Survey evaluation with cross-validation for non-developers is simplified by ***flexcv***.

Less coding, more research.

Python Package for Regression on Tabular Data

Fabian Rosenthal, Patrick Blättermann and Siegbert Versümer
Institute of Sound and Vibration Engineering, Hochschule Düsseldorf

Introduction

Predicting continuous outcomes such as sound perception often requires machine learning models. For small to medium datasets, nested k-fold cross-validation (CV) is recommended for model tuning, selection, and evaluation [1]. When dealing with data grouped by subjects or measurements, group-based splitting is crucial to prevent data leakage. `scikit-learn` [2] lacks built-in support for nested group-k-fold CV, necessitating custom implementations.

Our solution is `flexcv`, an open-source Python package providing essential CV tools, including nested group-k-fold splitting, for robust analysis of continuous outcomes and grouped data.

Features

- Model fixed and mixed effects, including slopes
- Fit and evaluate multiple models in one run
- Tune hyperparameters efficiently with `Optuna` [3]
- Employ your own objective functions for hyperparameter tuning, e.g. to adapt to domain specific goals or penalize overfitting
- Use group-based split strategies for inner folds
- Split the data with stratification for continuous outcomes, i.e. to ensure outcome distribution
- Apply correction for mixed effects to any non-linear model with `MERF` [4]
- Scale outer and inner folds independently, i.e. without leaking distribution information
- Perform repeated CV to reduce variance
- Track your experiments (no plot is left behind)
- Store your model configurations in YAML files
- Use under MIT license

Easy to use

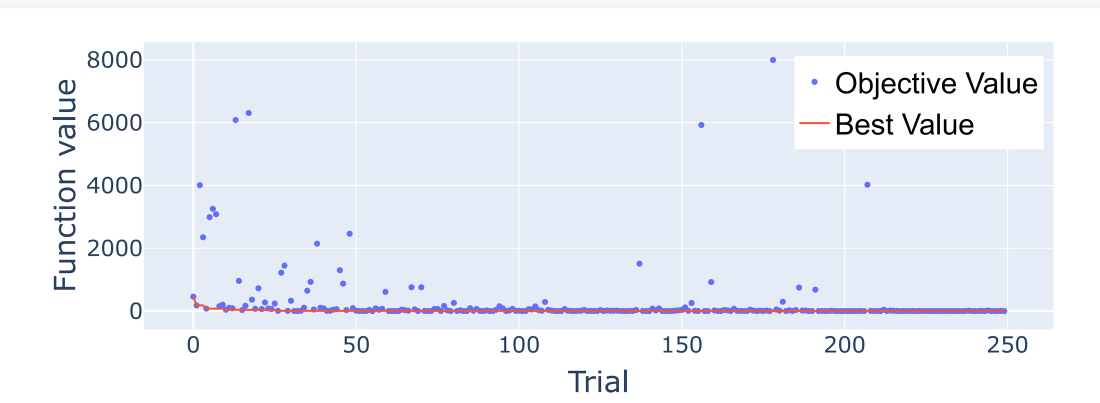
In this example we evaluate a random forest with nested group-k-fold CV. The class interface allows you to write setup code that is expressive and easy to read. You can use method chaining to define your experiment. This scheme provides helpful hints in your code editor and reduces imports.

```python
from flexcv import CrossValidation, RepeatedCV
from flexcv.synthesizer import generate_regression
from sklearn.ensemble import RandomForestRegressor
from optuna.distributions import IntDistribution

# generate a randomly grouped example dataset
X, y, group, _ = generate_regression(n_groups=5)

# we will tune the max tree depth of the RandomForest
params = {"max_depth": IntDistribution(5, 100)}

results = (
    CrossValidation()  # or use RepeatedCV()
    .set_data(X, y, group)
    .set_splits("GroupKFold", "GroupKFold")  # nested group strategy
    .add_model(RandomForestRegressor, requires_inner_cv=True, params=params)
    .perform()
    .get_results()
)
```

Variance in the cross validation estimator can be reduced by performing repeated CV [5]. We simply replace `CrossValidation` with the `RepeatedCV` class.

Optimization

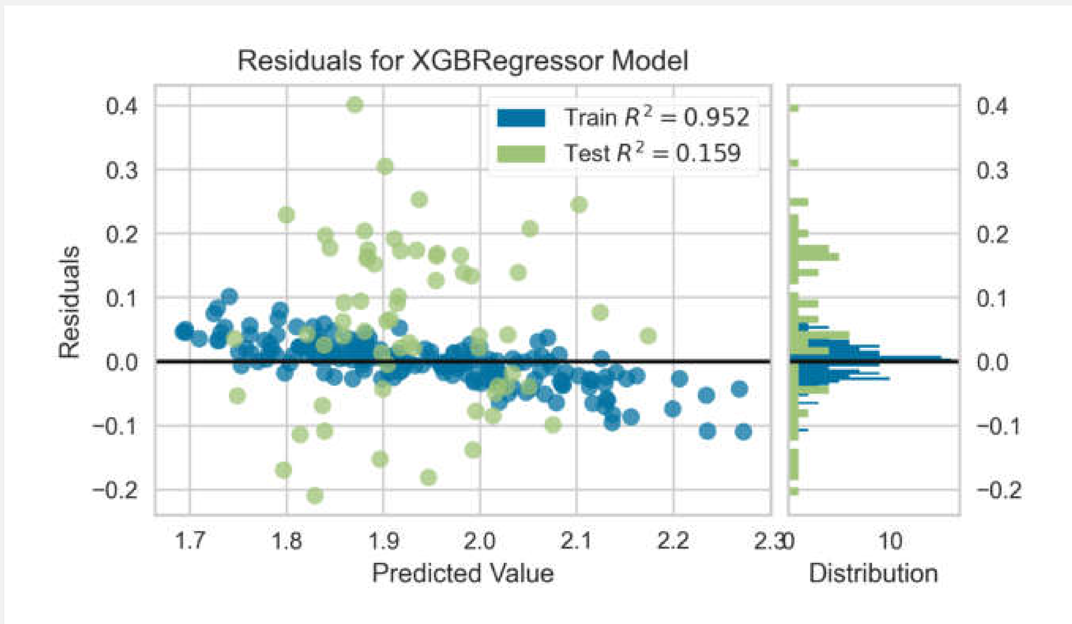
Hyperparameters of machine learning algorithms can be tuned with the integration of `Optuna`. It allows efficient sampling from the search space and often finds a good optimum in a fraction of the time consumed by a grid search. In this figure, we use `Optuna` to minimize the Rosenbrock function[1]:



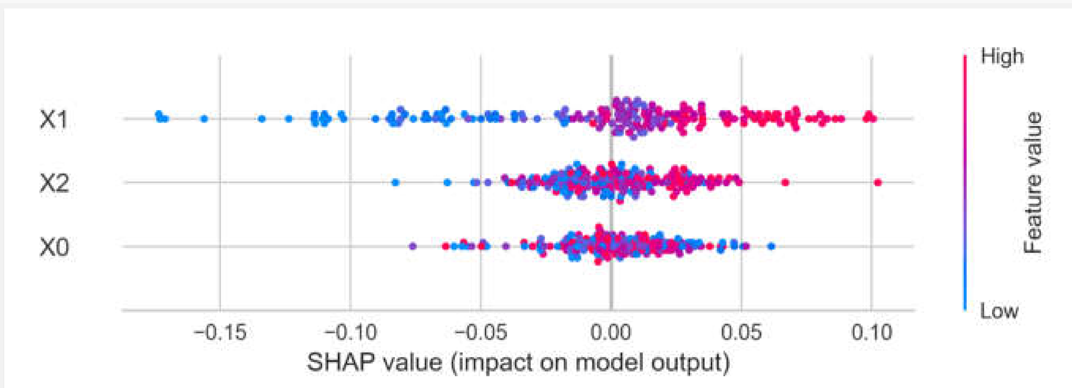
Logging & Analysis

We use `Neptune` [6] to track experiments. You can log data, models, metrics & statistics, and figures. Here are two examples of how you can leverage your experiment logs for analysis:

1. Find problems in your model fit with regression diagnostics. This XGBoost model clearly overfits on the training set of a specific fold and needs regularization:



2. `SHAP` plots let you observe the importance of predictors, even for complex machine learning models. In this example, $X1$ seems to have the greatest impact on model output. We might also consider building a model without $X2$ and $X0$, since their SHAP values are poorly distributed:



References

[1] Sebastian Raschka. *Model Evaluation, Model Selection, and Algorithm Selection in Machine Learning*. en. arXiv:1811.12808 [cs, stat]. Nov. 2020. URL: `http://arxiv.org/abs/1811.12808` (visited on 12/11/2023).

[2] *scikit-learn: machine learning in Python — scikit-learn 1.4.1 documentation*. URL: `https://scikit-learn.org/stable/index.html` (visited on 02/21/2024).

[3] Takuya Akiba et al. "Optuna: A Next-generation Hyperparameter Optimization Framework". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2019.

[4] Ahlem Hajjem, François Bellavance, and Denis Larocque. "Mixed-effects random forest for clustered data". In: *Journal of Statistical Computation and Simulation* 84 (2010), pp. 1313–1328.

[5] Ron Kohavi. "A study of cross-validation and bootstrap for accuracy estimation and model selection". In: *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 2*. IJCAI'95. event-place: Montreal, Quebec, Canada. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1995, pp. 1137–1143. ISBN: 1-55860-363-8.

[6] *neptune.ai | The MLOps stack component for experiment tracking*. en-US. URL: `https://neptune.ai/` (visited on 02/21/2024).

[1] $f(x,y) = (a-x)^2 + b*(y-x^2)^2$. Here $a = 1, b = 100$. Local minimum at $(1,1)$. We sampled 250 pairs (x,y) from the search space $S = \{(x,y) \mid -3 \leq x \leq 3, -3 \leq y \leq 3\}$ using the TPE sampler in `Optuna`. The lowest function value was yielded in trial 235 ($f(x,y) < 0.002$).

repository

contact
rosenthal.fabian@gmail.com
orcid.org/0009-0001-1258-9196

Hochschule Düsseldorf
University of Applied Sciences

Institute of Sound and Vibration Engineering