UNIVERSITY OF EDINBURGH
Business School

Predictive Analytics and Modelling of Data

CMSE11428 (2020-2021)

Dr Xuefei Lu
The University of Edinburgh Business School

Bias and variance

Bias and variance

- Bias and variance capture these two concepts formally.
  Bias captures whether we're truly predicting what we say we're predicting.
- Assume we have the 'true' relationship between the independent and dependent variables as
  $$y = f(x) + \epsilon$$
- where $f(x)$ is the relationship, and error term ϵ with zero mean and variance σ^2. Then we want find a function $\hat{f}\,(x; Data)$ to approximate the true function $f(x)$ as well as possible, e.g. the $\left(y - \hat{f}\,(x; Data)\right)^2$ to be minimal for both given $Data$ (D) and new points.
- The **bias** is
  $$Bias_D\left[\hat{f}\,(x; D)\right] = E_D\left[\hat{f}\,(x; D)\right] - f(x)$$
- The bias of your model should be as close to 0 as possible, otherwise, we would systematically predict the wrong value.

Bias and variance

- The **variance** is a measure of reliability and variation in the estimate, i.e., it shows how much our result will remain similar over a small range of values if we apply our model multiple times.
- Formally, the variance of our predictive model is

$$Var_D[\hat{f}\ (x; D)] = E_D\left[\left(E_D[\hat{f}\ (x; Data)] - \hat{f}\ (x; Data)\right)^2\right]$$

$$= E_D[\hat{f}\ (x; Data)^2] - E_D[\hat{f}\ (x; Data)]^2$$

If we put bias and variance together for the mean root squared error (note that we can do the same thing for classification by making use of the indicator function), we get:

$$\boxed{E_D\left[\left(y - \hat{f}\ (x; Data)\right)^2\right] = \left(Bias_D[\hat{f}\ (x; D)]\right)^2 + Var_D[\hat{f}\ (x; D)] + \sigma^2}$$

The prediction error is based on **the bias**, which is hopefully 0, **a variance**, also preferably low, and **the intrinsic noise** σ^2.

The expectation E_D ranges over different choices of the training set D.

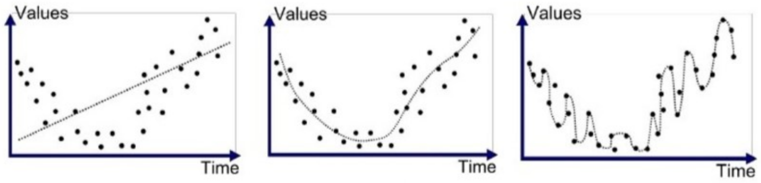
All three terms are non-negative ➔ a lower bound on the expected error on unseen samples.

The bias-variance trade-off

$$E_D\left[\left(y-\hat{f}\,(x;Data)\right)^2\right] = \left(Bias_D[\hat{f}\,(x;D)]\right)^2 + Var_D[\hat{f}\,(x;D)] + \sigma^2$$

Generally, we always try to be as close to the true relationship f and have low bias. At the same time, we want to be able to churn out a model that has a low variance and is not too susceptible to the input.

Unfortunately, it is typically impossible to do both simultaneously. High-variance learning methods may be able to represent their training set well but are at risk of overfitting to noisy or unrepresentative training data. In contrast, algorithms with high bias typically produce simpler models that don't tend to overfit but may underfit their training data, failing to capture important regularities.



Underfitted

Good Fit/Robust

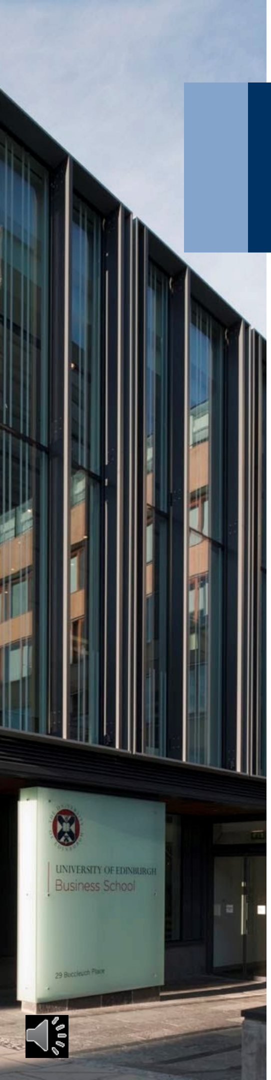
Overfitted

The more complex the model $\hat{f}$ is, the more data points it will capture, and the lower the bias will be. However, complexity will make the model "move" more to capture the data points, and hence its variance will be larger.

Cross-validation

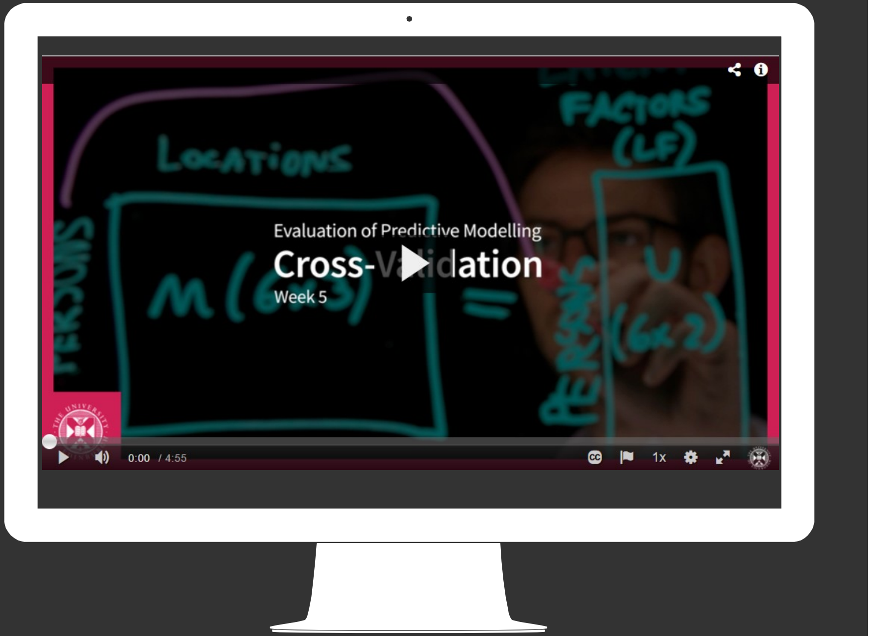
Using k-fold and leave-one-out cross-validation

- A stronger, more robust approach to ensure that our algorithms are not capturing the underlying patterns that are only particular to the training set is cross-validation (CV).
- Cross-validation divides the dataset into k segments. One of the k segments will be used to test the data, and the other $k - 1$ for training the data. This procedure is then repeated with a different segment to the first one we used. So, another segment is used as test set, and the $k - 1$, now including the first segment we used earlier, form the training set. After each of the k holdout scenarios, the evaluation metric of your choice is calculated, accuracy for example, and after all the k runs the average (and sometimes other summary statistics such as the standard deviation) of that metric is used.
- The segments used for the test set are called folds: k-fold CV.



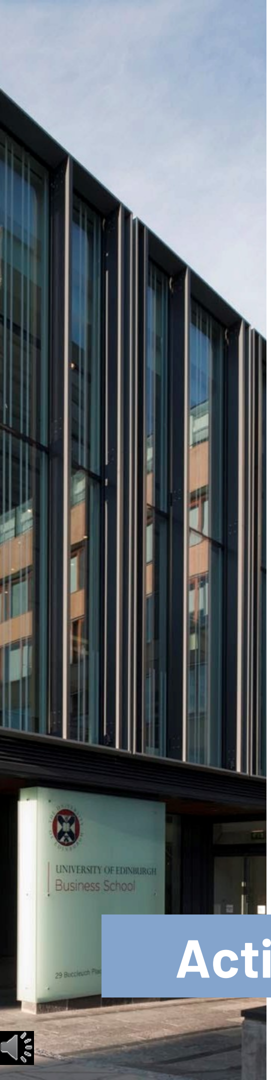
Using k-fold and leave-one-out cross-validation

- In case of class imbalance: in stratified k-fold CV, the balance of the classes is also maintained over the folds.
- The value of k can vary, but is typically set between 5 and 10.
- An extreme version of k-fold cross-validation is **leave-one-out cross-validation**.
  - n-folds CV, n as the dataset size n
  - computationally very expensive
- In general, the higher the value of k, the more times the algorithms need to be run. – Can be problematic for large datasets.



Cross-validation

- Watch the following video via this link
- https://media.ed.ac.uk/media/Cross-validation/1_50jm9mwz/114521421



- You will now learn how to code cross-validation, using both standard CV, as well as with pre-modelling activities, and stratified sampling.

- Please study the following file:
  14 - How_to_code_cross_validation.ipynb
- Then try to complete the following exercise:
  15 - Activity_9_coding_cross_validation.ipynb + absent.csv

Activity: How to code cross-validation

UNIVERSITY OF EDINBURGH
Business School