Techniki eksploracji danych

Projekt zaliczeniowy

Projekt polegał na przeglądzie, dopracowaniu i użyciu budowanych algorytmów uczenia maszynowego. Wykonanie projektu uwzględniało kroswalidację, wpływ parametrów modeli i porównanie wyników z bibliotekami R, takimi jak Caret, rpart czy nnet. Badane algorytmy to:

- k-najbliższych sąsiadów
- drzewa decyzyjne
- sieci neuronowe.

1) Klasyfikacja binarna - *Wholesale customers*
   https://archive.ics.uci.edu/ml/datasets/Wholesale+customers
- Liczba wierszy: 440
- Liczba kolumn: 7
- Kolumna Y: „CHANNEL” (kanał sprzedażowy)

2) Klasyfikacja wieloklasowa - *Balance Scale*
   https://archive.ics.uci.edu/ml/datasets/Balance+Scale
- Liczba obserwacji: 625
- Liczba atrybutów: 4
- Zmienna celu: 3 klasy (L, B, R)

3) Regresja - *Computer Hardware*
   https://archive.ics.uci.edu/ml/datasets/Computer+Hardware
- Liczba wierszy: 209
- Liczba kolumn: 9
- Kolumna Y: zmienna numeryczna (wydajność sprzętu)
- Usunięto kolumnę 2 i 10

Najlepsze modele dla poszczególnych problemów klasyfikacji i regresji

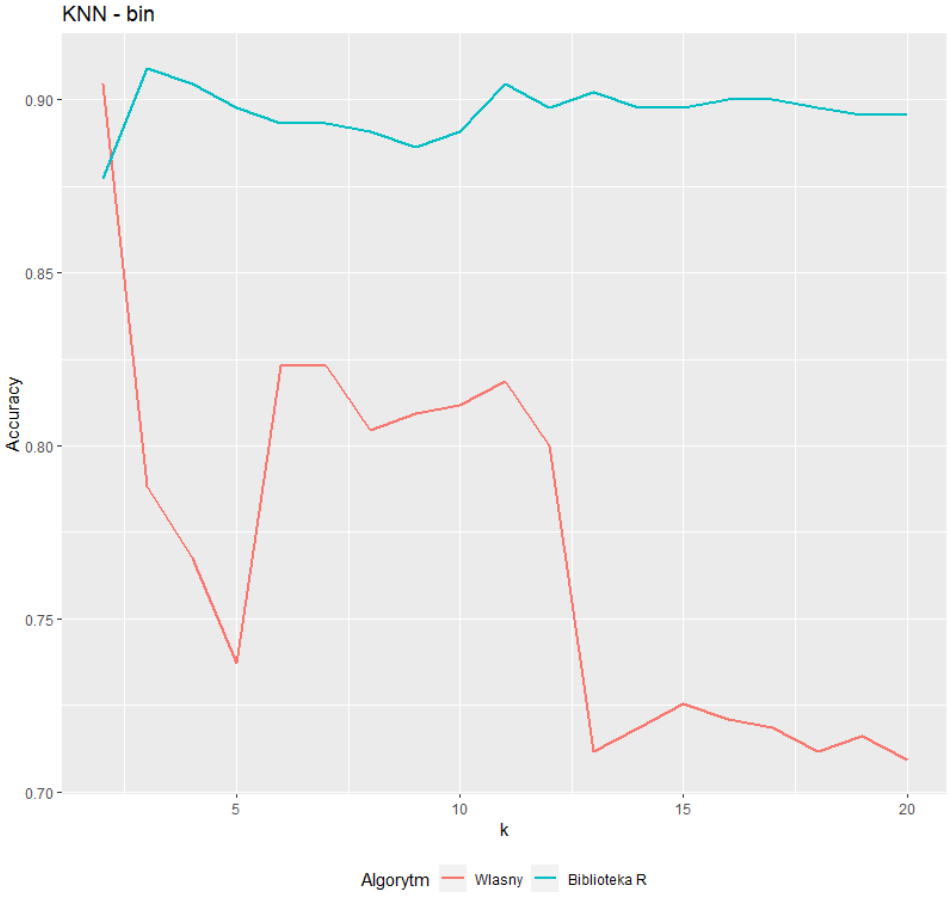
(wyniki dla zbioru walidacyjnego)

	Model	**Implementacja**	**Parametry**	**Dokładność predykcji**
Klasyfikacja Binarna	KNN	Własna	k = 2	0.91
	Drzewa decyzyjne	Własna	depth = 7, minobs = 4, Gini, overfit = prune	0.92
	Sieci NN	Własna	h = (4,6), lr = 0.01, *iter* = 20000	0.91
	KNN	R - caret	k = 3	0.91
	Drzewa decyzyjne	R - Rpart	maxdepth = 3	0.92
	Sieci NN	R - nnet	h = 4	0.91
Klasyfikacja Wieloklasowa	KNN	Własna	k = 20	0.88
	Drzewa decyzyjne	Własna	depth = 7, minobs = 2, Gini, overfit = prune, cf = 0.2	0.75
	Sieci NN	Własna	h = (5,10) / (4,6,8,10), lr = 0.01, *iter* = 100000	0.92
	KNN	R - caret	k = 9	0.91
	Drzewa decyzyjne	R - Rpart	maxdepth = 8	0.78
	Sieci NN	R - nnet	h = 7	0.96
Regresja	KNN	Własna	k = 2	38.74
(MAE)	Drzewa decyzyjne	Własna	depth = 5, minobs = 2, SS	41.22
	Sieci NN	Własna	h = (8), lr = 0.01, *iter* = 100000	34.29
	KNN	R - caret	k = 3	31.54
	Drzewa decyzyjne	R - Rpart	maxdepth = 3	49.96
	Sieci NN	R - nnet	h = 8	0.0016

KNN – Wyniki dla własnej implementacji

Klasyfikacja Binarna

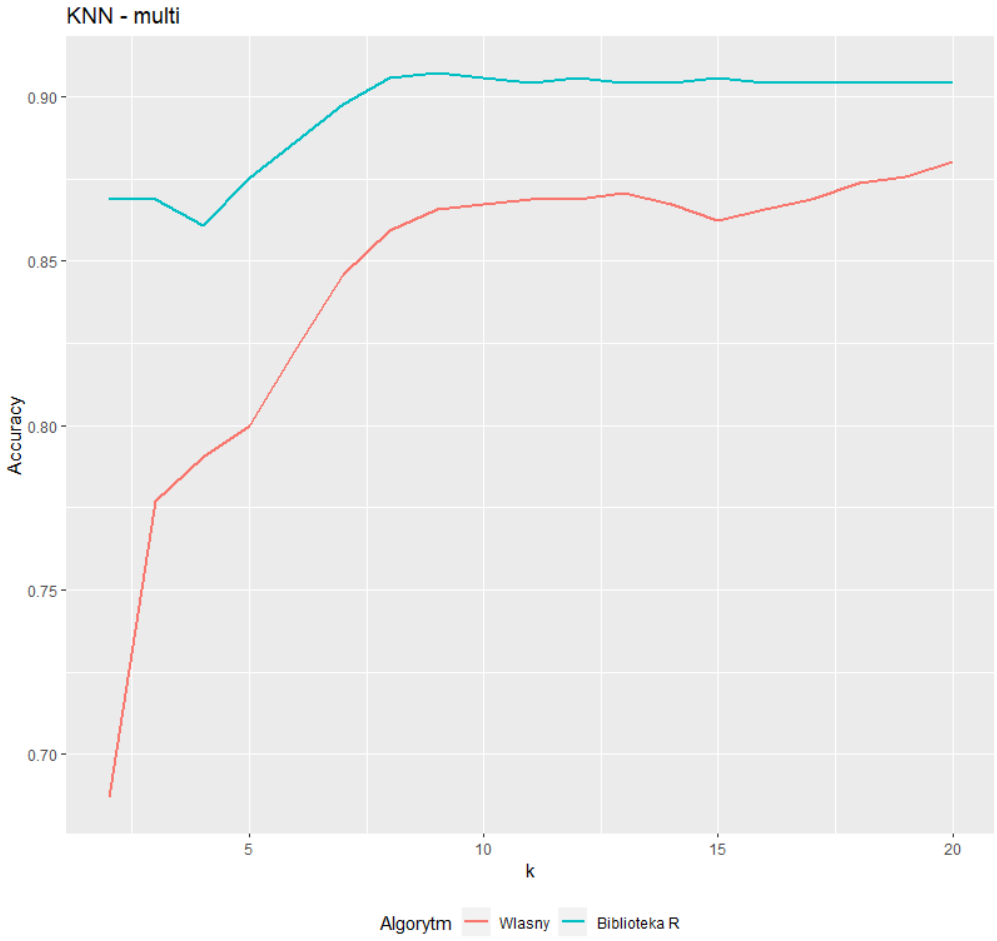
k	AUCW	SensitivityW	SpecificityW	AccuracyW
2	0.94183	0.81615	0.95412	0.90464
3	0.94751	0.64253	0.86730	0.78837
4	0.95311	0.56268	0.87024	0.76745
5	0.95540	0.43187	0.88020	0.73722
6	0.95740	0.47476	0.98020	0.82326
7	0.95857	0.46734	0.98020	0.82326
8	0.95991	0.41289	0.98020	0.80467
9	0.95816	0.41561	0.98323	0.80931
10	0.96060	0.39936	0.99016	0.81163
11	0.96206	0.42951	0.99016	0.81860
12	0.96586	0.37326	0.99016	0.79999
13	0.97088	0.32577	0.89373	0.71162
14	0.97035	0.34173	0.89373	0.71860
15	0.97148	0.37993	0.89016	0.72558
16	0.96954	0.37210	0.89016	0.72094
17	0.96950	0.36580	0.89016	0.71862
18	0.97138	0.34288	0.89016	0.71164
19	0.97204	0.35538	0.89016	0.71629
20	0.97253	0.32894	0.89016	0.70931



KNN – Wyniki dla własnej implementacji

Klasyfikacja Wieloklasowa

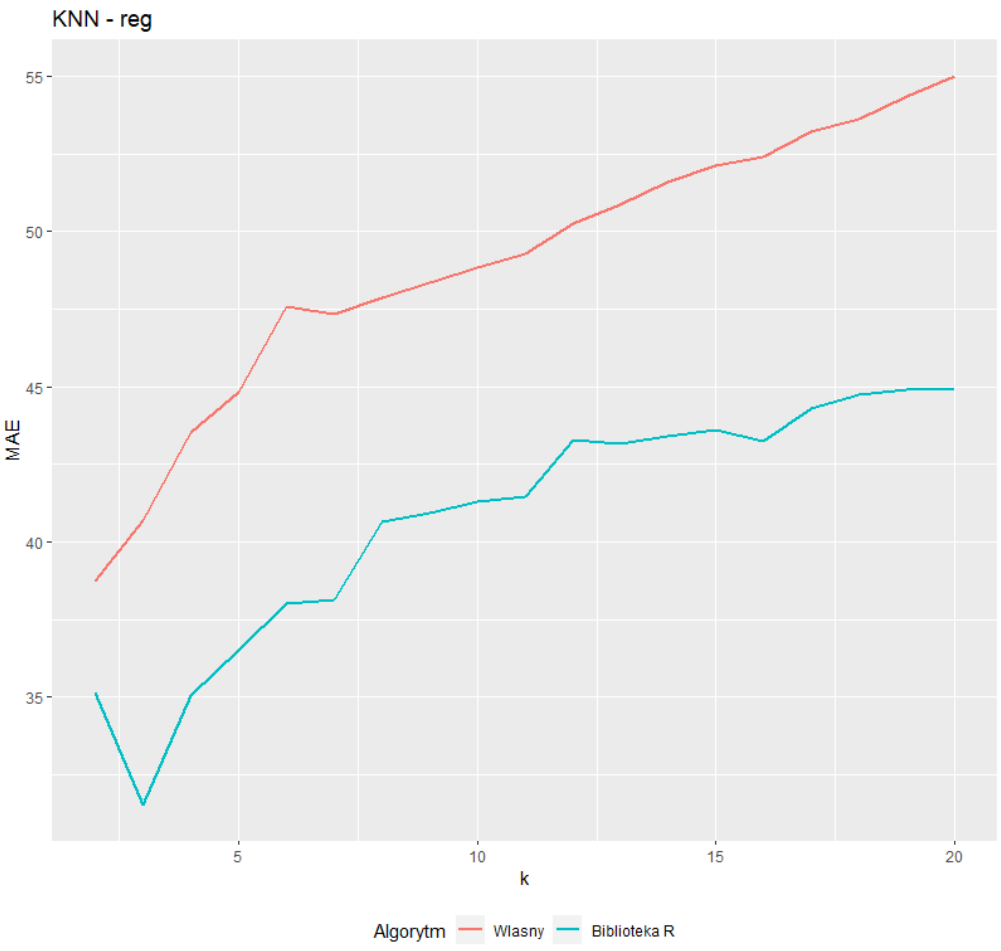
k	ACCW
2	0.6868852
3	0.7770492
4	0.7901639
5	0.8000000
6	0.8229508
7	0.8459016
8	0.8590164
9	0.8655738
10	0.8672131
11	0.8688525
12	0.8688525
13	0.8704918
14	0.8672131
15	0.8622951
16	0.8655738
17	0.8688525
18	0.8737705
19	0.8754098
20	0.8803279



KNN – Wyniki dla własnej implementacji

Regresja

k	MAEW	MSEW	MAPEW
2	38.74211	811.3515	43.11552
3	40.69474	1013.0540	41.78618
4	43.53289	1230.0950	41.89342
5	44.82105	1294.0467	41.99954
6	47.58596	1540.0055	43.54153
7	47.32030	1613.0729	43.52290
8	47.87566	1686.0894	43.72202
9	48.35322	1799.7434	44.41493
10	48.82579	1902.6099	44.35821
11	49.26746	1949.8705	44.82073
12	50.26842	2042.3201	45.92237
13	50.85870	2110.2325	46.36960
14	51.59398	2173.4785	47.09954
15	52.10316	2257.3574	47.49210
16	52.39868	2328.0126	48.02771
17	53.21858	2414.4291	48.94602
18	53.61754	2483.6265	48.91152
19	54.36731	2554.4632	49.27877
20	54.99789	2619.6037	49.52266



Drzewa Decyzyjne (własna implementacja)

Klasyfikacja Binarna

Depth = c(3,5,7), minobs = c(2,4,7), type = c('Entropy', 'Gini'), overfit = c('none', 'prune'), cf = c(0.08, 0.2)

	depth	minobs	type	overfit	cf	AUCW	SensitivityW	SpecificityW	AccuracyW
1	3	2	Entropy	none	0.08	0.95610	0.45860	0.97314	0.81396
2	3	2	Entropy	none	0.20	0.95610	0.45860	0.97314	0.81396
3	3	2	Entropy	prune	0.08	0.95281	0.41934	0.66900	0.59303
4	3	2	Entropy	prune	0.20	0.95217	0.41164	0.66900	0.59071
5	3	2	Gini	none	0.08	0.94613	0.66115	0.85776	0.79767
6	3	2	Gini	none	0.20	0.94613	0.66115	0.85776	0.79767
7	3	2	Gini	prune	0.08	0.94643	0.17837	0.27595	0.25581
8	3	2	Gini	prune	0.20	0.94643	0.17837	0.27595	0.25581
9	3	4	Entropy	none	0.08	0.95546	0.45091	0.97314	0.81163
10	3	4	Entropy	none	0.20	0.95546	0.45091	0.97314	0.81163
11	3	4	Entropy	prune	0.08	0.95281	0.41934	0.66900	0.59303
12	3	4	Entropy	prune	0.20	0.95281	0.41934	0.66900	0.59303
13	3	4	Gini	none	0.08	0.95481	0.66115	0.86070	0.80000
14	3	4	Gini	none	0.20	0.95481	0.66115	0.86070	0.80000
15	3	4	Gini	prune	0.08	0.94905	0.17837	0.17889	0.17907
16	3	4	Gini	prune	0.20	0.94905	0.17837	0.17889	0.17907
17	3	7	Entropy	none	0.08	0.95989	0.42783	0.97981	0.80931
18	3	7	Entropy	none	0.20	0.95989	0.42783	0.97981	0.80931
19	3	7	Entropy	prune	0.08	0.95308	0.41934	0.66900	0.59303
20	3	7	Entropy	prune	0.20	0.95308	0.41934	0.66900	0.59303
21	3	7	Gini	none	0.08	0.95513	0.66115	0.86070	0.80000
22	3	7	Gini	none	0.20	0.95513	0.66115	0.86070	0.80000
23	3	7	Gini	prune	0.08	0.94926	0.17837	0.17889	0.17907
24	3	7	Gini	prune	0.20	0.94926	0.17837	0.17889	0.17907
25	5	2	Entropy	none	0.08	0.94234	0.65307	0.94562	0.84883
26	5	2	Entropy	none	0.20	0.94234	0.65307	0.94562	0.84883
27	5	2	Entropy	prune	0.08	0.94714	0.59059	0.95482	0.83722
28	5	2	Entropy	prune	0.20	0.94650	0.58290	0.95482	0.83489
29	5	2	Gini	none	0.08	0.93787	0.78640	0.94410	0.89534

30	5	2	Gini	none	0.20	0.93787	0.78640	0.94410	0.89534
31	5	2	Gini	prune	0.08	0.94519	0.74907	0.84106	0.81394
32	5	2	Gini	prune	0.20	0.94519	0.74907	0.84106	0.81394
33	5	4	Entropy	none	0.08	0.95266	0.70921	0.94409	0.87210
34	5	4	Entropy	none	0.20	0.95266	0.70921	0.94409	0.87210
35	5	4	Entropy	prune	0.08	0.95157	0.69851	0.94457	0.86279
36	5	4	Entropy	prune	0.20	0.95157	0.69851	0.94457	0.86279
37	5	4	Gini	none	0.08	0.94226	0.75641	0.94457	0.88604
38	5	4	Gini	none	0.20	0.94226	0.75641	0.94457	0.88604
39	5	4	Gini	prune	0.08	0.96194	0.69948	0.74799	0.73488
40	5	4	Gini	prune	0.20	0.96194	0.69948	0.74799	0.73488
41	5	7	Entropy	none	0.08	0.95673	0.65460	0.94732	0.85582
42	5	7	Entropy	none	0.20	0.95673	0.65460	0.94732	0.85582
43	5	7	Entropy	prune	0.08	0.95499	0.52350	0.76335	0.68140
44	5	7	Entropy	prune	0.20	0.95401	0.61239	0.85747	0.77442
45	5	7	Gini	none	0.08	0.95071	0.74568	0.94429	0.88140
46	5	7	Gini	none	0.20	0.95071	0.74568	0.94429	0.88140
47	5	7	Gini	prune	0.08	0.95632	0.59282	0.65444	0.63721
48	5	7	Gini	prune	0.20	0.95561	0.59282	0.65444	0.63721
49	7	2	Entropy	none	0.08	0.92662	0.81605	0.92063	0.88373
50	7	2	Entropy	none	0.20	0.92662	0.81605	0.92063	0.88373
51	7	2	Entropy	prune	0.08	0.92242	0.72121	0.93677	0.87210
52	7	2	Entropy	prune	0.20	0.92216	0.71352	0.93677	0.86977
53	7	2	Gini	none	0.08	0.93022	0.83171	0.92653	0.89534
54	7	2	Gini	none	0.20	0.93022	0.83171	0.92653	0.89534
55	7	2	Gini	prune	0.08	0.93107	0.85532	0.92673	0.90463
56	7	2	Gini	prune	0.20	0.93107	0.85532	0.92673	0.90463
57	7	4	Entropy	none	0.08	0.94478	0.78890	0.94266	0.89302
58	7	4	Entropy	none	0.20	0.94478	0.78890	0.94266	0.89302
59	7	4	Entropy	prune	0.08	0.94620	0.76418	0.93716	0.88140
60	7	4	Entropy	prune	0.20	0.94620	0.76418	0.93716	0.88140
61	7	4	Gini	none	0.08	0.93171	0.75733	0.93394	0.87675
62	7	4	Gini	none	0.20	0.93171	0.75733	0.93394	0.87675
63	7	4	Gini	prune	0.08	0.95609	0.87031	0.94058	0.91860
64	7	4	Gini	prune	0.20	0.95609	0.87031	0.94058	0.91860
65	7	7	Entropy	none	0.08	0.95337	0.70733	0.94732	0.86977
66	7	7	Entropy	none	0.20	0.95337	0.70733	0.94732	0.86977
67	7	7	Entropy	prune	0.08	0.95499	0.52350	0.76335	0.68140
68	7	7	Entropy	prune	0.20	0.95716	0.67406	0.95102	0.85814
69	7	7	Gini	none	0.08	0.94443	0.66025	0.94800	0.85814
70	7	7	Gini	none	0.20	0.94443	0.66025	0.94800	0.85814
71	7	7	Gini	prune	0.08	0.95632	0.59282	0.65444	0.63721
72	7	7	Gini	prune	0.20	0.95924	0.66782	0.74799	0.72558

Drzewa Decyzyjne (własna implementacja)

Klasyfikacja Wieloklasowa

Depth = c(3,5,7), minobs = c(2,4,7), type = c('Entropy', 'Gini'), overfit = c('none', 'prune'), cf = c(0.08, 0.2)

	depth	minobs	type	overfit	cf	ACCW
1	3	2	Entropy	none	0.08	0.6393443
2	3	2	Entropy	none	0.20	0.6393443
3	3	2	Entropy	prune	0.08	0.6393443
4	3	2	Entropy	prune	0.20	0.6393443
5	3	2	Gini	none	0.08	0.6704918
6	3	2	Gini	none	0.20	0.6704918
7	3	2	Gini	prune	0.08	0.6622951
8	3	2	Gini	prune	0.20	0.6622951
9	3	4	Entropy	none	0.08	0.6393443
10	3	4	Entropy	none	0.20	0.6393443
11	3	4	Entropy	prune	0.08	0.6393443
12	3	4	Entropy	prune	0.20	0.6393443
13	3	4	Gini	none	0.08	0.6704918
14	3	4	Gini	none	0.20	0.6704918
15	3	4	Gini	prune	0.08	0.6622951
16	3	4	Gini	prune	0.20	0.6622951
17	3	7	Entropy	none	0.08	0.6393443
18	3	7	Entropy	none	0.20	0.6393443
19	3	7	Entropy	prune	0.08	0.6393443
20	3	7	Entropy	prune	0.20	0.6393443
21	3	7	Gini	none	0.08	0.6704918
22	3	7	Gini	none	0.20	0.6704918
23	3	7	Gini	prune	0.08	0.6622951
24	3	7	Gini	prune	0.20	0.6622951
25	5	2	Entropy	none	0.08	0.7098361
26	5	2	Entropy	none	0.20	0.7098361
27	5	2	Entropy	prune	0.08	0.7114754
28	5	2	Entropy	prune	0.20	0.7098361
29	5	2	Gini	none	0.08	0.7245902
30	5	2	Gini	none	0.20	0.7245902
31	5	2	Gini	prune	0.08	0.7327869
32	5	2	Gini	prune	0.20	0.7262295
33	5	4	Entropy	none	0.08	0.7098361
34	5	4	Entropy	none	0.20	0.7098361
35	5	4	Entropy	prune	0.08	0.7114754
36	5	4	Entropy	prune	0.20	0.7098361
37	5	4	Gini	none	0.08	0.7262295
38	5	4	Gini	none	0.20	0.7262295
39	5	4	Gini	prune	0.08	0.7344262
40	5	4	Gini	prune	0.20	0.7278689
41	5	7	Entropy	none	0.08	0.7163934
42	5	7	Entropy	none	0.20	0.7163934
43	5	7	Entropy	prune	0.08	0.7163934
44	5	7	Entropy	prune	0.20	0.7147541
45	5	7	Gini	none	0.08	0.7311475
46	5	7	Gini	none	0.20	0.7311475
47	5	7	Gini	prune	0.08	0.7393443
48	5	7	Gini	prune	0.20	0.7327869
49	7	2	Entropy	none	0.08	0.7245902
50	7	2	Entropy	none	0.20	0.7245902
51	7	2	Entropy	prune	0.08	0.7311475
52	7	2	Entropy	prune	0.20	0.7327869
53	7	2	Gini	none	0.08	0.7491803
54	7	2	Gini	none	0.20	0.7491803
55	7	2	Gini	prune	0.08	0.7475410
56	7	2	Gini	prune	0.20	0.7524590
57	7	4	Entropy	none	0.08	0.7278689
58	7	4	Entropy	none	0.20	0.7278689
59	7	4	Entropy	prune	0.08	0.7278689
60	7	4	Entropy	prune	0.20	0.7278689
61	7	4	Gini	none	0.08	0.7409836
62	7	4	Gini	none	0.20	0.7409836
63	7	4	Gini	prune	0.08	0.7475410
64	7	4	Gini	prune	0.20	0.7475410
65	7	7	Entropy	none	0.08	0.7245902
66	7	7	Entropy	none	0.20	0.7245902
67	7	7	Entropy	prune	0.08	0.7213115
68	7	7	Entropy	prune	0.20	0.7229508
69	7	7	Gini	none	0.08	0.7409836
70	7	7	Gini	none	0.20	0.7409836
71	7	7	Gini	prune	0.08	0.7508197
72	7	7	Gini	prune	0.20	0.7459016

Drzewa Decyzyjne (własna implementacja)

Regresja

Depth = c(3,5,7), minobs = c(2,4,7), type = c('SS'), overfit = c('none')

depth	minobs	MAEW	MSEW	MAPEW
3	2	47.65351	421.7153	54.91685
3	4	50.97593	525.7710	55.20389
3	7	51.81590	594.9487	55.44479
5	2	41.21984	483.0207	47.08697
5	4	44.68776	444.5735	44.12559
5	7	48.54410	626.7076	44.81811
7	2	41.36897	449.9343	45.63965
7	4	44.62963	443.0277	41.38539
7	7	48.26312	634.2982	43.09095

Sieci Neuronowe - własna implementacja

Klasyfikacja Binarna

h = c(4), c(8), c(4,6), c(5,10), c(6,6,6), c(4,6,8,10)

lr = 0.01 / iteracje = c(20000, 100000)

h	lr	iter	AUCW	SensitivityW	SpecificityW	AccuracyW
4	0.01	2e+04	0.97056	0.67725	0.98062	0.88372
4	0.01	1e+05	0.97073	0.78449	0.96079	0.90233
8	0.01	2e+04	0.97297	0.68796	0.98042	0.88372
8	0.01	1e+05	0.97026	0.71713	0.86011	0.81163
(4, 6)	0.01	2e+04	0.97392	0.77268	0.97369	0.90929
(4, 6)	0.01	1e+05	0.96271	0.80031	0.92492	0.88139
(5, 10)	0.01	2e+04	0.97165	0.76046	0.96664	0.90001
(5, 10)	0.01	1e+05	0.95933	0.81336	0.93452	0.89301
(6, 6, 6)	0.01	2e+04	0.97335	0.56463	0.87661	0.77675
(6, 6, 6)	0.01	1e+05	0.92951	0.70476	0.92479	0.85351
(4, 6, 8, 10)	0.01	2e+04	0.97495	0.51879	0.66920	0.61628
(4, 6, 8, 10)	0.01	1e+05	0.93945	0.69586	0.92532	0.84885

Sieci Neuronowe - własna implementacja

Klasyfikacja Wieloklasowa

h = c(4), c(8), c(4,6), c(5,10), c(6,6,6), c(4,6,8,10)

lr = 0.01 / iteracje = c(20000, 100000)

h	lr	iter	ACCW
4	0.01	2e+04	0.8524590
4	0.01	1e+05	0.8459016
8	0.01	2e+04	0.8508197
8	0.01	1e+05	0.8459016
(4, 6)	0.01	2e+04	0.8524590
(4, 6)	0.01	1e+05	0.9180328
(5, 10)	0.01	2e+04	0.8491803
(5, 10)	0.01	1e+05	0.9213115
(6, 6, 6)	0.01	2e+04	0.8524590
(6, 6, 6)	0.01	1e+05	0.9196721
(4, 6, 8, 10)	0.01	2e+04	0.5737705
(4, 6, 8, 10)	0.01	1e+05	0.9213115

Sieci Neuronowe - własna implementacja

Regresja

h = c(8), c(4,6), c(6,6,6), c(4,6,8,10)

lr = 0.01 / iteracje = c(20000, 100000)

h	lr	iter	MAEW	MSEW	MAPEW
8	0.01	2e+04	33.76962	136.4112	44.59327
8	0.01	1e+05	34.29277	141.4849	43.65135
(4, 6)	0.01	2e+04	43.96494	850.8665	82.75593
(4, 6)	0.01	1e+05	39.13082	622.0212	61.74953
(6, 6, 6)	0.01	2e+04	40.24712	615.4576	73.14972
(6, 6, 6)	0.01	1e+05	36.89039	386.7591	58.19373
(4, 6, 8, 10)	0.01	2e+04	68.68417	3093.4157	159.18592
(4, 6, 8, 10)	0.01	1e+05	41.42146	423.8286	66.62513

Podsumowanie i wnioski

Zaimplementowane modele KNN, Drzew decyzyjnych i Sieci neuronowych bazują na podstawowej idei tych algorytmów. Nie zostały wprowadzone żadne dodatkowe algorytmy, które powodowałyby zwiększenie celności predykcji tych algorytmów. Sieci neuronowe zostały uogólnione dzięki czemu można wprowadzić dowolną ilość warstw o różnej ilości neuronów i sprawdzić ich działanie. Jeśli chodzi o kroswalidację, to w przypadku KNN zrównoleglono obliczenia, ponieważ pojedynczy model, w zależności od rozmiaru danych, model liczył się nawet kilka minut. Wykorzystano do tego funkcję „foreach" ustawiając odpowiednie parametry wejściowe, aby każda z instancji miała potrzebne dane do uzupełnienia tabeli wyjściowej. Dzięki temu siatka modeli z wybranymi parametrami, z uwzględnieniem kroswalidacji (kFold = 10 dla wszystkich obliczeń), policzyła się odpowiednio szybciej - w zależności od sprzętu.

W przypadku klasyfikacji binarnej wszystkie algorytmy zaimplementowane własnoręcznie osiągnęły bardzo podobne maksymalne wyniki w okolicy 0.90 - 0.92. Są to bardzo zadowalające wyniki w porównaniu do modeli z bibliotek R, które osiągnęły precyzję na tym samym poziomie. Dane do klasyfikacji wieloklasowej wykazały słabość algorytmu Drzew decyzyjnych. W przypadku obu implementacji drzewa osiągnęły najgorsze wyniki – 0.75 dla własnej implementacji oraz 0.78 dla Rpart. KNN osiągnął dokładność predykcji 0.88, a sieć neuronowa 0.92 . Tutaj funkcje z bibliotek okazały się nieco lepsze, z wynikami odpowiednio - 0.91 dla KNN oraz 0.96 dla sieci neuronowej. Ostatnim problemem była regresja. Modele zbudowane przy pomocy bibliotek R osiągnęły wyniki podobne do własnych implementacji. Najlepszym modelem dla regresji okazały się sieci neuronowe z biblioteki R, gdzie MAE wyniosło 0.0016 . Najlepszym własnym modelem jest w tym przypadku jest sieć neuronowa z MAE wynoszącym 34.29 . Pozostałe własne modele osiągnęły odpowiednio: KNN – 38.74 (caret = 31.54) oraz drzewa decyzyjne – 41.22 (Rpart = 49.96).

Wyniki, które zostały porównane dotyczą wybranych modeli i użytych parametrów. Może się okazać, że najlepsze parametry zostały pominięte w siatce hiper-parametrów. Podsumowując wyniki można śmiało stwierdzić, że własne implementacje nie odbiegają, aż tak bardzo od dopracowanych implementacji z bibliotek R. Można śmiało stwierdzić, że algorytmy z bibliotek są dużo lepiej zoptymalizowane pod kątem szybkości działania.