Darie Vlad-Andrei, Seria CC, Grupa 311

Partea I:

1. Tipul problemei

   Problema aleasa este de clasificare.

2. Generarea setului de date si impartirea lui in subsetul antrenare si cel de testare

Sursa datelor din tabel este wikipedia ul aferent fiecarei specii de animal, iar informatiile pe care nu le am gasit acolo, le am pus eu.

Metoda de generare folosita este cea sintetica.

Pentru inceput importam bibliotecile numpy, pandas si train_test_split. Acum setam un seed pentru a putea genera aceleasi valori aleatoare de fiecare data cand rulam programul (seed ul folosit a fost 42 asa cum am vazut si in laborator si pentru ca e o conventie).

Apoi facem o lista cu numele speciilor pe care le vom folosi pentru tabel si facem functii ce vor genera date pentru tabel.

Functia genereaza_nume_specie primeste ca parametru un numar si intoarce o lista cu numarul de nume de specii dat. Folosind aceasta functie generam o lista de 1000 de nume de specii pe care o vom da ca parametru pentru restul functiilor de generare.

Functia de generare_categorie primeste ca parametru lista de 1000 de nume de specii si intoarce o lista cu 1000 de categorii in care pentru fiecare nume din lista ca parametru se va pune categoria sa (Mamifer/Pasare/Reptila) specifica animalului.

Functia genereaza_greutate_kg are acelasi parametru ca cea de dinainte si intoarce o lista cu 1000 de greutati in kg pentru fiecare nume de specie in care greutatea va fi un numar float aleator dintr un interval specificat cu np.random.uniform, rotunjit la 2 zecimale cu np.round(, 2). Pentru Papagal, Bufnita, Vipera si Sarpe Boa punem valori lipsa.

Functia genereaza_durata_viata_ani primeste acelasi parametru si intoarce o lista cu 1000 de durate de viata in ani pentru fiecare nume de specie in care durata de viata va fi un numar intreg aleator dintr un interval specificat, cu np.randint.

Functia genereaza_habitat primeste acelasi parametru si intoarce o lista cu 1000 de habitate in care pentru fiecare nume de specie se va pune habitatul corespunzator fiecarui nume(Padure/Jungla/Desert). Totusi unele animale pot fi regasite in mai multe habitate asa ca pentru ele vom genera un numar float aleator iar daca acesta e mai mic decat o valoare aleasa de mine va primi un habitat, altfel pe celalalt.

Functia genereaza_pericol_disparitie primeste acelasi parametru si intoarce o lista cu 1000 de Da sau Nu. Alegerea e facuta dupa generarea unui numar float, daca acesta e mai mic decat 0.2 va primi Da, altfel Nu.

Functia genereaza_viteza_Km_h primeste acelasi parametru si intoarce o lista cu 1000 de viteze in km/h pentru fiecare nume de specie in care viteza va fi un numar float aleator dintr

un interval specificat, rotunjit la 2 zecimale. Totusi pentru Maimuta, Papagal, Bufnita si Strut punem valori lipsa.

Functia genereaza_inaltime_m primeste acelasi parametru si intoarce o lista cu 1000 de inaltimi alese aleator dintr un interval specificat pentru fiecare animal. Totusi pentru Lup, Papagal, Bufnita, Vipera si Sarpe Boa punem valori lipsa.

Functia genereaza_culoare primeste acelasi parametru si intoarce o lista cu 1000 de culori specifice pentru fiecare animal. Totusi unele animale pot fi gasite avand culori diferite asa ca voi genera pentru fiecare un numar aleator si in functie daca acesta e mai mic decat un numar ales de mine va primi o culoare, altfel o va primi pe cealalta.

Acum vom face un dataframe tabel, cu pd.DataFrame, pentru a putea lucra cu datele noastre, in care specificam numele fiecarei coloane din dataframe si apelam functia corespunzatoare fiecareia.

In final, impartim dataframe ul nostru in 700 de randuri, cu tabel.sample(n=700, random_state=42), pe care le punem in dataframe ul train, iar restul de 300 in dataframe ul test, cu tabel.drop(train.index).

Acum salvam aceste 2 dataframe uri in 2 fisiere csv diferite, train.csv si test.csv, cu train si test .to_csv('nume_fisier.csv', index=False).

Ipotezele facute sunt ca am presupus ca fiecare animal are o singura valoare fixa pentru anumite coloane, dar pentru altele acelasi animal poate avea alte caracteristice toate alese printr o decizie aleatoare.

Tabelul contine cateva valori lipsa pe coloanele Greutate (Kg), Viteza (Km/h) si pe Inaltime (m).

3. Analiza valorilor lipsa

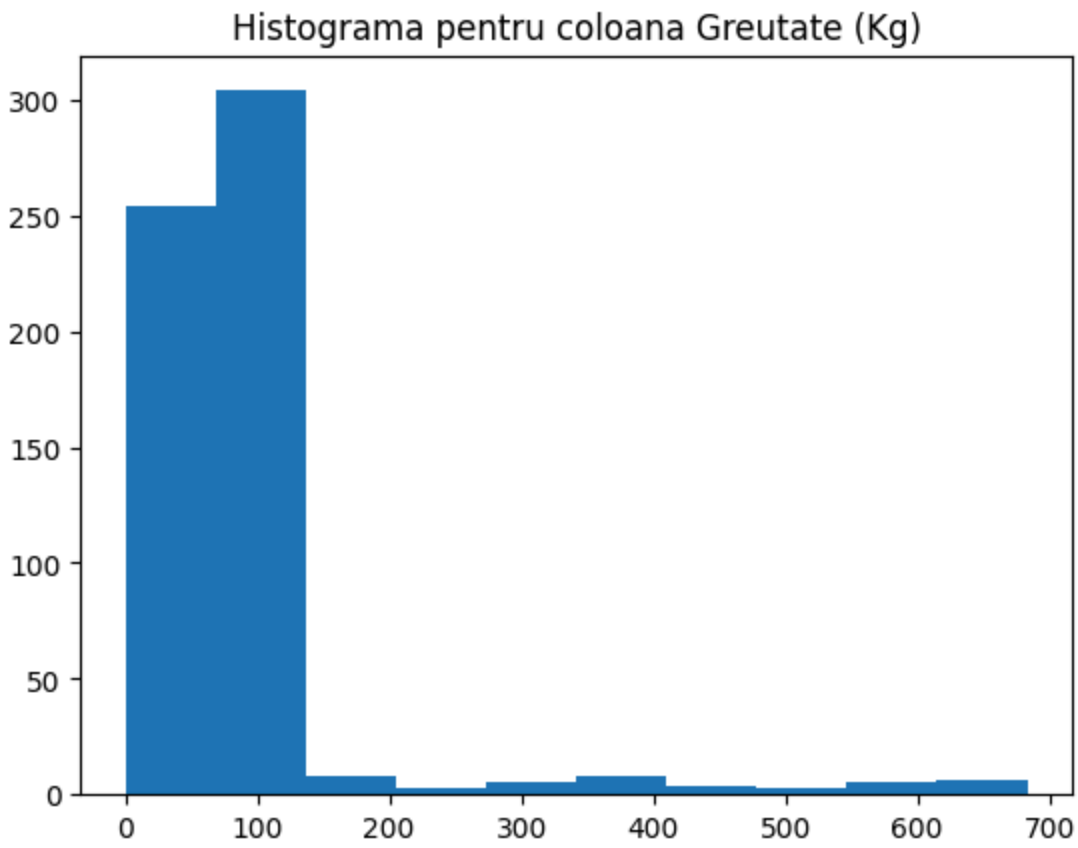
Vom testa pe rand pentru setul Test si pentru setul Train daca avem valori lipsa astfel: Folosim functia .isnull().sum() ce va face suma valorilor lipsa pentru fiecare coloana, apoi pentru procentul lor calculam numarul de randuri cu .shape[0], iar procentul pentru fiecare coloana va fi (numarul de valori lipsa / numarul de randuri) * 100. Pentru coloanele gasite ca avand valori lipsa le vom completa cu media celorlalte valori de pe coloana respectiva folosind .fillna(coloana_respectiva.mean()). Dupa aceasta, verificam daca s au completat si le dam export din nou in fisierele lor csv corespunzatoare.

4. Statistici descriptive

Vom apela pentru Test si pentru Train functia .describe() si vom selecta coloanele categorice cu .select_dtypes(include='object') si vom da describe si la ele.

5. Analiza distributiei variabilelor

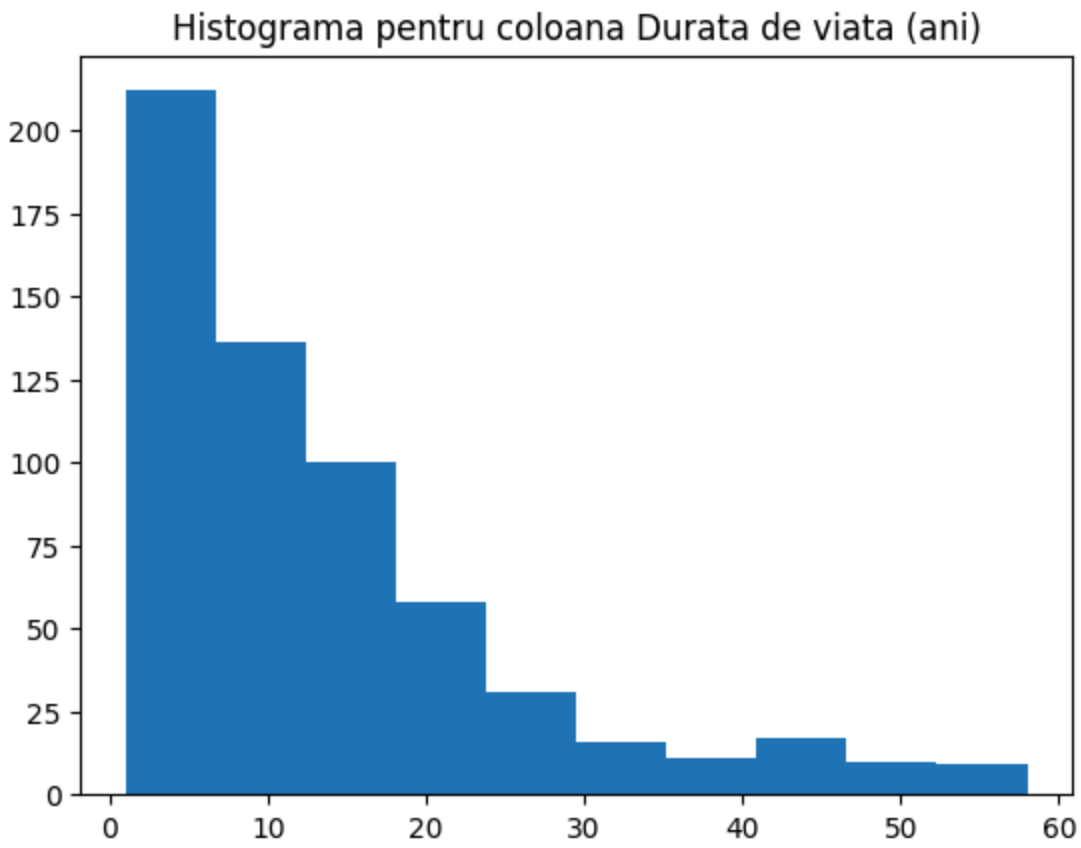
Vom incepe cu setul Train.



1 Graficul reprezinta distributia valorilor din coloana Greutate (Kg). Sunt mai multe valori intre 0 si 120 (aproximativ).

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca sunt foarte putine valori de la 120 (aproximativ) spre final.

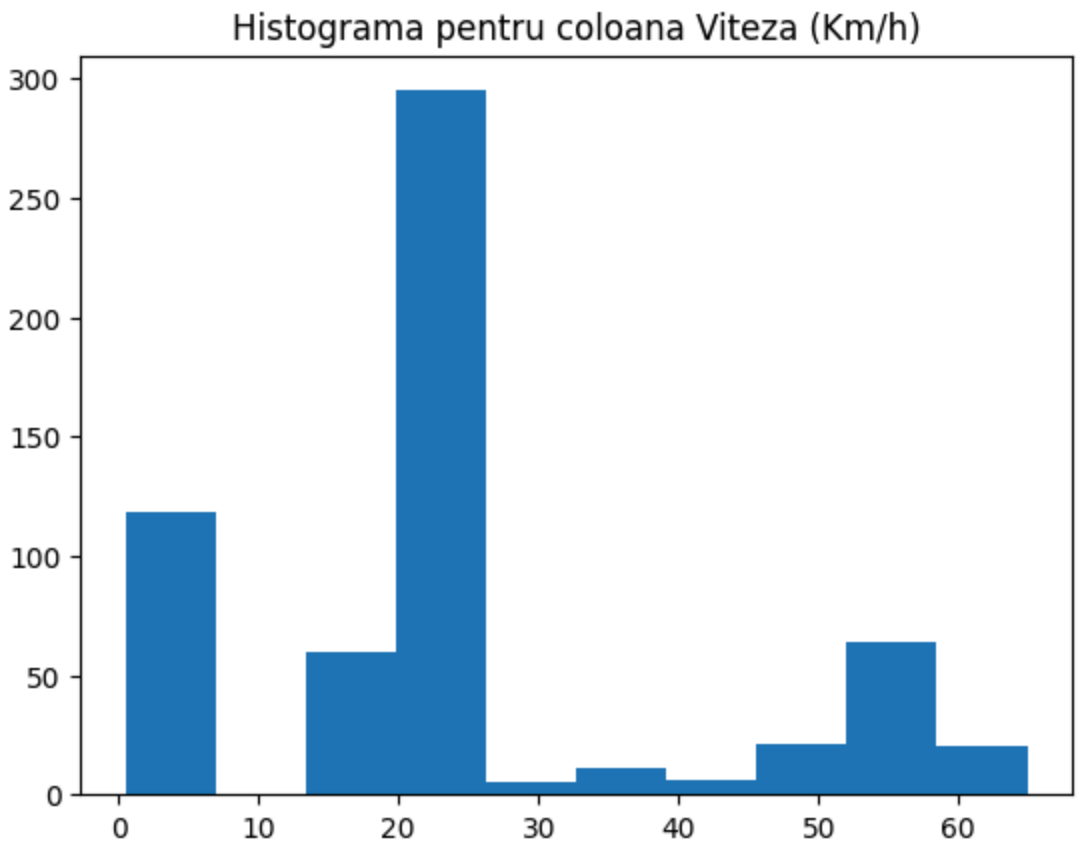
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Durata de viata (ani). Sunt mai multe valori la inceput si exista si un varf izolat intre 40 si 50.

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca valorile incep sa scada progresiv.

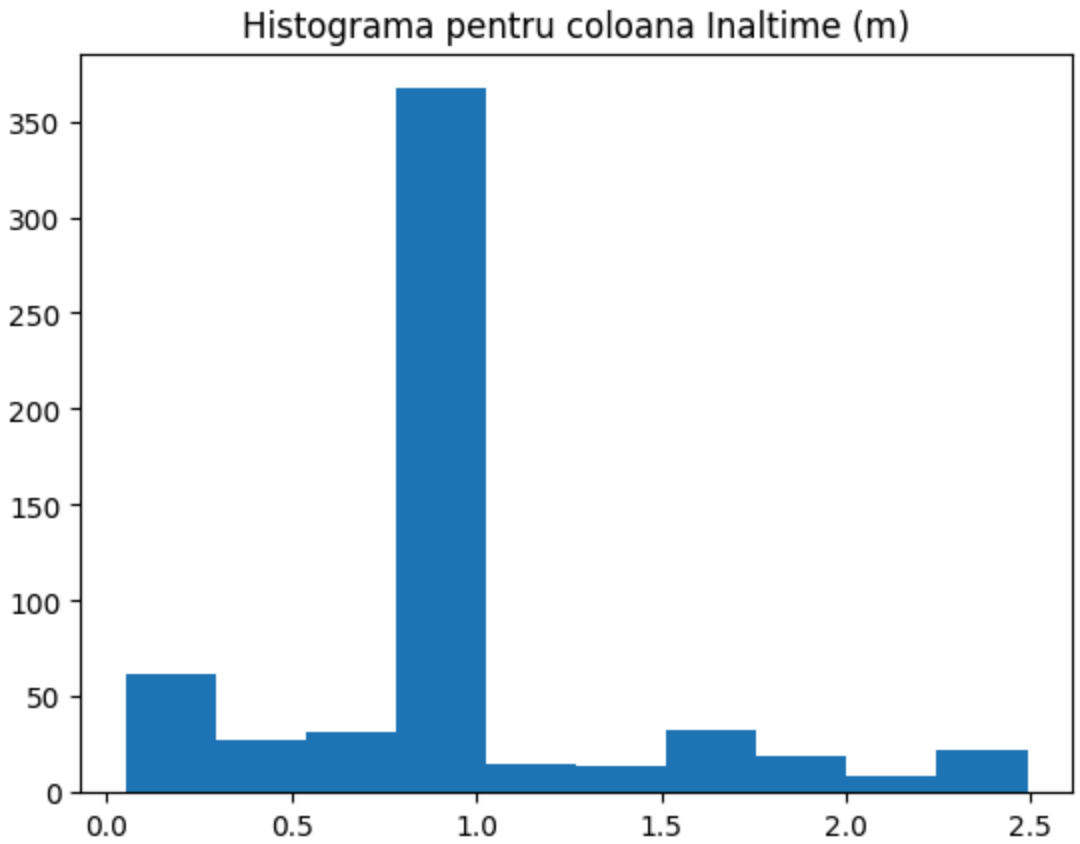
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Viteza (Km/h). Sunt mai multe valori intre 20 si 30.

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca sunt valori foarte aleatoare.

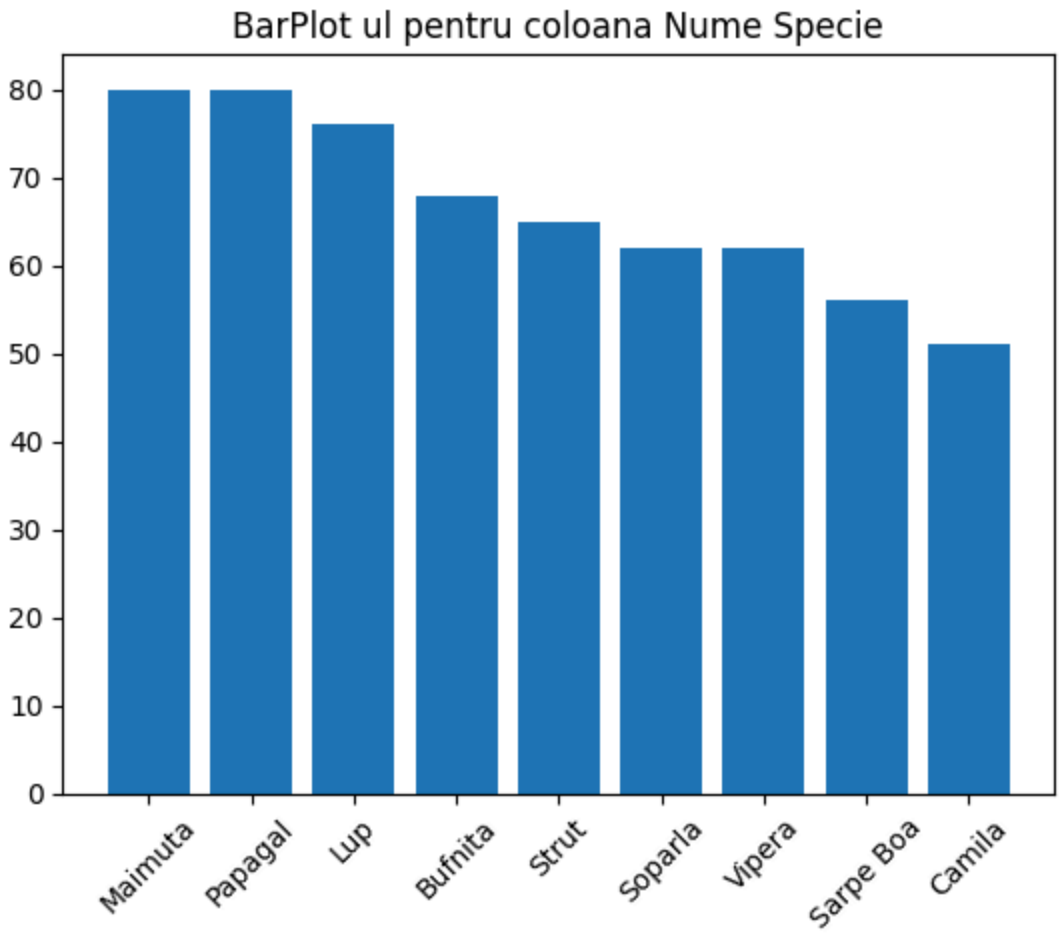
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Inaltime (m). Sunt mai multe valori intre 0.5 si 1.

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca e un singur varf iar restul sunt aleatoare.

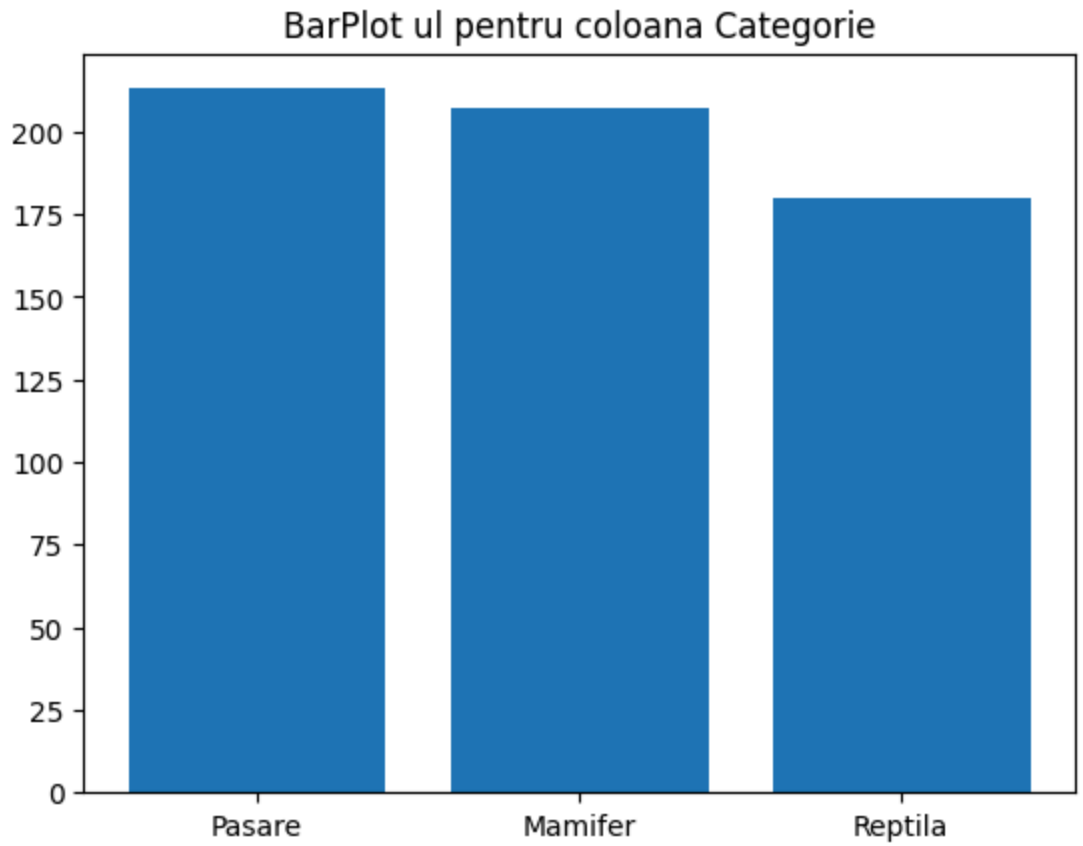
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Nume Specie. Sunt mai multe valori de Maimuta, iar de la ea scad progresiv.

2 Putem presupune ca e o distributie dezechilibrata a numelor.

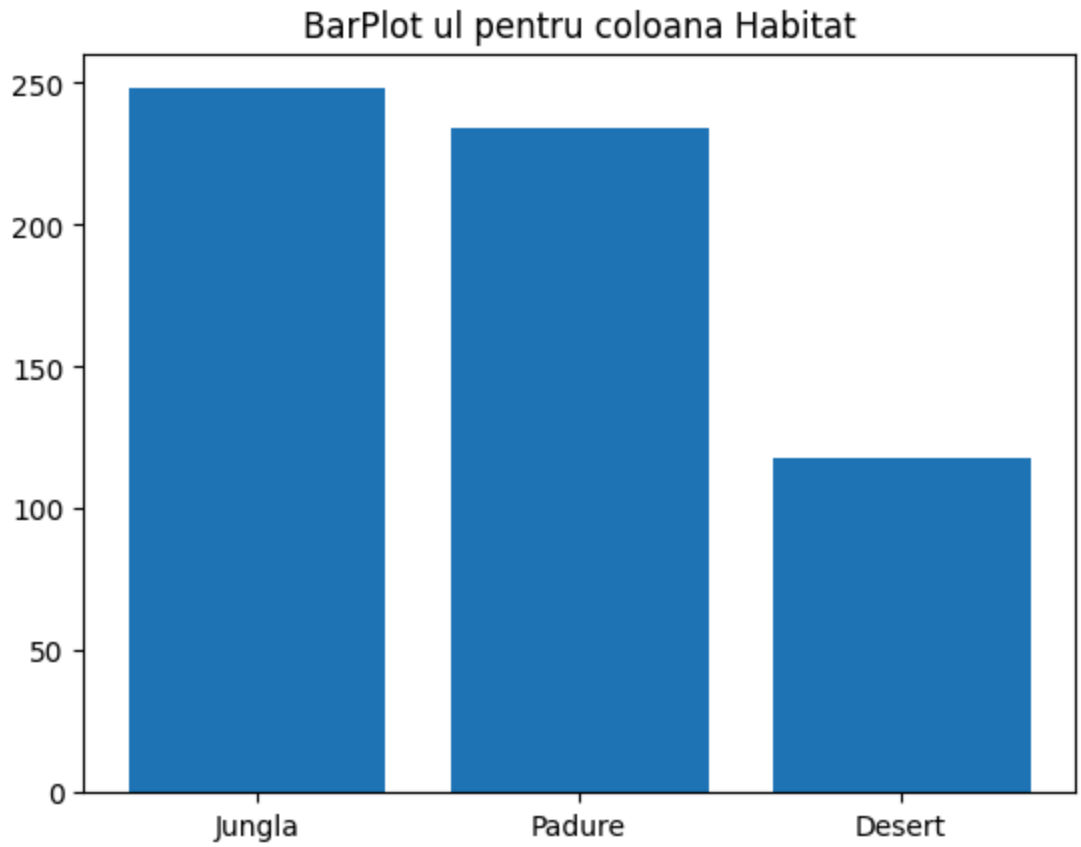
3 Le lasam asa.



1 Graficul reprezinta distributia valorilor din coloana Categorie. Sunt mai multe valori de Pasare, iar de la ea scad progresiv.

2 Putem presupune ca e o distributie dezechilibrata a categoriilor.

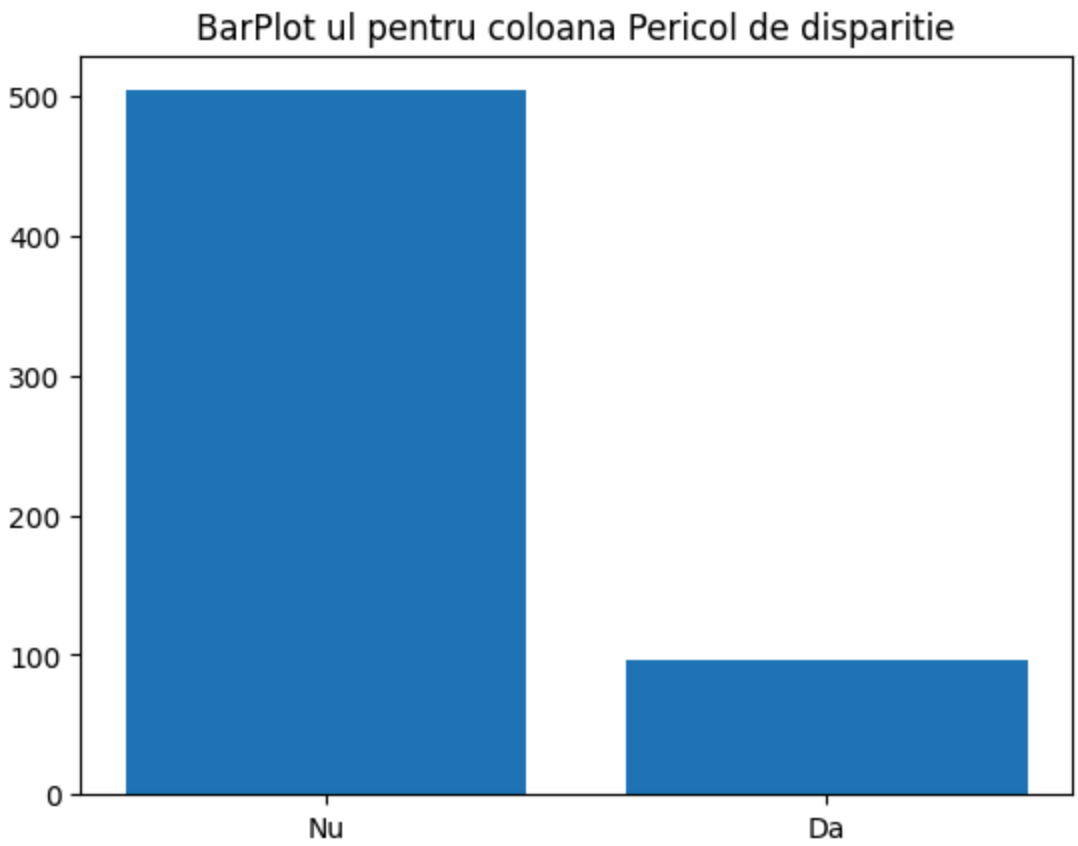
3 Putem aplica un OneHotEncoding a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Habitat. Sunt mai multe valori de Jungla, iar de la ea scad progresiv.

2 Putem presupune ca e o distributie dezechilibrata a habitatelor.

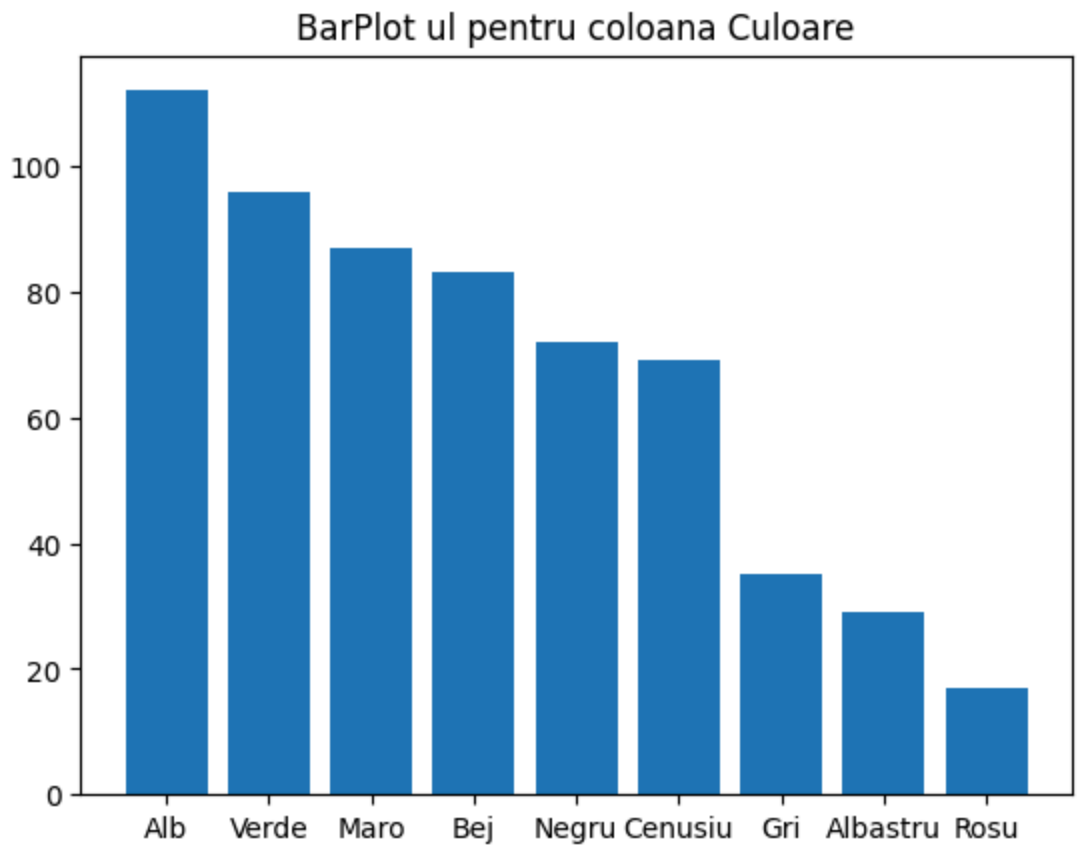
3 Putem aplica un OneHotEncoding a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Pericol de disparitie. Sunt mai multe valori de Nu.

2 Putem presupune ca e o distributie dezechilibrata a pericolelor.

3 Putem aplica un OneHotEncoding a acestor date.

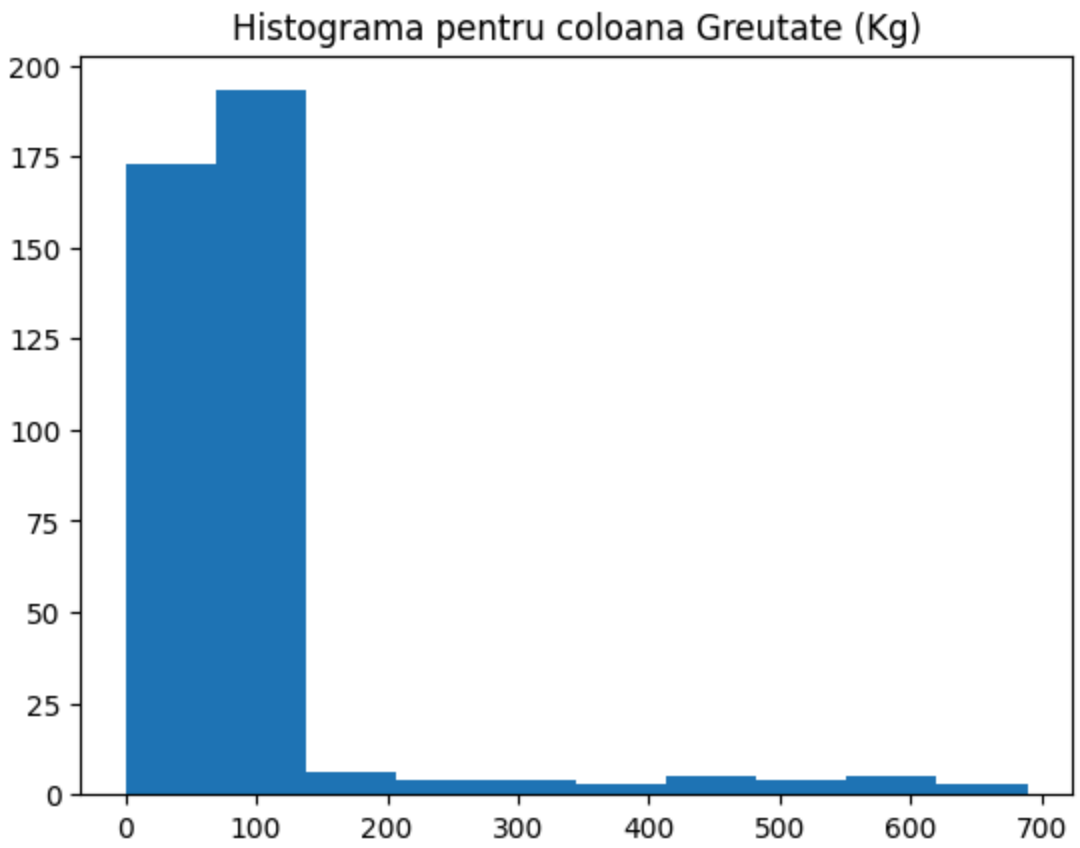


1 Graficul reprezinta distributia valorilor din coloana Culoare. Sunt mai multe valori de Alb, iar de la ea scad progresiv.

2 Putem presupune ca e o distributie dezechilibrata a culorilor.

3 Putem aplica un OneHotEncoding a acestor date.

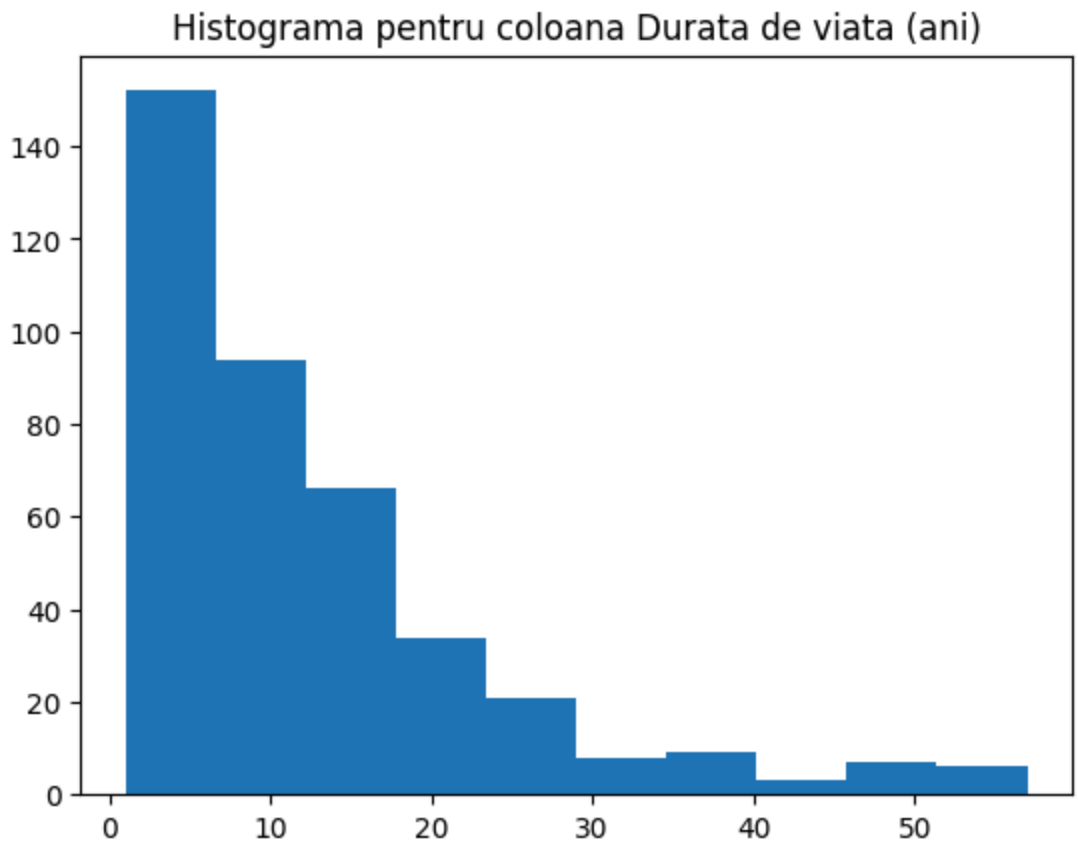
Vom continua cu setul Test.



1 Graficul reprezinta distributia valorilor din coloana Greutate (Kg). Sunt mai multe valori intre 0 si 120 (aproximativ).

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca sunt foarte putine valori de la 120 (aproximativ) spre final.

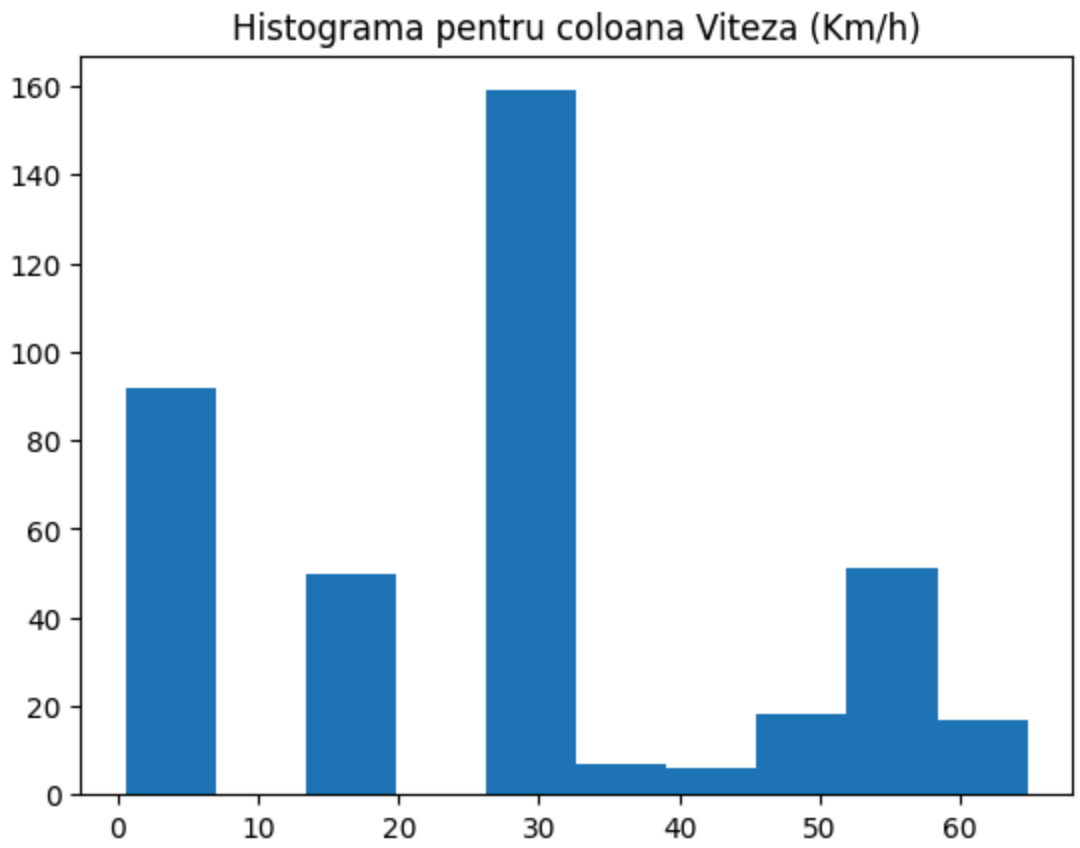
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Durata de viata (ani). Sunt mai multe valori la inceput si exista si un varf izolat intre 30 si 40 si intre 40 si 50.

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca valorile incep sa scada progresiv.

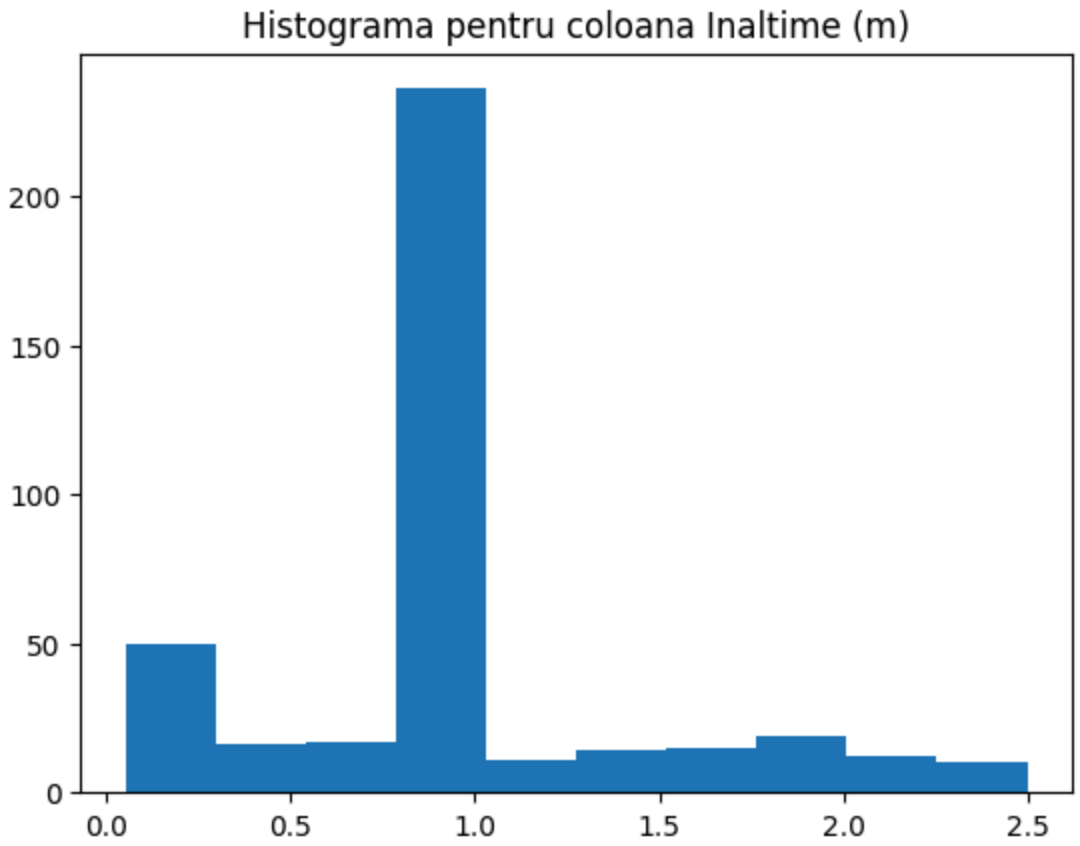
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Viteza (Km/h). Sunt mai multe valori intre 20 si 30.

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca sunt valori foarte aleatoare.

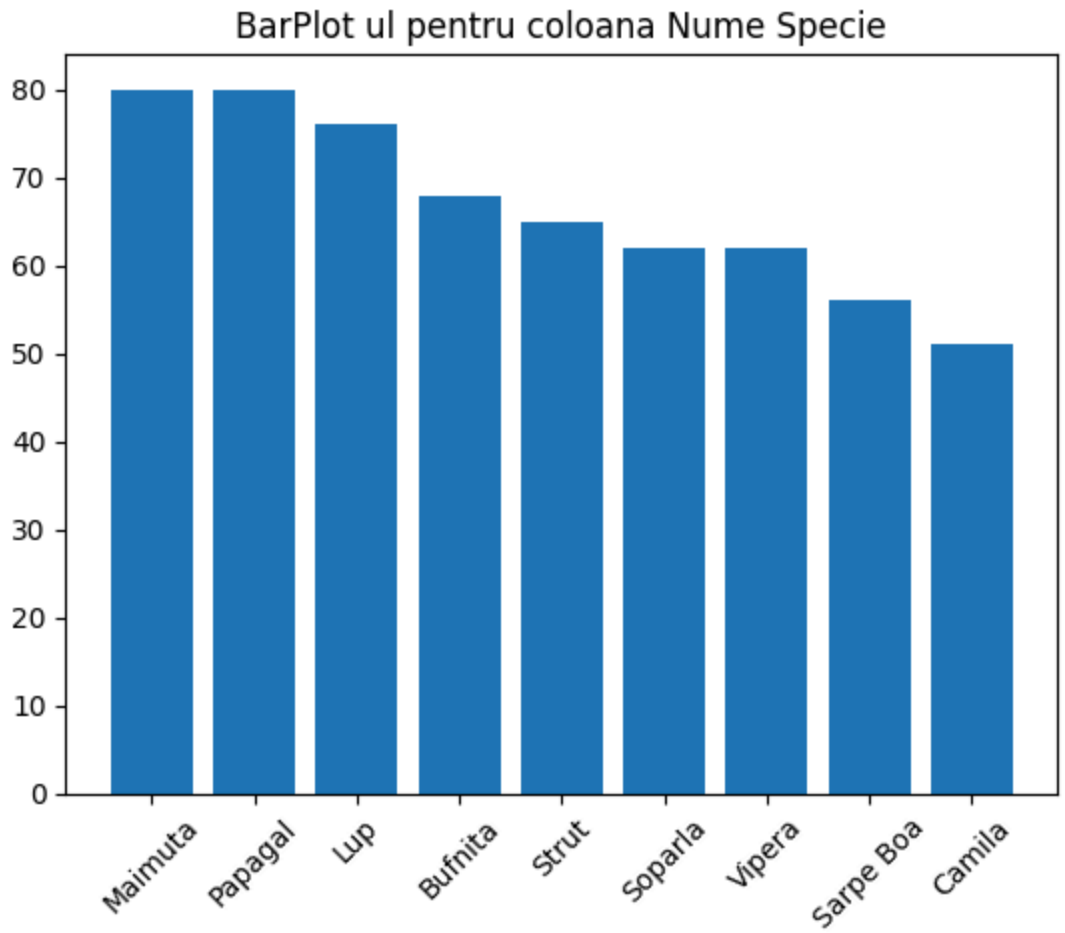
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Inaltime (m). Sunt mai multe valori intre 0.5 si 1.

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca e un singur varf iar restul sunt aleatoare.

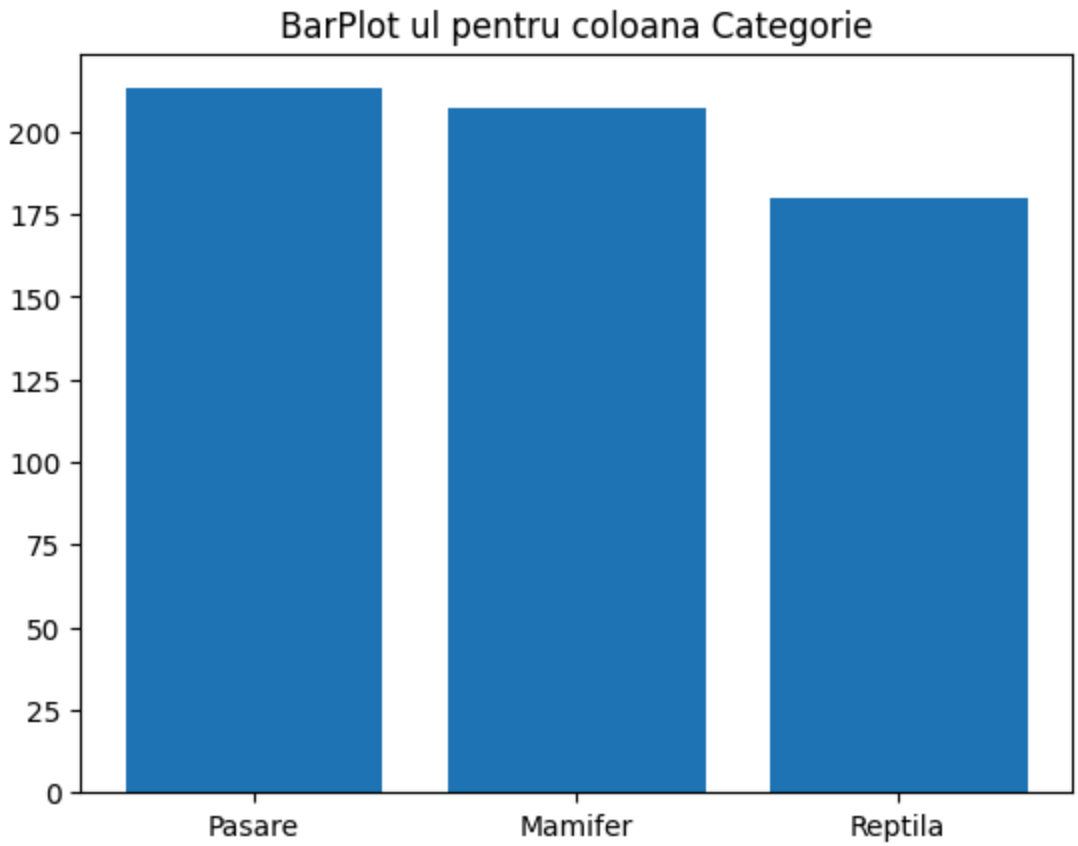
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Nume Specie. Sunt mai multe valori de Maimuta, iar de la ea scad progresiv.

2 Putem presupune ca e o distributie dezechilibrata a numelor.

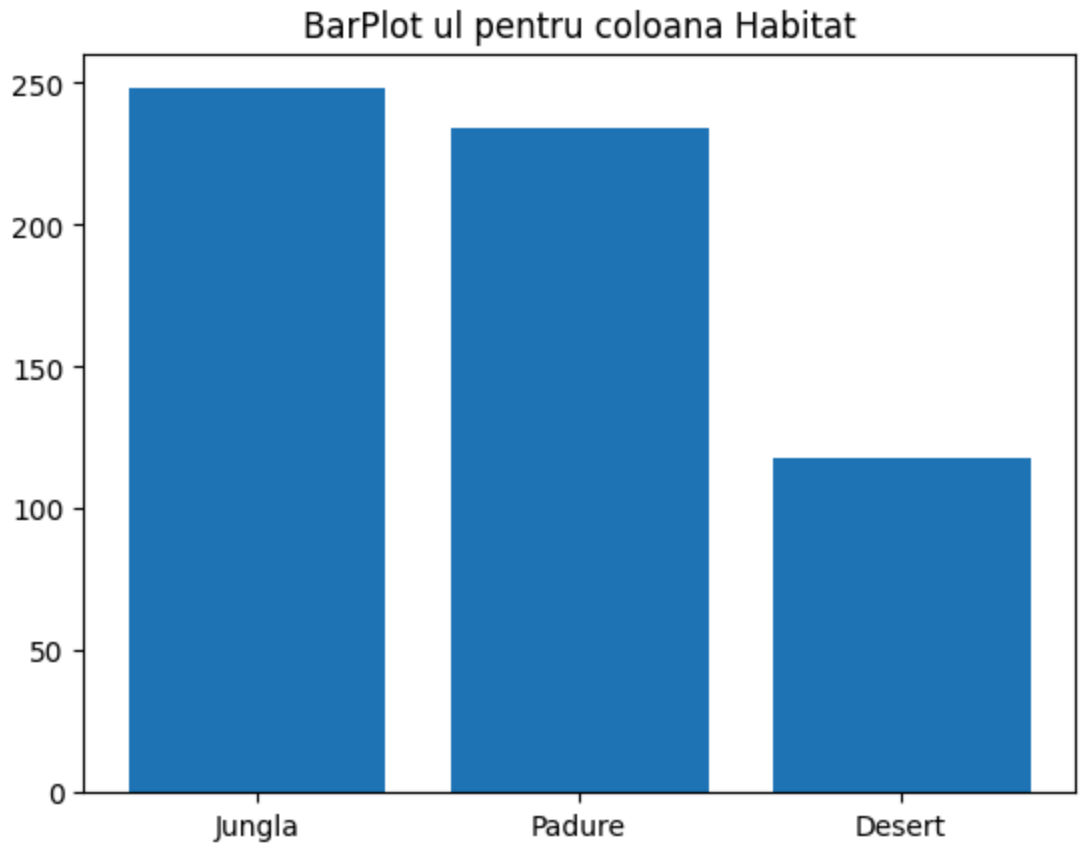
3 Le lasam asa.



1 Graficul reprezinta distributia valorilor din coloana Categorie. Sunt mai multe valori de Pasare, iar de la ea scad progresiv.

2 Putem presupune ca e o distributie dezechilibrata a categoriilor.

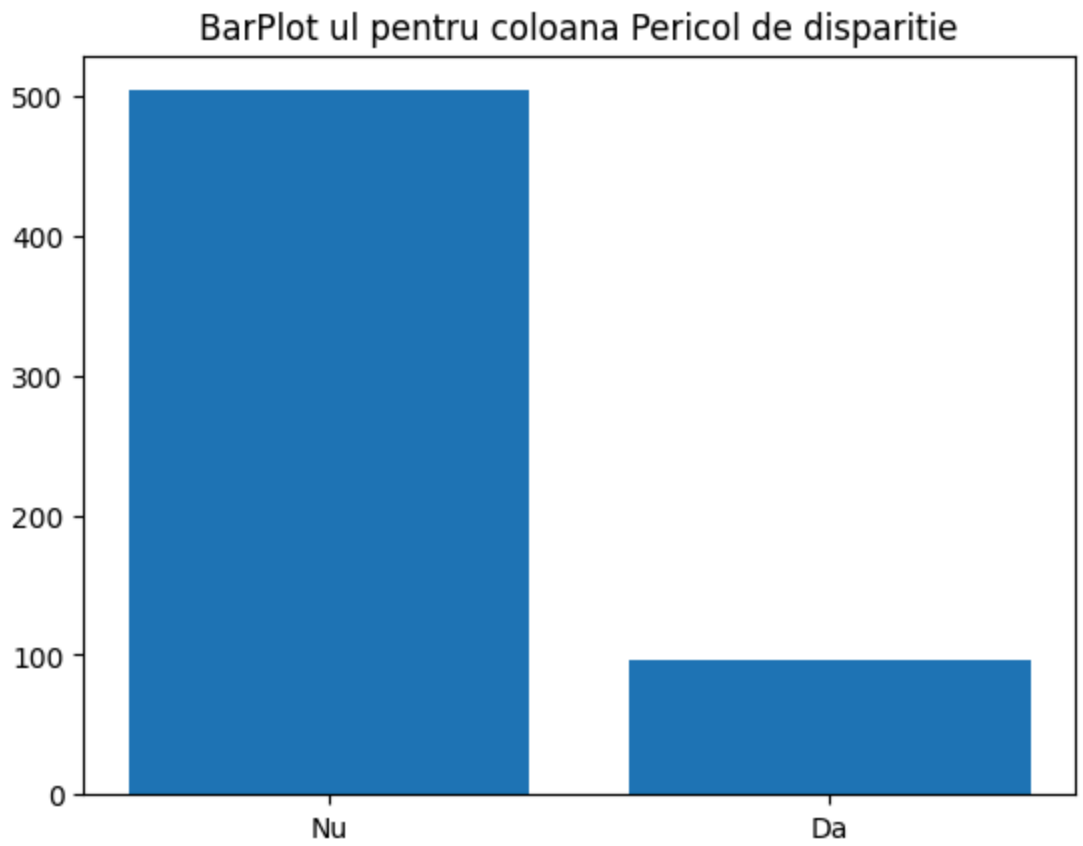
3 Putem aplica un OneHotEncoding a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Habitat. Sunt mai multe valori de Jungla, iar de la ea scad progresiv.

2 Putem presupune ca e o distributie dezechilibrata a habitatelor.

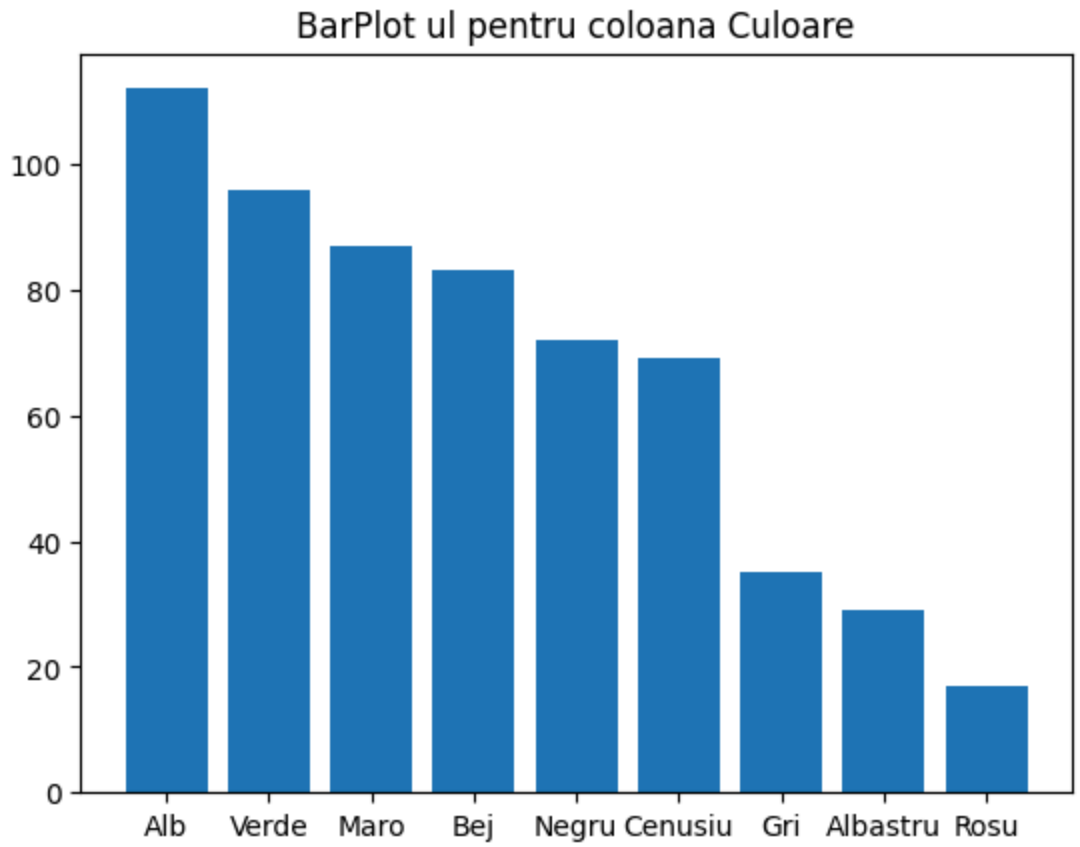
3 Putem aplica un OneHotEncoding a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Pericol de disparitie. Sunt mai multe valori de Nu.

2 Putem presupune ca e o distributie dezechilibrata a pericolelor.

3 Putem aplica un OneHotEncoding a acestor date.



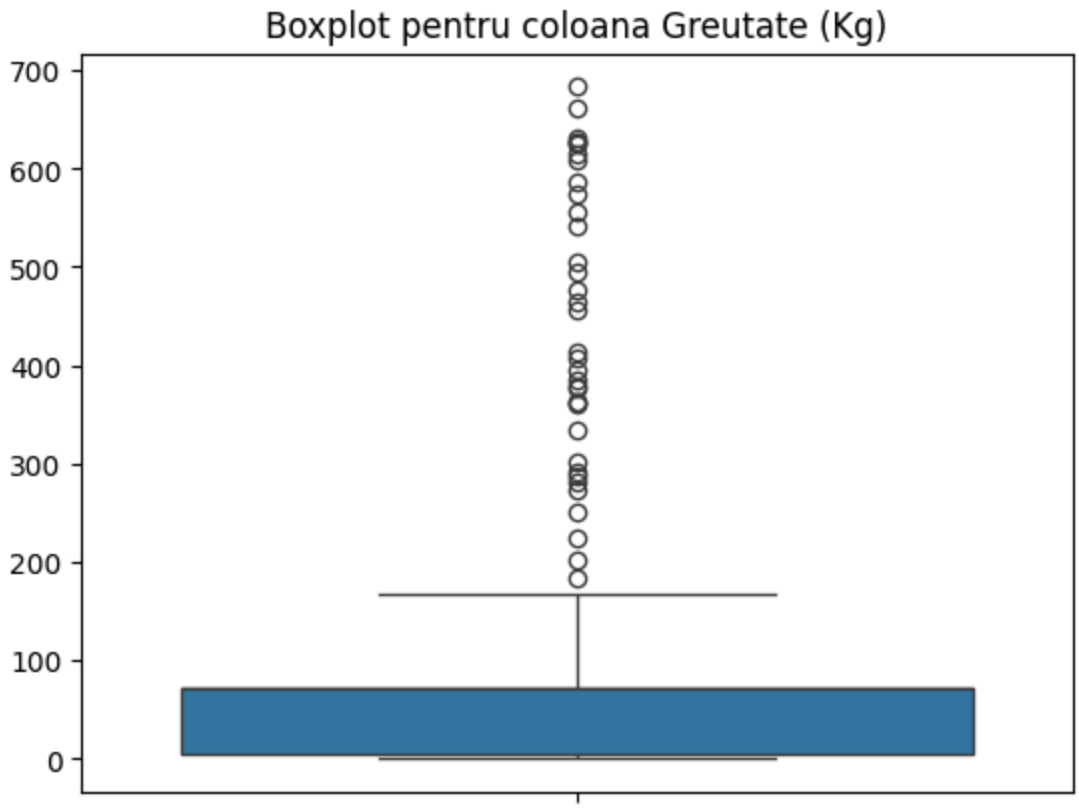
1 Graficul reprezinta distributia valorilor din coloana Culoare. Sunt mai multe valori de Alb, iar de la ea scad progresiv.

2 Putem presupune ca e o distributie dezechilibrata a culorilor.

3 Putem aplica un OneHotEncoding a acestor date.

6. Detectarea outlierilor

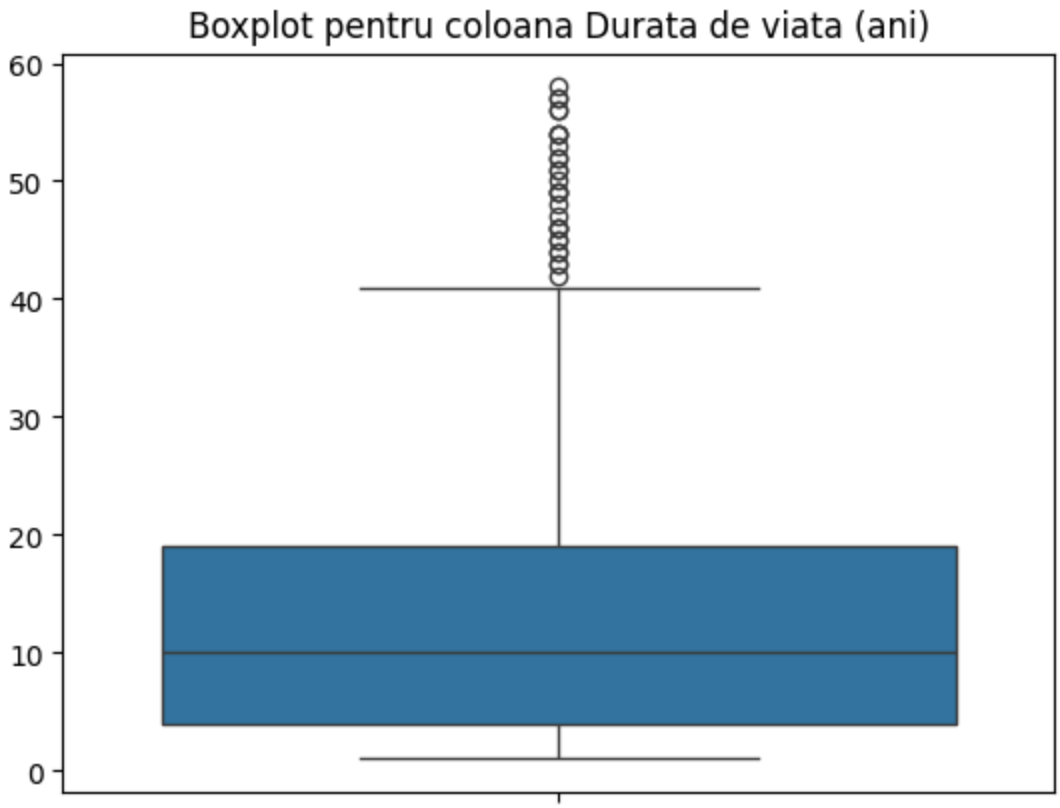
Vom incepe cu setul Train.



1 Graficul reprezinta distributia valorilor din coloana Greutate (Kg). Majoriatatea valorilor sunt pana la 80 Kg (aproximativ).

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca apar valori de la 170 la 700 Kg.

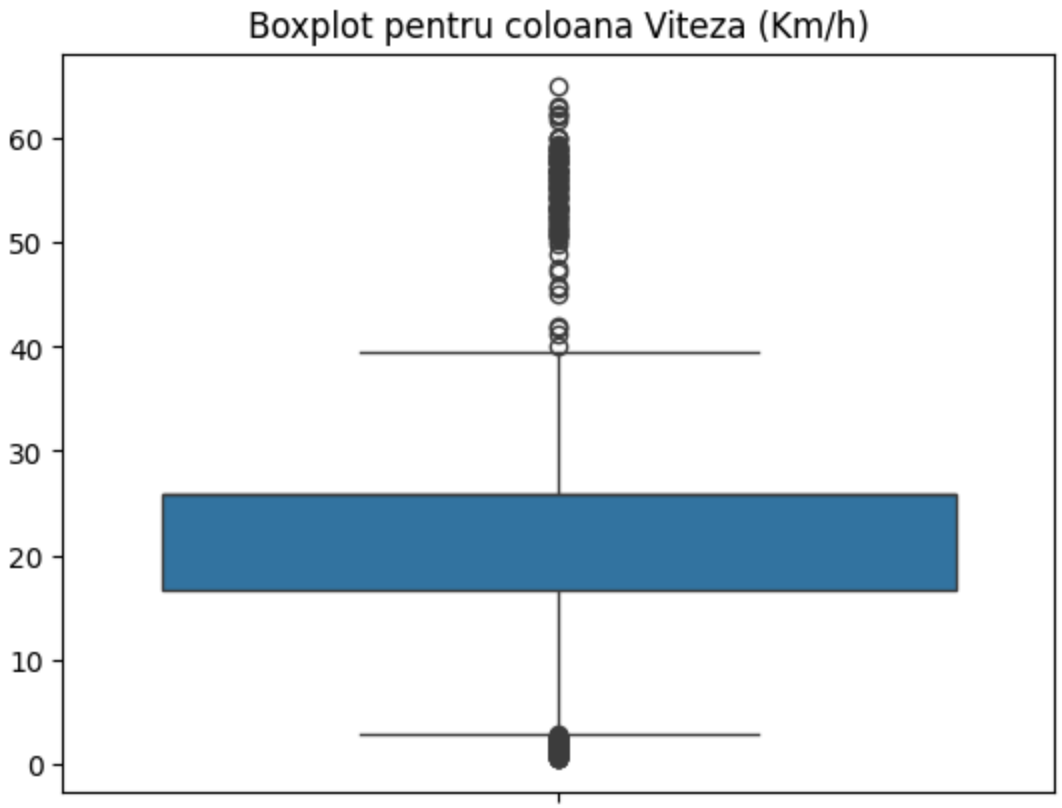
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Durata de viata (ani). Majoriatatea valorilor sunt pana la 20 ani (aproximativ). Mediana se afla la 10 ani.

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca apar valori de la 40 la 60 ani.

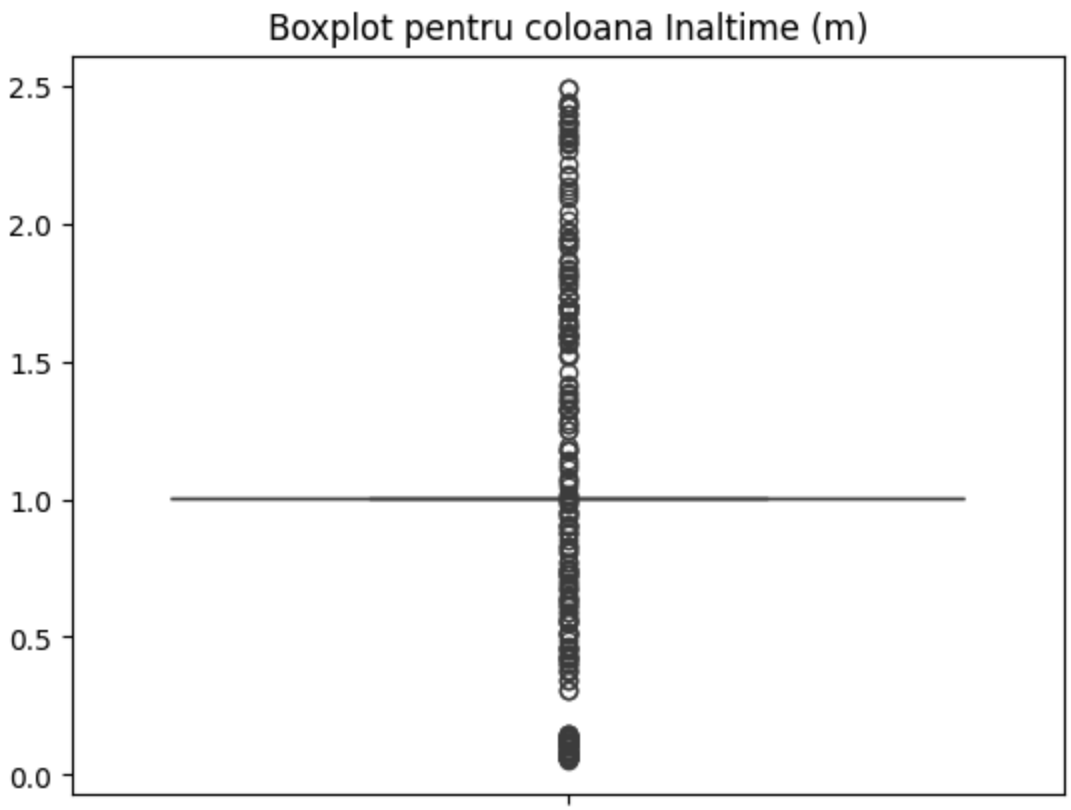
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Viteza (Km/h). Majoriatatea valorilor sunt de la 17 pana la 20 Km/h (aproximativ).

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca apar valori de la 40 la 70 Km/h si de la 4 la 0 KM/h.

3 Putem aplica o normalizare a acestor date.

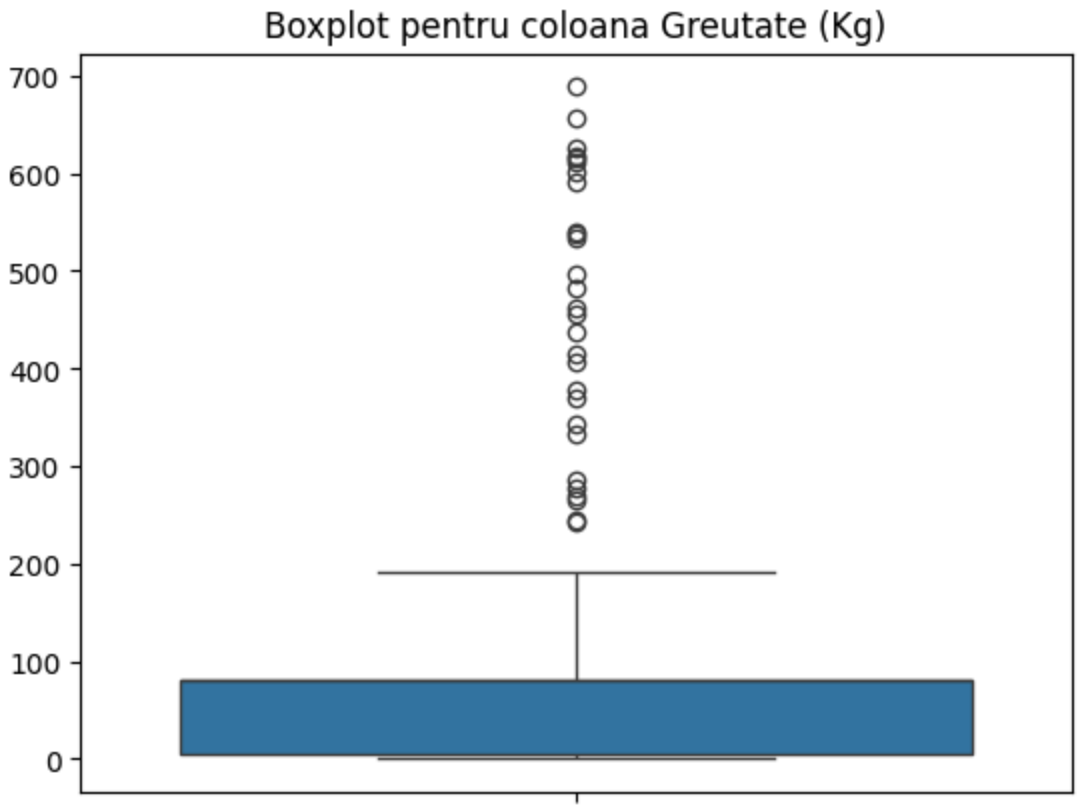


1 Graficul reprezinta distributia valorilor din coloana Inaltime (m). Majoriatatea valorilor sunt la 1 m (aproximativ).

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca apar valori de la 1 la 0 m si de la 1 la 2.5 m.

3 Putem aplica o normalizare a acestor date.

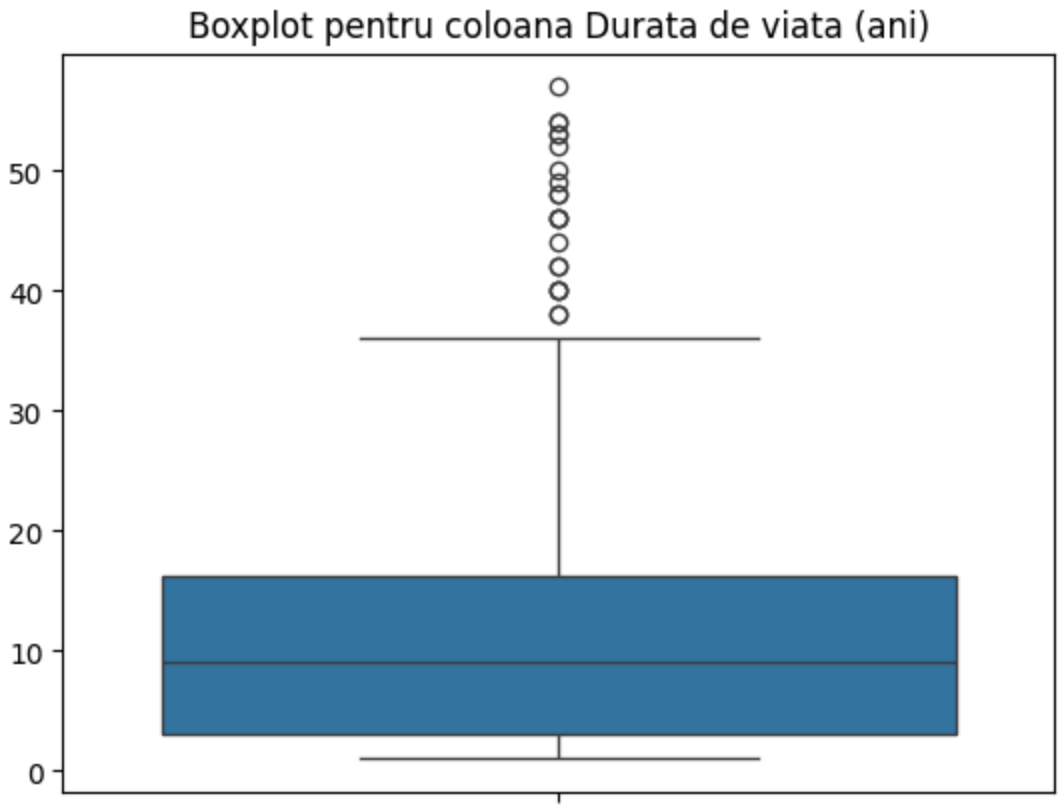
Vom continua cu setul Test.



1 Graficul reprezinta distributia valorilor din coloana Greutate (Kg). Majoriatatea valorilor sunt pana la 80 Kg (aproximativ).

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca apar valori de la 240 la 700 Kg.

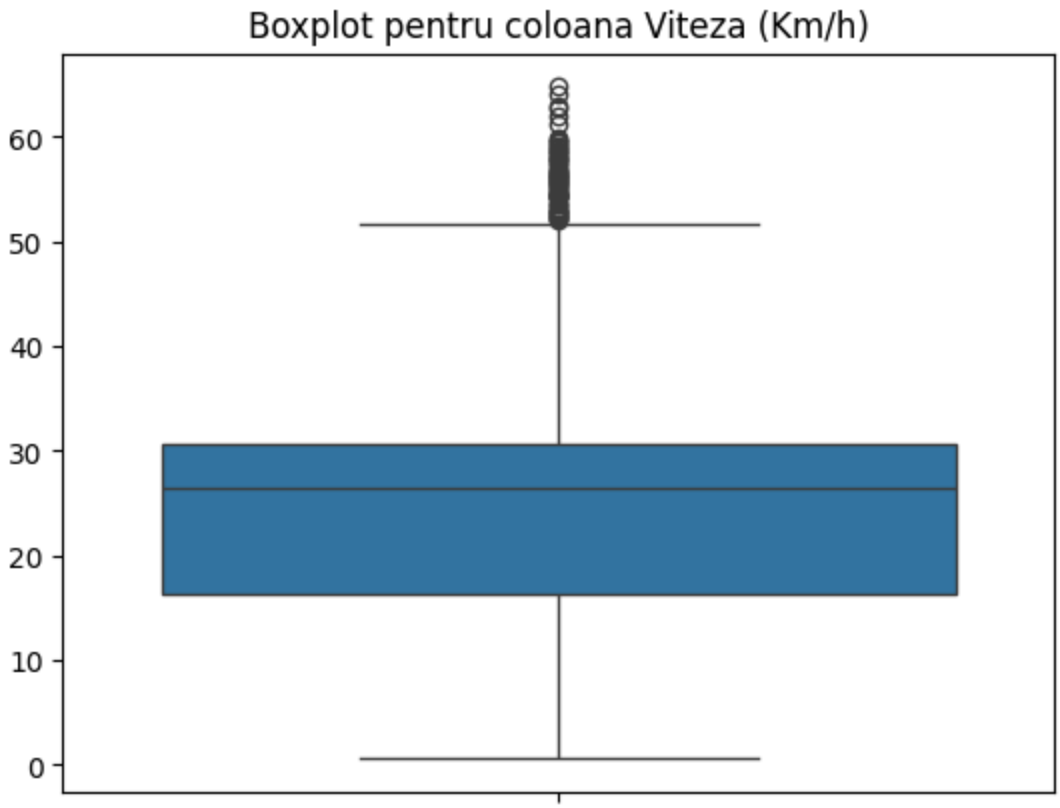
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Durata de viata (ani). Majoriatatea valorilor sunt pana la 17 ani (aproximativ). Mediana se afla la 9 ani.

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca apar valori de la 37 la 60 ani.

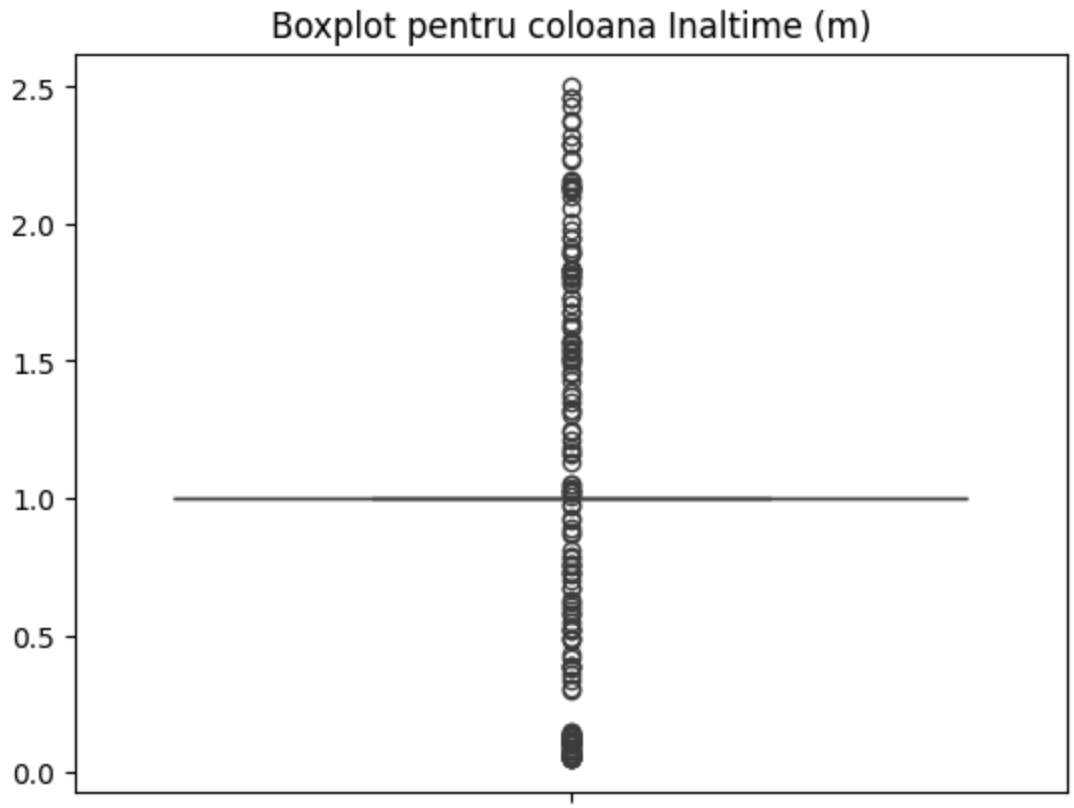
3 Putem aplica o normalizare a acestor date.



1 Graficul reprezinta distributia valorilor din coloana Viteza (Km/h). Majoriatatea valorilor sunt de la 17 pana la 30 Km/h (aproximativ). Mediana se afla la 27 Km/h (aproximativ).

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca apar valori de la 53 la 70 Km/h.

3 Putem aplica o normalizare a acestor date.



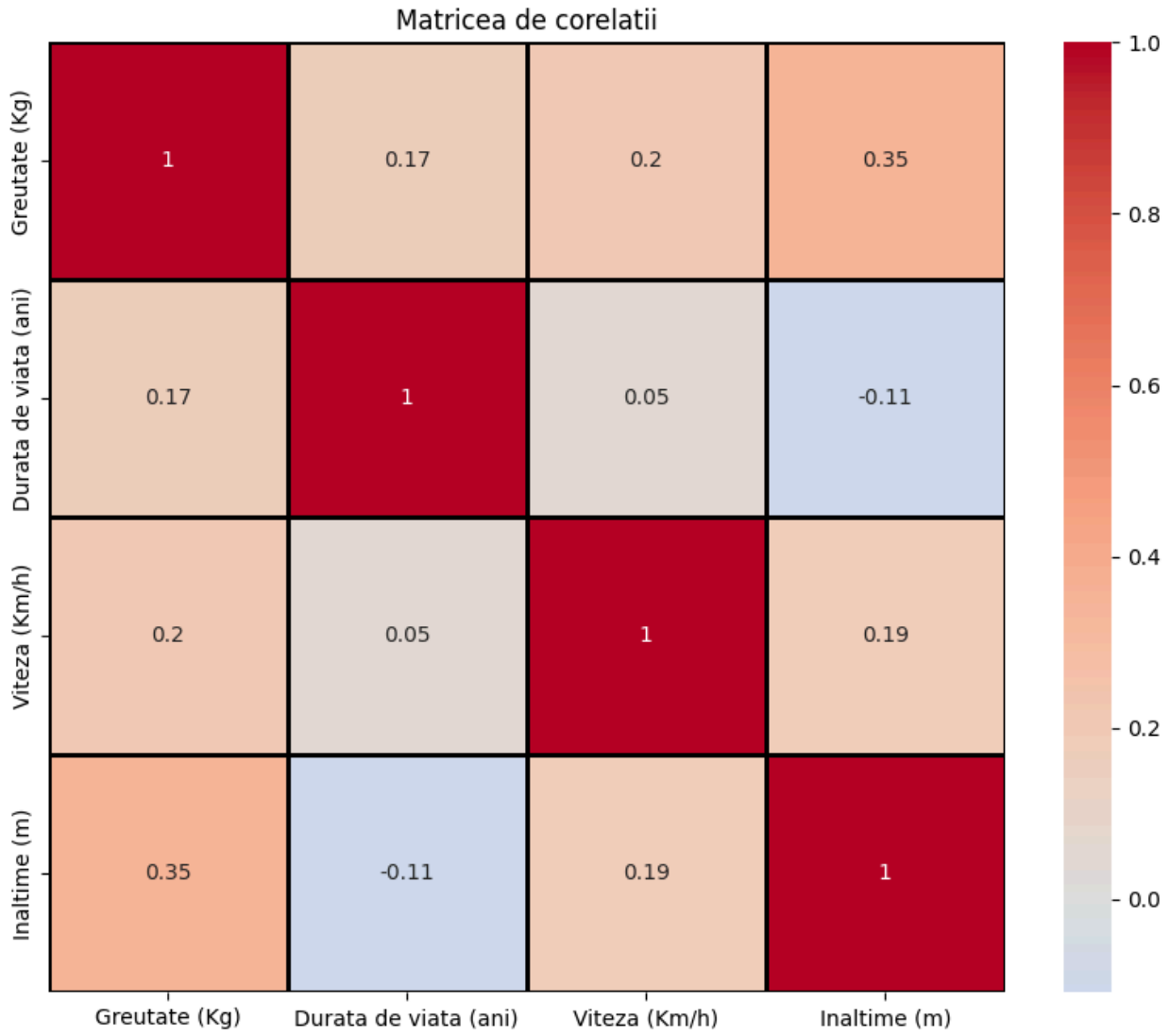
1 Graficul reprezinta distributia valorilor din coloana Inaltime (m). Majoriatatea valorilor sunt la 1 m (aproximativ).

2 Putem presupune ca vor fi outliere in aceasta coloana din cauza ca apar valori de la 1 la 0 m si de la 1 la 2.5 m.

3 Putem aplica o normalizare a acestor date.

7. Analiza corelatiilor

Heatmap ul pentru Train dintre coloanele numerice:

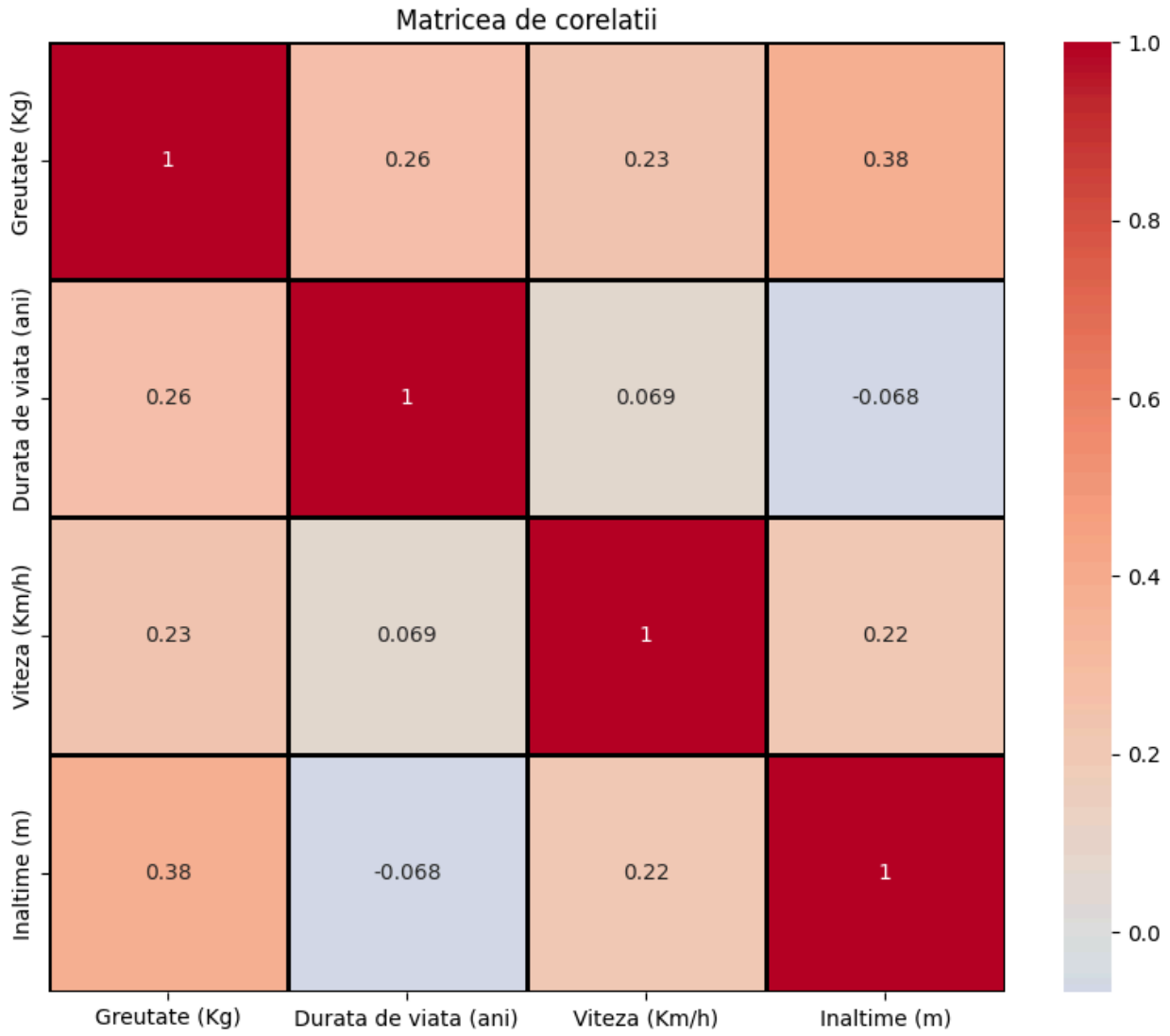


1 Aceasta este matricea de corelatii (heatmap) intre coloanele numerice. Avem o corelatie moderata pozitiva intre Greutate (Kg) si Inaltime (m) de 0.35, o corelatie mai slaba tot pozitiva intre Greutate (Kg) si Viteza (Km/h) de 0.2, o corelatie aproape asemanatoare intre Viteza (Km/h) si Inaltime (m) de 0.19, inca una putin mai slaba intre Durata de viata (ani) si Greutate (Kg) de 0.17, iar restul sunt corelatii slabe pozitive sau negative.

2 Nu exista corelatii foarte puternice, deci putem pastra valorile.

3 Pastram valorile cum sunt.

Heatmap ul pentru Test dintre coloanele numerice:



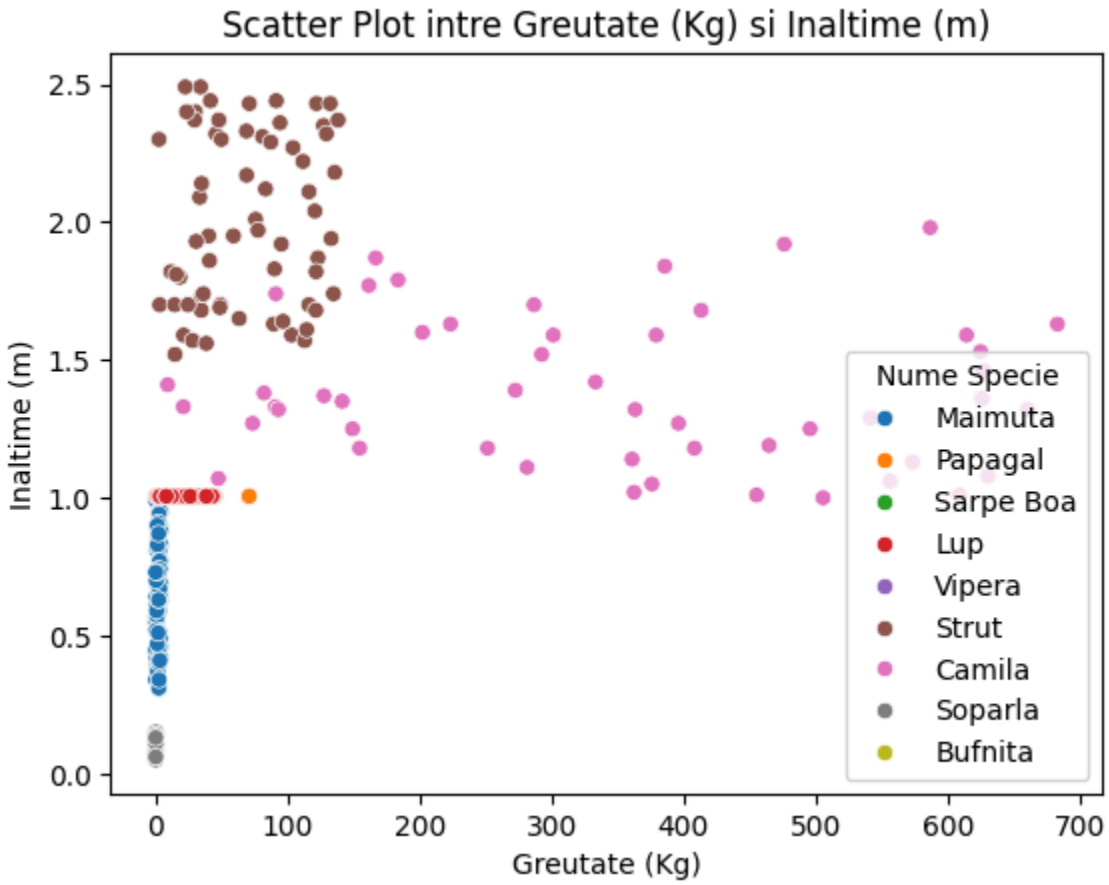
1 Aceasta este matricea de corelatii (heatmap) intre coloanele numerice. Avem o corelatie moderata pozitiva intre Greutate (Kg) si Inaltime (m) de 0.38, o corelatie mai slaba tot pozitiva intre Greutate (Kg) si Viteza (Km/h) de 0.23, o corelatie aproape asemanatoare intre Viteza (Km/h) si Inaltime (m) de 0.22, inca una putin mai puternica intre Durata de viata (ani) si Greutate (Kg) de 0.26, iar restul sunt corelatii slabe pozitive sau negative.

2 Nu exista corelatii foarte puternice, deci putem pastra valorile.

3 Pastram valorile cum sunt.

8. Analiza relatiilor cu variabila tinta

Vom incepe cu Train:

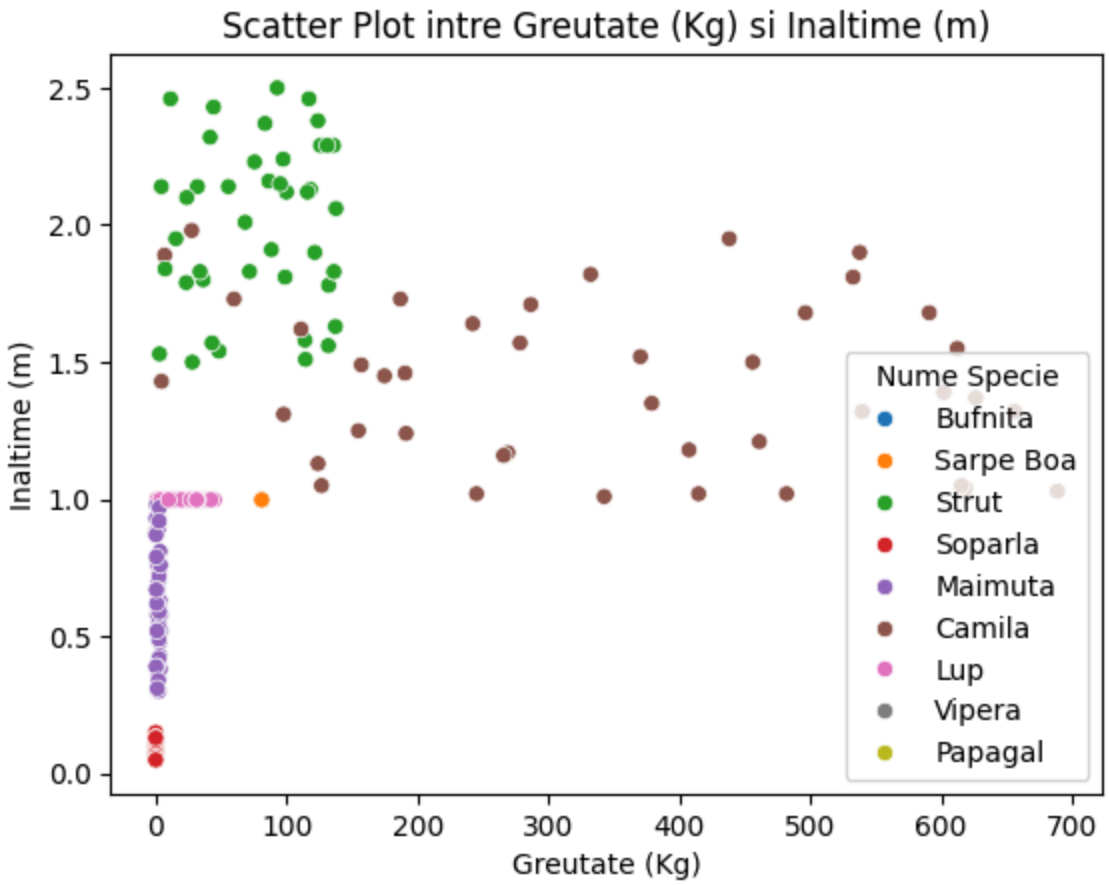


1 Acesta e un grafic ce reprezinta relatiile dintre coloanele Greutate (Kg) si Inaltime (m) cu cea Nume Specie. Speciile sunt bine separate una de alta in functie de cele 2 caracteristici.

2 Aceste 2 caracteristici sunt foarte relevante pentru distingerea numelor pentru ca sunt bine delimitate.

3 Se aplica normalizarea ambelor coloane.

Vom continua cu Test:



1 Acesta e un grafic ce reprezinta relatiile dintre coloanele Greutate (Kg) si Inaltime (m) cu cea Nume Specie. Speciile sunt bine separate una de alta in functie de cele 2 caracteristici.

2 Aceste 2 caracteristici sunt foarte relevante pentru distingerea numelor pentru ca sunt bine delimitate.

3 Se aplica normalizarea ambelor coloane.

9. Pregatirea datelor

Punem in X_train si X_test datele din train si test fara coloana Nume Specie pentru ca pe aia o prezicem si pentru asta folosim .drop(columns=[‘coloana’]). In y_test si y_train punem doar coloana Nume Specie.

10. OneHotEncodingpentru datele categorice si Normalizare pentru datele numerice

Importam modelul LogisticRegression si StandardScaler. Facem 2 liste in care punem in una coloanele numerice si in cealalta pe cele categorice. Facem X_train_codificat si X_test_codificat prin functia pd.get_dummies(Matricea pe care o codificam, columns=lista coloanelor categorice, drop_first=True). Acum actualizam X_train_codificat si X_test_codificat aplicandule o normalizare pentru coloanele numerice cu ajutorul functiei scaler.fit_transform(X_train_codificat sau X_test_codificat [lista coloanelor numerice]).

11. Antrenarea modelului

Definim model ca fiind LogisticRegression cu maxim 1000 de iteratii si random_state = 42 (seed ul de la inceput), iar dupa ii dam fit cu X_train_codificat si y_train.

12. Evaluare pe test

Importam accuracy_score, classification_report, confusion_matrix si ConfusionMatrixDisplay, dar si seaborn. Facem vectorul y_prezis prin model.predict(X_test_codificat). Acuratetea va fi facuta cu accuracy_score dintre y_test si y_prezis si o afisam rotunjita cu 2 zecimale.

Preciza, recall si F1-score vor fi afisate cu classification_report tot dintre y_test si y_prezis.

Pentru matricea de confuzie folosim functia confusion_matrix intre y_test si y_prezis si facem un display care va fi ConfusionMatrixDisplay de matricea de confuzie si etichetele vor fi model.classes_. Facem un plot din display cu cmap Albastru ii dam si un titlu si rotim etichetele de pe axa x la 45 de grade pentru a nu se suprapune, punem denumiri la axele x si y si il afisam.

Pentru graficul de erori facem un dictionar ce are cheile real: y_test si prezis: y_prezis. Facem un dataframe cu acest dictionar si luam doar randurile in care real e diferit de prezis. Din aceste randuri scoatem cate sunt si le punem intr un barplot pe axa x indexul si pe axa y valoarea, paleta folosita este cea Rosie, iar hue va fi tot indexul. Facem aceleasi prelucrari ca la matricea de confuzie si afisam graficul de erori. Observam ca au fost gresiti 3 valori de Sarpe Boa si o valoare de Vipera.

Bonus folosirea GitHub: https://github.com/Theodor17/PCLP3