Prediction and Crossvalidation

Mark M. Fredrickson (mfredric@umich.edu)

Computational Methods in Statistics and Data Science (Stats 406)

Cross Validation

Prediction

Suppose we have a sample of the form

$$(Y_i, \mathbf{X}_i) \overset{\text{iid}}{\sim} F$$

where $\mathbf{X}$ is a vector of **predictors** or **features**. We may or may not assume F

Goal: **predict $\hat{Y}$ at certain $\mathbf{X}$**

- $\mathbf{X}$ may be cheap/easy to sample, but Y is expensive/difficult
- Wish to get understanding of Y where we have not observed $\mathbf{X}$

In many cases we will **condition on $\mathbf{X} = \mathbf{x}$** and **model $Y \mid \mathbf{X} = \mathbf{x}$** using some function $f(\mathbf{x})$.

Examples

- Using the mean or medians of the sample Y values (no predictors).
- Linear regression: $f(x_1, x_2, \ldots) = \hat{\beta}_0 + \hat{\beta}_1 x_1 + \hat{\beta}_2 x_2 + \ldots$
- Machine learning technique (support vector machines, deep learners, random forests)

Predictions and Loss Functions

A **loss function** scores how close a prediction and the true value are:

$$l(f(\mathbf{x}^*), y \mid X = \mathbf{x}^*)$$

The goal of the loss function is to capture **how expensive is an error**?

We'll come back to loss functions, but for now a concrete example is **squared error loss**:

$$l(f(\mathbf{x}^*), y \mid \mathbf{x}^*) = (y - f(\mathbf{x}^*))^2$$

Sources of Randomness

Notice, f implicitly **depends on the original sample**:

$$f(\mathbf{x}^*) = f(\mathbf{x}^*, Y_1, \mathbf{X}_1, \ldots, Y_n, \mathbf{X}_n)$$

so l^* is a random quantity.

Additionally, if we are thinking about a new observation $Y^* = Y \mid \mathbf{x}^*$, then

$$l(f(\mathbf{x}^*), Y^*)$$

has **two sources of randomness** (the original sample and Y^*).

Operating Characteristics

As you might expect, we want to understand **operating characteristics** of loss functions. We could look at many things, but we'll focus on **expected loss**:

$$\mathrm{E}\left(l(f(\mathbf{x}^*), Y^*)\right)$$

where the expectation is taken over the joint distribution of the original sample and a new observation (equiv.: by IID assumption, a sample of size $n+1$). In other words, over many samples $(Y, \mathbf{X})$ and many new observations Y^*, **how far from the truth is our prediction on average**?

Notice: when using **squared error loss** this is **similar but different than MSE**:

$$\mathrm{E}\left((\hat{\theta} - \theta)^2\right)$$

because θ is **fixed**.

Computing and Estimating Expected Loss

As with other operating characteristics, we have several options:

- Rely on **model assumptions** to compute (e.g., linear model with Normal errors)
- Assume F and use Monte Carlo to **estimate**.
- Split the sample into **training** and **test** sets. Fit in one, predict in the other.

Want to note one way **not to compute**: use the sample:

$$\frac{1}{n}\sum_{i=1}^{n} l(f(\mathbf{x}_i), Y_i)$$

Since f and Y are **not independent** we risk **under estimating prediction error (overfitting)**.

Overfitting Example

Suppose we are predicting $Y \mid \mathbf{X} = \mathbf{x}$ using **an estimator of the conditional mean**: $\mu(\mathbf{x}) = \mathrm{E}\left(Y \mid \mathbf{x}\right)$.

We'll see some examples later in the semester of **smoothing estimators** $\hat{\mu}(\mathbf{x})$ that are **unbiased for the conditional mean**:

$$\mathrm{E}\left(\hat{\mu}(\mathbf{x})\right) = \mu(\mathbf{x})$$

Consider the case of using **squared error loss** and a new observation Y^*. What is our (true) expected loss?

True Expected Loss

$$
\begin{aligned}
\mathrm{E}\left((Y^* - \hat{\mu}(\mathbf{x}^*))^2\right) &= \mathrm{E}\left(Y^{*2}\right) - 2\mathrm{E}\left(Y^*\hat{\mu}(\mathbf{x}^*)\right) + \mathrm{E}\left(\hat{\mu}(\mathbf{x}^*)^2\right) \\
&= \mathrm{E}\left(Y^{*2}\right) - \mathrm{E}\left(Y^*\right)^2 + \mathrm{E}\left(Y^*\right)^2 + \\
&\quad \mathrm{E}\left(\hat{\mu}(\mathbf{x}^*)^2\right) - \mathrm{E}\left(\hat{\mu}(\mathbf{x}^*)\right)^2 + \mathrm{E}\left(\hat{\mu}(\mathbf{x}^*)\right)^2 - 2\mathrm{E}\left(Y^*\hat{\mu}(\mathbf{x}^*)\right) \\
&= \mathrm{Var}\left(Y^*\right) + \mathrm{Var}\left(\hat{\mu}(\mathbf{x}^*)\right) + \mathrm{E}\left(Y^*\right)^2 - 2\mathrm{E}\left(Y^*\hat{\mu}(\mathbf{x}^*)\right) + \mathrm{E}\left(\hat{\mu}(\mathbf{x}^*)\right)^2 \\
&= \mathrm{Var}\left(Y^*\right) + \mathrm{Var}\left(\hat{\mu}(\mathbf{x}^*)\right) + \mu(\mathbf{x}^*)^2 - 2\mathrm{E}\left(Y^*\right)\mathrm{E}\left(\hat{\mu}(\mathbf{x}^*)\right) + \mu(\mathbf{x}^*)^2 \\
&= \mathrm{Var}\left(Y^*\right) + \mathrm{Var}\left(\hat{\mu}(\mathbf{x}^*)\right) + 2\mu(\mathbf{x}^*)^2 - 2\mu(\mathbf{x}^*)^2 \\
&= \mathrm{Var}\left(Y^*\right) + \mathrm{Var}\left(\hat{\mu}(\mathbf{x}^*)\right)
\end{aligned}
$$

Using the same sample to estimate prediction error

What if we didn't have a new Y^* and just used the sample average squared error?

$$\frac{1}{n}\sum_{i=1}^{n}\left[\hat{\mu}(\mathbf{x}_i) - Y_i\right]^2$$

This is generally **not unbiased** for the true prediction error because the $\hat{\mu}$ and Y_i are not independent and

$$\mathrm{E}\left(\hat{\mu}(\mathbf{x}_i) Y_i\right) \neq \mathrm{E}\left(\hat{\mu}(\mathbf{x}_i)\right) \mathrm{E}\left(Y_i\right)$$

Solution: find another sample that is independent of the original to make our prediction error estimate.

Leave-one-out prediction

Recall, the **jackknife** estimates bias by **repeatedly dropping one observation**.

$$T_j = T(X_1, \ldots, X_{j-1}, X_{j+1}, \ldots, X_n)$$

Since we assumed the observations were **independent**:

$$X_j \perp X_1, \ldots, X_{j-1}, X_{j+1}, \ldots X_n$$

it is also the case that

$$X_j \perp T_j$$

Idea: we could **predict** each X_j with T_j and **average over all** n. We call this "leave-one-out" (LOO) prediction.

Example: Student academic test scores

ELSEVIER

Available online at www.sciencedirect.com

SciVerse ScienceDirect

Research in Social Stratification and Mobility 30 (2012) 97–112

Research in Social Stratification and Mobility

http://elsevier.com/locate/rssm

Differences between Hispanic and non-Hispanic families in social capital and child development: First-year findings from an experimental study

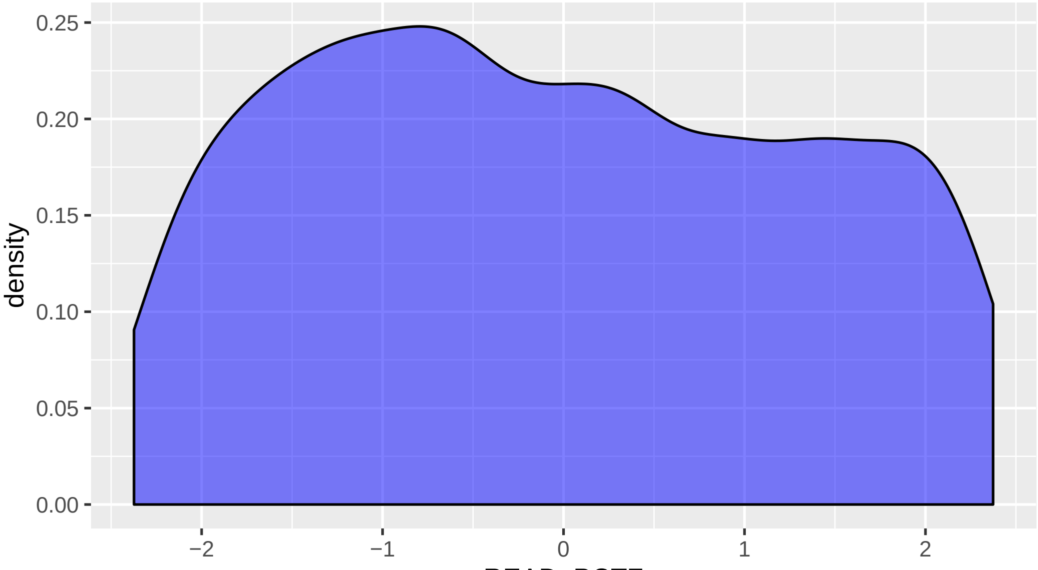
Adam Gamoran [a,*], Ruth N. López Turley [b], Alyn Turner [a], Rachel Fish [a]

[a] University of Wisconsin-Madison, , United States

[b] Rice University, , United States

Received 15 March 2011; received in revised form 5 August 2011; accepted 9 August 2011

Reading test scores



Question: what would have less prediction error, the mean or mode?

Suppose we want to predict the reading score of a student not in the original study. What would have smaller error:

- The sample mean of the students in the study?

```
> mean(reading, na.rm = TRUE)
[1] -0.04814175
```

- The sample mode of a smoothed density?

```
> d <- density(reading)
> d$x[which.max(d$y)]
[1] -0.790038
```

Using the mean

Observe that if we drop X_j from the mean, we get:

$$\bar{X}_j = \frac{1}{n-1}\sum_{i \neq j} X_i = \frac{1}{n-1}\left(\left[\sum_{i=1}^{n} X_i\right] - X_j\right)$$

```
> n <- length(reading)
> barxj <- 1 / (n - 1) * (sum(reading) - reading)
```

We will estimate our squared error with the average of $(\bar{X}_j - X_j)^2$.

```
> err_mean <- barxj - reading
> (err2_mean <- mean(err_mean^2))

[1] 1.800303
```

Using the mode

```
> modej <- sapply(1:n, function(i) {
+     d <- density(reading[-i])
+     d$x[which.max(d$y)]
+ })
> err_mode <- modej - reading
> (err2_mode <- mean(err_mode^2))
```

```
[1] 2.349879
```

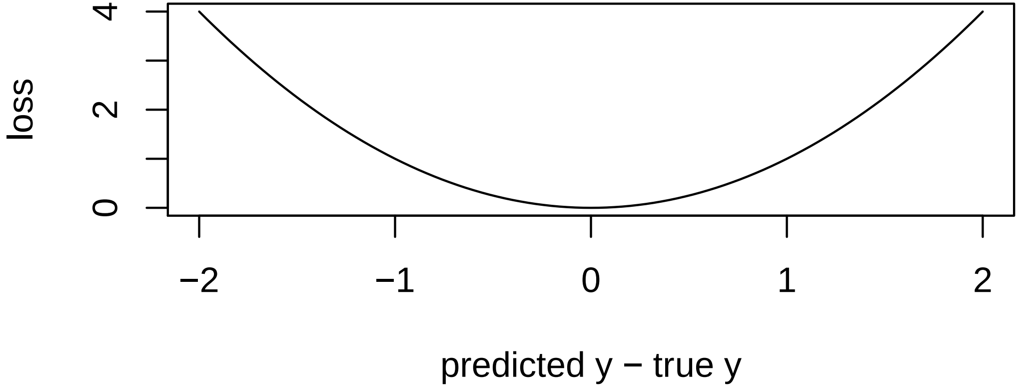
Which is larger than the predicted error using the mean:

```
> err2_mean / err2_mode
```

```
[1] 0.7661257
```

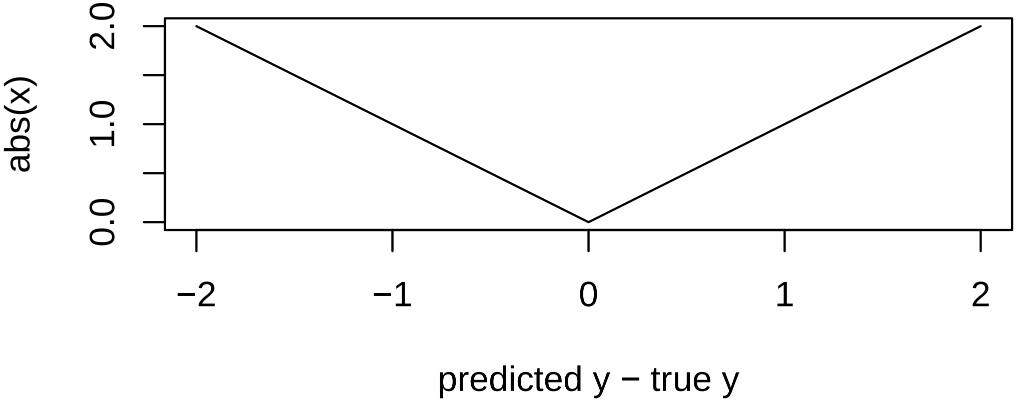
More on loss functions

We have been using **squared loss**

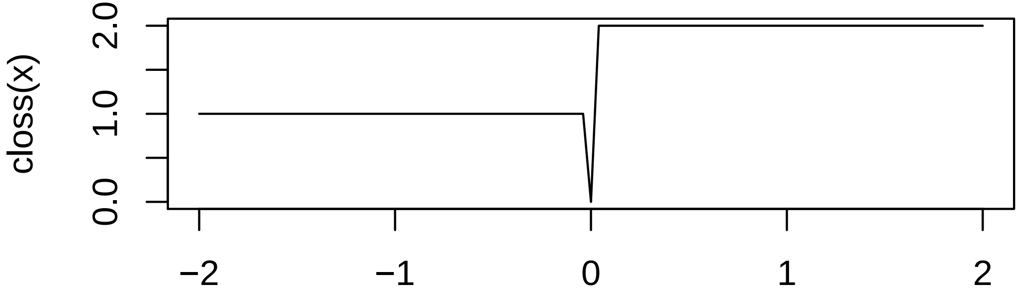


but we could consider other loss functions.

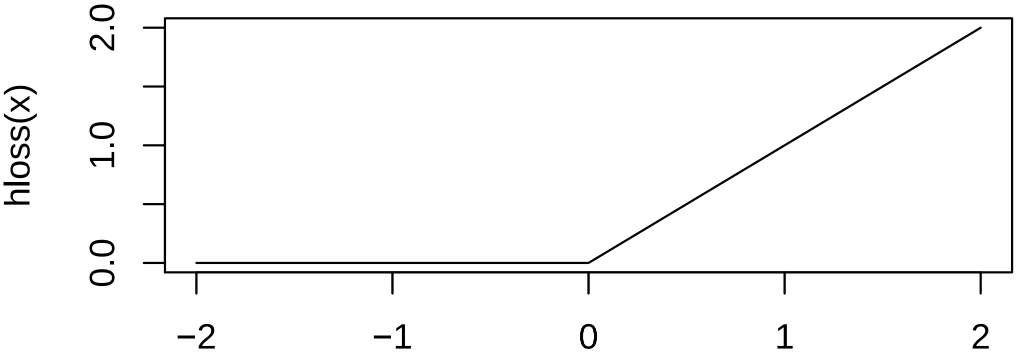
Absolute loss: $|Y^* - f(\mathbf{x}^*)|$



Constant loss: $a \times I(\hat{Y} > Y^*) + b \times I(\hat{Y} < Y^*)$



Hinge loss: $\max(0, \hat{Y} - Y^*)$



Different Loss, Different Conclusion

```
> mean(err_mean^2) / mean(err_mode^2) # squared error
[1] 0.7661257
> mean(abs(err_mean)) / mean(abs(err_mode)) # absolute error
[1] 0.9217158
> mean(closs(err_mean)) / mean(closs(err_mode)) # constant loss
[1] 1.121518
> mean(hloss(err_mean)) / mean(hloss(err_mode)) # hinge loss
[1] 2.268704
```

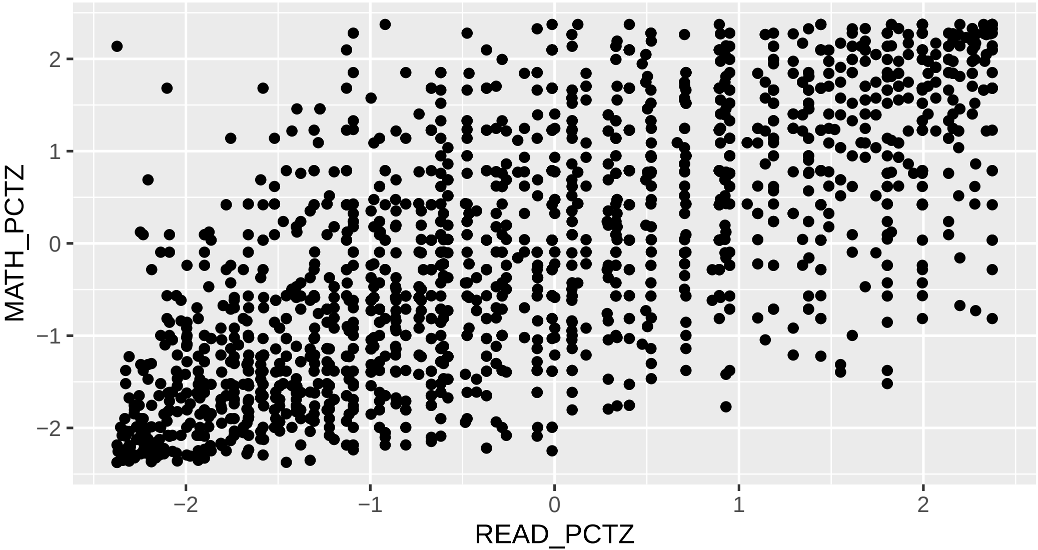
Picking a loss function

Now that we know we can work with different loss functions, **which should we pick?**

Best way: write down how costly mistakes are and deduce proper loss function.

More commonly, we use loss functions that are common to a discipline or industry.
(Similar to picking $\alpha = 0.05$ or $\alpha = 0.001$)

Predicting Math from Reading Scores



Using a linear model

We recently spent time working with a **linear model**:

$$Y = \beta_0 + \beta_1 X + R$$

where Y is an **outcome**, X is a **predictor**, and R is a **residual term**.

Y is often sometimes referred to as the **independent variable** and X as the **dependent variable**.

We often use **ordinary least squares** (OLS) to fit our model, which minimizes the within sample squared error:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - (\beta_0 + \beta_1 x_i))^2$$

If the **residual terms are Normal**, then OLS minimizes prediction error, but what if errors aren't Normal?

Cross Validation

On the previous example, we used a "leave-one-out" strategy, where we predicted each point using a model built without that point.

For this example, we'll generalize this idea by **leaving out half of the sample** and **predicting on the other half**.

We could just use a single split into a **training set** and a **testing set**.

We will repeat this process many times, **cross validation**.

Basic model fitting

```
> mod_lm <- lm(math ~ read, data = read_math)
> coef(mod_lm)

(Intercept)        read
 0.08581827  0.72495925
```

Cross validating

```
> k <- 1000
> n <- dim(read_math)[1]
> half <- round(n/2)
> err_lm <- replicate(k, {
+     rand.order <- sample.int(n)
+     train.idx <- rand.order[1:half]
+     test.idx <- rand.order[(half + 1):n]
+
+     mod <- lm(math ~ read, read_math[train.idx, ])
+     preds <- predict(mod, newdata = read_math[test.idx, ])
+     read_math$math[test.idx] - preds
+ })
```

Alternative to OLS

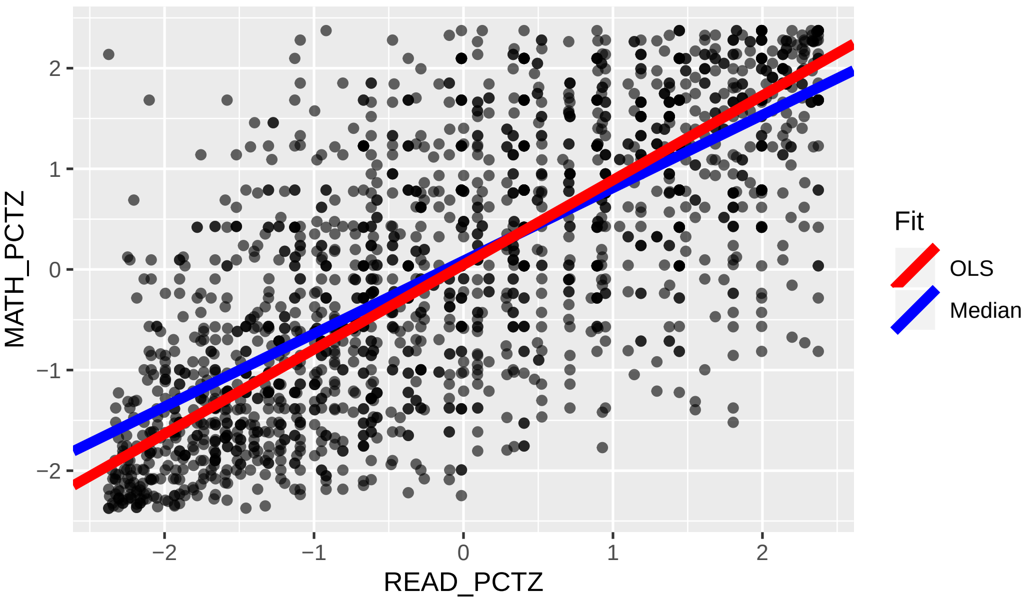
Let's compare the error we get from prediction using OLS to that we would get if pick β_0 and β_1 to minimize **absolute error** (**median regression**):

$$\min_{\beta} \sum_{i=1}^{n} |y_i - (\beta_0 + \beta_1 x_i)|$$

The `quantreg` package provides a routine to find β_0 and β_1:

```
> library(quantreg)
> mod_rq <- rq(math ~ read, data = read_math)
> coef(mod_rq)
```

```
(Intercept)        read
 0.04719007  0.84076874
```



MATH_PCTZ
2
1
0
−1
−2
−2
−1
0
1
2
READ_PCTZ
Fit
OLS
Median

```
> err_abs <- replicate(k, {
+     rand.order <- sample.int(n)
+     train.idx <- rand.order[1:half]
+     test.idx <- rand.order[(half + 1):n]
+
+     mod <- rq(math ~ read, data = read_math[train.idx, ]) # only differen
+     preds <- predict(mod, newdata = read_math[test.idx, ])
+     read_math$math[test.idx] - preds
+ })
```

Comparing OLS and median regression

```
> mean(err_lm^2) / mean(err_abs^2)
[1] 0.9712884
> mean(abs(err_lm)) / mean(abs(err_abs))
[1] 1.019969
```

Other Cross Validation Uses: Picking a model

We've used cross validation for **estimating prediction error**. Other uses include selecting a model to fit:

$$Y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 \quad \text{vs.} \quad Y = \beta_3 + \beta_4 x_1$$

The first model will always have smaller **in-sample prediction error**, but can **overfit** if x_2 actually is not important.

Also, using the same data set to pick a model as **perform inference** can lead to biased estimates, improper coverage rates, etc. **Post-selection inference** is a hot topic today, and many techniques boil down to cross-validation.

Tuning parameters

Many statistical methods have **tuning parameters**. For example the **LASSO shrinkage estimator**:

$$\min_{\beta} \sum_{i=1}^{n} (y_i - (\sum_{j=0}^{p} \beta_j x_{ji}))^2 + \lambda \sum_{j=1}^{n} |\beta_j|$$

This method tries to make β_j small, often taking them to be zero.

This is traded off the squared error using the tuning parameter λ. CV can be used to pick λ.

We'll see examples later, for example picking a **"bandwidth" for density estimation.**

Summary

- **Prediction** guesses an unobserved Y^* based on observed $\mathbf{x}^*$ based on a prediction function $f(\mathbf{x}^*)$ which is created using a sample $(Y_i, \mathbf{X}_i)$.
- The quality of the prediction is given by **loss function** $l(Y^*, f(x^*))$
- There are **two sources of randomness**, the original sample and the Y^*.
- Key operating characteristic: **expected loss**
- Loss functions encode **costs of making mistakes** in predictions.
- Estimating expected loss can be achieved using **independent training and test sets**
- Many training/test splits are called **crossvalidation**.