Inteligenta Artificiala
Tema 2
Algoritmi de Machine Learning pentru predictia diabetului si a riscului de credit

Adrian-George Dumitrache
334CC

May 25, 2024

Abstract

Acest document prezinta analiza, prelucrarea si utilizarea a doua seturi de date cu scopul de antrena doua modele de Machine Learning (paduri aleatoare si Multi-Level Perceptron) sa prezica diabetul, respectiv riscul de credit a unei persoane.

1 Prezicerea diabetului

Pentru a antrena modelele folosite sa rezolvam aceasta problema ne vom folosi de un **set de date de 10000 de intrari** ce contin diverse informatii (precum varsta, abuzul de alcool sau rating-ul psihologic) despre cate o persoana si una din 3 etichete: **no diabetes**, **pre-diabetes** si **diabetes**. Vom incepe prin a analiza datele, urmand sa le preprocesam si ulterior sa aplicam algoritmii de machine learning pe acest set de date.

1.1 Analiza datelor

Incepem prin analiza datelor numerice. Din start se observa faptul ca atributul **Metabolical Rate** are valori lipsa, deci putem semnala de pe acum ca acestea trebuie imputate.

	psychological-rating	BodyMassIndex	Age
count	10000.00	10000.00	10000.00
mean	4.37	28.25	8.06
std	8.89	6.46	3.04
min	0.00	14.00	1.00
25%	0.00	24.00	6.00
50%	0.00	27.00	8.00
75%	3.00	31.00	10.00
max	30.00	92.00	13.00

	CognitionScore	Body Stats	Metabolical Rate
count	10000.00	10000.00	9000.00
mean	3.13	194.96	221.59
std	7.31	82.44	60.48
min	0.00	105.06	71.60
25%	0.00	156.72	180.54
50%	0.00	174.04	224.22
75%	2.00	197.74	262.69
max	30.00	553.00	327.94

De asemenea, se observa ca unele dintre aceste atribute au valori anormale:

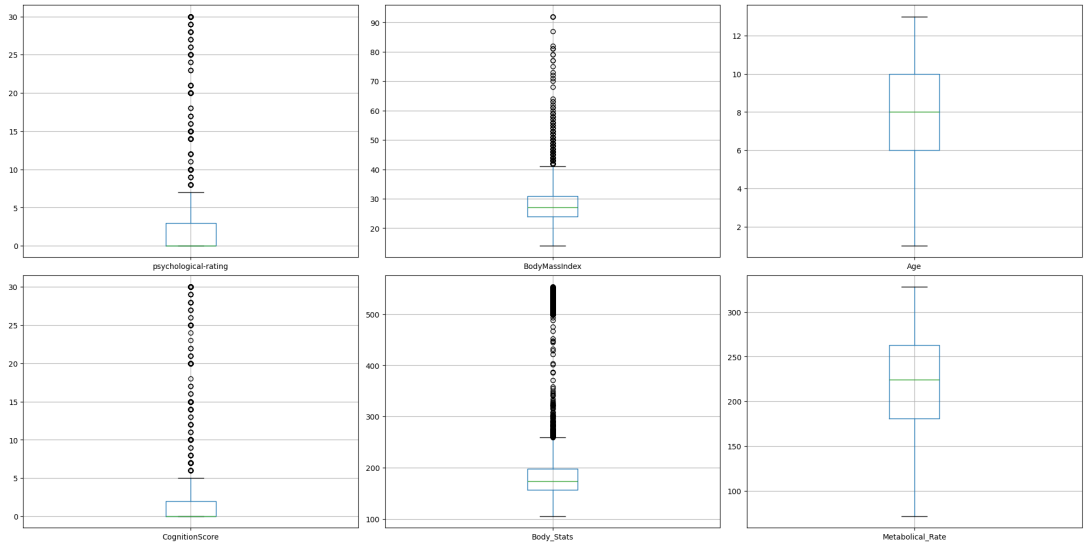

Figure 1: Boxplot al tuturor atributelor numerice

Cele mai mici/mari percentile vor fi sterse si reaproximate ulterior prin metode imputare, fiind probabil valori eronate.

Similar, analizam atributele categorice si observam ca ne **lipsesc valori** pentru **CompletedEduLevel**, acestea vor trebui de asemenea imputate ulterior.

	HealthcareInterest	PreCVA	RoutineChecks	CompletedEduLvl
count	10000	10000	10000	9000
unique	2	2	2	6
top	Pos	0	0	PhD
freq	9534	9580	9158	3778

	alcoholAbuse	cholesterol ver	vegetables	HighBP	Unprocessed fructose	Jogging
count	10000	10000	10000	10000	10000	10000
unique	2	2	2	2	2	2
top	0	1	1	0	1	Rarely
freq	9442	9635	8143	5755	6368	8308

	IncreasedChol	gender	HealthScore	myocardial infarction	SalaryBraket
count	10000	10000	10000	10000	10000
unique	2	2	5	2	8
top	No	Male	2	0	8
freq	5696	5598	3587	9006	3616

	Cardio	ImprovedAveragePulmonaryCapacity	Smoker
count	10000	10000	10000
unique	2	2	2
top	1	1	non-smoker
freq	7560	6368	5612

Se observa de asemenea si o variatie destul de mica in valorile anumitor atribute categorice, ceea ce ar putea sa aduca dificultati.

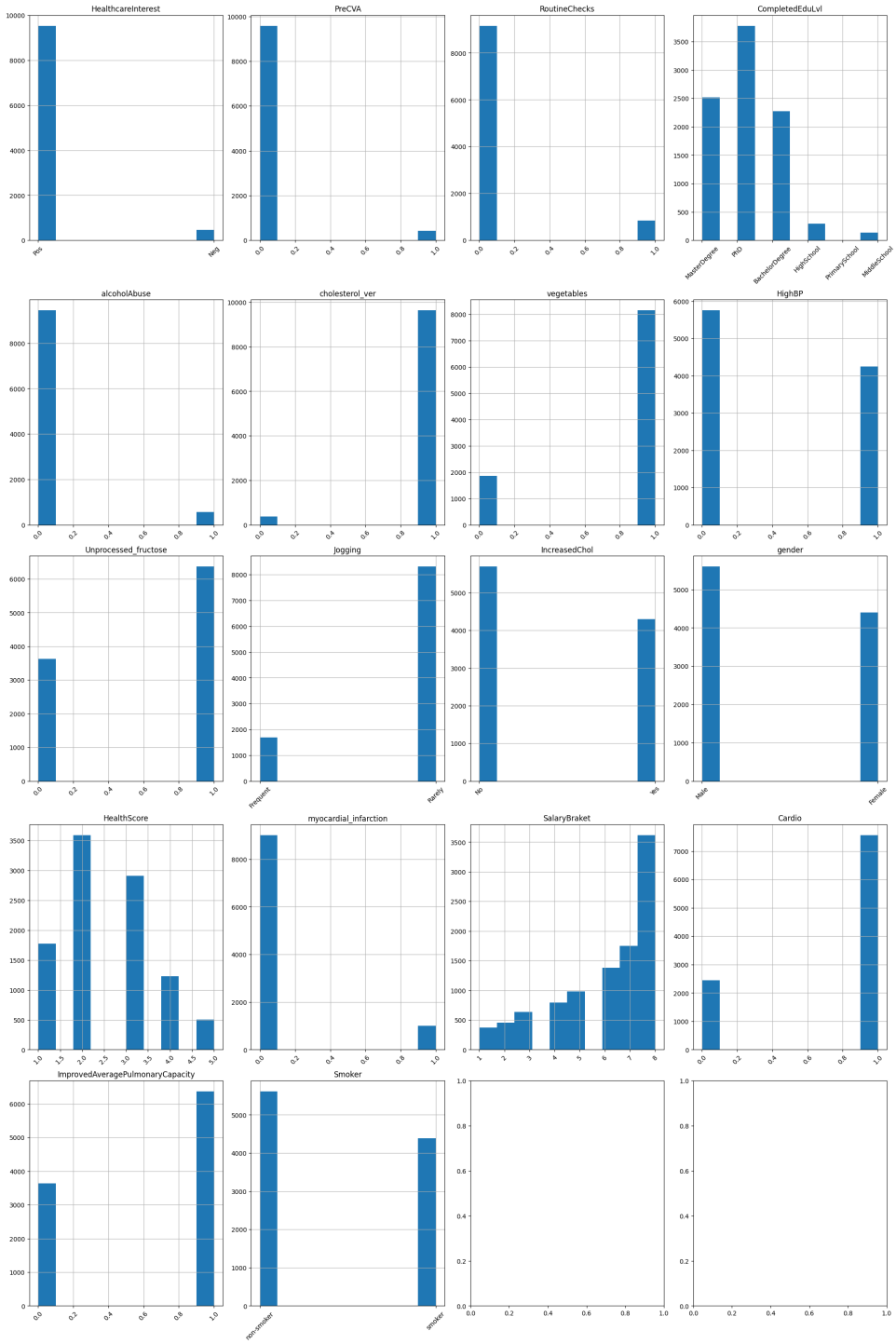

Figure 2: Histograme ale atributelor categorice

In urma analizei claselor apare cea mai mare problema pe care o sa o avem cu acest set de date: o imbalanta mare intre etichete.

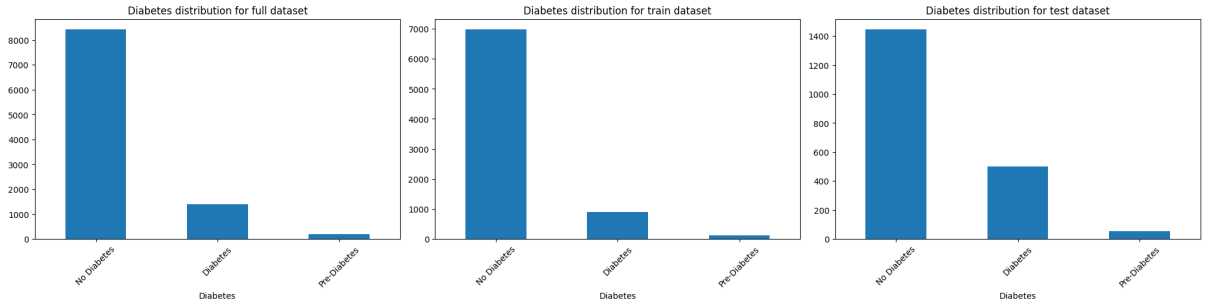

Figure 3: Balanta claselor

Se observa faptul ca numarul de intrari cu eticheta pre-diabet este extrem de mic, motiv pentru care sunt sanse mari sa avem dificultati in prezicerea acestui atribut, neavand prea multe exemple de astfel de cazuri. Ne vom focusa pe metrici precum precizie, recall si F1 in loc de accuracy (pentru ca oricum putem avea un accuracy de peste 70% clasificand totul cu eticheta fara diabet).

Analizand atributele numerice utilizand **metoda Pearson**, se observa o corelare puternica intre varsta si metabolism, ceea ce are sens. Din motive de performanta, vom ignora atributul pentru metabolism (ceea ce este convenabil, pentru ca oricum avea valori lipsa).

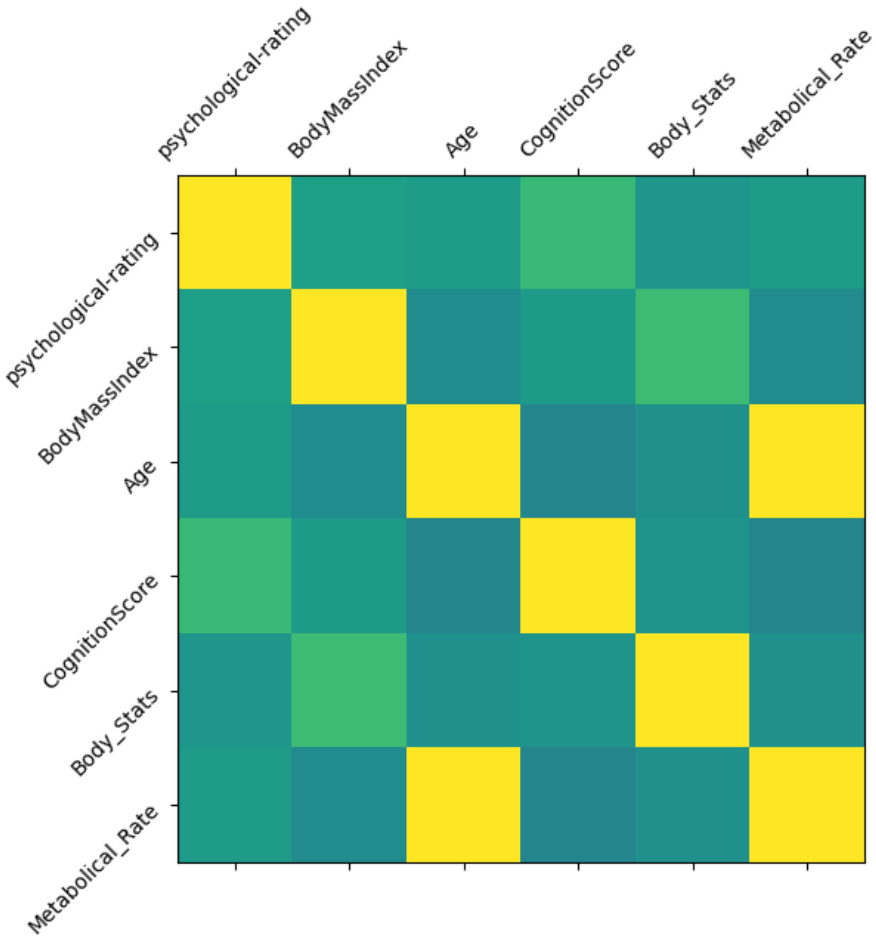

Figure 4: Corelarea atributelor numerice

Aplicam si metoda **Chi-Squared Test** si observam ca unele atribute sunt dependente de aproape toate celelalte atribute, motiv pentru care vom ignora urmatorele atribute: 'Metabolical Rate', 'Smoker', 'CompletedEduLvl', 'Unprocessed fructose'.

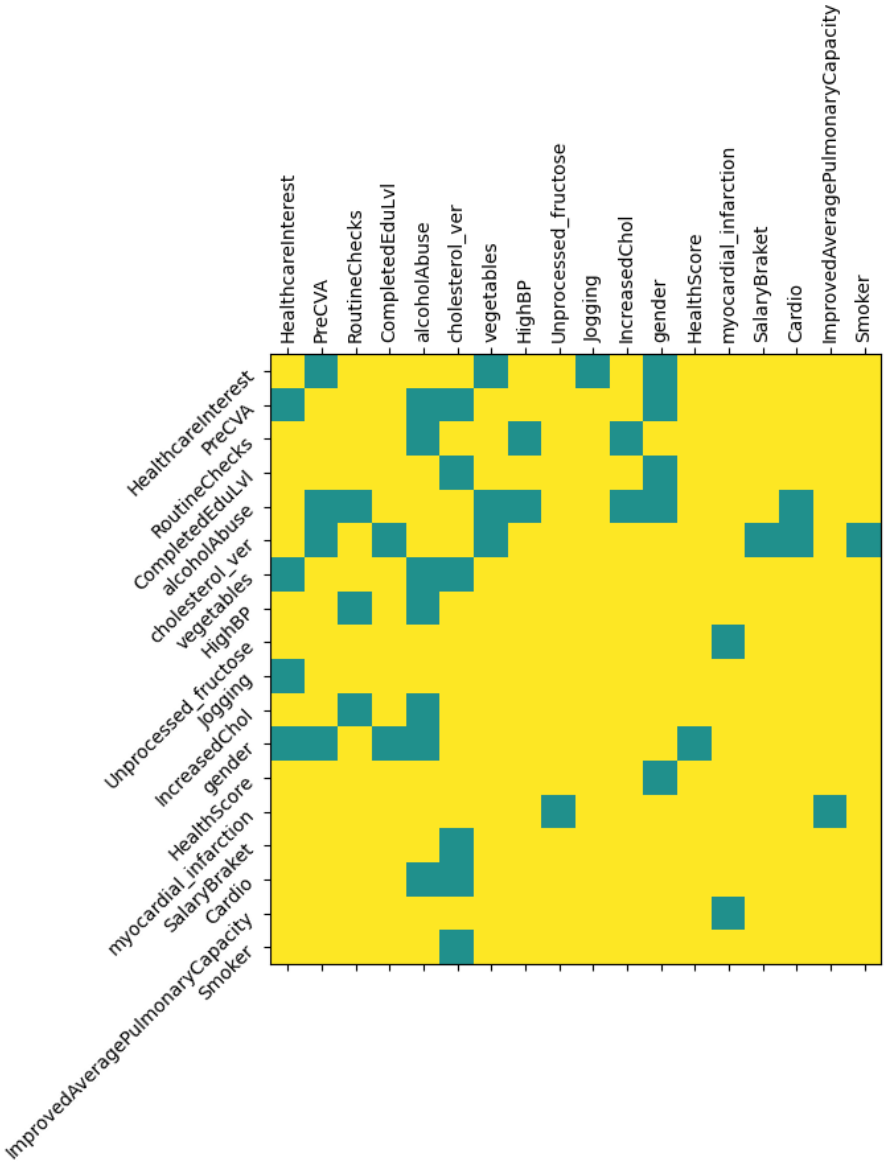

Figure 5: Corelarea atributelor categorice

1.2 Padure aleatoare

Algoritmul aplicat fara vreun parametru adecvat da rezultate destul de mediocre, prezicand aproape toate intrarile ca neavand diabet. Acest lucru poate fi combatat prin intermediul hiperparametrului **classweight**, acest crescand semnificativ toate metricile de prezicere ale atributului diabet si putin si pentru diabet. Am ales sa folosesc doar 10 estimatori deoarece algoritmul pare sa stagneze ca performanta dupa acest numar, deci e inutil sa il incetim. Cea mai buna performanta a fost atinsa cu o adancime de 12, fix jumatate din atribute.

label	precision	recall	f1-score	support
No diabetes	0.79	0.90	0.84	1446
Pre-diabetes	0.15	0.09	0.11	54
Diabetes	0.57	0.36	0.44	500

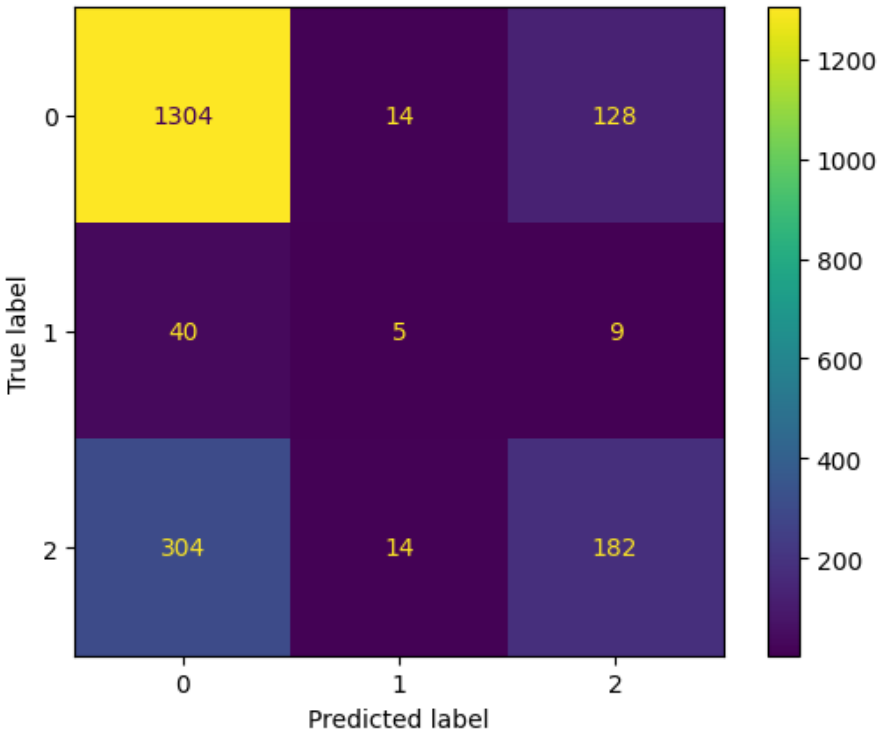

Figure 6: Matrice de confuzie pentru algoritmul padure aleatoare

Varianta implementata de mine are probleme majore din punct de vedere al performantei si al faptului ca nu are implementat hiperparametrul de classWeight. Motiv pentru care rezultatele sunt... proaste. Nu am putut experimenta cu un numar mai mare de estimatori fara sa dau de runtime-uri de peste 30 de minute. Deci rezultatele nu se duc mai departe de preziciri inutile de genul "toate intrarule nu au diabet".

1.3 Multi-Level Perceptron

Am scos performante destul de bune cu cel putin cateva preziceri corecte de pre-diabet fara sa fac prea multe optimizari. Apoi am decis sa folosesc RandomSearchCV pentru a gasi cei mai buni parametrii si am ajuns solver-ul **SGD** alaturi de parametrul de adaptive learning si Nesterov's momentum.

label	precision	recall	f1-score	support
No diabetes	0.76	0.90	0.83	1446
Pre-diabetes	0.08	0.06	0.07	54
Diabetes	0.50	0.26	0.34	500

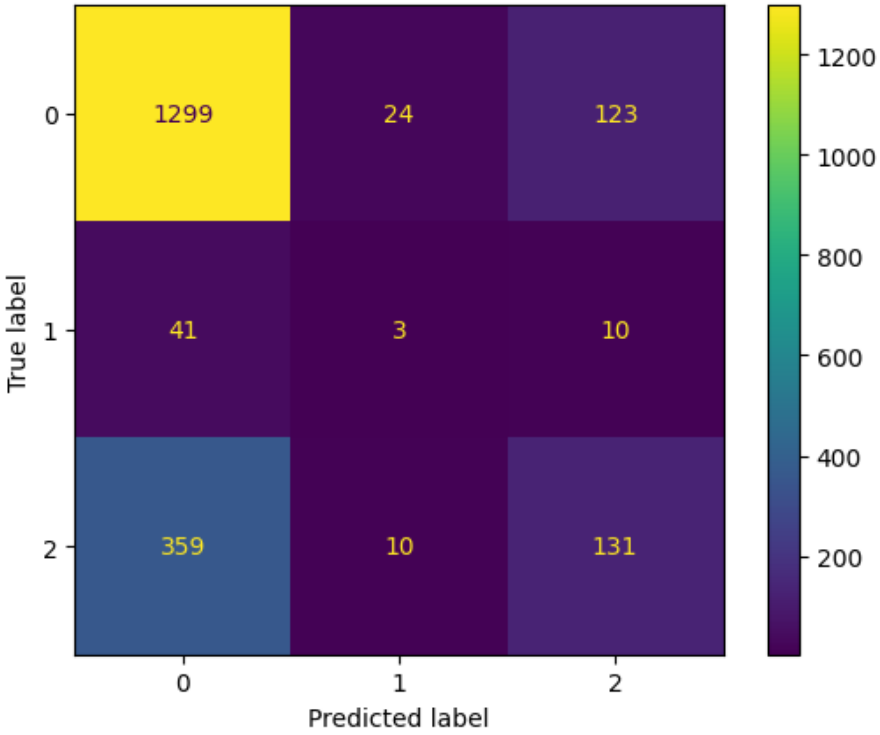

Figure 7: Matrice de confuzie pentru algoritmul MLP

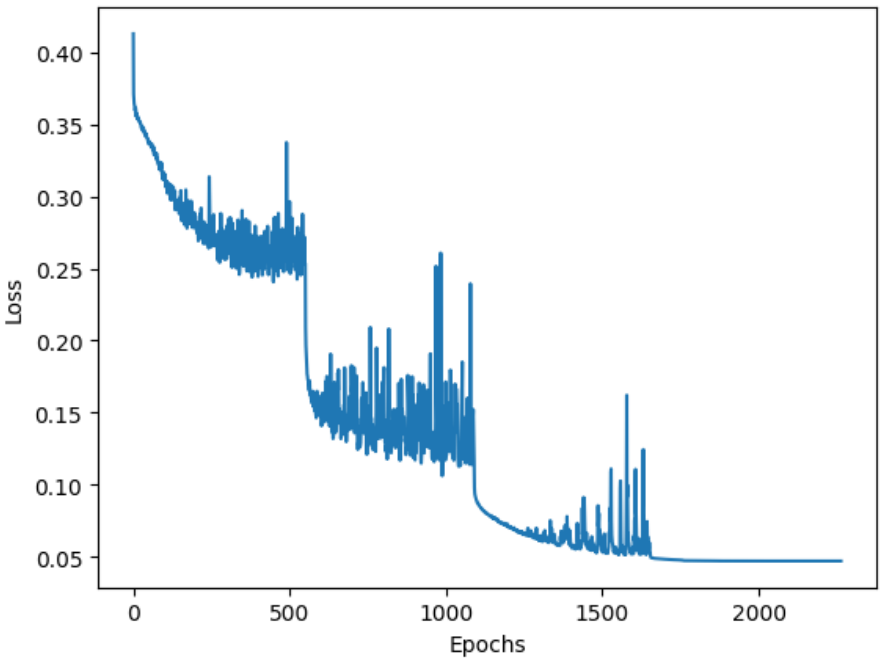

Figure 8: Grafic al loss-ului

Rezultatul algoritmului implementat de mine este urmatorul:

label	precision	recall	f1-score	support
No diabetes	0.77	0.90	0.83	1446
Pre-diabetes	0.00	0.00	0.00	54
Diabetes	0.49	0.28	0.35	500

Se observa din start o imbunatatire fata de padurea aleatoare, dar din cauza lipsei unui hiperparametru pentru invatare adaptiva si un set de date slab calitativ, avem rezultate foarte dezamagitoare in prezicerea pre-diabetului.

1.4 Comparatie

Rezultatele indica ca algoritmul de padure aleatoare a fost putin mai bun in toate metricele, cu exceptia recall-ului, unde a fost mult mai bun. Motivul este probabil faptul ca am decis sa fac mai putine straturi ascunse pentru MLP din considerente de performanta (in special considerand ca antrenarea padurii este de ordinul secundelor).

algorithm	accuracy	precision	recall	f1-score
RF	**0.75**	**0.79, 0.15, 0.57**	**0.90, 0.9, 0.36**	**0.84, 0.11, 0.44**
MLP	0.72	0.76, 0.08, 0.50	0.90, 0.06, 0.07	0.83, 0.07, 0.34

2 Prezicerea riscului de credit

Pentru a antrena modelele folosite sa rezolvam aceasta problema ne vom folosi de un **set de date de 10000 de intrari** ce contin diverse informatii (precum varsta, suma imprumutata si statusul rezidential) despre cate o persoana si una din 3 etichete: **approved** si **declined**. Vom incepe prin a analiza datele, urmand sa le preprocesam si ulterior sa aplicam algoritmii de machine learning pe acest set de date.

2.1 Analiza datelor

Se observa faptul ca atributele **jobTenureYears** si **loanRate** au valori lipsa, deci putem semnala de pe acum ca acestea trebuie imputate.

	applicantAge	applicantIncome	jobTenureYears	loanAmount	loanRate
count	10000	10000	9736	10000	9060
mean	27.745100	65734.211300	4.785744	9568.037500	11.007179
std	6.360155	56944.387081	4.353122	6350.431581	3.266393
min	20.000000	4200.000000	0.000000	500.000000	5.420000
25%	23.000000	38595.000000	2.000000	5000.000000	7.900000
50%	26.000000	55000.000000	4.000000	8000.000000	10.990000
75%	30.000000	78997.000000	7.000000	12200.000000	13.470000
max	123.000000	2039784.000000	123.000000	35000.000000	23.220000

	loanIncomeRatio	creditHistoryLengthYears	creditHistoryLengthMonths
count	10000	10000	10000
mean	0.170130	5.811100	75.760700
std	0.106814	4.050217	48.677362
min	0.000000	2.000000	25.000000
25%	0.090000	3.000000	41.000000
50%	0.150000	4.000000	57.000000
75%	0.230000	8.000000	102.000000
max	0.760000	30.000000	369.000000

De asemenea, se observa ca unele dintre aceste atribute au valori anormale:

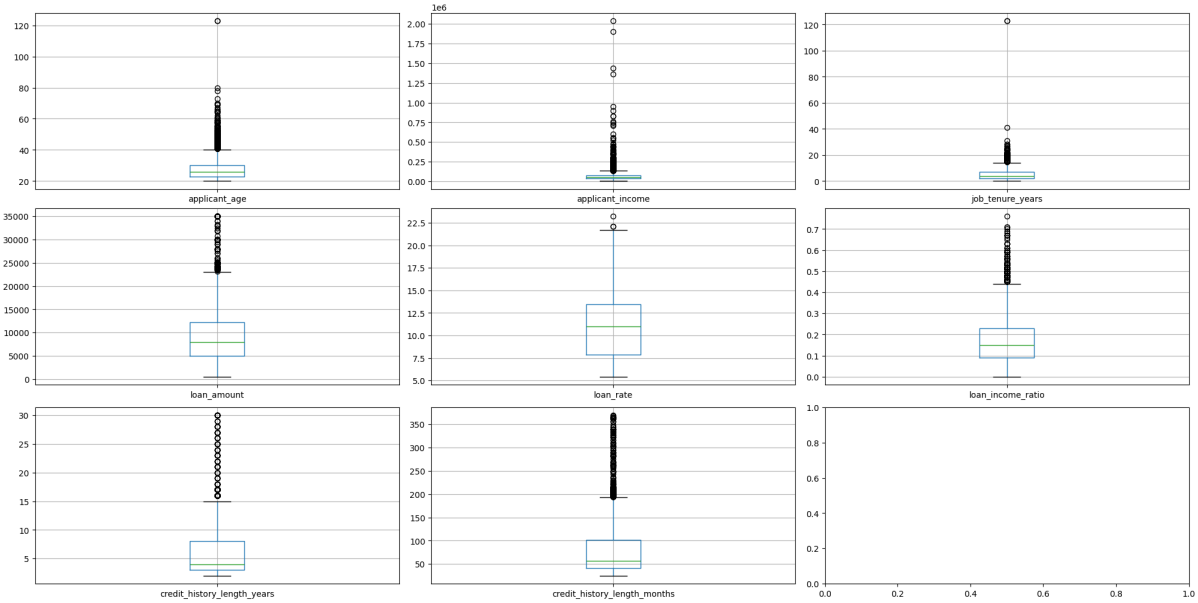

Figure 9: Boxplot al tuturor atributelor numerice

Cele mai mici/mari percentile vor fi sterse si reaproximate ulterior prin metode imputare, fiind probabil nefolositoare

Atributele categorice nu par sa aibe valori lipsa.

	residentialStatus	loanPurpose	loanTating	creditHistoryDefaultStatus	stabilityRating
count	10000	10000	10000	10000	10000
unique	4	6	7	2	4
top	Renter	Study	Excellent	No	C
freq	5056	1971	3325	8264	5056

Se observa de asemenea ca atributele au valori destul de variate, spre deosebire de datasetul anterior.

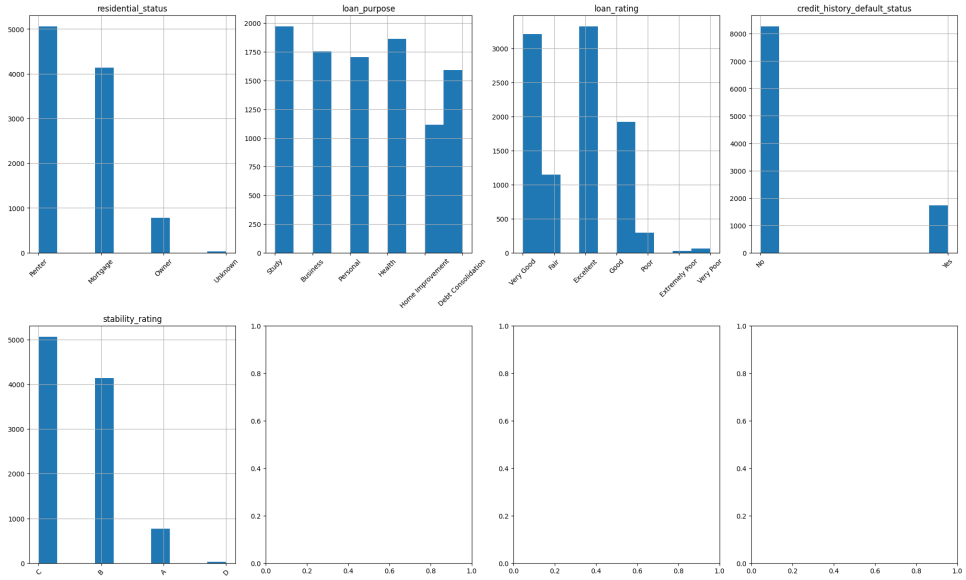

Figure 10: Histograme ale atributelor categorice

In urma analizei claselor observam din nou o situatie mai buna de cat in trecut: nu mai avem aceeasi imbalanta de mai devreme.

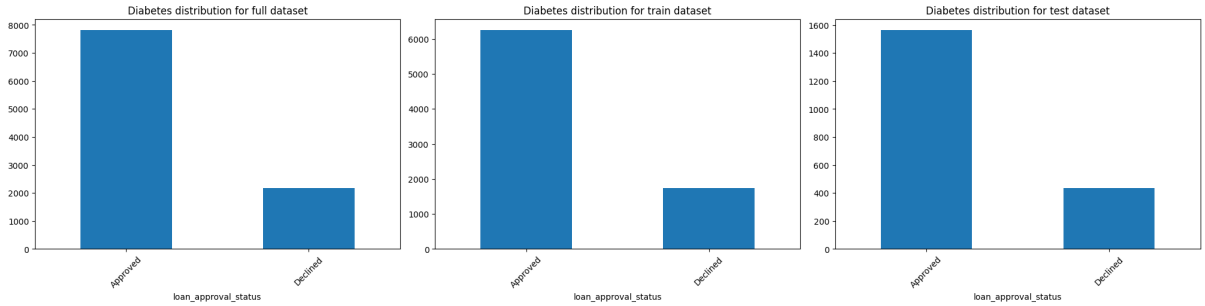

Figure 11: Balanta claselor

Analizand atributele numerice utilizand **metoda Pearson**, se observa o corelare puternica intre vechimea in luni si vechimea in ani de credit, nesurprinzator. Vom elimina una dintre acestea pentru performanta modelului.

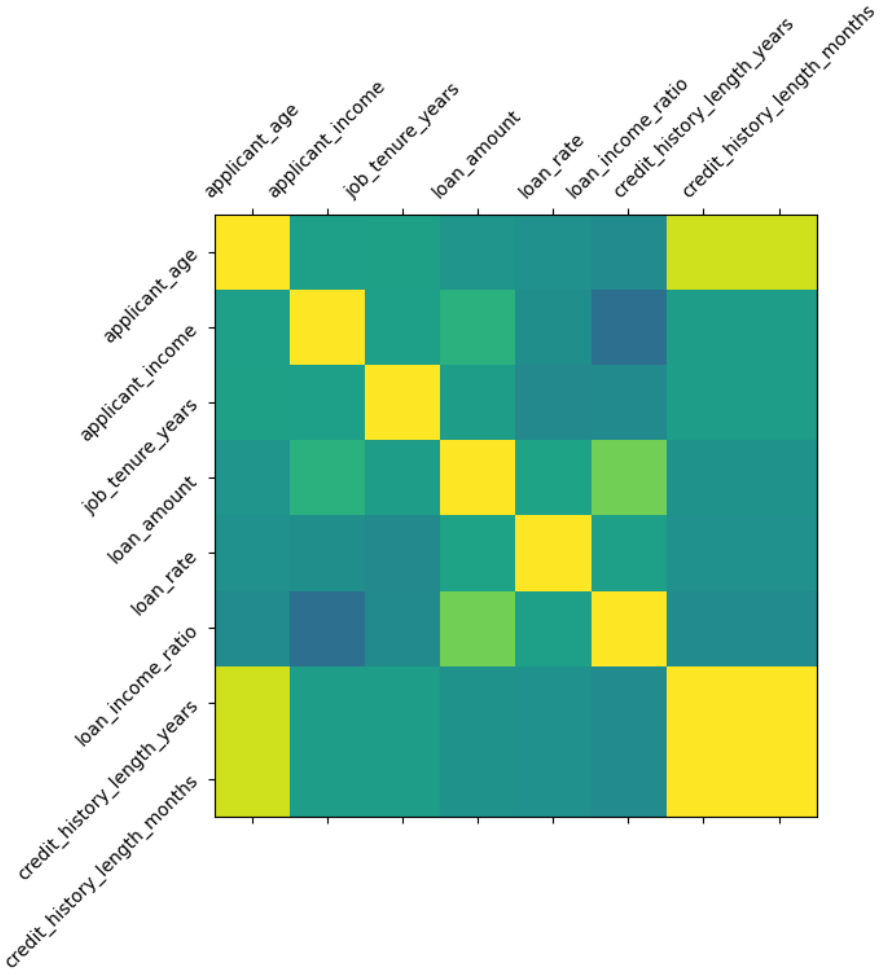

Figure 12: Corelarea atributelor numerice

Aplicam si metoda **Chi-Squared Test** si observam ca unele atribute sunt dependente de aproape toate celelalte atribute, motiv pentru care vom ignora urmatorele atribute: 'residentialStatus' si 'stabilityRating'.

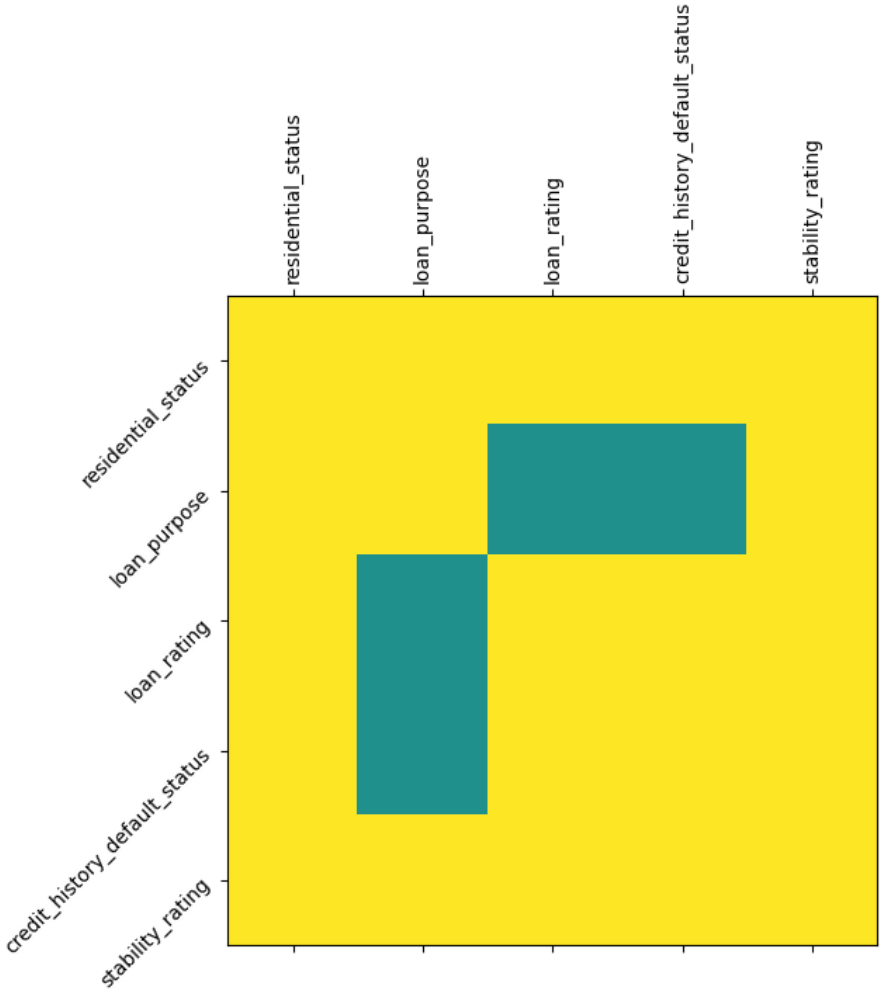

Figure 13: Corelarea atributelor categorice

2.2 Padure aleatoare

Fara problema balansului de clase, putem obtine rezultate foarte bune destul de usor, doar alegand o adancime adecvata pentru problema (dupa un GridSearch am ajuns la 7), optimizand pentru entropie si utilizand ca limitator de atribute log2.

label	precision	recall	f1-score	support
Approved	0.89	0.95	0.92	1564
Declined	0.77	0.58	0.66	436

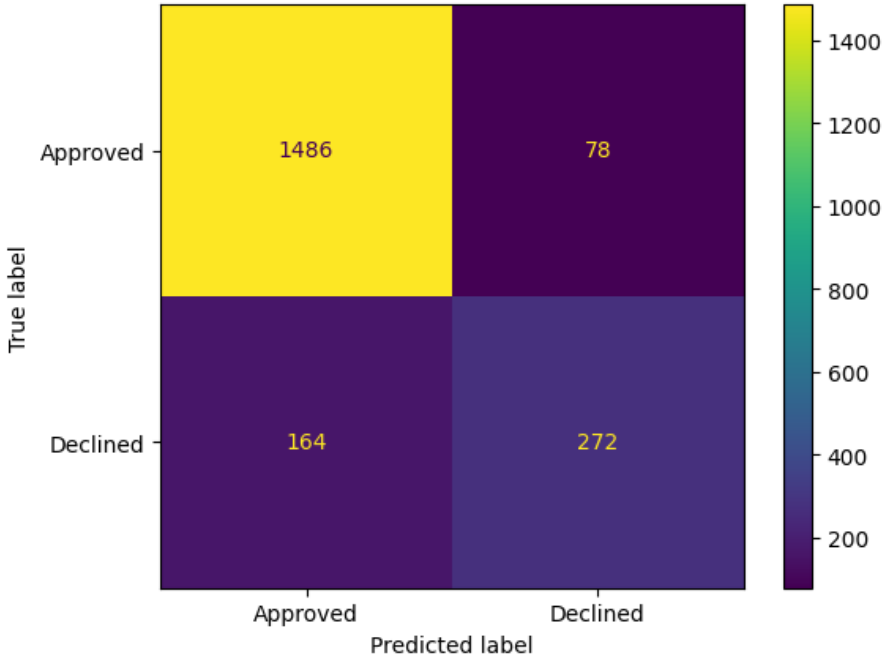

Figure 14: Matrice de confuzie pentru algoritmul padure aleatoare

Implementarea mea sufera de aceeasi problema si aici: performanta foarte slaba. Concluzie? Algoritmii complicati se scriu in C++ si se apeleaza in Python.

label	precision	recall	f1-score	support
Approved	0.72	0.85	0.87	1564
Declined	0.50	0.58	0.65	436

2.3 Multi-Level Perceptron

Am aplicat cam aceeasi parametrii ca la setul de date diabet si am obtinut rezultate extrem de bune, ceea ce demonstreaza cat de importanta este calitatea datelor pe care le folosim.

label	precision	recall	f1-score	support
Approved	0.89	0.95	0.92	1564
Declined	0.75	0.58	0.66	436

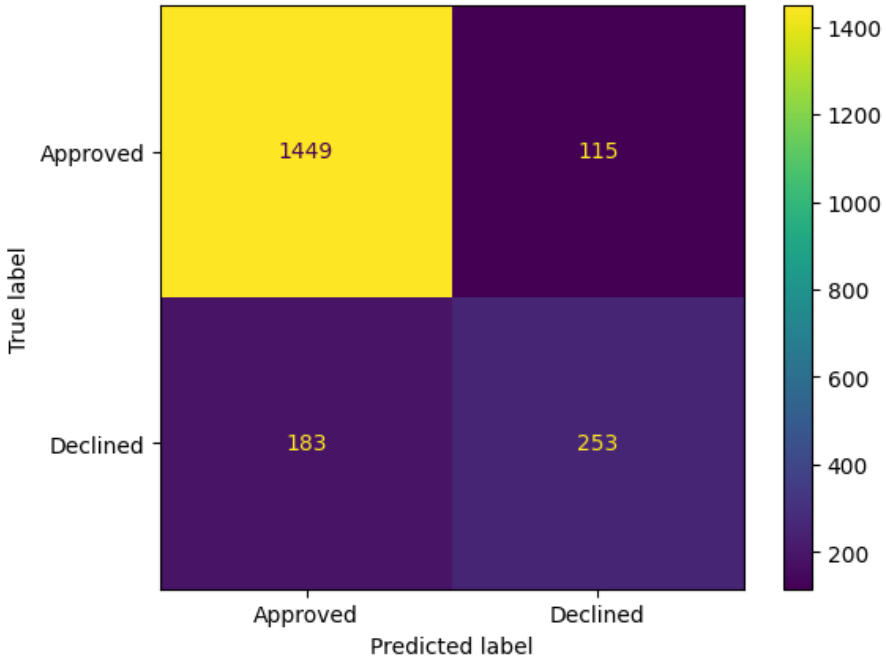

Figure 15: Matrice de confuzie pentru algoritmul MLP

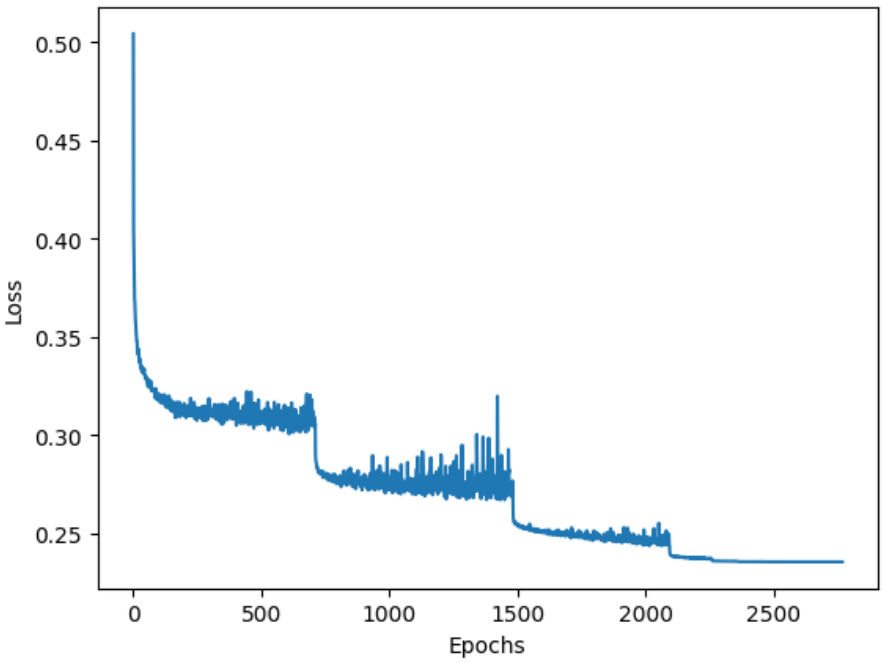

Figure 16: Grafic al loss-ului

Intre timp, implementarea mea are urmatoarele rezultate:

label	precision	recall	f1-score	support
approved	0.87	0.94	0.90	1564
denied	0.70	0.47	0.57	436

Care sunt incredibil de bune all things considered si chiar deloc departe de implementarea din scikit learn, very proud of myself.

2.4 Comparatie

Rezultatele sunt destul de haioase, fiind aproape identice. Calitatea datelor s-a vazut instant. Probabil se pot scoate performante si mai bune de la MLP daca as creste numarul de straturi ascunse.

algoritm	accuracy	precision	recall	f1-score
RF	**0.89**	**0.89, 0.75**	**0.95, 0.58**	**0.92, 0.66**
MLP	0.87	**0.89, 0.75**	**0.95, 0.58**	**0.92, 0.66**