The Journal of Open Source Software

PyDTS: A Python Package for Discrete-Time Survival Analysis with Competing Risks and Optional Penalization

Tomer Meir [1¶], **Rom Gutman** [1], **and Malka Gorfine** [2]

1 Technion - Israel Institute of Technology, Haifa, 3200003, Israel **2** Department of Statistics and Operations Research, Tel Aviv University, Tel Aviv, 6997801, Israel ¶ Corresponding author

DOI: 10.21105/joss.08815

Software

- Review
- Repository
- Archive

Editor: Blake Rayfield

Reviewers:

- @rich2355
- @CamDavidsonPilon

Submitted: 30 June 2025
Published: 12 November 2025

Summary

Time-to-event (survival) analysis models the time until a pre-specified event occurs. When time is measured in discrete units or rounded into intervals, standard continuous-time models can yield biased estimators. In addition, the event of interest may belong to one of several mutually exclusive types, referred to as competing risks, where the occurrence of one event prevents the occurrence or observation of the others. *PyDTS* is an open-source Python package for analyzing discrete-time survival data with competing-risks. It provides regularized estimation methods, model evaluation metrics, variable screening tools, and a simulation module to support research and development.

Statement of need

Time-to-event analysis is applied when the outcome of interest is the time until a pre-specified event occurs. In some settings, the time variable is inherently or effectively discrete, for example, when time is measured in weeks or months, or when event times are rounded or grouped into intervals. Competing risks arise when observations are at risk of experiencing multiple mutually exclusive event types, such that the occurrence of one event precludes the occurrence or observation of the others. Discrete-time survival data with competing risks are encountered across a wide range of scientific disciplines. For instance, in healthcare, the time to death from cancer is often recorded in months, with death from other causes considered a competing event.

While excellent Python packages for continuous-time survival-analysis exist (Davidson-Pilon, 2019; Pölsterl, 2020), a comprehensive, user-friendly Python toolkit specifically designed for discrete-time survival analysis is still missing. Moreover, in the continuous-time setting, competing-risks data can often be analyzed using methods developed for non-competing events, since the full likelihood function factorizes into separate likelihoods for each cause-specific hazard function (Kalbfleisch & Prentice, 2011). In contrast, this factorization does not hold in the discrete-time setting (Lee et al., 2018; Meir & Gorfine, 2025), and dedicated estimation procedures are required to correctly account for the competing risk structure.

PyDTS bridges this gap by providing tools for analyzing discrete-time survival data with competing risks, designed to support both expert and non-expert researchers. Specifically, it offers:

- Discrete-time competing-risks regression models, based on the methods of Lee et al. (2018) and Meir & Gorfine (2025).
- Automated procedures for hyperparameter tuning.
- Sure Independence Screening methods for feature selection (Zhao & Li, 2012).

- Model evaluation metrics for predictive accuracy and calibration (Meir & Gorfine, 2025).
- Simulation tools for generating synthetic datasets for research and testing.

To the best of our knowledge, *PyDTS* is the first open-source Python package dedicated for discrete-time survival analysis with competing risks. Details on the statistical models and methods implemented in *PyDTS* are summarized in the package documentation and described in great detail in Meir & Gorfine (2025).

Key Features

PyDTS can be easily installed via *PyPI* as follows:

```
pip install pydts
```

It includes the following key features:

1. **Estimation procedures:** Two methods are implemented, `TwoStagesFitter` of Meir & Gorfine (2025), and `DataExpansionFitter` of Lee et al. (2018). The `TwoStagesFitter` supports both regularization and the inclusion of time-dependent covariates, features that are not available in the `DataExpansionFitter` implementation.
2. **Sure Independence Screening:** The `SISTwoStagesFitter` class implements the Sure Independence Screening (SIS) of Zhao & Li (2012). SIS is a powerful dimensionality reduction technique designed for ultra-high-dimensional settings, where the number of covariates far exceeds the number of observations, a situation often encountered in genomic studies and other high-throughput domains. It works by filtering out a large number of uninformative covariates based on their marginal association with the outcome. After screening, penalized variable selection methods (e.g., LASSO) are typically applied to the reduced set of covariates to perform more refined modeling and selection.
3. **Evaluation Metrics:** The package includes functions for computing key performance metrics for discrete-time survival data with competing risks and right-censoring, including the cause-specific cumulative/dynamic area under the receiver operating characteristic curve (AUC) and the Brier score (BS). Formal definitions of all implemented evaluation metrics are provided in Meir & Gorfine (2025).
4. **Hyperparameters tuning:** The package provides automated procedures for hyperparameter selection, including grid search combined with cross-validation, enabling robust model calibration and improved generalization performance.
5. **Data Generation:** The `EventTimesSampler` module facilitates the generation of discrete-time survival data with competing risks and right censoring. Given user-specified model parameters, including the number of discrete event times, true regression coefficients, and covariate values for each observation, `EventTimesSampler` simulates both event times and event types. The module supports two types of right censoring: administrative censoring, applied when the simulated event time exceeds a user-defined maximum follow-up duration, and random censoring, which can be either covariate-dependent or independent. This flexible simulation framework is useful for benchmarking models, testing estimation procedures, and conducting methodological research.

Case Study

The utility of *PyDTS* is demonstrated through an analysis of patients' length of stay (LOS) in intensive care unit (ICU), conducted by Meir & Gorfine (2025). This analysis uses the publicly accessible, large-scale Medical Information Mart for Intensive Care (MIMIC-IV, version 2.0) dataset (Goldberger et al., 2000; Johnson et al., 2022).

Meir & Gorfine (2025) developed a discrete-time survival model to predict ICU LOS based on patients' clinical characteristics at admission. The dataset comprises 25,170 ICU patients. For each patient, only the last admission is considered, and features related to prior admission history are included. The LOS is recorded in discrete units from 1 to 28 days, resulting in many patients sharing the same event time on each day. Three competing events are considered: discharge to home (69.0%), transfer to another medical facility (21.4%), and in-hospital death (6.1%). Patients who left the ICU against medical advice (1.0%) are treated as right-censored, and administrative censoring is applied to those hospitalized for more than 28 days (2.5%). The analysis includes 36 covariates per patient. For a full description of the data, see Meir & Gorfine (2025).

Three estimation procedures were compared: the method of Lee et al. (2018) without regularization, two-step approach of Meir & Gorfine (2025) without and with LASSO regularization. When applying the two-step procedure with LASSO regularization, we need to specify the hyperparameters that control the amount of regularization applied to each competing event. *PyDTS* provides functionality for tuning these hyperparameters via K-fold cross-validation. By default, the optimal values are those that maximize the out-of-sample global-AUC metric, as defined in Meir & Gorfine (2025), Appendix I. Additional tuning options are also available. Here, a grid search with 4-fold cross-validation was performed to select the optimal hyperparameters that maximize the global-AUC. The code below illustrates such tuning procedure

```python
import numpy as np
from pydts.cross_validation import PenaltyGridSearchCV

penalizers = np.exp(range(-12, -1))
penalty_cv_search = PenaltyGridSearchCV()
gauc_cv_results = penalty_cv_search.cross_validate(
    full_df=mimic_df, l1_ratio=1, penalizers=penalizers, n_splits=4)
```

where `mimic_df` is the full dataframe containing the covariates, an event-type column, an event-time column, and an event indicator column; `penalizers` is the set of penalization values evaluated for each risk, denoted as η_j, with $j = 1, 2, 3$; `n_splits` is the number of folds; and `l1_ratio` controls the balance between L1 and L2 regularization, with `l1_ratio = 1` corresponding to pure L1 (LASSO) regularization. Figure 1 presents the results of the selection procedure. Panels A–C illustrate the number of non-zero estimated coefficients, denoted as β_j, as a function of the regularization hyperparameter η_j for each competing event. Panels D–F illustrate the coefficient values as a function of η_j for each competing event. Panels G–I illustrate the $\widehat{\text{AUC}}_j(t)$ metric for the selected set of η_j, $j = 1, 2, 3$. A comprehensive description of the case study settings and results can be found in Meir & Gorfine (2025).

Additional examples demonstrating *PyDTS*'s functionality are also provided in Meir & Gorfine (2025) and in the package documentation. These include analyses with regularized regression across varying sample sizes and levels of covariates' correlation, as well as the application of Sure Independence Screening in ultra-high-dimensional settings (Zhao & Li, 2012). These examples make use of the package's built-in data generation tools, underscoring its usefulness for methodological development and evaluation.

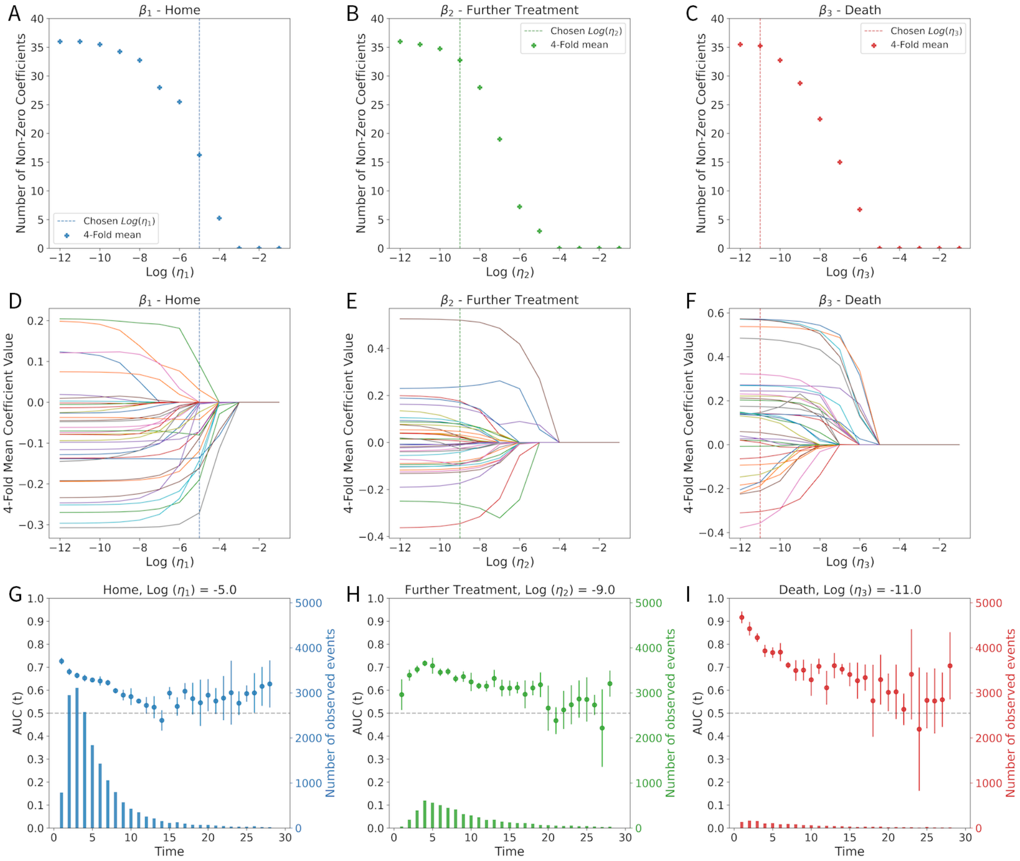

Figure 1: MIMIC dataset - LOS analysis. Regularized regression with 4-fold CV. The selected values of η_j are shown in dashed-dotted lines on panels **A**-**F**. **A**-**C.** Number of non-zero coefficients for $j = 1, 2, 3$. **D**-**F.** The estimated coefficients, as a function of η_j, $j = 1, 2, 3$. **G**-**I.** Mean (and SD bars) of the 4 folds $\widehat{\text{AUC}}_j(t)$, $j = 1, 2, 3$, for the selected values $\log \eta_1 = -5$, $\log \eta_2 = -9$ and $\log \eta_3 = -11$. The number of observed events of each type is shown by bars.

Acknowledgemnts

T.M. is supported by the Israeli Council for Higher Education (Vatat) fellowship in data science via the Technion; M.G. work was supported by the ISF 767/21 grant and Malag competitive grant in data science (DS).

References

Davidson-Pilon, C. (2019). Lifelines: Survival analysis in python. *Journal of Open Source Software*, *4*(40), 1317. https://doi.org/10.21105/joss.01317

Goldberger, A. L., Amaral, L. A. N., Glass, L., Hausdorff, J. M., Ivanov, P. Ch., Mark, R. G., Mietus, J. E., Moody, G. B., Peng, C.-K., & Stanley, H. E. (2000). PhysioBank, PhysioToolkit, and PhysioNet: Components of a new research resource for complex physiologic signals. *Circulation*, *101*(23). https://doi.org/10.1161/01.CIR.101.23.e215

Johnson, A., Bulgarelli, L., Pollard, T., Horng, S., Celi, L. A., & Mark, R. (2022). MIMIC-IV (version 2.0). *PhysioNet*. https://doi.org/10.13026/7vcr-e114

Kalbfleisch, J. D., & Prentice, R. L. (2011). *The statistical analysis of failure time data* (2nd ed.). John Wiley & Sons. https://doi.org/10.1002/9781118032985

Lee, M., Feuer, E. J., & Fine, J. P. (2018). On the analysis of discrete time competing risks data. *Biometrics*, *74*(4), 1468–1481. https://doi.org/10.1111/biom.12881

Meir, T., & Gorfine, M. (2025). Discrete-Time Competing-Risks Regression with or without Penalization. *Biometrics*, *81*. https://doi.org/10.1093/biomtc/ujaf040

Pölsterl, S. (2020). Scikit-survival: A library for time-to-event analysis built on top of scikit-learn. *Journal of Machine Learning Research*, *21*(212), 1–6. https://dl.acm.org/doi/10.5555/3455716.3455928

Zhao, S. D., & Li, Y. (2012). Principled sure independence screening for cox models with ultra-high-dimensional covariates. *Journal of Multivariate Analysis*, *105*(1), 397–411. https://doi.org/10.1016/j.jmva.2011.08.002