# Model Training

Tuning, Training and Evaluation

# Steps to Training a Model

Split dataset into training and testing

Declare hyperparameters to tune

Fit and tune with cross validation

Evaluate the models

# Splitting Dataset

Think of your data as a limited resource.

- You can spend some of it to train your model (i.e. feed it to the algorithm).
- You can spend some of it to evaluate (test) your model.
- But you can’t reuse the same data for both!

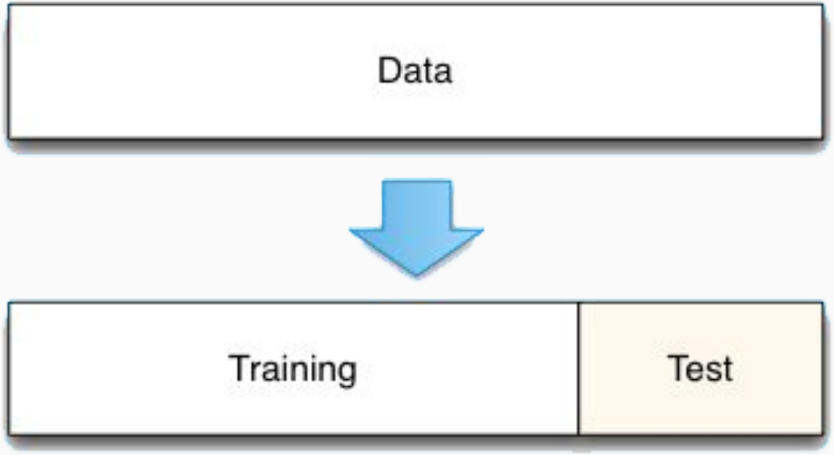
# Splitting Dataset

If you evaluate your model on the same data you used to train it, your model could be very overfit and you wouldn’t even know!

A model should be judged on its ability to predict new, unseen data.

# Splitting Dataset

Therefore, you should have separate training and test subsets of your dataset.



# Splitting Dataset

Training sets are used to fit and tune your models. Test sets are put aside as "unseen" data to evaluate your models.

- You should always split your data before doing anything else.
- This is the best way to get reliable estimates of your models' performance.
- After splitting your data, **don't touch your test set** until you're ready to choose your final model!

# Model Parameters vs Hyperparameters?

The key distinction is that model parameters can be learned directly from the training data while hyperparameters cannot.

# Model Parameters

Model parameters are learned attributes that define individual models

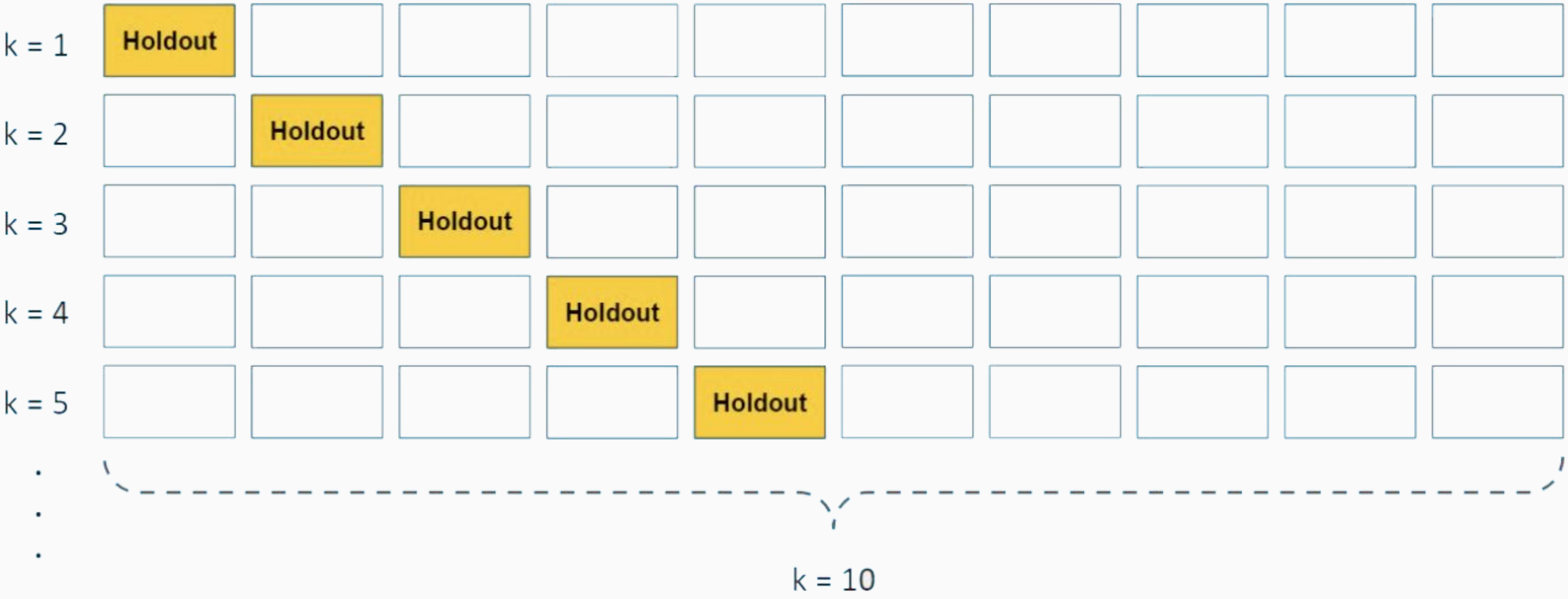
- e.g. regression coefficients
- e.g. decision tree split locations
- They can be learned directly from the training data

# Hyperparameters

Hyperparameters express "higher-level" structural settings for algorithms.

- e.g. strength of the penalty used in regularized regression
- e.g. the number of trees to include in a random forest
- They are decided before fitting the model because they can't be learned from the data

# Cross-Validation



| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| k = 1 | Holdout | | | | | | | | | |
| k = 2 | | Holdout | | | | | | | | |
| k = 3 | | | Holdout | | | | | | | |
| k = 4 | | | | Holdout | | | | | | |
| k = 5 | | | | | Holdout | | | | | |

.
.
.

k = 10

# Cross-Validation

Steps for 10-fold cross-validation:

1. Split your data into 10 equal parts, or "folds".
2. Train your model on 9 folds (e.g. the first 9 folds).
3. Evaluate it on the 1 remaining "hold-out" fold.
4. Perform steps (2) and (3) 10 times, each time holding out a different fold.
5. Average the performance across all 10 hold-out folds.

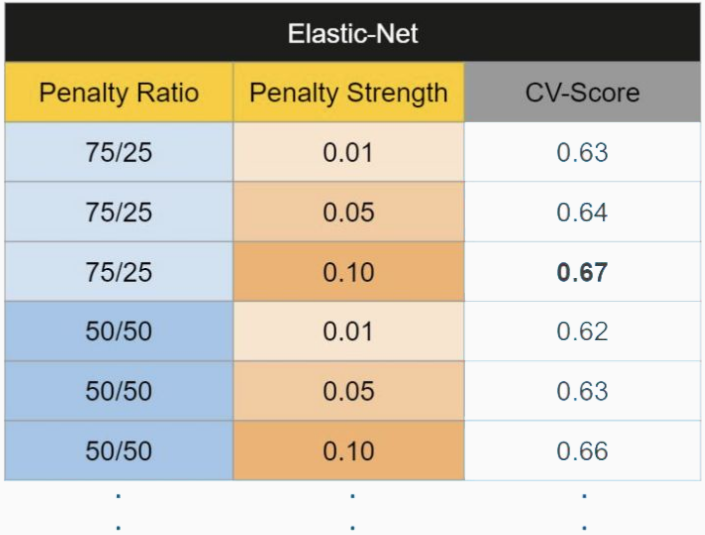
# Fit and Tune Models

The high-level pseudocode looks like this:

```
For each algorithm (i.e. regularized regression, random forest, etc.):
    For each set of hyperparameter values to try:
        Perform cross-validation using the training set.
        Calculate cross-validated score.
```

# Fit and Tune Models

At the end of this process, you will have a cross-validated score for each set of hyperparameter values... for each algorithm.



| Elastic-Net | | |
|---|---|---|
| Penalty Ratio | Penalty Strength | CV-Score |
| 75/25 | 0.01 | 0.63 |
| 75/25 | 0.05 | 0.64 |
| 75/25 | 0.10 | **0.67** |
| 50/50 | 0.01 | 0.62 |
| 50/50 | 0.05 | 0.63 |
| 50/50 | 0.10 | 0.66 |
| : | : | : |

# Model Evaluation

There are a variety of performance metrics you could choose from:

- For regression tasks, we recommend Mean Squared Error (MSE) or Mean Absolute Error (MAE). (Lower values are better)
- For classification tasks, we recommend Area Under ROC Curve (AUROC). (Higher values are better)

# Model Evaluation

The process is very straightforward:

1. For each of your models, make predictions on your test set.
2. Calculate performance metrics using those predictions and the "ground truth" target variable from the test set.

# Model Evaluation

Finally, use these questions to help you pick the winning model:

- Which model had the best performance on the test set? (performance)
- Does it perform well across various performance metrics? (robustness)
- Did it also have (one of) the best cross-validated scores from the training set? (consistency)
- Does it solve the original business problem? (win condition)