PROIECT APLICAȚIE STREAMLIT

Bank Churn

Proiect realizat de: Andrei Hornaru

Introducere

1. Scopul proiectului

Scopul acestui proiect este de a analiza dacă poate fi prezis comportamentul clienților care sunt predispuși să părăsească o bancă și să se mute la alta. Predicția este realizată pe baza unui set larg de date care include informații demografice, date financiare, nivel de educație și alți factori relevanți. Prin această analiză se urmărește identificarea tiparelor care pot indica probabilitatea ca un client să renunțe la serviciile unei instituții financiare.

2. Sursa datelor

Datele utilizate în acest proiect provin dintr-un set de date public disponibil pe platforma Kaggle. Acest set de date conține răspunsuri colectate de la un număr mare de respondenți și este folosit frecvent pentru analiza comportamentului clienților. Setul de date poate fi regăsit și accesat direct din repository-ul meu de pe GitHub, unde este inclus ca parte a proiectului.

3. Tehnologii și implementare

Aplicația a fost construită folosind limbajul de programare Python și framework-ul Streamlit, care permite dezvoltarea rapidă a aplicațiilor interactive pentru analiza datelor. În cadrul aplicației sunt aplicate procese de curățare a datelor, transformări, normalizare și standardizare, precum și vizualizări grafice. De asemenea, sunt utilizate analize statistice și algoritmi de machine learning pentru a studia relațiile dintre variabile și pentru a realiza predicții.

4. Analiza și interpretarea rezultatelor

Rezultatele obținute în urma analizelor statistice și a algoritmilor de machine learning sunt interpretate din perspectiva data science. Accentul este pus pe înțelegerea comportamentului clienților, pe relevanța variabilelor analizate și pe modul în care modelele pot fi utilizate pentru a susține decizii informate. Interpretarea nu urmărește doar acuratețea modelelor, ci și valoarea practică a insight-urilor obținute.

Meniul de navigare

Meniul de navigare al aplicației a fost construit utilizând componenta `sidebar` din Streamlit, cu scopul de a organiza clar și intuitiv toate etapele procesului de analiză a datelor.

Structura principală este realizată printr-un selector de tip *selectbox*, care permite utilizatorului să aleagă categoria dorită, începând de la o pagină introductivă și continuând cu etape esențiale precum vizualizarea datelor, curățarea acestora, îmbunătățirea acurateței, transformări statistice și aplicarea algoritmilor de machine learning. Pentru anumite categorii, cum ar fi curățarea datelor, analiza statistică sau algoritmii ML, au fost implementate submeniuri de tip *radio button*, care oferă acces la pagini dedicate fiecărei operații sau metode.

Organizare meniu navigare

- Intro (page)
- Vizualizare Data(page)
- Categorie: Curarea datelor
    1. Valori lipsa(page)
    2. Duplicate(page)
- Categorie: Imbunatarirea acuratetei datelor
    3. Anomalii(page)
- Categorie: Standardizarea si normalizarea valorilor
    1. Transformari(page)
- Categorie: aplicarea unor statistici descriptive:
    2. Skewness(page)
    3. Kurtos kolac(page)
    4. Matrice corelatie si covarianta(page)
    5. Reprezentari Grafice (pychart, histograme, boxplot)
    6. Agregari (min,max) (page)

Modificările setului de date efectuate:

1. **10% dintre celule șterse aleatoriu**:
    - Am introdus valori NaN (null) în 10% dintre celulele din dataset.
2. **15 clienți cu același CustomerId**:
    - 15 rânduri au acum un CustomerId duplicat, identic cu primul client din fișier.
3. **5% anomalii**:
    - 5% dintre rânduri au fost modificate astfel încât valorile Age și Tenure să fie negative.
4. **5% outliers**:
    - Pentru Age: valorile au fost modificate pentru a fi între 9-13 ani sau peste 90 de ani.
    - Pentru Tenure: valorile au fost setate mai mari decât 1.5 ori media sau mai mici decât media curentă.
5. **10% persoane cu salarii mari**:
    - 10% dintre rânduri au EstimatedSalary peste 200,000.
6. **10% persoane cu salarii foarte mici**:
    - 10% dintre rânduri au EstimatedSalary mult mai mic decât media (sub jumătate din media curentă).

Aplicarea de statistici descriptive

1. Skewness

Rezultatele Skewness (asimetria distribuției) oferă informații despre simetria sau lipsa de simetrie a distribuției valorilor pentru EstimatedSalary în funcție de fiecare categorie din Geography. Interpretarea este următoarea:

Ce reprezintă valorile Skewness:

Skewness = 0 (sau aproape de 0):

- Distribuția este simetrică (similară cu o distribuție normală).
- Valorile sunt uniform distribuite pe ambele părți ale mediei.

Skewness > 0 (pozitivă):

- Distribuția este asimetrică spre **dreapta** (skewed right).
- Majoritatea valorilor sunt mai mici decât media, iar coada lungă a distribuției este pe partea dreaptă (valori mari).

Skewness < 0 (negativă):

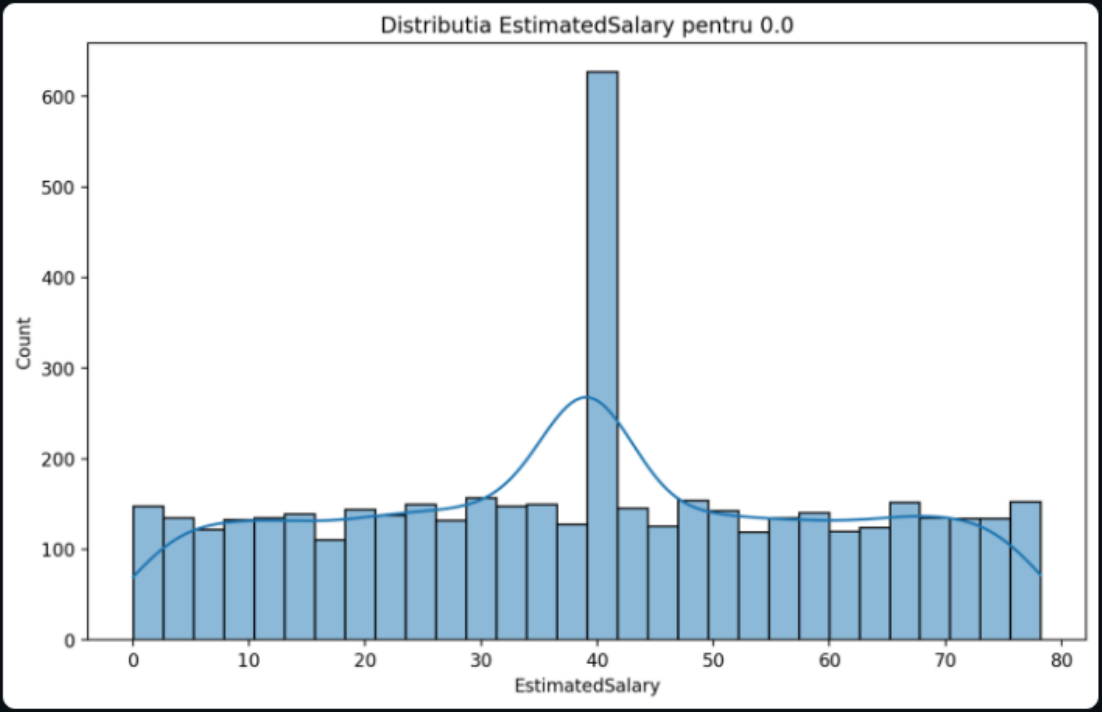
- Distribuția este asimetrică spre **stânga** (skewed left).
- Majoritatea valorilor sunt mai mari decât media, iar coada lungă a distribuției este pe partea stângă (valori mici).

Coeficientul de Skewness pentru EstimatedSalary in functie de Geography

Geography	EstimatedSalary
0	0.007
50	-0.0403
100	-0.0259

Distributia EstimatedSalary pentru fiecare valoare Geography

Distributia pentru Geography: 0.0



Distributia pentru Geography: 100.0

2. Kurtosis

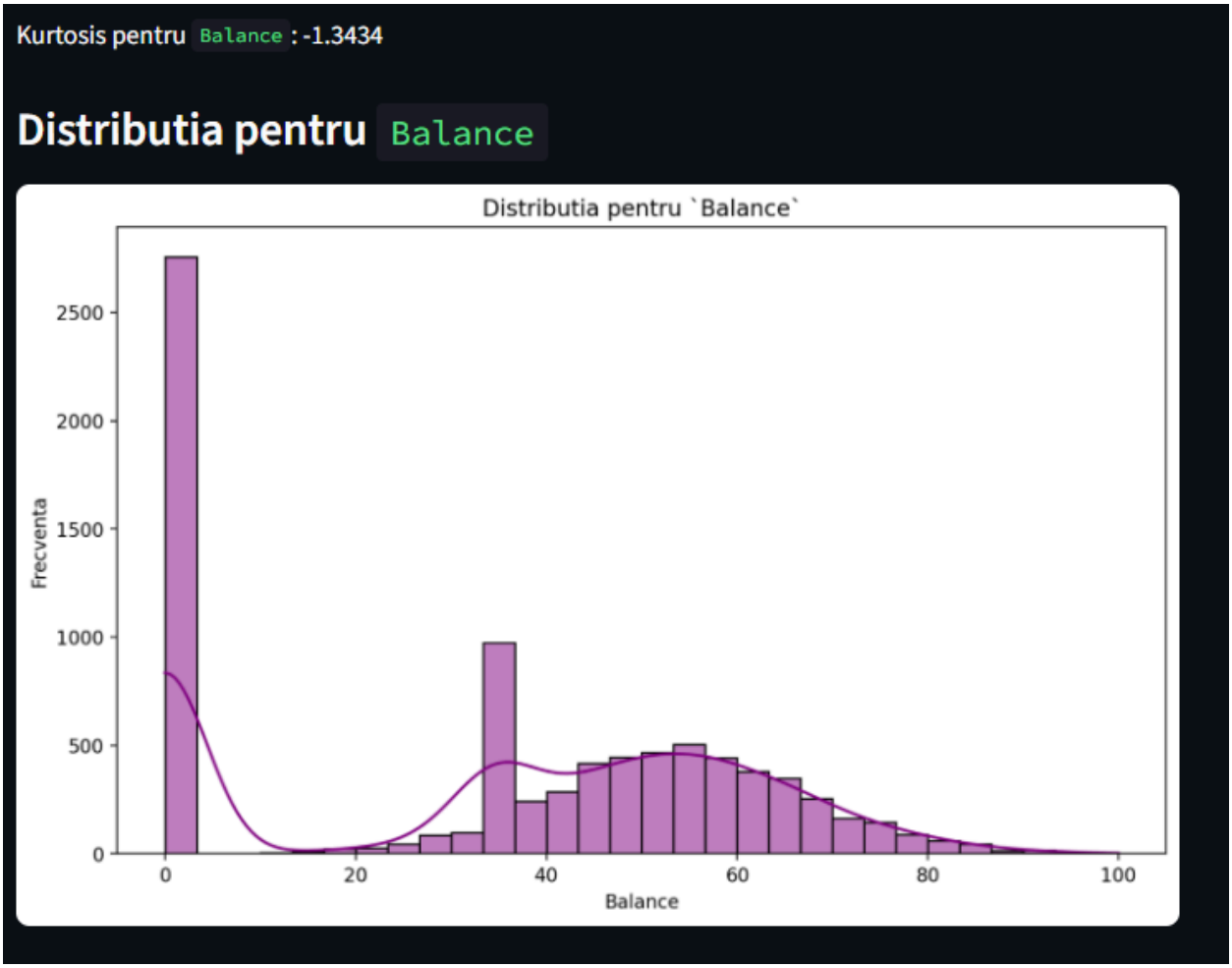
Interpretarea valorilor Kurtosis:

1. Kurtosis pentru Tenure: 1.6730

- **Kurtosis < 3**: Distribuția este **platikurtică**, ceea ce înseamnă că distribuția are cozi mai scurte și mai plate comparativ cu o distribuție normală.

- **Implicații**:
    - Valorile extreme (foarte mari sau foarte mici) sunt mai puțin frecvente decât într-o distribuție normală.
    - Majoritatea valorilor pentru Tenure sunt concentrate în jurul mediei.

2. Kurtosis pentru Balance: -1.4893



- **Kurtosis negativă**: Deși acest lucru este mai rar întâlnit, sugerează că distribuția este **extrem de platikurtică**.
- **Implicații**:
    - Distribuția este mult mai plată, cu valori răspândite uniform și aproape fără cozi semnificative.
    - Nu există multe valori extreme, iar distribuția este mai uniformă.

Ce reprezintă metoda asupra variabilelor?

Kurtosis măsoară "forma cozii" distribuției unei variabile și ne oferă informații despre:

1. **Prezența valorilor extreme**:
    - Distribuțiile leptokurtice (kurtosis > 3) au cozi mai lungi și sunt mai sensibile la valori extreme.
    - Distribuțiile platikurtice (kurtosis < 3) au cozi mai scurte și mai puține valori extreme.
2. **Concentrarea valorilor**:
    - Kurtosis mică (< 3) indică faptul că valorile sunt mai uniform distribuite.
    - Kurtosis mare (> 3) indică faptul că valorile sunt mai concentrate în jurul mediei.

Ce înțelegem pentru aceste variabile?

1. **Tenure (1.6730)**:
    - Variabila are o distribuție uniformă, cu majoritatea valorilor concentrându-se aproape de medie.
    - Potrivit pentru modelare directă, fără transformări.
2. **Balance (-1.4893)**:
    - Distribuția este foarte uniformă și plată, ceea ce ar putea indica:
    - Date bine răspândite (nu există concentrații mari).
    - Necesitatea unei posibile transformări pentru modele care presupun distribuții normale.

Concluzie:

- **Kurtosis este utilă** pentru a înțelege cum sunt distribuite valorile unei variabile și cât de des apar extremele.
- În cazul datelor:

- Tenure este relativ concentrat în jurul mediei, fără a avea valori extreme.
- Balance este foarte uniform distribuit, ceea ce poate necesita analize suplimentare în funcție de scopul proiectului.

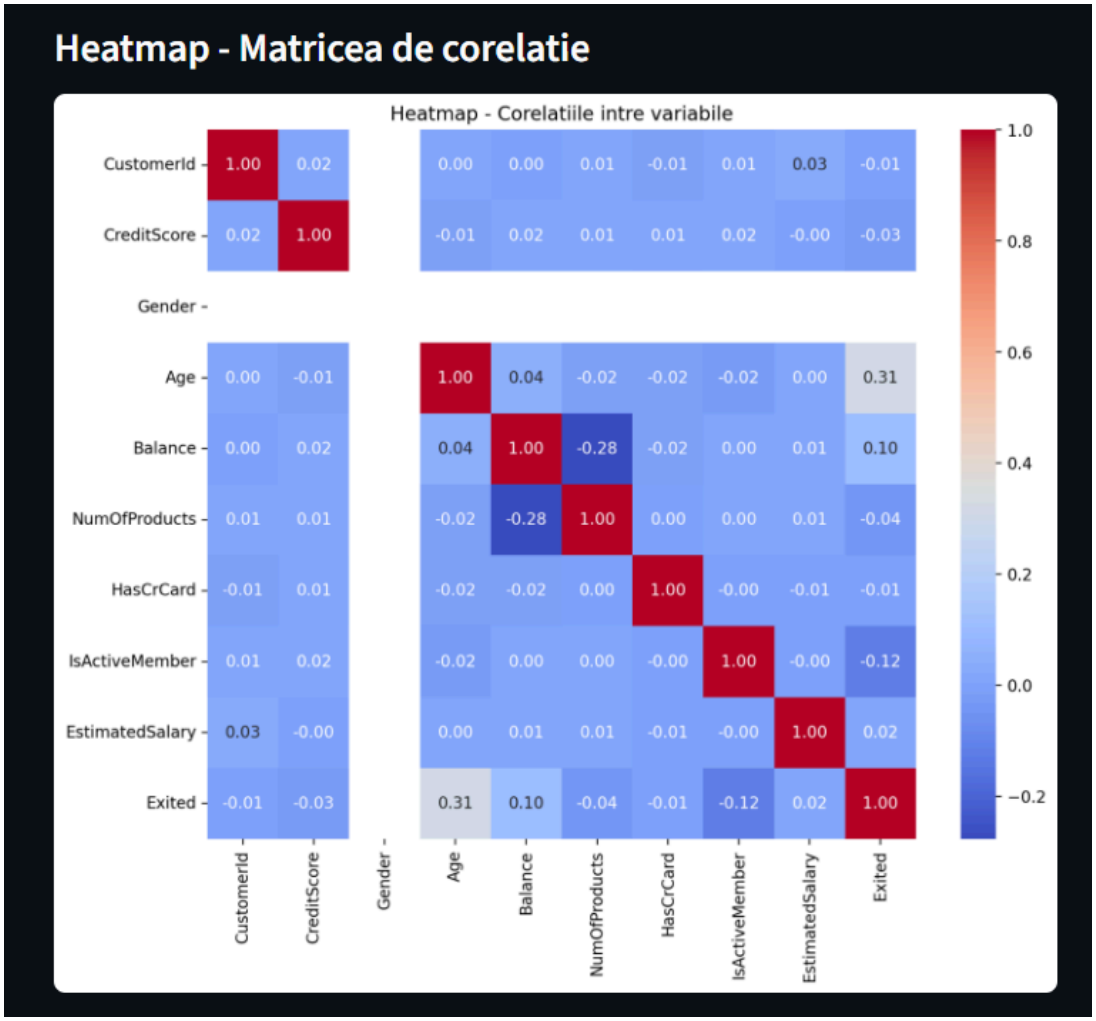
Matricea de covarianta

Vârsta (Age) este un factor influent asupra deciziei clienților de a părăsi banca. Acest lucru sugerează că ar trebui să fie un punct de atenție pentru strategii de retenție.

Balanța (Balance) joacă un rol moderat în influențarea comportamentului clienților, dar impactul său nu este la fel de pronunțat ca al vârstei.

Membrii activi (IsActiveMember) tind să rămână mai loiali, deși corelația nu este foarte puternică.

Numărul de produse (NumOfProducts) are relații interesante, dar inverse, cu alte variabile (Balance și Age).



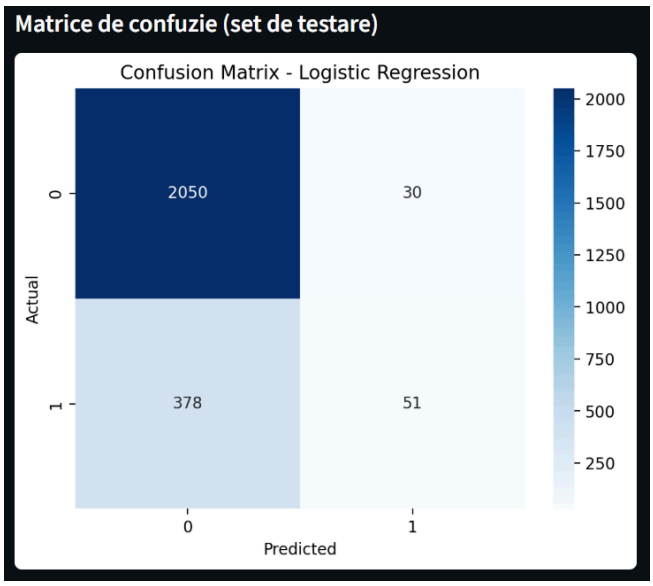
Algoritmi de machine learning

Regresie logistică

Modelul de regresie logistică a fost antrenat pe un set de date împărțit în date de antrenare și date de testare, cu un procent de 50% alocat setului de test. În urma acestei împărțiri, setul de antrenare a conținut 5.852 observații, iar setul de testare 2.509 observații. Variabila țintă, *Exited*, a fost convertită în format numeric pentru a putea fi utilizată de algoritm. Modelul a fost antrenat cu succes, iar performanța sa a fost evaluată atât pe datele de antrenare, cât și pe cele de testare.

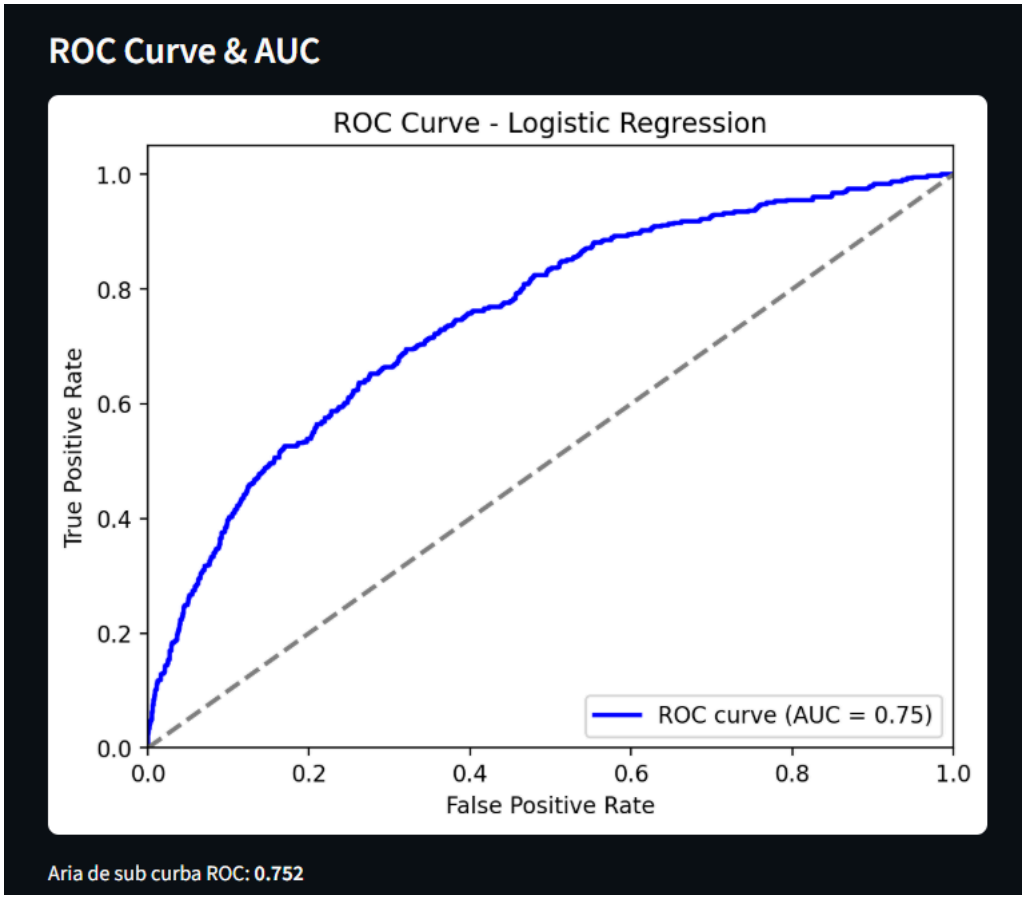
Acuratețea obținută este de **0.840 pe setul de antrenare** și **0.837 pe setul de testare**, ceea ce indică un comportament stabil al modelului și o bună capacitate de generalizare. Diferența foarte mică dintre cele două valori sugerează că modelul nu suferă de overfitting și reușește să învețe relații relevante din date fără a se adapta excesiv la setul de antrenare.

Analiza matricei de confuzie și a raportului de clasificare arată însă o **diferență semnificativă între performanța pe cele două clase**. Modelul identifică foarte bine clienții care rămân (*Stayed*), obținând un recall de 0.99 și un f1-score de 0.91 pentru această clasă. În schimb, performanța pentru clasa *Exited* este mult mai scăzută, cu un recall de doar 0.12 și un f1-score de 0.20. Acest lucru indică faptul că modelul are dificultăți în a detecta corect clienții care părăsesc banca, clasificând o mare parte dintre aceștia ca fiind clienți care rămân.



Această diferență de performanță poate fi explicată prin **dezechilibrul claselor** din setul de date, unde numărul clienților care rămân este semnificativ mai mare decât numărul celor care pleacă. Deși acuratețea globală este ridicată, aceasta nu reflectă în totalitate capacitatea modelului de a identifica clienții cu risc de părăsire, motiv pentru care este importantă analiza metricilor precum precision, recall și f1-score pentru fiecare clasă în parte.

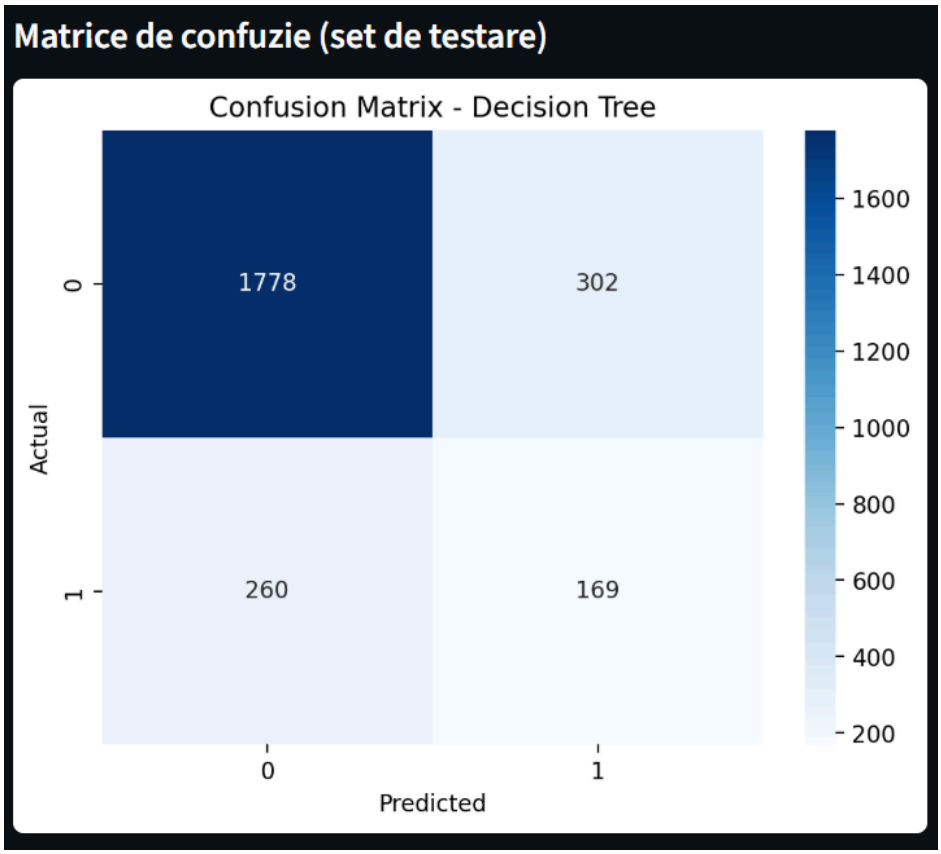
Curba ROC și valoarea **AUC de 0.752** indică o capacitate moderată a modelului de a separa cele două clase. Deși regresia logistică oferă un punct de plecare solid și ușor de interpretat, rezultatele sugerează că, pentru o predicție mai bună a clienților care vor părăsi banca, ar putea fi necesare modele mai complexe sau tehnici suplimentare, precum echilibrarea claselor sau utilizarea unor algoritmi de tip ensemble.



Arbori de decizie

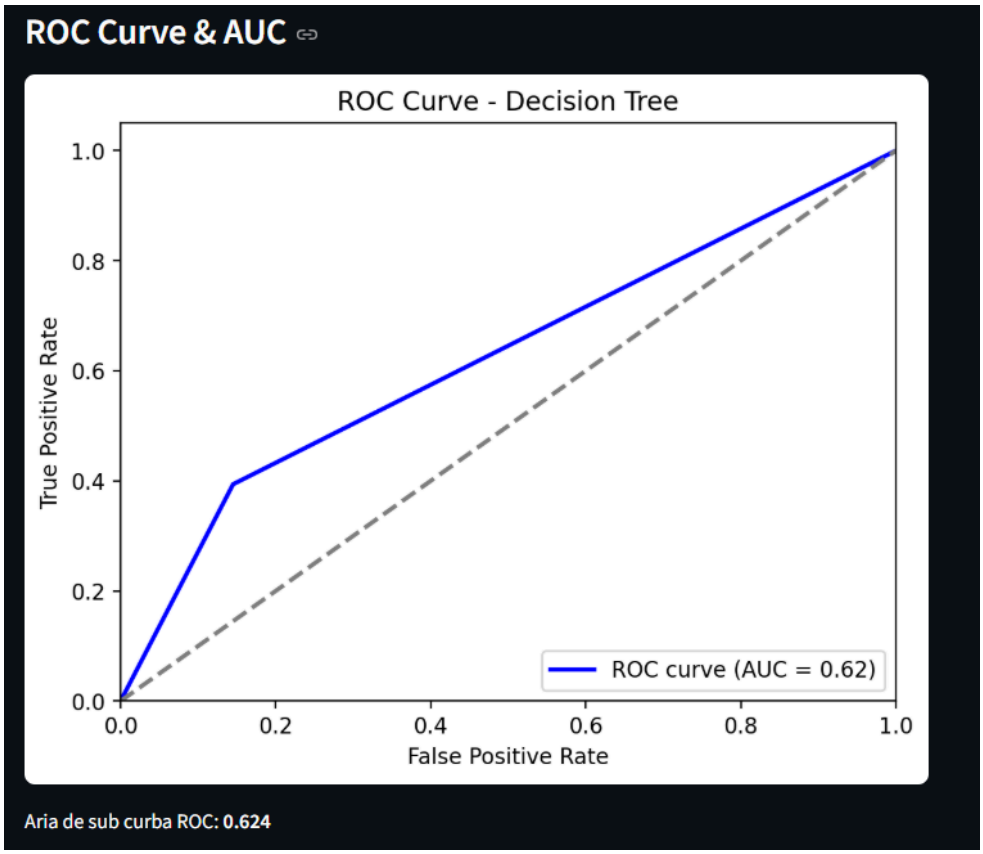
Modelul de Arbori de Decizie a fost antrenat folosind același set de date și aceeași împărțire între datele de antrenare și cele de testare, cu 50% din date alocate setului de test. Setul de antrenare a conținut 5.852 observații, iar setul de testare 2.509 observații, iar variabila țintă *Exited* a fost convertită în format numeric pentru a putea fi procesată de algoritm. Modelul a fost antrenat cu succes, oferind rezultate care permit o analiză detaliată a performanței sale.

Acuratețea obținută pe setul de antrenare este de 1.000, ceea ce indică faptul că modelul a învățat perfect datele de antrenare. Totuși, acuratețea pe setul de testare scade la 0.776, ceea ce sugerează o problemă de overfitting, specifică arborilor de decizie atunci când nu sunt controlați prin limitarea adâncimii sau prin alte tehnici de regularizare. Modelul se adaptează foarte bine datelor de antrenare, dar generalizează mai slab pe date noi.



Raportul de clasificare arată că modelul obține performanțe bune în identificarea clienților care rămân (*Stayed*), cu un f1-score de 0.86. În cazul clienților care părăsesc banca (*Exited*), performanța este semnificativ mai scăzută, cu un f1-score de 0.38 și un recall de 0.39. Aceste rezultate indică faptul că, deși modelul reușește să identifice o parte dintre clienții care pleacă, o proporție considerabilă este clasificată incorect.

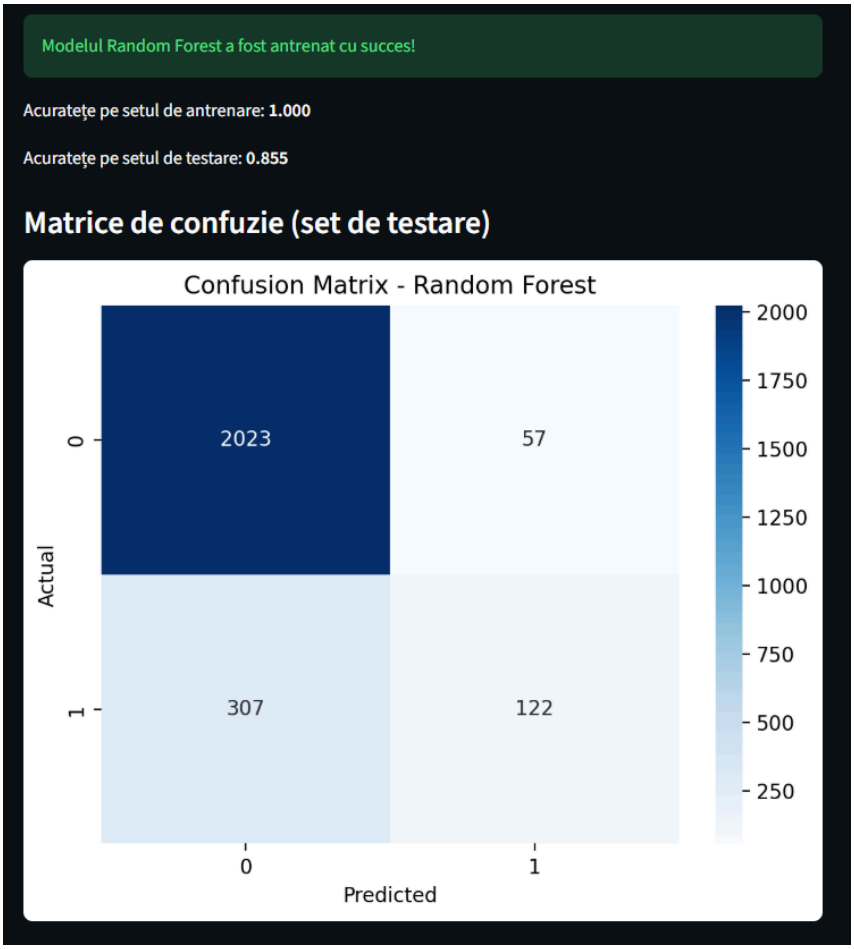
Analiza curbei ROC și valoarea AUC de 0.624 confirmă capacitatea limitată a modelului de a separa eficient cele două clase. Comparativ cu regresia logistică, arborele de decizie oferă o interpretabilitate ridicată, însă performanța generală și stabilitatea sunt mai reduse. Acest lucru sugerează că arborii de decizie simpli pot beneficia de optimizare suplimentară sau de integrarea într-un model de tip ensemble, precum Random Forest, pentru a obține rezultate mai robuste.



Random forest

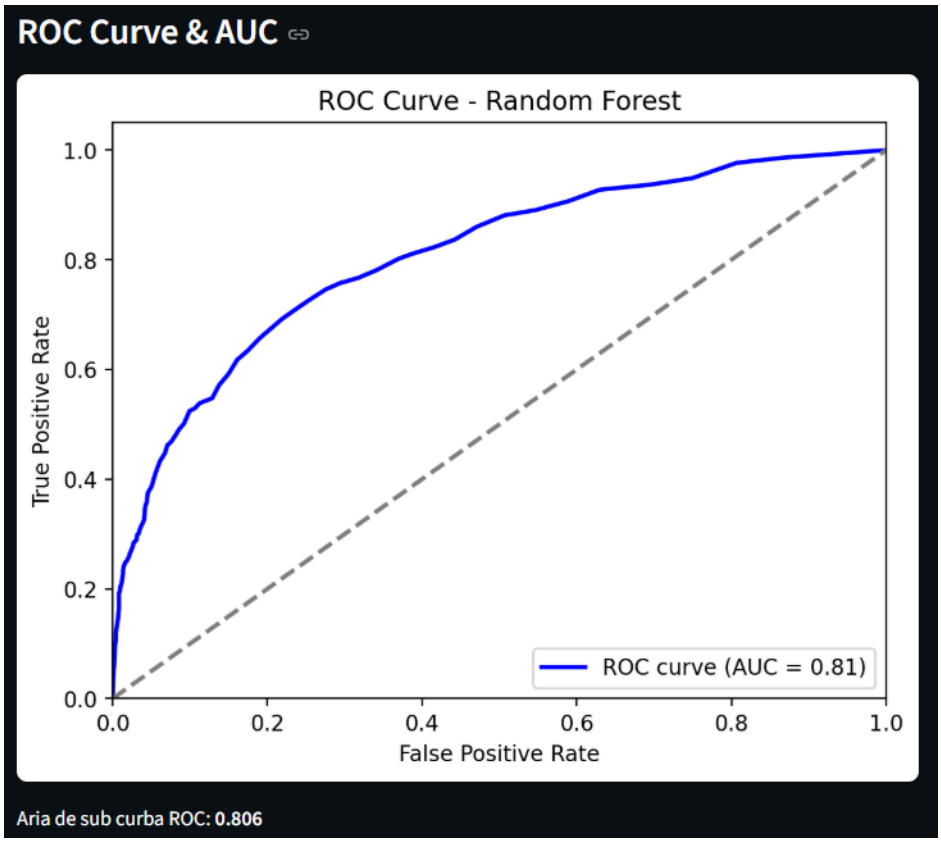
Modelul Random Forest a fost antrenat folosind același set de date și aceeași împărțire, iar acesta a inclus 5.852 observații, iar setul de testare 2.509 observații. Variabila țintă *Exited* a fost convertită în format numeric, iar câmpurile relevante au fost selectate automat pentru a construi un model robust bazat pe mai mulți arbori de decizie.

Acuratețea pe setul de antrenare este de 1.000, ceea ce indică faptul că modelul se potrivește perfect pe datele de antrenare. Cu toate acestea, spre deosebire de arborele de decizie simplu, Random Forest reușește să generalizeze mai bine, obținând o acuratețe de 0.855 pe setul de testare, cea mai ridicată dintre modelele analizate. Acest lucru confirmă avantajul abordării de tip ensemble, care reduce overfitting-ul prin combinarea mai multor modele.



Raportul de clasificare evidențiază o performanță foarte bună în identificarea clienților care rămân (*Stayed*), cu un recall de 0.97 și un f1-score de 0.92. Pentru clasa *Exited*, modelul obține rezultate mai bune decât regresia logistică și arborele de decizie, cu un f1-score de 0.40 și un recall de 0.28. Deși predicția clienților care părăsesc banca rămâne o provocare, Random Forest reușește să identifice un număr mai mare dintre aceștia comparativ cu celelalte modele.

Analiza curbei ROC și valoarea AUC de 0.806 indică o capacitate bună de separare a celor două clase, superioară regresiei logistice și arborilor de decizie. Aceste rezultate sugerează că Random Forest este cel mai performant model din această analiză, oferind un echilibru bun între acuratețe, stabilitate și capacitatea de a captura relații complexe din date, fiind o opțiune solidă pentru aplicații de predicție a comportamentului clienților.



Concluzii

În concluzie, analiza comparativă a celor trei modele de clasificare aplicat pentru predicția părăsirii băncii (Exited) de către clienți evidențiază performanțe și limitări distincte:

1. Performanță și generalizare:

Random Forest a obținut cea mai bună acuratețe globală pe setul de testare (0.855) și cea mai înaltă valoare AUC (0.806), demonstrând o capacitate robustă de generalizare și de separare a claselor, în ciuda unui overfitting pe datele de antrenare (acuratețe 1.000).

Regresia Logistică s-a dovedit a fi cel mai stabil model, cu o diferență minimă între acuratețea pe antrenare (0.840) și testare (0.837), și o capacitate de discriminare moderată (AUC=0.752), fără semne de overfitting.

Arborele de Decizie simplu a prezentat cel mai clar overfitting (acuratețe 1.000 la antrenare vs. 0.776 la test) și cea mai scăzută capacitate de separare (AUC=0.624), limitându-i utilitatea în forma sa neoptimizată.

2. Abordarea problemei dezechilibrului de clasă:

Toate modelele au reflectat dificultatea intrinsecă de a prezice clasa minoritară (clienții care părăsesc). Întâlnind aceeași problemă a setului dezechilibrat de date, ele au obținut performanțe excelente în clasificarea clienților care rămân, dar slabe în identificarea celor care pleacă.

Random Forest a înregistrat cele mai bune rezultate pentru clasa Exited (F1-Score: 0.40, Recall: 0.28), în ciuda faptului că acestea sunt încă modeste.

Arborele de Decizie s-a situat pe locul doi (F1-Score: 0.38, Recall: 0.39).

Regresia Logistică a obținut cele mai slabe performanțe pentru această clasă (F1-Score: 0.20, Recall: 0.12).

3. Recomandări și perspective:

Modelul recomandat pentru implementare în stadiul actual este Random Forest, deoarece oferă cel mai bun echilibru între acuratețea globală, capacitatea de a identifica parțial clienții cu risc de părăsire și stabilitate.

Pentru a îmbunătăți substanțial detectarea clienților care părăsesc (clasa Exited), aplicarea tehnicilor de echilibrare a datelor (e.g., SMOTE, sub-sampling, supra-eșantionare) este esențială pentru oricare dintre modele.

Regresia Logistică rămâne o opțiune valabilă datorită simplității și interpretabilității sale, iar rezultatele sale ar putea fi îmbunătățite semnificativ prin echilibrarea prealabilă a claselor.

Arborii de Decizie, în forma lor simplă, necesită optimizare (prunarea, setarea adâncimii) sau includerea în structuri ensemble (ca Random Forest) pentru a deveni competitivi.

În final, studiul confirmă că obținerea unei acurateți globale ridicate nu este sinonimă cu un model eficient pentru probleme cu clase dezechilibrate. Performanța reală este dată de capacitatea de a prezice clasa de interes minoritară. Implementarea unui model ensemble precum Random Forest, combinată cu o strategie proactivă de echilibrare a datelor, reprezintă direcția optimă pentru dezvoltarea unei soluții predictive viabile în gestionarea riscului de părăsire a clienților.

Github Repo

Pentru a rula aplicația local, este necesară clonarea repository-ului GitHub și instalarea dependențelor. În primul rând, repository-ul se clonează folosind comanda:

```
git clone https://github.com/andreihornaru/Bank-Churn-Streamlit
```

După clonare, se navighează în directorul proiectului:

```
cd Bank-Churn-Streamlit
```

Se recomandă utilizarea unui mediu virtual Python (opțional, dar recomandat), urmată de instalarea librăriilor necesare:

```
pip install -r requirements.txt
```

Aplicația este dezvoltată folosind Streamlit și poate fi pornită cu următoarea comandă:

```
python -m streamlit run app.py
```

După rulare, aplicația va fi disponibilă în browser la adresa http://localhost:8501, unde utilizatorul poate interacționa cu interfața și funcționalitățile implementate.