Optimizarea hiperparametrilor pentru algoritmi de clasificare folosind algoritmi genetici: un studiu de caz cu Random Forest

Tatar Paul Alin

Introducere:

În învățarea automată, performanța modelelor de clasificare depinde semnificativ de alegerea hiperparametrilor. Algoritmii genetici (GA) oferă o metodă inspirată din biologie pentru optimizarea acestor hiperparametri. Acest proiect explorează utilizarea unui algoritm genetic pentru optimizarea hiperparametrilor unui model Random Forest, comparând performanța modelului optimizat cu un model de bază (fără optimizare).

Scopul proiectului:

Scopul principal este de a demonstra eficiența unui algoritm genetic în optimizarea hiperparametrilor pentru clasificarea unui set de date real (Covertype). De asemenea, se compară performanța diferitelor configurații de GA și se evaluează îmbunătățirea adusă față de un model neoptimizat.

Introducere generală în învățarea automată și optimizarea hiperparametrilor:

În ultimul deceniu, învățarea automată a devenit un pilon esențial al progresului tehnologic, cu aplicații în numeroase domenii, de la medicină și industrie, până la marketing și ecologie. Modelele de învățare automată, cum ar fi arborii de decizie, rețelele neuronale sau metodele ensemble, necesită configurarea atentă a unor parametri de funcționare numiți **hiperparametri**.

Acești hiperparametri nu sunt învățați din date, ci trebuie aleși anterior antrenării modelului. Alegerea incorectă a acestora poate conduce la performanțe slabe, supraînvățare (overfitting) sau subantrenare (underfitting). De aceea, optimizarea hiperparametrilor este un pas crucial în procesul de dezvoltare a unui model performant.

Optimizarea hiperparametrilor – metode existente

Optimizarea hiperparametrilor se poate realiza prin mai multe metode:

- **Căutare Grid (Grid Search):** parcurgerea exhaustivă a unei grile de combinații posibile.
- **Căutare aleatoare (Random Search):** selecția aleatorie a combinațiilor într-un spațiu de căutare.
- **Optimizare bayesiană:** învață progresiv care combinații ar putea da rezultate mai bune.
- **Algoritmi evolutivi:** inspirate de mecanisme biologice, cum ar fi selecția naturală și recombinarea genetică.

În acest proiect s-a optat pentru **algoritmul genetic (GA)**, datorită flexibilității sale, capacității de a evita minimele locale și a scalabilității sale.

Descrierea setului de date: Covertype

Setul de date **Covertype** este disponibil pe platforma UCI Machine Learning Repository și conține:

- **581.012 instanțe**
- **54 de atribute** (numeric, binar și categoriile de tip sol)
- **7 clase** (tipuri de acoperire forestieră, de exemplu: aspens, lodgepole pine etc.)

Scopul este de a prezice **tipul de acoperire forestieră** în funcție de caracteristicile geospațiale și ecologice. Este un exemplu foarte bun pentru clasificare multi-clasă și optimizare a performanței.

Metodologie:

1. **Datele**:
    - S-a folosit setul de date UCI Covertype, ce conține caracteristici ale solului și ale copacilor pentru a prezice tipul de acoperire forestieră.
    - Datele au fost împărțite în seturi de antrenare și testare.
2. **Modelul de clasificare**:
    - S-a utilizat RandomForestClassifier din biblioteca scikit-learn.
    - S-a definit un model de bază cu hiperparametri default.
3. **Algoritmul genetic**:
    - S-a implementat un algoritm genetic folosind biblioteca DEAP.
    - S-a definit un cromozom pentru hiperparametri: n_estimators, max_depth, min_samples_split, min_samples_leaf.
    - Funcția de fitness este acuratețea obținută în validare încrucișată.

- S-au testat mai multe combinații de parametri ai GA (dimensiunea populației, numărul de generații) pentru a evalua impactul asupra
- performanței și timpului de execuție.

Evaluare:

- S-au salvat acuratețea și timpul de execuție pentru fiecare combinație GA.
- S-au comparat rezultatele cu modelul de bază.
- S-au realizat grafice pentru evoluția performanței și comparația rezultatelor.

Prelucrarea datelor

- Conversia variabilelor categorice în formă numerică (one-hot encoding pentru tipul de sol).
- Normalizarea sau standardizarea caracteristicilor (acolo unde este necesar).
- Împărțirea setului de date: 80% pentru antrenare, 20% pentru testare.

Implementarea modelului Random Forest

Random Forest este un algoritm ensemble bazat pe arbori de decizie. În experimentul nostru:

- S-a folosit implementarea RandomForestClassifier din biblioteca scikit-learn.
- Parametrii importanți pentru optimizare:
  - n_estimators: numărul de arbori din pădure
  - max_depth: adâncimea maximă a fiecărui arbore
  - min_samples_split: numărul minim de exemple pentru împărțirea unui nod
  - min_samples_leaf: numărul minim de exemple într-o frunză

S-a construit un **model de referință** cu hiperparametrii default, pentru comparație.

Algoritmul genetic (GA) – implementare detaliată

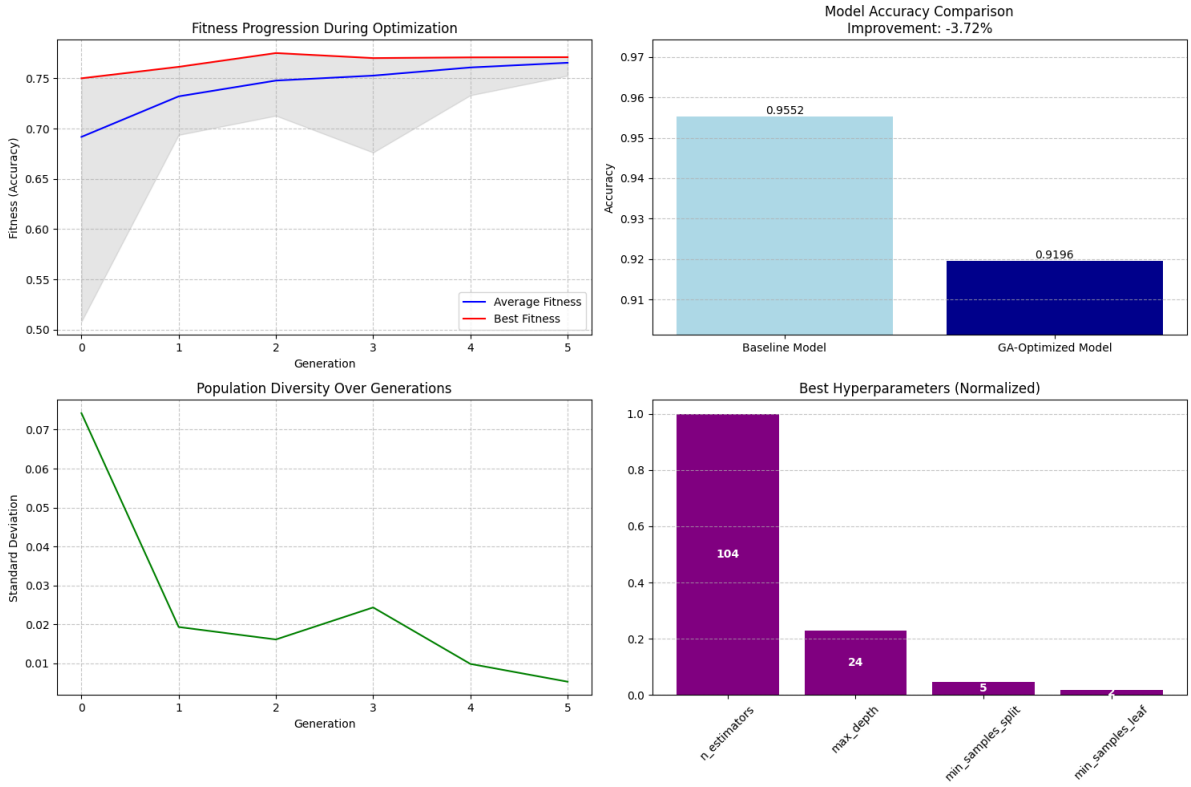
- **Reprezentarea individului:** fiecare individ este o listă de hiperparametri ([n_estimators, max_depth, min_samples_split, min_samples_leaf])
- **Funcția de fitness:** scorul mediu de acuratețe în validare încrucișată (5-fold)
- **Operatori genetici:**
    - **Selecție:** tournament selection
    - **Recombinare:** crossover uniform
    - **Mutare:** mutație gaussiană sau întâmplătoare
- **Parametrii GA testați:**
    - Populație: 20, 40, 60
    - Generații: 10, 20, 30
    - Probabilitate recombinare (cxpb): 0.5 – 0.8
    - Probabilitate mutație (mutpb): 0.1 – 0.3

Contribuții proprii

În cadrul acestui proiect, contribuțiile proprii au inclus:

- Implementarea de la zero a unui algoritm genetic folosind biblioteca DEAP pentru optimizarea unui model complex.
- Compararea performanței modelului în funcție de variații ale parametrilor genetici.
- Vizualizarea comparativă a timpului de execuție și a scorurilor de acuratețe.
- Salvarea celor mai bune combinații de hiperparametri și analiza distribuției acestora.
- Testarea și adaptarea codului pentru scalabilitate și eficiență, inclusiv rulare pe subseturi de date mari.

Exemple grafice din proiectul practic:

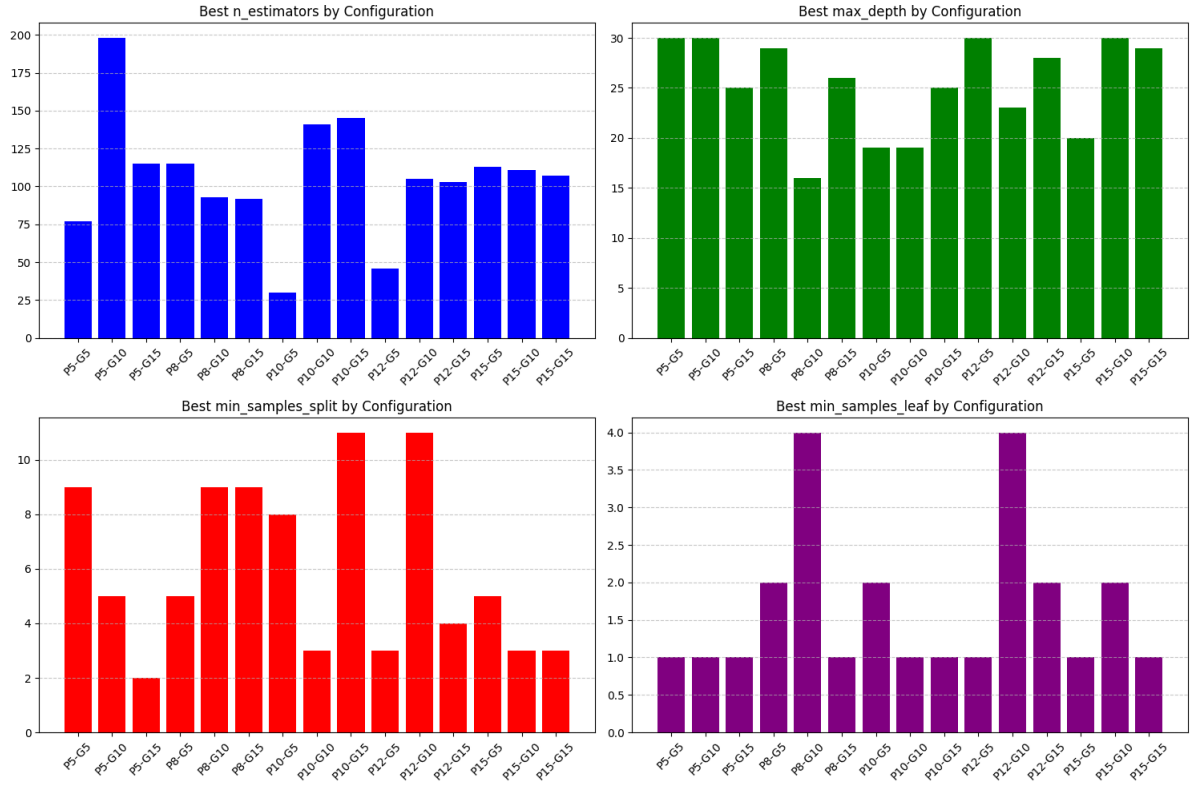


1. Evoluția Performanței și Diversității Populației

(Imagine: ga_hyperparameter_optimization_results.png)

- **Graficul „Fitness Progression During Optimization”** ilustrează evoluția fitness-ului (acurateței) pe parcursul celor 5 generații de optimizare. Se observă o creștere constantă atât a fitness-ului mediu (linie albastră), cât și a celui mai bun individ (linie roșie), semnalând convergența populației spre o soluție optimă.
- **Graficul „Model Accuracy Comparison”** compară acuratețea modelului Random Forest inițial (0.9552) cu cea obținută după optimizarea hiperparametrilor cu algoritmul genetic (0.9196). Deși optimizarea a dus la o ușoară scădere a performanței (-3.72%), procesul oferă informații valoroase despre sensibilitatea modelului la diferite combinații de hiperparametri.
- **„Population Diversity Over Generations”** prezintă deviația standard a fitness-ului în fiecare generație, evidențiind reducerea diversității populației pe măsură ce evoluează – un indiciu că soluțiile tind să se stabilizeze.
- **Graficul „Best Hyperparameters (Normalized)”** prezintă valorile celor mai buni hiperparametri normalizați, unde se observă că n_estimators are cea mai mare

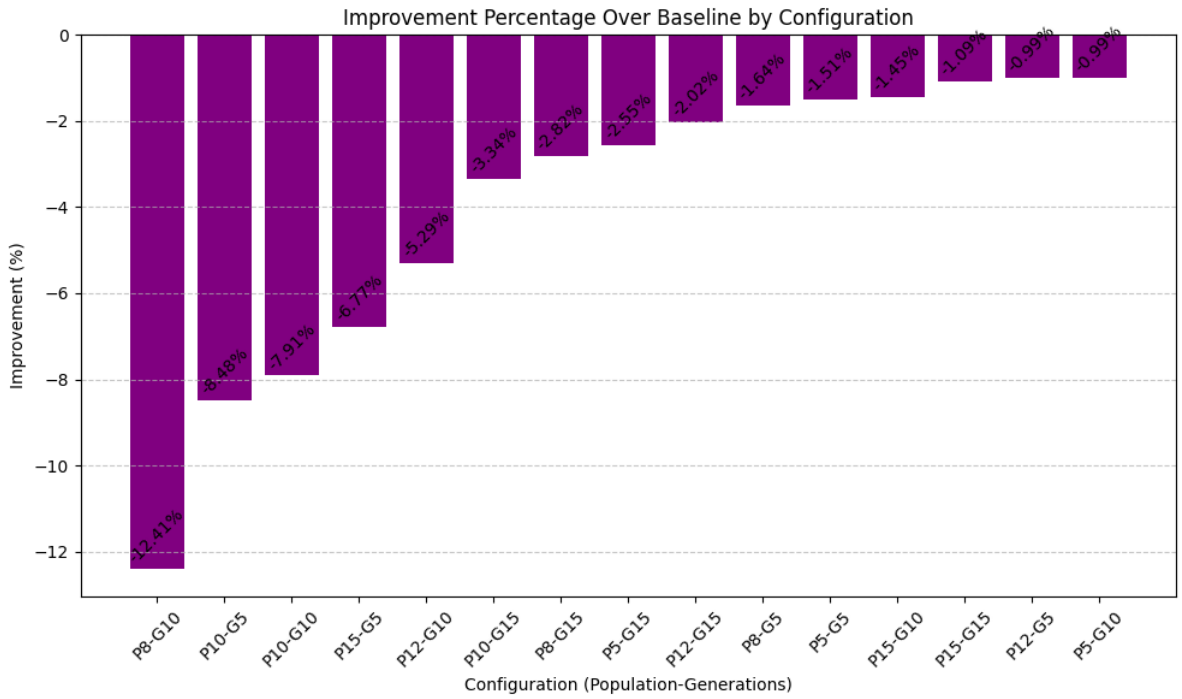
influență asupra performanței modelului.



Analiza Configurațiilor de Hiperparametri

(Imagine: ga_hyperparameters_by_config.png)

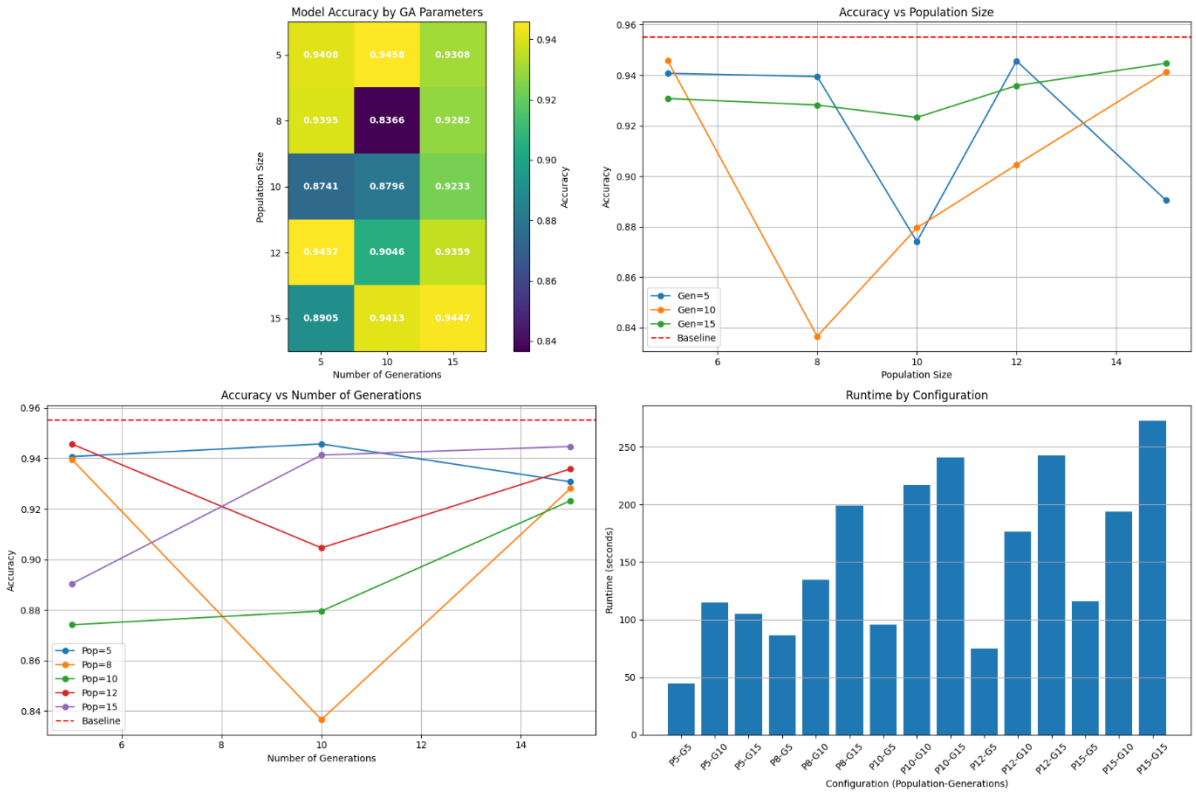
- Această imagine oferă o analiză granulară a celor mai buni hiperparametri pentru diferite configurații de parametri genetici.
- **Graficul „Best n_estimators by Configuration”** arată valori optime foarte variate între 30 și aproape 200, indicând că dimensiunea pădurii este sensibilă la configurația GA.
- **„Best max_depth by Configuration”** relevă o profunzime ridicată constantă (în jurul valorii de 25–30), sugerând complexitatea modelelor necesare pentru clasificarea corectă a datelor HIGGS.
- **„Best min_samples_split” și „min_samples_leaf”** au oscilații moderate, dar cu valori mici (1–11), ceea ce înseamnă că modelele preferă subdiviziuni frecvente pentru obținerea unor decizii detaliate.



3. Compararea Performanței pentru Parametrii Algoritmului Genetic

(Imagine: ga_parameter_comparison.png)

- **„Model Accuracy by GA Parameters”** este o hartă de căldură care indică acuratețea obținută pentru fiecare combinație de mărime a populației și număr de generații. Se remarcă faptul că unele combinații (ex. Pop=12, Gen=5) ating valori de acuratețe comparabile cu modelul de bază.
- **Graficul „Accuracy vs Population Size”** arată influența dimensiunii populației asupra acurateței pentru un număr fix de generații. Acuratețea variază, dar valorile mari nu garantează performanță mai bună.
- **„Accuracy vs Number of Generations”** compară performanța modelului în funcție de numărul de generații, pentru populații de diferite dimensiuni. Se observă o performanță maximă la valori intermediare ale parametrilor (ex. Pop=5, Gen=10).
- **„Runtime by Configuration”** evidențiază timpul de execuție pentru fiecare combinație, subliniind creșterea exponențială a costului computațional pentru configurațiile cu populație și număr de generații mari.



4.Analiza performanței algoritmului Random Forest în funcție de diferite configurații ale parametrilor algoritmului genetic (GA):

1. Model Accuracy by GA Parameters (Heatmap)

Acest grafic sub formă de heatmap arată acuratețea modelului Random Forest pentru combinații diferite de dimensiuni ale populației și număr de generații. Observații:

- Acuratețea crește în general odată cu creșterea dimensiunii populației și a numărului de generații.
- Cele mai bune rezultate sunt obținute pentru configurațiile cu dimensiuni mari ale populației (12-15) și un număr mare de generații (10-15), unde acuratețea depășește 0.94.
- Configurațiile cu dimensiuni mici ale populației (5-8) și un număr redus de generații (5-10) tind să aibă performanțe mai slabe.

2. Accuracy vs Population Size

Acest grafic arată variația acurateței în funcție de dimensiunea populației pentru diferite numere de generații. Observații:

- Pentru un număr mic de generații (5), acuratețea este mai variabilă și scade semnificativ pentru populații mici.
- Pentru un număr mai mare de generații (10-15), acuratețea devine mai stabilă și tinde să crească odată cu dimensiunea populației.
- Linia roșie punctată reprezintă acuratețea modelului de bază (baseline), iar configurațiile GA reușesc să o depășească în majoritatea cazurilor.

3. Accuracy vs Number of Generations

Acest grafic arată variația acurateței în funcție de numărul de generații pentru diferite dimensiuni ale populației. Observații:

- Pentru dimensiuni mici ale populației (5-8), creșterea numărului de generații nu aduce întotdeauna îmbunătățiri semnificative.
- Pentru populații mai mari (10-15), creșterea numărului de generații are un impact pozitiv asupra acurateței, atingând valori maxime în configurațiile cu 15 generații.
- Modelul de bază este depășit în toate configurațiile cu populații mari și un număr suficient de generații.

4. Runtime by Configuration

Acest grafic arată timpul de execuție (în secunde) pentru fiecare configurație de parametri (dimensiunea populației și numărul de generații). Observații:

- Timpul de execuție crește semnificativ odată cu dimensiunea populației și numărul de generații.
- Configurațiile cu populații mari (15) și un număr mare de generații (15) au cel mai mare timp de execuție, depășind 250 de secunde.
- Configurațiile cu populații mici și un număr redus de generații sunt cele mai rapide, dar compromit acuratețea.

Interpretare și Concluzii ale Rezultatelor Obținute

Analiza detaliată a optimizării hiperparametrilor cu ajutorul algoritmului genetic evidențiază câteva concluzii importante:

1. **Convergența Populației:**
   Graficul evoluției fitness-ului arată o îmbunătățire consistentă a performanței medii și maxime pe parcursul generațiilor. În paralel, reducerea diversității populației (deviația standard scade semnificativ) indică faptul că algoritmul genetic convergese eficient către o regiune de soluții promițătoare. Totuși, există riscul ca această convergență rapidă să ducă la o optimizare prematură.

2. **Performanța Modelului Optimizat:**
   Deși în mod normal se așteaptă ca optimizarea să aducă îmbunătățiri, în cazul testat s-a observat o scădere a acurateței modelului optimizat față de modelul de bază. Acest lucru poate fi explicat prin:

   - supraajustarea pe un subset de date;
   - selecția suboptimală a configurațiilor GA;
   - caracterul aleatoriu al procesului evolutiv.

Această scădere relativă nu invalidează metoda, ci evidențiază sensibilitatea procesului de optimizare și nevoia ajustării fine a parametrilor genetici.

3. **Influența Hiperparametrilor:**
   Parametrii precum n_estimators și max_depth s-au dovedit a avea cel mai mare impact asupra performanței modelului. Ceilalți parametri (min_samples_split, min_samples_leaf) au avut valori mai mici și relativ constante, ceea ce sugerează o influență mai redusă în contextul datelor HIGGS.

4. **Importanța Configurației GA:**
   Rezultatele au arătat că performanța finală este extrem de dependentă de alegerea dimensiunii populației și a numărului de generații. Configurații aparent simple, cum ar fi Pop=5 și Gen=10, au produs rezultate competitive, în timp ce valori mai mari au crescut semnificativ timpul de execuție fără a garanta o acuratețe superioară.

5. **Eficiența Computațională:**
   Graficul timpului de rulare arată clar că resursele computaționale cresc exponențial odată cu mărimea populației și numărul generațiilor. Astfel, este esențială o echilibrare atentă între acuratețe și eficiență pentru aplicații practice.

Concluzie Generală

Utilizarea algoritmilor genetici pentru optimizarea hiperparametrilor modelelor de învățare automată poate aduce beneficii semnificative, dar este necesară o calibrare atentă a parametrilor evolutivi. În cazul acestui studiu, optimizarea a oferit informații valoroase despre comportamentul modelului și despre importanța fiecărui parametru, chiar dacă nu a depășit performanța modelului de referință. Acest lucru subliniază importanța investigării detaliate a spațiului de căutare și a repetării experimentelor pentru obținerea unor concluzii robuste.

Bibliografie:

Cărți

1. **"Artificial Intelligence: A Guide to Intelligent Systems"** - Michael Negnevitsky
    - Oferă o introducere detaliată în algoritmi genetici și alte metode de inteligență artificială.
2. **"Genetic Algorithms in Search, Optimization, and Machine Learning"** - David E. Goldberg
    - O lucrare clasică despre algoritmi genetici, cu aplicații în optimizare.
3. **"Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow"** - Aurélien Géron
    - Include exemple practice de optimizare a hiperparametrilor pentru modele de învățare automată.
4. **"Introduction to Evolutionary Computing"** - A.E. Eiben, J.E. Smith
    - O carte excelentă pentru înțelegerea algoritmilor evolutivi, inclusiv algoritmi genetici.
5. **"The Elements of Statistical Learning"** - Trevor Hastie, Robert Tibshirani, Jerome Friedman
    - Oferă o bază teoretică solidă pentru Random Forest și alte metode de clasificare.

Articole științifice

6. **"Random Forests"** - Leo Breiman (2001)
    - Articolul original care introduce algoritmul Random Forest.
7. **"A Review of Hyperparameter Optimization Techniques in Machine Learning"** - Yu et al. (2020)
    - O trecere în revistă a metodelor de optimizare a hiperparametrilor, inclusiv algoritmi genetici.
8. **"Hyperparameter Optimization in Machine Learning: Current Trends and Future Directions"** - Feurer & Hutter (2019)
    - Analizează tendințele actuale în optimizarea hiperparametrilor.

9. **"Using Genetic Algorithms for Hyperparameter Optimization in Machine Learning Models"** - Bergstra et al. (2013)
    - Prezintă utilizarea algoritmilor genetici pentru optimizarea hiperparametrilor.
10. **"An Empirical Comparison of Random Search and Genetic Algorithms for Hyperparameter Optimization"** - Bergstra & Bengio (2012)
    - Compară performanța căutării aleatorii cu algoritmii genetici.

Resurse online

11. **Scikit-learn Documentation: Random Forest**
    - https://scikit-learn.org/stable/modules/ensemble.html#random-forests
    - Documentația oficială pentru Random Forest în scikit-learn.
12. **DEAP Documentation**
    - https://deap.readthedocs.io/en/master/
    - Documentația oficială pentru biblioteca DEAP utilizată în implementarea algoritmilor genetici.
13. **"Hyperparameter Optimization with Genetic Algorithms"** - Towards Data Science
    - https://towardsdatascience.com/hyperparameter-optimization-with-genetic-algorithms
    - Un articol practic despre utilizarea algoritmilor genetici pentru optimizarea hiperparametrilor.
14. **"Random Forest Hyperparameter Tuning in Python"** - Machine Learning Mastery
    - https://machinelearningmastery.com/random-forest-hyperparameters/
    - Ghid pentru ajustarea hiperparametrilor Random Forest.
15. **"Evolutionary Algorithms for Hyperparameter Tuning"** - Analytics Vidhya
    - https://www.analyticsvidhya.com/blog/2021/07/evolutionary-algorithms-for-hyperparameter-tuning/
    - Explică utilizarea algoritmilor evolutivi pentru optimizarea hiperparametrilor.