AUTO ML

Praca domowa nr 1

Weronika Dyszkiewicz, Katarzyna Mamla

21.11.2023

1 Wstęp i zbiory danych

Celem projektu jest przeanalizowanie tunowalności hiperparametrów 3 algorytmów uczenia maszynowego, w naszym przypadku:

- Random Forest,
- Light GBM,
- XgBoost.

na 4 zbiorach danych: Titanic, Speed dating, Diabetes oraz Credit-g. Algorytmy te obejmują szereg hiperparametrów, które muszą być być ustawione przed ich uruchomieniem. Do tunowania modeli wykorzystałyśmy 2 techniki losowania punktów: Random Search CV oraz Bayes Optimization. Do opytmalizacji bayesowskiej parametru użyłyśmy funkcji *gp_minimize* z pakietu **skopt**. A do Random Searcha, funkcji *RandomizedSearchCV* z **sklearn.model_selection**. Cały nasz eksperyment będzie dotyczył klasyfikacji binarnej.

1.1 Ramki danych

Pierwszym zbiorem danych jest zbiór Titanic (ozn. 1), opisujący status przeżycia poszczególnych pasażerów Titanica zdefiniowana kolumna jako *survived*. Zmienne w naszym wyodrębnionym zbiorze danych to m.in.: *pclass, survived, name, age, embarked*. Drugim zbiorem jest zbiór Speed Dating (ozn, 2). Dane te zostały zebrane od uczestników eksperymentalnych szybkich randek. Pod koniec randek uczestnicy byli pytani, czy chcieliby ponownie spotkać się ze swoją randką. Zostali również poproszeni o ocenę swojej randki pod względem sześciu atrybutów takich jak: *Attractiveness, Sincerity, Intelligence, Fun, Ambition oraz Shared Interests*. Predyktorem jest tzw 'match'. Trzecim zbiorem jest Credit-g (ozn. 3). Ten zbiór danych klasyfikuje osoby opisane przez zestaw atrybutów jako *good* lub *bad* pod względem ryzyka kredytowego. Ostatnim zbiorem jest zbiór Diabetes (ozn. 4). Tutaj przewidujemy czy osoba jest zakwalifikowana jako cukrzyk z *tested_negative* czy *tested_positive*.

2 Poszczególne algorytmy oraz wyniki

2.1 Random Forest

Wybrałyśmy ten algorytm, ponieważ jest jednym z popularniejszych i u nas będzie to algorytm, który dostawał średnio najmniejsze scory, co było już widoczne przy zastosowaniu autogluona do

Hiperparametry oraz accuracy score				
Zbiór danych	**Metoda samplingu**	{*'n_estimators', 'min_samples_split', 'min_samples_leaf', 'max_features','max_depth'*}	**Score**	Zbieżność iteracje
Titanic	RandomizedSearchCV	{37, 8, 1, 12, 18}	0.7354568238779	-
	Bayes	{34, 2, 1, 13, 20}	0.9083969465649	38
SpeedDating	RandomizedSearchCV	{107, 2, 1, 11, 15}	0.8646669337874	-
	Bayes	{112, 18, 1, 13, 19}	0.8621718377083	3
Credit-g	RandomizedSearchCV	{45, 4, 3, 2, 8}	0.72	-
	Bayes	{117, 17, 1, 2, 20}	0.745	21
Diabetes	RandomizedSearchCV	{50, 6, 7, 4, 10}	0.7720778355325	-
	Bayes	{119, 17, 1, 2, 20}	0.8116883116883	16

Tabela 1: Optymalne kombinacje hiperparametrów dla zbiorów danych 1-4 korzystając z RandomizedSearchCV i Optymalizacji Bayesowskiej z algorytmem RandomForest.

przewidywania najlepszych algorytmów dla poszczególnych ramek. Jednak użyłyśmy go, aby był do porównania innych, lepszych algorytmów. W pierwszym zbiorze danych, zestawienie najlepszych modeli dla domyślnych parametrów według pakietu **autogluon** Tabela 5.

W naszym projekcie, optymalizowałyśmy następujące parametry Random forest: *n_estimators, max_depth, min_samples_split, min_samples_leaf oraz max_features*. Wraz ze wzrostem liczby drzew wzrasta precyzja wyniku, a zatem lepsza dokładność, a nadmierne dopasowanie jest ograniczone. Jednak sprawi to, że model będzie wolniejszy. Stąd siatka parametru *n_estimators* mieści się w granicach [1,120]. Jak wiemy za duży *max_depth* mógłby powodować overfitting modelu, dlatego jako granice dałyśmy [3,20]. mniejsza wartość *min_sample_leaf* sprawi, że model będzie bardziej podatny na wykrywanie szumu, więc optymalizacja będzie w granicach [1,20]. Optymalne kombinacje hiperparametrów, score oraz liczba iteracji, po których score sie stabilizuje dla tego algorytmu przedstawia Tabela 1 oraz Rysunek 2 i 3.

2.2 Light GBM

LightGBM jest algorytmem opartym na drzewach, wykorzystujacym gradient boosting. Parametrami które chcemy zoptymalizować są *num_leaves, n_estimators,learning_rate* Optymalizacja tych parametrów odbywała się odpowiednio w granicach: $[50, 300], [0.01, 0.3], [10, 40]$. Optymalne kompinacje hiperparametrów, score oraz liczba iteracji, po których score sie stabilizuje dla tego algorytmu przedstawia Tabela 2 oraz Rysunek 2 i 3.

2.3 XG Boost

XGBoost lub eXtreme Gradient Boosting to kolejny algorytm również oparty na drzewach i wzmocnieniu gradientowym. Tutaj zdecydowałyśmy się na optymalizację parametrów {*'subsample', 'min_child_weight', 'max_depth', 'eta','colsample_bytree'*} Optymalizacja tych parametrów odbywała się odpowiednio w granicach: $[3, 10], [0.1, 1], [0, 7], [0, 1], [0.1, 1]$. Optymalne kombinacje hiperparametrów, score oraz liczba iteracji, po których score się stabilizuje dla tego algorytmu przedstawia Tabela 3 oraz Rysunek 2.

Hiperparametry oraz accuracy score				
Zbiór danych	**Metoda samplingu**	{*num_leaves', 'n_estimators', learning_rate'*}	**Score**	Zbieżność iteracje
Titanic	RandomizedSearchCV	{15, 74, 0.03}	0.9751742993848	-
	Bayes	{11, 51, 0.05}	0.9770992366412	2
SpeedDating	RandomizedSearchCV	{36, 272, 0.06}	0.875409306933	-
	Bayes	{11, 51, 0.05}	0.9770992366412	2
Credit-g	RandomizedSearchCV	{12, 67, 0.04}	0.7075	-
	Bayes	{11, 51, 0.05}	0.9770992366412	2
Diabetes	RandomizedSearchCV	{10, 141, 0.03}	0.7476342796215	-
	Bayes	{11, 51, 0.05}	0.9770992366412	2

Tabela 2: Optymalne kombinacje hiperparametrów dla zbiorów danych 1-4 korzystając z RandomizedSearchCV i Optymalizacji Bayesowskiej z algorytmem LightGBM.

Hiperparametry oraz accuracy score				
Zbiór danych	**Metoda samplingu**	{*'subsample', 'min_child_weight', 'max_depth', 'eta','colsample_bytree'*}	**Score**	Zbieżność iteracje
Titanic	RandomizedSearchCV	{0.6, 0, 6, 0.7, 0.2}	0.9761312371839	-
	Bayes	{0.10211525521493146, 0.056548909954715745, 3.154448299303527, 0.9753577689104947, 0.3745966931006509 }	0.9809160305344	33
SpeedDating	RandomizedSearchCV	{0.4, 1, 7, 0.1, 0.6}	0.868993065990	-
	Bayes	{0.20000000298023224, 1, 5, 0.10000000149011612, 0.20000000298023224}	0.9770992366412	1
Credit-g	RandomizedSearchCV	{0.3, 4, 7, 0.1, 0.9}	0.6975	-
	Bayes	{0.20000000298023224, 1, 5, 0.10000000149011612, 0.20000000298023224}	0.9770992366412	1
Diabetes	RandomizedSearchCV	{0.2, 0, 6, 0.4, 0.8}	0.759016393443	-
	Bayes	{0.20000000298023224, 1, 5, 0.10000000149011612, 0.20000000298023224}	0.9770992366412	1

Tabela 3: Optymalne kombinacje hiperparametrów dla zbiorów danych 1-4 korzystając z RandomizedSearchCV i Optymalizacji Bayesowskiej z algorytmem XGBoost.

Tunowalność		
Algorytm	Miara Tunowalności	Średnia tunowalność
RandomForest	$d^{(1)} = R^{(1)}(\theta^*) - R^{(1)}(\theta^{(1)*}) = -0.735 + 0.735 = 0$	0.00900
	$d^{(2)} = R^{(2)}(\theta^*) - R^{(2)}(\theta^{(2)*}) = -0.864 + 0.865 = 0.001$	
	$d^{(3)} = R^{(3)}(\theta^*) - R^{(3)}(\theta^{(3)*}) = -0.696 + 0.72 = 0.024$	
	$d^{(4)} = R^{(4)}(\theta^*) - R^{(4)}(\theta^{(4)*}) = -0.761 + 0.772 = 0.011$	
LightGBM	$d^{(1)} = R^{(1)}(\theta^*) - R^{(1)}(\theta^{(1)*}) = -0.975 + 0.975 = 0$	0.00425
	$d^{(2)} = R^{(2)}(\theta^*) - R^{(2)}(\theta^{(2)*}) = -0.863 + 0.875 = 0.012$	
	$d^{(3)} = R^{(3)}(\theta^*) - R^{(3)}(\theta^{(3)*}) = -0.708 + 0.708 = 0$	
	$d^{(4)} = R^{(4)}(\theta^*) - R^{(4)}(\theta^{(4)*}) = -0.743 + 0.748 = 0.005$	
XGBoost	$d^{(1)} = R^{(1)}(\theta^*) - R^{(1)}(\theta^{(1)*}) = -0.975 + 0.976 = 0.001$	0.00525
	$d^{(2)} = R^{(2)}(\theta^*) - R^{(2)}(\theta^{(2)*}) = -0.865 + 0.869 = 0.004$	
	$d^{(3)} = R^{(3)}(\theta^*) - R^{(3)}(\theta^{(3)*}) = -0.698 + 0.698 = 0$	
	$d^{(4)} = R^{(4)}(\theta^*) - R^{(4)}(\theta^{(4)*}) = -0.743 + 0.759 = 0.016$	

Tabela 4: Tunowalność dla algorytmów.

3 Tunowalność poszczególnych algorytmów

Z artykułu P. Probst, A. Boulesteix, B. Bischl 1, wiemy, że optymalną konfigurację hiperparametrów dla j-tego zbioru danych można wyznaczyć z

$$\theta^{(j)*} := arg\min_{\theta\in\vartheta} R^{(j)}(\theta), \tag{1}$$

gdzie R jest pewną miarą ryzyka, którą minimalizujemy. Optymalną konfigurację hiperparametrów dla danego algorytmu na m zbiorach danych obliczamy z

$$\theta^* := arg\min_{\theta\in\vartheta} g(R^{(1)}(\theta), ..., R^{(m)}(\theta). \tag{2}$$

Można obliczyć ogólną miarę tunowalności algorytmu dla zbioru danych w oparciu o różnicę między ryzykiem optymalnej konfiguracji referencyjnej a ryzykiem najlepszej możliwej konfiguracji na danym zbiorze danych

$$d^{(j)} := R^{(j)}(\theta^*) - R^{(j)}(\theta^{(j)*}), \;\; dla \;\; j = 1, ..., m \tag{3}$$

W naszych rozważaniach $m = 4$, za g przyjmiemy średnią, a za funkcję ryzyka R na wszystkich zbiorach danych weźmiemy minus dokładność (- *accuracy score*). Biorąc ujemą wartość dokłaności problem wyznaczenia konfiguracji hiperparametrów, która minimalizuje ryzyko jest równoważny szukaniu konfiguracji, która maksymalizuje dokładność.

W tabelach 6, 7, 8 dla każdego z omawianych algorytmów przedstawiono 40 ze 100 konfiguracji hiparametrów, których średia dokładność (po zbiorach danych) wytrenowanego modelu jest największa. Zaznaczono też $\theta^{(j)*}$ dla $j = 1, 2, 3, 4$ oraz optymalne θ^*.
Odczytując ze wspomnianych tabel dokładności uzyskane dla optymalnej konfiguracji hipparametrów dla j-tego zbioru (komórki zaznaczone na czerwono) oraz dokładności uzyskane dla optymalnego θ^* (w pierwszych wierszach) liczymy tunowlność algorytmów zgodnie ze wzorem (3). Wyniki prezentuje Tabela 4 oraz Rysunek 1.

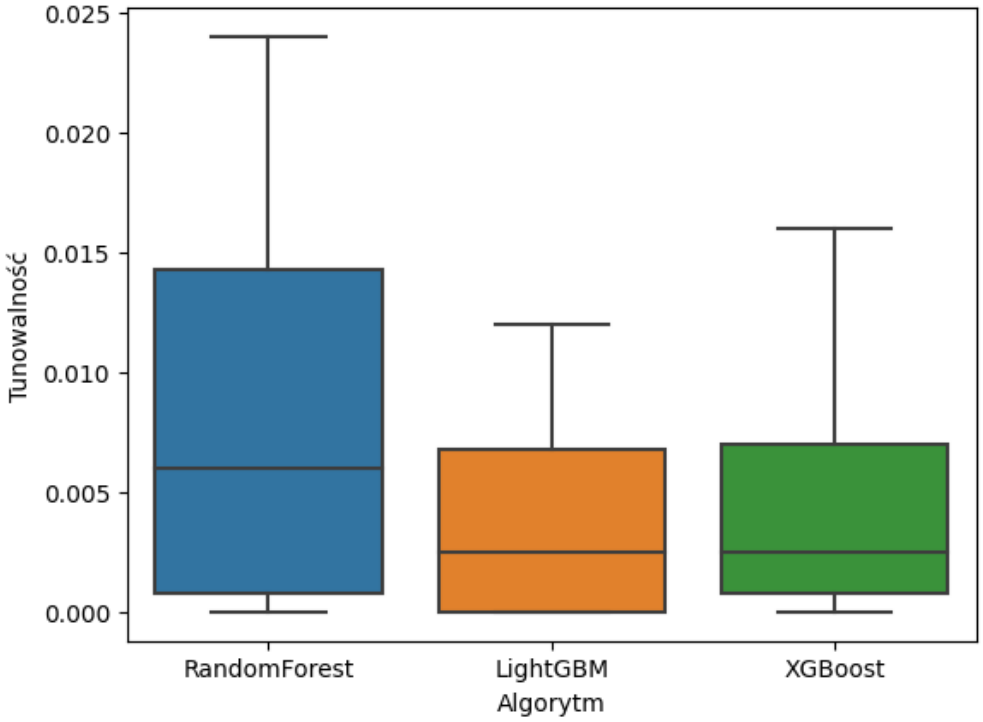
4 Podsumowanie

- Po optymalizacji średnio najlepszą dokładność uzyskuje algorytm LightGBM,
- Jak przewidywałyśmy na początku, Random Forest dawał średnio najgorsze scory,
- Random Forest jest średnio najbardziej tunowalnym algorytmem z rozważanych,
- Dobrze zdefiniowana początkowa siatka parametrów jest istotna, czasami można osiągnąć zbieżność do optymalnej wartości scora już w kilku początkowych krokach,
- Bayes zazwyczaj dawał lepszych accuracy score niż Random Search.
- Proces wyznaczania optymalnej konfiguracji hiperparametrów jest często bardzo czasochłonny.

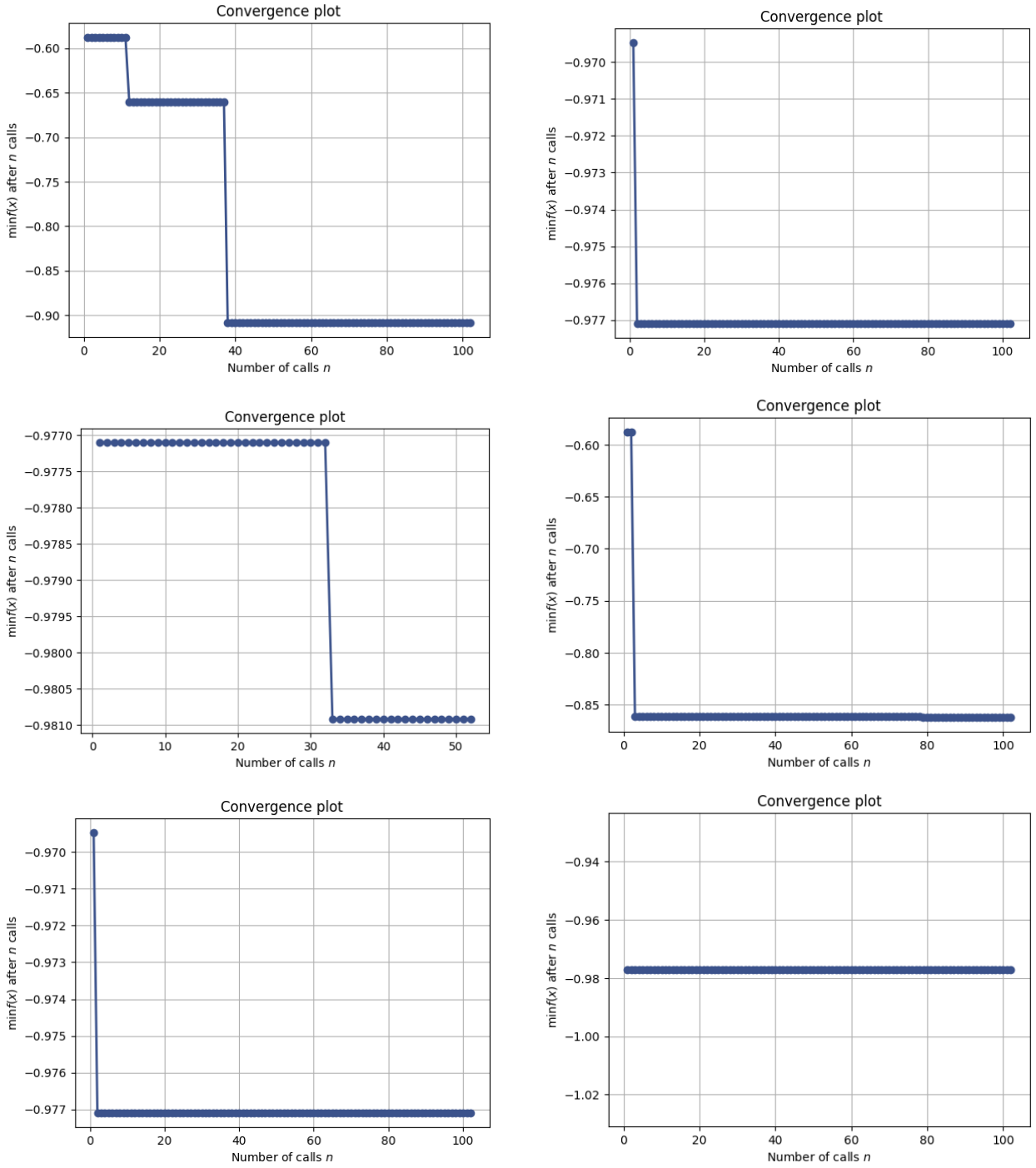
5 Appendix

Dokładność (accuracy score) dla RandomForestClassifier				
	model	score_val	pred_time_val	fit_time
0	LightGBMLarge	0.989130	0.013534	1.705375
1	WeightedEnsemble_L2	0.989130	0.014673	2.638971
2	LightGBM	0.983696	0.011058	1.061787
3	CatBoost	0.983696	0.017301	3.691258
4	XGBoost	0.983696	0.017997	0.908620
5	NeuralNetTorch	0.983696	0.036917	3.777592
6	NeuralNetFastAI	0.978261	0.029801	2.000375
7	RandomForestGini	0.978261	0.104045	1.600873
8	RandomForestEntr	0.978261	0.155280	1.713831
9	LightGBMXT	0.972826	0.029861	1.571203
10	ExtraTreesEntr	0.956522	0.097036	1.212818
11	ExtraTreesGini	0.951087	0.094927	1.172695
12	KNeighborsUnif	0.755435	0.015994	0.023318
13	KNeighborsDist	0.750000	0.018822	0.025105

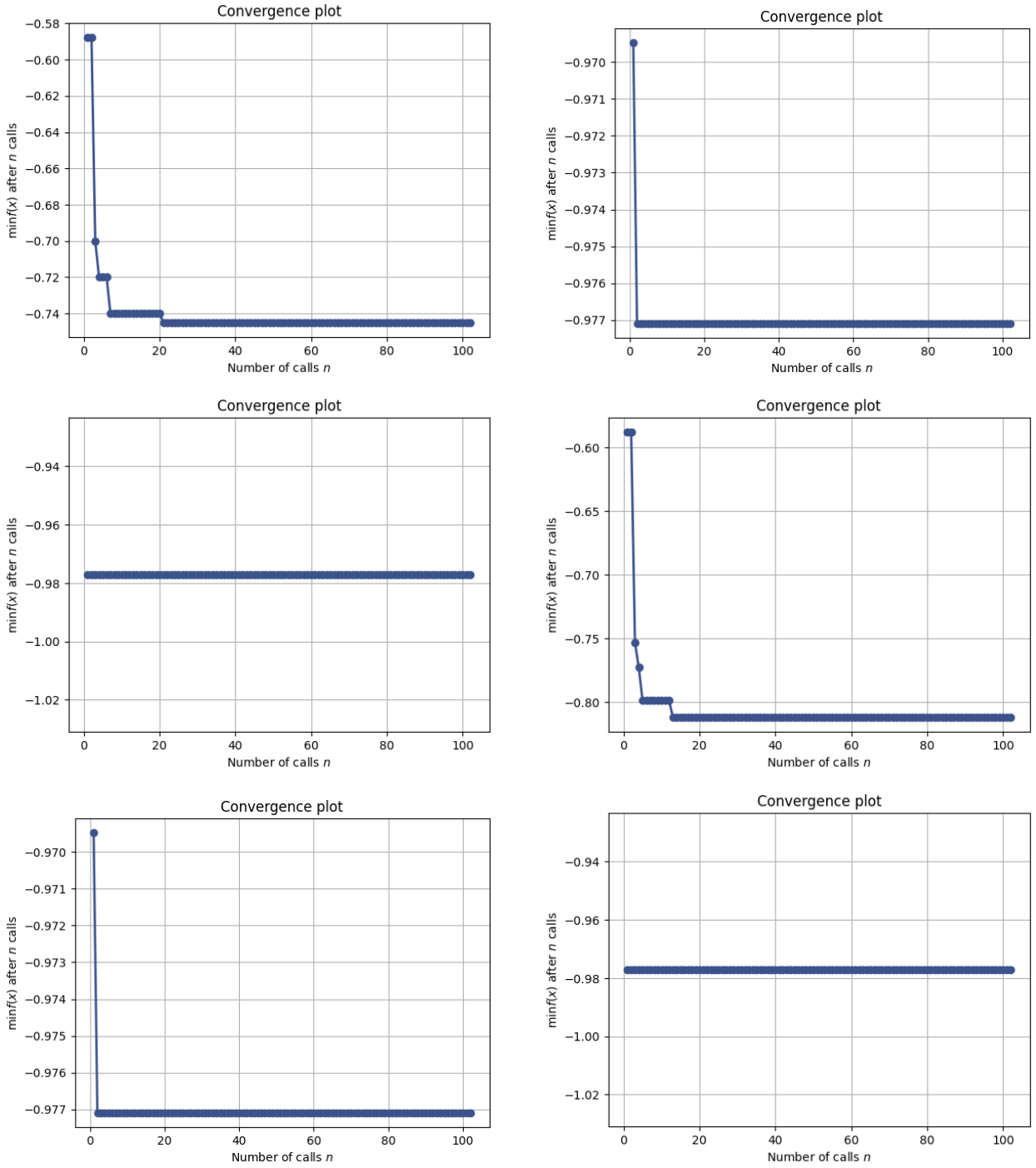
Tabela 5: Zestawienie najlepszych modeli dla domyślnych parametrów według pakietu **autogluon** dla ramki danych Titanic.



Rysunek 1: Boxploty tunowalności algorytmów z uwzglednieniem optymalnej konfiguracji parametrów.



Rysunek 2: Liczba iteracji optymalizacji Bayesowskiej po której uzyskujemy stabilne wyniki (3 pierwsze wykresy ramka danych Titanic, 3 kolejne ramka danych SpeedDating.)



Rysunek 3: Liczba iteracji optymalizacji Bayesowskiej po której uzyskujemy stabilne wyniki (3 pierwsze wykresy ramka danych Credit-g, 3 kolejne ramka danych Diabetes.)

Dokładność (accuracy score) dla RandomForestClassifier					
{'n_estimators', 'min_samples_split', 'min_samples_leaf', 'max_features','max_depth'}	Titanic	SpeedDating	Credit-g	Diabetes	Średnia
{37,8,1,12,18}=$\theta^{(1)*} = \theta^*$	**0,735**	0,864	0,696	0,761	**0,764**
{24,10,1,11,16}	0,714	0,862	0,706	0,736	0,755
{35,12,1,10,12}	0,696	0,862	0,695	0,756	0,752
{21,2,1,11,15}	0,714	0,863	0,678	0,751	0,751
{107,2,1,11,15}=$\theta^{(2)*}$	0,685	**0,865**	0,694	0,744	0,747
{119,2,9,11,8}	0,626	0,861	0,703	0,772	0,740
{109,18,3,11,19}	0,626	0,861	0,706	0,766	0,740
{83,16,3,3,18}	0,626	0,850	0,719	0,762	0,739
{71,8,5,2,13}	0,626	0,844	0,714	0,770	0,738
{71,2,17,6,16}	0,626	0,854	0,706	0,767	0,738
{83,8,9,10,19}	0,626	0,859	0,699	0,767	0,738
{61,14,17,11,16}	0,626	0,859	0,700	0,764	0,737
{65,12,7,9,16}	0,626	0,859	0,700	0,764	0,737
{99,8,15,8,8}	0,626	0,857	0,699	0,767	0,737
{65,8,13,11,12}	0,626	0,860	0,700	0,762	0,737
{96,4,11,3,19}	0,626	0,846	0,708	0,769	0,737
{23,8,13,12,10}	0,626	0,862	0,701	0,757	0,737
{101,18,11,6,13}	0,626	0,855	0,705	0,761	0,736
{34,18,13,10,7}	0,626	0,856	0,704	0,759	0,736
{70,4,17,11,6}	0,626	0,855	0,696	0,767	0,736
{114,4,11,11,12}	0,626	0,862	0,698	0,759	0,736
{46,14,17,12,7}	0,626	0,857	0,703	0,759	0,736
{57,6,3,4,10}	0,626	0,851	0,700	0,767	0,736
{70,4,5,12,13}	0,626	0,864	0,695	0,759	0,736
{101,12,19,8,17}	0,626	0,857	0,695	0,766	0,736
{88,18,1,7,8}	0,626	0,855	0,710	0,753	0,736
{54,16,9,11,12}	0,626	0,861	0,698	0,759	0,736
{82,14,19,10,7}	0,626	0,856	0,699	0,762	0,736
{32,14,17,8,15}	0,626	0,856	0,696	0,764	0,736
{27,2,19,11,17}	0,626	0,858	0,699	0,759	0,735
{93,6,15,5,9}	0,626	0,850	0,704	0,761	0,735
{83,12,1,4,9}	0,626	0,849	0,701	0,764	0,735
{100,16,7,4,6}	0,626	0,839	0,706	0,769	0,735
{41,8,5,8,16}	0,626	0,860	0,705	0,749	0,735
{99,4,5,7,11}	0,626	0,860	0,701	0,753	0,735
{45,4,3,2,8}=$\theta^{(3)*}$	0,626	0,840	**0,720**	0,754	0,735
{14,16,1,8,6}	0,628	0,851	0,713	0,748	0,735
{110,12,15,4,12}	0,626	0,846	0,708	0,759	0,735
{101,10,9,8,5}	0,626	0,847	0,700	0,766	0,735
{50,6,7,4,10}=$\theta^{(4)*}$	0,626	0,849	0,691	**0,772**	0,735

Tabela 6: Dokładność (miara accuracy score) dla ramek 1-4 przy użyciu RandomForestClassifier.

Dokładność (accuracy score) dla LGBMClassifier					
{num_leaves', 'n_estimators', learning_rate'}	Titanic	SpeedDating	Credit-g	Diabetes	Średnia
{12, 67, 0.04}=$\theta^{(3)*} = \theta^*$	0,975	0,863	**0,708**	0,743	**0,822**
{10, 141, 0.03}=$\theta^{(4)*}$	0,975	0,865	0,698	**0,748**	0,821
{27, 91, 0.02}	0,975	0,863	0,701	0,741	0,820
{25, 123, 0.02}	0,975	0,864	0,701	0,738	0,819
{35, 189, 0.01}	0,975	0,862	0,704	0,736	0,819
{24, 84, 0.02}	0,975	0,859	0,700	0,741	0,819
{13, 90, 0.069}	0,975	0,870	0,696	0,735	0,819
{28, 271, 0.01}	0,975	0,865	0,695	0,739	0,819
{25, 58, 0.05}	0,975	0,865	0,694	0,740	0,818
{14, 213, 0.02}	0,975	0,867	0,691	0,740	0,818
{38, 130, 0.01}	0,975	0,858	0,704	0,731	0,817
{15, 74, 0.03}=$\theta^{(1)*}$	**0,975**	0,862	0,696	0,733	0,817
{18, 79, 0.05}	0,975	0,867	0,691	0,733	0,817
{12, 67, 0.12}	0,973	0,867	0,684	0,740	0,816
{28, 67, 0.02}	0,975	0,856	0,700	0,731	0,816
{37, 66, 0.069}	0,974	0,866	0,688	0,733	0,815
{24, 104, 0.04}	0,975	0,869	0,684	0,726	0,814
{38, 90, 0.04}	0,975	0,870	0,685	0,723	0,813
{24, 62, 0.099}	0,968	0,872	0,675	0,733	0,812
{18, 50, 0.12}	0,972	0,872	0,671	0,730	0,811
{17, 67, 0.11}	0,969	0,872	0,680	0,723	0,811
{35, 71, 0.16}	0,969	0,867	0,665	0,743	0,811
{19, 69, 0.12}	0,966	0,871	0,669	0,735	0,810
{36, 272, 0.06}=$\theta^{(2)*}$	0,967	**0,875**	0,664	0,735	0,810
{12, 220, 0.069}	0,968	0,871	0,663	0,738	0,810
{29, 88, 0.08}	0,968	0,871	0,666	0,733	0,810
{29, 137, 0.05}	0,967	0,869	0,670	0,733	0,810
{25, 228, 0.03}	0,968	0,871	0,673	0,725	0,809
{36, 215, 0.26}	0,968	0,871	0,656	0,740	0,809
{38, 188, 0.26}	0,967	0,873	0,655	0,740	0,808
{15, 142, 0.08}	0,967	0,871	0,666	0,728	0,808
{20, 138, 0.19}	0,968	0,868	0,653	0,739	0,807
{15, 98, 0.18}	0,968	0,866	0,666	0,726	0,807
{35, 150, 0.16}	0,970	0,873	0,654	0,730	0,807
{36, 250, 0.05}	0,968	0,870	0,661	0,726	0,807
{20, 234, 0.04}	0,967	0,871	0,669	0,720	0,807
{13, 121, 0.14}	0,966	0,869	0,663	0,728	0,806
{39, 184, 0.13}	0,968	0,872	0,653	0,733	0,806
{36, 99, 0.11}	0,968	0,870	0,663	0,725	0,806
{36, 79, 0.26}	0,967	0,867	0,663	0,728	0,806

Tabela 7: Dokładność (miara accuracy score) dla ramek 1-4 przy użyciu LightGBM.

Dokładność (accuracy score) dla XGBClassifier					
{'subsample', 'min_child_weight', 'max_depth', 'eta','colsample_bytree'}	Titanic	SpeedDating	Credit-g	Diabetes	Średnia
{0.3, 4.0, 7, 0.1, 0.9}=$\theta^{(3)*} = \theta^*$	0,975	0,865	**0,698**	0,743	**0,820**
{0.4, 1.0, 7, 0.1, 0.6}=$\theta^{(2)*}$	0,975	**0,869**	0,690	0,744	0,820
{0.7, 4.5, 5, 0.2, 0.6}	0,975	0,868	0,681	0,744	0,817
{0.5, 3.0, 8, 0.1, 0.4}	0,975	0,866	0,671	0,752	0,816
{0.4, 5.0, 3, 0.3, 0.9}	0,975	0,860	0,685	0,738	0,814
{0.9, 4.0, 7, 0.1, 0.8}	0,975	0,867	0,679	0,733	0,814
{0.6, 4.0, 7, 0.2, 0.7}	0,975	0,860	0,674	0,744	0,813
{0.9, 0.0, 3, 0.5, 0.4}	0,976	0,863	0,675	0,739	0,813
{0.6, 5.0, 8, 0.2, 0.7}	0,975	0,865	0,678	0,735	0,813
{0.4, 5.5, 7, 0.3, 0.1}	0,975	0,850	0,678	0,746	0,812
{0.7, 3.5, 3, 0.3, 0.3}	0,975	0,865	0,671	0,733	0,811
{0.4, 6.5, 9, 0.4, 0.7}	0,975	0,848	0,691	0,728	0,811
{0.2, 0.0, 6, 0.4, 0.8}=$\theta^{(4)*}$	0,974	0,833	0,666	**0,759**	0,808
{0.7, 0.5, 5, 0.4, 0.4}	0,975	0,861	0,659	0,728	0,806
{0.9, 4.0, 3, 0.6, 0.3}	0,975	0,863	0,669	0,712	0,805
{0.8, 4.5, 6, 0.4, 0.9}	0,975	0,861	0,659	0,722	0,804
{0.5, 6.0, 5, 0.4, 0.8}	0,975	0,852	0,656	0,733	0,804
{0.8, 1.5, 8, 0.2, 0.1}	0,975	0,860	0,664	0,712	0,803
{0.5, 4.0, 9, 0.6, 0.2}	0,975	0,838	0,660	0,736	0,802
{0.1, 3.5, 5, 0.3, 0.9}	0,975	0,829	0,669	0,730	0,801
{0.9, 6.0, 7, 0.8, 0.7}	0,975	0,856	0,655	0,707	0,798
{0.6, 4.5, 9, 0.6, 0.8}	0,975	0,853	0,659	0,705	0,798
{0.6, 2.5, 6, 0.3, 0.2}	0,975	0,856	0,655	0,705	0,798
{0.1, 4.0, 9, 0.5, 0.8}	0,975	0,806	0,656	0,748	0,796
{0.2, 2.5, 3, 0.4, 0.2}	0,975	0,835	0,651	0,723	0,796
{0.7, 5.0, 6, 0.7, 0.1}	0,975	0,843	0,651	0,713	0,796
{0.4, 6.5, 4, 0.7, 0.8}	0,975	0,830	0,653	0,723	0,795
{0.5, 2.0, 9, 0.6, 0.3}	0,975	0,846	0,651	0,704	0,794
{0.6, 3.5, 5, 0.7, 0.3}	0,975	0,841	0,664	0,695	0,794
{0.3, 3.5, 4, 0.5, 0.3}	0,975	0,836	0,664	0,699	0,793
{0.8, 2.0, 7, 0.5, 0.9}	0,973	0,866	0,644	0,687	0,793
{0.7, 2.0, 7, 0.9, 0.7}	0,970	0,849	0,649	0,700	0,792
{0.2, 2.0, 8, 0.4, 0.1}	0,975	0,827	0,650	0,715	0,792
{0.8, 0.0, 8, 0.7, 0.2}	0,976	0,857	0,646	0,686	0,791
{0.3, 5.5, 8, 0.7, 0.2}	0,975	0,819	0,638	0,730	0,790
{0.6, 3.5, 4, 0.8, 0.4}	0,975	0,835	0,640	0,705	0,789
{0.6, 0.0, 6, 0.7, 0.2}=$\theta^{(1)*}$	**0,976**	0,851	0,626	0,699	0,788
{0.1, 6.0, 4, 0.6, 0.3}	0,975	0,812	0,634	0,708	0,782
{0.1, 5.0, 3, 0.9, 0.1}	0,975	0,804	0,615	0,708	0,776
{0.3, 3.5, 3, 0.8, 0.8}	0,975	0,814	0,635	0,666	0,773

Tabela 8: Dokładność (miara accuracy score) dla ramek 1-4 przy użyciu XGBoost.

Literatura

[1] Philipp Probst, Anne-Laure Boulesteix, Bernd Bischl *Tunability: Importance of Hyperparameters of Machine Learning Algorithms*, Journal of Machine Learning Research 20 (2019) 1-32