Analiza tunowalności hiperparametrów

Autorzy: Damian Lubaszka, Mikołaj Nowak

Zbiory danych

- **Założenia przy wyborze zbioru danych:**
  - Zbiory mają pochodzić ze strony open.ml,
  - Zbiory mają być dedykowane na problem klasyfikacji binarnej,
  - Klasa target dla każdego zbioru musi musi mieć rozkład jednostajny,
  - Liczba Features musi być mniejsza niż 20,
  - Liczba danych musi się mieścić w zakresie 10000-20000 rekordów.

- **Wybrane zbiory danych:**
  - MagicTelescope,
  - online-shoppers-intention,
  - bank-marketing,
  - credit

Algorytmy uczenia maszynowego

- Random Forest Classifier
- Gradient Boosting Classifier
- SVM Classifier

Algorytm uczenia maszynowego	Hyperparametr	dolny zakres	górny zakres
Random Forest Classifier	n_estimators	10	230
Random Forest Classifier	max_depth	1	15
Random Forest Classifier	min_samples_split	1	15
Random Forest Classifier	min_samples_leaf	1	10
Random Forest Classifier	max_features	['sqrt', 'log2']	
Gradient Boosting Classifier	n_estimators	10	150
Gradient Boosting Classifier	max_depth	1	8
Gradient Boosting Classifier	min_samples_split	1	10
Gradient Boosting Classifier	min_samples_leaf	1	5
Gradient Boosting Classifier	learning_rate	0.05	0.5
Gradient Boosting Classifier	subsample	0.75	1.0
SVM Classifier	C	0.01	10.0
SVM Classifier	kernel	['linear', 'rbf', 'sigmoid']	
SVM Classifier	gamma	0.001	6.5

Metody samplingu

- **Random Search** - jako przykład metody opierającej się na wyborze punktów z rozkładu jednostajnego,
- **Bayes Optimization** - jako przykład metody opierającej się na technice bayesowskiej.

Użyte pakiety

- **RandomizedSearchCV** z sklearn
- **HyperParameterOptimizationFacade** z smac

Liczba iteracji i średni czas wykonania

Algorytm uczenia maszynowego	Ilość iteracji dla Random Search	Ilość iteracji dla Bayesian Optimization	Czas obliczeń dla Random Search	Czas obliczeń dla Bayesian Optimization
Random Forest Classifier	450	150	57min	102min
Gradient Boosting Classifier	300	150	67min	110min
SVM Classifier	300	150	87min	149min

Ciekawostka:
Dla SVM uruchomiliśmy program dla 500/500 i nie wiele się poprawiło, co ma swoje uzasadnienie kilka slajdów dalej...

Znalezione optymalne hiperparametry

Metoda samplingu	n_estimators	max_depth	min_samples_split
Random Search	153	11	2
Bayes Optimization	227	12	4
Metoda samplingu	**min_samples_leaf**	**max_futures**	
Random Search	4	log2	
Bayes Optimization	3	log2	

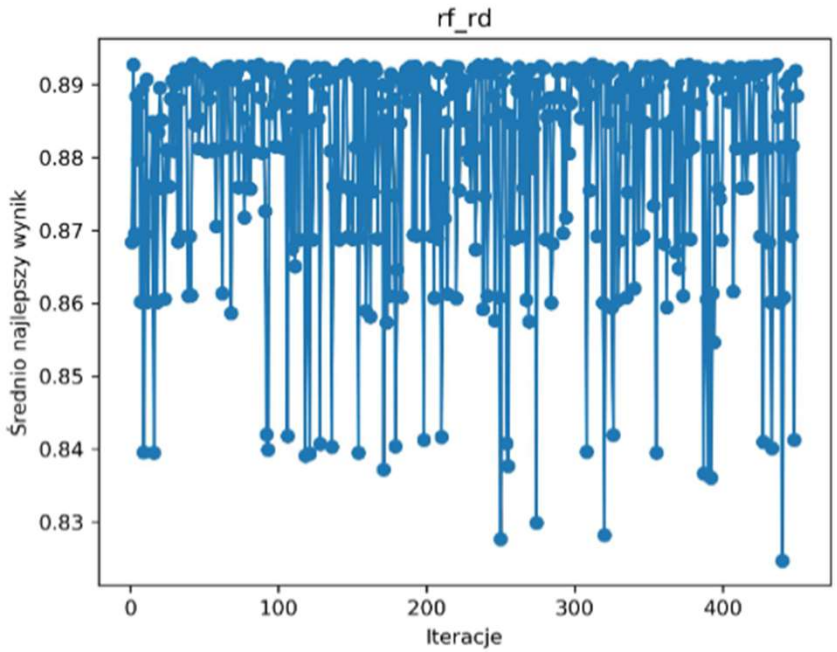
Metoda samplingu	n_estimators	max_depth	min_samples_split
Random Search	116	4	6
Bayes Optimization	141	5	5
Metoda samplingu	**min_samples_leaf**	**learning_rate**	**subsample**
Random Search	3	0.105102	0.775510
Bayes Optimization	5	0.051132	0.751077

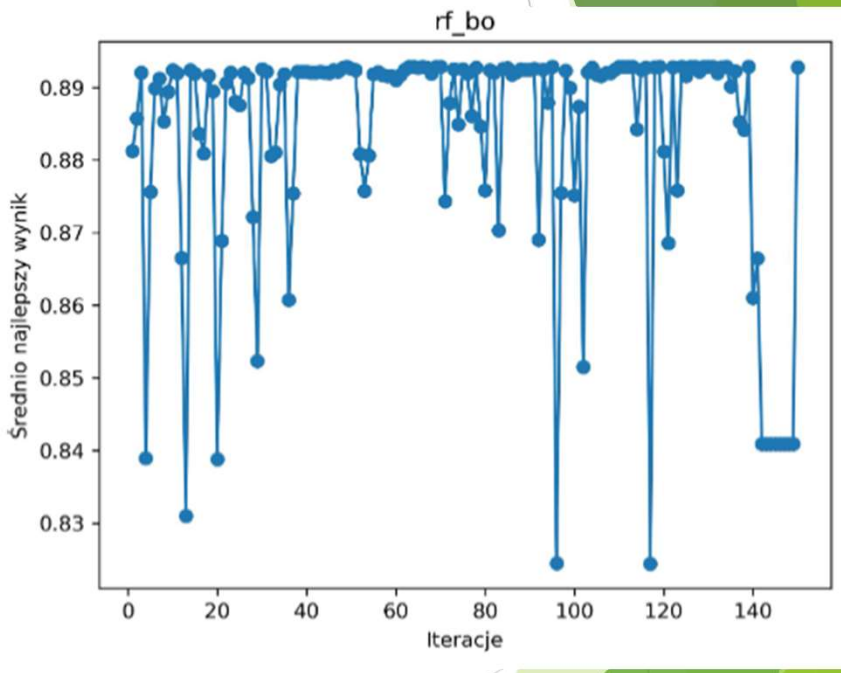
Metoda samplingu	C	kernel	gamma
Random Search	9.388367	rbf	3.316816
Bayes Optimization	9.08872	rbf	3.708863

Jak działał model na danych testowych

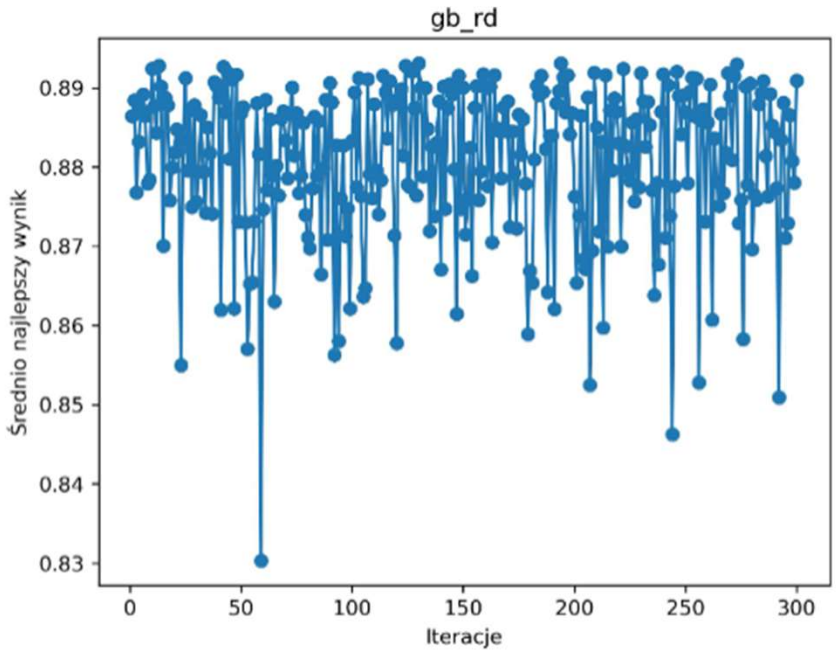
Algorytm	Random Search	Bayes Optimization	Default
Random Forest	0.8912434977360744	0.8921311162696635	0.8869180357772917
Gradient Boosting	0.8981076691283966	0.8986140892075709	0.8956905999452571
SVM	0.8667340916601738	0.8665752806450856	0.8599154568972678

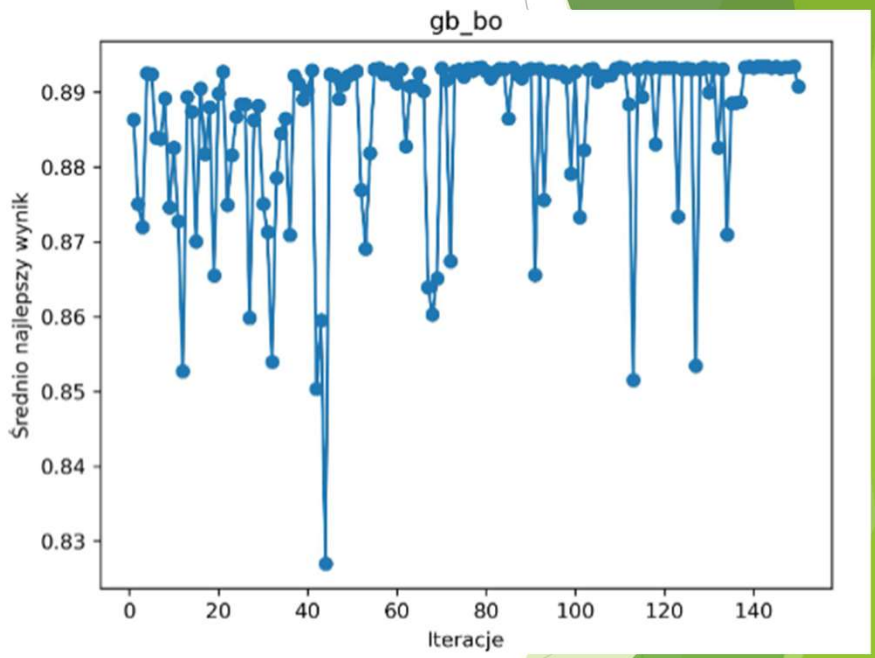
Stabilność wyników



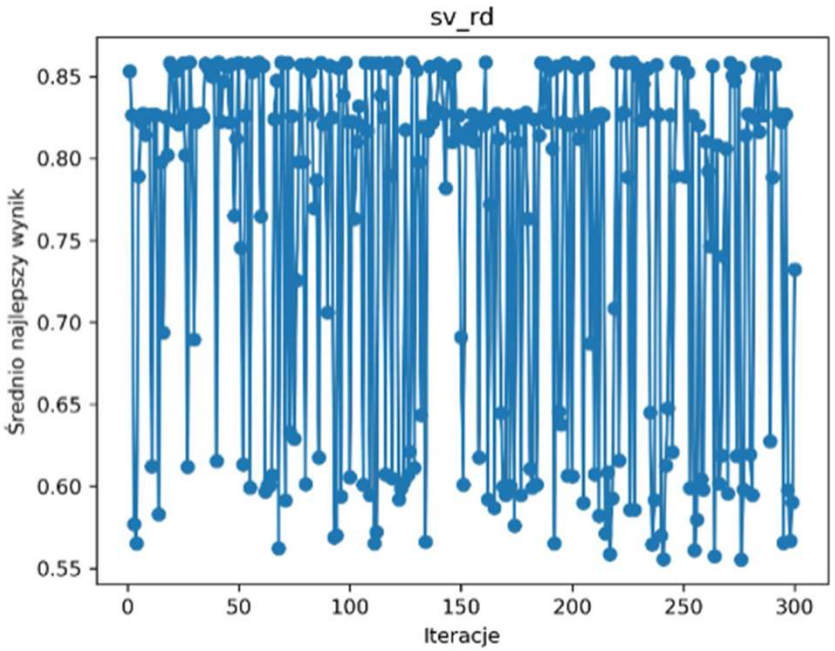


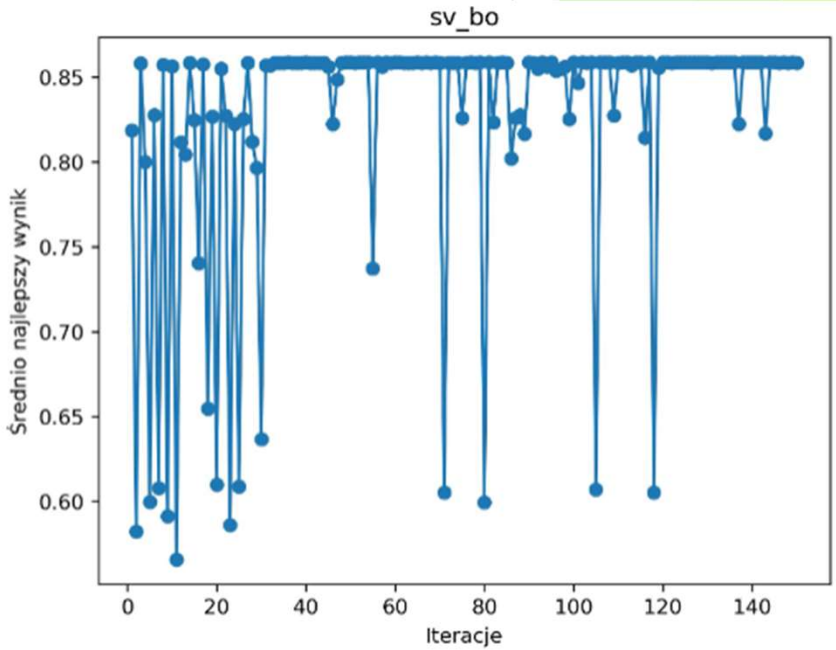
Stabilność wyników



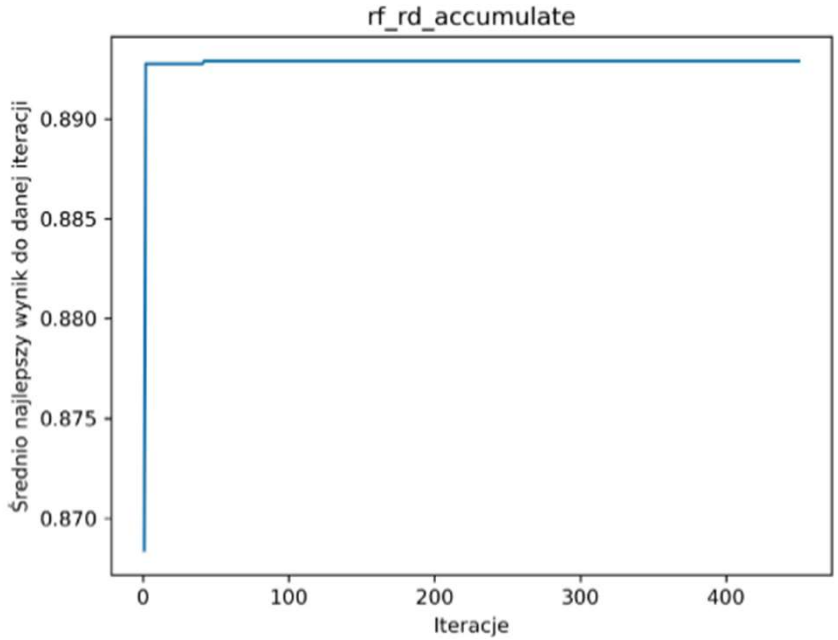


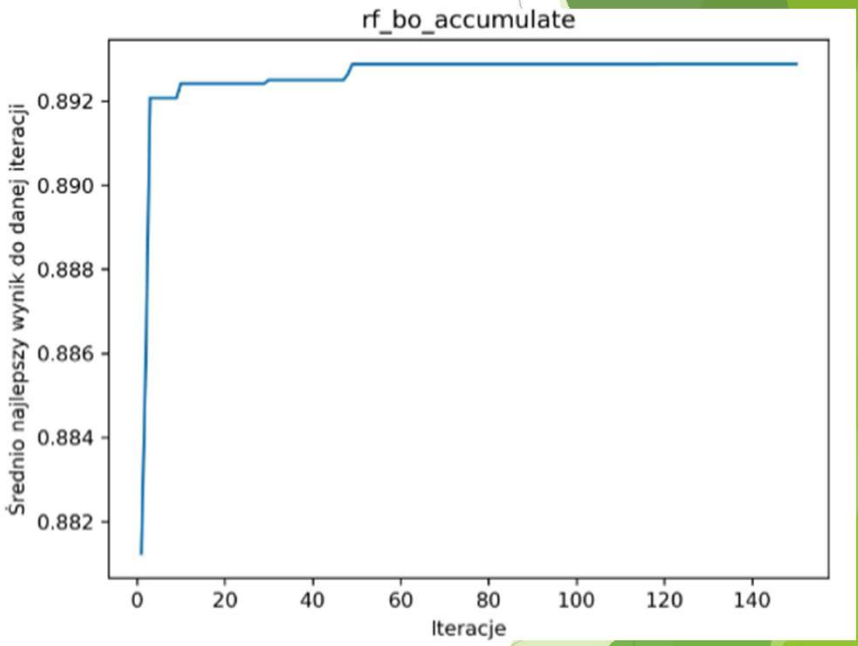
Stabilność wyników



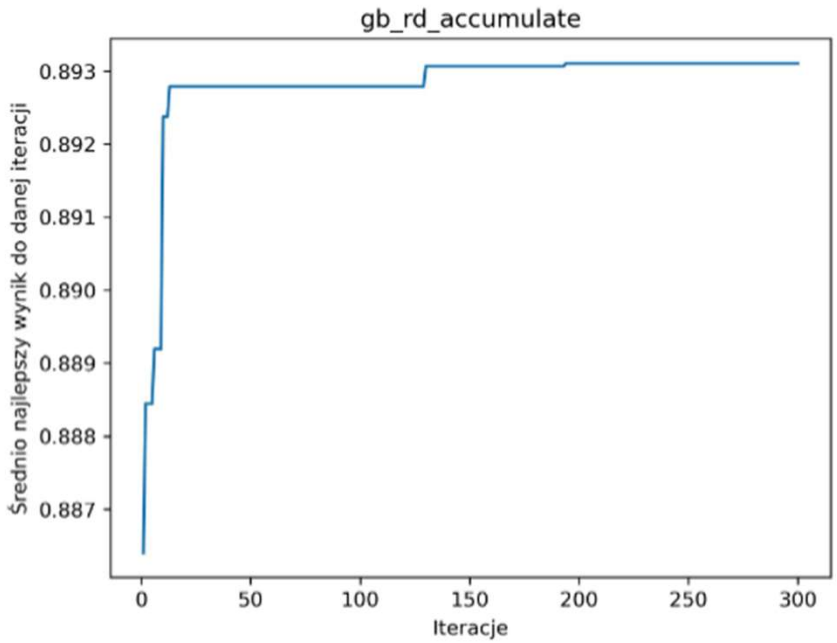


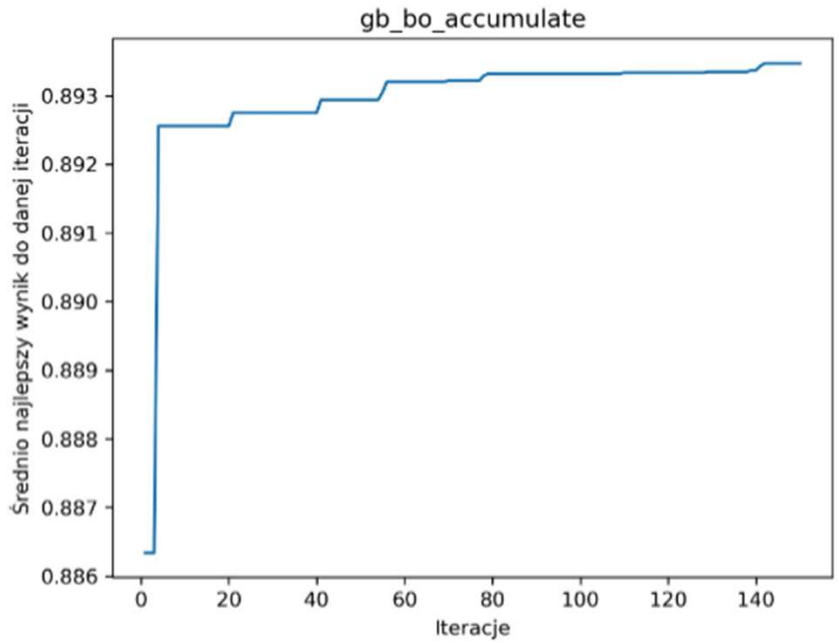
Kiedy zbieżność?



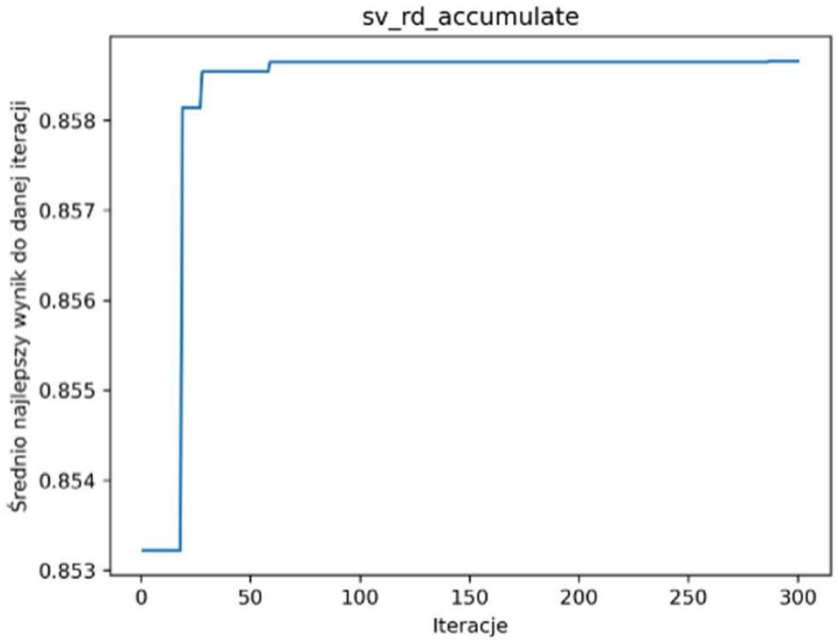


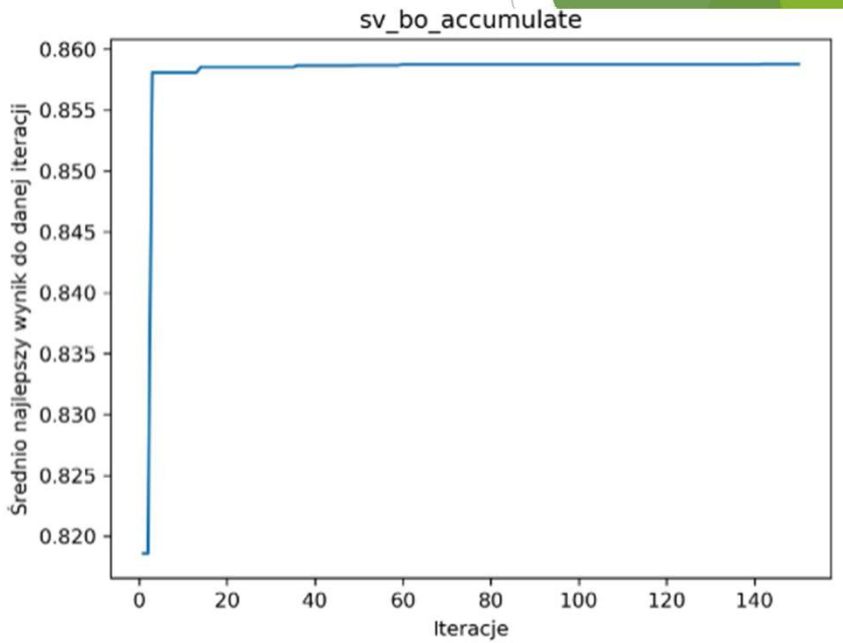
Kiedy zbieżność?



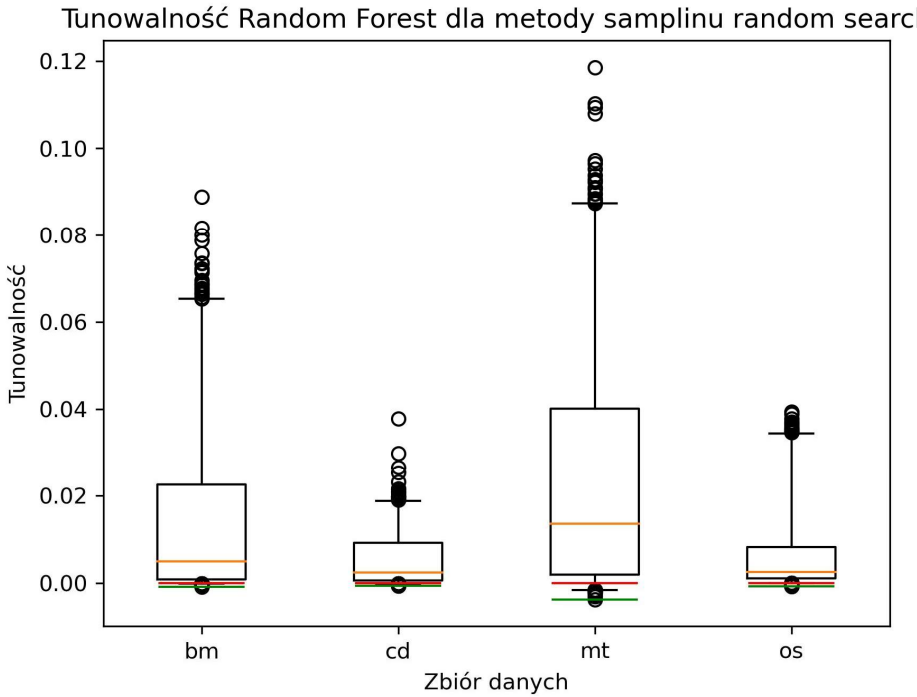


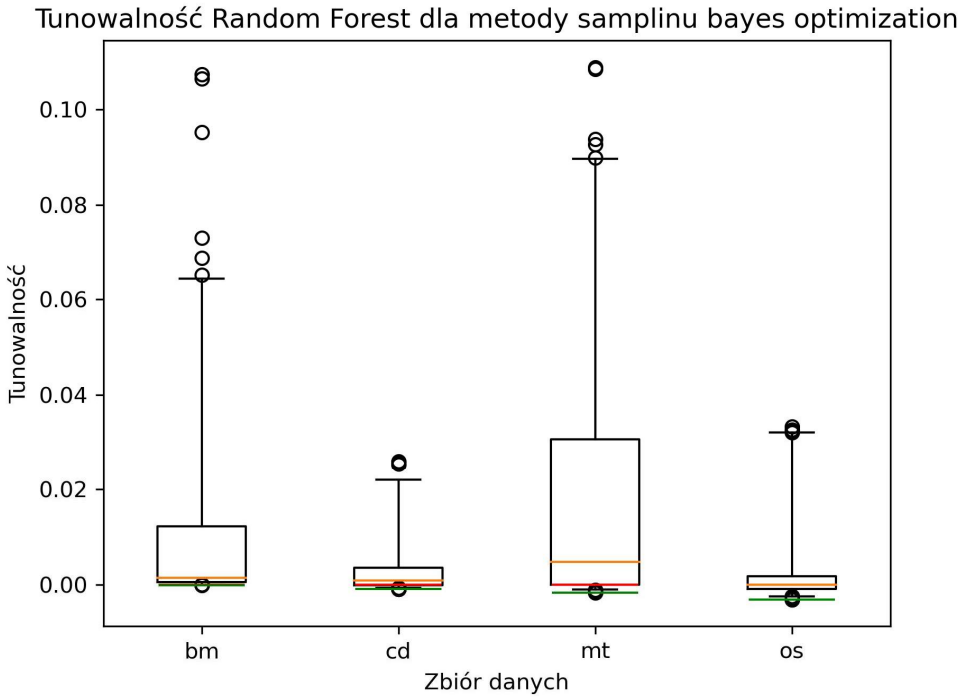
Kiedy zbieżność?



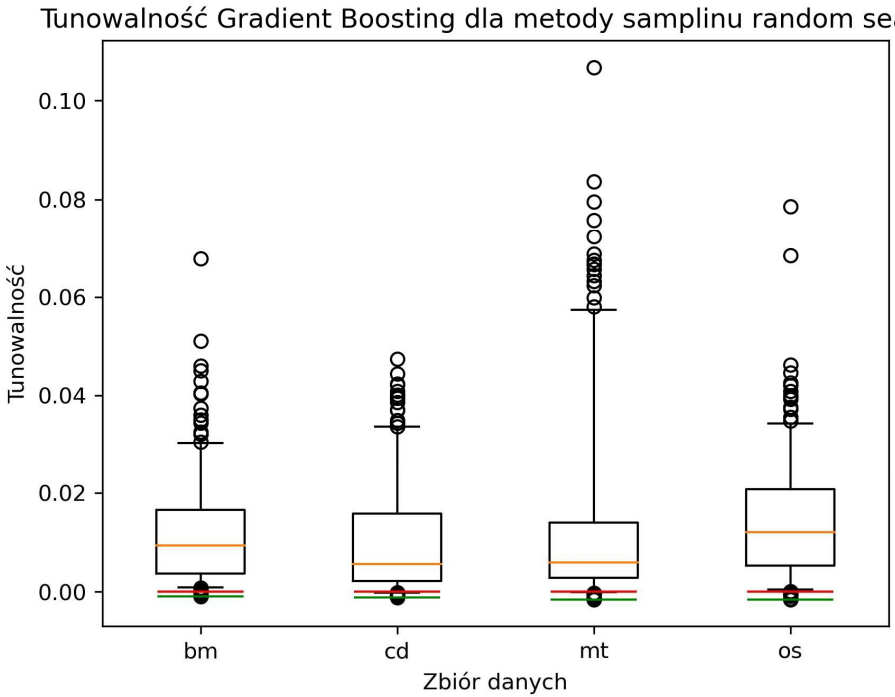


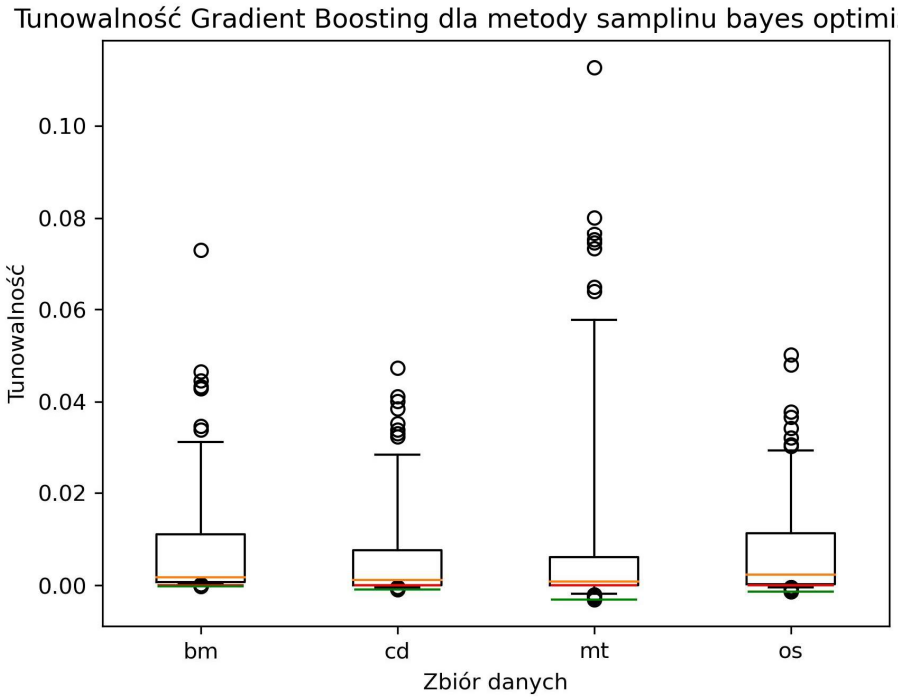
Tunowalność Algorytmów



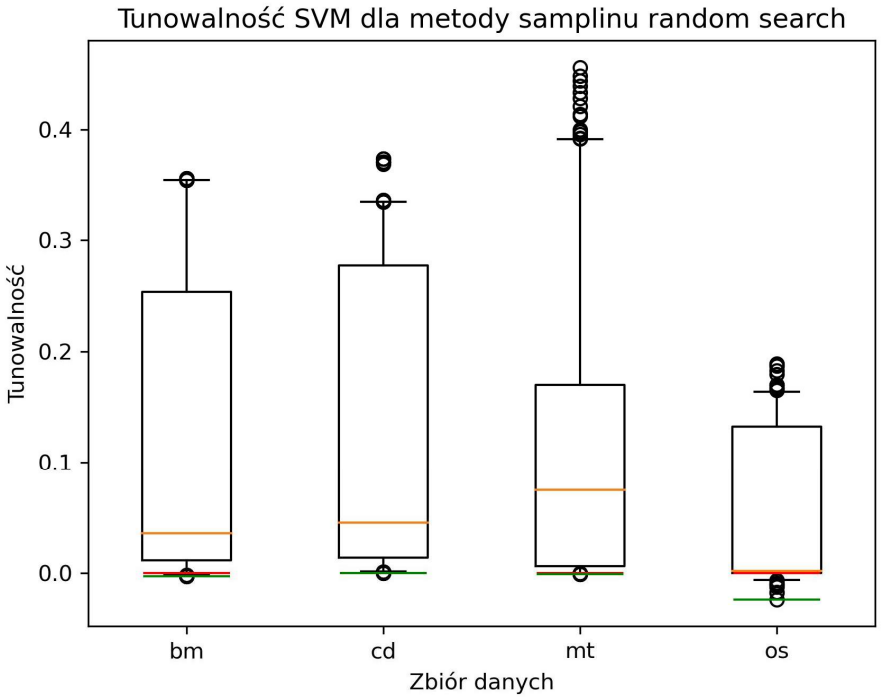


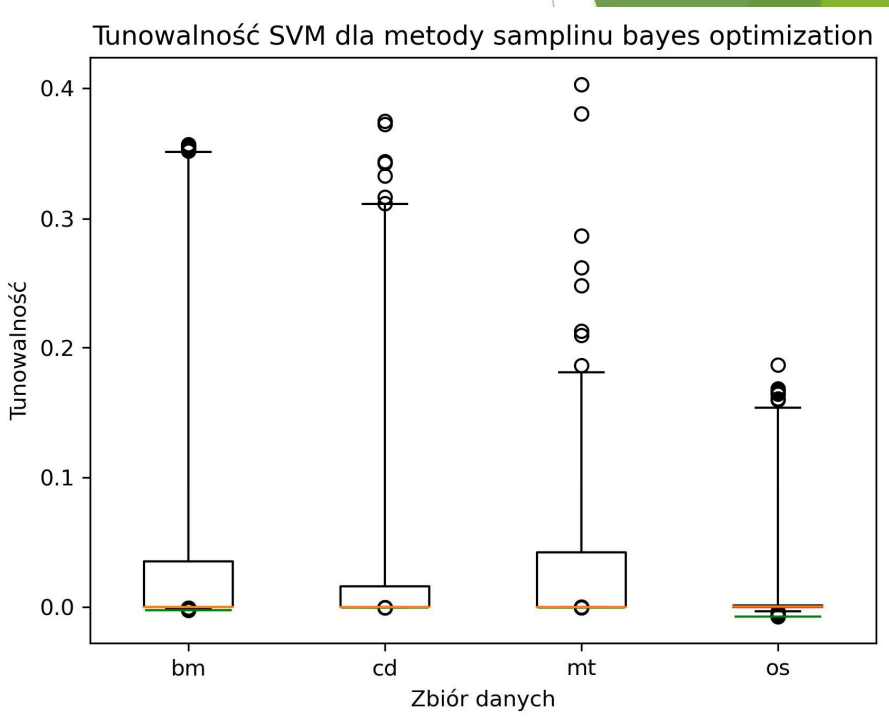
Tunowalność Algorytmów





Tunowalność Algorytmów





Tunowalność algorytmów

Algorytm	Random Search	Bayes Optimization
Random Forest	0.01250521997404487 7	0.00951204658035932 5
Gradient Boosting	0.01245747204463613 7	0.00765225734294827 8
SVM	0.10194249009495136	0.03233157903915992 4

Czy występuję Bias Sampling?

Dziękujemy za uwagę!

W celu analizy związków poszczególnych wartości tunowalności dla poszczególnych modeli, dla poszczególnych algorytmów i dla poszczególnych datasetów w oparciu o wcześniej pokazane boxploty należy użyć:

d_rf_rd_bm: mean: 0.014637162957257963 d_rf_rd_cd: mean: 0.005195224735522472
d_rf_rd_mt: mean: 0.023790088034786466 d_rf_rd_os: mean: 0.006398404168612613
d_rf_rd_all_datasets: mean: 0.012505219974044877 d_rf_bo_bm: mean:
0.012054188442511417 d_rf_bo_cd: mean: 0.0037774173658260955 d_rf_bo_mt: mean:
0.018674274475710656 d_rf_bo_os: mean: 0.003542306037389132 d_rf_bo_all_datasets:
mean: 0.009512046580359325 d_gb_rd_bm: mean: 0.011942517347072015 d_gb_rd_cd:
mean: 0.01032670428106464 d_gb_rd_mt: mean: 0.013444204644640998 d_gb_rd_os: mean:
0.014116461905766894 d_gb_rd_all_datasets: mean: 0.012457472044636137 d_gb_bo_bm:
mean: 0.007850805620328142 d_gb_bo_cd: mean: 0.0061690786852726465 d_gb_bo_mt:
mean: 0.008792356531272812 d_gb_bo_os: mean: 0.007796788534919513
d_gb_bo_all_datasets: mean: 0.007652257342948278 d_sv_rd_bm: mean:
0.11602077823340755 d_sv_rd_cd: mean: 0.12293660312340605 d_sv_rd_mt: mean:
0.12111411748983994 d_sv_rd_os: mean: 0.04769846153315187 d_sv_rd_all_datasets:
mean: 0.10194249009495136 d_sv_bo_bm: mean: 0.03838511695913659 d_sv_bo_cd: mean:
0.04041853876622925 d_sv_bo_mt: mean: 0.03555767160829614 d_sv_bo_os: mean:
0.01496498882297769 d_sv_bo_all_datasets: mean: 0.032331579039159924

Linki:

- https://www.openml.org/search?type=data&status=active&qualities.NumberOfInstances=between_10000_100000&qualities.NumberOfClasses=%3D_2&sort=runs&id=44116
- https://www.openml.org/search?type=data&status=active&qualities.NumberOfInstances=between_10000_100000&qualities.NumberOfClasses=%3D_2&sort=runs&id=45560
- https://www.openml.org/search?type=data&status=active&qualities.NumberOfInstances=between_10000_100000&qualities.NumberOfClasses=%3D_2&sort=runs&id=44126
- https://www.openml.org/search?type=data&status=active&qualities.NumberOfInstances=between_10000_100000&qualities.NumberOfClasses=%3D_2&sort=runs&id=45024
- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
- https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html
- https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html
- https://jmlr.org/papers/volume20/18-444/18-444.pdf
- https://www.analyticsvidhya.com/blog/2020/03/beginners-guide-random-forest-hyperparameter-tuning/
- https://www.analyticsvidhya.com/blog/2016/02/complete-guide-parameter-tuning-gradient-boosting-gbm-python/
- https://towardsdatascience.com/hyperparameter-tuning-for-support-vector-machines-c-and-gamma-parameters-6a5097416167