Modele predykcji cen nieruchomości

Metody wyboru hiperparametórw

Hiperparpetry dla każdego modelu wybieraliśmy metodą random search

Ocena modeli

Do wyboru najlepszych modeli posłużyliśmy się metryką RMSE czyli pierwiastkiem błędu kwadratowego

Poza RMSE obliczaliśmy także metrykę MAE oraz sporządziliśmy wykres pokazujący jak bardzo różniła się predykcja od wartości z obserwacji dla każdej z nich

Random Forest

Problem z prepocessingiem

Pomimo skorzystania z funkcji preprocessingu, dane i tak nie były w formacie jaki przyjmowała funkcja budująca model. Teoretycznie funkcja preprocessing zawiera funkcję usuwającą wartości NA, jednak i tak mimo to trzeba było ręcznie się usunąć kilku kolumn które wciąż zawierały takie dane.

Random Forest - model pierwszy

Hiperparametry:

- num.trees = 600
- max.depth = 3
- min.node.size = 7
- splitrule = "extratrees",

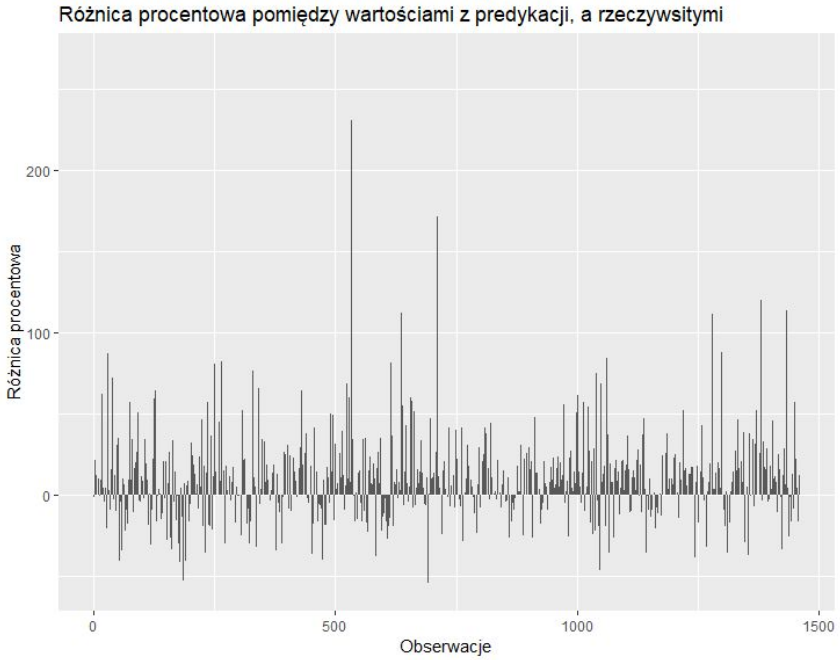
Treningowy:

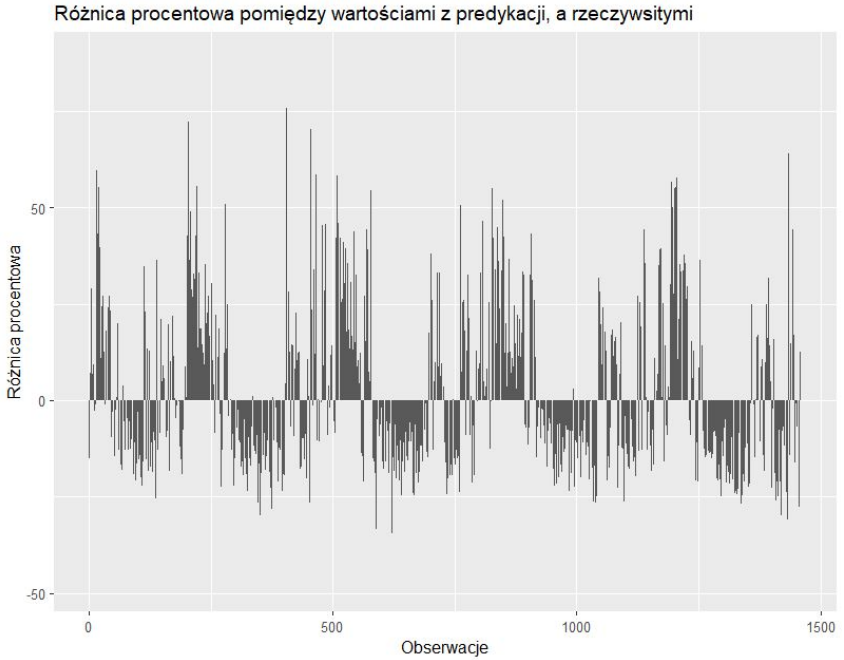
- RMSE = 49542.65
- MAE = 33107.8

Testowy:

- RMSE = 38349.32
- MAE = 31416.25

Random Forest - model pierwszy





Random Forest - model drugi

Hiperparametry:

- num.trees = 700
- max.depth = 4
- min.node.size = 3
- splitrule = "extratrees",

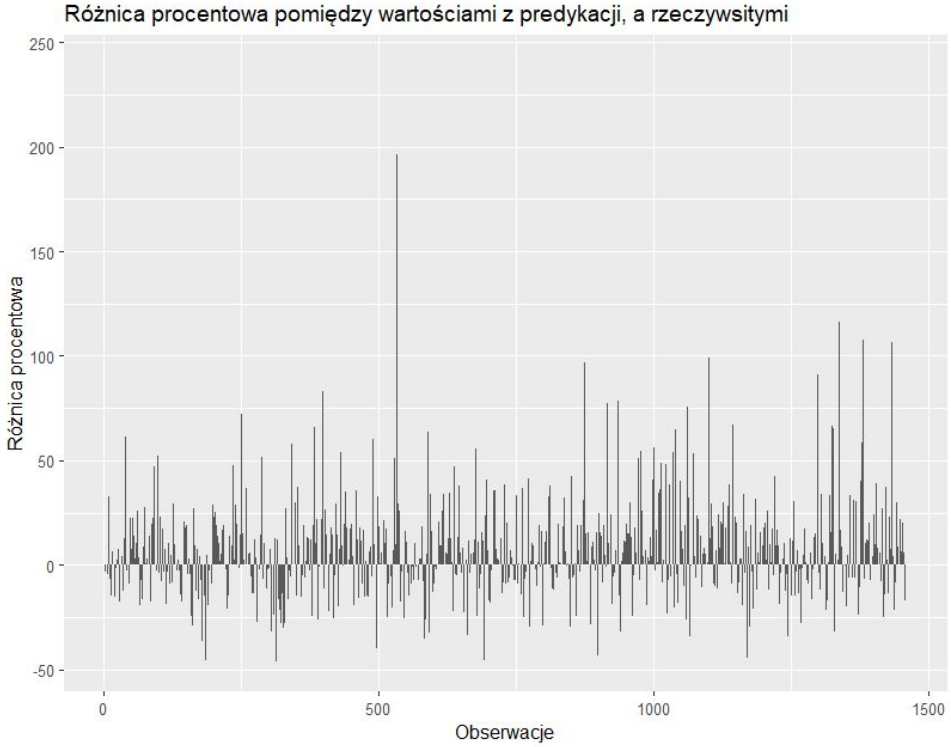
Treningowy:

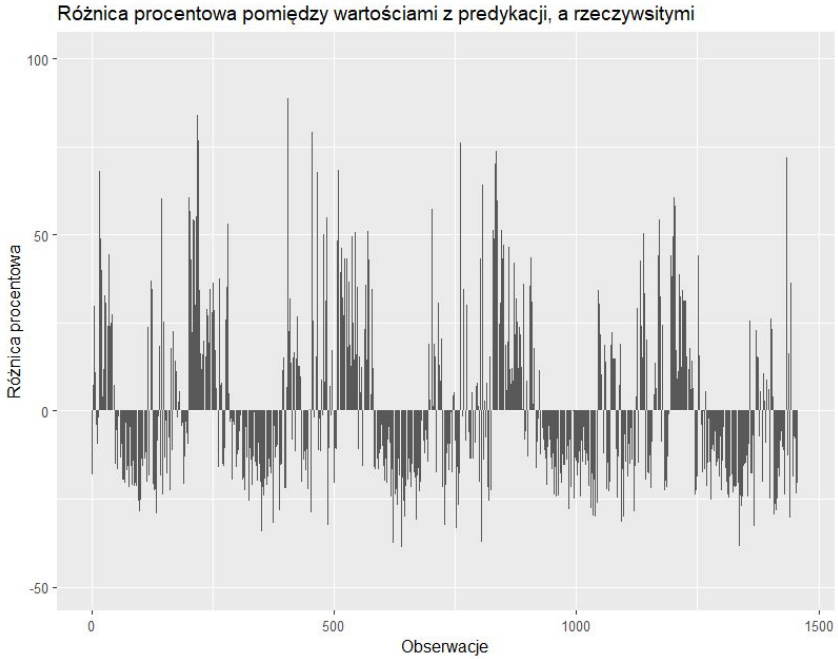
- RMSE = 43751.19
- MAE = 29260.3

Testowy:

- RMSE = 43696.35
- MAE = 35905.39

Random Forest - model drugi





Random Forest - model trzeci

Hiperparametry:

- num.trees = 700
- max.depth = 5
- min.node.size = 3
- splitrule = "extratrees",

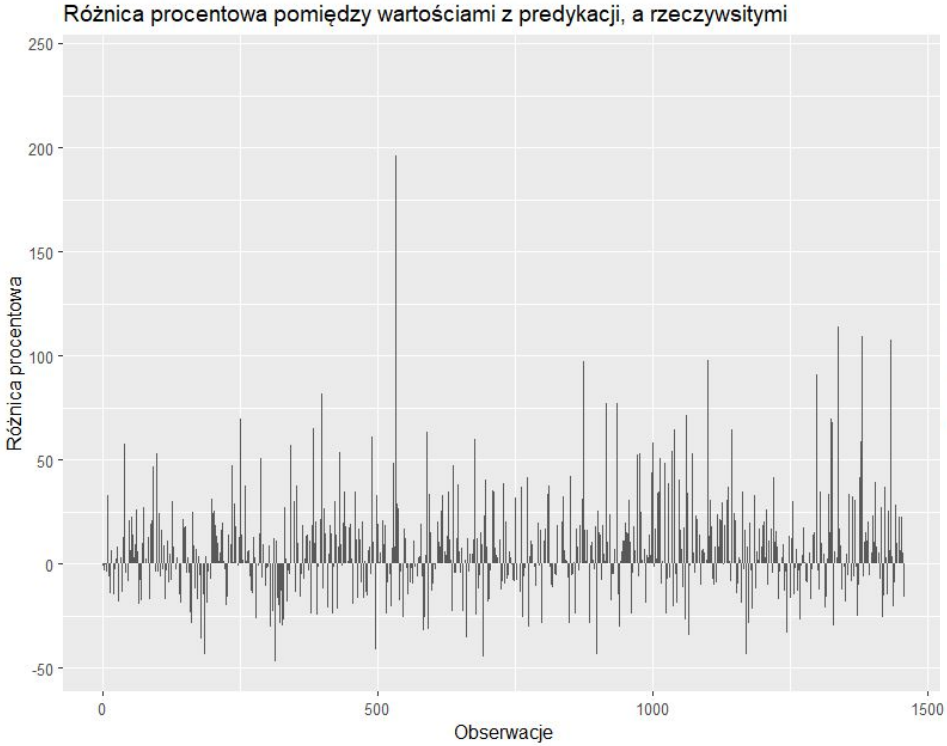
Treningowy:

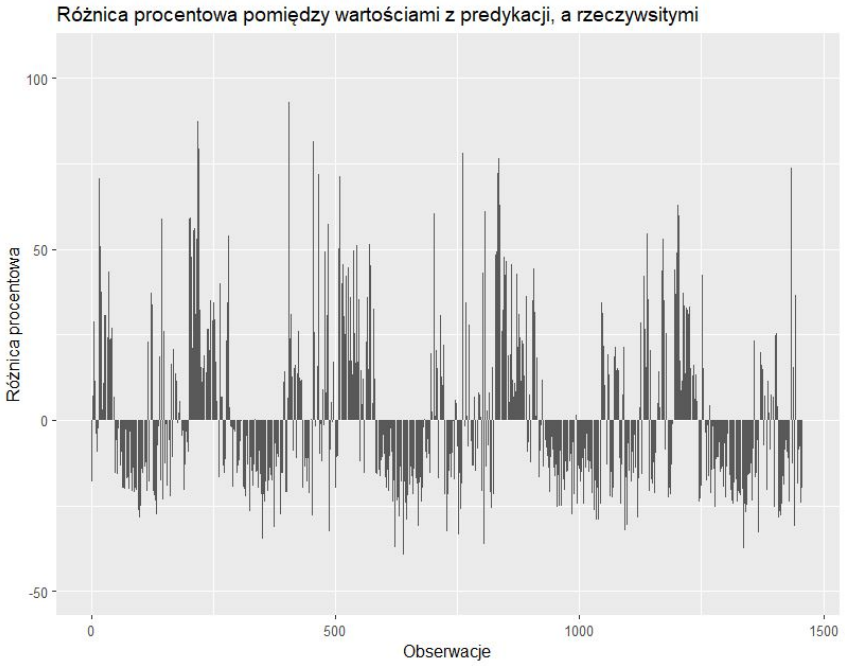
- RMSE = 43422.84
- MAE = 29046.91

Testowy:

- RMSE = 44024.12
- MAE = 35923.84

Random Forest - model trzeci





Random Forest - model czwarty

Hiperparametry:

- num.trees = 1000
- max.depth = 5
- min.node.size = 5
- splitrule = "variance",

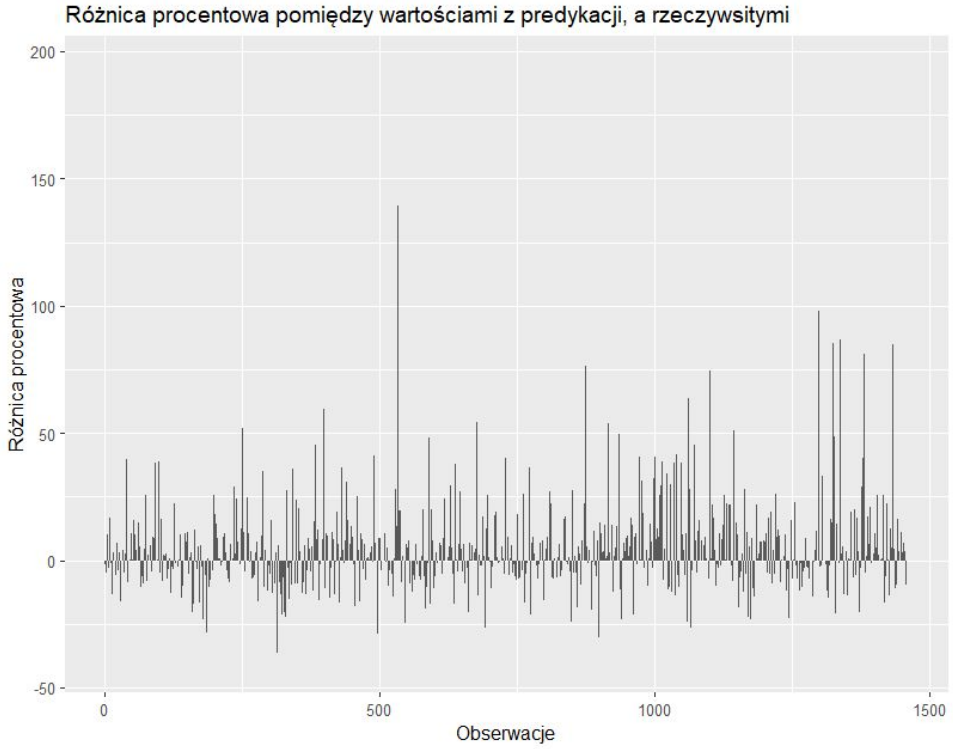
Treningowy:

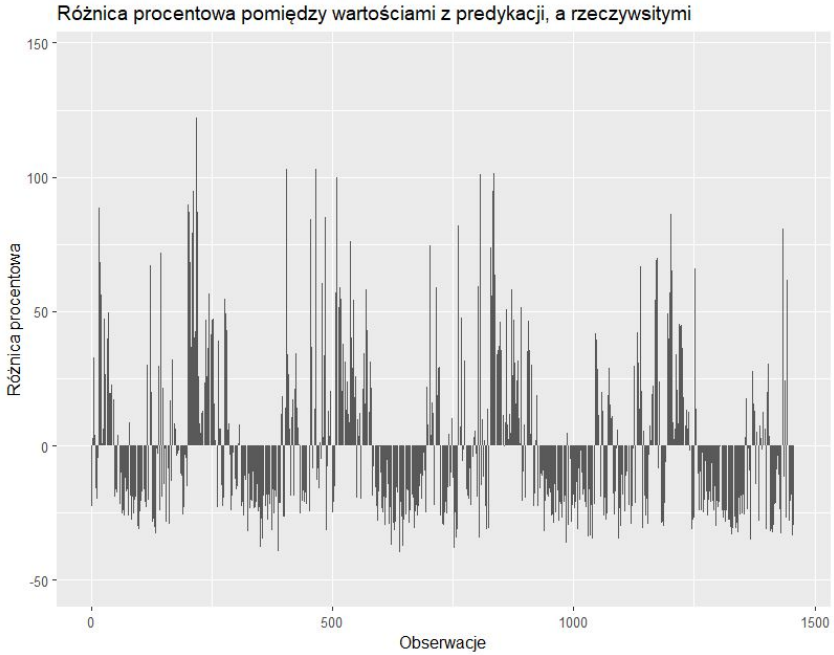
- RMSE = 30349.42
- MAE = 19958.47

Testowy:

- RMSE = 55754.17
- MAE = 44370.54

Random Forest - model czwarty





Random Forest - model piąty

Hiperparametry:

- num.trees = 400
- max.depth = 5
- min.node.size = 10
- splitrule = "variance",

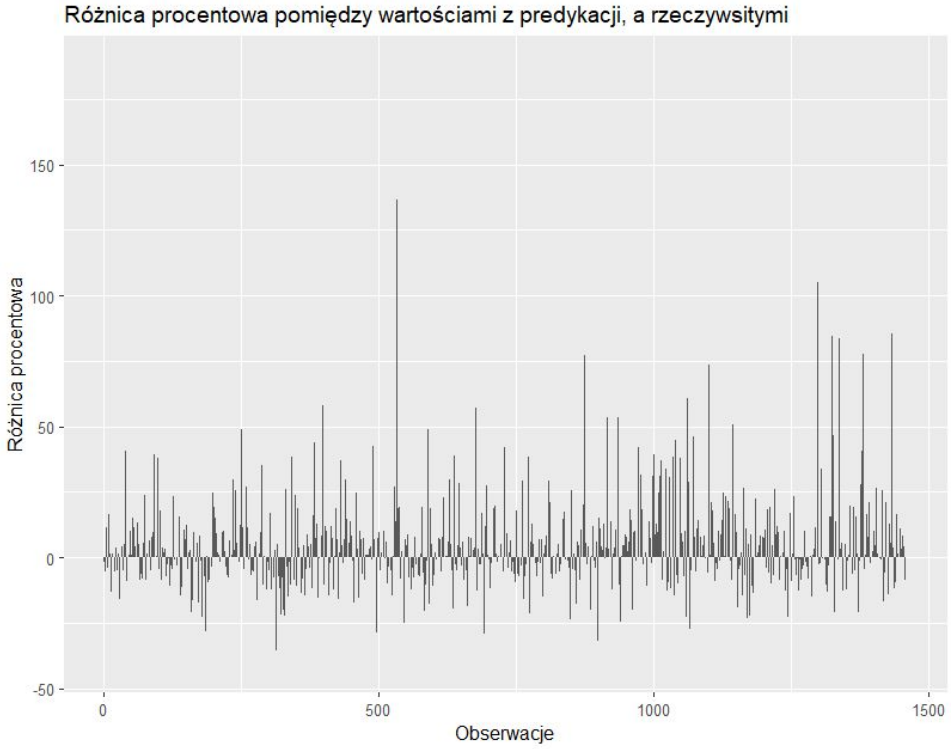
Treningowy:

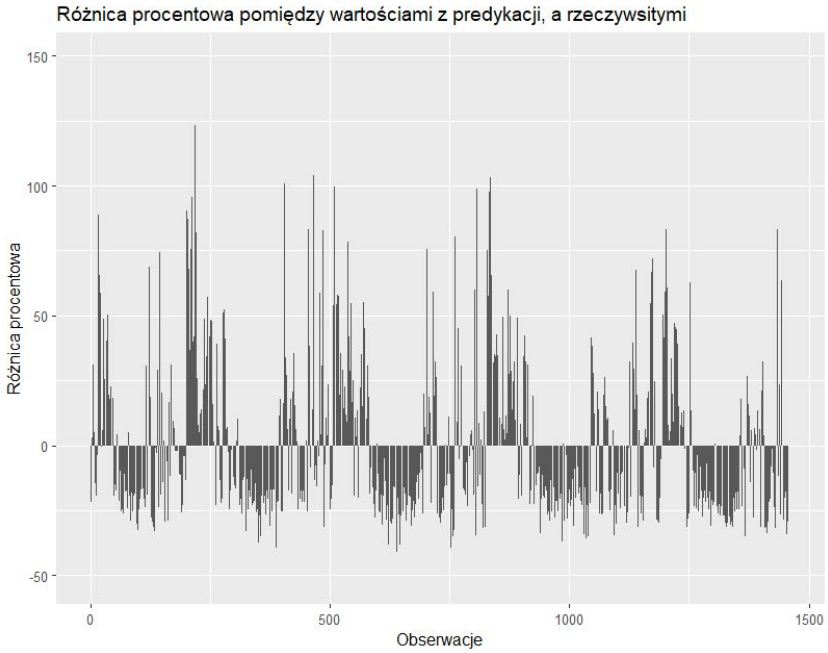
- RMSE = 30553.19
- MAE = 19950.2

Testowy:

- RMSE = 55836.22
- MAE = 44281.23

Random Forest - model piąty





XGBoost

Preprocessing

Wykonanie preprocessingu, który opracowaliśmy przy okazji EDA, w tym:

- **Zakodowanie braków danych jako wartości Absent lub -1.**
- **Zakodowanie zmiennych wyrażających ocenę**
- **Wybór faktycznie numerycznych zmiennych**

Alley	BsmtCond	MiscFeature	KitchenQual
NA	TA	NA	Gd
NA	TA	NA	TA
NA	TA	NA	Gd
NA	Gd	NA	Gd
NA	TA	NA	Gd
NA	TA	Shed	TA
NA	TA	NA	Gd
NA	TA	Shed	TA
NA	TA	NA	TA

Alley	BsmtCond	MiscFeature	KitchenQual
Absent	3	Absent	3
Absent	3	Shed	3
Absent	3	Absent	3
Absent	3	Absent	4
Absent	3	Absent	3
Absent	3	Absent	3
Pave	3	Absent	4
Absent	3	Absent	4

Proces tuningu

- **Baseline i wstępne zapoznanie z modelem**

Treningowy RMSE: 32475.83	**Treningowy RMSE: 3937.696**

- **Strojenie parametrów z pakietem `caret`**

Parametry: `eta`, `max_depth`, `gamma`, `colsample_bytree`, `min_child_tree`, `subsample`, `nrounds`

```
train_control = trainControl(method = "cv", number = 5, search = "random")
model = train(train, y = train$SalePrice,
              tuneLength = 200, metric = "RMSE",
              method = "xgbTree", trControl = train_control)
```

Wytrenowaliśmy 200 modeli używając pakietu `caret`.
Użyliśmy losowego doboru parametrów dla metody `xgbTree` z cross-walidacją.

Proces tuningu

- **Wybór 40 najlepszych modeli i dopasowanie parametrów do funkcji xgb_function i xgb_predict**

- **Selekcja 5 najlepszych pod względem RMSE i MAE**

eta	max_depth	gamma	colsample_bytree	min_child_weight	subsample	nrounds
0.10218812	2	3.090181	0.4472218	14	0.4589773	882
0.21937280	3	7.104679	0.4028896	1	0.5223975	95
0.04540008	8	8.428732	0.4297441	1	0.8770274	863
0.50138441	7	1.939117	0.5310363	12	0.5939928	820
0.37003314	5	6.501581	0.4433426	9	0.3200797	814

XGBoost - model pierwszy

Hiperparametry:

- eta = 0.04462844
- max_depth = 3
- gamma = 3.580246
- colsample_bytree = 0.6035825
- min_child_weight = 1
- subsample = 0.5339151
- nrounds = 464

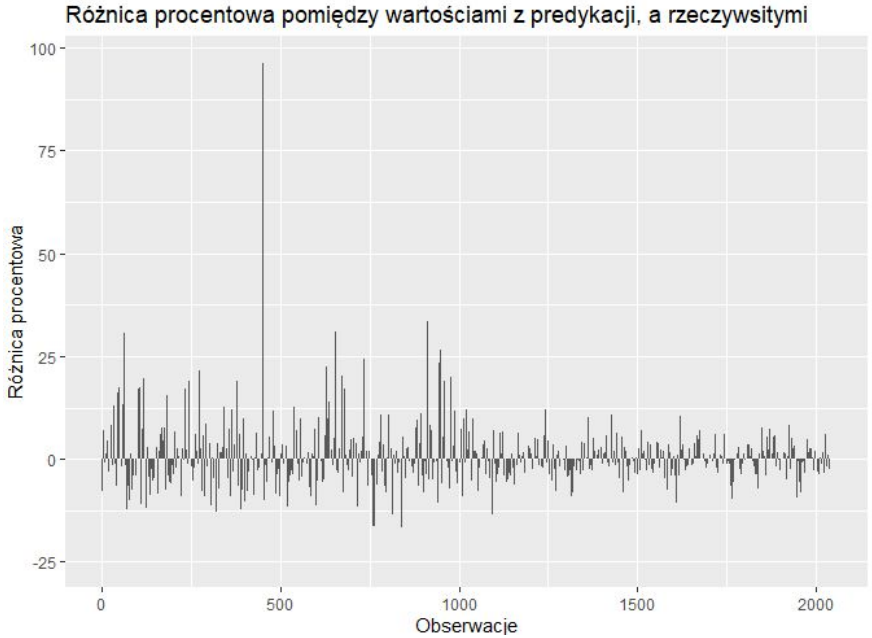
Treningowy:

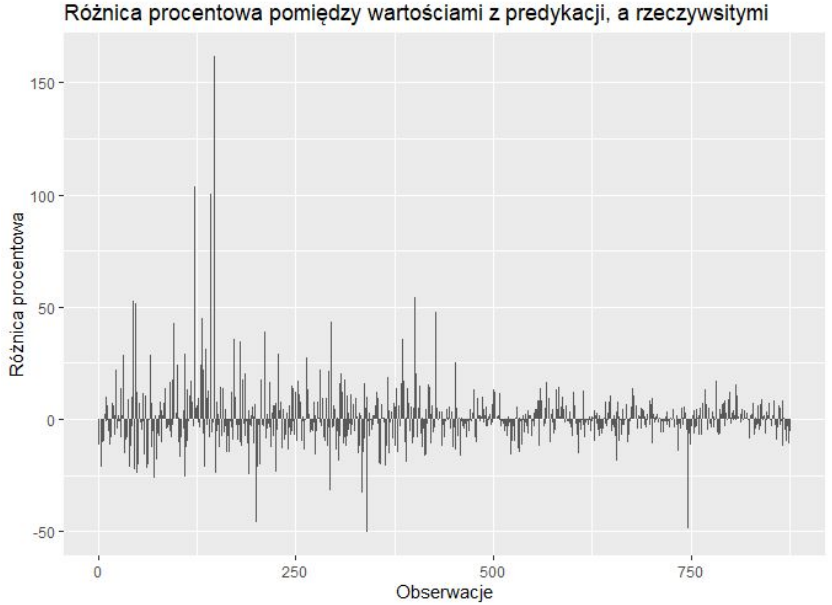
- RMSE = 12005.92
- MAE = 8635.486

Testowy:

- RMSE = 28219.85
- MAE = 13464.82

XGBoost - model pierwszy





XGBoost - model drugi

Hiperparametry:

- **eta = 0.05525418**
- **max_depth = 3**
- **gamma = 1.062662**
- **colsample_bytree = 0.6054747**
- **min_child_weight = 10**
- **subsample = 0.9131714**
- **nrounds = 305**

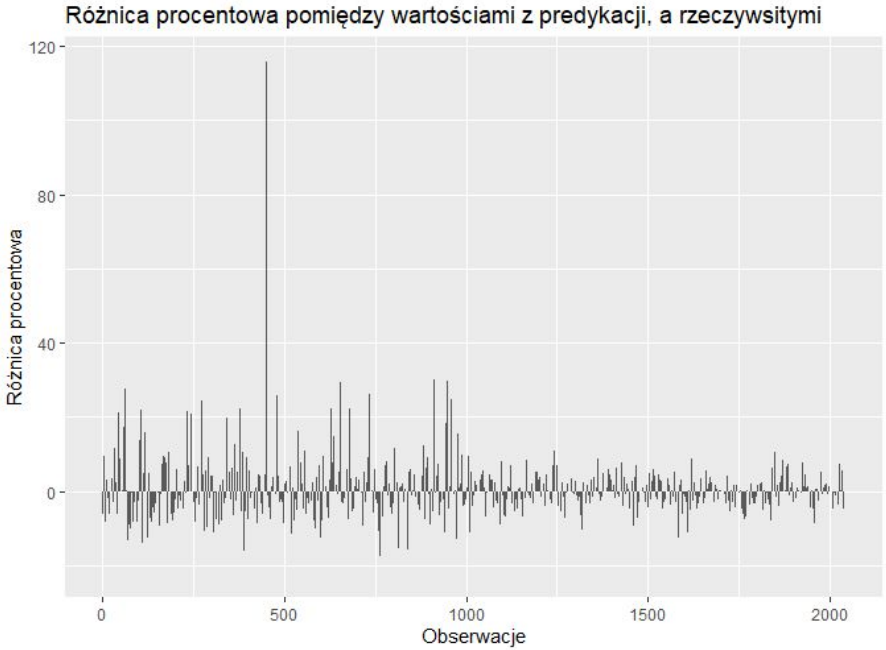
Treningowy:

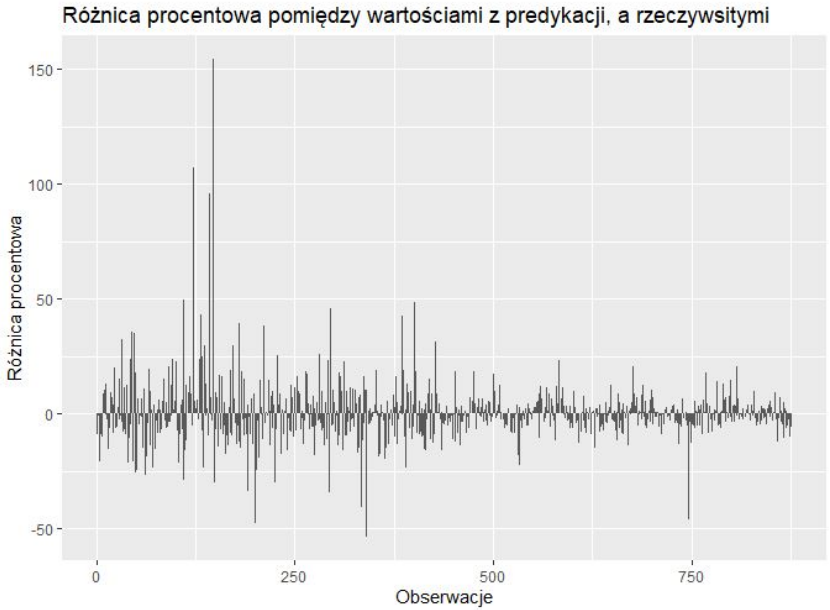
- **RMSE = 13583.38**
- **MAE = 9194.638**

Testowy:

- **RMSE = 30127.77**
- **MAE = 13843.16**

XGBoost - model drugi





XGBoost - model trzeci

Hiperparametry:

- eta = 0.1959231
- max_depth = 2
- gamma = 8.52637
- colsample_bytree = 0.6626087
- min_child_weight = 16
- subsample = 0.5217781
- nrounds = 81

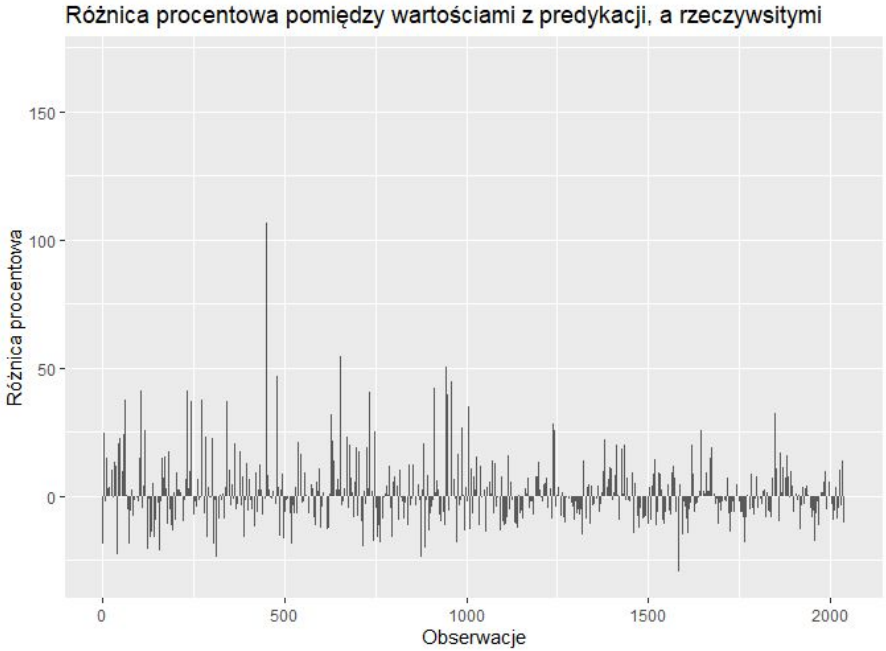
Treningowy:

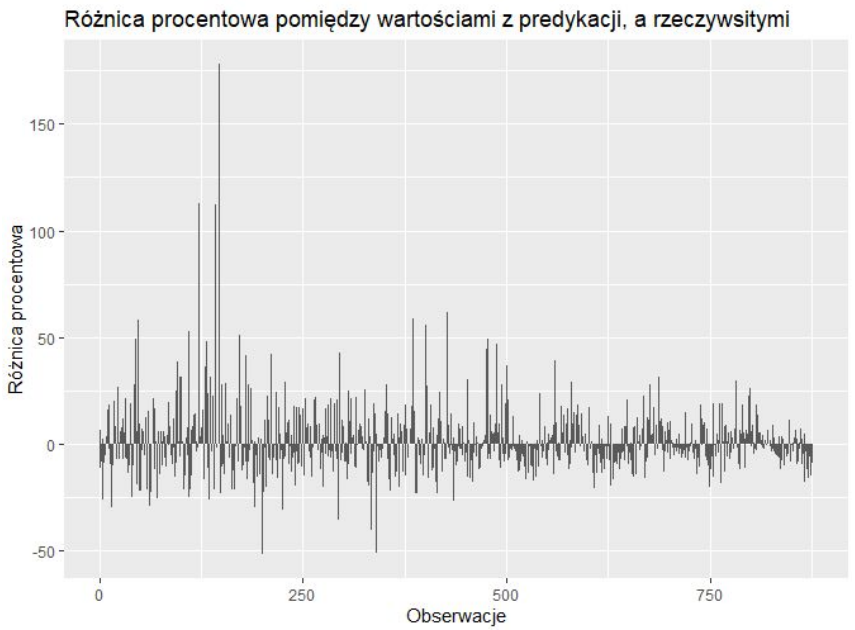
- RMSE = 22388.00
- MAE = 15754.058

Testowy:

- RMSE = 34033.15
- MAE = 18774.92

XGBoost - model trzeci





XGBoost - model czwarty

Hiperparametry:

- eta = 0.2041176
- max_depth = 2
- gamma = 3.240579
- colsample_bytree = 0.6905933
- min_child_weight = 1
- subsample = 0.6497563
- nrounds = 405

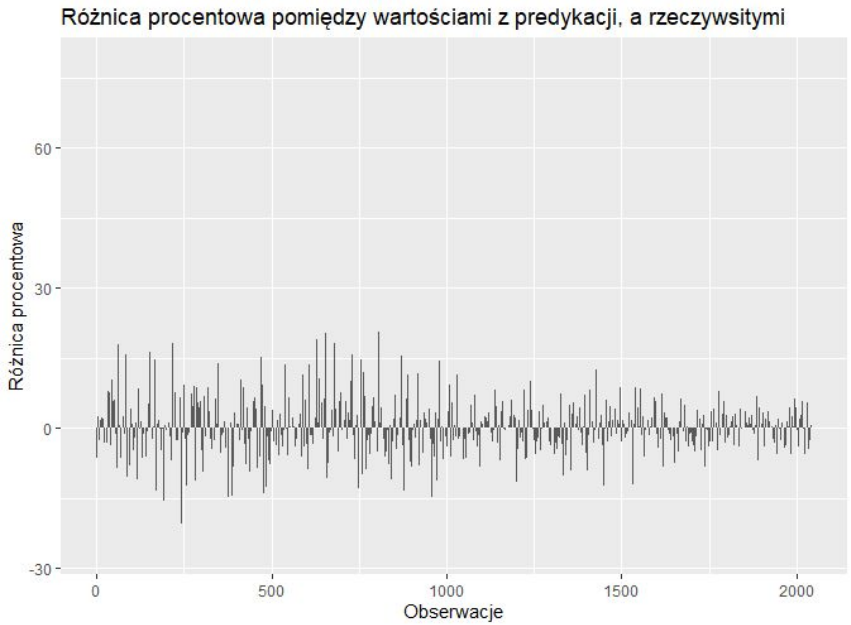
Treningowy:

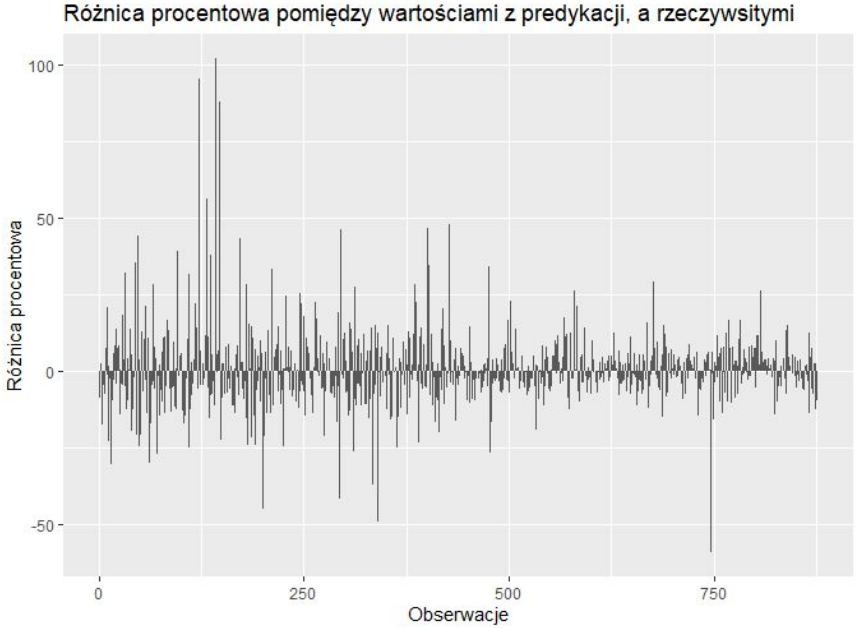
- RMSE = 10474.84
- MAE = 7800.387

Testowy:

- RMSE = 29589.40
- MAE = 14618.19

XGBoost - model czwarty





XGBoost - model piąty

Hiperparametry:

- **eta = 0.08603091**
- **max_depth = 2**
- **gamma = 3.549911**
- **colsample_bytree = 0.4167207**
- **min_child_weight = 0**
- **subsample = 0.6369859**
- **nrounds = 163**

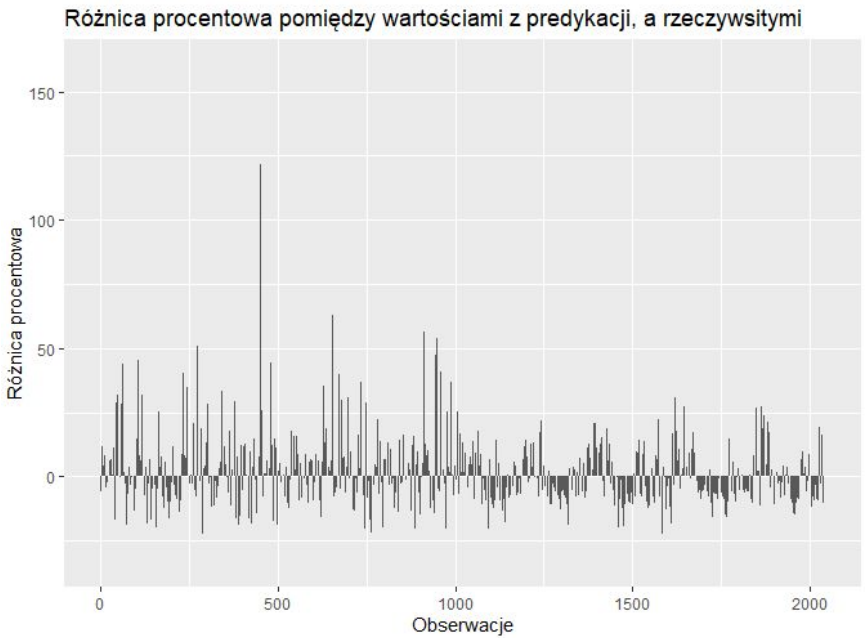
Treningowy:

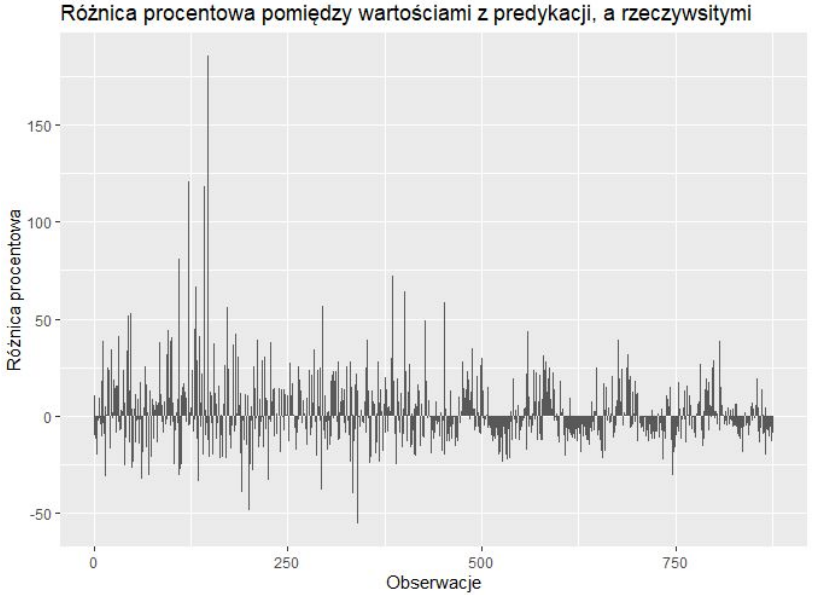
- **RMSE = 24614.30**
- **MAE = 18319.235**

Testowy:

- **RMSE = 36718.85**
- **MAE = 21625.45**

XGBoost - model piąty

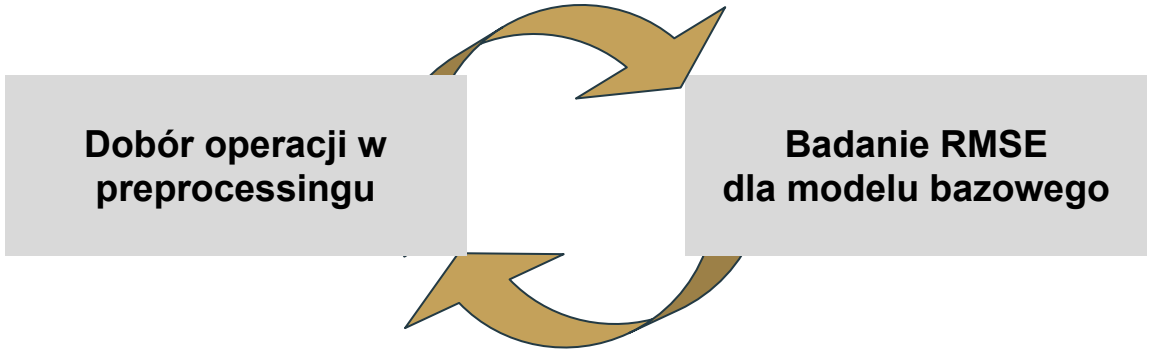




LightGBM

Preprocessing

- **Próba użycia naszego preprocessingu na zbiorze**



- **Wniosek: Najlepsze wyniki na surowych danych**

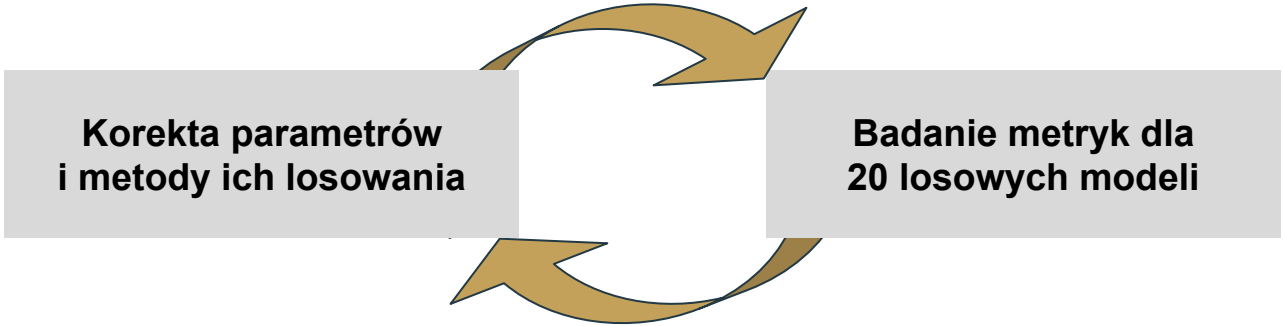
Treningowy RMSE: 42142.3 **Testowy RMSE: 17196.23**

Wykorzystanie parametrów `autofacotr` i `forceconvert`

Proces tuningu

- **Ręczne dobranie siatki parametrów, z uwzględnieniem zasad dobierania w LightGBM.**

Zadbaliśmy, aby wskazówki, których nauczyliśmy się podczas przygotowania prezentacji do odpowiedniego modyfikowania siatki parametrów. Dlatego zastosowaliśmy odpowiednie ograniczenia, a niektóre parametry uzależniliśmy od innych. (np. `num_leaves` od `max_depth`)



Proces tuningu

- **Trenowanie 200 modeli z najoptymalniejszą siatką parametrów**

```
max_depth = sample(3:10, 1)
num_leaves = sample(ceiling((2^max_depth)/10):2^max_depth, 1)
bagging_fraction = runif(1, 0.4, 1)
feature_fraction = runif(1, 0.4, 1)
learning_rate = runif(1, 0.05, 0.3)
num_iterations = sample(
  ceiling(log(1/learning_rate)*25):ceiling(log(1/learning_rate)*250), 1)
min_data_in_leaf = sample(0:40, 1)
lambda_l1 = rexp(1, 1)
```

- **Selekcja 5 najlepszych pod względem RMSE i MAE**

max_depth	num_leaves	bagging_fraction	feature_fraction	learning_rate	num_iterations	min_data_in_leaf	lambda_l1
3	2	0.5724640	0.4015881	0.06619538	242	31	1.37272030
4	2	0.5591560	0.7143656	0.18479657	182	39	0.02852181
4	2	0.8445833	0.6481415	0.14623834	54	27	1.24955739
3	2	0.5978079	0.4352913	0.17079440	347	33	0.72251183
3	2	0.9925450	0.6002572	0.14964913	98	21	0.40267346

LightGBM - model pierwszy

Hiperparametry:

- max_depth = 3
- num_leaves = 2
- bagging_fraction = 0.572464
- feature_fraction = 0.4015881
- learning_rate = 0.06619538
- num_iterations = 242
- min_data_in_leaf = 31
- lambda_l1 = 1.37272030

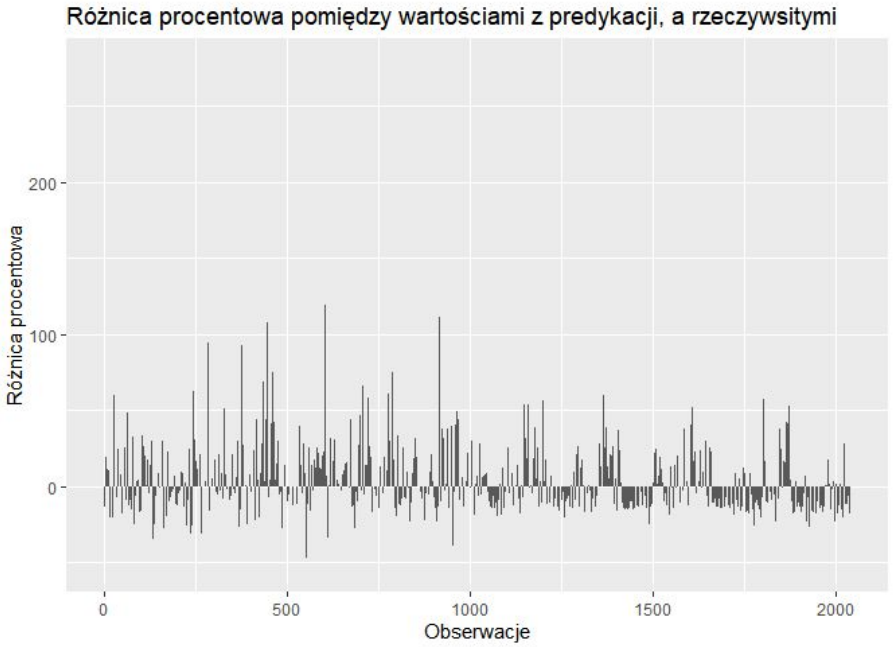
Treningowy:

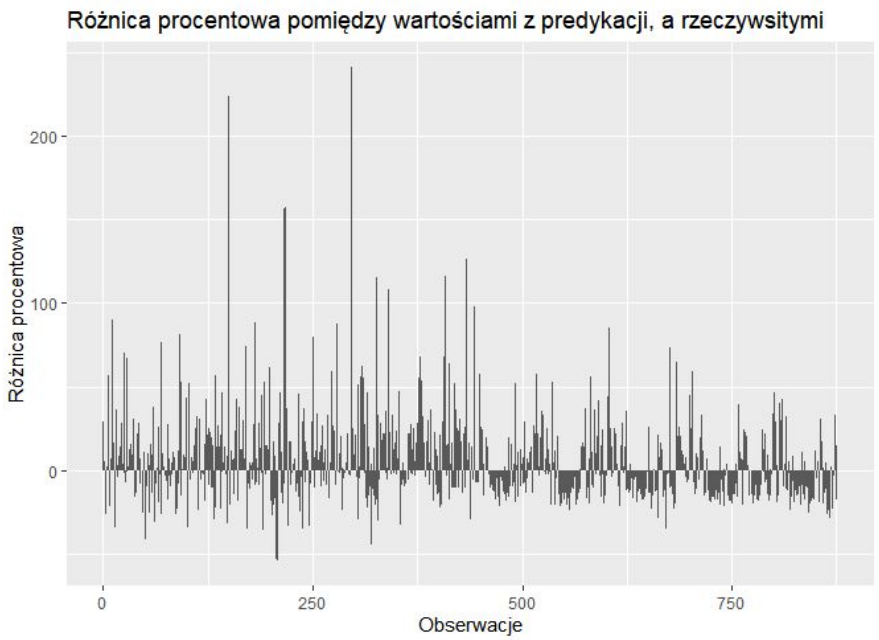
- RMSE = 39869.42
- MAE = 28334.95

Testowy:

- RMSE = 39968.16
- MAE = 28310.39

LightGBM - model pierwszy





LightGBM - model drugi

Hiperparametry:

- **max_depth = 4**
- **num_leaves = 2**
- **bagging_fraction = 0.559156**
- **feature_fraction = 0.7143656**
- **learning_rate = 0.1847966**
- **num_iterations = 182**
- **min_data_in_leaf = 39**
- **lambda_l1 = 0.02852181**

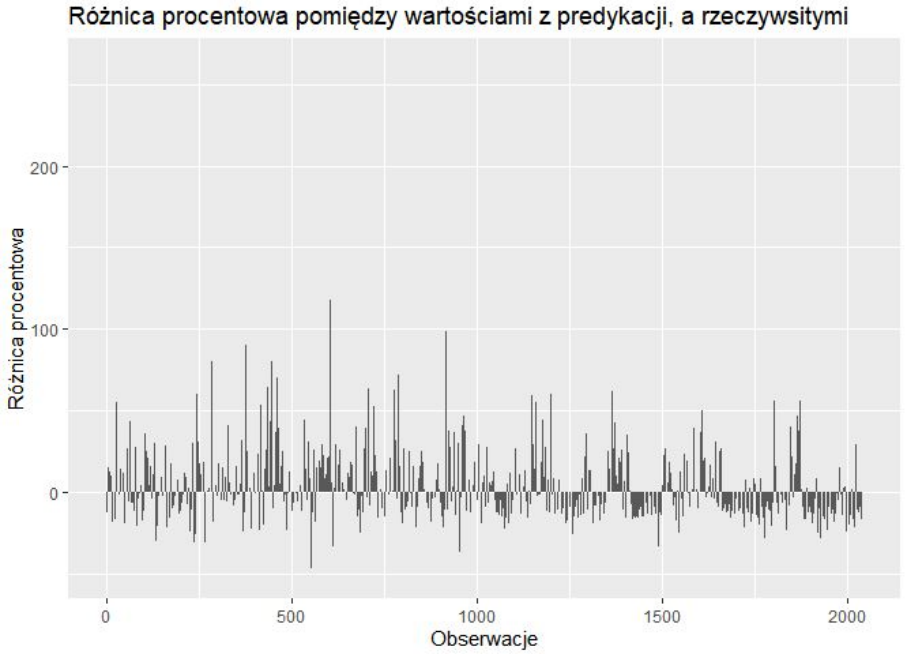
Treningowy:

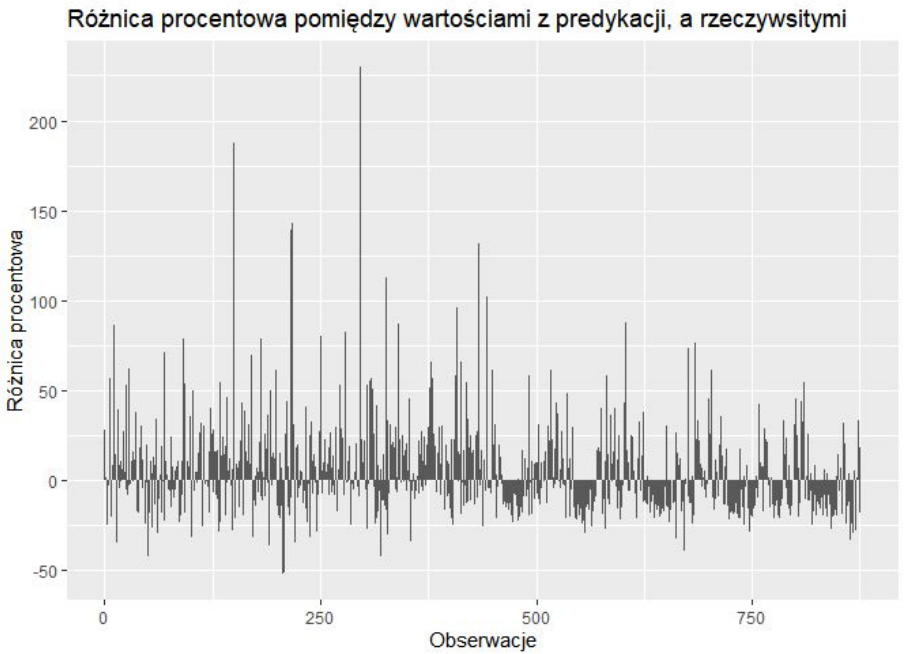
- **RMSE = 39101.91**
- **MAE = 28035.95**

Testowy:

- **RMSE = 40065.58**
- **MAE = 28835.33**

LightGBM - model drugi





LightGBM - model trzeci

Hiperparametry:

- max_depth = 4
- num_leaves = 2
- bagging_fraction = 0.8445833
- feature_fraction = 0.6481415
- learning_rate = 0.1462383
- num_iterations = 54
- min_data_in_leaf = 27
- lambda_l1 = 1.24955739

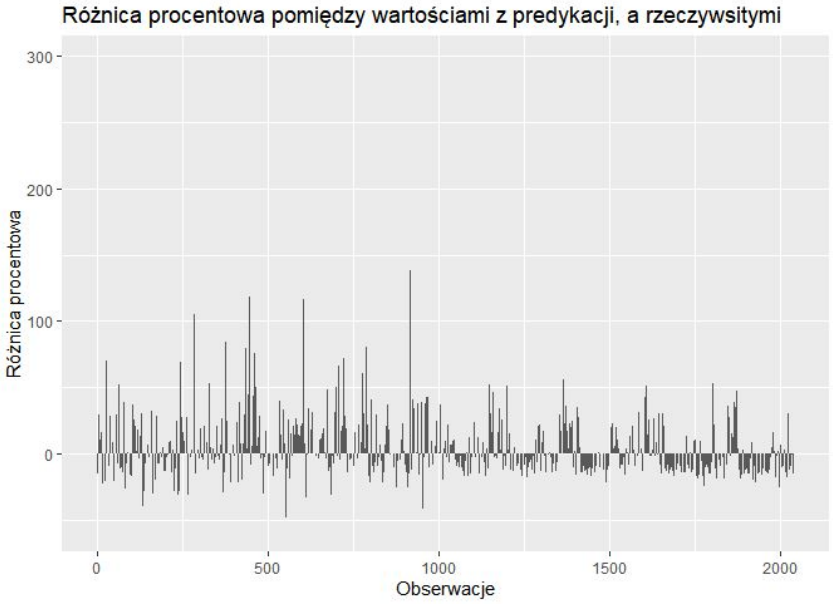
Treningowy:

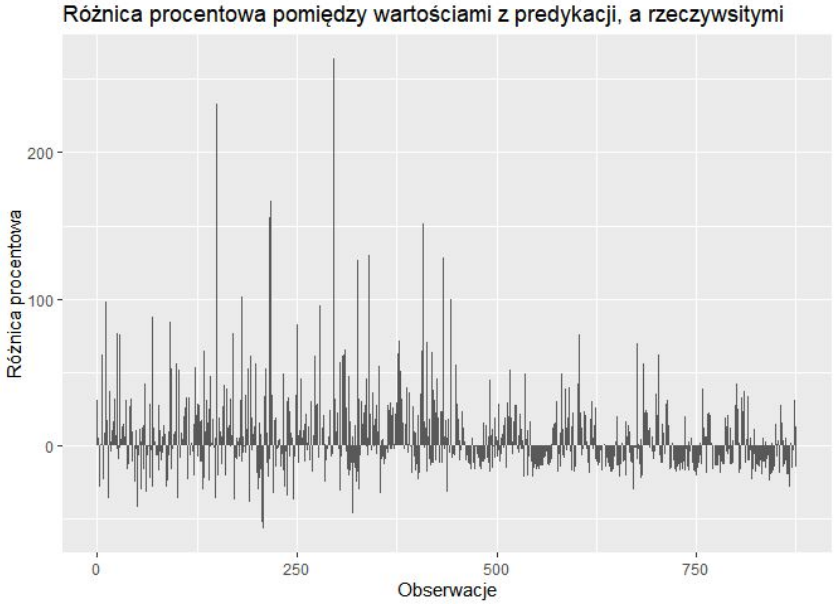
- RMSE = 40663.25
- MAE = 28951.90

Testowy:

- RMSE = 41350.35
- MAE = 28899.82

LightGBM - model trzeci





LightGBM - model czwarty

Hiperparametry:

- **max_depth = 3**
- **num_leaves = 2**
- **bagging_fraction = 0.5978079**
- **feature_fraction = 0.4352913**
- **learning_rate = 0.1707944**
- **num_iterations = 347**
- **min_data_in_leaf = 33**
- **lambda_l1 = 0.7225118**

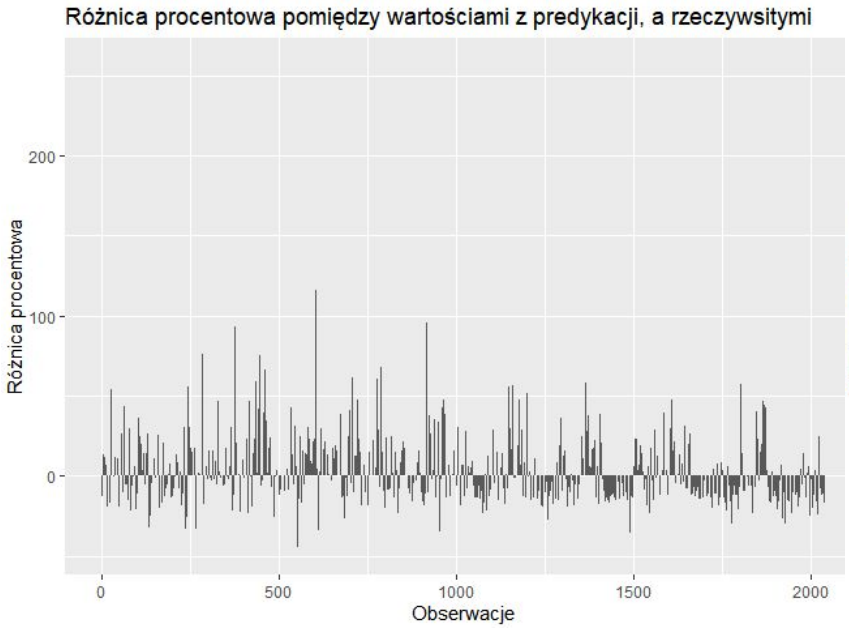
Treningowy:

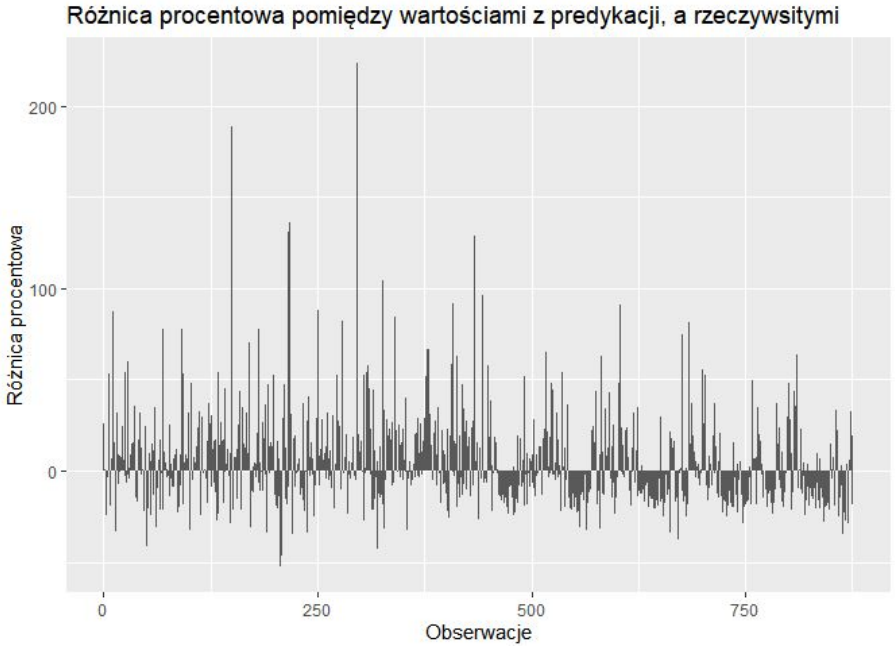
- **RMSE = 37729.43**
- **MAE = 27384.05**

Testowy:

- **RMSE = 41060.23**
- **MAE = 29089.3**

LightGBM - model czwarty





LightGBM - model piąty

Hiperparametry:

- max_depth = 3
- num_leaves = 2
- bagging_fraction = 0.992545
- feature_fraction = 0.6002572
- learning_rate = 0.1496491
- num_iterations = 98
- min_data_in_leaf = 21
- lambda_l1 = 0.40267346

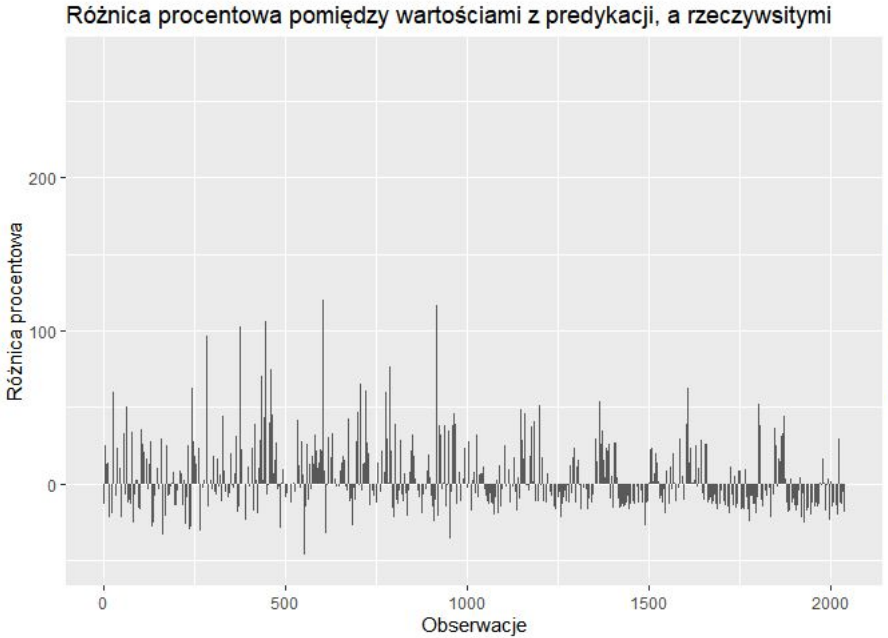
Treningowy:

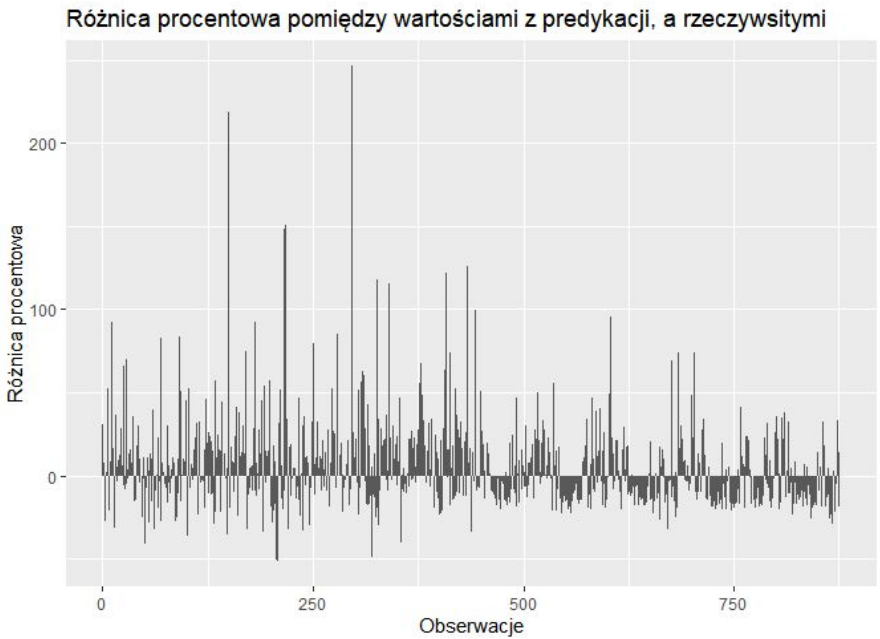
- RMSE = 39295.34
- MAE = 28323.76

Testowy:

- RMSE = 41700.98
- MAE = 29010.35

LightGBM - model piąty





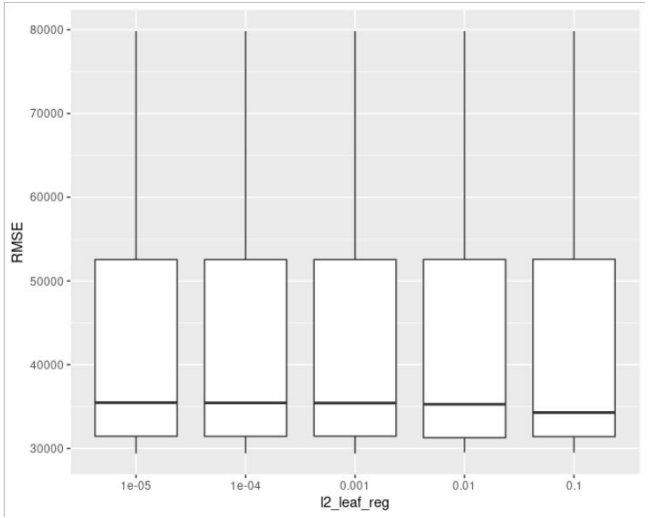
Catboost

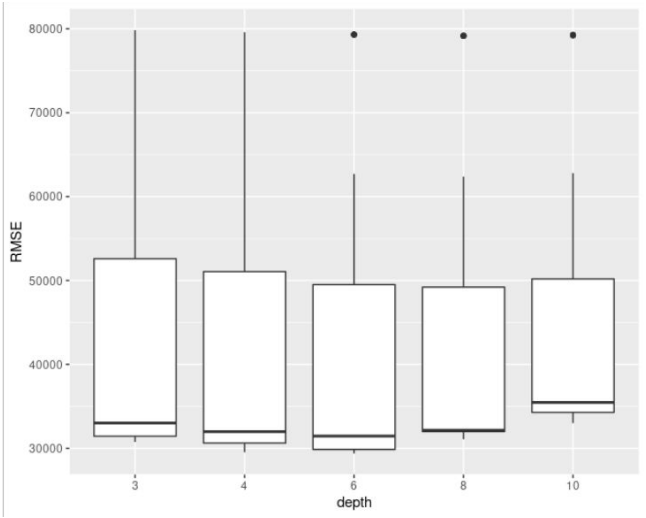
Rozpoznanie

```
catboost_grid <- expand.grid(
  iterations = c(100, 500, 1000),
  depth = c(3, 4, 6, 8, 10),
  l2_leaf_reg = c(1e-1, 1e-2, 1e-3, 1e-4, 1e-5),
  learning_rate = c(1e-1, 1e-2, 1e-3),
```

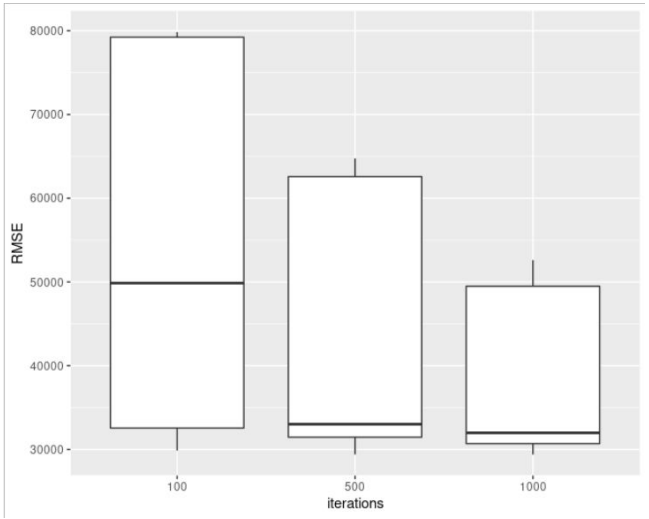
- Preprocessing przez wypełnienie wszystkich braków
  - Domyślne zachowanie w funkcji preprocessingowej to było usuwanie wierszy
  - Na naszym zbiorze zostały całe 2 wiersze
- Wykorzystaliśmy pakiet `caret`
  - `catboost` nie udostępnia niektórych parametrów do `caret`
  - Zwłaszcza tych ciekawiej brzmiących, jak `bagging_temperature`
- Najpierw przeszukaliśmy całą kratę z kilkoma wartościami (225 kombinacji)
- Potem przyjrzeliśmy się jak te bardziej wpływowe hiperparametry mają wpływ
- A następnie przeprowadziliśmy trochę random search (100 modeli)

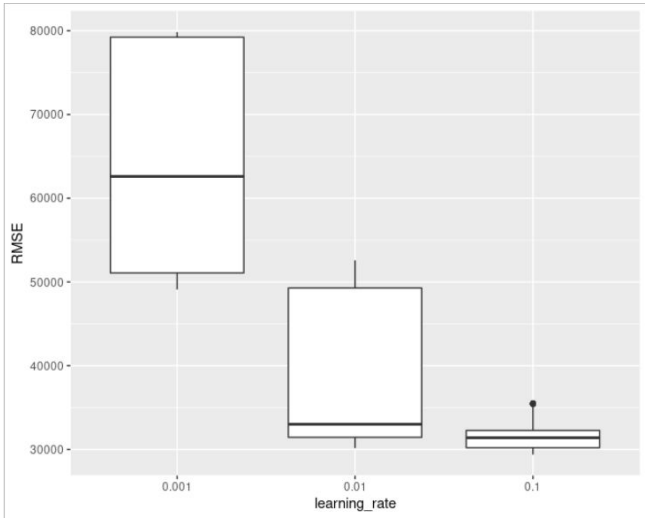
Mało wpływowe



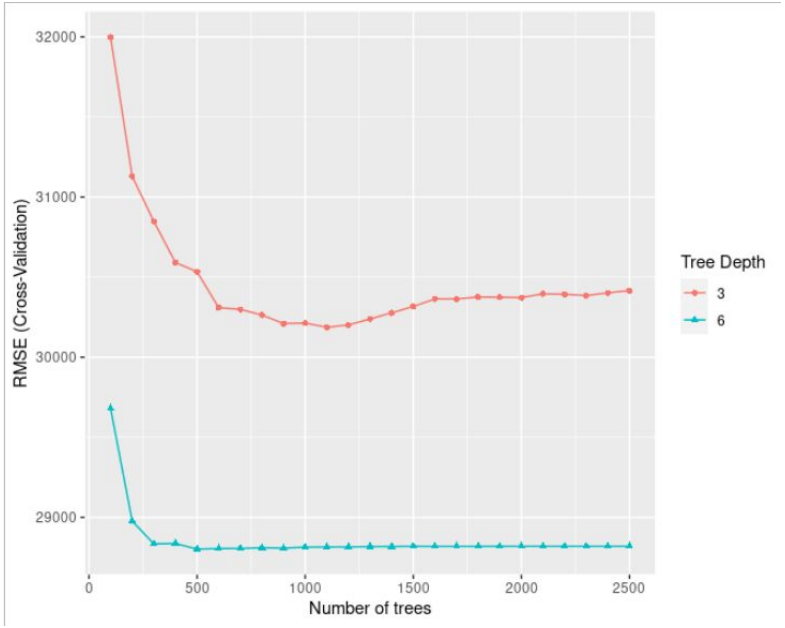


Bardziej wpływowe





Sprawdzenie wielu wartości



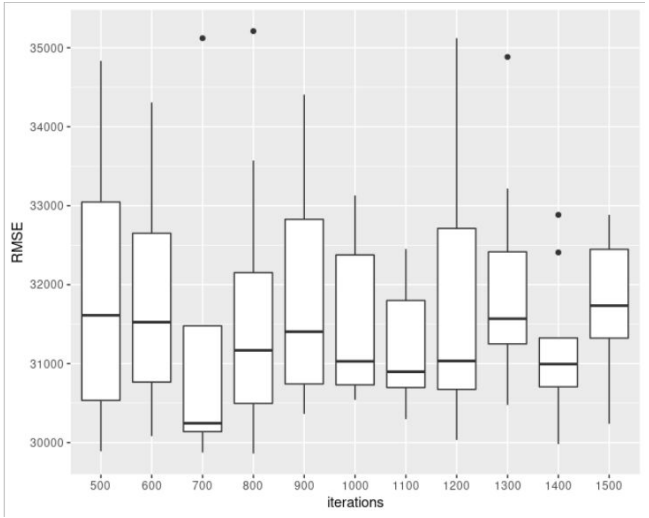
Wyniki

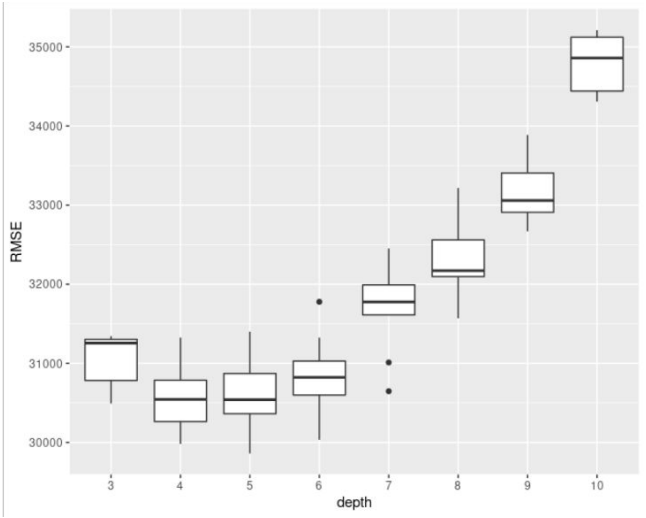
A tibble: 6 x 12

depth	learning_rate	l2_leaf_reg	rsm	border_count	iterations	RMSE	Rsquared	MAE	RMSESD	RsquaredSD	MAESD
<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>	<dbl>
5	0.1	0.05	0.75	254	800	29862.	0.877	18322.	4798.	0.0359	1309.
5	0.1	0.05	0.75	254	700	29875.	0.877	18307.	4804.	0.0360	1292.
5	0.1	0.05	0.75	254	500	29891.	0.877	18275.	4757.	0.0356	1235.
4	0.1	0.01	0.9	254	1400	29981.	0.875	18319.	6360.	0.0492	1906.
6	0.1	0.01	0.9	254	1200	30034.	0.875	18142.	6348.	0.0496	1658.

Wyniki bez kroswalidacji były lepsze, około 24 tyś RMSE na zbiorach testowych

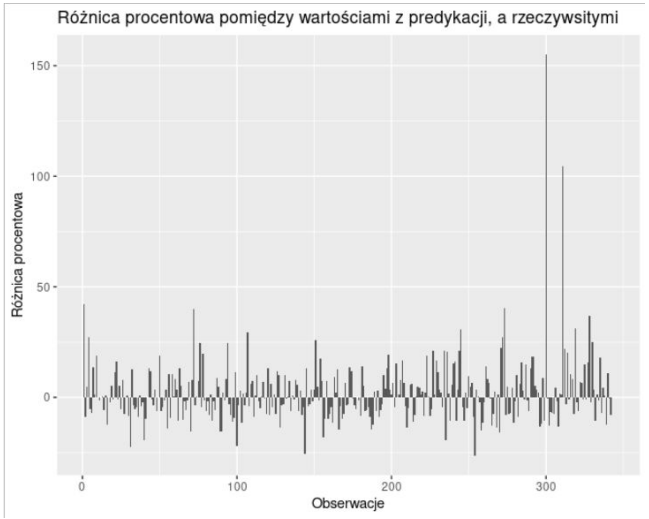
Wykresy

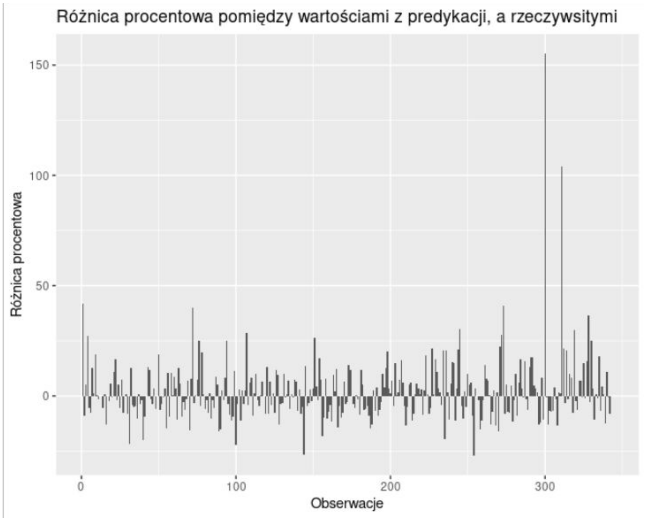

RMSE
35000
34000
33000
32000
31000
30000
500
600
700
800
900
1000
1100
1200
1300
1400
1500
iterations


RMSE
35000
34000
33000
32000
31000
30000
3
4
5
6
7
8
9
10
depth

Błędy - podobne

m1





m2