Inteligență Artificială: Tema 2 - ML aplicat

Seria CD – Ploscaru Alexandru – 331CD

Descrierea temei

Pentru implementarea acestei teme am folosit cunoștințele învățate în cadrul laboratoarelor și cursurilor de Inteligență Artificială, precum și informații de pe internet, cum ar fi librării deja existente și funcții matematice sau grafice.

Primul pas a fost analiza seturilor de date atribuite acestei teme, SalaryPrediction_full.csv și credit_risk_full.csv, pentru a determina caracteristicile fiecărui atribut, valorile lipsă și cele extreme și corelațiile dintre atribute. În urma acestei analize am constatat că datele trebuie modificate înainte de a fi procesate de un model de învățare automată pentru rezultate optime. Datele au fost preprocesate cu ajutorul explicațiilor oferite în enunțul temei și anumite funcții din librăriile folosite. Ulterior acestei preprocesări, am folosit 2 algoritmi, fiecare în două variante, una din librăria *scikit-learn* și una implementată manual, preluată din cadrul laboratorului.

Am folosit următoarele librării:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import chi2_contingency
import math

from copy import deepcopy
from typing import Optional, Dict, Callable

from sklearn.tree import DecisionTreeClassifier
from sklearn import metrics
from sklearn.model_selection import train_test_split

from sklearn.neural_network import MLPClassifier
from typing import List

from sklearn.experimental import enable_iterative_imputer
from sklearn.impute import IterativeImputer
from sklearn.impute import SimpleImputer

from sklearn import preprocessing
```

Analiza datelor

Salary Prediction

Setul de date are următoarele coloane, din care coloana *money* este rezultatul dorit:

	fnl	hpw	relation	gain	country	job	edu_int	years	loss	work_type	partner	edu	gender	race	prod	gtype	money
0	264627	40.00000	NotM	0	United-States	Adm-clerical	10	38	0	Priv	D	SC	NaN	White	77	DC	<=50K
1	151369	40.00000	NotF	0	United-States	Craft-repair	9	62	0	LGov	NM	HSG	M	White	87	AC	<=50K
2	188615	60.00000	H	0	United-States	Sales	13	42	0	SelfInc	MCS	B	M	White	77	AC	>50K
3	151089	55.00000	H	0	United-States	Exec-managerial	10	41	2415	SelfInc	MCS	SC	M	White	107	AC	>50K
4	177625	40.00000	NotF	0	United-States	Machine-op-inspct	9	25	0	Priv	NM	HSG	M	White	57	AC	<=50K
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...

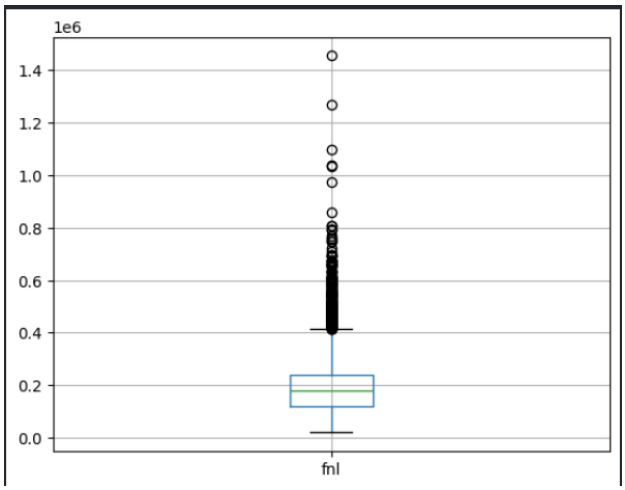
Date numerice – am folosit funcția *describe* din *pandas* pentru a afișa anumite valori relevante pentru fiecare atribut în parte:

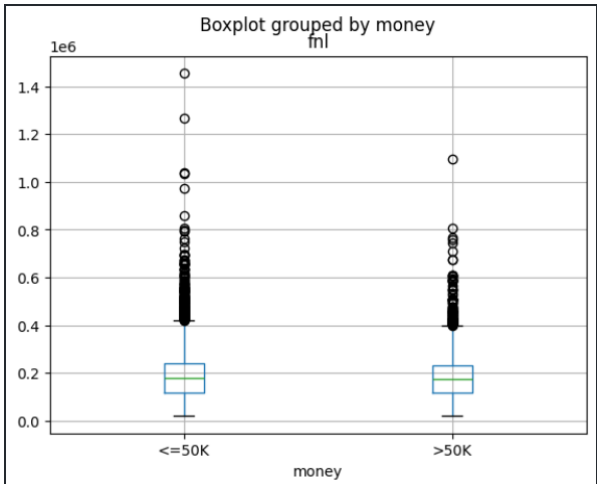
	fnl	hpw	gain	edu_int	years	loss	prod
count	9999.00000	9199.00000	9999.00000	9999.00000	9999.00000	9999.00000	9999.00000
mean	190352.90209	40.41624	979.85339	14.26203	38.64686	84.11141	2014.92759
std	106070.86269	12.51736	7003.79538	24.77084	13.74510	394.03548	14007.60450
min	19214.00000	1.00000	0.00000	1.00000	17.00000	0.00000	-28.00000
25%	118282.50000	40.00000	0.00000	9.00000	28.00000	0.00000	42.00000
50%	178472.00000	40.00000	0.00000	10.00000	37.00000	0.00000	57.00000
75%	237311.00000	45.00000	0.00000	13.00000	48.00000	0.00000	77.00000
max	1455435.00000	99.00000	99999.00000	206.00000	90.00000	3770.00000	200125.00000

Din acest tabel se poate observa că toate coloanele, mai puțin coloana *hpw* (număr de ore muncite pe săptămână), nu au date lipsă. Coloana *hpw* are câteva valori lipsă în acest set de date, existând doar 9199 valori în tabel, ceea ce înseamnă că lipsesc 800 de valori, care vor trebui adăugate în faza de preprocesare.

În continuare, am analizat datele din fiecare coloană separat, afișând pentru fiecare atribut: valoarea medie, deviația standard, valoarea minimă și maximă, valoarea percentilelor de 25%, 50% și 75%, și am afișat aceste valori cu ajutorul graficului de tip *boxplot.* Pentru fiecare coloană am afișat două *boxplot*-uri, unul cu toate valorile din coloana respectivă și unul cu valorile împărțite în funcție de atributul rezultat (*money*). Am mai afișat outputul funcției *describe* pentru fiecare atribut în parte, pentru valorile separate în funcție de atributul rezultat:

- fnl (Caracteristică socio-economică a populației din care provine individul)



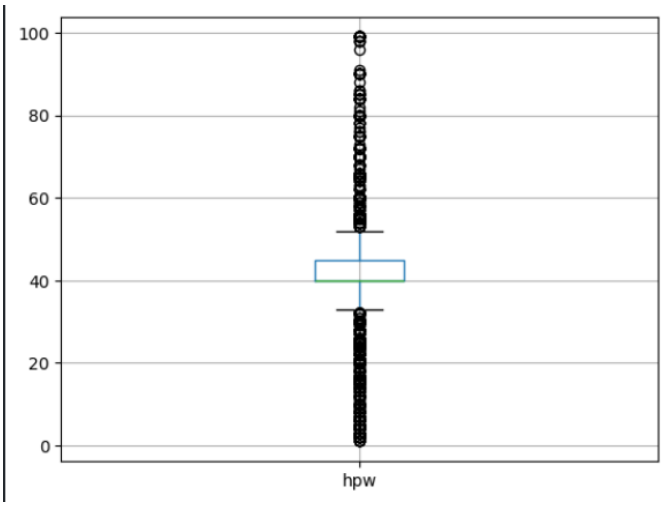


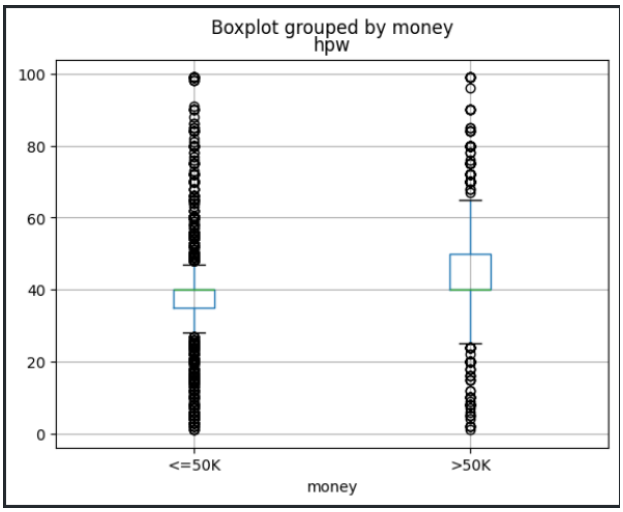

```
count      7591.00000
mean     190623.98419
std      106638.00497
min       19214.00000
25%      117854.50000
50%      180052.00000
75%      239145.50000
max     1455435.00000
Name: fnl, dtype: float64
```

```
count      2408.00000
mean     189498.34053
std      104280.09122
min       19302.00000
25%      119398.00000
50%      175804.00000
75%      231566.50000
max     1097453.00000
Name: fnl, dtype: float64
```

Valorile din stânga sunt pentru exemplele care au valoarea din coloana *money* = <=50K, iar valorile din dreapta sunt pentru *money* = >50K, iar acest lucru este valabil pentru toate coloanele în continuare.

- hpw (Număr de ore de muncă pe săptămână)



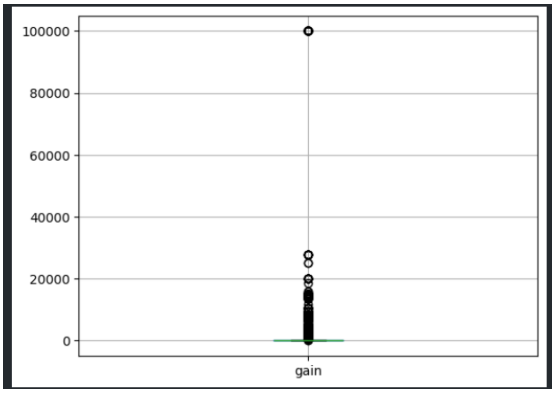


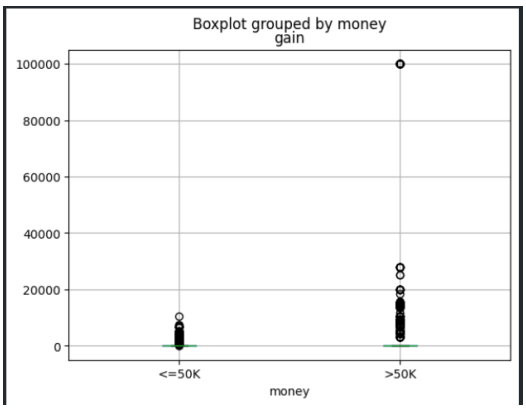
```
count   6988.00000
mean      38.86319
std       12.50687
min        1.00000
25%       35.00000
50%       40.00000
75%       40.00000
max       99.00000
Name: hpw, dtype: float64
```

```
count   2211.00000
mean      45.32474
std       11.21846
min        1.00000
25%       40.00000
50%       40.00000
75%       50.00000
max       99.00000
Name: hpw, dtype: float64
```

Valorile din stânga sunt pentru exemplele care au valoarea din coloana *money* = <=50K, iar valorile din dreapta sunt pentru *money* = >50K, dar cum coloana *hpw* are valori lipsă, aceasta diferă puțin.

- gain (Câștigul de capital)

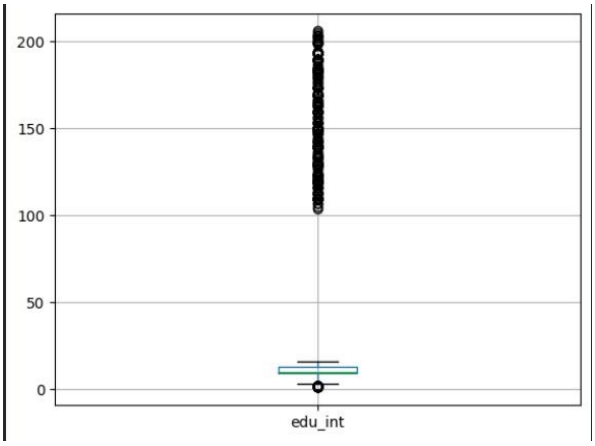


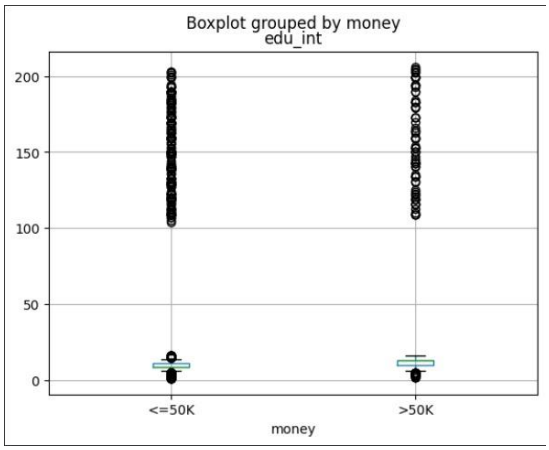



```
count    7591.00000
mean      122.01673
std       674.37591
min         0.00000
25%         0.00000
50%         0.00000
75%         0.00000
max     10566.00000
Name: gain, dtype: float64
```

```
count    2408.00000
mean     3684.10507
std     13880.99828
min         0.00000
25%         0.00000
50%         0.00000
75%         0.00000
max     99999.00000
Name: gain, dtype: float64
```

- edu_int (Numărul de ani de studiu)

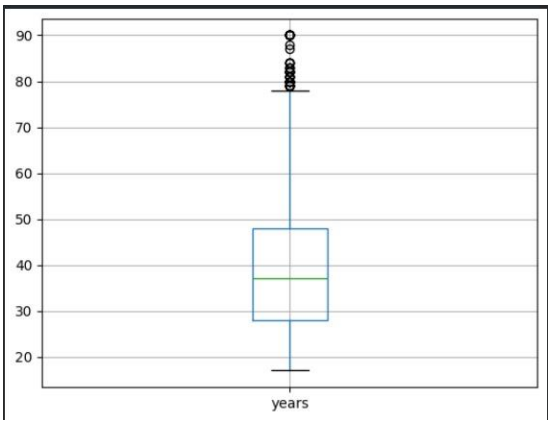


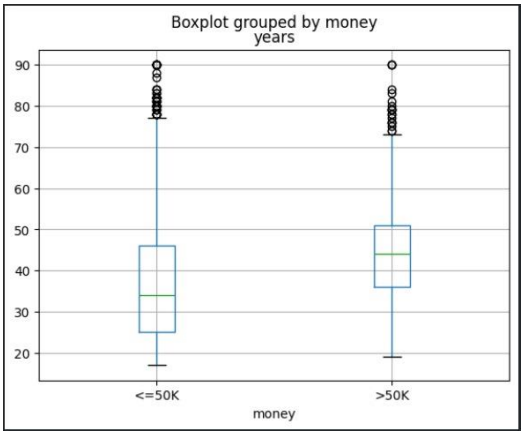



```
count   7591.00000
mean      13.74522
std       24.58773
min        1.00000
25%        9.00000
50%        9.00000
75%       11.00000
max      203.00000
Name: edu_int, dtype: float64
```

```
count   2408.00000
mean      15.89120
std       25.27549
min        2.00000
25%       10.00000
50%       13.00000
75%       13.00000
max      206.00000
Name: edu_int, dtype: float64
```

- years (Vârsta individului)

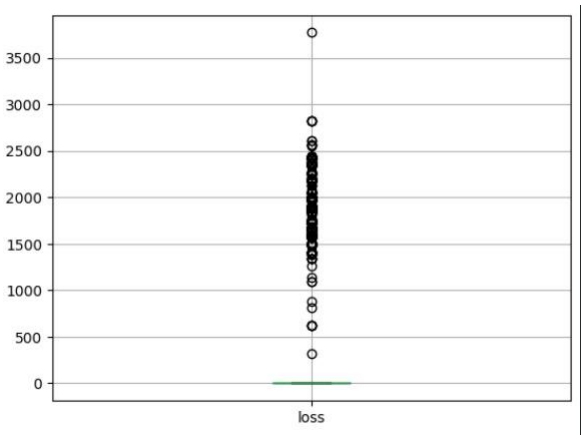


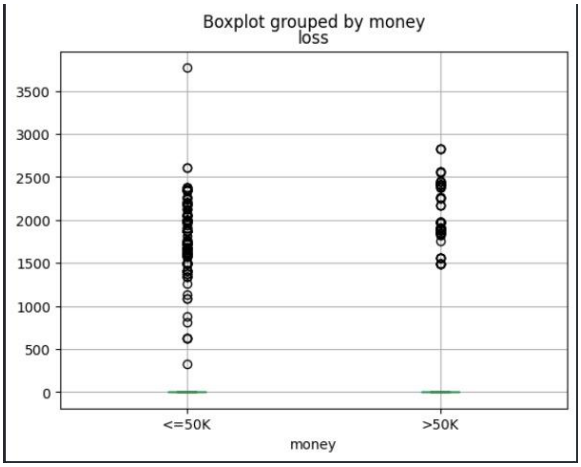



```
count   7591.00000
mean      36.85957
std       14.14134
min       17.00000
25%       25.00000
50%       34.00000
75%       46.00000
max       90.00000
Name: years, dtype: float64
```

```
count   2408.00000
mean      44.28115
std       10.59863
min       19.00000
25%       36.00000
50%       44.00000
75%       51.00000
max       90.00000
Name: years, dtype: float64
```

- loss (Pierderea de capital)

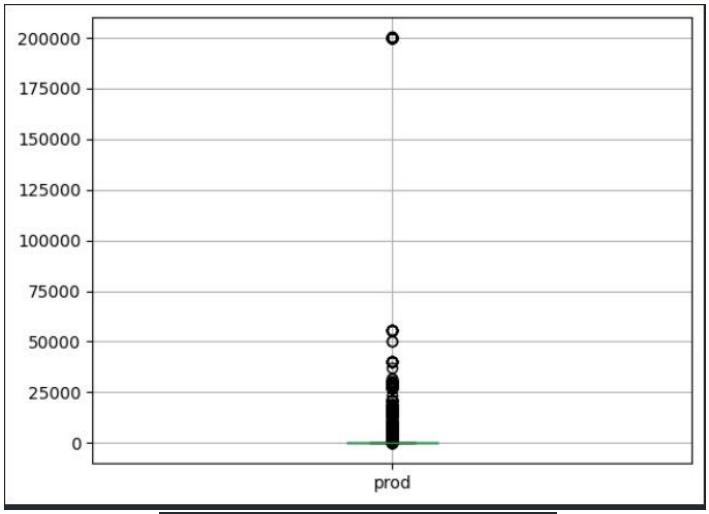


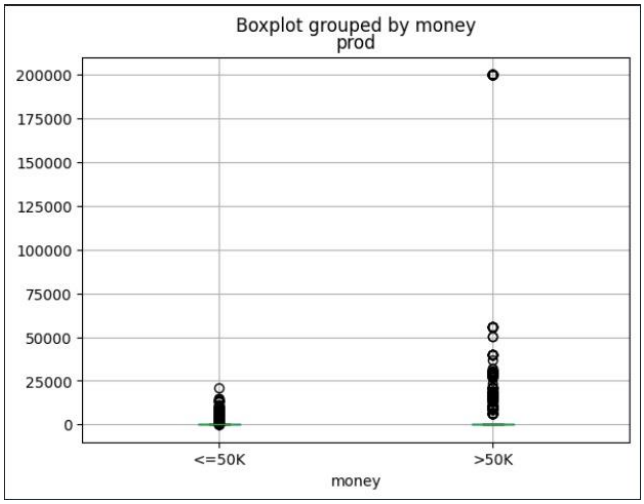



```
count   7591.00000
mean      56.66935
std      317.59182
min        0.00000
25%        0.00000
50%        0.00000
75%        0.00000
max     3770.00000
Name: loss, dtype: float64
```

```
count   2408.00000
mean     170.62002
std      563.03146
min        0.00000
25%        0.00000
50%        0.00000
75%        0.00000
max     2824.00000
Name: loss, dtype: float64
```

- prod (Producerea de capital)





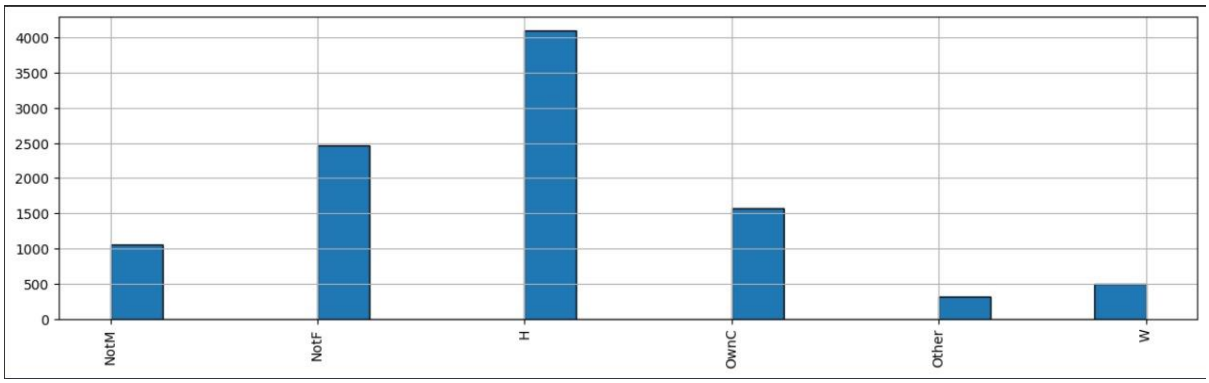

```
count    7591.00000
mean      299.23791
std      1348.93775
min       -28.00000
25%        37.00000
50%        57.00000
75%        72.00000
max     21179.00000
Name: prod, dtype: float64
```

```
count     2408.00000
mean      7423.48256
std      27761.98362
min        -23.00000
25%         42.00000
50%         62.00000
75%         97.00000
max     200125.00000
Name: prod, dtype: float64
```

Atribute discrete și ordinale:

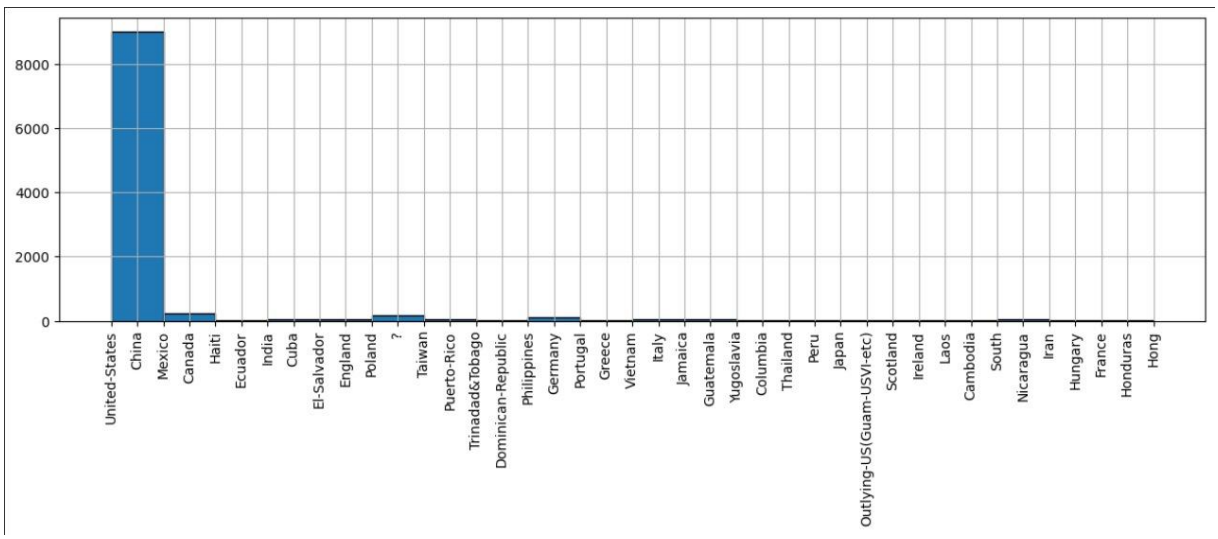
Am folosit funcția ***data_salary.describe(include=['O'])*** pentru a afișa un tabel cu coloanele care au valori categorice și nu numerice și numărul de valori unice din fiecare coloană, precum și cea mai frecventă valoare. În continuare, am folosit funcția ***data[column_name].unique()*** pentru a afișa o listă cu toate valorile unice din coloana respectivă și funcția ***data.value_counts(column_name)*** pentru a afișa frecvența de apariție a fiecărei valori. Aceste frecvențe au fost reprezentate cu ajutorul unei histograme.

- relation



['NotM' 'NotF' 'H' 'OwnC' 'Other' 'W']
relation
H 4097
NotF 2468
OwnC 1573
NotM 1054
W 491
Other 316
Name: count, dtype: int64

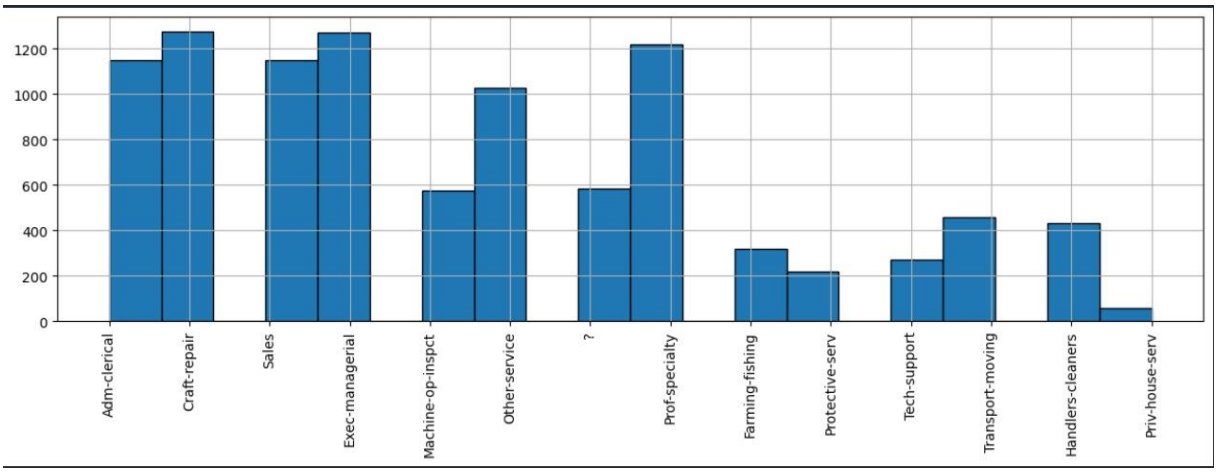
- country



['United-States' 'China' 'Mexico' 'Canada' 'Haiti' 'Ecuador' 'India'

'Cuba' 'El-Salvador' 'England' 'Poland' '?' 'Taiwan' 'Puerto-Rico'
'Trinadad&Tobago' 'Dominican-Republic' 'Philippines' 'Germany' 'Portugal'
'Greece' 'Vietnam' 'Italy' 'Jamaica' 'Guatemala' 'Yugoslavia' 'Columbia'
'Thailand' 'Peru' 'Japan' 'Outlying-US(Guam-USVI-etc)' 'Scotland'
'Ireland' 'Laos' 'Cambodia' 'South' 'Nicaragua' 'Iran' 'Hungary' 'France'
'Honduras' 'Hong']
country
United-States 8978
Mexico 193
? 158
Philippines 61
Germany 41
Canada 40
Cuba 34
Puerto-Rico 33
England 29
India 28
Vietnam 28
South 27
El-Salvador 26
Italy 24
...

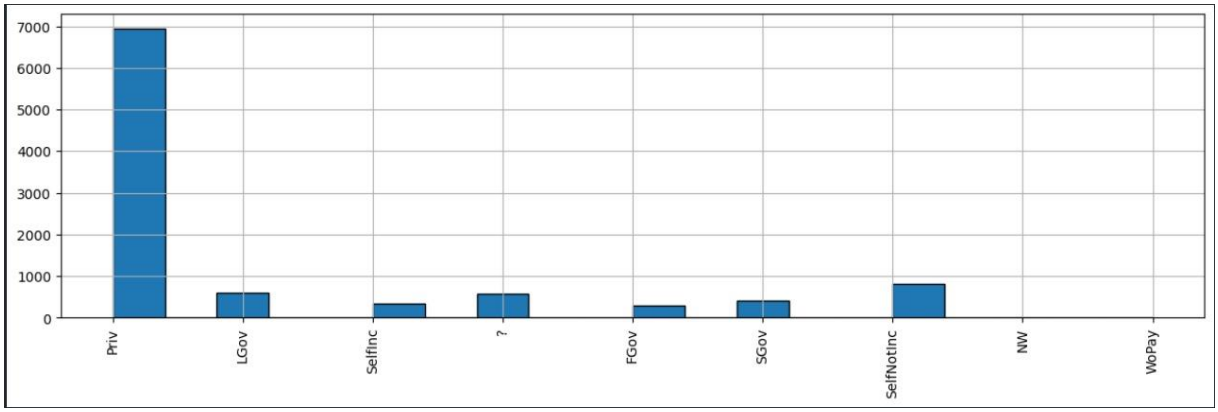
- Job



['Adm-clerical' 'Craft-repair' 'Sales' 'Exec-managerial'
'Machine-op-inspct' 'Other-service' '?' 'Prof-specialty'
'Farming-fishing' 'Protective-serv' 'Tech-support' 'Transport-moving'
'Handlers-cleaners' 'Priv-house-serv']
job
Craft-repair 1277
Exec-managerial 1271
Prof-specialty 1219
Sales 1150
Adm-clerical 1148
Other-service 1029
? 582
Machine-op-inspct 575

Transport-moving 456
Handlers-cleaners 432
Farming-fishing 316
Tech-support 270
Protective-serv 219
Priv-house-serv 55
Name: count, dtype: int64

- work_type



['Priv' 'LGov' 'SelfInc' '?' 'FGov' 'SGov' 'SelfNotInc' 'NW' 'WoPay']

work_type

Priv 6940

SelfNotInc 805

LGov 611

? 580

SGov 417

SelfInc 335

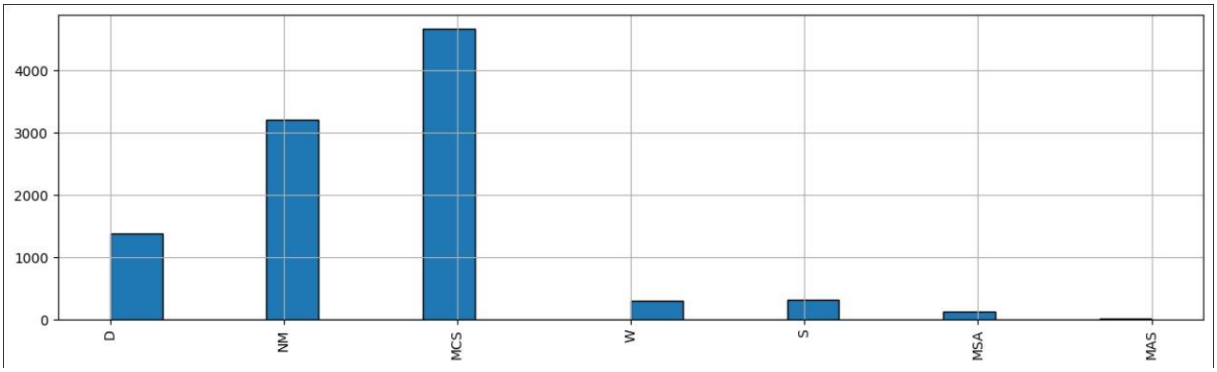
FGov 304

WoPay 5

NW 2

Name: count, dtype: int64

- partner



['D' 'NM' 'MCS' 'W' 'S' 'MSA' 'MAS']

partner

MCS 4667

NM 3209

D 1378

S 314

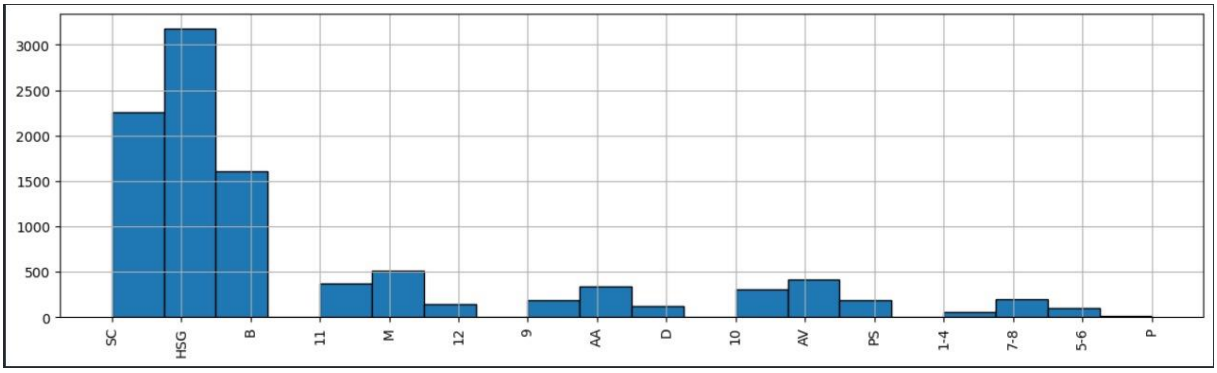
W 303

MSA 123

MAS 5

Name: count, dtype: int64

- edu



['SC' 'HSG' 'B' '11' 'M' '12' '9' 'AA' 'D' '10' 'AV' 'PS' '1-4' '7-8'
'5-6' 'P']

edu

HSG 3178

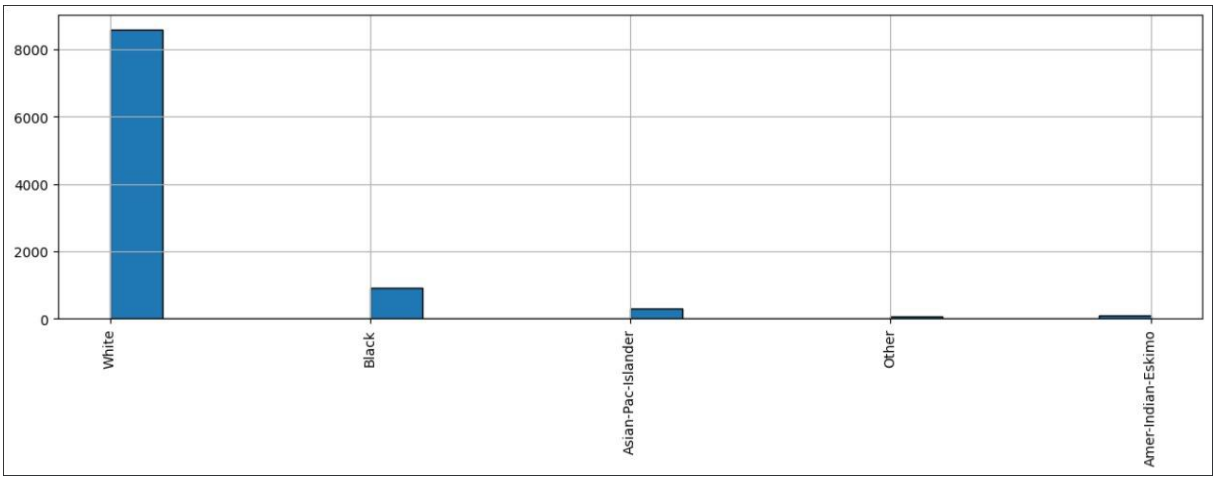
SC 2261

B 1613
M 509
AV 412
11 370
AA 341
10 306
7-8 196
PS 187
9 184
12 144
D 121
5-6 105
1-4 55
P 17
Name: count, dtype: int64

- gender

[nan 'M' 'F']
gender
M 6179
F 3020
Name: count, dtype: int64

- race



['White' 'Black' 'Asian-Pac-Islander' 'Other' 'Amer-Indian-Eskimo']

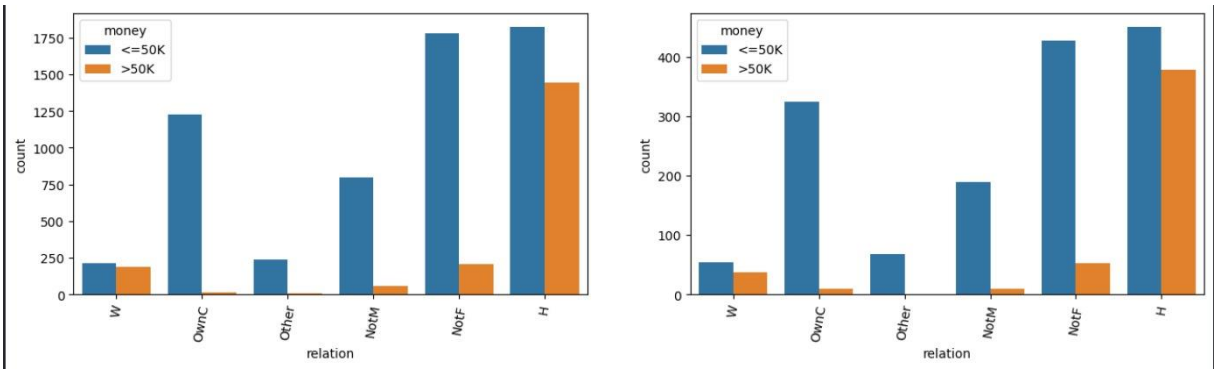
race
White 8588
Black 924
Asian-Pac-Islander 310
Amer-Indian-Eskimo 100
Other 77
Name: count, dtype: int64

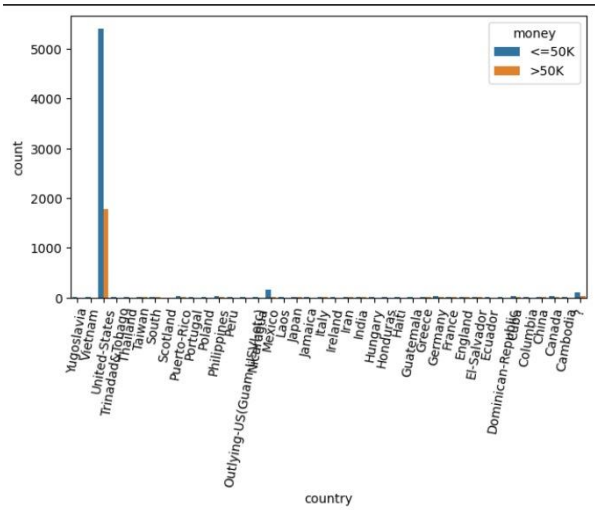
- gtype (tipul contractului de muncă)

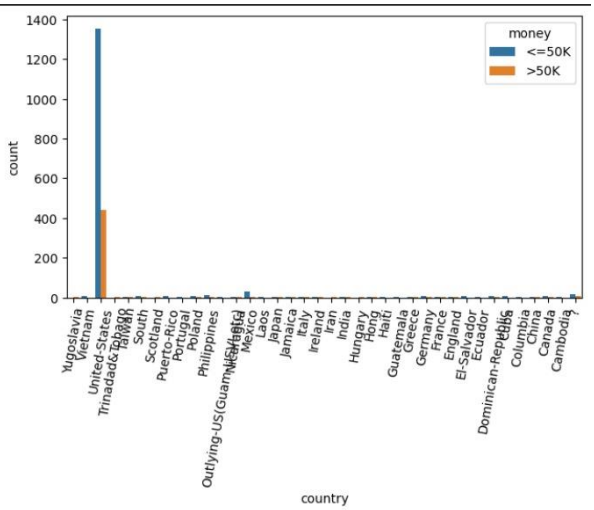
['DC' 'AC']
gtype
AC 6711
DC 3288
Name: count, dtype: int64

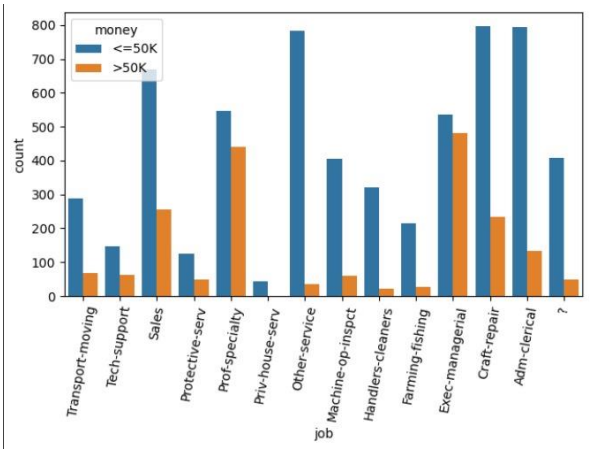
Analiza echilibrului de clase

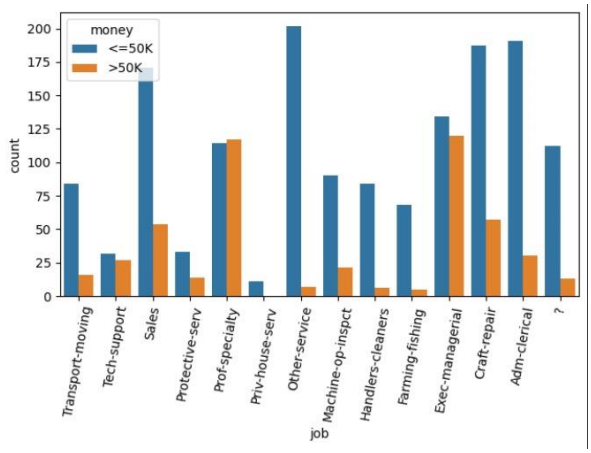
Pentru fiecare coloană am afișat două grafice (*boxplot* pentru numerice și *countplot* pentru categorice). Graficul din stânga este din setul de date de train, iar graficul din dreapta este din setul de date de test. Per total, se poate observa că sunt mai multe exemple de *money* = <=50K decât *money* = >50K, dar la nivelul *split-ului* pe fiecare atribut se poate observa că valorile sunt destul de echilibrate.

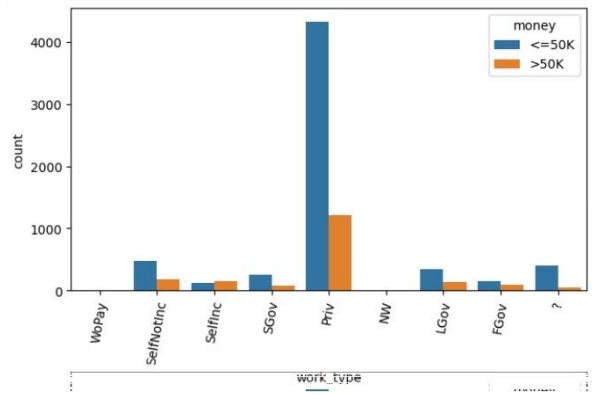


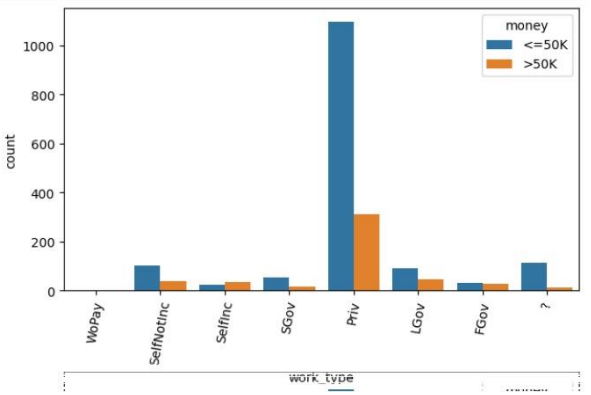

money
<=50K
>50K
count
country

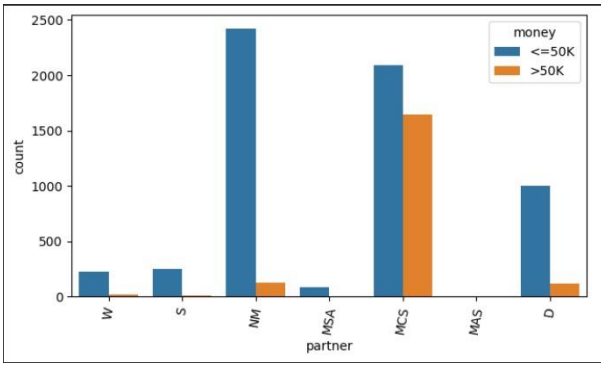

money
<=50K
>50K
count
country

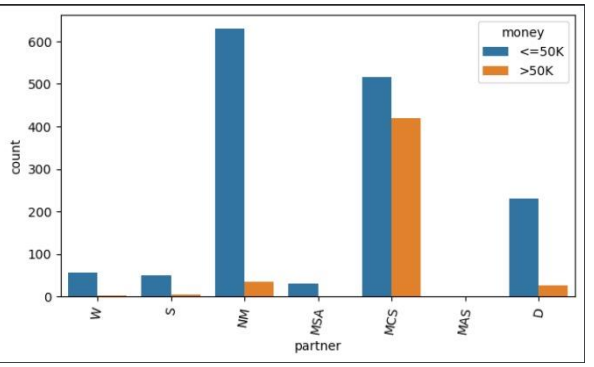

money
<=50K
>50K
count
job
Transport-moving
Tech-support
Sales
Protective-serv
Prof-specialty
Priv-house-serv
Other-service
Machine-op-inspct
Handlers-cleaners
Farming-fishing
Exec-managerial
Craft-repair
Adm-clerical
?

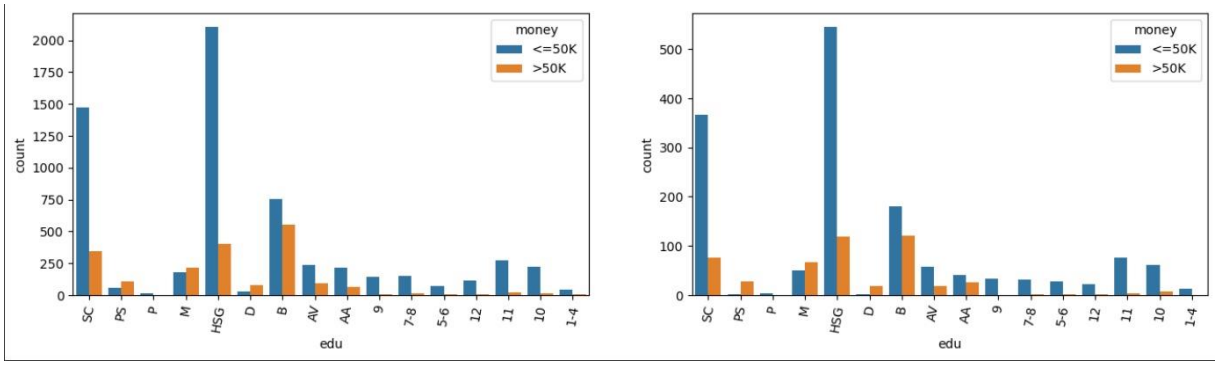

money
<=50K
>50K
count
job
Transport-moving
Tech-support
Sales
Protective-serv
Prof-specialty
Priv-house-serv
Other-service
Machine-op-inspct
Handlers-cleaners
Farming-fishing
Exec-managerial
Craft-repair
Adm-clerical
?

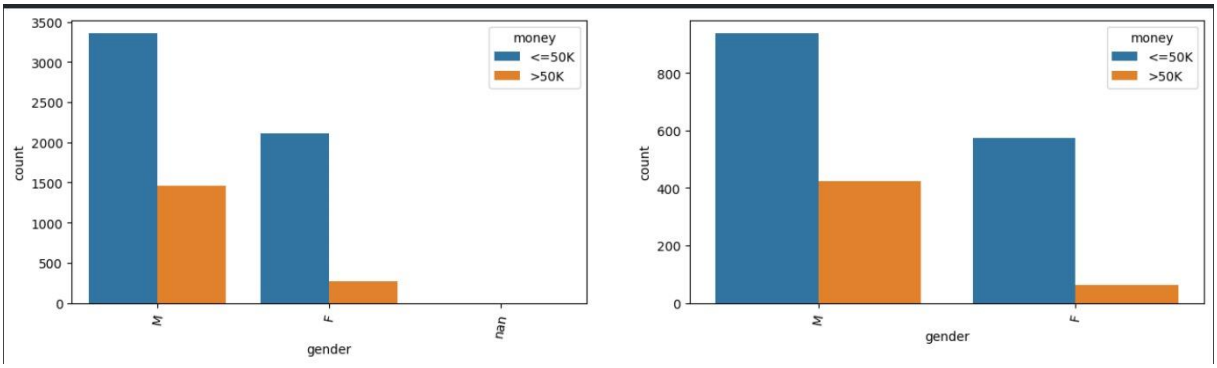

money
<=50K
>50K
count
WoPay
SelfNotInc
SelfInc
SGov
Priv
NW
LGov
FGov
?

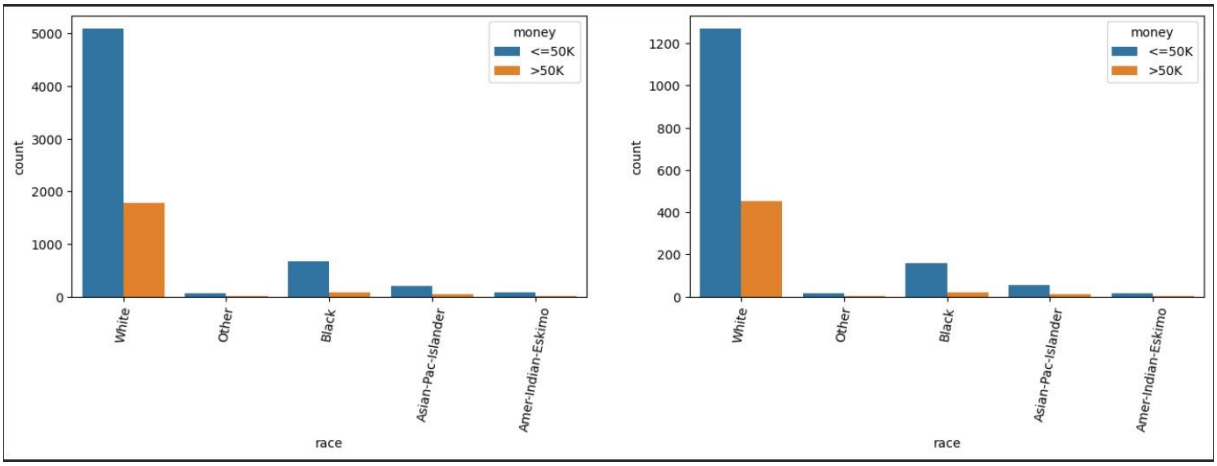

money
<=50K
>50K
count
WoPay
SelfNotInc
SelfInc
SGov
Priv
LGov
FGov
?

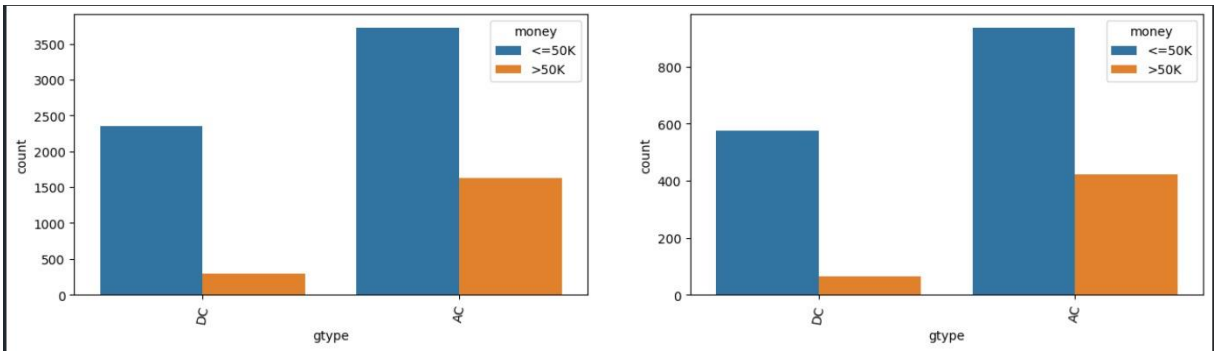

money
<=50K
>50K
count
partner
W
S
NM
MSA
MCS
MAS
D

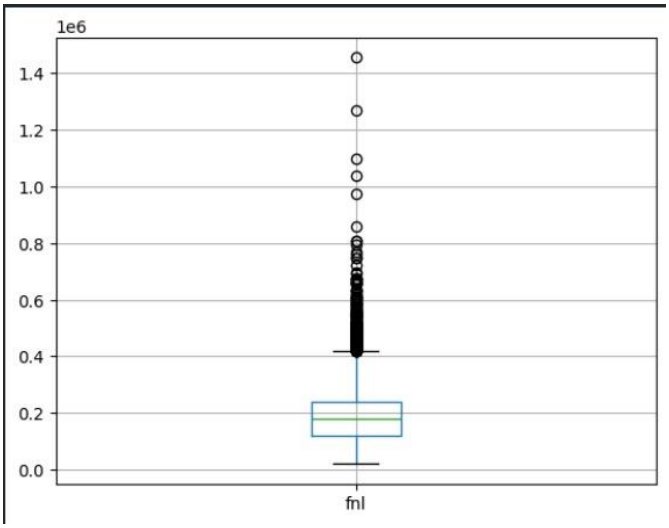

money
<=50K
>50K
count
partner
W
S
NM
MSA
MCS
MAS
D

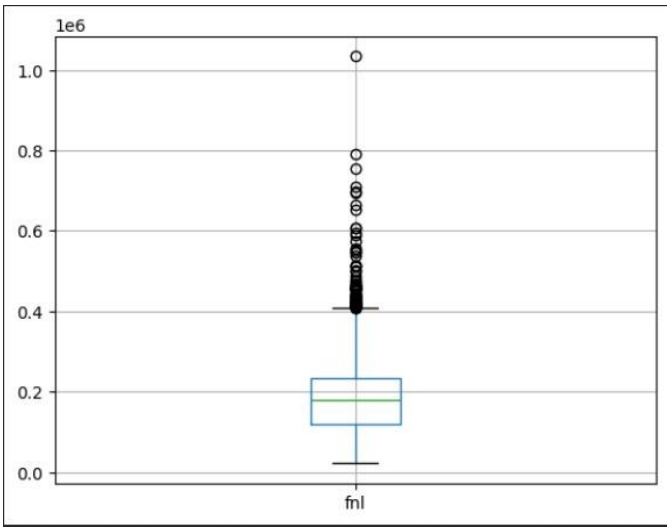

money
<=50K
>50K
count
edu
SC
PS
P
M
HSG
D
B
AV
AA
9
7-8
5-6
12
11
10
1-4

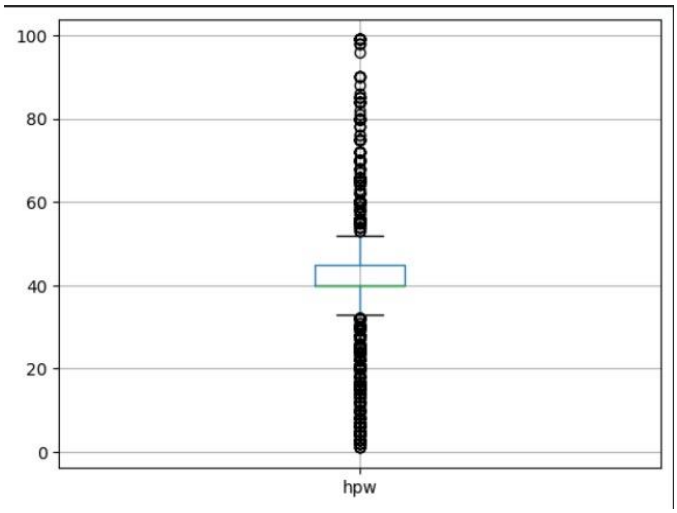

money
<=50K
>50K
count
gender
M
F
nan

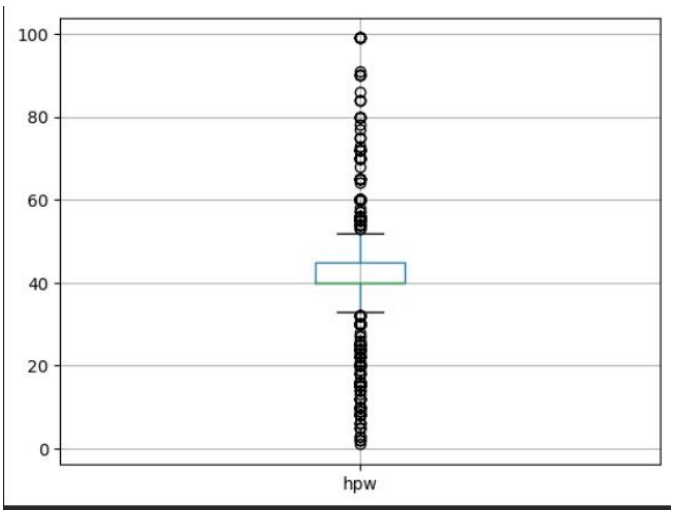

money
<=50K
>50K
count
race
White
Other
Black
Asian-Pac-Islander
Amer-Indian-Eskimo

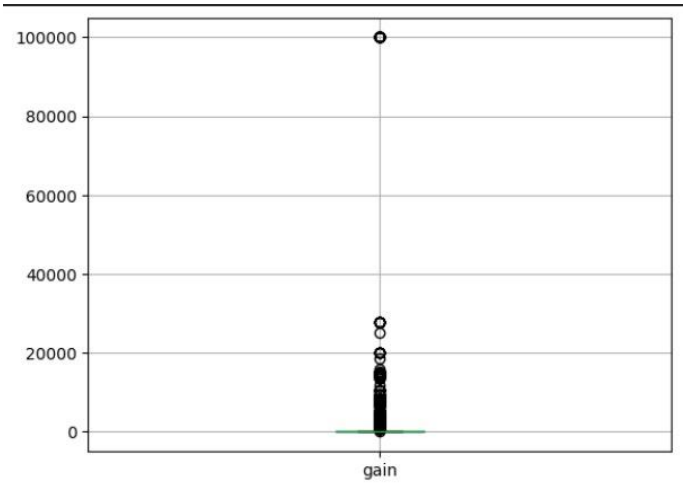

money
<=50K
>50K
count
gtype
DC
AC

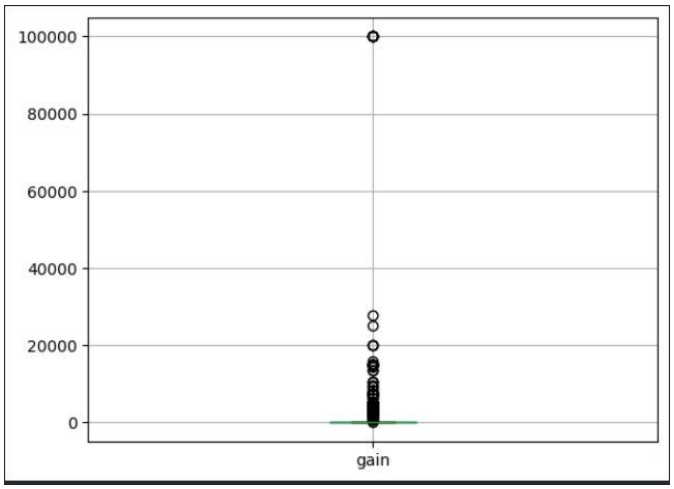

1e6
1.4
1.2
1.0
0.8
0.6
0.4
0.2
0.0
fnl

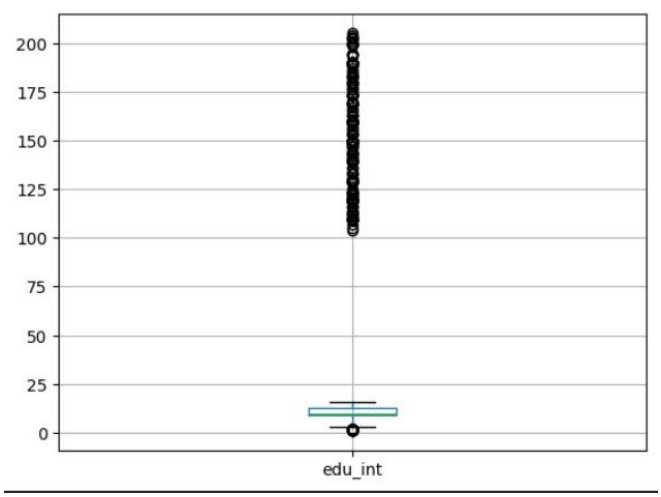

1e6
1.0
0.8
0.6
0.4
0.2
0.0
fnl

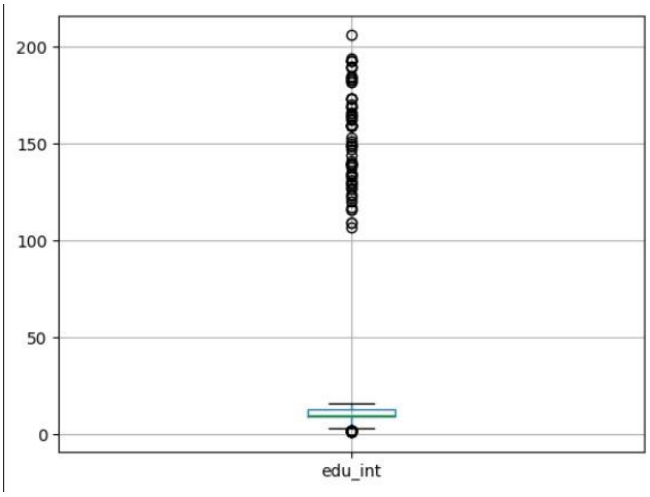

100
80
60
40
20
0
hpw

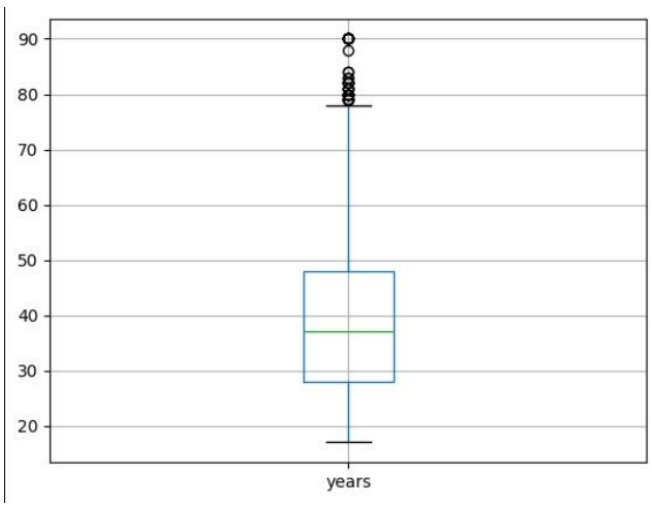

100
80
60
40
20
0
hpw

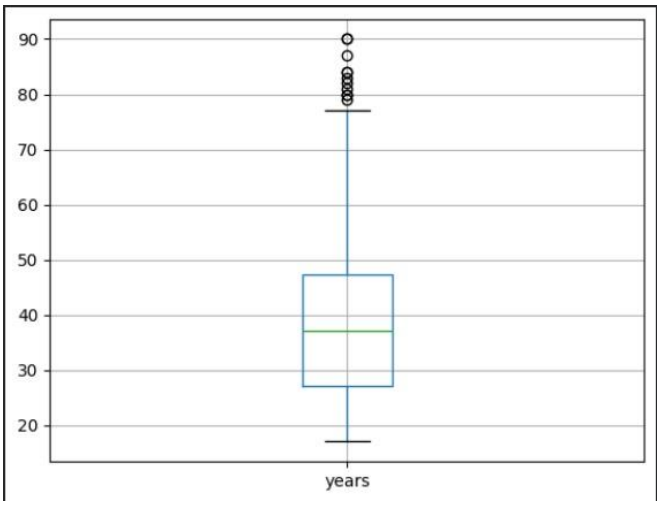

100000
80000
60000
40000
20000
0
gain

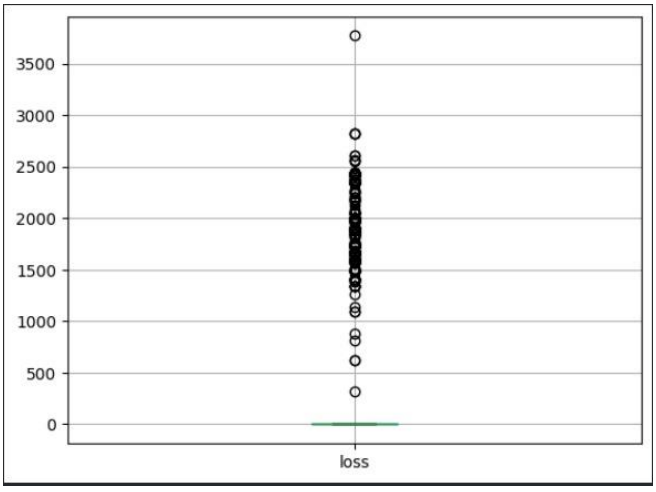

100000
80000
60000
40000
20000
0
gain

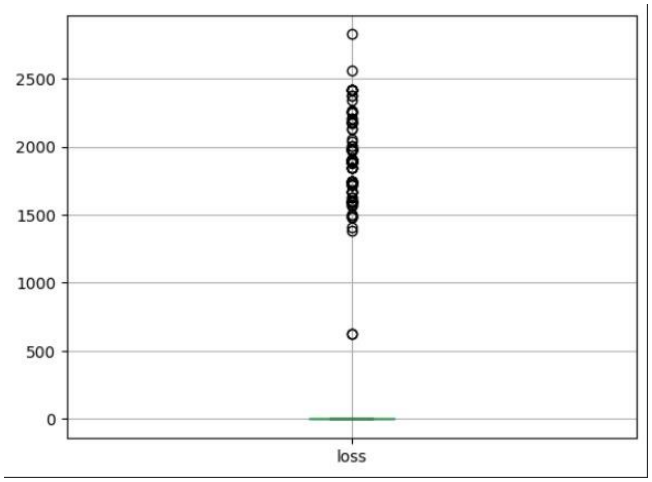

200
175
150
125
100
75
50
25
0
edu_int

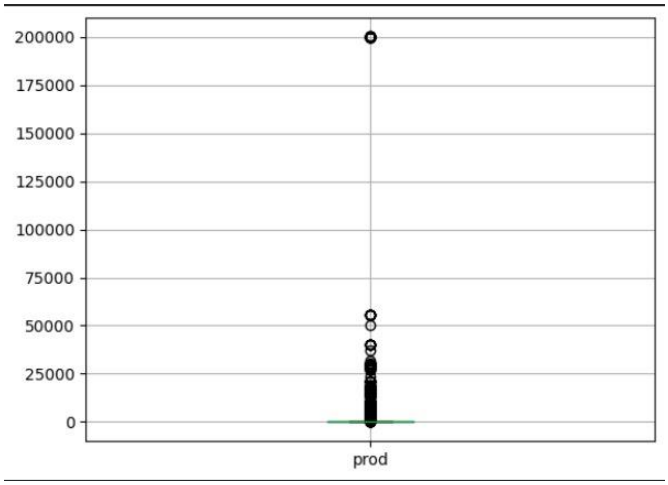

200
150
100
50
0
edu_int

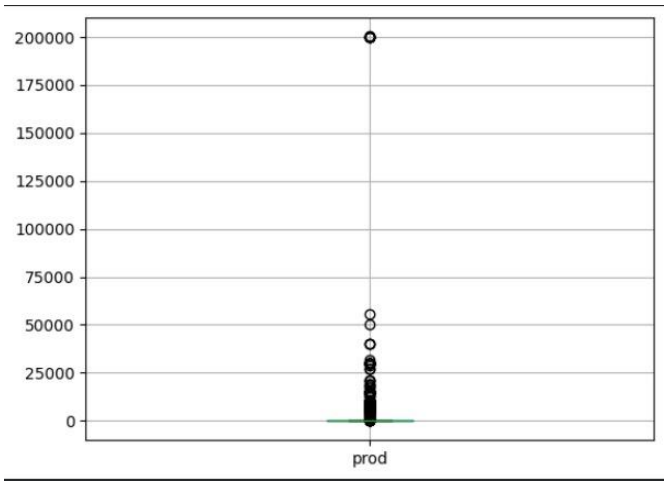

90
80
70
60
50
40
30
20
years


90
80
70
60
50
40
30
20
years


3500
3000
2500
2000
1500
1000
500
0
loss


2500
2000
1500
1000
500
0
loss



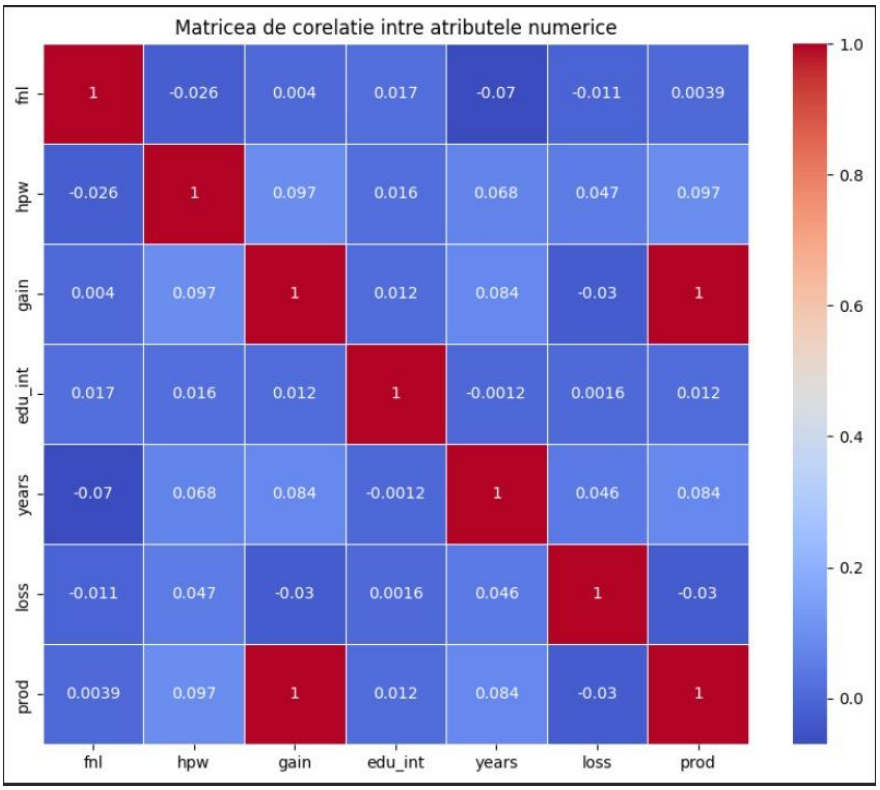


Corelația dintre atribute

Corelația dintre atribute ne ajută să observăm dacă avem atribute care sunt nefolositoare. Pentru datele numerice se poate folosi funcția din *pandas corr()* care poate fi afișată cu ajutorul unui *heatmap* din *seaborn.* Pentru cele categorice se poate folosi testul Chi-Pătrat, așa cum este specificat în enunțul temei.

Rezultatul testului Chi-Pătrat arată că aproape toate atributele categorice sunt corelate între ele, însă am decis să nu renunț la niciunul, fapt care poate a influențat rezultat algoritmilor ML.

Corelația dintre atributele numerice:



Corelația dintre atributele categorice:

	relation	country	job	work_type	partner	edu	gender	race	gtype
relation	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated
country	correlated	correlated	correlated	not-correlated	correlated	correlated	correlated	correlated	correlated
job	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated
work_type	correlated	not-correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated
partner	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated
edu	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated
gender	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated
race	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated
gtype	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated	correlated

Analiza setului de date Credit_Risk

Setul de date are următoarele coloane, din care coloana *loan_approval_status* este rezultatul dorit:

	residential_status	loan_rate	loan_amount	loan_purpose	loan_approval_status	job_tenure_years	credit_history_length_years	applicant_age	applicant_income	loan_rating	credit_history_default_status	loan_income_ratio	stability_rating	credit_history_length_months
0	Renter	9.99000	5600	Study	Approved	0.00000	4	25	36276	Very Good	No	0.15000	C	59
1	Mortgage	16.77000	12000	Business	Approved	5.00000	2	24	64000	Fair	No	0.19000	B	34
2	Renter	10.75000	5000	Business	Declined	NaN	4	21	12360	Very Good	No	0.40000	C	54
3	Mortgage	7.51000	13200	Personal	Approved	6.00000	12	36	83625	Excellent	No	0.16000	B	151
4	Mortgage	12.99000	3500	Health	Approved	5.00000	2	23	24091	Good	Yes	0.15000	B	31
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
9995	Renter	17.04000	7500	Business	Approved	7.00000	2	22	48000	Poor	Yes	0.16000	C	27
9996	Owner	10.59000	2400	Business	Approved	0.00000	3	22	19200	Very Good	No	0.13000	A	42
9997	Renter	NaN	6000	Home Improvement	Approved	5.00000	3	25	82000	Good	No	0.07000	C	44
9998	Renter	12.53000	1600	Personal	Approved	3.00000	3	24	25000	Good	No	0.06000	C	40
9999	Renter	6.62000	10000	Study	Approved	7.00000	4	23	45000	Excellent	No	0.22000	C	52

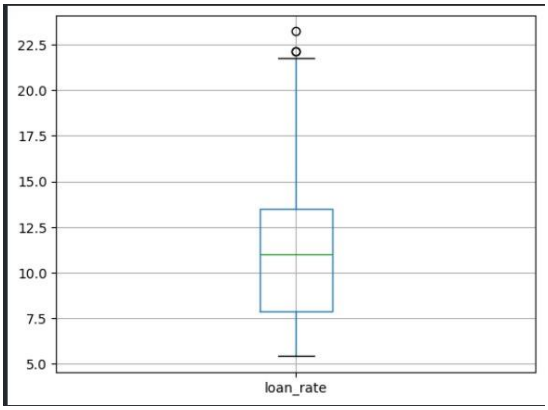
Date numerice – am folosit funcția describe din pandas pentru a afișa anumite valori relevante pentru fiecare atribut în parte:

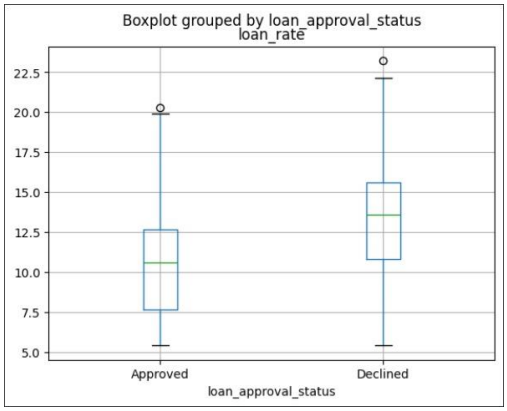
	loan_rate	loan_amount	job_tenure_years	credit_history_length_years	applicant_age	applicant_income	loan_income_ratio	credit_history_length_months
count	9060.00000	10000.00000	9736.00000	10000.00000	10000.00000	10000.00000	10000.00000	10000.00000
mean	11.00718	9568.03750	4.78574	5.81110	27.74510	65734.21130	0.17013	75.76070
std	3.26639	6350.43158	4.35312	4.05022	6.36015	56944.38708	0.10681	48.67736
min	5.42000	500.00000	0.00000	2.00000	20.00000	4200.00000	0.00000	25.00000
25%	7.90000	5000.00000	2.00000	3.00000	23.00000	38595.00000	0.09000	41.00000
50%	10.99000	8000.00000	4.00000	4.00000	26.00000	55000.00000	0.15000	57.00000
75%	13.47000	12200.00000	7.00000	8.00000	30.00000	78997.00000	0.23000	102.00000
max	23.22000	35000.00000	123.00000	30.00000	123.00000	2039784.00000	0.76000	369.00000

Din acest tabel se poate observa că toate coloanele, mai puțin coloanele *loan_rate* și *job_tenure_years,* nu au date lipsă. Coloana *loan_rate* are câteva valori lipsă în acest set de date, existând doar 9060 valori în tabel, ceea ce înseamnă că lipsesc 940 de valori, care vor trebui adăugate în faza de preprocesare. La fel și în coloana *job_tenure_years*, în tabel sunt doar 9736 de valori, ceea ce înseamnă că lipsesc 264 de valori.

În continuare, am analizat datele din fiecare coloană separat, afișând pentru fiecare atribut: valoarea medie, deviația standard, valoarea minimă și maximă, valoarea percentilelor de 25%, 50% și 75%, și am afișat aceste valori cu ajutorul graficului de tip *boxplot*. Pentru fiecare coloană am afișat două *boxplot-uri*, unul cu toate valorile din coloana respectivă și unul cu valorile împărțite în funcție de atributul rezultat (*loan_approval_status*). Am mai afișat outputul funcției *describe* pentru fiecare atribut în parte, pentru valorile separate în funcție de atributul rezultat:

- loan_rate



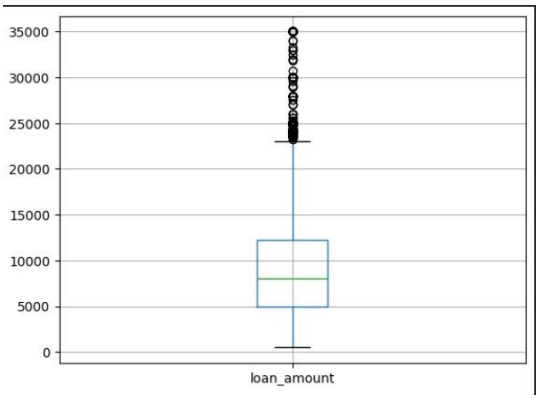


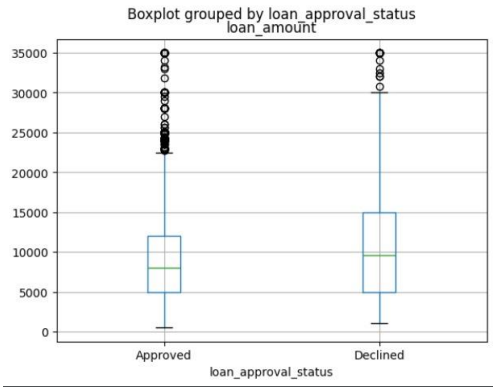

```
count   7047.00000
mean      10.39492
std        2.98781
min        5.42000
25%        7.66000
50%       10.59000
75%       12.69000
max       20.30000
Name: loan_rate, dtype: float64
```

```
count   2013.00000
mean      13.15056
std        3.29649
min        5.42000
25%       10.83000
50%       13.57000
75%       15.62000
max       23.22000
Name: loan_rate, dtype: float64
```

Valorile din stânga sunt pentru exemplele care au valoarea din coloana loan_approval_status = Approved, iar valorile din dreapta sunt pentru loan_approval_status = Declined, dar cum coloana loan_rate are valori lipsă, aceasta diferă puțin.

- loan_amount



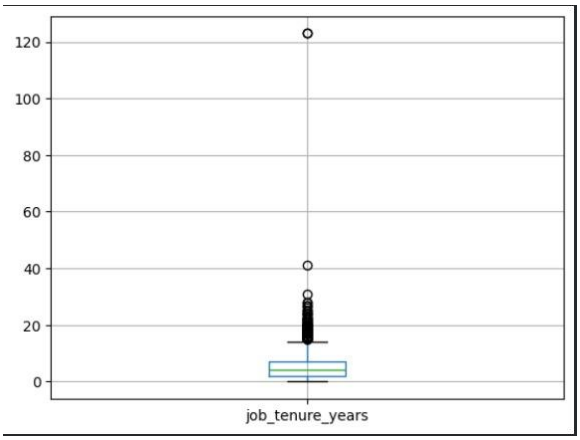


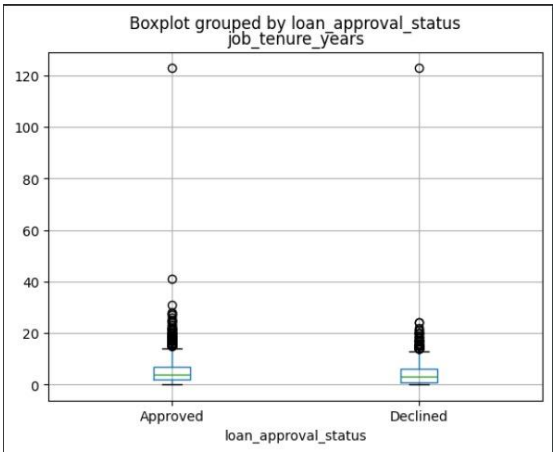

```
count    7818.00000
mean     9218.65886
std      6077.98429
min       500.00000
25%      5000.00000
50%      8000.00000
75%     12000.00000
max     35000.00000
Name: loan_amount, dtype: float64
```

```
count    2182.00000
mean    10819.84418
std      7104.74336
min      1000.00000
25%      5000.00000
50%      9600.00000
75%     15000.00000
max     35000.00000
Name: loan_amount, dtype: float64
```

Valorile din stânga sunt pentru exemplele care au valoarea din coloana loan_approval_status = Approved, iar valorile din dreapta sunt pentru loan_approval_status = Declined, iar acest lucru este valabil pentru toate coloanele în continuare

- job_tenure_years

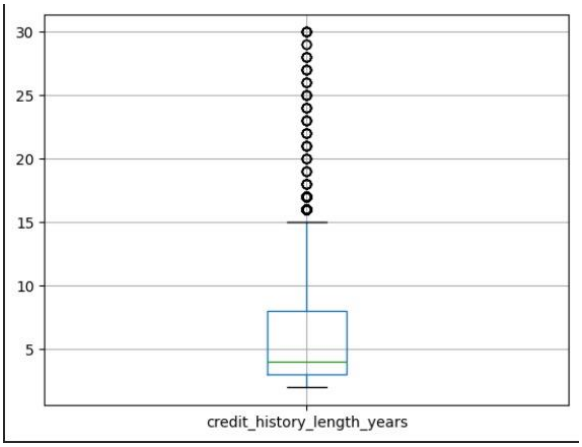


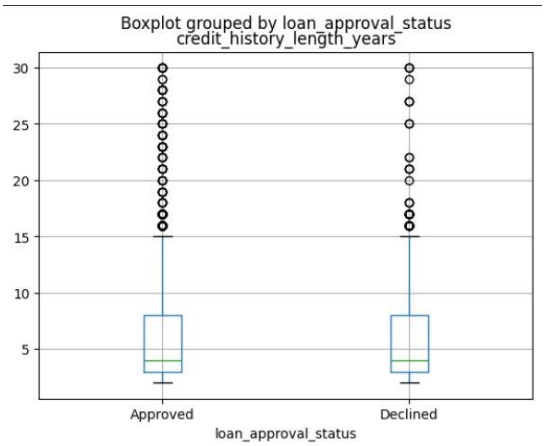



```
count   7643.00000
mean       4.96036
std        4.26361
min        0.00000
25%        2.00000
50%        4.00000
75%        7.00000
max      123.00000
Name: job_tenure_years, dtype: float64
```

```
count   2093.00000
mean       4.14811
std        4.61066
min        0.00000
25%        1.00000
50%        3.00000
75%        6.00000
max      123.00000
Name: job_tenure_years, dtype: float64
```

- credit_history_length_years

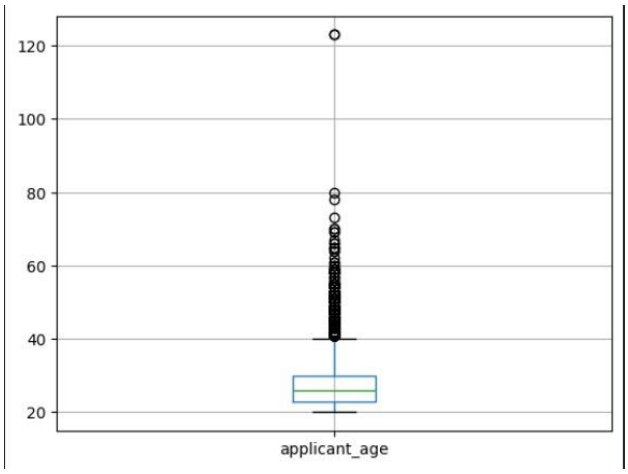


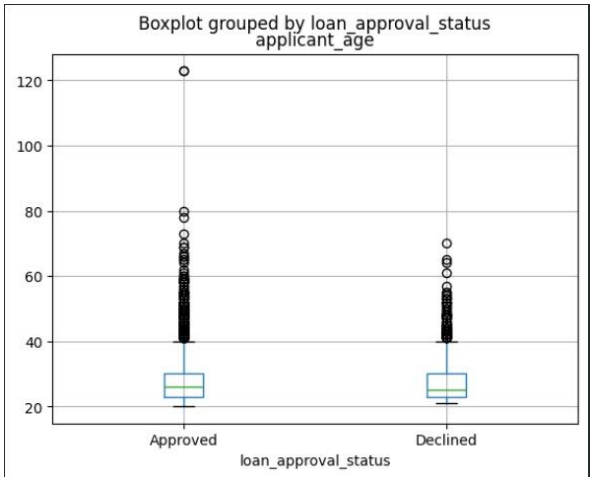



```
count   7818.00000
mean       5.86915
std        4.06738
min        2.00000
25%        3.00000
50%        4.00000
75%        8.00000
max       30.00000
Name: credit_history_length_years, dtype: float64
```

```
count   2182.00000
mean       5.60312
std        3.98210
min        2.00000
25%        3.00000
50%        4.00000
75%        8.00000
max       30.00000
Name: credit_history_length_years, dtype: float64
```

- applicant_age

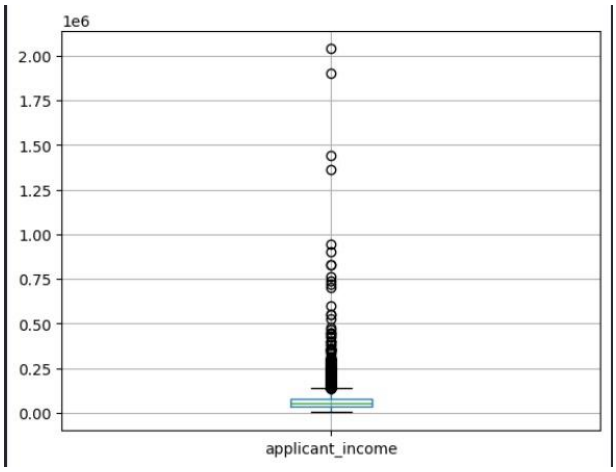


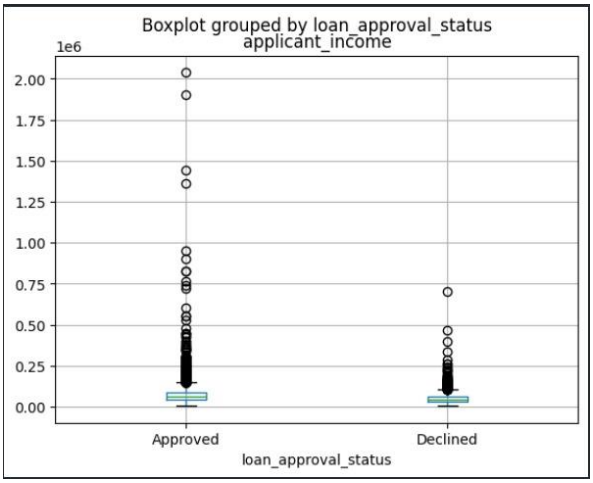



```
count   7818.00000
mean      27.84497
std        6.42734
min       20.00000
25%       23.00000
50%       26.00000
75%       30.00000
max      123.00000
Name: applicant_age, dtype: float64
```

```
count   2182.00000
mean      27.38726
std        6.10138
min       21.00000
25%       23.00000
50%       25.00000
75%       30.00000
max       70.00000
Name: applicant_age, dtype: float64
```

- applicant_income

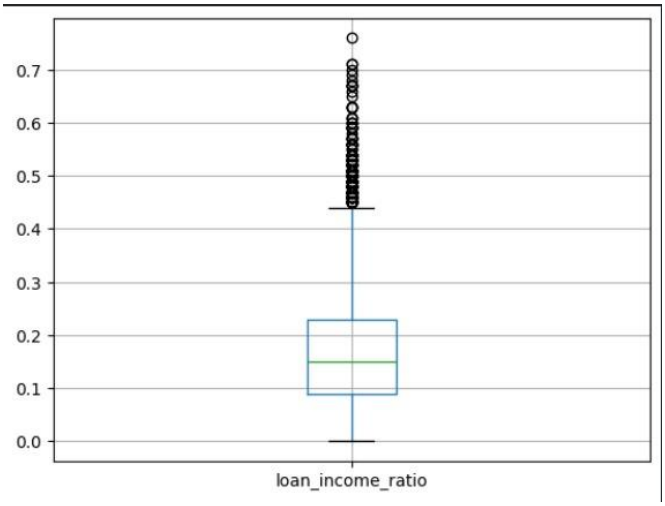


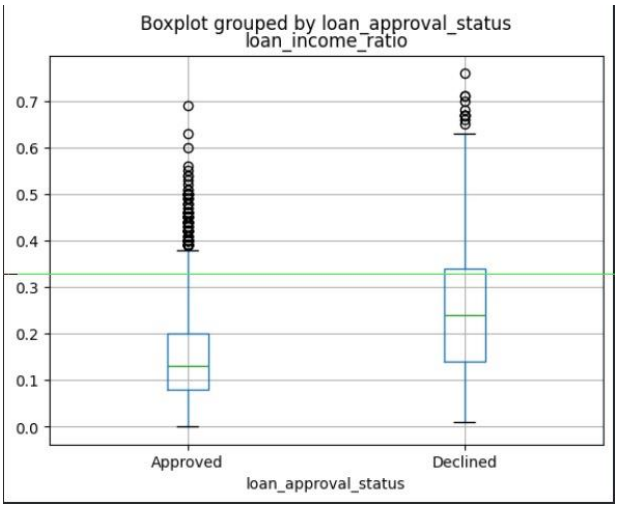



```
count     7818.00000
mean     70488.17997
std      60750.57851
min       7904.00000
25%      42000.00000
50%      60000.00000
75%      84000.00000
max    2039784.00000
Name: applicant_income, dtype: float64
```

```
count     2182.00000
mean     48700.97250
std      35599.00843
min       4200.00000
25%      29479.00000
50%      42000.00000
75%      60000.00000
max     703800.00000
Name: applicant_income, dtype: float64
```

- loan_income_ratio

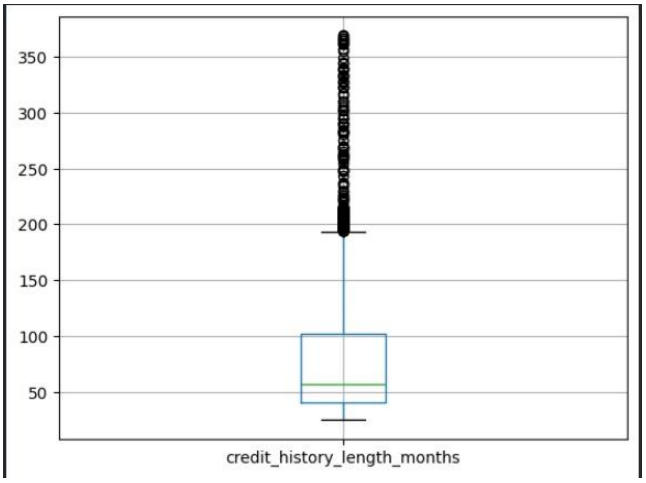


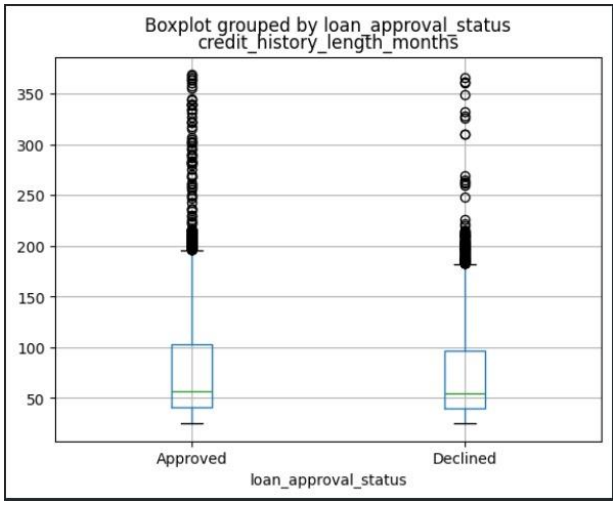



```
count   7818.00000
mean       0.14847
std        0.08661
min        0.00000
25%        0.08000
50%        0.13000
75%        0.20000
max        0.69000
Name: loan_income_ratio, dtype: float64
```

```
count   2182.00000
mean       0.24773
std        0.13310
min        0.01000
25%        0.14000
50%        0.24000
75%        0.34000
max        0.76000
Name: loan_income_ratio, dtype: float64
```

- credit_history_length_months





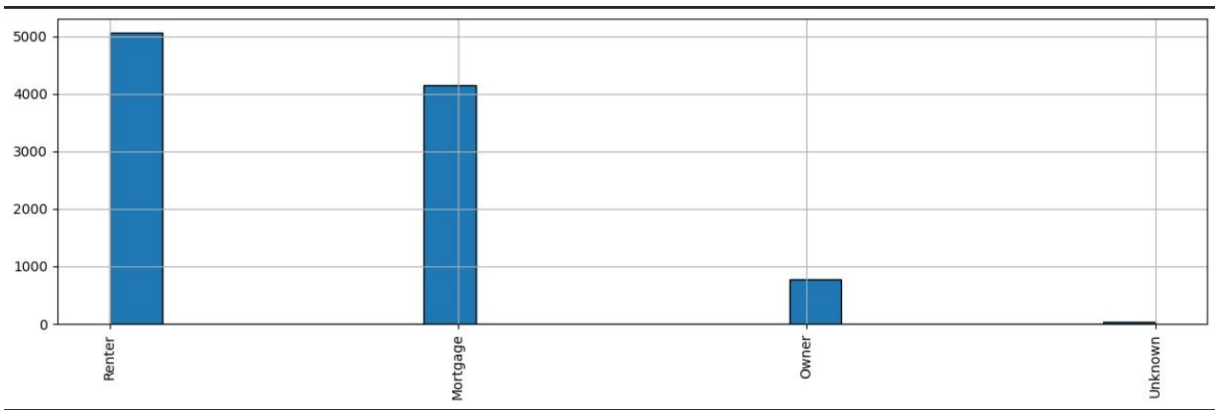

```
count   7818.00000
mean      76.48874
std       48.88821
min       25.00000
25%       41.00000
50%       57.00000
75%      103.00000
max      369.00000
Name: credit_history_length_months, dtype: float64
```

```
count   2182.00000
mean      73.15215
std       47.83445
min       25.00000
25%       40.00000
50%       55.00000
75%       97.00000
max      366.00000
Name: credit_history_length_months, dtype: float64
```

Atribute discrete și ordinale:

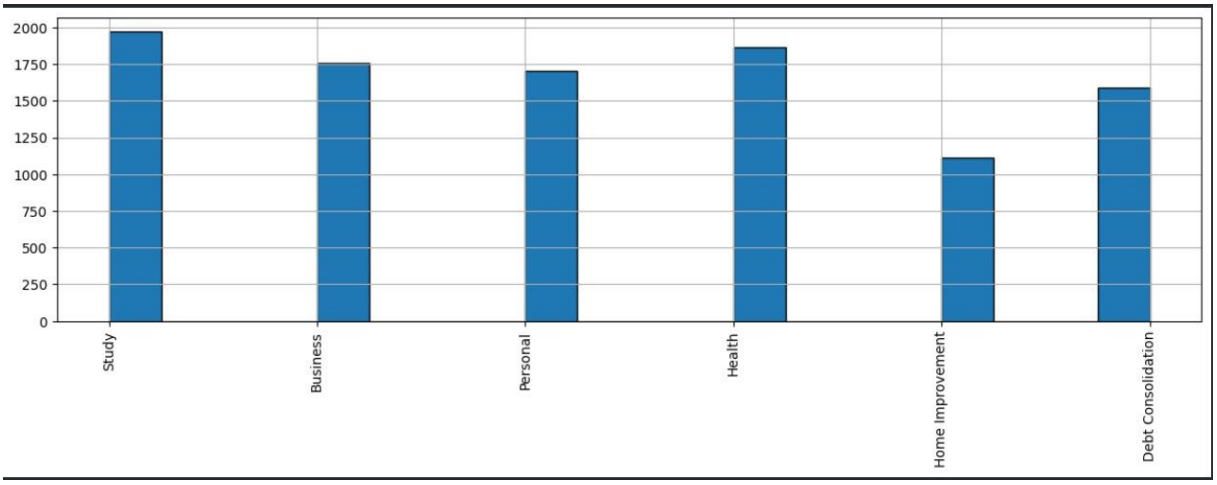
Am folosit funcția ***data_credit.describe(include=['O'])*** pentru a afișa un tabel cu coloanele care au valori categorice și nu numerice și numărul de valori unice din fiecare coloană, precum și cea mai frecventă valoare. În continuare, am folosit funcția ***data[column_name].unique()*** pentru a afișa o listă cu toate valorile unice din coloana respectivă și funcția ***data.value_counts(column_name)*** pentru a afișa frecvența de apariție a fiecărei valori. Aceste frecvențe au fost reprezentate cu ajutorul unei histograme.

- Residential_status



['Renter' 'Mortgage' 'Owner' 'Unknown']
residential_status
Renter 5056
Mortgage 4140
Owner 775
Unknown 29
Name: count, dtype: int64

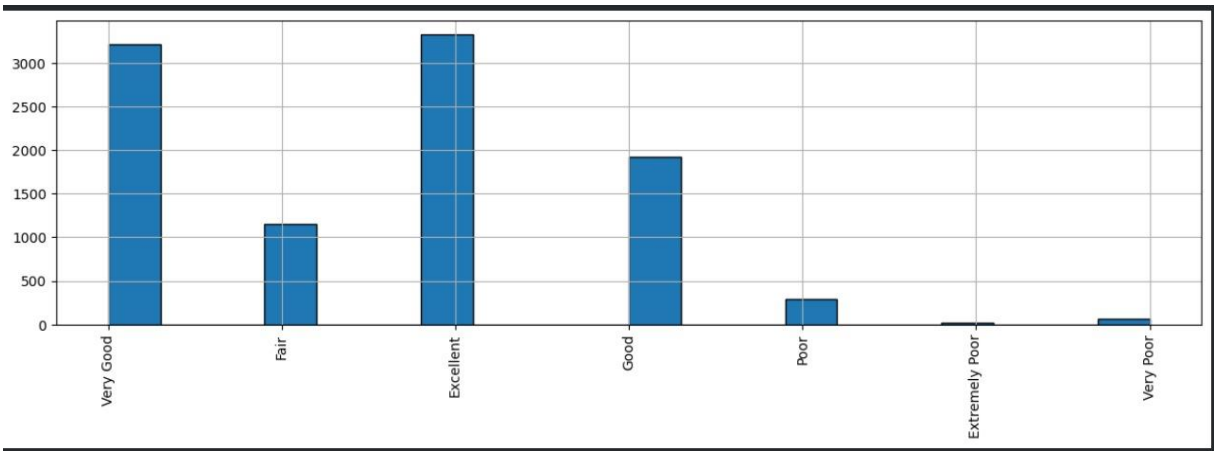
- Loan_purpose



['Study' 'Business' 'Personal' 'Health' 'Home Improvement'
'Debt Consolidation']
loan_purpose

Study 1971
Health 1865
Business 1755
Personal 1705
Debt Consolidation 1590
Home Improvement 1114
Name: count, dtype: int64

- Loan_rating



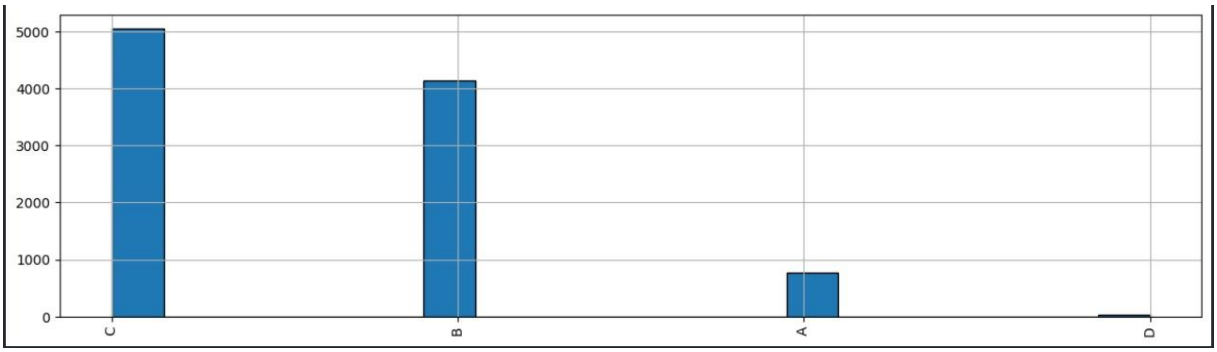
['Very Good' 'Fair' 'Excellent' 'Good' 'Poor' 'Extremely Poor' 'Very Poor']
loan_rating
Excellent 3325
Very Good 3216
Good 1925
Fair 1151
Poor 296
Very Poor 63
Extremely Poor 24
Name: count, dtype: int64

- Credit_history_default_status

['No' 'Yes']
credit_history_default_status
No 8264

Yes 1736
Name: count, dtype: int64

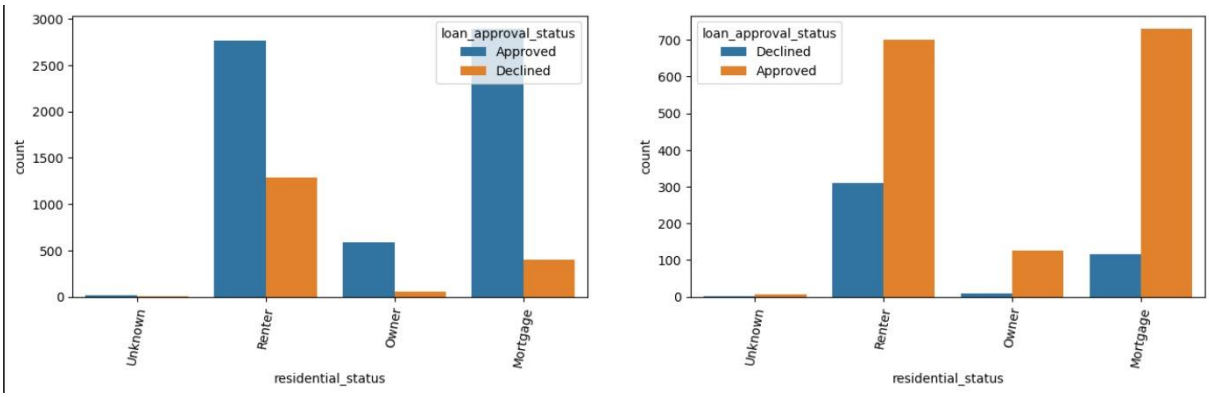
- Stability_rating

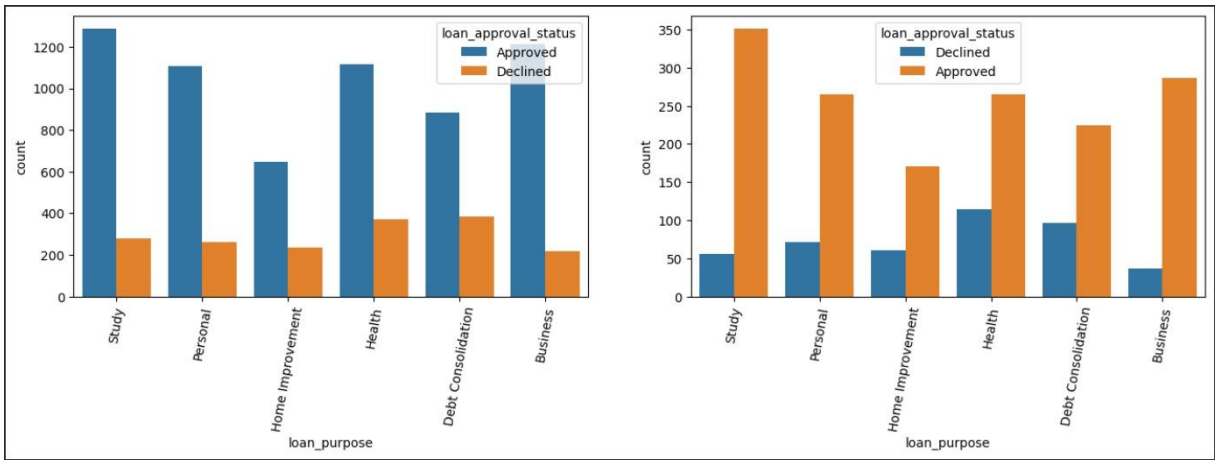


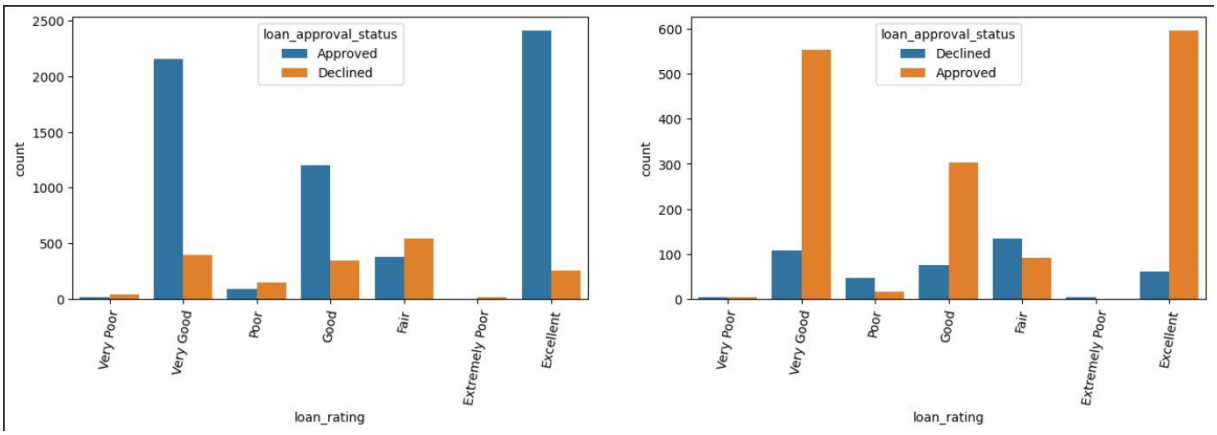
['C' 'B' 'A' 'D']
stability_rating
C 5056
B 4140
A 775
D 29
Name: count, dtype: int64

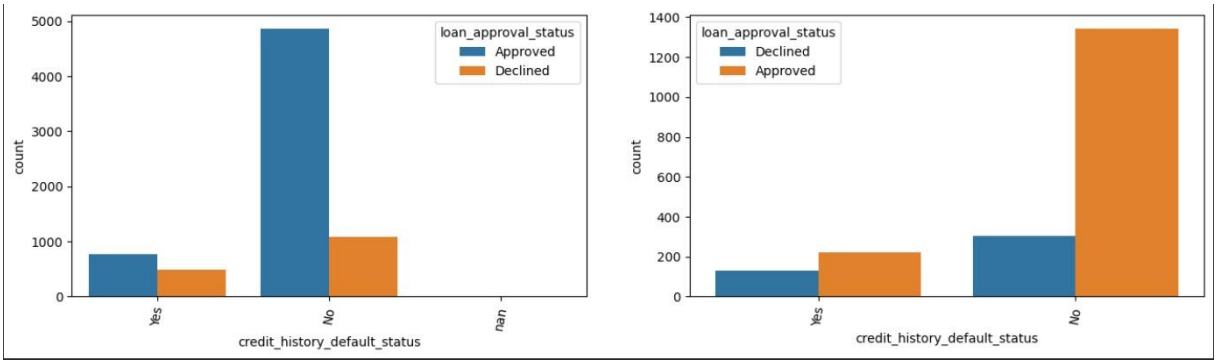
Analiza echilibrului de date

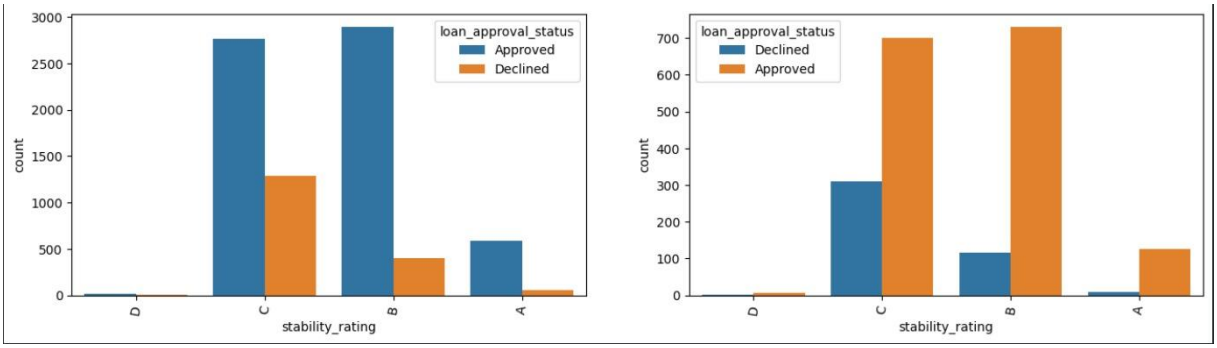
Pentru fiecare coloană am afișat două grafice (*boxplot* pentru numerice și *countplot* pentru categorice). Graficul din stânga este din setul de date de *train*, iar graficul din dreapta este din setul de date de *test*. Per total, se poate observa că sunt mai multe exemple de *loan_approval_status = Approved* decât *loan_approval_status = Declined*, dar la nivelul *split-ului* pe fiecare atribut se poate observa că valorile sunt destul de echilibrate. Se poate observa și o eroare în afișare deoarece culorile sunt inversate.

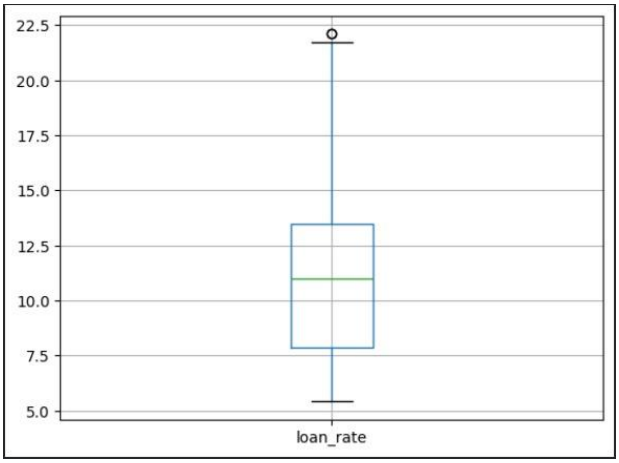


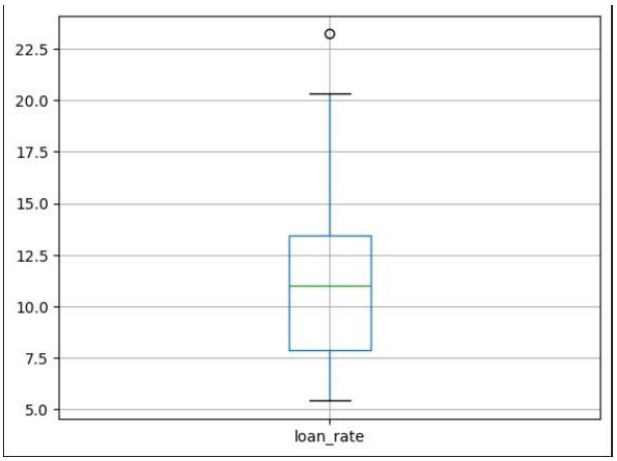

loan_approval_status
Approved
Declined
count
Study
Personal
Home Improvement
Health
Debt Consolidation
Business
loan_purpose
loan_approval_status
Declined
Approved
count
Study
Personal
Home Improvement
Health
Debt Consolidation
Business
loan_purpose

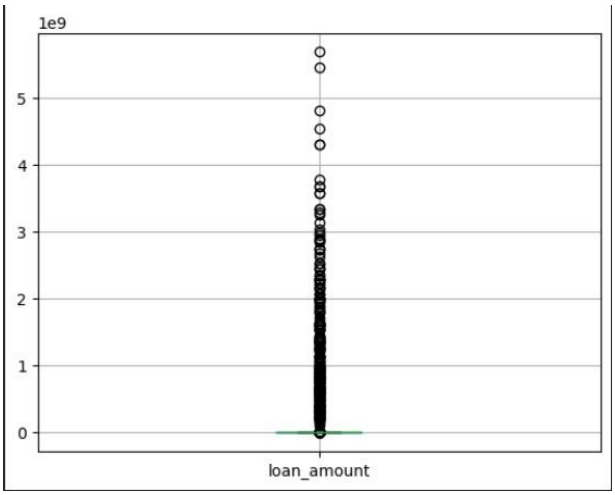

loan_approval_status
Approved
Declined
count
Very Poor
Very Good
Poor
Good
Fair
Extremely Poor
Excellent
loan_rating
loan_approval_status
Declined
Approved
count
Very Poor
Very Good
Poor
Good
Fair
Extremely Poor
Excellent
loan_rating

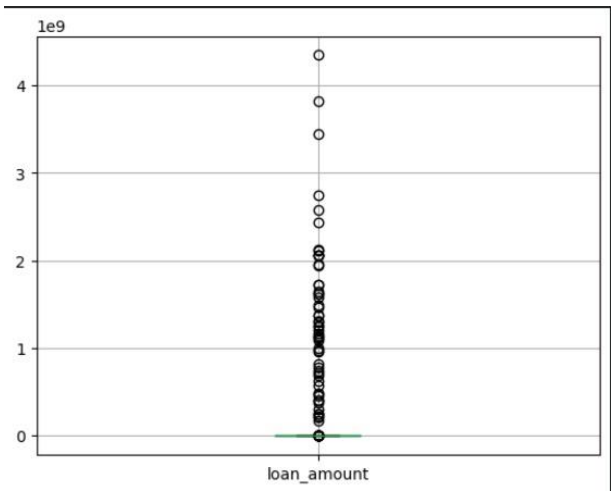

loan_approval_status
Approved
Declined
count
Yes
No
nan
credit_history_default_status
loan_approval_status
Declined
Approved
count
Yes
No
credit_history_default_status

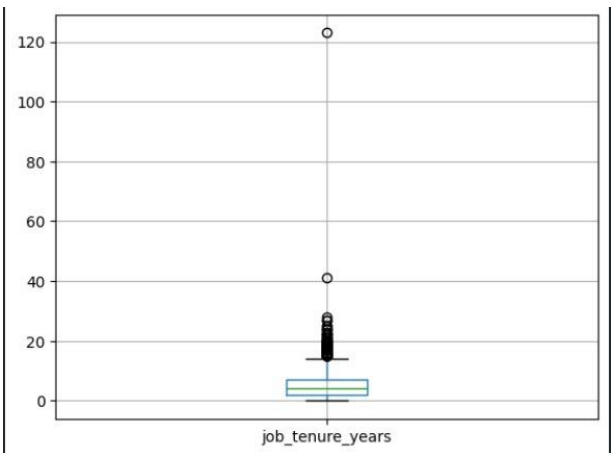

loan_approval_status
Approved
Declined
count
D
C
B
A
stability_rating
loan_approval_status
Declined
Approved
count
D
C
B
A
stability_rating

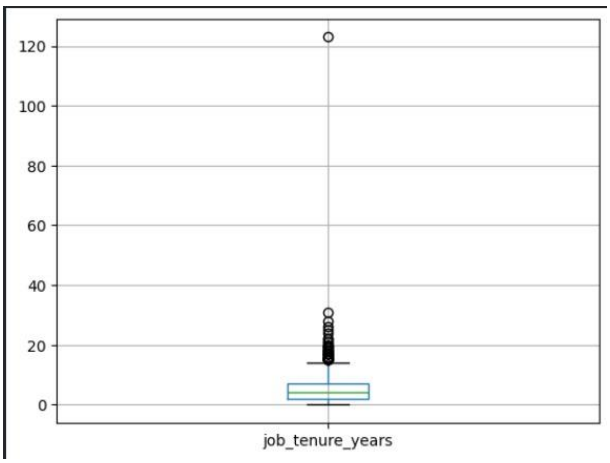

22.5
20.0
17.5
15.0
12.5
10.0
7.5
5.0
loan_rate

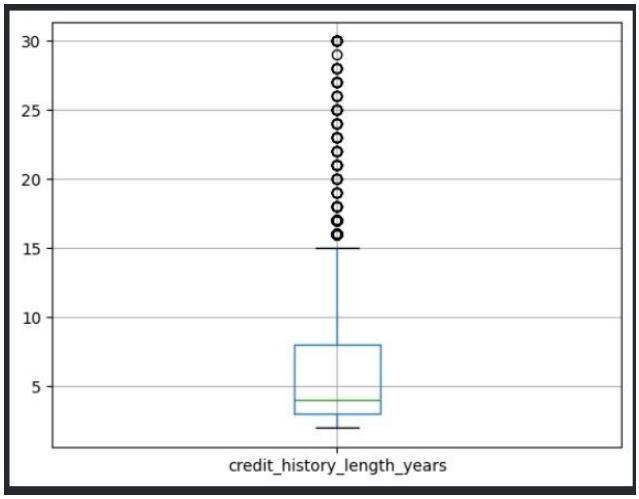

22.5
20.0
17.5
15.0
12.5
10.0
7.5
5.0
loan_rate

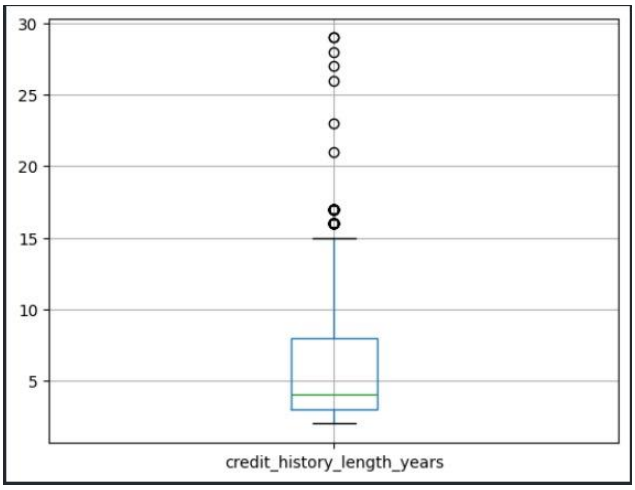

1e9
5
4
3
2
1
0
loan_amount

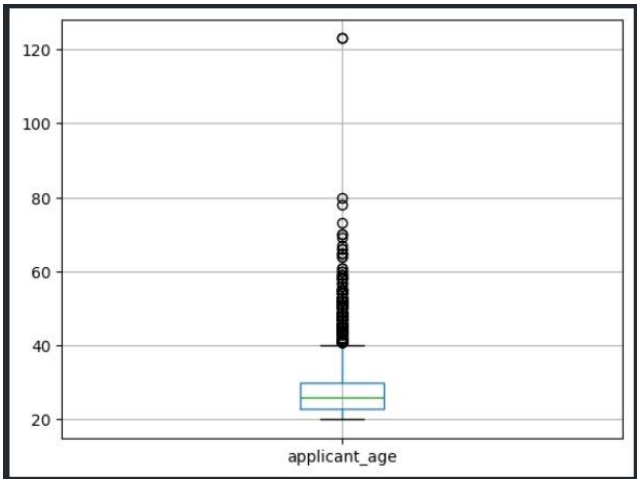

1e9
4
3
2
1
0
loan_amount

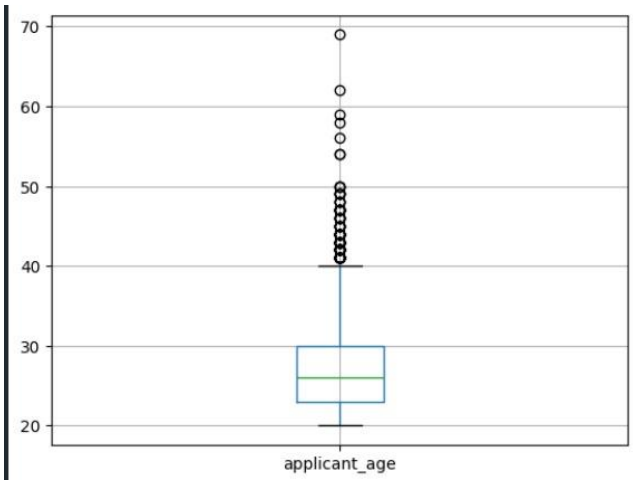

120
100
80
60
40
20
0
job_tenure_years

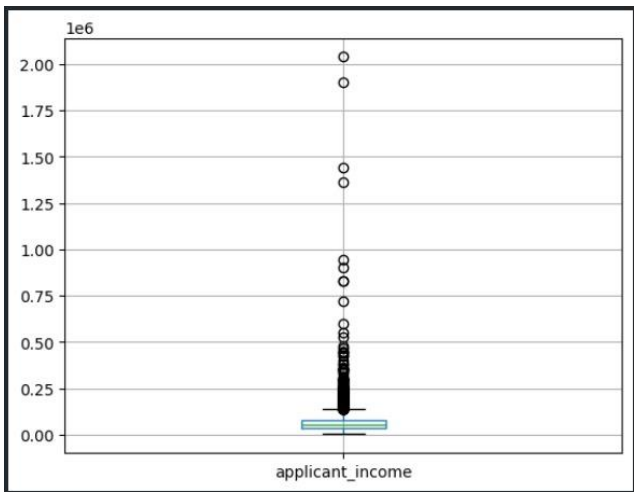

120
100
80
60
40
20
0
job_tenure_years

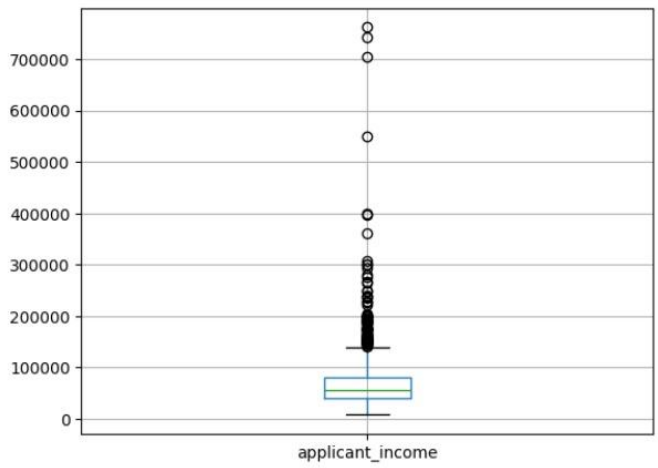

30
25
20
15
10
5
credit_history_length_years

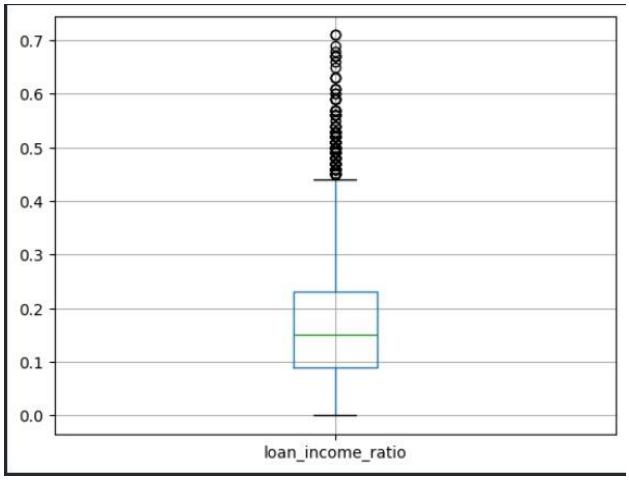

30
25
20
15
10
5
credit_history_length_years

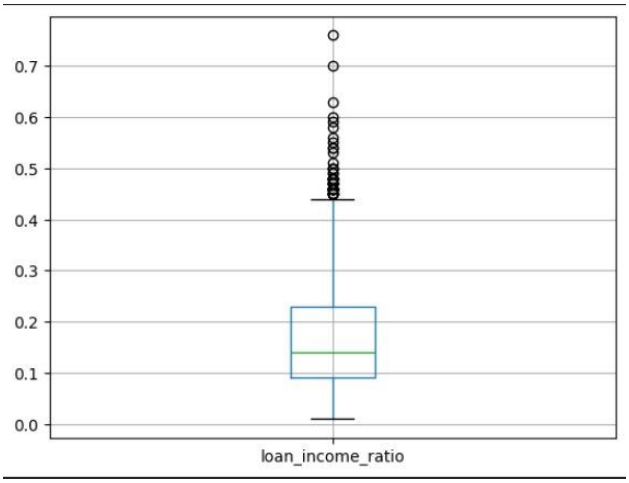

120
100
80
60
40
20
applicant_age

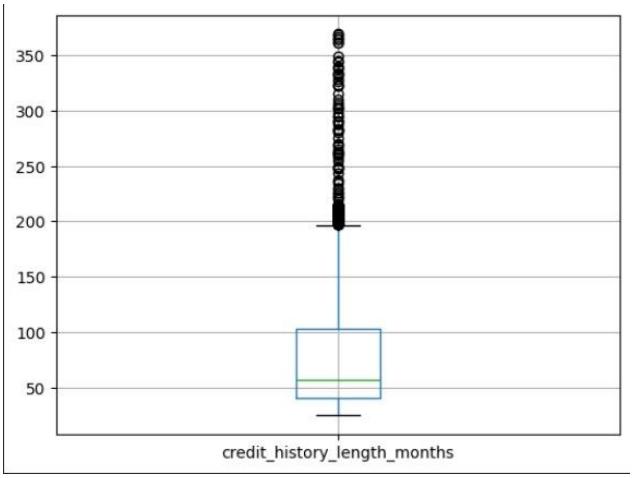

70
60
50
40
30
20
applicant_age

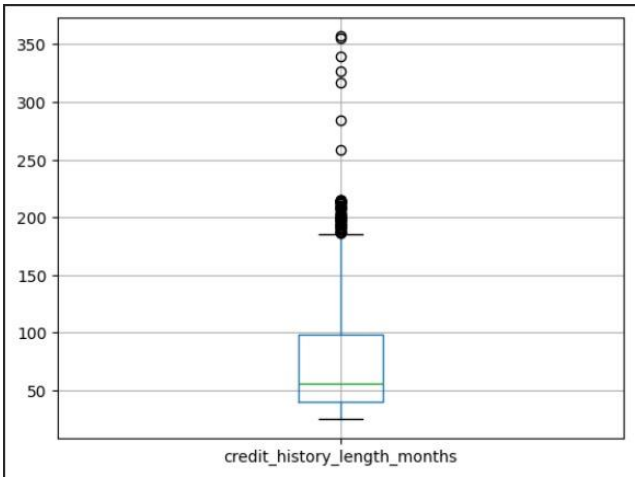

1e6
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
applicant_income


700000
600000
500000
400000
300000
200000
100000
0
applicant_income


0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
loan_income_ratio


0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
loan_income_ratio


350
300
250
200
150
100
50
credit_history_length_months

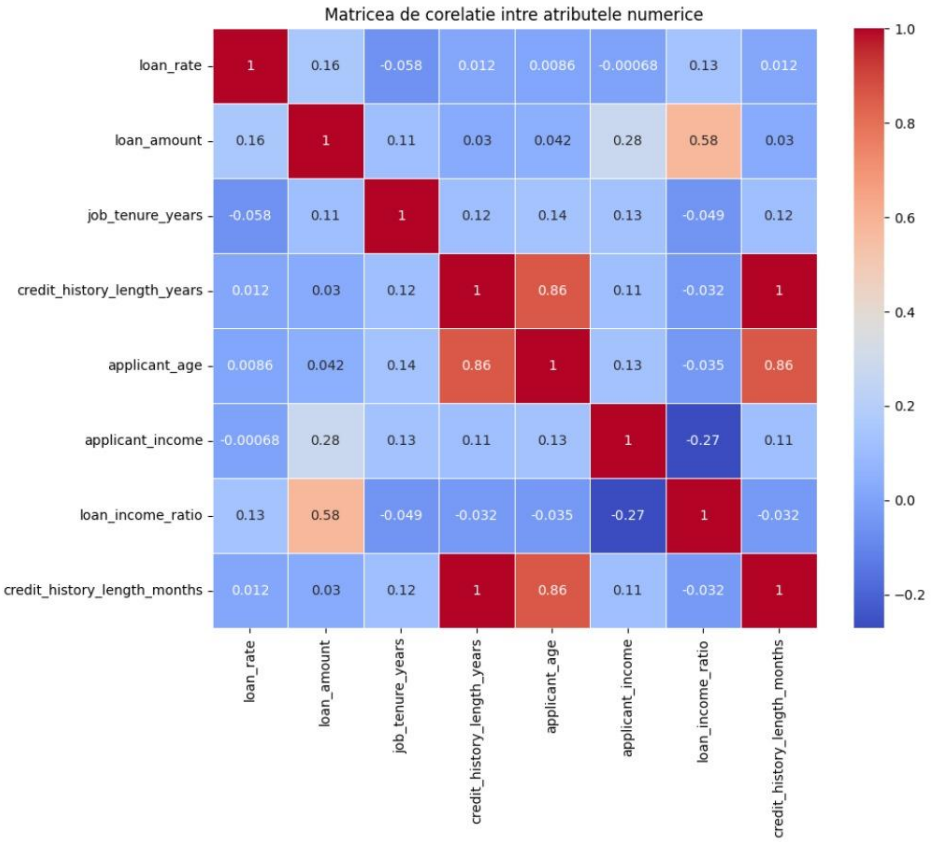

350
300
250
200
150
100
50
credit_history_length_months

Corelația dintre atribute

Corelația dintre atribute ne ajută să observăm dacă avem atribute care sunt nefolositoare. Pentru datele numerice se poate folosi funcția din pandas corr() care poate fi afișată cu ajutorul unui heatmap din seaborn. Pentru cele categorice se poate folosi testul Chi-Pătrat, așa cum este specificat în enunțul temei.

Rezultatul testului Chi-Pătrat arată că aproape toate atributele categorice sunt corelate între ele, însă am decis să nu renunț la niciunul, fapt care poate a influențat rezultat algoritmilor ML.

Corelația dintre atributele numerice:



Corelația dintre atributele categorice:

	residential_status	loan_purpose	loan_approval_status	loan_rating	credit_history_default_status	stability_rating
residential_status	correlated	correlated	correlated	correlated	correlated	correlated
loan_purpose	correlated	correlated	correlated	not-correlated	not-correlated	correlated
loan_approval_status	correlated	correlated	correlated	correlated	correlated	correlated
loan_rating	correlated	not-correlated	correlated	correlated	correlated	correlated
credit_history_default_status	correlated	not-correlated	correlated	correlated	correlated	correlated
stability_rating	correlated	correlated	correlated	correlated	correlated	correlated

Preprocesarea Datelor

Pentru a putea fi folosite de algoritmii de învățare automată, datele de mai sus trebuie prelucrate. Pașii de prelucrare sunt:

Detecția outliers (valorile extreme) - am folosit metoda explicată în enunțul temei, cu cuantilele de 25 și 75% (am afișat outliers pentru fiecare atribut în parte);

Imputarea valorilor lipsă - imputare înseamnă înlocuirea unei valori cu o altă valoare. Seturile de date au multe valori lipsă. Am folosit funcția IterativeImputer (care înlocuiește cu media valorilor) pentru a înlocui valorile numerice lipsă și funcția SimpleImputer (care înlocuiește cu cea mai frecventă valoare) pentru valorile categorice;

Imputarea valorilor extreme (outliers) - pentru outliers, i-am identificat ca mai sus și am înlocuit fiecare valoare lipsă cu null / nan, pentru a putea fi detectate ca valori lipsă și înlocuite cu ajutorul IterativeImputer și SimpleImputer;

Îndepărtarea atributelor puternic corelate (numerice);

Transformarea valorilor categorice în valori numerice (folosind Label Encoder) - valorile categorice trebuie înlocuite cu valori numerice pentru a putea fi procesate de algoritmii ML. LabelEncoder atribuie fiecărei valori unice un număr;

Scalarea datelor - am folosit algoritmul min_max_scaler pentru a standardiza toate datele.

În cazul setului de date salary, singura coloană care a fost îndepărtată, fiind detectată ca puternic corelată, este coloana *prod*.

În cazul setului de date credit, singura coloană care a fost îndepărtată, fiind detectată ca puternic corelată, este coloana *credit_history_length_years*.

Deși în ambele seturi de date au fost detectate și alte corelații puternice, am luat decizia să nu îndepărtez acele atribute, cu riscul de a afecta rezultatul algoritmilor ML, din cauza faptului că aproape toate atributele categorice erau corelate între ele. Iar din cele numerice, am ales să le elimin doar pe cele care aveau coeficientul de corelație 1 sau -1.

Algoritmii de învățare automată

Arbori de decizie

Am folosit modelul implementat în librăria scikit-learn, numit DecisionTreeClassifier(), acesta a fost antrenat pe seturile de date de train și apoi testat pe seturile de date de test.

Metrici pentru setul de date salary:

Accuracy: 0.766

Confussion Matrix:

[[1278 235]

[233 254]]

Precision: [0.84579749 0.5194274]

Recall: [0.84467944 0.52156057]

F1 score: [0.8452381 0.5204918]

Metrici pentru setul de date credit:

Accuracy: 0.842

Confussion Matrix:

[[1369 195]

[121 315]]

Precision: [0.91879195 0.61764706]

Recall: [0.87531969 0.72247706]

F1 score: [0.89652914 0.66596195]

Apoi am realizat implementarea manuală, pornind de la codul din cadrul laboratorului, modificându-l pentru a-l putea antrena și testa pe seturile de date din temă.

Metrici pentru setul de date salary:

Accuracy: 0.7555

Confussion Matrix:

[[1510   3]

[486   1]]

Precision: [0.75651303 0.25   ]

Recall: [0.99801718 0.00205339]

F1 score: [0.86064406 0.00407332]

Metrici pentru setul de date credit:

Accuracy: 0.7805

Confussion Matrix:

[[1536 28]

[411 25]]

Precision: [0.78890601 0.47169811]

Recall: [0.98209719 0.05733945]

F1 score: [0.8749644 0.10224949]

Multi-layered perceptron

Am folosit modelul implementat în librăria scikit-learn, numit MLPClassifier(), acesta a fost antrenat pe seturile de date de train și apoi testat pe seturile de date de test.

Metrici pentru setul de date salary:

```
Accuracy: 0.83
Classification Report:
              precision    recall  f1-score   support

         0.0       0.86      0.93      0.89      1513
         1.0       0.70      0.52      0.60       487

    accuracy                           0.83      2000
   macro avg       0.78      0.73      0.75      2000
weighted avg       0.82      0.83      0.82      2000

Confussion Matrix:
 [[1405  108]
 [ 233  254]]
```

Metrici pentru setul de date credit:

```
Accuracy: 0.89
Classification Report:
              precision    recall  f1-score   support

         0.0       0.90      0.96      0.93      1564
         1.0       0.81      0.62      0.70       436

    accuracy                           0.89      2000
   macro avg       0.86      0.79      0.82      2000
weighted avg       0.88      0.89      0.88      2000

Confussion Matrix:
 [[1501   63]
 [ 165  271]]
```

Apoi am realizat implementarea manuală, pornind de la codul din cadrul laboratorului, modificându-l pentru a-l putea antrena și testa pe seturile de date din temă.

Metrici pentru setul de date salary:

Accuracy: 0.768

Confussion Matrix:

[[1486 27]

[437 50]]

Precision: [0.77275091 0.64935065]

Recall: [0.98215466 0.1026694]

F1 score: [0.86495925 0.17730496]

Metrici pentru setul de date credit:

Accuracy: 0.799

Confussion Matrix:

[[1532 32]

[370 66]]

Precision: [0.80546793 0.67346939]

Recall: [0.97953964 0.15137615]

F1 score: [0.88401616 0.24719101]

Tabel pentru setul de date *salary* cu metrici:

	Acuratețe	Precizie	Recall	F1
Arbori de decizie (Scikit-learn)	0.766	0.6836	0.6831	0.6828
Arbori de decizie (Implementare manuală)	0.7555	0.5032	0.5	0.4323
MLP (Scikit-learn)	**0.83**	**0.78**	**0.73**	**0.75**
MLP (Implemenatre manuală)	0.768	0.711	0.5424	0.5211

Nota: Valorile pentru precision, recall, F1 au fost calculate ca media valorilor celor două clase, fără a lua în calcul weight.

Tabel pentru setul de date *credit* cu metrici:

	Acuratețe	Precizie	Recall	F1
Arbori de decizie (Scikit-learn)	0.842	0.7682	**0.7988**	0.7812
Arbori de decizie (Implementare manuală)	0.7805	0.6303	0.5197	0.4886
MLP (Scikit-learn)	**0.89**	**0.86**	0.79	**0.82**
MLP (Implementare manuală)	0.799	0.7394	0.5654	0.5656

Nota: Valorile pentru precision, recall, F1 au fost calculate ca media valorilor celor două clase, fără a lua în calcul weight.

Concluzie

Se poate observa că pentru ambele seturi de date, în general, modelul MLP din librăria *scikit-learn* are performanțele cele mai bune, poate din cauza faptului că algoritmii implementați manual nu au fost antrenați suficient sau pentru că MLP este mai eficient decât Arbori de decizie pentru aceste seturi de date. Se mai poate observa și o performanță **per total** relativ slabă care poate fi cauzată de faptul că am preprocesat datele prea puțin.