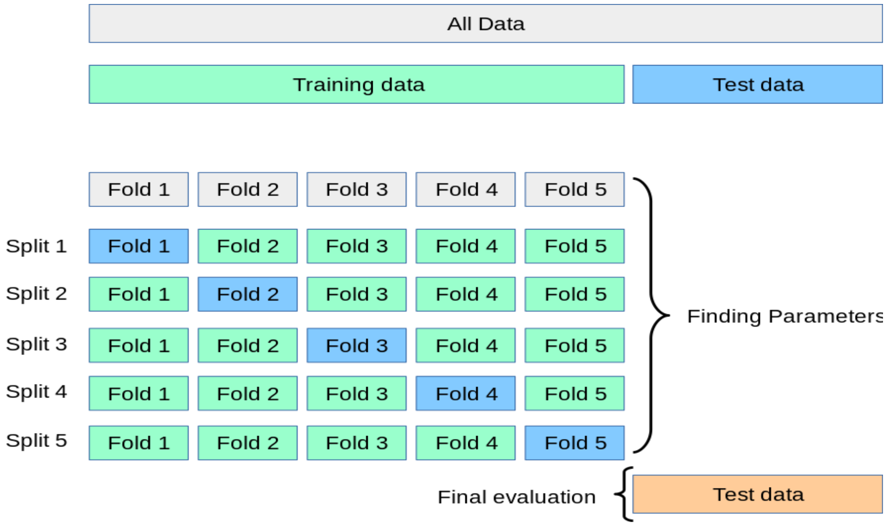
CROSSVALIDATION TECHNIQUES INFORMATION MANUAL

1.K-Fold Cross-Validation:

k-Fold cross-validation is a technique that minimizes the disadvantages of the hold-out method. k-Fold introduces a new way of splitting the dataset which helps to overcome the "test only once bottleneck".

The algorithm of the k-Fold technique:

1. Pick a number of folds – k. Usually, k is 5 or 10 but you can choose any number which is less than the dataset's length.
2. Split the dataset into k equal (if possible) parts (they are called folds)
3. Choose k – 1 folds as the training set. The remaining fold will be the test set
4. Train the model on the training set. On each iteration of cross-validation, you must train a new model independently of the model trained on the previous iteration
5. Validate on the test set
6. Save the result of the validation
7. Repeat steps 3 – 6 k times. Each time use the remaining fold as the test set. In the end, you should have validated the model on every fold that you have.
8. To get the final score average the results that you got on step 6.



Explanation: K-Fold Cross-Validation involves partitioning the dataset into k equally sized folds. Each fold is used as the testing set once while the remaining k-1 folds serve as the training set. The process is repeated k times, and the performance metrics are averaged.

CROSSVALIDATION TECHNIQUES INFORMATION MANUAL

Application: Effective for general-purpose model evaluation. Helps to assess model performance across different subsets of the data.

Benefits: Provides a good balance between computation time and reliable performance estimation.

Considerations: The choice of k affects the trade-off between bias and variance. Larger k values reduce bias but increase computation time.

Technique: Divides the dataset into K subsets (folds), trains on K-1 and validates on the remaining fold. Repeats K times, each fold serving as validation once.

Dataset Behavior: Effective for general datasets, minimizes bias and variance.

Computational Time: Requires K training and validation cycles, can be time-consuming for larger K.

Advantages: Provides a balanced estimate of model performance. All instances serve as both training and testing data.

Disadvantages: Can be computationally intensive, especially for larger values of k.

2. Stratified K-Fold Cross-Validation:

Sometimes we may face a large imbalance of the target value in the dataset. For example, in a dataset concerning wristwatch prices, there might be a larger number of wrist watch having a high price. In the case of classification, in cats and dogs dataset there might be a large shift towards the dog class.

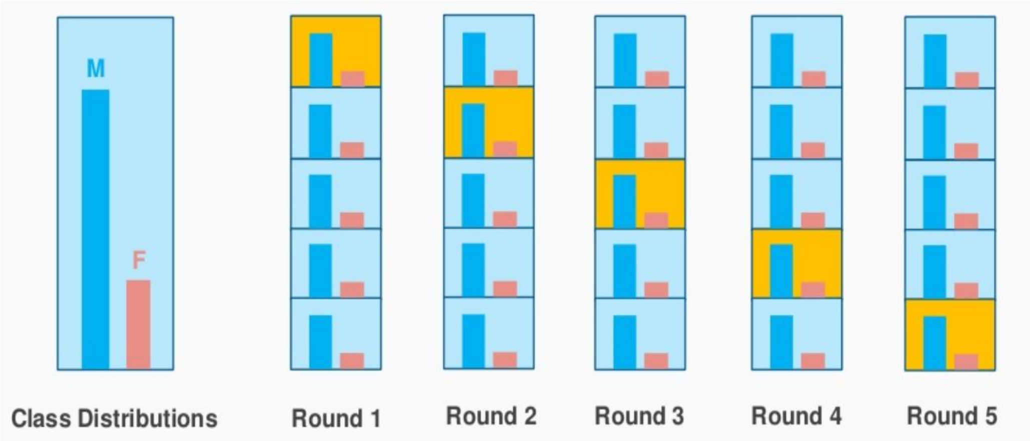
Stratified k-Fold is a variation of the standard k-Fold CV technique which is designed to be effective in such cases of target imbalance.

It works as follows. Stratified k-Fold splits the dataset on k folds such that each fold contains approximately the same percentage of samples of each target class as the complete set. In the case of regression, Stratified k-Fold makes sure that the mean target value is approximately equal in all the folds.

The algorithm of Stratified k-Fold technique:

CROSSVALIDATION TECHNIQUES INFORMATION MANUAL

1. Pick a number of folds – k
2. Split the dataset into k folds. Each fold must contain approximately the same percentage of samples of each target class as the complete set
3. Choose k – 1 folds which will be the training set. The remaining fold will be the test set
4. Train the model on the training set. On each iteration a new model must be trained
5. Validate on the test set
6. Save the result of the validation
7. Repeat steps 3 – 6 k times. Each time use the remaining fold as the test set. In the end, you should have validated the model on every fold that you have.
8. To get the final score average the results that you got on step 6.



Explanation: Similar to K-Fold, but it ensures that the class distribution remains consistent in each fold. This is particularly useful when dealing with imbalanced datasets, where one class has significantly fewer samples.

Application: Essential for maintaining a representative distribution of classes in each fold, thus improving model evaluation for imbalanced datasets.

Benefits: Reduces the risk of training and testing on unrepresentative subsets of the data.

Technique: Similar to K-Fold but preserves class distribution in each fold, ensuring representative subsets.

Dataset Behavior: Ideal for imbalanced datasets, where class proportions are uneven.

Computational Time: Similar to K-Fold, additional overhead for preserving class distribution.

Advantages: Helps prevent bias in performance estimates due to class imbalance.

Disadvantages: Similar computational intensity as K-Fold due to k iterations.

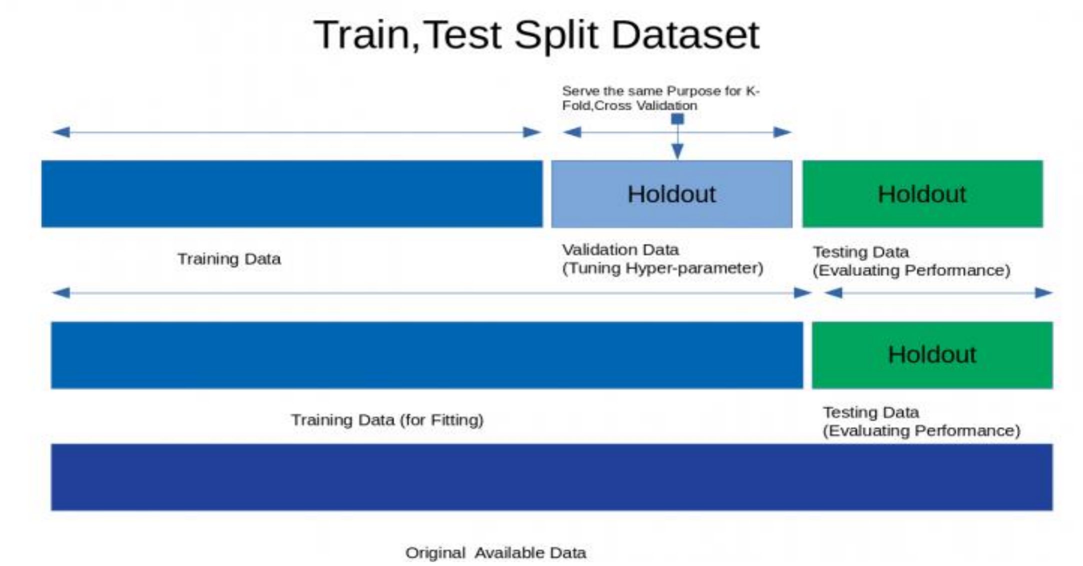
CROSSVALIDATION TECHNIQUES INFORMATION MANUAL

3. Shuffle-Split (Holdout) Cross-Validation:

Hold-out cross-validation is the simplest and most common technique. You might not know that it is a hold-out method but you certainly use it every day.

The algorithm of hold-out technique:

1. Divide the dataset into two parts: the training set and the test set. Usually, 80% of the dataset goes to the training set and 20% to the test set but you may choose any splitting that suits you better
2. Train the model on the training set
3. Validate on the test set
4. Save the result of the validation



Explanation: Holdout involves randomly splitting the dataset into training and testing sets, often with a predefined ratio. It can be iterated multiple times with different splits.

Application: Suitable for large datasets where computational constraints are a concern. Provides a quick estimate of model performance.

Benefits: Requires less computation time compared to K-Fold, making it a good choice for initial model assessment.

Technique: Randomly splits the dataset into training and validation sets with a specified ratio.

Dataset Behavior: Useful when the dataset is large or time-consuming to train on.

Computational Time: Requires training and validation on each iteration, quick for smaller datasets.

Advantages: Simple and quick. Useful for initial model assessment.

Disadvantages: Variability in performance estimates due to random splitting.

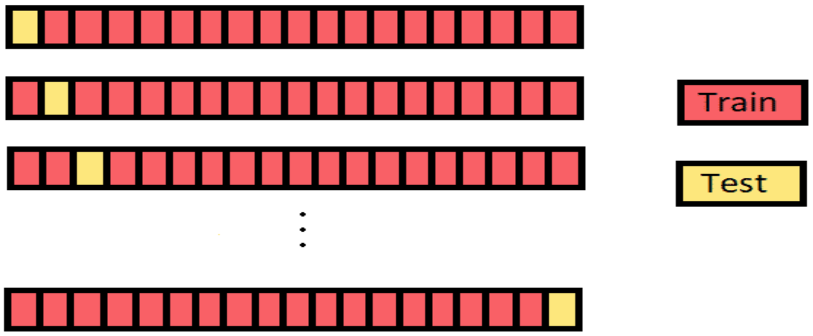
4. Leave-One-Out Cross-Validation:

Leave-one-out cross-validation (LOOCV) is an extreme case of k-Fold CV. Imagine if k is equal to n where n is the number of samples in the dataset. Such k-Fold case is equivalent to Leave-one-out technique.

The algorithm of LOOCV technique:

1. Choose one sample from the dataset which will be the test set
2. The remaining n – 1 samples will be the training set
3. Train the model on the training set. On each iteration, a new model must be trained
4. Validate on the test set
5. Save the result of the validation
6. Repeat steps 1 – 5 n times as for n samples we have n different training and test sets

7. To get the final score average the results that you got on step 5.



Explanation: LOOCV uses each instance in the dataset as a test set while the rest of the instances are used for training. This leads to n iterations, where n is the number of instances.

Application: Useful for small datasets where maximizing data utilization is crucial. Provides an unbiased estimate of model performance.

Benefits: Offers a low-bias evaluation since each instance is treated as a test set once.

Technique: Uses a single instance as the validation set and the rest for training. Repeats for all instances.

Dataset Behavior: Works well for small datasets, but can lead to high variance.

Computational Time: Requires training and validation for each instance, can be slow for large datasets.

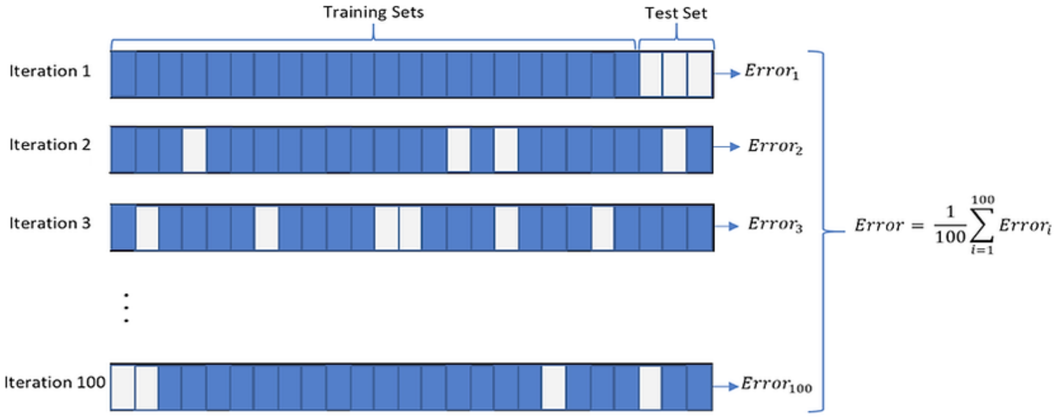
Advantages: Provides an unbiased estimate of performance since each instance acts as a test case.

Disadvantages: Highly computationally intensive for large datasets.

5. Monte Carlo Cross-Validation:

Monte Carlo operates rather differently. You randomly select (without replacement) some fraction of your data to form the training set, and then assign the rest of the points to the test set.

This process is then repeated multiple times, generating (at random) new training and test partitions each time.



Explanation: Monte Carlo Cross-Validation performs multiple random train-test splits and averages the results. It helps to assess the model's performance variability.

Application: Valuable for understanding the consistency of the model's performance across different data subsets.

Benefits: Accounts for variations in performance that might arise from random sampling.

Technique: Randomly samples subsets of the dataset for training and validation multiple times, calculating average performance.

Dataset Behavior: Robust against specific data splits and suitable for estimating generalization performance.

Computational Time: Requires multiple random sampling iterations, can be time-intensive.

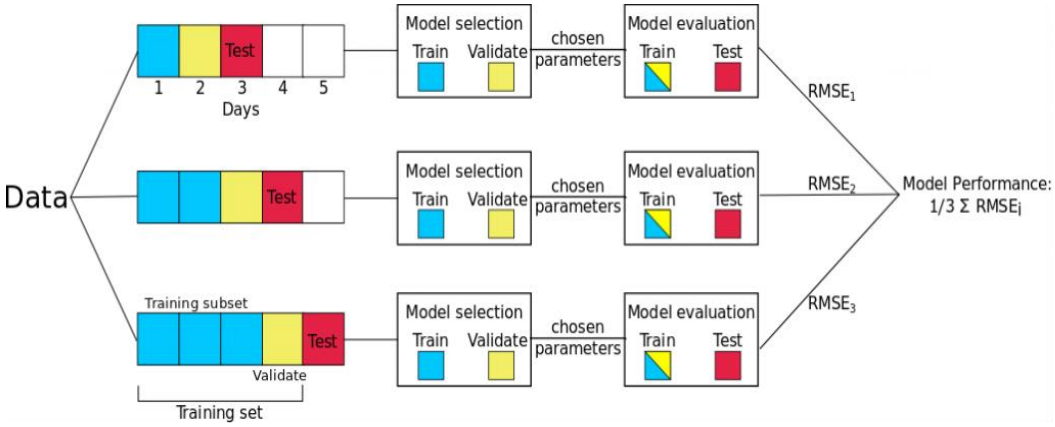
Advantages: Offers insights into the performance spread due to different splits, aiding in model selection.

Disadvantages: Requires more computational time due to multiple iterations.

6. Time Series Split Cross-Validation:

Regular cross validation techniques are not useful when working with time series datasets, time series datasets, can’t be randomly split and used for training and model validation, as we might miss on important components such as seasonality etc. With the order of the data being important it is difficult to split the data in any given interval. To tackle this issue we can use time series cross validation.

In this type of cross validation we take a small subsample of the data (keeping the order intact) and try and predict the immediate next examples for validation, this is also referred to as "forward chaining" or sometimes also refered to as "rolling cross validation", as we are continuously training and validating the model on the small snippets of data we are sure to found a good model if we can see that it is able to give good result on this rolling samples.



Explanation: Time Series Cross-Validation is designed for time-ordered data. It ensures that future time periods are not used for training past data, simulating real-world scenarios.

Application: Vital for time-dependent datasets like financial data or sensor readings, where temporal correlations exist.

Benefits: Preserves the temporal order and provides more realistic estimates of a model's performance in forecasting future data.

Technique: Splits time-ordered data into multiple folds, using earlier data for training and later data for validation.

Dataset Behavior: Tailored for time series data, ensuring that model evaluation reflects temporal patterns.

Computational Time: Training and validation times vary with the size of training windows.

Advantages: Reflects real-world temporal correlations and helps in evaluating the model's predictive ability.

Disadvantages: Similar computational intensity as K-Fold, but considerations vary based on the dataset's time span.

Each cross-validation technique serves the purpose of assessing a model's generalization performance while accounting for different aspects of the dataset. The choice of technique depends on your dataset's nature, size, and characteristics. Evaluating performance metrics and selecting the most appropriate technique is crucial for robust model evaluation and selection

While K-Fold and Stratified K-Fold are versatile, Holdout and LOOCV offer simplicity and unbiased estimates, respectively. Monte Carlo and Time Series Cross-Validation provide additional insights into performance stability and temporal patterns, respectively.