Sick dataset analysis part 2

Wojciech Bogucki

28/04/2020

Contents

Prepared dataset

For my analysis I once again used dataset *sick* with previous transormations: I removed three columns which gave no information and added constraint for age to avoid human mistakes.

```
sick_train <- sick_train %>% select(c(-TBG, -TBG_measured, -hypopituitary))
sick_test <- sick_test %>% select(c(-TBG, -TBG_measured, -hypopituitary))

sick_train <- sick_train %>% mutate(age=replace(age, age>130 | age<0, NA))
sick_test <- sick_test %>% mutate(age=replace(age, age>130 | age<0, NA))
```

As a reminder, I also created dataset with imputed missing values because some models required it. For imputation I used package `mice`.

Table 1: Impuation method for each variable

variable	imputaton method
sex	Logistic regression
TSH	Predictive mean matching
T3	Predictive mean matching
TT4	Predictive mean matching
T4U	Predictive mean matching
FTI	Predictive mean matching

Used models

In my previous analysis I used only interpretable models. Decision tree model from package `part` had best AUPRC score. With this model I compared three new so called 'black box' models:

- Random Forest (package `ranger`)
- Gradient Boosting Machine (package `gbm`)
- XGBoost (package `xgboost`)

Different versions of dataset

Different models have different requirements and limitations for input data. Decision tree and Gradient Boostting Machine models accept missing values in dataset so I used normal data after transformations. For Random Forest I used dataset with imputed missing values. Lastly, XGBoost accepts only numeric data, so I changed factors to numeric values.

Tuning model's hyperparameters

On every model I performed hyperparameter tuning with package `mlr`.

Table 2: Hyperparameters after tuning for

minsplit	minbucket	cp
21	7	0.000367

Table 3: Hyperparameters after tuning for Gradient Boosting Machine

n.trees	interaction.depth	n.minobsinnode	distribution	shrinkage
169	3	4	gaussian	0.0932

Table 4: Hyperparameters after tuning for Random Forest

mtry	min.node.size	splitrule	replace
7	3	gini	FALSE

Table 5: Hyperparameters after tuning for XGBoost

min_child_weight	max_depth	gamma	eta
4.97	4	3.86	0.374

Comparison of prediction measures

As in previous analysis, I calculated measures of goodness of predicton: agggregated AUC from 5-fold crossvalidation on training set, AUC on test set and AUPRC on test set. Results are presented in Table 6.

Table 6: Measures of goodness of prediction for each model

model	AUC on 5-fold crossvalidation	AUC on test data	AUPRC on test data
Decision trees	0.940	0.897	0.770
Decision trees with tune	0.961	0.973	0.893
Ranger	0.995	0.993	0.894
Ranger with tune	0.995	0.994	0.909
Gradient Boosting Machine	0.952	0.964	0.746
Gradient Boosting Machine with tune	0.988	0.996	0.922
XGBoost	0.966	0.995	0.911
XGBoost with tune	0.954	0.992	0.868

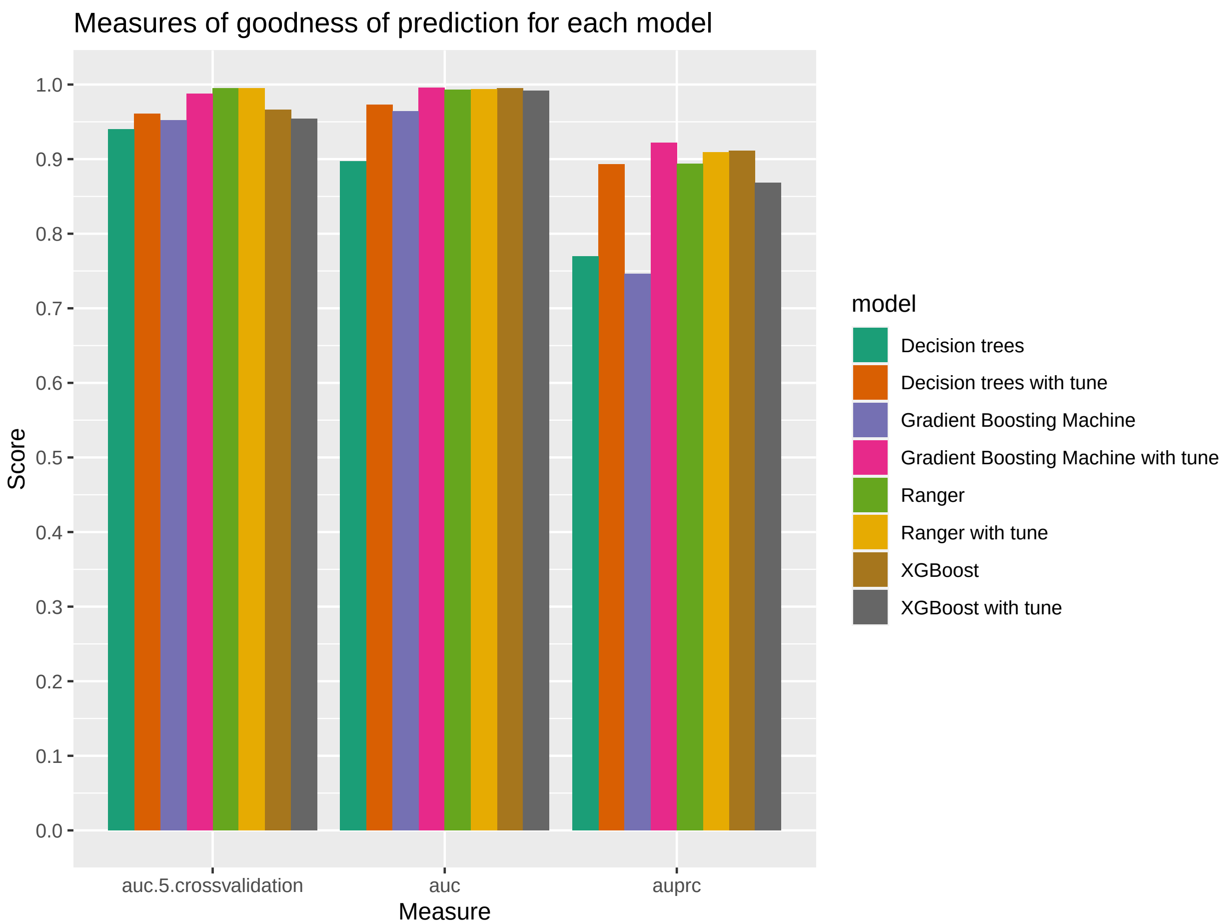

Figure 1: Models comparison

Conclusion

On Figure 1 we can notice that:

- On training dataset ranger models achieve the best results(over 0.99)
- On test dataset Gradint Boosting Machine model with tuned hyperparameters has the best AUC and AUPRC measures
- Surprisingly, GBM model with default hyperparametres has the worst AUPRC result(even worse than decision tree model)
- Generally, black box models performed better than interpretable model in this case but decision tree model with tuned hyperparameters has AUPRC score comparable with black box models
- Only for XGBoost model hyperparameters tuning yields worse results