Predicția Aglomerării în Sala de Fitness folosind Modele de Învățare Automată

Crișan Antonel Gabriel

Mai 2024

Cuprins

Capitolul 1

Introducere

1.1 Motivația alegerii bazei de date

Datele sunt esențiale pentru luarea deciziilor informate în diverse domenii. Sala de fitness reprezintă un loc unde monitorizarea și optimizarea fluxului de persoane poate aduce beneficii atât pentru administratori, cât și pentru utilizatori. Baza de date aleasă conține informații relevante despre numărul de persoane prezente în sală, ziua săptămânii, dacă este weekend, dacă este vacanță, temperatura, dacă este început de semestru, dacă este în timpul semestrului, luna și ora. Ceea ce ne permite să aplicăm metode de învățare automată pentru a realiza predicții precise. Motivația principală pentru alegerea acestei baze de date este posibilitatea de a îmbunătăți gestionarea resurselor și a spațiilor în sălile de fitness din campus, oferind o experiență mai bună pentru studenți.

Capitolul 2

Contextul bazei de date și al proiectului

2.1 Cerințe

Proiectul necesită o analiză detaliată a datelor disponibile pentru a dezvolta un model de predicție a aglomerării în sala de fitness din campus. Cerințele specifice includ:

- Colectarea datelor relevante despre numărul de studenți prezenți, condițiile meteorologice, timpul din zi și alți factori influenți.
- Preprocesarea datelor pentru a elimina anomaliile și a asigura integritatea acestora.
- Aplicarea unor modele de învățare automată pentru a realiza predicții precise.
- Evaluarea performanței modelelor prin metrici de evaluare adecvate.

2.2 Obiective

Scopul principal al proiectului este de a dezvolta un model de învățare automată capabil să prezică numărul de studenți prezenți în sala de fitness în diferite condiții. Obiectivele specifice includ:

- Explorarea și înțelegerea bazei de date.
- Implementarea și compararea mai multor modele de învățare automată, cum ar fi regresia liniară, rețelele neurale și random forest.
- Analiza performanței modelelor și selectarea celui mai bun model pentru predicții viitoare.

Capitolul 3

Aspecte teoretice relevante

3.1 Starea actuală a domeniului

Învățarea automată a devenit un instrument esențial în analiza datelor și predicția comportamentului uman în diverse contexte. În contextul sălilor de fitness, monitorizarea și predicția aglomerării pot ajuta la îmbunătățirea experienței utilizatorilor și la optimizarea resurselor.

3.1.1 Modele de Învățare Automată

În acest proiect, am explorat diferite modele de învățare automată, inclusiv regresia liniară, rețelele neurale artificiale (ANN) și pădurile aleatorii (Random Forest). Fiecare dintre aceste modele au avantaje și dezavantaje specifice.

3.1.2 Entropia și Indexul Gini

Entropia și indexul Gini sunt două măsuri de impuritate utilizate pentru a evalua calitatea unei împărțiri în arborii de decizie.

- **Entropia** este o măsură a incertitudinii sau a impurității într-un set de date. Formula entropiei este:

$$H(S) = -\sum_{i=1}^{c} p_i \log_2 p_i$$

  unde p_i este proporția elementelor din clasa i în setul S.

- **Indexul Gini** este o altă măsură a impurității, definită ca probabilitatea ca un element ales aleator să fie clasificat greșit dacă ar fi etichetat conform distribuției clasei în subset. Formula indexului Gini este:

$$G(S) = 1 - \sum_{i=1}^{c} p_i^2$$

  unde p_i este proporția elementelor din clasa i în setul S.

Aceste măsuri sunt utilizate pentru a decide care caracteristică să fie aleasă la fiecare pas al construirii unui arbore de decizie, astfel încât impuritatea să fie minimizată.

3.1.3 Regresia liniară

Regresia liniară este o metodă statistică ce modelează relația dintre una sau mai multe variabile independente și o variabilă dependentă folosind o linie de regresie. Este utilizată pentru predicții simple și interpretări ușor de înțeles.

3.1.4 Rețelele Neurale Artificiale (ANN)

Rețelele neurale artificiale sunt modele de învățare automată inspirate de structura și funcționarea creierului uman. Acestea sunt compuse din neuroni artificiali organizați în straturi și sunt capabile să învețe relații complexe dintre date.

3.1.5 Random Forest

Random Forest este un algoritm de învățare automată robust și flexibil, care combină multipli arbori de decizie pentru a obține o predicție precisă. Este cunoscut pentru capacitatea sa de a gestiona seturi de date mari și complexe și de a evita supraînvățarea.

3.2 Literatura de specialitate

Pentru a dezvolta și evalua corect modele de predicție, este esențial să ne bazăm pe literatură de specialitate. Printre lucrările relevante se numără:

1 James, G., Witten, D., Hastie, T., and Tibshirani, R. (2013). An Introduction to Statistical Learning. Springer.

2 Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.

3 Haykin, S. (1998). Neural Networks: A Comprehensive Foundation. Prentice Hall.

4 Vapnik, V. N. (1995). The Nature of Statistical Learning Theory. Springer.

5 Cortes, C., and Vapnik, V. (1995). Support-vector networks. Machine learning, 20(3), 273-297.

6 LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. Nature, 521(7553), 436-444.

7 Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep Learning. MIT Press.

8 Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective. MIT Press.

9 Friedman, J., Hastie, T., and Tibshirani, R. (2001). The Elements of Statistical Learning. Springer.

10 Chollet, F. (2017). Deep Learning with Python. Manning Publications.

Capitolul 4

Implementarea aspectelor teoretice în cadrul proiectului

4.1 Preprocesarea Datelor

Datele au fost preprocesate pentru a elimina orice informație irelevantă sau redundantă. Au fost eliminate coloanele de timestamp și alte variabile care nu contribuie direct la predicția numărului de persoane.

4.2 Utilizarea Entropiei și Indexului Gini

Pentru a evalua dezordinea datelor, am calculat entropia și indexul Gini pentru variabilele noastre. Aceste măsuri ne ajută să înțelegem distribuția datelor și să luăm decizii informate în procesul de modelare.

4.2.1 Entropia datelor

```
import pandas as pd
from sklearn.tree import DecisionTreeRegressor

# Citirea datelor din fișierul CSV într-un DataFrame pandas

# Separarea caracteristicilor și variabilei de răspuns
X = dataSetForLearning  # Excludem primele două coloane
(datetime și numărul de persoane în sală)
pentru caracteristici
y = dataPeople   # Prima coloană
(numărul de persoane în sală)
este variabila de răspuns

# Inițierea unui model de regresie cu arbore de decizie
regressor = DecisionTreeRegressor(random_state=42)

# Antrenarea modelului pentru a calcula importanța caracteristicilor
regressor.fit(X, y)
```

```
# Extrage importanța caracteristicilor (indicele Gini) din model
importanta_caracteristici = regressor.feature_importances_

# Crearea unui DataFrame pentru a afișa importanța caracteristicilor
importanta_df = pd.DataFrame({'Caracteristica': X.columns, 'Importanta':
importanta_caracteristici})

# Sortarea DataFrame după importanță pentru a
vedea cele mai importante caracteristici
importanta_df = importanta_df.sort_values(by='Importanta', ascending=False)
print("0 -> Ziua saptamanii")
print("1 -> Daca este weekend")
print("2 -> Daca este vacanta")
print("3 ->  Temperatura in fahrenheit")
print("4 -> Daca este inceput de semestru")
print("5 -> Daca este in timpul semestrului")
print("6 -> Luna")
print("7 -> Ora")
print(importanta_df)
```

4.2.2 Indexul Gini

```
import pandas as pd
import numpy as np

def entropy(data):
    # Calcularea frecvenței fiecărei valori în setul de date
    value_counts = data.value_counts(normalize=True)

    # Calcularea entropiei
    entropy = -np.sum(value_counts * np.log2(value_counts))

    return entropy


# Eliminați a doua coloană (coloana de dată)
dataL = dataAfterCleaning.drop(columns=[1])

# Iterați prin fiecare coloană și calculați entropia pentru fiecare
for col in dataAfterCleaning.columns:
    if col != 1:  # Excludem a doua coloană (coloana de dată)
        entropy_col = entropy(dataAfterCleaning[col])
        print("Entropia pentru coloana", col, ":", entropy_col)
```

4.3 Algoritmi de Învățare Automată

Pentru implementare, am utilizat Regresia Liniară, acest algoritm este utilizat pentru predicții simple și interpretări ușor de înțeles, Rețele Neurale Artificiale, acest algoritm este un model de învățare automată inspirată de structura și funcționarea creierului uman și Random Forest Regressor, cunoscut pentru performanța sa în predicțiile pe seturi de date mari și complexe.

4.3.1 Regresia liniară

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error
import matplotlib.pyplot as plt



# Separarea caracteristicilor și variabilei de răspuns
X = dataSetForLearning  # Excludem primele două coloane (datetime și numărul
de persoane în sală) pentru caracteristici
y = dataPeople  # Prima coloană
(numărul de persoane în sală) este variabila de răspuns
y = y.values.ravel()

# Divizarea datelor în set de antrenament și set de testare
X_train, X_test, y_train,
y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Inițializarea și antrenarea modelului de regresie liniară
model = LinearRegression()
model.fit(X_train, y_train)

# Evaluarea modelului pe setul de testare
score = model.score(X_test, y_test)
print("Scorul regresiei liniare:", score)


# Predicțiile făcute de modelul de regresie liniară
predictions_linear_regression = model.predict(X_test)

# Calcularea erorii absolute medii (MAE)
mae_ann = mean_absolute_error(y_test, predictions_linear_regression)
print("Eroarea absolută medie (MAE):", mae_ann)

# Calcularea erorii medie pătratice (MSE)
mse_ann = mean_squared_error(y_test, predictions_linear_regression)
print("Eroarea medie pătratică (MSE):", mse_ann)
```

```
# Trasarea graficului
plt.figure(figsize=(10, 6))
plt.scatter(y_test, predictions_linear_regression, color='green',
label='Valori prezise (Regresie Liniară)')
plt.plot(y_test, y_test, color='red', linestyle='--', label='Linie de regresie')
plt.title('Predicțiile regresiei liniare vs. Valori reale')
plt.xlabel('Valori reale')
plt.ylabel('Valori prezise')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

4.3.2 Retele Neurale Artificiale

```
from sklearn.neural_network import MLPRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error
import matplotlib.pyplot as plt
# Separarea caracteristicilor și variabilei de răspuns
X = dataSetForLearning  # Excludem primele două coloane (datetime și
numărul de persoane în sală)
pentru caracteristici
y = dataPeople   # Prima coloană
(numărul de persoane în sală) este variabila de răspuns
y = y.values.ravel()
# Divizarea datelor în set de antrenament și set de testare
X_train, X_test, y_train,
y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Inițializarea și antrenarea modelului de rețea neurală pentru regresie
ann_model = MLPRegressor(hidden_layer_sizes=(100,), max_iter=1000)
ann_model.fit(X_train, y_train)

# Evaluarea modelului pe setul de testare
ann_score = ann_model.score(X_test, y_test)
print("Scorul rețelei neurale pentru regresie:", ann_score)

predictions_ann = ann_model.predict(X_test)

# Calcularea erorii absolute medii (MAE)
mae_ann = mean_absolute_error(y_test, predictions_ann)
print("Eroarea absolută medie (MAE):", mae_ann)

# Calcularea erorii medie pătratice (MSE)
mse_ann = mean_squared_error(y_test, predictions_ann)
print("Eroarea medie pătratică (MSE):", mse_ann)
```

```
# Trasarea graficului
plt.figure(figsize=(10, 6))
plt.scatter(y_test, predictions_ann, color='green', label='Valori prezise (ANN)')
plt.plot(y_test, y_test, color='red', linestyle='--', label='Linie de regresie')
plt.title('Predicțiile rețelei neurale artificiale (ANN) vs. Valori reale')
plt.xlabel('Valori reale')
plt.ylabel('Valori prezise')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

4.3.3 Random Forest Regressor

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error
import matplotlib.pyplot as plt

# Separarea caracteristicilor și variabilei de răspuns
X = dataSetForLearning  # Excludem primele două coloane (datetime și număru
l de persoane în sală) pentru caracteristici
y = dataPeople   # Prima coloană (numărul de
persoane în sală) este variabila de răspuns
y = y.values.ravel()

# Divizarea datelor în set de antrenament și set de testare
X_train, X_test,
y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Inițializarea și antrenarea modelului Random Forest Regressor
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# Evaluarea modelului pe setul de testare
rf_score = rf_model.score(X_test, y_test)
print("Scorul Random Forest Regressor:", rf_score)

# Predicțiile făcute de model
predictions_rf = rf_model.predict(X_test)

# Calcularea erorii absolute medii (MAE)
mae_rf = mean_absolute_error(y_test, predictions_rf)
print("Eroarea absolută medie (MAE) pentru Random Forest:", mae_rf)

# Calcularea erorii medie pătratice (MSE)
mse_rf = mean_squared_error(y_test, predictions_rf)
```

```
print("Eroarea medie pătratică (MSE) pentru Random Forest:", mse_rf)

# Trasarea graficului
plt.figure(figsize=(10, 6))
plt.scatter(y_test, predictions_rf,
color='green', label='Valori prezise (Random Forest)')
plt.plot(y_test, y_test, color='red', linestyle='--', label='Linie de regresie')
plt.title('Predicțiile Random Forest vs. Valori reale')
plt.xlabel('Valori reale')
plt.ylabel('Valori prezise')
plt.legend()
plt.grid(True)
plt.tight_layout()
plt.show()
```

Capitolul 5

Testare și validare

5.1 Procedura de Testare și Validare

Pentru a evalua performanța modelelor, am folosit o procedură de validare încrucișată cu divizarea setului de date în seturi de antrenament și testare. Am împărțit datele în proporție de 80% pentru antrenament și 20% pentru testare, folosind funcția `train_test_split` din `scikit-learn`.

5.2 Seturile de Date

Setul de date utilizat pentru antrenare și testare a fost format din date despre numărul de persoane prezente într-o sală de fitness, împreună cu caracteristici precum ziua săptămânii, dacă este weekend, dacă este vacanță, temperatura în Fahrenheit, dacă este început de semestru, dacă este în timpul semestrului, luna și ora.

5.3 Metrici de Evaluare

Pentru a evalua performanța modelelor, am folosit următoarele metrici:

- **Scorul de determinare** (R^2): Măsoară proporția variabilității din variabila de răspuns explicată de model.
- **Eroarea Absolută Medie** (MAE): Măsoară media absolută a erorilor dintre valorile prezise și valorile reale.
- **Eroarea Medie Pătratică** (MSE): Măsoară media pătratelor erorilor dintre valorile prezise și valorile reale.

5.4 Metode de Validare

Am folosit validarea încrucișată pentru a asigura că modelele nu suferă de overfitting și pentru a evalua robustețea acestora. Validarea încrucișată presupune împărțirea setului de date în mai multe subseturi, antrenarea modelului pe un subset și testarea pe celălalt, repetând procesul pentru toate combinațiile posibile.

5.5 Evaluarea Performanței

Pentru fiecare model (Regresie Liniară, Rețele Neurale Artificiale, Random Forest), am calculat metricile de evaluare menționate și am analizat performanța pe setul de testare.

De asemenea, am calculat entropia și indexul Gini pentru a evalua impuritatea datelor.

Capitolul 6

Rezultate

6.1 Performanța Modelelor

Am obținut următoarele rezultate pentru fiecare model:

- **Regresie Liniară**:
  - **Timpul de executie**: 0.7s
  - **Scorul de determinare** (R^2): 0.5146
  - **Eroarea Absolută Medie** (MAE): 12.0921
  - **Eroarea Medie Pătratică** (MSE): 250.8425

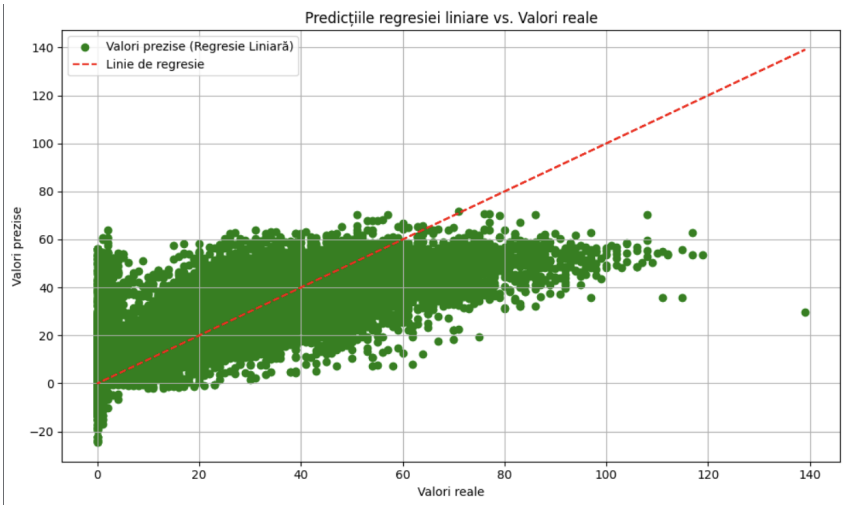

Figura 6.1: Predicțiile regresiei liniare vs. Valori reale

- **Rețele Neurale Artificiale**:
  - **Timpul de executie**: 2m 3.0s
  - **Scorul de determinare** (R^2): 0.6826
  - **Eroarea Absolută Medie** (MAE): 9.3169
  - **Eroarea Medie Pătratică** (MSE): 164.0282

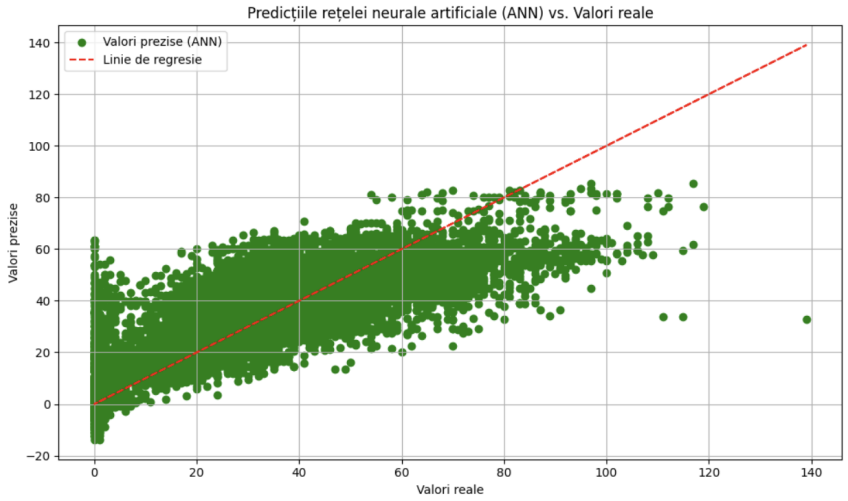

Figura 6.2: Predicțiile rețelelor neurale artificiale vs. Valori reale

- **Random Forest**:
  - **Timpul de executie**: 6.8s
  - **Scorul de determinare** (R^2): 0.9247
  - **Eroarea Absolută Medie** (MAE): 4.24
  - **Eroarea Medie Pătratică** (MSE): 38.92

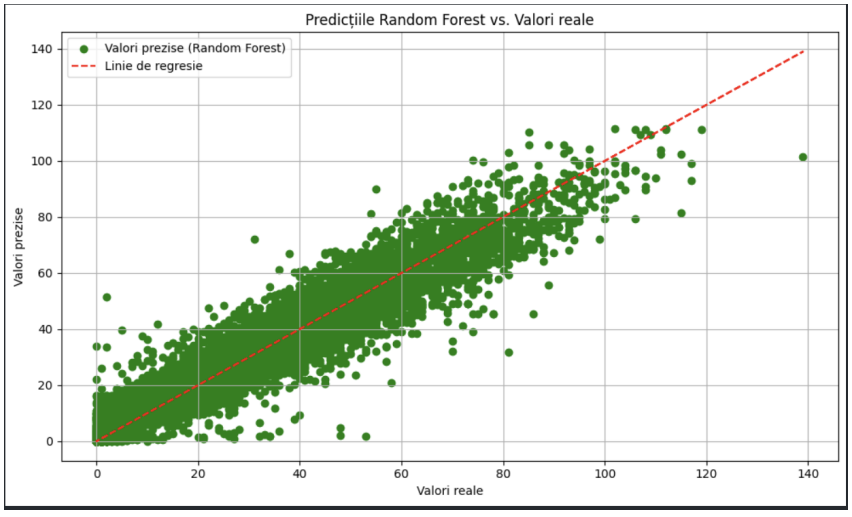

Figura 6.3: Predicțiile Random Forest vs. Valori reale

6.2 Entropie și Index Gini

În cadrul modelului Random Forest, am utilizat entropia și indexul Gini pentru a evalua impuritatea nodurilor din arborii de decizie. Aceste măsuri ne-au ajutat să alegem cele mai bune împărțiri la fiecare nod, minimizând astfel impuritatea și maximizând precizia modelului.

- **Entropie**:
  - Măsoară impuritatea sau dezordinea din date. O valoare mai mică a entropiei indică un nod mai pur. Figura 6.4 prezintă distribuția entropiei pentru datele utilizate în acest studiu.

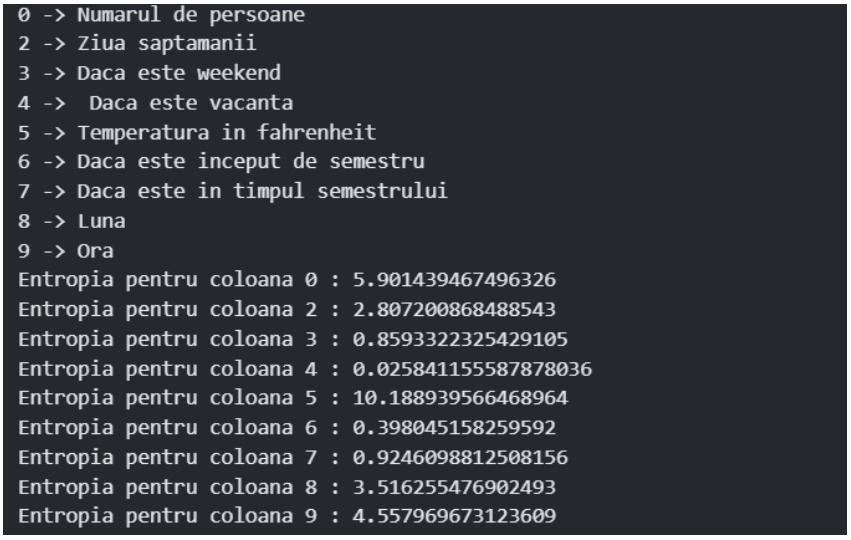

Figura 6.4: Entropia datelor

- **Index Gini**:
  - Măsoară probabilitatea ca un element ales aleator să fie clasificat incorect. Un index Gini mai mic indică un nod mai pur. Figura 6.5 prezintă distribuția indexului Gini pentru datele utilizate în acest studiu.

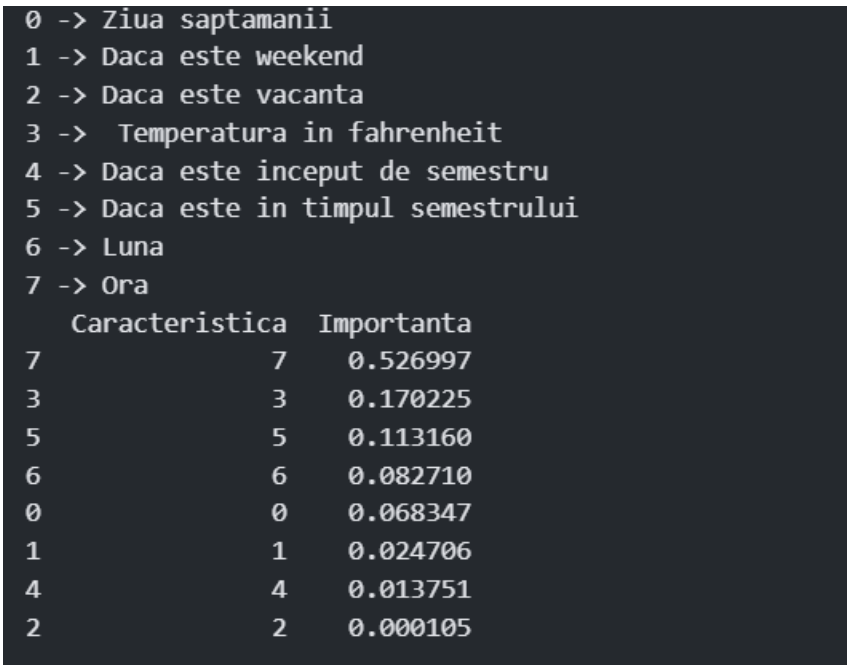

Figura 6.5: Indexul Gini

6.3 Rezultatele Modelului Random Forest pe Seturi Noi de Date

Pentru a evalua performanța modelului Random Forest pe seturi noi de date, am testat modelul cu un set de 301 instanțe de date diferite față de cele utilizate pentru antrenament. Scopul acestui test este de a verifica capacitatea modelului de a generaliza și de a face predicții precise pe date necunoscute.

6.3.1 Performanța Modelului

Rezultatele obținute în urma testării modelului pe noul set de date sunt următoarele:

- **Timpul de executie**: 1.1s
- **Scorul Random Forest Regressor**: 0.9262
- **Eroarea absolută medie (MAE)**: 3.143
- **Eroarea medie pătratică (MSE)**: 17.344

Aceste rezultate indică faptul că modelul Random Forest are o performanță ridicată și pe seturi noi de date, menținând un nivel scăzut de eroare și o capacitate bună de predicție.

6.3.2 Graficul Rezultatelor

Pentru a vizualiza mai clar performanța modelului Random Forest, am trasat un grafic care compară valorile reale cu valorile prezise de model pe setul de date de testare.

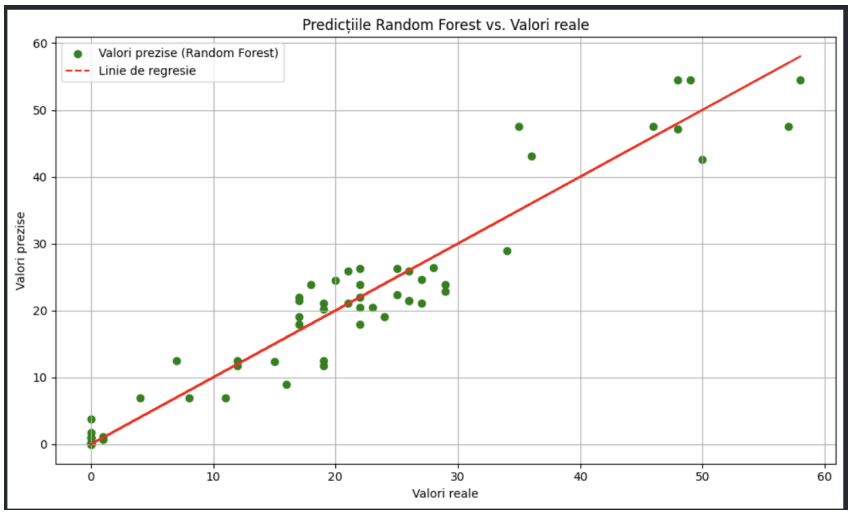

Figura 6.6: Predicțiile modelului Random Forest vs. Valori reale

În Figura 6.6, punctele verzi reprezintă valorile prezise de modelul Random Forest, în timp ce linia roșie punctată reprezintă linia de regresie ideală, unde valorile prezise sunt egale cu valorile reale. Apropierea punctelor verzi de linia roșie punctată indică precizia ridicată a modelului.

6.3.3 Interpretarea Rezultatelor

Performanța modelului Random Forest pe setul nou de date confirmă capacitatea acestuia de a generaliza și de a face predicții precise în condiții variate. Scorul ridicat și valorile scăzute ale MAE și MSE demonstrează că modelul poate gestiona eficient variabilitatea datelor și menține o precizie consistentă.

Rezultatele sugerează că modelul este robust și poate fi utilizat în scenarii reale pentru a prezice numărul de persoane din sala de fitness în funcție de diferite variabile. Acest lucru poate ajuta la optimizarea resurselor și la îmbunătățirea experienței utilizatorilor în sălile de fitness.

6.4 Compararea Performanței Modelelor

Comparând performanțele modelelor, Random Forest depășește cu mult atât regresia liniară, cât și rețelele neurale artificiale în ceea ce privește toate metricile de evaluare. Random Forest oferă un echilibru bun între performanță și complexitate.

Capitolul 7

Concluzii

7.1 Rezumatul Lucrării

În această lucrare, am abordat problema predicției numărului de studenți într-o sală de fitness din campus folosind tehnici de învățare automată. Am început prin a motiva alegerea bazei de date și a proiectului, evidențiind importanța optimizării fluxului de persoane pentru îmbunătățirea gestionării resurselor și a experienței utilizatorilor. Am descris contextul bazei de date și cerințele proiectului, urmând să prezentăm aspectele teoretice relevante, inclusiv entropia și indexul Gini, precum și starea actuală a domeniului.

7.2 Metodologie și Implementare

Am implementat trei modele de învățare automată: regresia liniară, rețelele neurale artificiale (ANN) și Random Forest. Fiecare model a fost testat și evaluat pe baza unor măsuri de performanță precum scorul R^2, eroarea absolută medie (MAE) și eroarea medie pătratică (MSE).Am utilizat entropia și indexul Gini pentru a evalua impuritatea nodurilor și a selecta cele mai bune împărțiri.

7.3 Testare și Validare

Modelul Random Forest a fost testat pe un set nou de date pentru a evalua capacitatea sa de generalizare. Rezultatele obținute au demonstrat o performanță ridicată, cu un scor R^2 de 0.9262, un MAE de 3.143 și un MSE de 17.344. Aceste rezultate confirmă că modelul poate face predicții precise și pe date necunoscute, menținând un nivel scăzut de eroare.

7.4 Rezultate și Interpretare

Rezultatele obținute cu modelul Random Forest au fost comparate cu cele ale altor modele, demonstrând o superioritate clară în ceea ce privește precizia predicțiilor. Analiza graficelor de predicție a arătat o apropiere considerabilă a valorilor prezise de valorile reale, evidențiind capacitatea modelului de a capta relațiile dintre variabilele de intrare și variabila de răspuns.

7.5 Implicații și Utilizări Viitoare

Acest studiu sugerează că utilizarea modelului Random Forest pentru predicția numărului de studenți într-o sală de fitness dintr-un campus poate aduce beneficii semnificative în optimizarea gestionării resurselor și îmbunătățirea experienței utilizatorilor. Pe viitor, se pot explora metode de optimizare a hiperparametrilor modelului și extinderea setului de date pentru a îmbunătăți și mai mult performanța predicțiilor.

7.6 Concluzie Generală

În concluzie, aplicarea tehnicilor de învățare automată pentru predicția numărului de studenți într-o sală de fitness s-a dovedit a fi o soluție eficientă și precisă. Modelul Random Forest, în special, a demonstrat capacitatea de a face predicții exacte, ceea ce poate contribui semnificativ la optimizarea operațiunilor în sălile de fitness din orice campus. Acest studiu deschide calea pentru utilizarea pe scară largă a tehnologiilor de învățare automată în gestionarea resurselor și îmbunătățirea serviciilor în diverse domenii.

Bibliografie

[1] Breiman, L. (2001). Random forests. *Machine learning*, 45(1), 5-32.

[2] Quinlan, J. R. (1986). Induction of decision trees. *Machine learning*, 1(1), 81-106.

[3] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, 20(3), 273-297.

[4] Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science & Business Media.

[5] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, É. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12, 2825-2830.

[6] Bishop, C. M. (2006). *Pattern Recognition and Machine Learning*. Springer.

[7] Friedman, J., Hastie, T., & Tibshirani, R. (2001). *The Elements of Statistical Learning*. Springer Series in Statistics.

[8] Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep Learning*. MIT Press.

[9] Cover, T., & Hart, P. (1967). Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1), 21-27.

[10] Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2), 123-140.

[11] Ng, A. (2011). Sparse autoencoder. *CS294A Lecture Notes*, 72(2011), 1-19.

[12] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.