Predictive performance

ST552 Lecture 23

Charlotte Wickham

2019-03-04

From last time: comments on criteria-based methods

These metrics are estimates, and like all estimates are subject to variability.

The ranking of models for one dataset might be different to another generated from the same data generating process.

There are some asymptotic results. Two common types:

- consistent for model selection: if you have enough data you will get the right model
- optimal for prediction: if you have enough data you will get the best predictions (in the sense of squared error)

Alternative estimate of model performance: use external test set, and estimate your desired metric directly. (The idea behind cross validation)

Predictive performance of models

A Tecator Infratec Food and Feed Analyzer working in the wavelength range 850 - 1050 nm by the Near Infrared Transmission (NIT) principle was used to collect data on samples of finely chopped pure meat. 215 samples were measured. For each sample, the fat content was measured along with a 100 channel spectrum of absorbances. Since determining the fat content via analytical chemistry is time consuming we would like to build a model to predict the fat content of new samples using the 100 absorbances which can be measured more easily.

How do we decide what a good model is?

How good is our model at prediction?

We build a model that takes x_0 for a new observation as input and predicts $\hat{y}_0$.

A common (but not the only) metric is the mean squared prediction error:

$$\mathrm{E}\left((y_0 - \hat{y}_0)^2\right)$$

How far on average are our predictions from the truth? (Expectation is over repeated samples from the data generating mechanism).

This error is for unseen data, so is often called the **extra-sample** or **generalization** error.

We can't find this expectation because we don't know the true model. So, we try to estimate it.

Illustration

One way, split the model in two. Build the model with one, the training set. Estimate the prediction error with the other, the test set.

```
set.seed(19128)
ind <- sample(nrow(meatspec), size = 172)
trainmeat <- meatspec[ind, ]
testmeat <- meatspec[-ind, ]

# fit model with training set
fit_meat <- lm(fat ~ ., data = trainmeat)

# predict on "unseen" data, the test set
testmeat$pred <- predict(fit_meat, testmeat)

# average squared error
mse <- with(testmeat, mean((fat - pred)^2))
# usually reported on root scale
sqrt(mse)
```

```
## [1] 3.339487
```

on average our predictions are about 3.3 percentage points away

Using the dataset for both training and evaluation leads to overconfidence

We could try to estimate the using the residuals (a.k.a training error)

$$1/n \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

but this tends to underestimate our error because we are using the same data to fit and evaluate the model.

```
sqrt(mean(residuals(fit_meat)^2)) # much too small!
```

```
## [1] 0.7320721
```

In general you can replace the mean squared error with any loss function you like, e.g median absolute error. The form will reflect what you are using the predictions for.

How could we improve our model?

```r
# fit model with training set
fit_meat <- lm(fat ~ ., data = trainmeat)
```

23-code.R Use laptops, do model selection. Evaluate on test set.

K-fold cross validation

- Extends the idea of a test and training set, by splitting the data into K sets.
- K-1 sets are used to fit the model, and the Kth to estimate the prediction error.
- Repeat K times, leaving out a different set each time.

Regularized regression

Next time. . .

Lasso and ridge.