MS&E 226: Fundamentals of Data Science

Lecture 5: In-sample estimation of prediction error

Ramesh Johari

Estimating prediction error

The road ahead

Thus far we have seen how we can select and evaluate predictive models using the train-validate-test methodology. This approach works well if we have "enough" data.

What if we don't have enough data to blindly train and validate models? We have to understand the behavior of prediction error well enough to intelligently explore the space of models.

The road ahead

Starting with this lecture:

- We develop methods of evaluating models using limited data.
- We develop measures of model performance that we can use to help us effectively search for "good" models.
- We characterize exactly how prediction error behaves through the ideas of *bias* and *variance*.

A word of caution: All else being equal, more data leads to more robust model selection and evaluation! So these techniques are not "magic bullets".

Estimating prediction error

We saw how we can estimate prediction error using validation or test sets.

But what can we do if we don't have enough data to estimate test error?

In this set of notes we discuss how we can use *in-sample* estimates to measure model complexity via *cross validation*.

Cross validation

Cross validation

Cross validation is a simple, widely used technique for estimating prediction error of a model, when data is (relatively) limited.

Basic idea follows the train-test paradigm, but with a twist:

- Train the model on a subset of the data, and test it on the remaining data
- Repeat this with *different* subsets of the data

K-fold cross validation

In detail, K-fold cross validation (CV) works as follows:

- Divide data (randomly) into K equal groups, called *folds*. Let A_k denote the set of data points $(Y_i, \mathbf{X}_i)$ placed into the k'th fold.[1]
- For $k = 1, \ldots, K$, train model on all except k'th fold. Let $\hat{f}^{-k}$ denote the resulting fitted model.
- Estimate prediction error as:

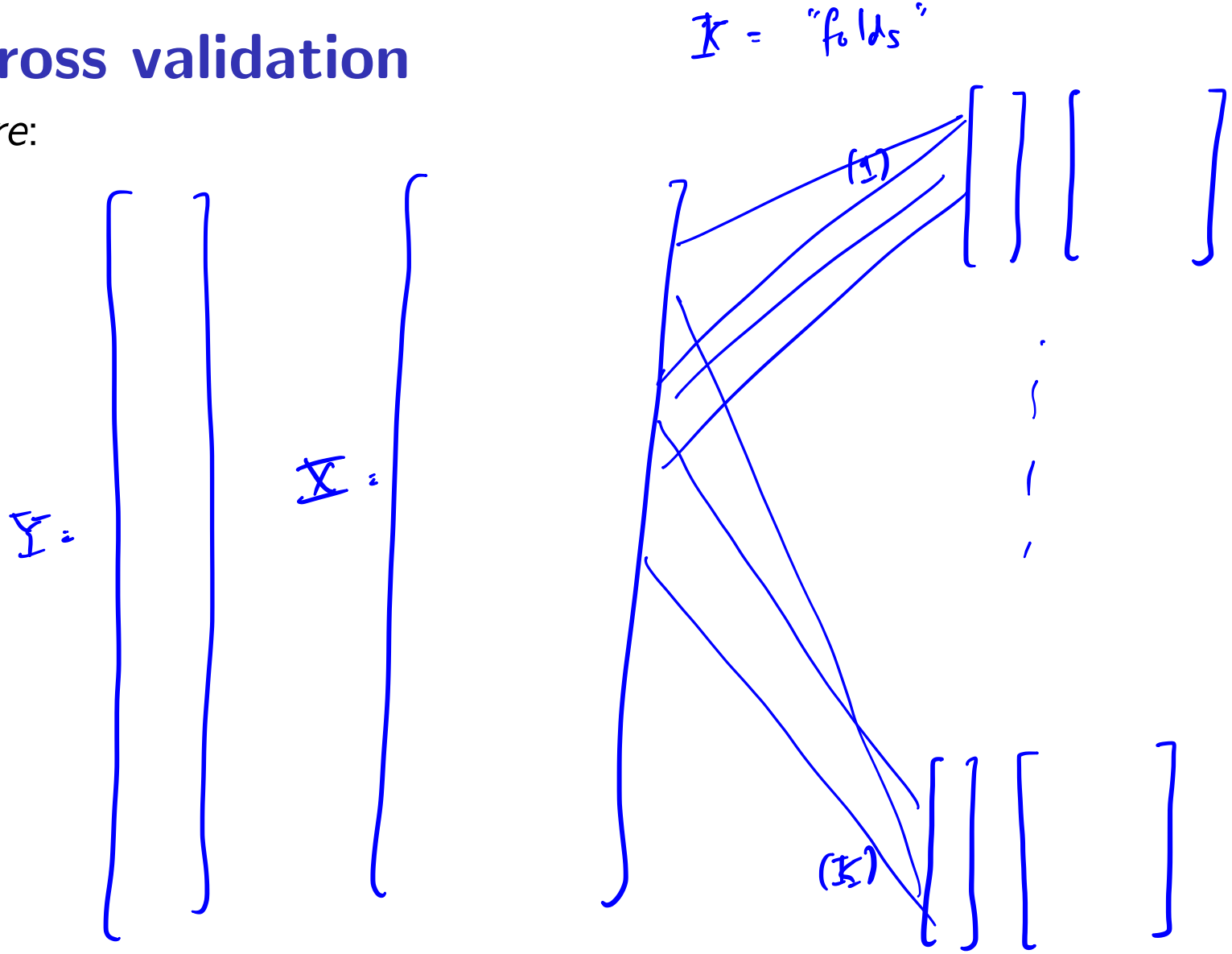
$$\text{Err}_{\text{CV}} = \frac{1}{K} \sum_{k=1}^{K} \left(\frac{1}{n/K} \sum_{i \in A_k} (Y_i - \hat{f}^{-k}(\mathbf{X}_i))^2 \right).$$

In words: for the k'th model, the k'th fold acts as a *validation set*. The estimated prediction error from CV Err_{CV} is the average of the test set prediction errors of each model.

[1] For simplicity assume n/K is an integer.

K-fold cross validation

A picture:



Using CV

After running K-fold CV, what do we do?

- We then build a model from *all* the training data. Call this $\hat{f}$.
- The idea is that Err_{CV} should be a good estimate of Err, the generalization error of $\hat{f}$.[2]

So with that in mind, how to choose K?

- If $K = n$, the resulting method is called *leave-one-out* (LOO) cross validation.
- If $K = 1$, then there is no cross validation at all.
- In practice, in part due to computational considerations, often use $K = 5$ to 10.

[2] Recall generalization error is the expected prediction error of $\hat{f}$ on new samples.

How to choose K?

There are two separate questions: how well Err_{CV} approximates the true error Err; and how sensitive the estimated error is to the training data itself.[3]

First: *How well does* Err_{CV} *approximate* Err*?*

- When $K = n$, the training set for each $\hat{f}^{-k}$ is nearly the entire training data.
  Therefore Err_{CV} will be nearly unbiased as an estimate of Err.
- When $K \ll n$, since the models use much less data than the entire training set, each model $\hat{f}^{-k}$ has higher generalization error; therefore Err_{CV} will tend to *overestimate* Err.

[3] We will later interpret these ideas in terms of concepts known as *bias* and *variance*, respectively.

How to choose K?

Second:: *How much does* Err_{CV} *vary if the training data is changed?*

- When $K = n$, because the training sets are very similar across all the models $\hat{f}^{-k}$, they will tend to have strong positive correlation in their predictions; in other words, the estimated Err_{CV} is very sensitive to the training data.
- When $K \ll n$, the models $\hat{f}^{-k}$ are less correlated with each other, so Err_{CV} is less sensitive to the training data.[4]

The overall effect is highly context specific, and choosing K remains more art than science in practice.

[4] On the other hand, note that each model is trained on significantly less data, which can also make the estimate Err_{CV} sensitive to the training data.

Leave-one-out CV and linear regression [$*$]

Leave-one-out CV is particularly straightforward for linear models fitted by OLS: there is no need to refit the model at all. This is a useful computational trick for linear models.

Theorem

Given training data $\mathbf{X}$ *and* $\mathbf{Y}$, *let* $\mathbf{H} = \mathbf{X}(\mathbf{X}^\top \mathbf{X})^{-1}\mathbf{X}^\top$ *be the* hat matrix, *and let* $\hat{\mathbf{Y}} = \mathbf{H}\mathbf{Y}$ *be the fitted values under OLS with the full training data.*
Then for leave-one-out cross validation:[5]

$$\text{Err}_{\text{LOOCV}} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{Y_i - \hat{Y}_i}{1 - H_{ii}}\right)^2.$$

Interpretation: Observations with H_{ii} close to 1 are very "influential" in the fit, and therefore have a big effect on generalization error.

[5] It can be shown that $H_{ii} < 1$ for all i.

LOO CV and OLS: Proof sketch [$*$]

- Let $\hat{f}^{-i}$ be the fitted model from OLS when observation i is left out.
- Define $Z_j = Y_j$ if $j \neq i$, and $Z_i = \hat{f}^{-i}(\mathbf{X}_i)$.
- Show that OLS with training data $\mathbf{X}$ and $\mathbf{Z}$ has $\hat{f}^{-i}$ as solution.
- Therefore $\hat{f}^{-i}(\mathbf{X}_i) = (\mathbf{HZ})_i$.
- Now use the fact that:

$$(\mathbf{HZ})_i = \sum_j H_{ij} Z_j = (\mathbf{HY})_i - H_{ii} Y_i + H_{ii} \hat{f}^{-i}(\mathbf{X}_i).$$

A hypothetical example

- You are given a large dataset with many covariates. You carry out a variety of visualizations and explorations to conclude that you only want to use p of the covariates.
- You then use cross validation to pick the best model using these covariates.
- Question: is Err_{CV} a good estimate of Err in this case?

A hypothetical example (continued)

No – You already used the data to choose your p covariates!

The covariates were chosen because they looked favorable on the training data; this makes it more likely that they will lead to low cross validation error.

Thus in this approach, Err_{CV} will typically be *lower* than true generalization error Err.[6]

MORAL: To get unbiased results, any model selection must be carried out without the holdout data included!

[6] Analogous to our discussion of validation and test sets in the train-validate-test approach.

Cross validation in R

In R, cross validation can be carried out using the `cvTools` package.

```
> library(cvTools)
> cv.folds = cvFolds(n, K)
> cv.out = cvFit(lm, formula =  ...,
                 folds = cv.folds, cost = mspe)
```

When done, `cv.out$cv` contains Err_{CV}. Can be used more generally with other model fitting methods besides `lm`.

An alternative to CV: Model scores

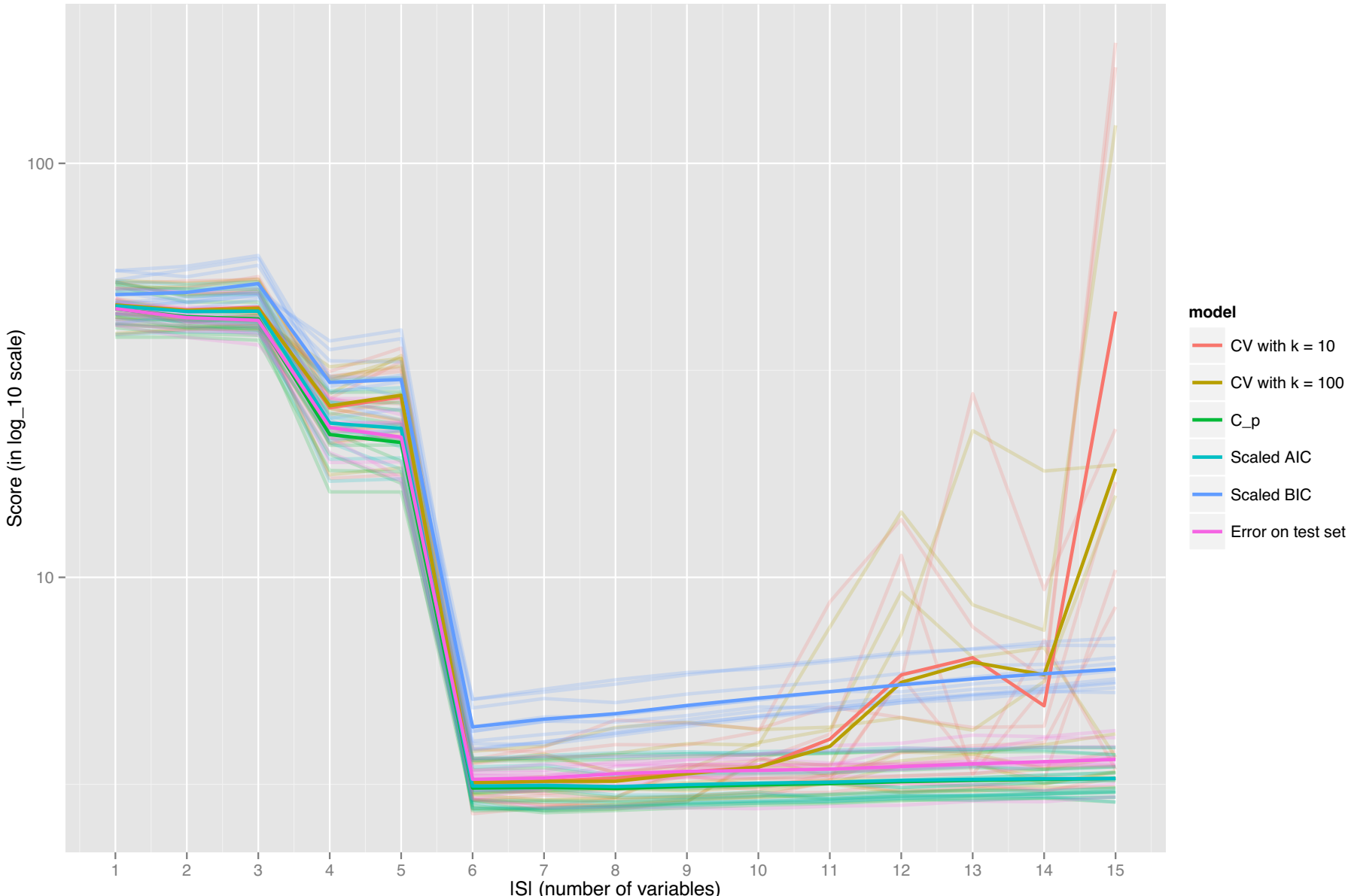
A different approach to in-sample estimation of prediction error uses the following approach:

- Choose a model, and fit it using the data.
- Compute a *model score* that uses the sample itself to estimate the prediction error of the model. (Such scores can then be used to select among competing alternative fitted models.)

Examples of model scores include C_p, AIC (the Akaike information criterion), and BIC (the Bayesian information criterion).

These scores and their underlying theory are discussed in detail in the lecture notes on "Model Scores" and "Model Selection Using Model Scores".

Simulation: Comparing C_p, AIC, BIC, CV



Train-test vs. CV

Train-test vs. CV

Suppose you have a *modeling strategy* that takes as input a training data set and produces as output a fitted model $\hat{f}$.

Example: The R call
`fm = lm(data = input, formula = Y ~ 1 + X1 + X2)` takes as input the data frame `input`, and produces as output a fitted model `fm`, using the two covariates `X1` and `X2` to predict the outcome.

Train-test vs. CV

If we are given a data set $\mathbf{X}, \mathbf{Y}$, we now have two ways to estimate generalization error:

- Train then test
- Cross validation

To make things simple, suppose we are comparing an 80%-20% train-test split to 5-fold CV. How do we choose between them?

Which $\hat{f}$?

Suppose the model of interest is *the fitted model* $\hat{f}$ *obtained by training on all the data*.

This is often the case in production systems, e.g., recommendation systems, etc.; even if train-test separation is used, in the end the model implemented in production will be the model trained on all the data.

In this case, note that 5-fold CV implements the same thing as the 80%-20% train-test split, but just does it 5 times. This is always going to yield a lower variance estimate of generalization error than just train-test.

Therefore, if the goal is to estimate the generalization error of a model that is fit on all the data, then CV will always provide the best estimate. The only restriction in this case to whether CV is used or not is computational: it may be prohibitive to run CV if the modeling strategy is quite complex.

Which $\hat{f}$?

On the other hand, suppose it is known in advance that the model of interest is one fit on only 80% of the data, and the 20% test set must be held out for model evaluation.

This is the case, e.g., in Kaggle competitions: Kaggle always maintains a held out test set that is not available to data scientists as they build their predictive models.

In this case, the test error on the 20% test set has the advantage that it will be an unbiased estimate of the true generalization error (as compared to 5-fold CV run on the training dataset).