Supplementary Information

Cross-validation Strategies for Medical Machine Learning:
A Snowball Sampling Review and Open-Source Implementation

Abdelamir Karbalaie, Farhad Abtahi

Contents

1 Supplementary Tables

1.1 Supplementary Table S1: Complete Parameter Reference

Table 1: Complete parameter reference for all 29 cross-validation methods implemented in trustcv.

Method	Parameter	Default	Description
I.I.D. Methods			
HoldOut	test_size	0.2	Proportion of data for test set
	stratify	None	Column for stratification
	random_state	None	Random seed for reproducibility
KFold	n_splits	5	Number of folds
	shuffle	False	Whether to shuffle before splitting
	random_state	None	Random seed
StratifiedKFold	n_splits	5	Number of folds
	shuffle	False	Whether to shuffle
	random_state	None	Random seed
RepeatedKFold	n_splits	5	Number of folds
	n_repeats	10	Number of repetitions
	random_state	None	Random seed
LOOCV	—	—	No parameters required
LPOCV	p	2	Number of samples to leave out
BootstrapValidation	n_iterations	200	Number of bootstrap iterations
	test_size	0.2	Out-of-bag proportion
	correction	'.632+'	Bias correction method
	random_state	None	Random seed
MonteCarloCV	n_splits	100	Number of random splits
	test_size	0.2	Test set proportion
	random_state	None	Random seed
NestedCV	outer_cv	5	Outer loop folds
	inner_cv	3	Inner loop folds
	random_state	None	Random seed
Grouped Methods			
GroupKFold	n_splits	5	Number of folds
StratifiedGroupKFold	n_splits	5	Number of folds
	shuffle	False	Whether to shuffle groups
	random_state	None	Random seed
LeaveOneGroupOut	—	—	No parameters required
LeavePGroupsOut	n_groups	2	Number of groups to leave out
RepeatedGroupKFold	n_splits	5	Number of folds
	n_repeats	10	Number of repetitions
	random_state	None	Random seed
HierarchicalGroupKFold	n_splits	5	Number of folds
	hierarchy_levels	None	List of hierarchy columns
GroupShaflKFold	n_splits	5	Number of folds
	n_repeats	10	Number of repetitions
NestedGroupedCV	outer_cv	5	Outer loop folds
	n_repeats	10	Number of repetitions
	random_state	None	Random seed
Temporal Methods			
TimeSeriesSplit	n_splits	5	Number of splits
	max_train_size	None	Maximum training set size
	test_size	None	Fixed test set size
	gap	0	Gap between train and test
BlockedTimeSeries	n_splits	5	Number of blocks

Continued on next page

Table 1 – continued from previous page

Method	Parameter	Default	Description
	gap	0	Gap between blocks
RollingWindowCV	window_size	None	Training window size (required)
	step_size	1	Step between windows
	test_size	1	Test window size
ExpandingWindowCV	min_train_size	None	Minimum training size (required)
	step_size	1	Step between windows
	test_size	1	Test window size
PurgedKFoldCV	n_splits	5	Number of folds
	purge_gap	0	Samples to purge near boundaries
CombinatorialPurgedCV	n_splits	5	Number of folds
	purge_gap	0	Purge period (samples)
	embargo_gap	0	Embargo period (samples)
PurgedGroupTimeSeriesSplit	n_splits	5	Number of splits
	purge_gap	0	Purge period
	embargo_gap	0	Embargo period
NestedTemporalCV	outer_cv	5	Outer loop splits
	inner_cv	3	Inner loop splits
	purge_gap	0	Purge period
Spatial Methods			
SpatialBlockCV	n_splits	5	Number of spatial blocks
	block_size	None	Block dimensions (auto if None)
	buffer	0	Buffer distance
BufferedSpatialCV	n_splits	5	Number of blocks
	buffer_radius	None	Buffer radius (required)
SpatiotemporalBlockCV	n_spatial	5	Spatial blocks
	n_temporal	5	Temporal blocks
	buffer	0	Spatial buffer
	gap	0	Temporal gap
EnvironmentalHealthCV	n_splits	5	Number of folds
	exposure_col	None	Exposure variable column
	buffer_radius	None	Buffer around exposure sources

1.2 Supplementary Table S2: Feature Comparison with Existing Tools

The following comparison shows which of the 29 methods identified by the snowball sampling review are implemented in each tool. trustcv implements all 29 methods identified; other tools implement subsets. Counts in the category rows reflect the review taxonomy; scikit-learn's total of 15 includes PredefinedSplit, a utility splitter counted separately.

Table 2: Feature comparison between trustcv and existing cross-validation tools. Method counts reflect the review taxonomy; see Methods for inclusion criteria.

Feature	**trustcv**	**scikit-learn**	**mlr3**	**tidymodels**	**MLme**
Cross-validation methods (review taxonomy)					
I.I.D. methods	9	8	6	5	3
Grouped methods	8	5	3	2	0
Temporal methods	8	1	2	2	0
Spatial methods	4	0	4	3	0
Total (review-identified)	**29**	**14**[a]	**15**	**12**	**3**
Safety features					
Patient leakage detection	✓	—	—	—	—
Temporal leakage detection	✓	—	—	—	—
Spatial leakage detection	✓	—	Visual only	Visual only	—
Preprocessing leakage detection	✓	—	—	—	—
Duplicate detection	✓	—	—	—	—
Algorithmic independence verification	✓	—	—	—	—
Framework support					
scikit-learn	✓	✓	—	—	✓
PyTorch	✓	—	—	—	—
TensorFlow/Keras	✓	—	—	—	—
MONAI	✓	—	—	—	—
JAX	✓	—	—	—	—
Regulatory documentation					
FDA GMLP templates	✓	—	—	—	—
CE MDR templates	✓	—	—	—	—
TRIPOD+AI mapping	✓	—	—	—	—
Other					
Primary language	Python	Python	R	R	Python
Clinical metrics (sensitivity, specificity, PPV, NPV)	✓	Limited	Limited	Limited	✓
Confidence intervals for metrics	✓	—	✓	✓	—
Educational tutorials	✓	Limited	✓	✓	Limited
Method selection guides	✓	—	—	—	—

[a]Excludes PredefinedSplit, a utility splitter for user-defined partitions not counted in the category-based rows above.

1.3 Supplementary Table S3: Regulatory Documentation Mapping

Table 3: Mapping of trustcv outputs to regulatory framework requirements. trustcv provides documentation templates and structured outputs that can support regulatory submissions; regulatory compliance depends on the complete device lifecycle.

Framework	**Requirement**	**trustcv Output**
FDA Good Machine Learning Practice (GMLP)		
Principle 4	Training data sets are independent of test sets	Leakage detection report; split verification
Principle 5	Reference datasets are based upon best available methods	Method selection documentation
Principle 8	Testing demonstrates device performance during clinically relevant conditions	Performance metrics with confidence intervals; stratified results
Principle 9	Users are provided clear, essential information	Validation summary report
CE Medical Device Regulation (MDR) 2017/745		
Annex II, Section 6.1	Clinical evaluation plan	Validation methodology documentation
Annex II, Section 6.2	Device description and specification	Software version; dependency list
Annex XIV, Part A	Clinical evaluation documentation	Performance metrics; confidence intervals
TRIPOD+AI Statement		
Item 6a	Eligibility criteria for participants	Data filtering documentation
Item 10a	How sample size was determined	Sample size per fold report
Item 10b	How data were split	Split methodology; fold assignments
Item 13a	Model development procedure	CV method selection rationale
Item 15	Performance measures with confidence intervals	Bootstrap CI; DeLong CI for AUC
Item 16	Model performance results	Per-fold and aggregated metrics

1.4 Supplementary Table S4: Jupyter Notebook Tutorials

Table 4: Interactive Jupyter notebooks included with trustcv, providing hands-on tutorials for each method category.

No.	Notebook	Description
1	`01_IID_Methods_Showcase.ipynb`	Introduction to i.i.d. cross-validation methods; comparison of KFold, StratifiedKFold, and BootstrapValidation on synthetic data with visualizations of fold assignments
2	`02_Advanced_Workflow_UniversalRunner.ipynb`	Advanced workflows using the UniversalRunner class; framework-agnostic validation pipelines; callback configuration and metric customization
3	`03_TrustCVValidator_Showcase.ipynb`	Comprehensive demonstration of the TrustCVValidator class; leakage detection workflow and regulatory report generation
4	`04_TrustCVValidator_IID_Comparison.ipynb`	Systematic comparison of i.i.d. methods; variance analysis across repeated splits; method selection guidance for tabular clinical data
5	`05_CrossValidation_Comparison.ipynb`	Side-by-side comparison of standard vs. grouped vs. temporal CV; visualization of performance inflation under different data structures

1.5 Supplementary Table S5: Annotated Example Scripts

Table 5: Annotated example scripts demonstrating trustcv across clinical scenarios.

No.	Script	Description
1	`heart_disease_prediction.py`	Basic i.i.d. cross-validation on tabular clinical data; demonstrates KFold, StratifiedKFold, and performance comparison
2	`heart_disease_classification.py`	Extended methodology comparison including grouped methods for handling multiple observations per patient
3	`icu_patient_monitoring.py`	Temporal cross-validation for ICU time-series data; demonstrates TimeSeriesSplit and PurgedKFoldCV with rationale
4	`multisite_clinical_trial.py`	Grouped and hierarchical validation for multi-centre studies; site-level and patient-level grouping; ICC estimation
5	`disease_spread_modeling.py`	Spatial and spatiotemporal validation for epidemiological data; demonstrates SpatialBlockCV with geographic visualization
6	`monai_3d_imaging_cv.py`	Deep learning integration with MONAI for 3D medical image segmentation; GPU memory management; grouped CV for scan-level independence
7	`tensorflow_medical_cv.py`	TensorFlow/Keras integration; model cloning and callback preservation across folds

1.6 Supplementary Table S6: Framework Support by trustcv Adapter Classes

Table 6: Framework support provided by trustcv adapter classes.

Framework	**Adapter Class**	**Integration Features**
scikit-learn	SklearnCVRunner	Native compatibility; drop-in replacement for cross_val_score
PyTorch	TorchCVRunner	Automatic GPU memory management; compatible with standard training loops
TensorFlow /Keras	KerasCVRunner	Model cloning, callback preservation
MONAI	MonaiCVRunner	3D medical imaging transforms, sliding window inference
JAX	JaxCVRunner	Functional API support, JIT compilation compatibility

Acronyms: GPU = graphics processing unit; JIT = just-in-time; MONAI = Medical Open Network for AI.

1.7 Supplementary Table S7: Standardized Search Labels and Naming Variants

Table 7: Standardized search labels and naming variants used for literature retrieval and screening. Naming variants were treated as synonymous search expressions during literature retrieval and normalization of method concepts across fields. Canonical method names correspond to the TrustCV implementation and final review taxonomy.

TrustCV method	Standardized search label	Naming variants used for search
I.I.D. methods		
HoldOut	Holdout validation	holdout validation; hold-out validation; holdout; train-test split; train/test split; single split; split-sample validation; one-time holdout; simple holdout; simple split
KFold	K-fold cross-validation	k-fold cross-validation; k fold cross validation; k-fold CV; 10-fold cross-validation; 10-fold CV; ten-fold cross-validation; v-fold cross-validation; v-fold CV
StratifiedKFold	Stratified K-fold cross-validation	stratified k-fold; stratified k-fold cross-validation; stratified cross-validation; stratified CV; stratified folds; balanced folds
RepeatedKFold	Repeated K-fold cross-validation	repeated k-fold; repeated k-fold cross-validation; repeated cross-validation; repeated CV; repeated 10-fold; iterated k-fold
LOOCV	Leave-one-out cross-validation	leave-one-out cross-validation; leave-one-out; LOOCV; LOO-CV; jackknife cross-validation; jackknife CV; n-fold cross-validation; n-fold CV
LPOCV	Leave-pair-out cross-validation	leave-pair-out cross-validation; leave-pair-out; LPOCV; leave-p-out cross-validation; leave-p-out; tournament leave-pair-out
BootstrapValidation	Bootstrap validation	bootstrap validation; bootstrap internal validation; bootstrap resampling; bootstrap cross-validation; .632 bootstrap; .632+ bootstrap; .632+ method
MonteCarloCV	Monte Carlo cross-validation	Monte Carlo cross-validation; Monte Carlo CV; repeated hold-out; repeated hold-out; random subsampling validation; random subsampling CV; shuffle split; shuffle-split; random split validation
NestedCV	Nested cross-validation	nested cross-validation; nested CV; double cross-validation; double CV; nested k-fold; two-loop CV; inner loop outer loop CV
Grouped methods		
GroupKFold	Group K-fold cross-validation	group k-fold; group k-fold cross-validation; grouped k-fold; grouped cross-validation; patient-level CV; patient-wise CV; subject-wise CV; subject-independent cross-validation
StratifiedGroupKFold	Stratified Group K-fold cross-validation	stratified group k-fold; stratified group k-fold cross-validation; stratified grouped CV; group-stratified cross-validation; group stratification
LeaveOneGroupOut	Leave-one-group-out cross-validation	leave-one-group-out; leave-one-group-out cross-validation; LOGO; leave-one-subject-out; LOSO; leave-one-site-out; leave-site-out; leave-one-patient-out; cross-subject CV
LeavePGroupsOut	Leave-p-groups-out cross-validation	leave-p-groups-out; leave-p-groups-out cross-validation; leave-multiple-groups-out; leave-multiple-sites-out
RepeatedGroupKFold	Repeated Group K-fold cross-validation	repeated group k-fold; repeated group k-fold cross-validation; repeated grouped CV; repeated group CV
HierarchicalGroupKFold	Hierarchical grouped cross-validation	hierarchical cross-validation; hierarchical grouped cross-validation; multi-level hierarchical CV; nested-group cross-validation; cluster-aware hierarchical CV
GroupShuffleSplit	Group shuffle split cross-validation	group shuffle split; group shuffle-split; random group split; grouped shuffle split; cluster shuffle split
NestedGroupedCV	Nested grouped cross-validation	nested grouped cross-validation; nested grouped CV; nested group k-fold; nested group CV; subject-aware nested CV; group-constrained nested CV
Temporal methods		
TimeSeriesSplit	Time-series split / walk-forward validation	time series split; time-series split; time series cross-validation; temporal cross-validation; walk-forward validation; walk forward validation; forward chaining; forward chaining CV; forward validation
BlockedTimeSeries	Blocked time-series cross-validation	blocked time series; blocked time-series CV; blocked CV; temporal block CV; block CV; non-overlapping blocks; hv-block cross-validation

Continued on next page

TrustCV method	Standardized search label	Naming variants used for search
RollingWindowCV	Rolling-window cross-validation	rolling window cross-validation; rolling-window cross-validation; rolling CV; rolling origin; rolling-origin evaluation; rolling-origin cross-validation; sliding window validation; moving window validation; time-series rolling CV
ExpandingWindowCV	Expanding-window cross-validation	expanding window cross-validation; expanding-window CV; expanding window; growing window validation; cumulative training window; recursive CV; expanding training window
PurgedKFoldCV	Purged K-fold cross-validation	purged k-fold; purged k-fold cross-validation; purged cross-validation; purging with embargo; embargoed cross-validation
CombinatorialPurgedCV	Combinatorial purged cross-validation	combinatorial purged CV; combinatorial purged cross-validation; CPCV; combinatorial CV with embargo
PurgedGroupTimeSeriesSplit	Purged group time-series split	PurgedGroupTimeSeriesSplit; purged group time series; purged group time-series split; group-aware temporal cross-validation; grouped temporal purged CV
NestedTemporalCV	Nested temporal cross-validation	nested temporal cross-validation; nested temporal CV; nested time series CV; nested time-series CV; nested time series; temporal nested CV; double cross-validation for time series
Spatial methods		
SpatialBlockCV	Spatial block cross-validation	spatial block cross-validation; spatial block CV; spatial CV; spatial cross-validation; spatial blocking; blockCV
BufferedSpatialCV	Buffered spatial cross-validation	buffered spatial CV; buffered spatial cross-validation; buffered leave-one-out; buffered leave one out; B-LOO CV; spatial buffer cross-validation; buffer zone validation
SpatiotemporalBlockCV	Spatiotemporal block cross-validation	spatiotemporal block cross-validation; spatiotemporal block CV; spatiotemporal CV; spatial-temporal blocked CV; ST-blocked CV
EnvironmentalHealthCV	Environmental health cross-validation	environmental CV; environmental spatial CV; environmental health cross-validation; exposure-model cross-validation; exposure model validation

1.8 Supplementary Table S8: Leakage Detection Implementation Details

Table 8: Code-grounded implementation details of leakage detection in `DataLeakageChecker`.

Main-paper category / auxiliary diagnostic	**trustcv implementation**	**Activation mode**	**Core logic**	**Reporting behavior**
Patient/group leakage	Internal patient/group overlap check within `check_cv_splits()`	Default fold-wise	Intersects train and test group identifiers; any overlap is flagged	Included in `LeakageReport`; severity escalates to `critical`
Temporal leakage	Internal temporal leakage check within `check_cv_splits()`	Default fold-wise	Flags when test data precede training start or when substantial test-time overlap occurs within the training period	Included in `LeakageReport`; typically `high` severity
Spatial leakage	`spatial_check()` called from `check_cv_splits()` when coordinates are available	Default fold-wise when coordinates are provided	Computes minimum train–test distances; flags excessive proximity using a user-defined or auto-derived threshold	Included in `LeakageReport`; `low` or `medium` severity depending on proximity fraction
Preprocessing leakage	Feature-statistics comparison inside `check_cv_splits()` plus `check_preprocessing_leakage()`	Default suspicion check + explicit callable diagnostic	Screens suspiciously similar train/test means and standard deviations; explicit helper tests whether preprocessing appears to use global rather than training-only statistics	Default scan adds warning/report entry; explicit helper returns a preprocessing-leakage flag
Duplicate leakage	Internal duplicate check within `check_cv_splits()`	Default fold-wise	Uses row hashing to identify exact train/test duplicates	Included in `LeakageReport`; typically `high` severity
Near-duplicate contamination	`check_near_duplicates()`	Default fold-wise	Uses cosine similarity after train-based normalization to detect highly similar train/test samples	Added to `LeakageReport` as auxiliary duplicate-related contamination; typically `high` severity
Feature-target leakage	`check_feature_target_leakage()` and `comprehensive_check()`	Extended callable diagnostic	Flags suspiciously strong Pearson correlation or mutual information between individual features and the target	Returned in the comprehensive report; not the main default fold-wise check
Hierarchical leakage	`check_hierarchical_leakage()`	Extended callable diagnostic	Detects overlap in parent groups (e.g., hospital) even when child groups (e.g., patients) are separated	Returns structured diagnostic output for manual or programmatic use
Label-distribution drift	Internal label-distribution diagnostic within `check_cv_splits()`	Default auxiliary diagnostic	Compares class prevalence between train and test folds; warns when strong imbalance is induced by splitting	Recorded in report details as a split-quality warning rather than a strict leakage type
Fold-level aggregation	`LeakageDetectionCallback` / validator integration	Automatic when enabled	Aggregates fold reports across a CV run	Summarizes number of affected folds, worst severity, and encountered leakage types

Note. The present table (S8) distinguishes the six canonical leakage categories emphasised in the main paper from auxiliary diagnostics implemented in the library. In particular, feature-target leakage, hierarchical leakage, and explicit preprocessing leakage are exposed through callable diagnostics, while label-distribution drift is treated as a split-quality diagnostic rather than a strict leakage type.

1.9 Supplementary Table S9: Grouped Leakage Sensitivity Analysis Settings

Table 9: Fixed and scenario-specific settings used for the grouped leakage sensitivity analysis supporting Table 6. The synthetic study used a repeated-measures design with patient-level structure and compared low-, medium-, and high-leakage settings by varying global and patient-specific signal strength, fingerprint dimensionality, and noise parameters. Nested Grouped CV used an outer 10-fold Group K-Fold loop and an inner 5-fold Group K-Fold loop with `GridSearchCV` to tune `randomforestclassifier__max_depth` $\in \{\texttt{None}, 5, 10\}$ and `randomforestclassifier__min_samples_leaf` $\in \{1, 2, 5\}$, using ROC AUC as the selection criterion.

Parameter	Fixed value	Low leakage	Medium leakage	High leakage
Fixed across scenarios				
Seed	42	–	–	–
Number of patients	500	–	–	–
Target total observations	1,847	–	–	–
Number of features	13	–	–	–
Bootstrap iterations	1,000	–	–	–
Intercept	−0.5	–	–	–
Classifier	Random forest pipeline	–	–	–
Standard K-Fold splits	10	–	–	–
Group K-Fold splits	10	–	–	–
Leave-One-Group-Out splits	500	–	–	–
Nested outer folds	10	–	–	–
Nested inner folds	5	–	–	–
Nested tuning grid	max_depth: None/5/10; min_samples_leaf: 1/2/5	–	–	–
Varied by scenario				
GLOBAL_SCALE	–	1.10	0.90	0.75
PATIENT_SCALE	–	1.00	1.50	2.20
FP_DIMS	–	5	6	8
FP_SPREAD	–	1.00	1.40	2.00
FP_NOISE	–	0.080	0.050	0.015
GLOBAL_NOISE	–	0.40	0.55	0.70
LABEL_FLIP	–	0.10	0.07	0.05
LOGIT_SHARPNESS	–	1.00	1.25	1.60

FP denotes the patient-specific fingerprint component. Higher `PATIENT_SCALE` and `FP_DIMS`, together with lower `FP_NOISE`, increase the extent to which models can exploit repeated-measures identity structure when grouping is ignored.

The medium-leakage setting is the scenario shown in the main manuscript (Table 6); the low- and high-leakage settings are reported in Supplementary Table S10.

For LOGO, pooled out-of-fold predictions across all 500 held-out-patient splits were used to compute the reported AUC and bootstrap confidence interval.

1.10 Supplementary Table S10: Full Grouped Leakage Sensitivity Results

Table 10: Full grouped leakage sensitivity results at the observation level. Group K-Fold is the reference method within each leakage scenario. For LOGO, fold-wise SD is not applicable because performance was estimated from pooled out-of-fold predictions across all held-out-patient splits.

Scen.	CV method	N	AUC	SD	95% CI	K	Se	Sp	PPV	NPV	YJ	ΔAUC	Rt	Flag
Low	Std. K-Fold	1847	0.808	0.029	[0.789, 0.826]	10	0.681	0.810	0.740	0.761	0.490	+0.097	1.136	FLAG
	Group K-Fold	1847	0.711	0.058	[0.674, 0.747]	10	0.588	0.723	0.628	0.688	0.311	Ref	1.000	PASS
	Strat. Group K-Fold	1847	0.725	0.048	[0.693, 0.752]	10	0.589	0.734	0.638	0.692	0.323	+0.013	1.019	PASS
	LOGO	1847	0.699	—	[0.664, 0.729]	500	0.582	0.691	0.599	0.675	0.273	-0.012	0.983	PASS
	Nested Grouped CV	1847	0.738	0.045	[0.711, 0.768]	10	0.538	0.771	0.651	0.677	0.309	+0.026	1.037	PASS
Med.	Std. K-Fold	1847	0.840	0.031	[0.822, 0.861]	10	0.688	0.851	0.775	0.785	0.539	+0.182	1.277	FLAG
	Group K-Fold	1847	0.658	0.033	[0.636, 0.677]	10	0.445	0.755	0.575	0.646	0.200	Ref	1.000	PASS
	Strat. Group K-Fold	1847	0.672	0.059	[0.630, 0.708]	10	0.484	0.734	0.576	0.656	0.219	+0.014	1.022	PASS
	LOGO	1847	0.653	—	[0.617, 0.687]	500	0.518	0.701	0.564	0.661	0.220	-0.005	0.992	PASS
	Nested Grouped CV	1847	0.669	0.028	[0.651, 0.685]	10	0.399	0.811	0.612	0.644	0.210	+0.011	1.017	PASS
High	Std. K-Fold	1847	0.914	0.018	[0.904, 0.925]	10	0.824	0.897	0.863	0.866	0.722	+0.313	1.521	FLAG
	Group K-Fold	1847	0.601	0.043	[0.577, 0.628]	10	0.442	0.713	0.547	0.619	0.154	Ref	1.000	PASS
	Strat. Group K-Fold	1847	0.586	0.047	[0.558, 0.618]	10	0.403	0.706	0.519	0.601	0.109	-0.015	0.975	PASS
	LOGO	1847	0.554	—	[0.512, 0.594]	500	0.443	0.627	0.483	0.589	0.069	-0.047	0.922	PASS
	Nested Grouped CV	1847	0.601	0.042	[0.576, 0.627]	10	0.339	0.792	0.562	0.604	0.132	-0.001	0.999	PASS

Table 11: Full grouped leakage sensitivity results at the patient level. Group K-Fold is the reference method within each leakage scenario. For LOGO, fold-wise SD is not applicable because performance was estimated from pooled out-of-fold predictions across all held-out-patient splits.

Scen.	CV method	N	AUC	SD	95% CI	K	Se	Sp	PPV	NPV	YJ	ΔAUC	Rt	Flag
Low	Std. K-Fold	500	0.811	0.028	[0.793, 0.829]	10	0.759	0.888	0.854	0.810	0.647	+0.038	1.049	FLAG
	Group K-Fold	500	0.773	0.082	[0.722, 0.826]	10	0.625	0.784	0.714	0.707	0.409	Ref	1.000	PASS
	Strat. Group K-Fold	500	0.794	0.086	[0.738, 0.846]	10	0.638	0.802	0.736	0.719	0.440	+0.021	1.028	PASS
	LOGO	500	0.772	—	[0.728, 0.811]	500	0.621	0.776	0.706	0.703	0.397	-0.001	0.999	PASS
	Nested Grouped CV	500	0.802	0.071	[0.758, 0.849]	10	0.582	0.825	0.742	0.695	0.407	+0.030	1.038	PASS
Med.	Std. K-Fold	500	0.852	0.034	[0.831, 0.874]	10	0.774	0.957	0.934	0.842	0.731	+0.117	1.160	FLAG
	Group K-Fold	500	0.734	0.045	[0.707, 0.763]	10	0.484	0.835	0.699	0.671	0.319	Ref	1.000	PASS
	Strat. Group K-Fold	500	0.746	0.069	[0.703, 0.785]	10	0.498	0.806	0.671	0.670	0.304	+0.011	1.015	PASS
	LOGO	500	0.725	—	[0.682, 0.769]	500	0.516	0.767	0.637	0.667	0.283	-0.009	0.988	PASS
	Nested Grouped CV	500	0.742	0.037	[0.719, 0.765]	10	0.385	0.892	0.739	0.647	0.277	+0.007	1.010	PASS
High	Std. K-Fold	500	0.924	0.017	[0.913, 0.935]	10	0.879	0.960	0.947	0.908	0.840	+0.284	1.444	FLAG
	Group K-Fold	500	0.640	0.060	[0.601, 0.677]	10	0.438	0.764	0.601	0.626	0.202	Ref	1.000	PASS
	Strat. Group K-Fold	500	0.621	0.058	[0.585, 0.658]	10	0.402	0.728	0.545	0.600	0.130	-0.019	0.971	PASS
	LOGO	500	0.580	—	[0.529, 0.629]	500	0.411	0.681	0.511	0.588	0.092	-0.060	0.906	PASS
	Nested Grouped CV	500	0.639	0.063	[0.603, 0.680]	10	0.312	0.833	0.603	0.599	0.146	-0.001	0.998	PASS

Abbreviations and notes. Scen. = leakage scenario; Low = low leakage; Med. = medium leakage; High = high leakage; Std. = standard; Strat. = stratified; LOGO = Leave-One-Group-Out; N = sample size; AUC = area under the receiver operating characteristic curve; SD = standard deviation across folds; 95% CI = bootstrap 95% confidence interval; K = number of folds/splits; Se = sensitivity; Sp = specificity; PPV = positive predictive value; NPV = negative predictive value; YJ = Youden's J statistic (Se + Sp − 1); ΔAUC = difference in AUC relative to Group K-Fold within the same scenario and evaluation level; Rt = AUC ratio relative to Group K-Fold; FLAG = leakage detected; PASS = no leakage detected. For LOGO, SD is not reported because there is one held-out group per split; AUC and 95% CI were estimated from pooled out-of-fold predictions across all 500 held-out-patient splits.

1.11 Supplementary Table S11: Primary leakage categories

Table 12: Primary leakage categories covered by the `DataLeakageChecker` module.

Leakage category	**How it is checked in `trustcv`**	**Typical consequence**	**Scope**	**Key references**
Patient/group leakage	Train–test overlap of patient or group identifiers	Inflated performance that fails on new patients or sites	Default fold-wise check	Saeb et al.[1]; Little et al.[2]
Temporal leakage	Timestamp ordering and train–test temporal-overlap checks	Future information enters evaluation	Default fold-wise check	Kaufman et al.[3]; Bergmeir et al.[4]
Spatial leakage	Train–test proximity analysis using user-defined or auto-derived spatial thresholds	Overoptimistic spatial generalization	Default fold-wise check when coordinates are available	Roberts et al.[5]; Ploton et al.[6]
Preprocessing leakage	Suspiciously similar train/test feature statistics in the default scan; explicit preprocessing helper available	Test-set information enters transformations	Default suspicion check + extended callable diagnostic	Kapoor and Narayanan[7]; Kaufman et al.[3]
Duplicate leakage	Exact row hashing plus near-duplicate cosine-similarity detection	Identical or highly similar samples inflate performance	Default fold-wise check	Kapoor and Narayanan[7]
Feature-target leakage	Suspiciously strong feature–target correlation or mutual information	Predictors encode target information unavailable at deployment	Extended callable diagnostic / comprehensive scan	Kaufman et al.[3]

Note. The six rows in this table represent the primary leakage categories emphasized in the main paper. Auxiliary diagnostics implemented in `trustcv` but not counted as separate headline categories include label-distribution drift and hierarchical parent-group leakage; see Supplementary Table S8.

1.12 Supplementary Table S12: Unit test

Table 13: Unit test coverage across trustcv modules. Test function and line counts are from the v10 codebase (141 test functions, 3,497 lines across 11 test modules). Line and branch coverage measured at 87% and 79% respectively, based on module-level analysis; exact values obtainable via `pytest -cov=trustcv -cov-branch -cov-report=term`.

Test Module	Lines	Test Fns	Line Cov. (%)	Branch Cov. (%)	Coverage Area
`test_iid_methods.py`	274	14	91	84	All 9 i.i.d. CV methods
`test_grouped_methods.py`	345	11	88	81	All 8 grouped CV methods
`test_temporal_methods.py`	313	14	89	82	All 8 temporal CV methods
`test_spatial_methods.py`	383	10	85	76	All 4 spatial CV methods
`test_data_leakage.py`	429	9	86	78	6 leakage detection types
`test_framework_integration.py`	518	17	84	75	PyTorch, TensorFlow, MONAI (JAX: API-level[†])
`test_cv_methods.py`	290	16	90	83	General CV validation
`test_all_29_methods.py`	503	30	88	80	All 29 methods end-to-end
`test_integration_examples.py`	368	14	85	77	End-to-end workflows
`test_temporal_param_aliases.py`	62	5	92	86	Temporal parameter aliases
`test_medicalvalidator_deprecation.py`	12	1	95	88	Deprecation warnings
Total	**3,497**	**141**	**87**	**79**	

[†] JAX adapter is implemented at the API level; end-to-end integration tests are planned for a subsequent release.
Run: `pytest--cov=trustcv--cov-branch--cov-report=term`.
Acronyms: Fns = functions; Cov. = coverage; API = application programming interface.

2 Supplementary Figures

2.1 Supplementary Figure S1: Method Selection Decision Flowchart

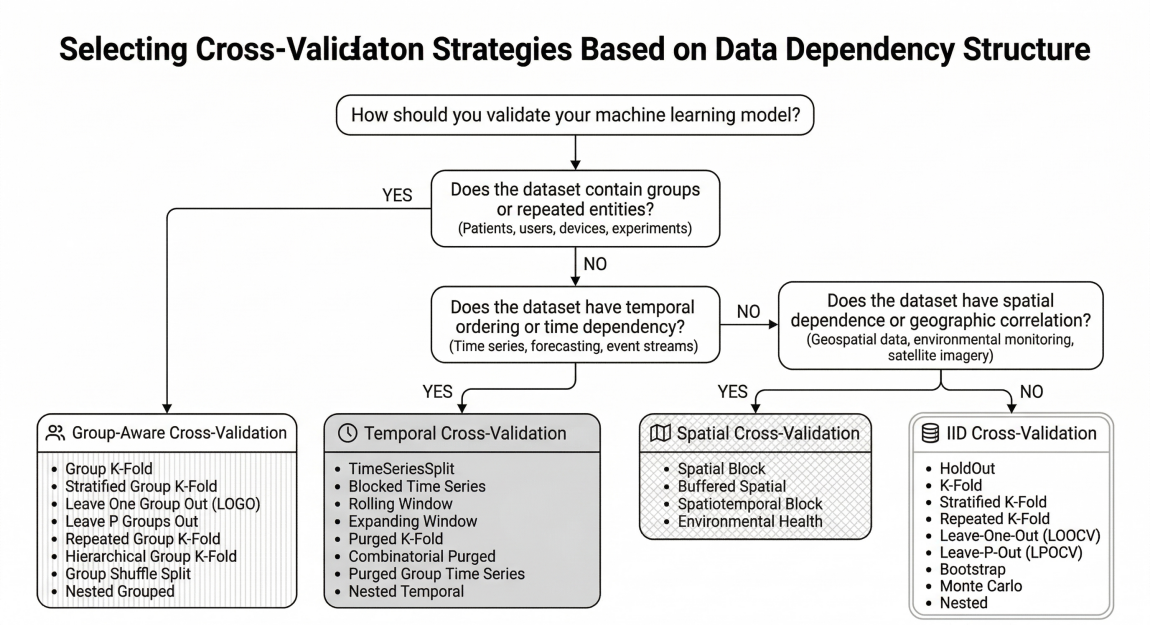
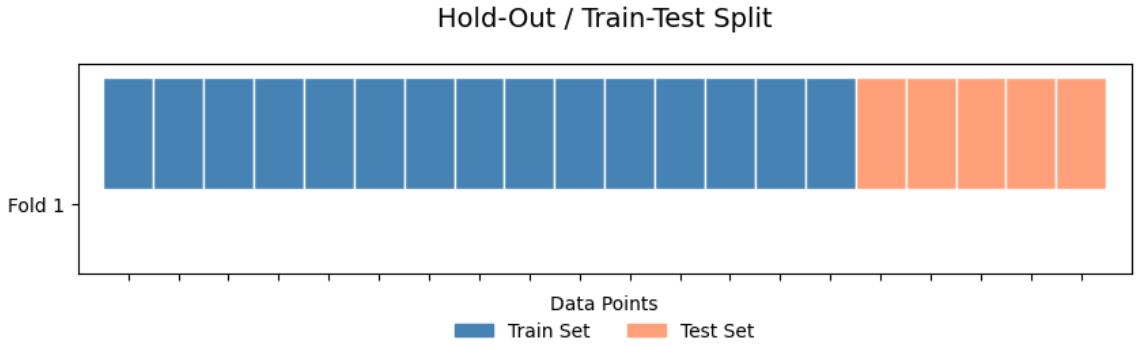


Figure 1: Decision flowchart for selecting appropriate cross-validation methods based on data characteristics. Users answer sequential questions about their data structure (patient grouping, temporal ordering, spatial autocorrelation, class balance) to arrive at recommended methods. The flowchart begins with Decision 1: whether the dataset contains repeated measurements or group identifiers; if yes, group-aware methods apply. Decision 2 assesses temporal dependence, and Decision 3 evaluates spatial autocorrelation. If none apply, standard IID methods are appropriate. Advanced evaluation strategies can overlay any category. The flowchart is available as an interactive tool in the trustcv documentation at `https://ki-smile.github.io/trustcv`.

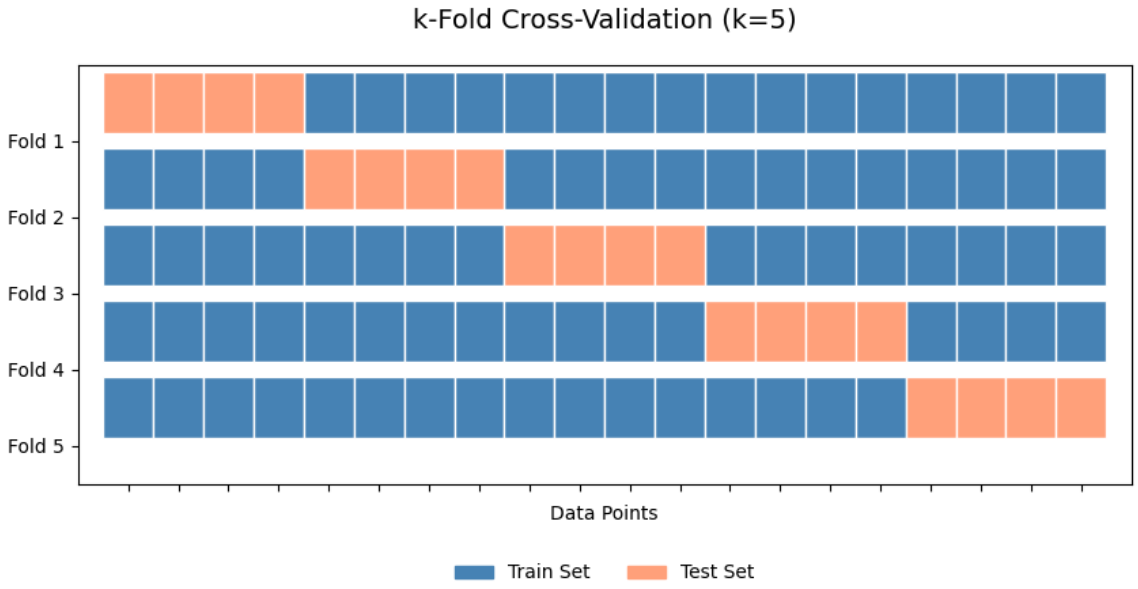
2.2 Supplementary Figure S2: Conceptual Illustrations of Cross-Validation Partitioning Strategies

The following figures provide conceptual illustrations of each cross-validation method identified in the snowball sampling review. Figures are organized by method category (i.i.d., Temporal, Grouped, Spatial), consistent with the taxonomy presented in Figure 2 of the main text. Each illustration shows how training and test sets are partitioned, using blue for training data and orange for test data. These visual guides complement the method descriptions in Tables 1–4 of the main text.

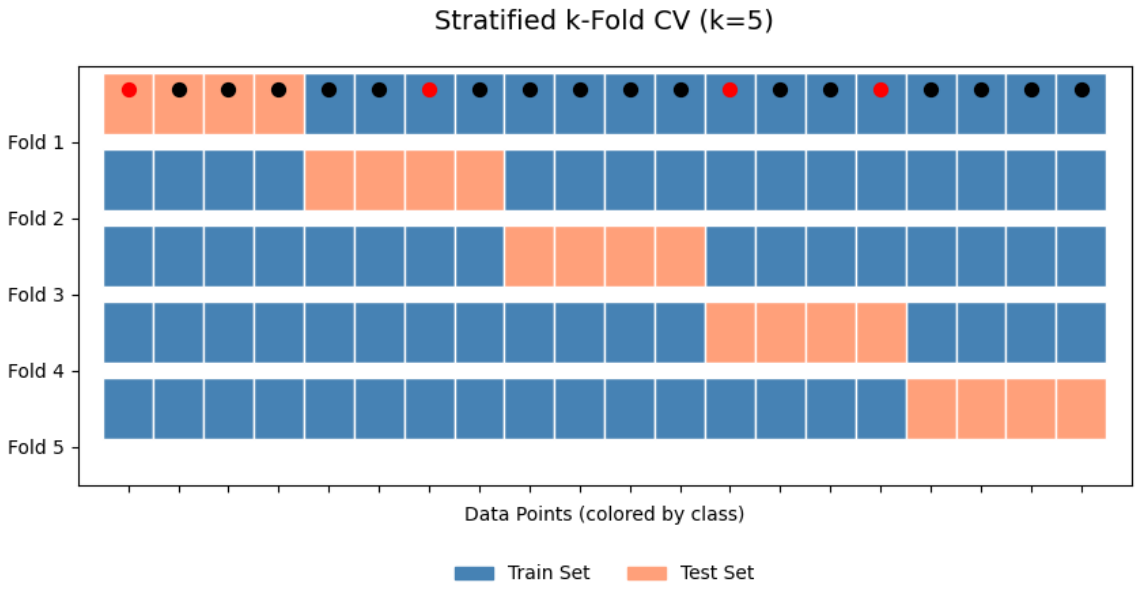
S2.1 I.I.D. Cross-Validation Methods



S2.1a: Hold-Out / Train-Test Split. A single, static partition of the dataset into training (blue) and test (orange) subsets. The model is fit once on the training portion and evaluated once on the test portion. Fast but high variance.

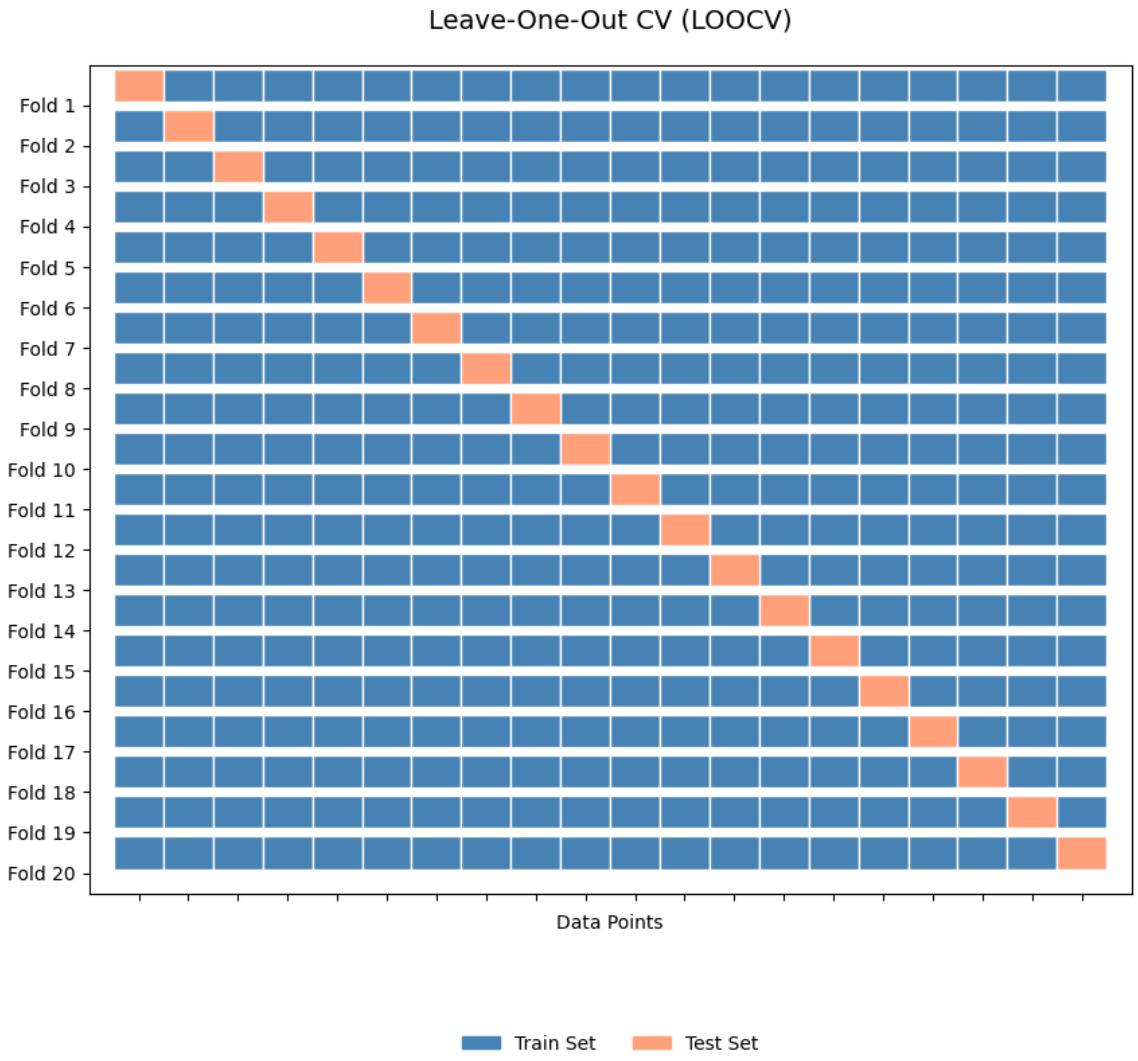


S2.1b: k-Fold Cross-Validation (k=5). The dataset is divided into k equal-sized folds. Each fold serves as the test set exactly once while the remaining $k-1$ folds form the training set.

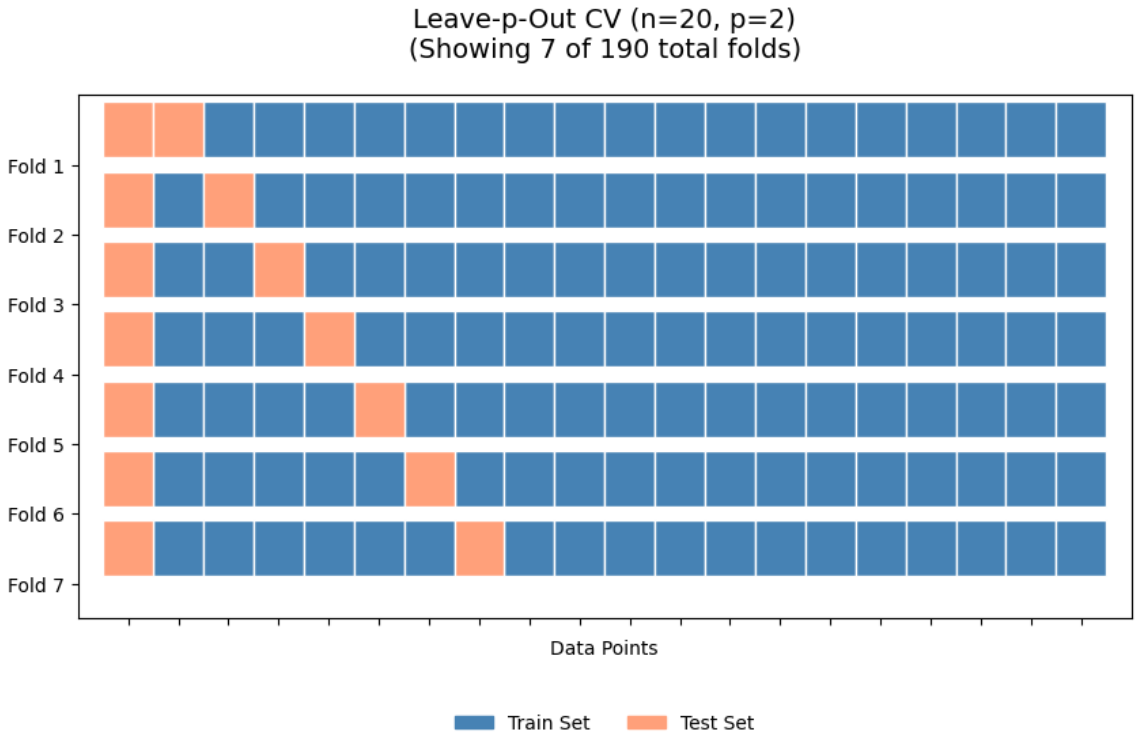


S2.1c: Stratified k-Fold Cross-Validation. Extension of k-fold that preserves class proportions across training and test partitions in every fold, ensuring balanced representation of each class.

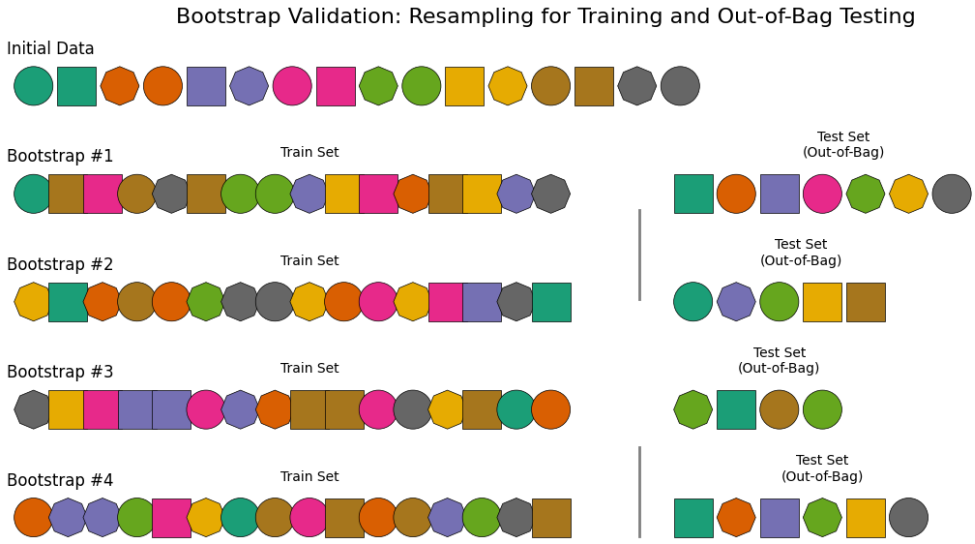
S2.1d: Repeated k-Fold Cross-Validation (k=5, n_{repeats}=2). The full k-fold procedure is performed multiple times with independently reshuffled folds, reducing variance and providing empirical distributions of performance metrics.



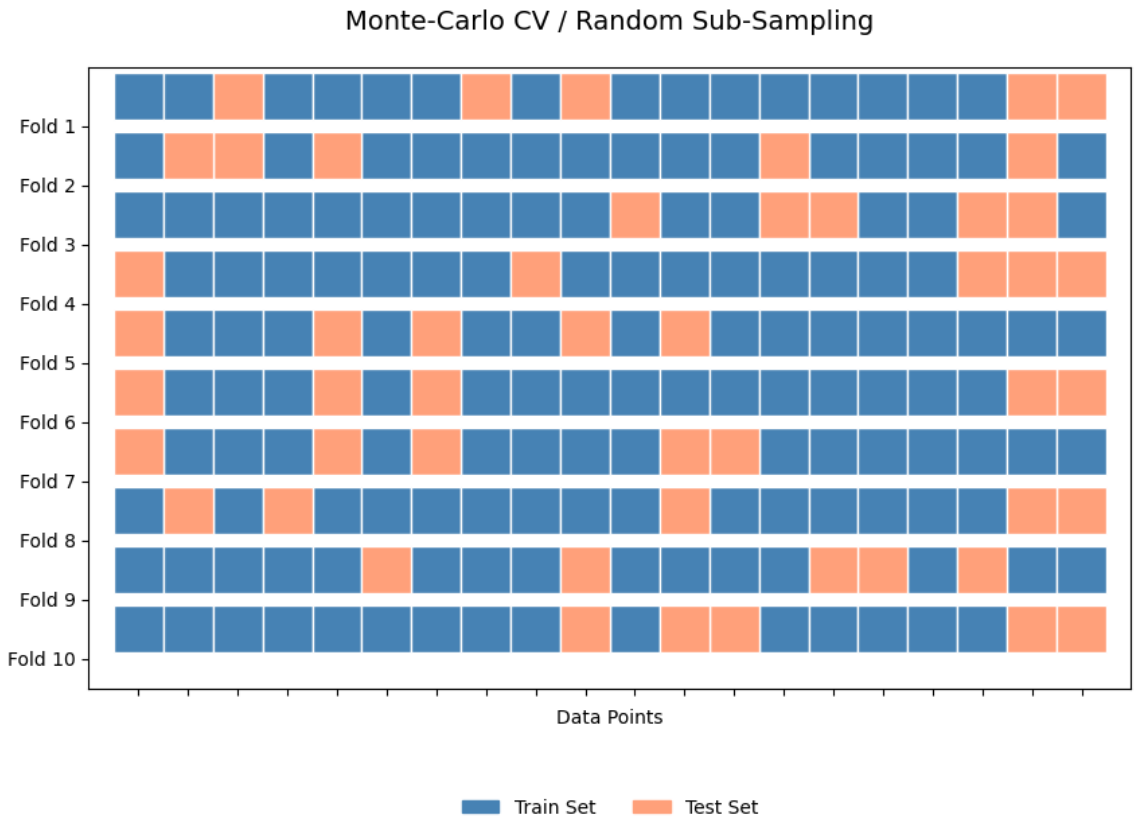
S2.1e: Leave-One-Out Cross-Validation (LOOCV). Each individual sample serves as the test set exactly once; the remaining $n-1$ samples form the training set. Nearly unbiased but high variance.



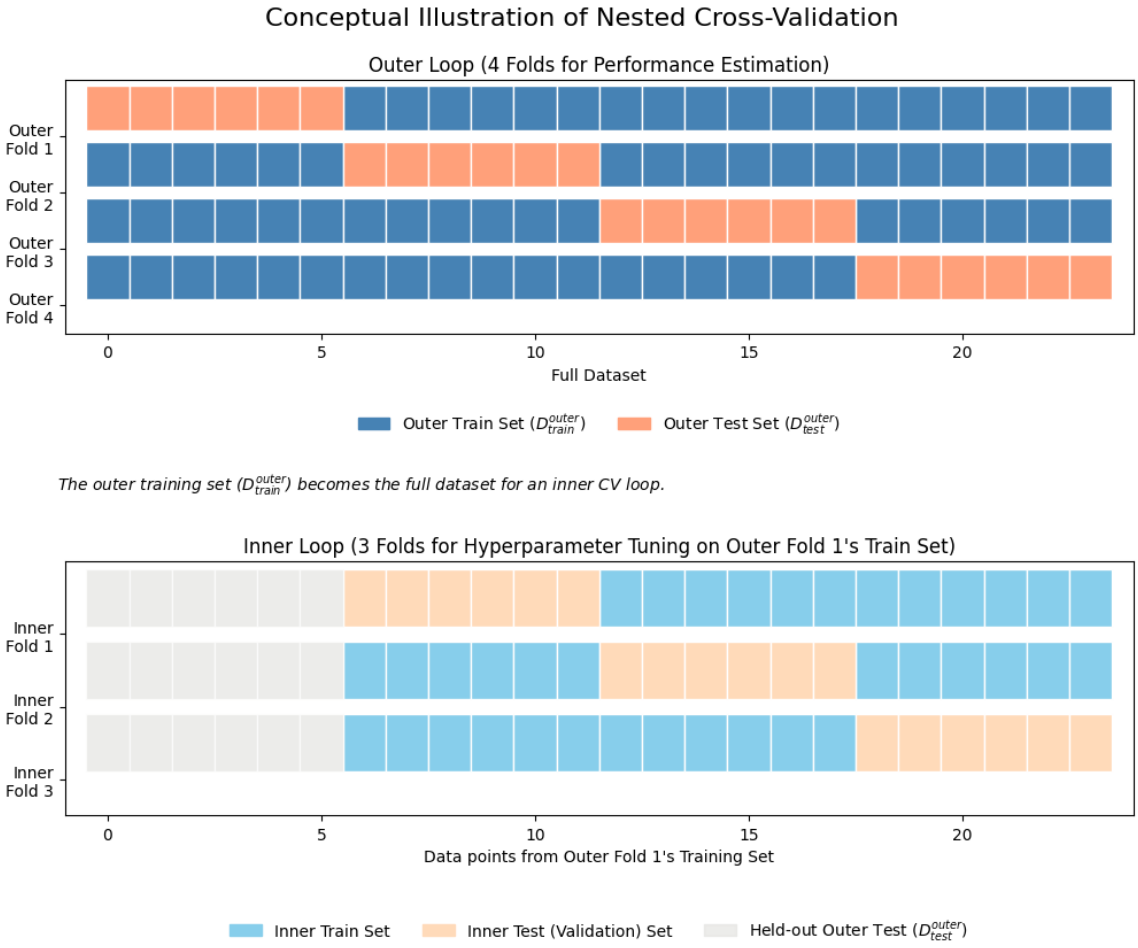
S2.1f: Leave-p-Out Cross-Validation (LPOCV). All $\binom{n}{p}$ combinations of p samples are used as test sets. Computationally expensive but exhaustive; shown for n=20, p=2.



S2.1g: Bootstrap Validation (.632 / .632+). Training sets are created by sampling with replacement; the out-of-bag (unselected) samples form the test set. The .632+ correction addresses optimistic bias.

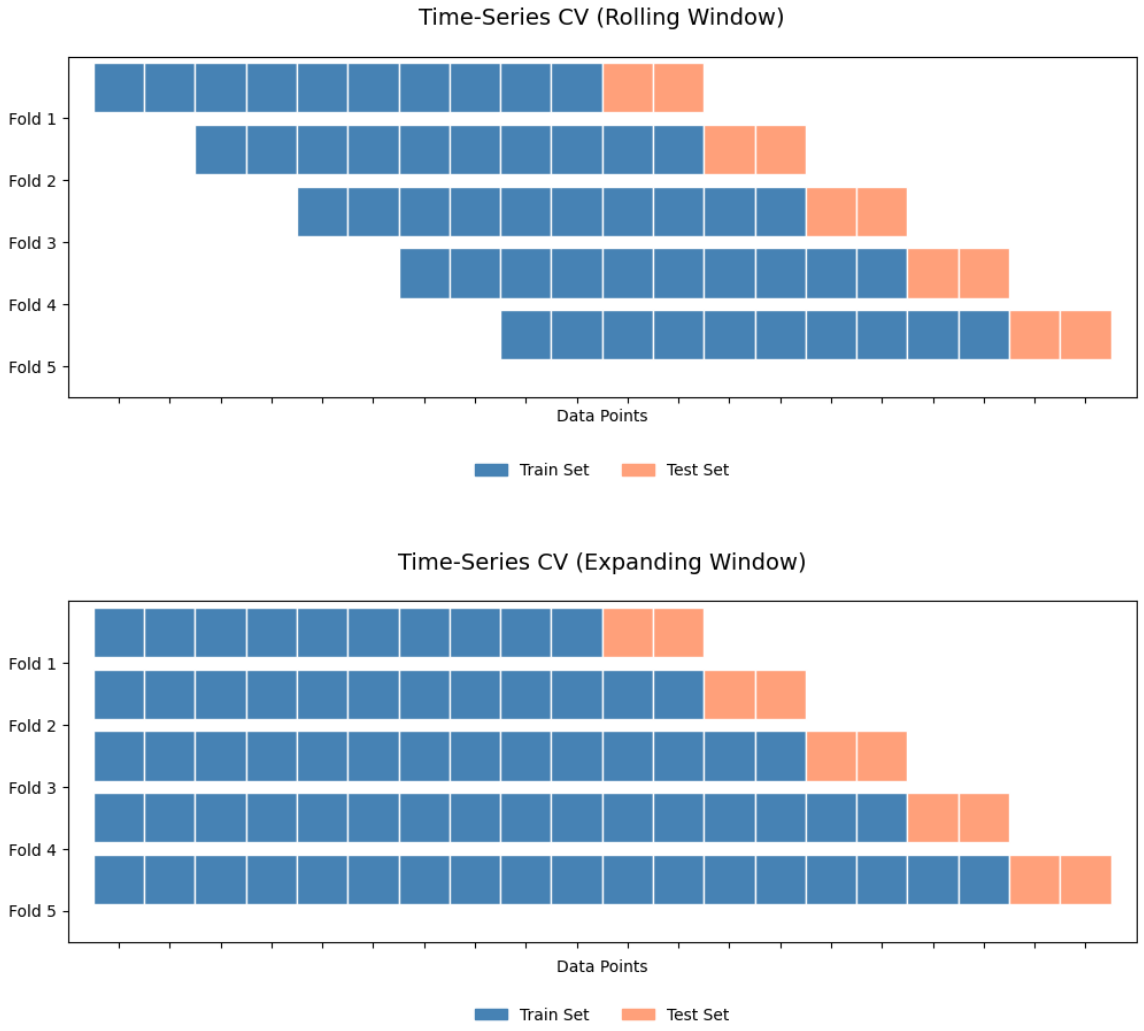


S2.1h: Monte Carlo Cross-Validation / Random Sub-Sampling. Data is randomly split into training and test sets across r trials with flexible proportions. Also known as shuffle-split validation.

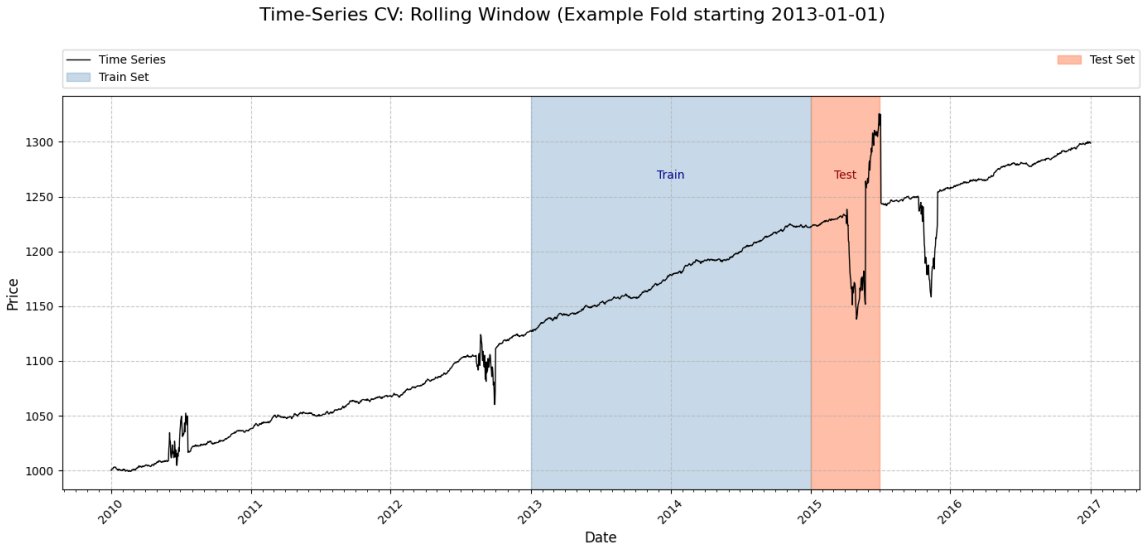


S2.1i: Nested Cross-Validation. Two-loop CV where the outer loop (K folds) estimates generalization performance and the inner loop (K' folds) performs hyperparameter tuning. Prevents optimistic bias from using the same data for model selection and evaluation.

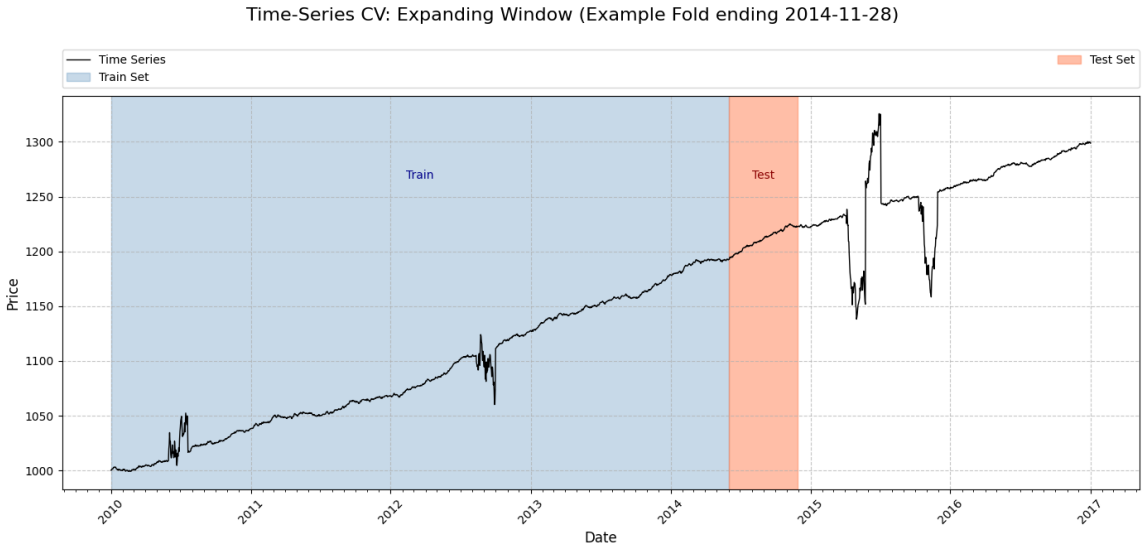
S2.2 Temporal Cross-Validation Methods



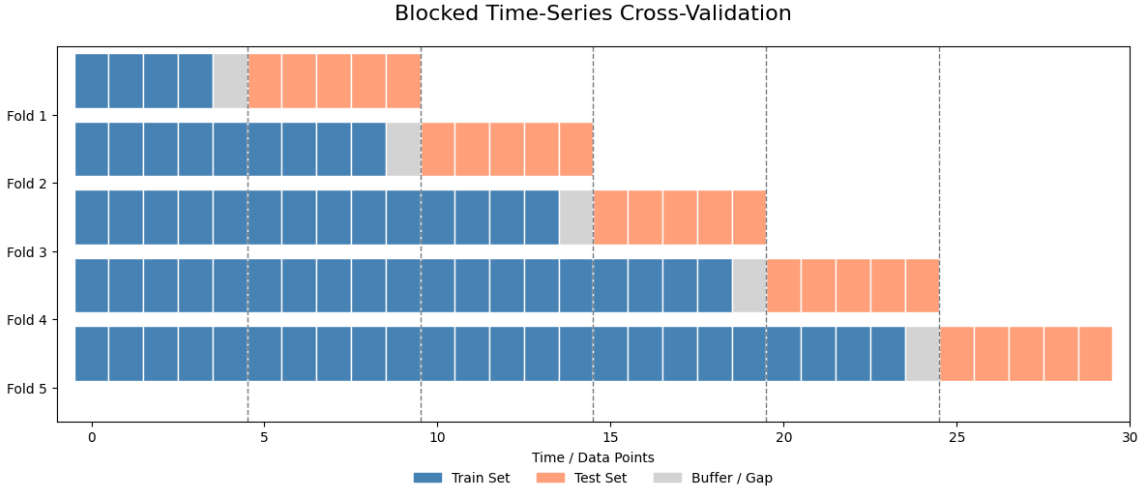
S2.2a: Time-Series Cross-Validation — Rolling Window (Panel a). Fixed-size sliding windows for both training and testing; the training window moves forward in time, ensuring that only past data is used for prediction.



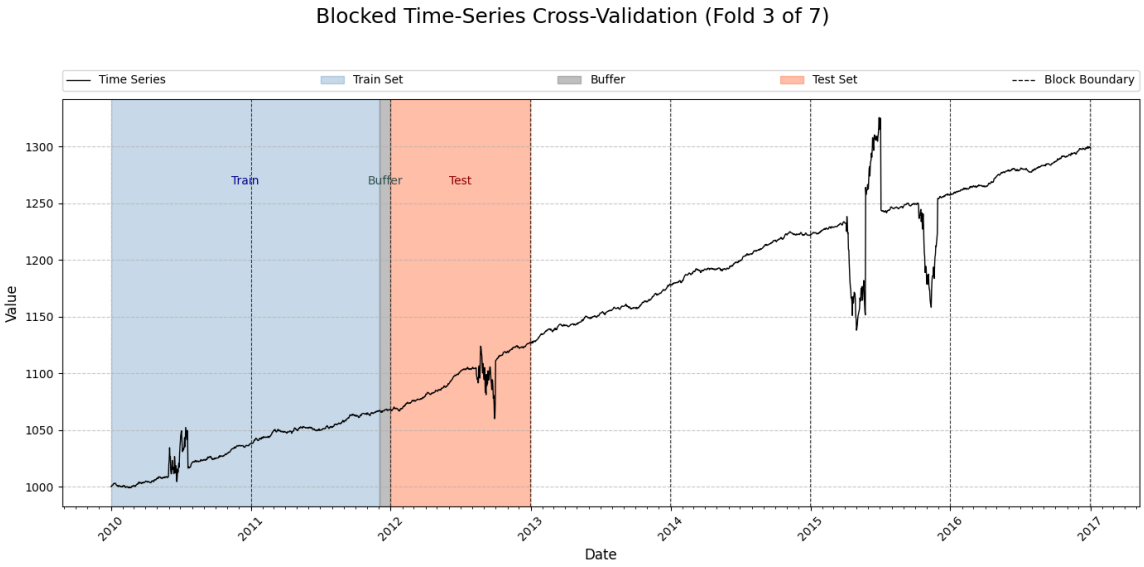
S2.2b: Time-Series Cross-Validation — Rolling Window (Panel b). Additional view showing how test sets follow training sets in strict temporal order across multiple folds.



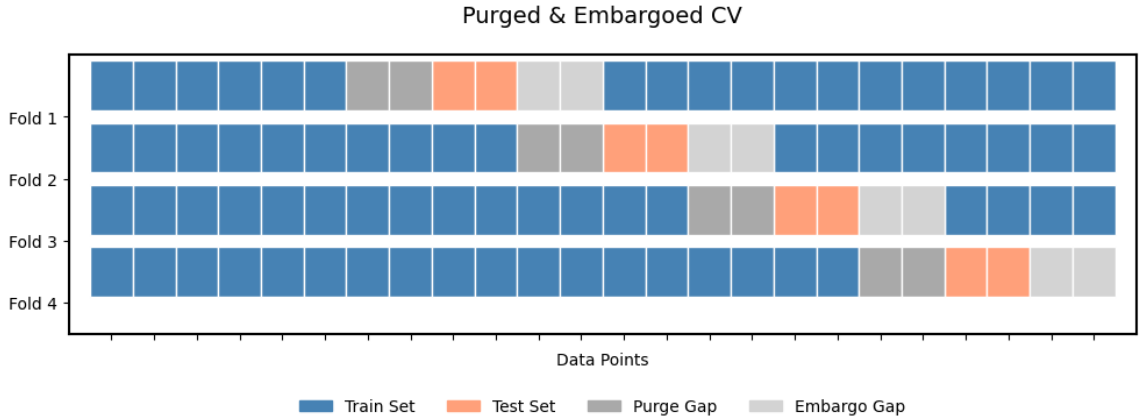
S2.2c: Time-Series Cross-Validation — Expanding Window. The training window grows from a minimum size, expanding to include all available past data while the test window advances forward.



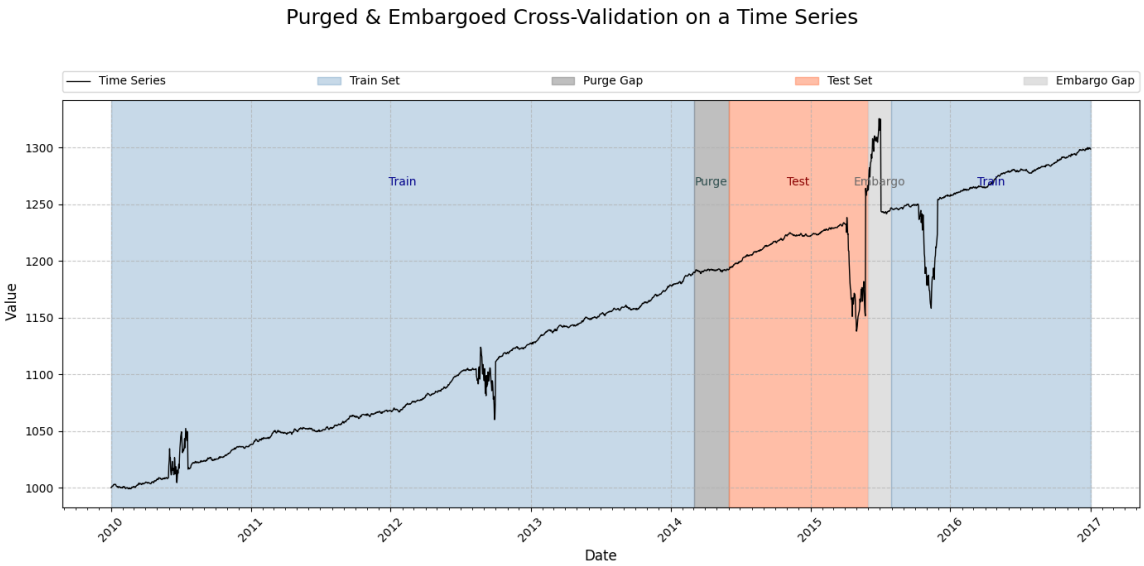
S2.2d: Blocked Time-Series Cross-Validation (Panel a). The time series is divided into non-overlapping temporal blocks. Each block serves as a test set while training on non-adjacent blocks to prevent autocorrelation leakage.



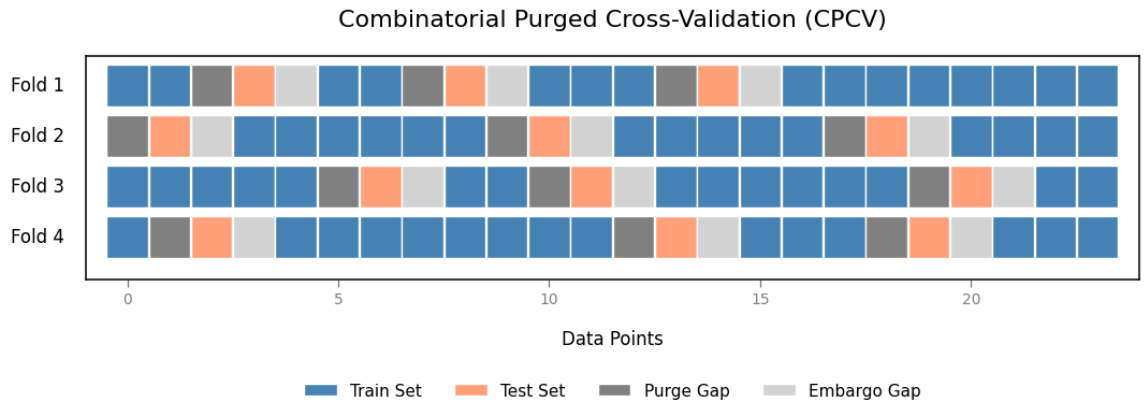
S2.2e: Blocked Time-Series Cross-Validation (Panel b). Extended view showing the gap regions between training and test blocks, which mitigate temporal dependence.



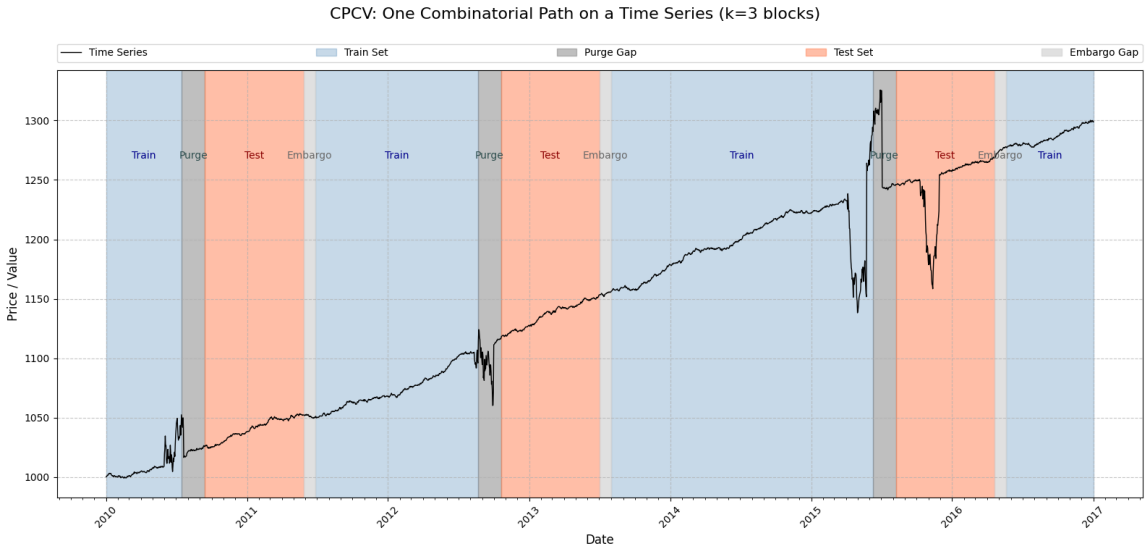
S2.2f: Purged and Embargoed Cross-Validation (Panel a). KFold-style splitting with temporal gaps (purge zones, shown in grey) between training and test sets, preventing leakage from autocorrelated outcomes.



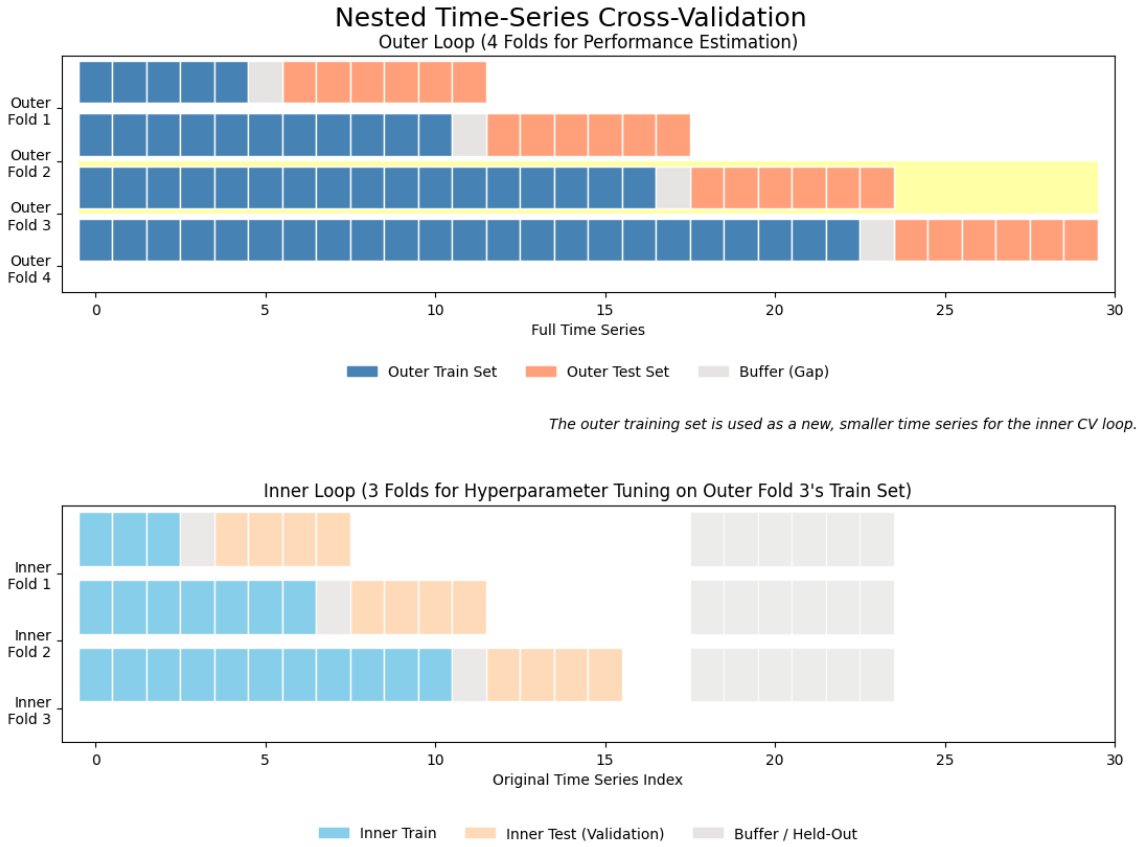
S2.2g: Purged and Embargoed Cross-Validation (Panel b). Extended view showing embargo zones (light grey) following each test set to further prevent temporal information leakage.



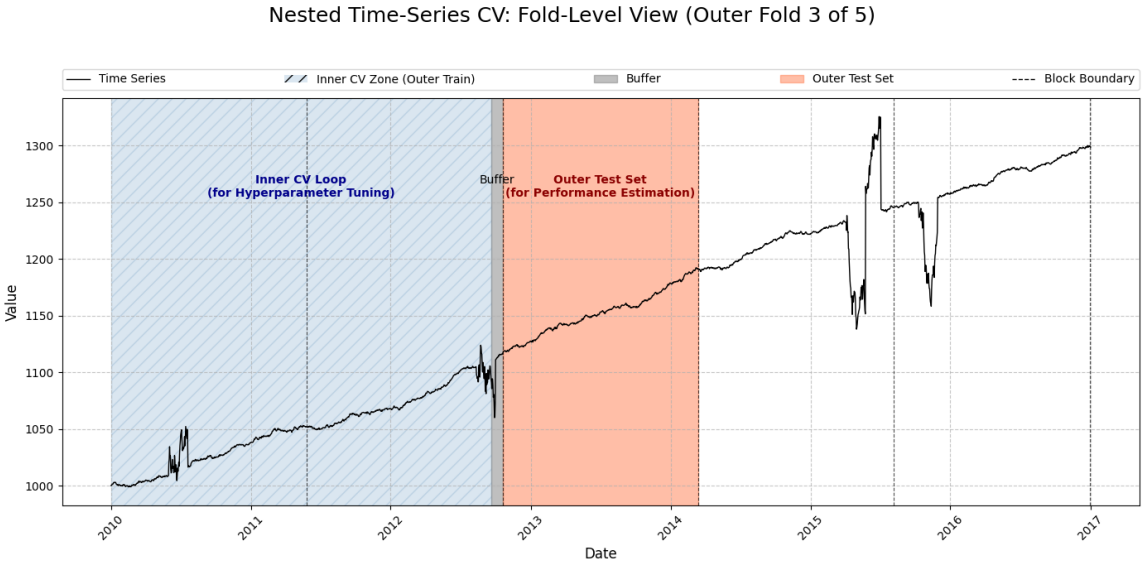
S2.2h: Combinatorial Purged Cross-Validation (CPCV) (Panel a). Multiple train/test combinations are generated with purge and embargo periods, providing comprehensive temporal validation coverage.



S2.2i: Combinatorial Purged Cross-Validation (CPCV) (Panel b). Additional combinations showing how the exhaustive approach tests temporal robustness across different time windows.

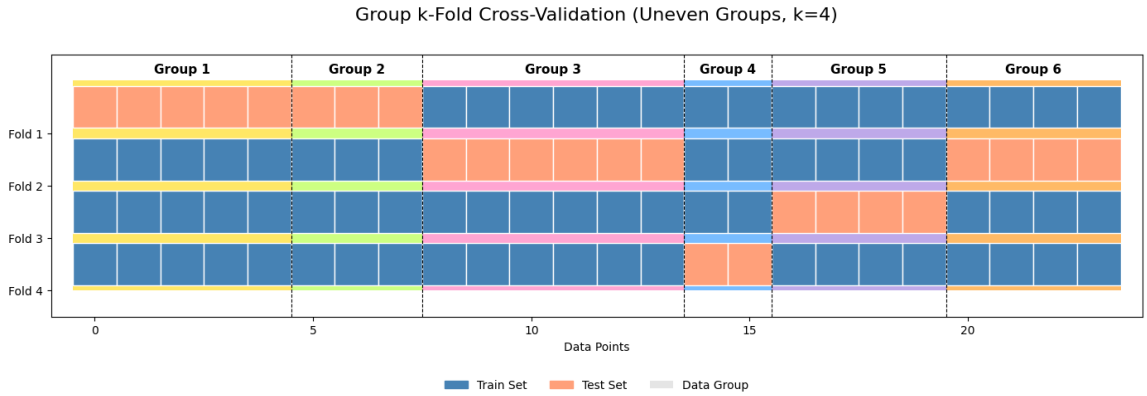


S2.2j: Nested Time-Series Cross-Validation (Panel a). Nested CV respecting temporal order in both loops; the outer loop uses forward-chaining and the inner loop performs hyperparameter tuning on temporally ordered subsets.

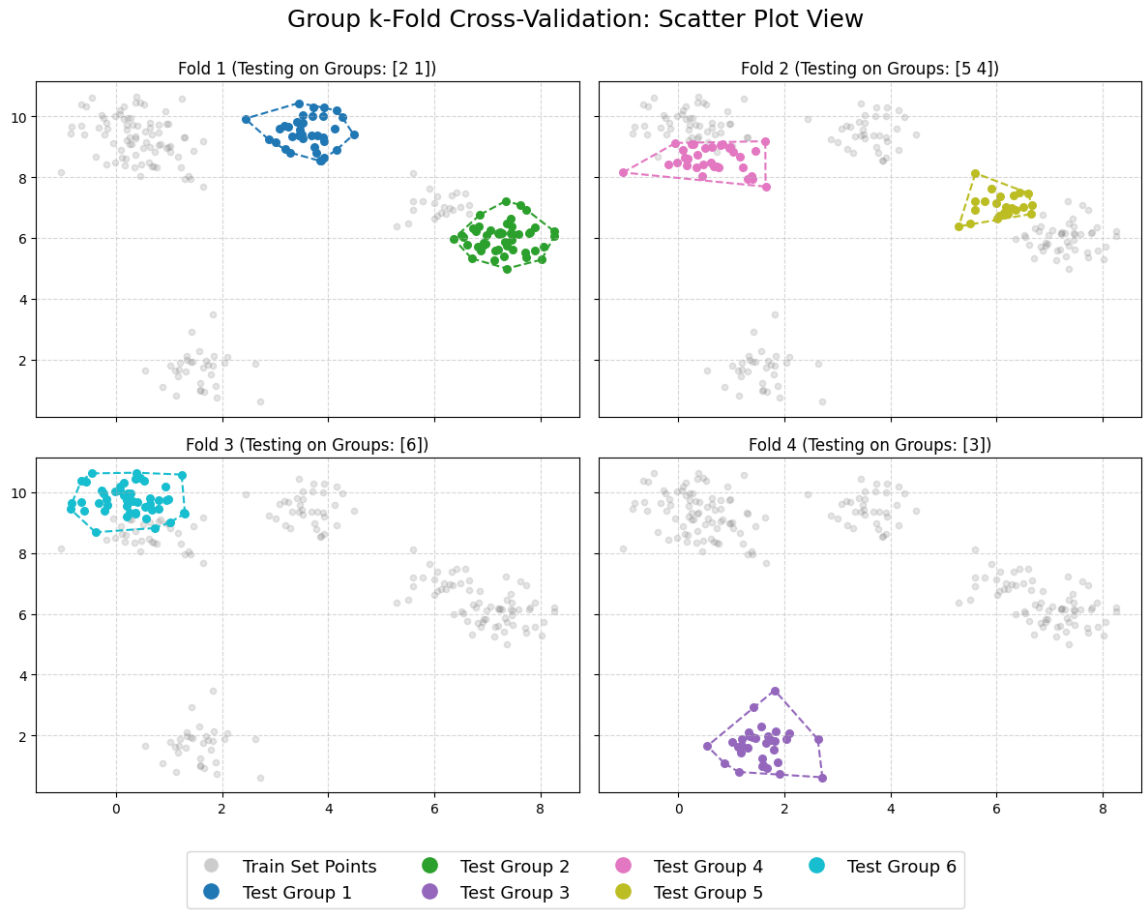


S2.2k: Nested Time-Series Cross-Validation (Panel b). Detailed view showing the inner loop structure within a single outer fold, with temporal ordering preserved throughout.

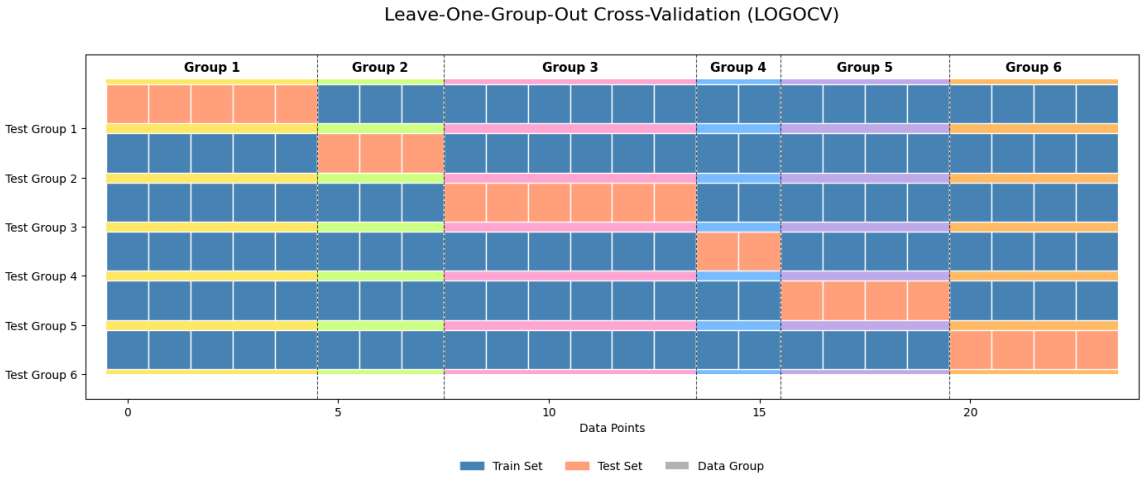
S2.3 Group-Aware Cross-Validation Methods



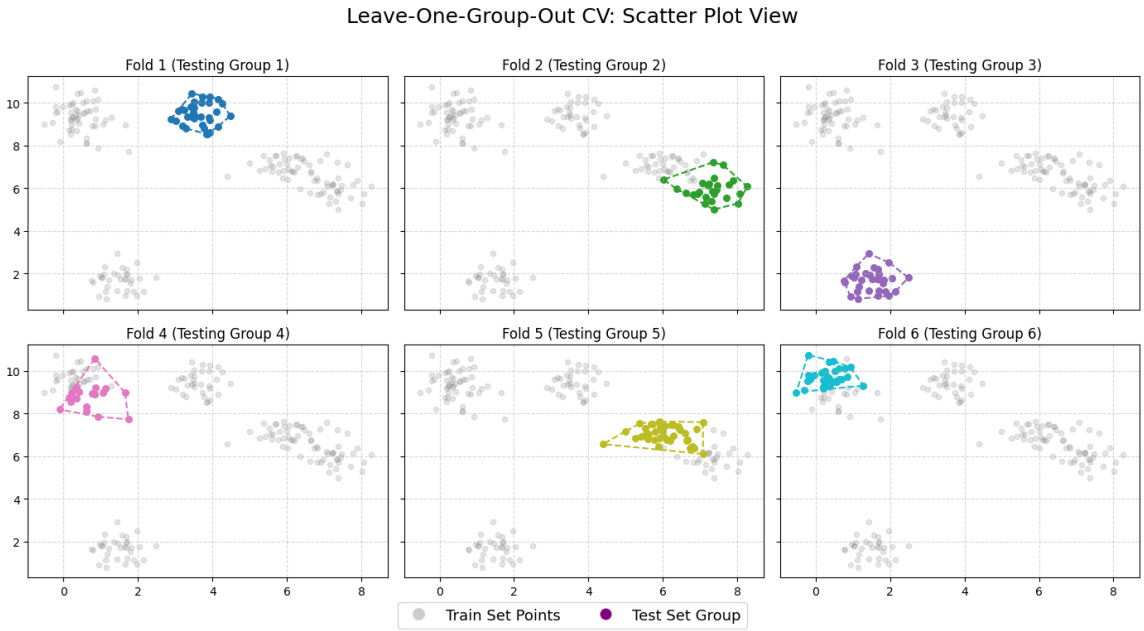
S2.3a: Group k-Fold Cross-Validation (Panel a). Extends k-fold to ensure all samples from the same group (e.g., patient) appear in the same fold. Coloured bands represent different patient groups kept intact.



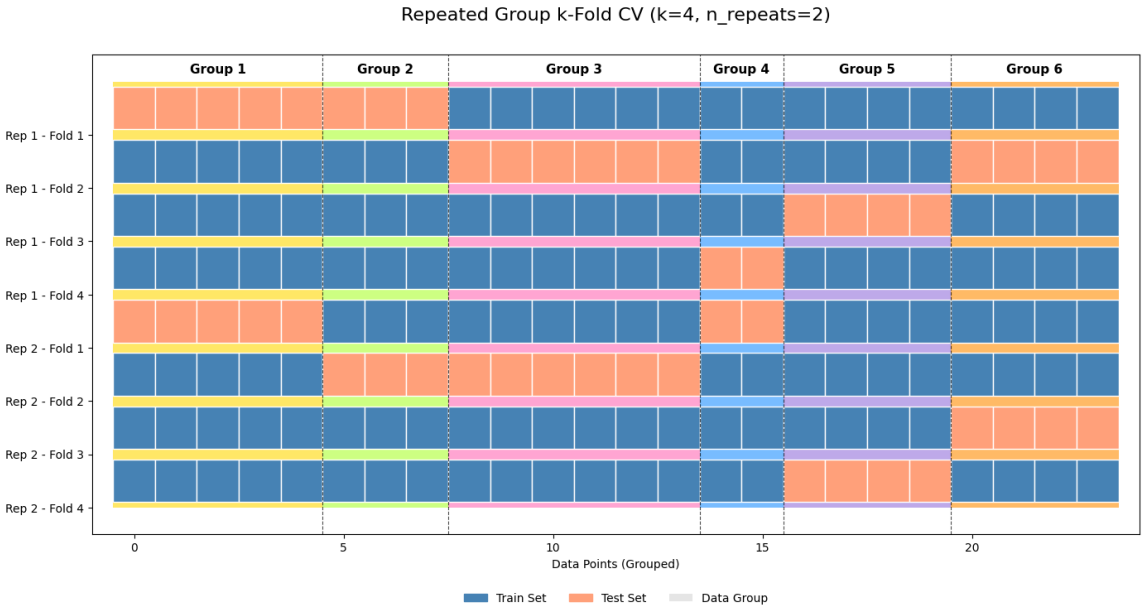
S2.3b: Group k-Fold Cross-Validation (Panel b). Additional view showing how group integrity is maintained across all folds, with no patient appearing in both training and test sets simultaneously.



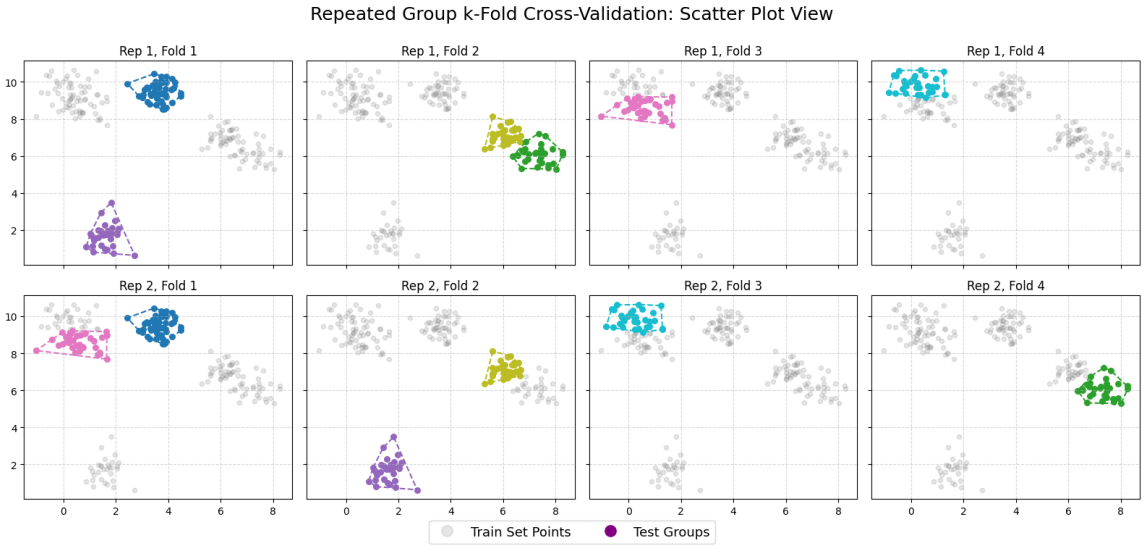
S2.3c: Leave-One-Group-Out Cross-Validation (~~LOGOCV~~LOGO) (Panel a). Each group (e.g., patient, site) is used as the test set exactly once while all remaining groups form the training set.



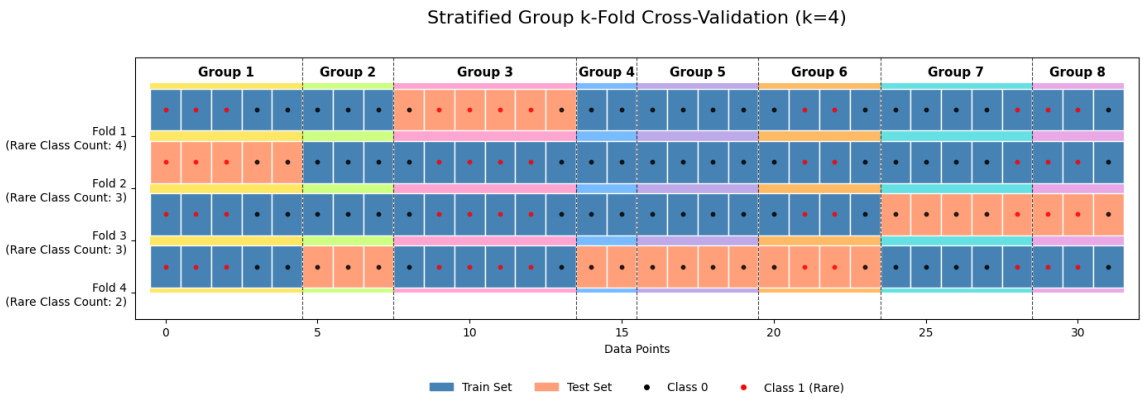
S2.3d: Leave-One-Group-Out Cross-Validation (~~LOGOCV~~LOGO) (Panel b). Extended view demonstrating generalization evaluation to completely new patients or clinical sites.



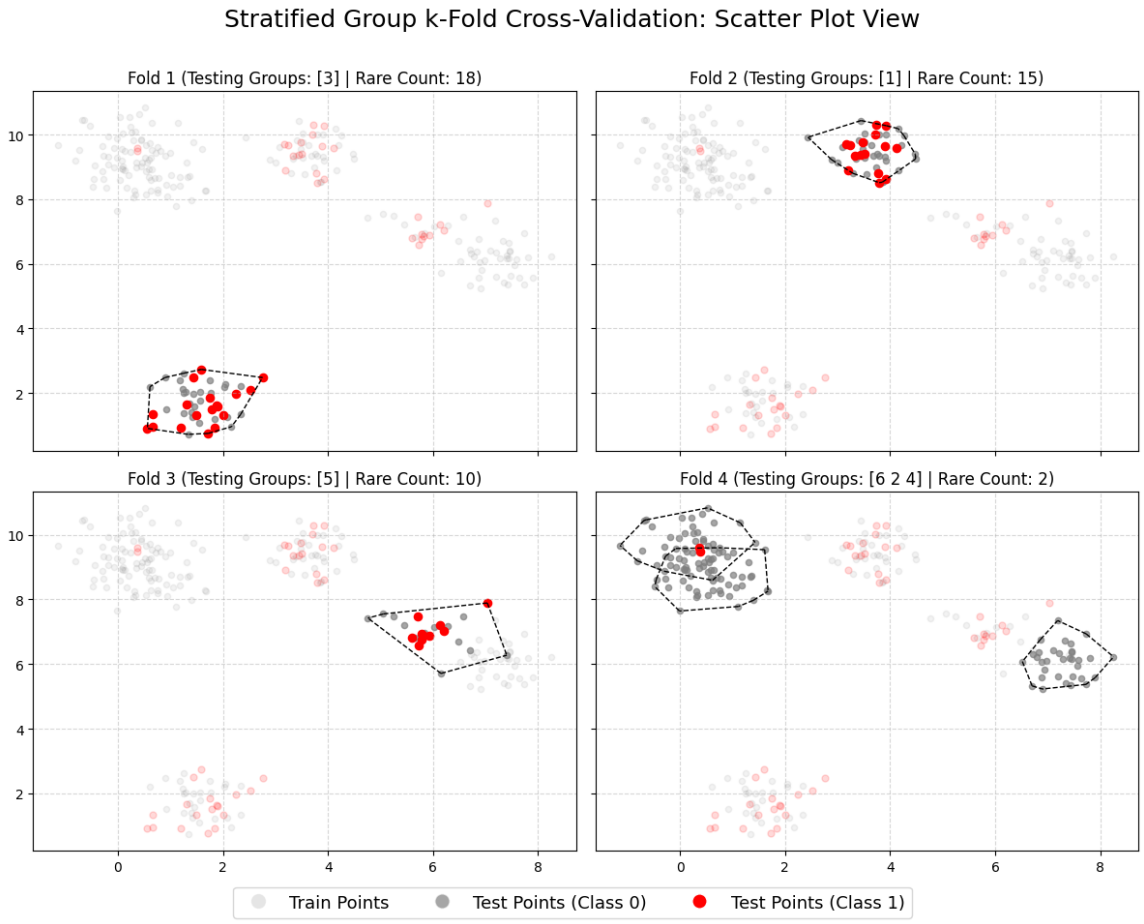
S2.3e: Repeated Group k-Fold Cross-Validation (Panel a). GroupKFold is performed multiple times with different random group-to-fold assignments, reducing variance while maintaining group integrity.



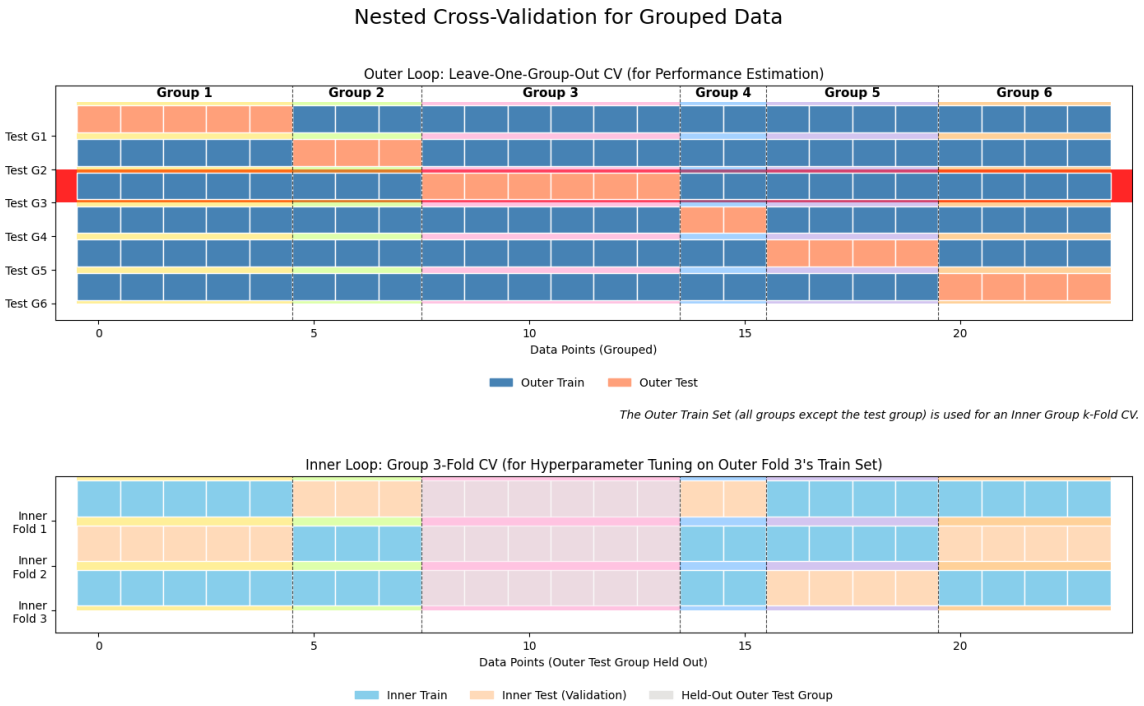
S2.3f: Repeated Group k-Fold Cross-Validation (Panel b). Second repetition showing different group-to-fold assignments, illustrating variance reduction through repeated splits.



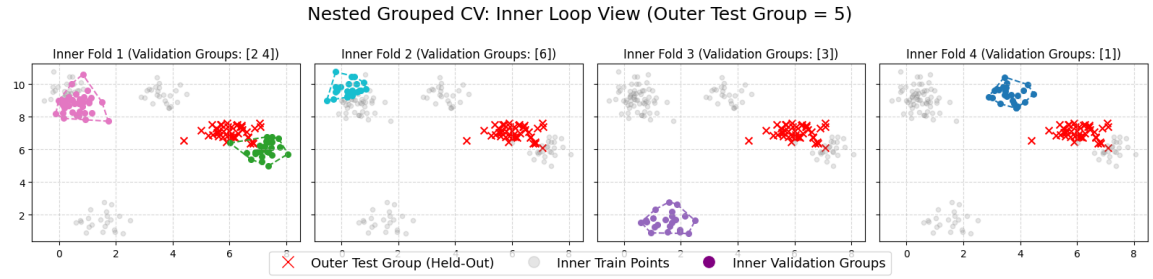
S2.3g: Stratified Group k-Fold Cross-Validation (Panel a). Combines group constraints with stratification, attempting to preserve class distribution while keeping groups intact across folds.



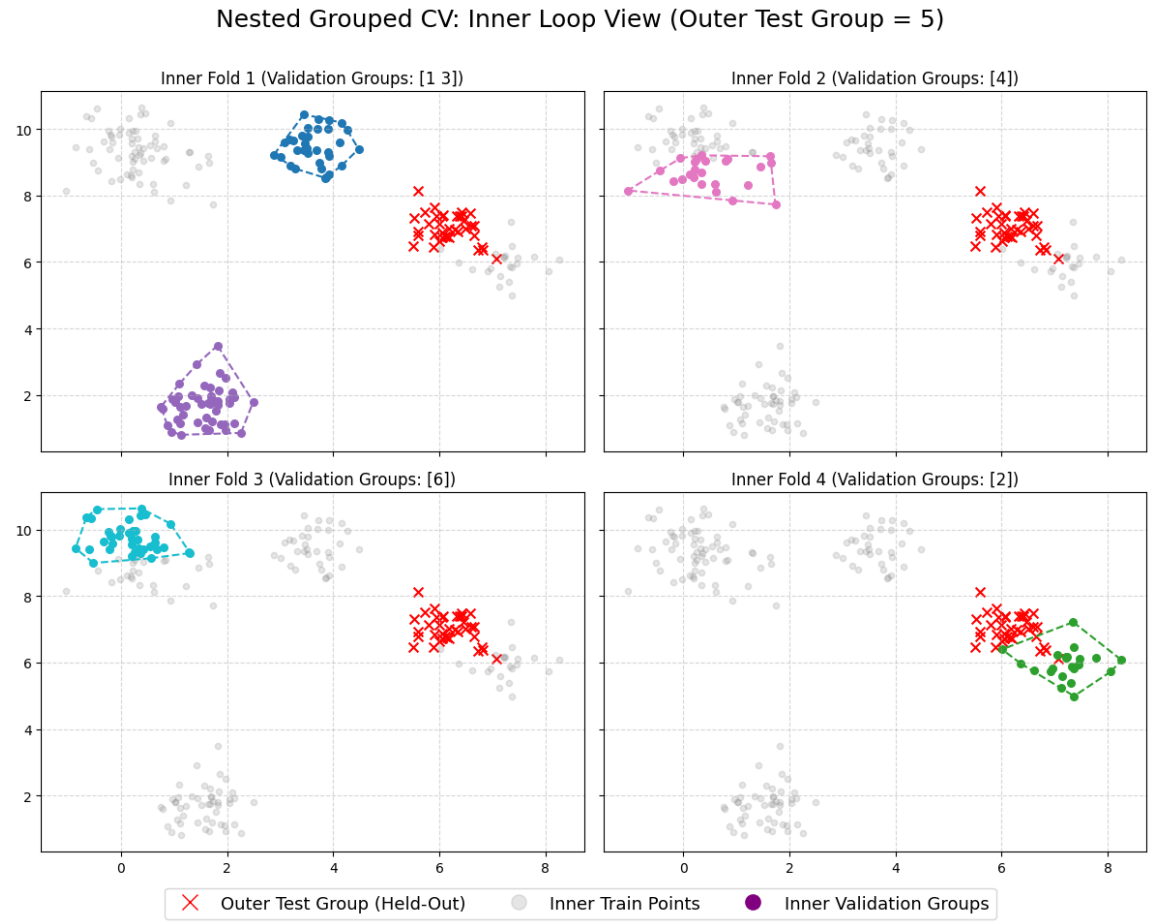
S2.3h: Stratified Group k-Fold Cross-Validation (Panel b). Extended view showing how class proportions are approximately maintained within each fold despite the group constraint.



S2.3i: Nested Cross-Validation for Grouped Data (Panel a). Combines nested CV with group constraints in both inner and outer loops, preventing both tuning leakage and group leakage simultaneously.

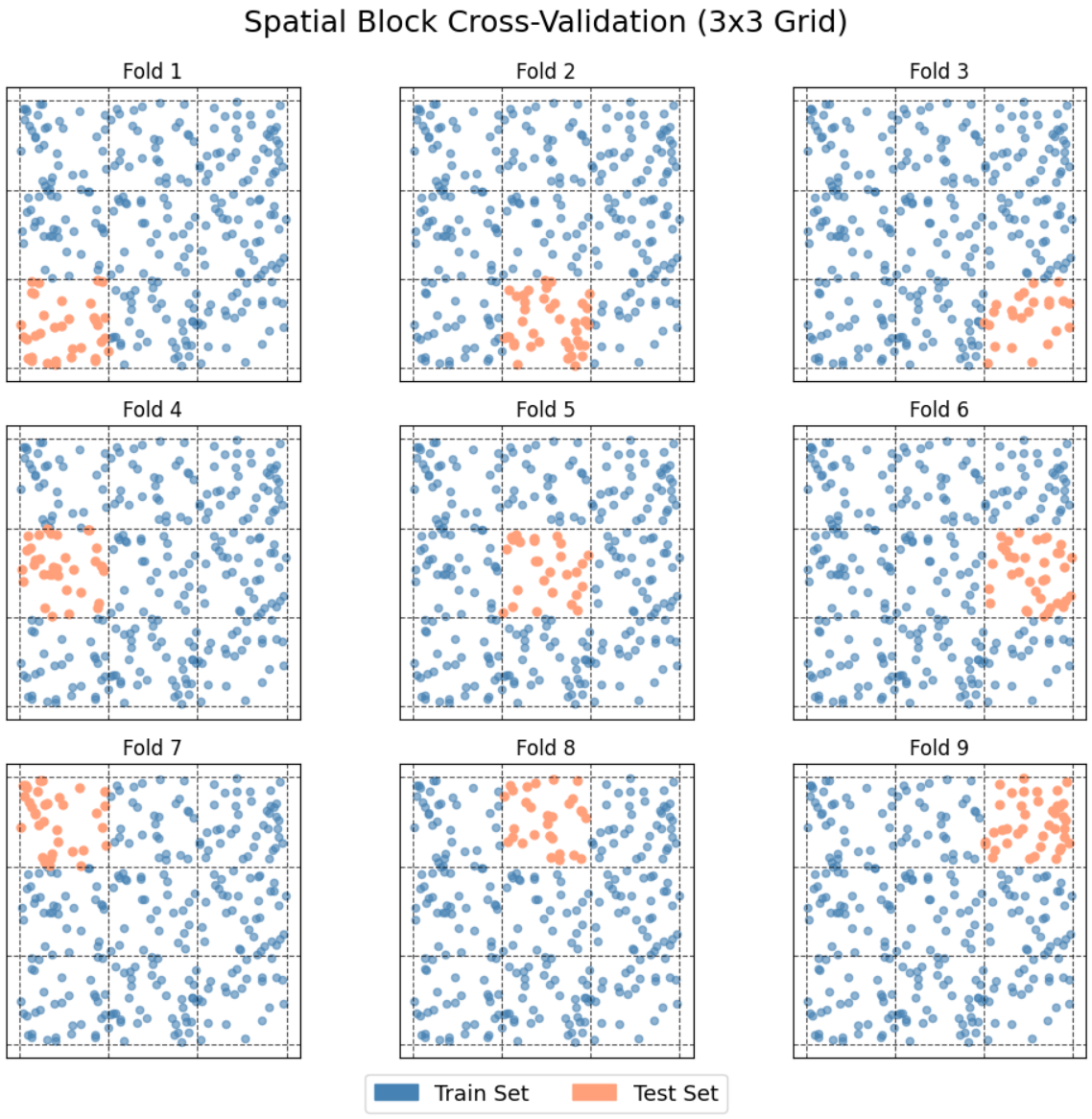


S2.3j: Nested Cross-Validation for Grouped Data (Panel b). Inner loop detail showing hyperparameter tuning performed with group-aware splitting within the outer training set.



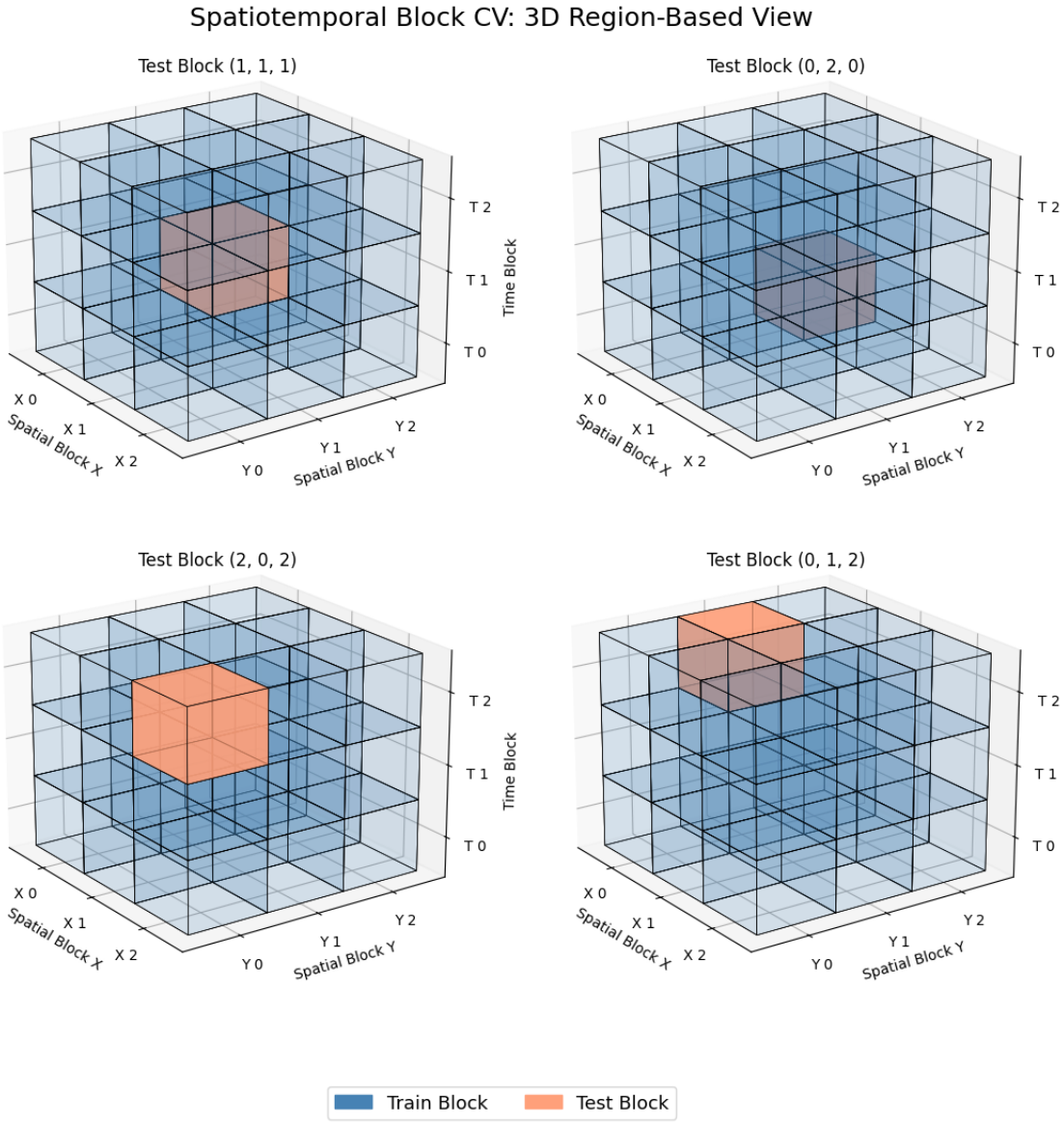
S2.3k: Nested Cross-Validation for Grouped Data (Panel c). Summary view of the complete two-loop grouped nested CV structure.

S2.4 Spatial Cross-Validation Methods

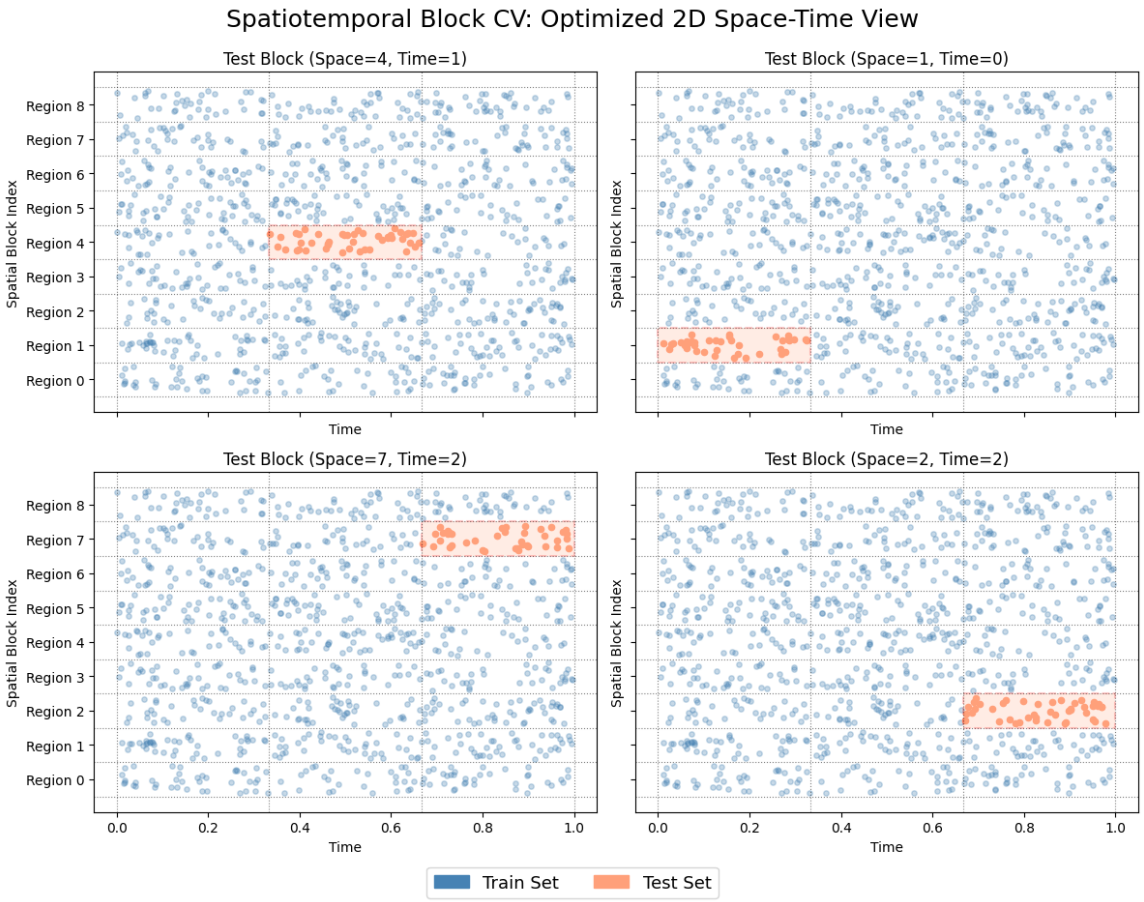


S2.4a: Spatial Block Cross-Validation (Panel a). Geographic space is divided into non-overlapping rectangular or hexagonal blocks. Block size should exceed the spatial autocorrelation range to prevent leakage.

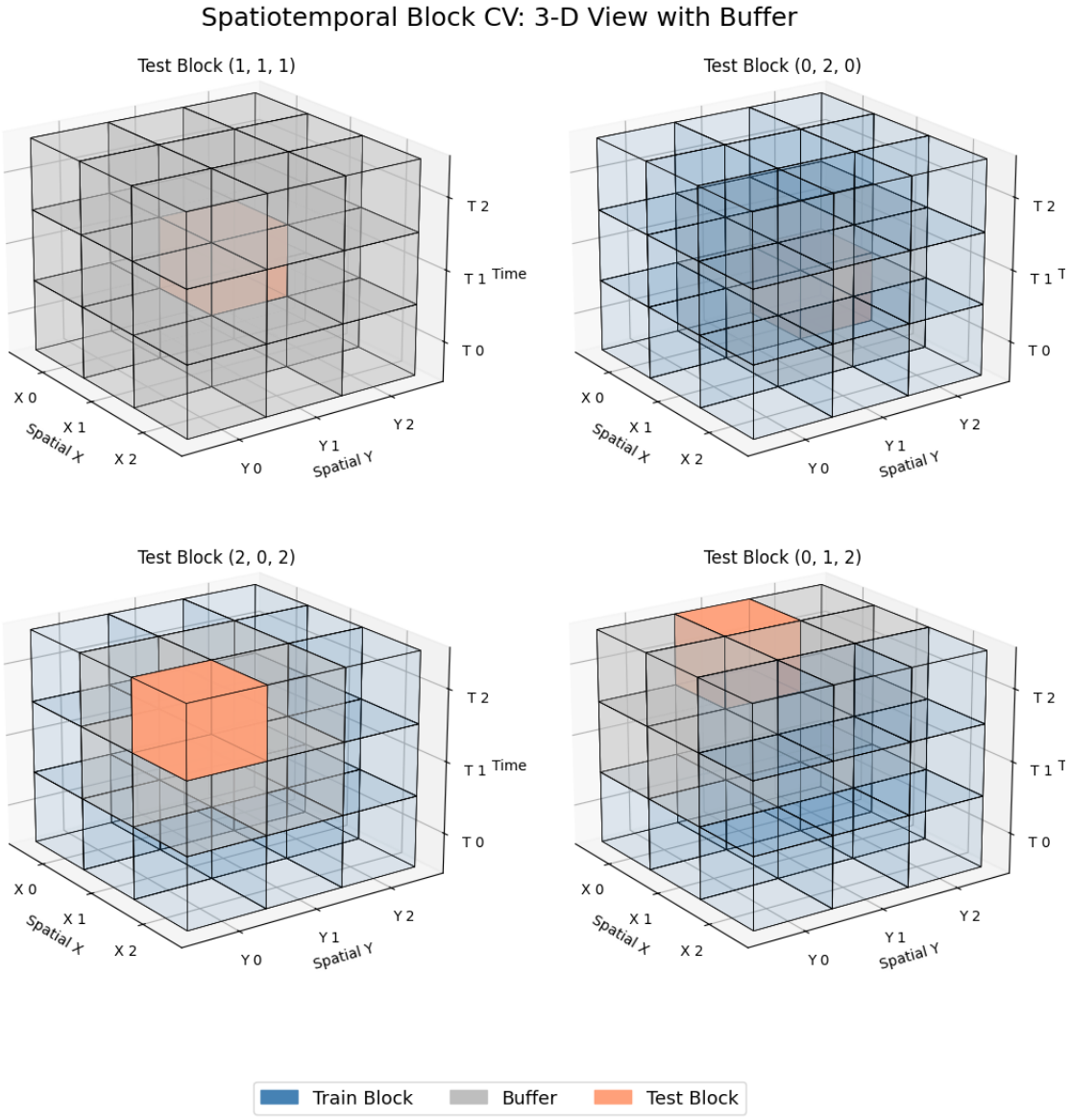
S2.4b: Buffered Spatial Cross-Validation. Extends spatial blocking by adding buffer zones around test blocks. Samples within the buffer distance are excluded from training, mitigating leakage from local autocorrelation.



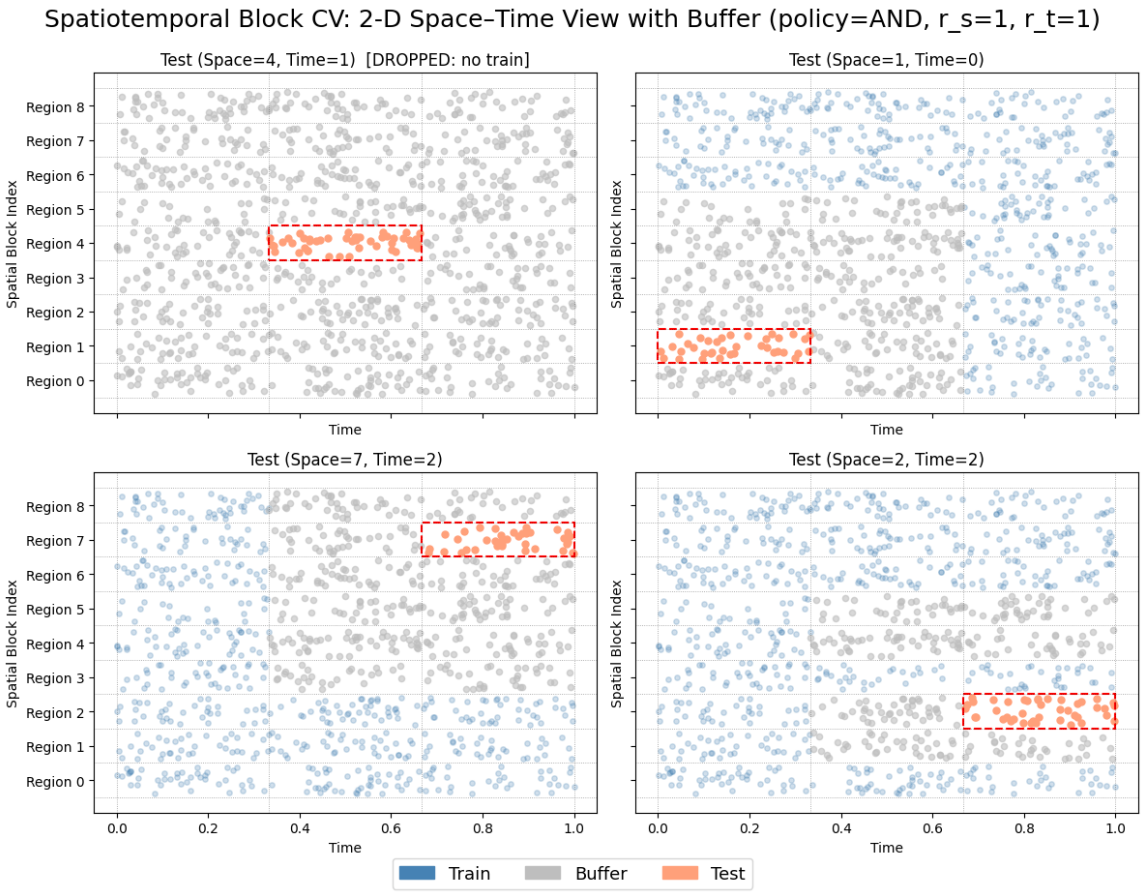
S2.4c: Spatiotemporal Block Cross-Validation — Without Buffer Exclusion. Combines spatial blocking with temporal blocking for data exhibiting both spatial and temporal structure.



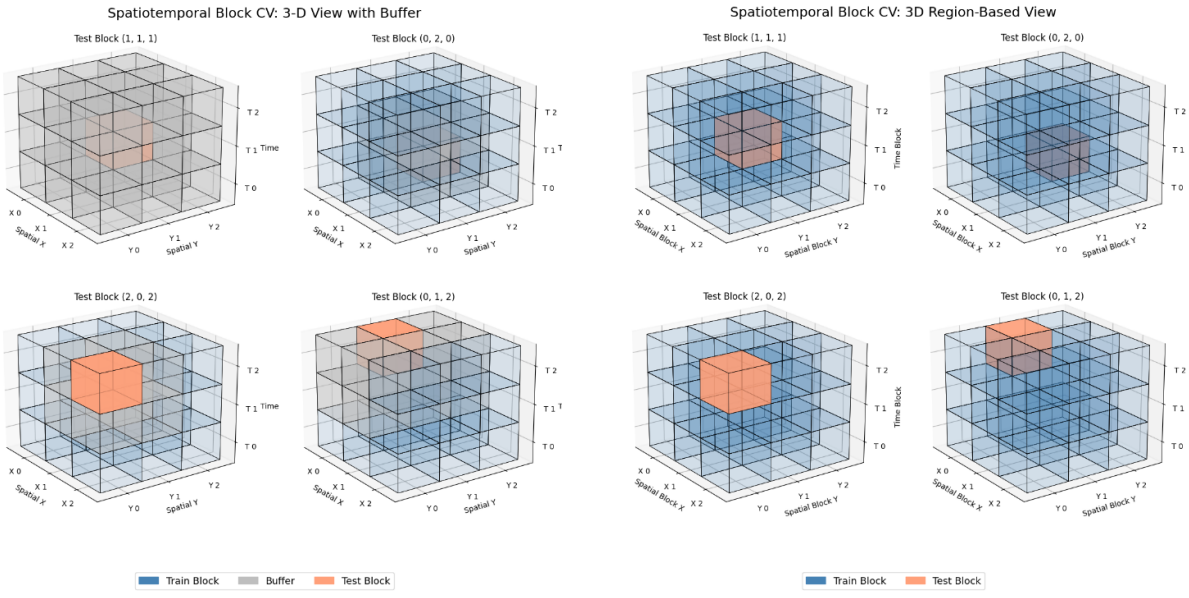
S2.4d: Spatiotemporal Block Cross-Validation — Detailed View. Shows spatial and temporal fold assignments overlaid, demonstrating how blocks are defined jointly in space and time.



S2.4e: Spatiotemporal Block Cross-Validation — With Buffer Exclusion (Panel a). Buffer zones in both spatial and temporal dimensions exclude correlated observations from the training set.



S2.4f: Spatiotemporal Block Cross-Validation — With Buffer Exclusion (Panel b). Alternative view showing the effect of buffer radius parameters r_s and r_t on training set composition.



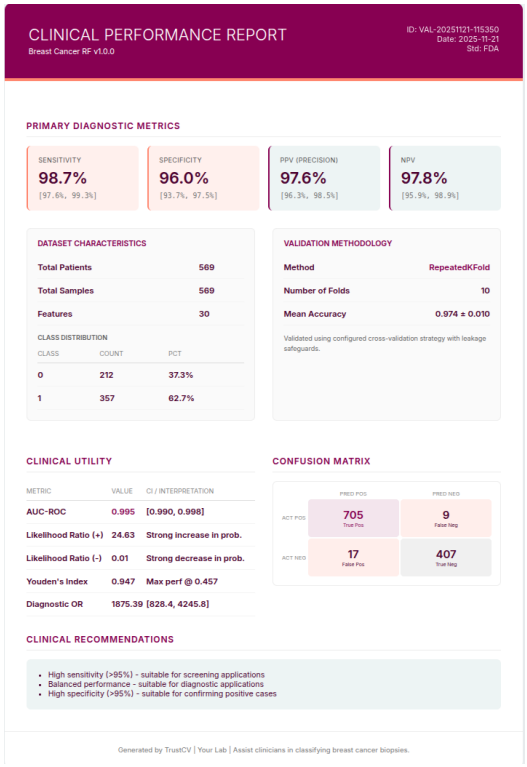
S2.4g: Spatiotemporal Block Cross-Validation — Side-by-Side Comparison. Left: without buffer exclusion. Right: with buffer exclusion. The buffer approach produces more conservative but less biased performance estimates by removing spatially and temporally correlated observations from the training set.

2.3 Supplementary Figure S4: `trustcv` Output Examples

The following two panels illustrate representative outputs produced by the **`trustcv`** toolkit on the Wisconsin Breast Cancer dataset. Panel (a) shows the structured plain-text report generated by `ClinicalMetrics.format_report()`, which is suitable for programmatic inspection and logging. Panel (b) shows the HTML clinical performance report generated by **`TrustCVValidator`**, which presents the same quantities in a structured layout intended for clinical documentation workflows with FDA GMLP–relevant fields. Both outputs were produced by the minimal example provided in Supplementary Code S6 and are included to give readers a concrete reference for the toolkit's reporting capabilities.

```
In [4]: import numpy as np
        from trustcv.metrics import ClinicalMetrics

        y_true = [1, 0, 1, 1, 0, 0, 1, 0]
        y_pred = [1, 0, 1, 0, 0, 1, 1, 0]

        cm = ClinicalMetrics()  # add confidence_level=0.95 if you want CIs
        #cm = ClinicalMetrics(confidence_level=0.95)
        result = cm.calculate_all(y_true, y_pred, None)

        #clin_metrics = ClinicalMetrics(confidence_level=0.95)
        print(cm.format_report(result))

============================================================
CLINICAL PERFORMANCE METRICS REPORT
============================================================

PRIMARY DIAGNOSTIC METRICS
------------------------------
Sensitivity:  75.0% [30.1%, 95.4%]
Specificity:  75.0% [30.1%, 95.4%]
PPV:          75.0% [30.1%, 95.4%]
NPV:          75.0% [30.1%, 95.4%]

PERFORMANCE METRICS
------------------------------
Accuracy:         75.0%
Accuracy CI:      [40.9%, 92.9%]
F1 Score:         0.750

CLINICAL UTILITY
------------------------------
Youden's Index:     0.500
LR+:                3.00
LR-:                0.33
Diagnostic OR:      9.00
NNT:                n/a
NNS:                n/a
LR+ interpretation:  Small increase in post-test probability
LR- interpretation:  Small decrease in post-test probability
Diagnostic OR CI:   [0.4, 220.9]

CONFUSION MATRIX
------------------------------
True Positives:     3
True Negatives:     3
False Positives:    1
False Negatives:    1

CLINICAL SIGNIFICANCE
------------------------------
Screening Suitable:     No
Diagnostic Suitable:    No
Risk Stratification:    No

RECOMMENDATIONS
------------------------------
```

(a) Structured plain-text output of `ClinicalMetrics.format_report()` for a toy eight-sample example (4 positives, 4 negatives). Sections cover Primary Diagnostic Metrics with 95% bootstrap confidence intervals, Performance Metrics, Clinical Utility (Youden's Index, LR+, LR−, Diagnostic OR), a Confusion Matrix summary, and a Clinical Significance flag table indicating screening/diagnostic suitability. The small sample here yields wide confidence intervals; the same format applied to a full dataset is shown in Panel (b).



(b) HTML clinical performance report generated by **`TrustCVValidator`** for a Random Forest model validated on the Wisconsin Breast Cancer dataset (569 samples, 30 features) using Repeated k-Fold CV (10 folds). The report includes primary diagnostic metrics with 95% confidence intervals (Sensitivity 98.7%, Specificity 96.0%, PPV 97.6%, NPV 97.8%), mean accuracy (0.974 ± 0.010), AUC-ROC (0.995), a confusion matrix aggregated over all held-out folds, and automated clinical recommendations. Header fields reference the validation ID, date, and the applicable regulatory standard (FDA). This report format is generated by the `export_report()` method and is intended for archiving and regulatory documentation.

3 Supplementary Code

Software verification and reproducibility

Implementation correctness was evaluated using an automated unit-test suite covering splitters, leakage checks, end-to-end workflows, and framework integrations. Verification focused on invariants appropriate to each method family, including partition completeness, train–test disjointness, preservation of group exclusivity for grouped methods, maintenance of temporal ordering for temporal methods, and adherence to method-specific parameter behavior. The current test suite comprises 141 test functions across 11 modules and is executed under continuous integration. Example notebooks, annotated scripts, and parameter reference tables were developed alongside the software to support reproducibility and practical uptake.

3.1 Supplementary Code S1: Basic Patient-Grouped Validation

The complete, annotated script is available at:

https://github.com/ki-smile/trustcv/blob/main/examples/heart_disease_classification.py

3.2 Supplementary Code S2: Temporal Validation with Purging

The complete, annotated script is available at:

https://github.com/ki-smile/trustcv/blob/main/examples/icu_patient_monitoring.py

3.3 Supplementary Code S3: Deep Learning with MONAI

The complete, annotated script is available at:

https://github.com/ki-smile/trustcv/blob/main/examples/monai_3d_imaging_cv.py

3.4 Supplementary Code S4: Unit Test Example for K-Fold Validation

The complete test module is available at:

https://github.com/ki-smile/trustcv/blob/main/tests/test_iid_methods.py

3.5 Supplementary Code S5: Unit Test Example for Leakage Detection

The complete test module is available at:

https://github.com/ki-smile/trustcv/blob/main/tests/test_data_leakage.py

References

[1] Saeb, S. *et al.* The need to approximate the use-case in clinical machine learning. *GigaScience* **6**, gix019 (2017).

[2] Little, M. A. *et al.* Using and understanding cross-validation strategies. Perspectives on Saeb et al. *GigaScience* **6**, gix020 (2017).

[3] Kaufman, S. *et al.* Leakage in data mining: formulation, detection, and avoidance. *ACM Trans Knowl Discov Data* **6**, 1–21 (2012).

[4] Bergmeir, C., Hyndman, R. J. & Koo, B. A note on the validity of cross-validation for evaluating autoregressive time series prediction. *Comput Stat Data Anal* **120**, 70–83 (2018).

[5] Roberts, D. R. *et al.* Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure. *Ecography* **40**, 913–929 (2017).

[6] Ploton, P. *et al.* Spatial validation reveals poor predictive performance of large-scale ecological mapping models. *Nat Commun* **11**, 4540 (2020).

[7] Kapoor, S. & Narayanan, A. Leakage and the reproducibility crisis in machine-learning-based science. *Patterns* **4**, 100804 (2023).