Validation

Monday, September 2, 2024 7:22 PM

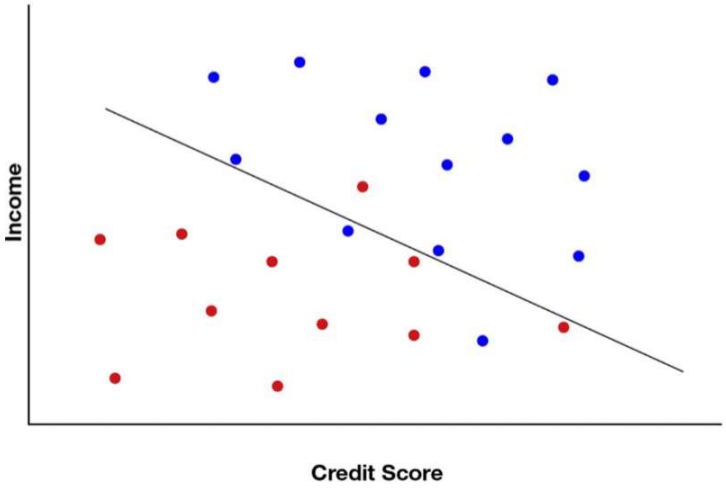
Validation Introduction

Validation: the process of assessing a model's performance to ensure it is working as intended; the process for measuring the uncertainty or variability in the model's estimates and identifying the sources of that uncertainty.

- How good is the model?
- How accurately does it classify data points?
- How well does it predict future data points?
- How often does it correctly determine an outcome?

A basic metric of evaluation is accuracy - how many of the datapoints were accurately handled by the model over all the datapoints handled by the model? In the classification example below, the model has correctly identified 21 data points out of 24, or 21/24=0.875.

However, this isn't a reliable metric to use in this case because it is too optimistic. This model was fitted to these datapoints; in other words, the model was trained on these datapoints. Validating the model using the training data will likely always end up too optimistic because the model has been learning about these specific datapoints, so it will more accurately predict on those points but be less accurate for previously unseen/new datapoints. Therefore the accuracy of 87.5% isn't reliable for new datapoints and overestimates how good the model actually is.



Data has two types of patterns, Real and Random. When training a model, the fitting matches both the Real and Random effects and we don't know which is which natively. This means that if you validate on the training data, you are matching the Real and Random effects of the training data and when the model is given new data, it only fits to the Real effects (and therefore will perform worse than it had on the training data).

- Real effects: the real relationship between attributes/features and the response/predicted variable. These effects will be the same in all datasets.
- Random effect: the data may appear to have a pattern, but that is actually caused by random chance. These effects will be different in all datasets.

Recall the birthdate example from the lectures: the model evaluates well on the training data but performs noticeably worse on previously unseen data.

Real example:

What day of the month were you born?

Error = 23	3
Error = 5	21
Error = 2	24
Error = 2	24
Error = 1	25
Error = 0	26
Error = –1	27
Error = –4	30
Error = –4	30
Error = –5	31

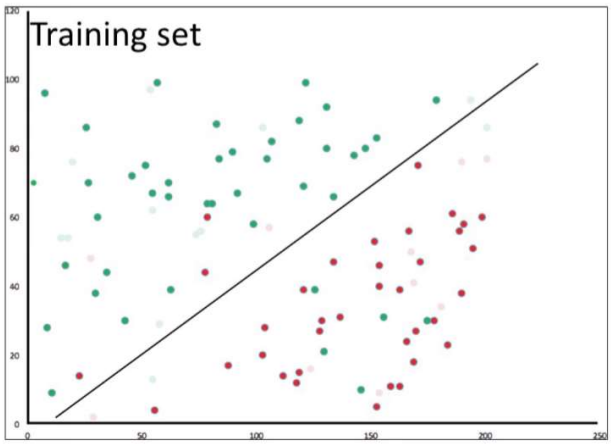
Best predictor: you were born on the 26th?

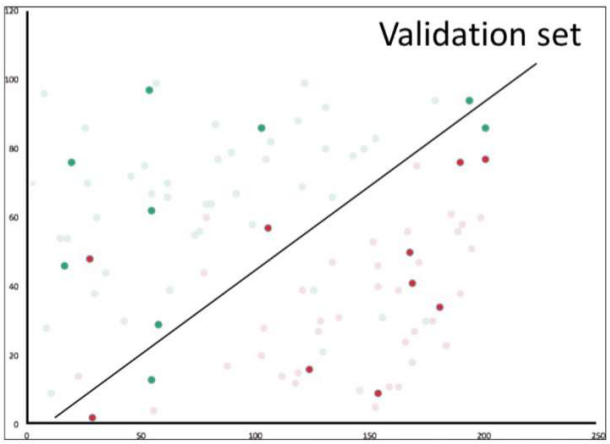
New data set

2	Error = 24
9	Error = 17
11	Error = 15
12	Error = 14
14	Error = 12
21	Error = 5
24	Error = 2
24	Error = 2
29	Error = –3
31	Error = –5

Training, Validation, and Testing Datasets

So what is a better way to evaluate model performance? Use a larger set of data to fit the model and a smaller set to evaluate the model. For example on the cc data, the training set has 90% accuracy while the validation set has 80% accuracy.





What if you want to compare different types of models, like KNN and SVM, and use the best performing one? This may still have issues of randomness, since the observed performance = real effects + random effects. So high-performing models are more likely to have above average random effects and the observed performance of a chosen model is still probably too optimistic.

Thus your dataset should be split into two to three groups. The size of each group depends on a couple of things, such as the number of samples in the data and the model you are training. Models with few hyperparameters will be easier to tune, so a smaller validation dataset is appropriate. If your model doesn't require hyperparameters, a validation set may not be required at all. If you aren't choosing between different models, a validation set may not be required.

Training Dataset: The sample of data used to fit the model, i.e. the data used to find the optimal model for a given problem and then training it. The model both sees and learns from this data.

Validation Dataset: The sample of data used to choose the best model; to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters. This set is used to frequently evaluate a model while fine-tuning the model hyperparameters. The model occasionally sees this data, but does not learn it. This may also be called the development dataset.

Testing Dataset: The sample of data used to estimate performance of a chosen model; to provide an unbiased evaluation of a final model fit on the training dataset. It is used only once a model is completely trained. The testing dataset should

be unlabeled as it provides the first real test against data the model has not seen before, providing a more accurate evaluation of the model's performance.

Splitting Data

There isn't a hard rule of how to split up the data, but below are some good rules of thumb. In general, the training set should be large so that the model learns about a good variety of important datapoints while the validation and testing sets need to be large enough that they can evaluate a variety of important datapoints.

Proportions:

- One model (only training and testing sets required):
  - 70-90% training, 10-30% testing
- Comparing models (training, validation, and testing sets required):
  - 50-70% training, the rest equally split between validation and testing.

Methods:

- Random: randomly choose a selected percentage of data points for training, testing, and validation (with no datapoint chosen with replacement; each datapoint can only be in one of the sets). This approach may accidentally introduce bias into the set by unevenly distributing the datapoints between groups (such as giving training most/all weekday datapoints, but few/no weekend datapoints).
- Rotation: take turns selecting points for each dataset. For example, with a five data point rotating sequence, a point would be chosen for each set in the following order until all datapoints have been split into a set: training-validation-training-testing-training. This approach guarantees that the data is equally separated into groups, which reduces the risk of bias being introduced to the data.
- Combined: randomly assign datapoints in groups to each set, such as 60% of weekdays and 60% of weekends into training with the rest equally distributed between validation and testing.

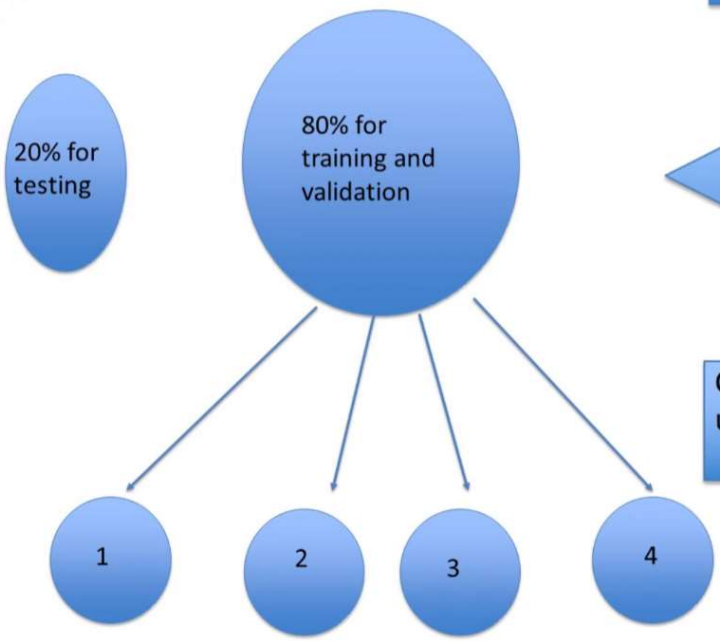
Cross Validation

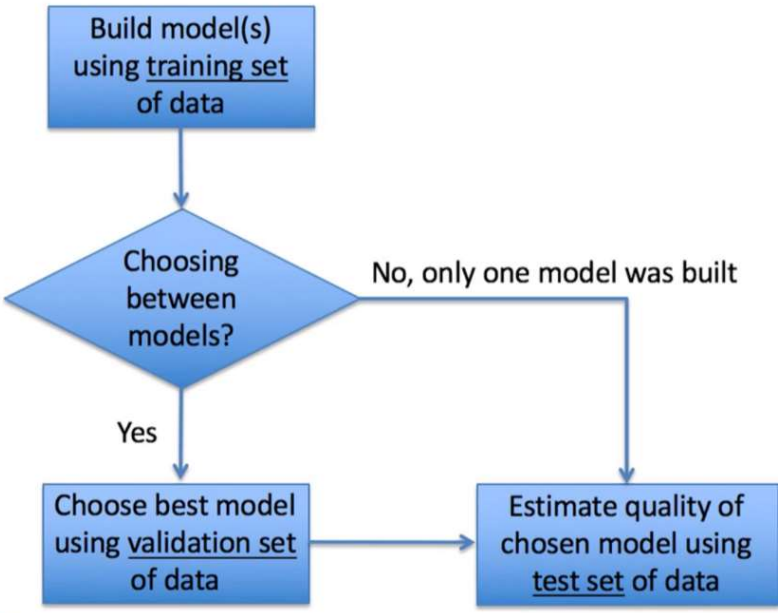
What if there are some important datasets that only appear in the validation or test sets, so the model never trains on that data or learns how to handle it? This problem can be avoided with Cross Validation. There are several kinds of Cross Validation, but this lesson focuses on k-Fold.

k-Fold Cross Validation: a resampling procedure used to evaluate models on a limited data sample so that the data can be evaluated multiple times, providing higher confidence in the model's design and evaluation. The general process is as follows:

1. The training dataset is randomly shuffled.
2. The data is split into *k* number of groups. There is no standard of what *k* value should be used, but *k* = 10 is common.
3. For each group:
   a. One of the *k* groups is used as a testing dataset while the remaining groups are the training dataset.
   b. The model is fitted on the training set and evaluated on the testing set.
   c. The evaluation is retained and the model discarded.
   d. Thus each of the *k* groups is used as a testing dataset once and used in the training set (*k-1*) times.
4. The model's performance is summarized based on the sample of evaluation scores previously retained. Commonly, the average of the *k* evaluations is used to estimate the model's true quality.

Note: none of the k models will be used as the final model. After the k-Fold Cross Validation is done, the k groups will be re-combined into the training dataset and a new model trained on all of that data.


20% for testing
80% for training and validation
1
2
3
4


Build model(s) using training set of data
Choosing between models?
No, only one model was built
Yes
Choose best model using validation set of data
Estimate quality of chosen model using test set of data