función proporcionada. La notación por utilizar (véase la Figura 3.3.3) en este tema es la siguiente:

		X_1	...	X_j	...	X_n	C
$(\mathbf{x}_1, c_1)$	1	x_{11}	...	x_{1j}	...	x_{1n}	c_1
	⋮	⋮		⋮		⋮	⋮
$(\mathbf{x}_i, c_i)$	i	x_{i1}	...	x_{ij}	...	x_{in}	c_i
	⋮	⋮		⋮		⋮	⋮
$(\mathbf{x}_N, c_N)$	N	x_{N1}	...	x_{Nj}	...	x_{Nn}	c_N
$\mathbf{x}$	$N+1$	$x_{N+1,1}$	...	$x_{N+1,j}$	...	$X_{N+1,n}$	?

Figura 3.3.3 Notación para el paradigma K

Donde indica un fichero de N casos, cada uno de los cuales esta caracterizado por n variables predictoras, X1, . . ., Xn y una variable a predecir, la clase C. Los N casos se denotan por:

$c^1, \ldots, c^m$ Denotan los m posibles valores de la variable clase C.

$$
\begin{aligned}
&(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_N, c_N) && \text{donde} \\
&\mathbf{x}_i = (x_{i,1} \ldots x_{i,n}) && \text{para todo } i = 1, \ldots, N \\
&c_i \in \{c^1, \ldots, c^m\} && \text{para todo } i = 1, \ldots, N
\end{aligned}
$$

Figura 3.3.4 Parámetros para el paradigma K

Tal y como puede observarse en el mismo, se obtienen las distancias de todos los elementos ya clasificados al nuevo caso, x, para poder clasificarlos.

COMIENZO
  Entrada: $D = \{(\mathbf{x}_1, c_1), \ldots, (\mathbf{x}_N, c_N)\}$
    $\mathbf{x} = (x_1, \ldots, x_n)$ nuevo caso a clasificar
  PARA todo objeto ya clasificado (x_i, c_i)
    calcular $d_i = d(\mathbf{x}_i, \mathbf{x})$
  Ordenar $d_i (i = 1, \ldots, N)$ en orden ascendente
  Quedarnos con los K casos $D_{\mathbf{x}}^K$ ya clasificados más cercanos a $\mathbf{x}$
  Asignar a $\mathbf{x}$ la clase más frecuente en $D_{\mathbf{x}}^K$
FIN

Figura 3.3.5 Algoritmo para el paradigma K

En este ejemplo, los predictores son X1 y X2 y se eligió K = 3. Un ejemplo de un algoritmo de agrupamiento es el algoritmo K-Means, un algoritmo de clasificación (agrupamiento) no supervisado que agrupa objetos según sus propiedades de grupo K. hay. La agrupación se realiza reduciendo la distancia total entre cada objeto y su grupo o

centro de grupo. Generalmente se utiliza la distancia al cuadrado. El algoritmo consiste en tres pasos:

1. *Inicialización:* Al elegir el número de grupos, K, donde los K centroides se configuran en el espacio de datos, por ejemplo, eligiéndolos al azar.
2. *Asignar objetos al centroide:* Cada objeto de los datos se asigna al centroide más cercano.
3. *Actualizar puntaje promedio:* La posición del puntaje promedio de cada grupo se actualiza tomando el nuevo baricentro como la posición media de los objetos en ese grupo.

El algoritmo k-mean resuelve problemas de optimización. El problema de optimización es una función que mejora (reduce) la suma de las distancias al cuadrado de cada objeto desde el centro del grupo. Los objetos están representados por vectores reales de tamaño d (x1, x2, ..., xn), y el algoritmo K-Mean determina que la suma de las distancias de los objetos en cada grupo es S = {S1, S2, ..., Sk }, en el centro de la misma. El problema se puede formular de la siguiente manera.

$$\min_{\mathbf{S}} E(\boldsymbol{\mu_i}) = \min_{\mathbf{S}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in S_i} \|\mathbf{x}_j - \boldsymbol{\mu_i}\|^2 \quad (1)$$

Figura 3.3.6 Algoritmo K-means

Se toma como nuevo centro la media de los elementos de cada grupo.

La principal ventaja del método K-mean es que es simple y rápido. Pero el valor de K debe especificarse y el resultado final depende de la configuración de los baricentros. En principio, no converge a un mínimo global sino a un mínimo local.

3.4 Algoritmos deep learning usados

1. Algoritmos de redes neuronales

   Una red neuronal artificial (ANN) consta de módulos dispuestos en una serie de capas, cada una conectada a capas adyacentes. Las redes neuronales artificiales se inspiran en los sistemas biológicos, como el cerebro, y en cómo procesan la información. Así que básicamente es una gran cantidad de elementos de

procesamiento interconectados que trabajan al unísono para resolver problemas específicos.

También aprende a través de ejemplos y experiencia, y es muy útil para modelar relaciones no lineales en datos dimensionales, o cuando las relaciones entre variables de entrada son difíciles de entender.

La señal de entrada, la información recibida por la neurona artificial, es la variable independiente. Los valores de entrada n se multiplican por sus respectivos pesos. Entonces, en resumen, el vector de entrada se multiplica por el vector de peso para obtener una combinación lineal de la entrada y el peso. Esto se llama una función de peso.

$$X * W^t = (x_1, x_2, \dots x_n) * \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_n \end{pmatrix} = \sum_{i=1} x_i * w_i$$

Figura 3.4.1 Ecuación para señales de entrada

A continuación, se aplicará una función activación:

$$\phi(\sum_{i=1} x_i * w_i)$$

Figura 3.4.2 Función de activación

Y finalmente el resultado se pasa a la salida. Este valor puede ser la nueva entrada a las neuronas que forman la red neuronal, o puede ser el resultado final, la variable de respuesta. La respuesta obtenida puede ser una variable continua como el precio de un objeto, una respuesta binaria (0,1) (sí, no) sobre si una persona tiene alguna enfermedad, o una respuesta categórica. Esto demuestra que la clasificación puede ayudar.

A continuación, podemos observar mediante un gráfico el comportamiento de una neurona artificial.

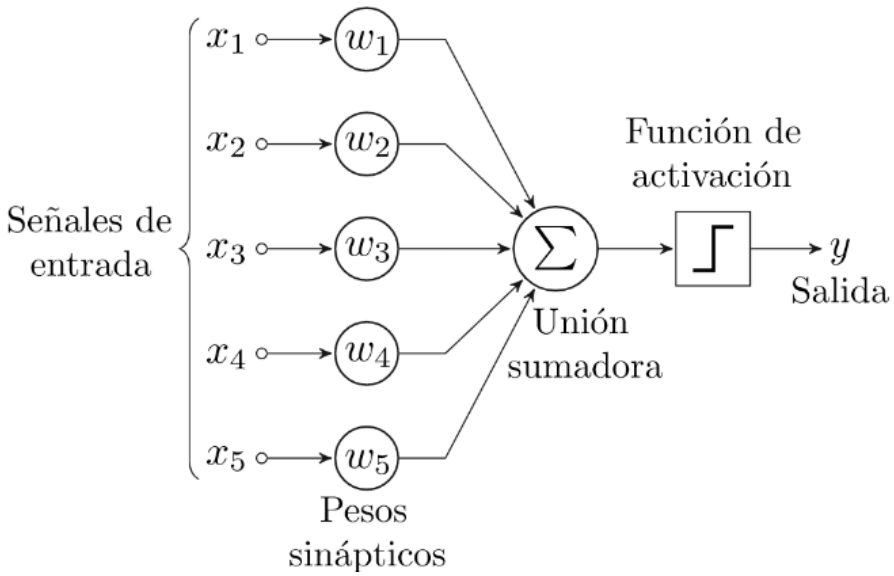


Figura 3.4.3. Esquema de un percepción o neurona artificial.:

Una función de activación es una función que envía información generada por una combinación lineal de pesos y entradas. Es decir, cómo se envía la información a través del enlace de salida. Puede enviar la información sin cambios. Hable sobre la función de identidad o no envía ninguna información. Las funciones de activación generalmente hacen que el modelo no sea lineal, ya que queremos que la red sea capaz de resolver problemas cada vez más complejos. Algunas de las funciones de activación más conocidas o utilizadas son:

Función Escalón, (similar a la función binaria.)

$$\phi(x) = \begin{cases} 0 \; si \; x < 0. \\ 1 \; si \; x \geq 0. \end{cases}$$

Figura 3.4.4. Función escalón.:

Función Sigmoidal.

$$\phi(x) = \frac{1}{1 + e^{-x}}$$

Figura 3.4.5. Función sigmoidal:

Función Rectificadora (ReLU).

$$\phi(x) = \max\{0, x\}, siendo \; x \geq 0.$$

Figura 3.4.6. Función rectificadora.:

Función Tangente Hiperbólica.

$$\phi(x) = \frac{1 - e^{-2x}}{1 + e^{-2x}}$$

Figura 3.4.7. Función tangente hiperbólica.:

En resumen, podemos definir una red neuronal como un gráfico, es decir, una capa de entrada que toma una señal de entrada y la envía a través de un estímulo a la siguiente capa oculta. Cuando llegamos a la última capa, que es la capa de salida, obtenemos la respuesta

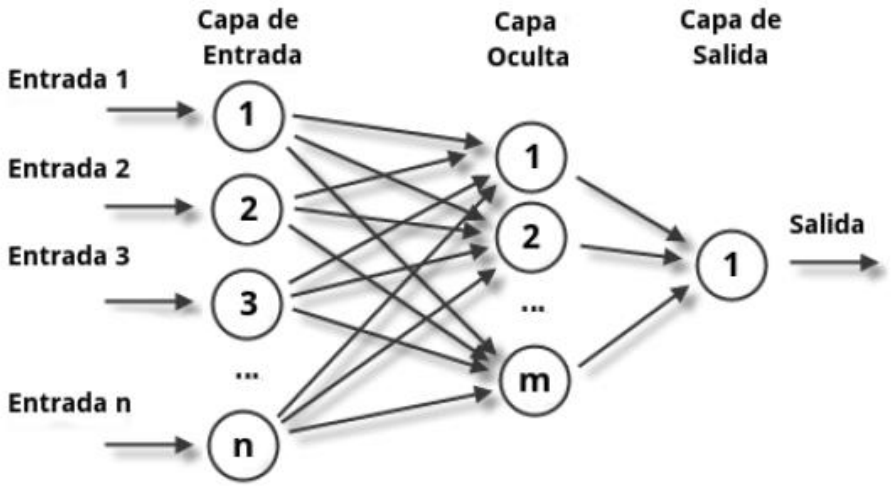


Figura 3.4.8. Estructura de una red neuronal.

3.5 Azure machine learning

Es un servicio en la nube que puede acelerar y gestionar el ciclo de vida de [18] proyectos de aprendizaje automático. Los profesionales del aprendizaje automático, los científicos de datos y los ingenieros pueden usarlo en sus flujos de trabajo diarios, como entrenar e implementar modelos y administrar MLOps (operaciones de aprendizaje automático).

Cree modelos en Azure Machine Learning o use modelos creados en plataformas de código abierto como Pytorch, TensorFlow y Scikit-learn. La herramienta MLOps ayuda

a monitorear, volver a entrenar y volver a implementar su modelo. Azure Machine Learning está destinado a personas y equipos que implementan operaciones de aprendizaje automático ("MLOps") dentro de una organización para operar modelos de aprendizaje automático en un entorno de producción seguro y auditable.

Los científicos de datos y los ingenieros de aprendizaje automático pueden encontrar herramientas para acelerar y automatizar sus flujos de trabajo diarios. Los desarrolladores de aplicaciones encuentran herramientas para integrar modelos en sus aplicaciones y servicios. Los desarrolladores de plataformas pueden encontrar un sólido conjunto de herramientas que aprovechan las API de larga duración de Azure Resource Manager para crear herramientas avanzadas de aprendizaje automático. Las empresas que trabajan en la nube de Microsoft Azure pueden aprovechar la seguridad familiar y el control de acceso basado en roles (RBAC) para su infraestructura.

Puede configurar su proyecto para denegar el acceso a los datos protegidos y seleccionar acciones. Azure Machine Learning le permite ejecutar sus scripts de entrenamiento en la nube o crear modelos desde cero. Los clientes suelen incorporar modelos creados y entrenados en marcos de código abierto y hacer que se ejecuten en la nube. Los científicos de datos pueden usar modelos creados con marcos populares de Python en Azure Machine Learning. por ejemplo.:

- PyTorch
- TensorFlow
- scikit-learn
- XGBoost
- LightGBM

También se admiten otros lenguajes y marcos, como:

- R
- .NET

En un proceso iterativo lento, los científicos de datos utilizan la experiencia y la intuición previas en el aprendizaje automático tradicional para seleccionar algoritmos y características de datos adecuados para el entrenamiento. El aprendizaje automático

automático (AutoML) acelera este proceso y está disponible a través de la interfaz de usuario de Studio o el SDK de Python.

La eficiencia del entrenamiento para el aprendizaje profundo y, a veces, los trabajos tradicionales de aprendizaje automático se pueden mejorar en gran medida mediante el entrenamiento distribuido de múltiples nodos. Los clústeres de proceso de Azure Machine Learning ofrecen las últimas opciones de GPU.

Las siguientes tecnologías de aprendizaje automático también son compatibles a través de Kubernetes conectado a Azure Arc (versión preliminar) y clústeres de proceso de Azure ML:

- PyTorch
- TensorFlow
- MPI

Capítulo 4. Resultados

Se implementó primero el algoritmo “Regresión Lineal”, con los datos ya obtenidos gracias a la página web de la universidad de Johns Hopkins, primero procedemos a hacer tratamiento de los datos.

En R Studio comenzamos exportando los datos de los CSV a una variable llamada “data1”, donde desde aquí podremos manipular y analizar los datos según sea necesario mediante la instrucción **read.csv.** Una vez hecho esto, debemos mostrar todos los detalles de los objetos en memoria, incluyendo las columnas de los marcos de datos (“data frames”), matrices y listas, lo cual puede generar una gran cantidad de información mediante el comando **str** y **summary** para mostrar información del tipo de objeto, así como las dimensiones (si es un data frame o similar), el nombre de las variables, los diez primeros elementos de las variables numéricas y los niveles o categorías de las variables categóricas, tal como se observa a continuación.

Después usamos **hist** para generar un histograma mostrando la relación entre el número de los casos y los fallecimientos en todo el mundo, por último una función **pairs y barplot** para tener una mejor noción y conocimiento de nuestros datos mediante las siguientes

gráficas. Con el fin de conocer las relaciones existentes entre cada par de variables podemos representar una matriz de diagramas de dispersión. Al parecer existe una relación lineal bastante clara entre **confirmed, deaths, recovered, actived**, pero no entre los otros dos pares de variables.

```
data1 = read.csv("C:/Users/user/Documents/Abdel/Tesis/Data/COVID-19-master/COVID-19-master/csse_cov

str(data1)
summary(data1)
hist(data1$Confirmed)
pairs(~Confirmed+Deaths+Recovered+Active, data = data1)
barplot(table(data1$Country_Region))
```

Figura 4.0.1. Código lectura y análisis del data set.:

```
> summary(data1)
      FIPS          Admin2          Province_State     Country_Region     Last_Update
 Min.   :   66   Length:4000        Length:4000        Length:4000        Length:4000
 1st Qu.:19049   Class :character   Class :character   Class :character   Class :character
 Median :30067   Mode  :character   Mode  :character   Mode  :character   Mode  :character
 Mean   :32387
 3rd Qu.:47039
 Max.   :99999
 NA's   :735
      Lat              Long_            Confirmed            Deaths          Recovered
 Min.   :-52.37   Min.   :-174.16   Min.   :     0.0   Min.   :    0.0   Min.   :     0
 1st Qu.: 33.21   1st Qu.: -96.58   1st Qu.:   700.8   1st Qu.:    9.0   1st Qu.:     0
 Median : 37.92   Median : -86.75   Median :  1964.0   Median :   31.0   Median :     0
 Mean   : 35.85   Mean   : -71.33   Mean   : 21051.1   Mean   :  472.7   Mean   : 11840
 3rd Qu.: 42.19   3rd Qu.: -77.39   3rd Qu.:  7752.8   3rd Qu.:  113.0   3rd Qu.:     0
 Max.   : 71.71   Max.   : 178.06   Max.   :2636045.0   Max.   :64731.0   Max.   :2114760
 NA's   :88       NA's   :88
     Active         Combined_Key       Incident_Rate    Case_Fatality_Ratio
 Min.   :-1083768   Length:4000        Min.   :    0   Min.   : 0.0000
 1st Qu.:     578   Class :character   1st Qu.: 3560   1st Qu.: 0.9636
 Median :    1657   Mode  :character   Median : 5840   Median : 1.5339
 Mean   :    8707                      Mean   : 5747   Mean   : 1.9314
 3rd Qu.:    4874                      3rd Qu.: 7758   3rd Qu.: 2.3070
 Max.   : 2400750                      Max.   :27388   Max.   :271.8750
```

Figura 4.0.2. Vista general del data set.

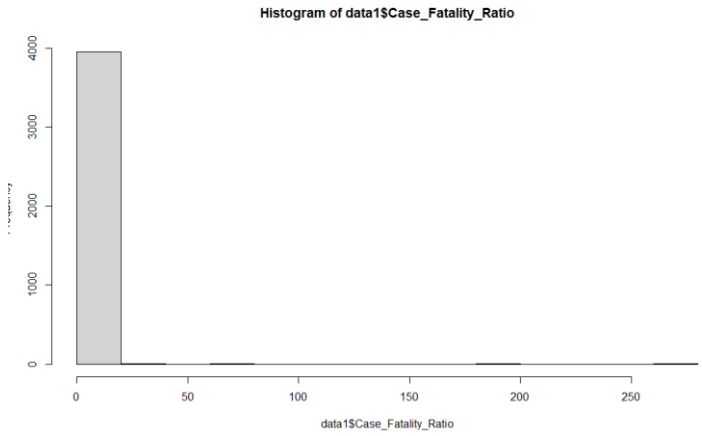


Figura 4.0.3. Histograma de frecuencia para el porcentaje de mortalidad en el dataset.:

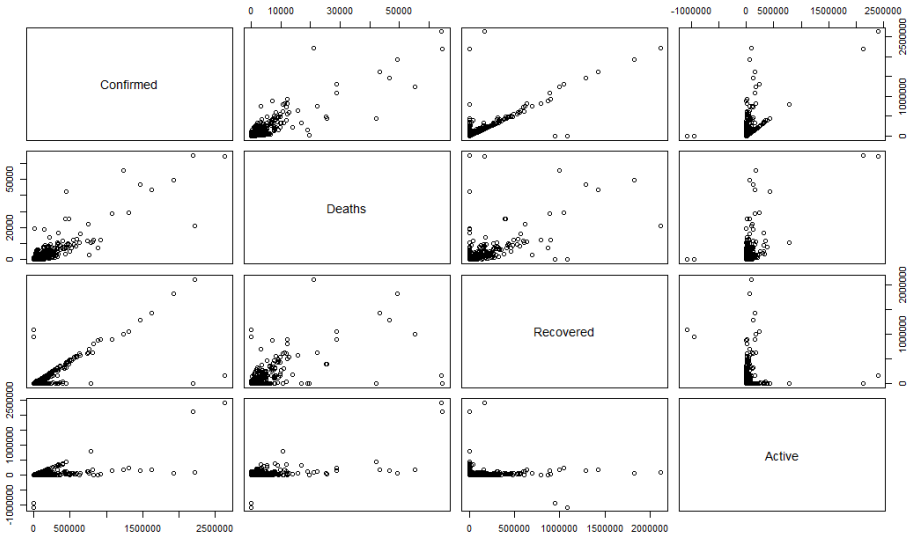


Figura 4.0.4. Relaciones entre las variables confirmaciones, defunciones, recuperaciones, activos dentro del dataset.

Una vez mostrado una perspectiva general de nuestro conjunto de datos covid-19, procedemos con una limpieza de los datos para minimizar los errores y tener mejor precisión al analizar con los algoritmos.

Procedemos a utilizar las siguientes instrucciones para la limpieza de los datos, mediante unas sencillas operaciones para disminuir el rango de los datos e incrementar la precisión.

```
#Outlier treatment

summary(data1$Confirmed)

quantile(data1$Confirmed, 0.99)
uv = 3*quantile(data1$Confirmed, 0.99)
data1$Confirmed[data1$Confirmed > uv] <- uv

summary(data1$Confirmed)
```

Figura 4.0.5. Código para la limpieza de los datos

Además, contamos con una función que permite completar los valores faltantes dentro de nuestro csv, para lo cual, usamos principalmente la función **mean** para encontrar la media de valores dentro de la tabla y así poderlos rellenar en con valores aproximados, tal como se muestra en la siguiente imagen.

```
#Missing value imputation
summary(data1$Case_Fatality_Ratio)

mean(data1$Case_Fatality_Ratio) #promedio tomando valores faltantes y no faltantes
mean(data1$Case_Fatality_Ratio, na.rm = TRUE) #promedio tomando cada valor faltante como true
which(is.na(data1$Case_Fatality_Ratio)) #posiciones de la tabla para valores faltantes
```

Figura 4.0.6. Código para tratar valores null o faltantes

Una vez hecho la limpieza, procedemos a implementar el algoritmo de regresión lineal simple, el comando básico es **lm** (linear models). El primer argumento de este comando es una fórmula y ~ x en la que se especifica cuál es la variable respuesta o dependiente (y) y cuál es la variable regresora o independiente (x). El segundo argumento, llamado data especifica cuál es el fichero en el que se encuentran las variables. El resultado lo guardamos en un objeto llamado regresión. Este objeto es una lista que contiene toda la información relevante sobre el análisis. Mediante el comando summary obtenemos un resumen de los principales resultados: ,el cual nos basta con las siguientes líneas.

```
#Simple Linear Regresion

simple_model <- lm(data1$Deaths ~ data1$Confirmed, data1=df)

summary(simple_model)

plot(data1$Confirmed, data1$Deaths)
abline(simple_model)
```

Figura 4.0.7. Código para algoritmo regresión lineal

Los siguientes comandos representan la nube de puntos (comando plot) y añaden la representación gráfica de la recta de mínimos cuadrados (comando abline aplicado al objeto generado por lm). Obteniendo la siguiente gráfica.

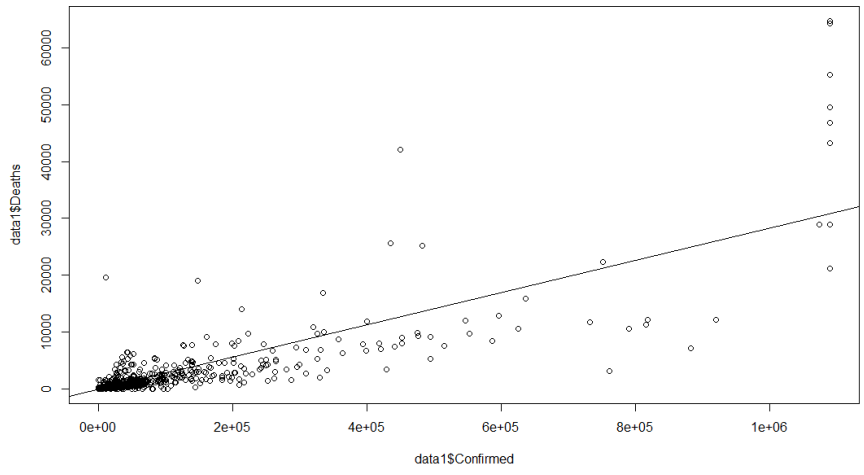


Figura 4.0.8. Regresión lineal entre las variables de casos confirmados y defunciones

Posteriormente implementamos el algoritmo llamado “subset selection” el cual la función **regsubsets()** se puede usar para identificar diferentes mejores modelos de diferentes tamaños. Debe especificar la opción nvmax, que representa el número máximo de predictores para incorporar en el modelo. Por ejemplo, si nvmax = 5, la función devolverá hasta el mejor modelo de 5 variables, es decir, devolverá el mejor modelo de 1 variable, el mejor modelo de 2 variables, …, los mejores modelos de 5 variables.

En nuestro ejemplo, solo tenemos 5 variables predictoras en los datos. Entonces, usaremos nvmax = 5.

La función **summary()** informa el mejor conjunto de variables para cada tamaño de modelo. Del resultado anterior, un asterisco especifica que una determinada variable está incluida en el modelo correspondiente.

Posteriormente, procederemos a implementar el algoritmo de **K – Means,** para el cual utilizaremos la librería **cluster,** y mediante una función seremos capaces de generar 3 clúster mediante los datos de personas confirmadas, defunciones y promedio de mortalidad iniciando con una seed “semilla” de 240 para que cada vez que nuestra función se vuelva a ejecutar se capaces de producir resultados aleatorios beneficiando la variedad de resultados en el algoritmo, tal cómo se muestra a continuación.

```
library(cluster)

# Removing initial label of
# Characters colum from original dataset
kmeansdata = data.frame(data1$Confirmed, data1$Deaths, data1$Recovered, data1$Active,
                        data1$Case_Fatality_Ratio, data1$CSR_Level)

# Fitting K-Means clustering Model
# to training dataset
set.seed(240) # Setting seed
kmeans.re <- kmeans(na.omit(kmeansdata),3)
kmeans.re

# Cluster identification for
# each observation
kmeans.re$cluster
```

Figura 4.0.9. Código para implementar algoritmo K-means

Los grupos que se formarán serán por compartir características similares, por lo cual, por ser un método no supervisado el conjunto de datos deberá darle una interpretación a la clasificación realizada.

Para ello, el algoritmo identificará K centroides, los cuales compararán su valor medio con el de los datos restantes, seleccionando aquellos más cercanos para realizar los clúster o agrupamiento. Obtenemos los siguientes resultados de los 3 clúster creados mediante los datos dentro de nuestro dataset.

```
K-means clustering with 3 clusters of sizes 87, 3896, 16

Cluster means:
  data1.Confirmed data1.Deaths data1.Recovered data1.Active data1.Case_Fatality_Ratio
1      308572.230    6733.7586      252497.368    49566.471                 2.094434
2        9238.229     211.0364        2701.183     6286.747                 1.926950
3      966926.615   30170.6250      929251.875   375970.375                 2.132583
```

Figura 4.0.10. Resultado algoritmos K-means

A continuación, mostramos su respectiva matriz de confusión generada mediante el siguiente código.

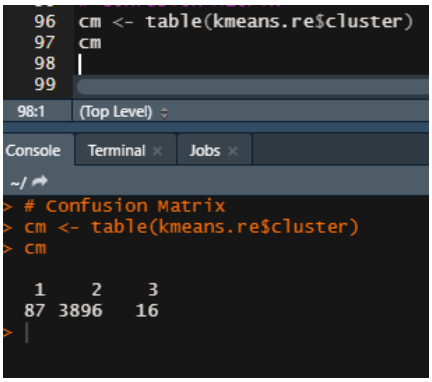


Figura 4.0.11. Matriz de confusión K-means

Ahora, para generar las gráficas de nuestros clústeres para este algoritmo, implementamos también un algoritmo llamado CLARA, el cual, en comparación con otros métodos de particionamiento como pam, puede manejar conjuntos de datos mucho más grandes. Internamente, esto se logra considerando subconjuntos de datos de tamaño fijo (samesize) de modo que los requisitos de tiempo y almacenamiento se vuelvan lineales en n en lugar de cuadráticos.

Cada subconjunto de datos se divide en k grupos utilizando el mismo algoritmo que en pam. Una vez que se han seleccionado objetos representativos del subconjunto de datos, cada observación de todo el conjunto de datos se asigna al centroide más cercano. La media (equivalente a la suma) de las diferencias de las observaciones con su centroide

más cercano se usa como una medida de la calidad de la agrupación. Se retiene el subconjunto de datos para el cual la media (o suma) es mínima. Un análisis adicional se lleva a cabo en la partición final, todo gracias a la función **pam,** tal cómo se muestra en un momento dentro de la librería **factorextra**

```
library(cluster)    # clustering algorithms
library(factoextra) # clustering algorithms & visualization

pam_clusters <- pam(x = kmeansdata, k = 5, metric = "manhattan")
pam_clusters



print(clara_clusters)

dd <- cbind(kmeansdata, cluster = clara_clusters$clustering)
head(dd, n = 4)

# Medoids
clara_clusters$medoids

# Clustering
```

Figura 4.0.12. Código algoritmo CLARA clustering

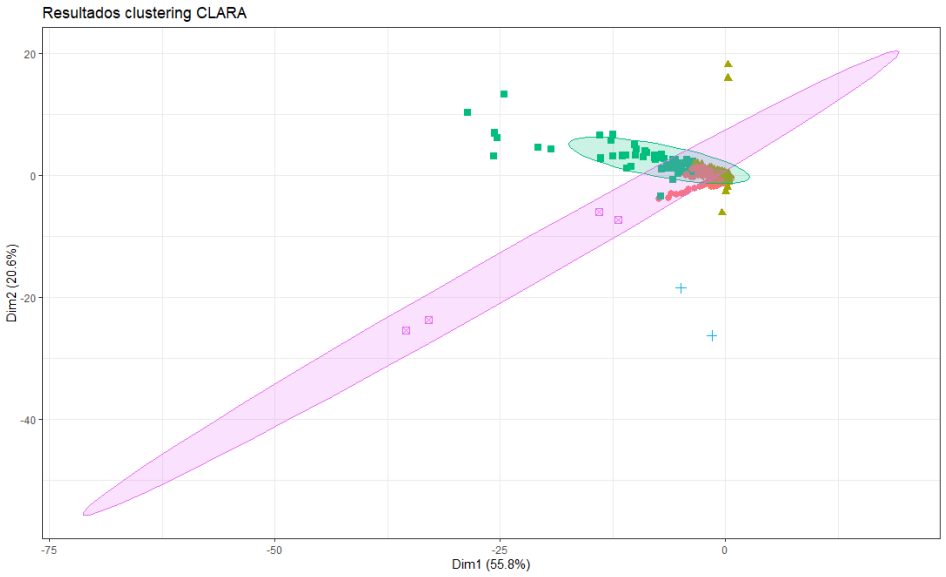


Figura 4.0.13. Visualización CLARA clustering

Se muestra a continuación los resultados del análisis para la clasificación mediante clústeres.

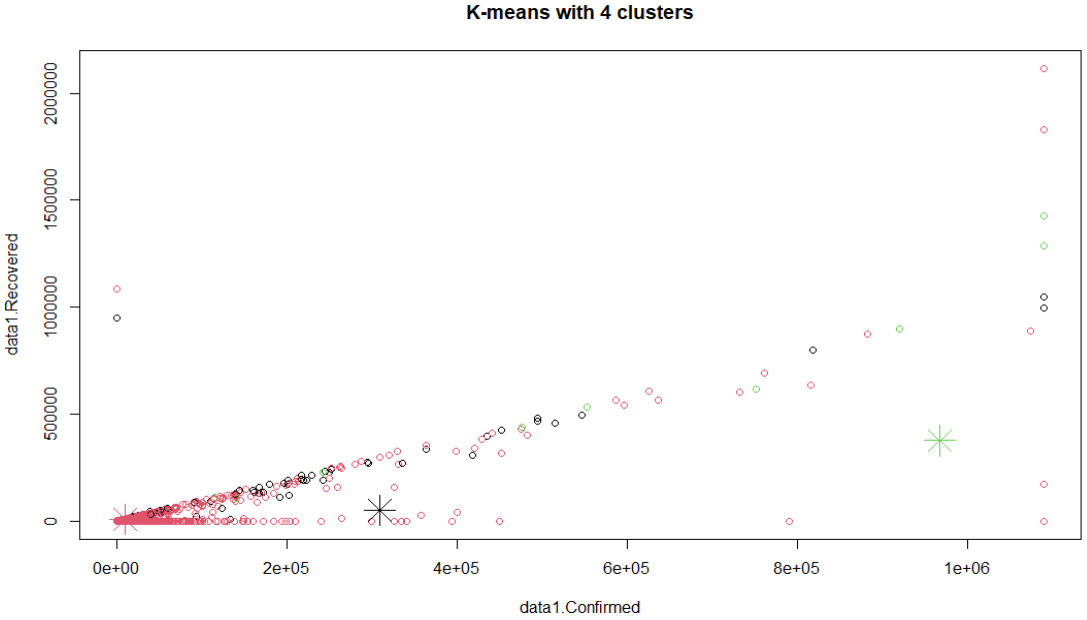


Figura 4.0.14. Resultados K-means con 4 clusters

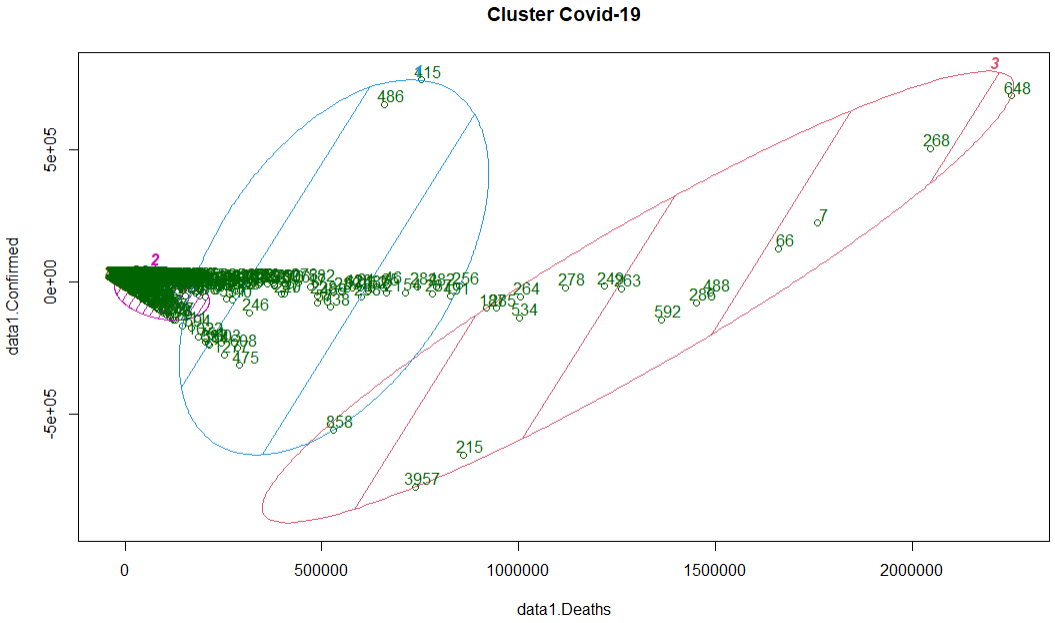


Figura 4.0.15. Clustering K-means entre variables casos confirmados y defunciones.

```
> # Clustering
> head(clara_clusters$clustering, 10)
 1  2  3  4  5  6  7  8  9 10
 1  1  1  2  2  2  3  1  2  2
```

Figura 4.0.16. Resultados CLARA Clustering

A continuación, analizaremos uno de los algoritmos deep learning más famosos y empleados en el mundo de la tecnología actual, las “redes neuronales”. Donde, cómo habíamos platicado, se usa principalmente para las siguientes situaciones.

- Aproximación de funciones, o el análisis de regresión, incluyendo la predicción de series temporales, funciones de aptitud y el modelado.
- Clasificación, incluyendo el reconocimiento de patrones y la secuencia de reconocimiento, detección y de la toma de decisiones secuenciales.
- Procesamiento de datos, incluyendo el filtrado, el agrupamiento, la separación ciega de las señales y compresión.
- Robótica, incluyendo la dirección de manipuladores y prótesis.
- Ingeniería de control, incluyendo control numérico por computadora.

Para poder utilizar este poderoso algoritmo, necesitamos utilizar la paquetería **neural net.**

Antes de crear la red es necesario escalar las variables para evitar el efecto de la escala de las variables. Existen varias formas de escalar, pero se usará una transformación para pasar los valores de las variables al intervalo (0,1). Con el siguiente código se va a convertir los datos originales a datos escalados y se almacenarán en el objeto scaled.

```
library(ggplot2)                                    # Se carga cada session
ggplot(kmeansdata, aes(x=data1.Confirmed, y=data1.Deaths)) + geom_point()
library(neuralnet)
head(kmeansdata)
maxs <- apply(kmeansdata, 2, max) # Máximo valor de las variables
mins <- apply(kmeansdata, 2, min) # Mínimo valor de las variables
scaled <- as.data.frame(scale(kmeansdata, center=mins, scale=maxs-mins))

head(cbind(kmeansdata, scaled))

#install.packages("gridExtra", dependencies = TRUE) # Se instala una sola vez
require(gridExtra)                                  # Se carga cada session
plot1 <- ggplot(kmeansdata, aes(x=data1.Confirmed, y=data1.Deaths)) + geom_point()
plot2 <- ggplot(scaled, aes(x=data1.Confirmed, y=data1.Deaths)) + geom_point()
grid.arrange(plot1, plot2, ncol=2)
```

Figura 4.0.17. Escalar *datos para redes neuronales*

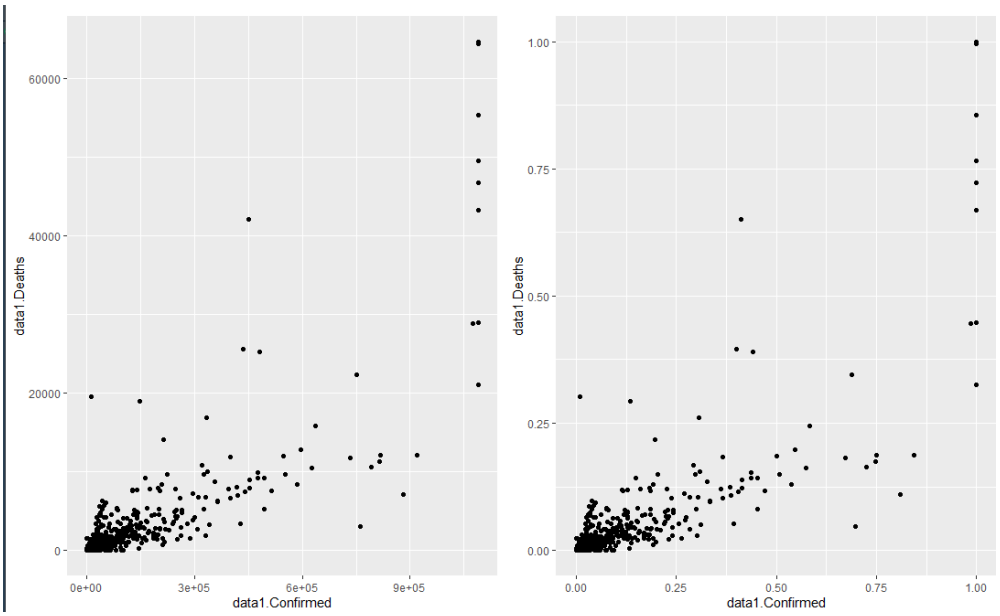


Figura 4.0.18. Visualización variables escaldas

En la siguiente figura se muestra el diagrama de dispersión para las variables escaladas. Al comparar ambos gráficos de dispersión (con datos y con scaled) se observa el mismo patrón en la nube de puntos, la única diferencia es que ahora los valores de las variables (Y y X) están en (0,1).

El elemento act.fct es la función de activación (logística por defecto) y el elemento weights contiene los pesos mostrados en la anterior figura. Estos dos elementos serán útiles más adelante. A continuación, el código para explorar lo que hay dentro de estos dos elementos.

```
> #Funcion de Activacion
> mod1$act.fct # Activation function
function (x)
{
    1/(1 + exp(-x))
}
<bytecode: 0x0000023eaacee970>
<environment: 0x0000023eaaceb438>
attr(,"type")
[1] "logistic"
```

Figura 4.0.19. Función de activación para red neuronal

Para la predicción de nuestros datos, usaremos el elemento $net.result del objeto myprediction, el cual, tiene la respuesta estimada pero en la forma escalada, por esta razón es necesario aplicar la transformación inversa para obtener el resultado en la escala original. A continuación, el código necesario para retornar a la escala original.

```
#El elemento $net.result del objeto myprediction tiene la respuesta estimada pero
#en la forma escalada, por esta razón es necesario aplicar la transformación inversa
#para obtener el resultado en la escala original. A continuación el código necesario
#para retornar a la escala original.
yhat_red <- myprediction$net.result * (max(kmeansdata$data1.Confirmed)-min(kmeansdata$dat
kmeansdata$yhat_red <- yhat_red
yhat_red[1:5] # Para ver los primeros 5 valores estimados
#yhat_red[1:200]
```

Figura 4.0.20. Código para predecir resultados de la red neuronal con escalda adecuada

Para comparar los resultados obtenidos con la red neuronal podemos dibujar los valores observados y contra los valores estimados de la variable respuesta. A continuación, se muestra el código para crear el gráfico de dispersión al cual se le agrega una línea recta a 45 grados como referencia; entre más cerca este un punto de la línea significa que la respuesta es lo más estimada posible y son cercanos, por lo que mostraremos cómo está conformada nuestra red neuronal, donde decidimos crearla con 2 capas y 3 nodos para hacer un análisis muy interesante.

Donde observamos los siguientes resultados.

```
> unlist(mod1$weights)  # Obtener en formas de vector los weigths=pesos
 [1]  -0.10172359 -11.49175599   0.30196737  -1.99177712  -0.84353207   0.45251030   0.08687909
 [8]  -0.67762314   0.72918272   1.77916692   1.57857232  -1.37324617  -2.91748511  -0.60072417
[15]   0.34809502  -0.32362625   2.16121457

> test <- data.frame(data1.Confirmed = scaled$data1.Confirmed)  #Conjunto de entrenamiento

> myprediction <- compute(x=mod1, covariate=test)

> myprediction$net.result[1:5] #Predicciones futuras
[1] 0.13033596 0.14437414 0.24060016 0.02149978 0.04461357
```

Figura 4.0.21. Resultados iniciales de la predicción de la red neuronal

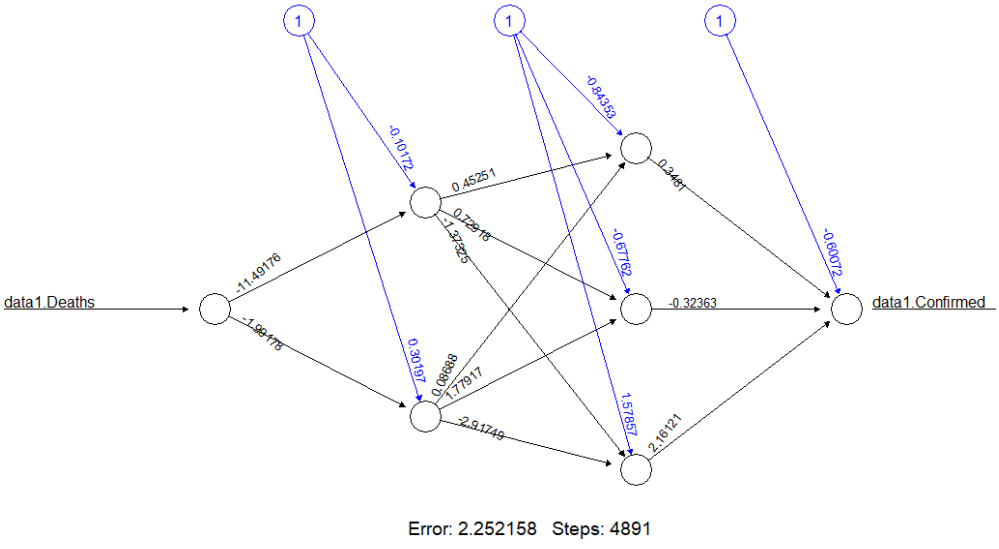


Figura 4.0.22. Estructura de red neuronal con 2 capas y 3 nodos

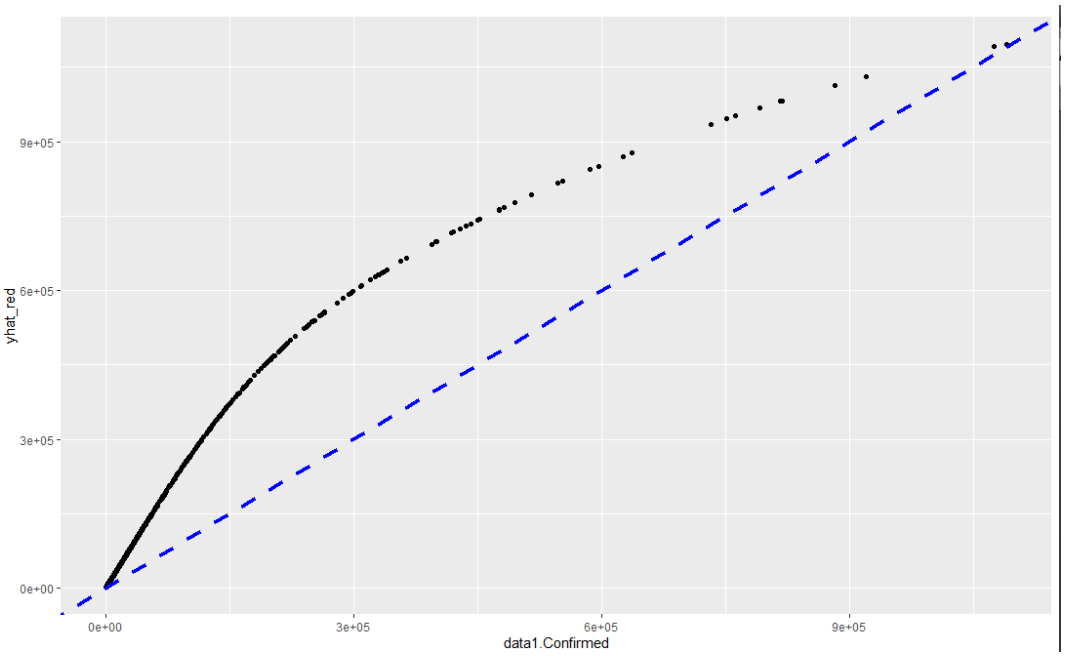
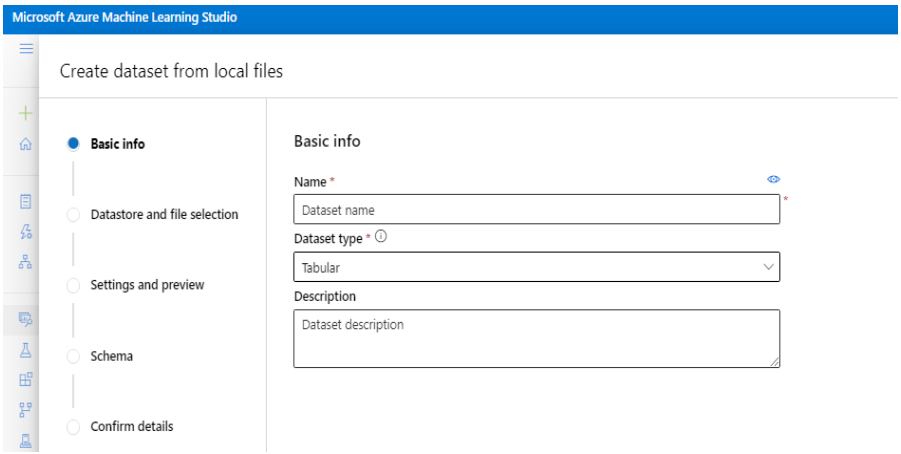


Figura 4.0.23. Comparación de resultados sin predicción contra los resultados escalados que ya predecimos

Por último, mostramos cómo la nube puede ser de gran ayuda para la inteligencia artificial gracias a Azure Virtual Studio, donde de manera grafica te solicita el conjunto de datos y el algoritmo a implementar, en este caso aplicaremos un algoritmo de regresión lineal.

Primero nos solita el CSV en donde Azure nos proporciona las herramientas para cargar nuestro dataset desde nuestro equipo y subirlo a la nube, tal cómo se muestra a continuación.



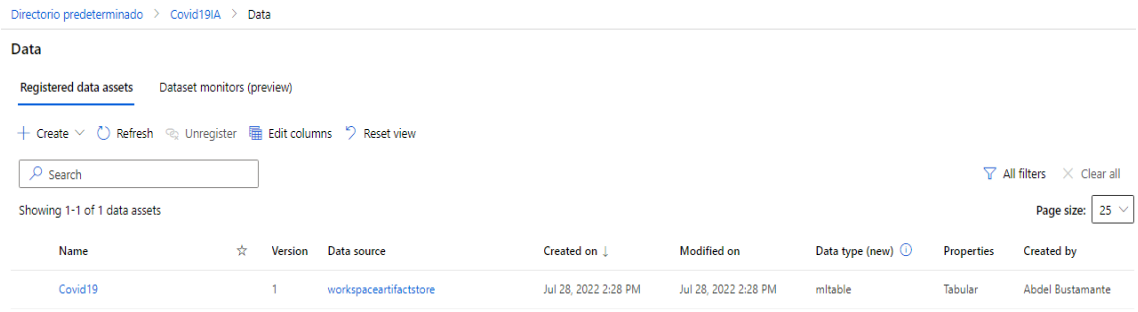


Figura 4.0.24. Carga del dataset a Azure Machine Learning Studio

Posteriormente procedemos a crear nuestro pipeline, en donde aquí definiremos que elementos queremos analizar en nuestro algoritmo, para lo cual, contamos con diversas opciones e incluso algoritmos machine learning/deep learning para implementar.

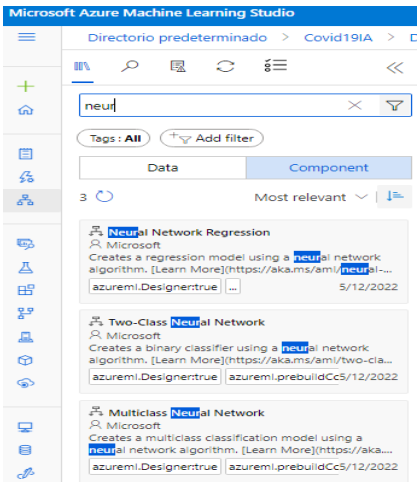


Figura 4.0.25. Componentes/Algoritmos en Azure Machine Learning Studio

Para esta propuesta, utilizaremos los componentes de.

- Covid Dataset
- Select Columns in Dataset
- Clean Missing Data
- Normalize Data
- Split Data
- Train Model
- Linear Regresion Model
- Score Model

- Evaluate Model
- Con esto, para poder ejecutar el algoritmo de regresión lineal, utilizaremos una estructura similar a la siguiente.

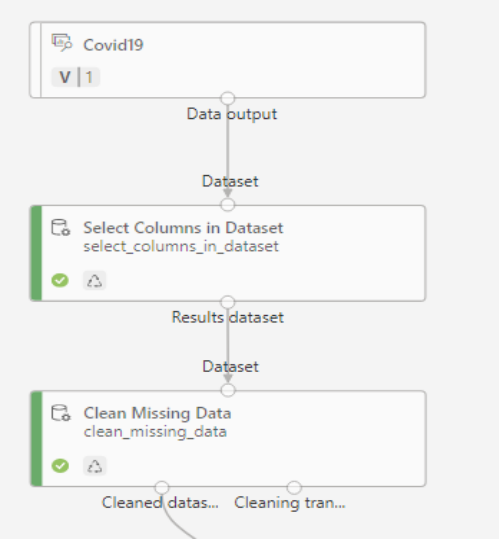


Figura 4.0.26. Uso, limpieza y selección de columnas del dataset en Azure Machine Learning Studio

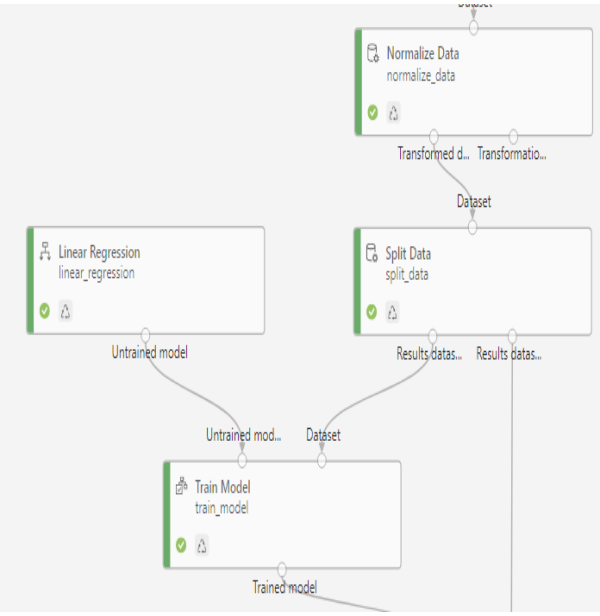


Figura 4.0.27. Normalización, división de los datos para posteriormente entrarlos y aplicar regresión lineal

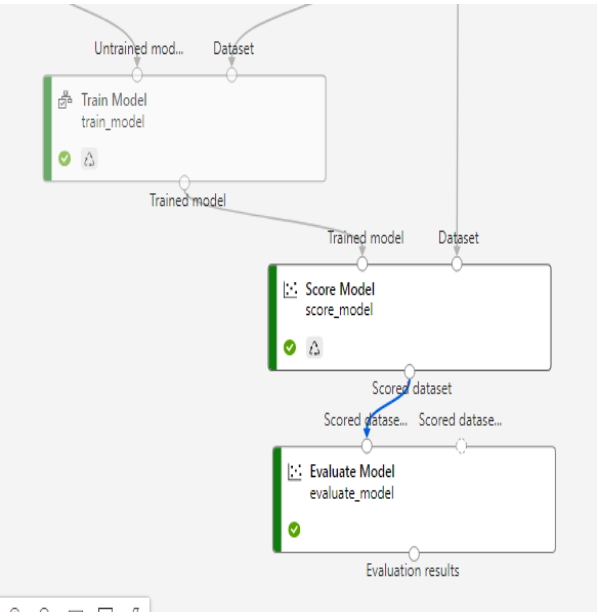


Figura 4.0.28. Por último, calificamos y evaluamos la precisión del modelo

Y una vez teniendo todos estos procesos, procedemos a correr nuestro modelo, para lo cual contamos fue ejecutado de manera correcta sin ningún problema, obteniendo los siguientes resultados.

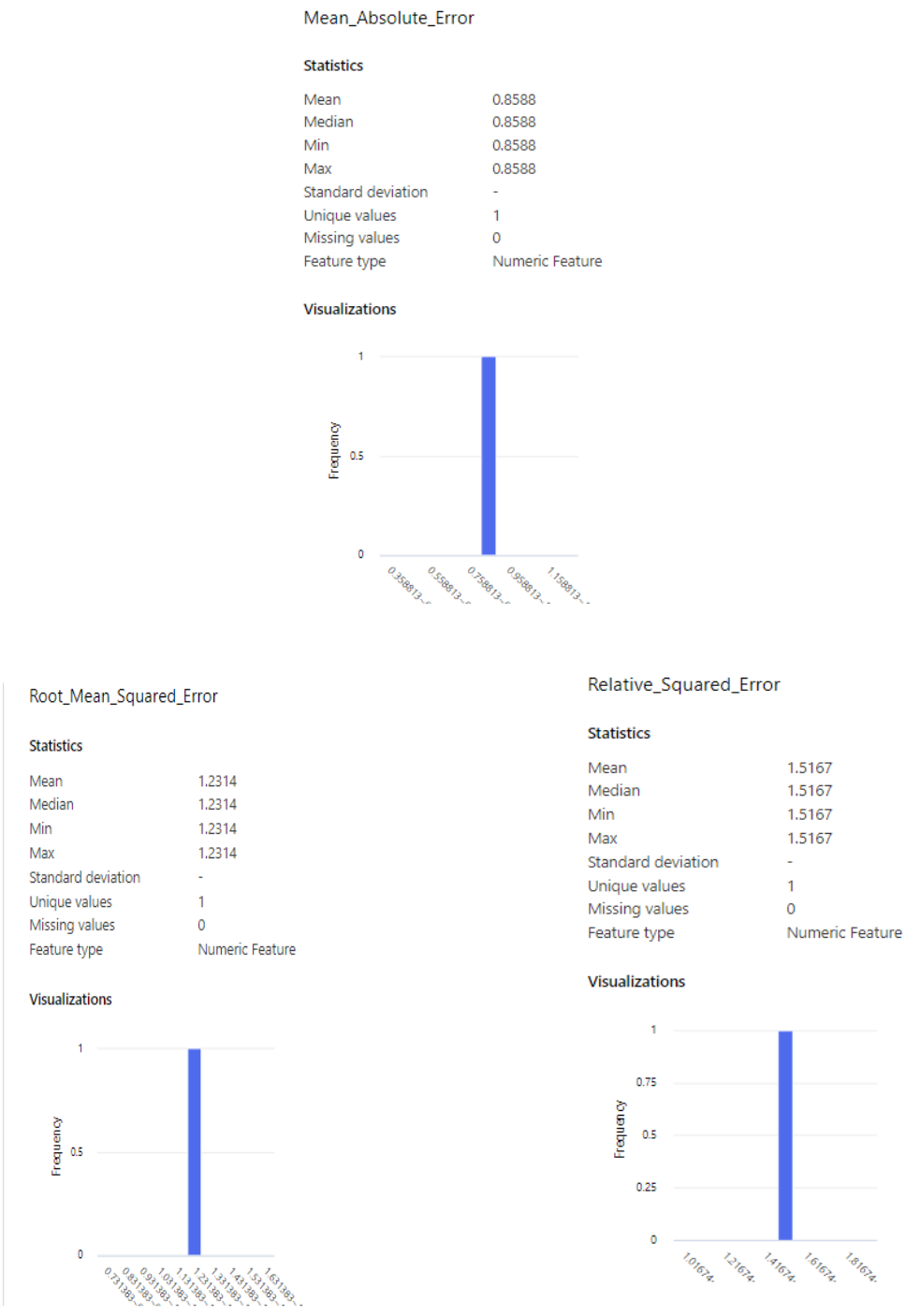


Figura 4.0.29. Resultados estadísticos regresión lineal de Azure Machine Learning Studio

Nos muestra los siguientes resultados. Actualmente, existen varios métodos para estimar el rendimiento y evaluar el ajuste del modelo, algunos de los cuales son: Root-square-

Error (RMSE), Mean Absolute Error (MAE, Mean Absolute Error), R-square. La medida más utilizada para las tareas de regresión es el error cuadrático medio (RMSE) y es la raíz cuadrada de la distancia cuadrática media entre los valores reales y esperados.

Indica el ajuste absoluto del modelo a los datos y qué tan cerca están los puntos de datos observados de los valores esperados del modelo. El error cuadrático medio, o RMSE, es una medida absoluta de relevancia. En este caso, proporcionó resultados prometedores. Un valor RMSE más bajo indica un mejor ajuste. El RMSE es una buena medida de qué tan bien un modelo predice una respuesta y es el criterio más importante para evaluar el ajuste si el propósito principal del modelo es predecir. En este caso, se muestran resultados muy débiles, por lo que nuestro modelo y predicción son muy prometedores para el análisis.

Figura 4.0.30. Resultados estadísticos regresión lineal de Azure Machine Learning Studio

Por otro lado, también nos muestra el error absoluto medio (MAE). Es la diferencia media absoluta entre los valores observados y esperados. El error absoluto medio o MAE es un punto lineal, lo que significa que todas las diferencias individuales se ponderan por igual en la media. Cómo se mencionó, no hay problema para analizar este

El coeficiente de determinación es un número entre 0 y 1 que mide qué tan bien el modelo estadístico predice el resultado. Por lo general, se escribe cómo R2 y se pronuncia cómo "r al cuadrado". Para la regresión lineal simple, a menudo se usa una r minúscula en lugar de (r2). Nuestros resultados no son cero, por lo que se pueden predecir muchas observaciones sin ningún problema, lo que hace una buena predicción para ellas.

Capítulo 5. Conclusiones y trabajo a futuro

A través de esta propuesta de tesis hemos analizado diferentes propuestas de como analizar una gran cantidad de información de diferentes maneras y con una gran variedad de usos dependiendo la situación. Observamos que por la pandemia mundial debido al virus SARS-CoV-2 (COVID-19) impactó de una manera descomunal en todas las áreas del mundo, en especial el área de IT, en donde se generó nuevas modalidades de organización, trabajo e información que serviría a la sociedad con predicciones de posibles eventos futuros, así como para una mejor toma de decisiones.

Mucho de esto ha sido gracias a la inteligencia artificial, que permite cada día a la sociedad una mejor comprensión de lo que sucede a nuestro alrededor dependiendo las circunstancias. Se demostró que ya sean algoritmos de aprendizaje automático (machine learning) o aprendizaje profundo (deep learning) son de gran ayuda, cuentan con diversos funcionalidades y sobre todo que pueden ser adaptables a cualquier situación que el mundo requiera, y acompañado de la tecnología como lo es la computación en la nube (cloud computing), donde otorga un conjunto de herramientas de alta confianza e infraestructura con alta disponibilidad y agilidad ante cualquier posible situación que pueda suceder, para lo cual la IA puede estar implementarse casi en cualquier área o sector dentro de la industria, donde veremos en un futuro hasta donde nos lleva estas tecnologías que avanzan cada vez más rápido conforme pasan los años y que sea siempre en favor de la humanidad.

Bibliografía

Aghion, P. J. (2019). *Artificial Intelligence and Economic Growth.* (pp. 237-290). University of Chicago Press.

Aizawa, K. (1992). *Connectionism and artificial intelligence: History and philosophical interpretation. Journal of Experimental & Theoretical Artificial Intelligence.*

Alain, G. a. (2013). *What regularized auto-encoders learn from the data generating distribution.* ICLR'2013 . also arXiv report.

APD, R. (2019, Abril 4). *¿Cuáles son los tipos de algoritmos del machine learning?* Retrieved from apd: https://www.apd.es/algoritmos-del-machine-learning/#:~:text=Se%20utilizan%20en%20el%20aprendizaje,representados%20por%20la%20variable%20K.

Brian, O. (2012, Noviembre 6). *Cloud Computing.* Retrieved from widmer: https://widmer.ch/fileadmin/templates/publikationen/20121106_SATW_WhitePaper_CloudComputing_EN.pdf

Deshmukh, R. R. (2019). *Data Cleaning: Current Approaches and Issues.*

Iberdrola. (2022, Agosto 30). *Descubre los principales beneficios del 'Machine Learning'.* Retrieved from iberdrola: https://www.iberdrola.com/innovacion/machine-learning-aprendizaje-automatico

Krizhevsky, A. S. (2012). *ImageNet classification with deep convolutional neural networks. In Proc. Advances in Neural Information Processing Systems.*

L., J. J. (n.d.). *Bases de datos: Modelos, lenguajes, diseño", 1era. Edición.* México: Oxford University Press México.

Lateef, Z. (2021, Diciembre 17). *Introduction To Machine Learning: All You Need To Know About Machine Learning*. Retrieved from edureka!: https://www.edureka.co/blog/introduction-to-machine-learning/

Microsoft. (2022, Julio 27). *¿Qué es Azure Machine Learning?* Retrieved from Microsoft Learn: https://docs.microsoft.com/es-es/azure/machine-learning/overview-what-is-azure-machine-learning

Microsoft. (2022, Julio 27). *Cognitive Services y Machine Learning.* Retrieved from Microsoft Learn: https://docs.microsoft.com/es-mx/azure/cognitive-services/cognitive-services-and-machine-learning

Microsoft. (2022, Julio 30). *Introducción a los aspectos básicos de Azure*. Retrieved from Microsoft Learn: https://docs.microsoft.com/es-mx/learn/modules/intro-to-azure-fundamentals/

Moujahid, A. (1991, Octubre 4). *Clasificadores K-NN*. Retrieved from sc: http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/t9knn.pdf

Nielsen, M. (n.d.). *Neural Networks and Deep Learning(Redes neuronales y aprendizaje profundo.*

Sotaquirá, M. (2018, Julio 16). *La Regresión Lineal en el Machine Learning*. Retrieved from codificandobits: https://www.codificandobits.com/blog/regresion-lineal/#:~:text=La%20relaci%C3%B3n%20lineal%20entre%20dos,intersecci%C3%B3n%20con%20el%20eje%20y.

Zúñiga, F. G. (2020, Octubre 2). *RStudio, IDE para programar con R. Instalación y primeros pasos*. Retrieved from arsys: https://www.arsys.es/blog/rstudio