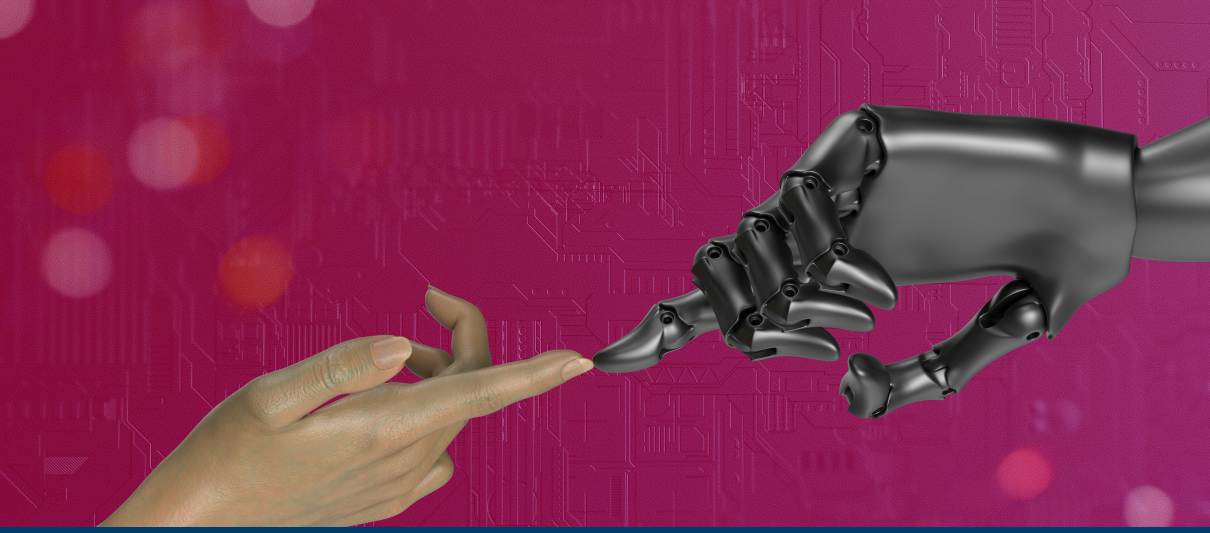


Hands-On Machine Learning

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

4. Regression

The data we use was collected by Dean De Cock in 2011 and made available on the web site of the Journal of Statistics Education, now more easily available on Kaggle. Each of the 2929 examples is a residential property in Ames, Iowa with around 80 features. The idea to use it (and to use a subset of it) comes from Sebastian Raschka: Machine Learning with PyTorch and Scikit-Learn, Packt Publishing, 2022.

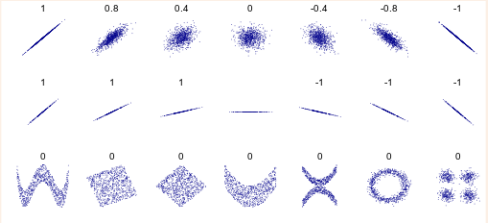
Correlation

- **Correlation coefficients** measure the strength and direction of the relationship between two variables (e.g. two features or a feature and the target)
- E.g. ice-cream sales and temperature are positively correlated.
- E.g. altitude and temperature are negatively correlated.

Pearson correlation coefficient

$$r = \frac{\sum_{i=1}^{m}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{m}(x_i - \overline{x})^2}\sqrt{\sum_{i=1}^{m}(y_i - \overline{y})^2}}$$

where m is the number of values, x_i & y_i are individual values, and $\overline{x}$ & $\overline{y}$ are their means. r will lie between -1 and 1.



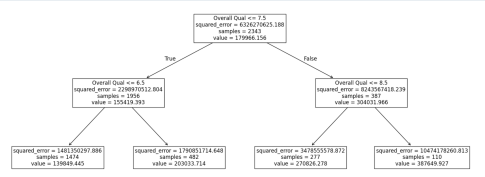
Some Uses of Correlation Coefficients in ML

- A feature that is not predictive of the target may be a candidate for removal.
- A feature that is strongly correlated with another (multicolinearity) may be a candidate for removal.
- Some Machine Learning algorithms assume that features are not correlated. If they are correlated, then either we should not use that algorithm or we should use dimensionality reduction techniques.
- If a feature is correlated with a sensitive feature (e.g. sex, gender, race, ...), then your model may be biased (even if it does not use the sensitive feature).

Warnings

- "Correlation does not imply causation." E.g. ice-cream sales and sunstroke are positively correlated.
- Correlation could be due to chance — especially in small datasets.
- Pearson correlation measures whether the two variables are linearly related — do they change at a constant rate. But their relationship might be curvilinear.

Decision Tree Regression



- `squared_error`: is the *MSE* of a node — this is used in place of Gini.
- `samples`: how many training examples the node applies to.
- `value`: the prediction if we stop at this node (the mean target value of the training examples that this node applies to)

Similarity & Distance

- In AI, we often want to know how *similar* one object is to another.
  - E.g. how similar is my house to yours?
  - E.g. which house in our dataset is most similar to yours?
- In fact, here we are instead going to measure how *different* they are using a **distance function**.

This is not about geographical distance.

Euclidean Distance

- Let $\boldsymbol{x}$ and $\boldsymbol{x}'$ be two examples with n features.

$$d(\boldsymbol{x}, \boldsymbol{x}') = \sqrt{(\boldsymbol{x}_1 - \boldsymbol{x}'_1)^2 + (\boldsymbol{x}_2 - \boldsymbol{x}'_2)^2 + \cdots + (\boldsymbol{x}_n - \boldsymbol{x}'_n)^2}$$
$$= \sqrt{\sum_{i=1}^{n} (\boldsymbol{x}_i - \boldsymbol{x}'_i)^2}$$

- Its minimum value is 0 (meaning identical) but there's no maximum value (depends on your data).
- Class exercise. What is the Euclidean distance between $\boldsymbol{x} = \begin{bmatrix} 100 \\ 1 \\ 4 \end{bmatrix}$ and $\boldsymbol{x}' = \begin{bmatrix} 100 \\ 5 \\ 1 \end{bmatrix}$?

Nearest-Neighbours Regressor

- To predict the target value $\hat{y}$ for unseen example $\boldsymbol{x}'$,
  - we find the example $\boldsymbol{x}$ in the labeled training set whose distance from $\boldsymbol{x}'$ is smallest;
  - we use the y-value for $\boldsymbol{x}$ as our prediction.
- We also refer to this as a **1-nearest-neighbour regressor** or **1NN** or **kNN for** $k = 1$.
- It has the problem that it can be incorrectly influenced by noisy examples (incorrect feature-values or incorrect labels).

k-Nearest-Neighbours Regressor

- To predict the target value $\hat{y}$ for unseen example $\boldsymbol{x}'$,
  - we find the k examples in the labeled training set whose distance from $\boldsymbol{x}'$ is smallest;
  - we use the *mean* of their y-values as our prediction.
- We abbreviate the name of this to **kNN**, e.g. 3NN.
- It is less influenced by noisy examples.
- k is a **hyperparameter** and we need a way of choosing its value — see later.

Question

- k is a **hyperparameter** – we'll soon learn how to choose a good value.
- Why are we unlikely to use $k = m$, where m is the number of examples in the training set?

Questions

- Which is more likely to result in **overfitting**? Small k or large k?
- Hence, which is more likely to result in **underfitting**?

Question

- One variant of kNN is **weighted kNN**, where we compute a distance-weighted mean.
- Now we could reasonably use $k = m$. Why? (Doing so is sometimes called Shepard's method.)

Questions

- How would we modify the algorithm to make a **kNN classifier?**
- Some classifiers output probabilities, one per class. How would we modify the algorithm to make a kNN classifier that outputs these **class probabilities**?

Problems with Euclidean distance

Problem

Features with different scales.

Solution

Scaling

Problem

Features that are correlated with each other.

Problem

The curse of dimensionality, where $n \gg m$.

Solution

Dimensionality reduction (e.g. Principal Components Analysis)

(These are problems with many other distance measures too.)

Scaling Numeric Values

- Different numeric-valued features often have very different ranges.
  - E.g. floor area ranges from a few tens to a few hundreds of square metres;
  - But the number of bedrooms and bathrooms ranges from 0 to a dozen or so.
- When computing the Euclidean distance, features with large ranges will dominate the distance calculations, thus giving features with small ranges negligible influence.
- E.g., consider your house $\boldsymbol{x} = \begin{bmatrix} 126 \\ 3 \\ 1 \end{bmatrix}$ and two others, $\boldsymbol{x}' = \begin{bmatrix} 131 \\ 3 \\ 1 \end{bmatrix}$ and $\boldsymbol{x}'' = \begin{bmatrix} 126 \\ 7 \\ 1 \end{bmatrix}$

  Intuitively, which is most similar to yours? According to Euclidean distance, which is most similar to yours?

Scaling

Scaling makes features more comparable.

Min-max scaling

$$x \leftarrow \frac{x - min}{max - min}$$

Robust scaling

$$x \leftarrow \frac{x - median}{inter\text{-}quartile\text{-}range}$$

Standardization

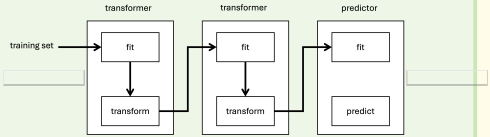
$$x \leftarrow \frac{x - mean}{standard\text{-}deviation}$$

Question

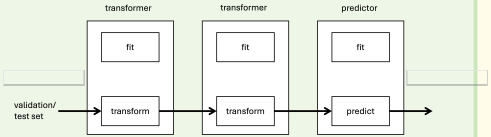
Consider a feature's *min*, *max*, *median*, *inter-quartile-range*, *mean* and *standard-deviation* when used for scaling.
Are these computed on (a) the whole dataset, (b) the training set and separately also the test set, or (c) just the training set?

scikit-learn pipelines

Training (`fit`)



Inference (`predict`)



Two types of Supervised Learning

Instance-Based Learning and Model-Based Learning

Differ in *when* they generalize

	Instance-Based	**Model-Based**
Train (fit)	Memorize the training examples	Generalize from the training examples – learn a function that captures the relationship between features and targets
Infer (predict)	Generalize from the training examples — use distance from training examples to predict unseen example's target value	Apply the learned function to the unseen example

Compare and Contrast

Compare model-based learning and instance-based learning.
Think about training time, inference time, incrementality and re-training.

Speeding-up kNN inference

There are data structures that support faster retrieval of neighbours (e.g. kd-trees and ball-trees).
During training (fit), we could insert the training examples into one of these data structures.

Image credits

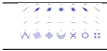


Image by Denis Boigelot, taken from Wikipedia