overfitting and model tuning

many modern classification and regression models are highly adaptable

capable of modeling complex relationships

BUT

can very easily overemphasize patterns that are not reproducible

this necessitates a methodological approach for evaluation to make sure the models are extensible to future, new data

almost all predictive modeling techniques have tuning parameters that enable the model to flex to find the structure in the data.

hence, we must use the existing data to identify settings for the model's parameters that yield the best and most realistic predictive performance (known as model tuning).

traditionally, this has been achieved by splitting the existing data into training and test sets. The training set is used to build and tune the model and the test set is used to estimate the model's predictive performance.

modern approaches to model building split the data into multiple training and testing sets, which have been shown to often find more optimal tuning parameters and give a more accurate representation of the model's predictive performance.

problem of overfitting

lots of modeling techniques can learn the structure of a data set perfectly — effectively this means that when the model is applied to the data on which the model was built, it correctly predicts every sample.

not only has the model learned the general patterns in the data it has also learned about each samples unique noise

this model is said to be overfit and will usually have poor accuracy when extrapolating

classic example of overfitting

every time you add a variable to a multiple regression a model's R-squared goes up

naive interpretation would be every additional predictive variable helps explain more of the response's variance

left to its own devices multiple regression will fit too closely to the training data

a large reason why modeling requires subject-matter expertise

problem of overfitting

to gauge how well the model is classifying samples one might the training set. In doing so, the estimated error rate for the model in the left- hand panel would be overly optimistic.

estimating the utility of a model by re-predicting the training set is referred to apparent performance of the model

in two dimensions, it is not difficult to visualize that one model is overfitting, but most modeling problems are in much higher dimensions

model tuning

many models have important parameters which cannot be directly estimated from the data

A parameter that cannot be directly estimated from the data is a tuning parameter because there is no analytical formula available to calculate an appropriate value

many of these parameters control the complexity of the model so poor choices for the values can result in overfitting.

one tuning parameters is generally referred to as the “cost” parameter. When the cost is large, the model will go to great lengths to correctly label every point while smaller values produce models that are not as aggressive.

model tuning

there are different approaches to searching for the best parameters

a general approach that can be applied to almost any model is to

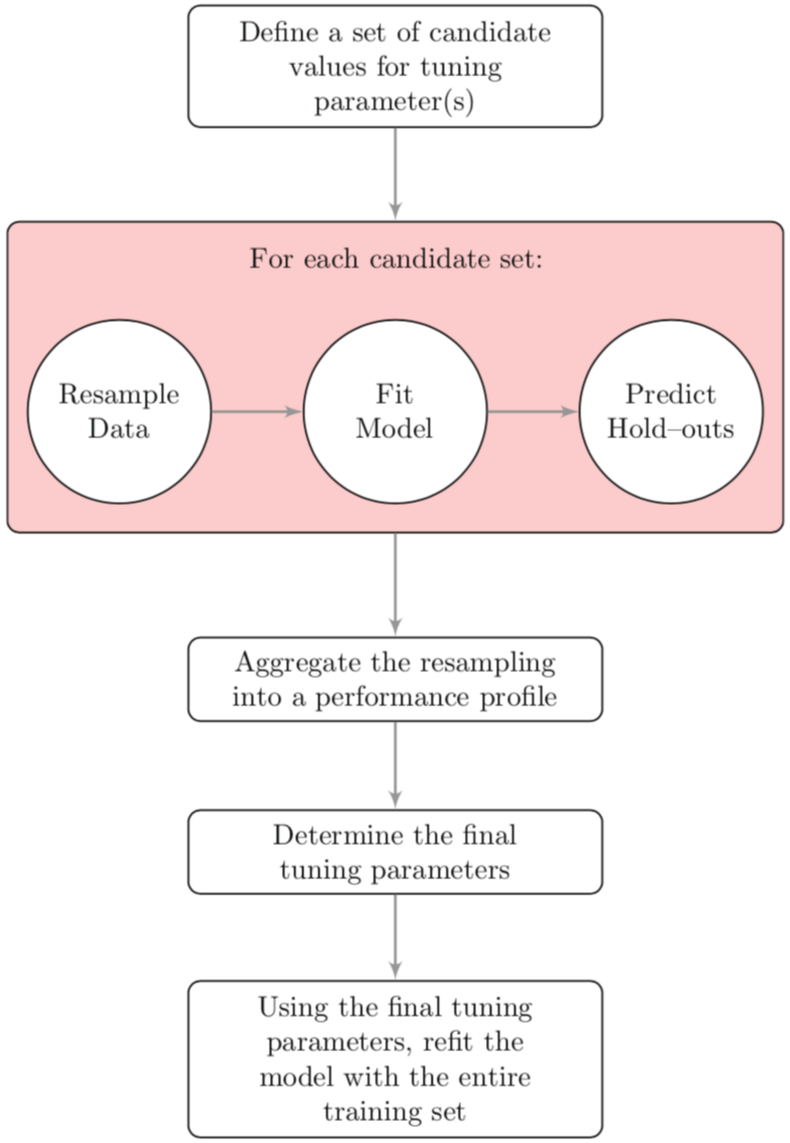
- define a set of candidate values
- generate reliable estimates of model utility across the candidates values
- choose the optimal settings

model tuning

once a candidate set of parameter values has been selected, then we must obtain trustworthy estimates of model performance

performance on the hold-out samples is aggregated into a performance profile used to determine the final tuning parameters

build a final model with all of the training data using the selected tuning parameters


Define a set of candidate values for tuning parameter(s)
For each candidate set:
Resample Data
Fit Model
Predict Hold–outs
Aggregate the resampling into a performance profile
Determine the final tuning parameters
Using the final tuning parameters, refit the model with the entire training set

model tuning

a more difficult problem is obtaining trustworthy estimates of model performance for these candidate models

a better approach is to test the model on samples that were not used for training

evaluating the model on a test set is the obvious choice, but, to get reasonable precision of the performance values, the size of the test set may need to be large

data splitting

one of the first decisions to make when modeling is to decide which samples will be used to evaluate performance

ideally, the model should be evaluated on samples that were not used to build or fine-tune the model, so that they provide an unbiased sense of model effectiveness

when a large amount of data is at hand, a set of samples can be set aside to evaluate the final model. of course, this is the `test` data set

while the `training` data set is the general term for the samples used to create the model

when the number of samples is not large, a strong case can be made that a test set should be avoided because every sample may be needed for model building

additionally, the size of the test set may not have sufficient power or precision to make reasonable judgements

much research has show that validation using a single test set can be a poor choice

data splitting

if a test set is deemed necessary, there are several methods for splitting the samples.

nonrandom approaches to splitting the data are sometimes appropriate. for example,

> If a model was being used to predict patient outcomes, the model may be created using certain patient sets (e.g., from the same clinical site or disease stage), and then tested on a different sample population to understand how well the model generalizes.
>
> Or in the case of spam filtering; if its more important for the model to catch the new spamming techniques rather than prior spamming schemes that is currently being investigated rather than the space that was evaluated years prior.

data splitting

in most cases, there is the desire to make the training and test sets as homogeneous as possible

the simplest way to split the data into a training and test set is to take a simple random sample

caveat, this does not control for any of the data attributes, such as the percentage of data in the classes

> when one class has a disproportionately small frequency compared to the others, there is a chance that the distribution of the outcomes may be substantially different between the training and test sets.

data splitting

we can also account for the outcome when splitting the data, stratified random sampling applies random sampling within subgroups (such as the classes).

in this way, there is a higher likelihood that the outcome distributions will match

when the outcome is a number, a similar strategy can be used; numeric values are broken into similar groups (e.g., low, medium, and high) and the randomization is executed within these groups

resampling techniques

generally, resampling techniques for estimating model performance operate similarly: a subset of samples are used to fit a model and the remaining samples are used to estimate the efficacy of the model

this process is repeated multiple times and the results are aggregated and summarized

the differences in techniques usually center around the method in which subsamples are chosen

k-Fold Cross-Validation

the samples are randomly partitioned into k sets of roughly equal size

a model is fit using all samples except the first subset (called the first fold)

held-out samples are predicted by this model and used to estimate performance measures

the first subset is returned to the training set and procedure repeats with the second subset held out, and so on

the k resampled estimates of performance are summarized (usually with the mean and standard error) and used to understand the relationship between the tuning parameter(s) and model utility

k-Fold Cross-Validation

a slight variant of this method is to select the k partitions in a way that makes the folds balanced with respect to the outcome

stratified random sampling creates balance with respect to the outcome

LOOCV cross-validation

another version, leave-one-out cross-validation (LOOCV), is the special case where k is the number of samples.

in this case, since only one sample is held-out at a time, the final performance is calculated from the k individual held-out predictions

additionally, repeated k-fold cross-validation replicates the procedure multiple times

for example, if 10-fold cross-validation was repeated five times, 50 different held-out sets would be used to estimate model efficacy

k-fold cross-validation

the choice of k is usually 5 or 10, but there is no formal rule

as k gets larger, the difference in size between the training set and the resampling subsets gets smaller

as this difference decreases, the bias of the technique becomes smaller (i.e., the bias is smaller for k = 10 than k = 5).

in this context, the bias is the difference between the estimated and true values of performance

k-fold cross-validation

another important aspect of a resampling technique is the uncertainty

an unbiased method may be estimating the correct value (e.g., the true theoretical performance) but may pay a high price in uncertainty

this means that repeating the resampling procedure may produce a very different value — done enough times, it will estimate the true value

k-fold cross-validation generally has high variance compared to other methods and, for this reason, might not be attractive. It should be said that for large training sets, the potential issues with variance and bias become negligible

k-fold cross-validation

from a practical viewpoint, larger values of k are more computationally burdensome

LOOCV is most computationally taxing because it requires as many model fits as data points and each model fit uses a subset that is nearly the same size of the training set

empirical results have shown that LOOCV and k =10-fold cross-validation yield similar results, indicating that k = 10 is more attractive from the perspective of computational efficiency

also, small values of k, say 2 or 3, have high bias but are very computationally efficient

k-fold cross-validation

However, the bias that comes with small values of k is about the same as the bias produced by the bootstrap but with much larger variance.

Research indicates that repeating k-fold cross-validation can be used to effectively increase the precision of the estimates while still maintaining a small bias

generalized cross-validation

for linear regression models, there is a formula for approximating the leave- one-out error rate. the generalized cross-validation (GCV) statistic does not require iterative refitting of the model to different data subsets. The formula for this statistic is the i_{th} training set outcome

$$\mathrm{GCV} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{y_i - \hat{y}_i}{1 - df/n}\right)^2$$

where y_i is the i_{th} in the training set set outcome, $\hat{y}_i$ is the model prediction of that outcome, and df is the degrees of freedom of the model. The degrees of freedom are an accounting of how many parameters are estimated by the model and, by extension, a measure of complexity for linear regression models. Based on this equation, two models with the same sums of square errors (the numerator) would have different GCV values if the complexities of the models were different.

repeated training/test splits

repeated training/test splits is also known as “leave-group-out cross-validation” or “Monte Carlo cross-validation.”

this technique simply creates multiple splits of the data into modeling and prediction sets

the proportion of the data going into each subset is controlled by the practitioner as is the number of repetitions.

the bias of the resampling technique decreases as the amount of data in the subset approaches the amount in the modeling set. A good rule of thumb is about 75–80%. Higher proportions are a good idea if the number of repetitions is large.

the number of repetitions is important

Increasing the number of subsets has the effect of decreasing the uncertainty of the performance estimates

> to get a gross estimate of model performance, 25 repetitions will be adequate if the user is willing to accept some instability in the resulting values

to get stable estimates of performance, it is suggested to choose a larger number of repetitions (say 50–200)

This is also a function of the proportion of samples being randomly allocated to the prediction set; the larger the percentage, the more repetitions are needed to reduce the uncertainty in the performance estimates

the bootstrap

a bootstrap sample is a random sample of the data taken with replacement

this means that, after a data point is selected for the subset, it is still available for further selection

the bootstrap sample is the same size as the original data set.

As a result, some samples will be represented multiple times in the bootstrap sample while others will not be selected at all. The samples not selected are usually referred to as the “out-of-bag” samples. For a given iteration of bootstrap resampling, a model is built on the selected samples and is used to predict the out-of-bag samples

in general, bootstrap error rates tend to have less uncertainty than k-fold cross-validation

however, on average, 63.2% of the data points the bootstrap sample are represented at least once, so this technique has bias similar to k-fold cross-validation when $k \approx 2$

if the training set size is small, this bias may be problematic, but will decrease as the training set sample size becomes larger

a few modifications of the simple bootstrap procedure have been devised to eliminate this bias.

the “632 method” addresses this issue by creating a performance estimate that is a combination of the simple bootstrap estimate and the estimate from re-predicting the training set

For example, if a classification model was characterized by its error rate, the 632 method would use

(0.632 × simple bootstrap estimate) + (0.368 × apparent error rate).

the bootstrap

the modified bootstrap estimate reduces the bias, but can be unstable with small samples sizes

this estimate can also result in unduly optimistic results when the model severely overfits the data, since the apparent error rate will be close to zero

choosing final tuning parameters

once model performance has been quantified across sets of tuning parameters, there are several philosophies on how to choose the final settings.

the simplest approach is to pick the settings associated with the numerically best performance estimates.

in general, it may be a good idea to favor simpler models over more complex ones and choosing the tuning parameters based on the numerically optimal value may lead to models that are overly complicated.

other schemes for choosing less complex models should be investigated as they might lead to simpler models that provide acceptable performance

data splitting recommendations

there is a strong technical case to be made against a single, independent test set

a test set is a single evaluation of the model and has limited ability to characterize the uncertainty in the results

proportionally large test sets divide the data in a way that increases bias in the performance estimates.

with small sample sizes the model may need every possible data point to adequately determine model values.

data splitting recommendations

no resampling method is uniformly better than another; the choice should be made while considering several factors

if the samples size is small, we recommend repeated 10-fold cross-validation for several reasons: the bias and variance properties are good and, given the sample size, the computational costs are not large

If the goal is to choose between models, as opposed to getting the best indicator of performance, a strong case can be made for using one of the bootstrap procedures since these have very low variance.

for large sample sizes, the differences between resampling methods become less pronounced, and computational efficiency increases in importance

once the settings for the tuning parameters have been determined for each model, the question remains

how do we choose between multiple models?

this largely depends on the characteristics of the data and the type of questions being answered. however, predicting which model is most fit for purpose can be difficult.

choosing between models

given this, we suggest the following scheme for finalizing the type of model:

1. start with several simple models that are not complete black boxes. set a baseline for accuracy

2. investigate more complex, less interpretable models that are more likely to produce higher accuracy

3. consider the tradeoffs of interpretability while also considering using the simplest model that reasonably approximates the performance of the more complex methods

choosing between models

using this methodology, the modeler can compare a to a ‘naive baseline’ as well as `performance ceiling` for the data set before settling on a model

often, a range of models will be equivalent in terms of performance so weigh the benefits of different methodologies

- computational complexity
- easy of prediction
- interpretability

German credit scoring

1,000 samples that have been given labels of good and bad credit

70% were rated as having good credit

of course the baseline accuracy to beat is 70%

credit history, employment, account status,

some are numeric, such as the loan amount

most of the predictors are categorical in nature, such as the purpose of the loan, gender, or marital status

categorical predictors were converted to “dummy variables” that related to a single category.

> for example, the applicant’s residence information was categorized as either “rent,”“own,” or “free housing” converted to three yes/no bits of information for each category

a stratified random sample of 800 customers to use for training models. The remaining samples will be used as a test set to verify performance when a final model is determined

for the credit scoring example, a nonlinear support vector machine model was evaluated over cost values ranging from 2^-2 to 2^7

each model was evaluated using five repeats of 10-fold cross-validation