ADIA Lab Structural Break Open Benchmark Challenge

Furui Wang

Shanghai Business School

Feishu Institute

Abstract

This report investigates the *Structural Break Open Benchmark*, a benchmark proposed by ADIA Lab and CrunchDAO that aims to determine whether a given boundary point in a univariate time series corresponds to a true structural break, rather than localizing the breakpoint itself. Under this setting, raw time series are often affected by noise and natural fluctuations, making structural changes difficult to identify through visual inspection or individual statistical measures. To address this challenge, we adopt a deterministic and model-agnostic feature engineering strategy that summarizes each time series from multiple perspectives, including global distributional properties, temporal shape characteristics, and contrasts between pre- and post-boundary segments. Based on the resulting fixed-dimensional feature representation, multiple standard classification models are evaluated under a unified experimental protocol, with ROC & AUC used as the primary performance metric. The experimental results indicate that individual statistical tests exhibit limited discriminative power when used in isolation, whereas feature-based non-linear models are more effective at capturing structural change patterns. Overall, the proposed framework provides a clear, reproducible, and stable baseline for this benchmark.

Keywords: Structural break detection; Time series analysis; Model selection

Contents

1 Introduction

Structural breaks refer to abrupt changes in the underlying statistical properties of a time series, such as its mean, variance, or temporal dependence structure. These phenomena are pervasive in economic and financial data and, if left unaccounted for, can severely compromise parameter estimation, inference, and forecasting performance [Perron et al. (2006), Aue and Horváth (2013)]. As a result, the detection and analysis of structural breaks have long been a central topic in time series analysis.

The classical literature on structural break detection has primarily focused on parametric and regression-based frameworks. Early contributions, such as R. L. Brown, Durbin, and Evans (1975), proposed stability tests based on recursive residuals, including the CUSUM family of tests. Subsequent work by Bai and Perron (1998) extended these ideas to allow for multiple structural changes and established consistent estimation and hypothesis testing procedures for break locations. These methodologies have been widely applied in empirical finance, where structural breaks are closely linked to regime shifts, policy interventions, and market crises [Andreou and Ghysels (2009), Casini and Perron (2018)].

More recently, increasing data complexity and application-driven requirements have motivated a shift toward feature-based and model-agnostic approaches to structural break detection. The ADIA Lab Structural Break Open Benchmark Challenge provides a standardized data generation and evaluation framework, enabling systematic and reproducible comparison across competing methods. The publicly available GitHub baseline created by aParsecFromFuture (2025) adopts large-scale feature engineering and model stacking to effectively capture statistical discrepancies between pre- and post-boundary segments. Building on the core ideas of this baseline, we redesign the pipeline from the perspective of structural clarity and implementation simplicity. Specifically, we organize the feature construction process into several interpretable modules that describe global distributional properties, temporal shape information, and explicit pre/post contrasts, and employ a two-stage prediction fusion strategy in which base model outputs generated on a compact feature subset are integrated with change-sensitive statistical features by a meta-learner. This design retains the essential modeling philosophy of the baseline while reducing overall complexity, improving interpretability, and maintaining competitive detection performance under the benchmark setting.

2 Dataset Description

The dataset used in this competition consists of a large collection of *synthetic univariate time series* specifically designed for the task of structural break detection. Each time series is associated with a predefined boundary point, at which a potential structural break may occur.

In the training set, each time series is assigned a binary label indicating whether a structural break occurs at the boundary point, where 1 denotes the presence of a break and 0 indicates no break. The data are provided in a long-format representation: each row corresponds to a single time observation, identified by a unique series ID and its corresponding time index and value. Series-level labels are stored separately.

The training input comprises approximately 2.37×10^7 observations, corresponding to 10,001 distinct time series, while the test set contains around 2.34×10^5 observations from 101 time series. Each observation includes two core fields: a time index and a univariate signal value.

The synthetic time series are constructed to reflect a wide range of real-world data-generating processes, encompassing varying levels of noise, break magnitudes, and structural patterns. These scenarios are motivated by applications commonly encountered in financial markets, climate analysis, industrial sensor monitoring, and biomedical signal processing. As a result, the competition emphasizes not only accurate detection of structural breaks, but also the ability of proposed methods to generalize across heterogeneous and previously unseen time series behaviors.

3 Exploratory Data Analysis

Before introducing statistical features and predictive models, we first examine the task from a qualitative and visual perspective. In the ADIA Lab Structural Break Open Benchmark, each univariate time series is associated with a known boundary point that partitions the sequence into a pre-boundary and a post-boundary segment. The objective is not to localize the breakpoint, but to determine whether this provided boundary corresponds to a genuine change in the underlying data-generating process.

Visual inspection of raw time series can be inconclusive. In many cases, the mean level appears stable across the boundary, while the differences are reflected in more subtle properties such as volatility, distributional shape, or higher-order temporal structure. Under substantial noise or gradual transitions, the true structural change may be difficult to distinguish from ordinary fluctuations by looking at the original values alone.

To illustrate this challenge, Figure 1 shows an example series whose label in the dataset is positive (`True`), meaning a structural break is known to occur at the provided boundary. The red dashed line indicates this ground-truth boundary location. While the breakpoint is not immediately obvious in the original series, several simple transformations (normalization, cumulative aggregation, rank-based mapping, absolute value, and rolling statistics) enhance the contrast between the pre- and post-boundary segments. These transformed representations suppress high-frequency noise and amplify changes in variance and distributional characteristics, making the break more visually apparent.

This observation motivates systematic feature engineering: instead of relying on raw trajectories, we summarize each series using transformation-informed statistics and breakpoint-based contrasts, which provides a stable and reproducible input for subsequent statistical testing, interpretability analysis, and model comparison.

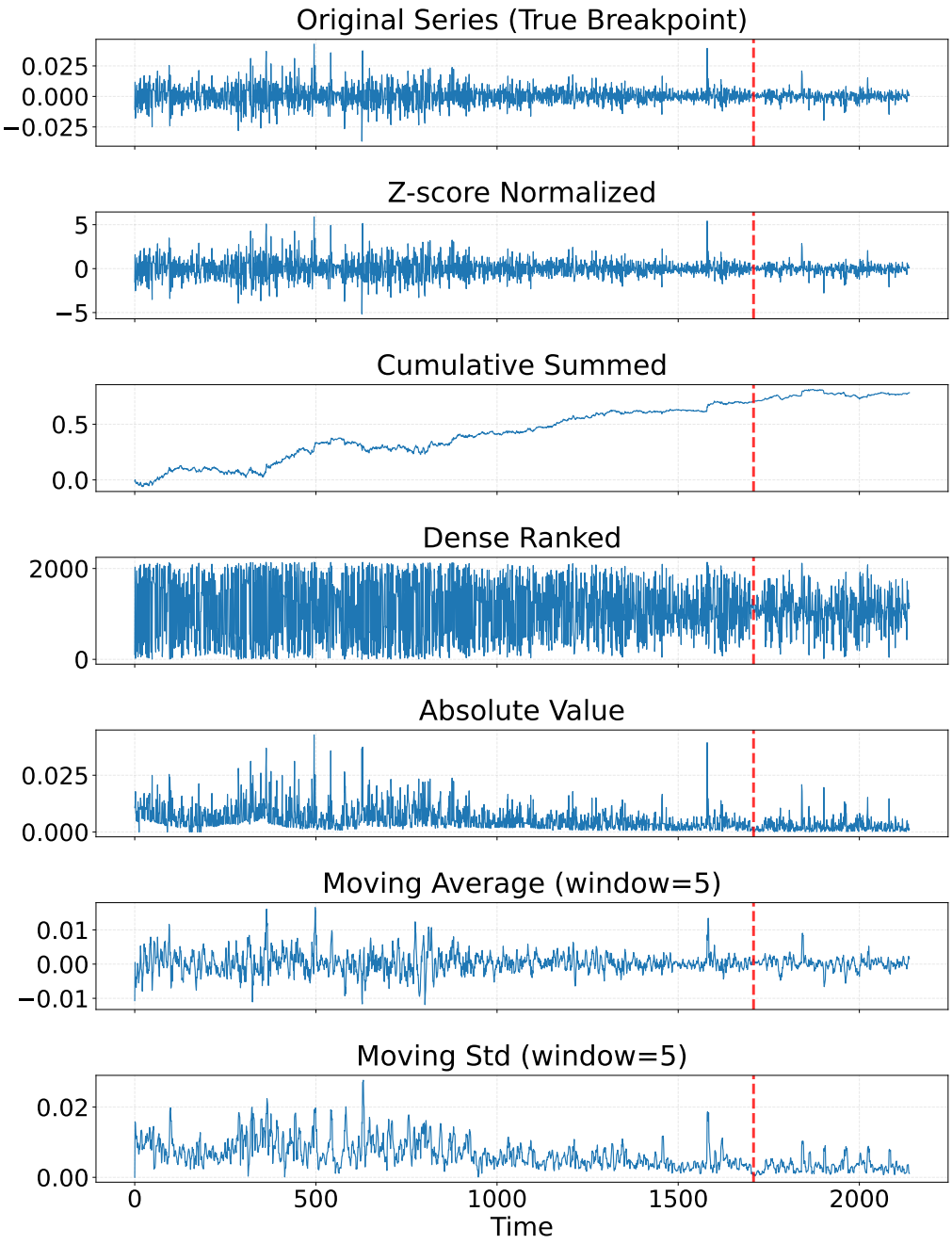


Figure 1: Original time series and transformed representations for a sample with a ground-truth positive structural break (`True`). The red dashed line marks the known boundary provided by the dataset, which corresponds to a true breakpoint for this example. Compared with the raw series, transformed views reveal clearer contrasts between the pre- and post-boundary segments, making the break more visually distinguishable.

4 Feature Engineering

In structural break detection tasks, raw time series often exhibit substantial noise and heterogeneous change patterns, making direct inference from the original observations unreliable. To address this issue, we adopt a deterministic and model-agnostic feature engineering strategy that maps each univariate time series into a fixed-dimensional feature representation. This representation serves as a unified input for subsequent interpretability analysis and model-based evaluation.

All features are constructed independently at the time-series level and do not rely on any

supervised learning model or training process. This design ensures that the feature construction stage itself does not introduce model-specific bias. Specifically, for each time series (identified by a unique *id*), four complementary feature generators are designed to characterize potential structural changes from different perspectives: global distributional properties, temporal shape dynamics, pre-/post-boundary contrasts, and classical statistical hypothesis tests.

4.1 Global Statistical Features

The first feature generator captures global distributional characteristics of each time series without distinguishing between pre- and post-boundary segments. These features provide background information about the overall scale and shape of the series and serve as coarse but robust descriptors.

Specifically, a set of descriptive statistics is computed, including the mean, median, maximum, minimum, standard deviation, and skewness. In addition, normalized statistics are constructed by scaling location measures (e.g., mean and median) with respect to the standard deviation, which helps mitigate scale differences across time series.

4.2 Temporal Shape Features

The second feature generator focuses on temporal dynamics and shape characteristics of the time series. To enhance sensitivity to local patterns and distributional structure, several deterministic transformations are applied to the original sequence.

These transformations include z-score normalization, absolute value transformation, dense ranking, rolling mean, and rolling standard deviation. For each transformed sequence, aggregated statistics such as the mean, median, extrema, standard deviation, and skewness are computed. This procedure compresses temporal information into a fixed-dimensional representation while preserving key aspects of dynamic behavior.

4.3 Pre/Post-Boundary Contrast Features

The third feature generator explicitly exploits the known boundary point provided in the dataset. Each time series is divided into a pre-boundary segment and a post-boundary segment, and structural changes are quantified by directly comparing statistical properties between the two segments.

Computed features include differences, absolute differences, and ratios of location statistics (mean and median), scale measures (standard deviation, interquartile range, and median absolute deviation), as well as quantile-based spread measures. These features are designed to capture changes in the underlying data-generating process across the boundary.

4.4 Statistical Test-based Features

The fourth feature generator is constructed upon classical statistical hypothesis testing methods that are designed to quantify distributional differences between the pre- and post-boundary segments of a time series. Unlike conventional approaches that rely solely on binary hypothesis test outcomes, this generator incorporates both the test statistics and their associated p-values as con-

tinuous features, thereby preserving richer information regarding the magnitude and significance of distributional changes.

Specifically, we employ the variance ratio F-test to assess changes in scale Fisher (1970), a robust variant of Levene's test for equality of variances M. B. Brown and Forsythe (1974), the two-sample Kolmogorov–Smirnov test to capture general distributional discrepancies Smirnov (1939), and Welch's t-test to evaluate differences in mean under unequal variances Welch (1947). Taken together, these complementary tests enable a comprehensive characterization of location shifts, scale changes, and overall distributional alterations induced by structural breaks.

4.5 Summary of Feature Generators

Table 1 summarizes the four feature generators and the types of information they capture.

Table 1: Summary of feature generators and extracted feature types.

Feature Generator	Perspective	Input Series	Main Feature Types	Boundary
Global Statistics	Overall distribution	Original value series	Mean, median, extrema; standard deviation; skewness; normalized statistics	No
Temporal Shape	Dynamic characteristics	Original and transformed series	Aggregated statistics; rolling mean and variance; rank-based and normalized features	No
Pre/Post Contrast	Structural magnitude	Pre- and post-boundary segments	Differences, absolute differences, and ratios of location and scale measures	Yes
Statistical Tests	Distributional significance	Pre- and post-boundary segments	Test statistics and p-values (F-test, Levene test, KS test, Welch t-test)	Yes

In summary, the proposed feature engineering framework systematically encodes global distributional properties, temporal dynamics, structural contrasts, and statistical significance signals into a unified feature representation. This representation provides a stable and reproducible foundation for subsequent interpretability analysis and model-based evaluation.

5 Methodology

This section presents the methodological framework adopted in this study. After constructing a unified and model-agnostic feature representation for each time series, multiple detection and classification models are applied and compared. The considered approaches range from classical ensemble methods to modern gradient boosting techniques and a pre-trained tabular inference model. Unless otherwise stated, all models operate on the same engineered feature space.

5.1 Evaluation of Individual Statistical Tests

To establish a baseline for structural breakpoint detection, we first evaluate the discriminative power of individual statistical tests without introducing any machine learning models. Each method

is assessed independently to examine whether a single statistical measure can serve as an effective indicator of structural changes in time series data.

For each time series, the observations are divided into two segments according to the known breakpoint position, namely the pre-break segment (`period = 0`) and the post-break segment (`period = 1`). Based on these two segments, a set of commonly used statistical tests and distributional difference measures are computed, and their abilities to distinguish breakpoint and non-breakpoint series are evaluated using ROC AUC value.

5.1.1 Description of Statistical Tests

The following eight statistical methods are considered in this study:

1. **Welch's t-test**: examines whether the means of the pre-break and post-break segments differ significantly.
2. **Mann–Whitney U test**: non-parametric test for distribution differences [Mann and Whitney (1947)].
3. **Kolmogorov–Smirnov (KS) test**: maximum distance between empirical CDFs.
4. **Mean Difference**: absolute mean difference between segments.
5. **Variance Difference**: absolute variance difference.
6. **Standard Deviation Difference**: absolute std difference.
7. **Quantile Difference**: absolute median difference.
8. **Interquartile Range (IQR) Difference**: absolute IQR difference.

5.1.2 Discriminative Performance Analysis

Figure 2 presents the ROC curves of the eight statistical methods evaluated on the training set. The corresponding AUC values are reported within each subplot.

Overall, the discriminative performance of individual statistical tests is limited, with AUC values generally ranging between 0.49 and 0.60. Among all methods, the KS test achieves the highest AUC (approximately 0.60), indicating a relatively stronger ability to capture distributional changes associated with structural breakpoints. These results motivate the integration of multiple statistical features and learning-based approaches.

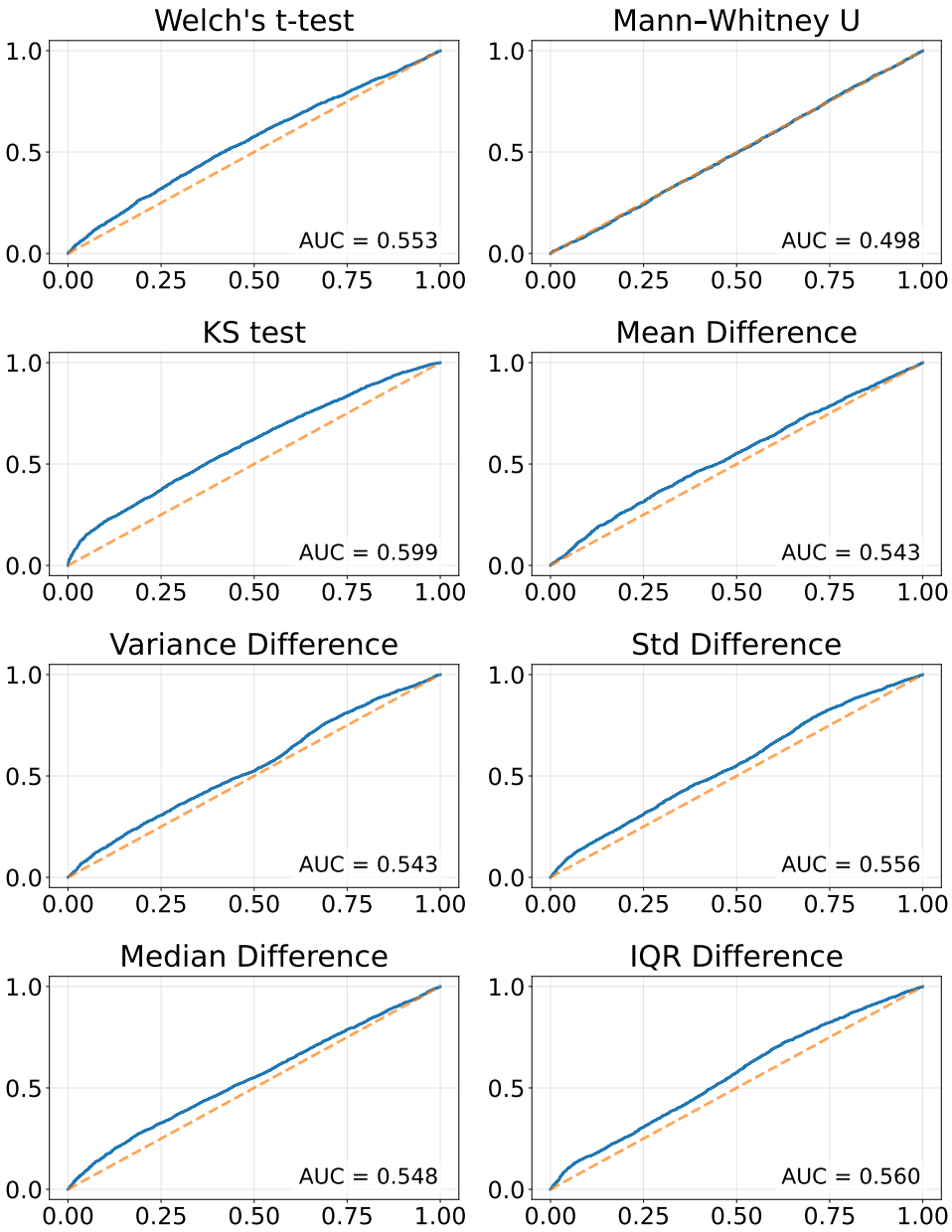


Figure 2: ROC curves of individual statistical tests for structural breakpoint detection. Each subplot corresponds to a single statistical method, with the AUC value reported inside the panel.

5.2 Tree-Based and Pre-trained Models for Comparison

To comprehensively evaluate the effectiveness of the proposed feature representation, we compare a diverse set of tree-based ensemble models and a pre-trained tabular inference model under a unified experimental protocol. These models differ substantially in their learning paradigms, inductive biases, and optimization strategies, providing a broad perspective on structural break detection performance.

Logistic Regression. Logistic Regression (LR) is a widely used linear classification model that estimates the conditional probability of a binary outcome given the input features. Unlike tree-based ensemble methods, Logistic Regression assumes a linear relationship between the input features and the log-odds of the target variable [Perron et al. (2006)].

The prediction probability of LR is defined as

$$p(y = 1 \mid x) = \sigma\big(w^\top x + b\big)\,, \tag{1}$$

where $w \in \mathbb{R}^d$ denotes the weight vector, b is the bias term, and $\sigma(\cdot)$ is the logistic sigmoid function given by

$$\sigma(z) = \frac{1}{1 + \exp(-z)}. \tag{2}$$

The model parameters are estimated by minimizing the negative log-likelihood, which is equivalent to the binary cross-entropy loss:

$$\mathcal{L}(w, b) = -\sum_{i=1}^{n} \left[y_i \log p(y_i = 1 \mid x_i) + (1 - y_i) \log \big(1 - p(y_i = 1 \mid x_i)\big) \right], \tag{3}$$

where n is the number of training samples.

In this study, Logistic Regression is implemented using the `LogisticRegression` class from the scikit-learn library and is included as a linear baseline model for comparison.

Support Vector Machine with RBF Kernel. Support Vector Machine (SVM) is a margin-based classification method that aims to find an optimal separating hyperplane by maximizing the margin between different classes in a high-dimensional feature space. To handle nonlinear decision boundaries, SVM can be extended through kernel functions [Cortes and Vapnik (1995)].

In this study, we adopt the Radial Basis Function (RBF) kernel, which implicitly maps the input features into an infinite-dimensional Hilbert space. The RBF kernel is defined as

$$K(x_i, x_j) = \exp\big(-\gamma \|x_i - x_j\|^2\big)\,, \tag{4}$$

where $\gamma > 0$ controls the width of the kernel.

For binary classification, SVM solves the following optimization problem:

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{n} \xi_i, \tag{5}$$

subject to

$$y_i \left(w^\top \phi(x_i) + b\right) \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n, \tag{6}$$

where w is the weight vector (normal vector) of the separating hyperplane, b is the bias term of the hyperplane, $\phi(\cdot)$ denotes the implicit feature mapping induced by the kernel, ξ_i are slack variables, $C > 0$ is a regularization parameter that balances margin maximization and classification error, and n is the number of training samples.

In this study, the SVM with RBF kernel is implemented using the `SVC` class from the scikit-learn library and is included as a nonlinear baseline model for comparison.

Random Forest. Random Forest is an ensemble learning method based on the Bagging (Bootstrap Aggregating) principle. Multiple decision trees are trained independently using bootstrap

samples drawn from the original training data. During node splitting, each tree considers only a randomly selected subset of features, which reduces correlation among trees and improves generalization[Breiman (2001)].

Let $f_t(\cdot)$ denote the prediction of the t-th decision tree and T the total number of trees. The Random Forest prediction is given by:

$$\hat{y}(x) = \frac{1}{T}\sum_{t=1}^{T} f_t(x). \tag{7}$$

RF typically employs the Gini impurity as the splitting criterion. For a node containing K classes, the Gini impurity is defined as:

$$\text{Gini} = 1 - \sum_{k=1}^{K} p_k^2, \tag{8}$$

where p_k denotes the proportion of samples belonging to class k.

The impurity reduction of a candidate split is:

$$\Delta\text{Gini} = \text{Gini}_{\text{parent}} - \frac{n_L}{n}\text{Gini}_L - \frac{n_R}{n}\text{Gini}_R. \tag{9}$$

Here, $\text{Gini}_{\text{parent}}$ is the Gini impurity of the parent node (the original node before splitting), n_L and n_R are the number of samples in the left and right child nodes after splitting, $n = n_L + n_R$ is the total number of samples in the parent node, and Gini_L and Gini_R are the Gini impurities of the left and right child nodes, respectively.

Gradient Boosting Decision Trees. Gradient Boosting Decision Trees (GBDT) construct an additive model by sequentially fitting decision trees to the residuals of previous iterations. Unlike Random Forests, where individual trees are trained independently, GBDT trains trees in a serial manner, allowing later trees to focus on samples that are difficult to predict[Friedman (2001)].

The prediction function of GBDT is defined as

$$\hat{y}(x) = \sum_{t=1}^{T} f_t(x), \quad f_t \in \mathcal{F}, \tag{10}$$

where $\mathcal{F}$ denotes the space of regression trees and T is the total number of boosting iterations.

At iteration t, the base learner is obtained by minimizing the following objective:

$$f_t = \arg\min_{f\in\mathcal{F}} \sum_{i=1}^{n} L\left(y_i, \hat{y}_i^{(t-1)} + f(x_i)\right), \tag{11}$$

where $L(\cdot)$ denotes a differentiable loss function, $\hat{y}_i^{(t-1)}$ represents the prediction of the ensemble at iteration $t-1$, and n is the number of training samples.

In this study, the GBDT model is implemented using the `GradientBoostingClassifier` provided by the scikit-learn library and serves as a classical boosting baseline for performance comparison.

XGBoost. XGBoost is a regularized and computationally efficient implementation of GBDT that incorporates second-order gradient information and explicit regularization to control model complexity[Chen (2016)].

Its objective function is defined as:

$$\mathcal{L} = \sum_{i=1}^{n} L\left(y_i, \hat{y}_i\right) + \sum_{t=1}^{T} \Omega(f_t), \tag{12}$$

where the regularization term is:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \|w\|^2. \tag{13}$$

Using a second-order Taylor approximation, the loss at iteration t becomes:

$$\mathcal{L}^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2 \right] + \Omega(f_t), \tag{14}$$

where g_i and h_i denote the first- and second-order derivatives of the loss with respect to the prediction.

The split gain is computed as:

$$\text{Gain} = \frac{G_L^2}{H_L + \lambda} + \frac{G_R^2}{H_R + \lambda} - \frac{(G_L + G_R)^2}{H_L + H_R + \lambda} - \gamma. \tag{15}$$

LightGBM. LightGBM adopts the same objective formulation as XGBoost but introduces several engineering optimizations. It employs histogram-based feature discretization to reduce computational complexity and uses a leaf-wise tree growth strategy, where the leaf with the largest split gain is expanded first. Model complexity is controlled via parameters such as `num_leaves` and `max_depth`, enabling efficient learning on high-dimensional features[Ke et al. (2017)].

CatBoost. CatBoost is a gradient boosting framework designed to address prediction shift and categorical feature bias[Prokhorenkova et al. (2018)]. It introduces an ordered boosting mechanism, in which the prediction for the i-th sample is computed using only samples preceding it in a random permutation:

$$\hat{y}_i = f\left(x_i \mid \{(x_j, y_j)\}_{j<i}\right). \tag{16}$$

For categorical features, CatBoost employs target statistics:

$$\text{TS}_i(c) = \frac{\sum_{j<i} \mathbf{1}(x_j = c)\, y_j + a \cdot \text{prior}}{\sum_{j<i} \mathbf{1}(x_j = c) + a}. \tag{17}$$

CatBoost further uses symmetric (oblivious) decision trees, leading to regular tree structures and efficient inference.

TabPFN. Tabular Prior-Data Fitted Network (TabPFN) is a pre-trained model for tabular data that performs direct conditional inference instead of task-specific parameter optimization.

Given a training dataset

$$\mathcal{D} = (X_{\text{train}}, y_{\text{train}}), \tag{18}$$

TabPFN models the conditional probability:

$$p(y_{\text{test}} \mid x_{\text{test}}, X_{\text{train}}, y_{\text{train}}). \tag{19}$$

From a Bayesian perspective, the predictive distribution can be expressed as:

$$p(y \mid x, \mathcal{D}) = \int p(y \mid x, \theta)\, p(\theta \mid \mathcal{D})\, d\theta, \tag{20}$$

which is approximated using a Transformer-based architecture:

$$\hat{y} = f_\phi(X_{\text{train}}, y_{\text{train}}, x_{\text{test}}), \tag{21}$$

where ϕ denotes fixed pre-trained parameters.

Summary. By integrating bagging-based ensembles, boosting-based methods, and a pre-trained tabular inference model, this study provides a comprehensive comparison across fundamentally different modeling paradigms. All models are evaluated using the same engineered feature set and experimental protocol, ensuring a fair and consistent assessment of structural break detection performance.

5.3 Model Selection

Based on the engineered feature representation described in the previous sections, we conduct a systematic comparison of multiple classification models to evaluate their effectiveness in structural break detection. The evaluated models include linear classifiers, kernel-based methods, tree-based ensemble models, and a pre-trained tabular inference model. All models are trained and assessed under a unified experimental protocol to ensure fair and reproducible comparison.

We first report the performance on the training set using *strict out-of-fold* (OOF) predictions obtained via stratified K-fold cross-validation. In this setting, each sample is predicted by a model that has not been trained on it, thereby eliminating information leakage and providing an unbiased estimate of generalization performance.

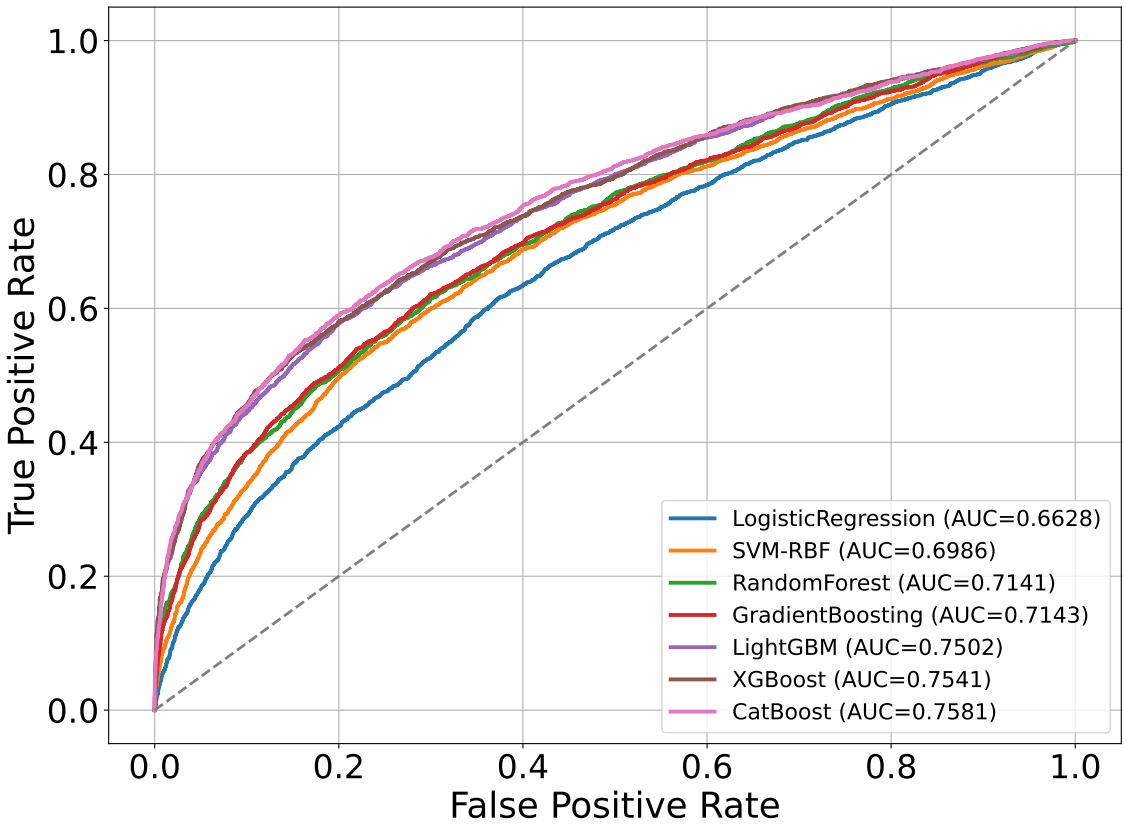


Figure 3: Strict OOF ROC curves of different models evaluated under stratified K-fold cross-validation on the engineered feature set.

As shown in Figure 5, linear models such as Logistic Regression exhibit limited discriminative capability, with ROC curves remaining close to the random baseline. This indicates that linear decision boundaries are insufficient to capture the complex structural characteristics encoded in the engineered features. In contrast, non-linear models demonstrate substantially improved performance. Kernel-based SVMs and tree-based ensemble methods benefit from their ability to model feature interactions, leading to higher AUC values.

Among the compared traditional machine learning models, GBDT methods consistently outperform other approaches. In particular, LightGBM, XGBoost, and CatBoost achieve the highest OOF AUC values, surpassing Logistic Regression, SVM-RBF, Random Forest, and classical Gradient Boosting. These results suggest that boosting-based ensembles are particularly effective in exploiting the high-dimensional statistical and temporal features produced by the proposed feature engineering pipeline. Based on this observation, LightGBM, XGBoost, and CatBoost are selected as the strongest traditional models for subsequent comparison and integration.

To further assess real-world applicability, all traditional models are trained on the full training data and evaluated on an independent held-out test set. In this evaluation, models are directly applied to unseen test samples without additional model adaptation.

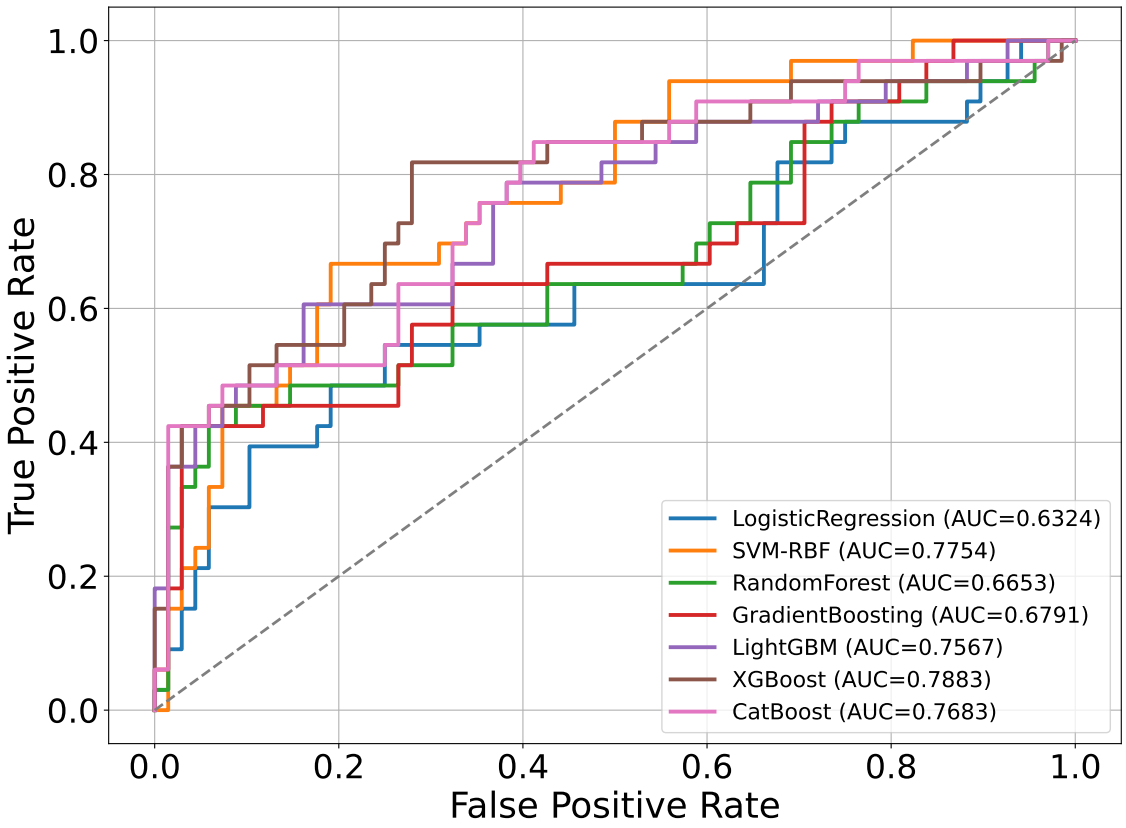


Figure 4: ROC curves evaluated on the independent test set. Traditional models are trained on the full training data and applied to held-out test samples.

Figure 6 shows that the relative performance ordering observed in the OOF evaluation largely persists on the test set. Linear and kernel-based models continue to underperform, while gradient boosting tree ensembles maintain stable and competitive results. LightGBM, XGBoost, and CatBoost demonstrate robust generalization performance across the entire range of false positive rates, confirming their suitability as strong conventional baselines.

Motivated by these results, we further incorporate TabPFN together with the selected GBDT models to construct a hybrid modeling framework. While the traditional boosting models provide flexible discrimination through data-driven tree ensembles, TabPFN contributes a strong prior-informed predictive signal at the sample level. This complementary combination forms the foundation for the subsequent integrated and stacked modeling strategy.

5.4 Integration of TabPFN with LightGBM, XGBoost, and CatBoost

In Section 5, we discussed how boosting models such as LightGBM, XGBoost, and CatBoost consistently demonstrated strong performance in capturing complex interactions within the engineered features. To further enhance the predictive capabilities of these models, we integrate them with TabPFN, a pre-trained model that provides additional prior-driven generalization capabilities.

TabPFN generates prior-informed probability estimates based on global statistical features, which are then concatenated with the structural change features. These combined features are fed into the three boosting models (LightGBM, XGBoost, and CatBoost), forming a stacked ensemble that benefits from both prior knowledge and data-driven feature interactions. This hybrid

approach aims to capitalize on the strengths of both models: TabPFN's generalization power and the boosting models' ability to capture non-linear feature interactions.

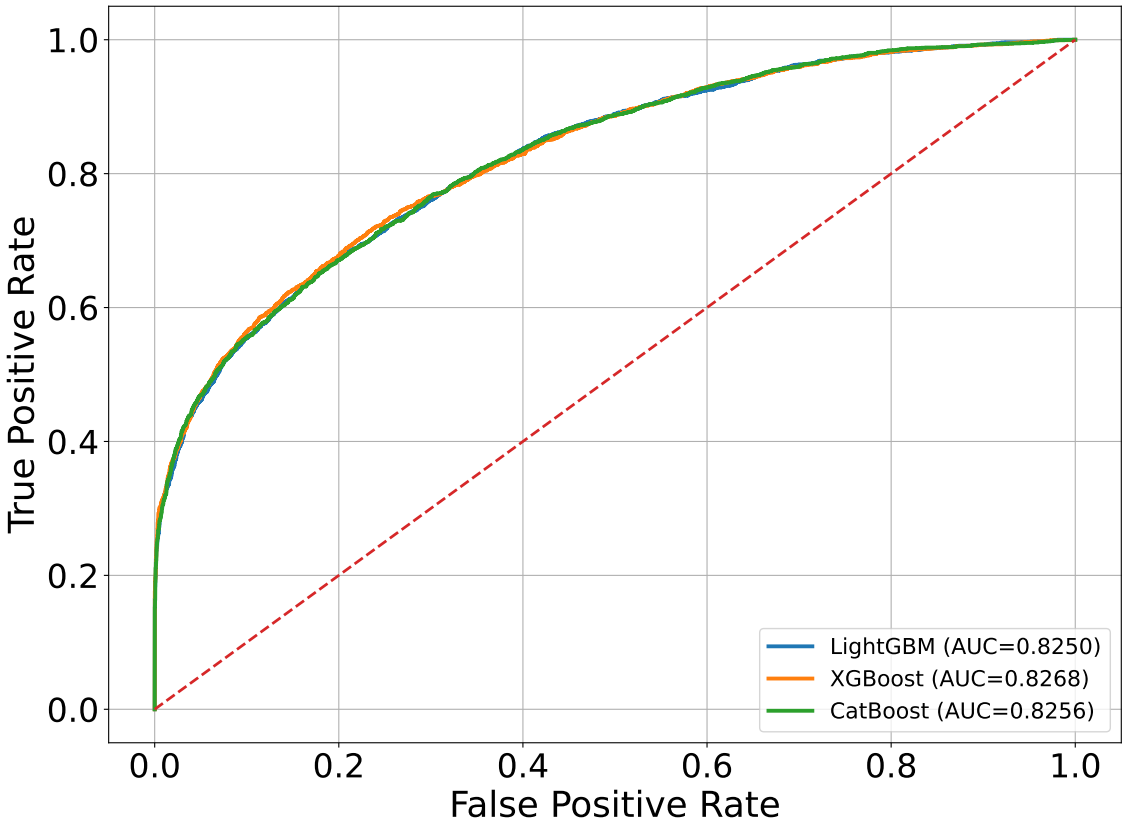


Figure 5: Strict OOF ROC curves for the three TabPFN-augmented boosting models. TabPFN generates OOF probability estimates based on global statistical features (x_1), which are concatenated with structural change features (x_s) and passed into LightGBM, XGBoost, and CatBoost. These models achieve comparable and strong performance (AUC $\approx$ 0.82–0.83), showing the effectiveness of combining prior knowledge with feature interactions.

The combination of TabPFN and the boosting models leads to a final model that leverages both the prior-driven generalization of TabPFN and the flexible discrimination capabilities of boosting ensembles. In Figure 6, we present the ROC curves for the final model evaluated on the held-out test set. This figure confirms that the integrated TabPFN + boosting model consistently outperforms the individual boosting models, showing superior generalization on unseen data.

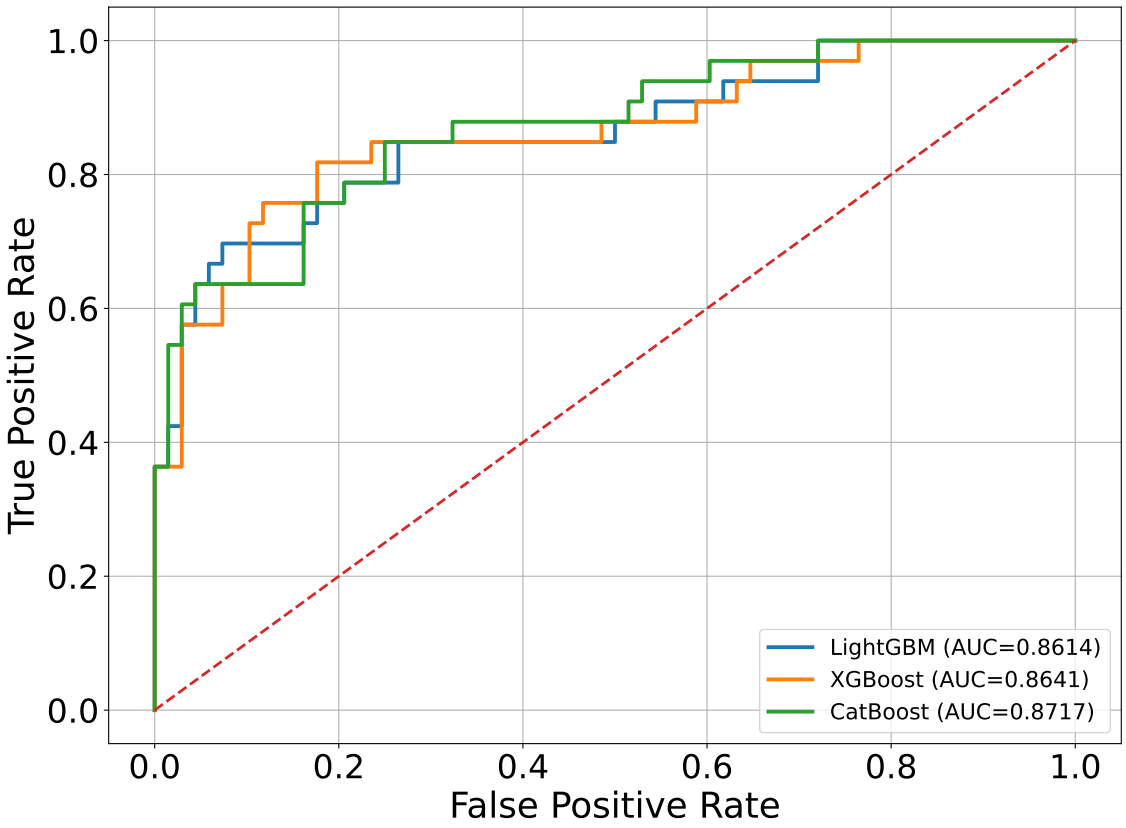


Figure 6: ROC curves evaluated on the independent test set. The final stacked model, combining TabPFN with CatBoost, achieves the highest AUC, demonstrating robust generalization performance. This model is selected as the final choice for structural break detection.

The integration of TabPFN with LightGBM, XGBoost, and CatBoost provides a robust foundation for the final stacked model. In the next section, we will explore the final model's architecture, training strategy, and interpretability in more detail.

6 Final Stacked Model: TabPFN and CatBoost

6.1 Model Architecture and Design Rationale

Building upon the feature engineering pipeline and the comparative analysis conducted in Section 5, we construct a final stacked model that integrates TabPFN with a selected gradient boosting model. The primary objective of this design is to combine the complementary strengths of prior-informed inference and data-driven nonlinear discrimination in a unified and leakage-free framework.

Figure 7 illustrates the overall architecture of the proposed stacked model. Starting from raw univariate time-series data, deterministic feature engineering is applied to obtain a tabular representation at the series level. The engineered features are then partitioned into two subsets: global statistical features and structural change features. The global statistical features summarize distributional properties of the entire series, while the structural change features capture temporal shape variations, contrast statistics, and hypothesis-test-based indicators associated with potential breakpoints.

TabPFN is employed as the base model and trained exclusively on the global statistical feature

subset. Under a strict out-of-fold (OOF) protocol, TabPFN produces probability estimates for each training sample, resulting in a leakage-free probability feature that reflects prior-informed predictive signals at the sample level. This OOF probability feature is then concatenated with the structural change features and fed into a CatBoost meta-model, which serves as the final classifier. By design, this two-stage architecture allows the meta-model to exploit both the calibrated prior signal provided by TabPFN and the rich nonlinear interactions captured by tree-based ensembles.

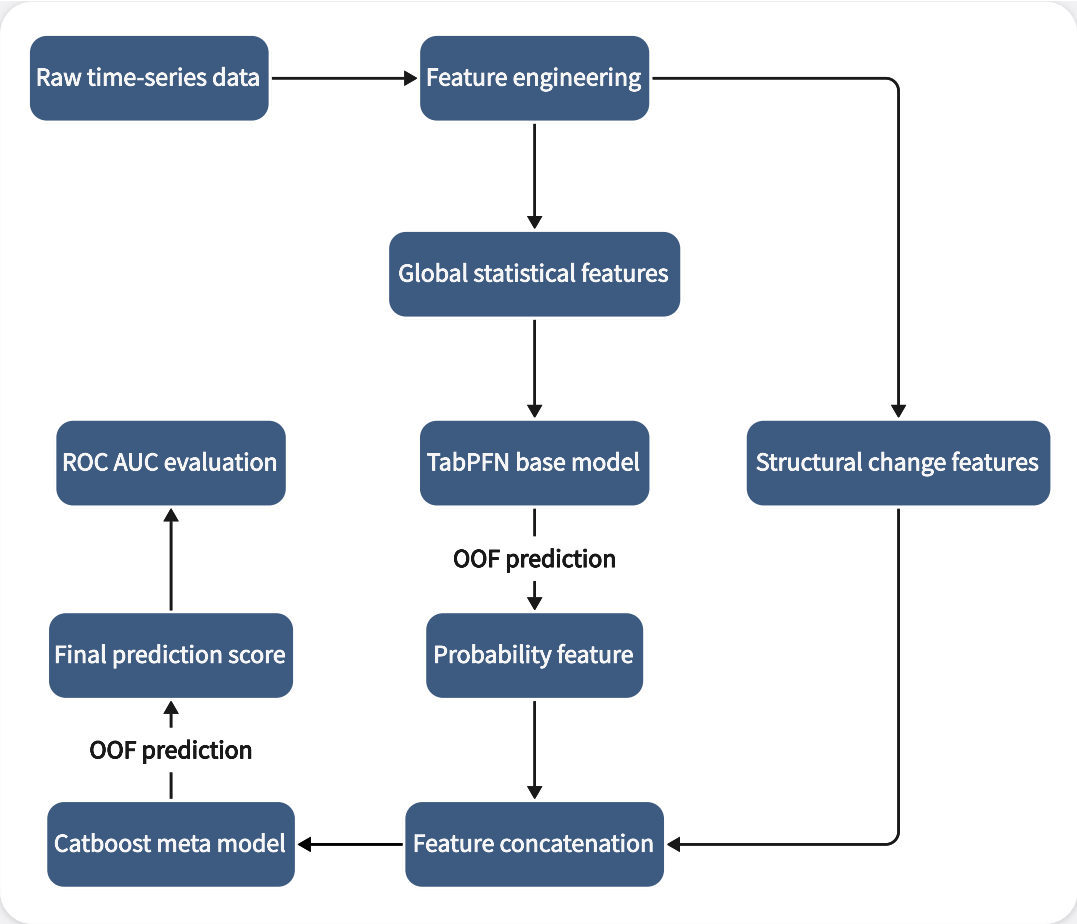


Figure 7: Overall architecture of the final stacked model. Raw time-series data are transformed into engineered features, which are split into global statistical features and structural change features. TabPFN serves as the base model to generate strict out-of-fold probability estimates, which are concatenated with structural change features and fed into a CatBoost meta-model to produce final predictions.

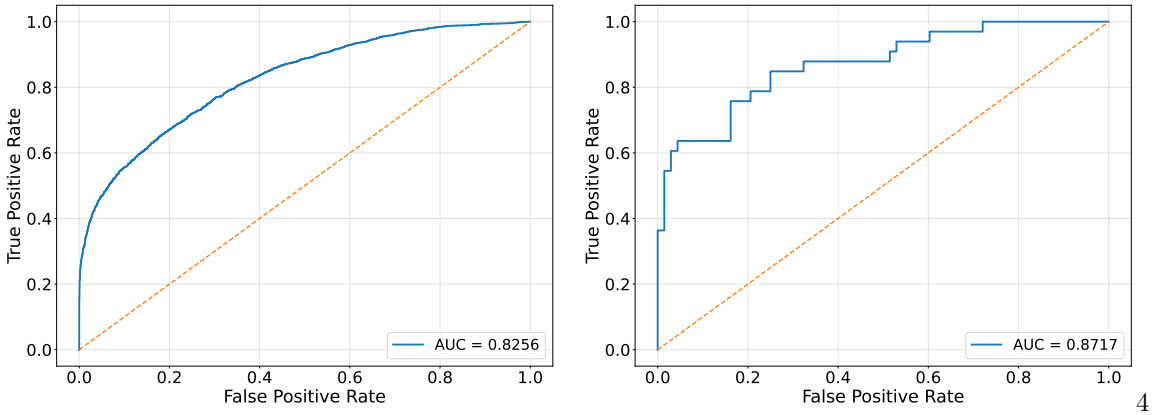
6.2 Training Strategy and Leakage-Free Evaluation

To ensure an unbiased performance assessment, a strict leakage-free training and evaluation strategy is adopted throughout the stacking procedure. In the first stage, TabPFN is trained using stratified K-fold cross-validation. For each fold, the model is fitted on the training subset and generates probability predictions for the held-out validation fold. Aggregating predictions across all folds yields a complete OOF probability vector for the training set, which is subsequently treated as a fixed feature.

In the second stage, the CatBoost meta-model is trained on the augmented feature space consisting of the structural change features and the TabPFN-derived OOF probability feature.

The same stratified cross-validation protocol is applied to obtain strict OOF predictions for the stacked model. This design guarantees that no information from the validation folds leaks into the training process at any stage.

Figure 8 reports the ROC curves of the final stacked model on both the training and test sets. On the training set, evaluated under the strict OOF protocol, the model achieves an AUC of approximately 0.8256, providing a reliable and unbiased estimate of in-sample generalization performance. On the independent test set, the stacked model attains an AUC of approximately 0.8718, indicating strong robustness and effective generalization to unseen time series.



(a) Training set ROC curve under strict OOF evaluation (AUC ≈ 0.8256).

(b) Test set ROC curve on the held-out data (AUC ≈ 0.8718).

Figure 8: ROC curves of the final stacked model (TabPFN base + CatBoost meta) evaluated on the training set using strict OOF predictions and on the independent test set.

6.2.1 SHAP-Based Global Importance and Directional Effects

To further validate and refine the global importance analysis, we employ SHapley Additive exPlanations (SHAP), which provide a theoretically grounded attribution of feature contributions. Figure 9 presents the mean absolute SHAP values for the most influential features. The resulting ranking is largely consistent with the CatBoost importance scores, confirming the stability of the identified key features.

Beyond global ranking, SHAP also reveals how individual feature values influence model outputs. Figure 10 illustrates the SHAP beeswarm plot, where each point corresponds to a single sample and color indicates feature magnitude. Clear directional patterns emerge: higher values of the TabPFN probability feature and extreme values of several structural change features consistently push predictions toward the presence of a structural break, whereas more stable feature values tend to favor the non-break class.

These consistent directional effects enhance the transparency of the stacked model and demonstrate that its predictions are driven by interpretable and meaningful signals rather than spurious correlations.

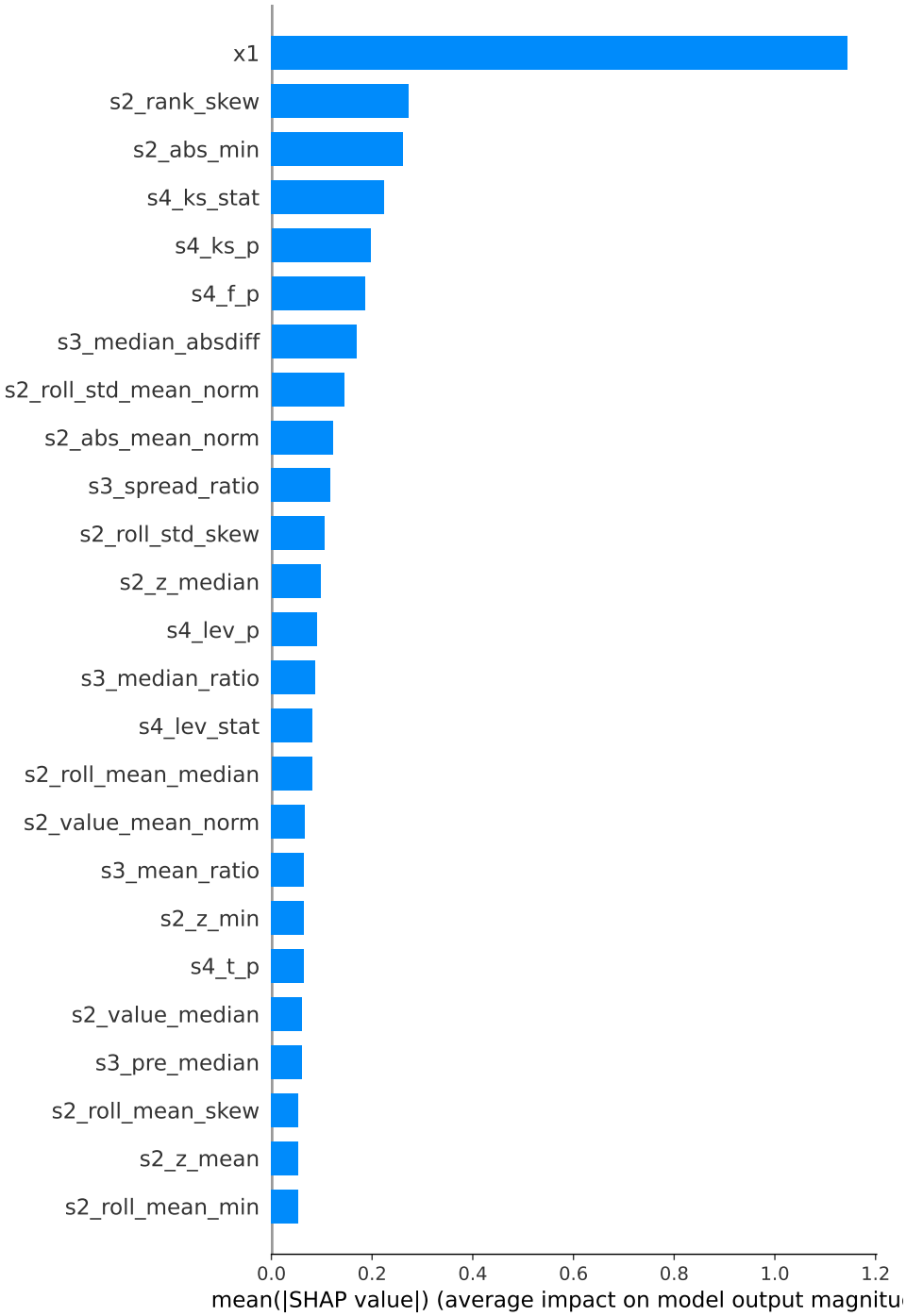


Figure 9: Global SHAP feature importance measured by mean absolute SHAP values. Larger values indicate greater overall contribution to model predictions.

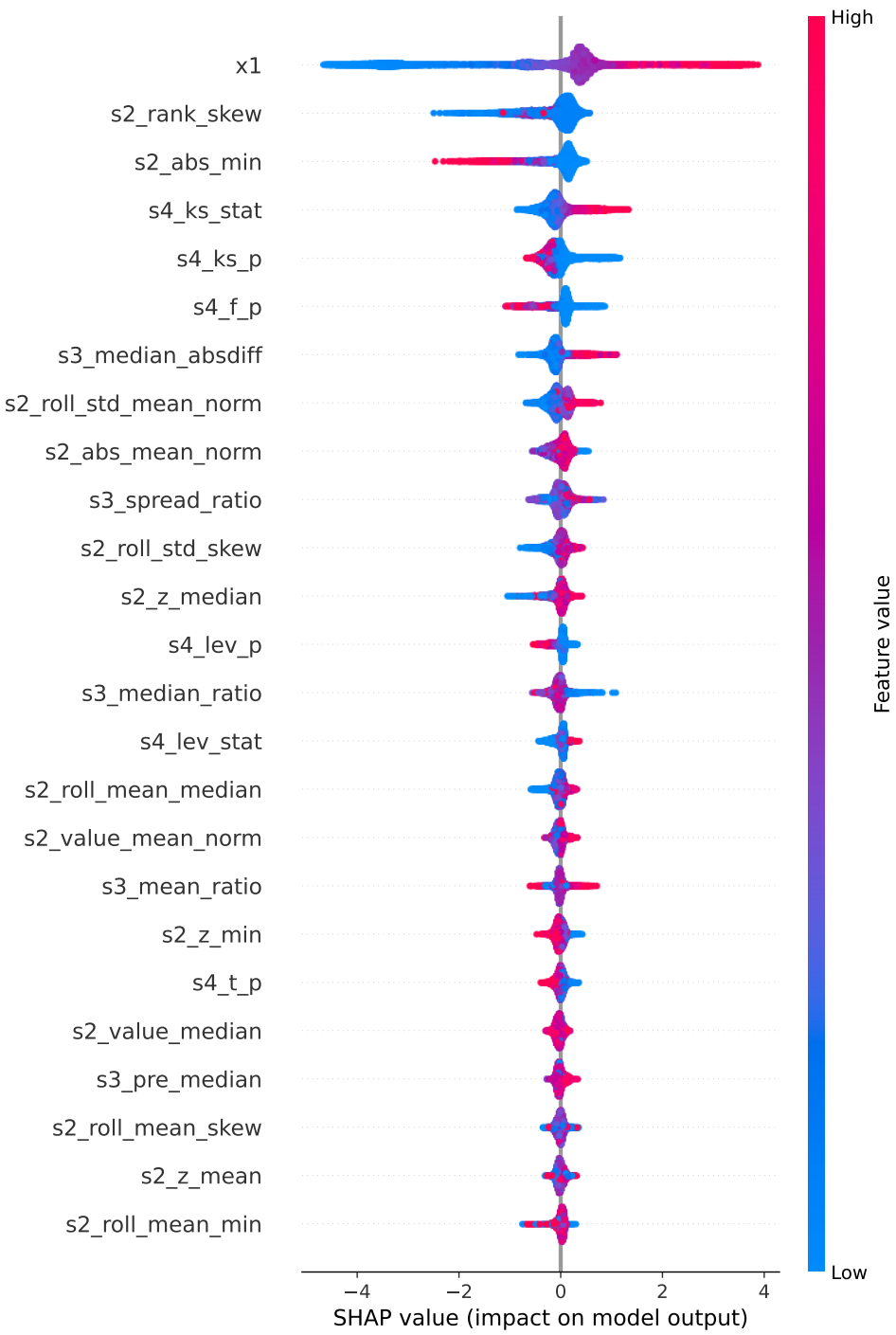


Figure 10: SHAP beeswarm plot illustrating feature-level contributions across samples. Color represents feature magnitude, and horizontal displacement reflects the direction and strength of impact on the predicted outcome.

6.2.2 Distributional Separation of High-Importance Features

After identifying the most influential features at the model level, we further examine whether these features exhibit meaningful distributional differences at the data level. Figure 11 visualizes the empirical distributions of the top eight SHAP-ranked features for samples with and without structural breaks. For each feature, histograms are shown for the two classes, together with vertical dashed lines indicating class-wise means.

The results indicate that many high-importance features display clear distributional separation

between break and non-break samples. Several temporal shape features exhibit heavier tails or increased dispersion for break samples, while statistical test-based features often concentrate in distinct value ranges. This observation suggests that the features emphasized by the model correspond to genuine structural differences in the underlying data, rather than artifacts introduced by the learning algorithm.

Overall, the consistency between feature importance rankings, SHAP-based directional effects, and empirical distributional separation provides strong evidence that the final stacked model captures meaningful and interpretable signals relevant to structural break detection.

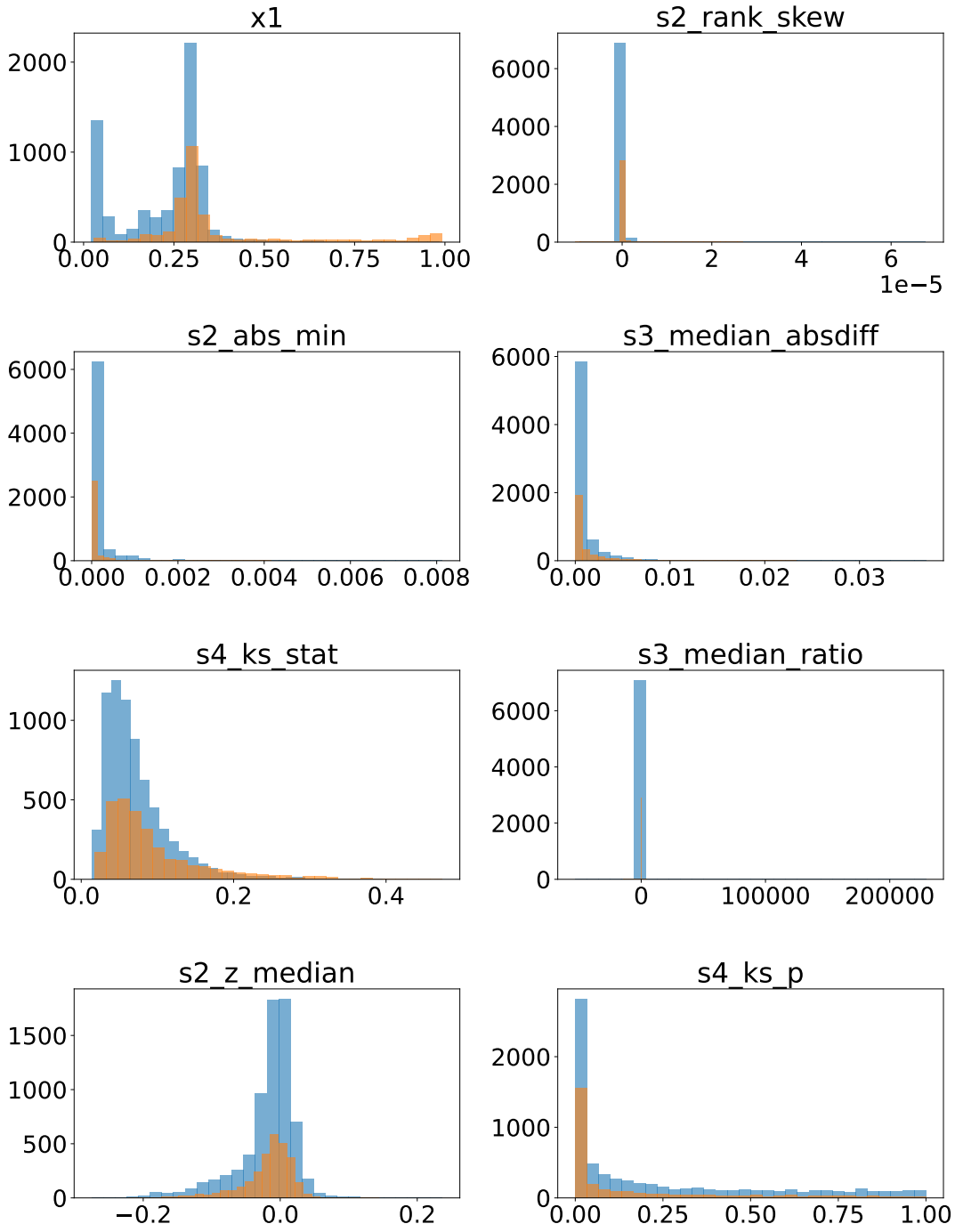


Figure 11: Distributions of the top eight SHAP-ranked features for samples with and without structural breaks. Dashed lines indicate class-wise means.

7 Conclusion

This study investigates structural break detection at a known boundary under the ADIA Lab Structural Break Open Benchmark. To address the challenges posed by noisy and heterogeneous time series, we propose a deterministic and model-agnostic feature engineering framework that systematically summarizes global distributional properties, temporal shape dynamics, and explicit pre/post-boundary contrasts. This unified feature representation provides a stable foundation for fair model comparison, robust evaluation, and interpretability analysis.

Through extensive experiments, we demonstrate that individual statistical tests exhibit limited discriminative power when used in isolation, motivating the adoption of learning-based approaches. A comprehensive comparison across linear models, kernel-based classifiers, tree-based ensembles, and a pre-trained tabular inference model shows that gradient boosting methods consistently outperform conventional baselines. In particular, LightGBM, XGBoost, and CatBoost emerge as the strongest traditional models under both strict out-of-fold evaluation and independent test set assessment.

Building on these findings, we introduce a two-stage stacked architecture that integrates TabPFN with CatBoost. By using TabPFN to generate leakage-free out-of-fold probability estimates as a meta feature and combining them with structurally informative features, the final stacked model effectively merges prior-informed generalization with data-driven nonlinear discrimination. The resulting model achieves strong and stable performance on both the training set (under strict OOF evaluation) and the held-out test set, indicating robust generalization and minimal overfitting.

Furthermore, interpretability analysis based on CatBoost feature importance and SHAP explanations confirms that the model relies on meaningful signals closely aligned with structural break mechanisms. The TabPFN probability feature and structural change features jointly contribute to the final decision, enhancing both transparency and reliability.

Several directions for future work remain. First, the current framework focuses on a single known boundary; extending it to unknown or multiple breakpoints would be a natural next step. Second, incorporating multivariate time series or domain-specific transformations may further improve detection performance. Finally, uncertainty-aware evaluation and probabilistic calibration analysis could provide deeper insight into model reliability in real-world applications.

Overall, this work highlights the importance of combining principled feature engineering, leakage-free evaluation, and complementary modeling paradigms for reliable and interpretable structural break detection.

References

[1] Elena Andreou and Eric Ghysels. "Structural breaks in financial time series". In: *Handbook of financial time series* (2009), pp. 839–870.

[2] aParsecFromFuture. *ADIA Lab Structural Break Challenge Solution*. https://github.com/aParsecFromFuture/ADIA-Lab-Structural-Break-Challenge-Solution. GitHub repository. 2025.

[3] Alexander Aue and Lajos Horváth. "Structural breaks in time series". In: *Journal of Time Series Analysis* 34.1 (2013), pp. 1–16.

[4] Jushan Bai and Pierre Perron. "Estimating and Testing Linear Models with Multiple Structural Changes". In: *Econometrica* 66.1 (1998), pp. 47–78.

[5] Leo Breiman. "Random forests". In: *Machine learning* 45.1 (2001), pp. 5–32.

[6] Morton B Brown and Alan B Forsythe. "Robust tests for the equality of variances". In: *Journal of the American statistical association* 69.346 (1974), pp. 364–367.

[7] Robert L Brown, James Durbin, and James M Evans. "Techniques for testing the constancy of regression relationships over time". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 37.2 (1975), pp. 149–163.

[8] Alessandro Casini and Pierre Perron. "Structural breaks in time series". In: *arXiv preprint arXiv:1805.03807* (2018).

[9] Tianqi Chen. "XGBoost: A Scalable Tree Boosting System". In: *Cornell University* (2016).

[10] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". In: *Machine learning* 20.3 (1995), pp. 273–297.

[11] Ronald Aylmer Fisher. "Statistical methods for research workers". In: *Breakthroughs in statistics: Methodology and distribution*. Springer, 1970, pp. 66–70.

[12] Jerome H Friedman. "Greedy function approximation: a gradient boosting machine". In: *Annals of statistics* (2001), pp. 1189–1232.

[13] Guolin Ke et al. "Lightgbm: A highly efficient gradient boosting decision tree". In: *Advances in neural information processing systems* 30 (2017).

[14] Henry B Mann and Donald R Whitney. "On a test of whether one of two random variables is stochastically larger than the other". In: *The annals of mathematical statistics* (1947), pp. 50–60.

[15] Pierre Perron et al. "Dealing with structural breaks". In: *Palgrave handbook of econometrics* 1.2 (2006), pp. 278–352.

[16] Liudmila Prokhorenkova et al. "CatBoost: unbiased boosting with categorical features". In: *Advances in neural information processing systems* 31 (2018).

[17] Nikolai V Smirnov. "On the estimation of the discrepancy between empirical curves of distribution for two independent samples". In: *Bull. Math. Univ. Moscou* 2.2 (1939), pp. 3–14.

[18] Bernard L Welch. “The generalization of ‘STUDENT’S’problem when several different population varlances are involved”. In: *Biometrika* 34.1-2 (1947), pp. 28–35.