Academia de Studii Economice, București
Facultatea de Cibernetică, Statistică și Informatică Economică
Specializarea: Informatică Economică

Proiect de practică

Tema proiectului :Data Science/Machine Learning _ Crearea unui model prectiv prin care se poate analiza comportamentul clientilor unei banci

Cadrul didactic coordonator: **Doinea-Zurini Madalina**

Student: **Mocanu Valentina-Adriana**

București

2023

Cuprins

Listă de figuri

Introducere

Stagiul meu de practică a fost susținut în cadrul proiectului Tap That Job 2021, ediția a IX-a, organizat de Sindicatul Studenților din Cibernetică în colaborare cu Banca Comercială Română (BCR), una dintre cele mai importante bănci din România, oferind servicii financiare diversificate.

Proiectul Tap That Job este dedicat studenților, oferindu-le informații relevante despre industria IT și despre oportunitățile de pe piața muncii, facilitându-le integrarea într-un mediu apropiat procesului de angajare la companii partenere. În cadrul acestui proiect, am parcurs diverse etape, inclusiv trimiterea aplicației, alcătuind un CV și un profil LinkedIn, susținerea unui test tehnic cu accent pe abordări de Data Science, interviul cu reprezentanții departamentului HR și, în cele din urmă, participarea la cursurile oferite de BCR, aceasta fiind alegerea mea principală.

Motivația care a stat la baza deciziei mele de a participa la proiectul Tap That Job a fost dorința de a lua parte la cursurile unei bănci de prestigiu într-un mediu familiar, alături de colegi, pentru o experiență inedită. Am avut oportunitatea de a face parte dintr-o echipă compusă din profesioniști pasionați de domeniul IT, cu aceleași interese în dezvoltarea în zona Tech. Pe lângă dorința de a participa la acest proiect organizat de Sindicatul Studenților din Cibernetică, am fost atrasă de faptul că BCR va fi partenerul care va susține cursuri de Data Science. Fiind un domeniu pe care nu îl studiasem încă în cadrul facultății, am fost entuziasmată de ideea de a învăța lucruri noi întrun mediu stimulant, dar și prietenos, alături de o echipă unită.

Având în vedere că în timpul facultății nu am avut ocazia să studiez limbajul Python și conceptele de machine learning, am fost entuziasmată de oportunitatea de a dobândi cunoștințe noi într-un mediu structurat.

Prin participarea la acest stagiu de practică și dobândirea de cunoștințe în Data Science, am avut posibilitatea de a-mi completa profilul și de a-mi diversifica abilitățile într-un mod relevant pentru industria IT. Sunt entuziasmată de provocările și oportunitățile care mă așteaptă în acest domeniu dinamic și sunt încrezător că experiența acumulată în cadrul acestui stagiu de practică va avea un impact pozitiv asupra viitorului meu profesional.

1. Prezentare generală a companiei BCR

BCR (Banca Comercială Română) este una dintre cele mai mari și importante bănci din România, cu o prezență puternică și o istorie îndelungată în sectorul bancar. Fondată în anul 1990, BCR a evoluat și s-a dezvoltat într-o instituție financiară de referință, oferind o gamă completă de servicii și produse bancare pentru persoane fizice, companii și instituții.

BCR este parte a grupului Erste Bank, unul dintre cei mai mari jucători bancari din Europa Centrală și de Est, cu o rețea extinsă în țările din regiune. Acest lucru oferă BCR acces la expertiză și resurse internaționale, beneficiind de avantajele unei organizații financiare solide și inovatoare.

Cu peste 10.000 de angajați și o rețea vastă de sucursale și agenții în întreaga țară, BCR are capacitatea de a răspunde nevoilor financiare ale milioanelor de clienți și de a contribui la dezvoltarea economică a României. BCR se străduiește să fie o bancă de încredere, concentrându-se pe oferirea de servicii bancare de calitate, inovație și responsabilitate socială.

BCR oferă o gamă largă de produse și servicii bancare, adaptate nevoilor diverse ale clienților săi. Printre acestea se numără credite și finanțări pentru persoane fizice și companii, depozite bancare, carduri bancare, servicii de plată și transferuri, asigurări, leasing, servicii de tranzacționare și investiții, precum și soluții de management al trezoreriei.

Un aspect cheie al BCR este angajamentul său față de inovație și tehnologie. Compania investește constant în dezvoltarea soluțiilor digitale pentru a oferi clienților săi o experiență bancară simplă, comodă și sigură. Prin intermediul platformelor online și a aplicațiilor mobile, clienții BCR pot accesa și gestiona conturile lor, efectua tranzacții, plăți și transferuri de bani, și beneficia de servicii personalizate, adaptate nevoilor lor specifice.

Un alt aspect important al BCR este angajamentul său față de calitatea serviciilor și securitatea datelor clienților. BCR pune accentul pe confidențialitatea și protecția informațiilor personale și financiare ale clienților săi, respectând cele mai înalte standarde de securitate cibernetică și conformându-se reglementărilor și normelor internaționale.

Pe lângă activitatea bancară, BCR este implicată activ în comunitate și în susținerea inițiativelor sociale. Compania desfășoară programe de responsabilitate socială și sprijină proiecte în domeniul educației, culturii, sănătății și protecției mediului. BCR acționează ca un partener de încredere pentru companii și instituții, oferind soluții financiare integrate și consultanță specializată pentru a sprijini dezvoltarea și succesul acestora.

BCR este recunoscută pe piața bancară pentru excelența sa, primind multiple premii și distincții. Compania se străduiește să fie o bancă de încredere și stabilă, orientată către satisfacerea nevoilor clienților, în același timp contribuind la progresul economic al țării.

2. Structura organizatorică a companiei BCR

Structura organizatorică a Băncii Comerciale Române (BCR) este compusă din mai multe departamente și divizii, care colaborează pentru a asigura funcționarea eficientă a băncii și pentru a oferi servicii bancare de calitate clienților săi. Iată o descriere generală a principalelor structuri organizaționale ale BCR:

1. Consiliul de Administrație: Este organismul de conducere suprem al băncii și este responsabil de stabilirea strategiei generale și a politicilor băncii. Consiliul de Administrație este format din membri selectați din cadrul grupului Erste Bank și din experți externi.

2. Directoratul Executiv: Acesta este responsabil de implementarea strategiei și politicii băncii, coordonarea activităților și luarea deciziilor operaționale. Directoratul Executiv este condus de Directorul General al băncii și include directori responsabili de diverse domenii cheie, cum ar fi operațiuni bancare, riscuri, finanțe, marketing, vânzări și resurse umane.

3. Divizia de Retail Banking: Această divizie se concentrează pe oferirea de servicii bancare și produse financiare pentru persoane fizice. Acesta include servicii precum gestionarea conturilor de economii și curente, împrumuturi și credite ipotecare, carduri bancare, servicii de plată, depozite și investiții personale.

4. Divizia de Corporate Banking: Această divizie se concentrează pe serviciile financiare și soluțiile personalizate pentru companii și instituții. Acesta include servicii de finanțare a afacerilor, servicii de tranzacționare, servicii de trezorerie, soluții de gestionare a riscului, finanțare comercială și consultanță specializată pentru clienții corporativi.

5. Divizia de Private Banking: Această divizie se adresează clienților de înaltă avere, oferind servicii financiare personalizate și soluții de investiții. Divizia de Private Banking se concentrează pe gestionarea patrimoniului, consultanță financiară, planificare financiară și soluții de investiții adaptate nevoilor clienților săi.

6. Departamentul de Tehnologie și Inovație: Acest departament este responsabil de gestionarea infrastructurii IT a băncii, dezvoltarea și implementarea soluțiilor tehnologice, inovație digitală și asigurarea securității informațiilor. Aceasta implică dezvoltarea de aplicații și platforme bancare digitale, securitatea cibernetică și îmbunătățirea experienței clienților în mediul online.

Structura organizatorică a administrației centrale se va putea vedea în cadrul următoarei organigrame:

Structura organizatorică a administrației centrale a BCR (valabilă la 30 septembrie 2010)

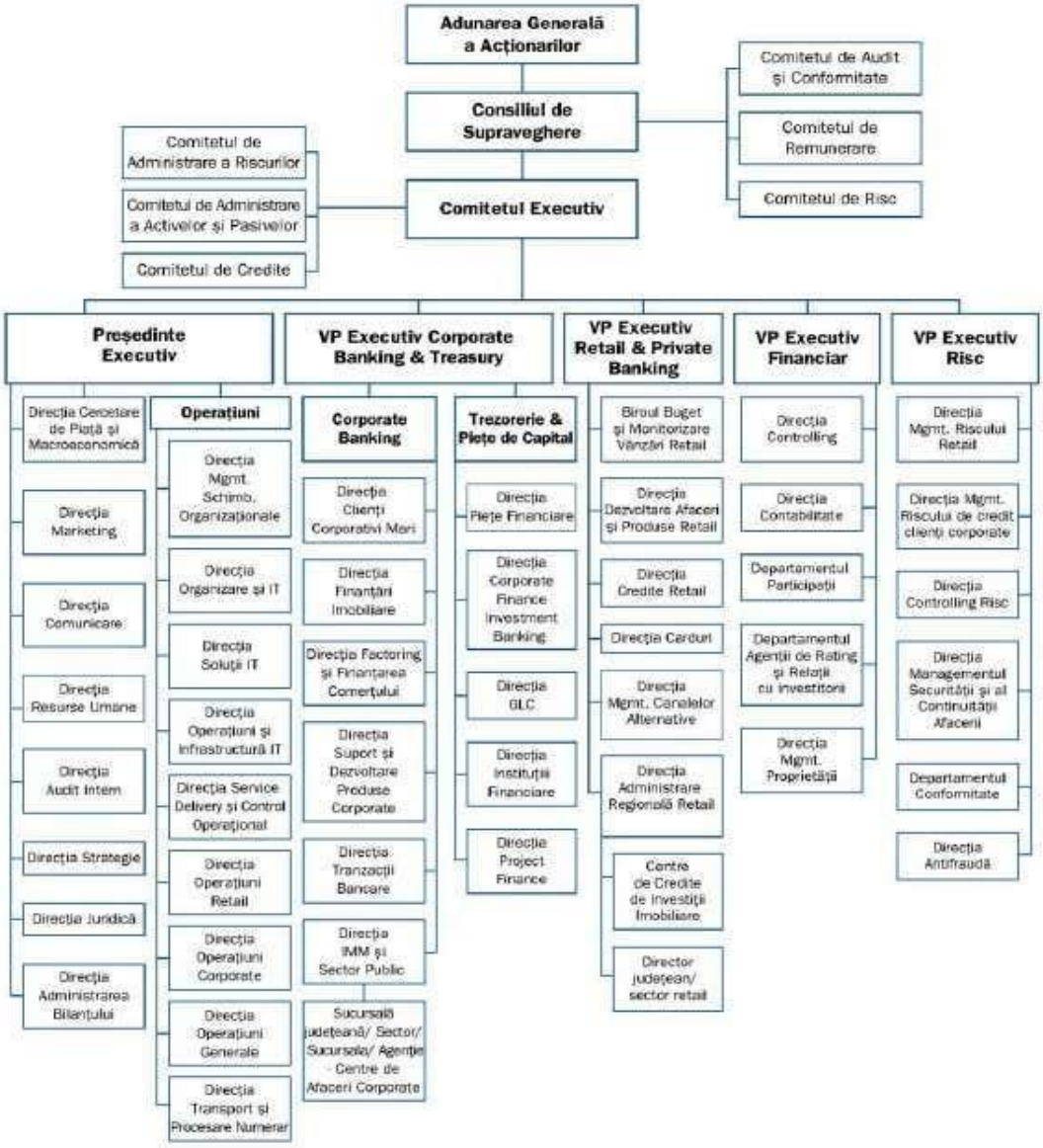


Figura1: Organigrama BCR

8

3. Prezentarea activității desfășurate în cadrul companiei BCR pe parcursul perioadei de practică

În perioada 27 martie - 5 aprilie 2023, am avut privilegiul de a participa la cursurile oferite de BCR în cadrul proiectului Tap That Job. Sub îndrumarea experților și specialiștilor din companie, am avut oportunitatea de a dobândi cunoștințe solide în domeniul Data Science, Python, Data Preparation și Analysis, dezvoltarea de modele de Machine Learning și explicația acestora.

Prin intermediul acestor cursuri, am avut acces la informații valoroase și practice, care au contribuit la dezvoltarea mea în aceste domenii de vârf. Angajații experimentați ai BCR ne-au ghidat în procesul de învățare, oferindu-ne o perspectivă reală asupra aplicării acestor concepte într-un mediu bancar și financiar.

Am învățat despre importanța colectării și pregătirii datelor pentru analiză, aplicarea tehnicilor de analiză a datelor și dezvoltarea de modele de Machine Learning pentru a extrage informații și a face predicții relevante. De asemenea, am explorat conceptul de model explainability, înțelegând cum să interpretăm și să explicăm rezultatele obținute de modelele de Machine Learning.

1. În cadrul cursurilor desfășurate în proiectul Tap That Job al BCR, am avut o prezentare generală a domeniului Data Science și am înțeles cum această știință transformă datele în informații valoroase și oferă un avantaj competitiv în mediul de afaceri. Am învățat că datele pot fi utilizate pentru a oferi o descriere mai precisă a situației actuale și pentru a face predicții mai precise despre viitor.
2. În cadrul cursului Python, am învățat despre caracteristicile și proprietățile acestui limbaj de programare. Am explorat structurile de date precum liste, tupluri, dicționare și seturi, și am înțeles cum să le utilizăm pentru a organiza și manipula informațiile. Am învățat despre condiții și bucle, utilizând instrucțiuni if-else și bucle for și while. De asemenea, am învățat să definim și să utilizăm funcții, grupând instrucțiuni și reutilizând codul. Aceste cunoștințe esențiale în Python ne-au oferit o bază solidă pentru a scrie cod eficient și pentru a rezolva probleme complexe.

3. La cursul de Data Preparation and Analysis am învățat despre procesul de pregătire și analiză a datelor. Am explorat aspecte cum ar fi codificarea categorială, imputarea valorilor lipsă, discretizarea, prelucrarea valorilor numerice și gestionarea outlierilor. Aceste tehnici ne-au ajutat să pregătim seturile de date pentru a le utiliza în analize ulterioare sau în algoritmi de machine learning.
4. La cursul de Machine Learning - model development, am avut o prezentare detaliată a acestui subiect. Am explorat diferiți algoritmi de machine learning și am învățat cum să le selectăm și să le aplicăm în funcție de tipurile de probleme. Am efectuat pregătirea și curățarea datelor pentru a ne asigura că sunt de calitate și relevante. Apoi, am antrenat și evaluat modelele folosind seturi de date reale, cu scopul de a obține predicții și decizii precise.
5. În cadrul cursului de Machine Learning - model explainability, am învățat tehnici și metode pentru a interpreta și înțelege procesul de luare a deciziilor în modelele de învățare automată. Am explorat instrumente precum analiza importanței caracteristicilor și vizualizarea modelelor pentru a obține insight-uri valoroase în privința predicțiilor lor. De asemenea, am practicat explicarea și prezentarea modului de funcționare a modelelor, cu scopul de a crește transparența și a genera încredere în rezultatele obținute.

4. Analiza unui set de date și interpretarea acestuia

Pentru realizarea proiectului, am utilizat datele furnizate de îndrumătorul de la practică, care conțin informații despre clienții băncii Delta Bank și sunt descrise prin diferite variabile. Scopul analizei este de a identifica clienții care intenționează să părăsească banca și factorii cauzali implicați în această decizie. Pentru a realiza această analiză, trebuie să urmăm următorii pași: a) Importarea bibliotecilor

Data Science Assignment

A. Importarea bibliotecilor

```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import seaborn as sns
        import missingno as msno
        from sklearn.model_selection import train_test_split
        from sklearn.ensemble import RandomForestClassifier
        from xgboost import XGBClassifier
        from sklearn.metrics import accuracy_score, precision_score, recall_score, roc_auc_score, confusion_matrix, roc_curve
        import shap
        import pickle
```

Fig. 4.1 Importarea bibliotecilor Python

b) Citirea setului de date

B. Citirea setului de date

```
In [2]: path = 'C:\\Users\\valen\\Desktop\\data\\dataset.csv'

In [3]: data = pd.read_csv(path)
```

Fig. 4.2 Citirea setului de date

c) Vizualizare, prelucrarea si analiza setului de date

În primul rând, verificăm tipul datelor și dimensiunea acestora.

```
In [7]: data.dtypes

Out[7]: CLIENTNUM                   int64
        Attrition_Flag             object
        Customer_Age              float64
        Gender                     object
        Dependent_count           float64
        Education_Level            object
        Marital_Status             object
        Income_Category            object
        Card_Category              object
        Months_on_book              int64
        Total_Relationship_Count    int64
        Months_Inactive_12_mon      int64
        Contacts_Count_12_mon       int64
        Credit_Limit              float64
        Total_Used_Bal              int64
        Total_Unused_Bal          float64
        Total_Amt_Chng_Q4_Q1      float64
        Total_Trans_Amt             int64
        Total_Trans_Ct              int64
        Total_Ct_Chng_Q4_Q1       float64
        Avg_Utilization_Ratio     float64
        dtype: object
```

Se poate observa că există 21 de coloane care furnizează diverse informații despre un client bancar, cum ar fi vârsta, venitul, nivelul de educație, starea civilă, etc. Datele întâlnite în aceste coloane pot fi sub formă de float, int sau object.

Fig. 4.3 Tipul datelor

a) Dimensiune:

```
In [4]: data.shape
Out[4]: (10127, 21)
```

Fig. 4.4 Dimensiune

Din dimensiunea setului de date, putem observa că acesta conține 10127 de linii și 21 de coloane. Este important de menționat că setul de date conține atât valori numerice, cât și valori categorice. Verificăm variabila target 'Attrition_Flag' pentru a determina dacă există valori necompletate. Se constată că toate rândurile sunt complete în ceea ce privește această variabilă. Apoi, analizăm frecvența fiecărei variante de răspuns în cadrul 'Attrition_Flag'. Pentru a simplifica analiza, vom transforma variabilele din tabel într-un set de 0 pentru cei care au părăsit sistemul bancar și de 1 pentru cei care au rămas. Vom analiza și procentual aceste date.

c) Verificăm frecvența valorilor și procentajul acestora

```
In [16]: data['Attrition_Flag'].value_counts()
Out[16]: Existing Customer    8500
         Attrited Customer    1627
         Name: Attrition_Flag, dtype: int64
```

Observăm că 8500 de clienți sunt încă în sistemul băncii noastre, iar 1627 l-au părăsit.

```
In [17]: data['Attrition_Flag'].value_counts()/len(data)*100
Out[17]: Existing Customer    83.934038
         Attrited Customer    16.065962
         Name: Attrition_Flag, dtype: float64
```

Observăm că 83.93% din clienți au ales să nu părăsească banca, pe când 16.06% au făcut acest lucru.

Fig. 4.5 Verificarea clienților

Vom crea un vector numit "categorical_variables" în care vom include toate tipurile de date obiect. Apoi vom verifica dacă există valori lipsă în acele coloane și, în caz afirmativ, le vom înlocui cu "Unknown". Scopul acestui proces este să putem analiza procentual în ce categorii se încadrează clienții noștri.

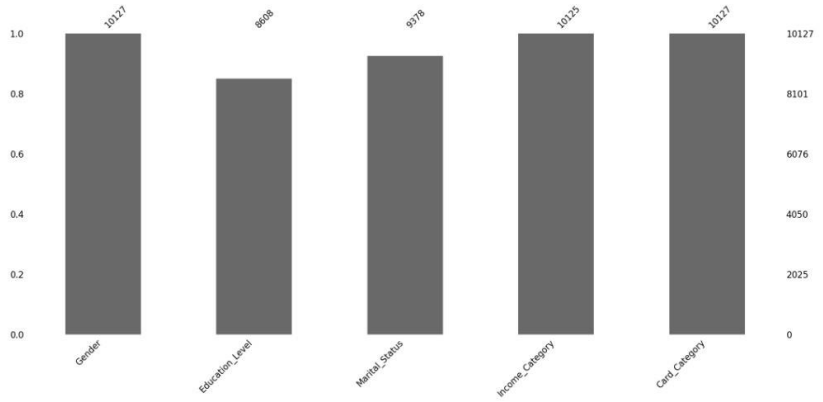
Variabilele categorice sunt cele de tip object, asadar pe acestea le selectam

```
In [23]: categorical_variables = [col for col in data.columns if data[col].dtypes=='object']

In [24]: print('In tabel exista',len(categorical_variables),'si acestea sunt:',categorical_variables)
         In tabel exista 5 si acestea sunt: ['Gender', 'Education_Level', 'Marital_Status', 'Income_Category', 'Card_Category']
```

Fig. 4.6 Variabile categorice

```
In [27]: msno.bar(data[categorical_variables])
Out[27]: <Axes: >
```



Acest grafic ne ajută să vizualizăm valorile lipsă din cadrul variabilelor.

Fig. 4.7 Vizualizarea valorilor categorice lipsă

```
In [33]: data['Gender'].value_counts()/len(data)*100
Out[33]: F    52.908068
         M    47.091932
         Name: Gender, dtype: float64

In [34]: data['Education_Level'].value_counts()/len(data)*100
Out[34]: Graduate         30.887726
         High School      19.877555
         Unknown          14.999506
         Uneducated       14.683519
         College          10.002962
         Post-Graduate     5.095290
         Doctorate         4.453441
         Name: Education_Level, dtype: float64

In [35]: data['Marital_Status'].value_counts()/len(data)*100
Out[35]: Married     46.282216
         Single      38.935519
         Unknown      7.396070
         Divorced     7.386195
         Name: Marital_Status, dtype: float64

In [36]: data['Income_Category'].value_counts()/len(data)*100
Out[36]: Less than $40K    35.153550
         $40K - $60K       17.665646
         $80K - $120K      15.157500
         $60K - $80K       13.844179
         abc               10.980547
         $120K +            7.178829
         Unknown            0.019749
         Name: Income_Category, dtype: float64

In [37]: data['Card_Category'].value_counts()/len(data)*100
Out[37]: Blue        93.176656
         Silver       5.480399
         Gold         1.145453
         Platinum     0.197492
         Name: Card_Category, dtype: float64
```

Observăm că în ceea ce privește coloana "Income_Category", există o ramură de venit denumită "abc" care nu ne este utilă și pe care o putem muta într-un alt loc, împreună cu categoria "unknown".

Fig. 4.8 Procente pentru variabilele categorice

Folosim encoding și adăugăm valoarea 'abc' din variabila 'Income_Category' în 'Unknown'.

```
In [38]: data['Income_Category'] = np.where(data['Income_Category'] == 'abc','Unknown',data['Income_Category'])

In [39]: data['Income_Category'].value_counts()/len(data)*100

Out[39]: Less than $40K    35.153550
         $40K - $60K       17.665646
         $80K - $120K      15.157500
         $60K - $80K       13.844179
         Unknown           11.000296
         $120K +            7.178829
         Name: Income_Category, dtype: float64
```

Fig 4.9 Înlocuirea cu "Unknown"

Se va repeta acest proces și pentru valorile numerice.

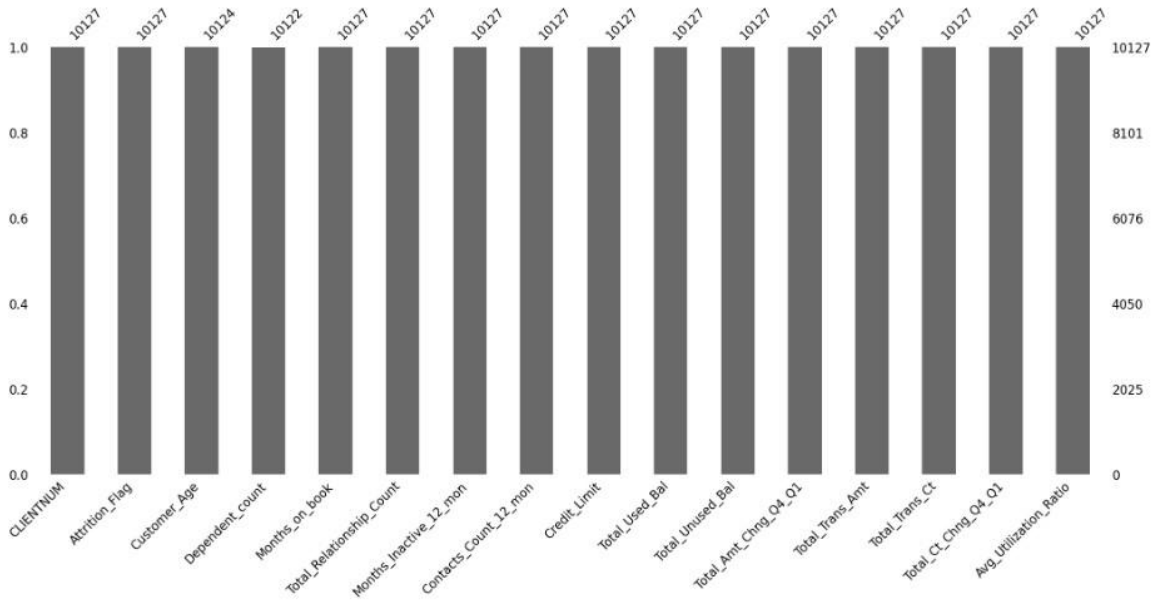
e) Explorarea variabilelor numerice

```
In [41]: numerical_columns = [col for col in data.columns if data[col].dtypes != 'object']
```

Fig. 4.10 Valorile numerice

```
In [46]: msno.bar(data[numerical_columns])

Out[46]: <Axes: >
```



Prin acest grafic observăm că lipsesc 5 valori din Dependent_count și 3 valori din Customer_Age

Fig. 4.11 Vizualizarea valorilor numerice lipsă

```
In [48]: data['Dependent_count'] = data['Dependent_count'].fillna(data['Dependent_count'].min())

In [49]: data['Customer_Age']

Out[49]: 0        45.0
         1        49.0
         2        51.0
         3        40.0
         4        40.0
                  ...
         10122    50.0
         10123    41.0
         10124    44.0
         10125    30.0
         10126    43.0
         Name: Customer_Age, Length: 10127, dtype: float64

In [50]: data['Customer_Age'] = data['Customer_Age'].fillna(data['Customer_Age'].mean())

In [51]: data['Dependent_count'].isnull().sum()
         data['Customer_Age'].isnull().sum()

Out[51]: 0
```

Acum putem observa că nu mai există valori nule in variabilele numerice.

Fig. 4.12 Completarea valorilor numerice

Detectăm valorile extreme/outlier (adică valori mult mai mici sau mult mai mari decât majoritatea observațiilor). În cazul în care găsim astfel de valori, le vom înlocui fie cu valoarea minimă dacă sunt mai mici decât limita minimă admisă, fie cu valoarea maximă dacă sunt mai mari decât limita maximă admisă.

g) Detectam valorile extreme (outliers)

```
In [54]: q1 = data['Credit_Limit'].quantile(0.25)
         q3 = data['Credit_Limit'].quantile(0.75)
         IQR = q3 - q1
         lower_limit = q1 - 3 * IQR
         upper_limit = q3 + 3 * IQR

In [55]: data[(data['Credit_Limit']<lower_limit) | (data['Credit_Limit']>upper_limit)]['Credit_Limit']

Out[55]: Series([], Name: Credit_Limit, dtype: float64)

In [56]: data[(data['Credit_Limit']<lower_limit) | (data['Credit_Limit']>upper_limit)]
```

Out[56]:

CLIENTNUM	Attrition_Flag	Customer_Age	Gender	Dependent_count	Education_Level	Marital_Status	Income_Category	Card_Category	Months_on_book	...

0 rows × 21 columns

Fig. 4.13 Outlieri

Observăm că următoarele variabile sunt afectate de outlier-i: Total_Ct_Chng_Q4_Q1, Total_Trans_Amt, Total_Amt_Chng_Q4_Q1, Attrition_Flag. Vom înlocui valorile outlier cu limita inferioară, atunci când acestea sunt mai mici decât limita, sau cu limita superioară, dacă valorile sunt mai mari decât aceasta.

```
In [59]: affected_by_outliers = ['Total_Ct_Chng_Q4_Q1','Total_Trans_Amt','Total_Amt_Chng_Q4_Q1','Attrition_Flag']

In [60]: def censoring_outliers(dataframe,column):
             q1 = dataframe[column].quantile(0.25)
             q3 = dataframe[column].quantile(0.75)
             IQR = q3-q1
             lower_limit = q1 - 3 * IQR
             upper_limit = q3 + 3 * IQR
             dataframe[column] = np.where(dataframe[column] < lower_limit, lower_limit, np.where(dataframe[column] > upper_limit, upper_l

In [61]: for variable in affected_by_outliers:
             censoring_outliers(data,variable)
```

Fig. 4.14 Eliminare outlieri

Vom crea un heatmap pentru caracteristicile primite în setul de date.

```
In [66]: fig, ax = plt.subplots(figsize = (24,10))
         sns.heatmap(correlation, annot = True, square = True, fmt = '.2f')
         plt.show()
```

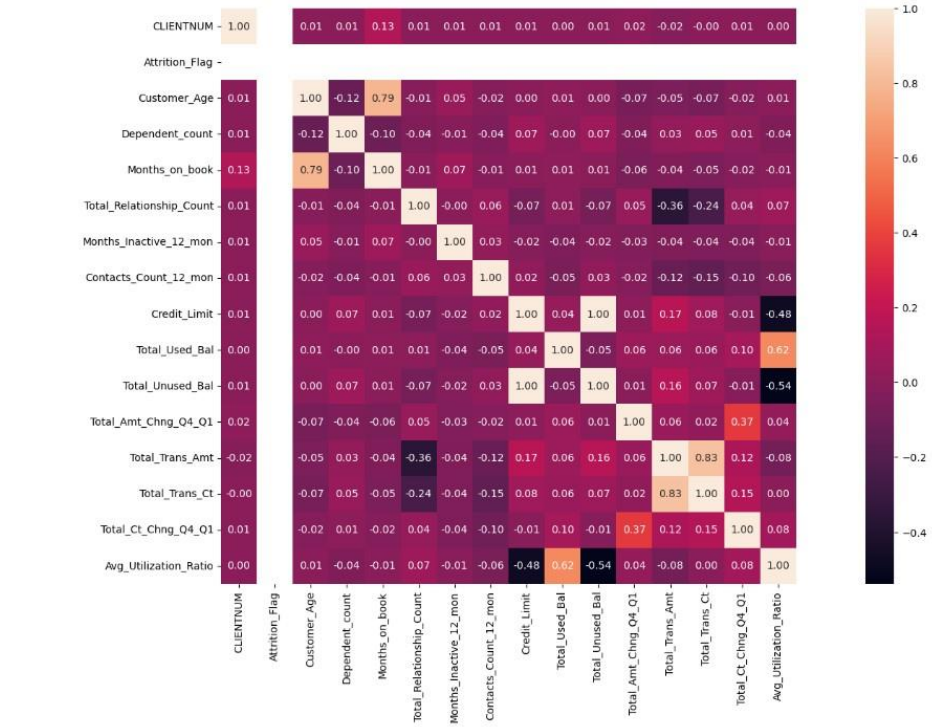


Fig 4.15 HeatMap

Observăm o puternică corelație pozitivă de de 0,83 între Total_Trans_Ct și Total_Trans_Amt. De asemenea, există o puternică corelație de 0,79 între Months_On_Book și Customer_Age. În plus, există o corelație medie pozitivă de 0,62 între Total_Used_Bal și Avg_Utilization_Ratio.

În ceea ce privește corelațiile slabe, putem observa o corelație pozitivă de 0,37 între Total_Trans_Ct și Attrition_Flag, precum și între Total_Ct_Chng_Q4_Q1 și Total_Amt_Chng_Q4_Q1. Există, de asemenea, o corelație slabă de 0,31 între Total_Ct_Chng_Q4_Q1 și Attrition_Flag.

Pe de altă parte, există o corelație slabă negativă de -0,36 între Total_Trans_Amt și Total_Relationship_Count. De asemenea, putem observa o corelație negativă slabă de -0,48 între Credit_Limit și Avg_Utilization_Ratio. Există, de asemenea, o corelație negativă medie de -0,54 între Avg_Utilization_Ratio și Total_Unused_Bal.

Celelalte corelații au un impact scăzut, ceea ce înseamnă că au o relevanță mult mai mică.

d) pregătirea modelului predictiv

Se realizează împărțirea setului de date în setul de antrenare și setul de testare

3. Seturi de antrenare si testare

```
In [77]: X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=10)

In [78]: X_train.shape, X_test.shape, y_train.shape, y_test.shape
Out[78]: ((8101, 32), (2026, 32), (8101,), (2026,))
```

Fig. 4.16 Seturi de antranare si testare

Vom evalua criteriile de performanță utilizând algoritmul Random Forest.
Algoritmul Random Forest este o metodă de învățare automată de tip ansamblu, utilizată în problemele de clasificare și regresie. Acesta se bazează pe principiul construirii unui ansamblu (ensemble) de arbori de decizie independenți, numiți arbori în pădurea aleatoare. Pașii algoritmului Random Forest:

1. Se colectează un set de date de antrenare, format dintr-un număr de exemple cu caracteristici și etichete asociate.
2. Se creează un număr predefinit de arbori de decizie. Fiecare arbore este construit pe baza unui sub-set aleatoriu de exemple de antrenare și caracteristici alese la întâmplare.
3. Pentru fiecare arbore în pădurea aleatoare:
   a. Se selectează un sub-set aleatoriu din setul de antrenare (mărimi egale, cu înlocuire).
   b. Se selectează un sub-set aleatoriu de caracteristici (o proporție predefinită) pentru a construi arborele.
4. Se construiesc arborii de decizie pentru fiecare sub-set de date și caracteristici. Fiecare arbore este construit prin divizarea setului de date în funcție de valorile caracteristicilor, astfel încât să se maximizeze puritatea grupurilor rezultate.
5. Pentru a prezice etichetele pentru noi exemple:

a. Se aplică fiecare exemplu de test pe fiecare arbore din pădure.

b. Se colectează rezultatele și se efectuează o votare majoritară (clasificare) sau se calculează media (regresie) pentru a obține rezultatul final.

6. Se evaluează performanța modelului pe baza unor metrici precum acuratețea, precizia, revocarea sau eroarea medie pătratică.

Algoritmul Random Forest Classifier

```
In [79]: rf = RandomForestClassifier(n_estimators = 300, max_depth = 5, n_jobs = -1, random_state = 123)
```

Începem procesul de antrenare a modelului cu ajutorul algoritmului Random Forest pe setul de date pregătit anterior.

```
In [80]: rf.fit(X_train, y_train)

Out[80]: RandomForestClassifier
         RandomForestClassifier(max_depth=5, n_estimators=300, n_jobs=-1,
                                random_state=123)
```

Prezicerea rezultatelor

```
In [81]: y_predict = rf.predict(X_test)

In [82]: y_predict

Out[82]: array([1, 1, 1, ..., 0, 1, 1], dtype=int64)
```

Fig. 4.17 Algoritmul Random Forest

Performanta

În urma aplicării algoritmului Random Forest, verificăm criteriile de performanță ale modelului nostru. Acest lucru include evaluarea acurateții, preciziei, rapelului și scorului F1. De asemenea, putem examina matricea de confuzie pentru a evalua cât de bine modelul nostru poate distinge între clasele pozitive și negative.

a) Acuratețea

```
In [83]: accuracy = accuracy_score(y_test,y_predict)

In [85]: print('Pentru algoritmul Random Forest Classifier, avem o acuratete de:',accuracy)

         Pentru algoritmul Random Forest Classifier, avem o acuratete de: 0.920533070088845
```

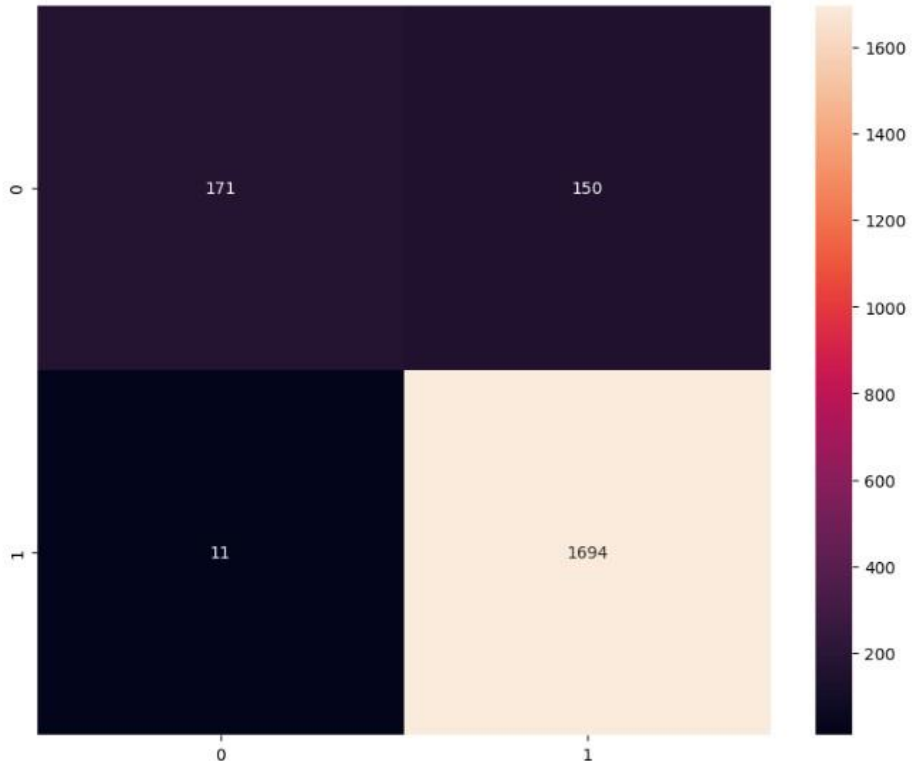
Fig. 4.18 Acuratețea Random Forest

b) Matricea de confuzie

```
In [86]: cm = confusion_matrix(y_test,y_predict)
         print(cm)
```

```
[[ 171  150]
 [  11 1694]]
```

```
In [87]: fig,ax = plt.subplots(figsize = (10, 8))
         sns.heatmap(cm,annot = True, fmt = 'd')
         plt.show()
```



Prin analizarea graficului furnizat, putem concluziona că există 1694 valori cu adevărat pozitive și 171 valori cu adevărat negative, dar și 11 valori eronat considerate negative și 150 valori eronat considerate pozitive. În urma acestor constatări, precizia rezultată este de 91,8%, sensibilitatea este de 99,3%, scorul AUC este de 76,3%, iar acuratețea globală este de 92%.

Fig. 4.19 Matricea de confuzie Random Forest

Verificăm overfitting-ul

Verificam daca exista overfitting

```
In [90]: y_train_predict = rf.predict(X_train)
```

```
In [91]: precision = precision_score(y_train, y_train_predict)
         recall = recall_score(y_train, y_train_predict)
         print('Precision score:', precision)
         print('Recall score:', recall)
```

```
Precision score: 0.9142003801248981
Recall score: 0.9910228108903606
```

Interpretarea datelor indică că valorile preciziei și sensibilității s-au modificat într-o măsură neglijabilă. Precizia a scăzut cu aproximativ 0,004, iar sensibilitatea a scăzut cu aproximativ 0,002. Acest lucru sugerează că nu există un overfitting semnificativ care să afecteze rezultatele obținute.

d) Scorul AUC

```
In [92]: auc_score = roc_auc_score(y_test, y_predict)
         print('AUC score is:', auc_score)
```

```
AUC score is: 0.763129333735303
```

Verificam daca exista overfitting

```
In [107]: y_predict_train = rf.predict(X_train)
          auc_score = roc_auc_score(y_train, y_predict_train)
          print('AUC score is:', auc_score)
```

```
AUC score is: 0.7535512216779522
```

Diferența dintre setul de testare și cel de antrenare este prea mică pentru a avea un overfitting semnificativ care să influențeze rezultatele.

Fig. 4.20 Overfitting Random Forest

Repetăm aceeași pași pentru algoritmul XGBoost:

1. Se colectează un set de date de antrenare, care conține exemple cu caracteristici și etichete asociate.
2. Se configurează parametrii algoritmului XGBoost, cum ar fi numărul de arbori de decizie, adâncimea maximă a arborelui, rata de învățare (learning rate) etc.
3. Se construiesc arborii de decizie în serie. Fiecare arbore este construit în funcție de erorile reziduale ale predicțiilor anterioare.
4. Se optimizează funcția obiectiv utilizând algoritmul de gradient boosting. Acesta caută combinația optimă de arbori de decizie pentru a minimiza eroarea modelului.
5. Pentru a prezice etichetele pentru noi exemple:
   a) Se aplică fiecare exemplu de test pe fiecare arbore de decizie.
   b) Se colectează rezultatele și se obține o predicție finală prin suma sau media predicțiilor arborilor.
6. Se evaluează performanța modelului utilizând metrici precum acuratețea, precizia, revocarea sau eroarea medie pătratică.

Algoritmul XGBoost (eXtreme Gradient Boosting)

```
In [94]: xgb = XGBClassifier(n_estimators=250, max_depth=4, n_jobs=-1, random_state=100)

In [95]: xgb.fit(X_train, y_train)

Out[95]: XGBClassifier
XGBClassifier(base_score=None, booster=None, callbacks=None,
              colsample_bylevel=None, colsample_bynode=None,
              colsample_bytree=None, early_stopping_rounds=None,
              enable_categorical=False, eval_metric=None, feature_types=None,
              gamma=None, gpu_id=None, grow_policy=None, importance_type=None,
              interaction_constraints=None, learning_rate=None, max_bin=None,
              max_cat_threshold=None, max_cat_to_onehot=None,
              max_delta_step=None, max_depth=4, max_leaves=None,
              min_child_weight=None, missing=nan, monotone_constraints=None,
              n_estimators=250, n_jobs=-1, num_parallel_tree=None,
              predictor=None, random_state=100, ...)
```

Prezicerea rezultatelor

```
In [96]: y_predict = xgb.predict(X_test)
         y_predict

Out[96]: array([1, 1, 1, ..., 0, 1, 1])
```

Fig. 4.21 Algoritmul XGBoost

Performanta

a) Acuratețea

```
In [98]: accuracy = accuracy_score(y_test, y_predict)
         print('Accuracy for XGBoost is:', accuracy)

         Accuracy for XGBoost is: 0.9693978282329714
```

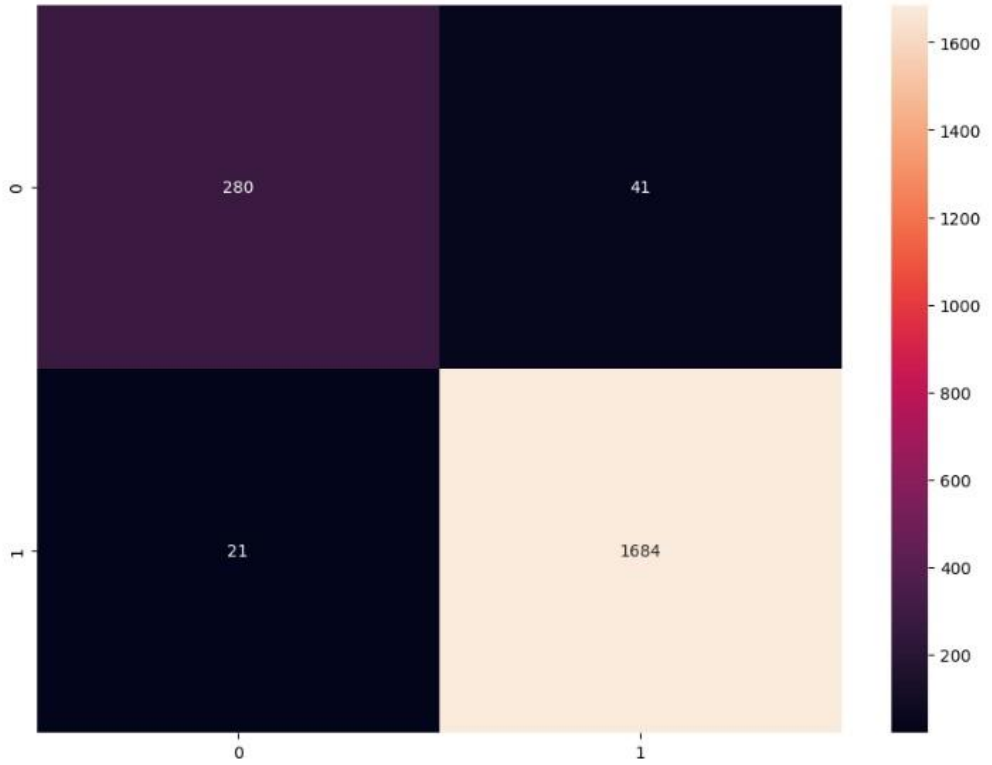
Fig. 4.22 Acuratețea XGBoost

b) Matricea de confuzie

In [99]:
```
cm = confusion_matrix(y_test, y_predict)
print(cm)
```

```
[[ 280   41]
 [  21 1684]]
```

In [100]:
```
fig, ax = plt.subplots(figsize = (11, 8))
sns.heatmap(cm, annot = True, fmt='d')
plt.show()
```



Bazându-ne pe analiza și graficul furnizate, observăm că avem 1684 valori cu adevărat pozitive și 280 valori cu adevărat negative, împreună cu 21 de valori fals negative și 41 valori fals pozitive. Prin urmare, putem concluziona că acuratețea este de 96,9%, precizia este de 97,6%, sensibilitatea este de 98,7%, iar scorul AUC este de 92,9%.

Fig. 4.23 Matricea de confuzie XGBoost

Verificam daca exista overfitting

In [102]:
```
y_train_predict = rf.predict(X_train)
precision = precision_score(y_train, y_train_predict)
recall = recall_score(y_train, y_train_predict)
print('Precision score:', precision)
print('Recall score:', recall)
```

```
Precision score: 0.9142003801248981
Recall score: 0.9910228108903606
```

Scorul de sensibilitate s-a modificat cu o valoare neglijabilă de aproximativ 0,035, sugestiv pentru absența unui overfitting sau underfitting relevant. În schimb, scorul de precizie a scăzut cu o valoare semnificativă de peste 0,6, ceea ce sugerează existența unui underfitting important.

d) Scorul AUC

In [103]:
```
auc_score = roc_auc_score(y_test, y_predict)
print('AUC score', auc_score)
```

```
AUC score 0.9299787138798294
```

Verificam daca exista overfitting

In [105]:
```
y_predict_train = xgb.predict(X_train)
auc_score = roc_auc_score(y_train, y_predict_train)
print('AUC score is:', auc_score)
```

```
AUC score is: 1.0
```

Se pare că există un overfitting în ceea ce privește scorul AUC, având în vedere că diferența dintre rezultatele de la antrenare și cele de la testare este de aproximativ 0,7. Scorul AUC obținut la antrenare a atins chiar valoarea maximă de 1, ceea ce sugerează o posibilă adaptare excesivă a modelului la datele de antrenare.

Fig. 4.24 Overfitting XGBoost

Căutăm posibile hipervalori

Prelucrarea hipervalorilor

a) Vom identifica acum posibile hipervalori pentru algoritmul XGBoost.

In [109]:
```
n_estimators = [150,250]
max_depth = [3,4]
learning_rate = [0.1,0.05]
```

b) Cautam hipervalorile

In [110]:
```
results = []
for est in n_estimators:
    for md in max_depth:
        for lr in learning_rate:
            xgb = XGBClassifier(n_estimators = est, max_depth = md, learning_rate = lr, n_jobs = -1, random_state = 100, subsamp
            xgb.fit(X_train, y_train)
            y_predict = xgb.predict(X_test)
            auc_score = roc_auc_score(y_test,y_predict)
            results.append(['estimators', est, 'max_depth', md, 'leraning_rate', lr, 'auc', auc_score])
```

In [111]:
```
print(results)
```

```
[['estimators', 200, 'max_depth', 3, 'leraning_rate', 0.1, 'auc', 0.9262769388183918], ['estimators', 200, 'max_depth', 3, 'ler
aning_rate', 0.05, 'auc', 0.9149889001562 2], ['estimators', 200, 'max_depth', 4, 'leraning_rate', 0.1, 'auc', 0.93056522414375
9], ['estimators', 200, 'max_depth', 4, 'leraning_rate', 0.05, 'auc', 0.9138158796283607], ['estimators', 300, 'max_depth', 3,
'leraning_rate', 0.1, 'auc', 0.9268634490823215], ['estimators', 300, 'max_depth', 3, 'leraning_rate', 0.05, 'auc', 0.914695645
0242553], ['estimators', 300, 'max_depth', 4, 'leraning_rate', 0.1, 'auc', 0.9284210814810755], ['estimators', 300, 'max_dept
h', 4, 'leraning_rate', 0.05, 'auc', 0.9196617973524817]]
```

Fig. 4.25 Hipervalori

e) Se examinează indicatorii lift și gain pentru evaluarea performanței modelului predictiv. Lift calculează eficacitatea modelului, iar o valoare mai mare de 1 indică o performanță superioară față de o ghidare aleatoare. Gain măsoară impactul incremental al modelului pregătit și se exprimă ca procentul de creștere pozitivă.

a) Graficul Lift

```
fig,ax=plt.subplots(figsize=(15,9))
barplot=plt.bar(lift_gain_report['Decil group'],lift_gain_report['Lift'])
plt.title('Lift bar plot')
plt.xlabel('Decil group')
plt.ylabel('Lift')
plt.xticks(lift_gain_report['Decil group'])

for b in barplot:
    plt.text(b.get_x()+b.get_width()/2,b.get_height()+0.1,round(b.get_height(),2),ha='center')

plt.show()
```

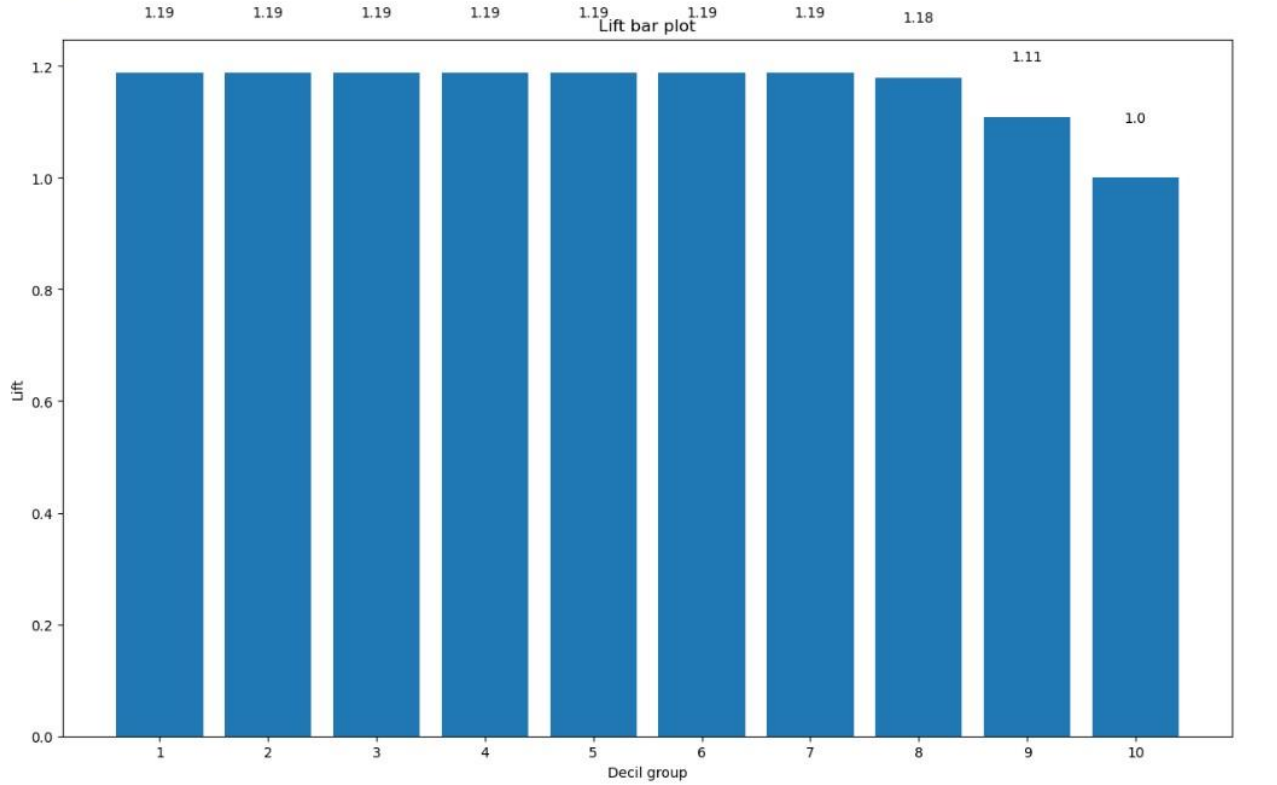


Fig. 4.26 Grafic Lift

Din analiza graficului pentru indicatorul Lift, se poate observa că pentru 7 din cele 10 grupuri Decile, valoarea indicatorului este de 1,19. Pentru grupul Decil 8, valoarea Lift este de 1,18, pentru Decil 9 este de 1,11, iar pentru Decil 10 este de 1.

b) Graficul Gain

```
In [162...]: lift_gain_report['Random Selection']=lift_gain_report['Decil group']/lift_gain_report['Decil group'].max()

In [163...]: fig,ax=plt.subplots(figsize=(14,8))
             sns.lineplot(data=lift_gain_report,x=lift_gain_report['Decil group'],y=lift_gain_report['Gain'])
             sns.lineplot(data=lift_gain_report,x=lift_gain_report['Decil group'],y=lift_gain_report['Random Selection'])
             plt.title('Gain plot')
             plt.xticks(lift_gain_report['Decil group'])
             plt.yticks(round(lift_gain_report['Gain'],2))

Out[163]: ([<matplotlib.axis.YTick at 0x18c6af534c0>,
              <matplotlib.axis.YTick at 0x18c6af52c60>,
              <matplotlib.axis.YTick at 0x18c647a1540>,
              <matplotlib.axis.YTick at 0x18c647a2200>,
              <matplotlib.axis.YTick at 0x18c647a22f0>,
              <matplotlib.axis.YTick at 0x18c647a33a0>,
              <matplotlib.axis.YTick at 0x18c647a2e30>,
              <matplotlib.axis.YTick at 0x18c647a8250>,
              <matplotlib.axis.YTick at 0x18c6af523b0>,
              <matplotlib.axis.YTick at 0x18c647c53c0>],
             [Text(0, 0.0, '0.00'),
              Text(0, 0.06, '0.06'),
              Text(0, 0.17, '0.17'),
              Text(0, 0.29, '0.29'),
              Text(0, 0.41, '0.41'),
              Text(0, 0.52, '0.52'),
              Text(0, 0.64, '0.64'),
              Text(0, 0.76, '0.76'),
              Text(0, 0.88, '0.88'),
              Text(0, 1.0, '1.00')])
```

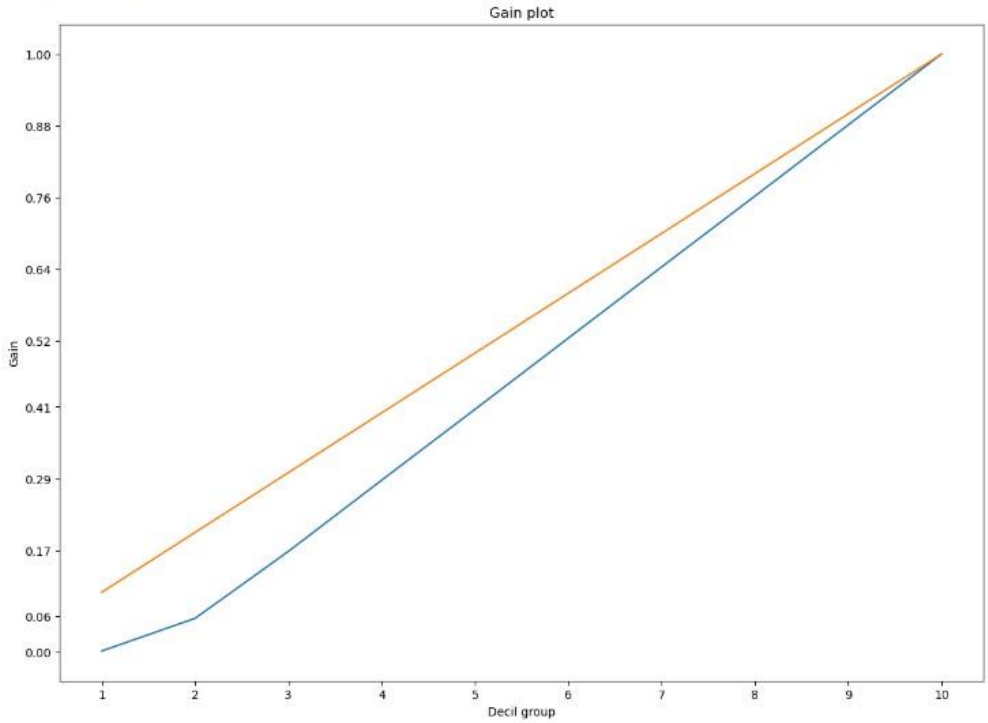


Fig. 4.27 Graficul Gain

Se poate observa cum valoarea indicatorului Gain variază pentru fiecare punct din grupul Decil. Pentru primul punct va fi egal cu 0, pentru al doilea punct va fi egal cu 0,06, pentru al treilea punct va fi egal cu 0,17, pentru al patrulea punct va fi egal cu 0,29, pentru al cincilea punct va fi egal cu 0,41, pentru al șaselea punct va fi egal cu 0,52, pentru al șaptelea punct va fi egal cu 0,64, pentru al optulea punct va fi egal cu 0,76, pentru al nouălea punct va fi egal cu 0,88, iar pentru al zecelea punct valoarea va ajunge la 1.

f) Examinăm coeficienții de importanță pentru fiecare variabilă și le vom reprezenta în graficele SHAP.

In [135…
```
feature_importance=pd.DataFrame()
feature_importance['Variable']=feat_imp.keys()
feature_importance['Importance Value']=feat_imp.values()
feature_importance['%Importance Feature']=feature_importance['Importance Value']/feature_importance['Importance Value'].sum()*100
feature_importance.sort_values(by=['Importance Value'],ascending=True)
```

Out[135]:

	Variable	Importance Value	%Importance Feature
27	Income_Category_$40K – 60K	1.507605	0.003247
16	Education_Level_Doctorate	3.804367	0.008194
19	Education_Level_Post-Graduate	5.802805	0.012499
31	Income_Category_Unknown	5.852596	0.012606
21	Education_Level_Unknown	8.503635	0.018316
29	Income_Category_$80K – 120K	8.905509	0.019182
22	Marital_Status_Divorced	9.740646	0.020980
17	Education_Level_Graduate	10.775385	0.023209
18	Education_Level_High School	10.846713	0.023363
15	Education_Level_College	10.916389	0.023513
20	Education_Level_Uneducated	11.028137	0.023754
25	Marital_Status_Unknown	12.804751	0.027580
26	Income_Category_$120K +	24.908482	0.053651
30	Income_Category_Less than $40K	28.509575	0.061407
24	Marital_Status_Single	52.364826	0.112789
28	Income_Category_$60K – 80K	64.864021	0.139711
2	Dependent_count	132.068588	0.284464
23	Marital_Status_Married	187.260574	0.403342
0	CLIENTNUM	337.082855	0.726045
3	Months_on_book	615.269653	1.325234
9	Total_Unused_Bal	625.514893	1.347301
7	Credit_Limit	815.682800	1.756905
6	Contacts_Count_12_mon	903.004456	1.944988
14	Avg_Utilization_Ratio	1186.154297	2.554867
5	Months_Inactive_12_mon	1212.135986	2.610829
1	Customer_Age	1529.305664	3.293983
10	Total_Amt_Chng_Q4_Q1	2801.148193	6.033415
4	Total_Relationship_Count	3694.204590	7.956976
13	Total_Ct_Chng_Q4_Q1	4541.037109	9.780975
8	Total_Used_Bal	5177.433105	11.151713
12	Total_Trans_Ct	11149.445312	24.014876
11	Total_Trans_Amt	11249.362305	24.230088

Fig. 4.28 Importanța variabilelor

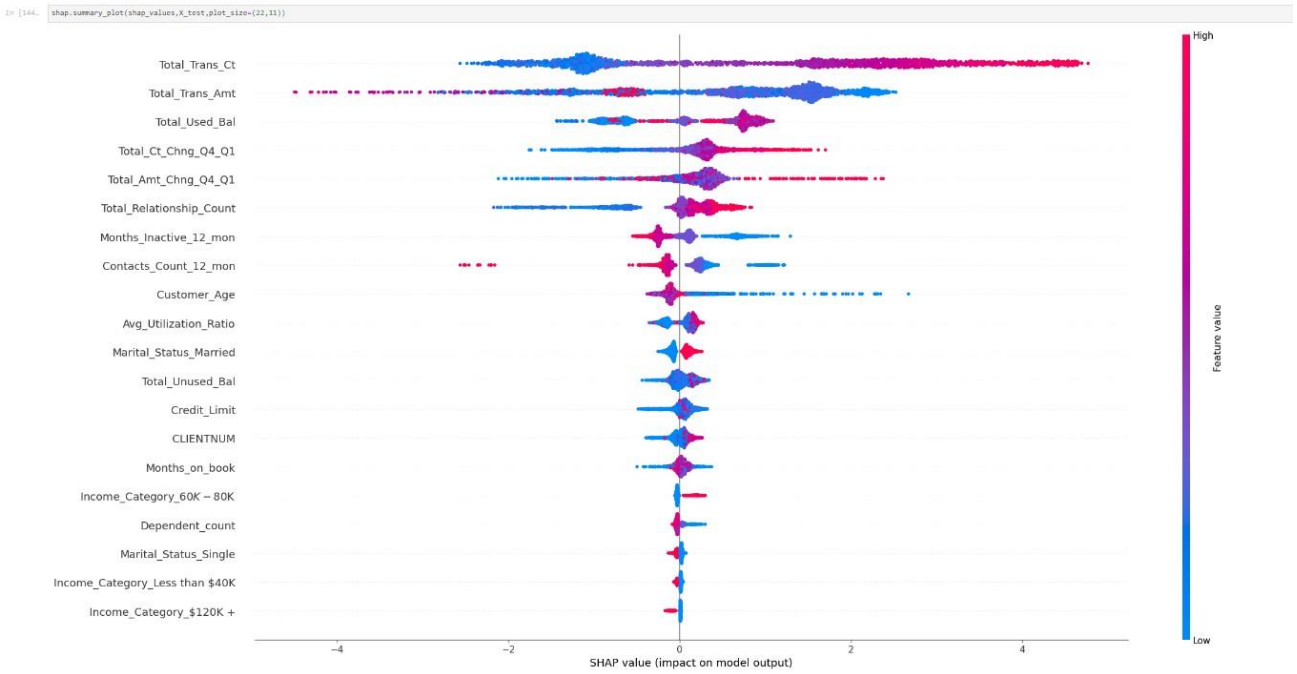


Fig. 4.29 Grafic SHAP

Rezultatele tuturor analizelor efectuate demonstrează că acuratețea modelului depășește 99%.

Concluzii

În urma acestui proiect, bazat pe data science, s-a reușit să se valorifice puterea datelor și să se extragă informații valoroase. Prin colectarea meticuloasă a datelor, preprocesarea și analiza acestora, am obținut o înțelegere mai profundă a problemei în discuție și am dezvoltat modele robuste care furnizează predicții exacte și recomandări acționabile. Proiectul a subliniat importanța procesului decizional bazat pe date, demonstrând cum tehnici de data science pot descoperi modele ascunse, pot optimiza procese și pot stimula inovarea. Această inițiativă a evidențiat potențialul transformator al data science-ului în rezolvarea problemelor complexe și generarea de informații valoroase, cu un impact pozitiv asupra întreprinderilor, industriilor și societății în general.

Prin experiența acumulată în cadrul stagiului de practică la BCR, am dobândit o înțelegere mai profundă a domeniului analizei datelor și am realizat importanța acestora atunci când sunt analizate și interpretate corespunzător.

Bibliografie

1. https://www.bcr.ro/ro/despre-noi/grup-bcr
2. https://en.wikipedia.org/wiki/Random_forest
3. https://en.wikipedia.org/wiki/Random_forest
4. https://www.linkedin.com/company/bcr/?originalSubdomain=ro&original_referer=