Tema ML: Logistic Regression + Recomandări + Ranking

Ciurariu Raluca-Iuliana, Darie Mihnea Stefan

18 ianuarie 2026

1 Descrierea problemei și a dataset-ului

Dataset-ul conține bonuri de cumpărături (coșuri) cu produsele achiziționate și timestamp-ul bonului. Obiectivul general este de a construi modele care:

- prezic dacă un produs (ex. sos) apare în coș, pe baza contextului și a produselor din coș;
- recomandă sosuri prin Top-K recomandări;
- realizează ranking pentru upsell folosind scoruri bazate pe probabilitate și venit.

2 Preprocesare și feature engineering

În toate experimentele, am aplicat pași comuni de curățare și preprocesare.

2.1 Curățare

- Standardizare denumiri produse: `str.strip()` pe `retail_product_name`
- Conversie `data_bon` la `datetime`
- Verificări pentru valori lipsă și eventuale prețuri invalide (ex. ≤ 0)

2.2 Features temporale

Din `data_bon` am derivat:

- `hour` – ora din zi
- `day_of_week` – ziua săptămânii (1–7)
- `is_weekend` – indicator binar weekend

2.3 Reconstrucția coșului

Bonul este unitatea de analiză (`id_bon`). Feature-urile sunt construite astfel:

- Vector de produse: `pd.crosstab(id_bon, retail_product_name)` (count per produs)
- Agregări per bon:
  - `cart_size` = număr total poziții în bon
  - `distinct_products` = număr produse distincte
  - `total_value` = suma prețurilor
  - `avg_price` = prețul mediu

2.4 Evitarea leakage-ului (Anti-leakage)

În problemele în care prezic apariția unui sos, am exclus coloana sosului curent din features pentru a evita reguli triviale de tip:

$$y = 1 \Rightarrow X(\text{sos}) > 0$$

care ar produce performanțe nerealiste.

3 EDA (Exploratory Data Analysis)

Am realizat o analiză exploratorie pentru a înțelege distribuția datelor și dependențele dintre feature-uri și target.

3.1 Grafice salvate

Graficele au fost salvate în folderul `plots/`. Exemple:

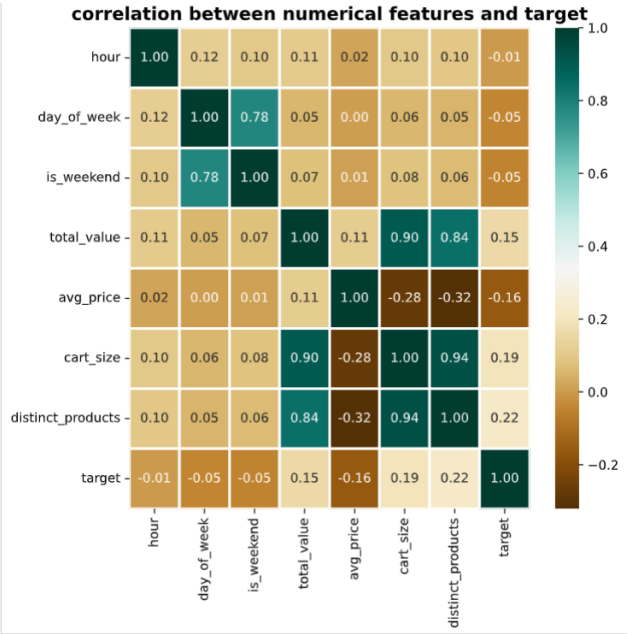

Figura 1: Enter Caption

Figura 2: Corelații între feature-urile numerice și target

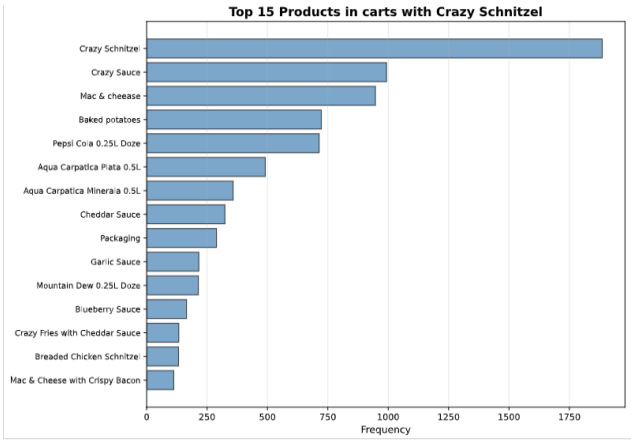

Figura 3: Enter Caption

Figura 4: Top produse în coșurile cu Crazy Schnitzel

Figura 5: Enter Caption

Figura 6: Distribuția target-ului (coș cu / fără Crazy Sauce)

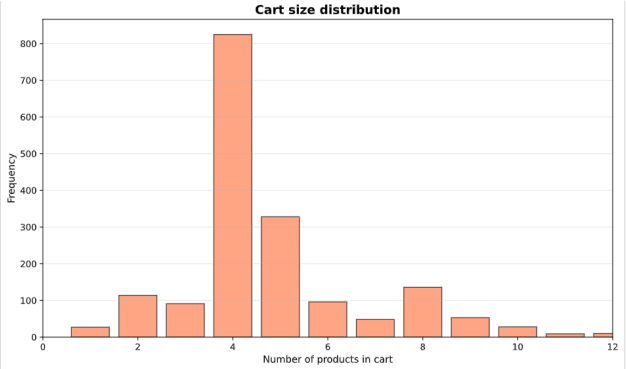

Figura 7: Enter Caption

Figura 8: Distribuția mărimii coșului

3.2 Observații EDA

- Dataset-ul este dezechilibrat: în multe cazuri, clasa majoritară domină.
- Heatmap-ul de corelație a fost folosit ca suport pentru selecția feature-urilor numerice.
- `cart_size` și `distinct_products` sunt predictori importanți, deoarece coșurile mai mari au probabilitate mai mare să includă sosuri.

4 Exercițiul 2.1: Logistic Regression (Crazy Sauce | Crazy Schnitzel)

4.1 Definirea problemei

Consider doar bonurile ce conțin `Crazy Schnitzel`.

$$y = \begin{cases} 1, & \text{dacă bonul conține și Crazy Sauce} \\ 0, & \text{altfel} \end{cases}$$

Feature-urile includ context temporal, agregări de coș și produse din coș (fără `Crazy Sauce`).

4.2 Implementare Logistic Regression (own code)

Am implementat manual două metode de antrenare:

- Gradient Descent (GD)
- Newton Method (cu Hessian și pseudo-inversă `np.linalg.pinv`)

Sigmoid:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Cost log-loss:

$$J(\theta) = -\frac{1}{m} \sum_{i=1}^{m} [y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

4.3 Metrici de evaluare

Am folosit:

- Accuracy
- Precision / Recall / F1
- ROC-AUC
- Matrice de confuzie
- Baseline: majority class

4.4 Analiza coeficienților (interpretabilitate)

Pentru fiecare model Logistic Regression am salvat coeficienții în CSV și am analizat:

- coeficienți pozitivi: cresc probabilitatea ca target-ul să fie 1;
- coeficienți negativi: reduc probabilitatea.

4.5 Problema Newton: coeficienți care explodează

Pe același set de date pe care GD funcționa stabil, Newton Method a avut uneori:

- accuracy foarte mare,
- dar coeficienți extrem de mari,

ceea ce indică instabilitate numerică sau separare aproape perfectă.

Cauze probabile:

- multicoliniaritate (coloane foarte corelate / duplicate);
- features rare (produse care apar în foarte puține bonuri);
- Hessian aproape singular (condiționare slabă).

4.6 Rezolvare: feature selection

Pentru stabilizare am introdus:

- feature selection:
  - selecție numerică ghidată de correlation map (heatmap),
  - eliminare coloane constante (std=0),
  - eliminare coloane duplicate.

4.7 Regularizare L2 (stabilizarea coeficienților)

Pentru a rezolva problema în care coeficienții obținuți cu Newton deveneau foarte mari (instabilitate numerică / separare aproape perfectă), am introdus regularizare L2 (ridge). Aceasta penalizează valorile mari ale coeficienților și îmbunătățește stabilitatea modelului.

În termeni matematici, funcția de cost devine:

$$J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\Big[y_i \log(\hat{y}_i) + (1-y_i)\log(1-\hat{y}_i)\Big] + \lambda \cdot \|\mathbf{W}\|_2^2$$

Regularizarea a fost combinată cu standardizarea features-urilor numerice.

4.8 Rezultate numerice (baseline + RAW/STD + FS+STD)

Mai jos sunt rezultatele obținute pe test (cu train unde este disponibil).

4.8.1 Baseline (majority class)

Tabela 1: Baseline: majority class

Model	Acc_{test}	Prec_{test}	Rec_{test}	F1_{test}	ROC-AUC_{test}
Baseline majority=1	0.5322	0.5322	1.0000	0.6947	–

4.8.2 Logistic Regression: RAW vs STD

Tabela 2: Logistic Regression - RAW vs STD (test metrics)

Model	Acc_{test}	Prec_{test}	Rec_{test}	F1_{test}	ROC-AUC_{test}
GD RAW	0.5518	0.5434	0.9895	0.7015	0.5361
Newton RAW	0.9440	0.9337	0.9632	0.9482	0.9678
GD STD	0.8880	0.8440	0.9684	0.9020	0.9676
Newton STD	0.9300	0.9146	0.9579	0.9357	0.9677

4.8.3 Logistic Regression: Feature Selection + Standardizare

Tabela 3: Logistic Regression - FS+STD (test metrics)

Model	Acc_{test}	Prec_{test}	Rec_{test}	F1_{test}	ROC-AUC_{test}
GD FS+STD	0.9216	0.8932	0.9684	0.9293	0.9766
Newton FS+STD	0.9300	0.9188	0.9526	0.9354	0.9646

4.9 Logistic Regression + L2: (Baseline / RAW / Standardizare)

Tabela 4: Performanță Logistic Regression (Baseline, RAW, Standardizat)

Model	**Acc**	**Prec**	**Rec**	**F1**	**ROC-AUC**
Baseline (majority=1)	0.5322	0.5322	1.0000	0.6947	0.6947
GD RAW	0.5518	0.5434	0.9895	0.7015	0.5361
Newton RAW	0.9440	0.9337	0.9632	0.9482	0.9678
GD STD	0.8880	0.8440	0.9684	0.9020	0.9676
Newton STD	0.9300	0.9146	0.9579	0.9357	0.9677

5 Exercițiul 2.2: Logistic Regression pentru fiecare sos + recomandare Top-K

5.1 Definire

Pentru fiecare sos s:

$$y_s = \mathbb{1}[s \in bon]$$

Am antrenat un model logistic separat pentru fiecare sos din listă.

5.2 Anti-leakage

Pentru sosul curent s, am folosit:

$$X^{(-s)} = X \setminus \{s\}$$

și pentru recomandare am creat context fără sosuri:

$$X_{\text{context}} : \text{toate coloanele sos} = 0$$

5.3 Recomandare Top-K

Pentru fiecare bon din test:

- calculez probabilități $P(s|coș)$ pentru fiecare sos;
- recomand Top-K sosuri cu probabilitatea cea mai mare, care nu sunt deja în coș.

5.4 Metrici Top-K

- Hit@K
- Precision@K

5.5 Comparație cu baseline popularitate

Baseline-ul recomandă Top-K sosuri după popularitatea globală în train.

5.6 Rezultate (exemplu)

```
MODELS:   K=1  Hit@K=0.130  Precision@K=0.130
MODELS:   K=3  Hit@K=0.638  Precision@K=0.223
MODELS:   K=5  Hit@K=0.888  Precision@K=0.200

BASELINE: K=1  Hit@K=0.384  Precision@K=0.384
BASELINE: K=3  Hit@K=0.749  Precision@K=0.273
BASELINE: K=5  Hit@K=0.949  Precision@K=0.216
```

6 Exercițiul 3: Ranking (Upsell) cu Naive Bayes

6.1 Scop

Construiesc un ranking de produse candidate pentru upsell pe baza scorului:

$$Score(p|coș) = P(p|coș) \cdot price(p)$$

6.2 Algoritm implementat: Bernoulli Naive Bayes

Am implementat Bernoulli Naive Bayes folosind input binar:

$$X_b = \mathbb{1}[X > 0]$$

cu smoothing Laplace ($\alpha = 1$).

6.3 Split temporal

Am folosit split temporal 80/20 (train vechi, test nou) pentru a simula un scenariu real.

6.4 Evaluare (coș parțial)

Din fiecare bon eligibil:

- elimin un produs real (ground truth);
- verific dacă apare în Top-K ranking.

6.5 Rezultate

```
naive bayes ranking | test_cases=829
hit@1/3/5: {1: 0.3148, 3: 0.6441, 5: 0.8528}

popularity baseline
hit@1/3/5: {1: 0.2992, 3: 0.6960, 5: 0.9276}
```

7 Concluzii

- Logistic Regression standardizat a fost mai stabil, în special pentru GD.
- Newton poate obține rezultate excelente, dar poate deveni instabil numeric (coeficienți foarte mari).
- L2 regularizare + feature selection au stabilizat Newton și au redus coeficienții exagerați.
- Recomandarea Top-K pentru sosuri funcționează, dar baseline popularitate rămâne puternic.
- Ranking cu Naive Bayes + scor $P \cdot price$ este simplu și interpretabil, competitiv pentru Hit@1.

8 Direcții de îmbunătățire

- Extinderea candidate set (băuturi, garnituri).
- Tuning de prag pentru LR (nu neapărat 0.5).
- Compararea cu alte metode pentru ranking: k-NN, ID3, AdaBoost.
- Evaluare mai strictă: eliminare mai multe produse din coș (nu doar 1).

9 Contribuții:

Ne-am împarțit task-urile în doi, am lucrat împreuna la tot ce am făcut, nu a facut vreo unul un exercițiu singur, astfel încât amândoi am invațat tot din temă, nu doar o parte.