Non-quadratic Regularizers

Sewoong Oh

CSE/STAT 416
University of Washington

Regularizers

- consider a linear predictor

$$f(x) \;=\; w_0 + w_1 x[1] + w_2 x[2] + \cdots + w_d x[d]$$

- if $|w_i|$ is large then the predictor is very **sensitive** to small changes in x_i lead to large changes in the prediction

- this suggests that we would like w or $(w_{1:d}$ if $x[0] = 1)$ not to be large

- recall Ridge regression with **quadratic** or **L2 regularizer**

$$r(w) \;=\; w_1^2 + w_2^2 + \cdots + w_d^2$$

this penalizes having large parameters

L1 Regularizer

- **sum absolute** or **L1 regularizer** uses

$$r(w) \;=\; |w_1| + |w_2| + \cdots + |w_d|$$

- this is the same as **L1 norm** of the weight vector

$$\|w_{1:d}\|_1 \;\triangleq\; |w_1| + |w_2| + \cdots + |w_d|$$

- we write **L2 norm** (the **Euclidean norm**) as

$$\|w_{1:d}\|_2 \;\triangleq\; \sqrt{w_1^2 + w_2^2 + \cdots + w_d^2}$$

  such that the quadratic regularizer is

$$r(w) \;=\; \|w_{1:d}\|_2^2$$

- they are both members of the **p-norm family**, defined as

$$\|w_{1:d}\|_p \;\triangleq\; (|w_1|^p + \cdots + |w_d|^p)^{1/p}$$

Lasso regression

- we use squared loss $\text{MSE} = \frac{1}{N}\sum_{i=1}^{N}(\hat{y}_i - y_i)^2$
- with L2 regularizer is called **Ridge regression**

$$\text{minimize}_w = \text{MSE}(w) + \lambda\|w\|_2^2$$

- with L1 regularizer is called **Lasso regression**

$$\text{minimize}_w = \text{MSE}(w) + \lambda\|w\|_1$$

- widely used in machine learning
- since it is a convex function, can be efficiently minimized
- it has interesting properties, making it attractive in practice (sparsification)

Sparse coefficient vectors via L1 regularization

Sparse coefficient vector

- suppose w is sparse, i.e. many of its entries are zero

- prediction $\hat{y} = w^T x$ does not depend on features of x_i for which $w_i = 0$

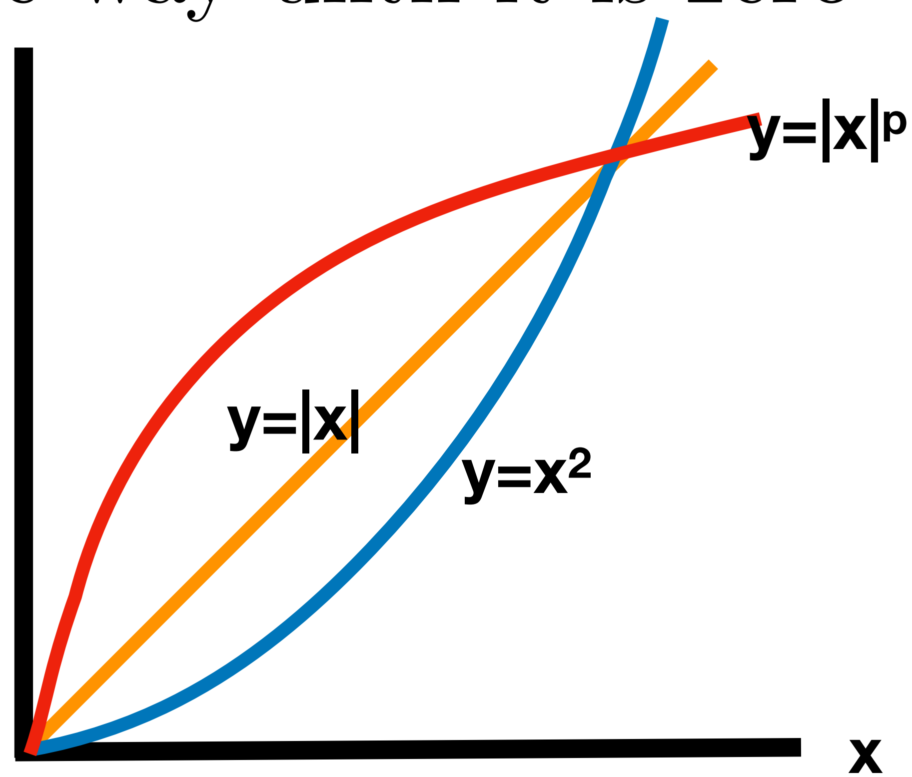
- this means we select **some** features to use (i.e. those with $w_i \neq 0$)

- (potential) practical benefits of **sparse** w
  - true model might be sparse in real applications
  - Sparsity (i.e. the number of features used in prediction) is the simplest measure of complexity of a model
  - Makes prediction model **simpler to interpret**
  - But manually engineering correct sparse set of features is extremely challenging

Using L1 regularization leads to sparse coefficient vectors

- $r(w) = \|w\|_1$ is called a sparsifying regularizer

- rough idea:
  - for L2 regularizer, once w_i is small, w_i is very small
  - so not much incentive to make coefficients go all the way to zero
  - for L1 regularizer, incentive to make w_i smaller keeps up all the way until it is zero

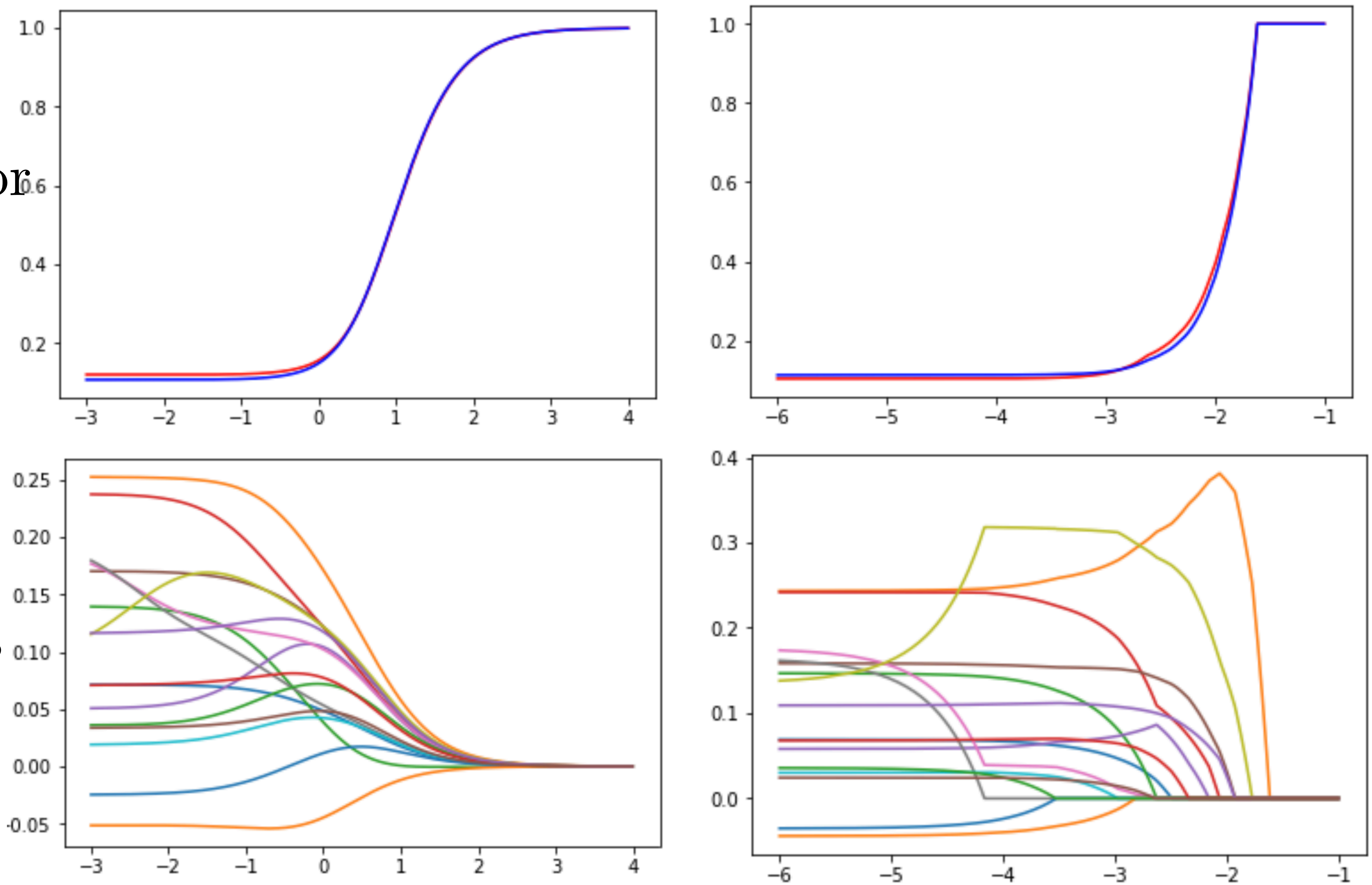


Example: house price

test error is red and train error is blue

error

w_i's



$\log_{10} \lambda$

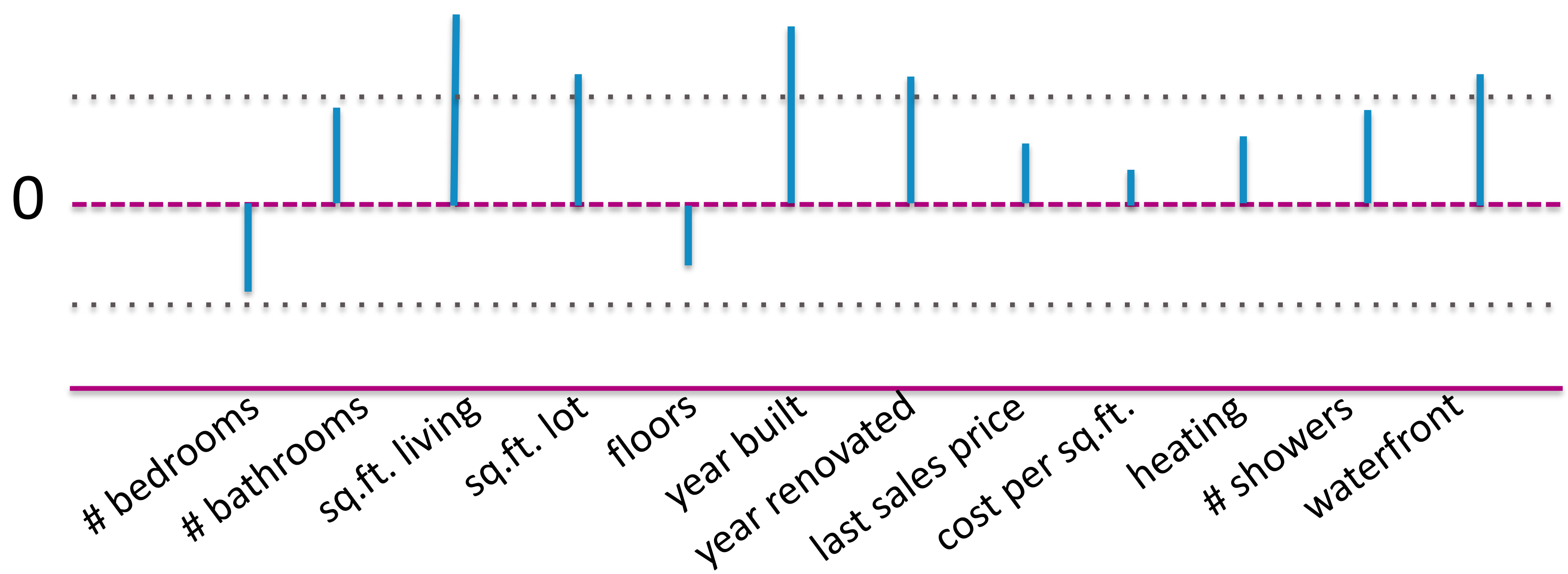
Ridge regression

$\log_{10} \lambda$

Lasso regression

Selecting sparse features based on Ridge regression (L2 regularizer) can be problematic

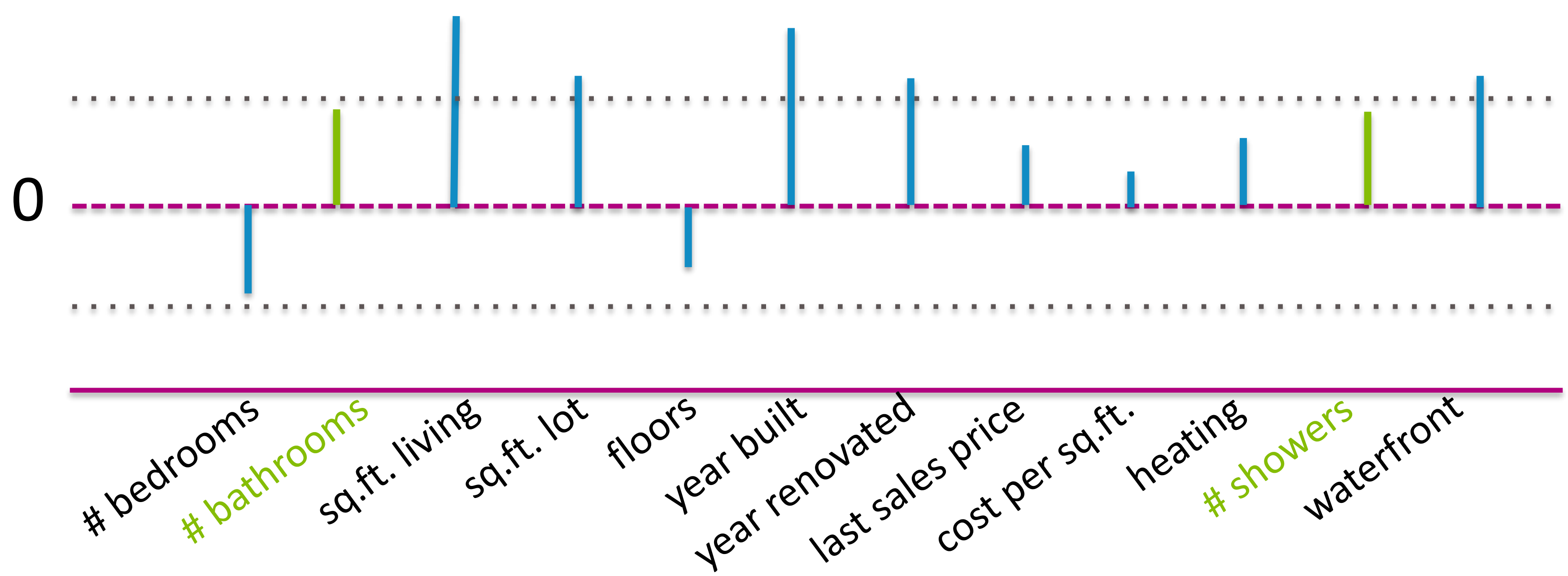
- sometimes sparse features are desired in practice
- consider running the following sparse feature selection method
  - run Ridge regression, with optimal lambda
  - Set to zero (shrink) those parameters that are smaller than a threshold



- Set threshold in order to keep the top 5, for example, parameters
- What is wrong with this approach?

Selecting sparse features based on Ridge regression (L2 regularizer) can be problematic

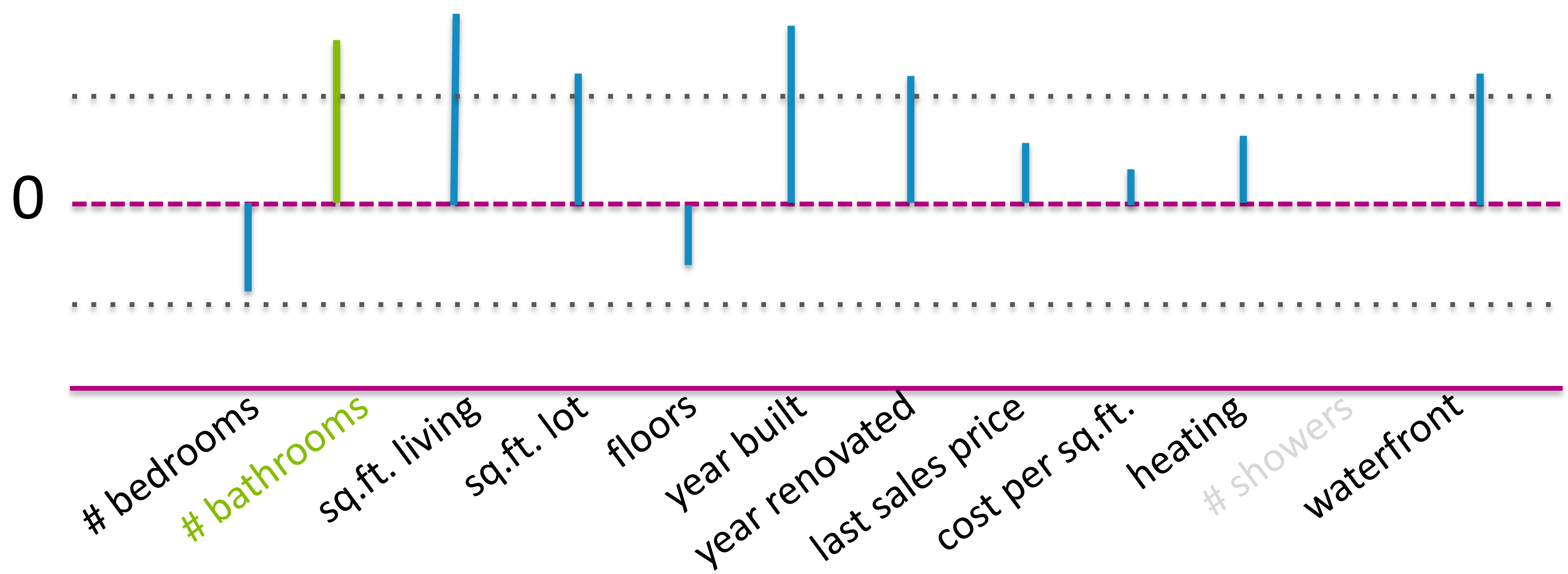
- sometimes sparse features are desired in practice
- consider running the following sparse feature selection method
    - run Ridge regression, with optimal lambda
    - shrink parameters that are smaller than a threshold



- nothing measuring bathrooms is included!!

Selecting sparse features based on Ridge regression (L2 regularizer) can be problematic

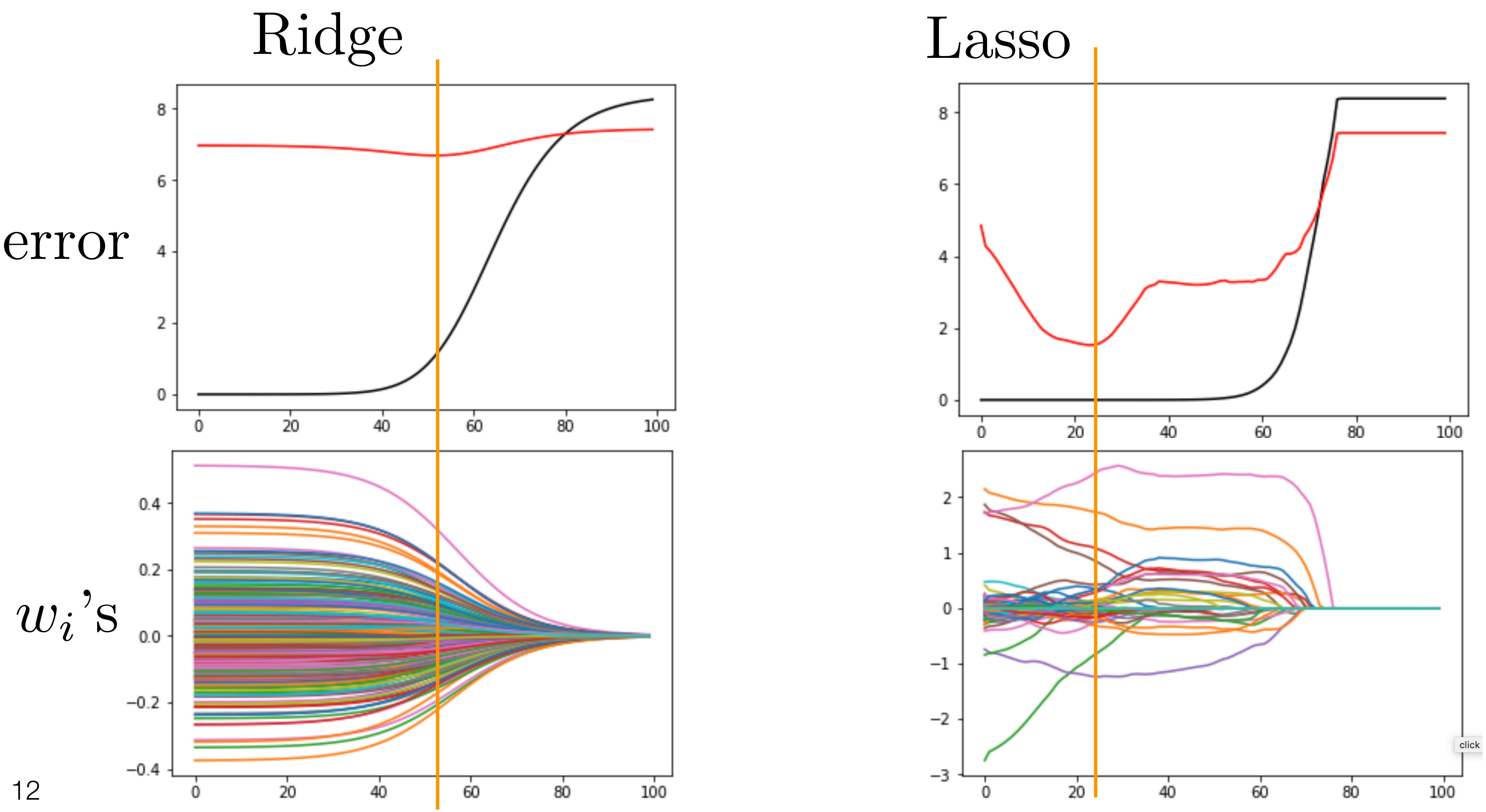
- If only one of the features were included when running Ridge regression, it would have survived



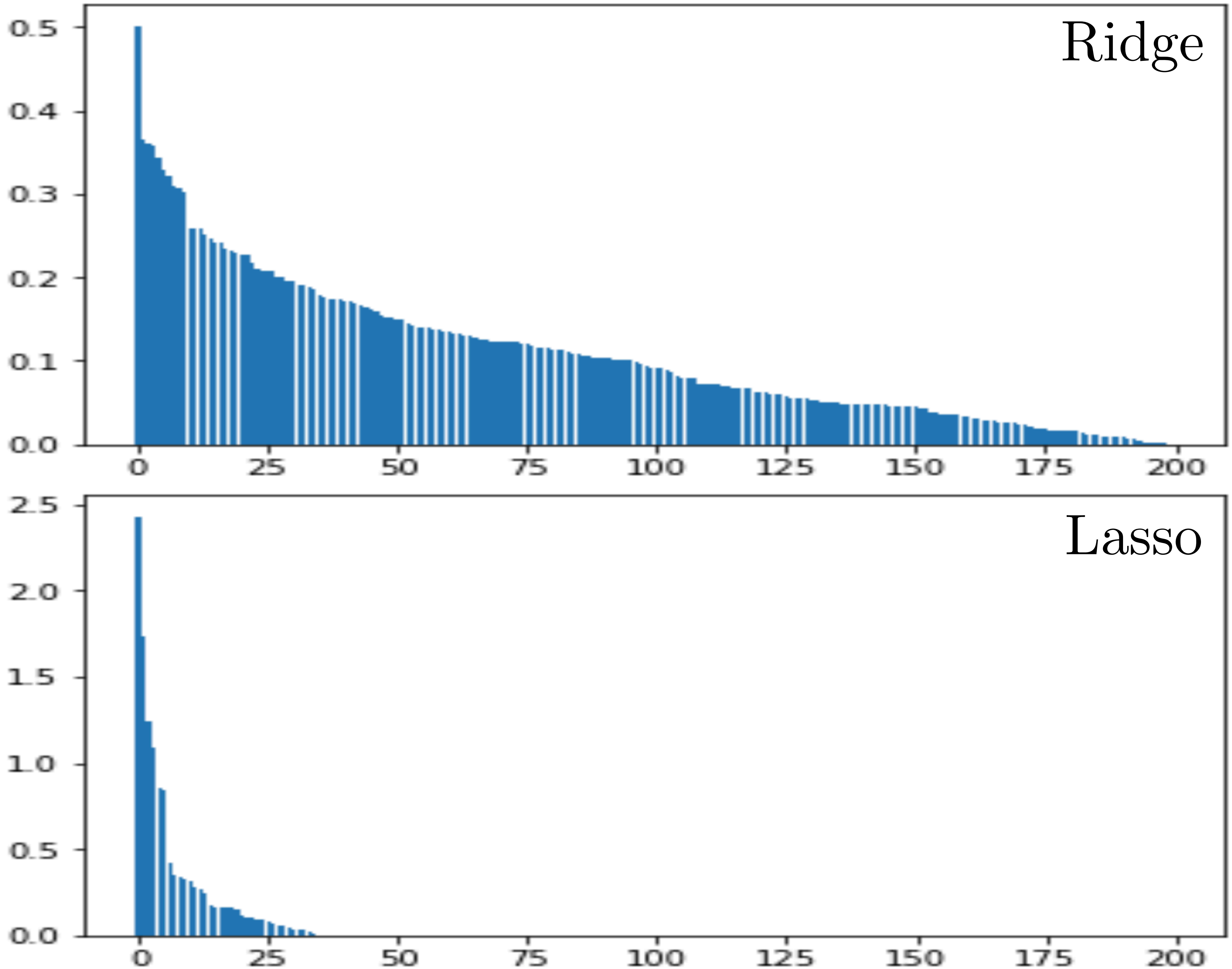
- thresholding Ridge regression parameters unnecessarily penalizes multiple similar features
- Lasso is a more principled way of selecting sparse features

Lasso regression naturally gives sparse features

- feature selection with Lasso regression
  - choose lambda based on regularization path with test data
  - keep features with largest parameters in w
  - retrain with lambda=0

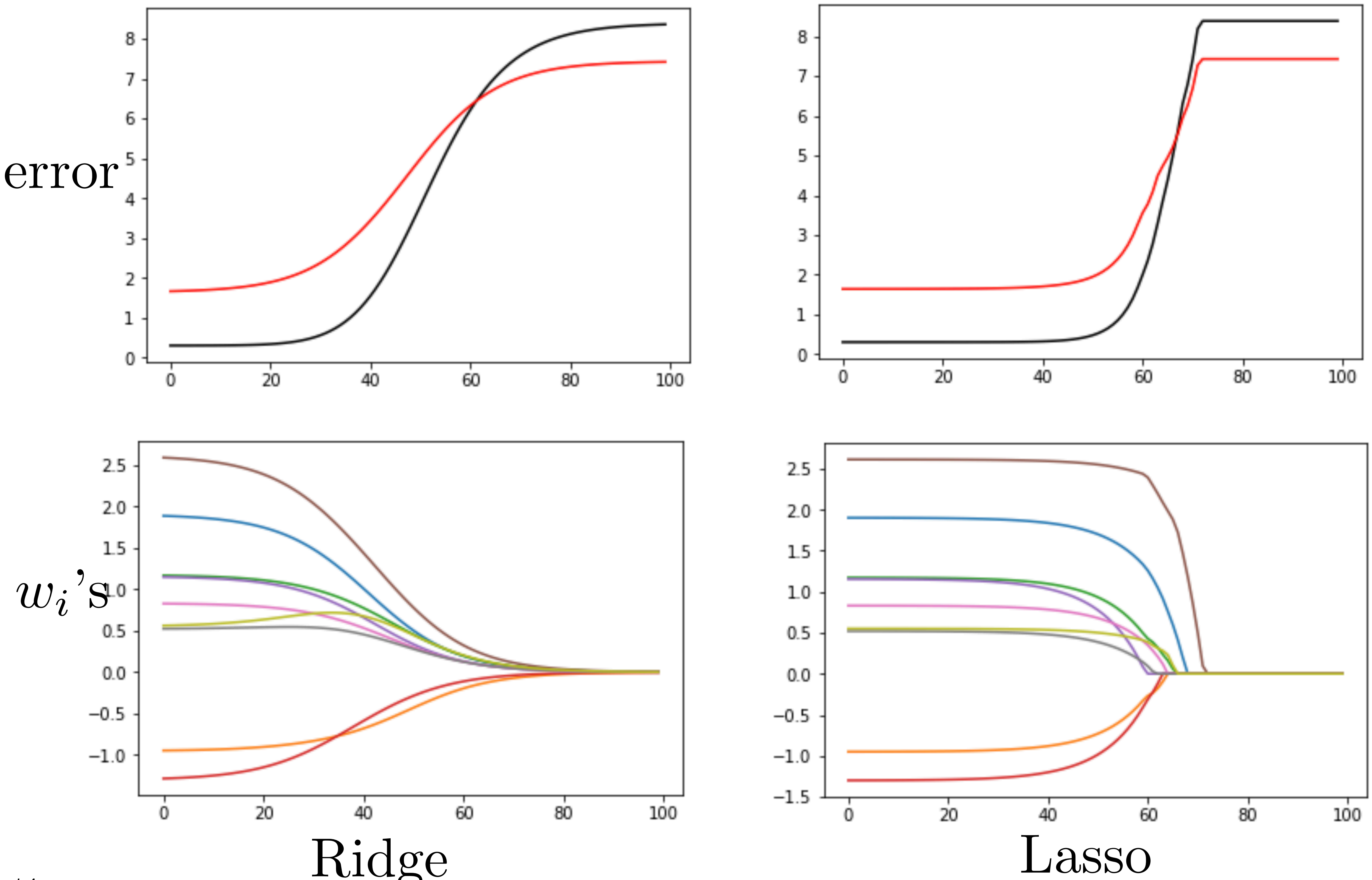


- at optimal lambda, the sorted $|w_i|$'s are
- Lasso has only 35 non-zero components



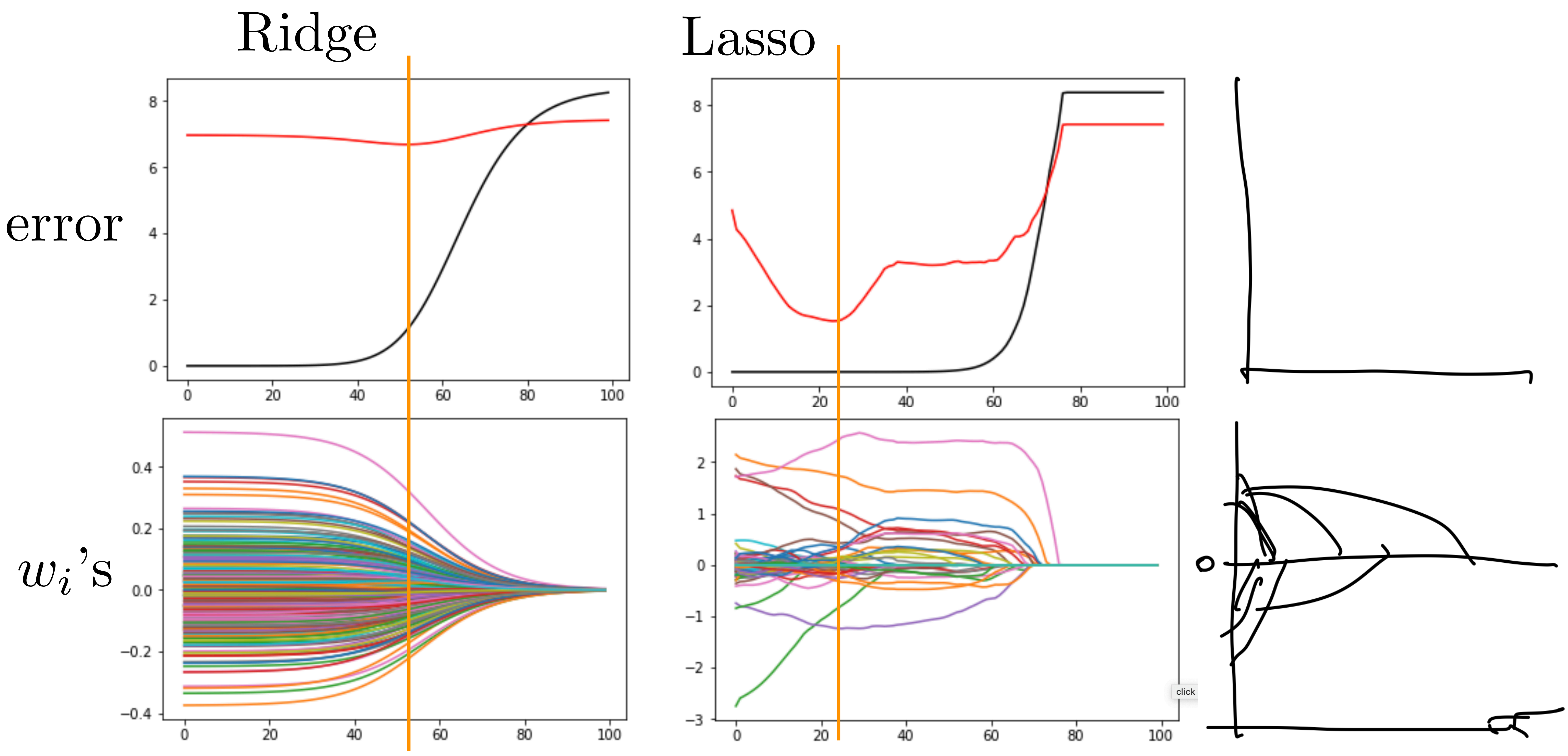
After retrain

- Retrain with only 9 features identified by lasso



- The test error is small and robust for broad range of lambda

- What if we use p-norm regularizer with p<1 ?

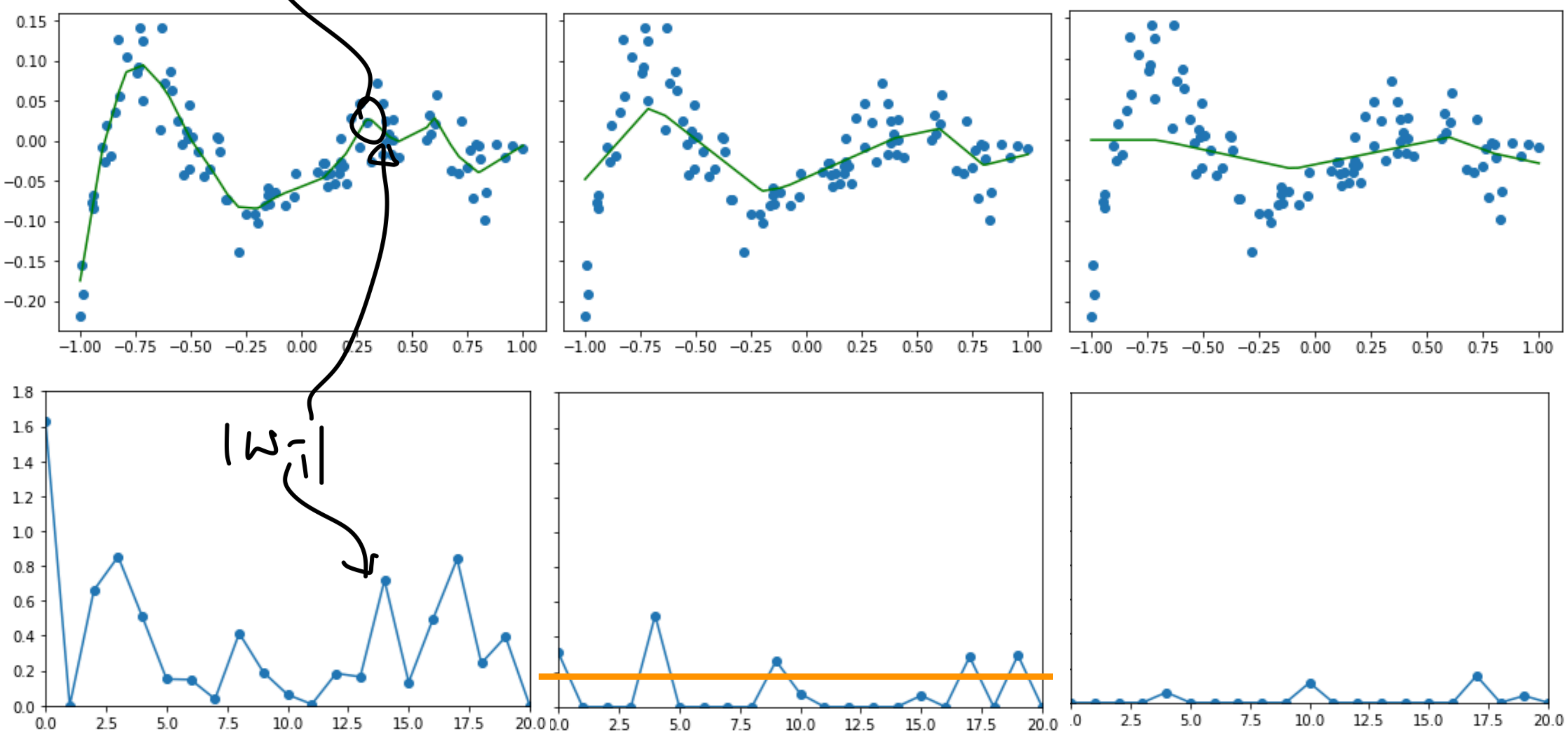


Example: piecewise-linear fit

- We use Lasso on the piece-wise linear example

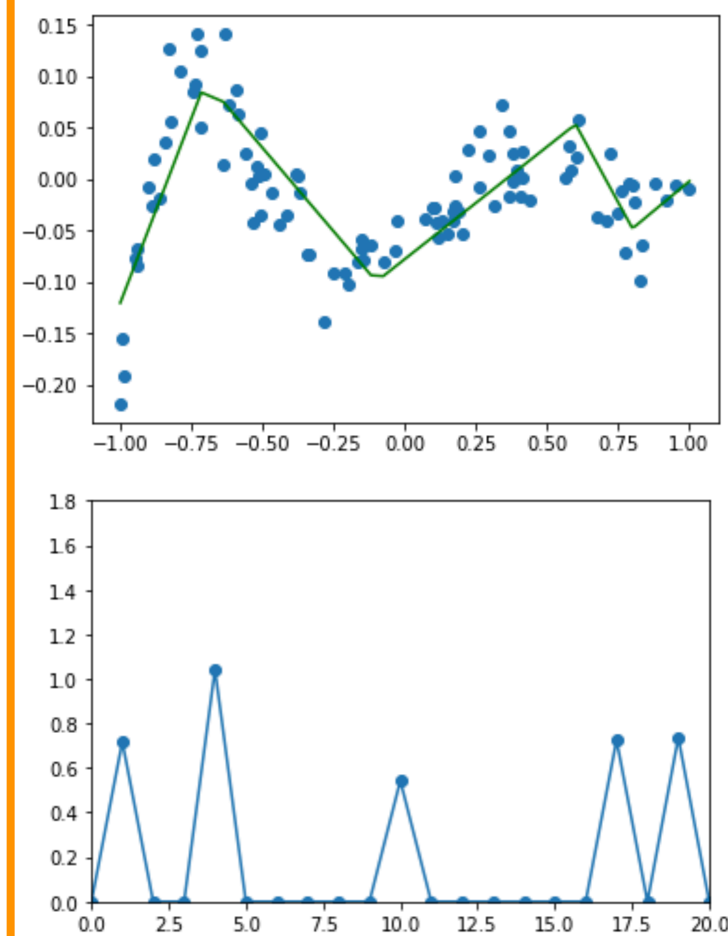
$$h_0(x) = 1$$
$$h_i(x) = [x + 1.1 - 0.1i]^+$$

$$\text{minimize}_w = \text{MSE}(w) + \lambda \|w\|_1$$



$\lambda = 10^{-8}$ $\lambda = 10^{-4}$ $\lambda = 2 \times 10^{-4}$

$$\text{minimize}_w = \text{MSE}(w)$$



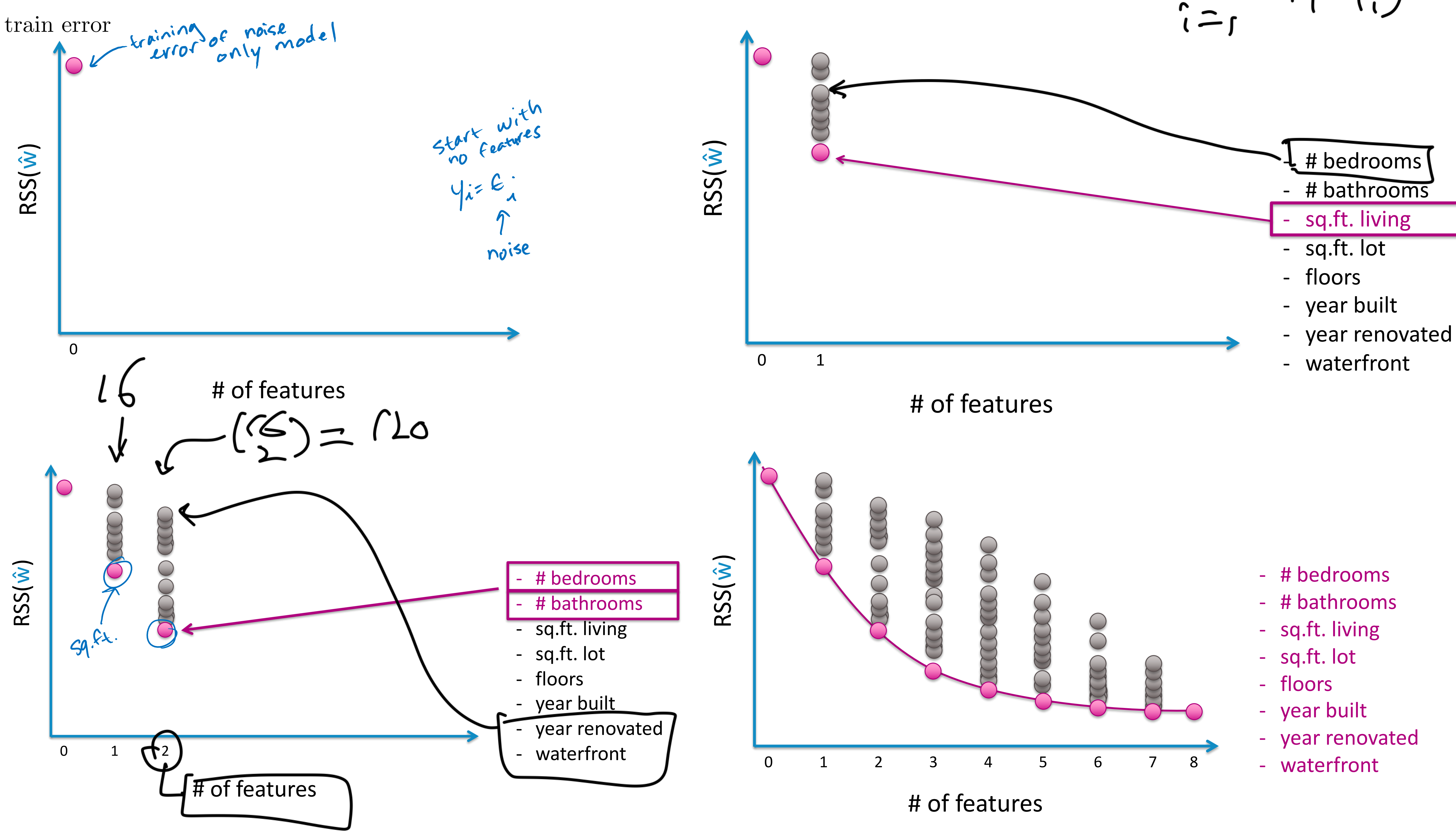
$\lambda = 0$

but only use selected features

- de-biasing is critical!

Slow but optimal model selection

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



- The best single-feature might not be included in best pair-of-features

Greedy algorithm: matching pursuit

- Choose how many features to select, say k

- Repeat for i=1,…,k
  - Choose a single feature, such that minimizes the loss when optimized together with (i-1) features chosen from the previous steps
  - Let f_i denote this feature
  - $S_i \leftarrow S_{i-1} \cup f_i$