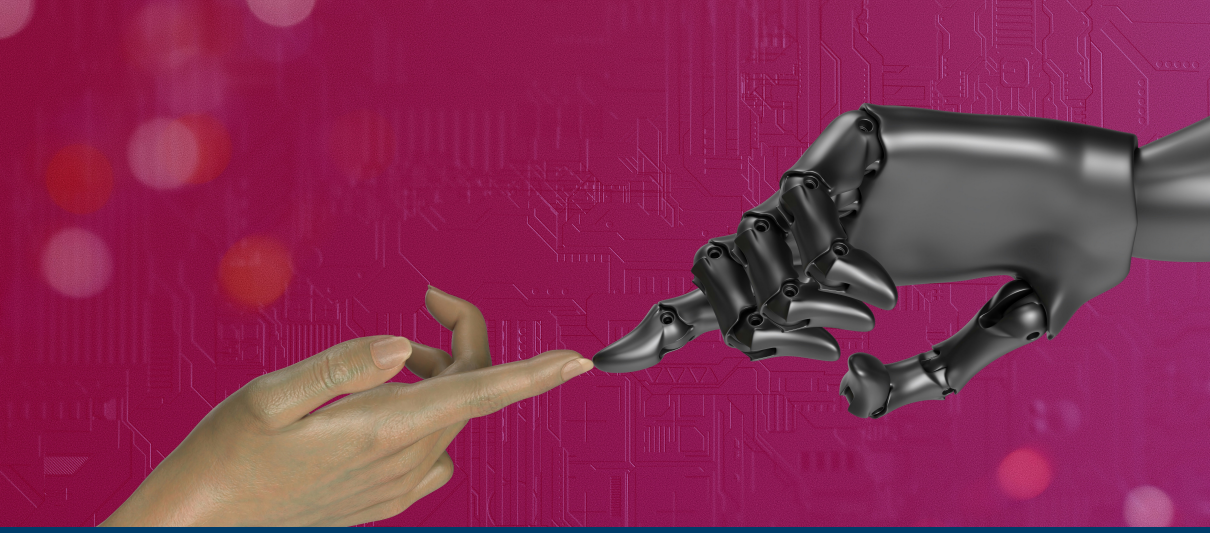


Hands-On Machine Learning

Dr. Derek Bridge | 2026/2027

UCC Coláiste na hOllscoile Corcaigh
University College Cork, Ireland

School of Computer Science and Information Technology
Scoil na Ríomheolaíochta agus na Teicneolaíochta Faisnéise

5. Model Selection

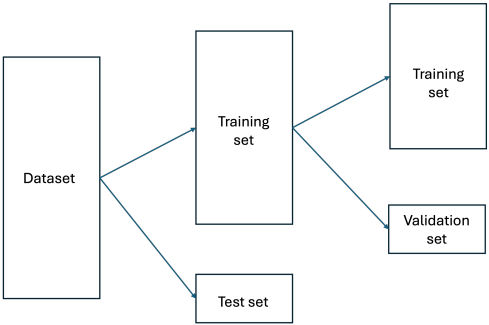
In this lecture, we use the same housing data that we used in the previous lecture.

Model Selection

- **Model Selection** involves evaluating multiple algorithms and hyperparameter configurations to identify the best-performing model(s) for your data.
- If you (like many people) use the test set for model selection, then information about the test set is being used to develop the model — **leakage**! Your final error estimate will likely be over optimistic.
- We need another test set — one that we use during model selection. This is called the **validation set**.

Model Selection using Holdout

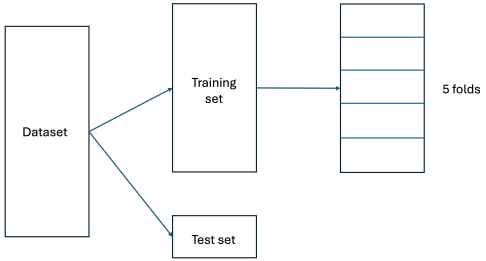
- The simplest approach is to shuffle then randomly partition the dataset into three, e.g. 60%-20%-20%:
  - training set;
  - validation set;
  - test set.
- Train the various configurations on the training set.
- Test them on the validation set.
- Choose the configuration with the lowest validation error.
- Train the winning configuration on the union of the training set and validation set.
- Test this model on the test set.



- This method requires an even bigger dataset: one we can split into three large enough partitions!
- When we have a smaller dataset, we use a **resampling** method, where the examples get used and re-used for training and validation.

Model Selection using k-Fold Cross-Validation

- A common resampling method is k-**fold cross-validation**.
- Shuffle, then randomly partition the dataset into two: training set and test set.
- Then partition the training set into k equal-sized partitions.
- For $i \in [1, \ldots, k]$:
  - Train the various configurations on all folds except fold k.
  - Test them on fold k, acting as the validation set.
- Choose the configuration with the lowest *mean* validation error.
- Train the winning configuration on the original training set.
- Test this model on the test set.



Holdout

Advantage

- The validation error is independent of the training set.

Disadvantages

- Results can vary quite a lot across different runs. Informally, you might get lucky — or unlucky. In other words, in any one split, the data used for training or testing might not be representative.
- We are training on only a subset of the available dataset, perhaps as little as 60% of it. From so little data, we may learn a worse model and so our error measurement may be pessimistic.

k-Fold CV

Advantage

- The validation errors of the folds are independent — because examples are included in only one validation set.
- Better use is made of the dataset: for $k = 10$, for example, we train using $9/10$ of the dataset.
- You now have k validation errors. So, not only can you compute their mean, you could also compute their standard deviation, or you could even use tests for statistical significance when comparing different models.

Disadvantages

- While the validation sets are independent of each other, the training sets are not. They will overlap with each other to some degree.
- The number of folds is constrained by the size of the dataset and the desire sometimes on the part of statisticians to have folds of at least 30 examples.
- It can be costly to train the learning algorithm k times.
- There may still be some variability in the results due to 'lucky'/'unlucky' splits.

Important

- In the past, students have tried holdout and k-fold CV as if they were in competition with each other. This betrays a misunderstanding. You do not try them both and see which one gives the lower error. You pick one of them — the one that makes most sense for your data — and use it.
- In practice, we only use the holdout method when we have a very large dataset, e.g. 10s of 1000s. The size of the dataset mitigates the disadvantages.
- We use k-fold CV when we have a smaller dataset, e.g. several 100s.
- (There are some alternatives, e.g. repeated holdout, leave-one-out CV,...)
- Whichever you choose (holdout of k-fold CV), remember to **stratify** for classification.

Grid Search

- Suppose you have multiple hyperparameters or other ways of configuring an algorithm.
- **Grid Search** will try all combinations.

Randomized Search

- When the number of combinations of hyperparameter values/configurations is high, Grid Search's exhaustive approach may take too long.
- We can instead use **Randomized Search**, which tries a certain number of them, chosen at random.

There are yet further alternatives to these two, e.g. Halving Grid Search. And there are advanced Bayesian methods.