Cross Validation and Linear Regression Project

Cross Validation and Project

- Now that we've spent some time cleaning and working with the Ames Data Set, we're ready to let you test your new Regression skills on it to create a house sale price prediction model!
- However, there are two more general topics we want to learn before we jump to the project!

Cross Validation and Project

- Section Overview
  - Cross Validation in Detail
    - Train | Test Split
    - Train | Validation | Test Split
    - Scikit-Learn cross_val_score
    - Scikit-Learn cross_validate
  - Grid Search
  - Linear Regression Project Exercise

Cross Validation and Project

- We've already discussed models with built-in cross validation (e.g. RidgeCV).
- We will expand on this by exploring Scikit-Learn's general tools for utilizing cross-validation for any model.
- This will also allow us to later perform grid searches for the optimal combination of multiple hyperparameters.

Cross Validation and Project

- We'll begin by reviewing the most basic cross validation process we know so far (Train | Test split) and then slowly build up to the full k-fold cross validation.
- Let's get started!

Cross Validation

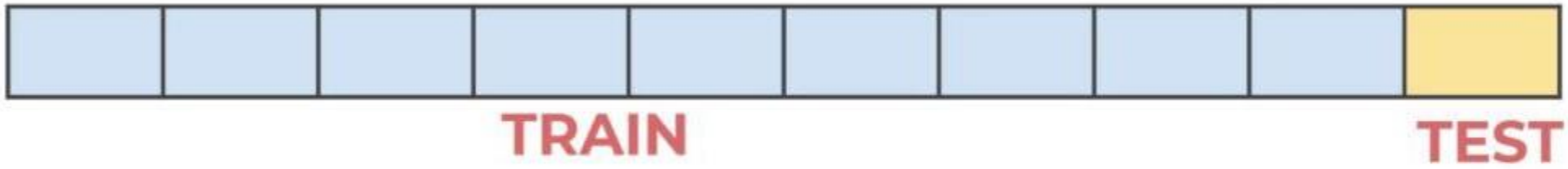
Train | Test Split

Cross Validation

- Begin with entire data set

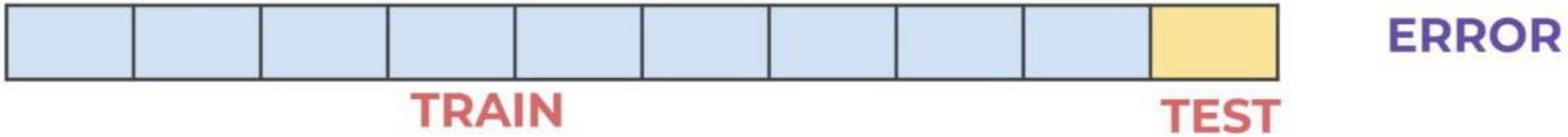
Cross Validation

- Split into two sets Train and Test



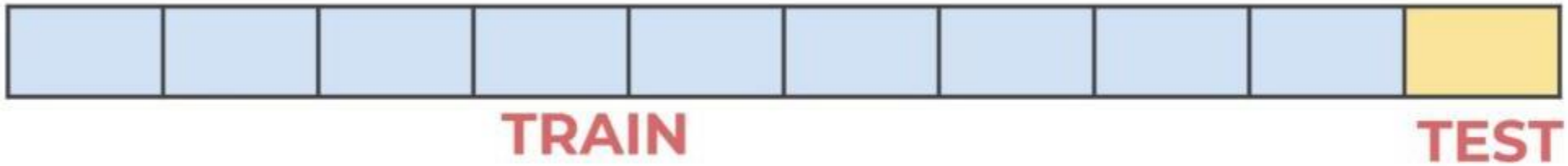
Cross Validation

- Train model then evaluate error on Test.



Cross Validation

- We “allow” ourselves model adjustments.



Cross Validation and Project

- Let's briefly review this process!
- We'll later expand on it until we reach full k-fold cross validation.

00-Cross-Validation.ipynb

Cross Validation

Train | Validation | Test Split

Cross Validation

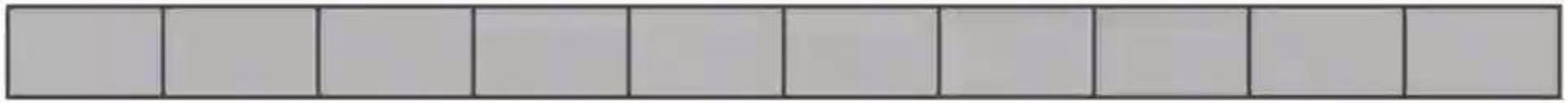
- We just saw that Train | Test split method has a disadvantage of not having a portion of data that can report a performance metric on truly “unseen” data.
- While adjusting hyperparameters on test data is a fair technique and not typically referred to as “data leakage”, it is a potential issue in regards to reporting.

Cross Validation

- If we want a truly fair and final set of performance metrics, we should get these metrics from a **final** test set that we do not allow ourselves to adjust on.
- Let's quickly review this process in theory and application!

Cross Validation

- Begin with entire data set

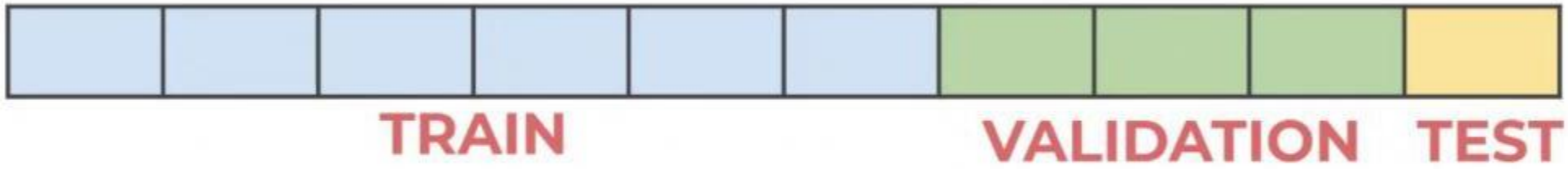


Cross Validation

- Begin with entire data set

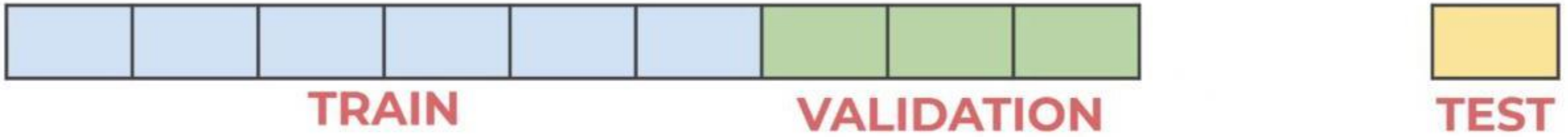
Cross Validation

- Split into: Train, Validation, and Test.



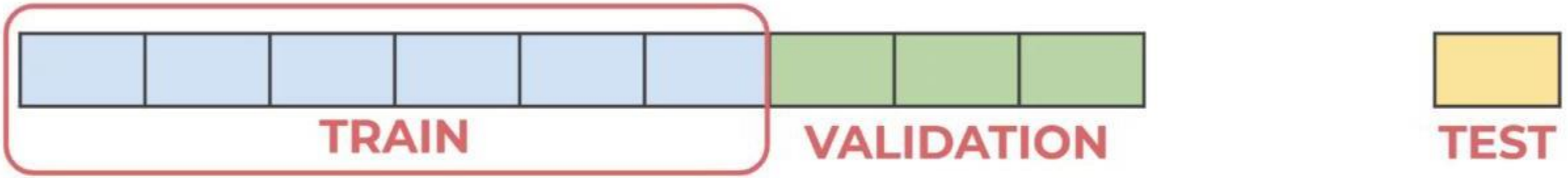
Cross Validation

- Set aside Test set for final metrics.



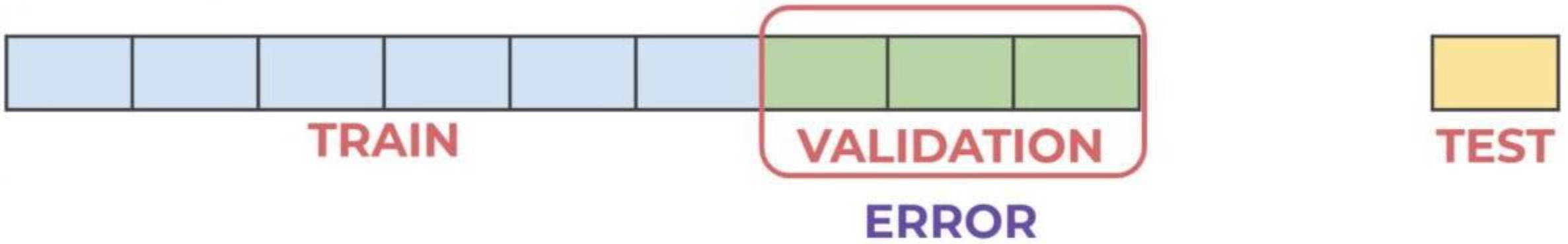
Cross Validation

- Fit model on Train set.



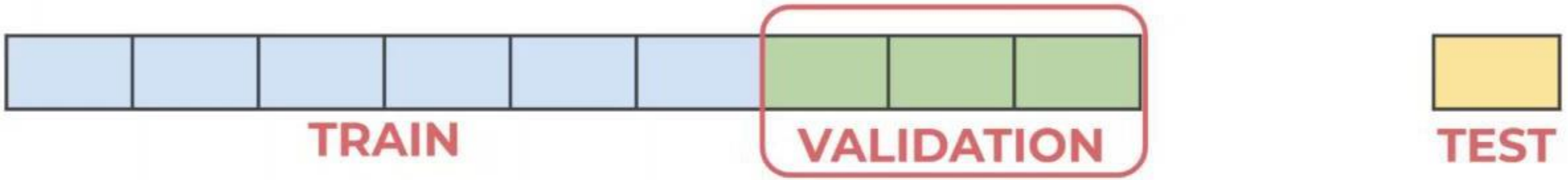
Cross Validation

- Evaluate performance on validation set.



Cross Validation

- Adjust hyperparameters as needed...

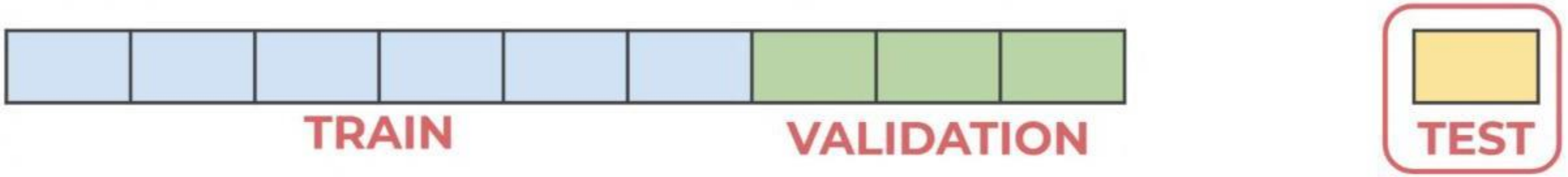


Cross Validation

- Train, validate, and adjust as necessary.

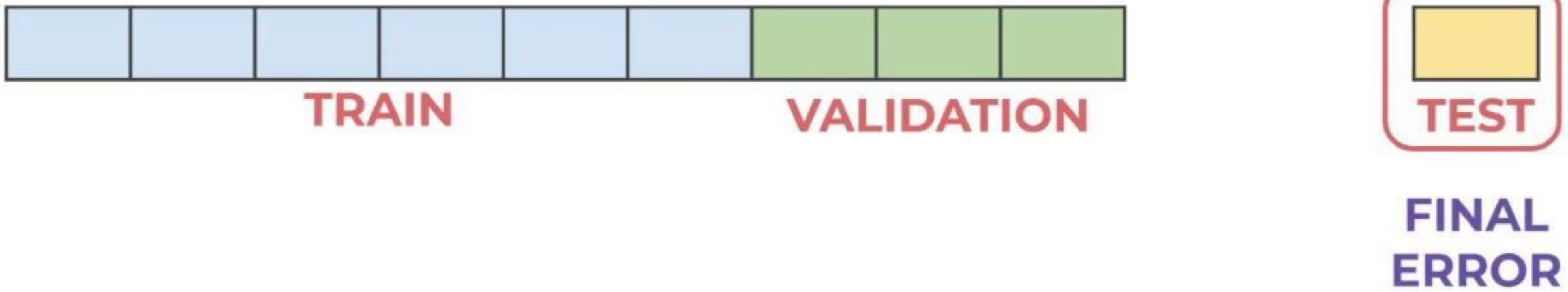
Cross Validation

- Perform final evaluation on Test set.



Cross Validation

- Do not adjust! Report this as final metric.



Cross Validation

- Recall the entire reason to not adjust after the final test data set is to get the fairest evaluation of the model.
- The model was not fitted to the final test data **and** the model hyperparameters were not adjusted based off final test data.
- This is truly never before seen data!

Cross Validation

- To achieve this in Python with Scikit-Learn we simply perform the **train_test_split()** function call twice.
  - Once to split off larger training set.
  - Second time to split remaining data into a validation set and test set.
- Let's jump to a notebook to see how its done!

00-Cross-Validation.ipynb

Cross Validation

Using the cross_val_score function

Cross Validation

- Start with entire data set:

Cross Validation

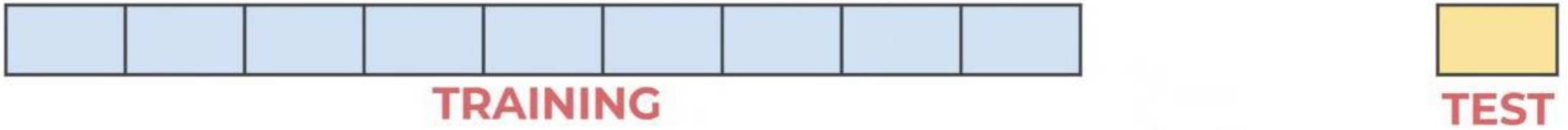
- Split data Training data and Test Data:

Cross Validation

- Split data Training data and Test Data:

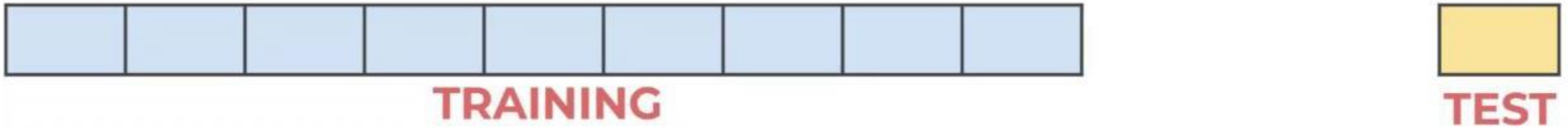
Cross Validation

- Remove Test data for final evaluation:



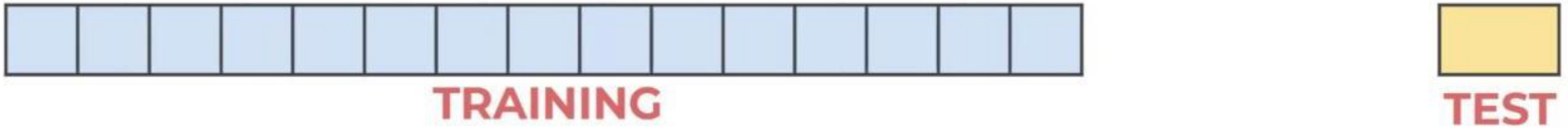
Cross Validation

- Choose K-Fold Split Value for Training Data



Cross Validation

- Recall larger K means more computation!



Cross Validation

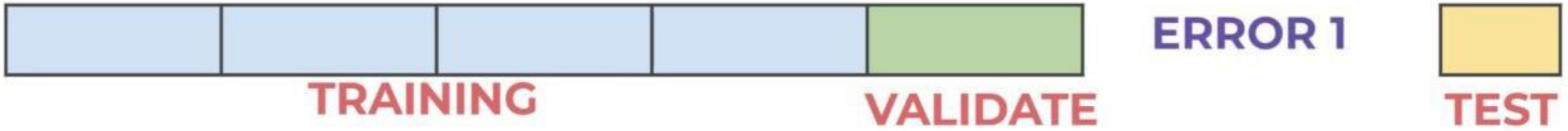
- Here K = 5

Cross Validation

- Train on K-1 folds and Validate on 1 Fold

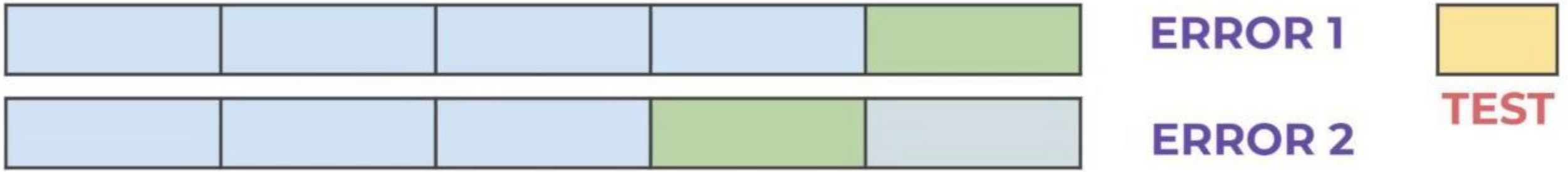
Cross Validation

- Obtain an error metric for this fold:



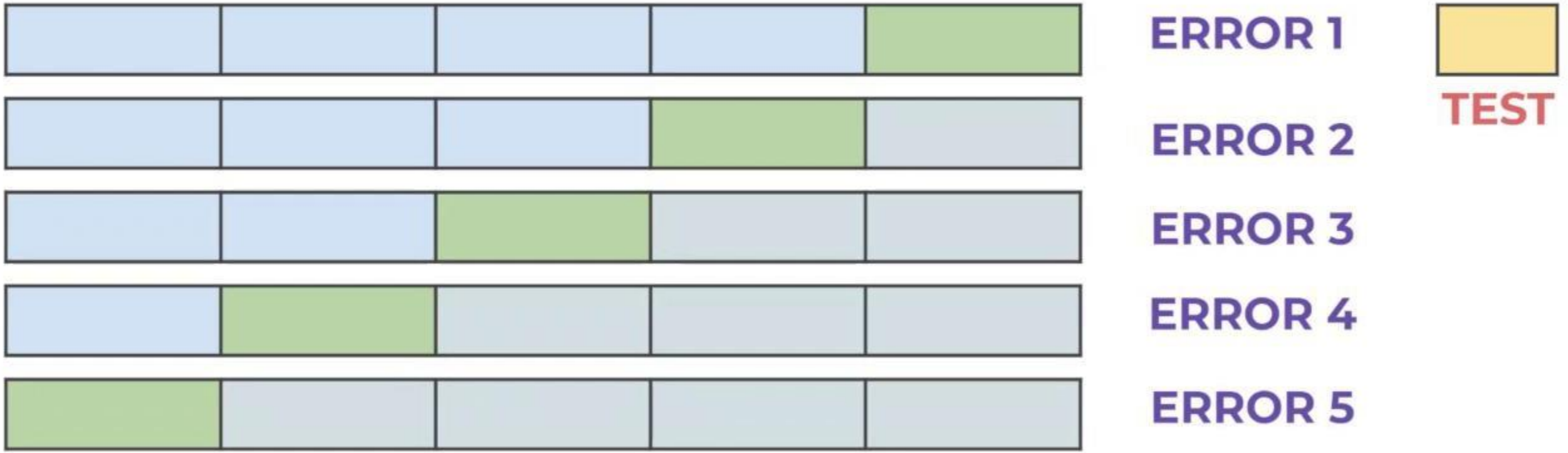
Cross Validation

- Repeat for another combination



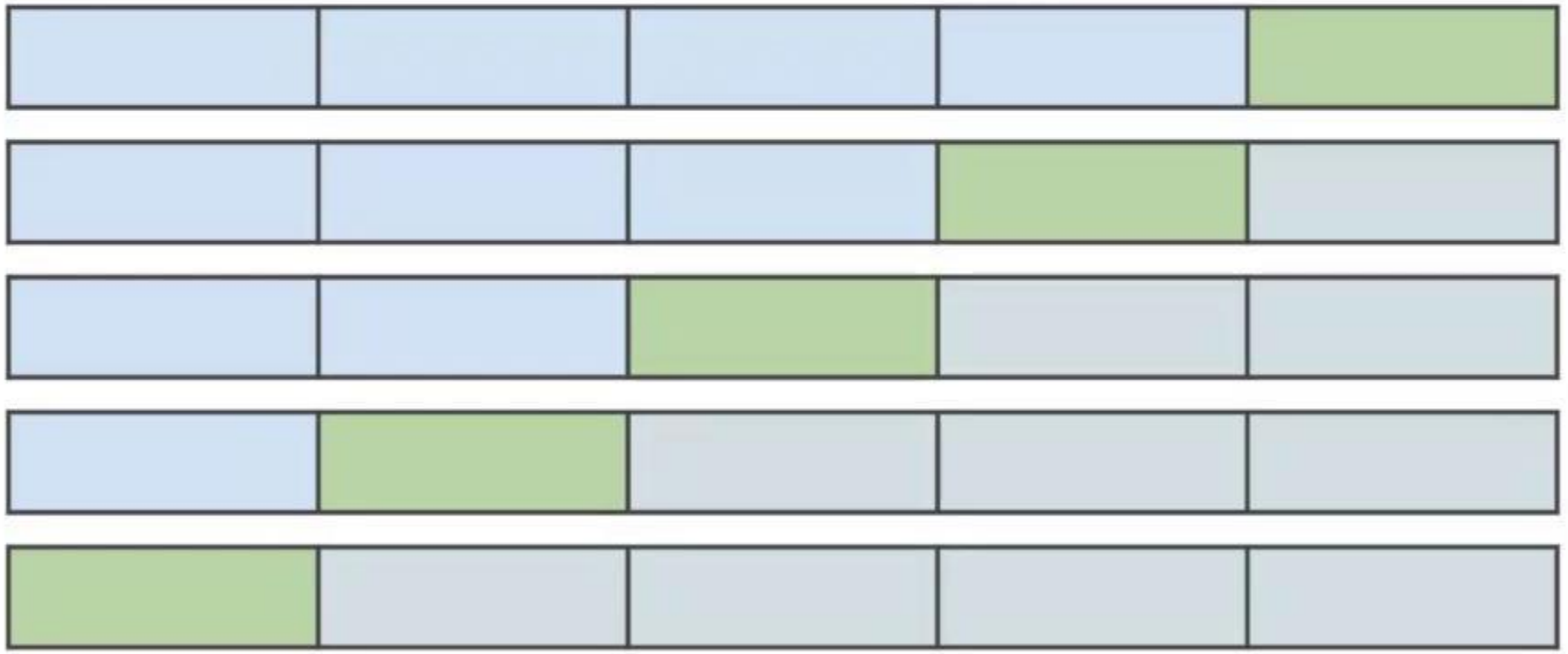
Cross Validation

- Continue for all fold combinations



Cross Validation

- Use mean error for parameter adjustments



ERROR 1

ERROR 2

ERROR 3

ERROR 4

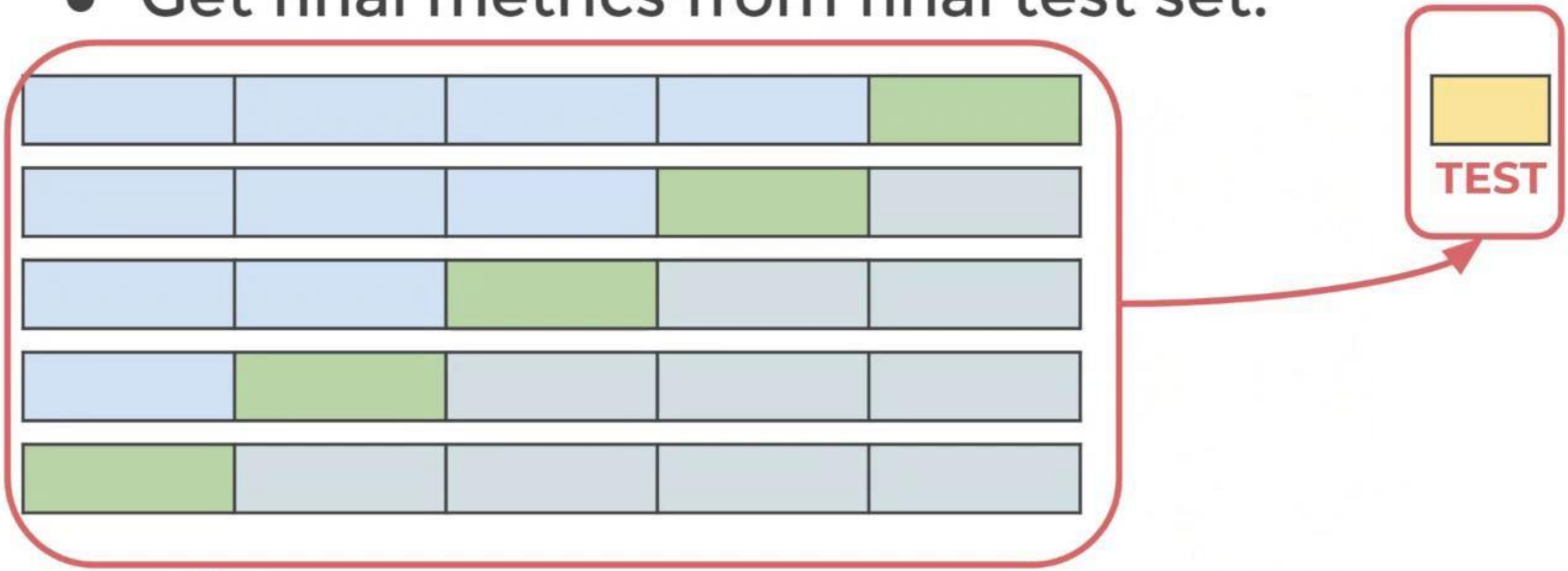
ERROR 5

AVG ERROR

TEST

Cross Validation

- Get final metrics from final test set.



Cross Validation

- The **cross_val_score** function uses a model and training set (along with a K and chosen metric) to perform all of this for us automatically!
- This allows for K-Fold cross validation to be performed on any model.
- Let's explore how to use it!

00-Cross-Validation.ipynb

Cross Validation

Using the cross_validate function

Cross Validation

- The **cross_validate** function allows us to view multiple performance metrics from cross validation on a model and explore how much time fitting and testing took.
- Let's quickly review how to use this function call!

00-Cross-Validation.ipynb

Grid Search

Grid Search

- Often more complex models have multiple adjustable hyperparameters.
- A grid search is a way of training and validating a model on every possible combination of multiple hyperparameter options.

Grid Search

- Scikit-Learn includes a **GridSearchCV** class capable of testing a dictionary of multiple hyperparameter options through cross-validation.
- This allows for both cross-validation and a grid search to be performed in a generalized way for any model.

01-Grid-Search.ipynb

Linear Regression Project Overview

02-Linear-Regression-Project-Exercise.ipynb