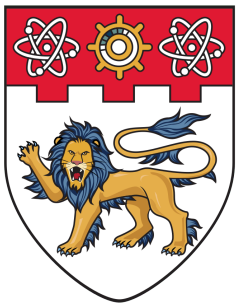


NANYANG TECHNOLOGICAL UNIVERSITY

SINGAPORE

NANYANG TECHNOLOGICAL UNIVERSITY

INVESTIGATING SPATIAL AND NON-SPATIAL CROSS-VALIDATION TECHNIQUES

Kan Huai Feng, Kai

School of Computer Science and Engineering

2024

NANYANG TECHNOLOGICAL UNIVERSITY

SC4079

INVESTIGATING SPATIAL AND NON-SPATIAL CROSS-VALIDATION TECHNIQUES

Submitted In Partial Fulfillment of the Requirements for the degree of Bachelor of Engineering in Computer Engineering of the Nanyang Technological University

by

Kan Huai Feng, Kai

College of Computing and Data Science
2024

Abstract

Accurately evaluating predictive models, especially when built for spatial data, remains a challenge due to the limitations of traditional cross-validation techniques. While spatial cross-validation techniques have been developed to address those challenges, there is still a lack of clear guidance on when a spatial or non-spatial technique would yield the most precise and dependable assessment of a model's performance.

This study investigates the effectiveness of the cross-validation method, spatial or non-spatial, in yielding accurate estimates by comparing three spatial techniques (Spatial K-Fold, Blocked, Buffered) and three non-spatial techniques (Random K-Fold, Bootstrap, Importance-Weighted). Using simulated landscapes, model performance is assessed across a range of metrics including Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Coefficient of Determination (R^2), and Bias. The findings emphasize the importance of considering spatial autocorrelation and covariate shifts, offering practical guidance on the selection of cross-validation techniques in spatial modelling contexts.

Acknowledgments

I would like to express my sincere gratitude to my project supervisor, **Dr Michele Nguyen**, for her continuous support, guidance, and encouragement throughout this project. Her invaluable insights, thoughtful critiques, and expertise have been critical to the successful completion of this work. I am deeply appreciative and grateful for the opportunity to learn under her mentorship.

Table of Contents

List of Tables

List of Figures

1. Introduction

Spatial Prediction (SP) is an important tool across various research fields as it allows researchers to estimate geographic variables, such as housing prices [1] and air quality [2], at unsampled locations based on existing geospatial data. A key aspect of SP is evaluating the model's performance in predicting geographic variables using Cross-Validation (CV). An accurate assessment of prediction models is essential to ensure that findings are not misinterpreted and that well-informed decisions can be made based on model outputs.

Conventional CV techniques, often referred to as non-spatial CV techniques, are widely adopted in model validation. However, these techniques typically disregard spatial autocorrelation; an inherent characteristic of spatial data described by Tobler's first law of geography. It states that "everything is related to everything else, but nearby things are more related than distant things" [3], [4]. Ignoring spatial autocorrelation can result in biased model assessments and overestimated predictive performance, particularly in fields where spatial relationships are important. For example, non-spatial CV techniques can underestimate the prediction errors in models that depend heavily on geographical variables, resulting in models appearing more accurate than they truly are.

To address these challenges, spatial CV techniques have been developed. These techniques aim to retain the spatial relationships and autocorrelation present in the data during validation, thus offering a potentially more reliable performance assessment. However, despite their theoretical advantages, their ability to evaluate models accurately and reliably compared to conventional CV techniques remains under-explored. Previous research has highlighted the limitations of non-spatial techniques, but a systematic comparison of both approaches in diverse spatial contexts is still lacking [5], [6].

This study aims to bridge this gap by systematically comparing the performance of commonly used spatial and non-spatial techniques in validating spatial predictive models. Using a landscape simulation model inspired by [7], the study applies three non-spatial CV techniques - Random K-fold CV, Bootstrap CV and Importance-Weighted CV, alongside three spatial CV techniques - Spatial K-fold CV, Blocked CV and Buffered CV. Spatial Prediction is particularly well suited for landscapes as it accounts for inherent spatial autocorrelation and geographic dependencies in environmental data, leading to more accurate predictions across diverse spatial regions. Unlike the simulation in

[7], which primarily focused on structured, predefined datasets, this study introduces more complex spatial patterns and covariate shift scenarios to assess model robustness under real-world conditions. This broader investigation allows for a more comprehensive evaluation of CV techniques and their ability to handle spatial heterogeneity and distribution shifts.

The study aims to answer the following questions:

1. How do spatial and non-spatial CV techniques compare in terms of accuracy metrics Root Mean Square Error (RMSE), Mean Absolute Error (MAE), Coefficient of Determination (R^2)?
2. What are the inherent biases and limitations of each technique when applied to spatial predictive models?
3. How do covariates and noise variables influence the performance and effectiveness of cross-validation in model evaluation?

By addressing these questions, this study will contribute to refining model validation practices in spatial analysis. The findings are expected to guide the most appropriate CV techniques for SP modelling, enabling researchers to avoid common pitfalls such as over-fitting or biased results. This study aims to enhance the accuracy of CV techniques in assessing SP models across various applications, from environmental monitoring to urban planning, and improve the overall design and interpretation of studies relying on SP models.

2. Literature Review

2.1 Introduction to Cross-Validation Techniques

CV is a technique used in SP modelling to assess how well a model generalises an independent dataset. The basic process involves splitting a dataset into the train and the test sets: the model is trained on the train set, and its performance is assessed on the test set.

Non-spatial CV techniques such as Random K-fold CV, Bootstrap CV and Importance-Weighted CV are primarily designed for datasets that adhere to the assumption of being independent and identically distributed (i.i.d.) [8], [9]. These techniques are widely adopted due to their simplicity and effectiveness in providing a robust estimate of a model's predictive accuracy [10].

Despite their effectiveness in many applications, non-spatial CV techniques face significant limitations when the dataset contains distinct properties, such as temporal or spatial dependencies. In such cases, the assumption of i.i.d. observations does not hold as autocorrelation is often present [7], [11]. Temporal dependencies introduce autocorrelation, where past observations influence future values, while spatial dependencies result in closer observations being more similar than those farther apart. When non-spatial CV techniques are applied to such datasets without considering these dependencies, the resulting model assessments can be biased and overly optimistic as they fail to account for the inherent autocorrelation [7], [11]. This results in an inflated perception of model performance, which may not generalise well to real-world applications.

As SP modelling involves increasingly complex datasets with inherent characteristics, addressing these limitations becomes crucial. This has led to the development of spatial CV techniques designed to better accommodate such dependencies seen in the data.

2.2 Challenges of Applying Non-Spatial CV to Spatial Data

Spatial data serves a unique challenge in model validation due to spatial autocorrelation - a phenomenon where observations located near each other in space are more likely to have similar values than those further apart [12]. Spatial autocorrelation is a testament to Tobler's first law of geography, which states that everything is related to everything else, but nearby things are more related than distant things [3], [13]. Spatial autocorrelation violates the assumption of independence that non-spatial CV techniques rely on, leading to inaccurate model evaluation [4]. Non-spatial CV techniques typically disregard these spatial relationships, leading to splits in the data that do not account for the spatial structure.

Another issue that could arise is covariate shift (CS), where the distribution of covariates (predictor variables) differs between the train and the test sets. This shift can occur in spatial datasets when different regions display distinct spatial patterns. This may cause the model to do well during cross-validation but perform poorly when applied to another geographical region with different spatial characteristics. Factors such as topography, climate, and land use can cause covariates to vary significantly between regions. For example, urban areas may display a stronger relationship between traffic and pollution when predicting air quality, while rural areas might show a greater influence from agricultural activities. Suppose the train set consists of data from urban regions. In that case, the model may struggle to generalise when predicting air quality in rural areas where the covariate relationships are fundamentally different [14].

When applying non-spatial CV techniques to spatial data, there is a significant risk of introducing spatial leakage—a situation where geographically proximate points are included in both the train and the test sets. For instance, when using Random K-fold CV on spatial data, it is common for nearby observations to appear in both the train and the test sets, allowing the model to inadvertently "see" the test data during training. This overlap can lead to inflated performance estimates, as the model may perform well on the test set due to the spatial similarity between the train and the test data points rather than its ability to generalise to new and unseen data [15]. Consequently, the model's cross-validation results may suggest high accuracy, but its true performance on independent datasets may be significantly lower.

This issue has been observed in numerous applications, including environmental mod-

elling, epidemiology, and remote sensing. In these fields, models trained on spatially autocorrelated data often show strong performance during validation but struggle to generalise when applied to regions or locations outside the sampled areas. The inability of non-spatial CV techniques to account for spatial dependency leads to over-fitting and reduces the accuracy of model assessments, especially in geographically diverse datasets [16].

2.3 Development and Application of Spatial Cross-Validation Techniques

To address the inherent limitations of non-spatial CV techniques—primarily their failure to account for spatial autocorrelation—spatial cross-validation techniques were introduced. Spatial CV aims to preserve the spatial relationships in the dataset, reducing the risk of spatial leakage. By incorporating spatial dependencies into the validation process, spatial CV techniques provide a more robust and accurate measure of model performance, particularly in spatially structured datasets.

Geographically Weighted Regression (GWR), a spatial modelling technique, benefits significantly from spatial CV techniques. GWR accounts for spatial heterogeneity by allowing local variations in the relationships between variables. [17] demonstrated that using spatial CV techniques in GWR can lead to more realistic assessments of model performance by respecting the underlying spatial structure of the data. This approach differs from non-spatial CV, which would otherwise ignore the spatial dependencies, leading to biased evaluations.

However, while spatial CV techniques help to mitigate overly optimistic assessment of models, they are not without limitations. In cases where the test regions exhibit spatial patterns drastically different from the training regions, spatial CV can lead to overly pessimistic performance estimates. A case study by [18] found that some models were penalized more than necessary for spatial dependencies with no significant improvement over non-spatial CV techniques in certain cases. This variability in results suggests that the effectiveness of spatial CV techniques may depend on the specific spatial characteristics of the dataset. While spatial CV techniques can reduce the over-fitting of a model, they could also underestimate a model's predictive ability if the test set is too dissimilar from the train set.

2.4 Current Limitations and Research Needs

Most existing studies on CV techniques focus on specific applications or datasets, such as environmental modelling, geostatistics, or urban planning, which limits the generalisability of their findings. While the theoretical advantages of spatial CV are well documented, there is a need for more systematic evaluations of spatial and non-spatial CV techniques under controlled scenarios. Their practical implementation and potential drawbacks require further investigation, particularly in comparison with non-spatial CV techniques. This study addresses these needs by conducting a controlled landscape simulation, allowing for a direct comparison between spatial and non-spatial CV techniques in multiple different scenarios. By evaluating a diverse range of CV techniques under standardised conditions, this study provides broader insights into their strengths, limitations, and applicability in different spatial contexts.

2.5 Non-Spatial Cross-Validation Techniques

2.5.1 Random K-fold Cross-Validation

Random K-Fold CV (RKFCV) splits the dataset into k folds of equal sizes. The model is then trained on $k-1$ of these folds and validated on the remaining fold. This process is repeated k times, with each repeat having a different test set such that each fold is used as the test set exactly once. The final estimate of the model's performance is then obtained by averaging the error across all iterations. The following is a step-by-step guide to how RKFCV works:

1. **Randomly Shuffle the Data**: The data is randomly shuffled to eliminate any inherent order that may influence the model's performance. This ensures that each fold is representative of the dataset.

2. **Splitting into k fold**: The shuffled dataset is divided into k folds. The value of k is typically chosen between 5 to 10, though it could vary depending on the size of the dataset. Each fold has approximately the same number of data points.

3. **Training and Testing**:

- In each iteration, one of the k folds is held out as a test set, while the remaining $k-1$ folds are included in the training set.
- The process is repeated k times such that each fold serves as the test set exactly once.

4. **Model Evaluation**: After k iterations, the model's performance is averaged across all folds to provide a more robust estimate.

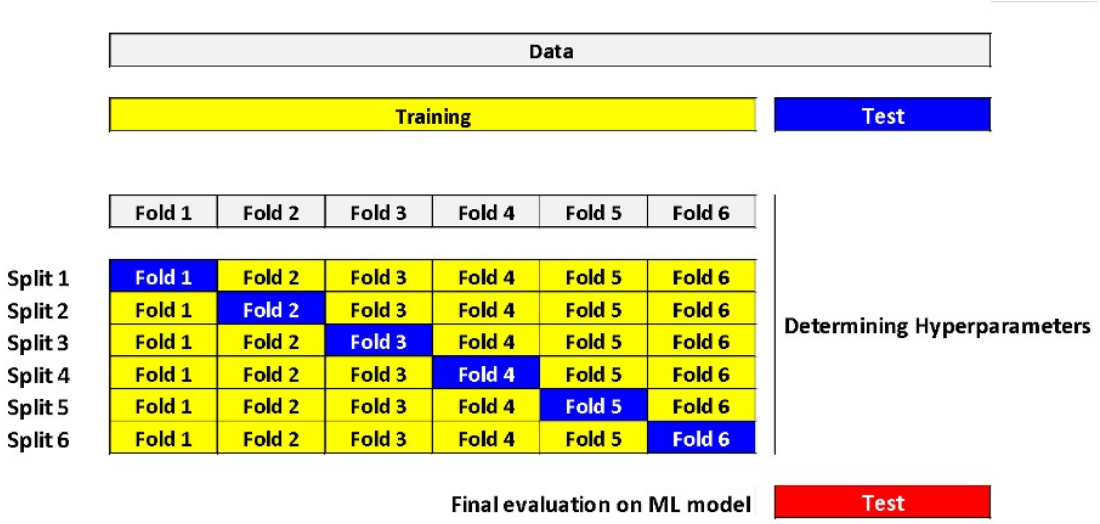


Figure 2.5.1: Random K-fold Cross-Validation [19]

RKFCV is versatile and widely adopted as it can provide a robust estimate of a model's generalization performance across different data splits [10]. However, in spatial contexts, where data points closer together are likely to be more similar, RKFCV can result in data leakage and overestimated performance.

2.5.2 Bootstrap Cross-Validation

Bootstrap CV (BootCV) creates multiple training and test sets by repeatedly sampling the dataset with replacements. Each new sample serves as a train set, while the remaining data points serve as the test set [20]. The following is a step-by-step guide to how BootCV works:

1. **Sampling with Replacement**: From the dataset of size y, a new train set of the same size y will be generated by randomly selecting data points with replacement. This means that some data points could be selected multiple times or not at all.

2. **Training and Testing**:

    - The model is trained on the newly created sample (train set) and tested on the remaining data points that were not selected (test set). Typically, about 63% of the original data is used for training, and the remaining 37% are used for testing.
    - This process is repeated multiple times, depending on the specified number of iterations set. Each time, a different train and test set will be generated.

3. **Performance Estimation**: After all iterations, the model's performance is averaged across all iterations to provide a more robust estimate.

Figure 2.5.2: Bootstrap Cross-Validation: Multiple resampled train sets are created from the original dataset, with the remaining data points serving as the test sets. This process is repeated across several iterations to estimate model performance [21].

BootCV provides a flexible way to estimate the model's performance and is often used to quantify the variability of model accuracy across different samples. As it uses sampling, BootCV can work well with smaller datasets where splitting data might lead to under-representative test sets. However, it can suffer from the same limitations of RKFCV, whereby when applied to spatial data, the random sampling process may overlap the train and the test sets, resulting in a less accurate performance estimate.

2.5.3 Importance Weighted Cross-Validation

Importance Weighted CV (IWCV) adjusts for differences in the covariates distribution of the train and the test datasets. IWCV assigns weights to data points in the train set based on their importance or similarity to the test set. These weights adjust for distribution mismatches between the train and the test sets and help produce more accurate model performance estimates under actual test conditions. The weights are derived from the ratio of the probability densities of the test and train data.

This study used the Relative Unconstrained Least-Squares Importance Fitting (RuLSIF) method to estimate the density ratio. RuLSIF was chosen because of its robustness and computational efficiency in estimating density ratio with minimal tuning of hyperparameters. Unlike other methods, RuLSIF minimizes Bregman divergence between probability densities, making it more stable and reliable in handling real-world datasets [22]. RuLSIF's ability to directly estimate the relative density ratio allows for better control of distribution shifts between the train and the test set [23].

The following is a step-by-step guide to how IWCV works:

1. **Data Splitting**: The dataset is divided into several folds. In each iteration, 1 fold is used as the test set, while the remaining folds are used to train the model.

2. **Assigning Importance Weights**: Each training data instance is assigned an importance weight, reflecting how likely it is to represent the test data. These weights are typically derived through density ratio estimation, where the ratio between the probability densities of the test data and the train data is computed for each instance. The formula is given as:

$$w(x) = \frac{P_{\text{test}}(x)}{P_{\text{train}}(x)} \tag{2.5.1}$$

   where $P_{\text{test}}(x)$ is the probability of observing instance x, and $P_{\text{train}}(x)$ is the probability of observing the same instance in the train set.

3. **Model Training with Weights**: The model is trained on the weighted train set, with each instance reweighed per its importance. Instances that are more representative of the test set are given a higher weight, ensuring the model places more emphasis on these instances during training.

4. **Model Evaluation**: The model is evaluated on the test fold after training. This

step is repeated across all folds, and the overall performance metric is averaged over all iterations to assess the model's generalization capability.

By incorporating importance weights through RuLSIF, IWCV effectively adjusts for distribution shifts between the train and the test sets. This adjustment is particularly useful when the train set may be biased or originates from a different domain than the test data. RuLSIF provides an accurate and efficient mechanism for estimating the density ratio, reducing potential mismatches in distribution that could lead to bias model performance assessment.

2.6 Spatial Cross-Validation Techniques

2.6.1 Spatial K-fold Cross-Validation

Spatial K-fold CV (SKFCV), unlike RKFCV, ensures that the k folds are split in a way that respects the spatial structure inherent in the dataset. The dataset is divided into spatially contiguous blocks or regions, with each serving as a fold. The folds are generated by clustering spatially proximate points together such that each fold represents a spatial region. Like RKFCV, the model is trained on $k-1$ folds and validated on the remaining fold. The process repeats itself till all folds have been used as the test set exactly once. The final estimate of the model's performance is then obtained by averaging the error across all iterations. The following is a step-by-step guide to how SKFCV works:

1. **Splitting the dataset**: The data points in the dataset is divided into k folds based on their spatial location. This is usually done by dividing the area into spatial blocks (grids) or clustering observations based on their geographical coordinates.
2. **Training and Testing**:
    - For each iteration, one of the folds is used as the test set, while the remaining $k-1$ folds are used as the train set.
    - The key difference between SKFCV and RKFCV is that the test set in SKFCV is spatially distinct from the train set, ensuring minimal spatial leakage.

- The process is repeated k times, such that each fold serves as the test set exactly once.

3. **Performance Estimation**: After k iterations, the model's performance is averaged across all folds to provide a more robust estimate.

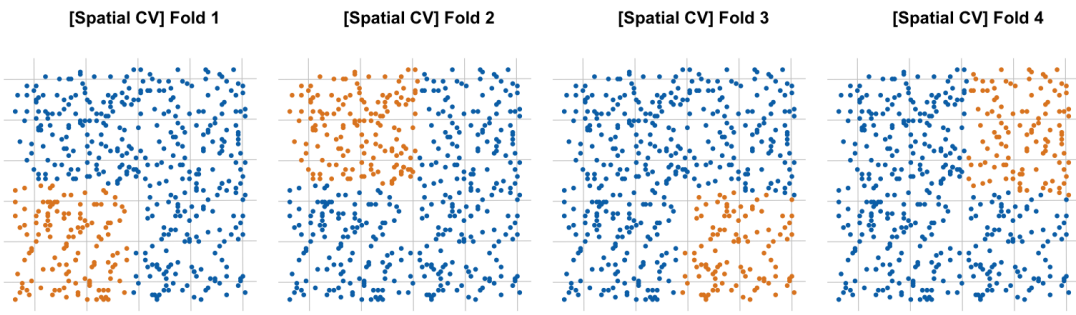


Figure 2.6.1: Visualisation of Spatial Cross-Validation Folds: Each subplot shows a different fold used for spatial cross-validation. The orange points represent the test set, while the blue points represent the train set. This technique ensures spatial separation between the train and the test sets to mitigate spatial autocorrelation effects and provides a more reliable model evaluation [24].

The spatial folds created by SKFCV ensure that the train and the test sets are geographically separated, reducing the risk of spatial leakage. SKFCV is exceptionally effective for datasets with spatial autocorrelation, as it prevents the model from leveraging spatial dependencies between nearby observations to inflate performance estimates.

2.6.2 Buffered Cross-Validation

Buffered CV (Buff) introduces a buffer zone around the test sets to mitigate spatial autocorrelation. If the data points are within a certain distance of the test set, they will be removed from the train and the test set. This ensures that the train and the test sets are spatially distant. The following is a step-by-step guide to how BuffCV works:

1. **Dividing data into folds**: Like other CV techniques, BuffCV splits the data into k folds. However, it adds a buffer zone to each test set to ensure that the train and the test sets are spatially separated.

2. **Buffer Zone**:

- A buffer zone is a region around the test fold where any data points in this region are excluded from both the train and the test sets.
- Typically, the buffer zone size is based on the spatial autocorrelation range. The buffer should be large enough to ensure that there is no strong spatial autocorrelation between the train and the test sets.

3. **Training and Testing**:
   - After defining the buffer, the remaining data outside the buffer and test set will be used for training.
   - The process is repeated k times such that each fold serves as the test set exactly once.
4. **Performance Estimation**: After k iterations, the model's performance is averaged across all folds to provide a more robust estimate.

Figure 2.6.2: Buffered Cross-Validation: Analysis (Train set) is used for model training, Assessment (Test set) is used for validation, and Buffer (Buffer Zone) ensures separation between the two to prevent spatial leakage.

By introducing the buffer zone, BufferCV helps prevent spatial autocorrelation from inflating the model's performance. However, it removes some data from the train and the test sets, which can be disadvantageous when dealing with small datasets. The size of the buffer zone must also be chosen carefully. A buffer that is too small would render it useless, while a buffer zone that is too big might leave insufficient data for training and testing.

2.6.3 Blocked Cross-Validation

Blocked CV (BCV) splits data into contiguous blocks to prevent spatial leakage between the train and the test sets. Unlike SKFCV, the dataset is divided based on spatial proximity. The resulting blocks will not be overlapping and hence ensure that there is no spatial overlap between the train and the test sets. The block size is usually determined based on the spatial structure (e.g., spatial autocorrelation range) to ensure minimal spatial dependence between blocks. The following is a step-by-step guide on how BCV works:

1. **Dividing data into blocks**:
    - The dataset is divided into several spatial blocks. These blocks are contiguous sections of the dataset and are grouped based on spatial proximity.
    - Block size is typically fixed based on criteria such as spatial autocorrelation range derived from variogram analysis. The aim is to define blocks large enough to ensure spatial independence between folds.
    - Each block is treated as a fold in BCV, with one block being set aside as the test set and the rest as the train set.
2. **Training and Testing**:
    - During each fold, one or more blocks are withheld as the test set, and the remaining ones become the train set.
    - The process is repeated such that each block becomes a test set exactly once.
3. **Model Evaluation**: After all iterations, the model's performance is averaged across all iterations to provide a more robust estimate.

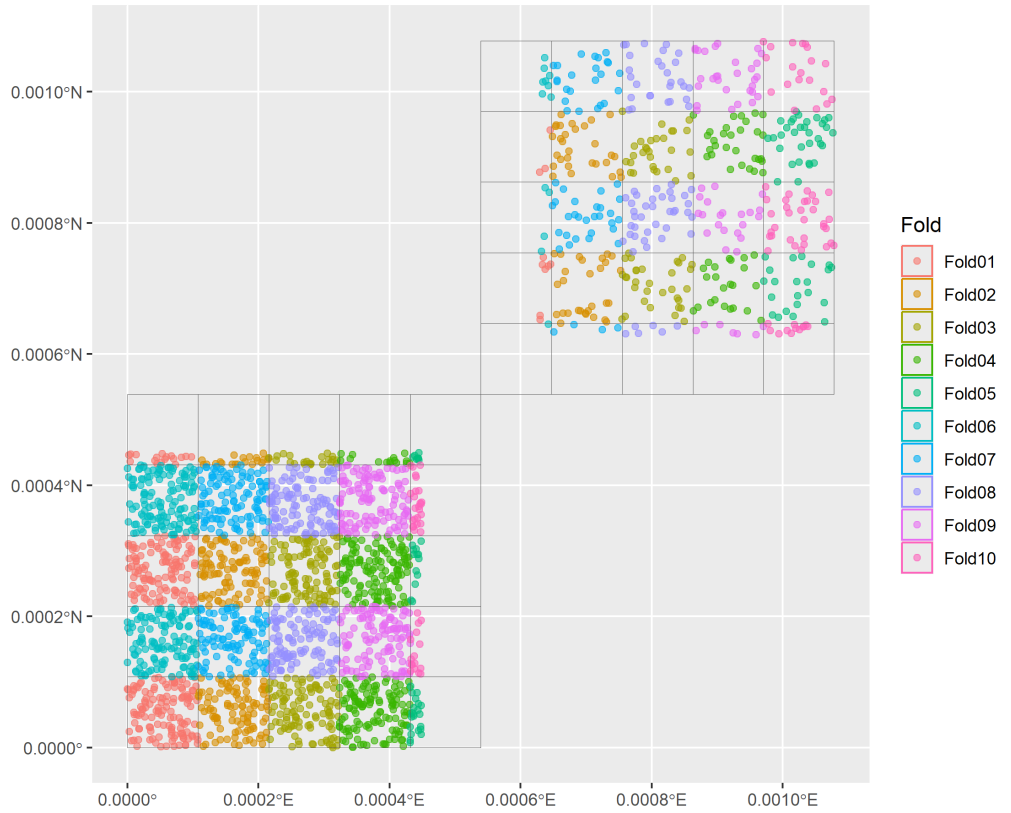


Figure 2.6.3: Blocked Cross-Validation: Each block (block of the same colours are of the same fold) represents a fold, ensuring spatial independence between the train and the test sets during model validation.

By creating blocks that respect the spatial autocorrelation in the dataset, BCV explicitly avoids any potential spatial dependence between the train and the test sets. providing a more accurate estimate of the model's performance. However, blocking reduces the number of data points available for training in each fold, as large blocks of data points will be used for the test set. This could affect model performance, especially in small datasets. The block size must also be considered carefully, as too large blocks can lead to insufficient train sets, and smaller blocks might result in spatial autocorrelation between the train and the test sets.

2.7 R Packages

2.7.1 Ranger

The Random Forest (RF) algorithm, designed by Breiman [25], is a popular ensemble method used for classification and regression. It creates multiple decision trees during training and outputs the mode of the classes or mean prediction of individual trees. RF are known for their robustness to over-fitting and ability to handle high-dimensional data, and capability to model both numerical and categorical variables effectively [26]. However, they can be computationally expensive.

The Ranger package in R provides a fast and memory-efficient implementation of RF. Unlike the base randomforest package in R, which is limited by its ability to scale according to data, Ranger is designed to manage high-dimensional datasets, which makes it ideal for handling tasks such as image processing and spatial modelling. Another huge benefit of using range is that Ranger supports parallel computing, which allows it to better process large datasets easily and efficiently by distributing the workload across multiple CPU cores [27].

2.7.2 Spatialsample

The Spatialsample package in R provides essential tools for users working with geographically structured data. This allows users to handle spatial structure data more effectively and simply [28]. Spatialsample simplifies the process of partitioning spatial data for model training and testing, ensuring that geographic characteristics are respected. By offering specialized resampling methods, Spatialsample helps users avoid biased model performance estimates, making it particularly useful for researchers in fields like ecology, environmental science, and geography.

A key advantage of Spatialsample is its seamless integration with R's tidymodels and caret frameworks, allowing users to incorporate spatial resampling without significant changes to their workflow. This package is vital for those working on predictive models in areas such as land use changes, environmental impact studies, or species distribution, ensuring that models are tested and validated appropriately for spatial contexts. Spatialsample is an irreplaceable tool for building more dependable and geographically robust models.

2.7.3 Caret

The Caret package, short for Classification and regression training, is designed to streamline the process of building predictive models. One of its core strengths is its ability to simplify pre-processing, resampling, and model evaluation, enabling users to focus on improving model performance.

Caret offers tools such as cross-validation, bootstrapping, and repeated k-fold validation, as well as support for hyperparameter tuning, which is critical in optimizing the performance of machine learning models. This also means that users have much control over the kind of models they can aim to create [29].

In addition, Caret supports external packages that provide specific cross-validation tools or model training algorithms not natively available in Caret itself. For instance, packages like Spatialsample and Ranger can be integrated into Caret to add spatial cross-validation techniques. This flexibility ensures that Caret's built-in methods do not constrain users, and users can extend its functionality by incorporating specialized techniques from other packages [28], [30].

Another key feature is the support for parallel processing, which speeds up the model training process by distributing the computational workload across multiple cores. Like Ranger, this functionality is particularly useful when working with large datasets or complex models.

3. Methods

3.1 Simulation Data Setup

An adapted version of the simulation approach by [7] was used to compare the spatial and non-spatial CV techniques. 50 landscapes were generated, representing independent simulations of the same data-generation process. Each landscape comprised a grid of 50 by 50 cells for 2,500 cells. For each landscape, 6 variables were generated using Gaussian variograms with varying ranges of spatial correlation and a target variable z was created based on a linear combination of the generated variables. Unlike the original setup by [7], which included more complex interactions between variables, the simulation this study adopts focuses on simpler spatial dependencies to facilitate the evaluation of predictive modelling techniques.

The spatial variables X_1, X_2, ..., X_6 were generated from a multivariate normal distribution by putting together a variogram to model spatial autocorrelation and a mean structure to represent underlying trends. For this simulation, the variables X_1, X_2 and X_3 were centered to have a mean of zero. This centering is catered so that there is no bias in the variables, allowing the model to focus solely on the spatial relationship and interactions among the variables.

The relationships between spatial data points were defined using the following components:

- The variogram defines how spatial correlation diminishes as distance increases. A Gaussian variogram model was used with partial sill, nugget, and range parameters.
- The mean structure captures the environmental trend across the landscape. In this study, however, X_1, X_2 and X_3 have a mean of zero, implying that there is no consistent trend for these variables across the landscape.

The spatial fields were simulated as follows:

$$\mathbf{X} \sim \mathcal{N}(0, \Sigma) \tag{3.1.1}$$

where:

- **X** represents the vector of spatial variables.
- 0 is the mean vector, indicating that X_1, X_2, and X_3 have a mean of zero.
- Σ is the covariance matrix derived from the variogram, which encodes the spatial relationships between data points.

Three parameters are used to describe the variograms:

1. **sills** (s): The total variance
2. **nugget** (nu): The variance at a lag distance of zero
3. **range** (r): The distance where two measurements are no longer considered correlated.

Two data points within the range r are considered spatially autocorrelated. The following equation was adopted for the Gaussian variogram used in this simulation.

$$\gamma(h) = (s - nu)\left(1 - \exp\left(-\frac{h^2}{r^2}\right)\right) + nu \tag{3.1.2}$$

where h is equal to the distance between any pair of data points sampled from the dataset.

This study generated a target variable z to represent the outcome that the trained model will predict. z was simulated as a linear combination of three environmental variables: X_1 (soil or topography), X_2 (precipitation) and X_3 (temperature). These variables are key factors that influence many spatial and environmental processes.

The relationship between z and its covariates was modelled using the following Equation 3.1.3:

$$z = X_1 + X_2 + X_3 + (X_2 * X_3) + \varepsilon \tag{3.1.3}$$

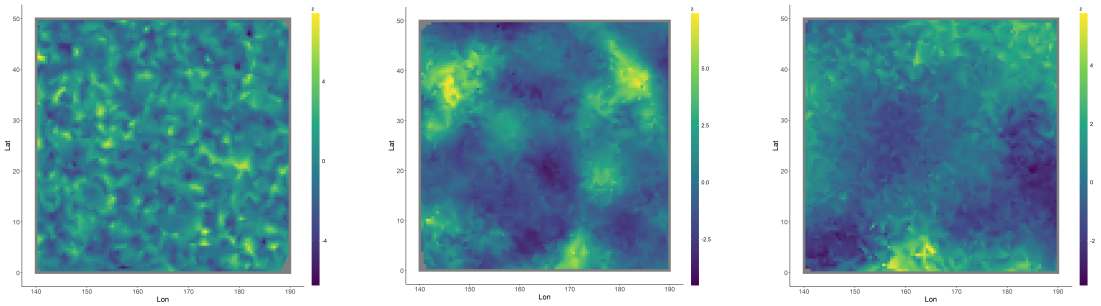
Here, ε represents a random noise term, modelled as Gaussian noise with a mean of 0 and a standard deviation of 0.1. This relatively small standard deviation was chosen to

control the level of noise, introducing variability without overpowering the influence of the covariates. Additionally, the value of 0.1 aligns with the scale of the environmental variables, ensuring that the noise remains proportionate to the magnitude of the data.

This setup realistically simulated an environmental outcome, such as crop yield, soil moisture, or habitat suitability.

Name	**Variable Definition**
X_1	Random Gaussian field with Gaussian covariance (partial sill = sill - nugget, sill = 1.0, nugget = 0.1, varying ranges)
X_2	Random Gaussian field with Gaussian covariance (partial sill = sill - nugget, sill = 1.0, nugget = 0.1, varying ranges)
X_3	Random Gaussian field with Gaussian covariance (partial sill = sill - nugget, sill = 1.0, nugget = 0.1, varying ranges)
X_4	Random Gaussian field with Gaussian covariance (partial sill = sill - nugget, sill = 1.0, nugget = 0.1, varying ranges)
X_5	Random Gaussian field with Gaussian covariance (partial sill = sill - nugget, sill = 1.0, nugget = 0.1, varying ranges)
X_6	Random Gaussian field with Gaussian covariance (partial sill = sill - nugget, sill = 1.0, nugget = 0.1, varying ranges)
z	$X_1 + X_2 + X_3 + (X_2 * X_3) + \varepsilon$

Table 3.1.1: Description of variables used in the simulation study.X_1, X_2, X_3 are covariates with mean = 0 and X_4, X_5, X_6 are noise variables with mean = 0



(a) No spatial autocorrelation $r = 1$

(b) Moderate spatial autocorrelation $r = 6$

(c) Strong spatial autocorrelation $r = 12$

Figure 3.1.1: Simulated target variable z with three different levels of spatial autocorrelation: (a) No spatial autocorrelation $r = 1$ shows a random distribution with no visible spatial patterns. (b) Moderate spatial autocorrelation $r = 6$ begins to show clustered regions, indicating some degree of spatial correlation. (c) Strong spatial autocorrelation $r = 12$ exhibits well-defined clusters, where nearby locations have highly similar values.

To further understand how the variables would affect the estimate of the model's performance, variables were added to the model iteratively. The model's performance was measured in each iteration to observe the effects of adding additional variables. The process unfolded in the following way:

- 1st Iteration: Trained using variables X_1
- 2nd Iteration: Trained using variables X_1 and X_2
- 3rd Iteration: Trained using variables X_1, X_2 and X_3
- 4th Iteration: Trained using variables X_1, X_2, X_3 and X_4
- 5th Iteration: Trained using variables X_1, X_2, X_3, X_4 and X_5
- 6th Iteration: Trained using all variables X_1, X_2, X_3, X_4, X_5 and X_6

Four distinct scenarios were created to evaluate the performance of the models under different conditions. These scenarios differed in how the train and the test sets were geographically distributed. The specifics of each scenario are detailed below:

- **Spatially Dependent (SD)**: All data points are within r, meaning that both the train and the test data points are sampled from the same spatial area such that there is spatial autocorrelation between the train and the test sets.

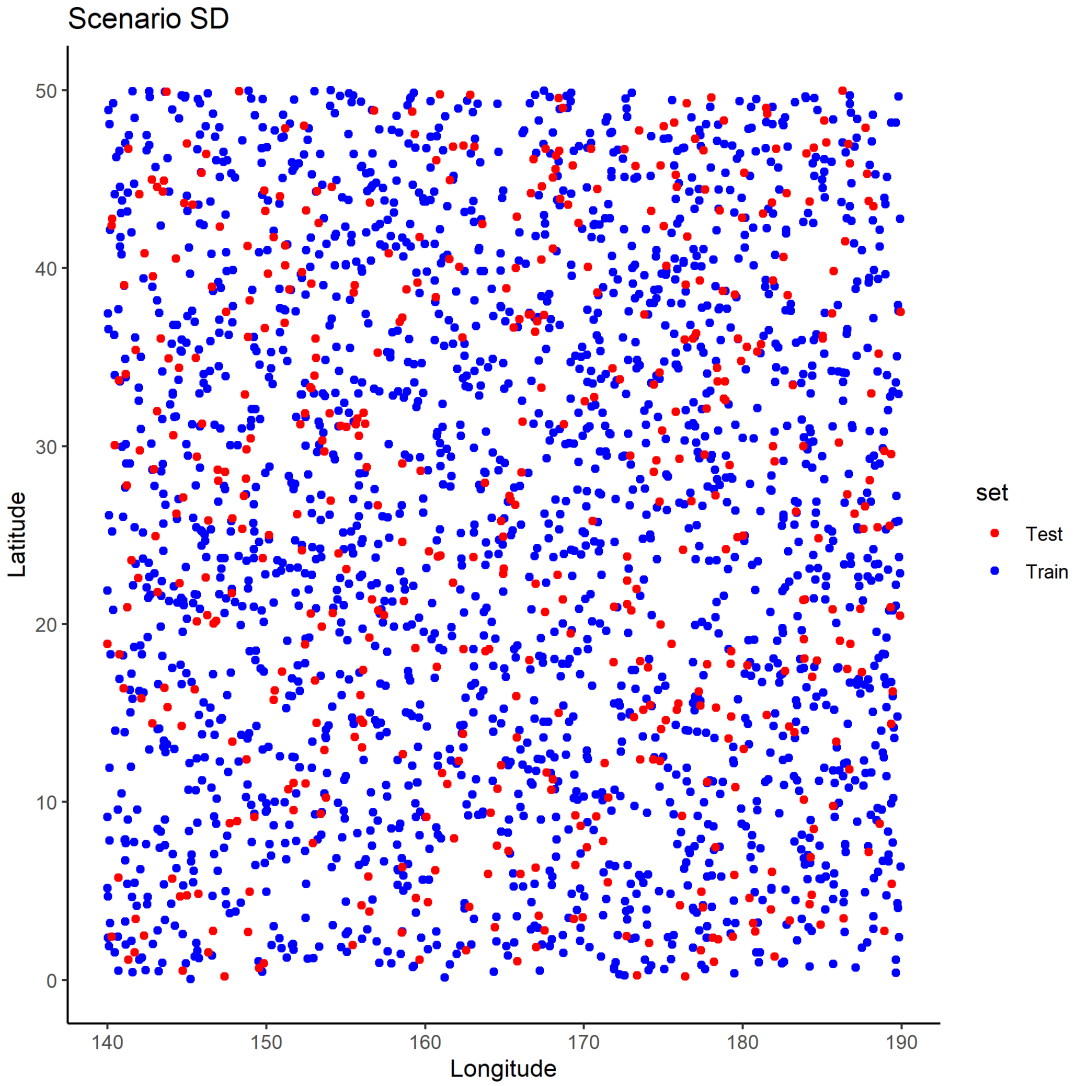


Figure 3.1.2: An example of Scenario SD, where $r = 6$.

- **Spatially Dependent + Covariate Shift (SDCS)**: The train and the test sets are split beforehand such that the nearest distance between any train and test data point is $r - 1$ (if $r = 1$, distance is 1 as well), ensuring spatial dependence. CS between the train and the test sets was tested using the Kolmogorov-Smirnov (KS) test with Bonferroni correction. Suppose the KS test does not reject the null hypothesis (indicating no CS); CS will be induced manually to simulate differences in the distribution between the train and the test sets. Details on the procedure to check and induce CS can be found in Section 3.2.

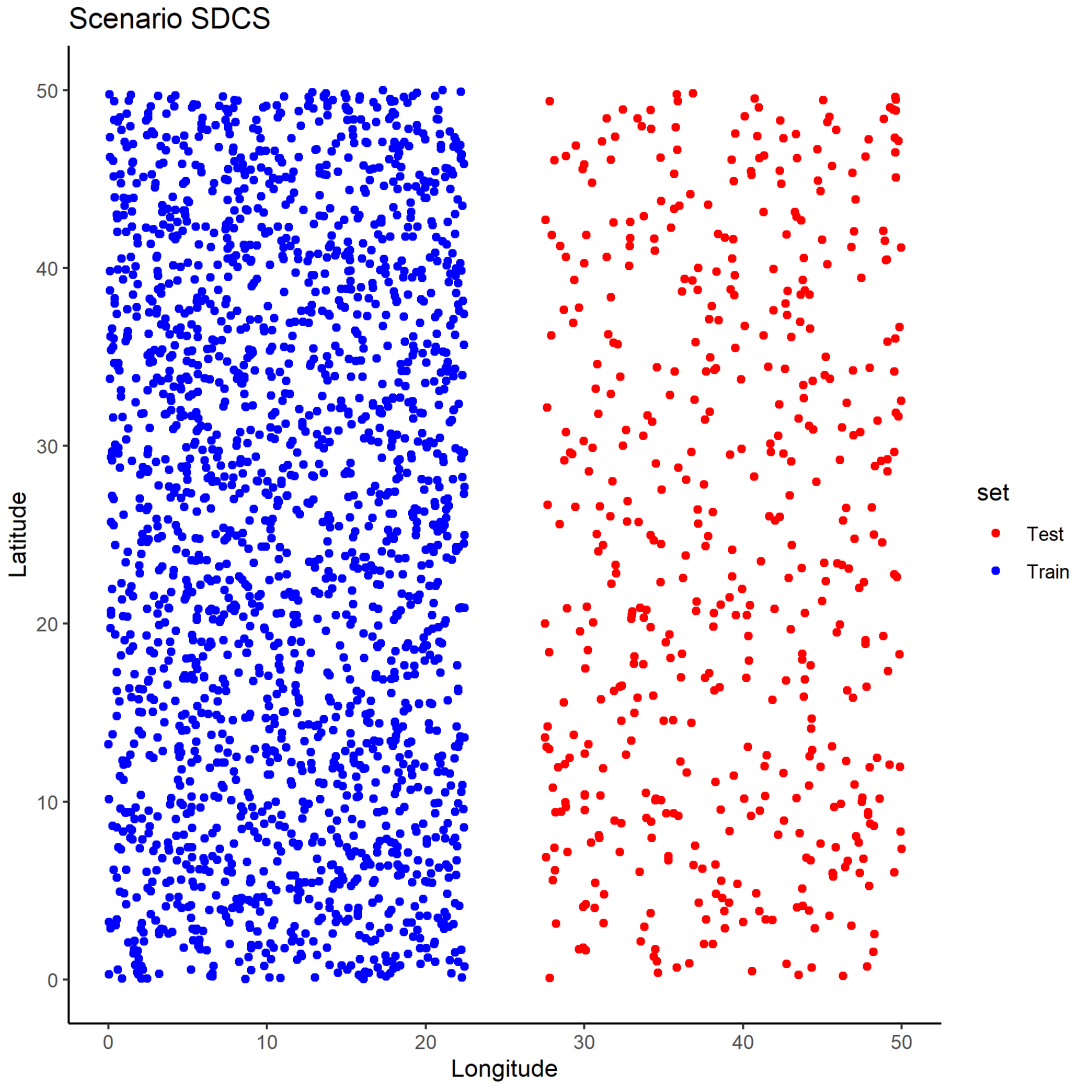


Figure 3.1.3: An example of SDCS is where the train and the test data points are geographically separated by $r-1$ distance. In this figure, $r=6$ and the nearest data point between the two sets is 5.

- **Spatially Independent (SI)**: The train and the test sets are generated in separate spatial regions, ensuring that the distance between the nearest data points in the train and the test sets is greater than r, thus making any spatial autocorrelation between the two sets negligible.

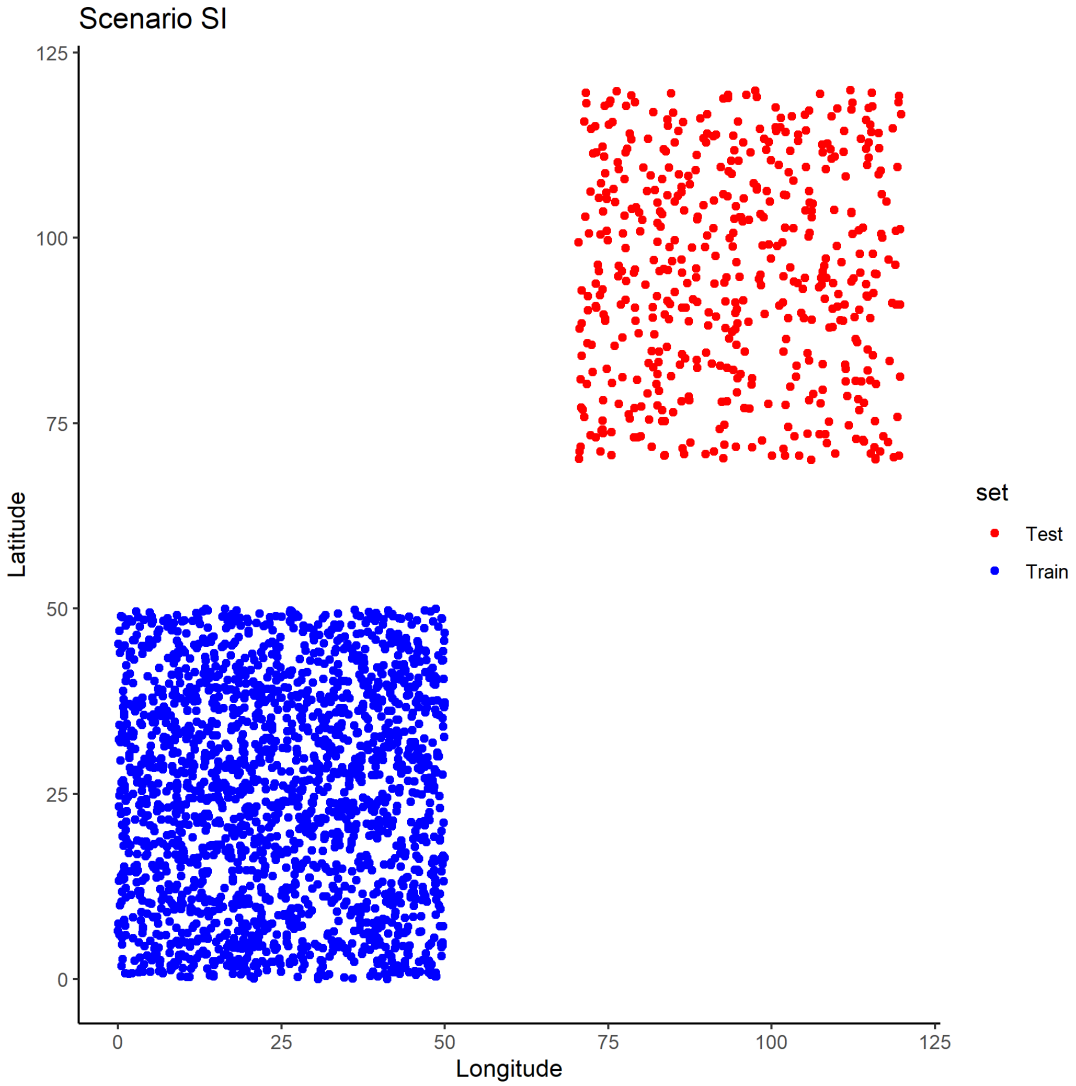


Figure 3.1.4: An example of Scenario SI, where train and test data points are geographically separated by at least distance $\geq r$.

- **Spatially Independent + Covariate Shift (SICS)**: Visually on a geographical region, SICS is similar to the SI scenario. However, additional checks for CS were placed using the KS test with Bonferroni correction. If the KS test indicates no CS, it will be manually induced between the train and the test sets.

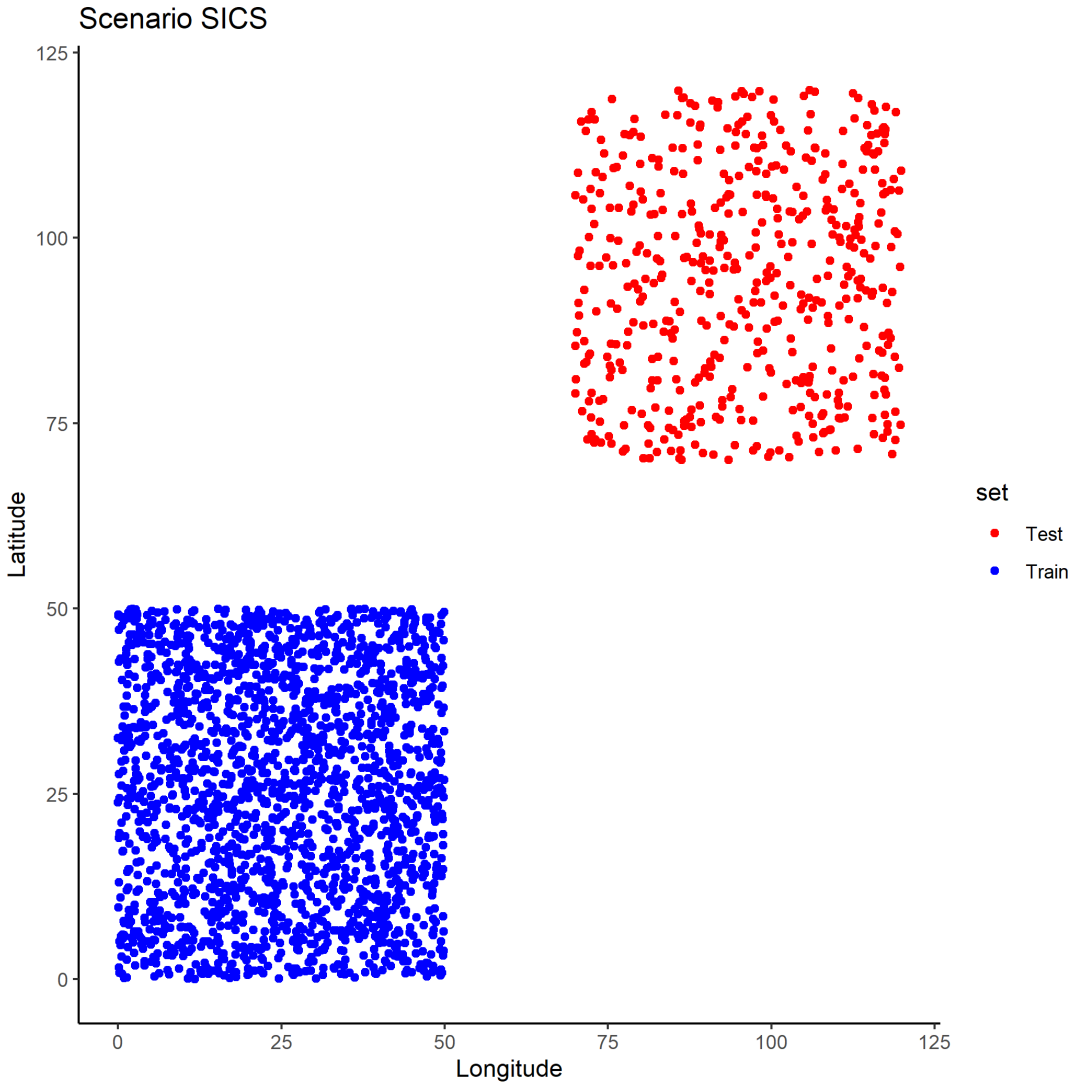


Figure 3.1.5: An example of Scenario SICS, where the train and the test data points are geographically separated by at least distance $\geq r$.

3.2 Procedure to Check and Induce Covariate Shift

In scenarios that involve CS (SDCS and SICS), the presence of CS was evaluated using the KS test. KS test assesses for CS between the train and the test sets. KS test is a non-parametric method that compares the cumulative distribution functions (CDFs) of two datasets to determine if they came from the same distribution. It measures the maximum vertical distance between the CDFs to determine how different the distributions between the two datasets are. If p-value is low, it would suggest that the two distributions are significantly different, indicating that CS is present. This test was

only applied to key variables X_1, X_2 and X_3.

To control the risk of type I errors when making multiple comparisons across different variables, the Bonferroni correction was applied to the significant level α. The corrected significance level α' is calculated as such:

$$\alpha' = \frac{\alpha}{m} \tag{3.2.1}$$

where m is the number of variables being tested. If the KS test rejects the null hypothesis $p \geq \alpha'$, it would indicate that CS is present. However, if the test fails to reject the null hypothesis, no CS is assumed to be present, and hence, manual intervention is applied to induce CS. Table 3.2.1 shows how CS is induced for each variable.

Variable	**What it represents**	**How CS will be induced**	**Reasoning**
X_1	Standard variable (topology, soil, etc.)	Shifting mean by addition	Shifting the mean of X_1 simulates a consistent, uniform change in environmental characteristics across the entire dataset.
X_2	Precipitation	Increase variance by multiplication	By multiplying the feature values, it would simulate greater deviations from the mean, which is more realistic for precipitation patterns.
X_3	Temperature	Shift mean by addition and then increase variance by multiplication	The combined approach of shifting both the mean and variance allows for simulating both gradual and extreme temperature changes, often observed together in the context of climate change.

Table 3.2.1: How CS is manually induced to each feature.

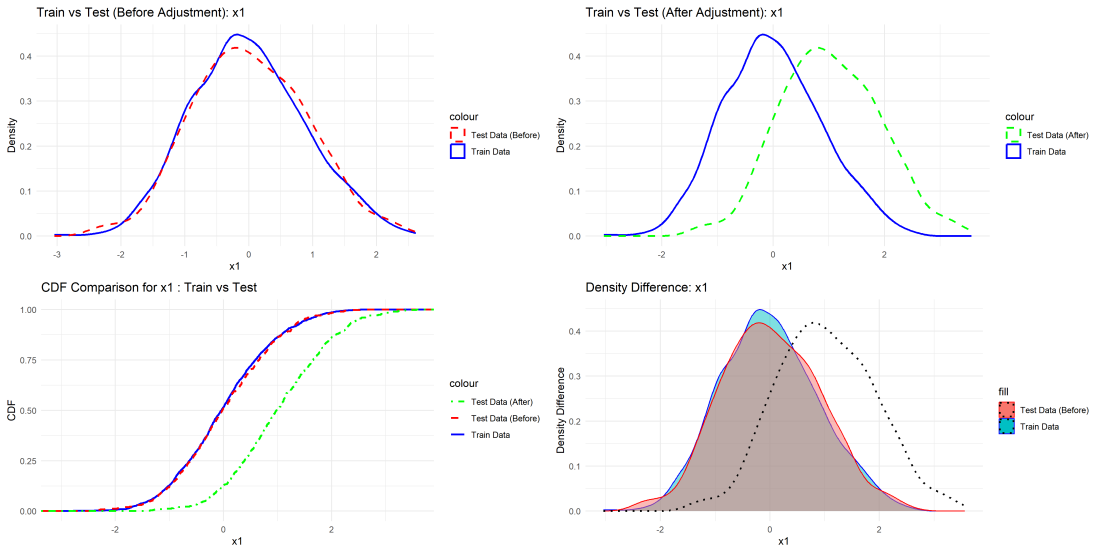


Figure 3.2.1: An example of inducing CS for the variable X_1 for scenario SD. Examples for X_2 and X_3 can be found in the Appendix (Figures 9.0.1 and 9.0.2)

- **Top Left**: The density plot shows minimal differences between the train set (blue) and the test set (before adjustment, red dashed).
- **Top Right**: After inducing CS, the test set (green dashed) shows a clear divergence from the train set.
- **Bottom Left**: The Cumulative Distribution Function (CDF) plot highlights how the test set's CDF (green) shifts significantly after adjustment, diverging from the train set (blue).
- **Bottom Right**: The density difference plot emphasizes the shift, with the black dotted line showing how the test set distribution changes after adjustment.

	Spatial Dependence (SD)	**Spatial Independence (SI)**
No Covariate Shift	Scenario SD: $d < r$ and $p \geq \alpha'$	Scenario SI: $d \geq r$ and $p \geq \alpha'$
Covariate Shift	Scenario SD + CS: $d < r$ and $p < \alpha'$	Scenario SI + CS: $d \geq r$ and $p < \alpha'$

Table 3.2.2: Conditions for classifying the dataset into the relevant scenarios. d is the distance between the nearest data points between the train and the test set. r is the spatial autocorrelation range of the dataset. p is the p-value from the KS test that assesses the distributional similarity between the train and the test sets. α' is the adjusted significance level derived from applying the Bonferroni correction to control for multiple comparisons.

For each scenario in Table 3.2.2, we simulate data for z using different spatial ranges

(r) for X_1, X_2, X_3. Here, the sill of the variogram (s) is fixed to 1 and the nugget (nu) is fixed to 0. Note that $r = 1$ indicates no spatial autocorrelation.

Intended Scenario	**Training Set**	**Test Set**
SD, SI, SD + CS	$s = 1, r = 1, 6, 12, nu = 0$	$s = 1, r = 1, 6, 12, nu = 0$
SI + CS	$s = 1, r = 1, 6, 12, nu = 0$	$s = 0.5, r = 1, 6, 12, nu = 0$

Table 3.2.3: Parameters of each scenario.

3.3 Experiment Design

For each simulated dataset, the random forest was used to model the target variable z and fitted using variables X_1, ..., X_6. Random forest models are known to be highly accurate, even without adjusting their parameters [31]. As such, the default parameters were used for all random forest models. After fitting, the performance of each model was assessed using the different CV metrics and several evaluation metrics commonly used to measure the model's accuracy. An external dataset was also used to assess the CV estimates of error. This dataset was simulated using the same process to be consistent with the scenarios and ranges that the model was trained upon, but also designed to reflect a completely different geographical region.

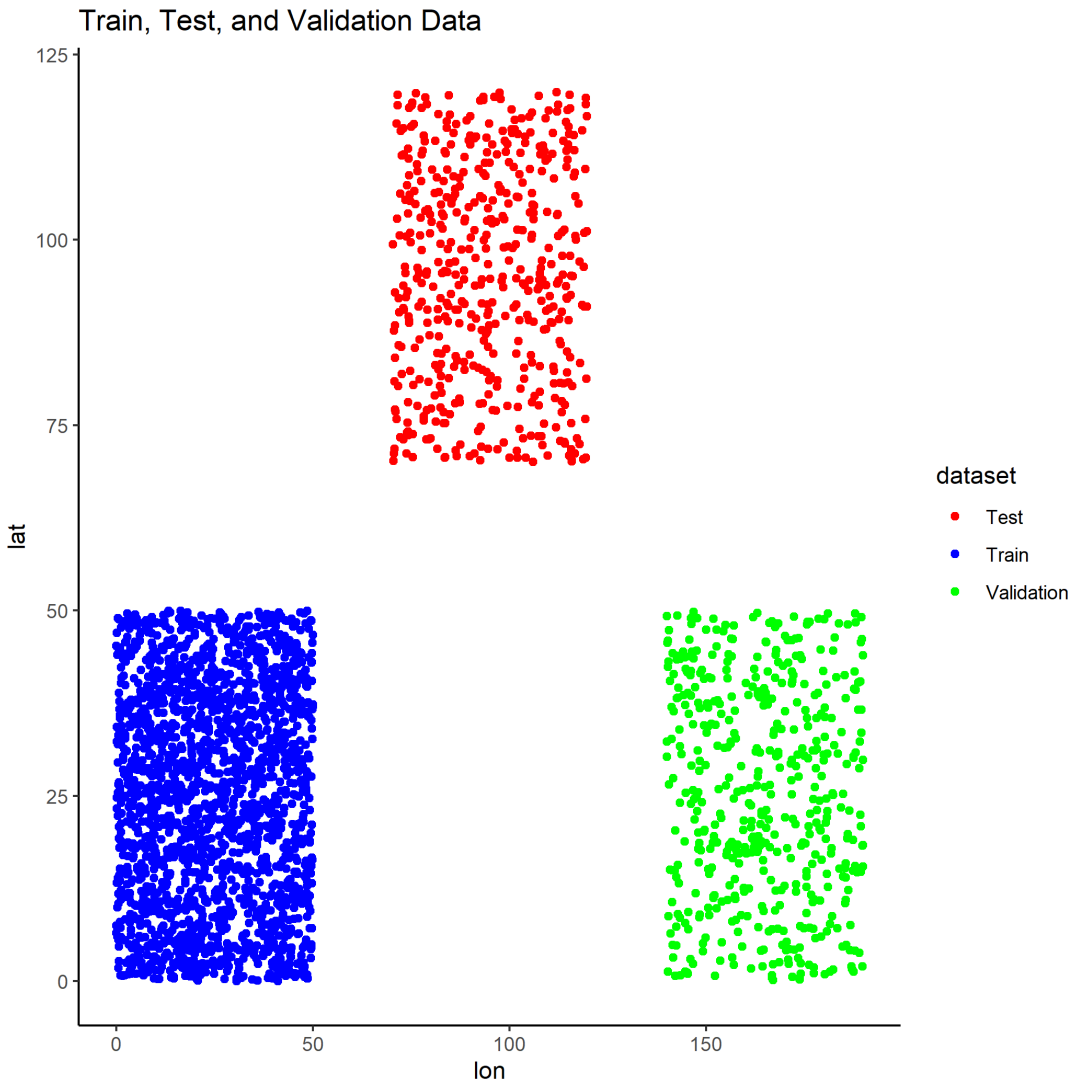


Figure 3.3.1: An example of the external dataset (green) and its relationship to the train and the test set. This example is for scenario SI where $r = 6$.

3.3.1 Root Mean Square Error (RMSE)

RMSE measures the average magnitude of the errors between the predicted values and actual values [32]. RMSE squares the individual prediction errors, meaning that larger errors would contribute more to the overall metric than smaller ones. This allows RMSE to show how well the model predicts the target variable, with lower values indicating better performance.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(z_i - \hat{z}_i)^2} \tag{3.3.1}$$

Where:

- n is the number of data points.
- z_i is the actual observed value.
- $\hat{z}_i$ is the predicted value.

3.3.2 Mean Absolute Error (MAE)

MAE is the average of the absolute differences between the predicted and the actual values [33]. Like RMSE, lower MAE values indicate more accurate predictions.

$$\text{MAE} = \frac{1}{n}\sum_{i=1}^{n}|z_i - \hat{z}_i| \tag{3.3.2}$$

Where:

- n is the number of observations.
- z_i is the actual observed value.
- $\hat{z}_i$ is the predicted value.

Unlike RMSE, which penalises large errors due to the squared error term, MAE treats all errors equally, giving a smaller value in the presence of large deviations that outliers could cause. This makes MAE less sensitive to outliers and particularly useful for measuring an average error without disproportionately penalizing larger errors.

3.3.3 Coefficient of Determination (R^2)

R^2 measures how well the variance in the target variable is explained by the model [34]. It ranges from 0 to 1, with higher values indicating better model performance.

$$R^2 = 1 - \frac{\mathrm{SS}_{\mathrm{res}}}{\mathrm{SS}_{\mathrm{tot}}} \tag{3.3.3}$$

Where:

- $\mathrm{SS}_{\mathrm{res}}$ is the sum of squares of residuals.
- $\mathrm{SS}_{\mathrm{tot}}$ is the total sum of squares.

Sum of Square of Residuals $(\mathbf{SS_{res}})$ $\mathrm{SS}_{\mathrm{res}}$ measures the amount of variance in the error term, or residuals, of a regression model. A smaller $\mathrm{SS}_{\mathrm{res}}$ indicates that the model fits the data well, while a larger sum suggests a poorer fit to the data.

$$\mathrm{SS}_{\mathrm{res}} = \sum_{i=1}^{n} (z_i - \hat{z}_i)^2 \tag{3.3.4}$$

Where:

- n is the number of observations.
- z_i is the observed value.
- $\hat{z}_i$ is the predicted value.

Total Sum of Squares $(\mathbf{SS_{tot}})$ $\mathrm{SS}_{\mathrm{tot}}$ measures the overall variance in the observed data. It represents the total variation in the dependent variable and serves as a benchmark to compare the fit of a regression model. A higher $\mathrm{SS}_{\mathrm{tot}}$ indicates more variability in the data, while a lower $\mathrm{SS}_{\mathrm{tot}}$ suggests less variability.

$$SS_{tot} = \sum_{i=1}^{n} (z_i - \bar{z})^2 \tag{3.3.5}$$

Where:

- n is the number of observations.
- z_i is the observed value.
- $\bar{z}$ is the mean of the observed values.

3.3.4 Bias

Bias measures the systematic error in the model's predictions, indicating if the model tends to over- or underestimate actual values. A bias value close to zero indicates that, on average, predictions are unbiased. A positive or negative bias indicates consistent over or underestimation.

$$Bias = \frac{1}{n} \sum_{i=1}^{n} (\hat{z}_i - z_i) \tag{3.3.6}$$

Where:

- n is the number of observations.
- z_i is the actual observed value.
- $\hat{z}_i$ is the predicted value.

3.4 Hyperparameters of Spatial and Non-Spatial CV Techniques

In our CV techniques, we varied the number of k folds for all techniques, exploring 2, 5 and 10 folds.

For RKFCV and IWCV, the train set was split randomly into k folds. For IWCV, density ratios were estimated using the RuLSIF method with $\alpha = 0$ and 50 kernels to optimise the importance weights [23].

BootCV was conducted with 50, 75, and 100 bootstrap samples. Studies have shown that increasing the number of bootstrap samples can improve the robustness and precision of predictions [35]. Additionally, a balance is required between computational efficiency and the accuracy gained from more samples. Hence, 50, 75 and 100 are often considered reasonable trade-offs [36].

For SKFCV, the train set was divided into spatially contiguous k folds. This technique ensures that each fold represents a distinct geographic area, minimizing spatial autocorrelation between the train and the test sets. The folds were created by partitioning the study area into blocks that maintain spatial integrity.

For BCV, the dataset was divided into blocks and then grouped into k folds. The block size was set to match the spatial autocorrelation range (block size is $2, 6, 12$ when r is $1, 6, 12$), as BCV aims to reduce spatial autocorrelation within the test and the train sets. In this study, all 3 common blocking methods (random, systematic [snake] and systematic [continuous]) were implemented to ensure a comprehensive evaluation of BCV's performance. However, BCV's effectiveness is sensitive to the block size, with larger blocks increasing the estimated test error due to higher inter-block CS [37].

Finally, BuffCV used a buffer size equal to r, and the block size was fixed at 12.5 to ensure the grid could be evenly split into 16 blocks. This decision was made to prevent cases whereby the buffer would remove all the train data points, leading to situations without valid train sets due to the interaction between buffer size and block size.

The train and the test sets were split beforehand for all non-spatial techniques to ensure that each technique used the same samples. This consistent sample split was necessary to maintain comparability across the different CV techniques. By ensuring the samples were constant, we can accurately assess the impact of spatial and non-spatial CV on model performance without introducing confounding variables due to different sample allocations.

Parameter	Values	Definition
r	1, 6, 12	The range fed into the Gaussian model to simulate data.
k	2, 5, 10	The number of folds to assign data into. Each fold was used precisely once.
Block Size	1, 6, 12; 12.5	The block size used when performing BCV and BuffCV where the former is for BCV and the latter for BuffCV.
Blocking Method	Random, Systematic (Snake), Systematic (continuous)	The method for assigning folds in BCV and BuffCV. Random: randomly assigns blocks to folds. Snake: labels the first row of blocks from left to right, then from right to left and repeats. Continuous: labels each row from left to right, moving from the bottom row up.
Buffer	2, 6, 12	The buffer zone size to apply around the test set.
Bootstrap Samples	50, 75, 100	The number of bootstrap samples performed.

Table 3.4.1: Parameter values and their definitions for CV techniques.

4. Results

4.1 Performance in Spatial Independence (SI) Scenario

Looking at the aggregated results from all spatial range settings, most techniques—BootCV, RKFCV, IWCV, SKFCV, BuffCV, and BCV—performed consistently well in SI scenarios. RMSE values ranged from 0.07 to 1.8 in the test datasets, with BuffCV being an outlier when the number of folds $k = 2$ and 0.22 to 1.36 in external datasets. MAE values ranged from 0.06 to 2.77 for the test dataset, with BuffCV again being an outlier when $k = 2$ and 0.17 to 1.07 for the external dataset. While BuffCV and BootCV exhibited the highest error values, the differences among most other techniques were minimal.

R^2 values are consistently high, typically above 0.95, reflecting strong predictive performance in spatially independent conditions, with BCV, BuffCV (when $k \geq 2$), and IWCV performing particularly well across all r and folds.

k / B	Technique	Test			External		
		Min	**Median**	**Max**	**Min**	**Median**	**Max**
2 / 50	Random K	0.322	0.492	0.906	0.258	0.318	0.811
	BootCV	0.277	0.461	1.803	0.244	0.313	1.350
	Spatial K	0.370	0.559	1.462	0.224	0.273	0.599
	BlockCV	0.088	0.131	0.188	0.226	0.271	0.599
	BuffCV	**0.143**	**0.408**	**3.247**	0.223	0.264	0.596
	IWCV	0.305	0.351	0.430	0.298	0.346	0.516
5 / 75	Random K	0.323	0.488	0.911	0.256	0.320	0.801
	BootCV	0.278	0.452	1.799	0.240	0.314	1.362
	Spatial K	0.299	0.387	1.167	0.226	0.272	0.595
	BlockCV	0.074	0.103	0.167	0.224	0.264	0.582
	BuffCV	0.091	0.121	0.257	0.226	0.272	0.610
	IWCV	0.254	0.303	0.375	0.262	0.297	0.442
10 / 100	Random K	0.321	0.492	0.916	0.253	0.318	0.799
	BootCV	0.277	0.452	1.804	0.244	0.313	1.362
	Spatial K	0.274	0.347	0.515	0.223	0.271	0.599
	BlockCV	0.066	0.097	0.191	0.225	0.273	0.599
	BuffCV	0.076	0.108	0.161	0.225	0.264	0.599
	IWCV	0.248	0.297	0.356	0.257	0.289	0.465

Table 4.1.1: Summary of the minimum, median, and maximum RMSE values across different folds/bootstrap samples k/B and techniques for both test and external datasets for SI. Techniques include Random K, BootCV, Spatial K, BlockCV, BuffCV, and IWCV. BuffCV $k = 2$ is highlighted as an outlier in the test dataset.

k / B	Technique	Test			External		
		Min	**Median**	**Max**	**Min**	**Median**	**Max**
2 / 50	Random K	0.184	0.267	0.608	0.171	0.204	0.549
	BootCV	0.176	0.264	1.508	0.167	0.204	1.079
	Spatial K	0.225	0.333	1.189	0.154	0.179	0.390
	BlockCV	0.062	0.083	0.127	0.155	0.179	0.399
	BuffCV	**0.085**	**0.260**	**2.777**	0.154	0.175	0.407
	IWCV	0.196	0.225	0.243	0.198	0.223	0.300
5 / 75	Random K	0.184	0.263	0.609	0.167	0.204	0.543
	BootCV	0.173	0.261	1.512	0.164	0.206	1.099
	Spatial K	0.178	0.241	0.921	0.155	0.180	0.394
	BlockCV	0.053	0.067	0.084	0.155	0.179	0.396
	BuffCV	0.062	0.075	0.163	0.155	0.176	0.395
	IWCV	0.194	0.223	0.206	0.172	0.193	0.290
10 / 100	Random K	0.182	0.266	0.610	0.167	0.204	0.551
	BootCV	0.171	0.259	1.512	0.167	0.204	1.095
	Spatial K	0.173	0.214	0.358	0.153	0.179	0.389
	BlockCV	0.048	0.065	0.081	0.155	0.179	0.389
	BuffCV	0.055	0.069	0.092	0.153	0.175	0.399
	IWCV	0.164	0.186	0.213	0.167	0.189	0.273

Table 4.1.2: Summary of the minimum, median, and maximum MAE values across different folds/bootstrap samples k/B and techniques for both test and external datasets for SI. Techniques include Random K, BootCV, Spatial K, BlockCV, BuffCV, and IWCV. BuffCV $k = 2$ is highlighted as an outlier in the test dataset.

k / B	Technique	Test			External		
		Min	Median	Max	Min	Median	Max
2 / 50	Random K	0.698	0.922	0.977	0.748	0.967	0.982
	BootCV	0.028	0.945	0.979	0.301	0.967	0.984
	Spatial K	0.209	0.908	0.962	0.862	0.976	0.985
	BlockCV	0.973	0.995	0.997	0.863	0.976	0.986
	BuffCV	**-1.765**	**0.952**	**0.995**	0.863	0.978	0.986
	IWCV	0.941	0.962	0.975	0.897	0.963	0.975
5 / 75	Random K	0.695	0.924	0.976	0.754	0.966	0.982
	BootCV	0.032	0.944	0.979	0.288	0.967	0.984
	Spatial K	0.287	0.950	0.976	0.864	0.976	0.986
	BlockCV	0.994	0.997	0.998	0.857	0.975	0.986
	BuffCV	0.977	0.996	0.998	0.870	0.978	0.986
	IWCV	0.957	0.971	0.981	0.925	0.973	0.981
10 / 100	Random K	0.683	0.924	0.976	0.755	0.966	0.983
	BootCV	0.027	0.945	0.979	0.289	0.967	0.984
	Spatial K	0.833	0.953	0.981	0.862	0.976	0.985
	BlockCV	0.992	0.997	0.998	0.864	0.976	0.986
	BuffCV	0.993	0.997	0.998	0.873	0.978	0.986
	IWCV	0.961	0.974	0.981	0.915	0.974	0.982

Table 4.1.3: Summary of the minimum, median, and maximum R^2 values across different folds/bootstrap samples k/B and techniques for both test and external datasets for SI. Techniques include Random K, BootCV, Spatial K, BlockCV, BuffCV, and IWCV. BuffCV $k = 2$ is highlighted as an outlier in the test dataset.

One thing to note is that using BuffCV when $k = 2$ resulted in poor performance of the model on the test sets. One possible reason could be due to a block size mismatch. The block size of BuffCV was set at 12.5 and may not have been appropriate for the spatial pattern displayed by the dataset. If the block size is too large or too small relative to the spatial autocorrelation range, the train and the test sets may not represent the overall spatial structure, leading to a high variance.

Additionally, while SKFCV exhibits some performance challenges with lower values of k , increasing k helps stabilize its performance.

4.2 Impact of Spatial Dependence (SD)

When SD is introduced, performance begins to differ for each technique. BCV, BootCV, IWCV, BuffCV (except for $k = 2$) and RKFCV seemed to perform well against the test set, with RMSE and MAE values ranging from 0.1 to 0.4. BuffCV, especially for $k = 2$ exhibits a large increase in error, particularly in the external dataset where RMSE values can exceed 3.0. SKFCV seems to struggle with performance, with both RMSE and MAE varying vastly between 0.2 and 0.8, but its performance improves with increasing folds. When all techniques were tested against the external dataset, they all performed similarly, with little to no difference in the error values. R^2 values remain relatively high and consistent across the techniques. The figures supporting these observations can be found in the Appendix (Figures 9.0.1 - 9.0.3).

From Figure 4.2.1, 4.2.2 and 4.2.3, we note that while SKFCV exhibits higher test error compared to the rest (apart from the case of BuffCV when $k = 2$), it seems to provide a more reliable estimate of out-of-sample performance.

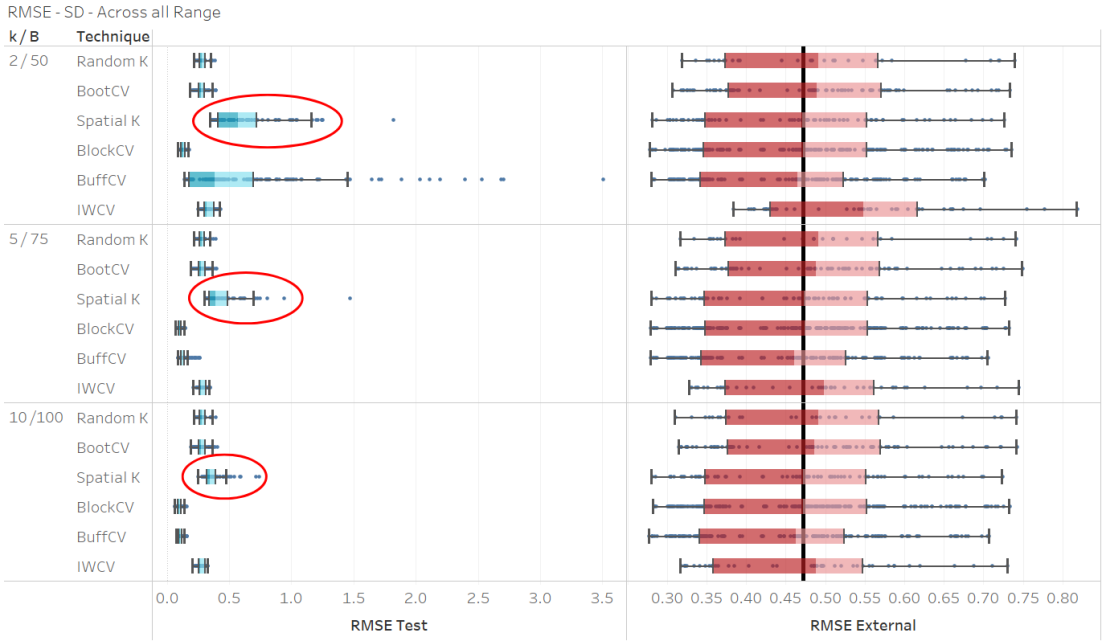


Figure 4.2.1: RMSE distribution for Spatial Dependence (SD) scenario across different cross-validation techniques and folds/bootstrap samples (k / B). All techniques (BootCV, RKFCV, IWCV, SKFCV, BuffCV, and BlockCV) demonstrate relatively low RMSE values, though SKFCV shows higher variability across folds. External dataset RMSE values are generally higher, reflecting the impact of spatial dependence on model performance.

Figure 4.2.2: MAE distribution for Spatial Dependence (SD) scenario across different cross-validation techniques and folds/bootstrap samples (k / B). Techniques such as BootCV, RKFCV, and IWCV show low MAE values for test datasets but display increased errors for external datasets, reflecting the challenges posed by spatial dependence. SKFCV demonstrates the largest spread in error values, improving with higher fold numbers.

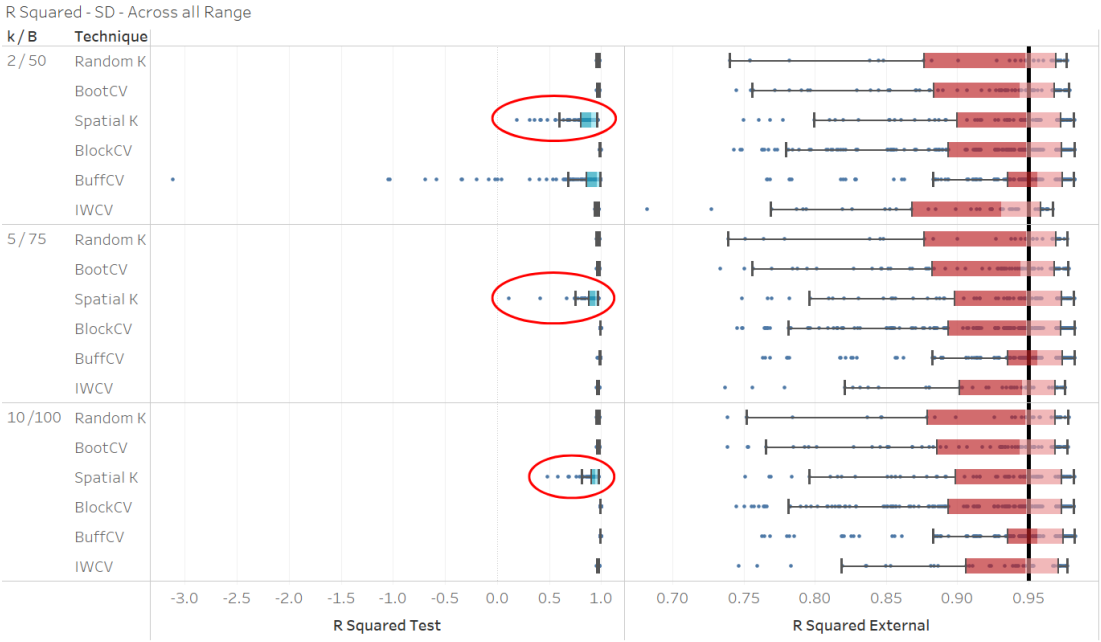


Figure 4.2.3: R^2 distribution for Spatial Dependence (SD) scenario across different cross-validation techniques and folds/bootstrap samples (k / B). All techniques maintain relatively high R^2 values, though slight decreases are observed with increasing folds. SKFCV shows more variance in test datasets.

4.3 Challenges with Covariate Shift (SDCS and SICS)

In both SDCS and SICS scenarios, CS presents the greatest challenge across all techniques, leading to significant increases in all error metrics. The percentage changes in external test errors were higher when transitioning from SD to SDCS than when shifting from SI to SICS. This suggests that spatial dependence compounds with CS to cause the additional increase in errors.

RKFCV and BootCV exhibit the highest sensitivity to CS, with RMSE increases of 165.5% and 159.8%, respectively, in the SDCS test set. Their performance on external datasets saw increases of 110.9% and 107.0%, respectively. In contrast, IWCV and SKFCV demonstrated greater resilience, with RMSE increases of only 7.3% and 8.1% in the SDCS test set.

Similarly, MAE values followed the same pattern, where RKFCV and BootCV showed increases of 164.9% and 157.4%, respectively, in the SDCS test set, compared to much smaller increases for IWCV (3.7%) and SKFCV (8.1%). R Squared values also dropped significantly, particularly for BootCV (by 50.2%) and RKFCV (by 44.6%), whereas IWCV showed minimal decline (0.6%).

Overall, these findings suggest that techniques like RKFCV and BootCV fit models which are more sensitive under covariate shift. In contrast, IWCV and SKFCV offer more robustness as shown in test performance.

Technique	**SD to SDCS**		**SI to SICS**	
	Test	**External**	**Test**	**External**
RKFC	165.5%	110.9%	35.6%	87.9%
BootCV	159.8%	107.0%	11.4%	44.9%
SKFCV	8.0%	87.2%	16.3%	80.6%
BCV	-6.6%	87.2%	-3.0%	80.6%
BuffCV	-8.0%	100.5%	-4.5%	64.9%
IWCV	7.3%	92.3%	7.9%	85.5%

Table 4.3.1: Percentage increase in RMSE between Spatial Dependence (SD) to Spatial Dependence with Covariate Shift (SDCS) and Spatial Independence (SI) to Spatial Independence with Covariate Shift (SICS). The table compares the performance in both Test and External datasets for each cross-validation technique.

Technique	SD to SDCS		SI to SICS	
	Test	External	Test	External
RKFC	164.9%	100.4%	33.6%	75.1%
BootCV	157.4%	97.2%	6.4%	34.8%
SKFCV	8.1%	79.3%	15.1%	69.8%
BCV	-6.2%	79.2%	-6.0%	69.9%
BuffCV	-6.5%	91.4%	-6.7%	55.7%
IWCV	3.7%	83.5%	1.7%	72.9%

Table 4.3.2: Percentage increase in Mean Absolute Error (MAE) between Spatial Dependence (SD) to Spatial Dependence with Covariate Shift (SDCS) and Spatial Independence (SI) to Spatial Independence with Covariate Shift (SICS). The table reports the percentage changes in error values across Test and External datasets for various cross-validation techniques.

Technique	SD to SDCS		SI to SICS	
	Test	External	Test	External
RKFC	-44.6%	-23.2%	-12.3%	-1.9%
BootCV	-50.2%	-20.6%	-21.2%	-18.1%
SKFCV	-9.1%	-12.2%	-10.4%	-8.3%
BCV	0.1%	-12.2%	-10.4%	-12.2%
BuffCV	5.6%	-9.5%	-8.0%	-8.0%
IWCV	-0.6%	-16.7%	-11.5%	-16.7%

Table 4.3.3: Percentage change in R^2 values between Spatial Dependence (SD) to Spatial Dependence with Covariate Shift (SDCS) and Spatial Independence (SI) to Spatial Independence with Covariate Shift (SICS). This table highlights the performance differences in model fit across Test and External datasets for each cross-validation technique.

4.4 Comparative Performance by fold and range *r*

The performance of the techniques varies depending on r and k. BCV consistently performs well across increasing values of r and k but shows significant performance deterioration when validated on different regions, particularly in external datasets as compared to test datasets. BootCV is highly sensitive to the number of bootstrap samples and r, performing the best at the lower values or r. BuffCV, while sensitive to r, maintains consistent performance across different numbers of folds. IWCV remains

stable across folds and r, showing minimal changes in error rates. The figures supporting these observations can be found in the Appendix (Figures 9.0.12 - 9.0.23).

RKFCV performs the best when r is low but declines in performance as r increases as shown in Figures 4.4.1, 4.4.2 and 4.4.3. Finally, SKFCV shows increased error rates with higher values of r, particularly when evaluated on test datasets. However, with an increased number of folds, its performance increases as shown in Figures 4.4.4, 4.4.5 and 4.4.6.

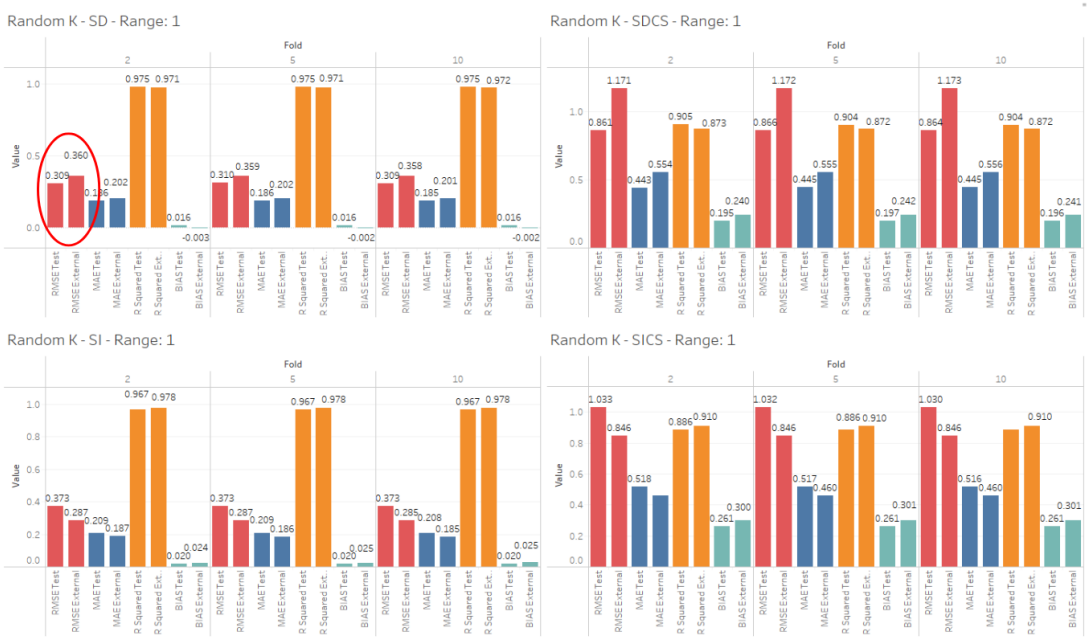


Figure 4.4.1: RKFCV $r = 1$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and different k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

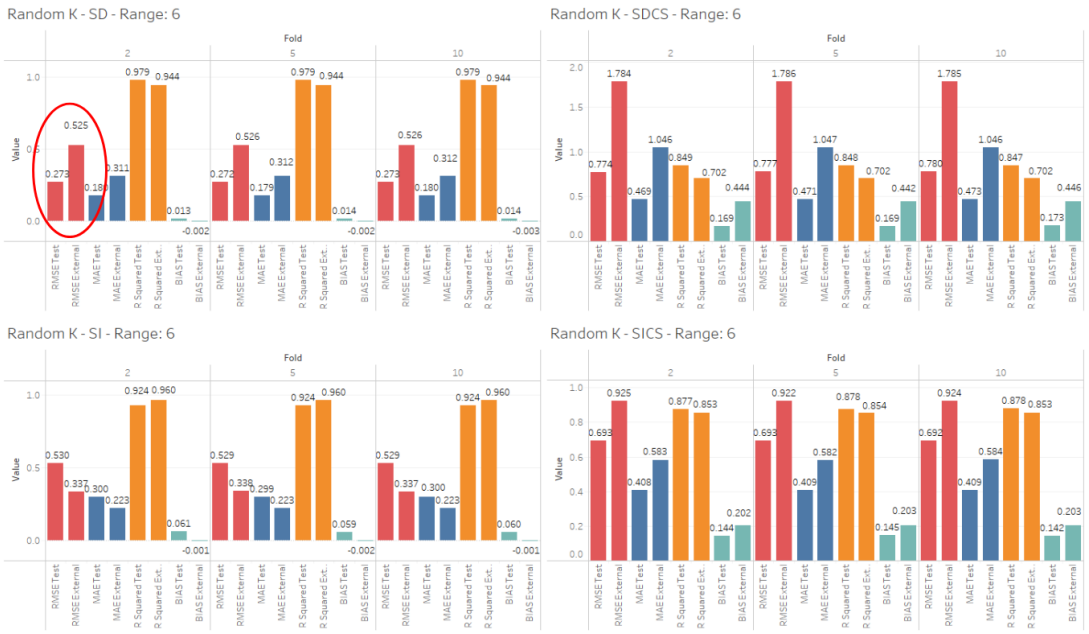


Figure 4.4.2: RKFCV $r = 6$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and different k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

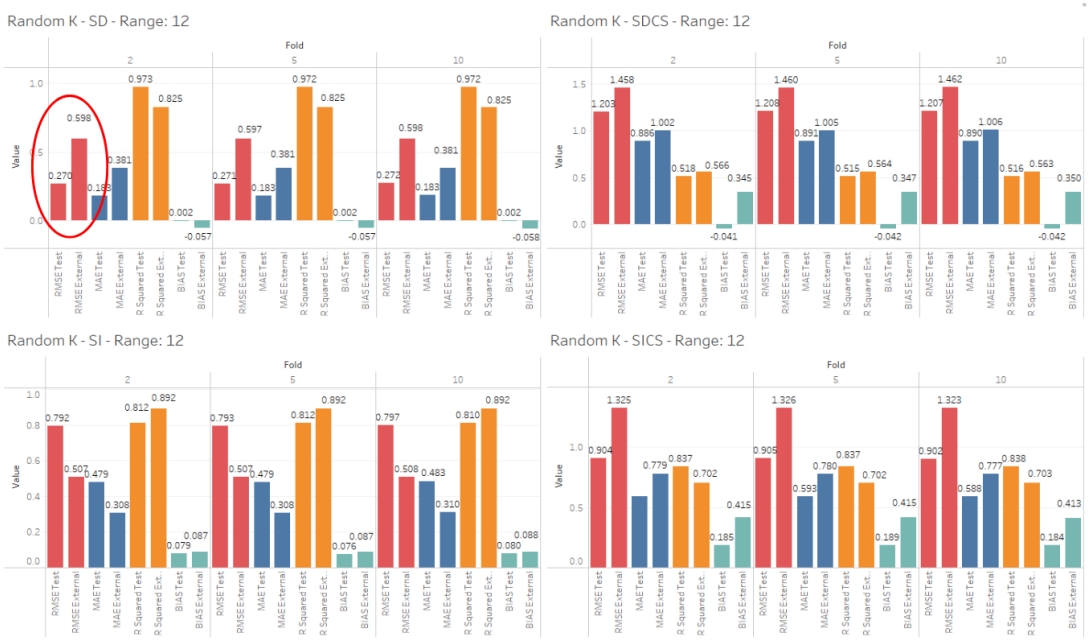


Figure 4.4.3: RKFCV $r = 12$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and different k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

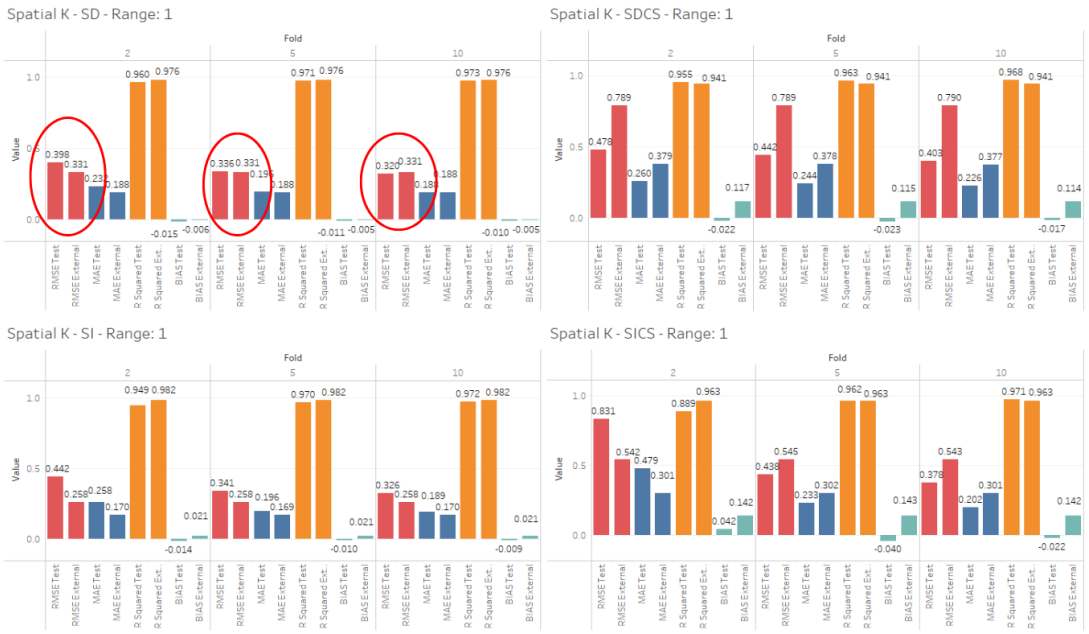


Figure 4.4.4: SKFCV $r = 1$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

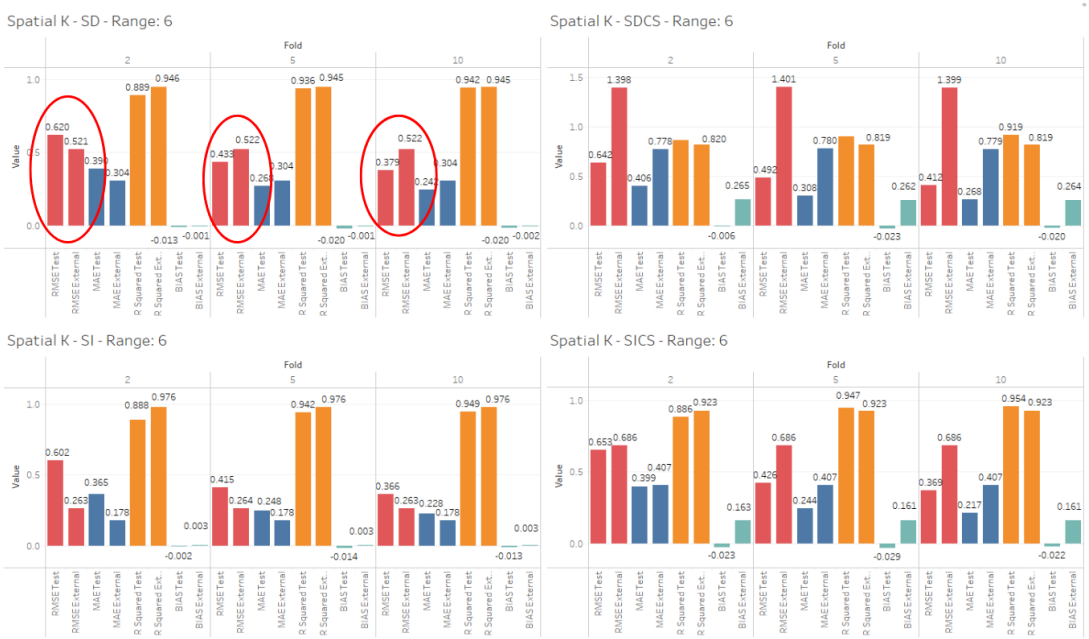


Figure 4.4.5: SKFCV $r = 6$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

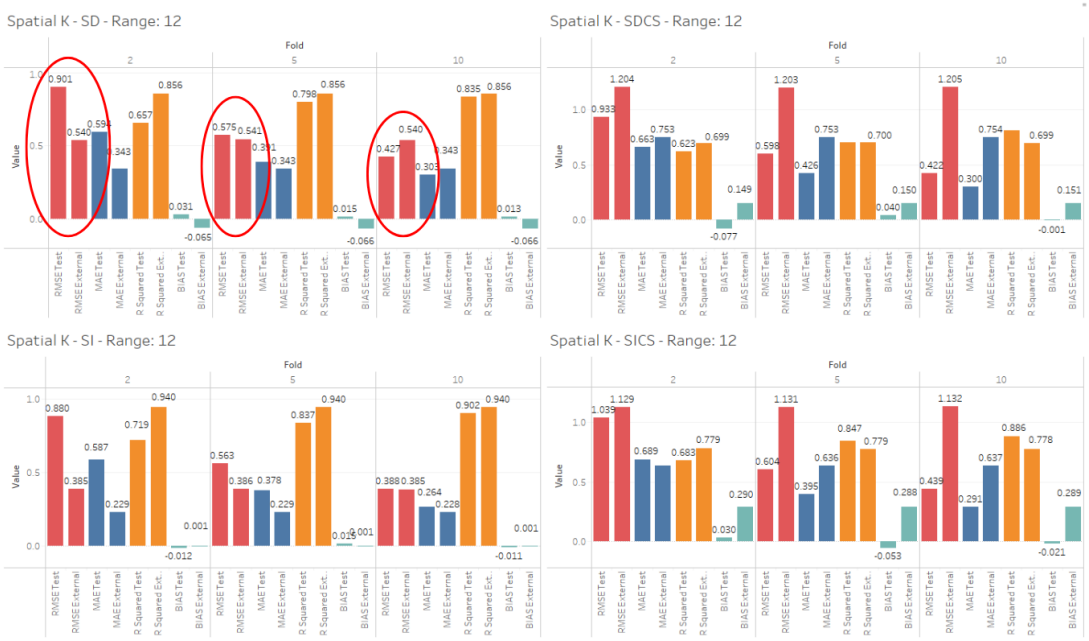


Figure 4.4.6: SKFCV $r = 12$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and different k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

4.5 Influence of Covariates and Noise Variables on Cross-Validation Performance

Across all metrics (RMSE, MAE, R^2 and Bias) and scenarios (SD, SDCS, SI, and SICS), the introduction of covariates X_1, X_2, X_3 to the random forest models consistently improves model performance. Adding noise variables X_4, X_5, X_6 leads to only minor increases in error metrics and minor decreases in predictive accuracy. This suggests that the RF models did not deem the noise variables used in the experiment important. It is likely due to them not explaining the variation in the z in the train sets well. Further work can involve noise terms which are more correlated to the covariates to explore the effects of model misspecification.

We present the plot for RKFCV here, with similar plots for the remaining CV techniques available in the Appendix (Figures 9.0.24 - 9.0.28) for reference.

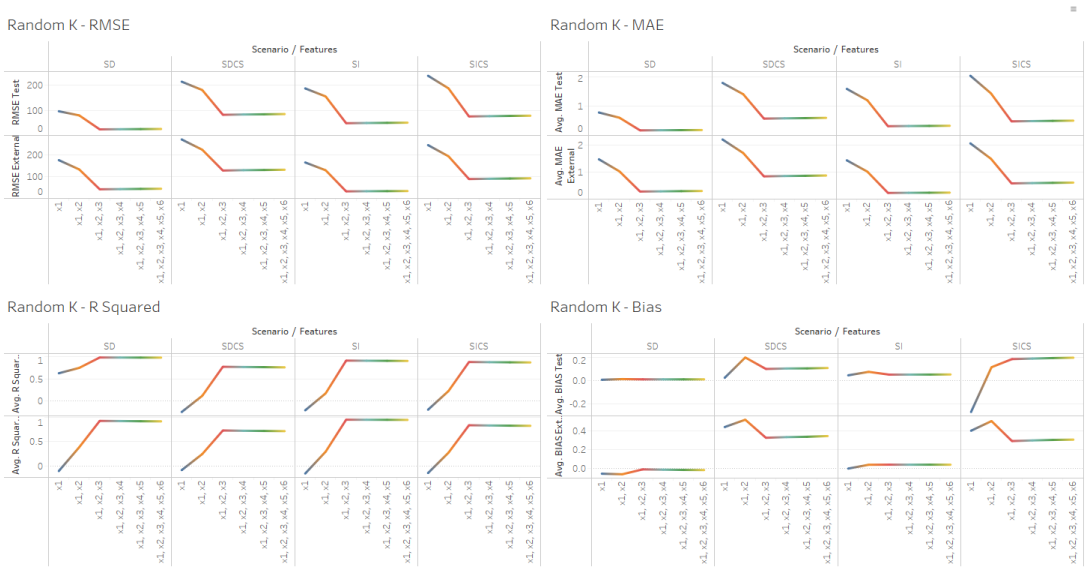


Figure 4.5.1: Impact of Covariates and Noise Variables on Random K-Fold Cross-Validation Performance Across Different Scenarios (SD, SDCS, SI, and SICS): Minimal performance degradation is observed with the addition of noise variables X_4, X_5, X_6, with covariates X_1, X_2, X_3 having a more positive influence on the result.

5. Discussion

5.1 Addressing Bias and Over-Optimism in Cross-Validation Techniques

The findings in this study are consistent with prior research, especially those highlighted by [7], which raised concerns regarding bias and over-optimism in non-spatial CV techniques. The results from BootCV and RKFCS highlight the issue of spatial leakage since these non-spatial techniques do not explicitly account for spatial dependence. When trained on datasets with spatial dependence, the model may ”see” patterns between the train and test sets, leading to overly optimistic results. This spatial leakage leads to a lower error rate during cross-validation but often does not accurately reflect the model's real-world performance. For instance, while RKFCV performs well in SI scenarios, the results show substantial performance degradation in SD and SDCS scenarios.

In the case of BuffCV, the results for $k = 2$ emerge as an outlier in its performance, with a significant decrease in performance in both test and external datasets. As noted in the results, block size mismatch can be a possible reason for this outlier. This observation reinforces the need for careful hyperparameter tuning when deploying CV techniques, similar to findings by [15].

5.2 Handling Covariate Shift and Spatial Variability

The results shown in SDCS and SICS scenarios highlight the challenges that CS brings to SP modelling. Non-spatial techniques tend to exhibit large jumps in error metrics, supporting the conclusions of papers like [38], which warned that distribution mismatches between the train and the test sets could drastically degrade model performance. Although spatial techniques fare slightly better, the increase in error is still notable.

These findings align with the conclusions of [7] and [39], who emphasized the need for spatial CV to mitigate bias and improve the accuracy of model performance estimates. The increase in error seen in SDCS and SICS for both non-spatial and spatial CV techniques indicates that CS, especially when combined with spatial dependence, can

present a serious challenge for SP modelling, a consideration often neglected.

5.3 Limitations of the Experimental Design

While this study offers valuable insights, several limitations must be acknowledged. Firstly, while Leave-One-Out CV (LOOCV) and Buffered Leave-One-Out CV (BLOO-CV) were considered initially, they were omitted due to their high computational costs. Future studies could benefit from testing these techniques to better account for spatial variability in a less biased manner.

Secondly, only random forest models were used across all techniques, scenarios and r. The performance of different models, such as linear regression or other machine learning models, will differ under these same CV techniques and parameters. This limitation could affect the generalizability of the results.

Third, little to no hyperparameter tuning was employed to better accommodate the CV techniques. Each technique was tuned using a generic set of parameters that could have worked well for some models but not others. A systematic hyperparameter tuning process could reveal further improvements in performance, particularly in handling real-world scenarios like CS.

Fourth, introducing more disruptive noise variables or increasing the variance in these variables could provide a more realistic test of the model's resilience in handling noisy, real-world data. The minimal impact of the noise variables used in this study demonstrates that while the model is robust, the experiment might not fully simulate the complexity of real-world scenarios where noise can have a greater effect.

Lastly, while the external dataset used in this study was geographically distant from the train set, it is still a simulated dataset with consistent scenarios and spatial autocorrelation ranges. A broader validation against real-world datasets from diverse geographical regions could provide different insights into the accuracy of these CV techniques.

5.4 Recommendations for Experiment Design

Based on these findings, researchers should follow a structured approach when doing their experiment designs with regard to SP modelling. The following guiding steps are recommended:

1. **Determine the Spatial Autocorrelation Range**: Before selecting a CV technique, it is important to assess the spatial autocorrelation present in the dataset. This will give a clearer picture as to which spatial techniques can be considered, as well as the plausible hyperparameters to use for specific techniques that require it. For instance the selection of block size for BCV and BuffCV, as well as the buffer size for BuffCV.

2. **Assess Covariate Shift**: Researchers should assess the presence of CS between the train and the test datasets. If the CS is significant, techniques like IWCV could be a preferable choice as it is able to handle this variability better.

3. **Hyperparameter Tuning**: As seen with BuffCV's performance, hyperparameter tuning can have a significant impact on results. Hyperparameter tuning should not be overlooked, as a one-size-fits-all approach will likely not work for every dataset or CV technique. Techniques with more parameters to work with, like BuffCV, need especially extra attention to hyperparameter tuning as they would produce vastly different results based on the combination of parameters.

4. **Validate Across External Datasets**: Depending on the context surrounding the research or experiment, models should be evaluated based on other real-world datasets from a different geographic region, if relevant. This would ensure a proper understanding of the model's accuracy, improving confidence in the model whilst avoiding unfounded confidence in it. This step is especially important when models are intended for application in new geographical regions.

6. Conclusion

In this study, we evaluated 6 CV techniques, RKFCV, BootCV, IWCV, SKFCV, BCV, and BuffCV, across multiple spatial scenarios: SD, SDCS, SI, and SICS. The results indicated that while all techniques perform well under SI conditions, significant differences in performance arise under SD, SDCS, and SICS scenarios.

One key takeaway is the importance of understanding both the dataset's spatial characteristics and the potential presence of CS. As our results show, models trained without accounting for spatial dependence may provide overly optimistic predictive models that do not generalise well to new geographic areas. This highlights the need for researchers to carefully choose and tune CV techniques based on the spatial properties of the data at hand.

Additionally, this study underscores the necessity for hyperparameter tuning, as demonstrated by BuffCV's block size sensitivity. Furthermore, validating models on external datasets from distinct geographical regions is crucial for understanding the model's actual performance when applied to real-world scenarios. As CS is often an unexplored aspect in spatial modelling, future works should aim to develop techniques or guidelines that can better manage spatial variability and CS, providing an overall improvement in the experiment design of spatial predictive modelling.

Ultimately, choosing the right CV technique strategy and carefully tuning the model to suit the use cases are essential steps in producing a robust model for spatial applications.

7. Code Availability

All codes used in this study are available online at **www.github.com/SlothKai**.

8. References

[1] S. C. Bourassa, E. Cantoni, and M. Hoesli, "Predicting house prices with spatial dependence: A comparison of alternative methods," School of Urban and Public Affairs, University of Louisville, and Department of Econometrics, University of Geneva, Louisville, KY, USA, and Geneva, Switzerland, Tech. Rep., 2023, available: email: steven.bourassa@louisville.edu, eva.cantoni@unige.ch, martin.hoesli@unige.ch.

[2] Y. Wang, K. Liu, Y. He, P. Wang, Y. Chen, H. Xue, C. Huang, and L. Li, "Enhancing air quality forecasting: A novel spatio-temporal model integrating graph convolution and multi-head attention mechanism," *Atmosphere*, vol. 15, no. 4, 2024. [Online]. Available: https://www.mdpi.com/2073-4433/15/4/418

[3] W. R. Tobler, "A computer movie simulating urban growth in the detroit region," *Economic Geography*, vol. 46, no. sup1, pp. 234–240, 1970.

[4] H. J. Miller and J. Han, Eds., *Geographic Data Mining and Knowledge Discovery*. CRC Press, 2004.

[5] P. Tziachris, M. Nikou, V. Aschonitis, A. Kallioras, K. Sachsamanoglou, M. Fidelibus, and E. Tziritis, "Spatial or random cross-validation? the effect of resampling methods in predicting groundwater salinity with machine learning in mediterranean region," *Water*, 2023.

[6] K. L. Rest, D. Pinaud, P. Monestiez, J. Chadoeuf, and V. Bretagnolle, "Spatial leave-one-out cross-validation for variable selection in the presence of spatial autocorrelation," *Global Ecology and Biogeography*, vol. 23, pp. 811–820, 2014.

[7] D. R. Roberts, V. Bahn, S. Ciuti, M. S. Boyce, J. Elith, G. Guillera-Arroita, S. Hauenstein, and et al., "Cross-validation strategies for data with temporal, spatial, hierarchical, or phylogenetic structure," *Ecography*, vol. 40, no. 8, pp. 913–929, 2017. [Online]. Available: https://doi.org/10.1111/ecog.02881

[8] D. M. Hawkins, "The problem of overfitting," *Journal of Chemical Information and Computer Sciences*, vol. 44, no. 1, pp. 1–12, 2004.

[9] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 36, no. 2, pp. 111–133, 1974.

[10] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," *International Joint Conference on Artificial Intelligence (IJCAI)*, 1995. [Online]. Available: https://www.researchgate.net/publication/2352264_A_Study_of_Cross-Validation_and_Bootstrap_for_Accuracy_Estimation_and_Model_Selection

[11] A. Sylvain and C. Alain, "A survey of cross-validation procedures for model selection," *Statistics Surveys*, vol. 4, pp. 40–79, 2010. [Online]. Available: https://doi.org/10.1214/09-SS054

[12] ScienceDirect, "Spatial autocorrelation," 2023, accessed: 2024-26-08. [Online]. Available: https://www.sciencedirect.com/topics/computer-science/spatial-autocorrelation#:~:text=The%20term%20spatial%20autocorrelation%20refers,map%20shows%20positive%20spatial%20autocorrelation.

[13] H. J. Miller, "Tobler's first law and spatial analysis," *Annals of the Association of American Geographers*, vol. 94, no. 2, pp. 284–289, 2004.

[14] J. Lelieveld, J. Evans, M. Fnais, D. Giannadaki, and A. Pozzer, "The contribution of outdoor air pollution sources to premature mortality on a global scale," *Nature*, vol. 525, pp. 367–371, 2015.

[15] A. Brenning, "Spatial cross-validation and bootstrap for the assessment of prediction rules in remote sensing: The r package sperrorest," in *2012 IEEE International Geoscience and Remote Sensing Symposium*, 2012, pp. 5372–5375.

[16] Y. Dong, F. Peng, H. Li, and Y. Men, "Spatial autocorrelation and spatial heterogeneity of underground parking space development in chinese megacities based on multisource open data," *Applied Geography*, 2023.

[17] A. S. Fotheringham, C. Brunsdon, and M. Charlton, *Geographically Weighted Regression: The Analysis of Spatially Varying Relationships*. Wiley, 2002.

[18] A. M.-C. Wadoux, G. B. Heuvelink, S. de Bruin, and D. J. Brus, "Spatial cross-validation is not the right way to evaluate map accuracy," *Ecological Modelling*, vol. 457, p. 11, 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0304380021002489

[19] K. Technology. (2024) Cross-validation in machine learning: Techniques and applications. Accessed: Oct. 20, 2024. [Online]. Available: https://kili-technology.com/data-labeling/machine-learning/cross-validation-in-machine-learning

[20] STHDA, "Bootstrap resampling essentials in r," 2024, accessed: 2024-10-20. [Online]. Available: http://www.sthda.com/english/articles/38-regression-model-validation/156-bootstrap-resampling-essentials-in-r/

[21] S. Raschka. (2023) Bootstrap and out-of-bag (oob) estimate of performance. Accessed: Oct. 20, 2024. [Online]. Available: https://rasbt.github.io/mlxtend/user_guide/evaluate/BootstrapOutOfBag/

[22] Y. Yamada, T. Suzuki, T. Kanamori, and M. Sugiyama, "Relative density-ratio estimation for robust distribution comparison," in *Advances in Neural Information Processing Systems*, vol. 24. Neural Information Processing Systems Foundation, 2011, pp. 594–602.

[23] M. Sugiyama, T. Suzuki, and T. Kanamori, *Density Ratio Estimation in Machine Learning*. Cambridge University Press, 03 2012.

[24] S. N. R. (2019) Spatial autocorrelation in r. Accessed: Oct. 15, 2024. [Online]. Available: https://doodles.mountainmath.ca/posts/2019-10-07-spatial-autocorrelation-co/

[25] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[26] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of machine learning research*, vol. 15, pp. 1929–1958, 2014.

[27] M. N. Wright and A. Ziegler, *ranger: A Fast Implementation of Random Forests*, 2023, r package version 0.15.1. [Online]. Available: https://CRAN.R-project.org/package=ranger

[28] M. J. Mahoney, L. K. Johnson, J. Silge, H. Frick, M. Kuhn, and C. M. Beier, "Assessing the performance of spatial cross-validation approaches for models of spatially structured data," *arXiv preprint arXiv:2303.07334*, 2023. [Online]. Available: https://arxiv.org/abs/2303.07334

[29] M. Kuhn, *caret: Classification and Regression Training*, 2023, r package version 6.0-94. [Online]. Available: https://topepo.github.io/caret/index.html

[30] M. N. Wright and A. Ziegler, "ranger: A fast implementation of random forests for high dimensional data in c++ and r," *Journal of Statistical Software*, vol. 077, pp. 1–17, 2015.

[31] R. Caruana and A. Niculescu-Mizil, "An empirical comparison of supervised learning algorithms," in *Proceedings of the 23rd international conference on Machine learning*, 2006, pp. 161–168.

[32] J. Frost. (2024) Root mean square error (rmse). Accessed: 2024-08-26. [Online]. Available: https://statisticsbyjim.com/regression/root-mean-square-error-rmse/

[33] Deepchecks, "Mean absolute error," n.d., accessed: 2024-08-26. [Online]. Available: https://deepchecks.com/glossary/mean-absolute-error/

[34] Scribbr. (n.d.) Coefficient of determination — r-squared. Accessed: 2024-08-26. [Online]. Available: https://www.scribbr.com/statistics/coefficient-of-determination/

[35] B. Saremi, M. Kohls, P. Liebig, U. Siebert, and K. Jung, "Measuring reproducibility of virus metagenomics analyses using bootstrap samples from fastq-files," *Bioinformatics*, 2020.

[36] Y. Wang, Z. Zhu, and X. He, "Reboot: Distributed statistical learning via refitting bootstrap samples," 2022.

[37] C. Bergmeir and J. M. Benítez, "On the use of cross-validation for time series predictor evaluation," *Information Sciences*, vol. 191, pp. 192–213, 2012. [Online]. Available: https://doi.org/10.1016/j.ins.2011.12.028

[38] M. Sugiyama, M. Krauledat, and K.-R. Müller, "Covariate shift adaptation by importance weighted cross-validation," *Journal of Machine Learning Research*, vol. 8, pp. 985–1005, 2008.

[39] P. Schratz *et al.*, "Hyperparameter tuning and performance assessment of spatial models using spatial cross-validation," *International Journal of Geographical Information Science*, vol. 33, no. 10, pp. 2135–2155, 2019.

9. Appendix

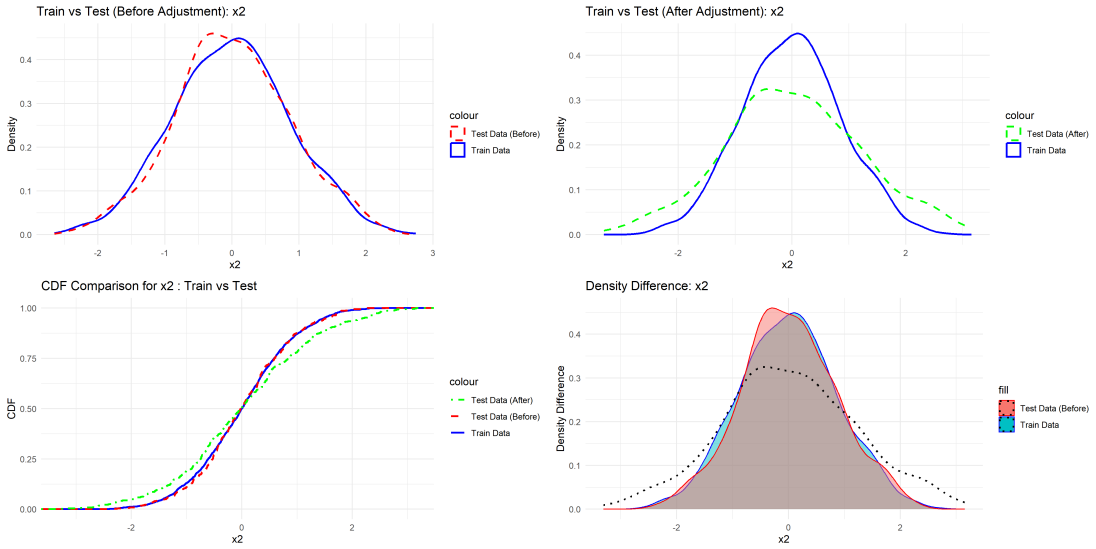


Figure 9.0.1: An example of inducing CS for the variable X_2 for scenario SD.

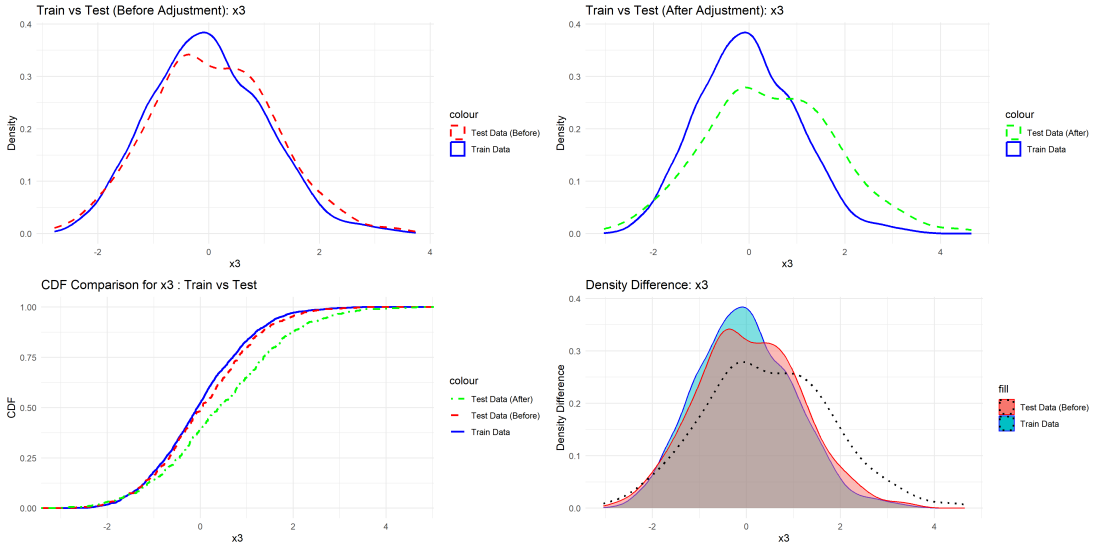


Figure 9.0.2: An example of inducing CS for the variable X_3 for scenario SD.

k / B	Technique	Test			External		
		Min	**Median**	**Max**	**Min**	**Median**	**Max**
2 / 50	Random K	0.223	0.279	0.389	0.319	0.491	0.740
	BootCV	0.191	0.274	0.393	0.308	0.490	0.734
	Spatial K	0.354	0.571	1.820	0.282	0.471	0.727
	BlockCV	0.093	0.132	0.178	0.279	0.472	0.736
	BuffCV	**0.143**	**0.381**	**3.505**	0.281	0.465	0.702
	IWCV	0.252	0.321	0.430	0.384	0.548	0.819
5 / 75	Random K	0.222	0.279	0.393	0.317	0.491	0.742
	BootCV	0.196	0.273	0.399	0.312	0.488	0.750
	Spatial K	0.306	0.389	1.470	0.281	0.473	0.728
	BlockCV	0.073	0.103	0.147	0.280	0.474	0.733
	BuffCV	0.092	0.117	0.266	0.280	0.461	0.706
	IWCV	0.214	0.275	0.346	0.329	0.498	0.745
10 / 100	Random K	0.223	0.279	0.390	0.311	0.491	0.742
	BootCV	0.193	0.276	0.403	0.315	0.486	0.742
	Spatial K	0.253	0.345	0.744	0.281	0.472	0.728
	BlockCV	0.066	0.097	0.158	0.278	0.462	0.707
	BuffCV	0.080	0.106	0.159	0.283	0.472	0.734
	IWCV	0.211	0.267	0.329	0.318	0.488	0.733

Table 9.0.1: Summary of the minimum, median, and maximum RMSE values across different folds/bootstrap samples k/B and techniques for both test and external datasets for SD. Techniques include Random K, BootCV, Spatial K, BlockCV, BuffCV, and IWCV. BuffCV $k = 2$ exhibits notably high variability in RMSE values in both test and external datasets.

k / B	Technique	Test			External		
		Min	Median	Max	Min	Median	Max
2 / 50	Random K	0.160	0.179	0.223	0.190	0.297	0.479
	BootCV	0.144	0.178	0.222	0.187	0.297	0.472
	Spatial K	0.216	0.350	0.905	0.171	0.283	0.478
	BlockCV	0.066	0.083	0.117	0.169	0.283	0.482
	BuffCV	**0.085**	**0.233**	**3.218**	0.172	0.278	0.430
	IWCV	0.177	0.217	0.240	0.231	0.338	0.524
5 / 75	Random K	0.161	0.181	0.219	0.189	0.298	0.480
	BootCV	0.148	0.179	0.221	0.189	0.296	0.488
	Spatial K	0.186	0.256	0.864	0.173	0.283	0.477
	BlockCV	0.053	0.067	0.082	0.172	0.282	0.480
	BuffCV	0.062	0.073	0.180	0.171	0.278	0.433
	IWCV	0.151	0.186	0.206	0.195	0.304	0.472
10 / 100	Random K	0.160	0.180	0.223	0.180	0.300	0.481
	BootCV	0.145	0.178	0.221	0.189	0.296	0.484
	Spatial K	0.174	0.227	0.496	0.173	0.283	0.472
	BlockCV	0.049	0.064	0.084	0.173	0.283	0.480
	BuffCV	0.057	0.067	0.094	0.173	0.277	0.433
	IWCV	0.150	0.182	0.195	0.186	0.295	0.451

Table 9.0.2: Summary of the minimum, median, and maximum MAE values across different folds/bootstrap samples k/B and techniques for both test and external datasets for SD. Techniques include Random K, BootCV, Spatial K, BlockCV, BuffCV, and IWCV. BuffCV $k = 2$ exhibits notably high variability in MAE values in both test and external datasets.

k / B	Technique	Test			External		
		Min	**Median**	**Max**	**Min**	**Median**	**Max**
2 / 50	Random K	0.959	0.977	0.988	0.740	0.948	0.978
	BootCV	0.959	0.978	0.989	0.744	0.944	0.979
	Spatial K	0.194	0.906	0.969	0.749	0.948	0.982
	BlockCV	0.985	0.995	0.998	0.743	0.949	0.983
	BuffCV	**-3.111**	**0.959**	**0.995**	0.766	0.956	0.983
	IWCV	0.940	0.968	0.981	0.682	0.931	0.967
5 / 75	Random K	0.959	0.977	0.988	0.739	0.948	0.978
	BootCV	0.958	0.978	0.989	0.733	0.944	0.979
	Spatial K	0.115	0.947	0.976	0.733	0.944	0.979
	BlockCV	0.993	0.997	0.998	0.745	0.948	0.983
	BuffCV	0.966	0.996	0.998	0.764	0.956	0.983
	IWCV	0.954	0.977	0.985	0.737	0.946	0.976
10 / 100	Random K	0.959	0.977	0.988	0.739	0.948	0.979
	BootCV	0.958	0.978	0.989	0.738	0.944	0.978
	Spatial K	0.487	0.944	0.979	0.751	0.949	0.983
	BlockCV	0.994	0.997	0.998	0.745	0.948	0.983
	BuffCV	0.994	0.997	0.998	0.763	0.956	0.983
	IWCV	0.958	0.978	0.986	0.746	0.948	0.978

Table 9.0.3: Summary of the minimum, median, and maximum R^2 values across different folds/bootstrap samples k/B and techniques for both test and external datasets for SD. Techniques include Random K, BootCV, Spatial K, BlockCV, BuffCV, and IWCV. BuffCV $k = 2$ exhibits notably high variability in MAE values in both test and external datasets.

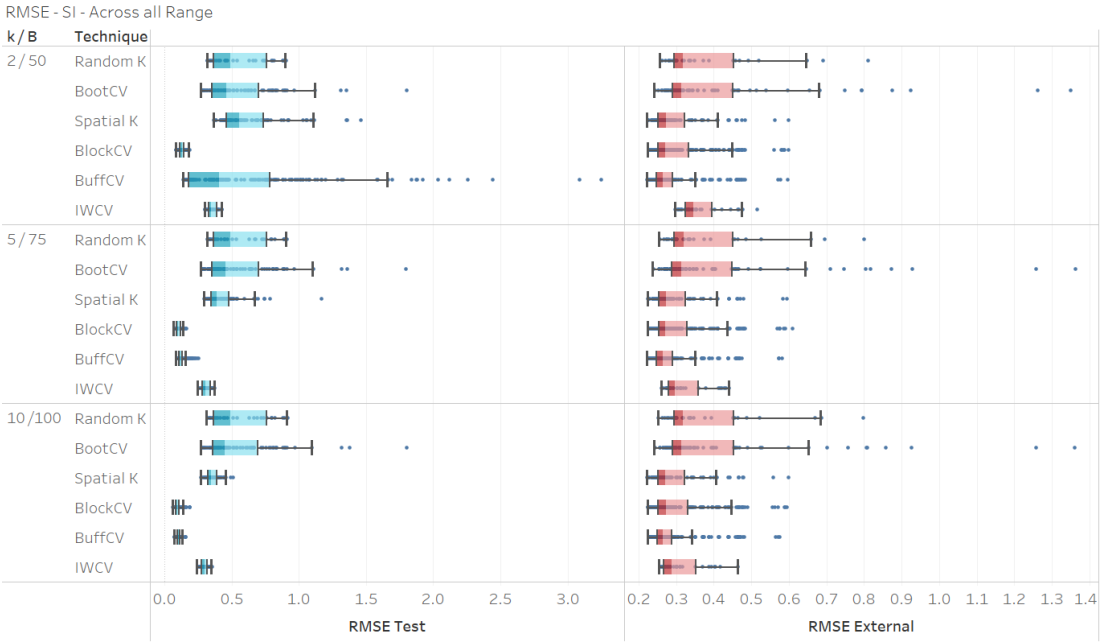


Figure 9.0.3: RMSE distribution for Spatial Independence (SI) scenario across different cross-validation techniques and folds/bootstrap samples (k / B). The boxplot highlights the relatively low error across all techniques (BootCV, RKFCV, IWCV, SKFCV, BuffCV, and BCV), with minimal variation between the test (blue) and the external (red) datasets. BuffCV shows a slight increase in variance, particularly when $k = 2$, which may be attributed to block size mismatch.

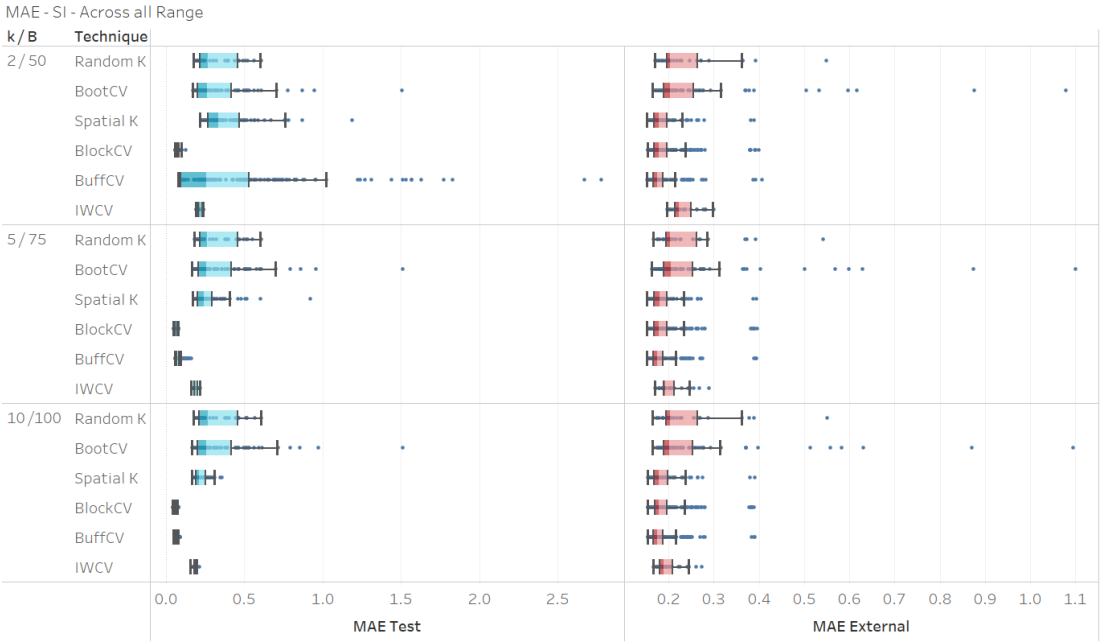


Figure 9.0.4: MAE distribution for Spatial Independence (SI) scenario across different cross-validation techniques and folds/bootstrap samples (k / B). The boxplot shows consistently low mean absolute errors across techniques (BootCV, RKFCV, IWCV, SKFCV, BuffCV, and BCV). BuffCV displays higher variance in when $k = 2$, likely due to block size mismatch, while the other techniques exhibit minimal variation between the test (blue) and the external (red) datasets.

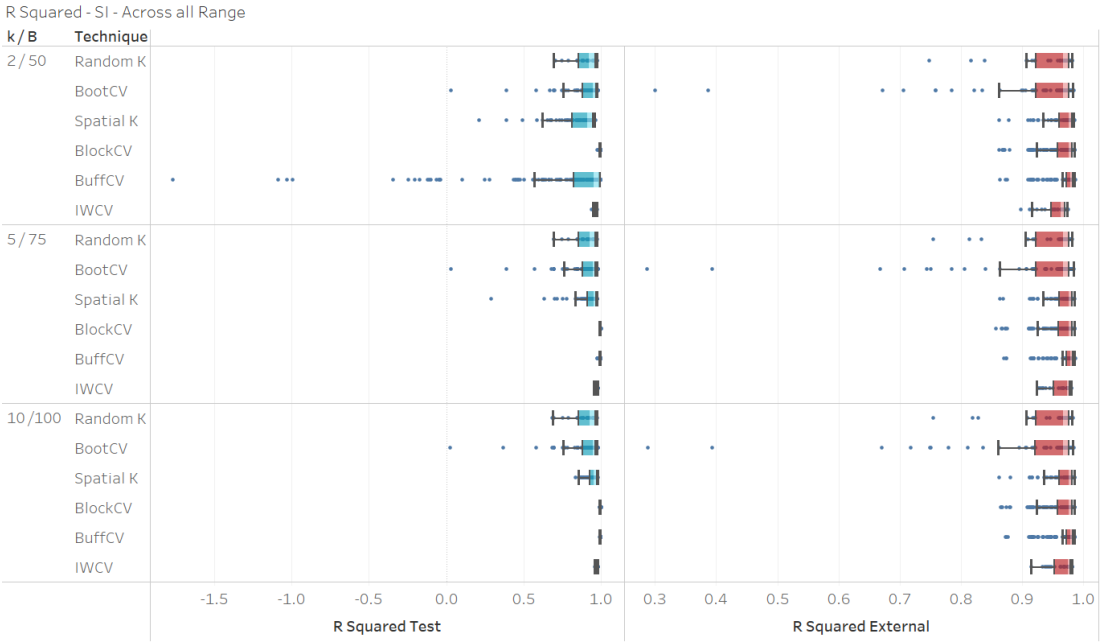


Figure 9.0.5: R^2 distribution for Spatial Independence (SI) scenario across different cross-validation techniques and folds/bootstrap samples (k / B). The boxplot demonstrates consistently high R^2 values across all techniques (BootCV, RKFCV, IWCV, SKFCV, BuffCV, and BCV), indicating strong predictive performance. Minor variance is observed, with most techniques showing values close to 1, while BuffCV exhibits slightly lower R^2 values when $k = 2$, potentially due to block size mismatch.

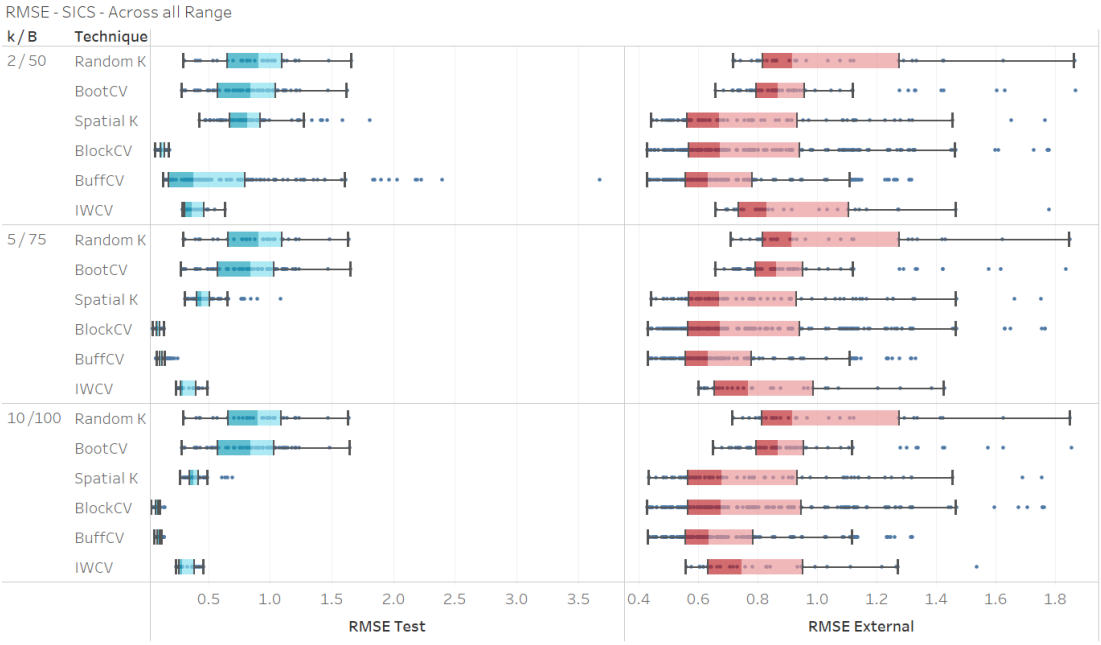


Figure 9.0.6: RMSE distribution for Spatial Independence with Covariate Shift (SICS) scenario across cross-validation techniques and folds/bootstrap samples (k / B). While error values remain relatively low, BootCV and RKFCV exhibit higher RMSE under covariate shift conditions, indicating a sensitivity to distribution changes. IWCV performs more consistently across test and external datasets.

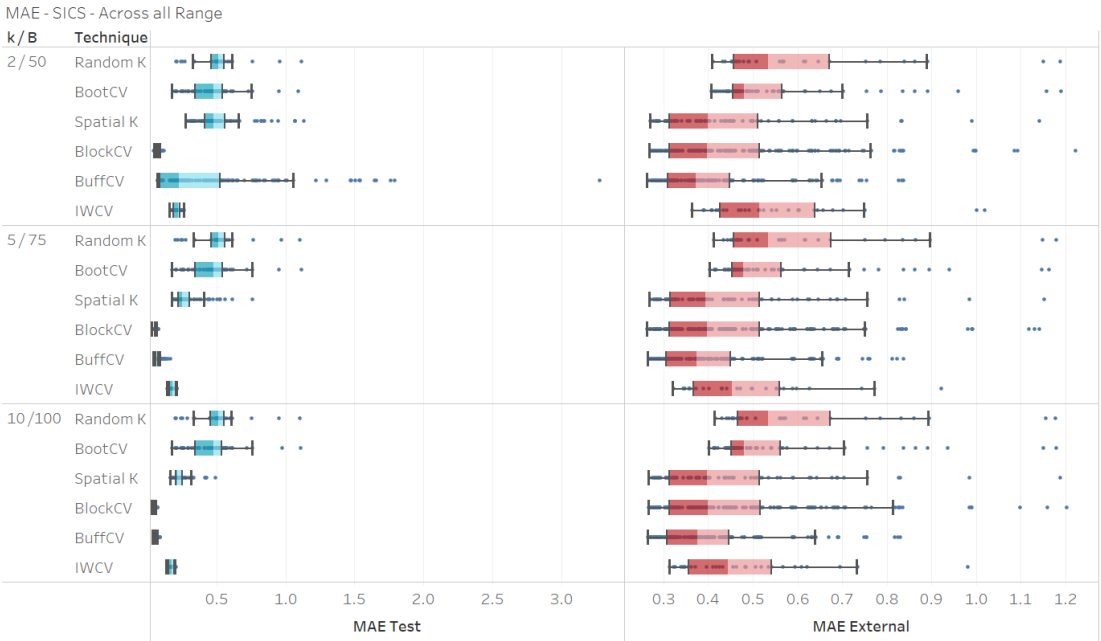


Figure 9.0.7: MAE distribution for Spatial Independence with Covariate Shift (SICS) scenario across different cross-validation techniques and folds/bootstrap samples (k / B). The impact of covariate shift is evident across techniques, with BootCV and BuffCV showing higher errors. IWCV and SKFCV performed better, with less pronounced differences between test and external datasets.

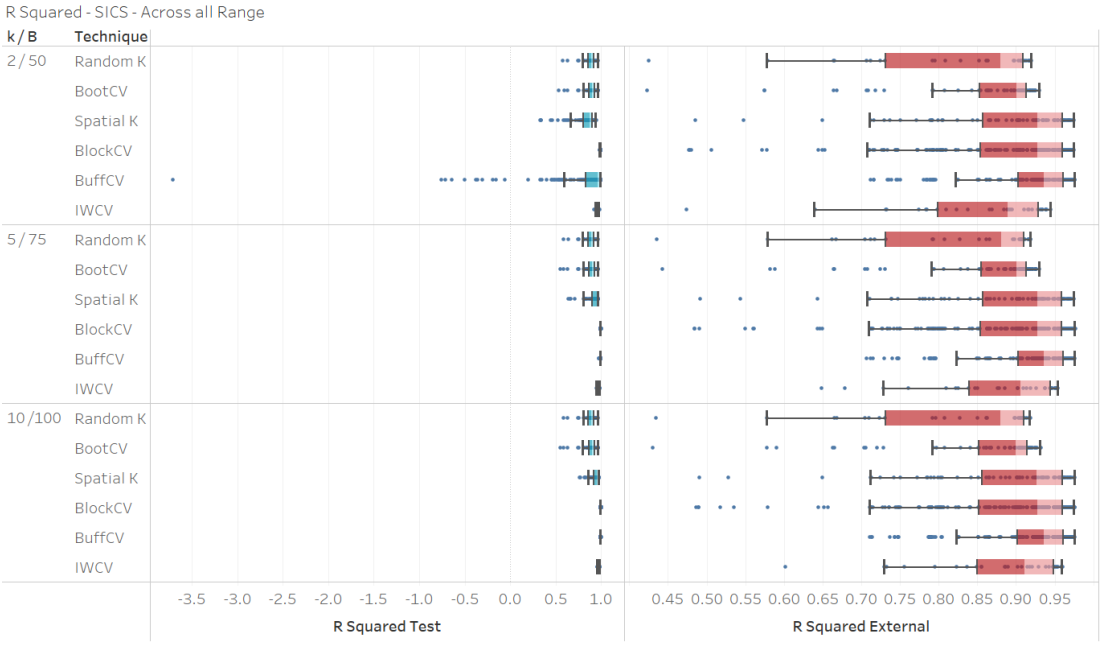


Figure 9.0.8: R^2 distribution for Spatial Independence with Covariate Shift (SICS) scenario across different cross-validation techniques and folds/bootstrap samples (k / B). IWCV and SKFCV maintain relatively high R^2 values, while BootCV and RKFCV show more variation and lower predictive accuracy under covariate shift conditions. Covariate shift appears to affect all techniques, though IWCV shows the least impact.

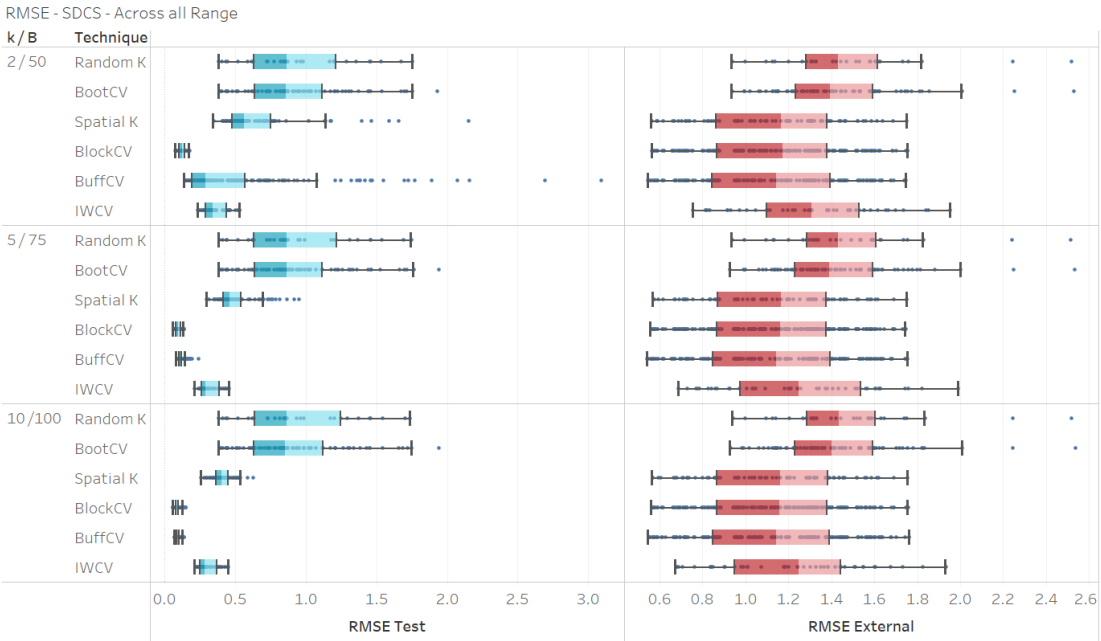


Figure 9.0.9: RMSE distribution for Spatial Dependence with Covariate Shift (SDCS) scenario across different cross-validation techniques and folds. A marked increase in RMSE values is observed for all techniques, with RKFCV and BootCV showing the highest error. Covariate shift amplifies the disparity between test and external datasets, particularly in higher folds.

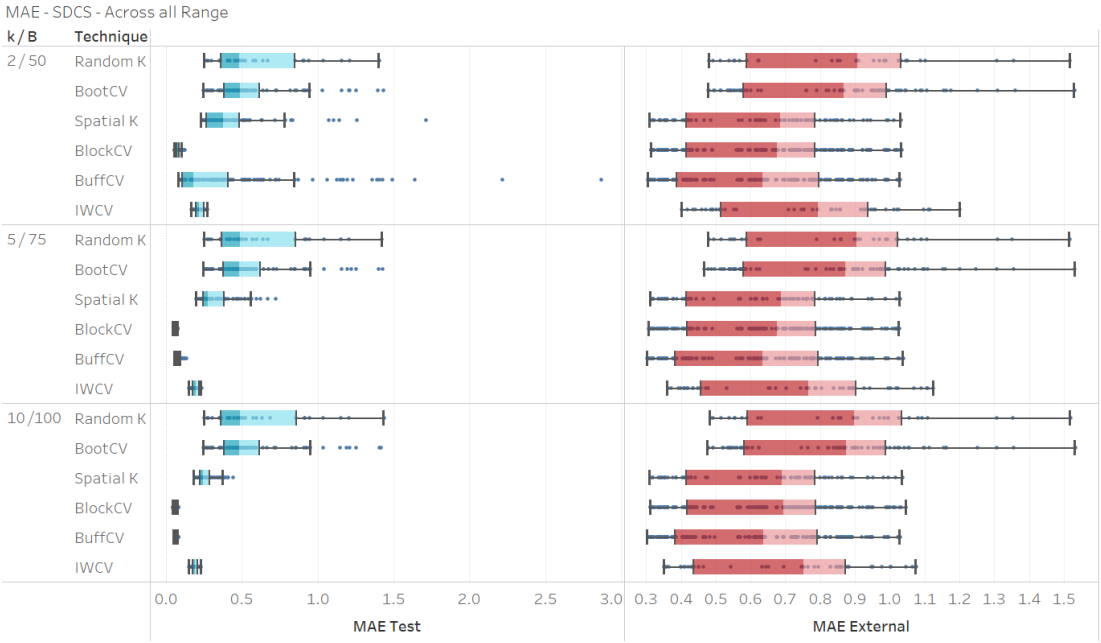


Figure 9.0.10: MAE distribution for Spatial Dependence with Covariate Shift (SDCS) scenario across different cross-validation techniques and folds/bootstrap samples (k / B). All techniques exhibit significant increases in MAE due to covariate shift, with BootCV and RKFCV showing the highest variability in test and external datasets. IWCV remains relatively stable, handling distribution shifts better than the other techniques.

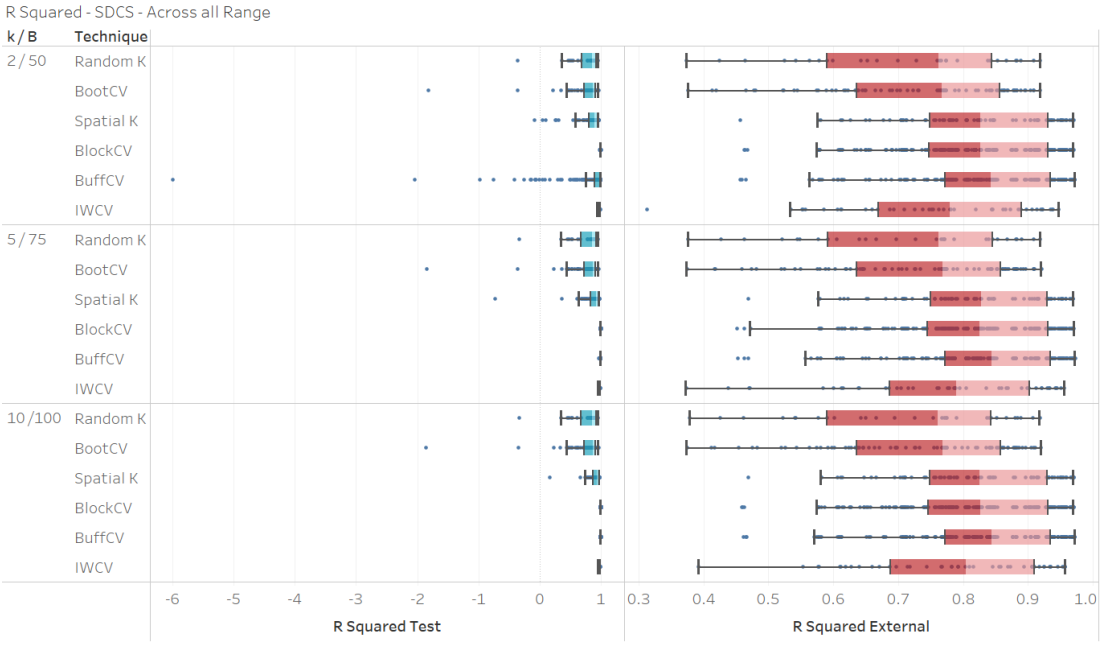


Figure 9.0.11: R^2 distribution for Spatial Dependence with Covariate Shift (SDCS) scenario across different cross-validation techniques and folds/bootstrap samples (k / B). The introduction of covariate shift leads to notable drops in R^2 values for all techniques, particularly in RKFCV and BootCV. IWCV and BuffCV demonstrated more stable performance, though their R^2 values still decrease under these challenging conditions.

Figure 9.0.12: BootCV $r = 1$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and different resample levels (50,75,100). Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

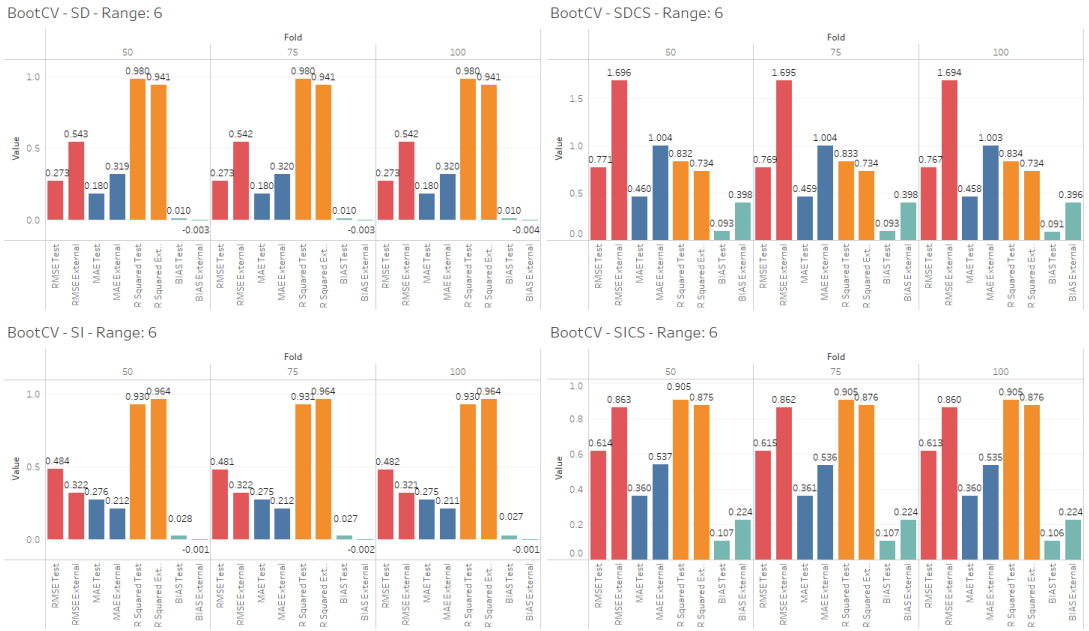


Figure 9.0.13: BootCV $r = 6$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and different resample levels (50,75,100). Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

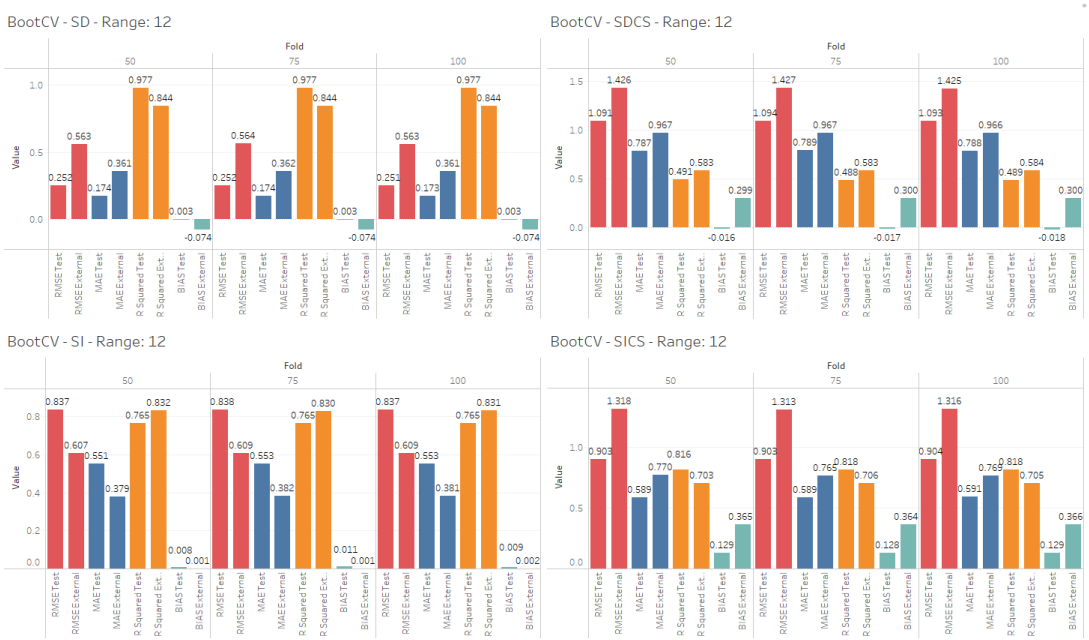


Figure 9.0.14: BootCV $r = 12$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and different resample levels (50,75,100). Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

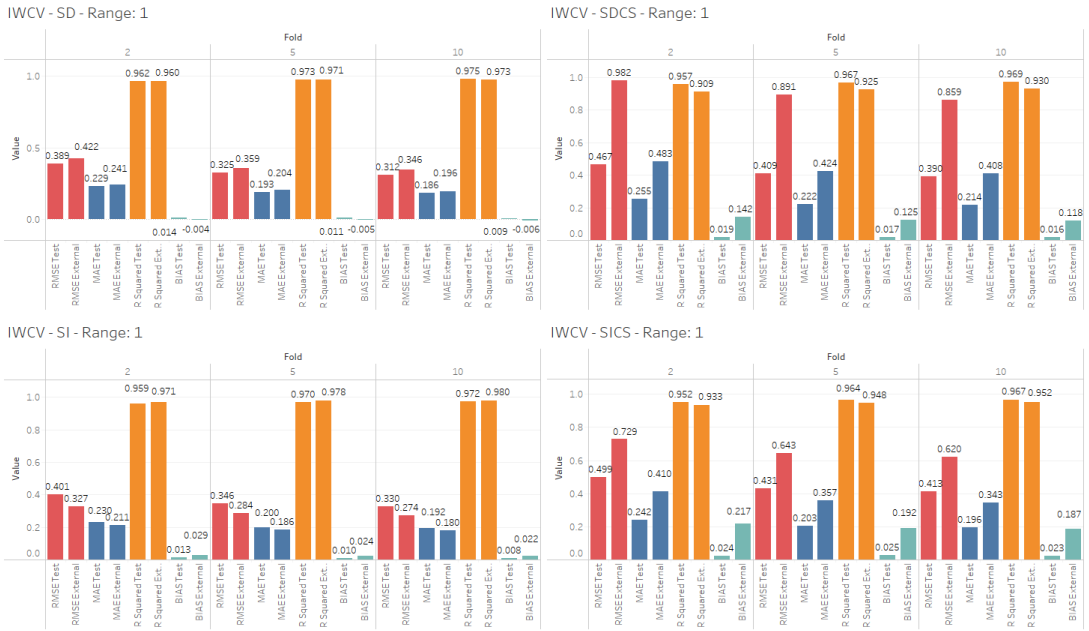


Figure 9.0.15: IWCV $r = 1$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

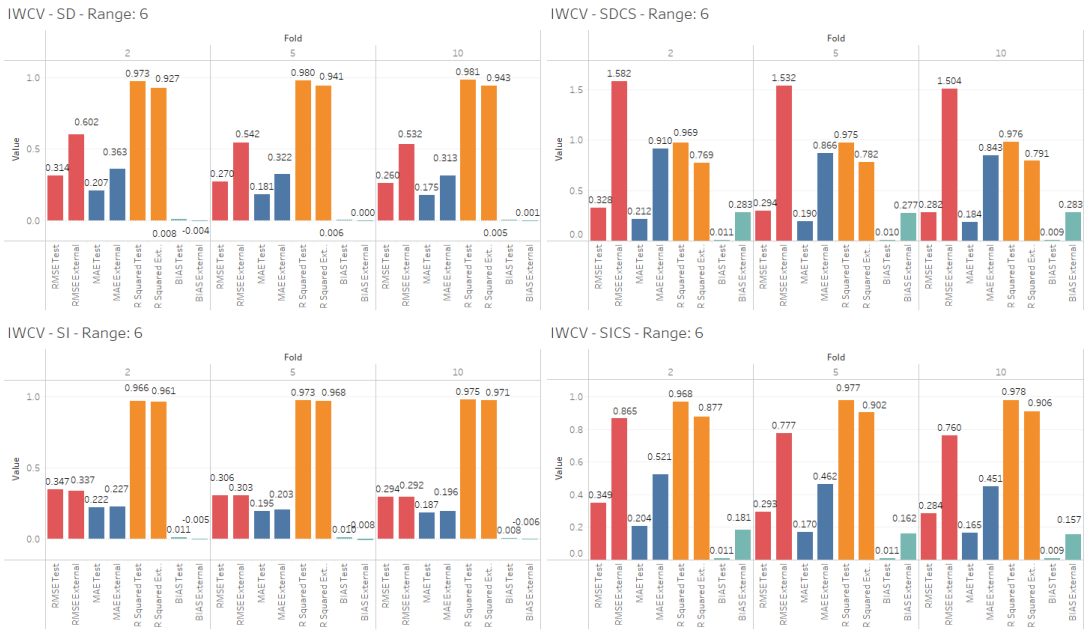


Figure 9.0.16: IWCV $r = 6$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

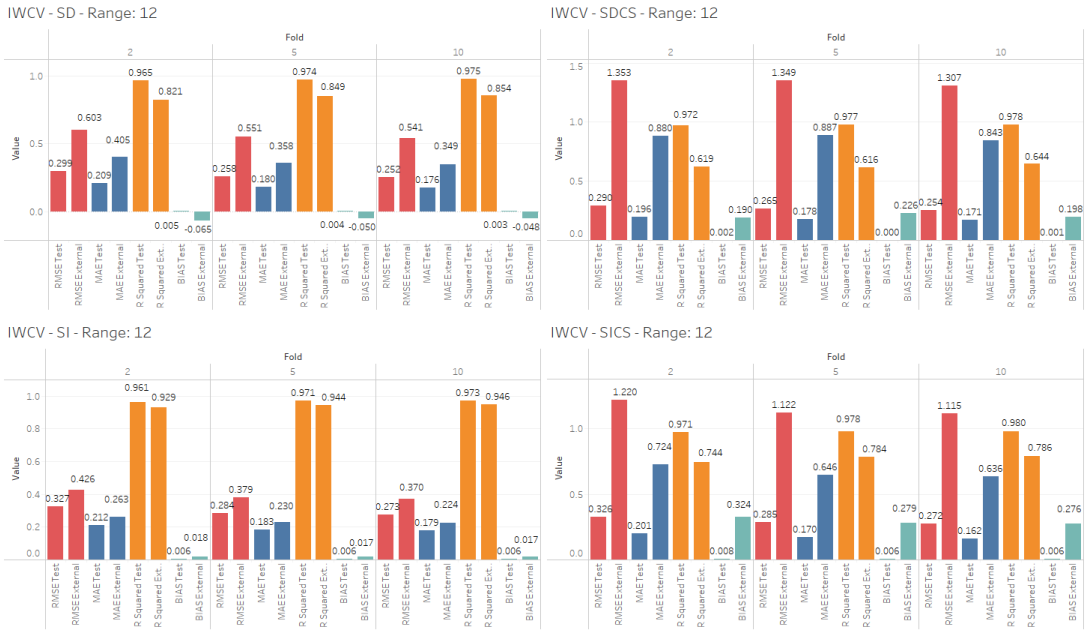


Figure 9.0.17: IWCV $r = 12$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and different k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

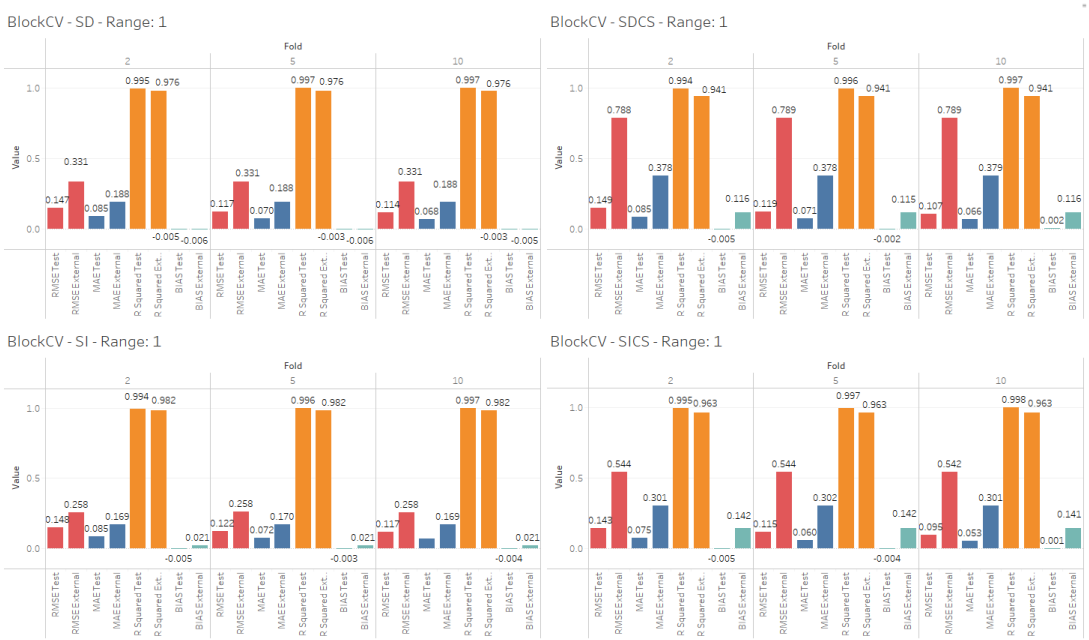


Figure 9.0.18: BCV $r = 1$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

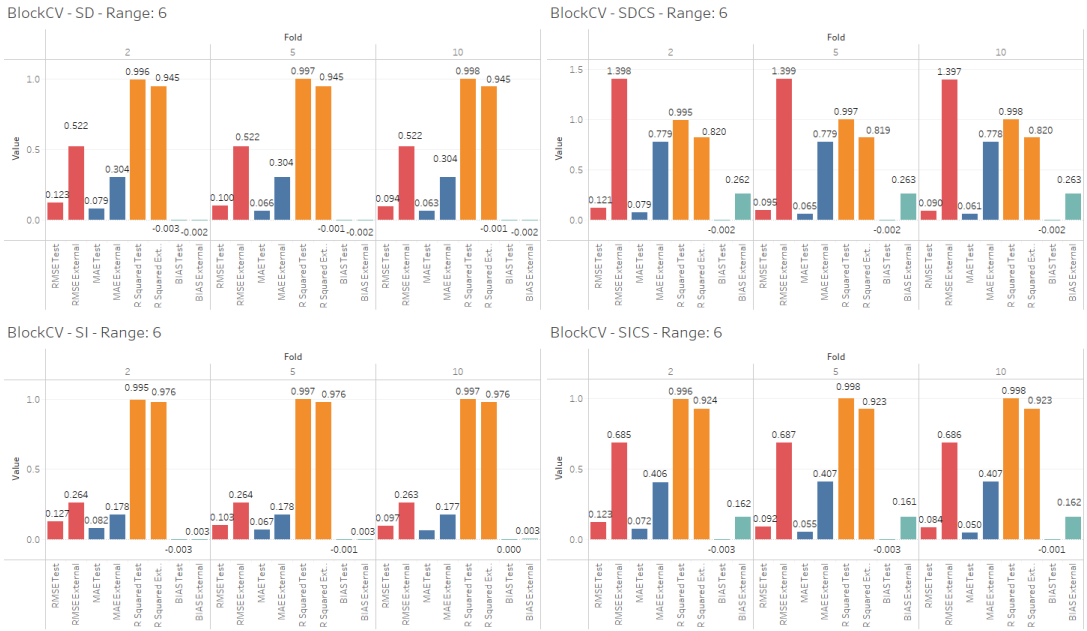


Figure 9.0.19: BCV $r = 6$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

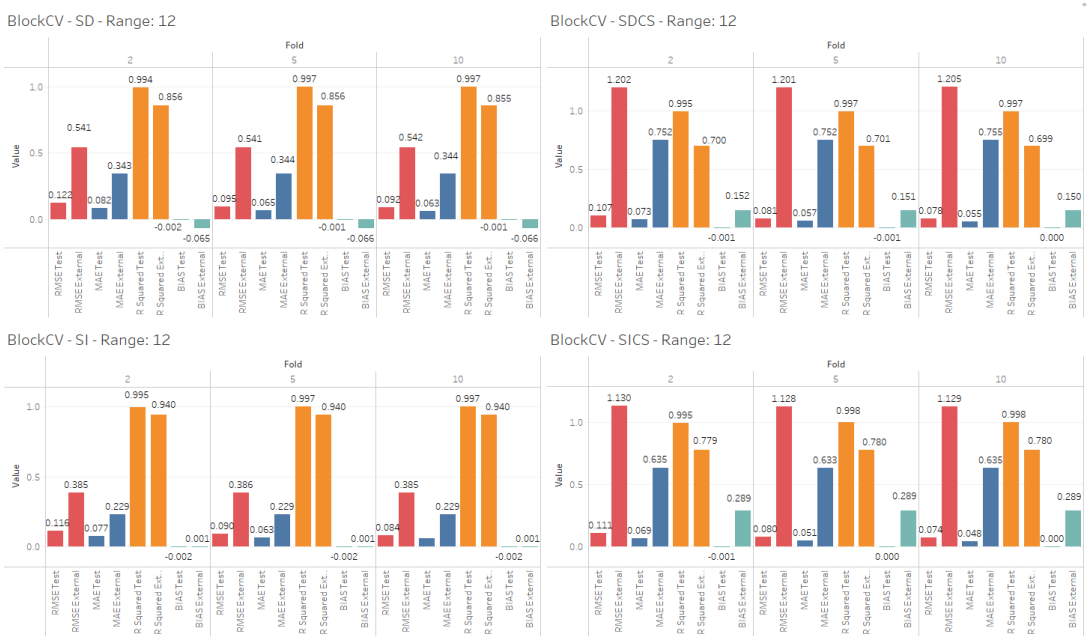


Figure 9.0.20: BCV $r = 12$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and different k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

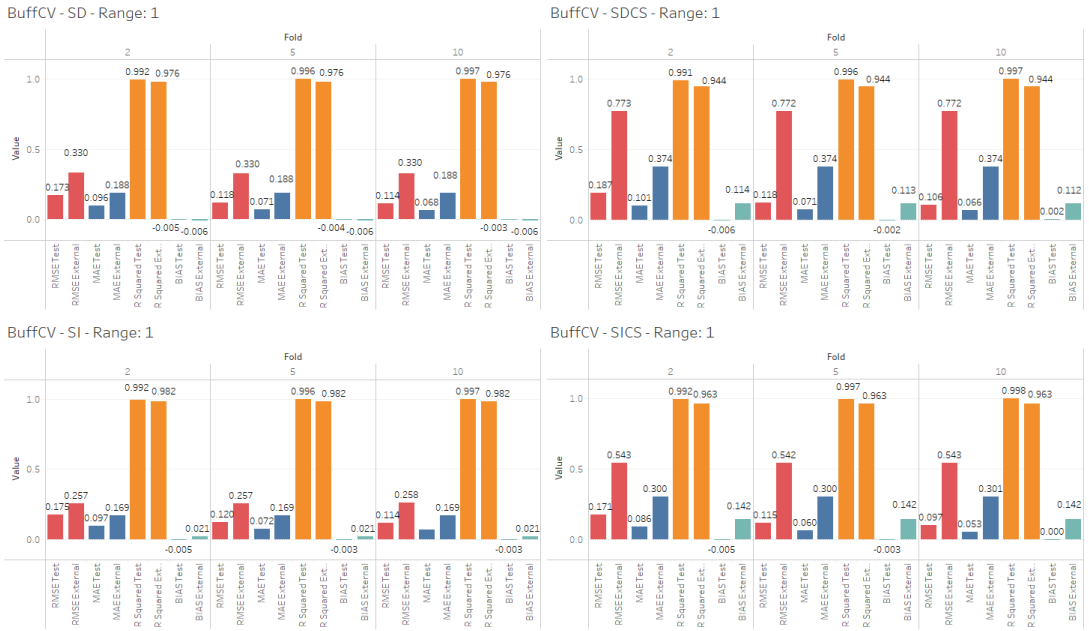


Figure 9.0.21: BuffCV $r = 1$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

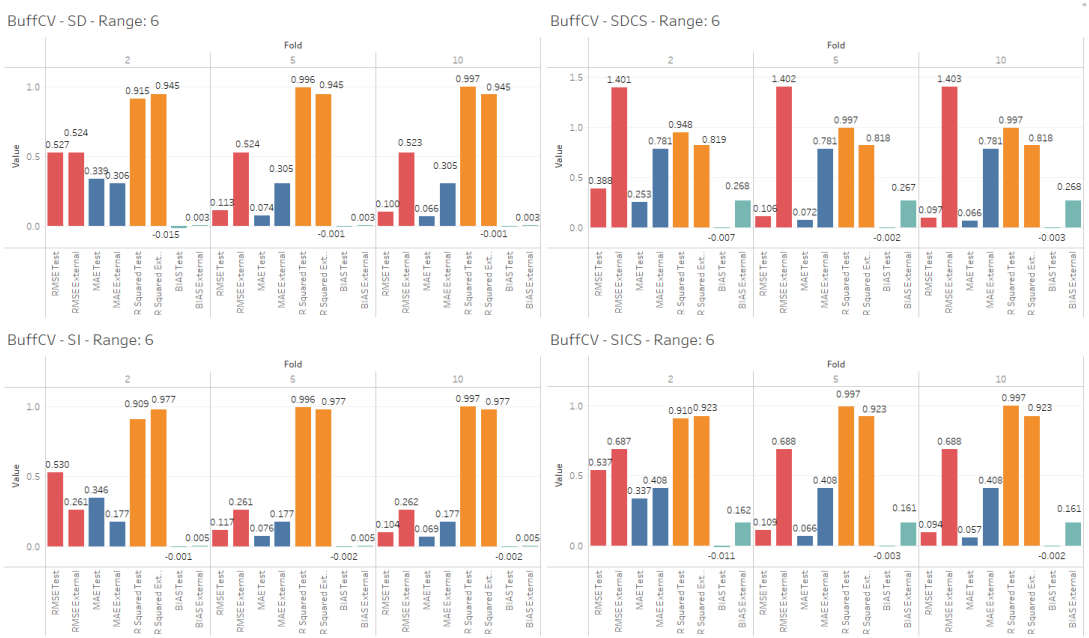


Figure 9.0.22: BuffCV $r = 6$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

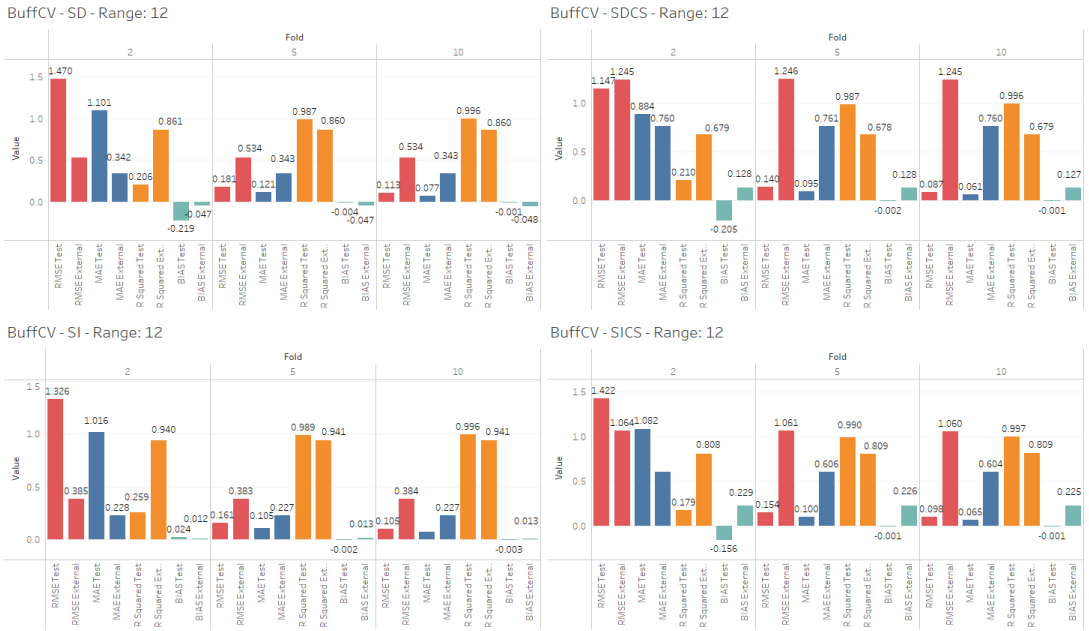


Figure 9.0.23: BuffCV $r = 12$: Comparison of performance between RMSE, MAE, R^2 and Bias across varying r $1, 6, 12$, scenarios (SD, SDCS, SI, SICS) and different k folds $2, 5, 10$. Performance metrics reports both test dataset and external dataset. The averages of the metrics were taken.

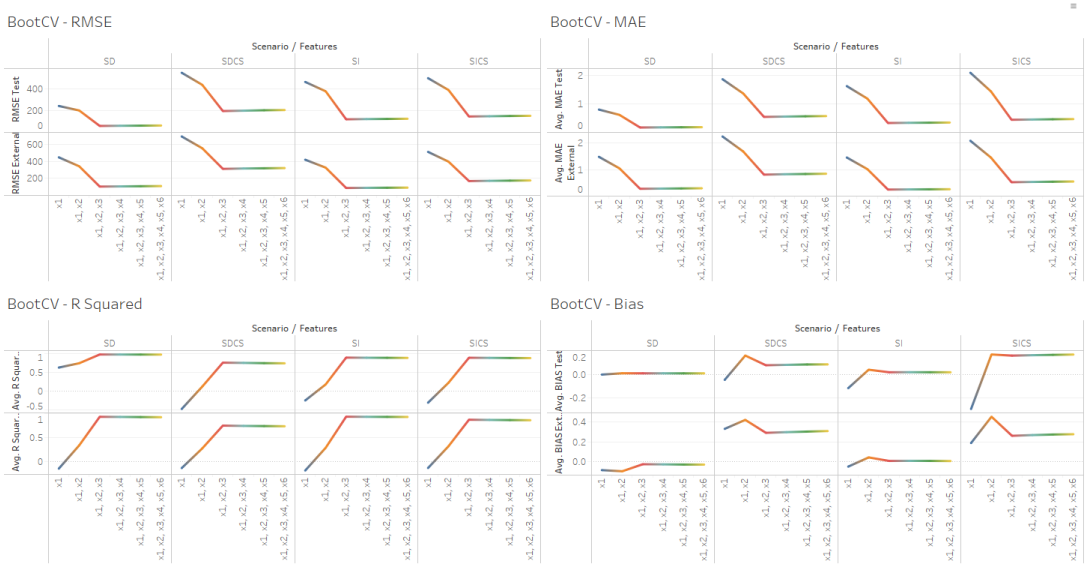


Figure 9.0.24: Impact of Covariates and Noise Variables on Bootstrap Cross-Validation Performance Across Different Scenarios (SD, SDCS, SI, and SICS): Adding noise variables X_4, X_5, X_6 introduces only slight changes in performance metrics, with covariates X_1, X_2, X_3 having the most significant impact.

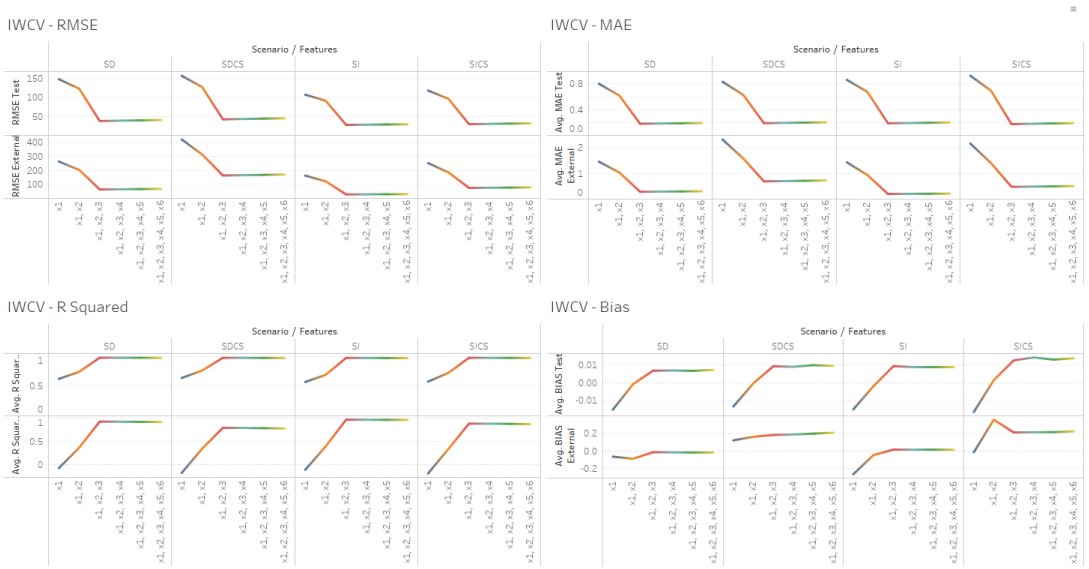


Figure 9.0.25: Impact of Covariates and Noise Variables on Importance Weighted Cross-Validation Performance Across Different Scenarios (SD, SDCS, SI, and SICS): Adding noise variables X_4, X_5, X_6 introduces only slight changes in performance metrics, with covariates X_1, X_2, X_3 having the most significant impact.

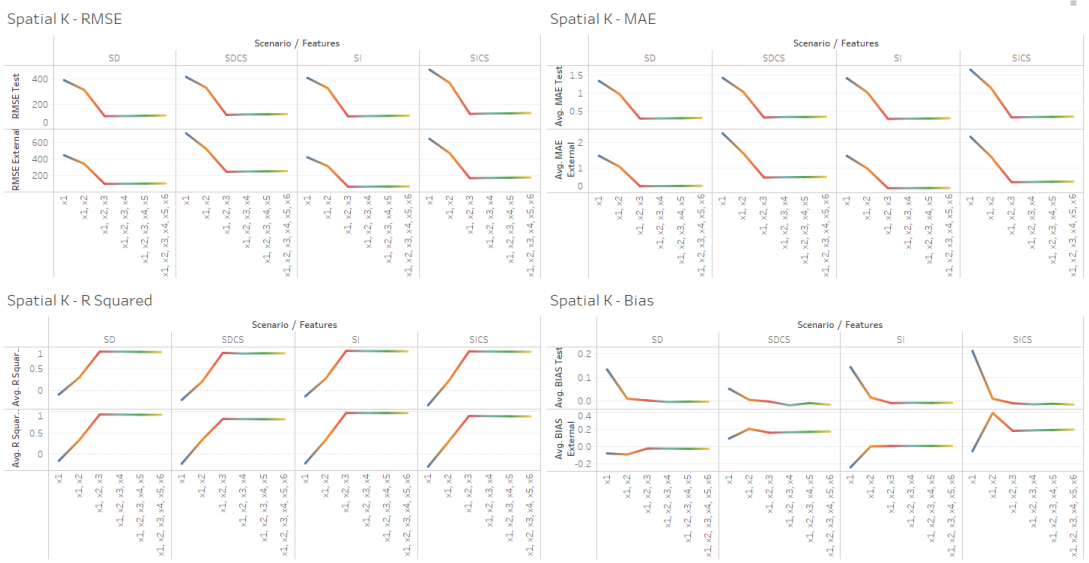


Figure 9.0.26: Impact of Covariates and Noise Variables on Spatial K-Fold Cross-Validation Performance Across Different Scenarios (SD, SDCS, SI, and SICS): Adding noise variables X_4, X_5, X_6 introduces only slight changes in performance metrics, with covariates X_1, X_2, X_3 having the most significant impact.

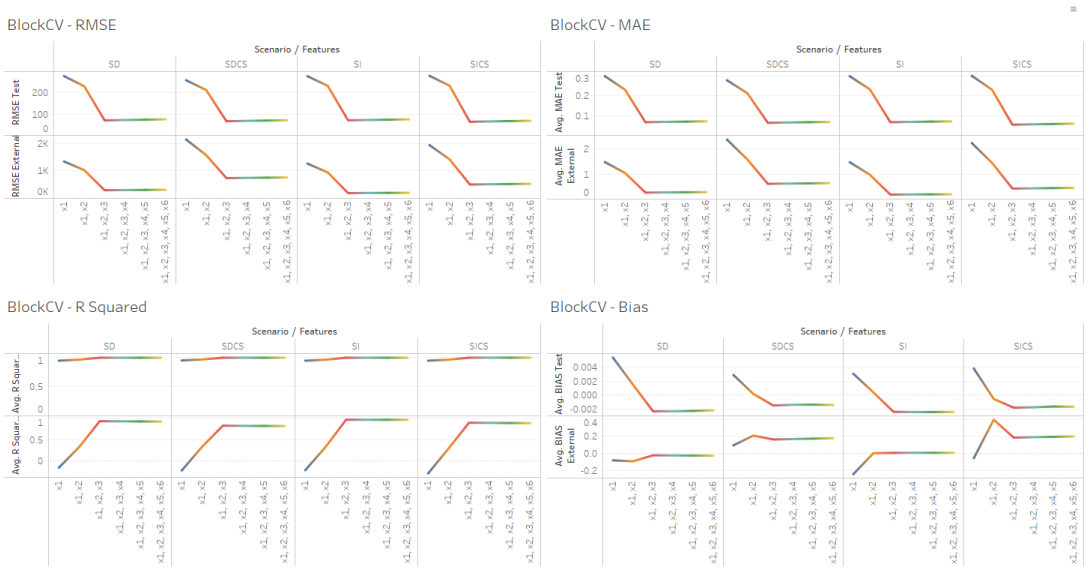


Figure 9.0.27: Impact of Covariates and Noise Variables on Blocked Cross-Validation Performance Across Different Scenarios (SD, SDCS, SI, and SICS): Adding noise variables X_4, X_5, X_6 introduces only slight changes in performance metrics, with covariates X_1, X_2, X_3 having the most significant impact.

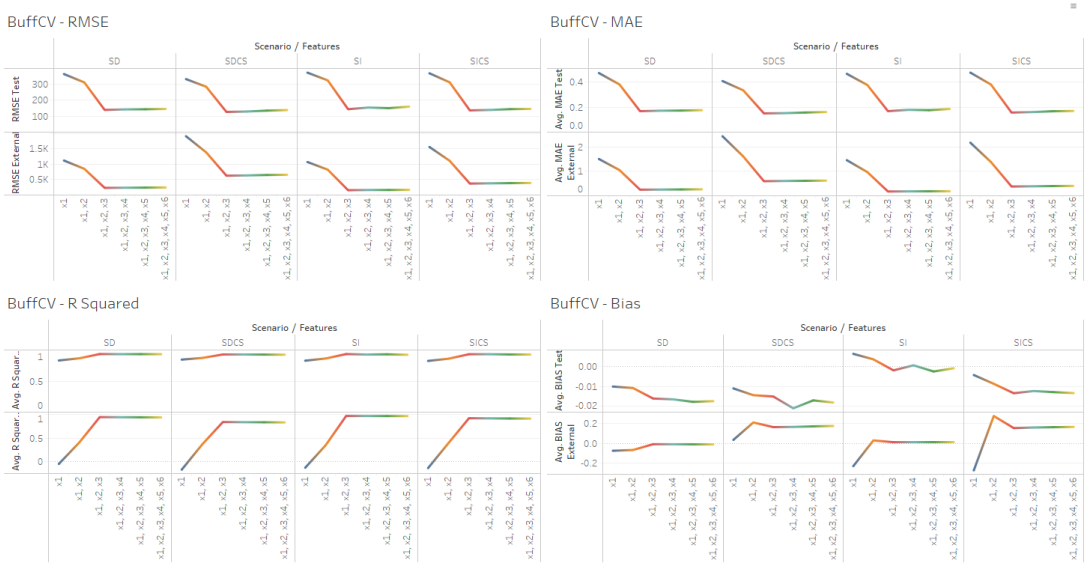


Figure 9.0.28: Impact of Covariates and Noise Variables on Buffered Cross-Validation Performance Across Different Scenarios (SD, SDCS, SI, and SICS): The addition of noise variables X_4, X_5, X_6 causes minor changes in performance metrics, with covariates having a more noticeable impact on model performance.