

An Interpretable Machine Learning Framework for Exploratory Survival and Safety Analysis Using Random Survival Forests and SHAP

Kun Shao, Evopoint Biosciences Co.,Ltd

ABSTRACT

In clinical trials, standardized CDISC Analysis Data Model (ADaM) datasets contain rich baseline characteristics and complex efficacy data results. Traditional survival models, particularly the Cox proportional hazards model, remain central to confirmatory analyses. However, exploratory clinical trial questions often involve high-dimensional baseline covariates, non-linear relationships, potential interactions, and a strong need for clinical interpretability. This paper presents an ADaM-compatible, interpretable machine learning framework for exploratory survival and safety analyses using Random Survival Forests (RSF) and SHapley Additive exPlanations (SHAP). The framework includes endpoint definition, feature governance to reduce information leakage, dual-path modeling with full-feature RSF and Cox-Lasso-selected RSF, SHAP-based interpretation, and traditional statistical benchmarking. The workflow is demonstrated using data from a clinical trial, focusing on Progression-Free Survival (PFS) and targeted risk endpoints derived from standard ADSL, ADTTE, and ADAE datasets. The proposed framework complements traditional statistical methods by supporting interpretable hypothesis generation, risk profiling, and exploratory signal detection within familiar clinical trial data structures.

INTRODUCTION

Clinical trial data are increasingly rich, structured, and multidimensional. In oncology studies, a typical analysis environment may include demographic variables, disease history, tumor characteristics, prior therapy, biomarkers, baseline laboratory results, treatment assignments, and multiple time-to-event efficacy and safety endpoints. The CDISC ADaM standard provides a common structure for these data, enabling traceable, analysis-ready datasets such as ADSL for subject-level characteristics, ADTTE for time-to-event endpoints, and ADAE for adverse events.

For confirmatory survival analyses, the Cox proportional hazards model and related semi-parametric methods remain foundational. These methods are transparent, statistically well understood, and well aligned with regulatory reporting. Nevertheless, exploratory analyses often ask a different type of question. Instead of estimating a pre-specified treatment effect under a confirmatory estimand, analysts may want to identify baseline factors associated with earlier disease progression, delayed response, or increased risk of a targeted adverse event. These exploratory questions may involve many candidate predictors, non-linear relationships, sparse event counts, and complex interactions that are not easily captured by a traditional linear predictor.

Machine learning methods can help address these exploratory needs, but they must be implemented carefully in clinical trial settings. A flexible algorithm is not sufficient by itself. The workflow must preserve the discipline of endpoint definition, time alignment, feature governance, leakage prevention, reproducibility, and interpretability. For clinical trial teams, especially statisticians, statistical programmers,

and clinical data scientists, an effective method should integrate with standard ADaM datasets and produce outputs that can be reviewed, explained, and challenged.

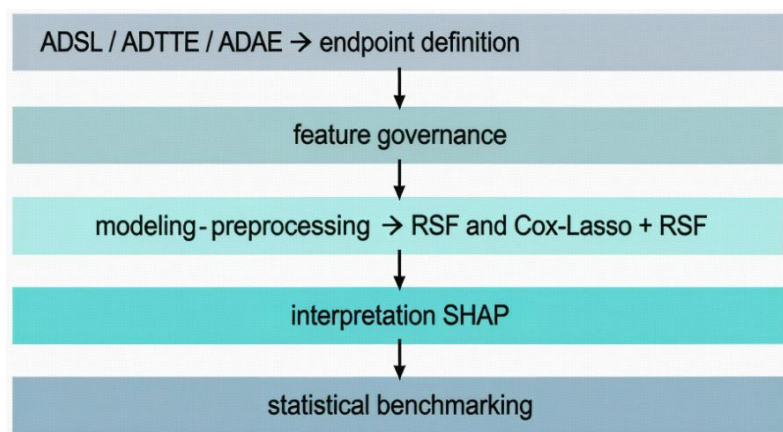
This paper describes a practical framework that combines Random Survival Forests (RSF) with SHapley Additive exPlanations (SHAP) for exploratory survival and safety analyses. RSF provides a non-parametric survival modeling approach that can capture non-linear effects and interactions without imposing a proportional hazards assumption. SHAP provides global and local explanations of model predictions, helping translate model behavior into clinically interpretable patterns. To improve stability in high-dimensional settings, the framework also includes a hybrid path in which Cox-Lasso is used for feature selection before fitting a tuned RSF model.

The goal of this paper is not to replace traditional statistical methods. Rather, it is to show how interpretable machine learning can complement them in a structured, ADaM-compatible exploratory workflow. PFS and targeted risk are used as demonstration endpoints to illustrate how the same framework can be adapted across efficacy and safety analyses.

FRAMEWORK OVERVIEW

The proposed workflow contains five layers: endpoint definition, feature governance, modeling, interpretation, and statistical benchmarking. Figure 1 provides a conceptual overview of the framework.

Figure 1. Proposed ADaM-compatible interpretable machine learning workflow



ENDPOINT DEFINITION LAYER

The first layer translates an ADaM analysis question into a survival modeling target. For each endpoint, the analyst defines the analysis population, time origin, event indicator, event time, censoring rule, and any competing event handling. This step is intentionally separated from the modeling algorithm because the scientific meaning of the endpoint must be established before any machine learning method is applied.

For ADTTE-based efficacy endpoints, such as PFS, the endpoint is typically derived from parameter-level records using `PARAMCD`, `AVAL`, and censoring variables. For ADAE-based safety endpoints, such as targeted risk, the endpoint may require identifying adverse events that meet clinical criteria such as body system, relationship to treatment, toxicity grade, and event start date. In all cases, the endpoint definition

should be documented in terms familiar to clinical trial statisticians: population, variable source, time scale, event definition, censoring, and exclusions.

FEATURE GOVERNANCE LAYER

The second layer constructs the baseline feature matrix from ADSL. This layer is critical because exploratory machine learning models can easily learn information that is not appropriate for prediction at baseline. To reduce information leakage, the framework excludes subject identifiers, date variables, end-of-treatment variables, disposition variables, death-related fields, data cutoff fields, follow-up end variables, and other fields that are temporally close to or downstream of the outcome.

The workflow also removes high-cardinality text fields and redundant numerical encodings of categorical variables when they do not provide independent clinical information. For example, when both a categorical variable and a numeric companion code are present, only the clinically interpretable representation should be retained unless there is a specific reason to include both. Treatment assignment variables can be handled deliberately, with one primary treatment variable retained while redundant arm or treatment coding fields are excluded.

This layer is not merely a programming convenience. It is a bridge between estimand thinking and machine learning implementation. By explicitly defining which variables are eligible as baseline predictors, the analyst improves the credibility and interpretability of the exploratory findings.

MODELING LAYER

The modeling layer contains two complementary paths.

The first path fits a full-feature RSF model using all eligible baseline predictors after preprocessing. This path is useful for broad exploration because it allows the model to detect non-linear relationships and interactions across the entire candidate feature set.

The second path uses Cox-Lasso for high-dimensional feature selection followed by a tuned RSF model on the reduced feature set. Cox-Lasso provides a sparse survival model by applying L1 regularization. In the implementation described here, the regularization parameter is selected using cross-validation based on Harrell's C-index. Features with non-zero coefficients are retained for the subsequent RSF model. If no non-zero features are selected, a fallback rule can retain the top features by absolute coefficient magnitude to avoid an empty modeling set.

The selected features are then passed to an RSF model with hyperparameters tuned through cross-validation. This hybrid strategy uses a convex sparse model for noise reduction and feature screening, then uses RSF to capture non-linear risk patterns in the lower-dimensional feature space.

INTERPRETABILITY LAYER

The interpretability layer uses SHAP to explain the fitted RSF model. SHAP values quantify the contribution of each feature to a model prediction relative to a background distribution. In this workflow, SHAP is applied to the RSF prediction function, producing explanations of the model-derived risk score.

Two outputs are particularly useful for clinical trial exploratory analysis. A SHAP summary plot ranks features by their average absolute contribution and shows whether high or low feature values are associated with higher model-predicted risk. A SHAP dependence plot focuses on one feature and displays potential non-linear relationships between the feature value and the risk contribution.

SHAP explanations should be interpreted as explanations of the fitted model, not as causal effects. They are useful for hypothesis generation, clinical review, and communication, but they remain model-dependent and data-dependent.

STATISTICAL BENCHMARKING LAYER

The final layer uses traditional statistical methods to benchmark or further examine machine learning findings. Cox proportional hazards models can provide hazard ratios, confidence intervals, and p-values for selected features. Kaplan-Meier curves and log-rank tests can display survival differences across clinically meaningful groups or thresholds suggested by exploratory analysis.

This layer is included to reinforce the relationship between machine learning and standard statistical practice. The framework is designed to complement, not replace, conventional methods. Machine learning can identify patterns worth examining; traditional methods can help summarize and communicate selected findings in familiar statistical terms.

IMPLEMENTATION USING STANDARD ADAM DATASETS

The framework was implemented in Python using standard analysis datasets. ADSL provided subject-level baseline predictors. ADTTE provided time-to-event efficacy endpoints. ADAE provided adverse event records for targeted safety endpoint derivation.

The main Python libraries included `pandas` and `numpy` for data handling, `scikit-learn` for preprocessing and model selection, `scikit-survival` for survival modeling, `shap` for model interpretation, `matplotlib` for visualization, and `lifelines` for traditional Cox, Kaplan-Meier, and log-rank analyses.

PREPROCESSING

Eligible baseline variables were separated into numerical and categorical predictors. Numerical variables were imputed using the median and standardized. Categorical variables were imputed using a constant value and encoded using one-hot encoding with handling for unknown categories. To avoid unstable sparse representations, categorical variables with excessive cardinality were excluded or capped according to the preprocessing rule.

The survival target was converted into the structured array format used by `scikit-survival`, with one Boolean event indicator and one numeric time variable. This target structure allowed a consistent modeling interface across PFS and targeted risk.

MODEL ASSESSMENT

Model discrimination was assessed using Harrell's C-index. For the full-feature RSF path, the model was fit on a training set and evaluated on a held-out test set. For the hybrid path, Cox-Lasso feature selection

was performed within the training process, followed by RSF hyperparameter tuning using cross-validation. The tuned model was then evaluated on the held-out test set.

C-index is appropriate for assessing whether the model ranks subjects according to earlier or later event occurrence. In this exploratory setting, a C-index between 0.65 and 0.7 would suggest modest to moderate discriminative ability, indicating that baseline covariates contain some prognostic signal for ranking subjects by relative event risk. It should not be interpreted as a direct measure of clinical utility, calibration, treatment effect, or causal validity.

REPRODUCIBILITY

The implementation used fixed random seeds for training/test splitting, sampling, and model fitting where applicable. Reproducibility can be further improved by recording package versions, input dataset creation dates, analysis population counts, event counts, selected hyperparameters, and exported figure names.

CASE STUDY APPLICATIONS

Two endpoint-level case studies were used to demonstrate the portability of the proposed framework across efficacy and safety analyses. These applications illustrate how the workflow can be implemented using standard ADaM datasets while keeping the reusable methodology as the focus of the paper.

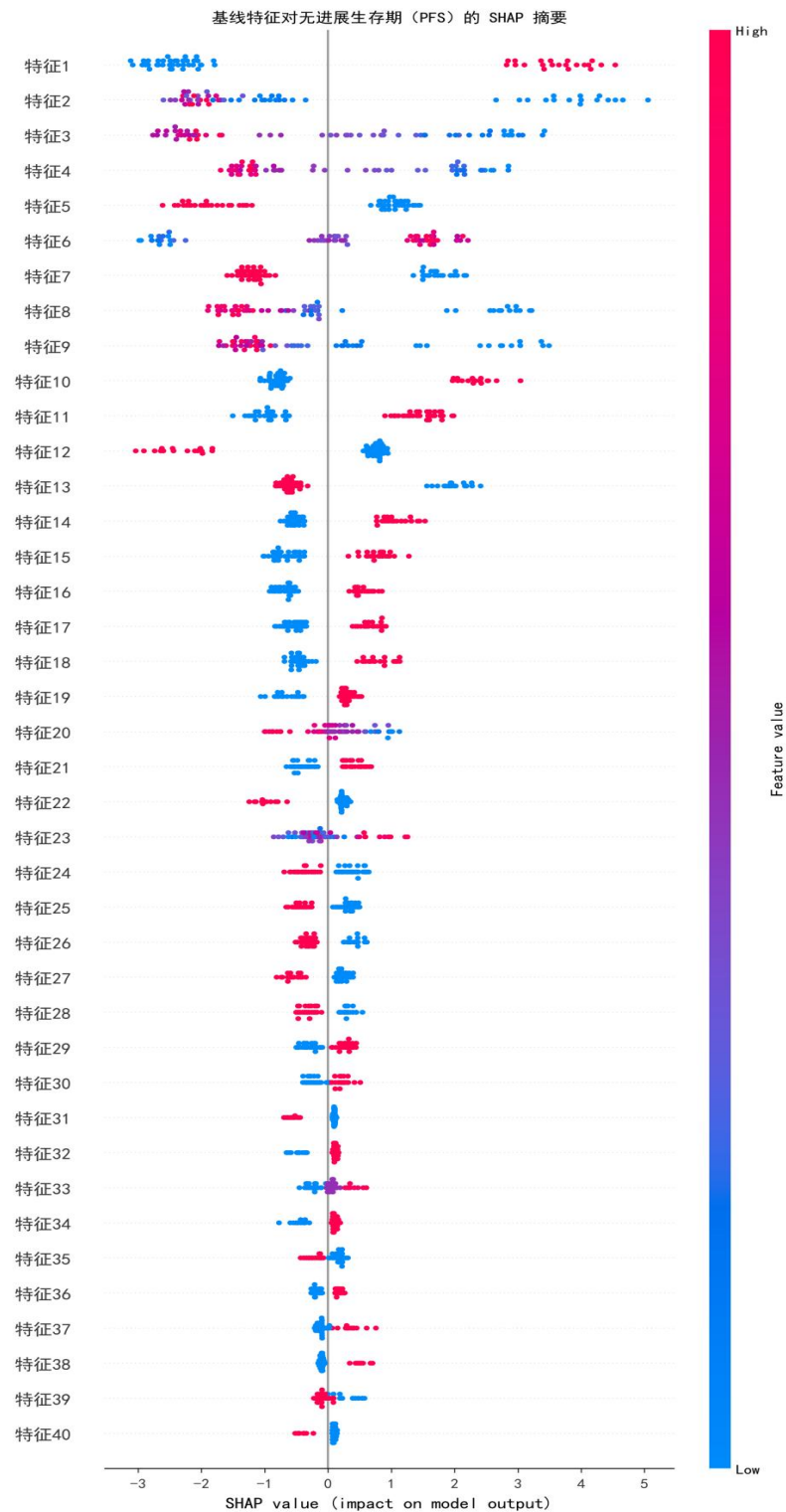
DEMONSTRATION 1: PROGRESSION-FREE SURVIVAL

PFS was used as a standard efficacy time-to-event example derived from ADTTE. Subjects were selected from the efficacy analysis population where available. PFS records were identified using ``PARAMCD = 'PFS'``. The analysis time was taken from ``AVAL``, and the event indicator was derived from the censoring variable, with ``CNSR = 0`` representing a PFS event and ``CNSR = 1`` representing censoring according to common ADTTE convention.

After merging PFS records with ADSL, records with missing or non-positive time values were excluded. Eligible baseline variables were preprocessed and passed through both modeling paths: full-feature RSF and Cox-Lasso-selected RSF.

The full-feature RSF path used all eligible baseline covariates after preprocessing. Its purpose was broad signal discovery: the model could evaluate the complete baseline feature space and capture non-linear relationships and interactions without imposing a pre-selection step. The resulting SHAP summary plot therefore reflected the relative contribution of the full candidate feature set to the RSF risk score.

Figure 2. SHAP summary plot for full-feature PFS RSF



In the SHAP summary plot, features are ordered by their average absolute contribution to the RSF risk score. Points to the right of zero increase the model-predicted risk, while points to the left decrease it; color

indicates whether the observed feature value is relatively high or low. To further examine individual feature effects, SHAP dependence plots were generated for the three highest-ranked features. In a dependence plot, the x-axis shows the feature value and the y-axis shows its SHAP contribution, allowing potential threshold effects, non-linear patterns, or heterogeneous effects across subjects to be visually assessed.

Figure 3. Combined SHAP dependence plots for the top three ranked features from full-feature PFS RSF

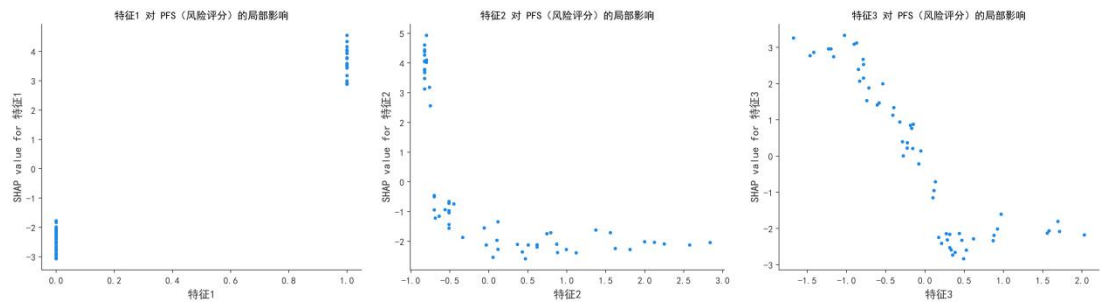
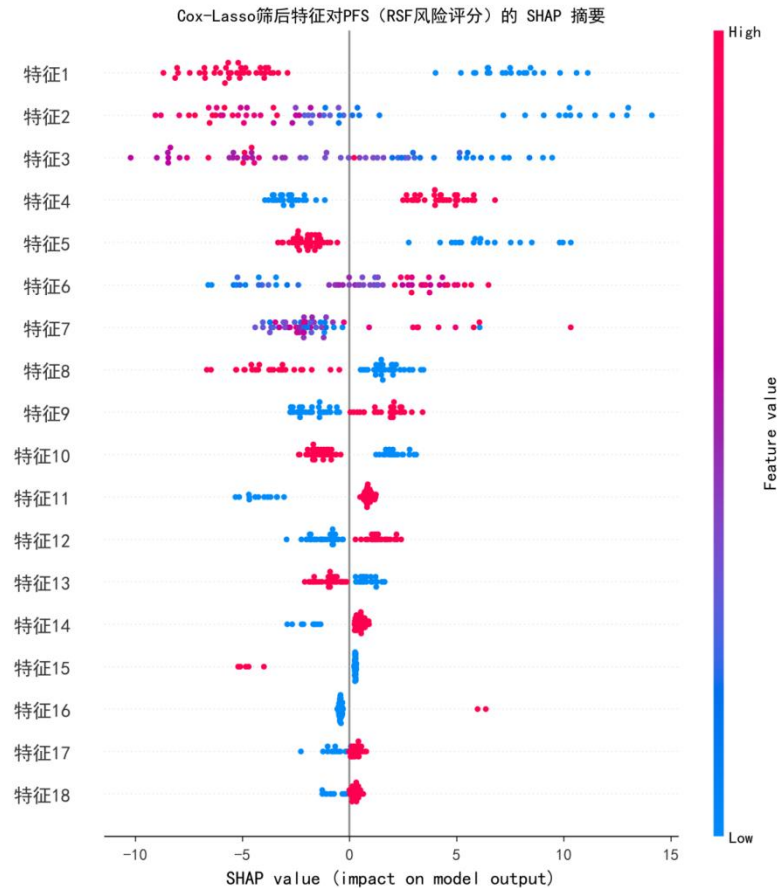


Figure 3 presents the Top 3 SHAP dependence plots side by side for the full-feature PFS model. Feature labels are anonymized for presentation (Feature 1–3); internally, Feature 1 corresponds to the highest mean |SHAP| rank in Figure 2. These local views complement the global ranking in the summary plot by illustrating how each top-ranked baseline covariate shifts the model-predicted risk across its observed range.

The Cox-Lasso-selected RSF path first applied Cox-Lasso to the preprocessed feature matrix, using cross-validation to select the regularization parameter and retaining features with non-zero coefficients. A tuned RSF model was then fit on this reduced feature set. Compared with the full-feature RSF, this hybrid path was designed to reduce noise, improve interpretability, and focus SHAP explanations on a smaller set of clinically reviewable features.

Figure 4. SHAP summary plot for Cox-Lasso-selected PFS RSF



For the Cox-Lasso-selected RSF model, the SHAP summary plot should be interpreted within the reduced feature space selected by Cox-Lasso. Therefore, the ranking reflects the relative importance of features retained by the sparse survival model, not the full original baseline feature set. The corresponding Top 3 dependence plots provide feature-level views of how each selected predictor contributes to the RSF risk score across its observed range.

In the hybrid path, Cox-Lasso selected a reduced set of baseline features from the broader candidate pool. Compared with the full-feature RSF path, the Cox-Lasso screening step produced a more focused predictor set and a clearer SHAP-based exploratory profile, illustrating the practical value of the dual-path framework design. These findings are consistent with the purpose of the framework: to identify clinically interpretable variables that may warrant further review, rather than to claim confirmatory prognostic effects.

DEMONSTRATION 2: TARGETED RISK

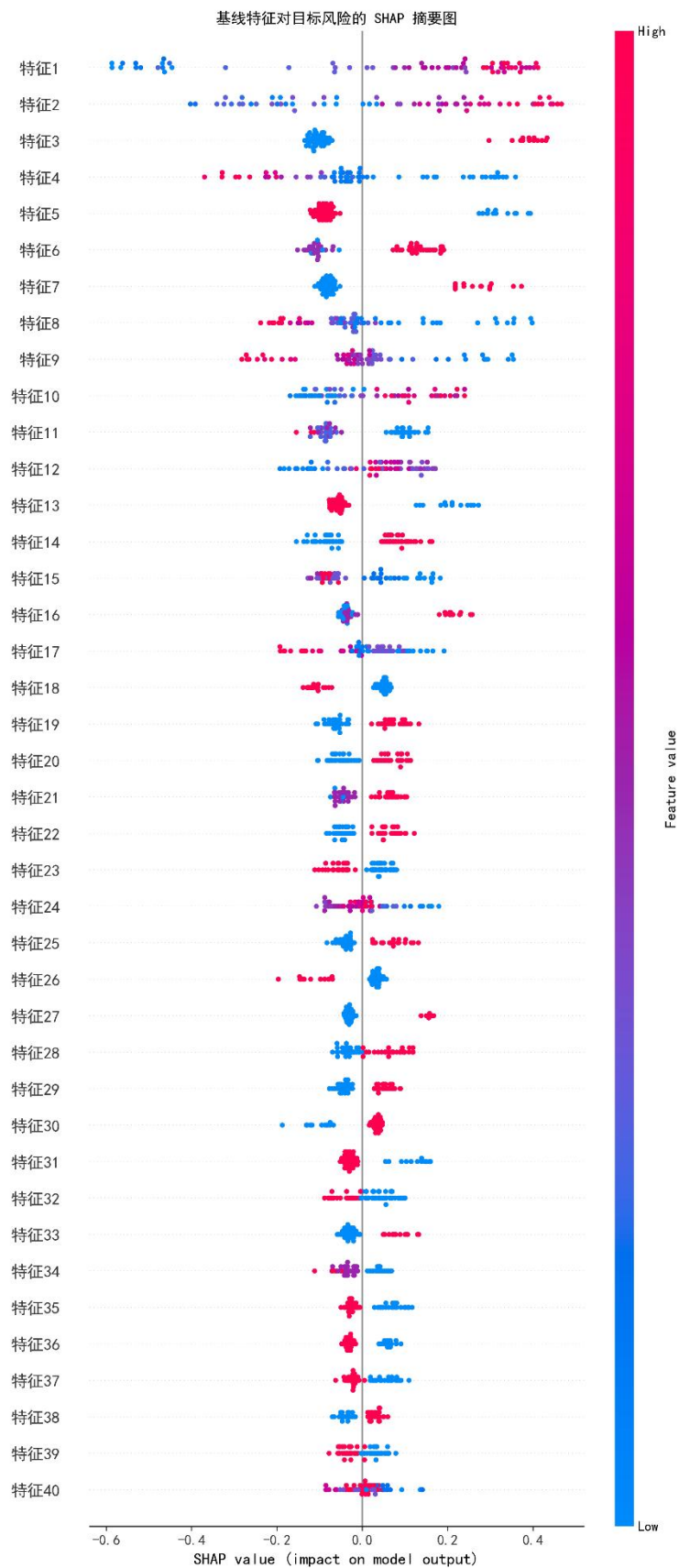
A targeted risk endpoint was used as a safety endpoint derived from ADAE. The event definition identified treatment-related adverse events in a predefined organ system with toxicity grade 2 or higher. For each subject, the earliest qualifying adverse event date was used as the target event date. The time origin was first treatment date where available. Censoring time was derived from treatment end, last known alive date, or reference end date according to the available data hierarchy.

The safety endpoint also incorporated a simplified competing event structure. Drug withdrawal due to adverse events and death were identified as competing events when they occurred before the target risk event. For the RSF model, the analysis used a cause-specific simplification in which competing events were treated as censored for the target event. This approach is practical for demonstration but should be described clearly, as it differs from a Fine-Gray subdistribution hazard model.

As with PFS, the targeted risk endpoint was analyzed using both the full-feature RSF path and the Cox-Lasso-selected RSF path.

The full-feature RSF path applied the same preprocessing and feature governance rules to eligible ADSL baseline covariates. The resulting SHAP summary plot characterizes the relative contribution of the candidate baseline feature set to the RSF risk score for the targeted risk endpoint.

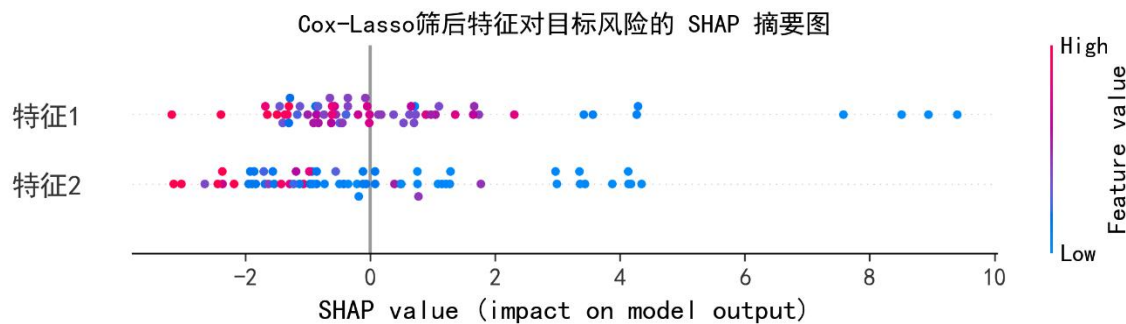
Figure 5. SHAP summary plot for full-feature targeted risk RSF



In the SHAP summary plot, features are ordered by their average absolute contribution to the RSF risk score for the targeted risk endpoint. Feature labels are anonymized for presentation (Feature 1, Feature 2, and so on).

The Cox-Lasso-selected RSF path reduced the baseline feature space before RSF modeling. Given the limited number of selected features and the exploratory nature of the safety endpoint, results from this path should be presented as potential safety signals for clinical review rather than definitive risk factors.

Figure 6. SHAP summary plot for Cox-Lasso-selected targeted risk RSF



For the Cox-Lasso-selected RSF model, the SHAP summary plot should be interpreted within the reduced feature space retained by Cox-Lasso sparse screening. Therefore, the ranking reflects the relative importance of features retained by the sparse survival model rather than the full original baseline feature set.

To connect the exploratory machine learning results with the statistical benchmarking layer of the framework, traditional survival analyses were applied to the leading risk factor identified from the preceding SHAP summary plot. In this demonstration, the top-ranked SHAP covariate was selected as the benchmarking variable, and subjects were grouped using a clinically motivated threshold for this single risk factor. The association with the targeted risk endpoint was then summarized using univariable and multivariable Cox models, Kaplan-Meier curves, and log-rank comparison. This step evaluates whether the SHAP-selected risk factor also separates subjects in a conventional survival analysis framework.

Figure 7. Kaplan-Meier comparison for targeted risk by baseline covariate grouping

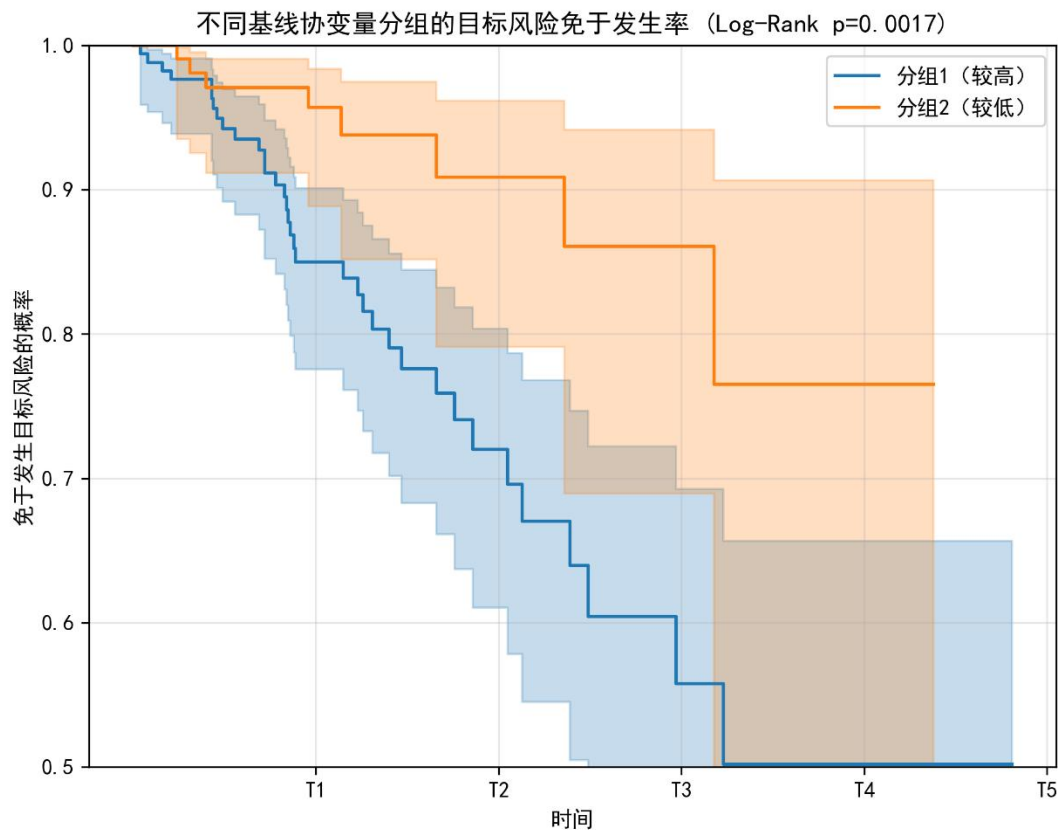


Figure 7 provides a practical supplemental check using familiar survival methods. By translating the top-ranked SHAP risk factor into a clinically reviewable baseline grouping, the Kaplan-Meier curves show a statistically significant difference between the two groups for targeted risk (log-rank $p = 0.0017$, < 0.05). This finding supports the clinical interpretability of the SHAP-selected risk factor and illustrates how the statistical benchmarking layer can help translate model-derived signals into evidence that is easier for clinical teams to review..

DISCUSSION

This paper presents a practical framework for applying interpretable machine learning to exploratory clinical trial survival and safety analyses. The framework is designed around ADaM data structures and clinical trial analysis principles. Rather than starting with an algorithm and searching for an application, the workflow begins with endpoint definition and feature governance, then applies RSF and SHAP within that disciplined structure.

The first contribution is ADaM compatibility. ADSL, ADTTE, and ADAE are familiar to clinical trial teams, and many exploratory questions can be expressed using these datasets. By mapping ADSL to baseline predictors, ADTTE to efficacy time-to-event endpoints, and ADAE to targeted safety endpoints, the framework can be implemented without requiring a new data model.

The second contribution is interpretability. RSF can capture complex survival patterns, but its value in clinical settings depends on whether the results can be understood and reviewed. SHAP summary and dependence plots provide a practical way to identify which features drive model predictions and how feature values relate to model-predicted risk. These outputs can support clinical discussion, data review, and hypothesis generation.

The third contribution is the dual-path modeling design. The full-feature RSF path supports broad exploration, while the Cox-Lasso-selected RSF path helps reduce dimensionality and noise in high-dimensional baseline feature spaces. This is particularly useful when the number of candidate predictors is large relative to the number of events, a common situation in targeted safety analyses or biomarker-rich oncology studies.

The fourth contribution is the explicit connection to traditional statistical methods. The framework does not position machine learning as a replacement for Cox models, Kaplan-Meier curves, or log-rank tests. Instead, it uses machine learning to identify patterns and uses traditional methods to summarize selected findings in familiar statistical terms. This relationship is important in clinical trial analysis settings, where reproducibility, traceability, and communication across statistical programming and clinical teams are central.

Several limitations should be acknowledged. First, the analyses are exploratory and should not be interpreted as confirmatory evidence. Feature selection, SHAP ranking, and endpoint-specific modeling can introduce selection effects and multiplicity concerns. In addition, machine learning methods generally require sufficient sample size and event counts to support stable model fitting, feature selection, and interpretation. When the number of events is limited relative to the number of candidate predictors, model performance estimates, selected features, and SHAP rankings may be unstable. Therefore, the sample size, event count, and feature dimensionality should be reported together, and findings from small or sparse datasets should be interpreted with particular caution. Second, SHAP explanations describe the fitted model, not causal relationships. Third, C-index assesses ranking performance but does not establish calibration, clinical utility, or transportability. Fourth, the targeted risk demonstration uses a cause-specific simplification for competing events. More formal competing-risk or multi-state methods may be appropriate for endpoint-specific confirmatory analyses. Finally, external validation and bootstrap stability assessment would strengthen confidence in the reproducibility of selected features.

Future extensions could include Fine-Gray models for competing-risk endpoints, multi-state models for response and progression pathways, bootstrap-based feature stability summaries, calibration assessment, automated generation of SHAP and Cox benchmark reports, and integration with standardized clinical trial reporting pipelines.

CONCLUSION

Random Survival Forests combined with SHAP provide a practical, interpretable, and ADaM-compatible approach to exploratory survival and safety analysis in clinical trials. The proposed workflow begins with clear endpoint definition and baseline feature governance, then applies flexible survival modeling and SHAP-based interpretation within a structure that can be reviewed and reproduced by statistical

programming and clinical teams. By positioning machine learning as a complement to—not a substitute for—Cox models, Kaplan-Meier displays, and log-rank testing, the framework supports transparent, hypothesis-generating analyses that remain aligned with familiar trial reporting practice.

REFERENCES

1. Clinical Data Interchange Standards Consortium (CDISC). Analysis Data Model Implementation Guide (ADaMIG), Version 1.3. CDISC: Austin, TX, USA; 2021. Available at: <https://www.cdisc.org/standards/foundational/adam/adamig-v1-3>
2. Cox DR. Regression models and life-tables. *Journal of the Royal Statistical Society: Series B (Methodological)*. 1972;34(2):187-220.
3. Kaplan EL, Meier P. Nonparametric estimation from incomplete observations. *Journal of the American Statistical Association*. 1958;53(282):457-481.
4. Mantel N. Evaluation of survival data and two new rank order statistics arising in its consideration. *Cancer Chemotherapy Reports*. 1966;50(3):163-170.
5. Harrell FE Jr, Califf RM, Pryor DB, Lee KL, Rosati RA. Evaluating the yield of medical tests. *Statistics in Medicine*. 1982;1(2):153-157.
6. Tibshirani R. The lasso method for variable selection in the Cox model. *Statistics in Medicine*. 1997;16(4):385-406.
7. Ishwaran H, Kogalur UB, Blackstone EH, Lauer MS. Random survival forests. *The Annals of Applied Statistics*. 2008;2(3):841-860.
8. Lundberg SM, Lee SI. A unified approach to interpreting model predictions. In: *Advances in Neural Information Processing Systems 30 (NeurIPS 2017)*; 2017:4765-4774.
9. Fine JP, Gray RJ. A proportional hazards model for the subdistribution of a competing risk. *Journal of the American Statistical Association*. 1999;94(446):496-509.
10. Pedregosa F, Varoquaux G, Gramfort A, Michel V, et al. Scikit-learn: machine learning in Python. *Journal of Machine Learning Research*. 2011;12:2825-2830.
11. Pölsterl S. scikit-survival: a library for time-to-event analysis built on top of scikit-learn. *Journal of Machine Learning Research*. 2020;21(212):1-6.
12. Davidson-Pilon C. lifelines: survival analysis in Python. *Journal of Open Source Software*. 2019;4(40):1317.

APPENDIX 1. CORE CODE FOR FULL-FEATURE PFS RSF + SHAP

The excerpt below shows the core workflow: PFS endpoint construction, preprocessing, full-feature RSF modeling, anonymized SHAP interpretation, and figure export. Helper functions for ADSL feature governance (``select_adsl_feature_columns``, etc.) are omitted for brevity.

```

# --- PFS endpoint from ADTTE + efficacy population from ADSL ---
pfs = adtte.loc[adtte["PARAMCD"].astype(str).str.strip() == "PFS",
               ["USUBJID", "AVAL", "CNSR"]].rename(
    columns={"AVAL": "PFS_MONTHS", "CNSR": "PFS_CNSR"})
adsl = adsl[adsl["EFFFL"].astype(str).str.strip().isin(["Y", "是"])]
df = adsl.merge(pfs, on="USUBJID", how="left")
df["PFS_Event"] = df["PFS_CNSR"] == 0
df["Time"] = pd.to_numeric(df["PFS_MONTHS"], errors="coerce")
df = df[df["Time"].notna() & (df["Time"] > 0) & df["PFS_Event"].notna()].copy()
y = np.array(list(zip(df["PFS_Event"].astype(bool),
                     df["Time"].astype(float))),
              dtype=[("Event", "?"), ("Time", "<f8")])

# --- Baseline features + preprocessing ---
feature_cols = select_adsl_feature_columns(df, adsl_columns)
X = df[feature_cols].copy()
preprocessor = ColumnTransformer([ # median impute + scale; one-hot for
    categorical_features
    ("num", Pipeline([("imputer", SimpleImputer(strategy="median")),
                      ("scaler", StandardScaler())]), numeric_features),
    ("cat", Pipeline([("imputer", SimpleImputer(strategy="constant",
    fill_value="Missing")),
                      ("onehot", OneHotEncoder(handle_unknown="ignore",
    sparse_output=False,
    max_categories=25))]),
    categorical_features),
])
X_train, X_test, y_train, y_test = train_test_split(
    X, y, test_size=0.2, random_state=42, stratify=df["PFS_Event"])
X_train_df = pd.DataFrame(preprocessor.fit_transform(X_train),
                          columns=preprocessor.get_feature_names_out())
X_test_df = pd.DataFrame(preprocessor.transform(X_test),
                        columns=preprocessor.get_feature_names_out())

# --- Full-feature RSF ---
rsf = RandomSurvivalForest(n_estimators=200, min_samples_split=10,
                          min_samples_leaf=15, n_jobs=-1, random_state=42)
rsf.fit(X_train_df, y_train)
print(f"C-index (test): {rsf.score(X_test_df, y_test):.3f}")

# --- SHAP with anonymized display labels (Feature 1, Feature 2, ...) ---

```

```

explainer = shap.Explainer(rsf.predict, shap.sample(X_train_df, 100,
random_state=42))
shap_values = explainer(X_test_df)
mean_abs = np.abs(shap_values.values).mean(axis=0)
display_names, rank_by_index = build_anonymized_display_names(feature_names,
mean_abs)
shap.summary_plot(shap_values, X_test_df, feature_names=display_names,
show=False)
plt.savefig("shap_summary_pfs.png", dpi=300)

# Top-3 dependence plots (anonymized labels + internal mapping CSV)
for rank, idx in enumerate(np.argsort(-mean_abs)[:3], 1):
    shap.dependence_plot(display_names[idx], shap_values.values,
                        X_test_df.set_axis(display_names, axis=1),
                        interaction_index=None, show=False)

plt.savefig(f"shap_dependence_pfs_{rank:02d}_feature{rank_by_index[idx]:02d}
.png", dpi=300)

```

APPENDIX 2. CORE CODE FOR COX-LASSO-SELECTED PFS RSF + SHAP

The excerpt below shows the hybrid path: Cox-Lasso screening with cross-validated alpha selection, tuned RSF on the reduced feature set, and anonymized SHAP outputs. Shared endpoint and preprocessing steps are abbreviated.

```

# --- Same PFS endpoint / feature matrix / preprocessor as Appendix 1 ---
X_train_p = preprocessor.fit_transform(X_train)
X_test_p = preprocessor.transform(X_test)
all_feature_names = np.array(preprocessor.get_feature_names_out())

# --- Cox-Lasso: CV selects alpha (Harrell C-index), then non-zero features ---
cox_init = CoxnetSurvivalAnalysis(l1_ratio=1.0, alpha_min_ratio=0.01,
                                n_alphas=100, max_iter=200000)
cox_init.fit(X_train_p, y_train)
best_alpha, _ = choose_alpha_via_cv(X_train_p, y_train, cox_init.alphas_,
                                n_splits=5, random_state=42)

cox_final = CoxnetSurvivalAnalysis(l1_ratio=1.0, alphas=cox_init.alphas_,
                                alpha_min_ratio=0.01,
                                n_alphas=len(cox_init.alphas_),
                                max_iter=200000)
cox_final.fit(X_train_p, y_train)

```

```

best_alpha_idx = int(np.argmin(np.abs(cox_final.alphas_ - best_alpha)))
selected_mask = np.abs(cox_final.coef[:, best_alpha_idx]) > 1e-12
selected_feature_names = all_feature_names[selected_mask]

X_train_sel_df = pd.DataFrame(X_train_p[:, selected_mask],
                              columns=selected_feature_names)
X_test_sel_df = pd.DataFrame(X_test_p[:, selected_mask],
                              columns=selected_feature_names)

# --- Tuned RSF on selected features ---
rsf_search = GridSearchCV(
    RandomSurvivalForest(n_jobs=-1, random_state=42),
    param_grid={"n_estimators": [300, 500],
                "min_samples_split": [5, 10, 20],
                "min_samples_leaf": [3, 5, 10],
                "max_features": ["sqrt", "log2", 0.4]},
    cv=3, n_jobs=-1, refit=True)
rsf_search.fit(X_train_sel_df, y_train)
rsf = rsf_search.best_estimator_
print(f"RSF test C-index: {rsf.score(X_test_sel_df, y_test):.4f}")

# --- SHAP on reduced feature space (anonymized for publication) ---
explainer = shap.Explainer(rsf.predict,
                           shap.sample(X_train_sel_df, min(100,
len(X_train_sel_df))),
                           random_state=42))
shap_values = explainer(X_test_sel_df)
mean_abs = np.abs(shap_values.values).mean(axis=0)
display_names, rank_by_index = build_anonymized_display_names(
    list(selected_feature_names), mean_abs)
shap.summary_plot(shap_values, X_test_sel_df, feature_names=display_names,
show=False)
plt.savefig("shap_summary_pfs_coxlasso_rsf.png", dpi=300)

```