Universitat Oberta de Catalunya (UOC)

Màster Universitari en Ciència de Dades (*Data Science*)

TREBALL FINAL DE MASTER

Àrea: 5

Detecció i predicció d'anomalies en dispositius IoT en l'Edge computing

---

Autor: Antoni Llussà Sala

Tutor: Sergio Trilles Oliver

Professor: Albert Solé Ribalta

Taradell, 3 de gener de 2021

Crèdits/Copyright

FITXA DEL TREBALL FINAL

Títol del treball:	Detecció i predicció d'anomalies en dispositius IoT en l'Edge computing
Nom de l'autor:	Antoni Llussà Sala
Nom del col·laborador/a docent:	Sergio Trilles Oliver
Nombre del PRA:	Albert Solé Ribalta
Data de entrega (mm/aaaa):	01/2021
Titulació o programa:	Màster Universitari en Ciència de Dades (Data Science)
Àrea del treball final:	5
Idioma del treball:	Català
Paraules clau:	Machine Learning, predicció d'anomalies, IoT

Cita

Amb la recopilació de dades,

'com més aviat millor' és sempre la millor opció.

Marissa Mayer

Agraïments

A la Núria, al Toni, a la Jana, al Pau i a la Berta. Gràcies per ser-hi i donar-me suport durant el transcurs d'aquest Màster Universitari en Ciència de Dades.

Abstract

Nowadays, Internet of Things (IoT) devices can run machine learning (ML) models. Taking advantage of the computational power of these devices, the incorporation of a ML model to detect and predict anomalies in the data (time series) is intended. Data is collected real time by sensors connected to the device.

Predicting and detecting anomalies within the IoT device can provide benefits such as reducing the sending of erroneous data to server, and so that saving on transmission as well as on the processing of these data in the cloud, and to make filtering of erroneous data.

The field of the project is the environmental and focuses on measuring the air quality. Sensors will measure air particles and IoT devices will be managed through Particle platform (https://www.particle.io/) . Two types of sensors will be used, Particulate Matter Sensor SPS30 [20] and Grove - Laser PM2.5 Dust Sensor [9], and several particle sizes will be measured.

This work aims to develop an ML model for the detection and prediction of anomalous data captured by sensors connected to IoT devices, and run it within the IoT devices.

Keyworks: Machine Learning, anomalies detection, anomalies prediction, Internet of things.

Resum

En l'actualitat, els dispositius de la Internet de les coses (IoT), tenen la capacitat per de poder executar models d'aprenentatge automàtic (ML). Aprofitant aquest potencial, es pretén incorporar en un dispositiu IoT un model de ML per detectar i predir anomalies en les dades (series temporals) que capturen, en temps real, els sensors connectats al dispositiu.

Detectar i predir dades anòmales dins del dispositiu pot aportar avantatges com la reducció d'enviament de les dades errònies i així aconseguir un estalvi en la seva transmissió i també en el posterior processament d'aquestes dades en el núvol, així com poder fer un filtratge de les dades errònies.

L'àmbit del treball és l'ambiental, en aquest cas, per mesurar la qualitat de l'aire. Els sensors mesuraran les partícules de l'aire. El dispositiu IoT, es gestionarà mitjançant la plataforma https://www.particle.io/, hi haurà disponibles dos tipus de sensors per mesurar diversos diàmetres de partícules de l'aire. Els sensors seran: Particulate Matter Sensor SPS30 [20] i Laser PM2.5 Dust Sensor [9].

Aquest treball pretén aconseguir desenvolupar un model de ML per la detecció i predicció de dades anòmales capturades pels sensors connectats al dispositius IoT, i executar-lo dins del dispositius IoT.

Paraules clau: Aprenentatge automàtic, Detecció d'anomalies, Predicció d'anomalies, Internet de les coses

Índex

Índex de figures

Índex de taules

Capítol 1

Introducció

1.1 Context i justificació del treball

L'elecció del treball, és degut que actualment en el meu lloc de treball, estic treballant en projectes de l'àmbit de la Internet de les coses (IoT), també penso que aquest treball, destaca el seu caràcter innovador i prometedor dins el sector de la IoT, que és la detecció i predicció d'anomalies en les dades que generen els dispositius. Aquest tipus de solució crec que es pot aplicar en diferents àmbits de la interconnectivitat de les coses, dins els camps de la Industria 4.0, Smart Cities i l'Smart Home entre altres.

La Internet de les coses, fa referència a la interconnexió digital dels objectes a Internet, on permet tenir un control integral sobre l'estat de l'objecte. La IoT, aporta una capacitat transformadora, on dins aquest treball està vinculada amb tecnologies com: la sensòrica, el big data i el núvol.[13, 12]

En aquest treball, es vol realitzar una detecció i predicció d'anomalies de les dades, sobre les dades que es recullen en els dispositius IoT dels diferents sensors que puguin tenir, aquestes dades seran series temporals. En aquest cas serà en l'àmbit de la monitorització ambiental, es disposarà de varis sensors per mesurar la qualitat de l'aire com el particulat de l'aire. Es disposaran de diferents sensors per mesurar diferents diàmetres de les partícules. Aquesta detecció i predicció es realitzarà en l'edge computing i en temps real, aprofitant el potencial de càlcul d'aquests dispositius. Aquest aspecte, és rellevant, perquè ens aportarà varis avantatges, com:

- Estalviar l'enviament de les dades, quan aquestes siguin errònies en el núvol. Beneficiarà als equips que no disposin d'una connexió WIFI cap a Internet, ens podrà aportar una reducció de costos en l'enviament de les dades.

- Estalvi de càlculs en el núvol. Moltes vegades es disposen de recursos en el nuvol, que estan hostatjats a servidors com podria ser Google Cloud [8], Amazon Web Services [2], Microsoft Azure [16],etc., que ofereixen serveis per realitzar processos en les dades. Aquests serveis acostumen a tenir un cost que pot arribar a ser elevat. Ens podrà aportar una reducció de costos en els càlculs/processos que es realitzarien amb dades erronies.

- Filtratge de les dades. Quan els sensors fallen, es podrien corregir les dades amb dades aproximades i així no hi haurien dades absents i/o incorrectes.

El desenvolupament d'un model d'aprenentatge automàtic, és una opció molt interessant per poder crear un sistema de detecció i predicció d'anomalies de les dades recollides en el propi dispositiu IoT, i poder aconseguir els avantatges comentats.

1.2 Objectius del treball

L'objectiu principal del treball és poder desenvolupar un model per la detecció i predicció d'anomalies de les dades que recullen els dispositius IoT en l'edge computing, en temps real. Les dades a tractar seran de l'àmbit ambiental, sobre la qualitat de l'aire, hi hauran sensors de partícules de l'aire que mesuraran diferents diàmetres de les partícules. Per aconseguir l'objectiu, s'hauran d'establir un conjunt d'objectius relacionats:

- Analitzar l'estat de l'art de treballs relacionats de Machine Learning, que tinguin models per a la detecció i predicció d'anomalies en les dades.

- Implementar, entrenar, testejar i verificar els models obtinguts amb les dades que hi ha recopilades.

- Desenvolupar el model òptim pel microcontrolador.

- Testejar el model en temps real dins del dispositiu.

- Documentar i justificar el resultat del treball dins la memòria del treball final de Màster.

1.3 Enfocament i mètode seguit

Per la realització del projecte, s'utilitzarà la metodologia CRISP-DM (Cross Industry Standard Process for Data Mining).

Aquesta metodologia penso que es adequada perquè busca un compromís per executar un projecte amb qualitat. Per poder-ho aconseguir, és important destacar la iteració i revisió de cada una de les fases i processos. També s'estableixen cicles petits de planificació, execució i revisió, no s'avancen els cicles fins que es donin com a correctes.

Consta de sis fases:

1. **Comprensió del negoci:** En aquesta primera fase s'ha de ser capaç de comprendre els objectius del projecte, avaluar la situació actual del negoci respecte els objectius plantejats, fixar els objectius a nivell de mineria de dades i obtenir un pla de projecte en que es detallarà les fases, tasques i activitats que ens durant a aconseguir els objectius plantejats.

2. **Comprensió de les dades:** La segona fase es tracta d'una fase critica degut a que es treballa amb la qualitat de les dades, s'ha d'aprofundir en el seu coneixement: En quines condicions arriben, quina es la seva estructura, quines propietats tenen, com es poden tractar o eliminar els inconvenients que presenten.

3. **Preparació de les dades:** En la tercera fase l'objectiu és disposar del joc de dades final sobre el qual s'aplicaran els models. S'haurà d'establir amb quines dades es treballaran, netejar de dades, construir un model apte per la realització del model, integrar dades externes si fos necessari. Llavors s'ha de desenvolupar la documentació descriptiva necessària sobre el joc de dades.

4. **Modelat:** L'objectiu de la quarta fase és obtenir un model que ens ajudi a aconseguir els objectius que s'han plantejat de la mineria de dades i del negoci.

5. **Avaluació del model:** En la cinquena fase es tracta d'avaluar el grau d'encert dels objectius de negoci i en la cerca, si n'hi ha, de les raons del perquè el model ha estat insuficient.

6. **Desplegament:** En la sisena fase, es tracta de dissenyar un pla de desplegament dels models i coneixements sobre la nostra organització. Realitzar el seguiment i manteniment de la part més operativa del desplegament i revisar el projecte amb la seva globalitat per identificar les lliçons apreses.

La seqüència de les fases no és estricta. Les fases del projecte avancen i retrocedeixen entre fases, si és necessari. Tal com es mostra en la figura 1.1

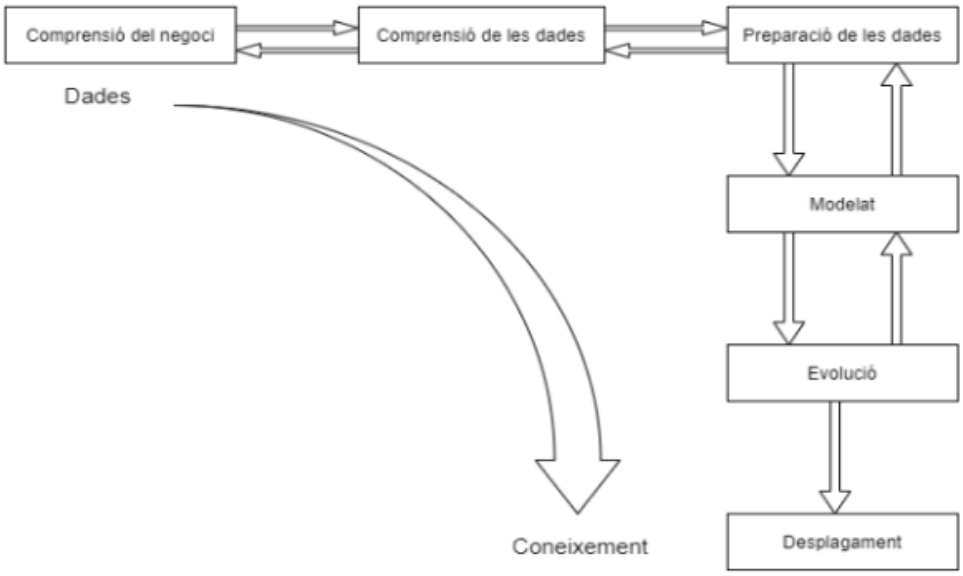


Figura 1.1: Fases de la metodologia CRISP-DM.

1.4 Planificació del treball

A continuació es mostra el resum de la planificació per la realització del treball:

Fase	Data de finalització	Descripció
1	27/09/2020	Definició i planificació del treball final
2	18/10/2020	Estat de l'art o anàlisi del mercat
3	20/12/2020	Disseny i implementació del treball
4	03/01/2021	Redacció de la memòria
5	10/01/2021	Presentació i defensa del projecte

Taula 1.1: *Dates clau.*

1.4.1 Fase 1

Les tasques que es duran a terme en aquesta fase són:

- Definició de la proposta.
- Definició de la metodologia que es durà a terme el treball.
- Planificació del treball.

1.4.2 Fase 2

En aquesta fase, es durà a terme un anàlisi de l'estat de l'art. On hi haurà les següents tasques:

- Cercar bibliografia relacionada.
- Justificar amb evidències científiques el treball.
- Refinar els objectius parcials si s'escau.
- Identificar la metodologia i les tècniques de generació de dades que s'utilitzaran.

1.4.3 Fase 3

En aquesta fase, es durà a terme el disseny i implementació del treball. On hi haurà les següents tasques:

- Preparació de les dades.
- Desenvolupar els models escollits.
- Avaluació dels diferents models.
- Desenvolupar el model òptim pel microcontrolador.
- Testejar el model en temps real dins del dispositiu.

1.4.4 Fase 4

En aquesta fase, es durà a terme la finalització del redactat de la memòria. On hi haurà les següents tasques:

- Revisar el contingut de tota la memòria, i si cal completar-ne les parts restants.
- Documentar i justificar el desenvolupament.
- Documentar i justificar el resultat del treball.

1.4.5 Fase 5

En aquesta darrera fase, es durà a terme la presentació i defensa del projecte. On hi haurà les següents tasques:

- Crear el document amb les diapositives de la presentació.
- Realitzar la defensa del treball final de màster.
- Resolució de les preguntes formulades pel tribunal.

Capítol 2

Estat de l'art

En aquest apartat es repassaran els antecedents i articles d'investigació vinculats a l'objecte del present treball: Detecció i predicció d'anomalies en dispositius IoT en l'edge computing.

La dificultat principal a la hora d'encarar aquesta recerca és la seva particularitat de on s'hauran d'executar els models de Machine Learning per poder realitzar la detecció i predicció de les anomalies, que és dins de l'edge computing. Això implica que els models generats hauran de ser executats dins del mateix dispositiu IoT, tenint en compte que té uns recursos limitats i on s'haurà d'utilitzar una versió del "TensorFlow Lite"[25] per poder executar els algorismes d'aprenentatges automàtic.

En l'apartat 2.2 es poden veure els treballs més destacats que s'han localitzat referent a la detecció d'anomalies en l'edge computing.

2.1 Antecedents

2.1.1 Aprenentatge Automàtic (Machine Learning)

L'aprenentatge automàtic [34] és una branca de la informàtica amb l'objectiu de permetre als computadors aprendre un nou comportament basat en dades empíriques. L'objectiu és dissenyar algorismes que permetin a la computadora mostrar el comportament après de l'experiència passada, enlloc de la interpretació humana.

L'aprenentatge automàtic és essencial pel desenvolupament de la intel·ligència artificial, però també es aplicable a moltes tasques informàtiques quotidianes.

L'aprenentatge automàtic es centra en aprendre de les dades amb l'objectiu de realitzar prediccions sobre altres dades en el futur.

Els algorismes d'aprenentatge automàtic es poden dividir en funció de com entrena la màquina:

- Aprenentatge supervisat: La màquina aprèn de les entrades que s'assignen a les sortides desitjades.
- Aprenentatge no supervisat: La màquina analitza l'entrada sense coneixement de la sortida desitjada.
- Aprenentatge semi-supervisat: Parts de les entrades s'emparellen amb una sortida desitjada i altres parts no.
- Transducció: La màquina intenta predir noves sortides basades en l'entrenament amb entrades i sortides anteriors.
- Reforç d'aprenentatges: La màquina ha de formar una política sobre com ha d'actuar sobre la base de l'observació de com determinades accions afecten al seu entorn.
- Aprendre a aprendre: Ensenya el biaix inductiu en l'experiència prèvia.

2.1.2 Anomalies en les series temporals

La detecció d'anomalies [33] serveix per detectar patrons en el qual el seu comportament es considera anormal en comparació amb els altres punts normals.

Hi ha diferents fonts d'anomalies: Detecció d'intrusos, detecció de fraus i la fuga de dades. La detecció d'anomalies s'utilitza en varies àrees: Smart Cities (fuites d'aigua, electricitat), seguretat en la Xarxa (detecció d'intrusions, de fraus), industries (inspecció superficial del dispositiu).

Hi ha tres categories principals d'anomalies en les series temporals:

1. Valors atípics globals: També conegudes com anomalies puntuals, aquests valors atípics existeixen més allà de la totalitat d'un conjunt de dades

2. Valors atípics contextuals: També conegudes com valors atípics condicionals, aquestes anomalies tenen valors que es desvien significativament dels altres punts de dades que existeixen en el mateix context. Una anomalia en el context d'un conjunt de dades pot no ser una anomalia en un altre context. Aquests valors atípics són comuns en les dades de series temporals perquè aquests conjunts de dades son registres de quantitats específiques en un període determinat. El valor existeix dins de les expectatives globals, però pot semblar anòmal dins de certs patrons de dades estacionals.

3. Valors atípics col·lectius: Sorgeix quan un subconjunt de punts de dades dins d'un conjunt és anòmal per tot el conjunt de dades. En aquesta categoria, els valors individuals no són anòmals ni de forma global ni contextual. Es comencen a veure quan al examinar diferents series temporals juntes. El comportament individual no pot desviar-se del rang normal en un conjunt de dades de series temporals especifiques. Però les anomalies més significatives tornen a ser clares quan es combina amb un altre conjunt de dades de series temporals.

Per abordar aquests tipus de problemes i detectar el comportament anòmal existeixen tres aproximacions:

1. Supervisat: S'assumeix que es disposa d'un conjunt de dades d'entrenament que està etiquetat tant els casos normals com anòmals. De manera que el model pot aprendre a classificar i utilitzar-o per nous casos.

2. Semi-supervisat: S'assumeix que únicament es tenen dades d'entrenament etiquetats per la classe normal i no per les anomalies. Amb aquesta informació pot descriure's un model per classificar els casos normals i si no ho compleix són anomalies.

3. No supervisat: S'assumeix que les dades normals son les més freqüents i per tant els menys freqüents són les dades anòmales. Aquest tipus de models sofreixen molts falsos positius si no és cert.

Problemes i desafiaments clau en la detecció d'anomalies:

- Falta de punts: Degut a la pèrdua de dades de l'entorn extrem, són difícils de detectar.
- Corrupció en les dades: Els factors externs o el mal funcionament del dispositiu corrompen les dades i dificulten la diferenciació de dades anòmales i corruptes.
- Dades encriptades: Detectar anomalies en dades encriptades és difícil.
- Fusió de sensors: És difícil recopilar dades de diferents sensors i després agregar-les per obtenir els resultats.
- Detecció en temps real: La detecció d'anomalies en temps real implica una transmissió de dades a alta velocitat i, per tant, requereix una resposta ràpida.
- Dades sorolloses: La transmissió electrònica genera dades amb soroll, per tant, és necessari eliminar-les dels dispositius informàtics de l'edge computing abans d'enviar-ho al núvol.
- Augment del tràfic: Una gran quantitat de dades pot sobrecarregar la tasca de detecció d'anomalies.
- Dades multi-variats: També s'ha de considerar els canvis freqüents en les dades.

Realitzar el ML en el dispositiu edge computing pot ajudar a millorar:

- Latència: No hi ha anada i tornada en un servidor.
- Privacitat: No es necessari que les dades surtin del dispositiu.
- Connectivitat: No es requereix una connexió a Internet.
- Consum d'energia: les connexions de xarxa consumeixen molta energia.

2.1.3 TensorFlow Lite

TensorFlow Lite és un conjunt d'eines per ajudar als desenvolupadors a executar models de TensorFlow en dispositius mòbils, integrats i de IoT. Permet la inferència d'aprenentatge automàtic (ML) en el dispositiu. Està dissenyat per executar-lo dins de l'edge computing, i així no haver d'enviar les dades cap a altres servidors.

2.1.4 Arquitectura edge computing en la Internet de les coses (IoT)

L'edge computing [6] és part d'una tecnologia de computació distribuïda on el processament de la informació s'ubica aprop de la vora, a on les coses i les persones produeixen o consumeixen aquesta informació.

La Internet de les coses està generant grans quantitats de dades difícils de gestionar per les infraestructures del núvol actuals. L'edge computing [28]ha sorgit com a un nou paradigma per superar les grans quantitats de dades que genera l'IoT, portant la computació cap a la vora de la xarxa, i reduint així la latència de les comunicacions al núvol i alliberant de les xarxes el coll d'ampolla que es derivaria d'aquest ample de banda. No obstant això, les infraestructures del núvol, no desapareixerien de les aplicacions IoT, més aviat els seus actius com ara l'alta disponibilitat i la gran capacitat pel processament i l'emmagatzematge complementaran aquestes aplicacions.

Això té una avantatge molt important, permet analitzar les dades importants casi a temps real, això pot esser una necessitat en moltes industries com la fabricació, la salud, les telecomunicacions o la industria financera.[7]

2.2 Treballs relacionats

A Heterogeneous IoT Data Analysis Framework with Collaboration of Edge-Cloud Computing: Focusing on Indoor PM10 and PM2.5 Status Prediction [29] presenta un marc col·laboratiu molt interessant a tenir en compte. S'utilitza un mètode per seleccionar el millor model per l'edge computing a partir dels models que hi ha al núvol basat en la correlació de les dades de mostra. Aquest mètode permet que l'edge computing usi el model més adequat sense cap tasca d'entrenament a dins l'edge computing i també minimitza problemes de privacitat de les dades. Utilitzen el mètode per predir la concentració futura de les partícules fines de l'aire, PM 10 i PM 2.5, en un espai tancat.

La plataforma del núvol crea models de xarxes LSTM per la predicció, mitjançant conjunt de dades de 29 espais diferents. Llavors es selecciona el millor model de predicció i es trasllada

en l'edge Computing per executar-lo en temps real.

El grau mitjà de precisió mínim en els diferents escenaris en PM10 és de 0.94. i en PM2.5 és de 0.92.

Anomaly detection on the edge [32] presenta un enfoc per la detecció d'anomalies de forma distribuïda, que utilitza auto-codificadors, xarxes neuronals d'aprenentatge profund especialitzades implementades en cada dispositiu de l'edge computing, per realitzar anàlisis i identificar observacions anòmales. Simultàniament, els auto-codificadors aprenen de les noves observacions per identificar noves tendències. Un servidor centralitzat agrega els models actualitzats i els distribueix de nou cap als dispositius perifèrics quan hi ha connexió disponible.

Aquesta arquitectura redueix els requisits de l'ampla de banda i la connectivitat entre els dispositius de l'edge computing i el servidor central. El model d'auto-codificadors i les observacions anòmales s'envien cap els servidors centrals en comptes de les dades observades.

Es presenten quatre experiments amb un conjunt de dades de 46462 punts amb 9 atributs. Amb un 1.89% de punts de dades anòmales. Un experiment (KDD) amb 620098 punts amb 9 atributs. Amb una mida de 1.2MB. Amb una mida de 75MB. Amb un 0.17% de dades anòmales.

Els auto-codificadors utilitzats en aquests experiments han estat compostos per cinc pares ocultes amb 64,32,16,32 i 64 neurones amb funció d'activació de ReLU respectivament. Les capes d'entrada i sortida tenen un número de neurones iguals al número d'atributs del conjunt de dades. Els pesos dels models entrenats ocupen 72KB de memòria pel conjunt de dades dels quatre experiments primers i 102KB per conjunt de dades del cinquè experiment KDD.

Els resultats dels experiments, mostren que un enfoc distribuït per la detecció d'anomalies mitjançant auto-codificadors pot produir resultats similars a un model no distribuït. La distribució permet que la majoria de càlculs es realitzin en paral·lel en els dispositius de l'edge computing i pot reduir la quantitat de dades que es transfereixen en el servidor central. I això fa que sigui interessant utilitzar-lo quan hi hagi situacions en que la connectivitat sigui limitada o inconsistent.

Hierarchical edge computing: A novel multi-source multi-dimensional data anomaly detection scheme for industrial Internet of Things [31] presenta un nou esquema de detecció d'anomalies en les dades multi-dimensionals de múltiples fonts basat en un model de computació d'edge computing. En primer lloc, es proposa un model de computació basat en l'edge computing per realitzar un equilibri de carga i el processament de dades de baixa latència en l'extrem del sensor i l'extrem de l'estació base. Després, es dissenya un algoritme de detecció d'anomalies de dades d'una font única basat en la teoria difusa, que pot analitzar de manera integral els resultats de detecció d'anomalies de dades en múltiples moments consecutius. Finalment, un algorisme de detecció d'anomalies de dades en múltiples fonts executat en l'extrem de l'edge computing per considerar els atributs de temps i espai associats a les dades de detecció. Els resultats experimentals revelen que l'esquema proposat té una major precisió de detecció i un menor retard de processament en comparació en solucions tradicionals.

L'objectiu de la detecció d'anomalies de dades d'una sola font és analitzar les dades d'un sensor. Aquest tipus de detecció només es base en el rang de valors normals d'un cert tipus de sensor per detectar anomalies, i les pot detectar en les dades d'un sensor en un moment determinat.

L'objectiu de la detecció d'anomalies de dades de múltiples fonts és analitzar les dades de múltiples sensors del mateix tipus de sensor. Així es pot analitzar les dades de varis sensors en diferents ubicacions. Així la dimensió espacial es suma a la dimensió temporal de l'equació. Així la detecció d'errors ha de determinar la ubicació del múltiples sensors i obtenir el conjunt de nodes que està més aprop d'un sensor segons la seva ubicació.

Els experiments han estat realitzats en quatre esquemes de detecció d'anomalies: Mètode tradicional de detecció d'anomalies (TADM), Mètode de detecció d'anomalies de teoria difusa (FTADM), mètode de detecció d'anomalies de dades d'una font única (SDADM) i el mètode de detecció d'anomalies de dades de múltiples fonts (MDADM).

Es pot observar que en els experiments la millor precisió és del mètode MDADM, però també és la que té el temps més alt d'execució.

Adaptive anomaly detection for IoT data in hierarchical edge computing [30] presenta una possible solució per poder emprar models complexes, com ara models d'aprenentatge profund, degut al potencial que puguin tenir alguns dispositius d'edge computing. La solució que es presenta, és transformar un model complex i gran en un que s'adapti a la capacitat del dispositiu IoT; per exemple, la compressió del model (Han, Mao iDally 2016) ho aconsegueix podant paràmetres redundants i poc importants (quasi zero), així com quantificant els pesos en contenidors. Tanmateix, aquest enfocament ha de gestionar els models de detecció d'anomalies cas per cas mitjançant una posada a punt, i només s'aplica a alguns tipus específics de xarxes neuronals profundes (DNN) amb una gran dispersió.

On es proposa un enfocament de detecció d'anomalies que consisteix en dos components:

1. Construir tres models de DNN basats en l'estat de l'art amb una complexitat creixent i associar-los a tres capes corresponents de sistemes HEC.

2. Dissenyar un esquema adaptatiu per seleccionar els models adequats basats en la informació contextual de les dades d'entrada, basant-se en el problema com un problema de bandit contextual que també es representa com un procés de decisió de Markov d'un sol pas.

Aquesta proposta, es testeja en dades IoT del món real i es demostra que supera a altres esquemes de referència. Proporciona el millor compromís aconseguint simultàniament una alta precisió de detecció i un baix retard de detecció.

Mobile Edge Computing-Based Data-Driven Deep Learning Framework for Anomaly Detection [26] presenta una possible solució d'aprenentatge automàtic (DL) per detectar problemes relacionats amb el rendiment de la xarxa (cèl·lules en repòs i tràfic elevat que podria provocar congestió) com anomalies. La investigació aborda els problemes de detecció en el punt de vista de DL i la computació de l'edge computing mòbil (MEC), basada en computació descentralitzada, administració de xarxes i emmagatzematge, en comparació amb l'arquitectura de computació centralitzada en el núvol, recentment ha cridat l'atenció per la seva utilitat potencial en xarxes 5G per impulsar la computació cap a punts d'accés, estacions bases.

El marc es base en que cada servidor monitoritza les activitats de l'usuari de múltiples cel·les i utilitza un xarxa neuronal profunda (DNN) de retroalimentació de capa L alimentada per un conjunt de dades de registres de detalls de trucades reals (CDR) per la detecció d'anomalies. Aconseguint una precisió de 98.8% i uns taxa de falsos positius de 0.44% (FPR).

S'utilitza una tècnica d'optimització avançada coneguda com estimació de moment adaptatiu (ADAM). Es configuren les capes ocultes amb la funció d'activació ReLU.

Squeezed Convolutional Variational AutoEncoder for unsupervised anomaly detection in edge device industrial Internet of Things [27] En aquest estudi es realitzen proves amb dos models per la detecció de valors anòmals, un auto-codificador variacional (CNN) i un auto-codificador variable convolucional comprimit (SCVAE). Opten per aquests models degut a que les xarxes neuronals que funcionen bé per aquests tipus de problemes, requereixen un us intensiu de la memòria i de computació, i els dispositius de l'edge computing podrien no tenir prou recursos.

El rendiment de la CNN-VAE és bo, però degut a la grandària del model i la inferència lenta, no es apropiat pel seu un en dispositius d'edge computing. Per tant es proposa el model SCVAE. Aquest model s'ha testejant amb un conjunt de dades etiquetades i no etiquetades, en els dos conjunts de dades els resultats han estat satisfactoris superant a algorismes convencionals d'aprenentatge automàtic.

A Multitiered Solution for Anomaly Detection in Edge Computing for Smart Meters [36] En aquest estudi tracta de predir si es produiran anomalies en l'us de l'electricitat en les pròximes setmanes. Per poder-ho aconseguir es proposen tècniques de predicció que utilitzen deep Neural Network (DNN), Support Vector Regressión (SVR) i k-Neighbors Neighbors (KNN). Aquests algorismes han estat provat en l'àrea de la Internet de les Coses dins de l'edge computing. Es mesuren el temps d'entrenament, el temps d'inferència. En les proves realitzades en una RaspberryPi, la millor opció ha estat el model DNN que tingui la latència més curta de 1.25ms, un mida d'arxiu persistent de 159kB i 128 passos de temps.

Proposen una solució amb varis nivells per agrupar els membres rellevants, expandir les etiquetes per augmentar les mostres minoritàries sense tenir que agrupar noves mostres de dades sintètiques, serialitzant les dades del mateix clúster per superar la falta de conjunt de

dades d'entrenament que només consten d'un cicle i determinar el tipus correcta de la màquina d'aprenentatge mitjançant la comparació del rendiment de les xarxes neuronals, especialment les LSTM, la SVR i la KNN per la detecció d'anomalies. On el resultat d'aquesta solució és el model d'inferència. Cada model generat per el cloud, és enviat a l'edge computing. Els nous clústers resultants s'entrenen després per produir els pesos del model que es guardaran en forma d'arxius de persistència i serviran perquè l'edge contribueixi en la formació de clústers.

2.3 Conclusions

Tant en els treballs mostrats com la literatura ens indiquen que portar la detecció d'anomalies en l'edge computing ens aporta diferents beneficis, com són: protegir la privacitat de les dades, reduir el coll d'ampolla que pot generar l'enviament de les dades en el Cloud per fer els càlculs i la reducció del cost dels medis d'emmagatzematge.

S'ha pogut observar que s'ha de realitzar una cerca de treballs relacionats, per poder trobar el millor model que s'adapti a l'objectiu del present treball, provant diferents enfocaments. Em pogut veure que moltes solucions utilitzen la compressió de models com en els articles [30] [27], i així facilitar l'execució en l'edge computing. Les xarxes neuronals profundes (DNN) i les xarxes neuronals recurrents simples (LSTM), poden ser una bona opció, però no poden descartar altres models com per exemple els auto-codificador variable convolucional comprimit (SCVAE).

Un punt interessant que hem trobat, han estat en els articles [29] i [32], que presenten un marc col·laboratiu per detectar anomalies de forma distribuïda, cadascun d'ells amb algorismes diferents a tenir en compte. En l'article [32] han demostrat que els resultats de models en entorns distribuïts i no distribuïts són similars, és important comentar-ho degut que en el present treball, no es realitzarà de forma distribuïda. Aquesta opció podria ser una possible millora en un futur, degut a que s'ha pogut veure que treballar de forma distribuïda permet fer els càlculs en paral·lel i reduir la quantitat de dades que es poden transmetre en el servidor central.

Cal comentar també l'article [31] on s'ha comparat la detecció d'anomalies de dades d'origen únic i la detecció d'anomalies de dades de múltiples fonts. Creiem que aquest treball degut a

l'enfoc que es vol donar en el present treball, no acaba d'adaptar-se. Degut a que els resultats obtinguts en l'enfoc en la detecció de dades d'origen únic, on realitza dos passos, primer el mètode de detecció d'anomalies de teoria difusa (FTADM) del sensor i el mètode de detecció d'anomalies de font única de l'estació base (SDADM), amb una sola estació base no s'obtenen gaire bons resultats comparant amb l'algorisme MSADM (la detecció d'anomalies de dades de múltiples fonts.

S'ha de tenir present l'article [26], on aprofiten tècniques modernes d'aprenentatge profund (DL) per millorar i oferir un rendiment òptim:

- Mètodes d'inicialització de pes: La explosió o desaparició del gradient és un problema important que hi ha durant la fase d'entrenament degut a a una inicialització de pes inadequada. La selecció curosa de l'estratègia de la inicialització dels pesos, pot solucionar i millorar el rendiment de DNN.

- Regularització: Les DNN tenen un gran desafiament que és el sobre-ajustament. La disminució de pes, és el tipus més comú de regularització.

- Mètodes d'optimització: Utilitzen ADAM[1] degut a que és un algorisme d'optimització de taxa d'aprenentatge adaptatiu més efectius per entrenar un DNN.

De l'article [36] cal destacar que les xarxes neuronals profundes tenen una latència i una mida del model millor que en les SVR i KNN.

S'ha de tenir present que per realitzar els models d'aprenentatge automàtic, disposem d'un conjunt de dades de 2 mesos aproximadament amb 4989 registres d'informació generada per tres sensors PM1, PM2.5 i PM10.

Capítol 3

Arquitectura

3.1 Components utilitzats

El Dispositiu IoT que s'utilitzarà per la realització del treball, és l'"Argon Wi-Fi Develompment Board"[4] és de la casa Particle [19]. El dispositiu es podrà gestionar mitjançant la plataforma de https://particle.io. Se'n pot veure la imatge a la Figura 3.1.

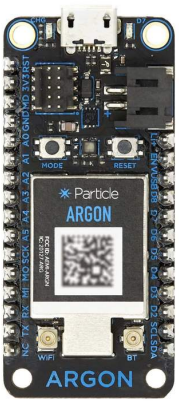


Figura 3.1: Argon Wi-Fi Development Board.

Aquest dispositiu té les següents característiques tècniques:

Procesador principal: Nordic Semiconductor nRF52840 SoC

- Processador ARM Cortex-M4F de 32 bits a 64 MHz
- Flash de 1 MB, RAM de 256 KB

- Suport central y perifèric Bluetooth LE (BLE)
- 20 GPIO de senyal mixta (6 x analògics, 8 x PWM), UART, I2C, SPI
- Admet instruccions DSP, càlculs d'unitat de punto flotant (FPU) accelerats per HW
- Mòdul criptogràfic i de seguretat ARM TrustZone CryptoCell-310
- Fins +8 dBm de potència TX (fins -20 dBm en passos de 4 dB)
- Ràdio NFC-A

Coprocessador de xarxa Wi-Fi de l'argón: Coprocessador Wi-Fi Espressif ESP32-D0WD 2.4 GHz

- Flaix integrada de 4 MB pel ESP32
- Compatibilitat amb 802.11 b / g / n
- 802.11 n (2.4 GHz), fins 150 Mbps

Especificacions generals de l'argón:

- Flaix SPI addicional de 4 MB incorporat
- Micro USB 2.0 de màxima velocitat (12 Mbps)
- Connector de bateria y carga Li-Po integrat
- Connector JTAG (SWD)
- LED d'estat RGB
- Botons de re-inici i mode
- Antena PCB integrada de 2,4 GHz per Bluetooth (no es compatible amb Wi-Fi)
- Dos connectors U.FL per antenes externes (un per Bluetooth, l'altre per Wi-Fi)

- Compleix amb l'especificació Feather en dimensions i distribució
- Certificació FCC, CE e IC
- Compleix con RoHS (sense plom)

La placa de desenvolupament Argon, tindrà un sensor connectat, que ens permetrà realitzar una detecció continua i amb temps real de la concentració de material particulat en l'aire, aquest sensor és:

Laser PM2.5 HM-3301 Dust Sensor [9], es base en la tecnologia de dispersió de la llum làser, i aconsegueix unes lectures precises, estables i consistents. Se'n pot veure la imatge a la Figura 3.2.

Figura 3.2: HM3001 Dust Sensor.

Les seves principals característiques i avantatges són:

- Segueix les normes ISO 21501-4, ISO 14644-1 i FS209E.
- Alta sensibilitat en partícules de pols de 0.3 μ m o més.
- Admet sortida de sis canals de 0.3 μm, 0.5 μm, 1.0 μm, 2.5 μ m, 5 μm, 10 μm.
- Sortida directa de concentració de massa PM2.5, PM10 amb unitat de μg / m3.
- Detecció contínua i en temps real de la concentració de pols a l'aire.
- Amb compensació d'humitat, escalable per al sensor de temperatura i humitat

- Consum d'energia molt baix. El consum d'energia és inferior a 150 mA en mode de repòs, i menys de 75 mA durant el funcionament.
- Admet modes de comunicació de les interfícies I2C i Uart per al desenvolupament secundari i la integració del sistema.
- Baix nivell de soroll, mida petita, pes lleuger i fàcil instal·lació.

3.2 Recol·lecció de les dades IoT

Per poder realitzar la recol·lecció de les dades que genera el sensor i poder-les guardar en una fulla de càlcul, s'utilitza la plataforma "Web IDE"https://build.particle.io, on permet gestionar aplicacions i gravar-les directament en la memòria flaix del dispositiu.

S'ha realitzat una aplicació que recull la telemetria cada 10 minuts obtinguda del sensor HM3001 que està connectat al dispositiu particle Argon i la grava en una fulla de càlcul de Google Sheets. Es guarda mitjançant una integració IFTTT (If This, Then That) [11], que és un servei basat en la web i que permet crear integracions a traves de declaracions condicionals simples, anomenades applets.

La telemetria registrada és 1.0 μm, 2.5 μm, 10 μm per cada un d'ells n'hi ha dos tipus: el particulat estàndard i el de l'entorn atmosfèric, també és registre la data i hora en que s'ha recollit la informació.

Capítol 4

Disseny i implementació

4.1 Dades utilitzades

Per la realització del treball, s'utilitzaran dos conjunts de dades. El primer conjunt de dades, són dades històriques que van estar recopilades en l'estudi del següent article [35]. Hi ha dades del 16 de setembre de 2019 fins al 18 de novembre de 2019. Hi ha tres períodes de recol·lecció de dades diferenciats. Del 16 de setembre al 14 d'octubre el sensor va està ubicat a l'interior de la Universitat Jaume I, del 15 al 30 d'octubre, va està ubicat a l'exterior, a la ciutat de Vila-real, en una ciutat industrial. I del 31 d'octubre al 18 de novembre un altre cop a l'interior de la Universitat Jaume I. Aquest primer conjunt de dades històriques, s'utilitzaran per realitzar la primera aproximació del model de Machine Learning per la detecció d'anomalies.

El segon conjunt de dades, són dades recopilades durant la realització del treball, el dia inicial que va començar el procés de captura va ser el 10 de novembre de 2020 i ha finalitzat el 13 de desembre de 2020, mitjançant el dispositiu i sensor esmentat anteriorment. El dispositiu està ubicat a l'exterior, al poble Vilafamés en la província de Castelló. Es poden veure imatges del dispositiu en les Figures 4.1 i 4.2.

Figura 4.1: Fotografia posterior del dispositiu presa de l'exterior del carrer.

Figura 4.2: Fotografia anterior del dispositiu presa des del balcó.

Aquest segon set de dades s'utilitzaran per testejar el model generat mitjançant el primer conjunt de dades i poder ajustar el model per llavors utilitzar-lo a dins del dispositiu IoT.

4.2 Exploració i preparació de les dades

Per realitzar l'exploració de les dades dels dos conjunts de dades, s'ha realitzat un anàlisis estadístic i exploratori de les dades.

4.2.1 Anàlisis estadístic del conjunt de dades històriques

En la Figura 4.8 es poden veure els cinc primer registres del conjunt de dades amb les 7 variables existents: Time, PM1, PM 2.5, PM 10, PM 1 ATM, PM 2.5 ATM, PM 10 ATM. La unitat dels sensors és de μg/m^3

Head

	Time	PM 1	PM 2.5	PM 10	PM 1 ATM	PM 2.5 ATM	PM 10 ATM
0	2019-09-16 10:53:47	9	13	15	9	13	15
1	2019-09-16 11:03:47	10	14	16	10	14	16
2	2019-09-16 11:14:31	9	13	15	9	13	15
3	2019-09-16 12:40:41	10	14	16	10	14	16
4	2019-09-16 13:12:30	9	13	15	9	13	15

Figura 4.3: Resum de les dades històriques.

En la Figura 4.9 es pot observar el resum estadístic de les dades, on es pot apreciar que hi ha 4989 registres. Es pot observar que les sis variables de particulat tenen nivells de concentració negatius. La variable que té el valor més petit i el més gran de concentració, és la "PM 1 ATM". La majoria de les variables el valor màxim és de 127μg/m^3.

	count	mean	std	min	25%	50%	75%	max
PM 1	4989.0	5.161355	6.405865	-97.0	2.0	3.0	7.0	127.0
PM 2.5	4989.0	7.285027	7.626725	-70.0	3.0	4.0	10.0	127.0
PM 10	4989.0	10.649629	8.625245	-65.0	6.0	7.0	14.0	127.0
PM 1 ATM	4989.0	5.156945	5.620091	-116.0	2.0	3.0	7.0	127.0
PM 2.5 ATM	4989.0	7.323913	7.775398	-13.0	3.0	4.0	10.0	127.0
PM 10 ATM	4989.0	10.552215	8.258378	-104.0	6.0	7.0	14.0	83.0

Figura 4.4: Resum estadístic de les dades històriques.

En la Figura 4.10 es pot observar que la variable "Time"és de tipus objecte, i la resta de variables són de tipus int64. No hi ha valors absents.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4989 entries, 0 to 4988
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Time        4989 non-null   object
 1   PM 1        4989 non-null   int64
 2   PM 2.5      4989 non-null   int64
 3   PM 10       4989 non-null   int64
 4   PM 1 ATM    4989 non-null   int64
 5   PM 2.5 ATM  4989 non-null   int64
 6   PM 10 ATM   4989 non-null   int64
dtypes: int64(6), object(1)
memory usage: 273.0+ KB
```

Figura 4.5: Informació de les variables de les històriques.

4.2.2 Anàlisis exploratori del conjunt de dades històriques

En la Figura 4.11 es pot observar que totes les variables de PM, segueixen un mateix patró.

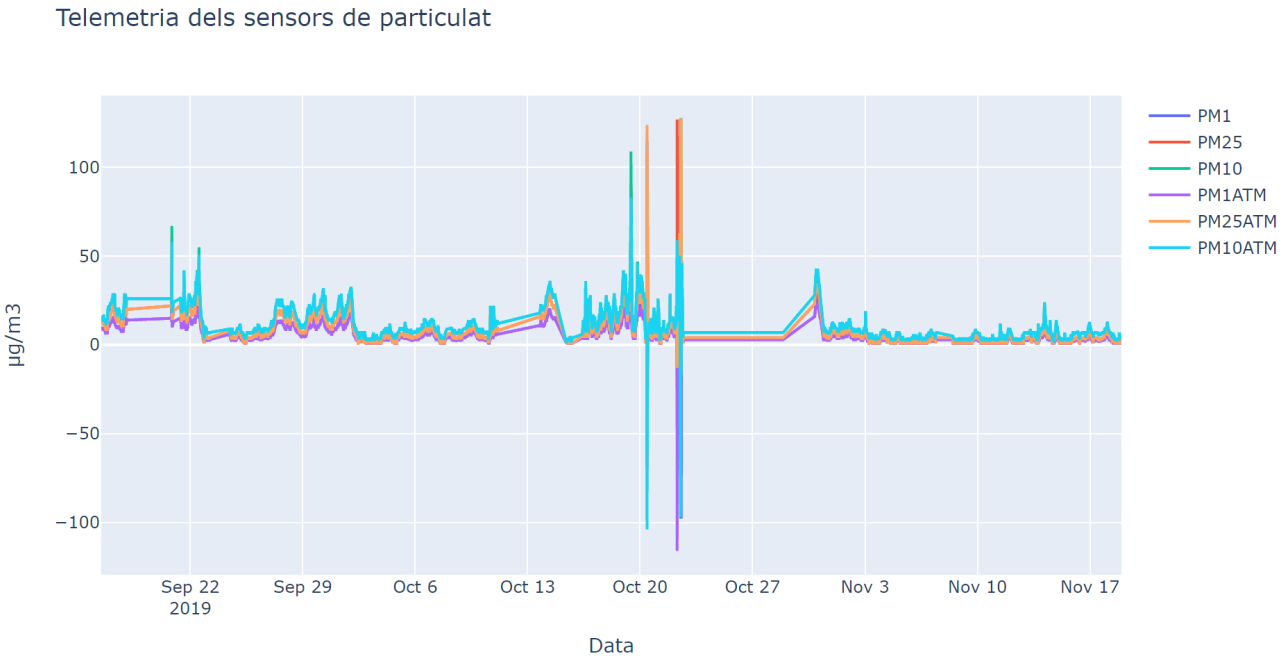


Figura 4.6: Timeseries de les dades històriques.

En la Figura 4.12 es pot observar que cada una de les variables PM amb la seva mesura atmosfèrica, segueixen un mateix patró en els quartils i en els bigotis. Es pot veure que cada una de les variables té valors atípics, s'identifiquen perquè són valors més grans al bigoti superior i més petits al bigoti inferior. Es pot veure que de valors atípics n'hi ha més per sobre dels bigotis superiors. La mediana ocupa la part inferior. En PM1 i PM1 ATM té un valor de 3, en PM2.5 i PM2.5 ATM de 4, en PM10 i PM10 ATM de 7. El Q1, ens indica que almenys un 25% de les observacions són menors o iguals a 2 en PM1 i PM1 ATM, 3 en PM2.5 i PM2.5 ATM i de 6 en PM10 i PM10 ATM. L'amplitud interquartíclica (IQR): Q3 - Q1 en PM1 i PM1 ATM de 5, en PM2.5 i PM2.5 ATM de 7 i en PM10 i PM10 ATM de 5. El Q3, ens indica que almenys un 75% de les observacions són menors o igual que 7 en PM1 i PM1 ATM, 10 en PM2.5 i PM2.5 ATM i de 14 en PM10 i PM10 ATM.

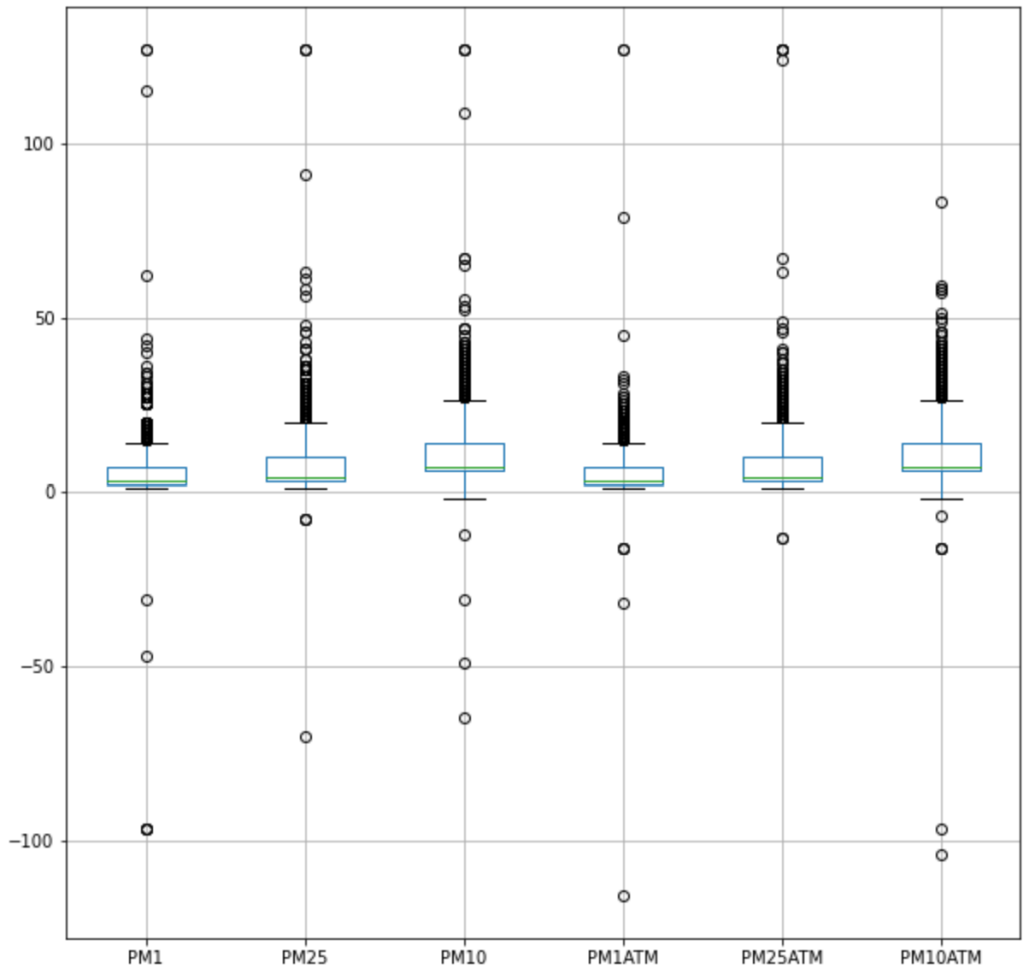


Figura 4.7: Boxplot de les variables de les històriques.

4.2.3 Anàlisis estadístic de les dades recopilades durant el treball

En la Figura 4.8 es poden veure els cinc primer registres del conjunt de dades amb les 7 variables existents: Time, PM1, PM 2.5, PM 10, PM 1 ATM, PM 2.5 ATM, PM 10 ATM. La unitat dels sensors és de $\mu g/m^3$. *Espotobservarquelavariable Time previamentsel'hihaassignatcomaindex*.

Head

Time	index	PM 1	PM 2.5	PM 10	PM 1 ATM	PM 2.5 ATM	PM 10 ATM
2020-11-10 23:12:18.699000+00:00	0	16	23	29	16	23	29
2020-11-10 23:22:18.480000+00:00	1	17	24	30	17	24	30
2020-11-10 23:32:18.549000+00:00	2	17	25	32	17	25	32
2020-11-10 23:42:18.563000+00:00	3	17	25	32	17	25	32
2020-11-10 23:52:18.543000+00:00	4	17	25	32	17	25	32

Figura 4.8: Resum de les dades recopilades durant el treball.

En la Figura 4.9 es pot observar el resum estadístic de les dades, on es pot apreciar que hi ha 3897 registres. Es pot observar que les sis variables de particulat tenen nivells de concentració negatius.El valor màxim l'ha registrat el sensor PM 10 ATM amb un valor de 121 μg/m^3.*Lavariableindexnoestindràencompteen L'anàlisi.*

	count	mean	std	min	25%	50%	75%	max
index	3897.0	1948.000000	1125.111328	0.0	974.0	1948.0	2922.0	3896.0
PM 1	3897.0	13.705414	13.269070	-126.0	3.0	9.0	19.0	118.0
PM 2.5	3897.0	18.627919	17.588151	-113.0	4.0	13.0	28.0	108.0
PM 10	3897.0	23.342828	19.427695	-125.0	7.0	18.0	35.0	117.0
PM 1 ATM	3897.0	12.047986	10.422024	-127.0	3.0	9.0	19.0	89.0
PM 2.5 ATM	3897.0	17.393893	15.022488	-128.0	4.0	13.0	28.0	117.0
PM 10 ATM	3897.0	22.613036	17.753231	-123.0	7.0	18.0	35.0	121.0

Figura 4.9: Resum estadístic de les dades recopilades durant el transcurs d'aquest treball.

En la Figura 4.10 es pot observar que les variables dels sensors són de tipus int64. No hi ha valors absents.

```
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 3897 entries, 2020-11-10 23:12:18.699000+00:00 to 2020-12-13 23:55:36.401000+00:00
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   index       3897 non-null   int64
 1   PM 1        3897 non-null   int64
 2   PM 2.5      3897 non-null   int64
 3   PM 10       3897 non-null   int64
 4   PM 1 ATM    3897 non-null   int64
 5   PM 2.5 ATM  3897 non-null   int64
 6   PM 10 ATM   3897 non-null   int64
dtypes: int64(7)
memory usage: 243.6 KB
```

Figura 4.10: Informació de les variables de les dades recopilades durant el transcurs d'aquest treball.

4.2.4 Anàlisis exploratori de les dades recopilades durant el treball

En la Figura 4.11 es pot observar que totes les variables de PM, segueixen un mateix patró. Es pot observar que hi ha diferents pics en negatiu de les variables i que estan en diferents franges de temps.

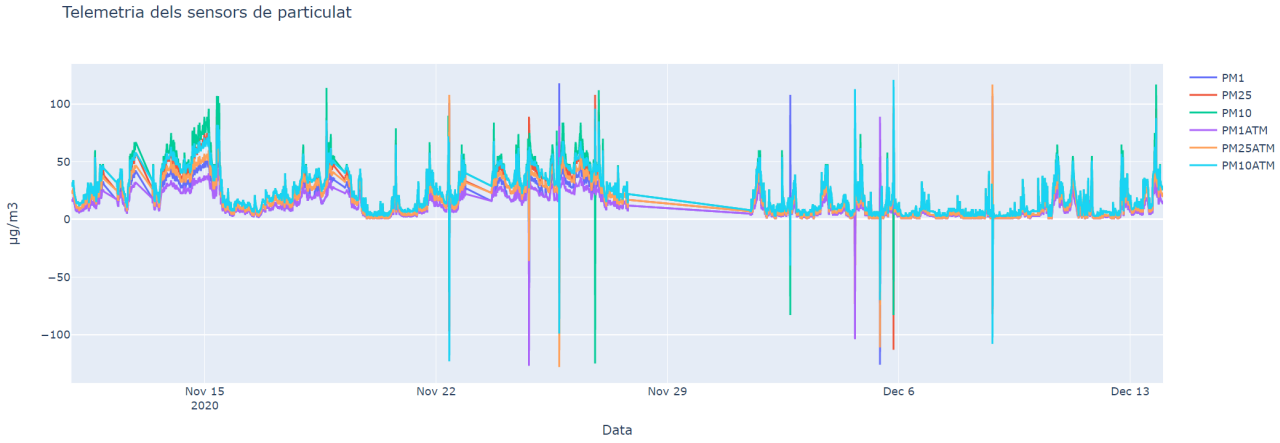


Figura 4.11: Timeseries de les dades recopilades durant el transcurs d'aquest treball.

En la Figura 4.12 es pot observar que cada una de les variables PM amb la seva mesura atmosfèrica, segueixen un mateix patró en els quartils i en els bigotis. Es pot veure que cada una de les variables té valors atípics, s'identifiquen perquè són valors més grans al bigoti superior

i més petits al bigoti inferior. Es pot veure que de valors atípics n'hi ha més per sobre dels bigotis superiors. La mediana ocupa la part inferior. En PM1 i PM1 ATM té un valor de 9, en PM2.5 i PM2.5 ATM de 13, en PM10 i PM10 ATM de 18. El Q1, ens indica que almenys un 25% de les observacions són menors o iguals a 3 en PM1 i PM1 ATM, 4 en PM2.5 i PM2.5 ATM i de 7 en PM10 i PM10 ATM. L'amplitud inerquartíclica (IQR): Q3 - Q1 en PM1 i PM1 ATM de 16, en PM2.5 i PM2.5 ATM de 24 i en PM10 i PM10 ATM de 28. El Q3, ens indica que almenys un 75% de les observacions són menors o igual que 19 en PM1 i PM1 ATM, 28 en PM2.5 i PM2.5 ATM i de 35 en PM10 i PM10 ATM.

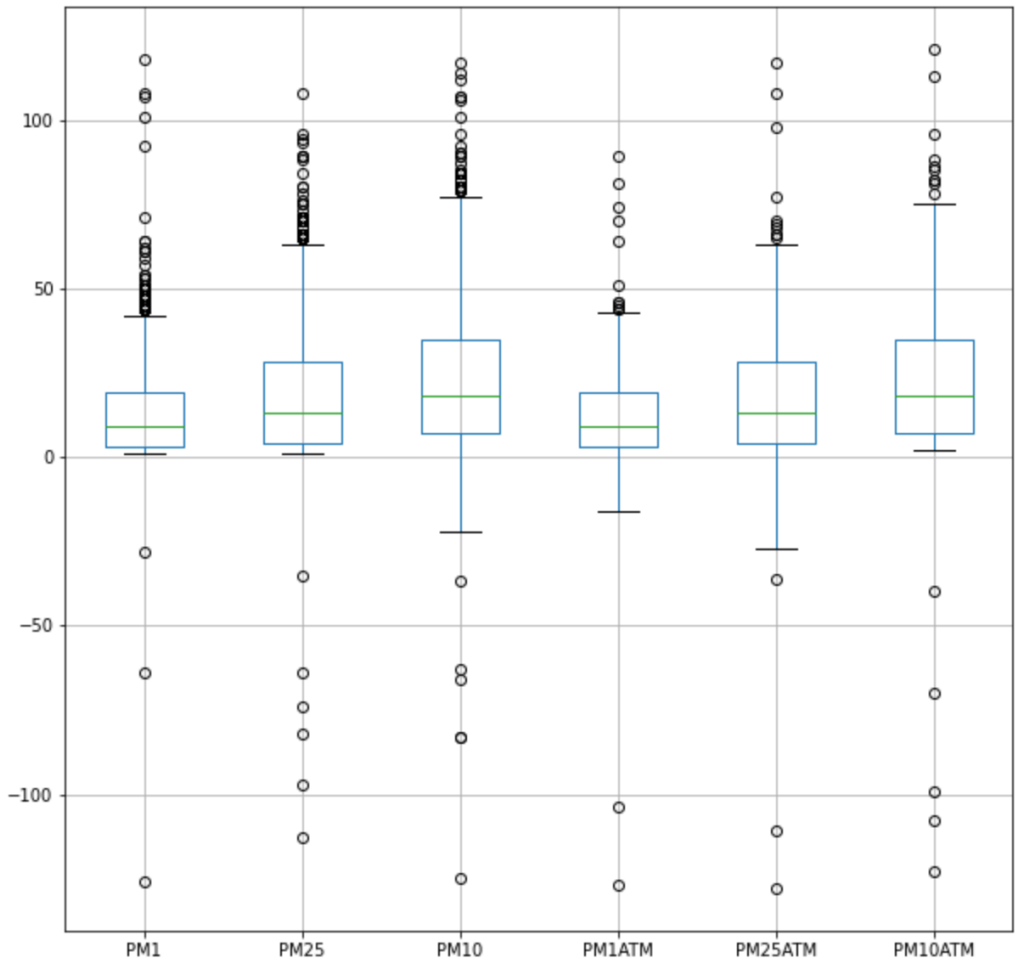


Figura 4.12: Boxplot de les variables de les dades recopilades durant el transcurs d'aquest treball.

4.2.5 Preparació de les dades

Segons la tipologia del model es realitzarà una preparació de les dades especifica per cadascun d'ells.

Xarxes Neuronals Es realitzarà un preprocessament de les dades amb tasques de normalització per facilitar una escala de valors equivalents que simplifiquin la comparació entre ells. Aquesta tècnica és útil per habitar que les dades quedin esbiaixades per la influència dels atributs amb valors més alts, distorsionant d'aquesta manera el resultat del model. La normalització serà la basada en la desviació estàndard, també anomenada estandarditzación de valors, que assegura que s'obtinguin els valors dins d'un rang elegit que tenen com a propietat que la seva mesura és 0 i la seva desviació estàndard és 1.Per realitzar aquesta normalització s'utilitzarà la funció *StandardScaler* de la llibreria *sklearn.preprocessing*.

Isolation Tree Es realitzarà un anàlisi de components principals (PCA), per reduir la dimensionalitat dels dos subconjunts de dades d'entrenament i test, es podrà veure en l'apartat 4.3.1. I poder treballar amb el model amb el component més adequat. Hi haurà un component per cada variable de particulat, en total 6. I s'utilitzarà la proba de *Dickey Fuller* [22] per saber el valor de p i saber si les dades són estacionaries i poder utilitzar el component òptim en el model de Isolation Tree.

4.3 Algoritmes de machine learning

4.3.1 Conjunts d'entrenament i de test

En la majoria dels models de machine learning, es necessita dividir el conjunt de dades en dos subconjunts:

- Conjunt d'entrenament: Un subconjunt per poder entrenar el model.
- Conjunt de test: Un subconjunt per poder testejar el model entrenat.

Aquesta divisió en subconjunts utilitzarem per dividir-los el principi de Pareto [21], diu que, en molts casos, el 80% dels efectes són conseqüència del 20% de les causes. El subconjunt

d'entrenament tindrà el 80% de les mostres del principi i el 20% restant seran del subconjunt de test. En la Figura 4.13 es pot veure el codi de com es creen els dos subconjunts de dades.

```
train_size = int(len(df2) * 0.8)
test_size = len(df2) - train_size
train, test = df2.iloc[0:train_size], df2.iloc[train_size:len(df2)]
train.shape, test.shape
```

```
((3991, 7), (998, 7))
```

Figura 4.13: Creació dels subconjunts d'entrenament i de test.

Un cop es tenen els subconjunts d'entrenament i de test, s'han de normalitzar les dades com hem vist en l'apartat 4.2.5.

Finalment perquè els models de les xarxes neuronals puguin predir les anomalies en una finestra de temps posterior, per exemple, predir-les en dades d'una hora, s'hauran de crear, per cada subconjunt de dades d'entrenament i de test, subseqüències segons la freqüència en la qual es vol fer la predicció. El model segons la configuració de les subseqüències tindrà un número d'entrades. En aquest treball es testejaran diferents finestres de temps per veure el funcionament dels diferents models, però en el model generat per testejar-lo en temps real dins el dispositiu IoT, s'utilitzarà la configuració d'una hora, que correspon a 6 lectures dels sensors, ja que aquestes lectures es realitzen cada 10 minuts. En l'apartat d'experimentació es podran veure les diferents proves realitzades.

En la següent Figura 4.14 es pot veure la funció.

```
TIME_STEPS=6 #6 registres hora x 24h --> equival a una finestra d'un dia

def create_sequences(X, y, time_steps=TIME_STEPS):
    Xs, ys = [], []
    for i in range(len(X)-time_steps):
        Xs.append(X.iloc[i:(i+time_steps)].values)
        ys.append(y.iloc[i+time_steps])

    return np.array(Xs), np.array(ys)

X_train, y_train = create_sequences(train[[columns[1]]], train[columns[1]])
X_test, y_test = create_sequences(test[[columns[1]]], test[columns[1]])
```

Figura 4.14: Creació de subseqüències de temps.

Per la detecció d'anomalies s'han utilitzat models no supervisats, les Xarxes Neuronals i l'Isolation Tree. De Xarxes Neuronals, s'han utilitzat les recurrents (RNN) i les profundes (DNN). De les RNN, s'han utilitzat dos models especifics, Long Short Term Memory (LSTM) i la Gated Recurrent Unit (GRU). Es poden veure en detall cadascuna d'elles en els apartats 4.3.5 4.3.6 4.3.7.

4.3.2 Tècniques per evitar el sobreentrenament

Per evitar el sobreentrenament dels models de les xarxes neuronals, s'utilitzen dues tècniques:

DropOut: La capa de dropout té la funció molt especifica en les xarxes neuronals, que és, prevenir el sobreentrenament. Aquesta capa desactiva un número aleatori d'entrades de la capa, posant a un valor igual a 0. El principal benefici és que està forçant a la xarxa a ser redundant, sent aquesta capa la responsable de donar la classificació i sortides correctes encara que algunes entrades estiguin inactives.

EarlyStopping: És una funcionalitat que ofereix la llibreria keras de python [5], el seu objectiu és evitar l'execució de masses èpoques d'entrenament degut a poden provocar un sobreajustament del conjunt de dades d'entrenament. Utilitzar-ne poques pot donar un model inadequat. La detecció anticipada és un mètode que permet especificar una gran quantitat arbitrària d'èpoques d'entrenament i parar l'entrenament un cop el rendiment del model deixa de millor en el conjunt de dades de validació d'espera.

4.3.3 Avaluació del models

En els models no supervisats, s'utilitzen mesures per saber la bondat del model. Per realitzar aquest mesurament, en les xarxes neuronals, podem utilitzar diferents mètriques de regressió que podem veure en la taula 4.1

$$\begin{array}{lcl} \text{Mean squared error} & \text{MSE} & = \frac{1}{n}\sum_{t=1}^{n} e_t^2 \\ \text{Root mean squared error} & \text{RMSE} & = \sqrt{\frac{1}{n}\sum_{t=1}^{n} e_t^2} \\ \text{Mean absolute error} & \text{MAE} & = \frac{1}{n}\sum_{t=1}^{n} |e_t| \end{array}$$

Taula 4.1: *Mètriques de regressió.*

Les tres mètriques són una mesura molt habitual per calcular les diferències entre els valors predits per un model i els valors realment observats. Com més baix sigui la mètrica, millor serà el nostre model.

En aquest treball s'utilitzarà l'RMSE perquè ens ha donat els valors més alts respecte a MSE i MAE, això passa perquè els erros s'eleven al quadrat abans de fer la mitjana, i l'RMSE atorga un pes relativament més alt als errors grans, i això vol dir que hauria de ser més útil quan els errors grans són indesitjables, que és el que es vol en aquest treball. En podem veure un exemple dels resultats dels càlculs en la Figura 4.15, on es pot observar que els valors de RMSE són més grans que MSE i MAE.

	model	sequence	activationDense	optimizer	dropout1	units	epochs	batchsize	validation_split	RMSE	MSE	MAE	Time
0	DNN	1h	tanh	adam	0.405196	87	80	31	0.1	0.019200	0.000369	0.009905	17.237772
0	DNN	1h	tanh	adam	0.118148	80	90	25	0.1	0.018856	0.000356	0.005032	22.277235
0	DNN	1h	tanh	adam	0.484764	12	52	57	0.1	0.017678	0.000313	0.011355	6.903105
0	DNN	1h	tanh	adam	0.405196	87	80	31	0.2	0.021332	0.000455	0.013577	16.152798
0	DNN	1h	tanh	adam	0.118148	80	90	25	0.2	0.021047	0.000443	0.008638	20.035282

Figura 4.15: Taula de resultats DNN per visualitzar els valors RMSE.

En l'algorisme de Isolation Tree, per calcular l'avaluació del model, es tindrà en compte l'anàlisi visual que s'ha realitzat, i es comprovarà a partir de quin valor el diagrama de boxplot

considera els valors atípics, seràn el nombre teòric de valors atípics. I calcularem els valors predits / nombre teòric de valors atípics. I amb el resultat decidirem la bondat del model.

4.3.4 Determinació d'anomalia

Un cop s'ha entrenat el model i avaluat el model, es la hora de fer la predicció usant les dades de test, i saber si és una anomalia o no.

En les xarxes neuronals es realitza mitjançant la funció de pèrdua com hem vist la subsecció 4.3.3 i en el calcul del *threshold*, es pot veure la funció en la Figura 4.16

```
#càlcul del threshold de test
def calculate_threshold(X_test, X_test_pred):
    distance = np.linalg.norm(X_test - X_test_pred, axis=1);
    """Sorting the scores/diffs and using a 0.80 as cutoff value to pick the threshold"""
    distance.sort();
    cut_off = int(0.80 * len(distance));
    threshold = distance[cut_off];
    return threshold
```

Figura 4.16: Funció del càlcul del threshold.

Es determina averia en un registre de particulat, si el resultat del seu valor de pèrdua és més gran al seu càlcul del threshold.

En el cas de l'Isolation Tree, utilitzarem la mateixa funció de predicció del model, *predict*, que determina si és anomalia sí el resultat és -1, i no és anomalia si és 1. En la configuració del model, es configura el paràmetre *contamination* que ajuda a ajustar el llindar en les puntuacions de les mostres. És la proporció de valors atípics en el conjunt de dades. [24]

4.3.5 Xarxes Neuronals Recurrents (RNN)

Les xarxes neuronals recurrents son molt útils per tractar seqüència de dades. Són una classe de les xarxes neuronals artificials on les connexions entre els estats presenten un o més cicles recurrents. Aquest cicles de retroalimentació es poden considera memòries internes del sistema.

En aquest treball s'han realitzat dos tipus de RNN, la Long Short Term Memory (LSTM) i la Gated Recurrent Unit (GRU), que són utilitzades en detecció d'anomalies en series temporals, com hem pogut veure en l'apartat de l'estat de l'art 2.

4.3.5.1 Long Short Term Memory (LSTM)

Long Short Term Memory, són una extensió de les RNN, que bàsicament amplien la seva memòria per aprendre de experiències importants que han passat fa molt temps. Les LSTM permeten recordar les seves entrades durant un període llarg de temps. La neurona d'una LSTM pot llegir, escriure i eliminar informació de la memòria. En la Figura 4.17 es pot veure l'esquema d'una cel·la LSTM. Com es pot veure en la figura, l'arquitectura LSTM es basa en controlar quina informació es guarda l'estat o memòria i influeix directament per la part superior i quina informació es produeix com a resposta de la xarxa. Aquest control es realitza mitjançat la porta de l'oblid, que observa el valor d'entrada a la cel·la juntament amb la sortida de la xarxa en el pas anterior i decideix quina part de memòria ha de conservar, i la porta d'entrada, que controla quina informació s'afegeix a la memòria de la xarxa.

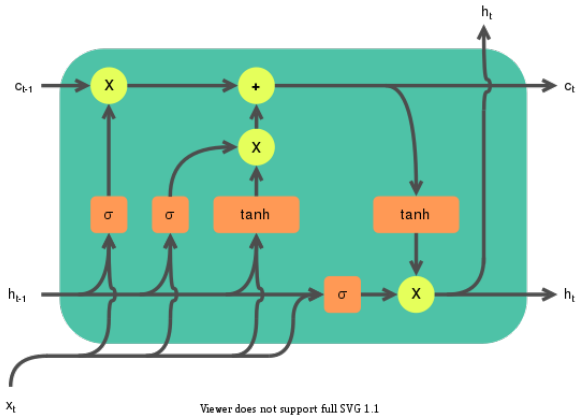


Figura 4.17: Esquema de LSTM. Font:[15]

En la Figura 4.18 es pot veure un exemple de codi que s'ha testejat en la realització del model.

```
model = Sequential()
model.add(LSTM(units = 64,time_major=False,return_sequences=True,input_shape=(X_train.shape[1], X_train.shape[2])))
model.add(Dropout(rate=0.3))
model.add(TimeDistributed(Dense(X_train.shape[2],kernel_initializer='normal',activation='tanh')))
model.compile(optimizer='adamax', loss='mae', metrics= [rmse,'mse'])
model.summary()
```

Figura 4.18: Algoritme LSTM d'una capa.

4.3.5.2 Gated Recurrent Unit (GRU)

Gated Recurrent Unit, usen el mateix principi que les LSTM, però estan simplificades de manera que el seu rendiment és similar però són més eficients computacionalment. Per aquesta rao les cel·les GRU son molt utilitzades quan el conjunt de dades no es molt gran, ja que una cel·la GRU hi ha menys paràmetres i per tant es necessita menys dades per l'entrenament. En la Figura 4.19 es pot veure l'esquema de la cel·la GRU. Com es pot veure en la figura, el fluxe d'informació es controla per la porta de reset, que permet seleccionar quina informació de la memòria serà utilitzada en un pas en concret. Per ell, observa l'estat anterior de la xarxa i les dades d'entrada i obté un vector que serà multiplicat posteriorment amb l'estat de la memòria, i la porta d'actualització es l'anàloga a les portes de l'oblid de la cel·la LSTM.

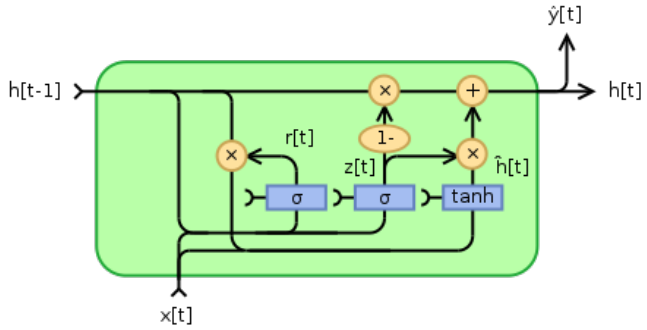


Figura 4.19: Esquema de GRU. Font:[10]

En la Figura 4.20 es pot veure un exemple de codi que s'ha testejat en la realització del model.

```
model = Sequential()
model.add(GRU(units = 128,time_major=False,return_sequences=True,input_shape=(X_train.shape[1], X_train.shape[2])))
model.add(Dropout(rate=0.8))
model.add(TimeDistributed(Dense(X_train.shape[2],kernel_initializer='normal',activation='tanh')))
model.compile(optimizer='adamax', loss='mae', metrics= [rmse,'mse'])
model.summary()
```

Figura 4.20: Algoritme GRU d'una capa.

4.3.6 Deep Neural Network (DNN)

Deep Neural Network, és una xarxa neuronal artificial amb múltiples capes entre les capes d'entrada i sortida. Normalment són xarxes de retroalimentació en que les dades flueixen desde

la capa d'entrada a la capa de sortida sense retrocedir. En la Figura 4.21 es pot veure un esquema de xarxa neuronal profunda amb una capa d'entrada, una capa oculta i una capa de sortida.

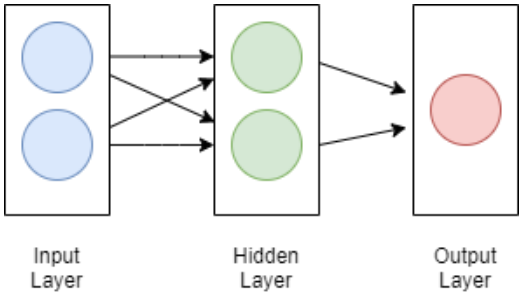


Figura 4.21: Esquema DNN amb una capa oculta.

En la Figura 4.22 es pot veure un exemple de codi que s'ha testejat en la realització del model.

```
model = Sequential()
model.add(Dense(units=21, input_shape=(X_train.shape[1], X_train.shape[2]), activation='relu'))
model.add(Dense(16, activation='relu'))
model.add(Dense(X_train.shape[2],activation='relu'))
model.compile(optimizer='adamax', loss='mae', metrics= [rmse,'mse'])

model.summary()
```

Figura 4.22: Algoritme DNN d'una capa.

4.3.7 Isolation Forest

Isolation Forest, bosc de aïllament, és un algorisme d'aprenentatge no supervisat per identificar anomalies quan les dades no estan etiquetades, no es coneix la classificació real (anomalia / no anomalia) de les observacions. El seu funcionament està inspirat en l'algorisme de classificació i regressió Random Forest. La tècnica que utilitza és aïllar explícitament els punts anòmals del conjunt de dades. És un algoritme molt ràpid amb una baixa sol·licitud de memòria. El model Isolation Forest s'obté al combinar múltiples *isolation tree*, cada un entrenant amb una mostra diferent generada per *bootstrapping* a partir de les dades originals. El valor predit per cada observació és el número mitjà de divisions que s'ha necessitat per aïllar la observació del conjunt d'arbres. Quant més petit és el valor, més probabilitat hi ha que est tracti d'una anomalia. [23]

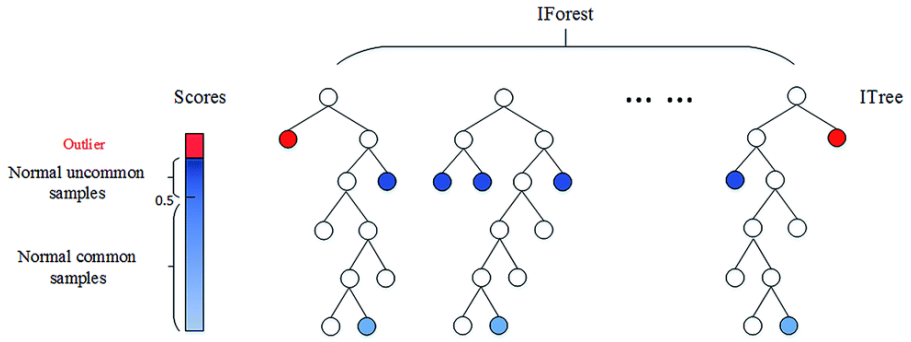


Figura 4.23: Isolation Forest. Font:[14]

```
outliers_fraction = 0.07
model =  IsolationForest(max_samples=6,contamination=outliers_fraction)
model.fit(principalDf.values)
```

Figura 4.24: Algoritme Isolation Forest.

4.4 Desplegament del model a producció

En el transcurs del desenvolupament del treball, hi ha hagut diferents plantejaments per arribar a la solució que porta el nom del títol del treball "Detecció i predicció d'anomalies en dispositius IoT en l'Edge computing".

4.4.1 Plantejament inicial

Inicialment el treball es va plantejar per dur a terme el desenvolupament d'un model de Machine Learning (ML) per detectar anomalies en temps real i poder-lo incorporar dins del dispositiu IoT en l'edge computing. La ídea inicial era que, el dispositiu particle Argon, recull del sensor del particulat HM3301 durant una seqüència de temps, i un cop ha recollit les dades que necessita el model de ML que està dins del microcontrolador, les hi passa, i aquest model retorna els resultats de les mètriques de predicció i llavors, mitjançant programació, es decideix si hi ha dades anòmales, i es guarden el resultats en una fulla de càlcul.

Durant el transcurs del desenvolupament, s'ha hagut de tenir en compte que la memòria del dispositiu IoT és limitada, per tant el model de Machine Learning havia de ser el més simple possible per detectar anomalies i que hi cabes dins de la memòria del dispositiu.

En les següents Figures 4.25 i 4.26 hi ha el codi per convertir el model perquè es pugui incorporar i executar-lo dins del dispositiu de l'edge computing.

```
run_model = tf.function(lambda x: model(x))
# This is important, let's fix the input size.
BATCH_SIZE = 1
STEPS = 6
INPUT_SIZE = 1
concrete_func = run_model.get_concrete_function(
    tf.TensorSpec([BATCH_SIZE, STEPS, INPUT_SIZE], model.inputs[0].dtype))

# model directory.
MODEL_DIR = "keras_lstm"
model.save(MODEL_DIR, save_format="tf", signatures=concrete_func)

converter = tf.lite.TFLiteConverter.from_saved_model(MODEL_DIR)
tflite_model = converter.convert()

open("converted_model.tflite", "wb").write(tflite_model)
```

Figura 4.25: Sentències per convertir el model cap a TFLite

```
!xxd -i converted_model.tflite > converted_model.cpp
```

Figura 4.26: Sentència per convertir el model TFLite a cpp.

Finalment no s'ha pogut acabar de dur a terme degut a la incompatibilitat del models no supervisats realitzats. Aquests models requereixen d'un preprocessament de les dades amb la seva estandarització i llavors un cop el model ha realitzat la predicció, s'han de realitzar càlculs de la pèrdua del RMSE i el Threshold, i llavors calcular si és anomalia o no. S'han estudiat les llibreries Math del dispositiu particle, i no compleixen el requeriment per dur a terme el procés i també manquen operacions com la mitjana [17]. En la documentació oficial del dispositiu Particle, hi ha exemples de ML [18], però són exemples de models supervisats i el model et retorna directament si és cert o fals, i això en aquest treball no ha estat possible.

Arribat en aquest punt, s'ha replantejat la solució per poder dur a terme les prediccions d'anomalies en les dades IoT.

4.4.2 Replantejament final

Per poder dur a terme la realització de l'estudi, es durà a terme el desenvolupament de REST-APIS desenvolupades amb django [3], que és un framework que s'utilitza per desenvolupar serveis de Python i s'instal·laran a un servidor amb sistema operatiu Linux per poder-ho realitzar. Es realitzaran diferents serveis, que se'ls passarà per paràmetre una matriu de valors, i cadascun d'ells amb un model diferent: LSTM, GRU, DNN i Isolation Forest retornaran si els valors que han rebut són anòmals. Dins del microcontrolador, recopilarà la informació dels valors cada deu minuts, i un cop n'hagi recopilat els valors necessaris per els models, en aquest cas 6 valors, els enviarà a cadascuna de les APIS, per poder-ne realitzar la predicció de cada un dels valors. El resultat de les prediccions es guardarà en temps real en una fulla de càlcul per poder comparar els resultats obtinguts dels diferents models.

4.4.3 Implementació

Els processos realitzats juntament amb el seu codi de programació, així com els conjunts de dades utilitzats pel desenvolupament d'aquest treball final de màster, es poden localitzar a l'enllaç de *Github* del treball:
https://github.com/tllussa/TFM-UOC-MU-DATA-SCIENCE sota la llicència Creative Commons (CC BY-NC-ND 4.0).

Capítol 5

Experimentació

En aquest treball s'han dut a terme diferents experimentacions:

- Recerca dels models òptims de Machine Learning.
- Visualització de les anomalies.
- Posta en producció dels models al cloud computing.

5.0.1 Recerca dels models òptims de Machine Learning

Per localitzar el model òptim per cada tipus de model de Machine Learning: xarxes neuronals recurrents (RNN), xarxes neuronals profundes (DNN) i Isolation Forest, s'han dut a terme varis processos per testejar diferents configuracions dels paràmetres de cadascun dels models. Les dades utilitzades per realitzar aquesta recerca seran les del conjunt de dades històriques. S'usaran les dades del sensor PM2.5 per realitzar aquests estudis. Cada model òptim de Machine Learning obtingut en la finestra de temps de predicció d'una hora, són els que s'utilitzaran en la programació de les REST-API per retornar la predicció en l'edge computing. Un cop realitzada la recerca se'n extrauran conclusions, si els models detecten les anomalies, si hi ha resultats similars segons la finestres de predicció, etc. Aquestes conclusions es podran veure en el capítol de conclusions 6.

5.0.1.1 Xarxes Neuronals

S'ha realitzat un procés de recerca de varies configuracions per cercar un model òptim per les xarxes RNN i DNN, i per cada finestra de temps de predicció: 1 hora, 3 hores, 6 hores, 12 hores, 1 dia, 3 dies i 7 dies. Es considerarà el model òptim, el que tingui el valor de la mètrica RMSE més baix en la seva finestra de temps de predicció.

Per realitzar aquest procés de recerca, primer de tot s'ha hagut de realitzar una preparació de les dades, com s'ha vist en l'apartat 4.2.5 i la creació dels subconjunts d'entrenament i test, com s'ha vist en l'apartat 4.3.1.

Recerca dels models òptims per les RNN: Per realitzar la recerca del model òptim en les RNN, s'han definit dos tipus de models: un model d'una capa i un model de dues capes. En la Figura 5.1 es pot observar la definició del model d'una capa, és un model seqüencial, amb una capa d'entrada RNN, una capa dropout i una Capa de sortida. També s'hi pot veure el llistat de paràmetres que s'utilitzaran per la recerca de l'estudi d'aquests dos tipus de models.

En la capa d'entrada, s'utilitzaran les xarxes LSTM i les GRU, per cadascuna d'elles es comprovarà el model amb les diferents finestres de temps de predicció: 1 hora, 3 hores, 6 hores, 12 hores, 1 dia, 3 dies i 7 dies. En la capa de sortida es comprovaran dos tipus d'activacions: la 'tanh' i la 'sigmoid'. Es provaran tres tipus d'optimitzadors: 'adam', 'adadelta' i 'adamax'. Es testejarà diferents velocitats d'aprenentatge, i s'utilitzaran valors aleatoris uniformes en la capa de dropout, les unitats, les èpoques i el batchsize.

```
def _model(cmodel,units,activationDense,dropout1,optimizer):
    model = Sequential()
    model.add(cmodel(units = units, return_sequences=True, input_shape=(X_train.shape[1], X_train.shape[2])))
    model.add(Dropout(rate=dropout1))
    model.add(TimeDistributed(Dense(1,kernel_initializer='normal',activation=activationDense)))
    model.compile(optimizer=optimizer, loss='mae')
    model.summary()

    return model

models = [LSTM,GRU]
nmodels = ["LSTM","GRU"]
sequences = ["1h","3h","6h","12h","1d", "3d", "7d"]
X_trains = [X_train1h,X_train3h, X_train6h, X_train12h,X_train1d,X_train3d, X_train7d]
y_trains= [y_train1h,y_train3h, y_train6h,y_train12h,y_train1d,y_train3d, y_train7d]
activations = ['relu']
activationsDense = ['tanh','sigmoid']
optimizers = ['adam','adadelta','adamax']
list_validationSplit = [0.1,0.2]
list_dropout1 =  np.random.uniform(0.1,0.8,5)
list_units = np.random.randint(6,high=100, size=5)
list_epochs = np.random.randint(5,high=100, size=5)
list_batchsize = np.random.randint(6,high=64, size=5)
```

Figura 5.1: Parametrització i configuració de RNN d'una capa

En la Figura 5.2, es pot observar la definició del model de dues capes, és un model seqüencial, amb una capa d'entrada RNN, una capa dropout, una capa intermitja RNN, una capa de dropout i una Capa de sortida. En la mateixa figura, es pot observar el llistat de paràmetres que s'utilitzaran per la recerca de l'estudi d'aquests dos tipus de models. En la capa d'entrada i la capa intermitja, s'usaran LSTM i les GRU, per cadascuna d'elles es comprovarà el model amb diferents finestres de temps de predicció: 1 hora, 3 hores, 6 hores, 12 hores, 1 dia, 3 dies i 7 dies. En la capa de sortida es comprovaran dos tipus d'activacions: la 'tanh' i la 'sigmoid'. Es provaran tres tipus d'optimitzadors: 'adam', 'adadelta' i 'adamax'. Es testejarà diferents velocitats d'aprenentatge, i s'utilitzaran valors aleatoris uniformes en la capa de dropout, les unitats, les èpoques i el batchsize.

```
def _model(cmodel,units,activationDense,dropout1,dropout2,optimizer):
    model = Sequential()
    model.add(cmodel(units = units, return_sequences=True, input_shape=(X_train.shape[1], X_train.shape[2])))
    model.add(Dropout(rate=dropout1))
    model.add(cmodel(units = units, return_sequences=True))
    model.add(Dropout(rate=dropout2))
    model.add(TimeDistributed(Dense(1,kernel_initializer='normal',activation=activationDense)))
    model.compile(optimizer=optimizer, loss='mae')
    model.summary()

    return model

models = [LSTM,GRU]
nmodels = ["LSTM","GRU"]
sequences = ["1h","3h","6h","12h","1d", "3d", "7d"]
X_trains = [X_train1h,X_train3h, X_train6h, X_train12h,X_train1d,X_train3d, X_train7d]
y_trains= [y_train1h,y_train3h, y_train6h,y_train12h,y_train1d,y_train3d, y_train7d]
activations = ['relu']
activationsDense = ['tanh','sigmoid']
optimizers = ['adam','adadelta','adamax']
list_validationSplit = [0.1,0.2]
list_dropout1 =  np.random.uniform(0.1,0.8,5)
list_dropout2 =  np.random.uniform(0.1,0.8,5)
list_units = np.random.randint(6,high=100, size=5)
list_epochs = np.random.randint(5,high=100, size=5)
list_batchsize = np.random.randint(6,high=64, size=5)
```

Figura 5.2: Parametrització i configuració de RNN de dues capes

Recerca dels models òptims per les DNN: Per realitzar la recerca del model òptim en les DNN, s'ha definit el model que es pot veure en la Figura 5.1, és un model seqüencial, amb una capa d'entrada de tipus dense, una capa dense, una capa dropout i una Capa de sortida dense. També es pot veure el llistat de paràmetres que s'utilitzaran per la recerca de l'estudi d'aquests dos tipus de models.

El model és testejarà amb diferents finestres de temps de predicció: 1 hora, 3 hores, 6 hores, 12 hores, 1 dia, 3 dies i 7 dies. En la capa de sortida es comprovaran dos tipus d'activacions: la 'tanh' i la 'sigmoid'. Es provaran tres tipus d'optimitzadors: 'adam', 'adadelta' i 'adamax'. Es testejarà diferents velocitats d'aprenentatge, i s'utilitzaran valors aleatoris uniformes en la capa de dropout, les unitats, les èpoques i el batchsize.

```
def _model(units,activationDense,dropout1,optimizer):
    model = Sequential()
    model.add(Dense(units=units, input_shape=(X_train.shape[1], X_train.shape[2]), activation='relu'))
    model.add(Dense(16, activation='relu'))
    model.add(Dropout(rate=dropout1))
    model.add(Dense(X_train.shape[2],activation=activationDense))
    model.compile(optimizer=optimizer, loss='mae')
    model.summary()

    return model

sequences = ["1h","3h","6h","12h","1d", "3d", "7d"]
X_trains = [X_train1h,X_train3h, X_train6h, X_train12h,X_train1d,X_train3d, X_train7d]
y_trains= [y_train1h,y_train3h, y_train6h,y_train12h,y_train1d,y_train3d, y_train7d]
activationsDense = ['tanh','sigmoid']
optimizers = ['adam','adadelta','adamax']
list_validationSplit = [0.1,0.2]
list_dropout1 =  np.random.uniform(0.1,0.8,5)
list_units = np.random.randint(5,high=100, size=5)
list_epochs = np.random.randint(5,high=100, size=5)
list_batchsize = np.random.randint(5,high=64, size=5)
```

Figura 5.3: Parametrització i configuració de DNN

Avaluació dels models: Per cada model de xarxa neuronal, es realitzarà el càlcul de les diferents mètriques per realitzar l'avaluació del model. La definició de les mètriques s'han vist en l'apartat 4.3.3. Per la validació dels models, ens basarem en la mètrica RMSE. Per cada franja horària de predicció, s'escollirà el model que obtingui el valor RMSE més petit degut a que atorga un pes relativament més alt als errors gran, i això vol dir que hauria de ser més útil quan els errors grans són indesitjables, que és el que es vol en aquest treball.

Resultats dels models: En les Taules 5.1, 5.2, 5.3, 5.4 i 5.5 es poden veure els resultats òptims aconseguits amb les proves realitzades en la recerca del millor model, segons els sistema d'avaluació.

En la Taula 5.1 es poden observar les prediccions de la RNN GRU d'una capa. Segons la seqüència, franja de temps a fer la predicció, comença a incrementar-se la mètrica RMSE a partir de 3 dies, però s'ha de tenir en compte el temps d'execució a partir de les 3 hores comencen a incrementar-se considerablement. El temps més baix és d'un minut i el més alt és de 27 minuts. Es pot veure que majoritàriament, la capa d'activació és la 'tanh', l'optimitzador és 'adamax' i el temps de validació és 0.2, han sortit en els millors resultats.

Sequence	Activation	Optimizer	DropOut	Units	Epochs	BatchSize	Validation	RMSE	MSE	MAE	Seconds
1h	tanh	adam	0.118	55	36	9	0.2	0.582	0.338	0.154	58.817
3h	tanh	adamax	0.118	55	36	9	0.2	0.584	0.341	0.165	58.063
6h	tanh	adamax	0.118	55	36	9	0.2	0.588	0.345	0.174	101.704
12h	tanh	adamax	0.405	40	68	45	0.1	0.603	0.363	0.200	96.051
1d	tanh	adamax	0.118	55	36	9	0.2	0.657	0.432	0.2568	204.052
3d	tanh	adadelta	0.118	55	36	9	0.2	0.969	0.9385	0.641	1631.155
7d	sigmoid	adamax	0.405	40	68	45	0.1	1.023	1.046	0.689	1643.763

Taula 5.1: *Resultats òptims de la xarxa neuronal recurrent GRU.*

En la Taula 5.2 es poden observar les prediccions de la RNN LSTM d'una capa. Segons la seqüència, franja de temps a fer la predicció, comença a incrementar-se la mètrica RMSE a partir de 3 dies, però s'ha de tenir en compte el temps d'execució a partir de les 3 hores comencen a incrementar-se considerablement. El temps més baix és de 27 segons i el més alt és de 57 minuts. Es pot veure que majoritàriament, la capa d'activació és la 'tanh', l'optimitzador és 'adamax' i el temps de validació és 0.2, han sortit en els millors resultats.

Sequence	Activation	Optimizer	DropOut	Units	Epochs	BatchSize	Validation	RMSE	MSE	MAE	Seconds
1h	tanh	adam	0.405	40	68	45	0.1	0.582	0.338	0.159	27.413
3h	tanh	adamax	0.118	55	36	9	0.2	0.586	0.343	0.169	105.727
6h	tanh	adamax	0.118	55	36	9	0.2	0.593	0.351	0.189	146.728
12h	tanh	adamax	0.405	40	68	45	0.1	0.659	0.434	0.259	72.936
1d	tanh	adamax	0.405	40	68	45	0.1	0.830	0.688	0.389	7 7.335
3d	tanh	adadelta	0.118	55	36	9	0.2	0.941	0.885	0.612	1523.291
7d	sigmoid	adamax	0.118	55	36	9	0.2	1.0230	1.046	0.689	3432.494

Taula 5.2: *Resultats òptims de la xarxa neuronal recurrent LSTM.*

En la Taula 5.3 es poden observar les prediccions de la RNN GRU de dues capes. Segons la seqüència, franja de temps a fer la predicció, comença a incrementar-se la mètrica RMSE a partir de 3 dies, però s'ha de tenir en compte el temps d'execució a partir de les 3 hores comencen a incrementar-se considerablement. El temps més baix és de 43 segons i el més alt és de 1,7 hores. Es pot veure que majoritàriament, la capa d'activació és la 'tanh', l'optimitzador és 'adamax' i el temps de validació és 0.1, han sortit en els millors resultats.

Sequence	Activation	Optimizer	DropOut	Units	Epochs	BatchSize	Validation	RMSE	MSE	MAE	Seconds	
1h	tanh	adam	0.118	0.243	45	43	30	0.2	0.574	0.329	0.157	43.191
3h	tanh	adamax	0.405	0.331	43	56	11	0.1	0.579	0.335	0.178	157.909
6h	tanh	adamax	0.118	0.243	45	43	30	0.2	0.583	0.340	0.190	145.750
12h	tanh	adamax	0.118	0.243	45	43	30	0.2	0.609	0.371	0.215	274.628
1d	tanh	adadelta	0.405	0.331	43	56	11	0.1	0.841	0.707	0.505	1747.804
3d	tanh	adadelta	0.405	0.331	43	56	11	0.1	0.923	0.852	0.562	4529.597
7d	sigmoid	adam	0.405	0.331	43	56	11	0.1	1.019	1.039	0.695	6280.207

Taula 5.3: *Resultats òptims de la xarxa neuronal recurrent GRU amb dues capes.*

En la Taula 5.4 es poden observar les prediccions de la RNN LSTM de dues capes. Segons la seqüència, franja de temps a fer la predicció, comença a incrementar-se la mètrica RMSE a partir de 3 dies, però s'ha de tenir en compte el temps d'execució a partir de les 3 hores comencen a incrementar-se considerablement. El temps més baix és de 27 segons i el més alt és de 57 minuts. Es pot veure que majoritàriament, la capa d'activació és la 'tanh', l'optimitzador és 'adamax' i 'adadelta' i el temps de validació és 0.2, han sortit en els millors resultats.

Sequence	Activation	Optimizer	DropOut	Units	Epochs	BatchSize	Validation	RMSE	MSE	MAE	Seconds	
1h	tanh	adam	0.118	0.243	45	43	30	0.2	0.573	0.329	0.155	40.125
3h	tanh	adamax	0.118	0.243	45	43	30	0.2	0.580	0.337	0.180	80.390
6h	tanh	adamax	0.118	0.243	45	43	30	0.2	0.591	0.349	0.207	127.388
12h	tanh	adamax	0.118	0.243	45	43	30	0.2	0.787	0.619	0.316	246.600
1d	tanh	adadelta	0.405	0.331	43	56	11	0.1	0.857	0.734	0.491	1,547.029
3d	tanh	adadelta	0.405	0.331	43	56	11	0.1	0.942	0.887	0.553	4,229.342
7d	tanh	adadelta	0.118	0.243	45	43	30	0.2	1.011	1.023	0.682	2,678.760

Taula 5.4: *Resultats òptims de la xarxa neuronal recurrent LSTM amb dues capes.*

En la Taula 5.5 es poden observar les prediccions de la DNN. Segons la seqüència, franja de temps a fer la predicció, els resultats són molt semblants a totes les franges de temps. El temps més baix és de 18 segons i el més alt és de 5 minuts. Es pot veure que majoritàriament, la capa d'activació és la 'sigmoid', l'optimitzador és 'adamax' i el temps de validació és 0.2, han sortit en els millors resultats.

Sequence	Activation	Optimizer	DropOut	Units	Epochs	BatchSize	Validation	RMSE	MSE	MAE	Seconds
1h	sigmoid	adamax	0.779	22	85	23	0.2	0.023	0.001	0.014	18.778
3h	sigmoid	adamax	0.779	22	85	23	0.2	0.032	0.001	0.020	22.037
6h	tanh	adamax	0.779	22	85	23	0.2	0.033	0.001	0.020	27.322
12h	sigmoid	adamax	0.779	22	85	23	0.2	0.041	0.002	0.025	29.853
1d	sigmoid	adam	0.779	22	85	23	0.2	0.043	0.002	0.026	39.955
3d	sigmoid	adamax	0.796	97	98	43	0.1	0.043	0.002	0.025	173.990
7d	tanh	adam	0.796	97	98	43	0.1	0.046	0.002	0.026	313.318

Taula 5.5: *Resultats òptims de la xarxa neuronal profunda.*

Detecció d'anomalies: En les xarxes neuronals recurrents, per determinar si el valor es un valor anòmal, s'utilitza la metodologia vista en l'apartat 4.3.4. S'ha realitzat per cada seqüència de temps i el model d'una capa i de dues capes la seva detecció d'anomalies, tant en les dades històriques com les dades recopilades durant el treball. L'entrenament obtingut pel sensor PM2.5, llavors s'ha testejat amb els 6 tipus de sensors: PM1, PM2.5, PM10, PM1 ATM, PM2.5 ATM i PM10 ATM, per poder comprovar si els són similars. En l'apartat 5.0.2.1 es podran veure resultats de mostra.

5.0.1.2 Isolation Forest

Per localitzar el model òptim per l'Isolation Forest, s'ha hagut de realitzar el procés de preparació de les dades de l'apartat 4.2.5 i la creació dels subconjunts d'entrenament i test de l'apartat 4.3.1. S'ha realitzat un procés de recerca segons els paràmetres de configuració del model, la variable 'max samples' i 'contamination'. S'ha executat el model per cada finestra de temps de predicció: 1h, 3h, 6h, 12h, 1d, 3d i 7d. En la Figura 5.4 es pot veure l'execució del model amb una finestra de temps de predicció de 1h.

```
# Import IsolationForest
from sklearn.ensemble import IsolationForest

outliers_fraction = 0.067
model =  IsolationForest(max_samples=6,contamination=outliers_fraction)
model.fit(principalDf.values)
```

Figura 5.4: Parametrització i configuració de L'Isolation Forest

Avaluació dels model: Per saber l'avaluació del model s'ha utilitzat la metodologia explicada en l'apartat 4.3.3, es tindrà en compte els valors atípics que determina el diagrama *box plot* per tenir un nombre teòric de valors atípics, i poder-los comparar amb els que determina el model, i així aconseguir la bondat del model. El propi model ja determina per ell mateix si una mostra és anòmala, com hem pogut veure en l'apartat 4.3.4.

Resultats dels models: En la Taula 5.6 es pot observar l'eficàcia del model. Es pot observar que aconsegueix una eficàcia en l'entrenament mínima del 94.55%, tenin en compte que aquesta accuracy s'ha mesurat a partir del nombre d'anomalies detectades pel *boxplot*.

Sequence	Count Outliers Train	Predict Outliers Train	Accuracy Train	Count Outliers Test	Predict Outliers Test	Accuracy Test
1h	275	260	94.55%	0	0	100%
3h	275	266	96.72%	0	3	-
6h	275	264	96%	0	0	-
12h	275	268	97.45%	0	1	-
1d	275	268	97.45%	0	0	100%
3d	275	268	97.45%	0	0	100%
7d	275	268	97.45%	0	1	-

Taula 5.6: *Resultats òptims de l'Isolation Forest del conjunt de dades històriques.*

Detecció d'anomalies: En l'Isolation Forest, per determinar si el valor es un valor anòmal, ho determina el mateix model, com s'ha vist en l'apartat 4.3.4. S'ha realitzat la detecció d'anomalies segons els dos components principals, com hem pogut veure en el capítol 4. En l'apartat 5.0.2.2 es poden veure el resultat de la visualització de la detecció d'anomalies.

5.0.2 Visualització de les anomalies

A continuació es poden veure els gràfics resultants de la visualització de les anomalies en la finestra de temps d'una hora de predicció. Les visualitzacions de la resta de finestres de temps de predicció, es poden visualitzar a l'apèndix A.

5.0.2.1 Xarxes Neuronals

En les Figures 5.5 i 5.6 es pot visualitzar les gràfiques de series de temps amb les anomalies detectades amb la RNN de tipus GRU d'una capa.

En la Figura 5.5, es pot observar que en la telemetria PM10, és on detecta un nombre més elevat d'anomalies. I a PM1 hi detecta menys anomalies que la resta. En aquest cas les telemetries de PM10 i PM2.5, tenen més similituds en la detecció d'anomalies. En la Figura 5.6, es pot observar que el model detecta més anomalies en PM1. També es pot observar el model detecta entre ells un patró similar d'anomalies.

Figura 5.5: Visualització d'anomalies de les dades històriques, RNN-GRU, freqüència 1h

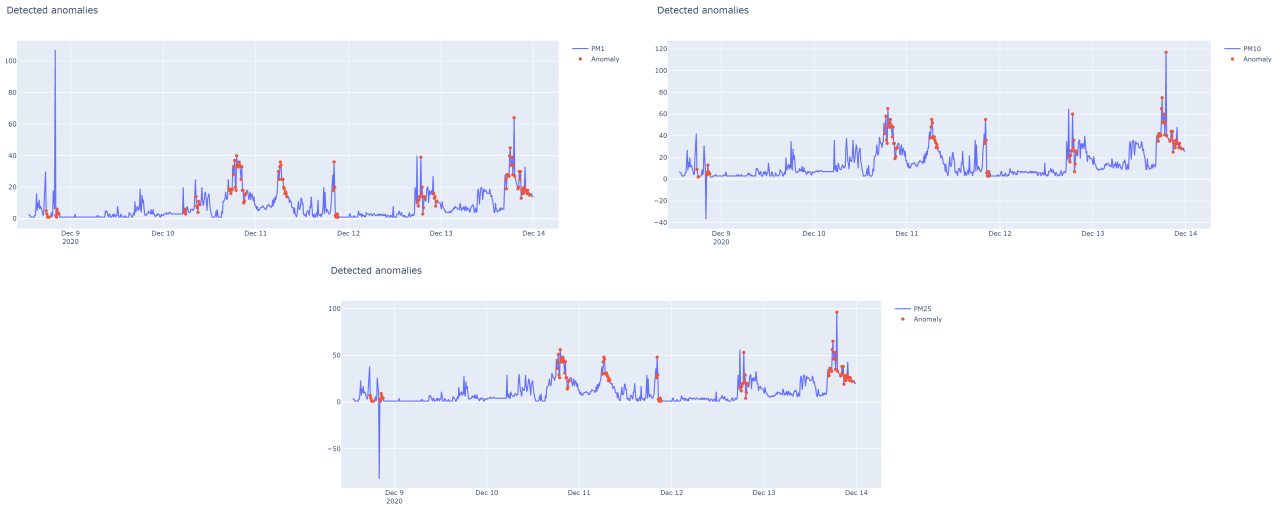


Figura 5.6: Visualització d'anomalies de les dades recopilades durant el treball, RNN-GRU, freqüència 1h

En les Figures 5.7 i 5.8 es pot visualitzar les gràfiques de series de temps amb les anomalies detectades amb la RNN de tipus LSTM d'una capa.

En la Figura 5.7, es pot observar que en la telemetria PM10, és on detecta un nombre més elevat d'anomalies. I a PM1 hi detecta menys anomalies que la resta. En aquest cas les telemetries de PM10 i PM2.5, tenen més similituds en la detecció d'anomalies. En la Figura 5.8, es pot observar que el model detecta més anomalies en PM1. També es pot observar el model detecta entre ells un patró similar d'anomalies.

Figura 5.7: Visualització d'anomalies de les dades històriques, RNN-LSTM, freqüència 1h

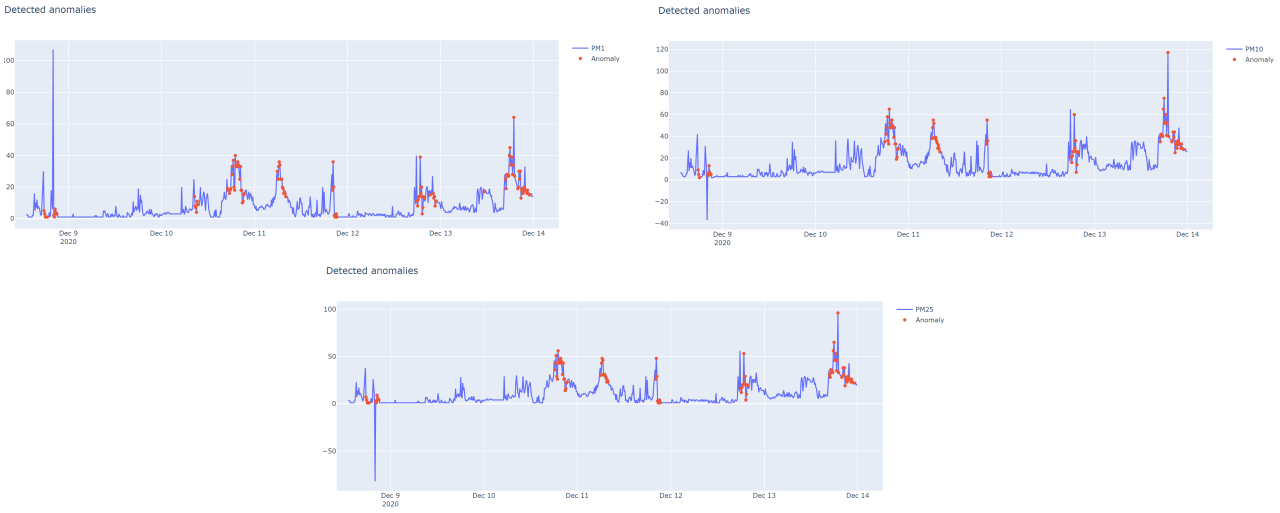


Figura 5.8: Visualització d'anomalies de les dades recopilades durant el treball, RNN-LSTM, freqüència 1h

En les Figures 5.9 i 5.10 es pot visualitzar les gràfiques de series de temps amb les anomalies detectades amb la DNN.

En la Figura 5.9 es pot observar que el model no detecta gens bé les anomalies en PM10 comparació PM1 i PM2.5. En PM2.5 detecta més anomalies que enn PM1.

En la Figura 5.10 es pot observar que PM10 i PM2.5, tenen un patró similar alhora de detectar les anomalies. I PM1 detecta menys anomalies.

Figura 5.9: Visualització d'anomalies de les dades històriques, DNN, freqüència 1h

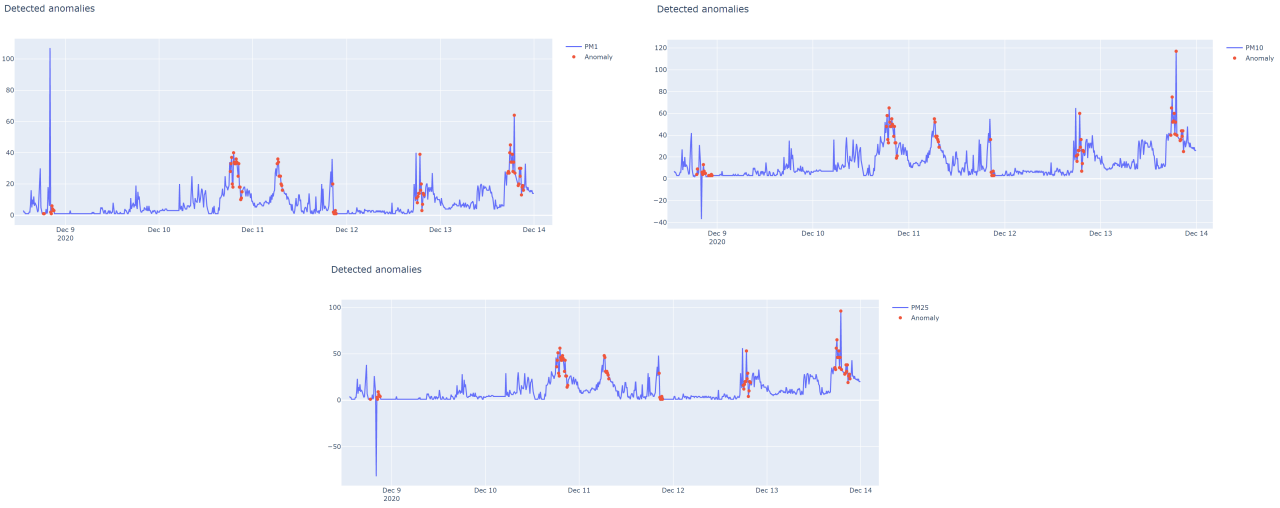


Figura 5.10: Visualització d'anomalies de les dades recopilades durant el treball, RNN-LSTM, freqüència 1h

5.0.2.2 Isolation Forest

En les Figures 5.11 i 5.12 es pot visualitzar les gràfiques de series de temps amb les anomalies detectades amb la Isolation Forest, del sensor PM2.5.

Es pot visualitzar que en les dades històriques, Figura 5.11, el model té una millor bondat alhora de detectar les anomalies. En canvi en les dades recopilades durant el treball, Figura 5.12, és pot observar que hi ha punts de dades que no les acaba de detectar.

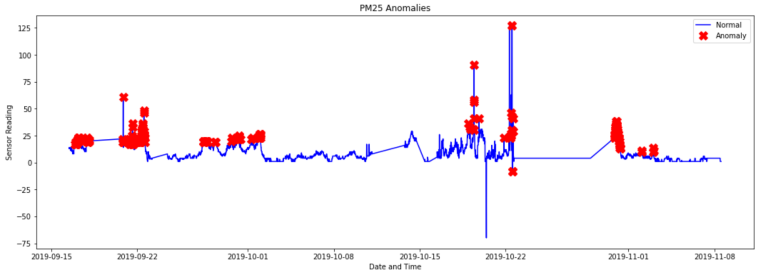


Figura 5.11: Anomalies detectades per l'Isolation Forest, en el conjunt de dades històriques del particulat PM2.5.

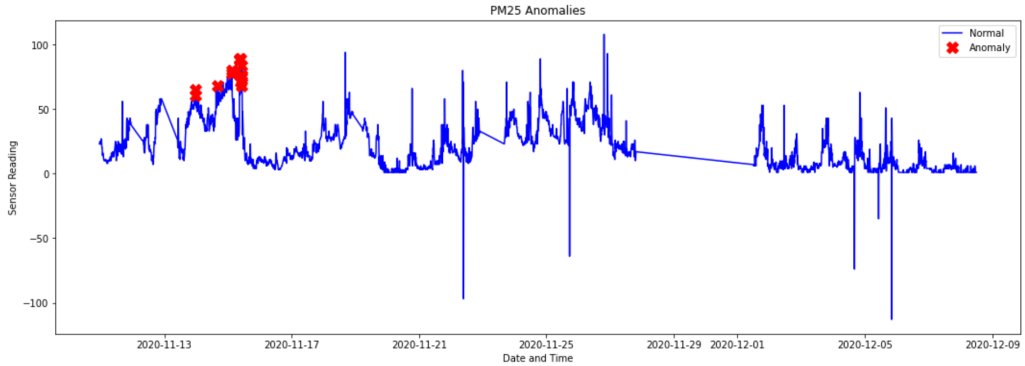


Figura 5.12: Anomalies detectades per l'Isolation Forest, en el conjunt de dades recollides durant el treball del particulat PM2.5.

5.1 Posta en producció dels models al cloud computing.

S'han realitzat dues REST-APIS, en un servidor web, amb el framework Django, com hem pogut veure en l'apartat 4.4, una per les xarxes neuronals i una altre per l'Isolation Forest, per dur a terme la predicció de les anomalies de les dades capturades en temps real mitjançant el dispositiu 'Argon' i el seu sensor 'Laser PM2.5 HM-3301 Dust Sensor'. Per dur a terme la predicció es requereixen 6 mostres de particulat de l'aire, en aquest cas de PM2.5, que els anirà recollint el dispositiu IoT, edge computing. Un cop tingui recopilades les 6 mostres, les enviarà per cada tipus de model, en el format Json, Figura 5.13, en les dues REST-APIS, perquè aquestes pugui realitzar la predicció per cada mesura enviada. La mateixa REST-API, un cop tingui calculades les prediccions, les guardarà en una fulla de càlcul de "Google Sheets"en temps real, com es pot veure en la Figura 5.15. Les REST-APIS, retornaran el resultat en el

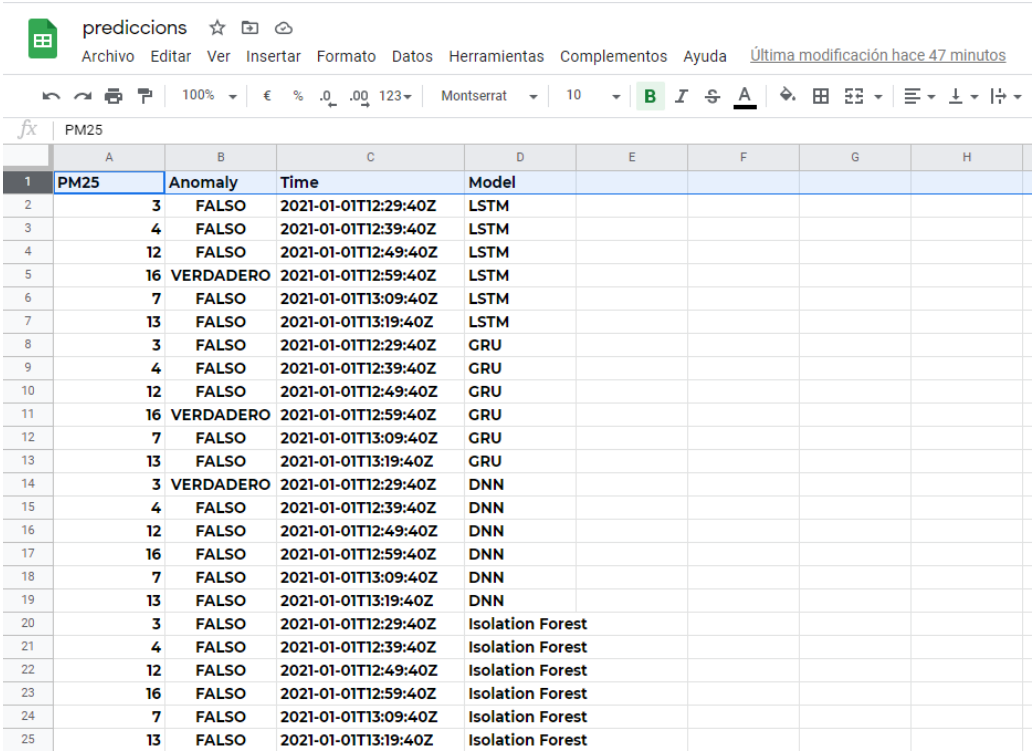
dispositiu, Figura 5.14, per si més endavant es vol realitzar algun tipus d'operació. En el codi del projecte, es pot visualitzar la fulla d'estils amb les prediccions que han realitzat en temps real.


```
{
"telemetry": [
    {"Time":"2020-12-19T08:46:02.271Z","PM25":225},
    {"Time":"2020-12-19T08:56:02.299Z","PM25":-100},
    {"Time":"2020-12-19T09:06:02.309Z","PM25":-175},
    {"Time":"2020-12-19T09:16:02.360Z","PM25":10},
    {"Time":"2020-12-19T09:26:02.325Z","PM25":175},
    {"Time":"2020-12-19T09:36:02.339Z","PM25":0}
],
"model":"GRU"
}
```


Figura 5.13: JSON d'exemple d'entrada cap als models


```
[{"PM25": 225.0, "Anomaly": true, "Time": "2020-12-19T08:46:02.271Z", "Model": "GRU"},
{"PM25": -100.0, "Anomaly": false, "Time": "2020-12-19T08:56:02.299Z", "Model": "GRU"},
{"PM25": -175.0, "Anomaly": false, "Time": "2020-12-19T09:06:02.309Z", "Model": "GRU"},
{"PM25": 10.0, "Anomaly": false, "Time": "2020-12-19T09:16:02.360Z", "Model": "GRU"},
{"PM25": 175.0, "Anomaly": false, "Time": "2020-12-19T09:26:02.325Z", "Model": "GRU"},
{"PM25": 0.0, "Anomaly": false, "Time": "2020-12-19T09:36:02.339Z", "Model": "GRU"}]
```


Figura 5.14: JSON d'exemple de sortida de les REST-APIs



	A	B	C	D	E	F	G	H
1	PM25	Anomaly	Time	Model				
2	3	FALSO	2021-01-01T12:29:40Z	LSTM				
3	4	FALSO	2021-01-01T12:39:40Z	LSTM				
4	12	FALSO	2021-01-01T12:49:40Z	LSTM				
5	16	VERDADERO	2021-01-01T12:59:40Z	LSTM				
6	7	FALSO	2021-01-01T13:09:40Z	LSTM				
7	13	FALSO	2021-01-01T13:19:40Z	LSTM				
8	3	FALSO	2021-01-01T12:29:40Z	GRU				
9	4	FALSO	2021-01-01T12:39:40Z	GRU				
10	12	FALSO	2021-01-01T12:49:40Z	GRU				
11	16	VERDADERO	2021-01-01T12:59:40Z	GRU				
12	7	FALSO	2021-01-01T13:09:40Z	GRU				
13	13	FALSO	2021-01-01T13:19:40Z	GRU				
14	3	VERDADERO	2021-01-01T12:29:40Z	DNN				
15	4	FALSO	2021-01-01T12:39:40Z	DNN				
16	12	FALSO	2021-01-01T12:49:40Z	DNN				
17	16	FALSO	2021-01-01T12:59:40Z	DNN				
18	7	FALSO	2021-01-01T13:09:40Z	DNN				
19	13	FALSO	2021-01-01T13:19:40Z	DNN				
20	3	FALSO	2021-01-01T12:29:40Z	Isolation Forest				
21	4	FALSO	2021-01-01T12:39:40Z	Isolation Forest				
22	12	FALSO	2021-01-01T12:49:40Z	Isolation Forest				
23	16	FALSO	2021-01-01T12:59:40Z	Isolation Forest				
24	7	FALSO	2021-01-01T13:09:40Z	Isolation Forest				
25	13	FALSO	2021-01-01T13:19:40Z	Isolation Forest				

Figura 5.15: Prediccions guardades en el Google Sheets

Capítol 6

Conclusions

En aquest treball, s'ha dut a terme un projecte de Machine Learning per la detecció i predicció d'anomalies en dispositius IoT en l'edge computing, en temps real. S'han desenvolupat quatre models de Machine Learning per la detecció i predicció d'anomalies en les dades (series temporals), mitjançant dades recopilades anteriorment pel mateix dispositiu IoT, per realitzar una detecció i predicció d'anomalies de les dades en el particulat de l'aire. El dispositiu IoT 'Particle Argon' mitjançant el sensor 'Laser PM2.5 HM-3301 Dust Sensor' captura dades del particulat de l'aire PM2.5, en temps real, cada 10 minuts, i un cop n'ha recopilat 6 observacions, s'envien en cadascun dels models per poder fer la detecció i predicció de les anomalies.

S'han realitzat dos tipus de xarxes neuronals recurrents (RNN), la Long Short Term Memory (LSTM) i la Gated Recurrent Unit (GRU), una xarxa neuronal profunda (DNN) i un Isolation Forest. Pels models de RNN, s'han preparat dues versions, models de dues capes i d'una capa. Els models d'una capa s'han dut a terme per poder-los incrustar dins del dispositiu, ja que aquest tipus de dispositius, tenen una memòria limitada. Els models de dues capes tenen una complexitat més elevada i el model generat supera la memòria del dispositiu.

S'han desenvolupat un parell de REST-APIS, una per les xarxes neuronals i l'altre per l'Isolation Forest, per ser cridades des del dispositiu. Les REST-APIS un cop tenen la predicció realitzada per cada observació, guarda en una fulla de càlcul de Google Sheets les prediccions. De la predicció, es guarda la observació, el temps en que ha estat capturada, el model que ha fet la predicció i si la observació ha estat predita anòmala.

Resultats obtinguts: Com hem pogut veure, les xarxes neuronals recurrents, tant la GRU

com la LSTM, els resultats han estat similars. El temps d'execució ha estat més ràpida la GRU, en seqüències de temps de predicció elevades. En la configuració que hem testejat, funcionen millor en finestres de temps de predicció curtes, per exemples 1h, 3h i 6h. Els resultats de les mètriques han estat similars. Això reflecteix alhora de detectar les anomalies en temps real, ja que es pot observar que detecten les mateixes anomalies. Cal destacar que en les proves realitzades no detectaven valors negatius, però s'ha vist que en la predicció en temps real si que detecten valors negatius.

En la detecció d'anomalies en les xarxes neuronals profundes, hem pogut veure que els resultats obtinguts alhora de fer la recerca dels models ha obtingut bons resultats segons la mètrica RMSE. En la detecció en temps real, s'ha vist que detectaven més anomalies que les RNN. Hem vist que detecta valors anòmals positius i negatius, i en més rang de números.

La detecció d'anomalies en l'Isolation Forest, hem vist que en les proves realitzades la detecció d'anomalies era bona, però quan hem posat el model dins del dispositiu, hem vist que no detecta cap anomalia. En comparació amb els altres models la rapidesa dels algorismes ha estat el més ràpid. i les prediccions han estat bones en la majoria de seqüències de temps de predicció.

Conclusions personals: Personalment, estic satisfet de la realització d'aquest treball, he pogut dur a terme els coneixaments adquirits durant el Màste. L'he trobat molt interessant, mai havia realitzat un projecte de Machine Learning de principi a fi: realitzar una recerca de l'estat de l'art, preparar les dades, desenvolupar els models escollits, avaluar-los, desenvolupar el model òptim pel microcontrolador i testejar el model en temps real. Durant el transcurs del treball, hi ha hagut algun imprevist, com el fet de no poder desenvolupar els models dins del microcontrolador, i m'ha fet prendre decisions ràpides per poder tirar endavant, i poder realitzar el treball, he hagut de muntar un servidor Linux i configurar-lo amb el framework Django i desenvolupar les REST-APIS. He escollit aquestes eines, perquè hi tenia una mica d'experiència.

El principi del desenvolupament, em vaig centrar amb els models de RNN de dues capes, però un cop vaig aconseguir exportar-los per dur-los a dins el dispositiu, vaig veure que ocupaven

massa espai, al voltant de 4 Mb. i això era massa, llavors vaig haver de simplificar els models i desenvolupar els models d'una capa. I un cop vaig tenir els models per posar-los a dins del microcontrolador, em vaig adonar que no seria possible tirar-lo endavant degut a la complexitat dels càlculs per dur-los dins del dispositiu. En les DNN, no hi havien preocupacions per la mida del model, però si per la complexitat per fer el càlcul de la predicció. L'Isolation Forest, no vaig arribar a intentar a desenvolupar-lo dins del dispositiu.

Referent als resultats, penso que es podrien millorar, perquè crec que amb els dos conjunts de dades, les històriques i les recopilades durant el treball, el fet que de la forma que s'han recopilat, com hem pogut veure en l'apartat 4.1, ha fet que els models no acabin d'estar del tot afinats, pel fet de que per l'obtenció dels models durant l'entrenament, les dades no han estat amb el mateix context que les de test, les dades recopilades durant el treball.

Penso que amb l'estat de l'art realitzat, m'ha ajudat alhora de decidir els algoritmes a utilitzar i les parametritzacions a testejar, com per exemple l'optimitzador ADM i la funció d'activació RELU en les capes intermitges. Hem vist que les RNN i DNN han donat bons resultats. I s'ha testejat l'Isolation Forest com a algoritme classic alhora de detectar anomalies en models no supervisats.

Línies futures: S'haurà de tenir en compte l'evolució del dispositiu particle Argon, per veure com va madurant el fet d'incorporar-hi més elements de Machine Learning, i poder-hi posar models més complexos, ja que amb el que hi ha actualment, no s'ha pogut aconseguir.

Com a idea, es podria realitzar el mateix estudi, però amb un dataset de dades més gran, i a ser amb dades separades, per exemple d'interior i d'exterior. I així veure com resoldrien les prediccions en cadascun dels casos.

Un altre punt a tenir en compte, seria fer un estudi multi-variant per veure el comportament de les dades, afegint nous sensors, com per exemple algun sensor de temperatura, d'humitat i estudiar com es comporta la detecció i anomalies en les dades en el particulat de l'aire.

Evidentment, es pot seguir aprofundint en la millora dels models proposats i un estudi més profund per veure si es pot aconseguir un model que no s'hagi d'esperar a tenir 6 registres, per veure si hi ha dades anòmales.

Bibliografia

[1] Adam. (Accedit: 17/10/2020).

[2] Amazon web services. (Accedit: 24/09/2020).

[3] Django. (Accedit: 15/12/2020).

[4] Documentació d'argon. (Accedit: 09/10/2020).

[5] Earlystopping. (Accedit: 9/12/2020).

[6] Edge computing. (Accedit: 04/10/2020).

[7] Edge computing. (Accedit: 04/10/2020).

[8] Google cloud. (Accedit: 24/09/2020).

[9] Grove - laser pm2.5 dust sensor. (Accedit: 21/09/2020).

[10] Gru. (Accedit: 11/12/2020).

[11] Integració ifttt. (Accedit: 15/11/2020).

[12] Internet de les coses. (Accedit: 21/09/2020).

[13] Internet of things. (Accedit: 21/09/2020).

[14] Isolation forest. (Accedit: 11/12/2020).

[15] Lstm. (Accedit: 11/12/2020).

[16] Microsoft azure. (Accedit: 24/09/2020).

[17] Particle docs math. (Accedit: 15/12/2020).

[18] Particle machine learning. (Accedit: 15/12/2020).

[19] Particle.io. (Accedit: 09/10/2020).

[20] Particulate matter sensor sps30. (Accedit: 21/09/2020).

[21] Principi de pareto. (Accedit: 7/12/2020).

[22] Prova de dickey-fuller. (Accedit: 8/12/2020).

[23] sklearn.ensemble.isolationforest. (Accedit: 10/12/2020).

[24] sklearn.ensemble.isolationforest-predict. (Accedit: 10/12/2020).

[25] Tensor flow lite. (Accedit: 7/10/2020).

[26] B. Hussain, Q. Du, S. Zhang, A. Imran, and M. A. Imran. Mobile edge computing-based data-driven deep learning framework for anomaly detection. *IEEE Access*, 7:137656–137667, 2019.

[27] D. Kim, H. Yang, M. Chung, S. Cho, H. Kim, M. Kim, K. Kim, and E. Kim. Squeezed convolutional variational autoencoder for unsupervised anomaly detection in edge device industrial internet of things. In *2018 International Conference on Information and Computer Technologies (ICICT)*, pages 67–71, 2018.

[28] C. Martín Fernández, M. Díaz Rodríguez, and B. Rubio Muñoz. An edge computing architecture in the internet of things. In *2018 IEEE 21st International Symposium on Real-Time Distributed Computing (ISORC)*, pages 99–102, 2018.

[29] Jaewon Moon, Seungwoo Kum, and Sangwon Lee. A heterogeneous iot data analysis framework with collaboration of edge-cloud computing: Focusing on indoor pm10 and pm2.5 status prediction. *Sensors*, 19(14), 2019.

[30] Mao V Ngo, Hakima Chaouchi, Tie Luo, and Tony QS Quek. Adaptive anomaly detection for iot data in hierarchical edge computing. *arXiv preprint arXiv:2001.03314*, 2020.

[31] Yuhuai Peng, Aiping Tan, Jingjing Wu, and Yuanguo Bi. Hierarchical edge computing: A novel multi-source multi-dimensional data anomaly detection scheme for industrial internet of things. *IEEE Access*, 7:111257–111270, 2019.

[32] Joseph Schneible and Alex Lu. Anomaly detection on the edge. In *MILCOM 2017-2017 IEEE Military Communications Conference (MILCOM)*, pages 678–682. IEEE, 2017.

[33] B. Sharma, L. Sharma, and C. Lal. Anomaly detection techniques using deep learning in iot: A survey. In *2019 International Conference on Computational Intelligence and Knowledge Economy (ICCIKE)*, pages 146–149, 2019.

[34] PhD Tantawi, Randa. Machine learning. *Salem Press Encyclopedia*, 2020.

[35] Sergio Trilles, Ana Belen Vicente, Pablo Juan, Francisco Ramos, Sergi Meseguer, and Laura Serra. Reliability validation of a low-cost particulate matter iot sensor in indoor and outdoor environments using a reference sampler. *Sustainability*, 11(24), 2019.

[36] Darmawan Utomo and Pao-Ann Hsiung. A multitiered solution for anomaly detection in edge computing for smart meters. *Sensors*, 20(18):5159, 2020.

Apèndix A

Visualització d'anomalies

A continuació es poden veure altres visualitzacions destacades que sobre la detecció d'anomalies que no s'han incorporat en el cos del treball.

GRU

A continuació es poden veure les gràfiques de la xarxa neuronal recurrent de tipus GRU.

En la Figura A.1 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

Figura A.1: Visualització d'anomalies de les dades històriques, RNN-GRU, freqüència 1d

En la Figura A.2 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

Figura A.2: Visualització d'anomalies de les dades recopilades durant el treball, RNN-GRU, freqüència 1d

En la Figura A.3 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants, dins el canvi que es visualitza que cadascun d'ells la serie temporal varia.

Figura A.3: Visualització d'anomalies de les dades històriques, RNN-GRU, freqüència 3h

En la Figura A.4 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

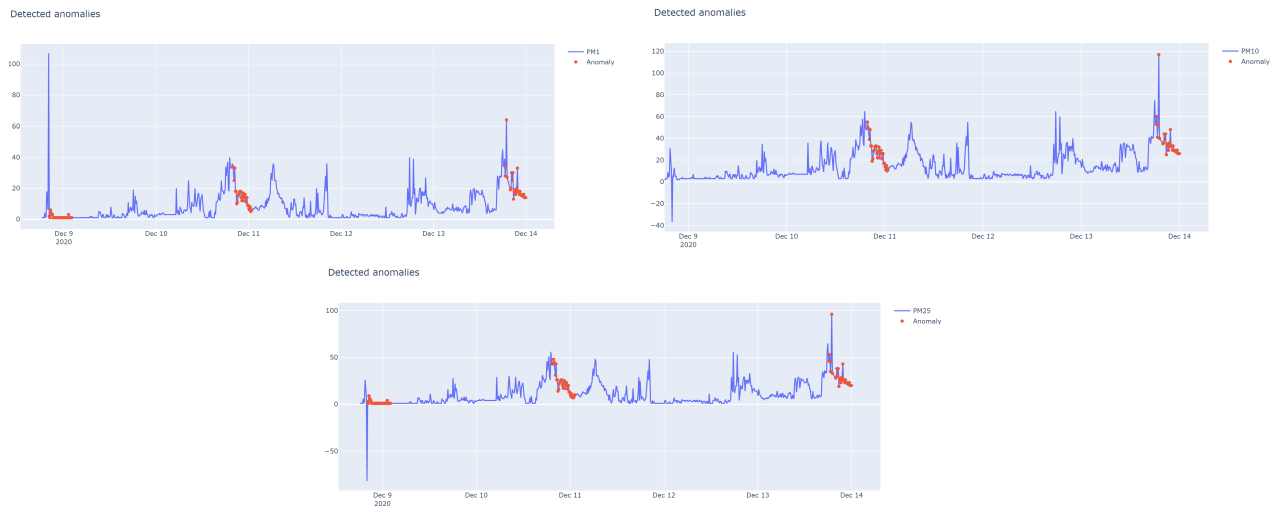


Figura A.4: Visualització d'anomalies de les dades recopilades durant el treball, RNN-GRU, freqüència 6h

En la Figura A.5 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions d'anomalies varien entre el PM1 i la resta.

Figura A.5: Visualització d'anomalies de les dades històriques, RNN-GRU, freqüència 12h

En la Figura A.6 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions d'anomalies varien entre el PM1 i la resta, degut a que la seva serie temporal difereix bastant a la resta.

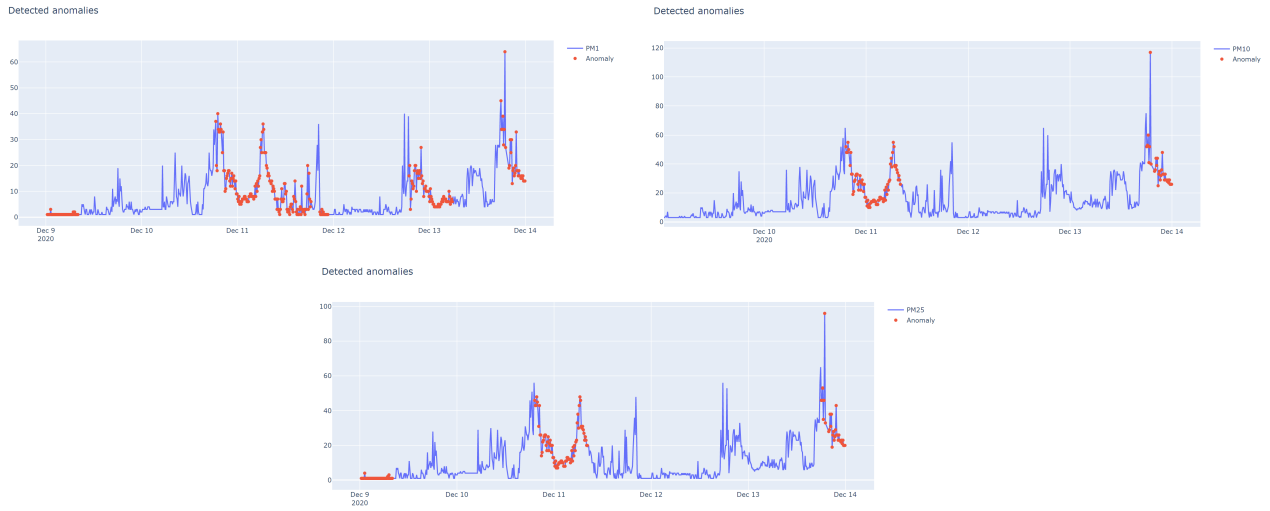


Figura A.6: Visualització d'anomalies de les dades recopilades durant el treball, RNN-GRU, freqüència 12h

En la Figura A.7 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions d'anomalies varien entre el PM2.5 i la resta, degut a que la seva serie temporal difereix bastant a la resta.

Figura A.7: Visualització d'anomalies de les dades històriques, RNN-GRU, freqüència 12h

En la Figura A.8 es pot veure que la detecció d'anomalies entre els diferents particulats, són diferents, ja que tenen una serie temporal diferents entre elles.

Figura A.8: Visualització d'anomalies de les dades recopilades durant el treball, RNN-GRU 2 capes, freqüència 1d

En la Figura A.9 es pot veure que la detecció d'anomalies entre els diferents particulats, és similar.

Figura A.9: Visualització d'anomalies de les dades històriques, RNN-GRU, freqüència 12h

En la Figura A.10 es pot veure que no hi ha hagut detecció d'anomalies.

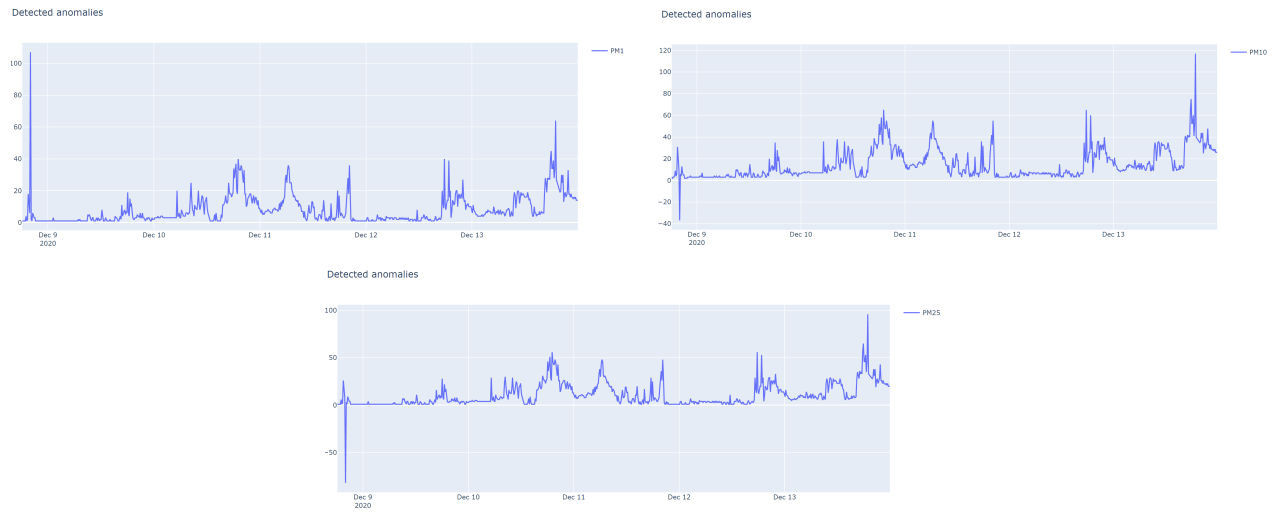


Figura A.10: Visualització d'anomalies de les dades recopilades durant el treball, RNN-GRU 2 capes, freqüència 6h

En la Figura A.11 es pot veure que la detecció d'anomalies és diferent entre els diferents particulats.

Figura A.11: Visualització d'anomalies de les dades històriques, RNN-GRU 2 capes, freqüència 3d

En la Figura A.18 es pot veure que la detecció d'anomalies és diferent entre els diferents particulats.

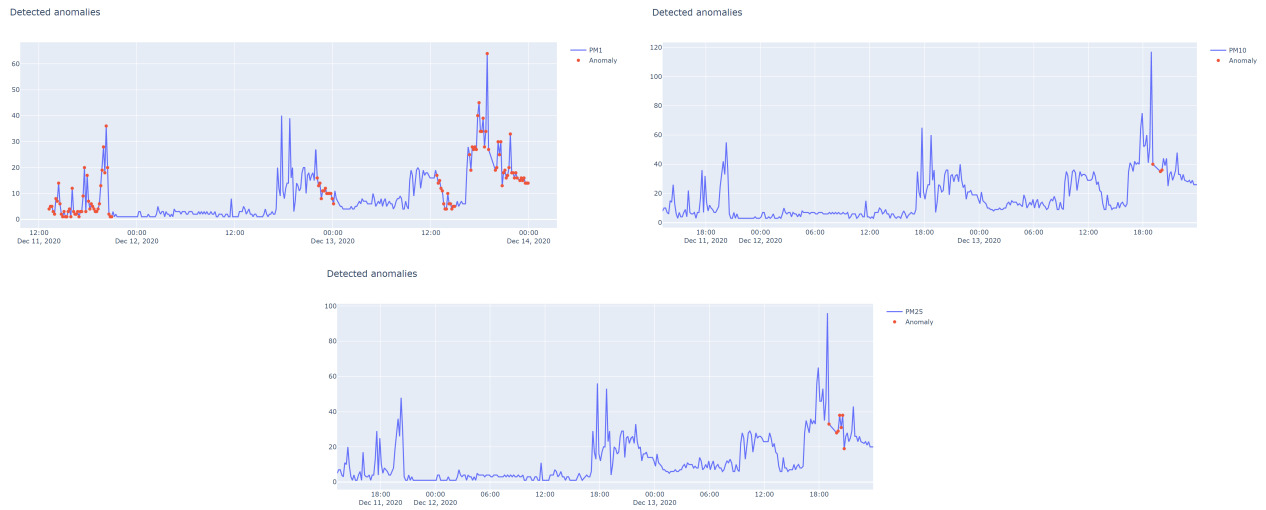


Figura A.12: Visualització d'anomalies de les dades recopilades durant el treball, RNN-GRU 2 capes, freqüència 3d

LSTM

A continuació es poden veure les gràfiques de la xarxa neuronal recurrent de tipus LSTM.

En la Figura A.13 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

Figura A.13: Visualització d'anomalies de les dades històriques, RNN-LSTM, freqüència 1d

En la Figura A.14 es pot veure que la detecció d'anomalies entre els diferents particulats són diferents.

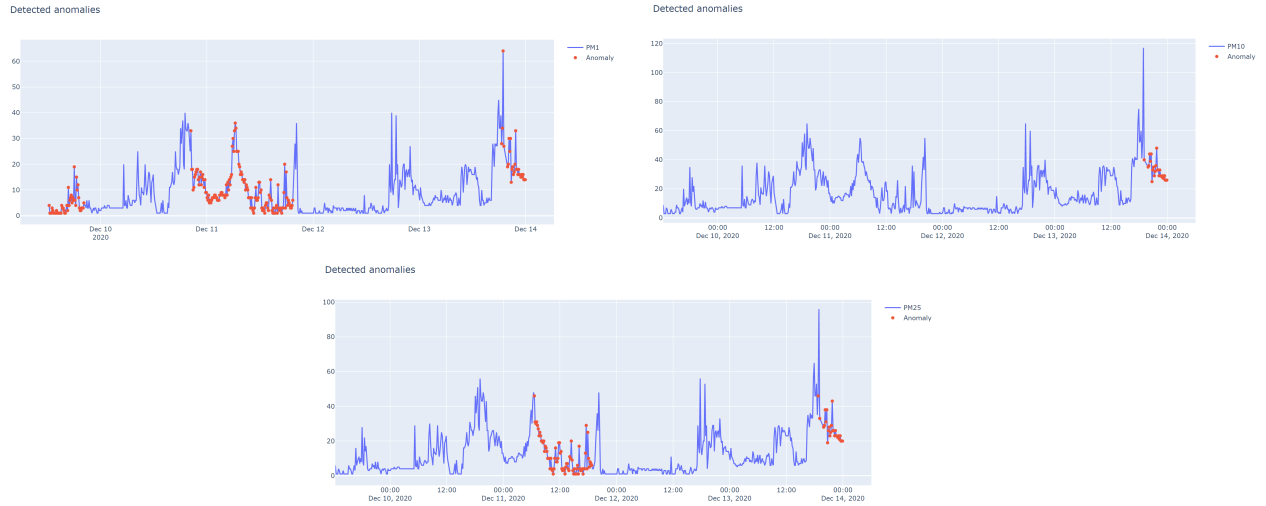


Figura A.14: Visualització d'anomalies de les dades recopilades durant el treball, RNN-LSTM, freqüència 1d

En la Figura A.15 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

Figura A.15: Visualització d'anomalies de les dades històriques, RNN-LSTM, freqüència 6h

En la Figura A.16 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

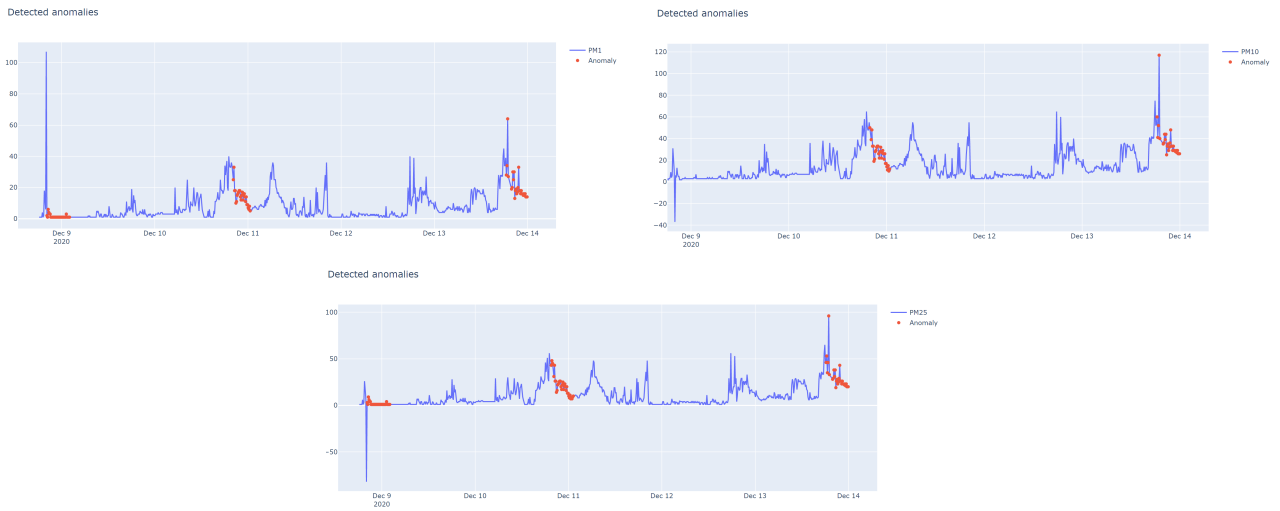


Figura A.16: Visualització d'anomalies de les dades recopilades durant el treball, RNN-LSTM, freqüència 6h

En la Figura A.17 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

Figura A.17: Visualització d'anomalies de les dades històriques, RNN-LSTM, freqüència 6d

En la Figura A.18 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són diferents.

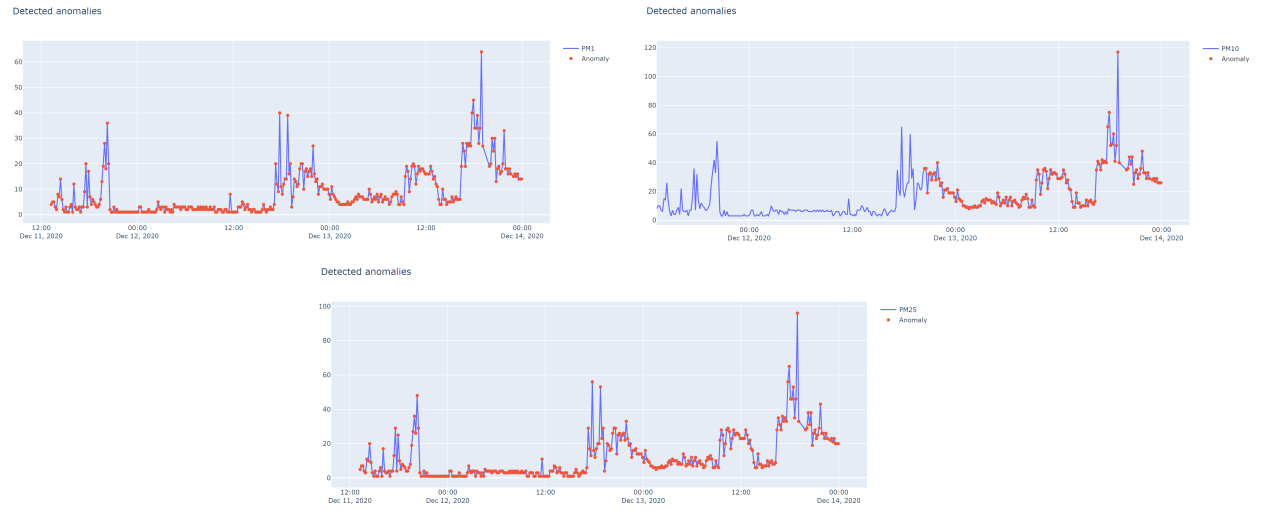


Figura A.18: Visualització d'anomalies de les dades recopilades durant el treball, RNN-LSTM, freqüència 3d

En la Figura A.19 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

Figura A.19: Visualització d'anomalies de les dades històriques, RNN-LSTM, freqüència 1d

En la Figura A.20 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són diferents.

Figura A.20: Visualització d'anomalies de les dades recopilades durant el treball, RNN-LSTM 2 capes, freqüència 1d

En la Figura A.21 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són similars.

Figura A.21: Visualització d'anomalies de les dades històriques, RNN-LSTM 2 capes, freqüència 6h

En la Figura A.22 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són diferents, degut a les series temporals que són diferents.

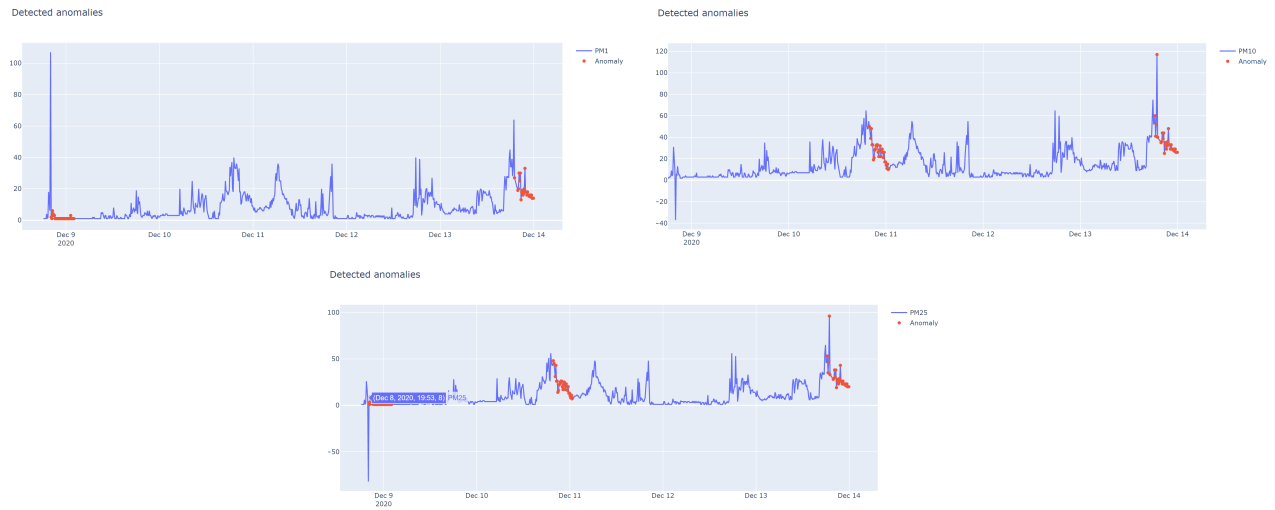


Figura A.22: Visualització d'anomalies de les dades recopilades durant el treball, RNN-LSTM 2 capes, freqüència 6h

En la Figura **??** es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són similars.

Figura A.23: Visualització d'anomalies de les dades històriques, RNN-LSTM 2 capes, freqüència 6d

En la Figura A.22 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són diferents, degut a les series temporals que són diferents.

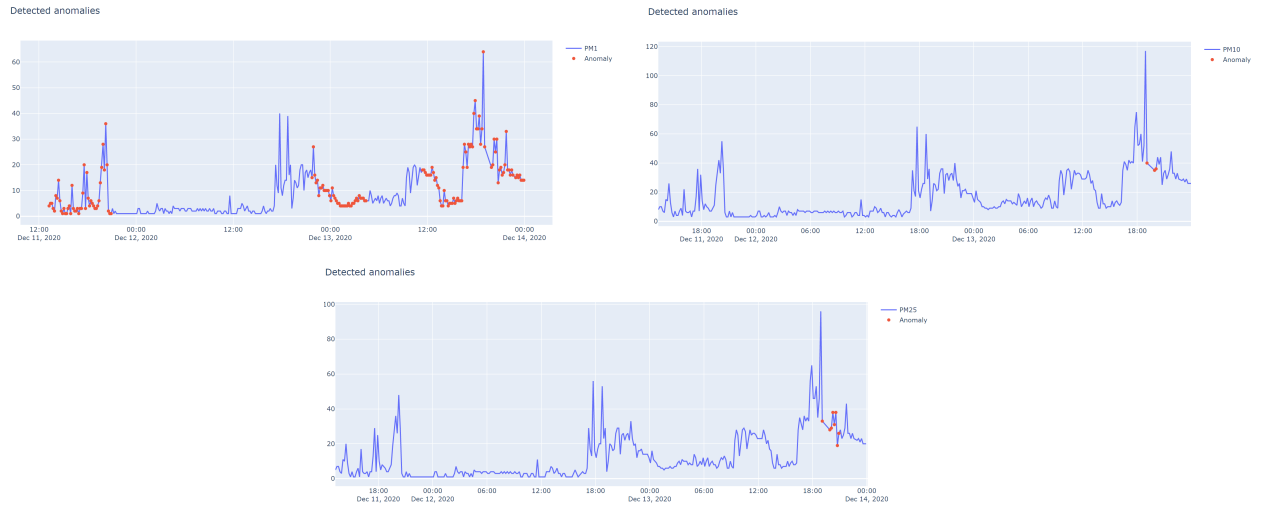


Figura A.24: Visualització d'anomalies de les dades recopilades durant el treball, RNN-LSTM 2 capes, freqüència 3d

DNN

A continuació es poden veure les gràfiques de la xarxa neuronal profundes.

En la Figura A.25 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

Figura A.25: Visualització d'anomalies de les dades històriques, DNN, freqüència 1d

En la Figura A.26 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

Figura A.26: Visualització d'anomalies de les dades recopilades durant el treball, DNN, freqüència 1d

En la Figura A.27 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

Figura A.27: Visualització d'anomalies de les dades històriques, DNN, freqüència 3h

En la Figura A.28 es pot veure que la detecció d'anomalies entre els diferents particulats, les deteccions són semblants.

Figura A.28: Visualització d'anomalies de les dades recopilades durant el treball, DNN, freqüència 6h

En la Figura A.29 es pot veure que no hi ha hagut detecció d'anomalies.

Figura A.29: Visualització d'anomalies de les dades històriques, DNN, freqüència 6d

En la Figura A.30 es pot veure que les deteccions d'anomalies entre els particulats és diferent.

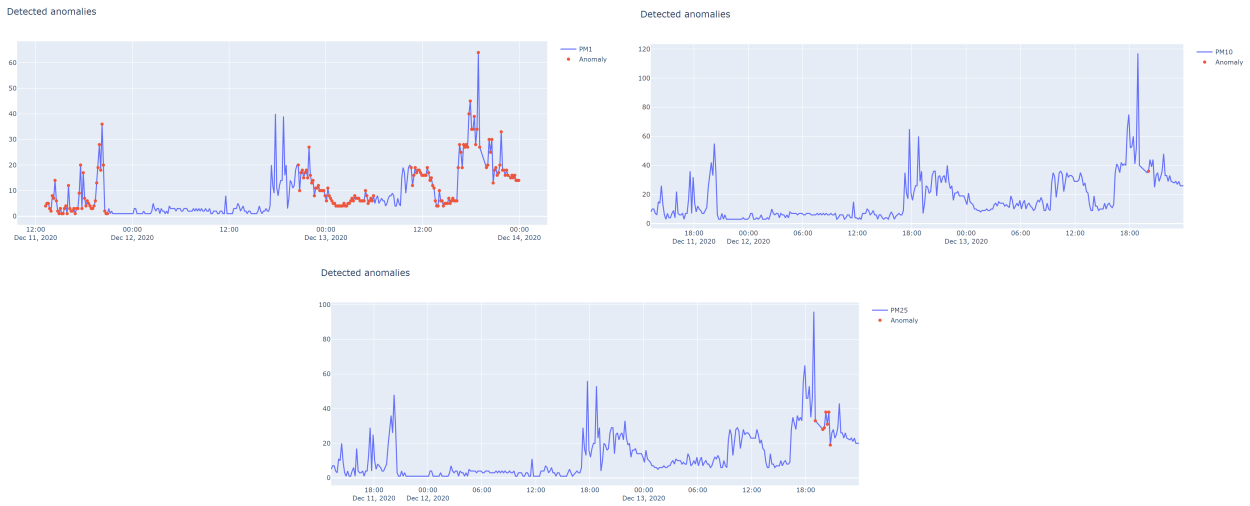


Figura A.30: Visualització d'anomalies de les dades recopilades durant el treball, DNN, freqüència 6d

Isolation Forest

A continuació es poden veure les gràfiques de l'Isolation Forest.

En la Figura A.31 es pot veure que la detecció d'anomalies en el subconjunt d'entrament agafa molts rangs, i aparentment no és correcte. En el conjunt de test no detecta anomalies.

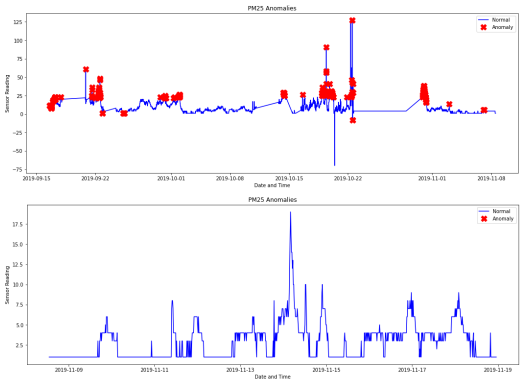


Figura A.31: Visualització d'anomalies de les dades històriques, Isolation Forest, freqüència 1d

En la Figura A.32 es pot veure que la detecció d'anomalies en el subconjunt d'entrament agafa molts rangs, i aparentment no és correcte. En el conjunt de test detecta anomalies no correctes.

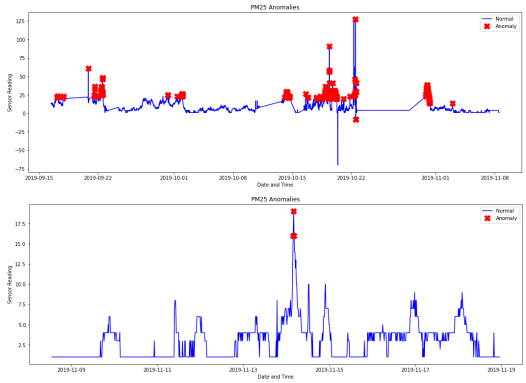


Figura A.32: Visualització d'anomalies de les dades recopilades durant el treball, Isolation Forest, freqüència 1d

En la Figura A.33 es pot veure que la detecció d'anomalies en el subconjunt d'entrament agafa molts rangs, i aparentment no és correcte. En el conjunt de test detecta anomalies no correctes.

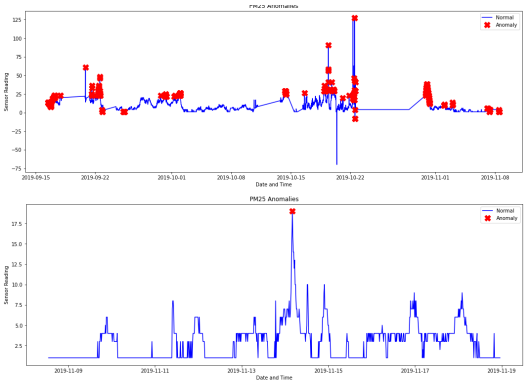


Figura A.33: Visualització d'anomalies de les dades històriques, Isolation Forest, freqüència 3h

En la Figura A.34 es pot veure que la detecció d'anomalies en el subconjunt d'entrament aparentment és correcte i en el subconjunt de test també és correcte.

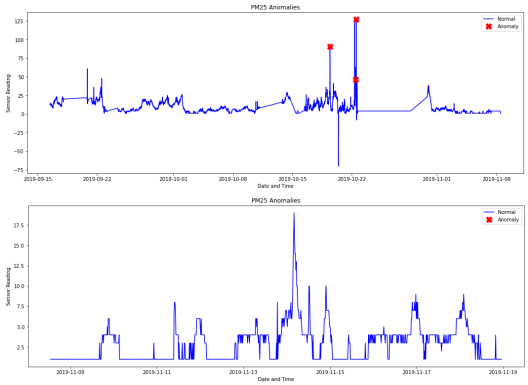


Figura A.34: Visualització d'anomalies de les dades recollides durant el treball, Isolation Forest, freqüència 1d

En la Figura A.34 es pot veure que la detecció d'anomalies en el subconjunt d'entrament aparentment és correcte, però amb poc precisió i en el subconjunt de test també és correcte.

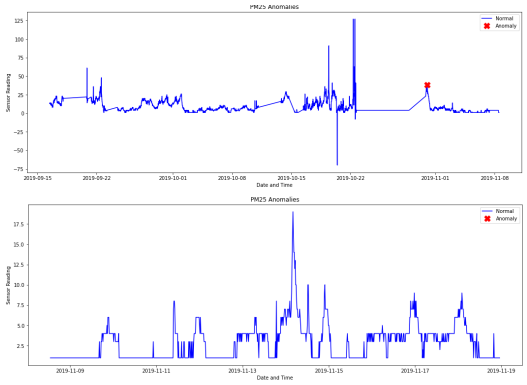


Figura A.35: Visualització d'anomalies de les dades recopilades durant el treball, Isolation Forest, freqüència 1d

En la Figura A.34 es pot veure que la detecció d'anomalies en el subconjunt d'entrament aparentment és correcte i en el subconjunt de test també és correcte.

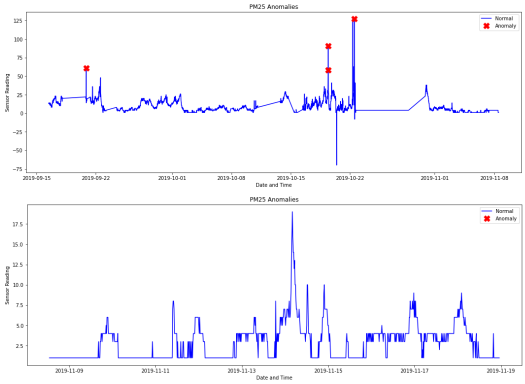


Figura A.36: Visualització d'anomalies de les dades històriques, Isolation Forest, freqüència 7d