Universitatea Națională de Știință și Tehnologie
POLITEHNICA București
Facultatea Automatică și Calculatoare

PSITR

Semestrul 2, 2024-2025

Tema de casă

Optimizarea calității aerului folosind date în timp real despre mediu

Profesor: Monica Drăgoicea

Studenți: Grigore Vlad-Gabriel
Din Andrei-Iulian
Grupa: 341B2
E-mail: vladgrigore55@gmail.com
diniulian63@gmail.com

Cuprins

1. Prezentarea generala a temei

Proiectul propune dezvoltarea unui sistem de monitorizare și analiză în timp real a calității aerului în spații închise, prin utilizarea datelor provenite de la senzori ambientali. Tema este deosebit de relevantă în contextul actual, în care optimizarea confortului și sănătății în mediile interioare — precum sălile de clasă, birourile sau alte spații colective — reprezintă o prioritate.

Scopul lucrării este de a evidenția modul în care datele colectate (precum temperatura, umiditatea, calitatea aerului, numărul de persoane, starea geamurilor și a ventilației) pot fi prelucrate și transformate în informații relevante, care, la rândul lor, generează cunoștințe utile și susțin luarea unor decizii automate sau asistate.

Această abordare reflectă aplicarea fluxului **date → informație → cunoștințe**, esențial în cadrul sistemelor moderne de procesare a informației în timp real, așa cum este prezentat și în curs. Sistemul dezvoltat colectează date din mediul înconjurător, le interpretează pentru a evalua condițiile de confort și construiește un model capabil să ia decizii automate (de exemplu, pornirea ventilației) sau să emită recomandări (precum deschiderea geamului).

Importanța unei astfel de soluții constă în capacitatea de a optimiza simultan **calitatea aerului** și **eficiența energetică**, contribuind la menținerea unui mediu interior mai sănătos și sustenabil. În plus, pe termen lung, cunoștințele obținute pot sprijini și la luarea de decizii strategice, precum alegerea unui domiciliu într-o zonă cu aer mai curat sau elaborarea unor politici de gestionare inteligentă a factorilor de mediu.

2. Descriere problema

Calitatea aerului în spațiile închise constituie un factor esențial în asigurarea sănătății și confortului utilizatorilor. Parametri precum temperatura, umiditatea relativă și concentrația poluanților variază în timp, fiind influențați de factori dinamici precum prezența umană, deschiderea sau închiderea geamurilor, funcționarea sistemelor de ventilație și condițiile de mediu exterior. În absența unui mecanism de monitorizare continuă și de reacție automată, aceste variații pot genera disconfort, acumularea de compuși nocivi sau pierderi energetice semnificative.

În acest context, proiectul propune dezvoltarea unui sistem de procesare a informației în timp real, capabil să preia, interpreteze și convertească datele ambientale brute în informații și cunoștințe relevante pentru optimizarea deciziilor automate sau asistate. Structura acestui sistem urmează modelul logic „date → informație → cunoștințe”, prezentat în cadrul Cursurilor 1 și 2, care oferă o bază conceptuală solidă pentru proiectarea aplicațiilor inteligente cu caracter adaptiv.

Componentele sistemului:

1. **Date (nivel primar):** Reprezintă valorile brute colectate de la senzori specifici și alți factori contextuali:

- Semnal analogic furnizat de senzorul MQ-135, utilizat pentru estimarea calității aerului;
- Valori de temperatură și umiditate oferite de senzorul DHT11;
- Metadate precum: ora măsurării (timestamp), starea geamurilor (deschis/închis), numărul de persoane din spațiu, starea sistemului de ventilație, precum și eventuale condiții externe (ex. nivelul de trafic).

2. **Informație (nivel intermediar):** Etapa în care datele sunt procesate: sincronizate temporal, validate, scalate și organizate în structuri analitice (de ex. tabele, DataFrame-uri Pandas, baze de date online precum ThingSpeak). Acest nivel facilitează:

    - identificarea corelațiilor între variabile,

- urmărirea tendințelor în timp,
- detectarea modificărilor semnificative în mediul monitorizat.

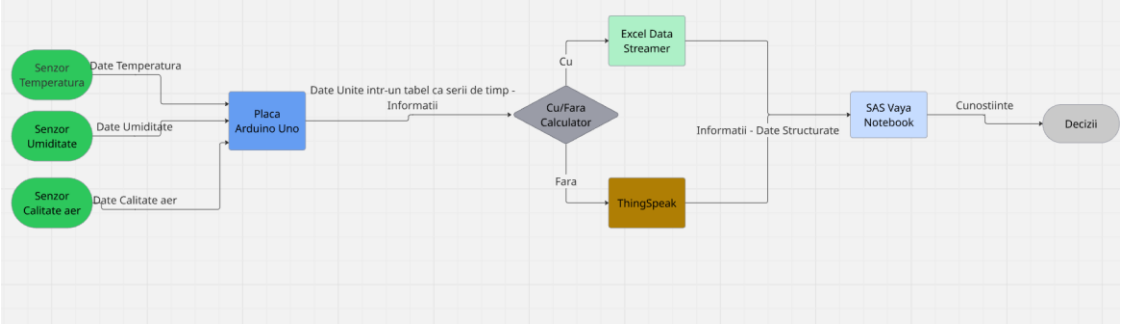
3. **Cunoștințe (nivel decizional):** Rezultatul interpretării informațiilor, orientat spre luarea unor decizii autonome sau asistate. Exemple:
   - Agregarea valorilor pe intervale temporale (orare, zilnice);
   - Corelarea condițiilor (ex.: temperaturi ridicate → umiditate scăzută → aer perceput ca fiind mai „curat”);
   - Detecția automată a anomaliilor și generarea de alerte;
   - Formularea de reguli acționabile, cum ar fi:

     „Dacă temperatura depășește 26°C, calitatea aerului este slabă, iar geamul este închis → activează ventilația automată.”

     „Dacă umiditatea depășește 70% în timp ce geamul este deschis → generează alertă pentru închiderea acestuia.”

Sistemul propus funcționează conform unei arhitecturi de tip streaming, în care datele sunt colectate, analizate și corelate în timp real, iar reacțiile sunt declanșate într-un interval de timp minim. Acest tip de abordare este optim pentru aplicațiile ce vizează mediul interior, caracterizat prin dinamism ridicat și nevoia de intervenții rapide, personalizate și eficiente.

În continuare este ilustrată schematic o variantă simplificată a fluxului de funcționare al sistemului, pentru a facilita înțelegerea etapelor de procesare a informației:



3. Colectarea datelor

Pentru realizarea analizei în timp real a calității aerului, a fost utilizat un prototip portabil construit pe baza platformei Arduino Uno, integrat cu un modul de comunicație ESP8266, conectat prin interfața SoftwareSerial (pinii D2 și D3). Structura hardware este completată de senzorul MQ-135 (conectat pe pinul analogic A0), utilizat pentru detecția compușilor poluanți volatili, respectiv senzorul DHT11 (pin D4), destinat măsurării temperaturii și umidității relative.

Componenta software include un firmware personalizat care inițializează conexiunea la rețea prin biblioteca WiFiEsp, cu o secvență de retry automat pe trei rețele predefinite (hotspot mobil al UPB, rețeaua domestică MERCUSYS și rețeaua alternativă FANTOMAS_5G), fiecare cu un timp de așteptare de 10 secunde. Achiziția datelor este realizată periodic, la un interval de 20 de secunde, simulând funcționalitatea unui *software timer* specific mediului FreeRTOS, conform recomandărilor din cadrul cursului de PATR (anul III).

Fiecare ciclu de achiziție implică:

- afișarea valorilor senzorilor pe interfața Serial Monitor;
- transmiterea acestora către platforma ThingSpeak, utilizând metodele ThingSpeak.setField() și ThingSpeak.writeFields();
- închiderea conexiunii TCP (client.stop()) și reconfigurarea acesteia pentru următorul ciclu de transmisie.

Datele sunt stocate într-un canal privat ThingSpeak (ID: 2945552) și pot fi exportate în format CSV pentru analize ulterioare. Această arhitectură permite o achiziție fiabilă și coerentă a datelor în timp real, inclusiv în condiții de rețea instabilă sau alimentare variabilă.

Sketch-ul complet este prezentat în *Listing A.1*, iar detaliile de implementare se regăsesc în *Anexa A.*

4. Explorarea datelor

În vederea fundamentării unor decizii automatizate privind controlul condițiilor ambientale, s-a realizat o analiză exploratorie a datelor colectate dintr-un set de locații distincte. Scopul a fost transformarea unui set brut de măsurători într-un ansamblu coerent de informații, capabil să susțină dezvoltarea unui sistem inteligent de monitorizare.

Setul de date analizat a fost preluat din platforma ThingSpeak și a fost încărcat în mediul Python prin intermediul bibliotecii pandas. Structura acestuia cuprinde peste 600 de observații, înregistrate în timp real, fiecare asociată unui marcaj temporal (Timestamp) și completată cu variabile descriptive și numerice, precum temperatura, umiditatea, calitatea aerului (ppm), numărul de persoane, starea geamurilor, activarea ventilației și alți factori de context (ex. tip incintă, vreme).

O primă etapă a constat în **evaluarea structurii tabelare și a tipurilor de date** aferente fiecărei variabile. Această examinare preliminară a relevat necesitatea conversiei unor coloane, în special a celor înregistrate sub formă de text (de exemplu, temperatura exprimată cu virgulă în loc de punct zecimal). Ulterior, au fost aplicate conversii riguroase pentru a aduce valorile la tipuri numerice corecte (float64) și pentru a asigura compatibilitatea cu metodele statistice și grafice ulterioare. De asemenea, marcajele temporale au fost transformate într-un format standard datetime64[ns, UTC], iar pentru facilitarea analizelor pe oră sau pe zi, s-au generat câmpuri suplimentare care separă data de ora exactă.

Procesul de curățare a datelor a fost completat prin redenumirea variabilelor pentru o mai bună lizibilitate (ex. „Temperatura (grad C)” a devenit „Temperatura”) și prin verificarea integrității datasetului — fără valori lipsă sau erori de conversie rămase după transformare.

În etapa de **analiză univariată**, au fost generate reprezentări grafice precum histograme și boxploturi pentru principalele variabile cantitative. De exemplu, distribuția temperaturii (Figura 1) arată o concentrare a valorilor între 24°C și 27°C, cu o ușoară asimetrie. Distribuția umidității (Figura 2) este moderată, oscilând între 40% și 60%, iar în cazul calității aerului (Figura 3), s-au identificat atât valori medii frecvente, cât și episoade izolate de poluare accentuată, care pot indica acumulări de compuși nocivi în lipsa ventilației.

Analiza **corelațiilor** între variabile (Figura 4) a scos în evidență o legătură semnificativă între temperatura ambientală, umiditate și calitatea aerului. De exemplu, s-a observat că valorile ridicate ale temperaturii sunt uneori corelate cu o scădere a calității aerului, în special în absența unei ventilații active. Această relație este vizibilă și în reprezentarea grafică a temperaturii versus calitatea aerului, diferențiată în funcție de starea sistemului de ventilație (Figura 5).

Pentru a evalua impactul variabilelor categorice asupra celor numerice, au fost realizate analize comparative (de tip barplot), care au confirmat că activarea ventilației contribuie constant la

îmbunătățirea calității aerului, în timp ce deschiderea geamurilor are un impact mai variabil, posibil influențat de factori externi, precum condițiile meteorologice și sursele de poluare din exterior.

În concluzie, analiza exploratorie sugerează că **activarea ventilației** este principalul determinant al calității aerului în spațiile închise, în timp ce alte variabile (precum temperatura sau deschiderea geamurilor) pot avea un efect mediat de contextul local. Diversitatea profilelor de corelație între locații confirmă nevoia unui model de tip adaptiv, personalizat în funcție de specificul fiecărui spațiu monitorizat, și nu a unui model general aplicabil tuturor.

Acest proces validat de analiză și curățare a datelor creează premisele pentru integrarea datasetului într-un pipeline robust de tip machine learning, fie pentru clasificarea condițiilor de confort, fie pentru predicția valorilor viitoare ale parametrilor de interes.

Figura 1: Distribuția valorilor temperaturii.

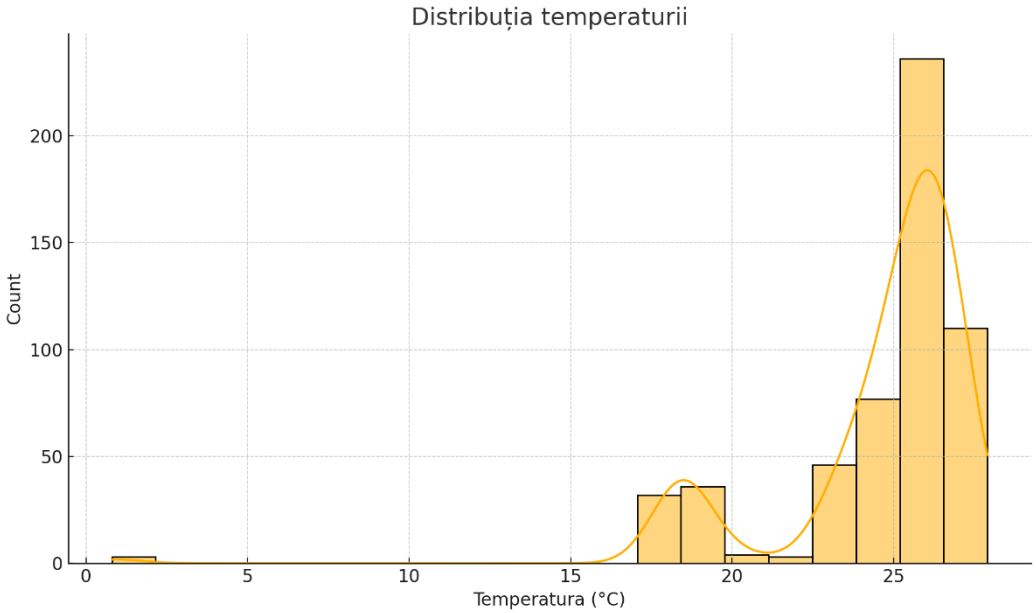


Figura 2: Distribuția valorilor umidității.

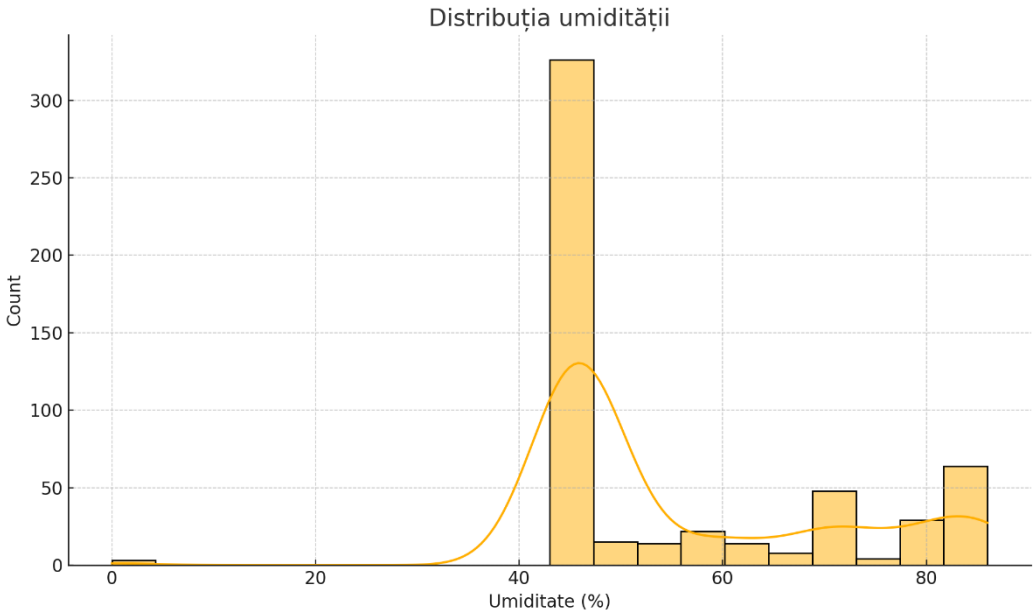


Figura 3: Distribuția calității aerului.

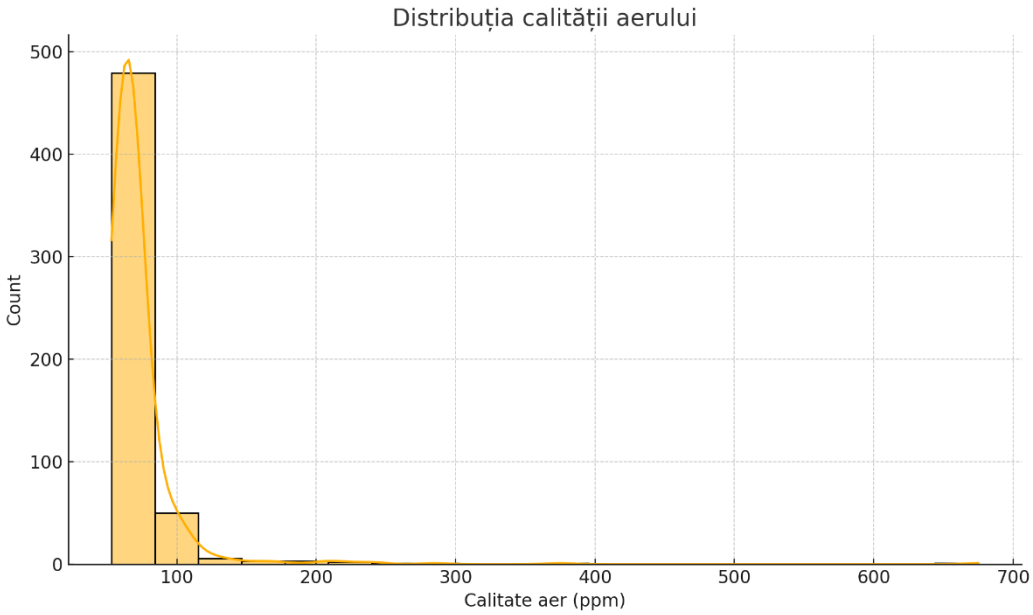


Figura 4: Matricea de corelație între variabile.

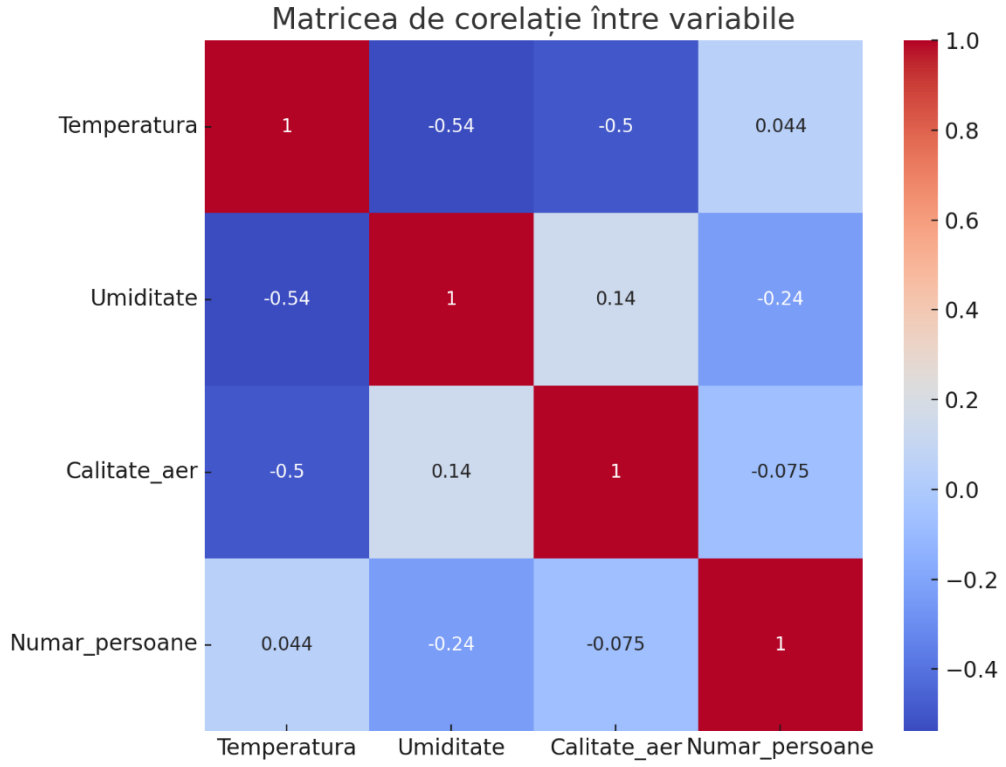
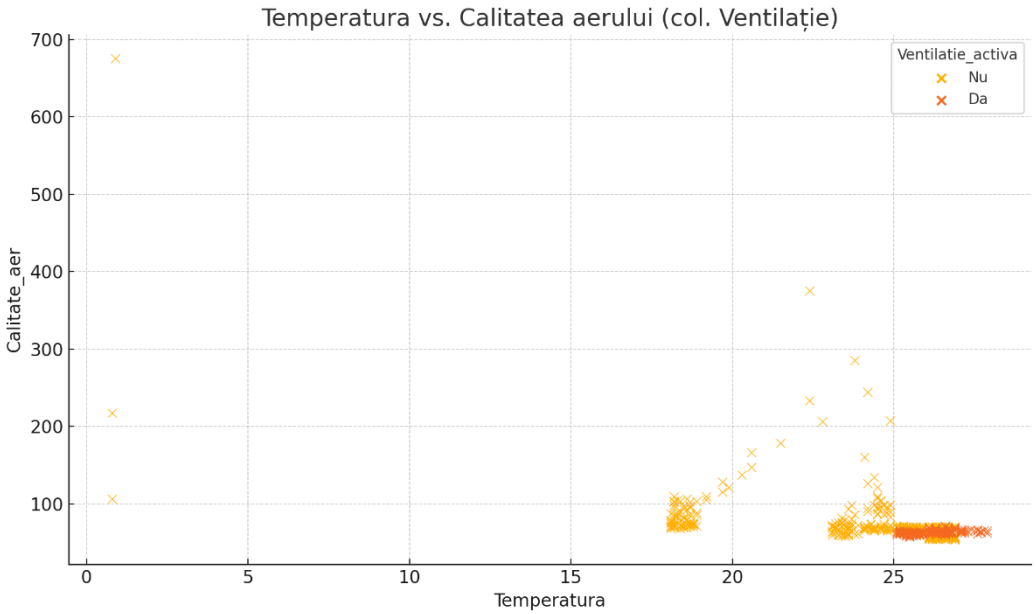


Figura 5: Relația dintre temperatură și calitatea aerului, în funcție de starea ventilației.



5. Extinderea explorării datelor

În cadrul analizei dedicate optimizării calității aerului în spații închise, o etapă esențială o reprezintă identificarea relațiilor semnificative dintre variabilele monitorizate și formularea unor întrebări relevante care pot fundamenta construirea unui model predictiv robust. Pornind de la principiile prezentate în cadrul cursului „Data Literacy in Practice” din platforma SAS Skills Builder for Students, această etapă vizează trecerea de la simple observații descriptive la extragerea de cunoștințe acționabile, cu aplicabilitate directă în controlul inteligent al mediului.

Pentru a susține această abordare, se propun următoarele întrebări analitice, menite să ghideze procesul de modelare:

- Cum influențează simultan „Ventilația activă” și „Geamul deschis” nivelul calității aerului?
- Care este importanța relativă a fiecărei variabile categorice în predicția calității aerului?
- În ce măsură condițiile meteo și tipul incintei modifică relațiile dintre variabilele ambientale?
- Pot fi identificate interacțiuni semnificative între factori care amplifică sau atenuează efectele asupra calității aerului?

Răspunsul la aceste întrebări presupune o înțelegere clară a structurii variabilelor din setul de date. Calitatea aerului, exprimată numeric, joacă rolul de variabilă dependentă, în timp ce factorii explicativi — atât categorici (precum starea ventilației, a geamurilor, condițiile meteo și tipul de incintă), cât și numerici (temperatura, umiditatea, numărul de persoane) — constituie predictori relevanți în model.

Pregătirea setului de date presupune mai multe etape de preprocesare. Variabilele categorice trebuie codificate (OneHotEncoding), iar informațiile temporale pot fi valorificate prin extragerea orei din timestamp, oferind astfel o perspectivă asupra variațiilor diurne. În plus, modelul poate fi îmbunătățit prin introducerea unor termeni de interacțiune, care reflectă relații combinate între variabile (ex. ventilație × geam deschis).

Din punct de vedere metodologic, se propune inițial o regresie liniară regularizată (Ridge), capabilă să furnizeze un model interpretabil, dar rezistent la supraînvățare. Aceasta va fi completată printr-o explorare cu algoritmi de tip Random Forest, utili pentru captarea relațiilor non-liniare și evaluarea importanței relative a predictorilor. Performanțele modelului vor fi validate prin intermediul unei validări încrucișate cu 5 fold-uri, utilizând ca metrici de referință RMSE și R^2.

Acest proces poate fi orchestrat într-un pipeline complet: de la împărțirea datelor în subseturi de antrenare și testare, la standardizarea și codificarea caracteristicilor, aplicarea modelului și interpretarea rezultatelor. Astfel, se creează premisele pentru integrarea într-un sistem de predicție automatizat, care să transforme datele colectate în timp real în decizii contextuale, orientate spre menținerea unui climat interior optim și sigur.

6. Concluzii si discuție

Implementarea acestui proiect a demonstrat fezabilitatea utilizării unui sistem simplu, compus din senzori de mediu și o platformă cloud, pentru a genera date în timp real ce pot fi analizate prin instrumente Python. Îmbinarea componentelor hardware cu procesele software de tip ETL (extragere, transformare, încărcare) a permis formularea unor concluzii relevante asupra calității aerului și a factorilor care o influențează. Prin acest demers, a fost ilustrată capacitatea unui prototip accesibil de a susține procese complexe de analiză exploratorie și modelare predictivă.

Una dintre lecțiile majore extrase din acest proces a fost importanța standardizării și curățării riguroase a datelor. Conversia valorilor numerice, tratarea timestamp-urilor și validarea tipurilor de variabile au fost esențiale pentru obținerea unei baze de date coerente. De asemenea, au fost necesare soluții robuste pentru menținerea conexiunii la rețea, cum ar fi mecanismele de reconectare și reinițializare TCP, care s-au dovedit vitale în asigurarea continuității achiziției de date în timp real.

Cu toate acestea, procesul a fost marcat și de multiple dificultăți. În primul rând, senzorii utilizați nu au oferit întotdeauna informații cu o precizie ridicată, ceea ce a impus o filtrare atentă a valorilor extreme și o interpretare critică a rezultatelor. Datele nu au fost centrate statistic (standardizate), deoarece acest lucru ar fi dus la eliminarea unei părți semnificative din observații — aspect nedorit într-un set de date deja limitat ca volum. În plus, identificarea unor relații semnificative între variabile s-a dovedit o provocare metodologică, în special în prezența interacțiunilor contextuale, cum ar fi vremea sau tipul incintei, care influențează comportamentul ambiental într-un mod dinamic și adesea non-liniar.

O altă limitare importantă o constituie portabilitatea hardware. Fără integrarea unui modul GPS, localizarea automată a măsurătorilor nu este posibilă, ceea ce afectează aplicabilitatea sistemului într-un context geografic mai larg. De asemenea, eșantionarea realizată la intervale relativ mari limitează granularitatea analizei. Un alt obstacol a fost lipsa unui dashboard de vizualizare live, ceea ce a îngreunat monitorizarea în timp real a datelor și interpretarea lor dintr-o perspectivă operațională.

Pentru dezvoltările viitoare, se preconizează integrarea unor componente suplimentare, precum un modul GPS, care să permită corelarea automată a datelor cu locația de proveniență. De asemenea, creșterea frecvenței de eșantionare la nivel de secundă ar îmbunătăți considerabil rezoluția temporală a analizei. Un alt pas firesc este automatizarea completă a procesului prin implementarea unui pipeline de tip machine learning integrat, capabil să formuleze predicții și recomandări în timp real. Complementar, dezvoltarea unui dashboard interactiv ar permite vizualizarea în direct a evoluției parametrilor și ar sprijini luarea deciziilor în contexte operaționale sau educaționale.

În concluzie, proiectul a validat potențialul unor soluții accesibile din punct de vedere tehnologic de a genera valoare analitică semnificativă, cu condiția integrării acestora într-un cadru metodologic riguros și flexibil.

7. Bibliografie

In aceasta secțiune prezentați bibliografia pe care ați utilizat-o pentru pregătirea temei de casa. Nu includeți doar link-uri la site-uri, analizați diverse lucrări, vezi si articolele exemplu încărcate in Moodle.

1. Drăgoicea, M. (2024). *Introducere în procesarea în timp real*. Universitatea Politehnica din București – PSITR, Curs 1.
2. Drăgoicea, M. (2024). *Arhitecturi și streaming de date*. Universitatea Politehnica din București – PSITR, Curs 2.
3. MathWorks. *Getting Started with ThingSpeak*. Documentație oficială ThingSpeak. Disponibil la: https://www.mathworks.com/help/thingspeak/
4. Google. *Google Sheets Help – IMPORTDATA, QUERY, ARRAYFORMULA*. Disponibil la: https://support.google.com/docs
5. McKinney, W. (2018). *Python for Data Analysis* (2nd ed.). O'Reilly Media.
6. DataCamp. *Exploratory Data Analysis in Python*. Capitolele 1–2. Platformă: https://www.datacamp.com
7. SAS Institute. (2024). *Data Literacy in Practice*. Platformă educațională SAS Skills Builder for Students.
8. Scikit-learn. *User Guide*. Disponibil la: https://scikit-learn.org/stable/user_guide.html

8. Anexa A – Cod sursă Arduino Uno + ESP8266 + DHT11 + MQ-135

Funcționalitate:

1. Conectare automată la una din cele trei rețele WiFi (timeout 10 secunde)
2. Citirea senzorilor de calitate aer (MQ-135) și temperatură/umiditate (DHT11) la fiecare 5 secunde
3. Trimiterea datelor către ThingSpeak (câmpurile 3, 4 și 5)
4. Reinițializarea conexiunii TCP după fiecare transmisie

```
#include <WiFiEsp.h>

#include <SoftwareSerial.h>

#include <ThingSpeak.h>

#include <DHT.h>


// SoftwareSerial pe D2/D3 către ESP-01

SoftwareSerial espSerial(2, 3);

#define DHTPIN 4

#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);


// Lista de SSID-uri și parole

const char* ssidList[] = {"UPB", "MERCUSYS_AEEA_2_4G", "FANTOMAS_5G"};

const char* passList[] = {"aglq8685", "wQc4kxdc-2020", "84153937"};

const int networks = sizeof(ssidList)/sizeof(ssidList[0]);
```

```
unsigned long channelID = 2945552;
const char* writeAPIKey = "QA7KPYAFGZ5S5LDN";
WiFiEspClient client;

void connectWiFi() {
  Serial.print("Conectare WiFi");
  for (int i = 0; i < networks; i++) {
    Serial.print("\n Încerc "); Serial.print(ssidList[i]);
    WiFi.begin(ssidList[i], passList[i]);
    unsigned long start = millis();
    while (millis() - start < 10000 && WiFi.status() != WL_CONNECTED) {
      Serial.print("."); delay(500);
    }
    if (WiFi.status() == WL_CONNECTED) {
      Serial.print(" → Conectat la "); Serial.println(ssidList[i]);
      return;
    }
    Serial.print(" eșuat");
  }
  Serial.println("\nNu am putut conecta la niciun WiFi!");
}

void setup() {
  Serial.begin(9600);
  espSerial.begin(9600);
  dht.begin();

  WiFi.init(&espSerial);
  connectWiFi();

  if (WiFi.status() == WL_CONNECTED)
    ThingSpeak.begin(client);
}

void loop() {
  if (WiFi.status() != WL_CONNECTED) {
    connectWiFi();
```

```
  ThingSpeak.begin(client);
 }

 int calAer = analogRead(A0);
 float temp = NAN, hum = NAN;

 for (int i = 0; i < 5; i++) {
  temp = dht.readTemperature();
  hum = dht.readHumidity();
  if (!isnan(temp) && !isnan(hum)) break;
  delay(1000);
 }

 Serial.print("Aer: "); Serial.print(calAer);

 if (!isnan(temp) && !isnan(hum)) {
  Serial.print(" | Temp: "); Serial.print(temp); Serial.print("°C");
  Serial.print(" | Umid: "); Serial.print(hum); Serial.println("%");
 } else {
  Serial.println(" | DHT11 invalid");
 }

 ThingSpeak.setField(3, calAer);
 if (!isnan(temp)) ThingSpeak.setField(4, temp);
 if (!isnan(hum))  ThingSpeak.setField(5, hum);

 int status = ThingSpeak.writeFields(channelID, writeAPIKey);
 Serial.print("TS status: "); Serial.println(status);

 client.stop();
 delay(1000);
 ThingSpeak.begin(client);

 Serial.println("----------------------");
 delay(5000);
}
```