Estimación de los niveles futuros de material particulado en el aire de Medellín y su área metropolitana

Santiago Larrea Henao

Trabajo de investigación para optar al título de Ingeniero Electrónico

Asesor
Gustavo Adolfo Patiño Álvarez, Doctor (PhD)

Universidad de Antioquia
Facultad de Ingeniería

Ingeniería Electrónica

Medellín, Antioquia, Colombia

2022

Cita		Larrea Henao [1]
Referencia Estilo IEEE (2020)	[1]	S. Larrea Henao, “Estimación de los niveles futuros de material particulado en el aire de Medellín y su área metropolitana”, Trabajo de grado profesional, Ingeniería Electrónica, Universidad de Antioquia, Medellín, Antioquia, Colombia, 2022.

Grupo de Investigación Sistemas Embebidos e Inteligencia Computacional (SISTEMIC).

UNIVERSIDAD DE ANTIOQUIA
Vicerrectoría de Docencia

Sistema de Bibliotecas

Repositorio Institucional: http://bibliotecadigital.udea.edu.co

Universidad de Antioquia - www.udea.edu.co

Rector: John Jairo Arboleda Céspedes.

Decano/Director: Jesús Francisco Vargas Bonilla.

Jefe departamento: Augusto Enrique Salazar Jiménez.

Resumen

La contaminación del aire es un tema de gran relevancia en el mundo y particularmente en la ciudad de Medellín, debido a que trae grandes afectaciones a la salud de los habitantes y puede ser el causante de enfermedades respiratorias e incluso la muerte. Con la finalidad de alertar de manera temprana a las autoridades ambientales y ciudadanía en general, el trabajo realizado en el presente proyecto consiste en realizar la estimación de los niveles futuros de material particulado presentes en la atmósfera de Medellín y su área metropolitana con dos días de anticipación. Esta estimación es realizada haciendo uso de técnicas de aprendizaje de máquina, como el regresor de bosque aleatorio y redes neuronales artificiales. Los resultados obtenidos muestran un error MAE de 0.058, RMSE de 0.077 y MSE de 0.0059 en la estimación de estos niveles.

Tabla de contenidos

1. Introducción

La ciudad de Medellín se encuentra en constante crecimiento, poblacional, industrial, de su parque automotor y económico, siendo reconocida en el pasado como la ciudad más innovadora del mundo. A pesar de que esto trae consigo múltiples beneficios para la ciudad, también puede generar problemas en materia de contaminación del aire, si no se toman las precauciones necesarias. La contaminación del aire es un problema que nos concierne a todos, debido a que puede ser el causante de grandes afectaciones de la salud de la población e incluso de la muerte.

El presente proyecto busca realizar una estimación a futuro de los niveles de material particulado presentes en la atmósfera de Medellín y su área metropolitana, con la finalidad de entender el comportamiento de este, en relación con las diferentes variables meteorológicas y las emisiones de gases de las diferentes fuentes fijas y móviles. Entender cómo se comporta este fenómeno, permitirá conocer con antelación los niveles, aproximados, de material particulado que habrá en la atmósfera y alertar de manera temprana para la toma de medidas que permitan contrarrestar el efecto de estas fuentes.

El desarrollo de este proyecto comienza realizando la recolección de los datos históricos de los niveles de material particulado medido por cada una de las estaciones de calidad del aire del Sistema de Alerta Temprana del Valle de Aburrá (SIATA) y los datos históricos de las medidas de las diferentes variables meteorológicas, tales como: humedad relativa, presión atmosférica, temperatura, velocidad y dirección del viento. Luego, estos datos son pasados por un proceso de limpieza de la información con la finalidad de eliminar datos erróneos y/o que presenten un comportamiento atípico en relación con la totalidad de la información. Luego, los datos limpiados son usados para entrenar un modelo de regresor de bosque aleatorio y una red neuronal artificial que servirán para realizar la estimación de los niveles futuros de material particulado presentes en la atmósfera de Medellín y su área metropolitana. Finalmente, se realiza un análisis de los resultados obtenidos.

Este trabajo está organizado de la siguiente manera: la sección 4 presenta el marco teórico del presente proyecto, la sección 5 presenta la metodología seguida, la sección 6 presenta el análisis de los resultados obtenidos y la sección 7 presenta las conclusiones.

2. Objetivos

Objetivo general:

Implementar algoritmos para la estimación de las concentraciones futuras de material particulado presente en la atmósfera de Medellín y su área metropolitana, teniendo en cuenta la caracterización y clasificación de las variables meteorológicas de mayor influencia en la calidad del aire identificados en las múltiples bases de datos de entidades de control como el SIATA, considerando el uso de técnicas de Aprendizaje de Máquina.

Objetivos específicos:

1. Recolectar información histórica de variables meteorológicas y de contaminación del aire, como lo son los niveles de PM2.5 y PM10, de las diferentes plataformas de medición que dispone la Alcaldía de Medellín y el área Metropolitana del Valle de Aburrá.
2. Realizar la evaluación experimental de diversas técnicas de aprendizaje de máquina apropiadas para la identificación y estimación de niveles futuros de material particulado presente en la atmósfera y su relación con variables meteorológicas en la región.
3. Realizar un análisis para la identificación de las variables meteorológicas de mayor influencia en relación con la contaminación del aire dentro de Medellín y el área Metropolitana del Valle de Aburrá.
4. Definir e implementar algoritmos para la estimación de la concentración de material particulado (PM2.5 y PM10) en diferentes momentos y sectores del área metropolitana.
5. Evaluar los resultados de la identificación y estimación realizadas, mediante el uso de estimadores de error aplicados a los resultados de predicción obtenidos y los valores reales de medida de la calidad del aire.

3. Marco Teórico

En este capítulo se presentan los conceptos relacionados con la estimación de los niveles de material particulado en la atmósfera, comenzando por las implicaciones que un aire contaminado tiene sobre la salud de los seres humanos, como se mide el nivel de contaminación y que tipos de partículas contaminantes existen. Luego se presenta la solución propuesta en el presente proyecto. Luego se presentan aspectos básicos relacionados con la recolección de los datos y su debida limpieza. También, se presentan los conceptos relacionados con aprendizaje de máquina, comenzando por las principales ramas y los tipos de problema que pueden resolverse, además de esto se presentan conceptos relacionados con las técnicas de aprendizaje de máquina utilizadas para resolver el problema: Redes neuronales y bosques aleatorios. Por último, se presentan los conceptos relacionados con la medida del error y los diferentes estimadores que existen

3.1. Formulación del problema

Actualmente el ser humano lucha contra un enemigo invisible, causado por sí mismo, que es la contaminación ambiental del aire. La exposición a esta contaminación trae consigo múltiples afectados a la salud, como lo son los problemas cardiorrespiratorios y puede ser incluso causante de enfermedades crónicas/terminales, como el cáncer. La contaminación ambiental es la causante de la muerte de una gran cantidad de personas alrededor del mundo. Según la Organización Mundial de la Salud (OMS), se estima que la contaminación ambiental del aire, tanto en las ciudades como en las zonas rurales, es la causa de 4.2 millones de muertes prematuras en todo el mundo por año; esta mortalidad se debe a la exposición a partículas pequeñas de 2.5 micrones de diámetro, o menos (PM2.5), que causan enfermedades cardiovasculares y respiratorias. El material particulado es el causante del 3% de los problemas cardiopulmonares y el 5% de las muertes por cáncer de pulmón en el mundo **[1].**

Además de esto, la exposición permanente al material particulado puede ser causante de otras enfermedades como el hígado graso no alcohólico. El material particulado posee metales pesados e hidrocarburos que al quemarse, pueden generar daños en el ADN y en el sistema neurológico **[2].**

En nuestro caso, la ciudad de Medellín y su área metropolitana poseen una geografía especial que genera que los gases de efecto invernadero se queden atrapados en la atmósfera, por lo que lo hace un objeto de estudio interesante, un piloto a nivel de Colombia con respecto al control y monitoreo de la calidad del aire y una ciudad con la necesidad de aplicar medidas que faciliten la reducción de los niveles de contaminantes atmosféricos. Medellín y su área metropolitana, conforman una región con 4.055.296 de habitantes (de acuerdo con las proyecciones poblacionales del DANE provenientes del censo 2018) que actualmente se encuentra en

acelerado crecimiento, tanto económico, como de población **[3]**. Lo anterior ha traído para sí graves problemas. En los últimos años el parque automotor de la ciudad ha aumentado considerablemente, pasando de tener (registrados) 173.389 vehículos en el año 2007 a tener 305.361 vehículos en el año 2018. Un crecimiento que es equivalente al 76.11 % **[4]**.

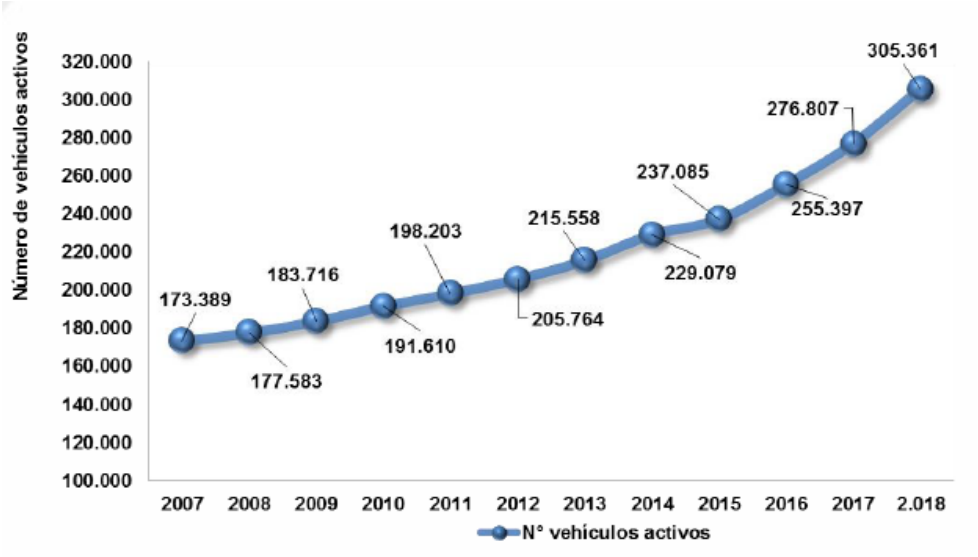


Figura 1. Crecimiento del parque automotor **[4]**.

El crecimiento del parque automotor trae consigo muchas desventajas para la calidad del aire ya que los vehículos son la principal fuente de emisión de material particulado (PM2.5 y PM10), según los indicadores ambientales de Medellín para el 2018 **[4]**.

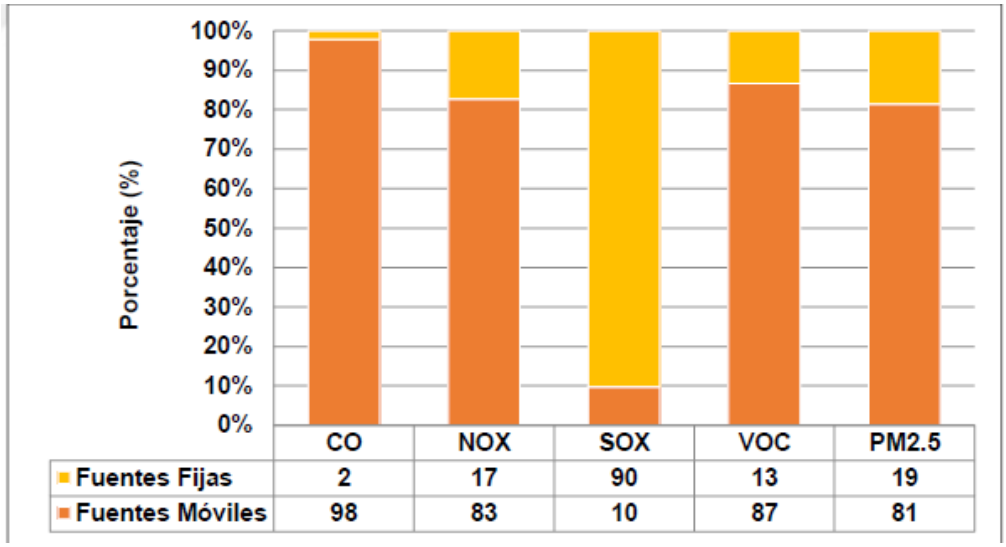


Figura 2. Porcentaje de material particulado generado por los distintos tipos de fuentes **[4]**.

En adición a esto, no solo el hecho de que existan vehículos en circulación es relevante en temas ambientales, sino que la congestión vial, que provoca que los vehículos estén deteniendo su marcha y retomando, genera un aumento en tres y cuatro veces de emisiones de contaminantes

al aire. Al existir un aumento en el parque automotor, la congestión vial aumenta y por lo tanto la contaminación.

A la atmósfera del Aburrá, desde distintas fuentes, móviles y fijas, la afectan 750 toneladas diarias, es decir, 273.750 tn/año, de agentes contaminantes entre monóxido de carbono, óxidos de nitrógeno y PM2.5. **[2]**

Según el DANE entre 1980 y 2012, en Medellín muere una persona cada tres horas por causas relacionadas con la contaminación del aire por enfermedades respiratorias crónicas, accidentes cerebrovasculares y cáncer de pulmón **[5].**

La población de Medellín es la más afectada del país por la contaminación medioambiental, ya que supera a Bogotá en un 92% y en un 87% si se compara con el país, en indicadores de mortalidad por enfermedades respiratorias crónicas **[2]**.

Basado en la preocupante situación del mundo y específicamente de Medellín y su área metropolitana, el presente proyecto busca alertar de manera temprana a las autoridades ambientales y población en general, cuando se estime que las concentraciones de material particulado en la atmósfera vayan a alcanzar niveles considerados como peligrosos (según el Índice de Calidad del Aire, ICA); con la finalidad de que puedan tomarse medidas preventivas y así evitar que la salud de la población se ponga en riesgo.

3.2. Solución propuesta

Para alertar a la población sobre los niveles de material particulado, es necesario poder realizar una estimación confiable sobre la concentración de contaminantes. Para esto se propone hacer uso de la información histórica de PM2.5, PM10, temperatura, humedad relativa, dirección y velocidad del viento. Esta información está presente en la base de datos del Sistema de Alerta Temprana de Medellín y el Valle de Aburrá (SIATA).

La información recolectada es utilizada para entrenar un algoritmo de aprendizaje de máquina con la capacidad de entender la tendencia de los datos históricos y arrojar una estimación de los niveles futuros de material particulado.

Con la finalidad de afinar los resultados obtenidos, es necesario que los datos pasen por un proceso conocido como *limpieza de la información.* La limpieza se encarga de descartar o transformar los datos que son considerados erróneos o que no aportan valor a la solución del problema.

Luego de entrenar el algoritmo, es necesario realizar un análisis para determinar la precisión de este. Este análisis puede realizarse en base a un conjunto de datos de prueba donde ya se

conocen las entradas y la salida esperada. Al comparar los resultados obtenidos con la salida esperada que ya se conoce, es posible determinar qué tan precisa es la estimación.

Luego de desarrollar un algoritmo que estime de manera precisa los niveles de material particulado presentes en la atmósfera para determinado momento y lugar es posible, basándonos en el ICA, alertar de manera temprana a las autoridades ambientales y la población en general.

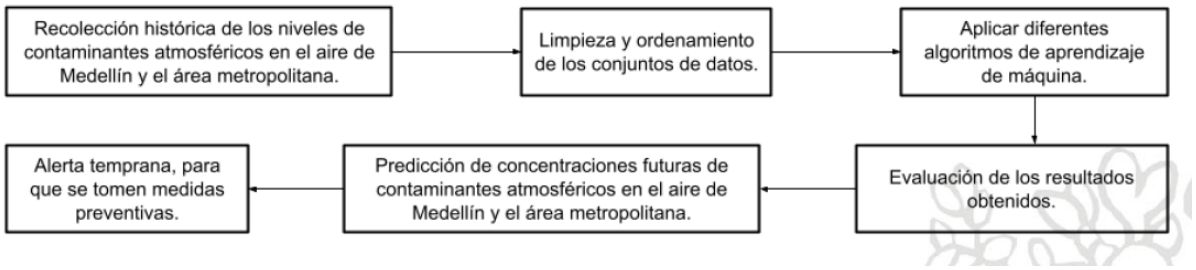


Figura 3. Diagrama de bloques de la solución propuesta.

3.3. Calidad del aire

Entender a qué hacemos referencia cuando hablamos de calidad del aire nos permite determinar cuando este es potencialmente peligroso para la salud del ser humano. Para esto es necesario responder las siguientes preguntas:

- ¿Qué es la contaminación del aire o polución?
- ¿Qué partículas son las causantes de la contaminación en el aire?
- ¿Qué tanta concentración de estas partículas es considerada peligrosa?

3.3.1. Polución

La polución es la contaminación del medio ambiente, en especial del aire o del agua, producida por los residuos procedentes de la actividad humana o de procesos industriales o biológicos". La polución es una alteración negativa de la calidad del aire producida por el ser humano. **[6]**

La contaminación del aire es causada por la concentración de diversas partículas en la atmósfera que pueden llegar a una persona y ser inhaladas por esta. Estas partículas son: PM2.5, PM10, O3, NO2 y SO2.

Debido a todas las implicaciones que trae consigo la contaminación del aire, mencionadas en la **sección 3.1**, la OMS ha trazado como objetivo la reducción gradual de los niveles de contaminantes atmosféricos en el aire, por lo que ha definido unas directrices sobre la calidad del aire (publicadas en el año 2005), asegurando así, que una reducción media anual de las concentraciones de partículas (PM10) de 35 microgramos/m3, común en muchas ciudades en

desarrollo, a 10 microgramos/m3, permitirá reducir el número de defunciones relacionadas con la contaminación aproximadamente un 15%. Las directrices se aplican a todo el mundo **[1]**.

Los valores fijados en dichas directrices son:

- PM2.5 - 10ug/m3 de media anual, 25ng/m3 de media en 24h.
- PM10 - 20ug/m3 de media anual, 50 ug/m3 de media en 24h.
- O3 - 100 ug/m3 de media en 8h.
- NO2 - 40 ug/m3 de media anual, 200 ug/m3 de media en 1h.
- SO2 - 20 ug/m3 media en 24h, 500 ug/m3 de media en 10min.

3.3.2. Material particulado

El material particulado (PM) es un conjunto de partículas sólidas y líquidas emitidas directamente al aire, tales como el hollín de diesel, polvo de vías, el polvo de la agricultura y las partículas resultantes de procesos productivos **[7]**.

Los niveles de material particulado (PM2.5 y PM10) son un indicador representativo común de la contaminación del aire. Afectan a más personas que cualquier otro contaminante. Los principales componentes del PM son los sulfatos, los nitratos, el amoníaco, el cloruro de sodio, el hollín, los polvos minerales y el agua. Consisten en una compleja mezcla de partículas sólidas y líquidas de sustancias orgánicas e inorgánicas suspendidas en el aire. Si bien las partículas con un diámetro de 10 micrones o menos (PM10) pueden penetrar y alejarse profundamente dentro de los pulmones, existen otras partículas más dañinas para la salud, que son aquellas con un diámetro de 2.5 micrones o menos (PM2.5). Las PM2.5 pueden atravesar la barrera pulmonar y entrar en el sistema sanguíneo. La exposición crónica a partículas contribuye al riesgo de desarrollar enfermedades cardiovasculares y respiratorias, así como cáncer de pulmón **[1].**

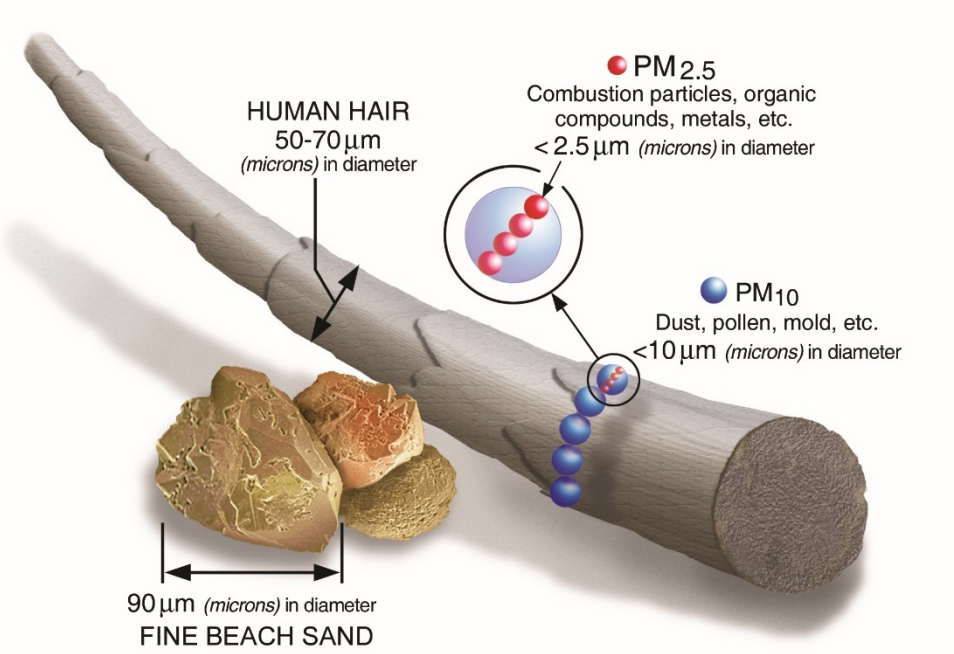


Figura 4. Diámetro de una partícula de PM2.5 y PM10 comparado con el diámetro de un cabello humano **[36]**.

En el presente proyecto se realizó la recolección de los niveles históricos de PM2.5 y PM10 con la finalidad de encontrar una tendencia y poder estimar así los niveles futuros de estos contaminantes.

3.3.3. Índice de Calidad del Aire

Con la finalidad de alertar a las autoridades ambientales y la población en general cuando se estime que la calidad del aire es potencialmente peligrosa para la salud, es necesario clasificar los niveles de concentración de contaminantes en grupos específicos. El ICA nos ayuda a definir estos intervalos.

El ICA es un indicador que nos da a entender gradualmente la cantidad de partículas contaminantes que hay presente en la atmósfera. Otra forma de verlo es que el ICA nos sirve como indicador para conocer la pureza del aire que respiramos.

En el caso del Valle de Aburrá, el ICA se mide de 0 a 300 y se clasifica en cinco escalas según su valor **[8]**.

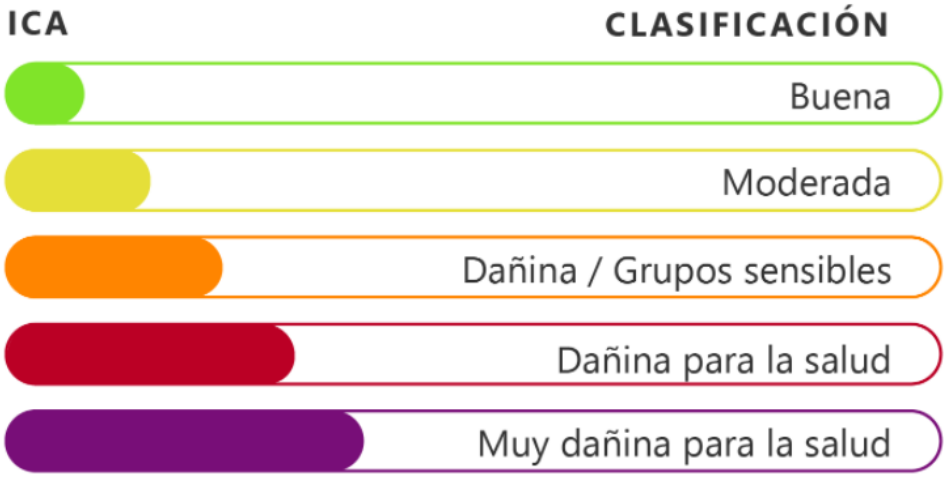


Figura 5. Clasificación del daño para la salud según el nivel del ICA **[8]**.

Indice de la Calidad del Aire			Puntos de corte del ICA ug/m^3						
ICA	Color	Categoría	PM_{10} 24 horas	$PM_{2,5}$ 24 horas	CO 8 horas	SO_2 1 hora	NO_2 1 hora	O_3 8 horas	O_3 1 hora (1)
0 - 50	Verde	Buena	0 - 54	0 - 12	0 - 5.094	0 - 93	0 - 100	0 - 106	.....
51 - 100	Amarillo	Aceptable	55 - 154	13 - 37	5.095 - 10.819	94 - 197	101 - 189	107 - 138	.....
101 - 150	Naranja	Dañina a la salud de Gupos Sensibles	155 - 254	38 - 55	10.820 - 14.254	198 - 486	190 - 677	139 - 167	245 - 323
151 - 200	Rojo	Dañina a la salud	255 - 354	56 - 150	14.255 - 17.688	487 - 797	678 - 1.221	168 - 207	324 - 401
201 - 300	Púrpura	Muy dañina a la salud	355 - 424	151 - 250	17.689 - 34.862	798 - 1.583	1.222 - 2.349	208 - 393	402 - 794
301 - 500	Marrón	Peligrosa	425 - 604	251 - 500	34.863 - 57.703	1.584 - 2.629	2.350 - 3.853	394 (2)	795 - 1.185

Figura 6. Valor del ICA según la cantidad de material particulado presente en la atmósfera **[4]**.

El nivel del ICA viene acompañado de varias recomendaciones, según la escala en la que se encuentre.

Figura 7. Recomendaciones para tener en cuenta según el valor del ICA **[8]**.

3.4. SIATA

El SIATA es la fuente principal de información del presente proyecto ya que es una plataforma de libre acceso, en la cual puede obtenerse información histórica de los niveles de material particulado y los valores medidos de las diferentes variables meteorológicas en Medellín y su área metropolitana.

El Sistema de Alerta Temprana del Valle de Aburrá es un proyecto estratégico de Medellín y su área metropolitana para la gestión ambiental y de riesgos, a través del conocimiento científico, el desarrollo tecnológico y la innovación. El SIATA tiene como objetivo identificar y alertar de forma temprana cuando van a ocurrir fenómenos de cualquier tipo (naturales o no) que puedan ser riesgosas para la población **[28]**.

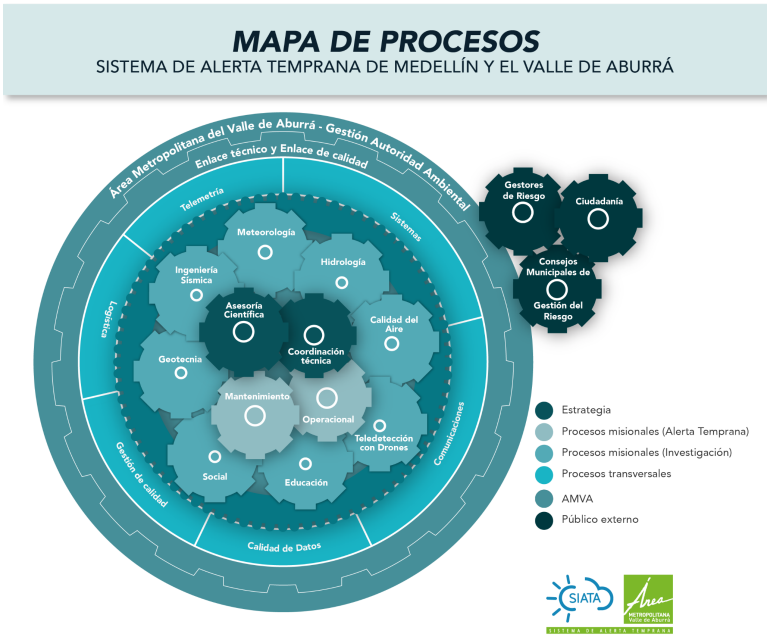


Figura 8. Mapa de procesos del SIATA **[28]**.

El SIATA posee una Red de Calidad del Aire del Valle de Aburrá que cuenta con estaciones para el monitoreo de distintos contaminantes, como ozono O3, óxidos de nitrógeno, monóxido de carbono CO, material particulado PM10, PM2.5, entre otros **[29]**.

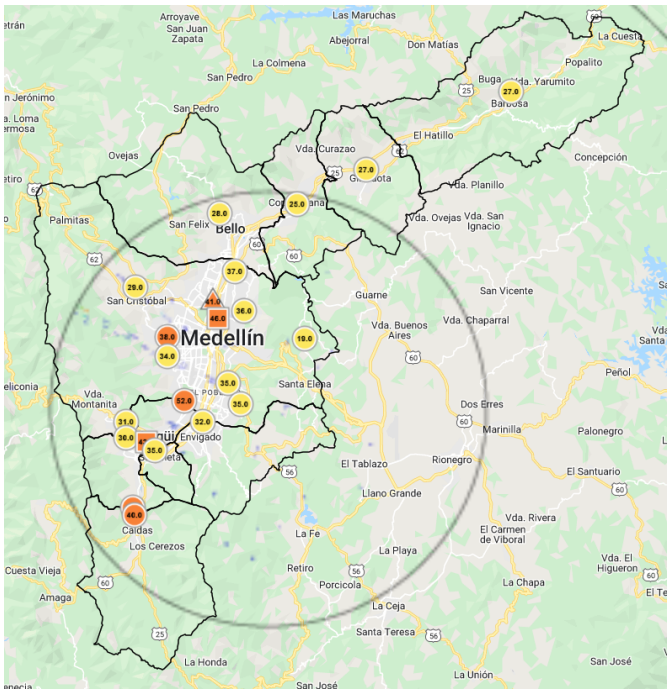


Figura 9. Distribución de las estaciones de medición de calidad del aire en el área metropolitana **[30]**.

Las estaciones de la red de calidad del aire del SIATA son:

Código	**Nombre**
12	Estación Trafico Centro
25	Universidad Nacional, sede El Volador
28	Casa de Justicia Itagüí
31	Corporación Universitaria Lasallista
38	I.E. Concejo Municipal de Itagüí
44	El Poblado - Tanques La Ye EPM
48	Estación Tráfico Sur
69	E U Joaquín Aristizábal
78	La Estrella - Hospital

79	I.E. Pedro Octavio Amado
80	Planta de producción de agua potable EPM
81	Torre Social
82	Ciudadela Educativa La Vida
83	I.E Pedro Justo Berrio
84	I.E INEM sede Santa Catalina
85	Parque Biblioteca Fernando Botero
86	I.E Ciro Mendía
87	I.E. Fernando Vélez
88	E.S.E. Santa Gertrudis
90	I.E. Rafael J. Mejía
94	Santa Elena

Tabla 1. Estaciones de la red de calidad del aire del SIATA.

Además de esto, el SIATA cuenta con estaciones de medición de variables meteorológicas que nos dan información en tiempo real de los valores de temperatura, humedad relativa, presión atmosférica, velocidad y dirección de los vientos.

Las estaciones de medición de las variables meteorológicas del SIATA son:

Código	Nombre
59	ISAGEN
68	Jardín Botánico
73	Ciudadela Educativa La Vida
82	I.E Manuel José Caicedo
83	Centro de Salud San Javier La Loma
105	Parque 3 Aguas
122	Tasajera
197	Universidad de Medellín
198	Politécnico Jaime Isaza Cadavid
201	Torre SIATA
202	AMVA
203	UNAL - Sede Agronomía
206	Colegio Concejo de Itagüí
207	Vivero EPM Piedras Blancas
229	Alcaldía La Estrella
249	Escuela CEDEPRO
252	Alcaldía Envigado
269	Parque de las Aguas

271	Jorge Eliecer Gaitán
313	La Ye
318	Institución Rafael J. Mejía
345	Villa del Socorro
349	Institución Educativa La Candelaria
354	Villa Niza
355	Escuela Piedras Gordas
360	Miraflores
362	Proyecto Ámsterdam
367	Joaquín Vallejo
368	Federico Carrasquilla
397	Concejo de Itagüí
399	Institución Educativa Jesús María Valle
403	Puerto Triunfo
419	SENA Medellín
427	ITM Castilla
448	CASD
450	Universidad Lasallista
477	Tanque EPM Girardota

478	Fiscalía General de la Nación
542	Santa Elena

Tabla 2. Estaciones de la red meteorológica del SIATA.

3.5. Recolección de los niveles históricos de PM y variables meteorológicas

Además de contar con una extensa red de sensores para el monitoreo constante de la calidad del aire, el SIATA, también posee una base de datos de acceso al público donde almacena la información de los niveles históricos de material particulado en los últimos años, tales como: PM2.5, PM10, NO, NO2, NOx, Ozono, CO, SO2 y de los valores históricos de las variables meteorológicas, tales como: Humedad, precipitación, presión, temperatura, viento y radiación. Haciendo uso de esta base de datos, es posible obtener los valores de los últimos 10 años medidos en diferentes sectores de Medellín y su área metropolitana.

En el caso del presente proyecto de investigación se descargaron los datos históricos de PM2.5, PM10, humedad, presión atmosférica, temperatura, velocidad y dirección del viento; comprendidos entre el periodo del 25 de octubre del 2011 al 25 de octubre del 2021.

3.6. Limpieza de datos

Cuando se trabaja con bases de datos de gran tamaño (que contienen miles y miles de datos), es normal que se presenten anomalías y errores en estos datos. La existencia de estos errores puede causar que los resultados obtenidos al analizar y tratar estos datos no sean correctos. Cuando los datos presentan errores el proceso se hace menos confiable.

Los errores en los datos pueden ser resultado de errores de medición, falta de validación al ingresarlos, omisiones mientras se coleccionan o mantienen, malas interpretaciones o cambios en el universo de los datos que no se han reflejado en la forma de representarlos. **[9]**.

Los errores en los datos pueden ser sintácticos, semánticos o de contexto. Los errores sintácticos hacen referencia a errores léxicos, de formato o de no estandarización de la información. Los errores semánticos hacen referencia a violaciones en la integridad de la información o contradicciones y los errores de contexto hacen referencia a la ausencia de valores **[9]**.

Cuando hay errores en los datos y estos se detectan, es necesario un proceso de limpieza que va desde la eliminación de los datos erróneos hasta su transformación. En el caso de datos numéricos, los valores erróneos o faltantes pueden ser reemplazados por la media aritmética de los datos correctos.

En el caso del presente proyecto en las bases de datos descargadas no se detectaron errores sintácticos o de contexto, ya que toda la información se encontraba bien estandarizada y no había información faltante. Pero, se detectaron algunos errores semánticos, ya que existía información que no era confiable, la confiabilidad en la información está determinada por el valor de unas banderas.

3.7. Inteligencia artificial

Haciendo uso de la información histórica recolectada, clasificada y posteriormente limpiada; en el presente proyecto se buscaba realizar una estimación de los niveles de concentraciones futuras de material particulado en la atmósfera de Medellín y su área metropolitana. A nivel computacional esto es posible haciendo uso de algoritmos de Inteligencia Artificial, específicamente Aprendizaje de máquina.

3.7.1. Aprendizaje de máquina

En el campo de las tecnologías de la información, el aprendizaje de máquina es conocido como el proceso de desarrollar algoritmos con la capacidad de simular el proceso por el cual un ser humano aprende.

Arthur Samuel lo definió como: “Un campo de estudio que da a las computadoras la capacidad de aprender sin ser programadas explícitamente” **[10]**, Ethern Alpaydin en su libro define el aprendizaje de máquina como: “Programación de computadoras para optimizar un criterio de desempeño utilizando datos de ejemplo o experiencia pasada” **[11]**. Tom Mitchell lo definió como: “Se dice que un programa de computadora aprende de la experiencia (E) con respecto a alguna clase de tareas (T) y la medida de desempeño (P), si su desempeño en las tareas en T, medido por P, mejora con la experiencia E” **[12]**

3.7.1.1. Ramas del aprendizaje de máquina

Dentro del aprendizaje de máquina existen dos ramas principales que son: Aprendizaje supervisado y Aprendizaje no supervisado. Las diferencias entre estos se explican a continuación.

3.7.1.1.1. Aprendizaje supervisado

La principal característica del aprendizaje supervisado es que al algoritmo se le provee un conjunto de datos de entrada **I** y a la vez la salida esperada **O** para ese conjunto de datos**.** Haciendo uso de esta información es posible calcular, luego de cada iteración del algoritmo, que tan preciso es. Haciendo uso de esta información, el algoritmo puede modificar sus parámetros internos con la finalidad de reducir la magnitud de la función de error global **[13].**

Si el algoritmo es lo suficientemente flexible, se espera que la precisión aumente y la diferencia entre la salida esperada y la salida obtenida, cada vez se acerque más a cero.

Cuando se entrena un modelo con aprendizaje supervisado, es muy importante que este no se aprenda los datos de entrada y los datos de la salida esperada de memoria, sino que sea capaz de generalizar y entregar valores de salida coherentes para datos que nunca haya visto. Cuando un algoritmo pierde la capacidad de generalizar, se dice que está **sobre ajustado.**

3.7.1.1.2. Aprendizaje no supervisado

En el caso del aprendizaje no supervisado, al algoritmo solo se le provee un conjunto de datos de entrada **I** y no existe una salida esperada **O**, por lo tanto, tampoco existe una medida de error, debido a que no hay una salida esperada con la cual comparar los resultados. El aprendizaje no supervisado es muy útil cuando es necesario aprender como un grupo de datos puede agruparse dependiendo de las similitudes existentes entre estos **[13].**

Un ejemplo de un algoritmo de aprendizaje no supervisado puede ser la clasificación de un grupo de zapatos según su color: Si un ser humano **H** es seleccionado para realizar esta tarea, aunque no se le haya entregado unos valores de salida esperada, **H** basándose simplemente en la observación es capaz de realizar esta tarea satisfactoriamente. Lo anterior es lo mismo que sucede en los algoritmos de aprendizaje no supervisado.

En el caso de la tarea de estimar las concentraciones futuras de PM2.5 y PM10, el aprendizaje requerido es del tipo supervisado debido a que se disponen de los valores de entrada y de salida esperada y se busca que el sistema entienda el comportamiento de los valores de salida con respecto a los valores de entrada dados.

3.7.1.2. Tipos de problemas a resolver con aprendizaje de máquina

Dentro del aprendizaje de máquina existen dos tipos de problemas que se pueden resolver: problemas de regresión y problemas de clasificación. La clasificación dentro de estos dos grupos depende de la naturaleza del problema.

Las diferencias entre los problemas de regresión y los problemas de clasificación se explican a continuación.

3.7.1.2.1. Regresión

Los problemas de regresión consisten en que, a partir de un conjunto de datos dados (entradas y salida esperada), se crea un algoritmo que tenga la capacidad de entender como estos datos van evolucionando (tendencia) y en base a ese entendimiento, ser capaz de inferir o predecir cuál es el valor siguiente (en base a esa tendencia) **[13]**.

En la siguiente imagen observamos como el comportamiento de los datos (puntos azules) pueden ser modelados a través de una línea recta (línea roja). En base a esa línea es posible calcular, por ejemplo, cuál será el valor aproximado de **Y** que habrá en **X** = 30.

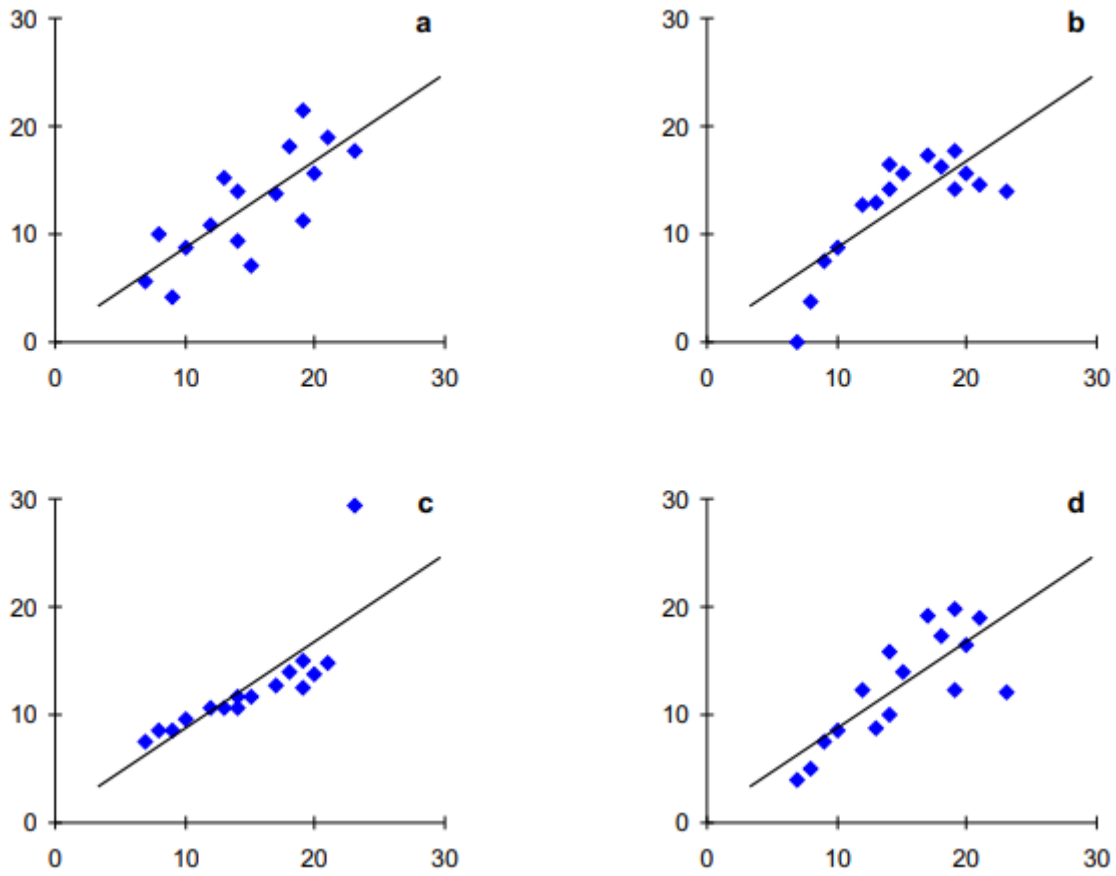


Figura 10. Ejemplos gráficos de regresión lineal **[14]**.

3.7.1.2.2. Clasificación

Los problemas de clasificación consisten, en que, a partir de un conjunto de datos dado (puede o no existir una salida esperada), se extraen las características de estos datos y en base a eso agruparlos según sus similitudes **[13]**.

En las siguientes imágenes, es posible detectar diferentes grupos de datos según las similitudes en sus características. En este caso se clasifican según su color o forma **[13]**.

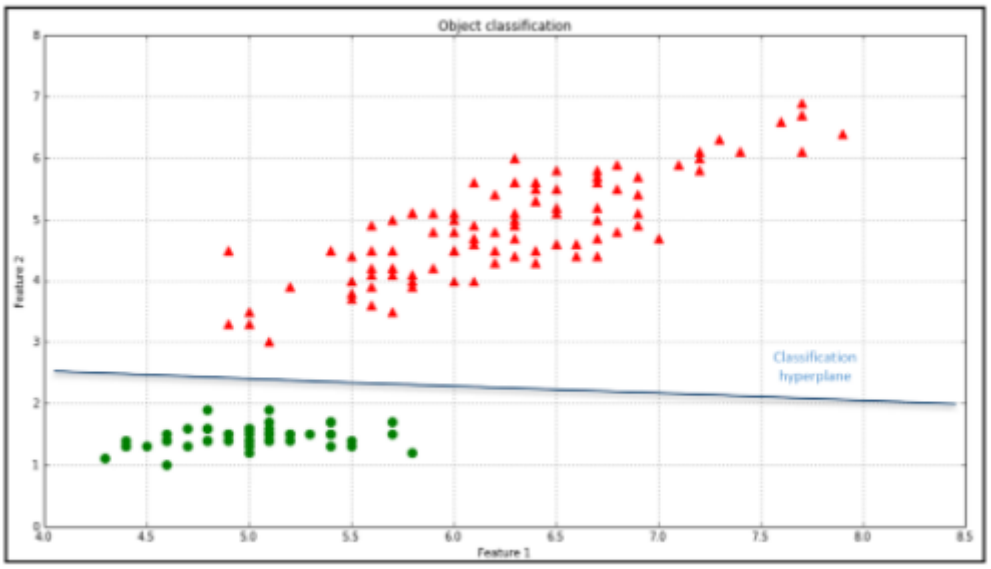


Figura 11. Representación de datos con características similares **[13].**

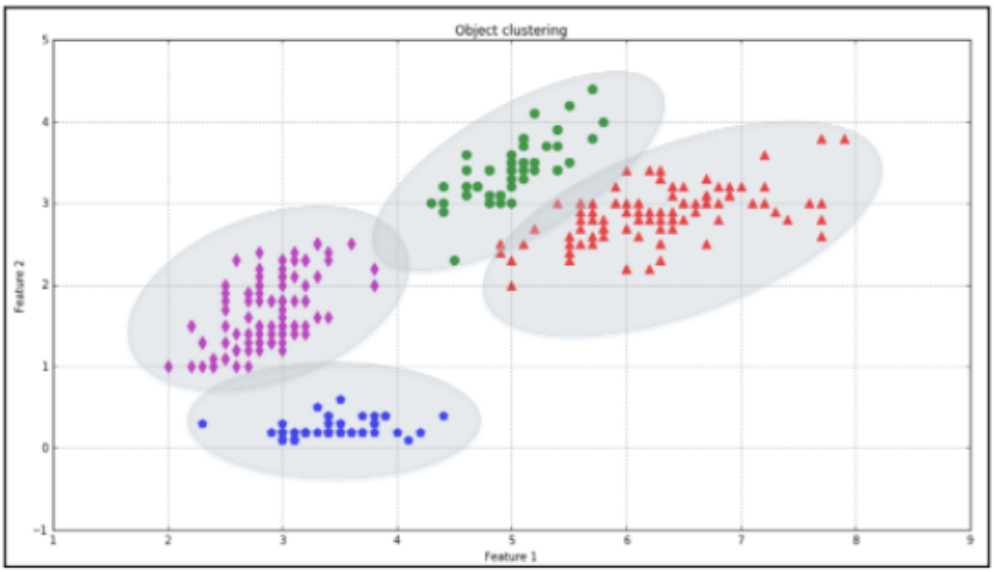


Figura 12. Clasificación de grupos de datos según la similitud en sus características **[13].**

En el caso del presente proyecto, el problema tratado es un problema de regresión, debido a que a partir del comportamiento de los valores de salida con respecto a unos valores de entrada dados, se espera obtener una salida numérica que siga ese comportamiento y que sea coherente con las entradas dadas.

3.7.2. Redes neuronales artificiales

En el presente proyecto se optó por entrenar un modelo de red neuronal, debido a que es un método que permite solucionar problemas de alta complejidad, donde las relaciones entre las variables de entrada con las variables de salida no son detectadas fácilmente. Además, en los

últimos años, las redes neuronales, se han convertido en el método más popular para solucionar problemas de aprendizaje de máquina.

Las redes neuronales artificiales son un modelo computacional cuyo propósito es simular el comportamiento del cerebro humano con respecto a cómo este aprende y cómo generaliza este conocimiento. Para poder comprender mejor cómo funciona una red neuronal artificial, es necesario entender sus conceptos fundamentales, que son: La neurona, la capa y la red.

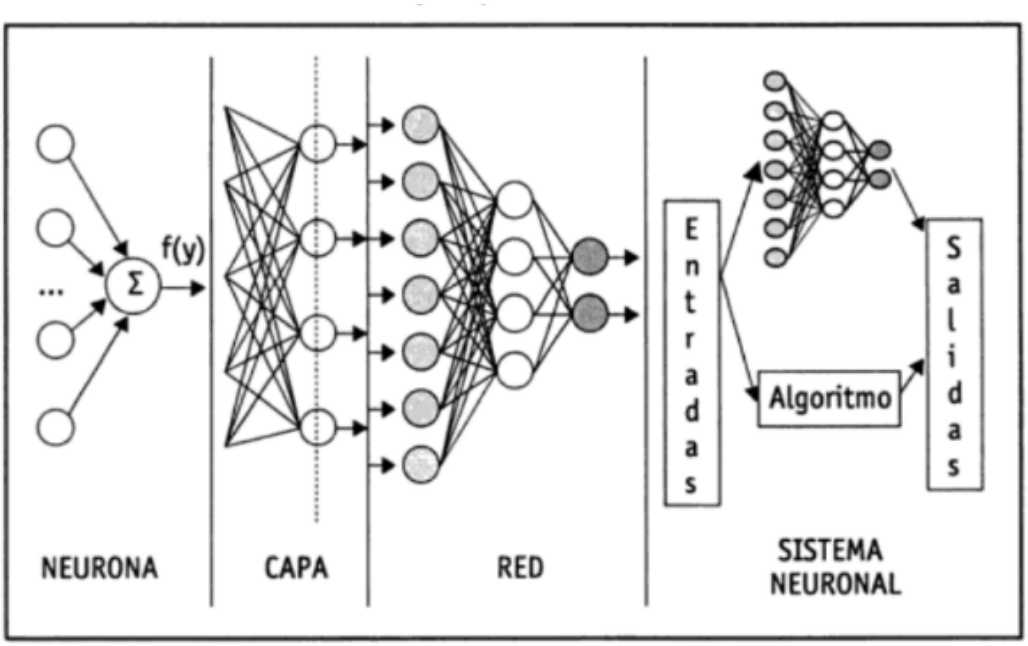


Figura 13. Conceptos principales de una red neuronal artificial **[15]**.

3.7.2.1. La neurona

Es la unidad básica de procesamiento dentro de una red neuronal. Recibe varios valores de entrada (x_1, x_2, x_3, ...x_n) y entrega una salida (y).

Como se observa en la siguiente imagen, cada una de las entradas tiene un valor de w asignado, este se conoce como el peso e indica la fortaleza de cada conexión. La neurona se encarga de realizar una suma ponderada de las entradas. La salida de esta suma es pasada por una función que se conoce como función de activación.

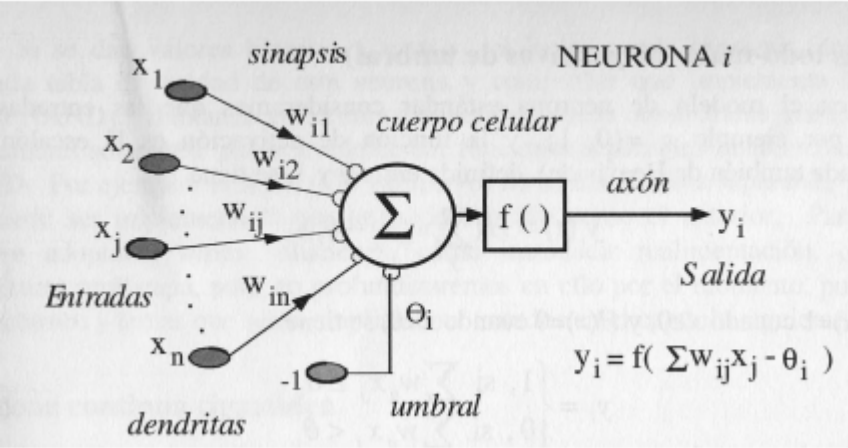


Figura 14. Esquema de una neurona **[16]**.

$$\boldsymbol{y = f\left(\sum_{i=1}^{n} x_i \cdot w_i\right)}$$

Ecuación 1. Salida de la neurona.

Donde $f(x)$ es la función de activación.

En el presente proyecto se optó por utilizar para todas las capas una función de activación sigmoide, debido a que los datos de salida de la red se encuentran en una escala de 0 a 1.

La función sigmoide está representada por la siguiente expresión:

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

Ecuación 2. Función sigmoide.

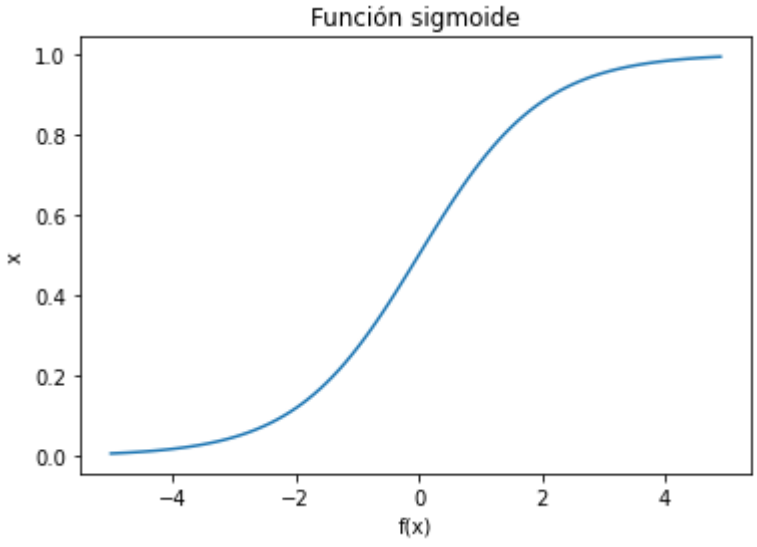


Figura 15. Representación gráfica de la función sigmoide.

3.7.2.2. La capa

Dentro de una red neuronal, se conoce como capa a un grupo de neuronas que están ubicadas de forma vertical y que reciben la misma información de entrada.

3.7.2.3. La red

La red neuronal consiste en ubicar de forma secuencial cada una de las capas. La información de entrada de cada una de las capas es la información de salida de la capa inmediatamente anterior.

Para el presente proyecto se decidió representar el problema de la estimación de los niveles futuros de PM2.5 y PM10 como una red neuronal con una capa de entrada de 9 neuronas, cuatro capas ocultas con 128, 256, 256 y 128 neuronas respectivamente y una capa de salida con 1 neurona.

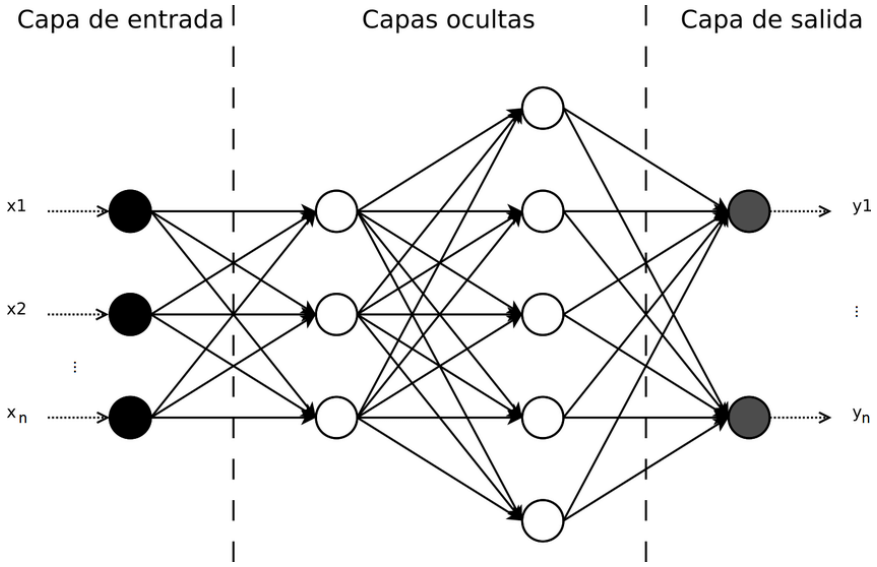


Figura 16. Esquema de la unión de capas de neuronas para formar una red **[17]**.

3.7.3. Bosques aleatorios

En el presente proyecto se decidió hacer uso de esta técnica de aprendizaje de máquina, con la finalidad de realizar la estimación de material particulado, debido a que es la técnica de aprendizaje de máquina basado en árboles de decisión más popular y flexible.

Los árboles de decisión son una estructura, que como su nombre indica, están basadas en la topología de tipo árbol. Un árbol de decisión es una estructura que sigue un proceso secuencial que inicia desde la raíz, donde se evalúa una característica particular. Basado en esta evaluación se elige una rama (se toma una decisión). El proceso se repite hasta que se llega a una *hoja* final que representa el objetivo de lo que se busca resolver **[13]**.

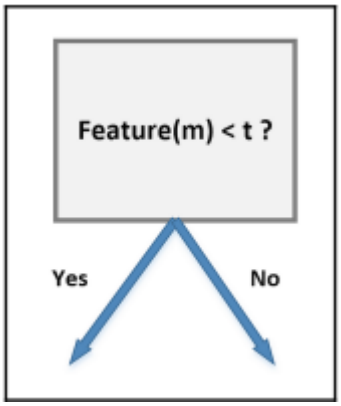


Figura 17. Ejemplo de decisión en base al valor de una característica **[13]**.

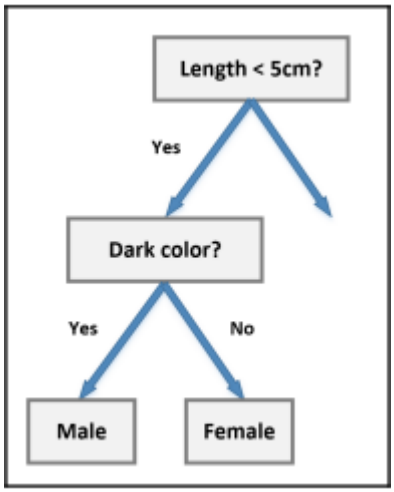


Figura 18. Ejemplo de un árbol de decisión básico para definir el género de una persona según el color y largo de su cabello **[13]**.

Cuando unimos varios árboles de decisión con sus propias reglas aplicadas a pequeños conjuntos de variables, tenemos un **bosque aleatorio**. El resultado del bosque aleatorio se obtiene promediando los resultados de los árboles de decisión que lo conforman **[13]**.

3.8. Python

La implementación de la solución del presente proyecto se realizó en Python. A continuación, se presenta una definición de este lenguaje y los motivos por los que fue seleccionado:

Según Arturo Fernández Montoro en su libro *Python 3 al descubierto*: "...Python es un lenguaje de programación de alto nivel, interpretado y multipropósito. En los últimos años su uso ha ido constantemente creciendo y en la actualidad es uno de los lenguajes de programación más empleados para el desarrollo de software." **[18]**.

Además de esto, Python es un lenguaje de programación con gran utilidad para realizar operaciones entre vectores y matrices, algo que es fundamental para el desarrollo de algoritmos

de aprendizaje de máquina. En los últimos años, este sector de la industria ha escogido a Python como su lenguaje por excelencia.

Existen un conjunto de librerías de Python que permiten que el trabajo con algoritmos de aprendizaje de máquina se haga más fácil, como: Pandas, numpy, matplotlib, scikit-learn, tensorflow.

3.8.1. Pandas [19]

Pandas es una herramienta de análisis y manipulación de datos de código abierto rápida, potente, flexible y fácil de usar, construida sobre el lenguaje de programación Python.

A continuación, se describen los métodos de pandas más usados para el desarrollo del presente proyecto:

- **read_csv**: Lee un archivo de valores separados por comas (csv) en un conjunto de datos.
- **to_csv**: Escribe un conjunto de datos en un archivo de valores separados por comas (csv).
- **concat**: Concatena conjuntos de datos de pandas a lo largo de un eje particular.
- **merge**: Combina conjuntos de datos al estilo de una base de datos.
- **drop_duplicates**: Devuelve un conjunto de datos con las filas duplicadas eliminadas.
- **set_index**: Establece el índice de un conjunto de datos usando las columnas existentes.
- **head**: Devuelve las primeras **n** filas de un conjunto de datos, por defecto n = 5.
- **tail**: Devuelve las últimas **n** filas de un conjunto de datos, por defecto n = 5.
- **describe**: Genera estadísticas descriptivas de las columnas numéricas de un conjunto de datos.

3.8.2. NumPy [20]

NumPy trae el poder computacional de lenguajes como C y Fortran a Python, un lenguaje mucho más fácil de aprender y usar. Con este poder viene la simplicidad: una solución en NumPy suele ser clara y elegante.

A continuación, se describen los métodos de numpy más usados para el desarrollo del presente proyecto:

- **mean**: Devuelve el promedio de los elementos un arreglo o matriz dados.

- **sqrt**: Devuelve la raíz cuadrada no negativa de un arreglo o matriz, por elementos.
- **square**: Devuelve el cuadrado de los elementos de un arreglo o matriz dados.
- **abs**: Devuelve el valor absoluto de los elementos de un arreglo o matriz dados.

3.8.3. Matplotlib [21]

Matplotlib es una biblioteca completa para crear visualizaciones estáticas, animadas e interactivas en Python.

A continuación, se describen los métodos de matplotlib más usados para el desarrollo del presente proyecto:

- **plot:** Gráfica *y vs x* como líneas y/o marcadores.
- **scatter:** Genera un gráfico de dispersión de *y vs x* con diferentes tamaños y/o colores de marcador.
- **matshow:** Gráfica una matriz en una nueva ventana de figura.

3.8.4. Scikit Learn [22]

Scikit-learn es una biblioteca de aprendizaje automático de código abierto que admite el aprendizaje supervisado y no supervisado. También proporciona varias herramientas para el ajuste de modelos, preprocesamiento de datos, selección de modelos, evaluación de modelos y muchas otras utilidades.

A continuación, se describen los métodos de scikit-learn más usados para el desarrollo del presente proyecto:

- **MinMaxScaler:** Este estimador escala y traduce cada característica individualmente de modo que esté en el rango dado en el conjunto de entrenamiento, por ejemplo, entre cero y uno.
- **make_column_transformer:** Permite aplicar un transformador a un conjunto de columnas.
- **train_test_split:** Divide arreglos o matrices en subconjuntos aleatorios de entrenamiento y prueba.
- **RandomForestRegressor:** Crea un regresor de bosque aleatorio.

3.8.5. TensorFlow [23]

TensorFlow es una plataforma de código abierto de extremo a extremo para el aprendizaje automático. Cuenta con un ecosistema integral y flexible de herramientas, bibliotecas y recursos de la comunidad que permite que los investigadores innoven con el aprendizaje automático (AA) y los desarrolladores creen e implementen aplicaciones con tecnología de AA fácilmente.

A continuación, se describen los métodos de scikit-learn más usados para el desarrollo del presente proyecto:

- **Sequential:** Agrupa una pila lineal de capas en un modelo.
- **Dense:** Genera una capa de red neuronal densamente conectada (todas las neuronas están conectadas a todas las variables de entrada).
- **InputLayer:** Capa que se utilizará como punto de entrada a una red.
- **Adam:** Optimizador que implementa el algoritmo de Adam.

3.9. Estimadores de error

Cuando se trabaja con aprendizaje de máquina supervisado, es necesario definir una manera de conocer la diferencia global existente entre las salidas obtenidas y las salidas esperadas de cada uno de los procesos. Dentro del estudio estadístico existen diferentes tipos de estimadores de error que pueden ser de utilidad para conocer esa diferencia global.

3.9.1. Error cuadrático medio (MSE)

El error cuadrático medio de un conjunto de datos mide el promedio de los errores al cuadrado, así:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}i - Yi)^2$$

Ecuación 3.

n: número de datos.

$\hat{Y}i$: Es el valor de salida obtenido.

Yi: Es el valor de salida esperado.

3.9.2. Raíz del error cuadrático medio (RMSE)

La raíz del error cuadrático medio de un conjunto de datos mide la raíz cuadrada del promedio de los errores al cuadrado, así:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{Y}i - Yi)^2}$$

Ecuación 4.

n: número de datos.

$\hat{Y}i$: Es el valor de salida obtenido.

Yi: Es el valor de salida esperado.

3.9.3. Error absoluto medio (MAE)

El error absoluto medio sirve para conocer la precisión de la estimación al comparar los valores de salida esperada con los valores de salida obtenida, así:

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|\hat{Y}i - Yi\right|$$

Ecuación 5.

n: número de datos.

$\hat{Y}i$: Es el valor de salida obtenido.

Yi: Es el valor de salida esperado.

3.9.4. Tasa de error (ER)

La tasa de error es útil para saber en un clasificador el porcentaje de datos mal clasificados, así:

$$ER = \left(\frac{Cantidad\ de\ datos\ mal\ clasificados}{Cantidad\ de\ datos}\right) \cdot 100\%$$

Ecuación 6.

En el trabajo desarrollado en el presente proyecto se hizo uso del MSE, RMSE y MAE; con la finalidad de analizar la precisión de la estimación y comparar el rendimiento de los diferentes algoritmos utilizados.

3.10. Estado del arte

Existen diversas investigaciones realizadas en torno a la estimación de los niveles de material particulado, haciendo uso de redes neuronales y otras técnicas de aprendizaje de máquina. Por ejemplo, en consecuencia, con el aumento de la contaminación (concentración de material particulado PM2.5) en las grandes ciudades del noreste chino, en el experimento presentado en **[24]** se busca realizar la predicción de concentración de PM2.5 con dos días de anticipación, por medio de técnicas especiales, como lo son: (1) Un modelo geográfico basado en la trayectoria de masa de aire, y (2) la transformada ondícula de las series de tiempo, y evaluar la eficiencia del uso de estas técnicas comparando con los resultados obtenidos omitiendo estas técnicas. Las variables recolectadas corresponden a la concentración de PM2.5 histórica, temperatura máxima por hora, temperatura mínima por hora, humedad, dirección y velocidad del viento, día del año, día de la semana y condiciones generales, estos valores corresponden al periodo comprendido entre el 1 de septiembre de 2013 y el 31 de octubre de 2014. El modelo geográfico basado en la trayectoria de masa de aire permite a los autores identificar regiones de influencia en la que se transporta aire "limpio" o aire "sucio" al interior de Beijing, mientras que, la transformada ondícula de las series de tiempo permite dividir la serie de información original de concentración de PM2.5, que tiene una muy alta variabilidad, en varias subseries con una variabilidad menor. Las variables recolectadas se usan como entradas de un perceptrón multicapa, que posee una función de activación sigmoide en la capa oculta. Los resultados del experimento para el modelo entrenado haciendo uso de las técnicas mencionadas anteriormente, muestran una reducción significativa en el RMSE (error cuadrático medio), MAE (error absoluto medio) e IA (índice de coincidencia) con respecto al modelo que no hace uso de estas técnicas.

Debido a la creciente importancia del estudio de la contaminación y sus efectos sobre la salud humana, como campo de investigación, en **[25]** se propone la predicción de concentración de material particulado PM2.5 como una estrategia para brindar una alerta temprana a la población. Se presenta un modelo neuro-difuso como una solución eficiente al problema de la predicción. Haciendo uso de técnicas y algoritmos, con la finalidad de aumentar la precisión de la predicción, como lo son: (1) analizar la correlación entre las diferentes variables de entrada con la finalidad de establecer los factores que más impacto tienen sobre la contaminación, (2) reducción de la dimensionalidad del conjunto de datos original por medio del análisis de componentes principales, (3) clasificación en grupos de los datos obtenidos en el numeral anterior, según la coincidencia entre estos, (4) obtener las reglas difusas de cada grupo, (5) optimización de la entrada y salida de las reglas difusas por medio de un algoritmo híbrido, producto de la combinación de algoritmos genéticos, optimización por enjambres de partículas y descenso del gradiente. Los resultados obtenidos con este modelo son comparados con modelos clásicos dentro del campo del aprendizaje de maquina y se observa una reducción

significativa en los estimadores de error como: MAE (error absoluto medio), RMSE (error cuadrático medio) y R2 (coeficiente de determinación ajustado).

China es uno de los países más afectados por la contaminación ambiental. Con la finalidad de encontrar un modelo eficiente para la predicción de la concentración de material particulado PM2.5, en **[26]** se busca evaluar el desempeño de un modelo basado en reducción de la dimensionalidad. Los datos recolectados de las estaciones de monitoreo del aire corresponden al periodo comprendido entre el 28 de marzo de 2013 al 31 de marzo de 2017. Con la finalidad de reducir la variabilidad de la información y eliminar información redundante, los datos son sometidos a un algoritmo conocido como Locally Linear Embedding. Estos datos son utilizados para entrenar una red neuronal profunda. Al medir los estimadores de error y comparar con los modelos clásicos aplicados sin manipular los datos, se observa una mejora en el desempeño.

Teniendo en cuenta que la oportuna predicción de niveles críticos de contaminación en el aire brinda la posibilidad de alertar de forma temprana a los ciudadanos del estado de calidad del aire, en **[27]** se busca evaluar el desempeño de dos redes neuronales recurrentes; una red neuronal recurrente de memoria a largo y corto plazo (LSTM), y una red neuronal basada en unidades de compuertas recurrentes (GRU); en la predicción de concentración de PM2.5. Gracias a once estaciones de monitoreo ubicadas en Santiago de Chile, se recogen datos meteorológicos y de concentración de PM2.5 en el periodo del 1 de junio hasta el 19 de agosto del 2011. Las redes neuronales son entrenadas con los datos conjuntos de todas las estaciones y con los datos de cada estación por separado. Los resultados obtenidos muestran que la precisión de la predicción de concentración de PM2.5 de las redes entrenadas con los datos conjuntos es mayor que la de las estaciones por separado y que la precisión de la red GRU es ligeramente mayor que la de la red LSTM.

4. Metodología

En este capítulo se presentan los pasos seguidos para realizar la estimación de los niveles futuros de material particulado presentes en la atmósfera para determinado momento y lugar de Medellín y su área metropolitana. Inicialmente se describe el proceso de recolección de la información a través de la plataforma del SIATA y cómo esta información fue presentada. A continuación, se describe el proceso de limpieza y unión que se realizó sobre los datos recolectados anteriormente. Luego, se presenta el análisis realizado para determinar la relación entre las diferentes variables meteorológicas y los niveles de material particulado presentes en la atmósfera. Por último, se presenta la implementación de los algoritmos de red neuronal y bosque aleatorio para realizar la estimación de los niveles futuros de material particulado presentes en la atmósfera.

4.1. Recolección de la información

4.1.1. Búsqueda de la información

Para obtener la información necesaria para dar solución al problema planteado, se hizo uso de la página web del Sistema de Alerta Temprana del Valle de Aburrá (SIATA). La página web del SIATA dispone de una sección llamada “Descargar información”, en donde cualquier persona puede acceder a las bases de datos públicas.

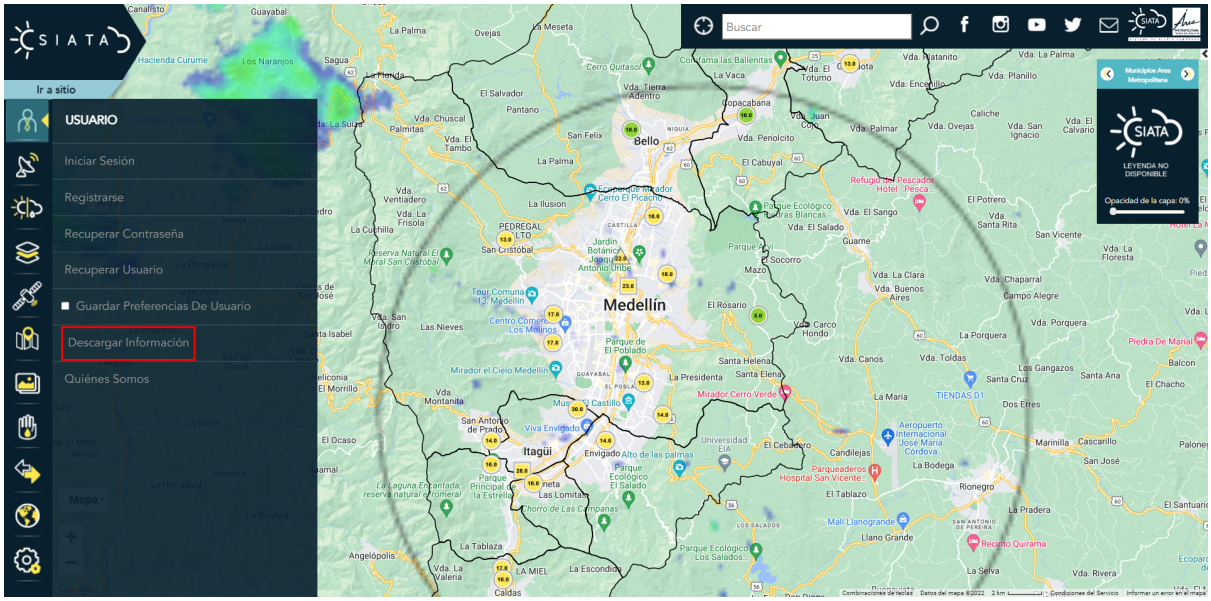


Figura 19. Sección del SIATA que nos permite acceder a la información histórica **[30]**.

4.1.2. Presentación de la información

La información histórica de calidad del aire de Medellín y su área metropolitana utilizada en el presente proyecto se encuentra almacenada en un lugar diferente a la información histórica de

mediciones de las variables meteorológicas. Además de esto, las estaciones de monitoreo de calidad del aire no son las mismas que las estaciones de medición de las variables meteorológicas.

Tanto la información histórica de calidad del aire, como la información histórica de mediciones de las variables meteorológicas están separadas por estación. Para el caso de la calidad del aire, hay 20 estaciones de monitoreo, mientras que para el caso de las variables meteorológicas hay 40 estaciones de medición y la información también está separada según la variable. Además de esto, la información se encuentra dividida en periodos mensuales.

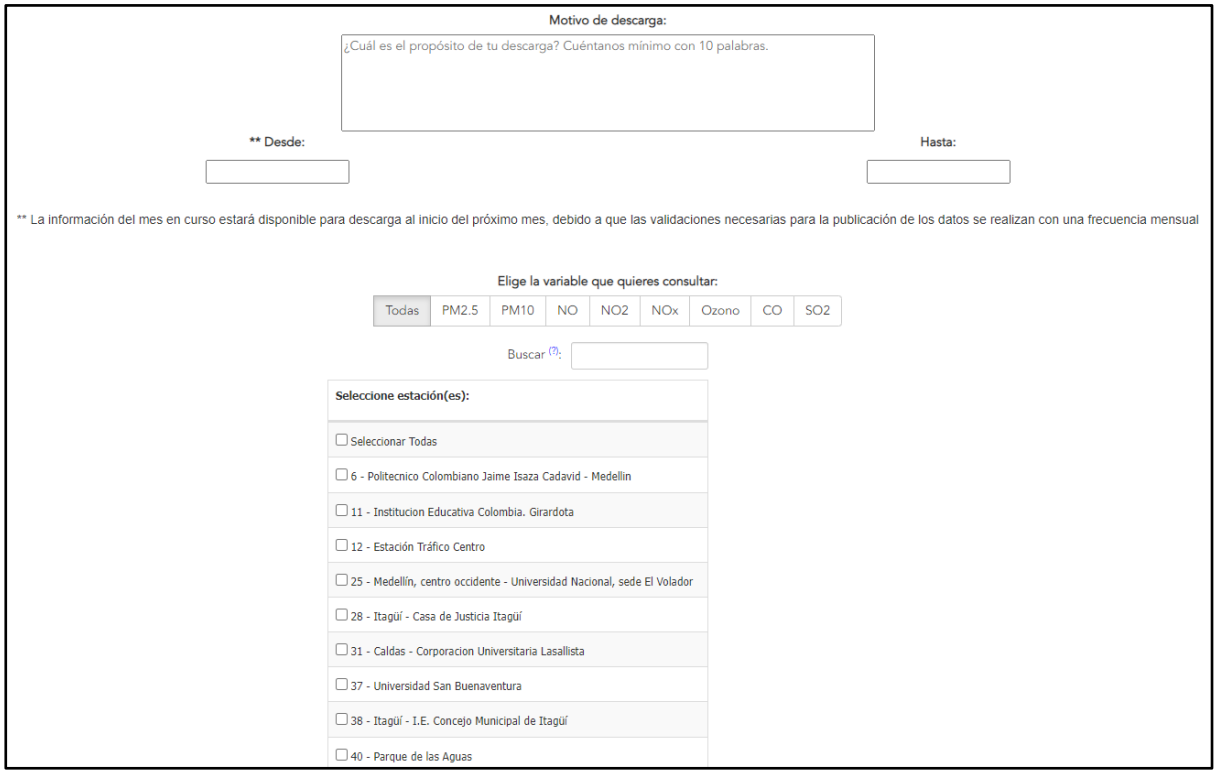


Figura 20. Página de descarga de los datos históricos de PM2.5 y PM10 **[33]**.

Figura 21. Página de descarga de los datos históricos de cada una de las variables meteorológicas **[34]**.

4.1.3. Reunión de los datos

Debido a lo anterior, se hizo necesario descargar la información estación por estación, variable por variable y mes a mes. La información descargada está comprendida entre el periodo del 25 de octubre del 2011 al 25 de octubre del 2021.

4.1.4. Clasificación de la información por estación

La información de cada una de las estaciones fue separada y almacenada en una carpeta.

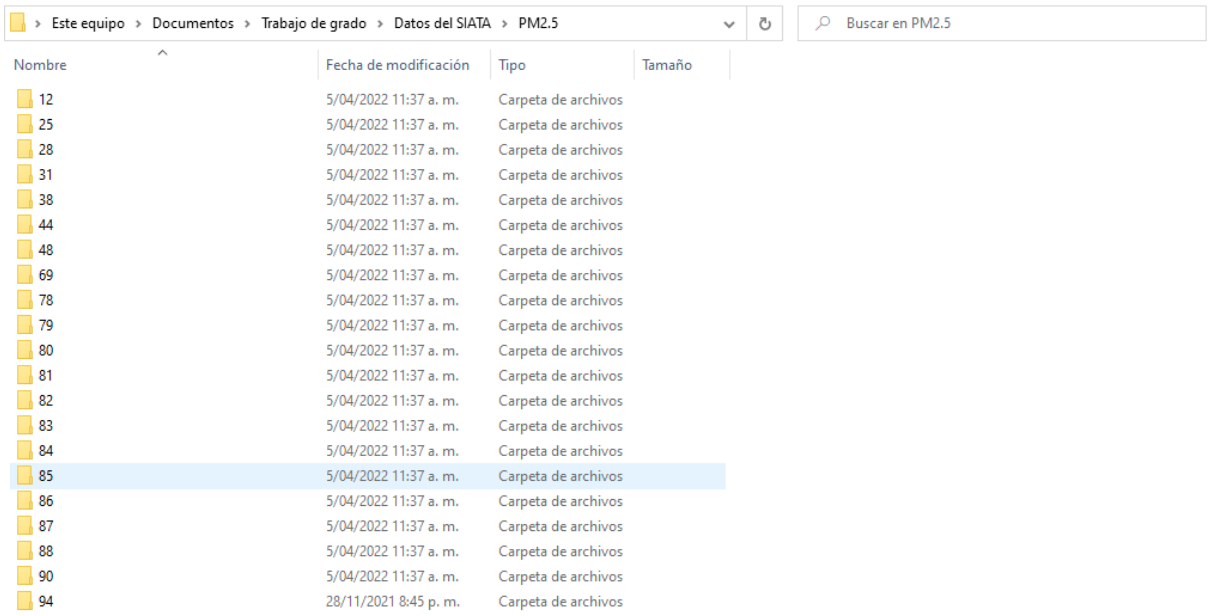


Figura 22. División de la información por carpetas, según la estación.

4.1.5. Análisis de la información recolectada

Se realizó un análisis preliminar de la información recolectada, para así identificar el valor que aporta a la solución del problema cada una de las variables y cada uno de los datos.

En el caso de la información histórica de calidad del aire, cada archivo cuenta con la siguiente información:

Contaminantes	**Nomenclatura**	**Unidades**
Material particulado menor a 1 micra	pm1	$\mu g/m^3$
Material particulado menor a 2.5 micras	pm25	$\mu g/m^3$
Material particulado menor a 10 micras	pm10	$\mu g/m^3$
Ozono	ozono	ppb
Monóxido de carbono	co	ppm
Monóxido de nitrógeno	no	ppb
Dióxido de nitrógeno	no2	ppb

Óxidos de nitrógeno	nox	ppb
Dióxido de azufre	so2	ppb

Tabla 3. Variables presentes en los archivos de calidad del aire **[31]**.

Por cada una de las variables existe un valor de bandera que indica la calidad de la medición (que tan confiable es el dato reportado), así:

Valor	**Calidad del dato**
1	Dato válido
-1	Dato válido por el operado anterior
1.8 - 2.5	Dato dudoso
2.6 - 3.9	Dato malo
>= 4.0	Dato faltante
Dato -9999 y calidad 1	Equipo fuera de operación

Tabla 4. Calidad del dato según el valor de la bandera **[31]**.

En el caso de la información histórica de las variables meteorológicas, cada estación cuenta con la siguiente información:

Variable	Unidades
Precipitación	mm
Temperatura	°C
Presión atmosférica	hPa
Humedad relativa	%
Magnitud de la velocidad del viento	m/s
Dirección del viento	grados
Magnitud de velocidad máxima de viento	m/s
Dirección del viento máximo	grados

Tabla 5. Variables presentes en los archivos de calidad del aire **[32]**.

Por cada una de las variables existe un valor de índice que indica la calidad de la medición (que tan confiable es el dato reportado), así:

Caso	Índice
Calidad confiable del dato en tiempo real	1
Calidad confiable del dato no obtenido en tiempo real	2
Calidad dudosa en dato de tiempo real	151
Calidad dudosa en dato del pluviometro 1 en tiempo real	1511
Calidad dudosa en dato del pluviómetro 2 en tiempo real	1512
Calidad dudosa en dato no obtenido en tiempo real	251

Calidad dudosa en dato no obtenido en tiempo real del pluviometro 1	2511
Calidad dudosa en dato no obtenido en tiempo real del pluviómetro 2	2512

Tabla 6. Indicadores de calidad para la red pluviométrica **[32]**.

Caso	**Índice**
Calidad confiable del dato en tiempo real	1
Calidad confiable del dato no obtenido en tiempo real	2
Calidad dudosa en dato de temperatura en tiempo real	153
Calidad dudosa en dato de humedad relativa en tiempo real	154
Calidad dudosa en dato de presión atmosférica en tiempo real	155
Calidad dudosa en dato de magnitud de viento promedio en tiempo real	156
Calidad dudosa en dato de precipitación en tiempo real	1511
Calidad dudosa en datos de todas las variables	151

Tabla 7. Indicadores de calidad para la red meteorológica **[32]**.

4.1.6. Instalación del software

Para la solución del problema técnico se hizo uso del lenguaje Python, ya que es excelente para el manejo de datos.

Para la manipulación, transformación y combinación de los conjuntos de datos se hizo uso de la librería Pandas que posee un extenso conjunto de funcionalidades. **[19]**

Para los cálculos matemáticos se hizo uso de la librería NumPy. **[20]**

Para graficar la distribución de la información (histogramas y nubes de puntos), los mapas de correlación entre variables y la evolución del error cuando se modifican los parámetros de los algoritmos se hizo uso de la librería Matplotlib. **[21]**

Para la implementación del algoritmo de regresor de bosque aleatorio y la funcionalidad para dividir el conjunto de datos en conjuntos de prueba y entrenamiento de forma aleatoria se hizo uso de la librería Scikit Learn. **[22]**

Para la implementación de la red neuronal artificial; la definición de la topología, métricas y funciones de error se hizo uso de la librería TensorFlow. **[23]**

4.1.7. Unión de la información

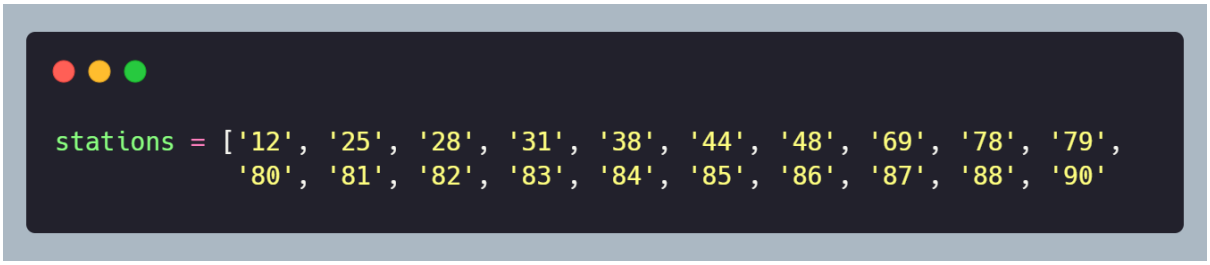
Ya que la información histórica de los niveles de material particulado y la medición de las diferentes variables meteorológicas está dividida en periodos mensuales (donde cada mes corresponde a un archivo diferente), se hizo necesario crear un algoritmo para realizar la unión de estas bases de datos. Los archivos se organizaron de más antiguo a más reciente y se realizó la unión así:

Figura 23. Representación gráfica de la unión mes a mes de todos los archivos.

Para el caso de la información de calidad del aire, la unión de los archivos se realizó así:

```
from os import listdir, chdir, remove
from os.path import isfile, join, exists
import pandas as pd
```

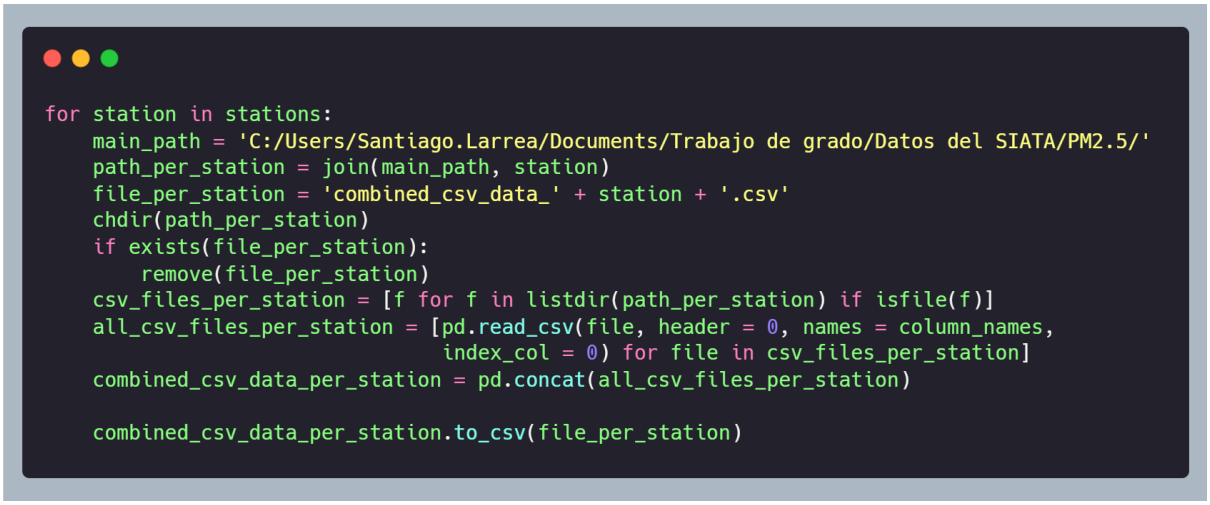
Figura 24. Librerías necesarias para realizar la unión de los archivos.



```
stations = ['12', '25', '28', '31', '38', '44', '48', '69', '78', '79',
            '80', '81', '82', '83', '84', '85', '86', '87', '88', '90']
```

Figura 25. Lista que contiene los códigos de todas las estaciones de calidad del aire.

Para cada una de las estaciones se creó un archivo llamado *combined_csv_data_[NÚMERO DE ESTACIÓN].csv*, luego haciendo uso del método **listdir** de os se obtuvo el nombre de cada uno de los archivos para así con el método **read_csv** y **concat** de pandas generar un archivo que agrupará la información contenida por todos estos archivos. Esta información se guardó en el archivo *combined_csv_data_[NÚMERO DE ESTACIÓN].csv*



```
for station in stations:
    main_path = 'C:/Users/Santiago.Larrea/Documents/Trabajo de grado/Datos del SIATA/PM2.5/'
    path_per_station = join(main_path, station)
    file_per_station = 'combined_csv_data_' + station + '.csv'
    chdir(path_per_station)
    if exists(file_per_station):
        remove(file_per_station)
    csv_files_per_station = [f for f in listdir(path_per_station) if isfile(f)]
    all_csv_files_per_station = [pd.read_csv(file, header = 0, names = column_names,
                                 index_col = 0) for file in csv_files_per_station]
    combined_csv_data_per_station = pd.concat(all_csv_files_per_station)

    combined_csv_data_per_station.to_csv(file_per_station)
```

Figura 26. Código utilizado para unir toda la información histórica de los últimos 10 años, por estación y variable.

Los archivos resultantes contienen entonces la información de los últimos 10 años para cada una de las estaciones y cada una de las variables.

Como se mencionó con anterioridad, ya que las estaciones de calidad del aire y las estaciones meteorológicas no se encuentran en la misma ubicación física, fue necesario realizar un análisis

para obtener las estaciones meteorológicas más cercanas a cada una de las estaciones de calidad del aire. Se realizó un trabajo manual que consistió en obtener las distancias entre las estaciones meteorológicas a cada una de las estaciones de calidad del aire, con ayuda de Google Maps. Las estaciones con menor distancia fueron relacionadas.

Estación de calidad del aire	**Estaciones meteorológicas**
12	202, 271, 419
25	203
28	252
31	450
38	206
44	313
48	229, 318
69	105, 450
78	206, 229, 397
79	197
80	367
81	82
82	73
83	197
84	198

85	83
86	345, 349, 354, 368
87	271
88	252
90	318

Tabla 8. Estaciones meteorológicas cercanas a cada estación de calidad del aire, en orden de cercanía.

Para el caso de la información de las variables meteorológicas, la unión de los archivos se realizó así:

Figura 27. Librerías necesarias para realizar la unión de los archivos.

```
stations_and_related = { '12' : ['202', '271', '419'],
                         '25' : ['203'],
                         '28' : ['252'],
                         '31' : ['450'],
                         '38' : ['206'],
                         '44' : ['313'],
                         '48' : ['229', '318'],
                         '69' : ['105', '450'],
                         '78' : ['206', '229', '397'],
                         '79' : ['197'],
                         '80' : ['367'],
                         '81' : ['82'],
                         '82' : ['73'],
                         '83' : ['197'],
                         '84' : ['198'],
                         '85' : ['83'],
                         '86' : ['345', '349', '354', '368'],
                         '87' : ['271'],
                         '88' : ['252'],
                         '90' : ['318']
                       }
```

Figura 28. Estaciones de calidad del aire relacionadas con las estaciones meteorológicas más cercanas.

```
for key, stations in stations_and_related.items():
    main_path = 'C:/Users/Santiago.Larrea/Documents/Trabajo de grado/Datos del SIATA/VARIABLE/Estación %s' %key
    for station in stations:
        path_per_station = join(main_path, station)
        file_per_station = 'combined_csv_data_' + station + '.csv'
        chdir(path_per_station)
        if exists(file_per_station):
            remove(file_per_station)
        csv_files_per_station = [f for f in listdir(path_per_station) if isfile(f)]
        all_csv_files_per_station = [pd.read_csv(file, header = 0, names = column_names, index_col = 0) for file
                                     in csv_files_per_station]
        combined_csv_data_per_station = pd.concat(all_csv_files_per_station)
        combined_csv_data_per_station.to_csv(file_per_station)
```

Figura 29. Código utilizado para unir toda la información histórica de los últimos 10 años, por estación y variable.

Luego de esto, se reunió por estación toda la información de las diferentes variables, según la fecha de medición de cada dato, así:

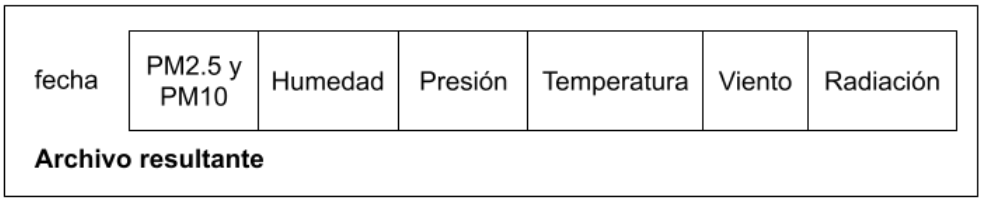


Figura 30. Representación gráfica de la unión de todos los valores de las variables, según la fecha.

```
import pandas as pd
import numpy as np
from datetime import date, timedelta
```

Figura 31. Librerías necesarias para realizar la combinación de los archivos.

```
stations_and_related = { '12' : ['202', '271', '419'],
                         '25' : ['203'],
                         '28' : ['252'],
                         '31' : ['450'],
                         '38' : ['206'],
                         '44' : ['313'],
                         '48' : ['229', '318'],
                         '69' : ['105', '450'],
                         '78' : ['206', '229', '397'],
                         '79' : ['197'],
                         '80' : ['367'],
                         '81' : ['82'],
                         '82' : ['73'],
                         '83' : ['197'],
                         '84' : ['198'],
                         '85' : ['83'],
                         '86' : ['345', '349', '354', '368'],
                         '87' : ['271'],
                         '88' : ['252'],
                         '90' : ['318']
                       }
```

Figura 32. Estaciones de calidad del aire relacionadas con las estaciones meteorológicas más cercanas, en orden de cercanía.

A nivel de implementación en Python, se creó una función llamada merge_data() que recibe como parámetros el número de estaciones de calidad del aire (station) y un arreglo con las estaciones meteorológicas cercanas (related_stations). Se obtuvo el archivo combined_csv_data_[NÚMERO DE ESTACIÓN].csv para cada una de las variables (PM2.5, PM10, humedad, presión, temperatura, velocidad del viento, dirección del viento). Se obtuvo cada uno de los archivos con las columnas de interés haciendo uso del método **read_csv** de pandas. Luego, se definió para todos los conjuntos de datos la columna fecha como índice, esto fue necesario debido a que cada fila del conjunto resultante debía contener el valor medido de todas las variables para una fecha determinada. A continuación, haciendo uso de la función **merge** de pandas, se realizó esta mezcla. La información resultante se guardó en el archivo merged_data_[NÚMERO DE ESTACIÓN].csv.

```python
def merge_data(station, related_stations):
    general_path = 'Estación %s/%s/combined_csv_data_%s.csv' %(station, related_stations[0], related_stations[0])
    pollution = 'C:/Users/Santiago.Larrea/Documents/Trabajo de grado/Datos del SIATA/PM2.5/%s/combined_csv_data_%s.csv' %(station, station)
    humedad = 'C:/Users/Santiago.Larrea/Documents/Trabajo de grado/Datos del SIATA/Humedad/' + general_path
    presion = 'C:/Users/Santiago.Larrea/Documents/Trabajo de grado/Datos del SIATA/Presión/' + general_path
    temperatura = 'C:/Users/Santiago.Larrea/Documents/Trabajo de grado/Datos del SIATA/Temperatura/' + general_path
    viento = 'C:/Users/Santiago.Larrea/Documents/Trabajo de grado/Datos del SIATA/Viento/' + general_path

    column_names_pollution = ['fecha', 'estacion', 'pm25', 'calidad_pm25', 'pm10', 'calidad_pm10', 'pm1', 'calidad_pm1', 'no', 'calidad_no',
                              'no2', 'calidad_no2', 'nox', 'calidad_nox', 'ozono',  'calidad_ozono', 'co', 'calidad_co', 'so2', 'calidad_so2',
                              'pst', 'calidad_pst','dviento_ssr', 'calidad_dviento_ssr', 'haire10_ssr', 'calidad_haire10_ssr', 'p_ssr',
                              'calidad_p_ssr', 'pliquida_ssr', 'calidad_pliquida_ssr', 'rglobal_ssr', 'calidad_rglobal_ssr', 'taire10_ssr',
                              'calidad_taire10_ssr', 'vviento_ssr', 'calidad_vviento_ssr']
    column_names_humedad = ['fecha', 'humedad', 'calidad_humedad']
    column_names_presion = ['fecha', 'presion', 'calidad_presion']
    column_names_temperatura = ['fecha', 'temperatura', 'calidad_temperatura']
    column_names_viento = ['fecha', 'velocidad_prom', 'velocidad_max', 'direccion_prom', 'direccion_max', 'calidad_viento']

    pollution_df = pd.read_csv(pollution, header = 0, names = column_names_pollution)
    humedad_df = pd.read_csv(humedad, header = 0, names = column_names_humedad)
    presion_df = pd.read_csv(presion, header = 0, names = column_names_presion)
    temperatura_df = pd.read_csv(temperatura, header = 0, names = column_names_temperatura)
    viento_df = pd.read_csv(viento, header = 0, names = column_names_viento)

    pollution_df = pollution_df[['fecha', 'estacion', 'pm25', 'calidad_pm25', 'pm10', 'calidad_pm10']]

    pollution_df.set_index(['fecha'])
    humedad_df.set_index(['fecha'])
    presion_df.set_index(['fecha'])
    temperatura_df.set_index(['fecha'])
    viento_df.set_index(['fecha'])

    merged_data = pd.merge(pollution_df, humedad_df, on='fecha', how='inner')
    merged_data = pd.merge(merged_data, presion_df, on='fecha', how='inner')
    merged_data = pd.merge(merged_data, temperatura_df, on='fecha', how='inner')
    merged_data = pd.merge(merged_data, viento_df, on='fecha', how='inner')

    merged_data.to_csv('C:/Users/Santiago.Larrea/Documents/Trabajo de grado/Datos del SIATA/Mezclado/Estación %s/merged_data_%s.csv' %(station,
        station))
```

Figura 33. Función encargada de realizar la combinación de todas las variables meteorológicas en un solo archivo.

Por último, se ejecutó la función merge_data() para cada una de las estaciones de calidad del aire.

```python
for station, related_stations in stations_and_related.items():
    merge_data(station, related_stations)
```

Figura 34. Llamado de la función de la **figura 33** para obtener un archivo combinado por cada una de las estaciones de calidad del aire.

Como resultado, se obtuvo un archivo por estación (de calidad del aire) que contiene toda la información de los niveles históricos de material particulado y los valores históricos de las variables meteorológicas registrados en los últimos 10 años.

fecha	estacion	pm25	calidad_pm25	pm10	calidad_pm10	humedad	calidad_humedad	presion	calidad_presion	temperatura	calidad_temperatura	velocidad_prom	velocidad_max	direccion_prom	direccion_max	calidad_viento
9/9/2015 18:00	12	14.0	-1.0	143.0	-1.0	40.4	1	846.8	1	27.1	1	1.7	3.5	43.0	346.0	1
9/9/2015 19:00	12	35.0	-1.0	106.0	-1.0	41.4	1	847.3	1	26.0	1	1.2	3.0	132.0	312.0	1
9/9/2015 20:00	12	30.0	-1.0	70.0	-1.0	49.5	1	848.5	1	25.0	1	1.5	3.2	127.0	329.0	1
9/9/2015 21:00	12	28.0	-1.0	57.0	-1.0	63.6	1	848.9	1	24.0	1	1.2	3.0	11.0	358.0	1
9/9/2015 22:00	12	18.0	-1.0	48.0	-1.0	67.3	1	849.0	1	23.2	1	2.5	4.8	2.0	359.0	1
9/9/2015 23:00	12	16.0	-1.0	40.0	-1.0	62.5	1	849.9	1	23.2	1	1.7	2.7	51.0	91.0	1
9/10/2015 0:00	12	9.0	-1.0	42.0	-1.0	66.1	1	849.9	1	22.2	1	2.2	3.2	66.0	106.0	1
9/10/2015 6:00	12	19.0	-1.0	22.0	-1.0	67.4	1	850.6	1	19.7	1	0.1	0.5	156.0	345.0	1
9/10/2015 7:00	12	32.0	-1.0	53.0	-1.0	70.3	1	850.5	1	19.3	1	2.2	4.1	209.0	236.0	1
9/10/2015 8:00	12	55.0	-1.0	93.0	-1.0	67.0	1	851.5	1	20.2	1	1.4	2.6	353.0	359.0	1
9/10/2015 9:00	12	60.0	-1.0	95.0	-1.0	59.1	1	851.5	1	21.5	1	0.6	1.7	268.0	345.0	1
9/10/2015 12:00	12	52.0	-1.0	108.0	-1.0	40.6	1	850.1	1	26.2	1	0.0	1.0	132.0	354.0	1
9/10/2015 13:00	12	40.0	-1.0	108.0	-1.0	39.7	1	849.1	1	26.3	1	1.0	2.7	340.0	359.0	1
9/10/2015 14:00	12	32.0	-1.0	82.0	-1.0	46.8	1	848.2	1	25.9	1	2.9	5.2	350.0	357.0	1
9/10/2015 19:00	12	35.0	-1.0	92.0	-1.0	71.5	1	849.6	1	20.2	1	1.4	2.3	285.0	308.0	1
9/10/2015 20:00	12	25.0	-1.0	42.0	-1.0	72.4	1	850.8	1	20.1	1	1.3	2.3	354.0	358.0	1
9/10/2015 21:00	12	31.0	-1.0	43.0	-1.0	71.1	1	851.6	1	20.5	1	1.0	2.7	220.0	325.0	1
9/10/2015 22:00	12	27.0	-1.0	43.0	-1.0	75.4	1	852.5	1	19.9	1	1.5	2.4	232.0	308.0	1
9/10/2015 23:00	12	31.0	-1.0	46.0	-1.0	82.6	1	852.4	1	19.3	1	0.1	0.8	233.0	335.0	1
9/11/2015 0:00	12	29.0	-1.0	43.0	-1.0	83.1	1	852.0	1	19.3	1	0.0	0.8	341.0	358.0	1
9/11/2015 1:00	12	31.0	-1.0	56.0	-1.0	81.0	1	851.3	1	19.5	1	0.8	1.2	303.0	346.0	1
9/11/2015 2:00	12	28.0	-1.0	48.0	-1.0	80.7	1	850.5	1	19.5	1	0.2	0.7	260.0	340.0	1
9/11/2015 3:00	12	37.0	-1.0	55.0	-1.0	78.1	1	850.7	1	19.7	1	0.4	0.7	244.0	297.0	1
9/11/2015 4:00	12	29.0	-1.0	53.0	-1.0	79.5	1	850.3	1	19.1	1	0.8	2.4	337.0	359.0	1
9/11/2015 5:00	12	35.0	-1.0	51.0	-1.0	78.8	1	851.0	1	19.3	1	1.2	1.8	356.0	358.0	1
9/11/2015 6:00	12	38.0	-1.0	81.0	-1.0	78.4	1	851.3	1	19.3	1	0.8	1.9	354.0	359.0	1
9/11/2015 7:00	12	53.0	-1.0	82.0	-1.0	81.8	1	851.9	1	19.0	1	2.2	3.2	321.0	354.0	1
9/11/2015 8:00	12	41.0	-1.0	62.0	-1.0	77.1	1	853.0	1	19.5	1	1.0	2.8	329.0	348.0	1
9/11/2015 9:00	12	32.0	-1.0	84.0	-1.0	71.6	1	853.7	1	20.0	1	1.2	3.1	350.0	359.0	1
9/11/2015 10:00	12	31.0	-1.0	61.0	-1.0	73.1	1	853.6	1	20.7	1	0.6	2.1	219.0	347.0	1
9/11/2015 11:00	12	64.0	-1.0	89.0	-1.0	66.6	1	853.5	1	21.5	1	0.5	2.0	248.0	339.0	1
9/11/2015 12:00	12	67.0	-1.0	102.0	-1.0	56.9	1	852.5	1	22.5	1	2.2	3.5	141.0	163.0	1
9/11/2015 13:00	12	43.0	-1.0	91.0	-1.0	52.6	1	851.3	1	24.6	1	1.2	3.2	359.0	358.0	1
9/11/2015 14:00	12	39.0	-1.0	69.0	-1.0	48.6	1	850.5	1	24.5	1	2.9	5.7	226.0	263.0	1
9/11/2015 15:00	12	29.0	-1.0	51.0	-1.0	51.7	1	850.0	1	24.2	1	2.8	5.8	216.0	286.0	1
9/11/2015 16:00	12	25.0	-1.0	42.0	-1.0	53.1	1	849.5	1	24.2	1	2.0	5.0	216.0	359.0	1
9/11/2015 17:00	12	7.0	-1.0	44.0	-1.0	47.9	1	849.6	1	23.6	1	2.9	3.9	221.0	266.0	1
9/11/2015 18:00	12	21.0	-1.0	53.0	-1.0	53.1	1	850.5	1	22.3	1	4.3	6.1	176.0	231.0	1
9/11/2015 19:00	12	16.0	-1.0	44.0	-1.0	57.8	1	851.4	1	21.6	1	4.2	5.3	192.0	211.0	1

Figura 35. Archivo resultante de combinar todos los datos y todas las variables.

4.1.8. Definición de las entradas y salidas del sistema

Ya que el objetivo del proyecto es estimar los niveles de material particulado presentes en la atmósfera en determinado momento y lugar de Medellín y su área metropolitana, se definieron las entradas y salidas del sistema de la siguiente manera:

- **Entradas:**
    - PM2.5
    - PM10
    - Humedad

- Presión atmosférica
- Temperatura
- Velocidad del viento
- Dirección del viento

- **Salidas:**
  - PM2.5 desplazado dos días
  - PM10 desplazado dos días

Haciendo uso de la función **read_csv** de pandas, como se muestra en la **figura 36**, se obtuvo el archivo generado en el numeral anterior. Luego, se eliminaron los datos que contenían la misma fecha para evitar conflictos a la hora de implementar los algoritmos de aprendizaje de máquina. Por último, se crearon dos columnas "y_pm25" y "y_pm10" que van a ser las que contengan la información de salida para entrenar el sistema.

```
df = pd.read_csv('C:/Users/Santiago.Larrea/Documents/Trabajo de grado/Datos del
                  SIATA/Mezclado/Estación 12/merged_data_12.csv', header = 0, index_col=0)
df = df.drop_duplicates(subset = ['fecha'])
df['y_pm25'] = np.nan
df['y_pm10'] = np.nan
```

Figura 36. Función para eliminar información que contiene la misma fecha y definición de las variables de salida y_pm25, y_pm10.

Ya que el objetivo es estimar los niveles futuros de material particulado con dos días de anticipación, se definió como salida del sistema los valores de PM desplazados dos días, lo que se hizo fue que para cada dato sus entradas corresponden a los valores medidos, mientras que su salida corresponde al valor de PM medido dos días después (e.g. Para el dato medido el 12/10/2015 a las 15:00 sus salidas fueron los valores de PM medidos el 14/10/2015 a las 15:00). Así:

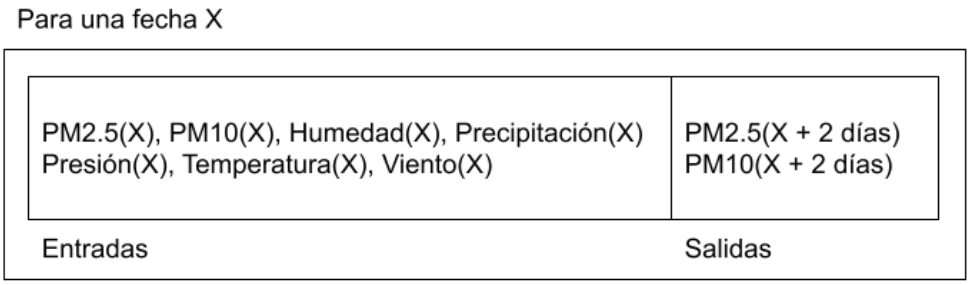


Figura 37. Definición de las entradas y salidas del sistema.

Como se observa en la **figura 38**, se obtuvo cada valor de fecha y hora dentro del archivo y se separó la fecha en una variable y la hora en otra. Luego se generó una fecha nueva adicionando dos días a la fecha actual, haciendo uso del método **timedelta** de datetime. A continuación, se unió la fecha resultante con la hora. Por último, se agregó a las columnas "y_pm25" y "y_pm10", la información correspondiente al PM2.5 y PM10 de la nueva fecha generada, tal como se explica en la **figura 37**.

```
for fecha in df['fecha']:
    _date = fecha.split()[0]
    hour = fecha.split()[1]

    new_date = (date.fromisoformat(_date) + timedelta(days=2)).isoformat()
    new_datetime = '%s %s' %(new_date, hour)

    if not df.loc[df['fecha'] == new_datetime].empty:
        row = df.loc[df['fecha'] == new_datetime]
        df.loc[df['fecha'] == fecha, 'y_pm25'] = float(row['pm25'])
        df.loc[df['fecha'] == fecha, 'y_pm10'] = float(row['pm10'])
```

Figura 38. Desplazamiento de las columnas pm25 y pm10.

4.1.9. Limpieza de la información

Ya que algunos de los datos recolectados presentaban inconsistencias, con el propósito de obtener mejores resultados se realizó un proceso de limpieza de la información.

```
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import os
```

Figura 39. Librerías necesarias para realizar la limpieza de los datos.

Antes que nada, se obtuvo la información resultante del numeral anterior (con las entradas y salidas del sistema), haciendo uso del método **read_csv** de pandas.

```
station = '12'
main_path = 'C:/Users/Santiago.Larrea/Documents/Trabajo de grado/Datos del SIATA/Mezclado/Estación %s' %station
file_name = 'merged_data_%s.csv' %station
full_path = os.path.join(main_path, file_name)

data = pd.read_csv(full_path, header = 0, index_col = 0)
data.head()
```

Figura 40. Porción de código usado para cargar la información combinada para la estación 12.

	fecha	estacion	pm25	calidad_pm25	pm10	calidad_pm10	humedad	calidad_humedad	presion	calidad_presion	temperatura	calidad_temperatura
26724	2015-09-09 18:00:00	12	14.0	-1.0	143.0	-1.0	40.4	1	846.8	1	27.1	1
26725	2015-09-09 19:00:00	12	35.0	-1.0	106.0	-1.0	41.4	1	847.3	1	26.0	1
26726	2015-09-09 20:00:00	12	30.0	-1.0	70.0	-1.0	49.5	1	848.5	1	25.0	1
26727	2015-09-09 21:00:00	12	28.0	-1.0	57.0	-1.0	63.6	1	848.9	1	24.0	1
26728	2015-09-09 22:00:00	12	18.0	-1.0	48.0	-1.0	67.3	1	849.0	1	23.2	1

Figura 41. Primeras 5 filas del conjunto de datos.

	fecha	estacion	pm25	calidad_pm25	pm10	calidad_pm10	humedad	calidad_humedad	presion	calidad_presion	temperatura	calidad_temperatura
82138	2021-10-29 19:00:00	12	41.0	1.0	101.0	1.0	75.8	1	850.2	1	21.1	1
82139	2021-10-29 20:00:00	12	29.0	1.0	77.0	1.0	80.4	1	851.2	1	20.6	1
82140	2021-10-29 21:00:00	12	27.0	1.0	64.0	1.0	82.6	1	852.0	1	20.3	1
82141	2021-10-29 22:00:00	12	25.0	1.0	54.0	1.0	84.1	1	852.1	1	20.0	1
82142	2021-10-29 23:00:00	12	37.0	1.0	61.0	1.0	87.3	1	851.8	1	19.7	1

Figura 42. Últimas 5 filas del conjunto de datos.

Luego, haciendo uso del atributo **shape**, se dio a conocer la cantidad de datos de los que se disponen.

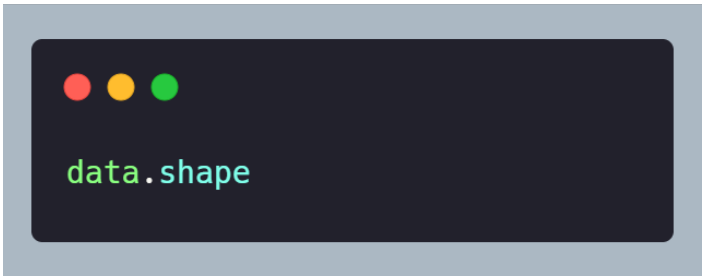


Figura 43. Código utilizado para conocer las dimensiones del conjunto de datos.

```
(43848, 19)
```

Figura 44. Dimensiones del conjunto de datos de la estación 12.

Para el caso de la estación 12, se disponen de 43848 filas y 19 variables (o columnas). Para conocer las variables del conjunto de datos se hizo uso del atributo **columns.values**.

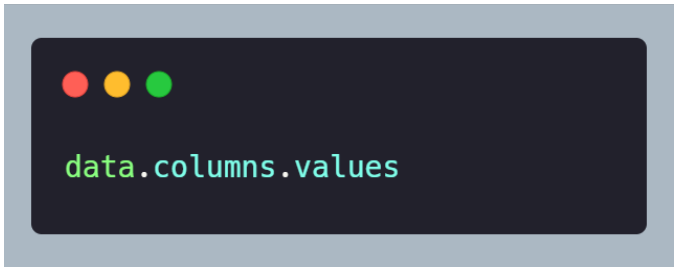


Figura 45. Porción de código para conocer las columnas del conjunto de datos.

```
['fecha' 'estacion' 'pm25' 'calidad_pm25' 'pm10' 'calidad_pm10' 'humedad'
 'calidad_humedad' 'presion' 'calidad_presion' 'temperatura'
 'calidad_temperatura' 'velocidad_prom' 'velocidad_max' 'direccion_prom'
 'direccion_max' 'calidad_viento' 'y_pm25' 'y_pm10']
```

Figura 46. Columnas del conjunto de datos.

A continuación, como muestra la **figura 47**, se validó si existía información faltante dentro del conjunto de datos. El resultado fue cero (no faltaba información).

```
null_fields = [pd.isnull(data[column]).values.ravel().sum() for column in data]
sum(null_fields)
```

Figura 47. Código para validar si hay datos faltantes.

Luego, haciendo uso de las banderas que indican la calidad de cada uno de los datos (ver **tabla 2, tabla 4** y **tabla 5**) se descartaron las medidas erróneas y los datos registrados por el equipo cuando estaba fuera de operación. Así, en la base de datos sólo se conservó la información clasificada como válida.

```
filtered_data = data
for column in data.columns:
        if ("calidad" in column):
            calidad = data[column]
            condition = (calidad == 1.0) | (calidad == -1.0) | ((calidad >= 1.8)
                        & (calidad <= 2.5))
            filtered_data = data.loc[condition]
        elif (column != 'estacion'):
            value = filtered_data[column]
            condition = (value != -9999) & (value != 9999)
            filtered_data = filtered_data.loc[condition]
```

Figura 48. Filtrado de la información según la información de las banderas de calidad.

Con la finalidad de conocer la fiabilidad de la información resultante después del filtrado, se hizo uso del método **describe**, que arroja información relevante para cada una de las columnas numéricas, como son: la media, la desviación estándar, el valor mínimo y máximo, etc. Con esta información es posible saber si algún dato está fuera de los rangos esperados.

```
filtered_data.describe()
```

Figura 49. Porción de código para describir la información filtrada de cada una de las columnas numéricas del conjunto de datos.

Como se observa en la **figura 50**, la información resultante luego del filtrado es consistente.

	estacion	pm25	calidad_pm25	pm10	calidad_pm10	humedad	calidad_humedad	presion	calidad_presion	temperatura
count	43848.0	43848.000000	43848.000000	43848.000000	43848.000000	43848.000000	43848.000000	43848.000000	43848.000000	43848.000000
mean	12.0	29.478948	0.841614	54.898522	0.848654	63.114359	1.005861	850.311371	1.005861	22.317241
std	0.0	15.885905	0.693952	24.926103	0.700360	16.324133	0.076334	1.805110	0.076334	3.152073
min	12.0	0.000000	-1.000000	2.400000	-1.000000	14.900000	1.000000	843.900000	1.000000	14.600000
25%	12.0	19.000000	1.000000	37.000000	1.000000	50.300000	1.000000	849.100000	1.000000	19.800000
50%	12.0	27.000000	1.000000	51.000000	1.000000	67.200000	1.000000	850.500000	1.000000	21.500000
75%	12.0	37.000000	1.000000	69.000000	1.000000	76.200000	1.000000	851.600000	1.000000	24.600000
max	12.0	170.000000	2.500000	242.000000	2.500000	97.900000	2.000000	855.800000	2.000000	32.000000

calidad_temperatura	velocidad_prom	velocidad_max	direccion_prom	direccion_max	calidad_viento	y_pm25	y_pm10
43848.000000	43848.000000	43848.000000	43848.000000	43848.000000	43848.000000	43848.000000	43848.000000
1.005861	1.635292	3.050739	164.125616	206.572044	1.005861	29.544344	54.976245
0.076334	1.350513	1.962890	115.469296	124.277797	0.076334	16.039491	25.162101
1.000000	0.000000	0.100000	0.000000	0.000000	1.000000	0.000000	2.400000
1.000000	0.700000	1.600000	57.000000	90.000000	1.000000	19.000000	37.000000
1.000000	1.300000	2.600000	144.000000	217.000000	1.000000	27.000000	51.000000
1.000000	2.200000	4.000000	269.000000	338.000000	1.000000	37.000000	69.000000
2.000000	13.000000	18.000000	359.000000	359.000000	2.000000	170.000000	242.000000

Figura 50. Descripción de las columnas numéricas luego del filtrado.

Luego, se graficó la distribución de los datos de PM2.5 a través del tiempo, con la finalidad de observar el comportamiento de estos al pasar de los años.

```
plt.scatter(filtered_data.index, filtered_data.pm25)
plt.xlabel('Tiempo')
plt.ylabel('Nivel de PM2.5')
plt.show()
```

Figura 51. Código utilizado para graficar la distribución de los datos de PM2.5 y PM10 a través del tiempo.

Con la finalidad de conocer cuáles son los valores de PM2.5 y PM10 más frecuentes, se graficó el histograma.

```
plt.hist(filtered_data.pm25, bins = 20)
plt.title('Distribución del PM2.5')
plt.xlabel('Nivel de PM2.5')
plt.ylabel('Frecuencia')
plt.show()
```

```
plt.hist(filtered_data.pm10, bins = 20)
plt.title('Distribución del PM10')
plt.xlabel('Nivel de PM10')
plt.ylabel('Frecuencia')
plt.show()
```

Figura 52. Código utilizado para graficar el histograma del PM2.5 y PM10.

4.2. Identificación de las variables meteorológicas de mayor influencia en relación con la contaminación de la atmósfera

4.2.1. Investigación sobre patrones meteorológicas

En esta etapa, se realizó una investigación sobre patrones climáticos con la finalidad de comprender qué relación existe entre los patrones climáticos presentes en Medellín y su área metropolitana con la contaminación del aire.

4.2.2. Investigación sobre variables meteorológicas

En esta etapa, se realizó una investigación de las diferentes variables meteorológicas, a que hace referencia cada una y como se realizan las mediciones de estas dentro de las estaciones meteorológicas de Medellín y su área metropolitana. Las variables meteorológicas presentes en el proyecto son: Humedad, presión atmosférica, temperatura, velocidad del viento y dirección del viento.

4.2.3. Análisis de la relación entre las variables meteorológicas y la contaminación del aire

Con la finalidad de entender mejor la relación entre las variables meteorológicas y la calidad del aire de Medellín y su área metropolitana, se hizo uso de los datos anteriormente recolectados, unificados y saneados.

Ya que dentro de la información se encontraban todavía las banderas de calidad y teniendo en cuenta que la información ya se había filtrado, las columnas correspondientes a esta información fueron eliminadas. También fueron eliminadas las columnas que no contenían información numérica.

```
float64_data = filtered_data.select_dtypes(include = ['float64'])
columns_of_interest = [column for column in float64_data.columns if "calidad_" not in column]
filtered_important_data = float64_data[columns_of_interest]
filtered_important_data.head()
```

Figura 53. Código utilizado para eliminar las columnas con datos no numéricos y las columnas correspondientes a las banderas de calidad.

pm25	pm10	humedad	presion	temperatura	velocidad_prom	velocidad_max	direccion_prom	direccion_max	y_pm25	y_pm10
14.0	143.0	40.4	846.8	27.1	1.7	3.5	43.0	346.0	21.0	53.0
35.0	106.0	41.4	847.3	26.0	1.2	3.0	132.0	312.0	16.0	44.0
30.0	70.0	49.5	848.5	25.0	1.5	3.2	127.0	329.0	32.0	76.0
28.0	57.0	63.6	848.9	24.0	1.2	3.0	11.0	358.0	36.0	74.0
18.0	48.0	67.3	849.0	23.2	2.5	4.8	2.0	359.0	34.0	61.0

Figura 54. Primeras 5 filas del conjunto de datos resultante.

Se hizo uso de la correlación, ya que esta es una medida que nos indica que tanta relación existe entre dos variables, es decir, si una variable cambia que tanto afecta esto a las demás. El método **corr()** de pandas arroja una matriz de correlación entre las variables del conjunto de datos.

```
correlation_matrix = filtered_important_data.corr()
correlation_matrix
```

Figura 55. Código utilizado para obtener la matriz de correlación de las variables del conjunto de datos.

	pm25	pm10	humedad	presion	temperatura	velocidad_prom	velocidad_max	direccion_prom	direccion_max	y_pm25	y_pm10
pm25	1.000000	0.805204	0.029894	0.198180	-0.070714	-0.172303	-0.169414	0.095614	0.169354	0.566011	0.421859
pm10	0.805204	1.000000	-0.131673	0.036768	0.089233	-0.053434	-0.021360	0.077939	0.152364	0.434176	0.451748
humedad	0.029894	-0.131673	1.000000	0.471273	-0.935875	-0.474374	-0.534552	0.037985	0.021984	-0.016869	-0.147392
presion	0.198180	0.036768	0.471273	1.000000	-0.564055	-0.400776	-0.442746	0.062611	0.044829	0.181555	0.048328
temperatura	-0.070714	0.089233	-0.935875	-0.564055	1.000000	0.489273	0.567994	-0.051628	-0.037362	-0.035654	0.096315
velocidad_prom	-0.172303	-0.053434	-0.474374	-0.400776	0.489273	1.000000	0.924022	-0.128844	-0.169488	-0.107479	-0.012345
velocidad_max	-0.169414	-0.021360	-0.534552	-0.442746	0.567994	0.924022	1.000000	-0.136680	-0.131800	-0.098938	0.008469
direccion_prom	0.095614	0.077939	0.037985	0.062611	-0.051628	-0.128844	-0.136680	1.000000	0.554510	0.069420	0.059296
direccion_max	0.169354	0.152364	0.021984	0.044829	-0.037362	-0.169488	-0.131800	0.554510	1.000000	0.162003	0.147775
y_pm25	0.566011	0.434176	-0.016869	0.181555	-0.035654	-0.107479	-0.098938	0.069420	0.162003	1.000000	0.807179
y_pm10	0.421859	0.451748	-0.147392	0.048328	0.096315	-0.012345	0.008469	0.059296	0.147775	0.807179	1.000000

Figura 56. Matriz de correlación de las variables del conjunto de datos.

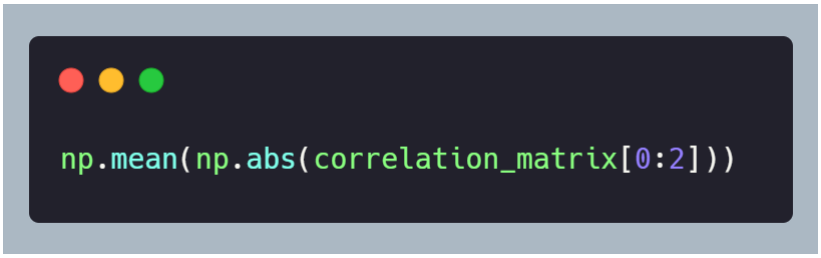
Con la finalidad de observar de forma más amigable la información obtenida en la matriz de correlación, se hizo uso del método **matshow()** que nos permite crear un mapa de calor de esta matriz.

```
fig = plt.figure()
ax = fig.add_subplot(111)
cax = ax.matshow(correlation_matrix, cmap='coolwarm', vmin=-1, vmax=1)
fig.colorbar(cax)
ticks = np.arange(0, len(columns_of_interest), 1)
ax.set_xticks(ticks)
plt.xticks(rotation = 90)
ax.set_yticks(ticks)
ax.set_xticklabels(columns_of_interest)
ax.set_yticklabels(columns_of_interest)
plt.show()
```

Figura 57. Código utilizado para crear un mapa de calor de la matriz de correlación.

4.2.4. Identificación de variables meteorológicas con mayor influencia en la calidad del aire

En base a los resultados obtenidos en la matriz de correlación, se hizo uso del código de la **figura 58**, para obtener la magnitud de la correlación existente entre cada variable, el PM2.5 y PM10. Para conocer las variables meteorológicas más influyentes en la calidad del aire, los valores de correlación con el PM2.5 y PM10 se promediaron haciendo uso del método **mean()** de numpy.



```
np.mean(np.abs(correlation_matrix[0:2]))
```

Figura 58. Código para obtener el promedio de la magnitud de la correlación de cada variable con el PM2.5 y PM10.

```
pm25              0.902602
pm10              0.902602
humedad           0.080784
presion           0.117474
temperatura       0.079973
velocidad_prom    0.112869
velocidad_max     0.095387
direccion_prom    0.086777
direccion_max     0.160859
y_pm25            0.500094
y_pm10            0.436804
```

Figura 59. Promedio de la magnitud de la correlación de cada variable con el PM2.5 y PM10.

4.3. Evaluación de técnicas de aprendizaje de máquina

4.3.1. Investigación sobre aprendizaje de máquina

Con la finalidad de identificar el mejor algoritmo de aprendizaje de máquina para realizar la predicción de material particulado presente en la atmósfera de la ciudad de Medellín y su área metropolitana con dos días de anticipación, se realizó una investigación sobre aprendizaje de máquina y las diferentes técnicas existentes. Se reconoce al aprendizaje de máquina con el potencial para dar solución al problema planteado.

4.3.2. Investigación sobre diferentes técnicas de aprendizaje de máquina

Se realizó una investigación sobre las diferentes técnicas de aprendizaje de máquina, que tipo de problema podía resolverse con cada una de ellas y cuáles eran las ventajas y desventajas de cada una. Ya que se tenía un problema de regresión, se tomó la decisión de implementar las siguientes técnicas:

- Regresor de bosque aleatorio.
- Red neuronal artificial (RNA).

4.4. Definir e implementar algoritmos para la estimación de los niveles futuros de material particulado

4.4.1. Aplicar varios algoritmos de aprendizaje de máquina a la información recolectada

En esta etapa, se aplicaron las diferentes técnicas de aprendizaje de máquina definidas anteriormente.

El sistema implementado fue el siguiente:

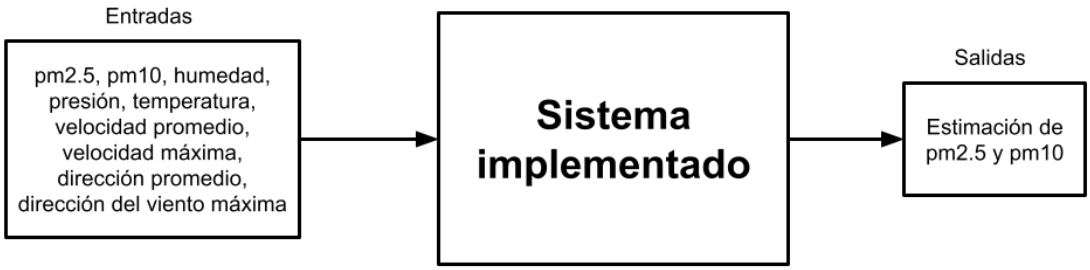


Figura 60. Sistema implementado para la estimación de los niveles de material particulado.

4.4.1.1. Bosque aleatorio

Se realizó la implementación de un bosque aleatorio con 100 árboles, haciendo uso del método **RandomForestRegressor** de la librería sklearn, la estructura de un bosque aleatorio es la siguiente:

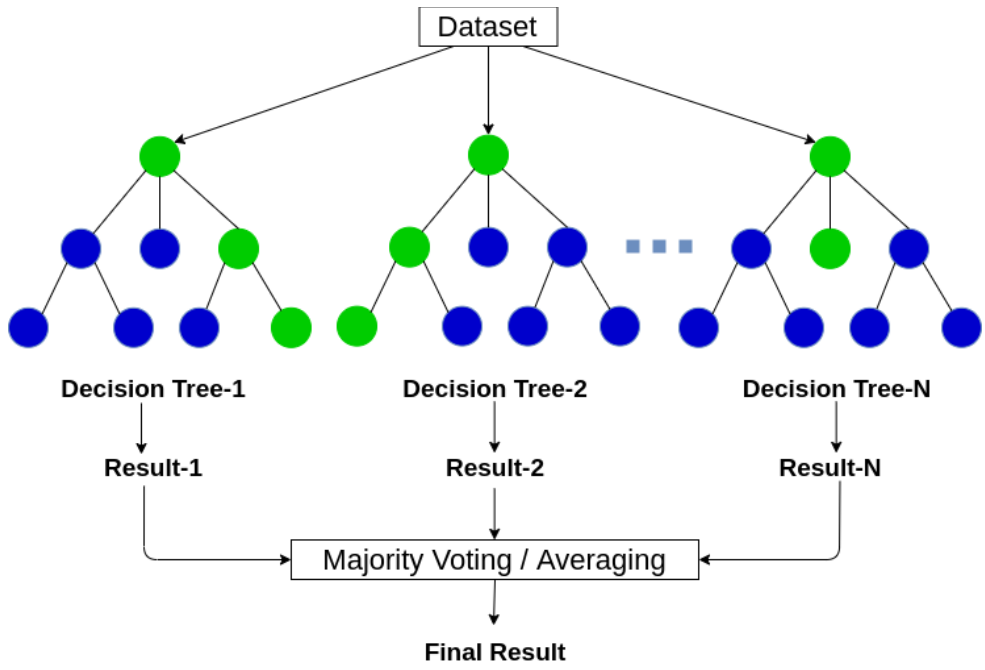


Figura 61. Diagrama de bosque aleatorio para N árboles **[35]**.

Inicialmente, se importaron todas las librerías necesarias para completar el trabajo.

```
import pandas as pd
import numpy as np
from sklearn.ensemble import RandomForestRegressor
from sklearn.preprocessing import MinMaxScaler
from sklearn.compose import make_column_transformer
from sklearn.model_selection import train_test_split
```

Figura 62. Librerías necesarias para realizar la implementación del regresor de bosque aleatorio.

Con la finalidad de analizar con qué conjunto de variables se obtienen mejores resultados y así obtener un modelo final más preciso, se entrenó este algoritmo de la siguiente manera: Primero se definió un conjunto de entrada con todas las variables (tanto material particulado como variables meteorológicas). Luego, en cada iteración, se fueron eliminando una a una las variables meteorológicas, en orden de menor relación con la contaminación ambiental (como se definió en la **sección 2.2**). Finalmente se compararon los resultados.

Para cada iteración se realizó el siguiente proceso:

Haciendo uso del método **read_csv**, se leyó el archivo anteriormente limpiado. Luego, se creó un array que contiene los nombres de las columnas a eliminar según la iteración. Así, se seleccionaron las variables de entrada al algoritmo para cada una de las iteraciones.

```
df = pd.read_csv('../../Datos del SIATA/Mezclado/Estación 12/merged_data_12.csv', index_col=0)
to_drop = ['fecha', 'estacion', 'temperatura', 'humedad', 'direccion_prom', 'velocidad_max',

to_drop = to_drop[0:i]
[to_drop.append(column) for column in df.columns if 'calidad' in column]

df.drop(columns=to_drop, inplace=True)
```

Figura 63. Código usado para seleccionar las variables de entrada al algoritmo.

Luego, se creó un código para normalizar las variables de entrada y salida haciendo uso del método **MinMaxScaler()** de sklearn que sirve para normalizar (definir los valores entre 0 y 1) un conjunto de datos y del método **make_column_transformer()** que permite aplicar cualquier transformación a un grupo de columnas determinado del conjunto de datos. Estos métodos usados para normalizar se conocen como transformadores.

```
columns_to_transform = ['pm25', 'pm10', 'humedad', 'presion', 'temperatura', 'velocidad_prom',
                        'velocidad_max', 'direccion_prom', 'direccion_max']

for j in to_drop:
    if j in columns_to_transform:
        columns_to_transform.remove(j)

transformer = make_column_transformer(
    (MinMaxScaler(), columns_to_transform)
)

y_transformer = make_column_transformer(
    (MinMaxScaler(), ['y_pm25', 'y_pm10'])
)
```

Figura 64. Código usado para normalizar las variables de entrada y salida.

Luego, el conjunto de datos se dividió en dos grupos, un 80% de los datos se usaron para entrenar el algoritmo y el 20% para validar la precisión de estimación del algoritmo entrenado.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figura 65. Código para dividir el conjunto de datos en subconjuntos para entrenamiento y pruebas.

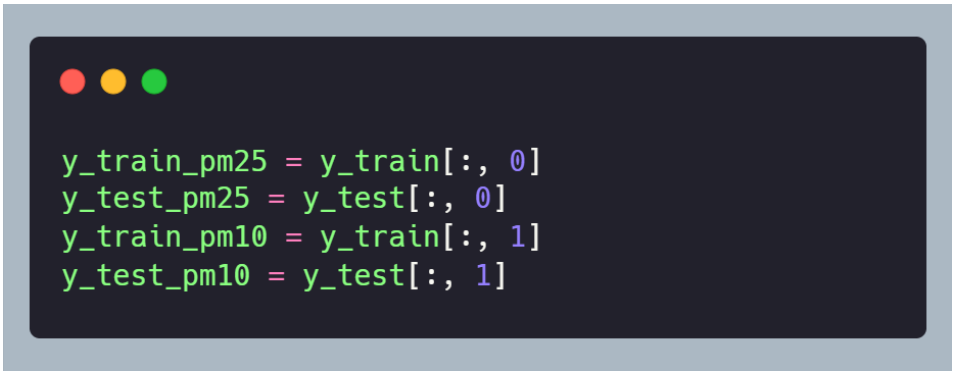
Luego se aplicaron los transformadores definidos en la **figura 64** a los subconjuntos de entrenamiento y pruebas.

```
transformer.fit(X_train)
X_train = transformer.transform(X_train)
X_test = transformer.transform(X_test)

y_transformer.fit(y_train)
y_train = y_transformer.transform(y_train)
y_test = y_transformer.transform(y_test)
```

Figura 66. Código para normalizar los subconjuntos de entrenamiento y pruebas.

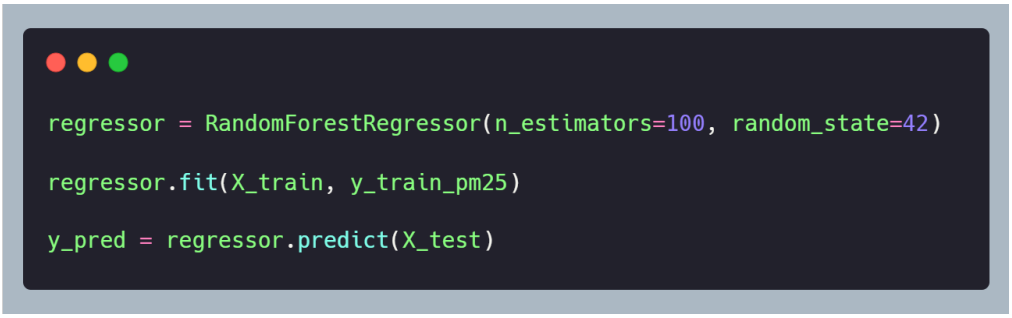
Ya que el algoritmo de bosques aleatorio solo nos permite tener una variable de salida, fue necesario entrenar y analizar los resultados en dos etapas, la primera para PM2.5 y la segunda para PM10.



```
y_train_pm25 = y_train[:, 0]
y_test_pm25 = y_test[:, 0]
y_train_pm10 = y_train[:, 1]
y_test_pm10 = y_test[:, 1]
```

Figura 67. Código para separar los subconjuntos de entrenamiento y pruebas por variable.

Luego, se entrenó el algoritmo haciendo uso del subconjunto de entrenamiento y se realizó la estimación haciendo uso del subconjunto de pruebas.



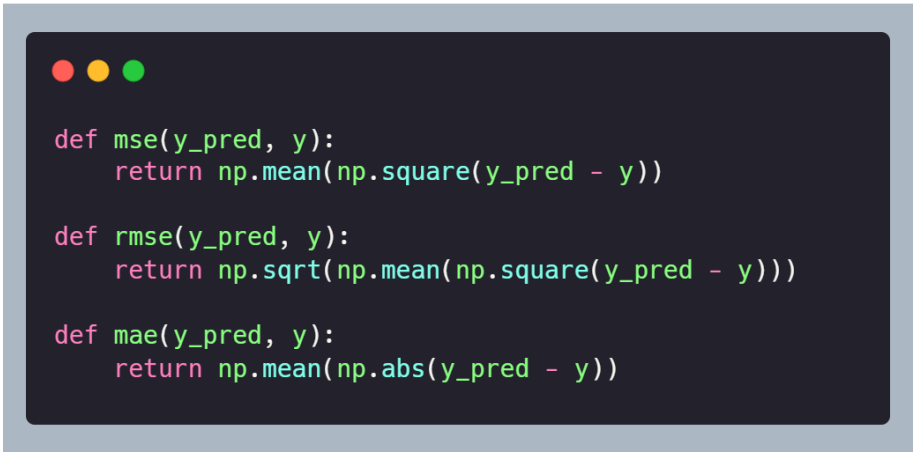
```
regressor = RandomForestRegressor(n_estimators=100, random_state=42)

regressor.fit(X_train, y_train_pm25)

y_pred = regressor.predict(X_test)
```

Figura 68. Código usado para entrenar el algoritmo y realizar la estimación.

Finalmente, se aplicaron los estimadores de error. Para este algoritmo se hizo uso del RMSE, MSE y MAE. Los resultados obtenidos fueron guardados en un arreglo para luego comparar.



```
def mse(y_pred, y):
    return np.mean(np.square(y_pred - y))

def rmse(y_pred, y):
    return np.sqrt(np.mean(np.square(y_pred - y)))

def mae(y_pred, y):
    return np.mean(np.abs(y_pred - y))
```

Figura 69. Funciones para calcular los diferentes estimadores de error utilizados.

```
error_mse = mse(y_pred, y_test_pm25)
error_rmse = rmse(y_pred, y_test_pm25)
error_mae = mae(y_pred, y_test_pm25)

error_info = {
    'mse': error_mse,
    'rmse': error_rmse,
    'mae': error_mae
}

errors.append(error_info)
```

Figura 70. Código utilizado para calcular los diferentes estimadores de error.

4.4.1.2. Red neuronal artificial

Se realizó la implementación de una red neuronal artificial con la siguiente topología:

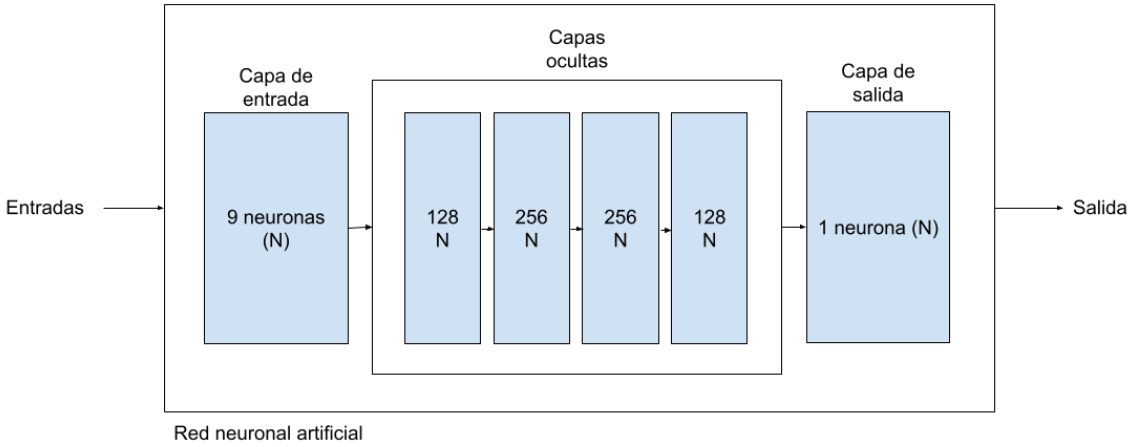
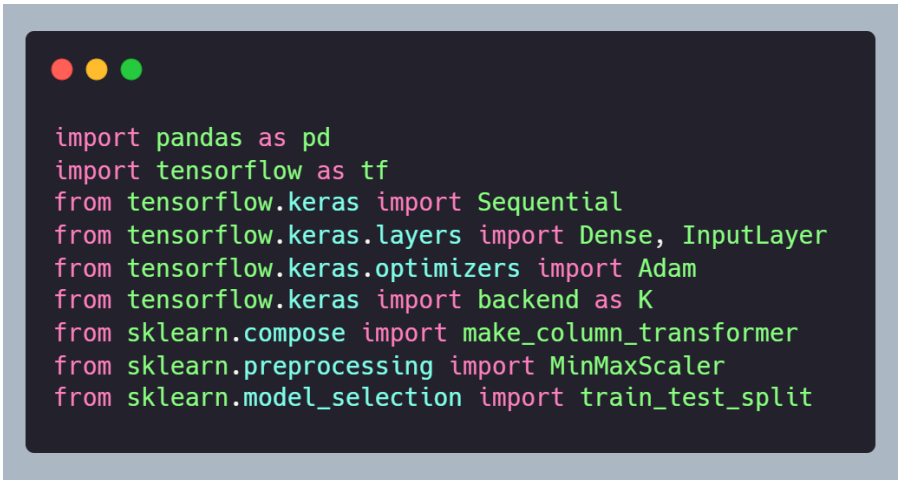


Figura 71. Representación en bloques de la topología de la red neuronal implementada.

Inicialmente se importaron todas las librerías necesarias para completar la tarea.



```
import pandas as pd
import tensorflow as tf
from tensorflow.keras import Sequential
from tensorflow.keras.layers import Dense, InputLayer
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import backend as K
from sklearn.compose import make_column_transformer
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
```

Figura 72. Librerías necesarias para realizar la implementación de la red neuronal artificial.

Con la finalidad de analizar con qué conjunto de variables se obtienen mejores resultados y así obtener un modelo final más preciso, se entrenó este algoritmo de la siguiente manera: Primero se definió un conjunto de entrada con todas las variables (tanto material particulado como variables meteorológicas). Luego, en cada iteración, se fueron eliminando una a una las variables meteorológicas, en orden de menor relación con la contaminación ambiental (como se definió en la **sección 4.2**). Finalmente se compararon los resultados.

Para cada iteración se realizó el siguiente proceso:

Haciendo uso del método **read_csv**, se leyó el archivo anteriormente limpiado. Luego, se creó un array que contiene los nombres de las columnas a eliminar según la iteración. Así, se seleccionaron las variables de entrada al algoritmo para cada una de las iteraciones.

```
df = pd.read_csv('../../Datos del SIATA/Mezclado/Estación 12/merged_data_12.csv', index_col=0)
to_drop = ['fecha', 'estacion', 'temperatura', 'humedad', 'direccion_prom', 'velocidad_max',

to_drop = to_drop[0:i]
[to_drop.append(column) for column in df.columns if 'calidad' in column]

df.drop(columns=to_drop, inplace=True)
```

Figura 73. Código usado para seleccionar las variables de entrada al algoritmo.

Luego, se creó un código para normalizar las variables de entrada y salida haciendo uso del método **MinMaxScaler()** de sklearn que sirve para normalizar (definir los valores entre 0 y 1) un conjunto de datos y del método **make_column_transformer()** que permite aplicar cualquier transformación a un grupo de columnas determinado del conjunto de datos. Estos métodos usados para normalizar se conocen como transformadores.

```
columns_to_transform = ['pm25', 'pm10', 'humedad', 'presion', 'temperatura', 'velocidad_prom',
                        'velocidad_max', 'direccion_prom', 'direccion_max']

for j in to_drop:
    if j in columns_to_transform:
        columns_to_transform.remove(j)

transformer = make_column_transformer(
    (MinMaxScaler(), columns_to_transform)
)

y_transformer = make_column_transformer(
    (MinMaxScaler(), ['y_pm25', 'y_pm10'])
)
```

Figura 74. Código usado para normalizar las variables de entrada y salida.

Luego, el conjunto de datos se dividió en dos grupos, un 80% de los datos se usaron para entrenar el algoritmo y el 20% para validar la precisión de estimación del algoritmo entrenado.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Figura 75. Código para dividir el conjunto de datos en subconjuntos para entrenamiento y pruebas.

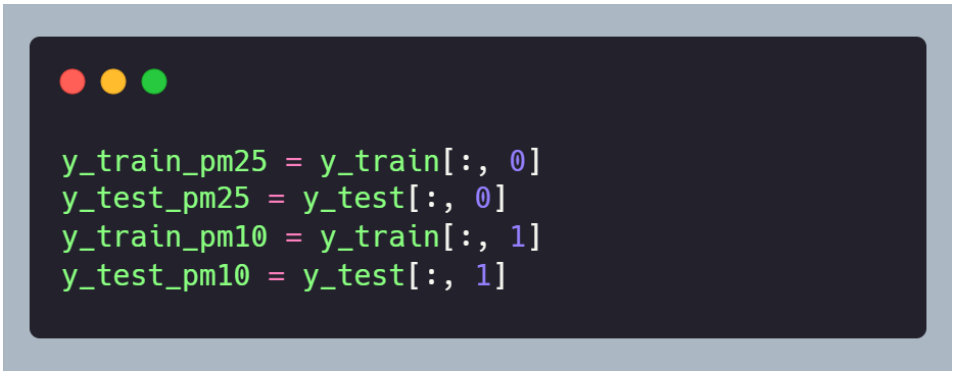
Luego se aplicaron los transformadores definidos en **figura 74** a los subconjuntos de entrenamiento y pruebas.

```
transformer.fit(X_train)
X_train = transformer.transform(X_train)
X_test = transformer.transform(X_test)

y_transformer.fit(y_train)
y_train = y_transformer.transform(y_train)
y_test = y_transformer.transform(y_test)
```

Figura 76. Código para normalizar los subconjuntos de entrenamiento y pruebas.

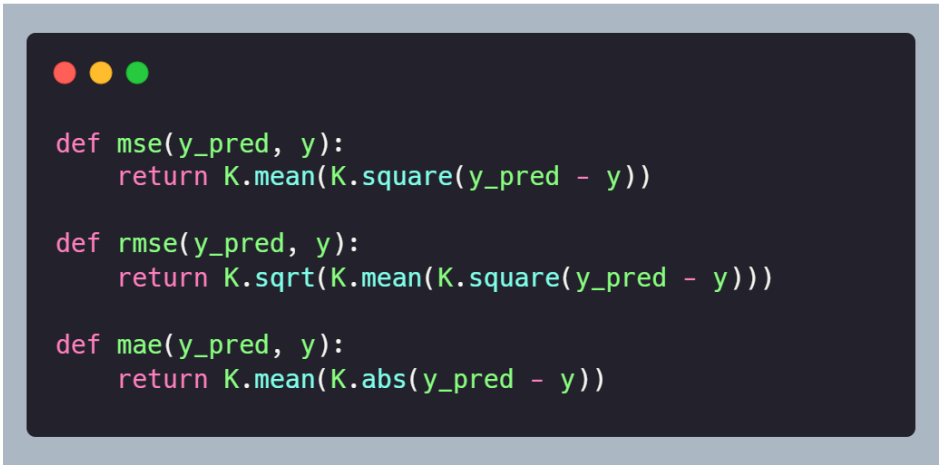
Ya que el algoritmo de bosques aleatorio solo nos permite tener una variable de salida, fue necesario entrenar y analizar los resultados en dos etapas, la primera para PM2.5 y la segunda para PM10.



```
y_train_pm25 = y_train[:, 0]
y_test_pm25 = y_test[:, 0]
y_train_pm10 = y_train[:, 1]
y_test_pm10 = y_test[:, 1]
```

Figura 77. Código para separar los subconjuntos de entrenamiento y pruebas por variable.

Luego, se crearon las funciones para calcular los diferentes estimadores de error. Para este algoritmo se hizo uso del RMSE, MSE y MAE.



```
def mse(y_pred, y):
    return K.mean(K.square(y_pred - y))

def rmse(y_pred, y):
    return K.sqrt(K.mean(K.square(y_pred - y)))

def mae(y_pred, y):
    return K.mean(K.abs(y_pred - y))
```

Figura 78. Funciones para calcular los diferentes estimadores de error utilizados.

Luego, se creó el modelo siguiendo la topología planteada en la **figura 71**, haciendo uso del método **Sequential** de tensorflow. La capa de entrada se definió de tipo **InputLayer** y las demás capas se definieron de tipo **Dense**.

```python
tf.random.set_seed(42)

number_of_inputs = 11 - i

model = Sequential([
    InputLayer(input_shape=(number_of_inputs,), dtype=tf.float64),
    Dense(128, use_bias=True, activation='sigmoid'),
    Dense(256, use_bias=True, activation='sigmoid'),
    Dense(256, use_bias=True, activation='sigmoid'),
    Dense(128, use_bias=True, activation='sigmoid'),
    Dense(1, use_bias=True)
])
```

Figura 79. Código usado para definir la topología de la red neuronal.

Luego, se compilo el modelo y se entrenó con el conjunto de datos de entrenamiento.

```python
model.compile(loss=rmse, optimizer=Adam(), metrics=[rmse])
model.fit(X_train, y_train, epochs=100)
```

Figura 80. Código usado para entrenar el modelo.

Finalmente, se realizó la estimación haciendo uso del subconjunto de pruebas y se aplicaron a los resultados obtenidos los estimadores de error. Los resultados obtenidos para cada iteración fueron guardados en un arreglo para luego comparar.

```python
predictions = model.predict(X_test)

error_mse = mse(predictions, y_test_pm25)
error_rmse = rmse(predictions, y_test_pm25)
error_mae = mae(predictions, y_test_pm25)

error_info = {
    'mse': error_mse.numpy(),
    'rmse': error_rmse.numpy(),
    'mae': error_mae.numpy()
}

errors.append(error_info)
```

Figura 81. Código utilizado para realizar la estimación y calcular los diferentes estimadores de error.

4.4.2. Realizar la estimación para diferentes momentos y sectores de Medellín y su área metropolitana

Luego de compilar los modelos de aprendizaje de máquina, haciendo uso del subconjunto reservado para pruebas (correspondiente a un 20% del total de la información), se realizó la estimación de los niveles futuros de material particulado. Se realizó la comparación del desempeño de los dos algoritmos implementados.

5. Resultados y análisis

En este capítulo se muestran las principales gráficas y tablas de los resultados obtenidos al comparar la presentación de los datos obtenidos antes y después de pasar por un proceso de limpieza. Además de esto, se presentan las gráficas correspondientes a los resultados obtenidos al realizar el análisis de la relación de las diferentes variables meteorológicas, con respecto a la calidad del aire (niveles de PM2.5 y PM10). Estas variables meteorológicas son: temperatura, humedad relativa, presión, velocidad del viento y dirección del viento. Por último, se presentan las gráficas correspondientes a la implementación de los diferentes algoritmos de aprendizaje de máquina aplicados a los datos recolectados, limpiados y normalizados. Los modelos implementados son: Regresor de bosque aleatorio y Red neuronal artificial. Se realiza una comparativa entre los resultados obtenidos al implementar estos dos modelos con la finalidad de establecer cuál es el que tiene mejor desempeño.

5.1. Limpieza de los datos

De la **figura 82** a la **figura 99**, se muestran la distribución de los datos a través del tiempo y la distribución de frecuencia de estos datos. Lo anterior se hace para cada una de las variables, tanto de material particulado como variables meteorológicas antes de realizar la limpieza de los datos. Estas imágenes son contrastadas con la información obtenida luego de realizar la limpieza, como se muestra de la **figura 100** a la **figura 117**.

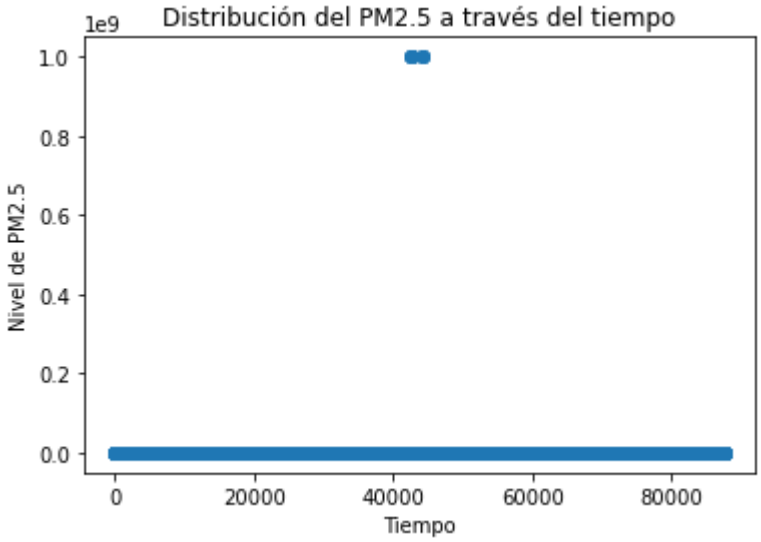


Figura 82. Distribución del PM2.5 a través del tiempo, antes de la limpieza.

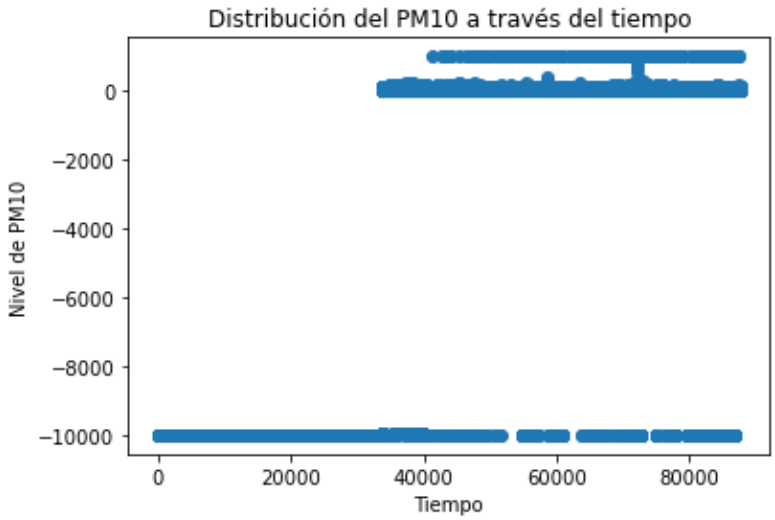


Figura 83. Distribución del PM10 a través del tiempo, antes de la limpieza.

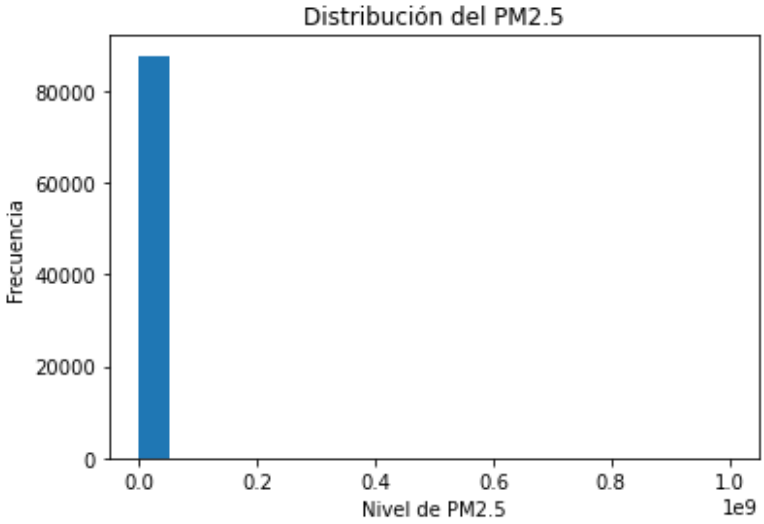


Figura 84. Distribución de frecuencia del PM2.5, antes de la limpieza.

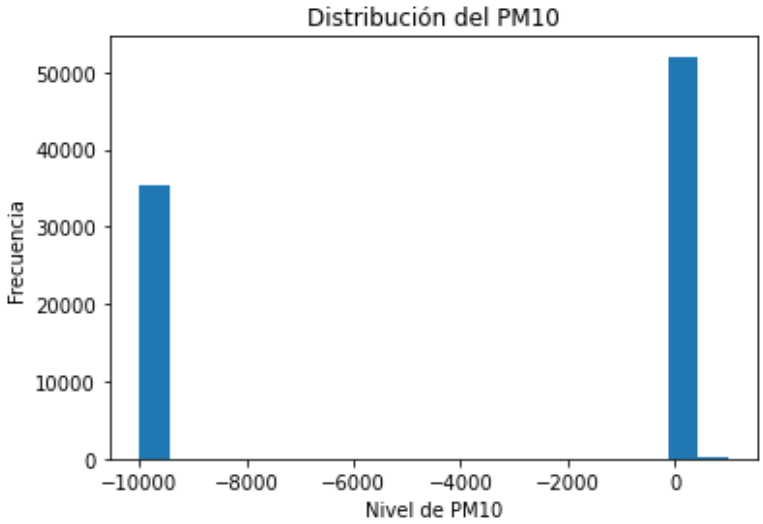


Figura 85. Distribución de frecuencia del PM10, antes de la limpieza.

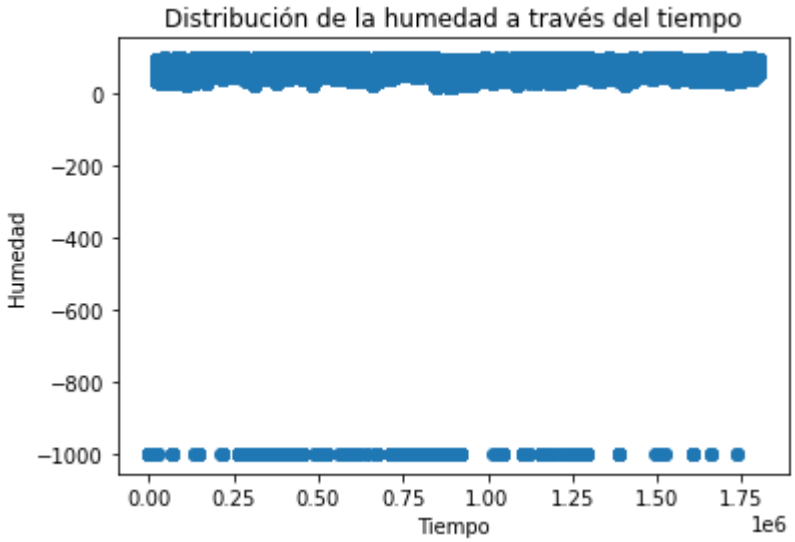


Figura 86. Distribución de la humedad a través del tiempo, antes de la limpieza.

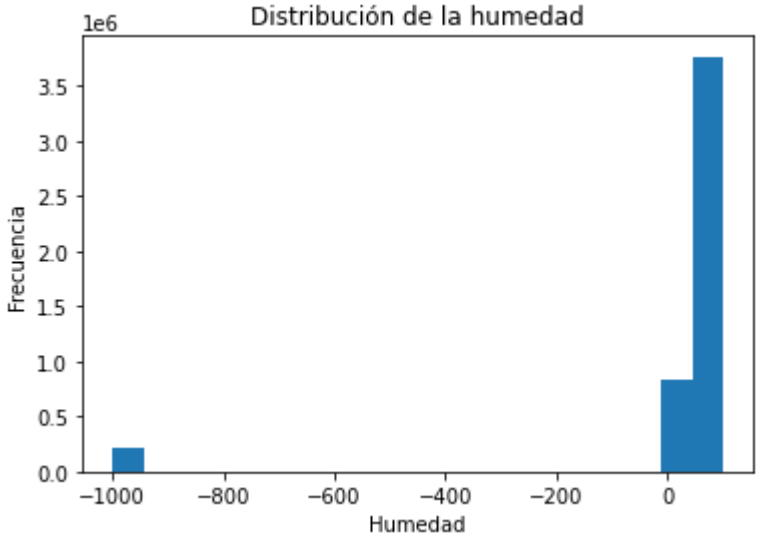


Figura 87. Distribución de frecuencia de la humedad, antes de la limpieza.

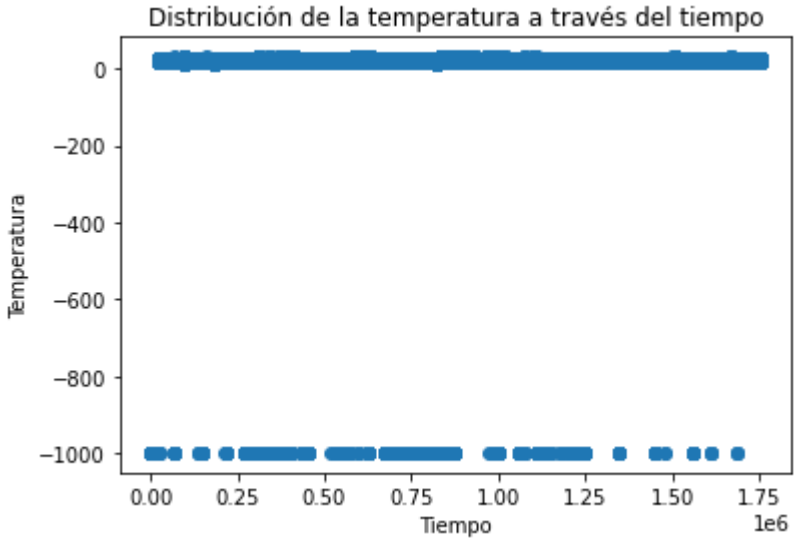


Figura 88. Distribución de la temperatura a través del tiempo, antes de la limpieza.

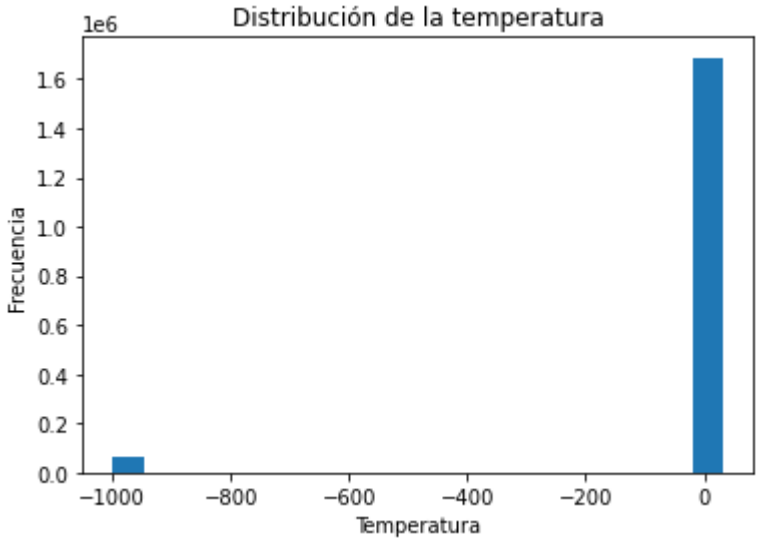


Figura 89. Distribución de frecuencia de la temperatura, antes de la limpieza.

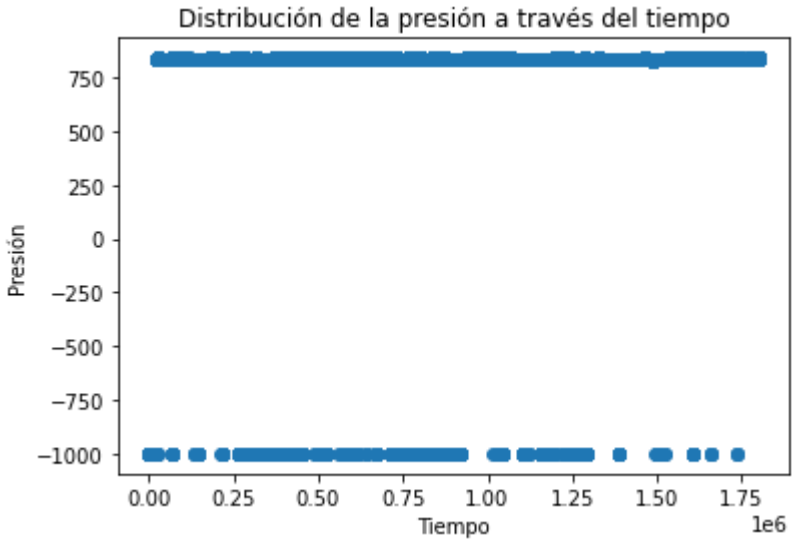


Figura 90. Distribución de la presión a través del tiempo, antes de la limpieza.

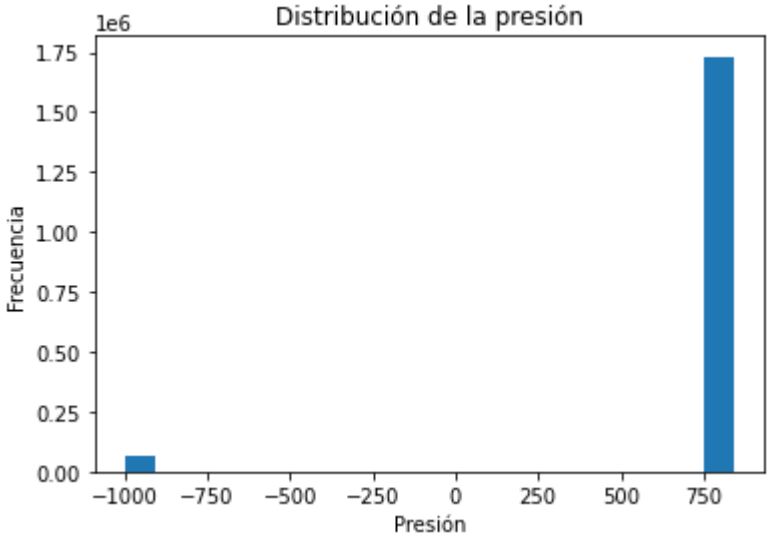


Figura 91. Distribución de frecuencia de la presión, antes de la limpieza.

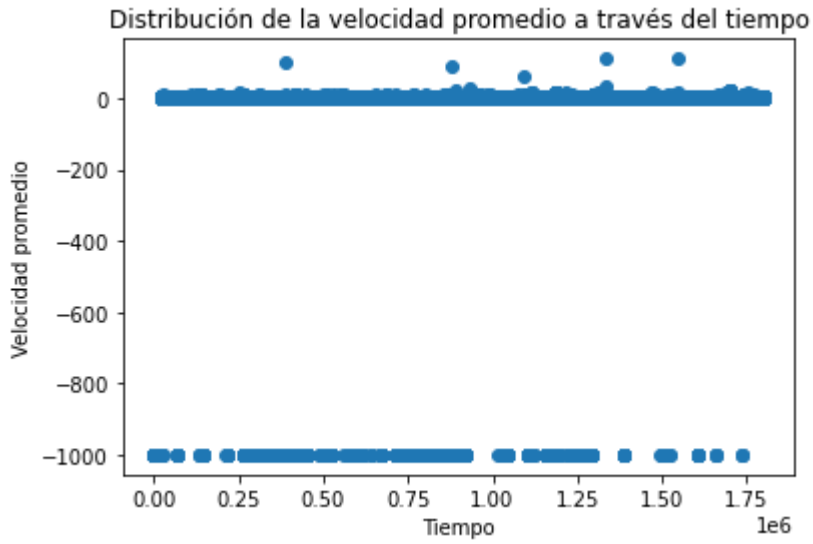


Figura 92. Distribución de frecuencia de la velocidad promedio, antes de la limpieza.

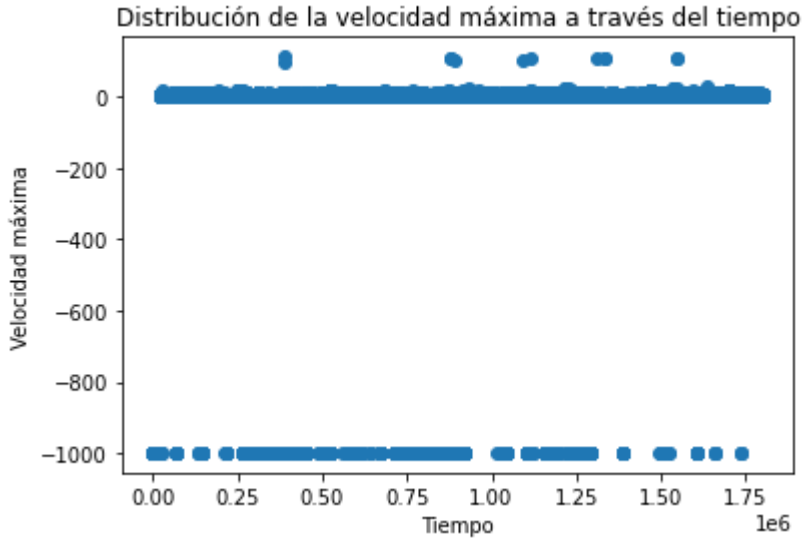


Figura 93. Distribución de la velocidad máxima a través del tiempo, antes de la limpieza.

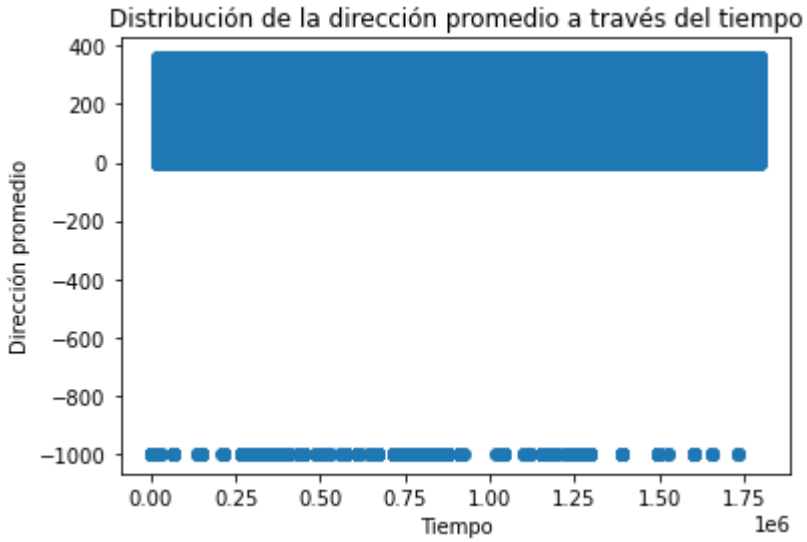


Figura 94. Distribución de la dirección promedio a través del tiempo, antes de la limpieza.

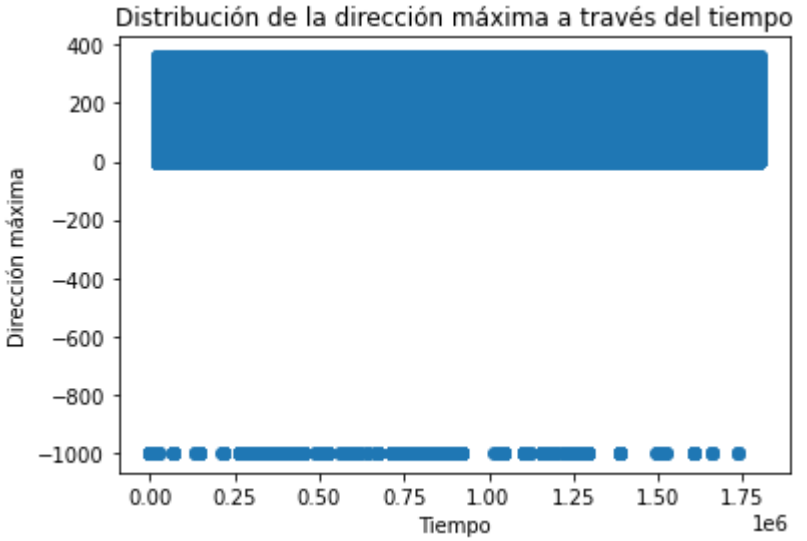


Figura 95. Distribución de la dirección máxima a través del tiempo, antes de la limpieza.

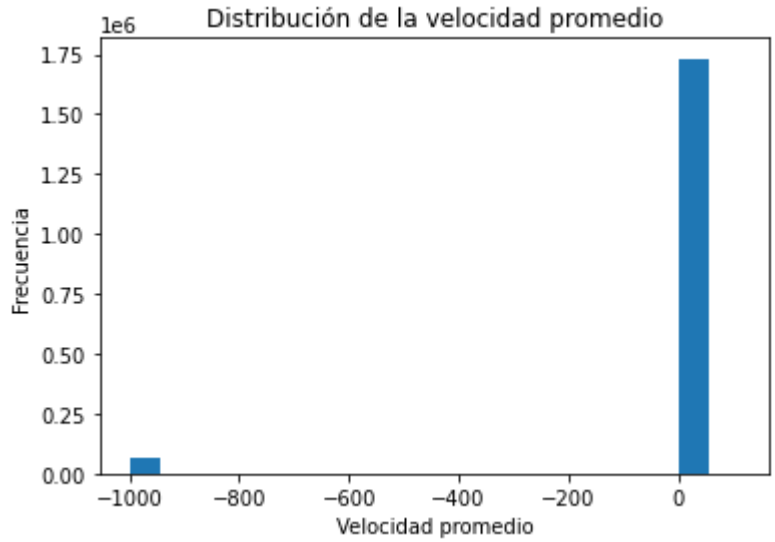


Figura 96. Distribución de la velocidad promedio a través del tiempo, antes de la limpieza.

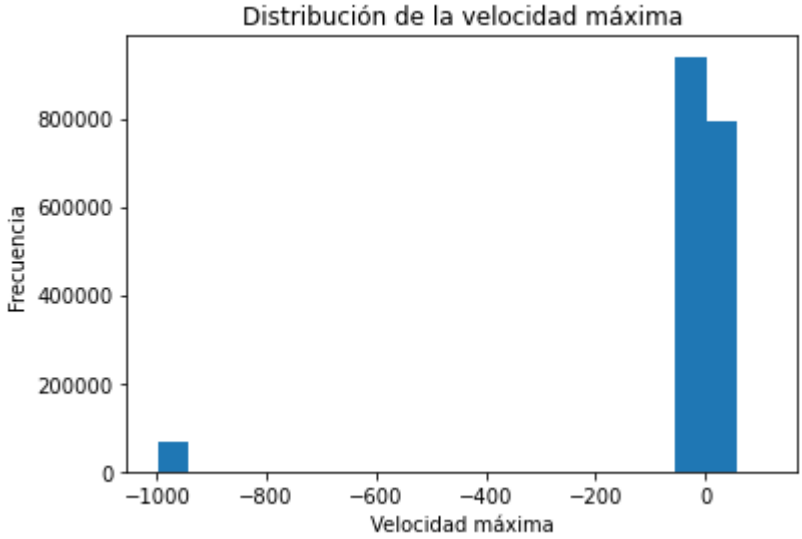


Figura 97. Distribución de frecuencia de la velocidad máxima, antes de la limpieza.

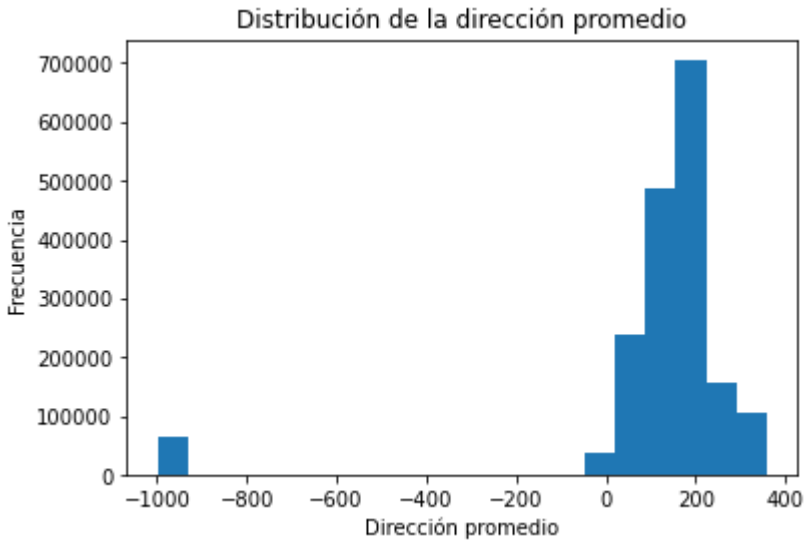


Figura 98. Distribución de frecuencia de la dirección promedio, antes de la limpieza.

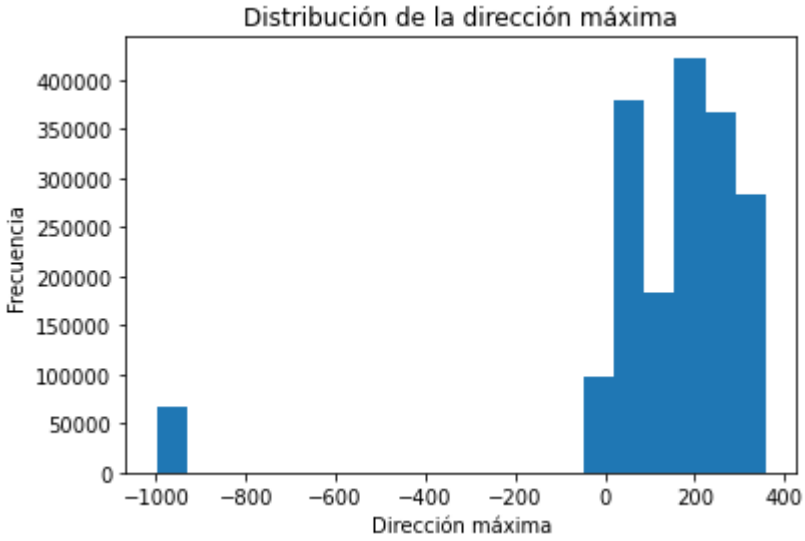


Figura 99. Distribución de frecuencia de la dirección máxima, antes de la limpieza.

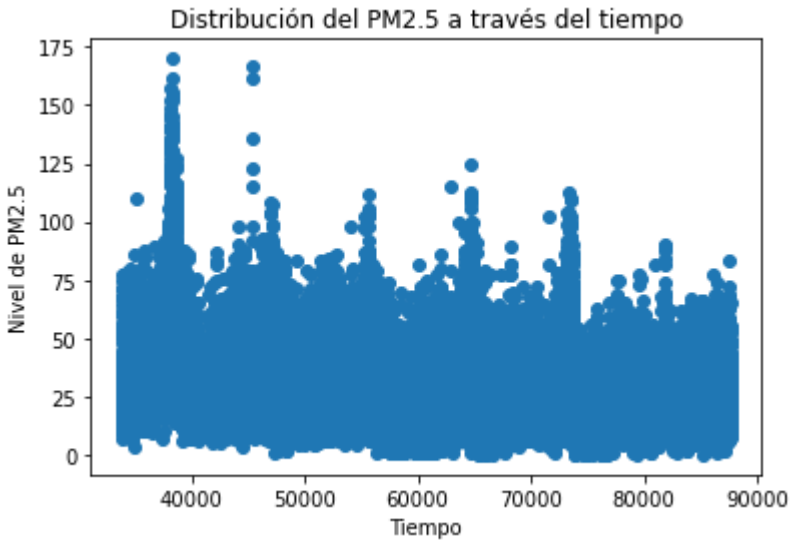


Figura 100. Distribución del PM2.5 a través del tiempo, después de la limpieza.

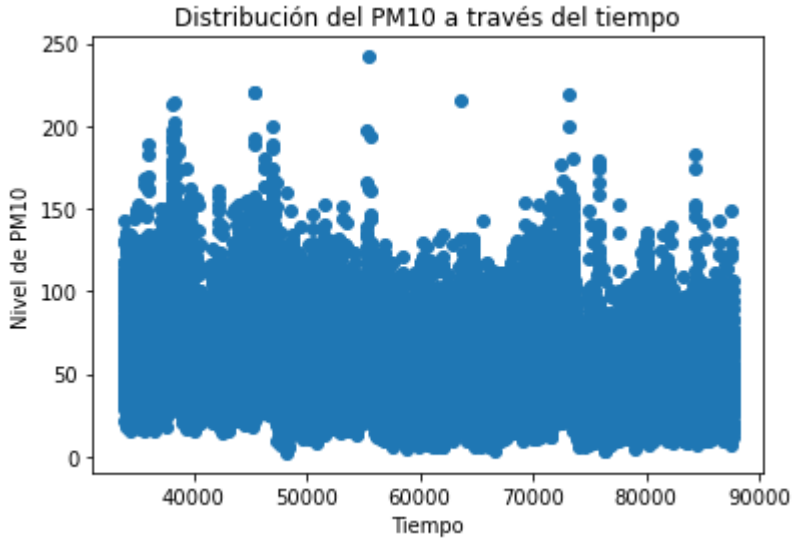


Figura 101. Distribución del PM10 a través del tiempo, después de la limpieza.

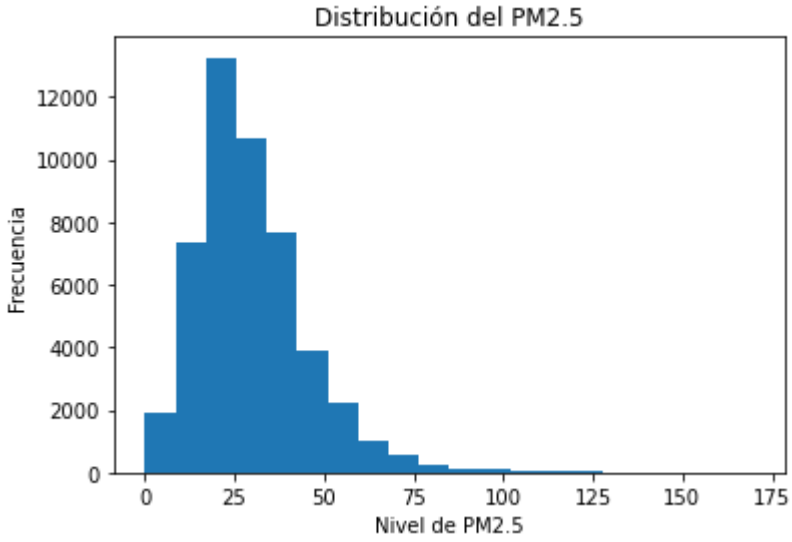


Figura 102. Distribución de frecuencia del PM2.5, después de la limpieza.

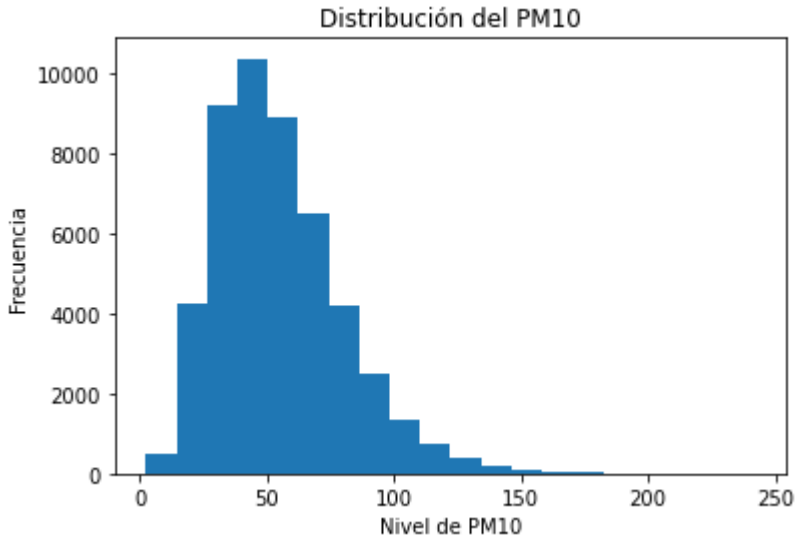


Figura 103. Distribución de frecuencia del PM10, después de la limpieza.

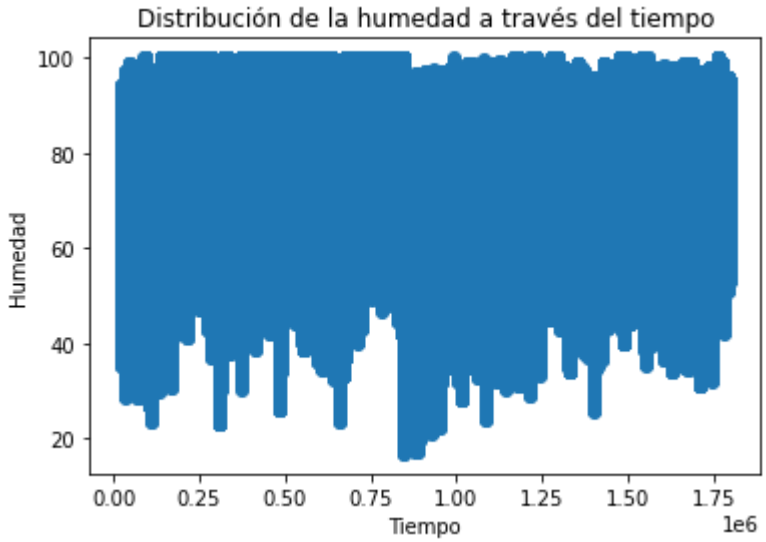


Figura 104. Distribución de la humedad a través del tiempo, después de la limpieza de los datos.

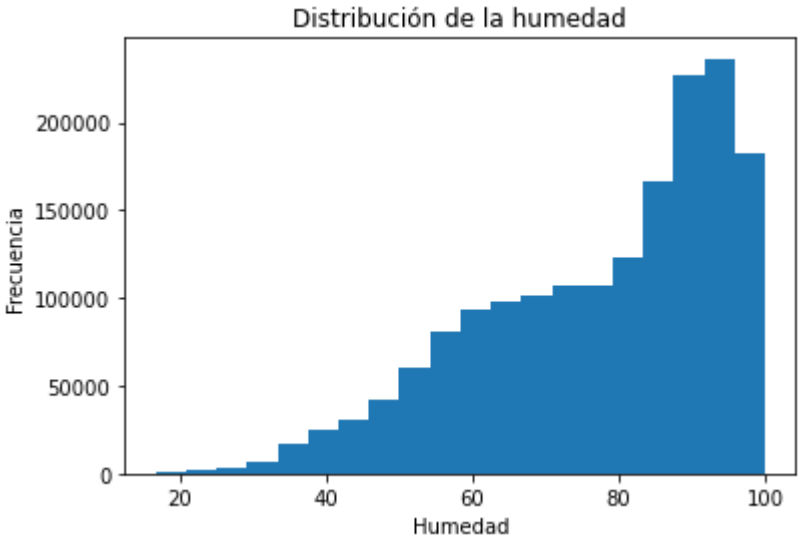


Figura 105. Distribución de frecuencia de la humedad, después de la limpieza.

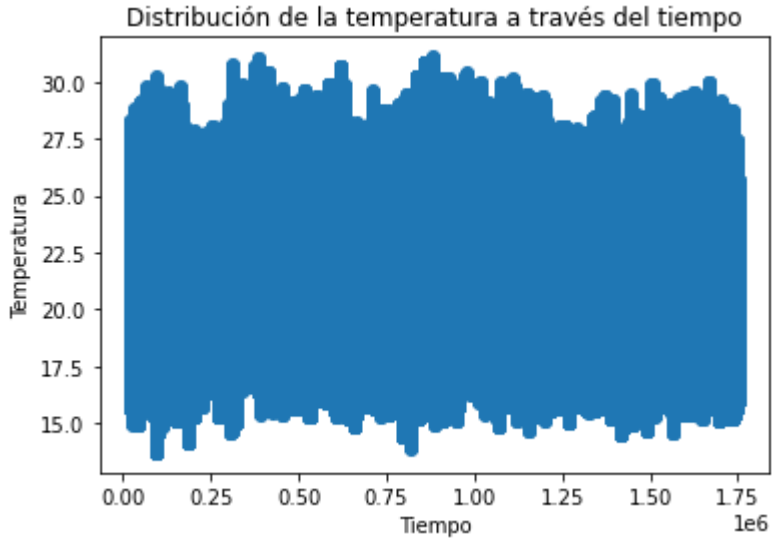


Figura 106. Distribución de la temperatura a través del tiempo, después de la limpieza.

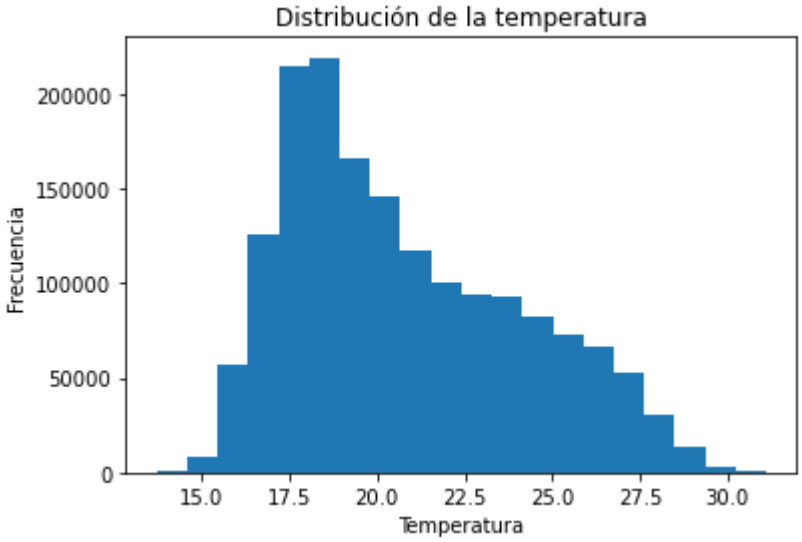


Figura 107. Distribución de frecuencia de la temperatura, después de la limpieza.

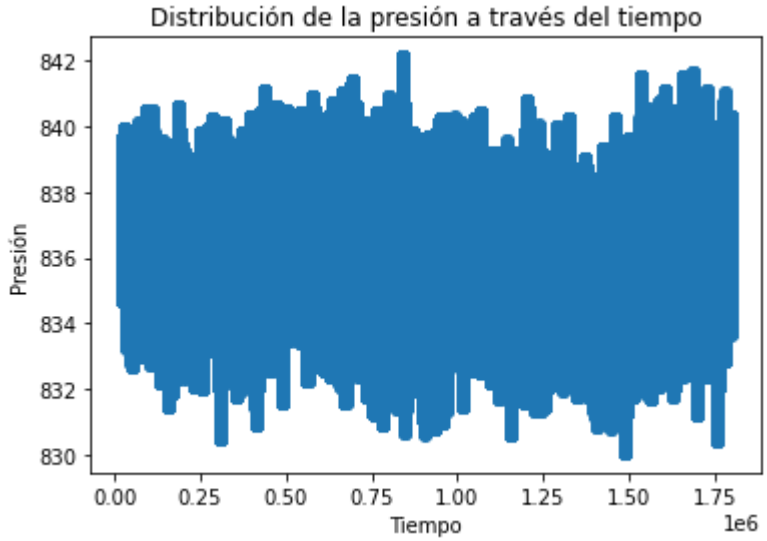


Figura 108. Distribución de la presión a través del tiempo, después de la limpieza.

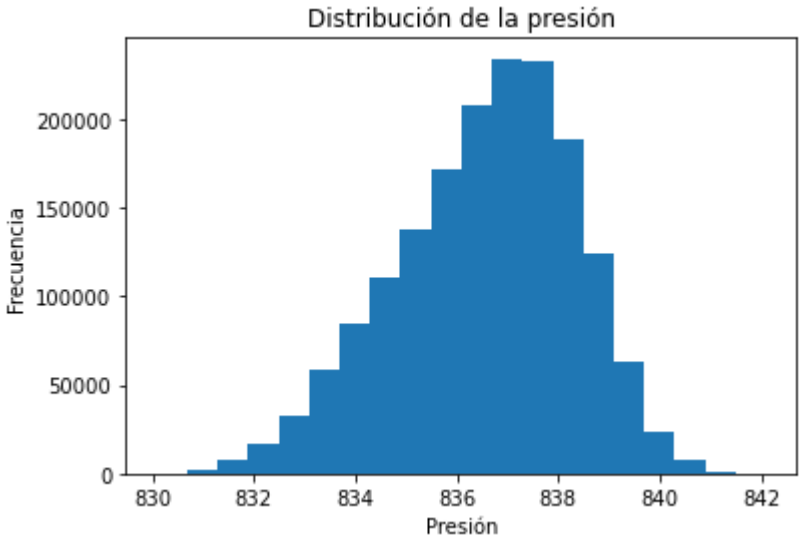


Figura 109. Distribución de frecuencia de la presión, después de la limpieza.

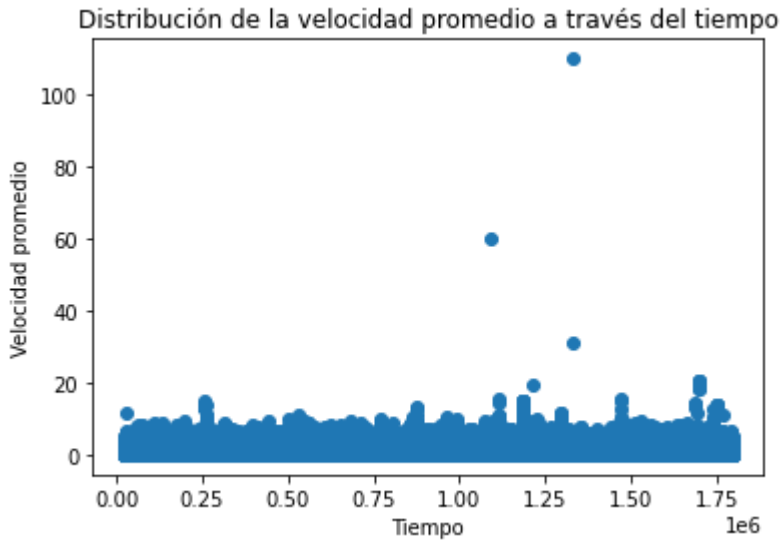


Figura 110. Distribución de la velocidad promedio a través del tiempo, después de la limpieza.

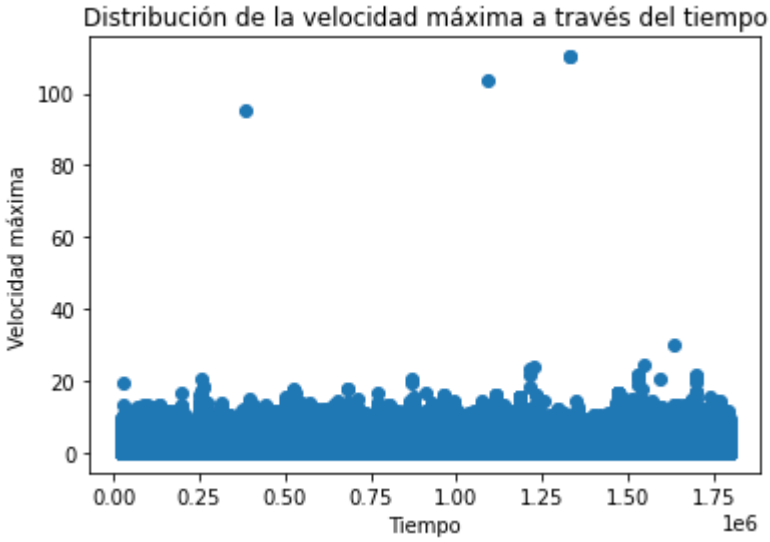


Figura 111. Distribución de la velocidad máxima a través del tiempo, después de la limpieza.

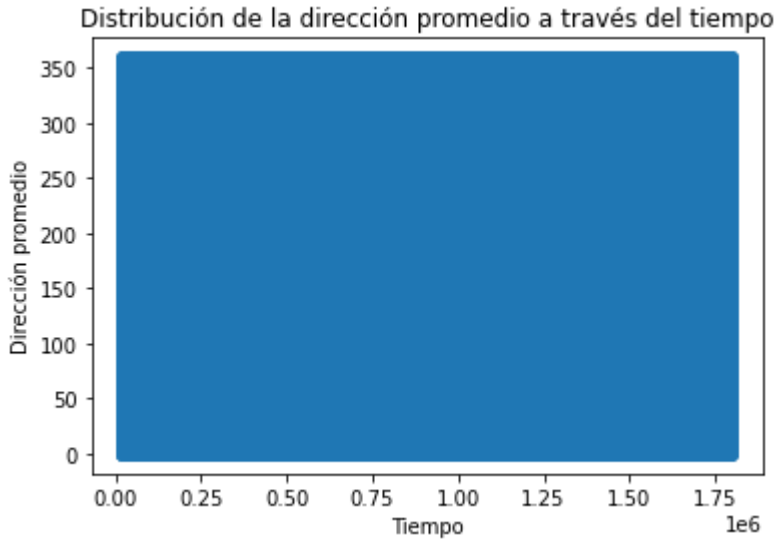


Figura 112. Distribución de la dirección promedio a través del tiempo, después de la limpieza.

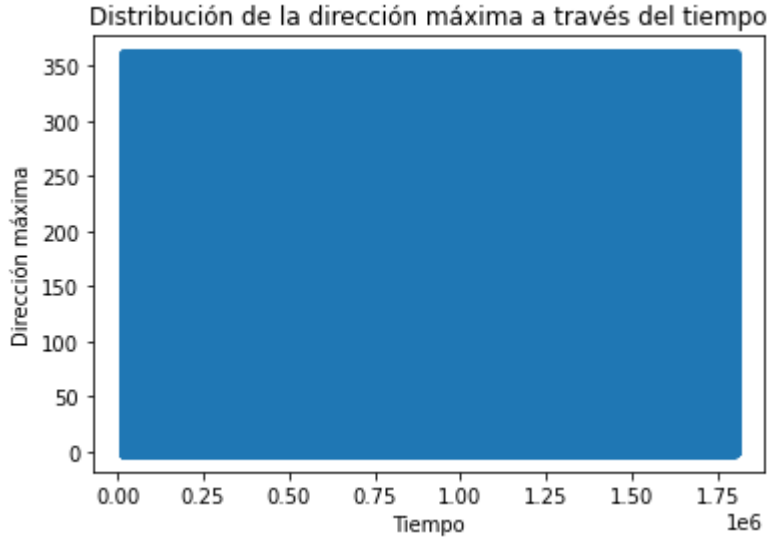


Figura 113. Distribución de la dirección máxima a través del tiempo, después de la limpieza.

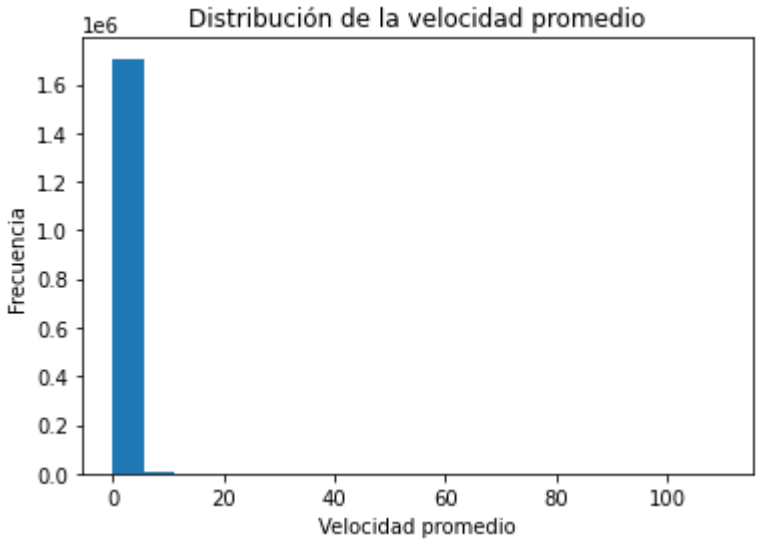


Figura 114. Distribución de frecuencia de la velocidad promedio, después de la limpieza.

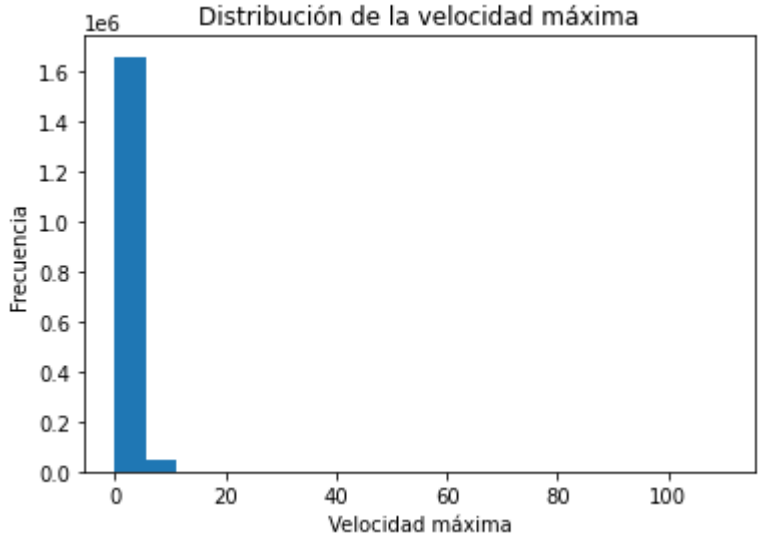


Figura 115. Distribución de frecuencia de la velocidad máxima, después de la limpieza.

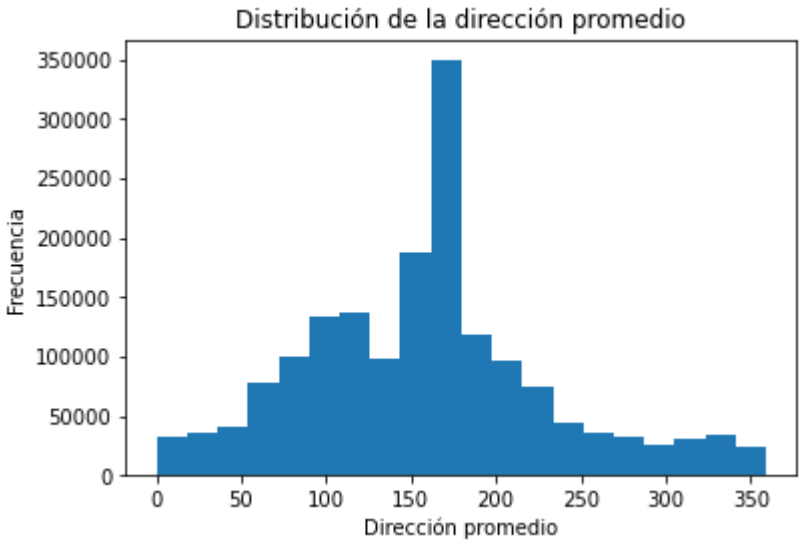


Figura 116. Distribución de frecuencia de la dirección promedio, después de la limpieza.

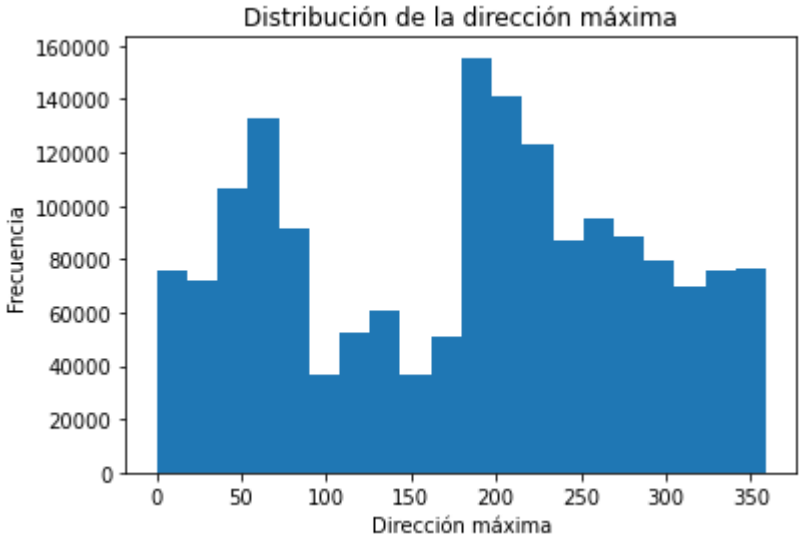


Figura 117. Distribución de frecuencia de la dirección máxima, después de la limpieza.

Análisis: De la **figura 82** a la **figura 99** (datos antes de la limpieza)**,** se observa la formación de grupos atípicos de datos que no tienen coherencia con el resto de la información. Estos datos son considerados erróneos debido a que pueden causar que los resultados obtenidos al implementar las diferentes técnicas de aprendizaje de máquina sean menos precisos. La información errónea debe eliminarse o ser transformada.

Mientras que, en el caso de la **figura 100** a la **figura 117** (datos después de la limpieza), observamos que la información es más consistente, permanece agrupada en un rango de valores normal (como se espera para esta información). Estos datos son considerados correctos y de gran valor para entrenar los algoritmos de aprendizaje de máquina implementados.

5.2. Identificación de las variables meteorológicas de mayor influencia en relación con la contaminación de la atmósfera

En la **figura 118**, se observa un mapa de calor de la correlación entre las diferentes variables. Los cuadrados con mayor intensidad de color representan una mayor relación entre las variables correspondientes. El color rojo indica que existe una correlación positiva entre las variables (cuando una aumenta de valor, la otra también aumenta), mientras que el color azul indica que existe una correlación negativa (cuando una aumenta de valor, la otra disminuye).

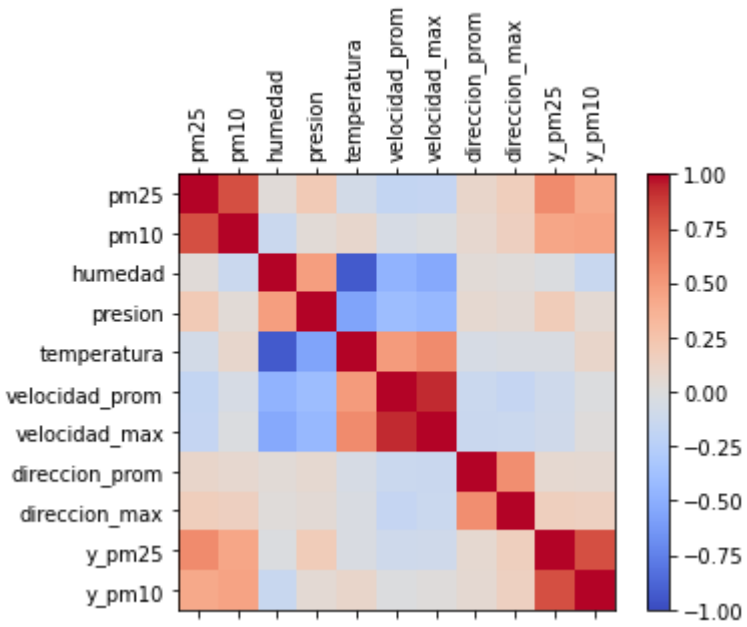


Figura 118. Mapa de calor de la correlación entre las diferentes variables.

Análisis: Según la información obtenida de la matriz de correlación, las variables meteorológicas con mayor influencia en la calidad del aire en orden descendente son:

1. Dirección del viento máximo.
2. Presión atmosférica.
3. Velocidad promedio del viento.
4. Velocidad máxima del viento.
5. Dirección promedio del viento.
6. Humedad.
7. Temperatura.

5.3. Implementación de los algoritmos de aprendizaje de máquina

5.3.1. Red neuronal artificial

De la **tabla 9** a la **tabla 11**, se muestran los valores obtenidos al aplicar los diferentes estimadores de error a los modelos de red neuronal artificial entrenados con diferentes números de variables como se explica en la **sección 4.4**. Esta información se representa gráficamente de la **figura 119** a la **figura 121**.

Número de entradas	MSE
9	0.0117
8	0.0117
7	0.0116
6	0.0115
5	0.0116
4	0.0116
3	0.0116
2	0.0114

Tabla 9. MSE por número de entradas de la red.

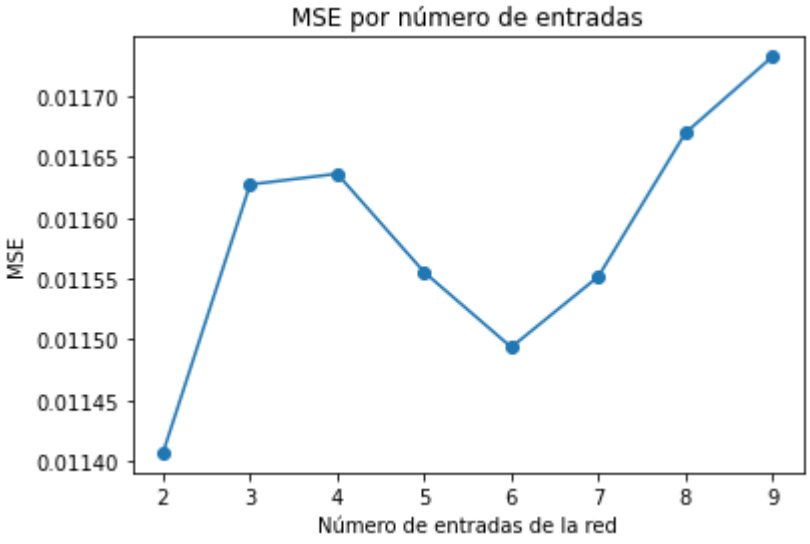


Figura 119. MSE por número de entradas de la red.

Número de entradas	RMSE
9	0.1078
8	0.1085
7	0.1082
6	0.1073
5	0.1078
4	0.1082
3	0.1078
2	0.1074

Tabla 10. RMSE por número de entradas de la red.

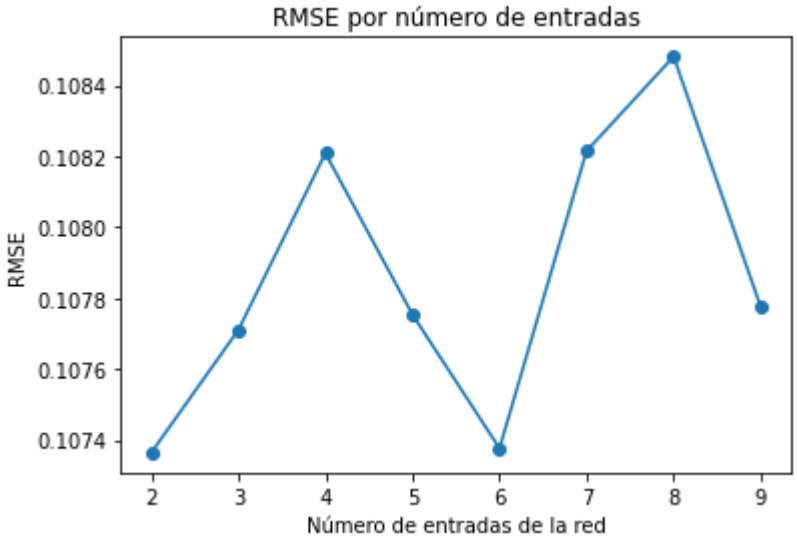


Figura 120. RMSE por número de entradas de la red.

Número de entradas	MAE
9	0.0801
8	0.0793
7	0.0798
6	0.0803
5	0.0797
4	0.0799
3	0.0802
2	0.0796

Tabla 11. MAE por número de entradas de la red.

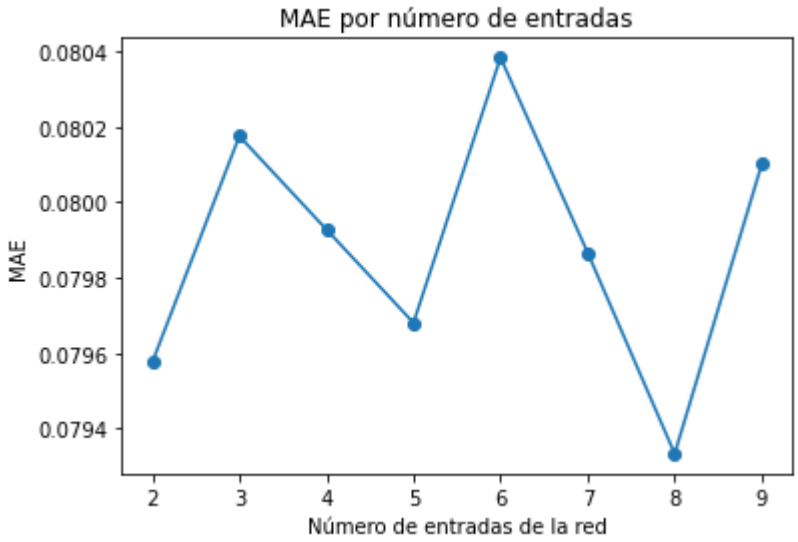


Figura 121. MAE por número de entradas de la red.

Análisis: Se observa como para el caso de la red neuronal artificial, el número de variables representa un cambio insignificante en la medida de los estimadores de error. No es posible observar una relación directa y no se observan mejoras significativas en el desempeño del modelo.

5.3.2. Regresor de bosque aleatorio

De la **tabla 12** a la **tabla 14**, se muestran los valores obtenidos al aplicar los diferentes estimadores de error a los modelos de regresor de bosque aleatorio entrenados con diferentes números de variables como se explica en la **sección 4.4**. Esta información se representa gráficamente de la **figura 122** a la **figura 124**.

Número de entradas	MSE
9	0.0059
8	0.0059
7	0.0062
6	0.0062
5	0.0063
4	0.0066
3	0.0072
2	0.007

Tabla 12. MSE por número de entradas del algoritmo.

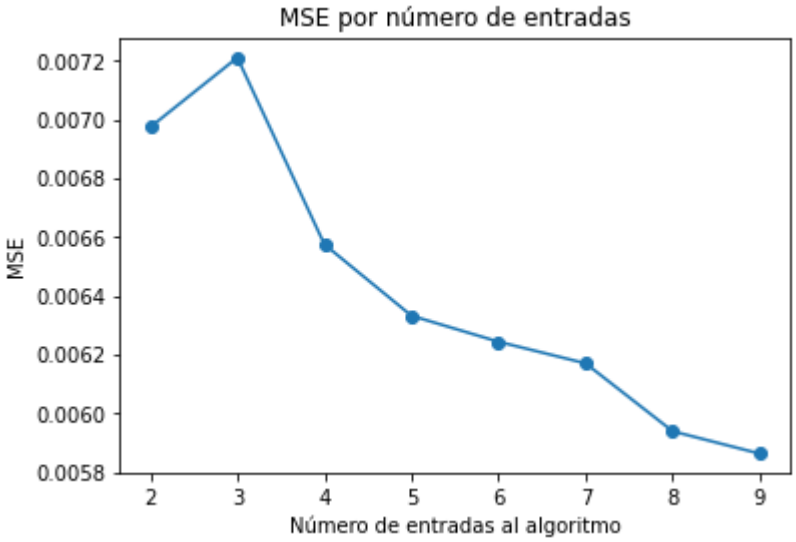


Figura 122. MSE por número de entradas del algoritmo.

Número de entradas	RMSE
9	0.0766
8	0.0771
7	0.0785
6	0.079
5	0.0796
4	0.0811
3	0.0849
2	0.0835

Tabla 13. RMSE por número de entradas del algoritmo.

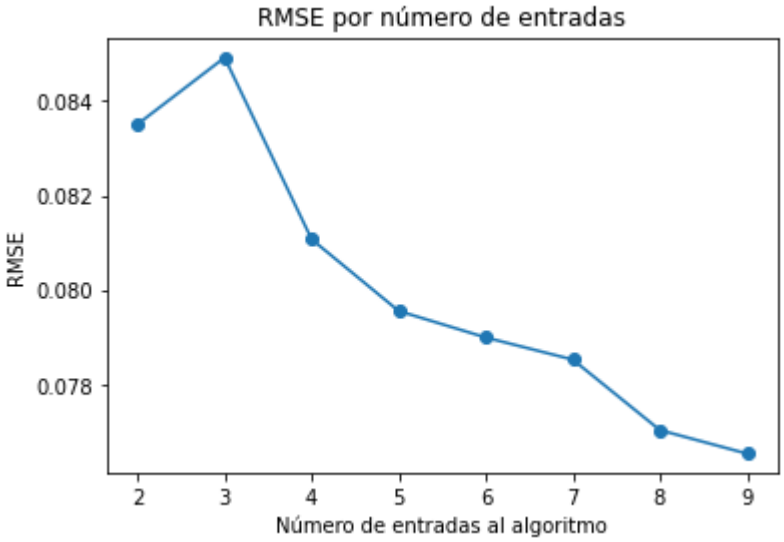


Figura 123. RMSE por número de entradas del algoritmo.

Número de entradas	MAE
9	0.0579
8	0.0584
7	0.0594
6	0.06
5	0.0602
4	0.0615
3	0.0641
2	0.0624

Tabla 14. MAE por número de entradas del algoritmo.

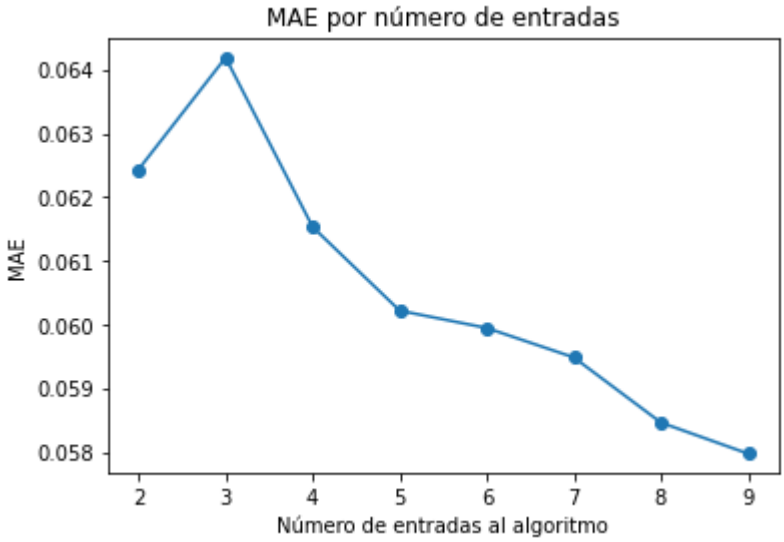


Figura 124. MAE por número de entradas del algoritmo.

Análisis: Se observa como para el caso del regresor de bosque aleatorio, el número de variables representa un cambio más significativo en la medida de los estimadores de error que en el caso de la red neuronal, pero a pesar de esto las diferencias son mínimas. Se observa un mejor desempeño con respecto a la disminución del número de variables de entrada, pero esta mejora no es significativa.

Al analizar los resultados obtenidos con ambas implementaciones se observa un desempeño sobresaliente y un error aceptable. Ambos algoritmos tienen la capacidad de realizar la estimación de los niveles futuros de material particulado en la atmósfera de manera acertada.

Por último, al comparar los resultados obtenidos en la implementación de la red neuronal artificial con los resultados obtenidos en la implementación del regresor de bosque aleatorio, se observa que este último tiene un mejor desempeño (en todos los estimadores de error).

5.4. Estimación de los niveles futuros de material particulado

En la **tabla 15**, se muestran los resultados obtenidos al realizar la estimación de los niveles de material particulado presentes en la atmósfera de Medellín y su área metropolitana, haciendo uso de los diferentes algoritmos de aprendizaje de máquina. Esta estimación es realizada para días específicos elegidos aleatoriamente, donde se conoce la información de entrada, pero también se conoce la información real de los niveles de material particulado. Los resultados obtenidos son comparados con los valores medidos por la red de calidad del aire.

Fecha	RNA		RFR		Medido	
	PM2.5	**PM10**	**PM2.5**	**PM10**	**PM2.5**	**PM10**
2015-09-09 18:00	23.42	81.50	29.28	76.83	21.0	53.0
2015-09-16 12:00	28.71	71.57	26.9	69.04	41.0	78.0
2017-02-06 13:00	42.58	74.66	43.46	90.11	43.0	86.0
2018-05-01 05:00	21.07	38.04	26.26	41.75	28.0	38.0
2018-10-24 00:00	20.05	43.89	14.0	32.42	10.0	23.0
2019-06-29 11:00	29.58	68.12	34.03	70.26	18.0	27.0
2020-06-22 19:00	17.89	39.60	31.74	102.21	45.0	137.0
2020-11-17 23:00	27.12	45.70	24.82	48.78	22.0	43.0

Tabla 15. Comparativa de la estimación realizada con la red neuronal artificial, el regresor de bosque aleatorio y los datos medidos (datos reales).

Análisis: Al realizar la estimación de los niveles de material particulado**,** se observa que, aunque existen valores alejados de los datos reales, en la mayoría de los casos se obtienen resultados cercanos a los resultados medidos. Lo anterior, nos permite poder generar alertas de forma temprana con respecto a la calidad del aire. Además de esto, se observa que el regresor de bosque aleatorio arroja resultados más precisos que la red neuronal artificial, como se observó en la sección anterior.

6. Conclusiones

- Para obtener mejores resultados en la estimación de los niveles futuros de material particulado presentes en la atmosfera de Medellín y su área metropolitana, es necesario que la información recolectada pase por un proceso de limpieza. Lo anterior, con la finalidad de eliminar valores atípicos dentro de los conjuntos de datos que pueden generar cierta incertidumbre o reducir la precisión del sistema.
- Existe una relación, aunque pequeña, entre la medida de las diferentes variables meteorológicas y la contaminación del aire de Medellín y su área metropolitana. Esta relación existe especialmente con la velocidad y dirección del viento, y la presión atmosférica.
- La inteligencia artificial, específicamente el aprendizaje de maquina resulto ser una herramienta poderosa para realizar la estimación de los niveles futuros de Material particulado en la atmosfera de Medellín y su área metropolitana. Aunque los valores obtenidos no son correctos en la totalidad de los casos, en su mayoría son bastante acertados y esto puede ser de gran utilidad para alertar a las autoridades ambientales y población en general cuando se considere que los niveles de material particulado van a ser peligrosos.
- El regresor de bosque aleatorio mostro ser un algoritmo más atinado que la red neuronal artificial, en la tarea de estimar los niveles futuros de material particulado presentes en la atmosfera de Medellín y su área metropolitana.

7. Referencias Bibliográficas

[1] World Health Organization, “Ambient (outdoor) air pollution”, 2021, [En línea]. Disponible en: https://www.who.int/news-room/fact-sheets/detail/ambient-(outdoor)-air-quality-and-health.

[2] José Guillermo Palacio, “El impacto de la polución en la salud”, Periódico El Colombiano, 2019, [En línea]. Disponible en https://www.elcolombiano.com/antioquia/el-impactode-la-polucion-en-la-salud-KF10378893.

[3] Medellín Como Vamos, “Área Metropolitana”, 2022, [En línea]. Disponible en: https://www.medellincomovamos.org/territorio/area-metropolitana-del-valle-de-aburra.

[4] Contraloría General de Medellín (CGM), “Indicadores Ambientales de Medellín”, 2018, [En línea]. Disponible en: http://www.cgm.gov.co/cgm/Paginaweb/IP/Informe%20Ambiental%202018/INDICADORES%20AMBIENTALES%20DE%20MEDELL%C3%8DN%20VIGENCIA%202018.pdf.

[5] C. A. Gómez, “Contaminación del aire de Medellín por pm10 y pm2.5 y sus efectos en la salud”. 2017.

[6] Oxford Languages, “Oxford Languages and Google”, 2021, [En línea]. Disponible en: https://languages.oup.com/google-dictionary-es/.

[7] C. A. Arciniegas, “Diagnóstico y control de material particulado: partículas suspendidas totales y fracción respirable PM10”, Luna Azul, n° 66, pp. 195-213, 2012.

[8] Área Metropolitana del Valle de Aburrá, “¿Qué es el ICA?”, 2019, [En línea]. Disponible en: https://www.metropol.gov.co/ambiental/calidad-del-aire/Paginas/Generalidades/ICA.aspx.

[9] B. López Porrero, “Limpieza de datos: reemplazo de valores ausentes y estandarización”, Tesis Doctoral, Universidad Central “Marta Abreu” de Las Villas, Facultad de Matemática, Física y Computación. Departamento Ciencias de la Computación, 2011.

[10] A. L. Samuel, “Some studies in machine learning using the game of checkers. II—Recent progress”, IBM Journal of research and development, vol. 11, n° 6, pp. 601-617, 1967.

[11] E. Alpaydin, “Introduction to Machine Learning”, 3 ed., 2014.

[12] T. M. Mitchell, “Machine Learning”, McGraw-Hill, New York, 1997, pp. 154-200.

[13] G. Bonaccorso, “Machine learning algorithms”, Packt Publishing Ltd, 2017.

[14] F. Carmona, “Modelos lineales”, Pub. Univ. de Barcelona, Barcelona, 2005.

[15] R. F. López, J. M. Fernández, ¨Las redes neuronales artificiales¨. Netbiblo, 2008.

[16] P. Larranaga, I. Inza, A. Moujahid, ¨Tema 8. redes neuronales¨. Redes Neuronales, U. del P. Vasco, vol. 12, pp. 17, 1997.

[17] F. D. Meneses Bautista, M. Alvarado, ¨Pronóstico del tipo de cambio USD/MXN con redes neuronales de retropropagación¨, Res. Comput. Sci., vol. 139, pp. 97-110., 2017.

[18] A. Fernández. ¨Python 3 al descubierto¨. Alfaomega Grupo Editor, 2013.

[19] Pandas, "Pandas Documentation", 2022, [En línea]. Disponible en: https://pandas.pydata.org/docs/.

[20] NumPy, "Numpy Documentation", 2022, [En línea]. Disponible en: https://numpy.org/doc/stable/.

[21] Matplotlib, "Matplotlib API Reference", 2022, [En línea]. Disponible en: https://matplotlib.org/stable/api/index.

[22] Scikit Lean, "ScikitLearn API Reference", 2022, [En línea]. Disponible en: https://scikit-learn.org/stable/modules/classes.html.

[23] TensorFlow, "TensorFlow Core v2.8.0", 2022, [En línea]. Disponible en: https://www.tensorflow.org/api_docs/python/tf?hl=es-419.

[24] X. Feng, Q. Li, Y. Zhu, J. Hou, L. Jin y J. Wang, "Artificial neural networks forecasting of PM2. 5 pollution using air mass trajectory based geographic model and wavelet transformation", Atmospheric Environment, vol. 107, pp. 118-128, 2015.

[25] Y. C. Lin, S. J. Lee, C. S. Ouyang y C. H. Wu. "Air quality prediction by neuro-fuzzy modeling approach". Applied soft computing, vol. 86, pp. 105898, 2020.

[26] J. Xie. "Deep neural network for PM2. 5 pollution forecasting based on manifold learning", 2017 international conference on sensing, diagnostics, prognostics, and control (SDPC). IEEE, pp. 236-240, 2017.

[27] T. Sepúlveda, O. Nicolis y B. Peralta, ¨Predictions of PM2.5 concentrations and critical events in Santiago, Chile using Recurrent Neural Networks¨. 2019 IEEE CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON). IEEE, pp. 1-7, 2019.

[28] Sistema de Alerta Temprana de Medellín y el Valle de Aburrá, "¿Quiénes somos?", 2022, [En línea]. Disponible en: https://siata.gov.co/sitio_web/index.php/quienesSomos.

[29] Sistema de Alerta Temprana de Medellín y el Valle de Aburrá, “Redes”, 2022, [En línea]. Disponible en: https://siata.gov.co/sitio_web/index.php/monitoreo#calidadAire.

[30] Sistema de Alerta Temprana de Medellín y el Valle de Aburrá, “SIATA”, 2022, [En línea]. Disponible en: https://siata.gov.co/siata_nuevo/.

[31] Sistema de Alerta Temprana de Medellín y el Valle de Aburrá, “Generalidades Información de Estaciones de Calidad del Aire”, 2022, [En línea]. Disponible en: https://siata.gov.co/descarga_siata/index.php/info/aire/.

[32] Sistema de Alerta Temprana de Medellín y el Valle de Aburrá, “Generalidades Información de Estaciones de Pluviograficas y Meteorológicas”, 2022, [En línea]. Disponible en: https://siata.gov.co/descarga_siata/index.php/info/pluviomet/.

[33] Sistema de Alerta Temprana de Medellín y el Valle de Aburrá, “Información de Calidad del Aire”, 2022, [En línea]. Disponible en: https://siata.gov.co/descarga_siata/index.php/index2/calidad_aire/.

[34] Sistema de Alerta Temprana de Medellín y el Valle de Aburrá. “Información de estaciones Meteorológicas”. 2022. Disponible en: https://siata.gov.co/descarga_siata/index.php/index2/estaciones/. Acceso en marzo del 2022.

[35] Analytics Vidhya, “Decision Tree vs. Random Forest – Which Algorithm Should you Use?”, 2020, [En línea]. Disponible en: https://www.analyticsvidhya.com/blog/2020/05/decision-tree-vs-random-forest-algorithm/.

[36] United States Environmental Protection Agency, “Particulate Matter (PM) Basics”, 2021, [En línea]. Disponible en: https://www.epa.gov/pm-pollution/particulate-matter-pm-basics.