# Predicția numărului de unități vândute ale seturilor Lego

Moise Victor-Ştefan 333AA

## Cuprins



## 1 Introducere

În contextul vânzărilor de seturi Lego, predicția numărului de unități vândute reprezintă un aspect esențial pentru optimizarea stocurilor și strategiilor de vânzare. Scopul acestui referat este de a analiza și compara performanța a doi algoritmi de învățare automată utilizate pentru predicții: *Regresia Liniară* și *Random Forest*.
Ambii algoritmi au fost antrenați și testați pe un calculator normal, cu placă video RTX 3060 TI. Inițial, mi-am propus să folosesc doar *Regresia Liniară*, însă, când am rulat același cod pe un MacBook Air cu procesor M2 (fără ventilatoare), am observat că timpul de antrenare este semnificativ mai mare. De aceea, am căutat alternative pentru *Regresia Liniară*, iar printre algoritmii găsiți s-a numărat *Random Forest*.
Pentru a evalua performanța celor doi algoritmi, am calculat, în ambele cazuri, *MSE (Mean Squared Error)*, *RMSE (Root Mean Squared Error)*, *MAE (Mean Absolute Error)* si $R^2$. De asemenea, am măsurat timpul și memoria de care algoritmii au avut nevoie pentru antrenare și predicție.

# 2 Metodologie

## 2.1 Setul de date

Am generat setul de date folosind un scriptul *generate_dataset.py*. Pentru a prezice numărul de unități vândute (*units sold*), am utilizat primele șase coloane ale setului de date: *piece count*, *date released*, *genre*, *genre popularity*, *price* și *price per piece*. *Piece count* și *price per piece* sunt generate aleatoriu, iar pe baza acestora se calculează *price*. De asemenea, *date released* este și ea aleasă aleatoriu, între 01-01-2000 si 01-01-2024, iar *genre* și *genre popularity* sunt alese dintr-un dicționar predefinit *(genres.py)*.
Setul de date conține 1000 de linii (seturi lego). Numărul de linii al setului de date poate fi schimbat modificând variabila *N_SAMPLES* din *generate_dataset.py*. Am alocat pentru antrenare 80% din setul de date, iar pentru testare resul de 20%.
Proiectul este disponibil si pe GitHub in format complet, aici.

## 2.2 Algoritmi

Cei doi algoritmi pe care i-am folosit sunt *Regresia Liniară* și *Random Forest*. Am ales să antrenez câte un model pentru fiecare, pentru a putea compara particularitățile și performanțele amândurora.

### 2.2.1 Regresia Liniara

În urma antrenării modelului de predicție folosind algoritmul de *Regresie Liniară*, am obținut următoarele predicții (comparate cu valorile reale):

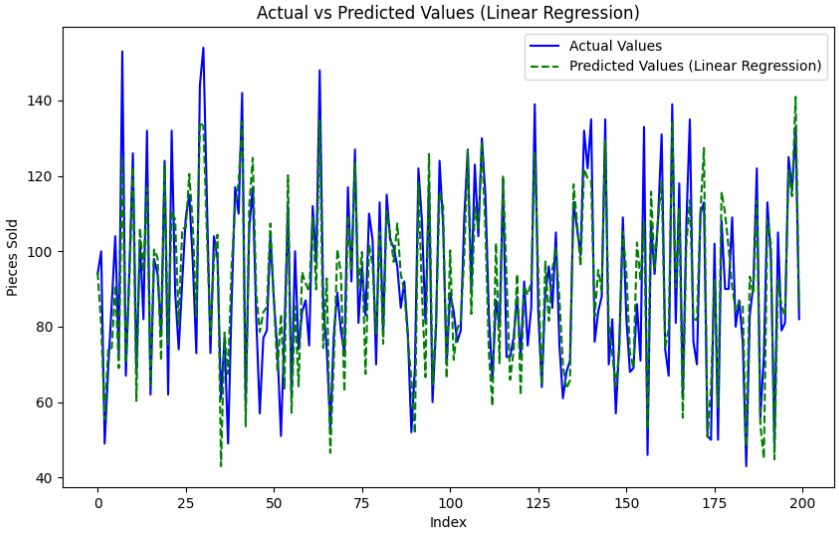


Figure 1: Linear Regression Results

În *Figure 1*, linia albastră reprezintă numărul de unități reale vândute, iar linia verde punctată reprezintă numărul de unități vândute prezise. Valorile reale sunt apropiate de cele prezise, cu mici abateri, ceea ce indică o performanță bună a modelului generat.
În termeni de performanță, am obținut următoarele date:

- **MSE**: 105.380
- **RMSE**: 10.265
- **MAE**: 8.352
- $R^2$: 0.816

Concluziile pe care le pot trage din datele prezentate mai sus sunt:

1. **RMSE** și **MSE** au o valoare relativ moderată în comparație cu valorile din coloana *units sold* din setul de date. Acest lucru indică faptul că modelul prezintă o eroare acceptabilă în etapa de predicție.
2. **MAE** de 8.352 sugerează că diferența medie absolută între valorile reale și cele prezise este relativ scăzută. Acest lucru confirmă o aliniere bună între predicțiile modelului și datele reale, cu excepția unor mici abateri.
3. Diferența între **MAE** și **RMSE** este mică, ceea ce indică faptul că erorile mari (outlier-ii) nu influențează semnificativ performanța generală a modelului.

4. Factorul $R^2$ de 0.816 indică faptul că modelul se potrivește bine datelor, explicând aproximativ 81.6% din variația variabilei *units sold* pe baza celorlalte variabile utilizate.

Performanțele "hardware" ale algoritmului sunt următoarele (voi comenta asupra lor după ce prezint pe scurt și concluziile în urma antrenării modelului cu algoritmul Random Forest):

- **Timp de antrenare:** 6.8 secunde (aproximativ)
- **Memorie utilizată:** 0 MB

### 2.2.2 Random Forest

În urma antrenării modelului de predicție folosind algoritmul *Random Forest*, am obținut următoarele predicții (comparate cu valorile reale):

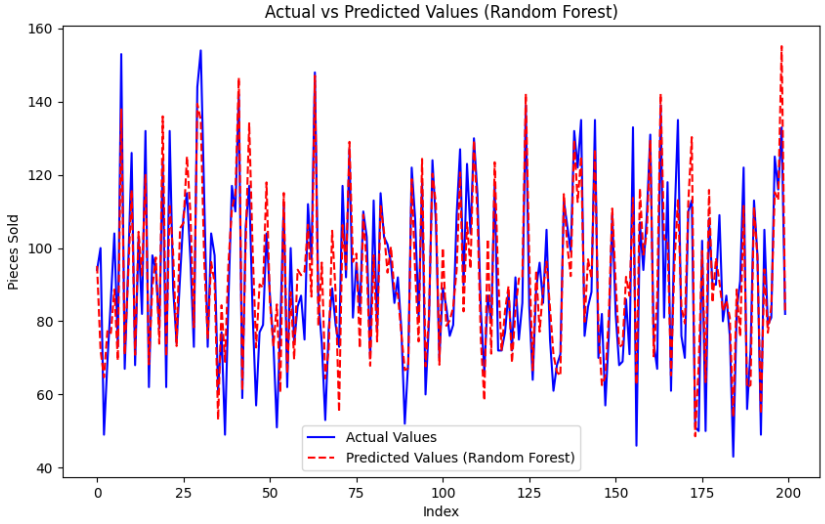


Figure 2: Random Forest Prediction Results

Asemenea modelului antrenat cu *Regresia Liniară*, predicțiile sunt în mare parte corecte, apropiate de valorile reale. Indicatorii de performanță au fost extrem de apropiați, așa că nu voi insista asupra lor. Timpul de antrenare și memoria utilizată, totuși, sunt complet diferite:

- **Timp de antrenare:** 2.3 secunde (aproximativ)
- **Memorie utilizată:** 7 MB (aproximativ)

### 2.2.3 Scurta Comparatie

Am observat că pentru un set relativ mic de date (1000 de rânduri), performatele modelului antrenat cu *Random Forest* au fost semnificativ mai bune decât cele ale modelului antrenat cu *Regresie Liniară*. Totuși, de curiozitate, am generat un set de date cu 50.000 de linii și am rulat aceiași algoritmi, așteptându-mă la rezultate similare, dar rezultatul a fost complet opus:

- **Timpi de antrenare:** 6s (Regr. Liniara) si 36s (Random Forest)
- **Memorie utilizată:** 6 MB (Regr. Liniara) si 339 MB (Random Forest)

Rezultatele au fost cel puțin surprinzătoare, timpul de antrenare pentru *Regresia Liniară* fiind aproximativ același, ba chiar mai mic (6.7 secunde), iar pentru *Random Forest* 36 secunde. De asemenea, cantitatea de memorie utilizată în cazul modelului *Random Forest* a fost enormă comparativ cu *Regresia Liniară*, unde memoria utilizată a rămas in jurul valorii de 0 (0.037 MB).
Pot concluziona astfel: pe seturi mici de date, algoritmul *Random Forest* a fost net superior *Regresiei Liniare*. Pentru seturi mari de date, totuși, *Regresia Liniară* și-a păstrat aceleași performanțe comparativ cu modelul antrenat pe setul mic de date.

# 3 Concluzii

Concluzionând, algoritmul de predicție *Regresie Liniară* este unul cu o acuratețe ridicată: factorul $R^2$ este unul ridicat, iar erorile, comparativ cu setul meu de date sunt mici, acceptabile.
Am observat de asemenea că orice fel de *noise* introdus în scriptul de generare a setului de date, în coloana de *units sold*, are un efect "distructiv" asupra modelului, predicțiile fiind complet eronate. Astfel, atât în cazul modelelor antrenate cu *Regresie Liniară*, cât și cu *Random Forest*, seturile de date trebuie trecute printr-un algoritm de **denoising**, pentru a asigura performanța ridicată.