Recap & Look ahead

Emily Fox & Carlos Guestrin
Machine Learning Specialization
University of Washington

©2015 Emily Fox & Carlos Guestrin

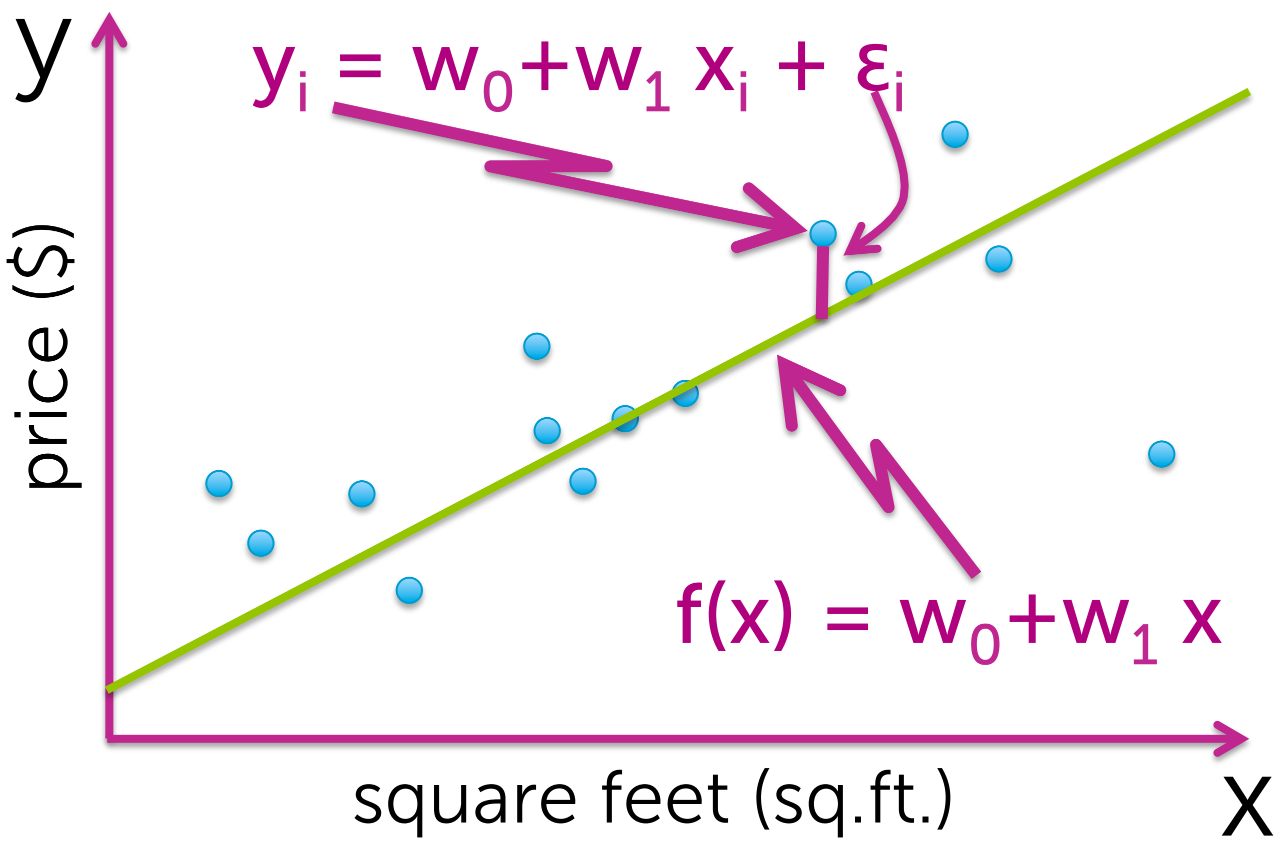
What we've learned

Module 1: Simple Regression

©2015 Emily Fox & Carlos Guestrin

Simple linear regression model

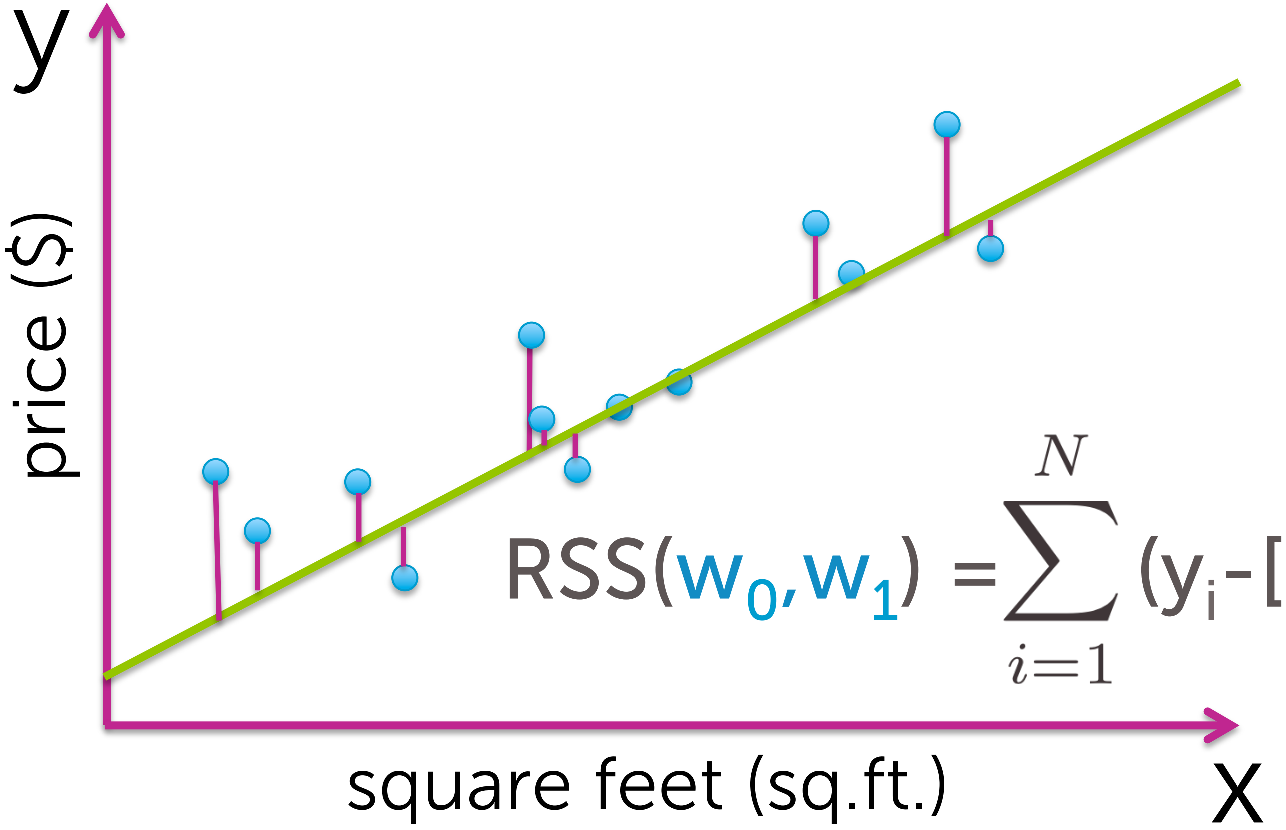
1 input and just fit a line to data



©2015 Emily Fox & Carlos Guestrin

"Cost" of using a given line

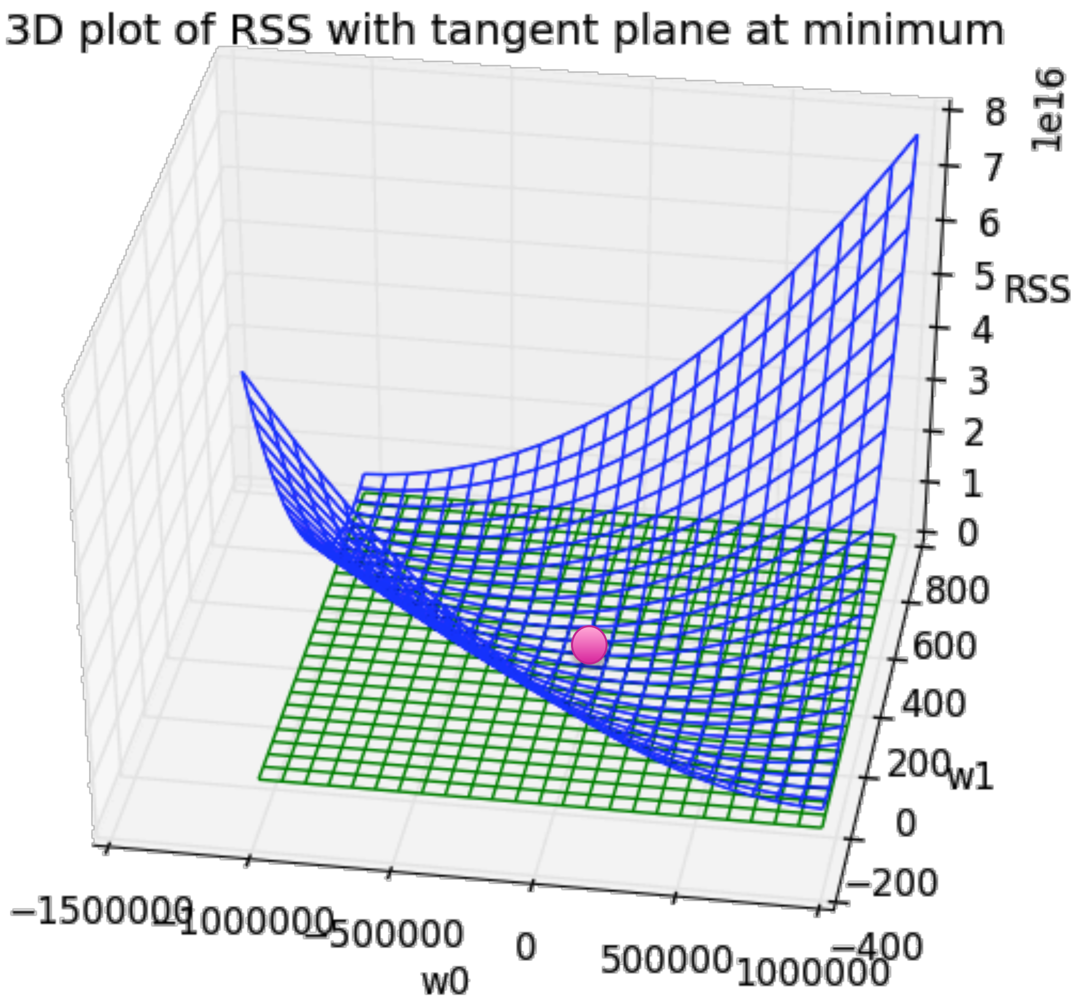
Residual sum of squares (RSS)



$$RSS(w_0,w_1) = \sum_{i=1}^{N}(y_i-[w_0+w_1x_i])^2$$

©2015 Emily Fox & Carlos Guestrin

Minimizing the cost



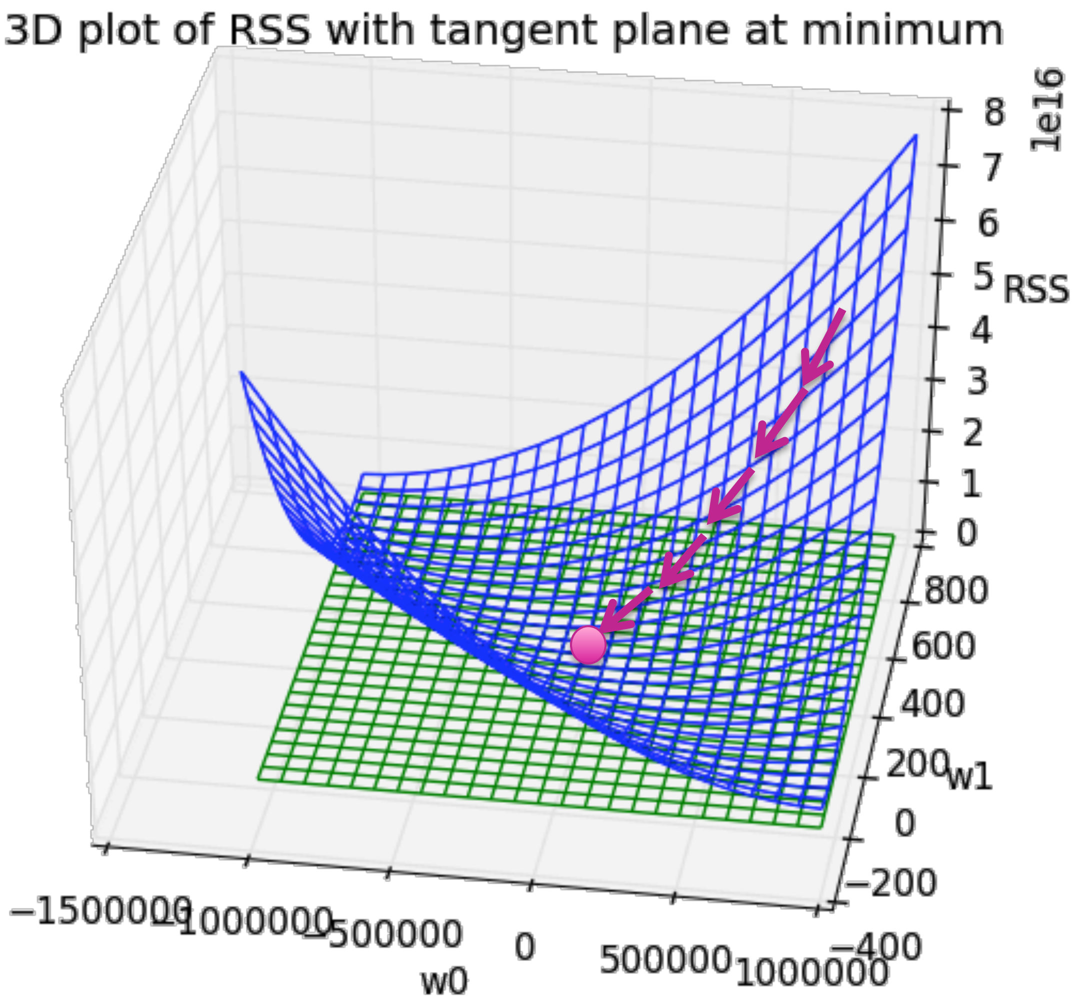
Minimize function over all possible w_0, w_1

$$\min_{w_0, w_1} \sum_{i=1}^{N} (y_i - [w_0 + w_1 x_i])^2$$

RSS(w_0,w_1) is a function of 2 variables

©2015 Emily Fox & Carlos Guestrin

Gradient descent



Algorithm:

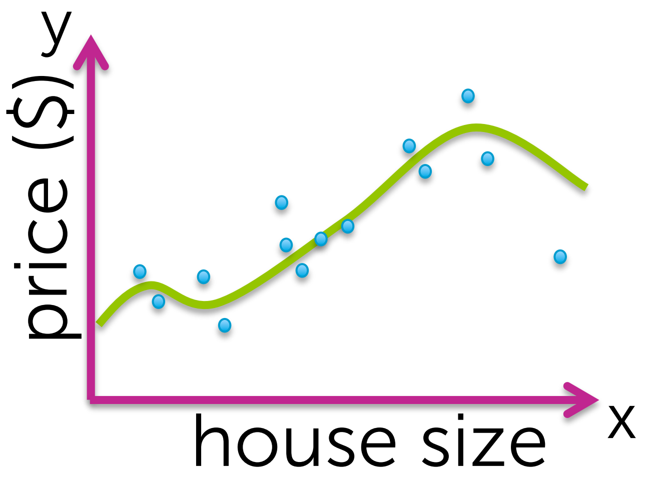
while not converged

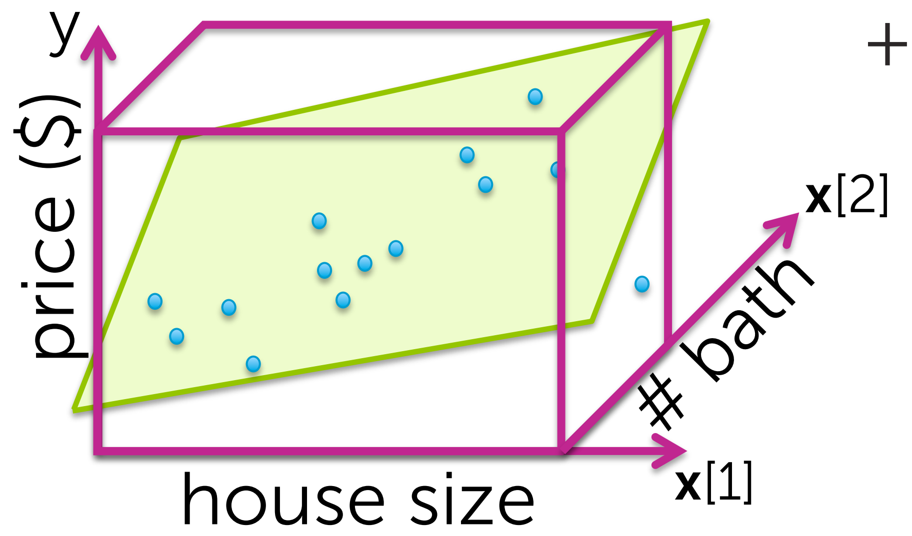
$w^{(t+1)} \leftarrow w^{(t)} - \eta \nabla RSS(w^{(t)})$

©2015 Emily Fox & Carlos Guestrin

Module 2: Multiple Regression

Regression with multiple features





Fit more complex relationships than just a line

Incorporate more inputs + features thereof

- Square feet
- # bathrooms
- # bedrooms
- Lot size
- Year built
- …

©2015 Emily Fox & Carlos Guestrin

Formally…

Model:

$$y_i = w_0\, h_0(\mathbf{x}_i) + w_1\, h_1(\mathbf{x}_i) + \ldots + w_D\, h_D(\mathbf{x}_i) + \varepsilon_i$$

$$= \sum_{j=0}^{D} w_j\, h_j(\mathbf{x}_i) + \varepsilon_i$$

feature 1 = $h_0(\mathbf{x})$ … e.g., 1

feature 2 = $h_1(\mathbf{x})$ … e.g., $\mathbf{x}[1]$ = sq. ft.

feature 3 = $h_2(\mathbf{x})$ … e.g., $\mathbf{x}[2]$ = #bath

or, $\log(\mathbf{x}[7])\ \mathbf{x}[2]$ = log(#bed) x #bath

…

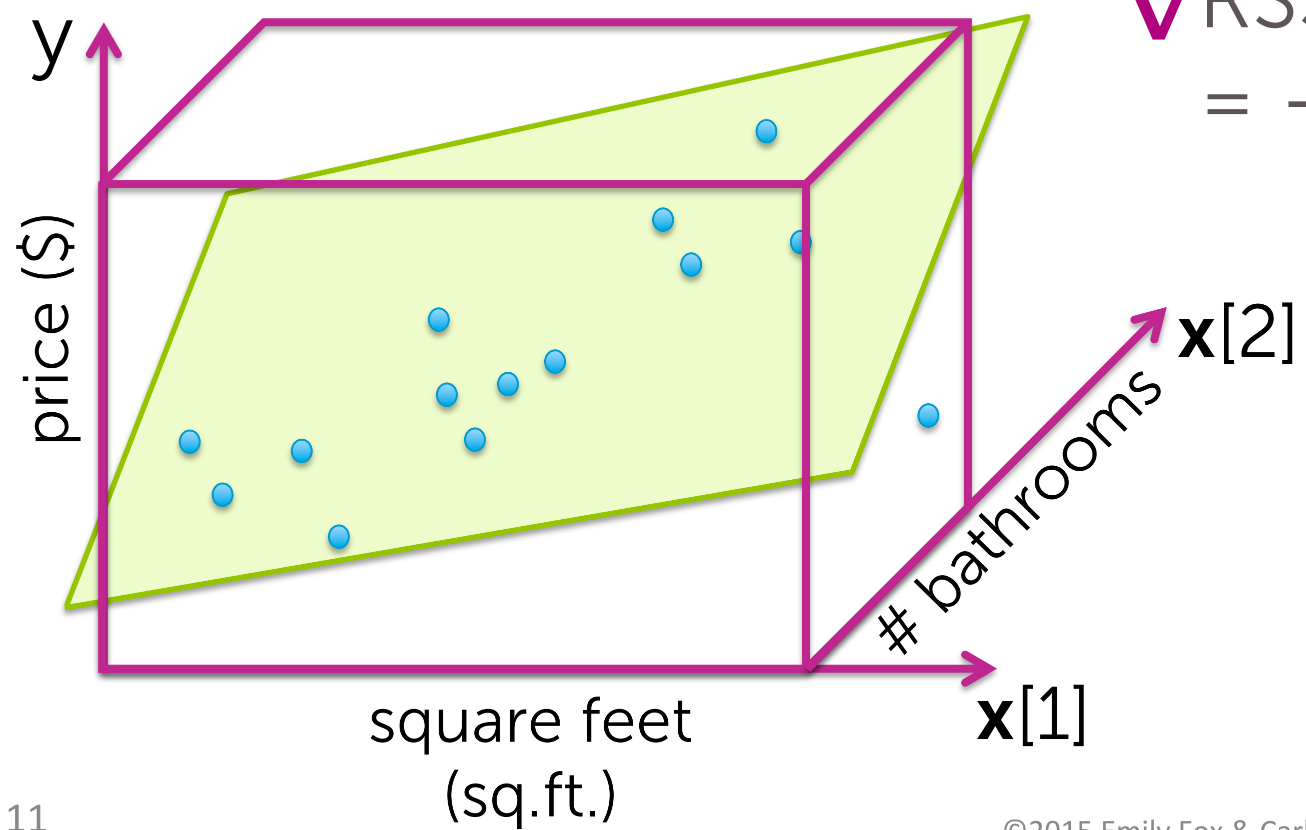
feature D+1 = $h_D(\mathbf{x})$ … some other function of $\mathbf{x}[1]$,…, $\mathbf{x}[d]$

©2015 Emily Fox & Carlos Guestrin

RSS for multiple regression

$$\text{RSS}(\mathbf{w}) = \sum_{i=1}^{N} (y_i - h(x_i)^T\mathbf{w})^2$$

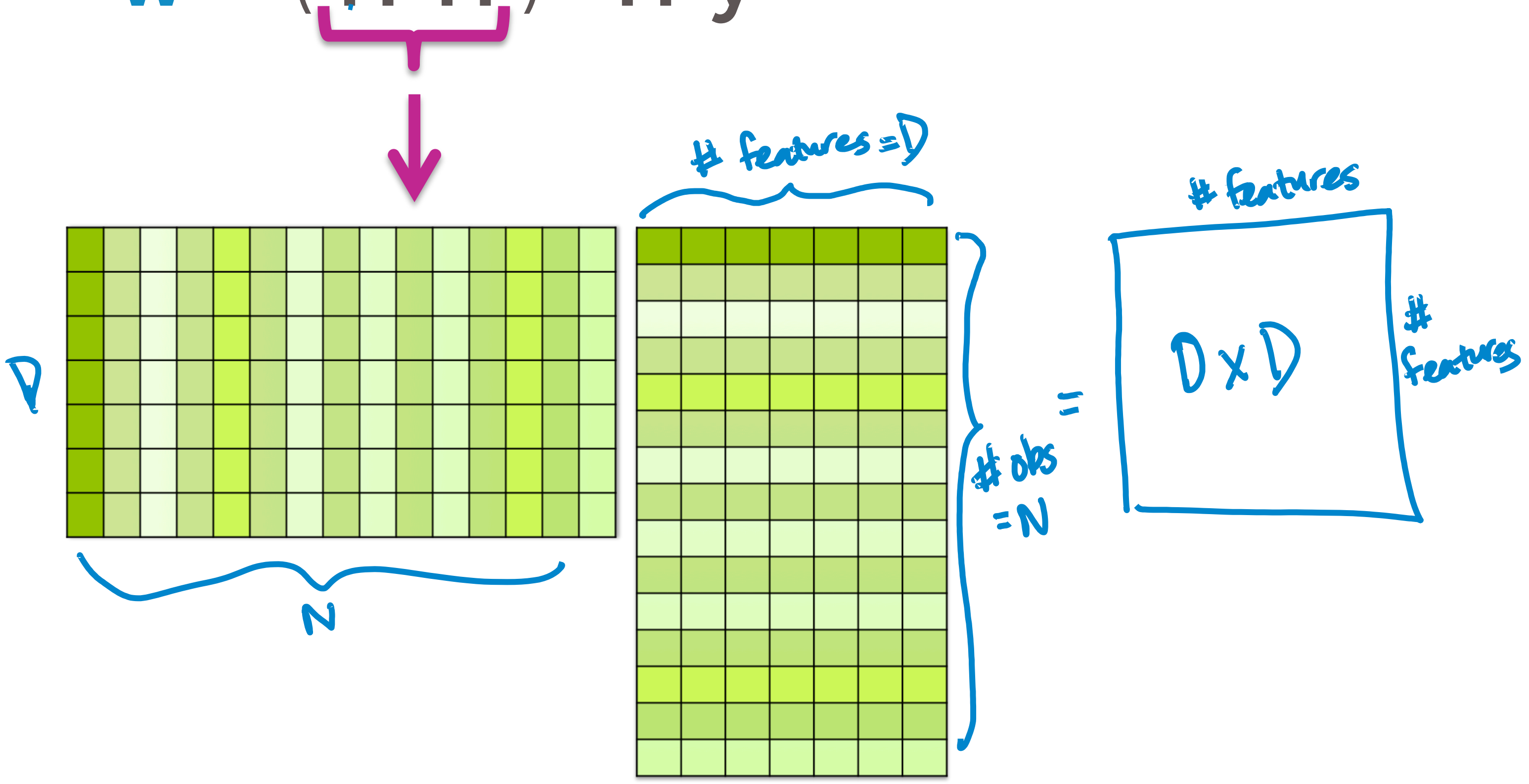
$$\nabla \text{RSS}(\mathbf{w}) = -2\mathbf{H}^T(\mathbf{y}-\mathbf{H}\mathbf{w})$$



©2015 Emily Fox & Carlos Guestrin

Closed-form solution

$$\hat{\mathbf{w}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y}$$



Invertible if:

In most cases is N > D

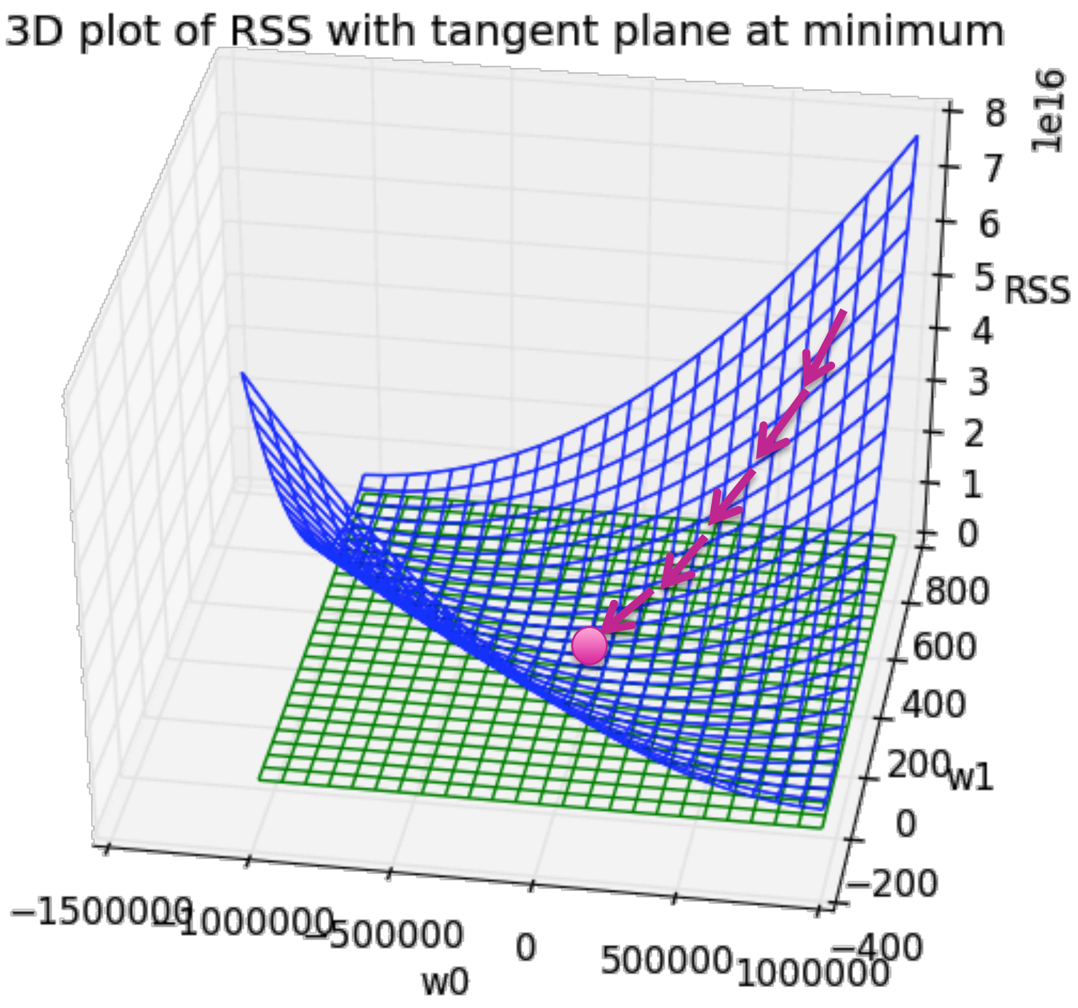
really, # of linearly ind. observations

Complexity of inverse:

$O(D^3)$

©2015 Emily Fox & Carlos Guestrin

Gradient descent for multiple regression



init $\mathbf{w}^{(1)}=0$ (or randomly, or smartly), $t=1$

while $||\nabla \text{RSS}(\mathbf{w}^{(t)})|| > \varepsilon$

for $j=0,\ldots,D$

$$\text{partial}[j] = -2\sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\mathbf{w}^{(t)}))$$

$$\mathbf{w}_j^{(t+1)} \leftarrow \mathbf{w}_j^{(t)} - \eta \, \text{partial}[j]$$

$t \leftarrow t+1$

©2015 Emily Fox & Carlos Guestrin

Module 3: Assessing Performance

Measuring loss

Loss function:

$L(y, f_{\hat{w}}(\mathbf{x}))$

actual value

$\hat{f}(\mathbf{x})$ = predicted value $\hat{y}$

Cost of using $\hat{w}$ at x when y is true

Examples:

(assuming loss for underpredicting = overpredicting)

Absolute error: $L(y, f_{\hat{w}}(\mathbf{x})) = |y - f_{\hat{w}}(\mathbf{x})|$

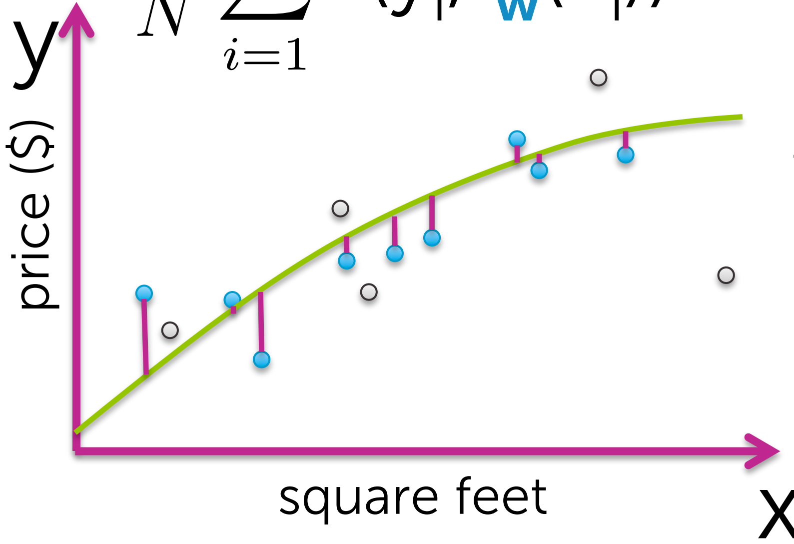
Squared error: $L(y, f_{\hat{w}}(\mathbf{x})) = (y - f_{\hat{w}}(\mathbf{x}))^2$

©2015 Emily Fox & Carlos Guestrin

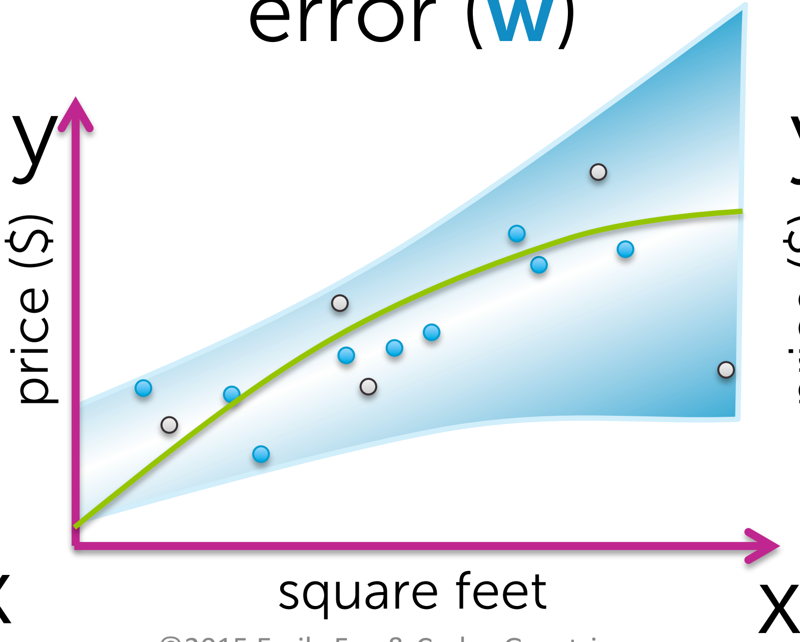
3 Measures of error

Training error ($\hat{\mathbf{w}}$) =

$$\frac{1}{N}\sum_{i=1}^{N} L(y_i, f_{\hat{\mathbf{w}}}(\mathbf{x}_i))$$

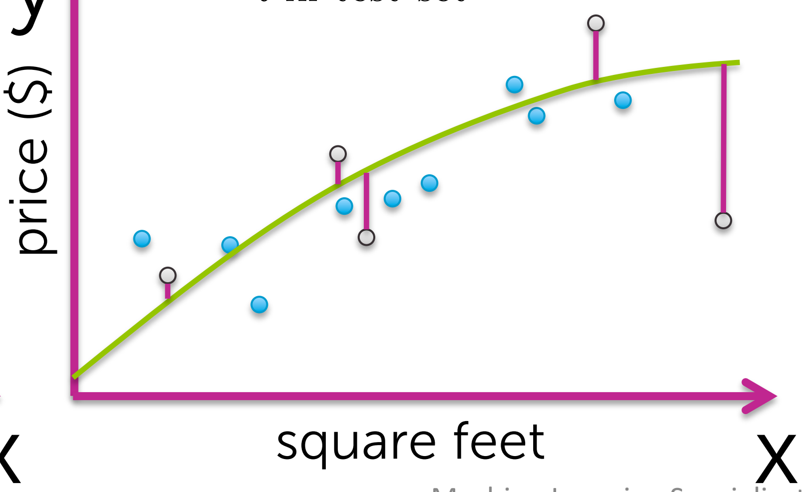


Generalization error ($\hat{\mathbf{w}}$)



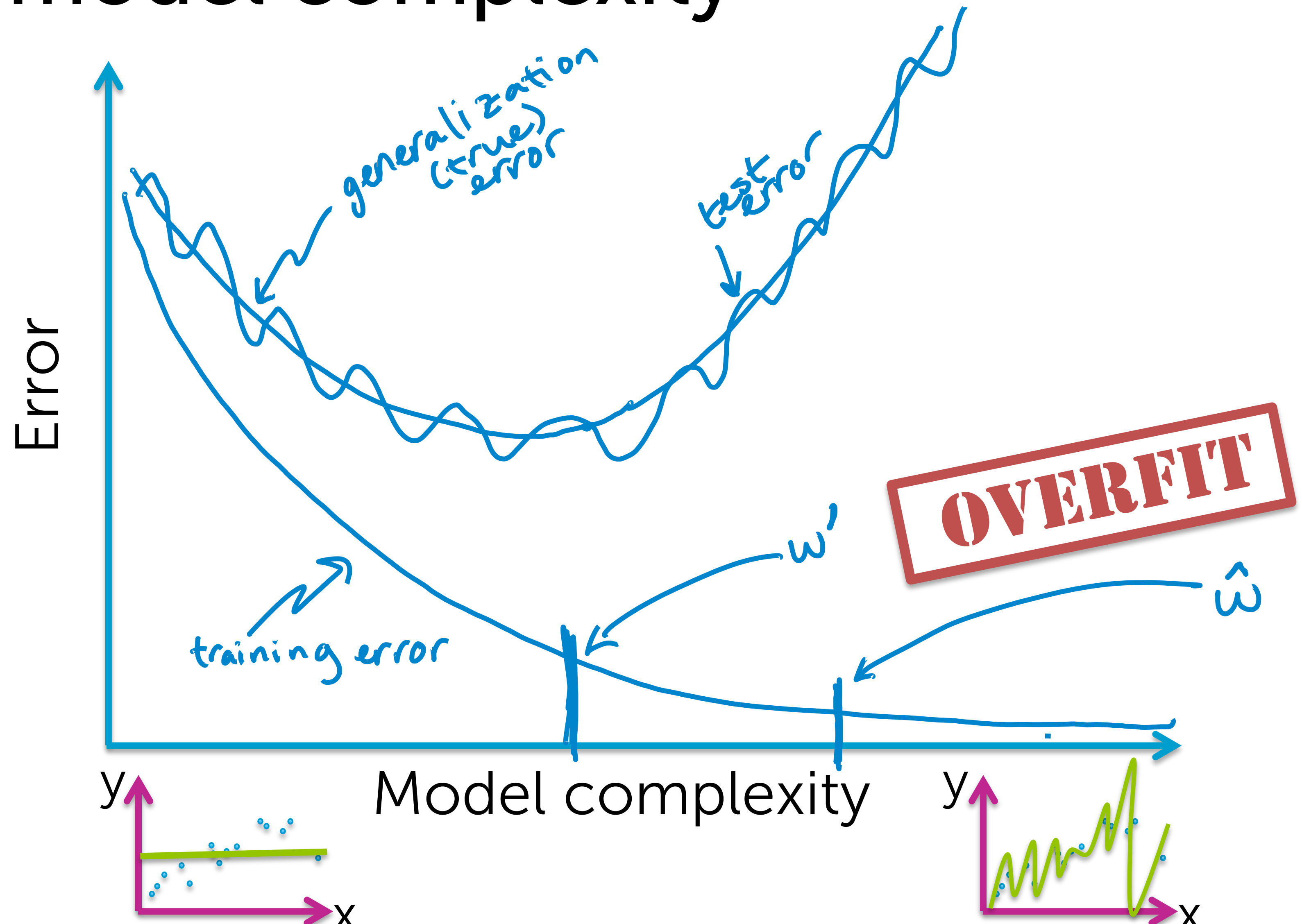
Test error ($\hat{\mathbf{w}}$) =

$$\frac{1}{N_{test}}\sum_{i \text{ in test set}} L(y_i, f_{\hat{\mathbf{w}}}(\mathbf{x}_i))$$



©2015 Emily Fox & Carlos Guestrin

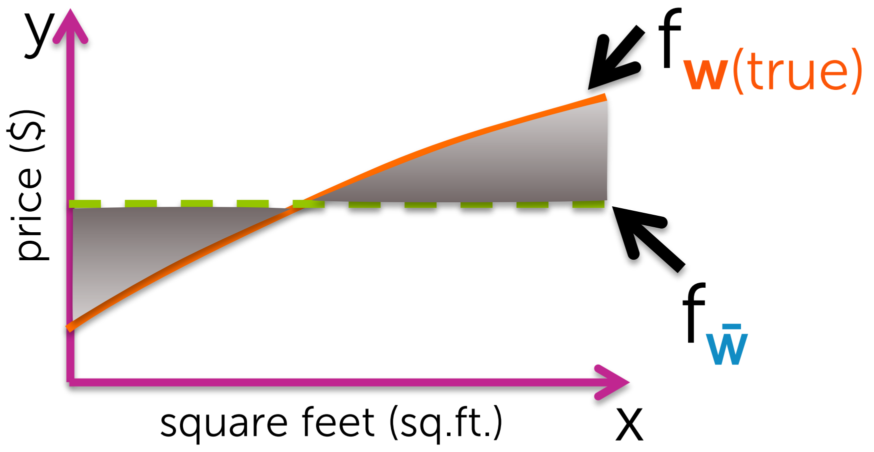
Training, true, & test error vs. model complexity

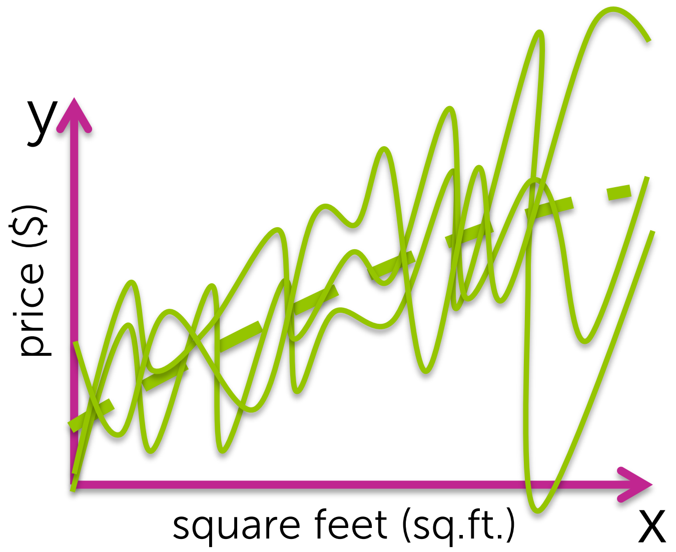


©2015 Emily Fox & Carlos Guestrin

3 Sources of prediction error

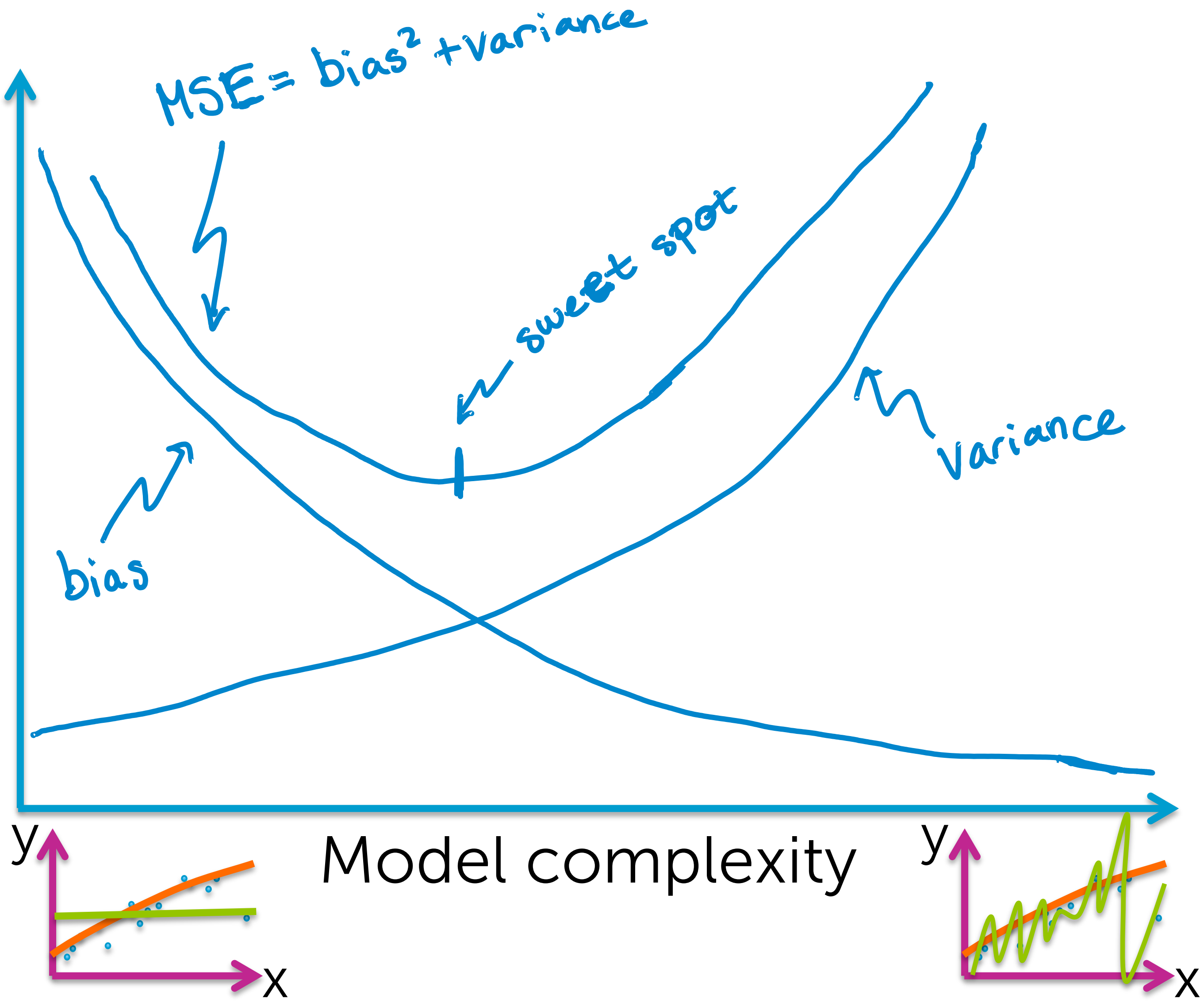
1. Noise
2. Bias
3. Variance





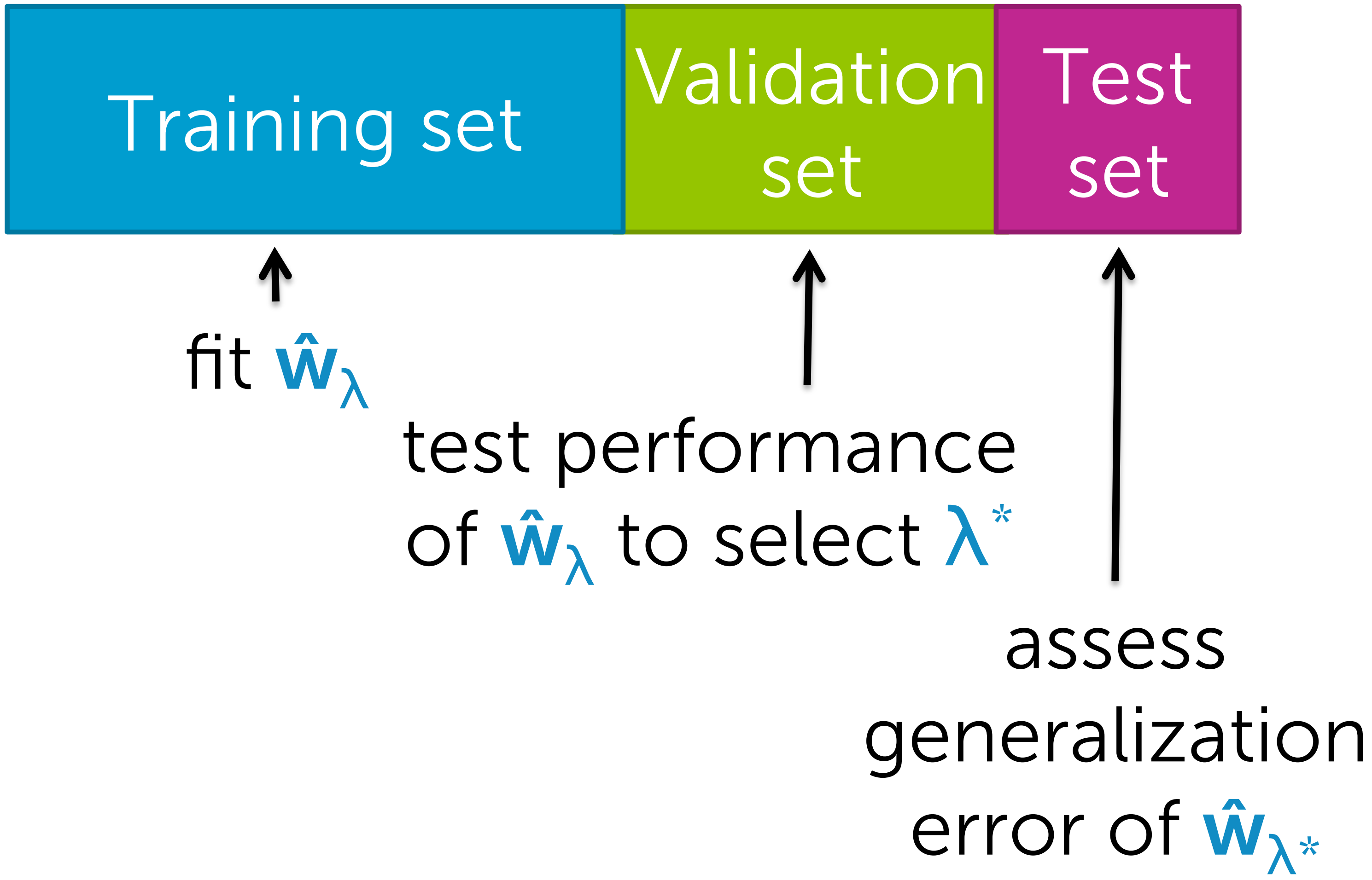
©2015 Emily Fox & Carlos Guestrin

Bias-variance tradeoff



©2015 Emily Fox & Carlos Guestrin

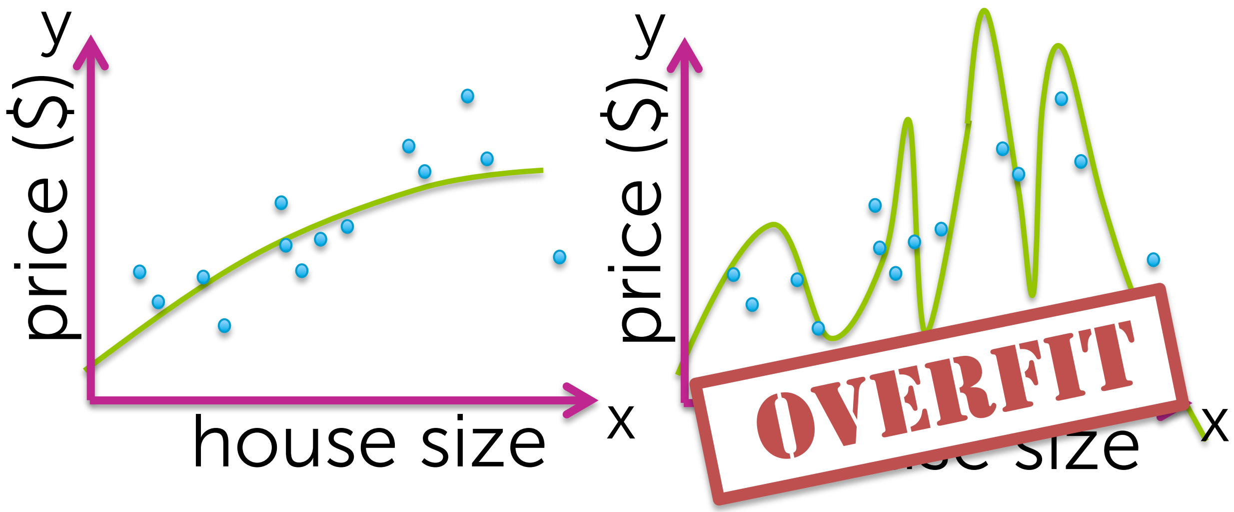
Model selection & assessment



©2015 Emily Fox & Carlos Guestrin

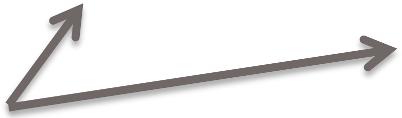
Module 4: Ridge Regression

Balancing fit and model complexity



Ridge total cost =

measure of fit + measure of magnitude of coefficients



bias-variance tradeoff

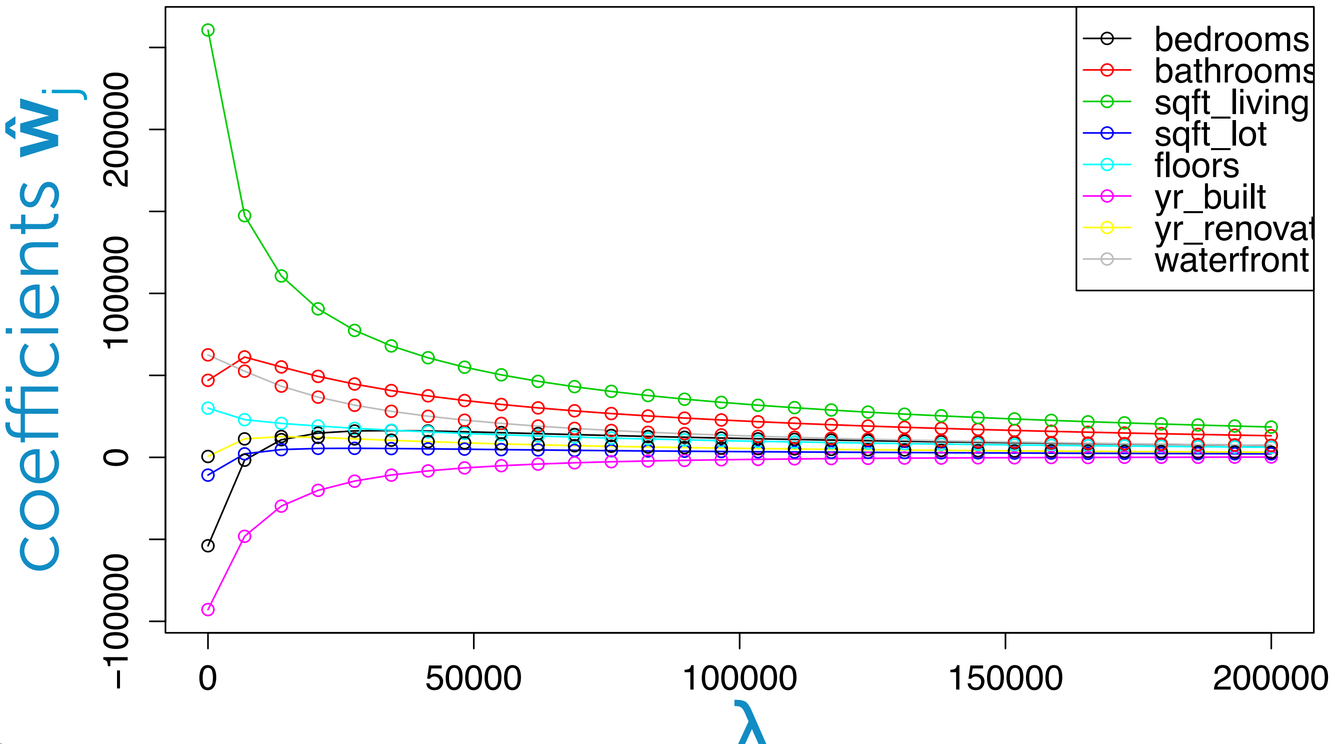
©2015 Emily Fox & Carlos Guestrin

Ridge objective function (L_2 regularized regression)

$\hat{\mathbf{w}}$ selected to minimize

tuning parameter = balance of fit and magnitude

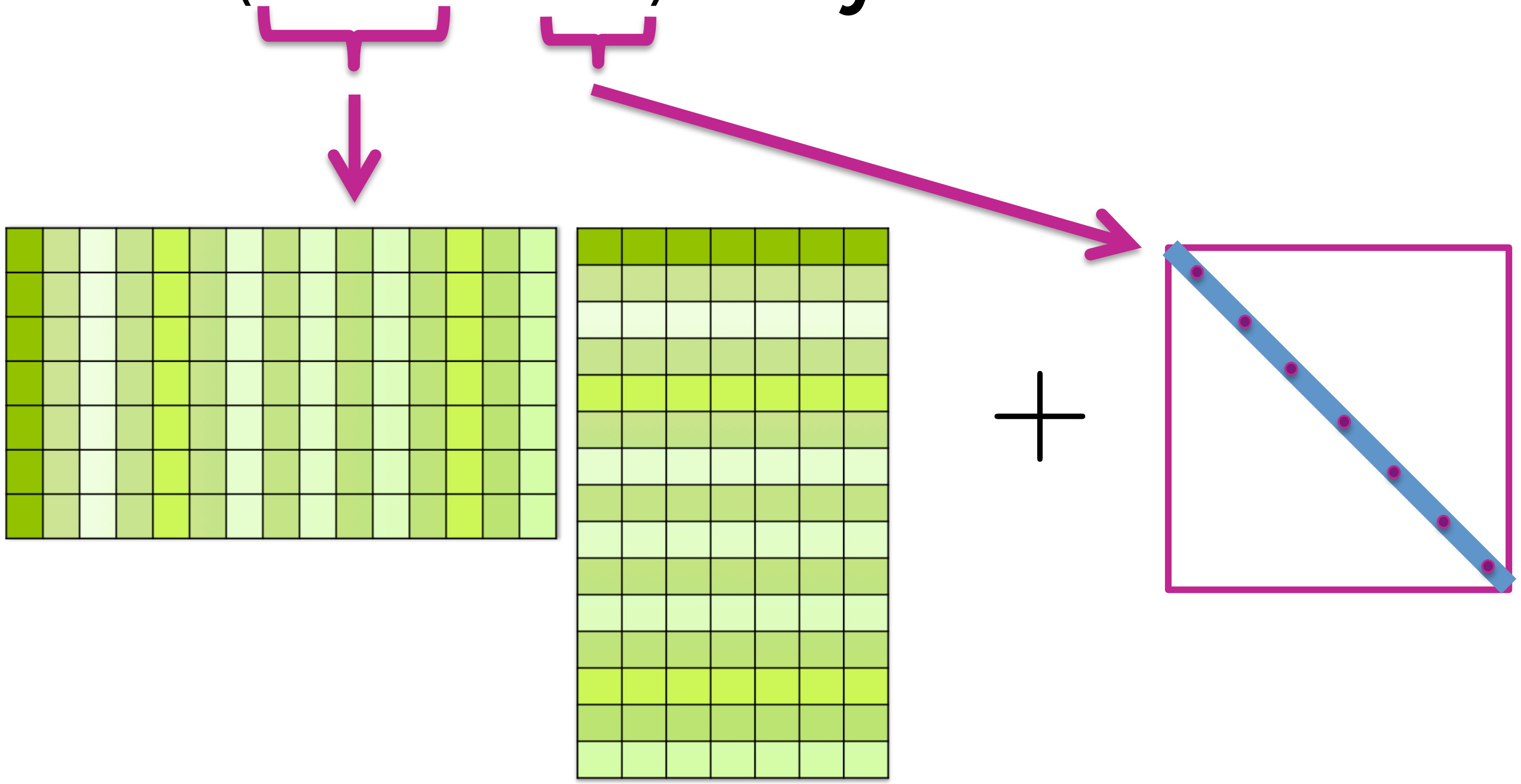
$$\text{RSS}(\mathbf{w}) + \lambda\|\mathbf{w}\|_2^2$$



©2015 Emily Fox & Carlos Guestrin

Ridge closed-form solution

$$\hat{\mathbf{w}} = (\mathbf{H}^T\mathbf{H} + \lambda\mathbf{I})^{-1}\mathbf{H}^T\mathbf{y}$$



Invertible if:

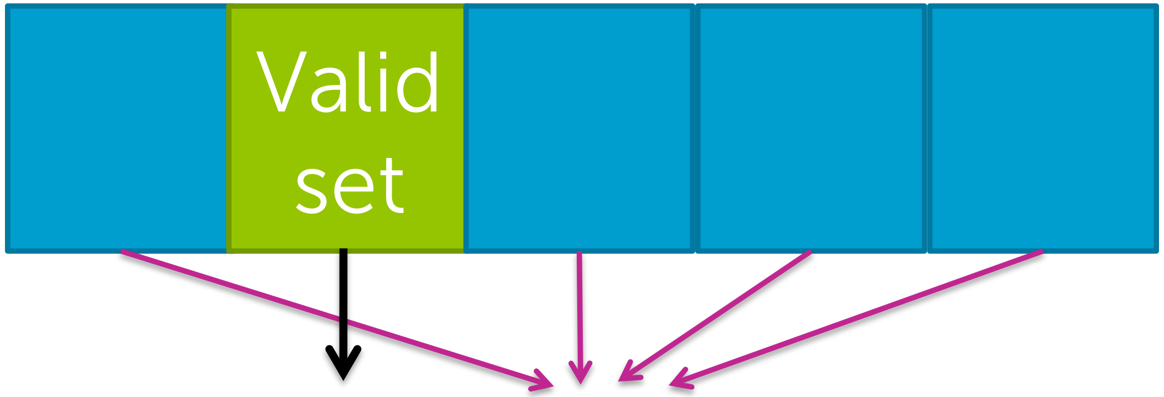
Always if $\lambda>0$, even if $N < D$

Complexity of inverse:

$O(D^3)$… big for large D!

©2015 Emily Fox & Carlos Guestrin

K-fold cross validation



$\text{error}_2(\lambda)$ $\hat{\mathbf{w}}_\lambda^{(2)}$

For k=1,…,K

1. Estimate $\hat{\mathbf{w}}_\lambda^{(k)}$ on the training blocks
2. Compute error on validation block: $\text{error}_k(\lambda)$

Compute average error: $CV(\lambda) = \frac{1}{K}\sum_{k=1}^{K} \text{error}_k(\lambda)$

©2015 Emily Fox & Carlos Guestrin

Module 5: Lasso Regression

Performing feature selection

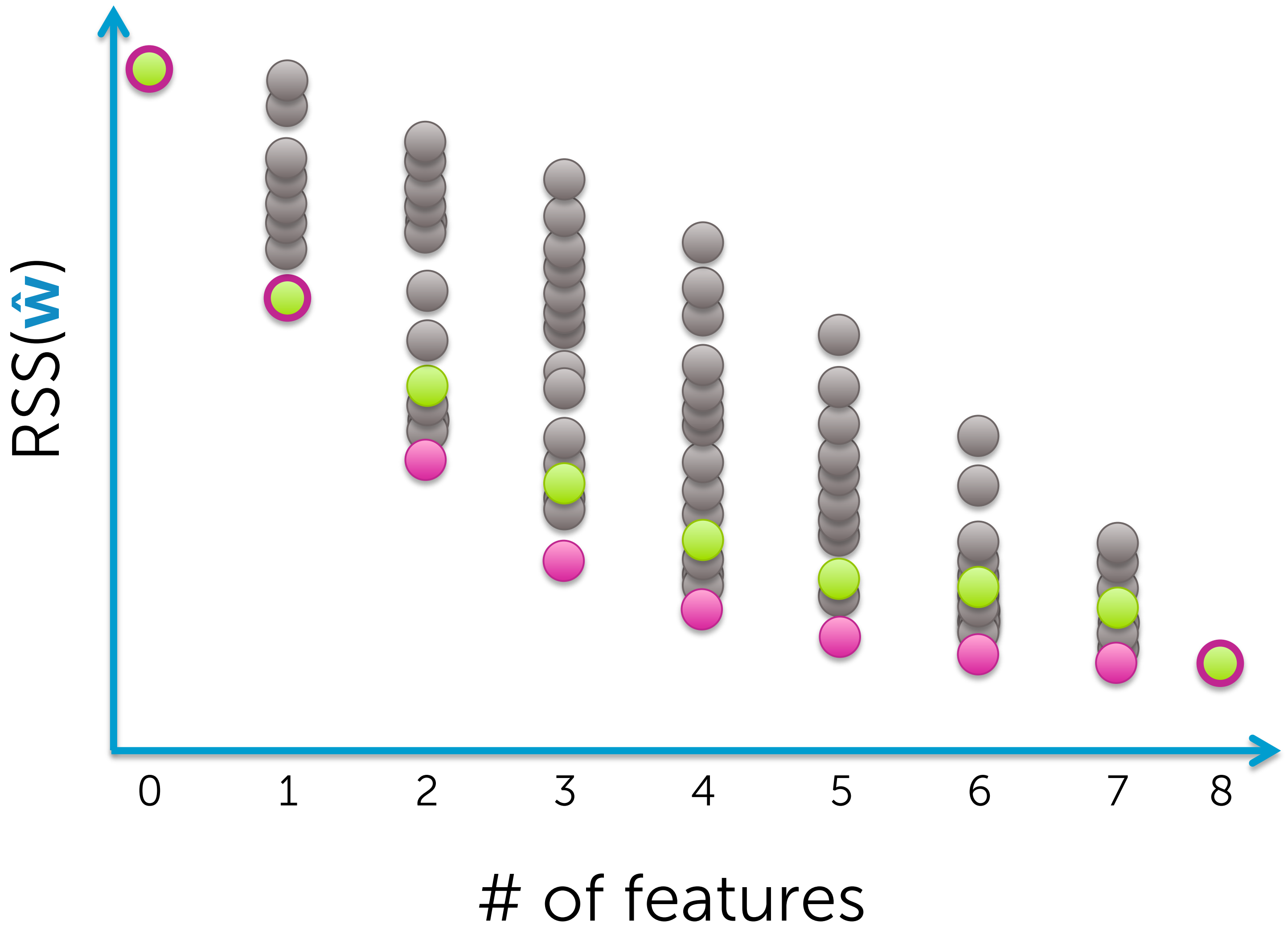
Useful for efficiency of predictions and interpretability

Lot size
Single Family
Year built
Last sold price
Last sale price/sqft
Finished sqft
Unfinished sqft
Finished basement sqft
floors
Flooring types
Parking type
Parking amount
Cooling
Heating
Exterior materials
Roof type
Structure style

Dishwasher
Garbage disposal
Microwave
Range / Oven
Refrigerator
Washer
Dryer
Laundry location
Heating type
Jetted Tub
Deck
Fenced Yard
Lawn
Garden
Sprinkler System
⋮

©2015 Emily Fox & Carlos Guestrin

All subsets vs. greedy



- # bedrooms
- # bathrooms
- sq.ft. living
- sq.ft. lot
- floors
- year built
- year renovated
- waterfront

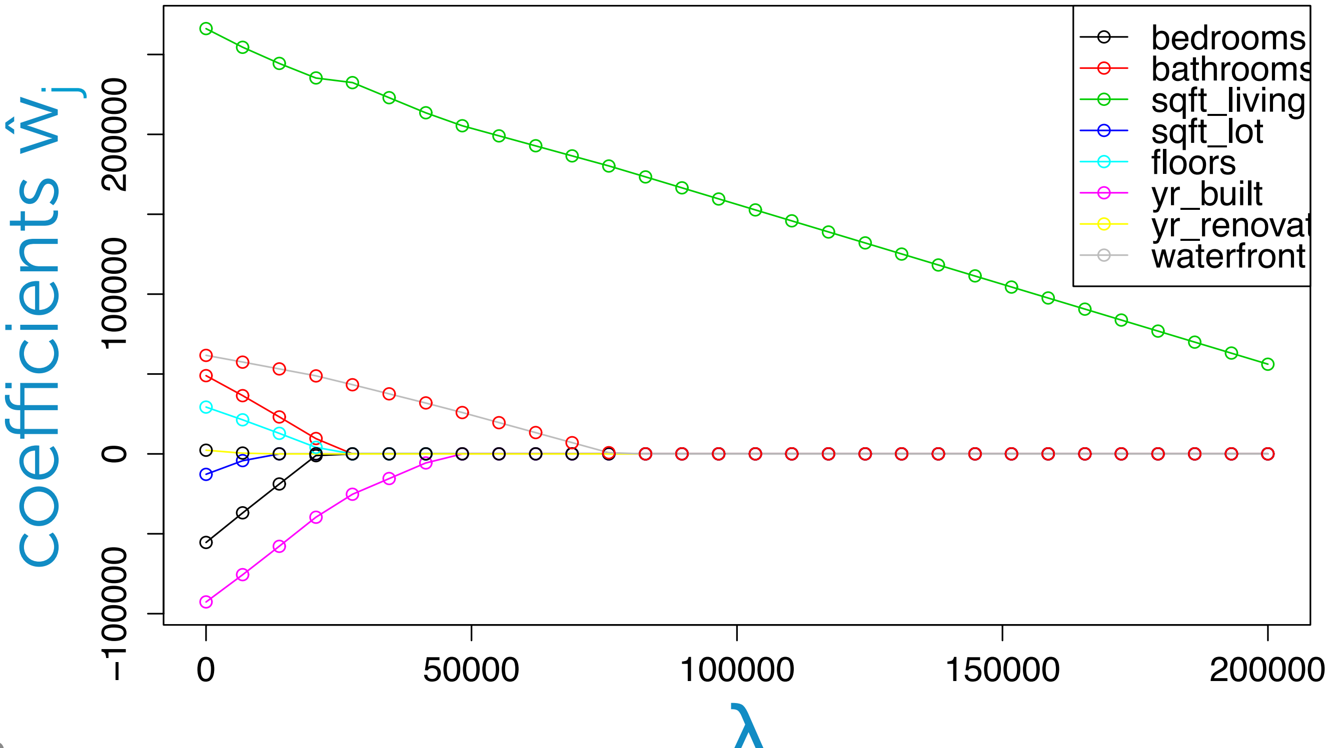
©2015 Emily Fox & Carlos Guestrin

Lasso objective function (L_1 regularized regression)

$\hat{\mathbf{w}}$ selected to minimize

tuning parameter = balance of fit and **sparsity**

$$RSS(\mathbf{w}) + \lambda ||\mathbf{w}||_1$$



©2015 Emily Fox & Carlos Guestrin

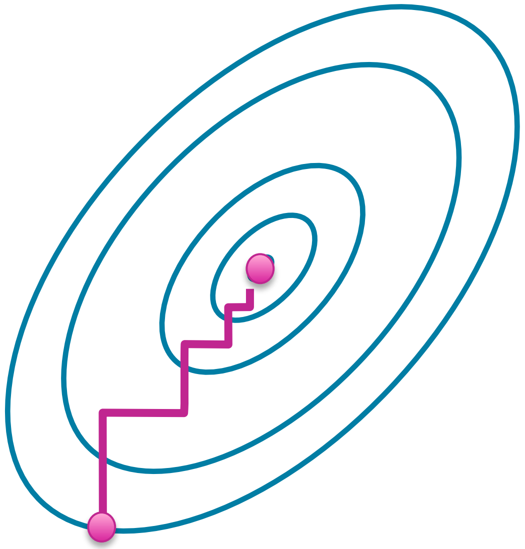
Coordinate descent for lasso

Precompute: $z_j = \sum_{i=1}^{N} h_j(\mathbf{x}_i)^2$

Initialize $\hat{\mathbf{w}}$ = 0 (or smartly…)

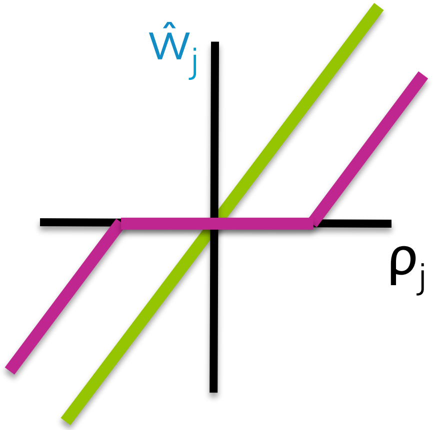
while not converged

for j=0,1,…,D



compute: $\rho_j = \sum_{i=1}^{N} h_j(\mathbf{x}_i)(y_i - \hat{y}_i(\hat{\mathbf{w}}_{-j}))$

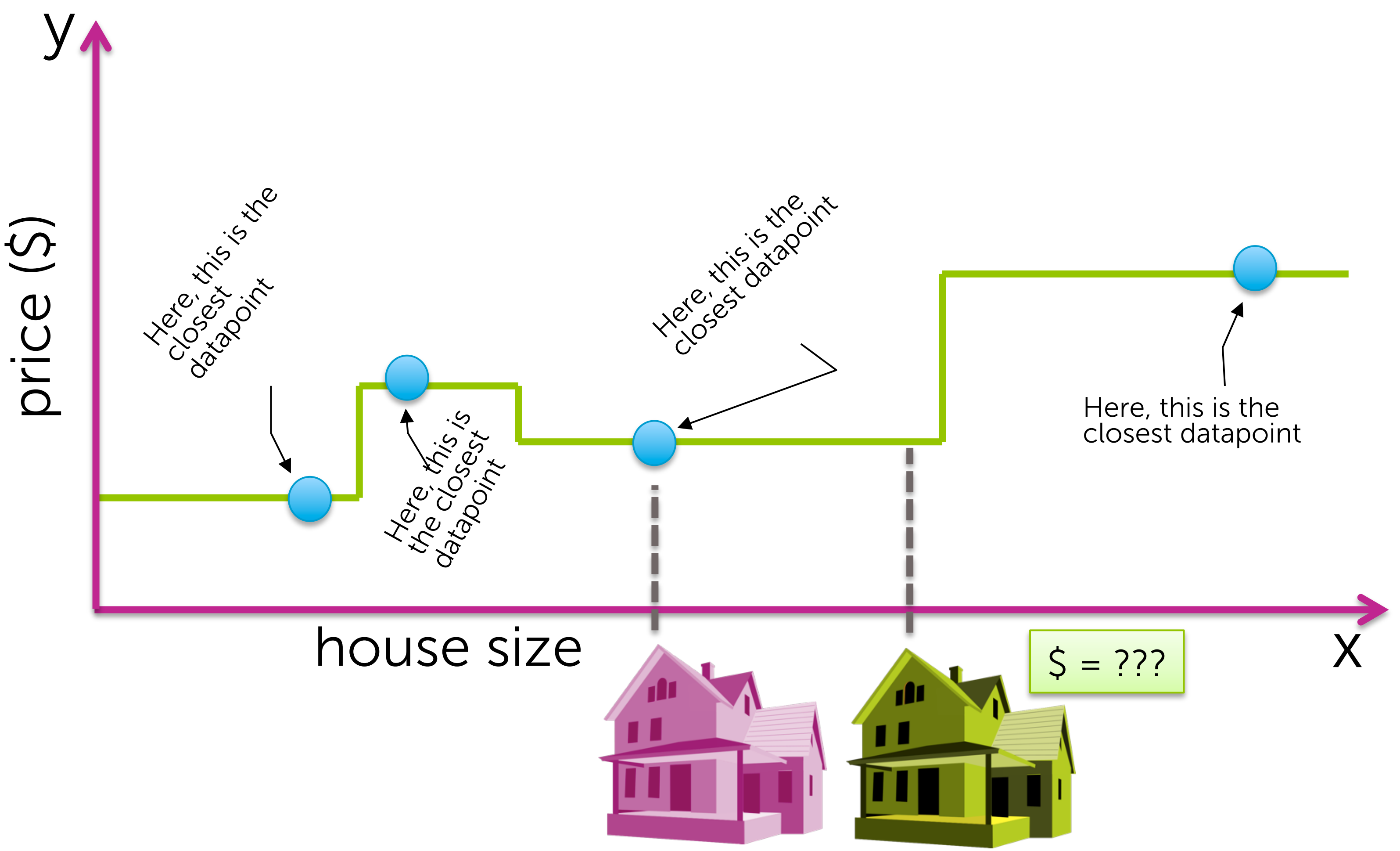
set: $\hat{w}_j = \begin{cases} (\rho_j + \lambda/2)/z_j & \text{if } \rho_j < -\lambda/2 \\ 0 & \text{if } \rho_j \text{ in } [-\lambda/2, \lambda/2] \\ (\rho_j - \lambda/2)/z_j & \text{if } \rho_j > \lambda/2 \end{cases}$



©2015 Emily Fox & Carlos Guestrin

Module 6: Nearest Neighbor & Kernel Regression

1-Nearest neighbor regression



©2015 Emily Fox & Carlos Guestrin

Weighted k-NN

Weigh more similar houses more than those less similar in list of k-NN

Predict:

weights on NN

$$\hat{y}_q = \frac{c_{qNN1}y_{NN1} + c_{qNN2}y_{NN2} + c_{qNN3}y_{NN3} + \ldots + c_{qNNk}y_{NNk}}{\sum_{j=1}^{k} c_{qNNj}}$$

©2015 Emily Fox & Carlos Guestrin

Kernel regression

Instead of just weighting NN, weight *all* points

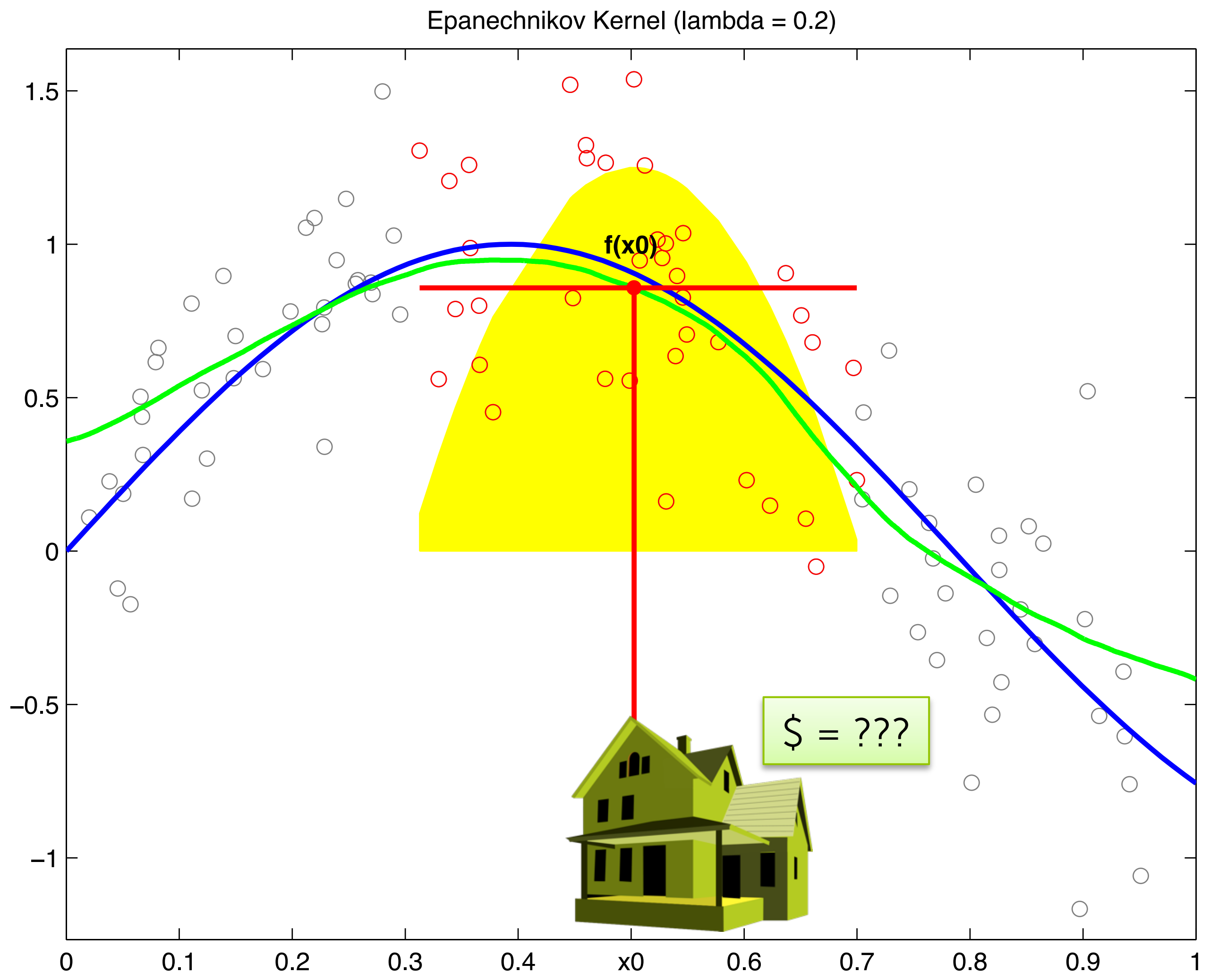
Predict:

weight on each datapoint

$$\hat{y}_q = \frac{\sum_{i=1}^{N} c_{qi} y_i}{\sum_{i=1}^{N} c_{qi}} = \frac{\sum_{i=1}^{N} \text{Kernel}_\lambda(\text{distance}(\mathbf{x}_i,\mathbf{x}_q)) * y_i}{\sum_{i=1}^{N} \text{Kernel}_\lambda(\text{distance}(\mathbf{x}_i,\mathbf{x}_q))}$$

©2015 Emily Fox & Carlos Guestrin

Visualizing kernel regression



©2015 Emily Fox & Carlos Guestrin

Summary of what we learned

Models

- Linear regression
- Regularization: Ridge (L2), Lasso (L1)
- Nearest neighbor and kernel regression

Algorithms

- Gradient descent
- Coordinate descent

Concepts

- Loss functions, bias-variance tradeoff, cross-validation, sparsity, overfitting, model selection, feature selection

©2015 Emily Fox & Carlos Guestrin

What we didn't cover

Other important regression topics

- Multivariate outputs **y**
  …when correlated
- Maximum likelihood estimation
  – Equivalent to least squares when errors are "normal"/Gaussian
- Statistical inferences
- "Generalized linear models"
  – Models for non-Gaussian error
  – E.g., outputs are
    (i) constrained to be positive or bounded
    (ii) discrete ("yes"/"no")
- Regression trees
- …

©2015 Emily Fox & Carlos Guestrin

What's ahead in this specialization

3. Classification

Case study: Analyzing sentiment

Models

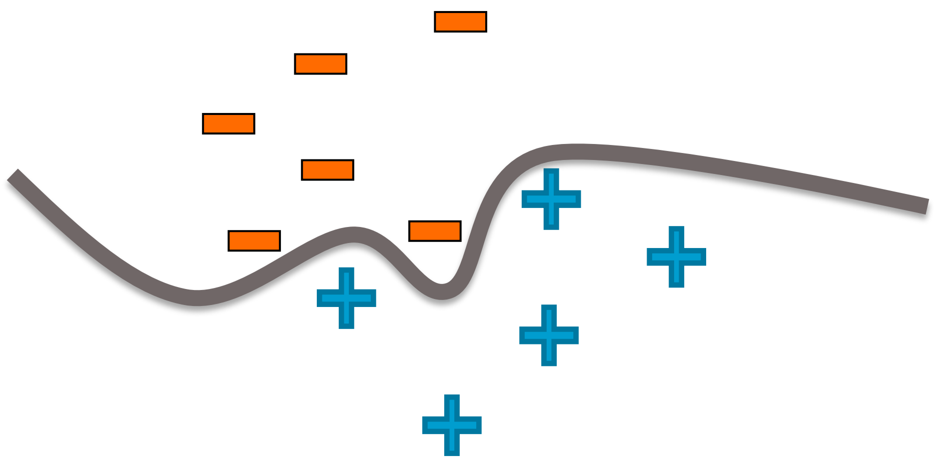
- Linear classifiers
- Decision trees
- Boosted trees and random forests

Algorithms

- Stochastic gradient descent
- Boosting

Concepts

- Decision boundaries, MLE, ensemble methods, online learning



©2015 Emily Fox & Carlos Guestrin

4. Clustering & Retrieval

Case study: Finding documents

Models
- Nearest neighbors
- Clustering, mixtures of Gaussians
- Latent Dirichlet allocation (LDA)

Algorithms
- KD-trees
- K-means
- Expectation-maximization (EM)

Concepts
- Distance metrics, approximation algorithms, sampling algorithms, scaling up with map-reduce

SPORTS

WORLD NEWS

ENTERTAINMENT

SCIENCE

©2015 Emily Fox & Carlos Guestrin

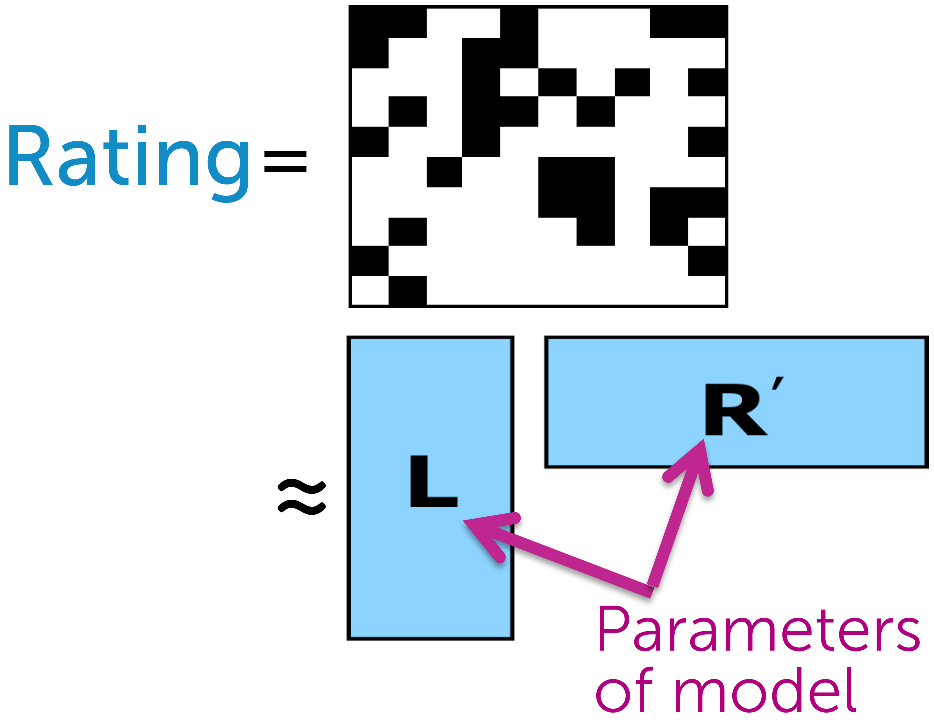
5. Recommender Systems & Dimensionality Reduction

Case study: Recommending Products

Models
- Collaborative filtering
- Matrix factorization
- PCA

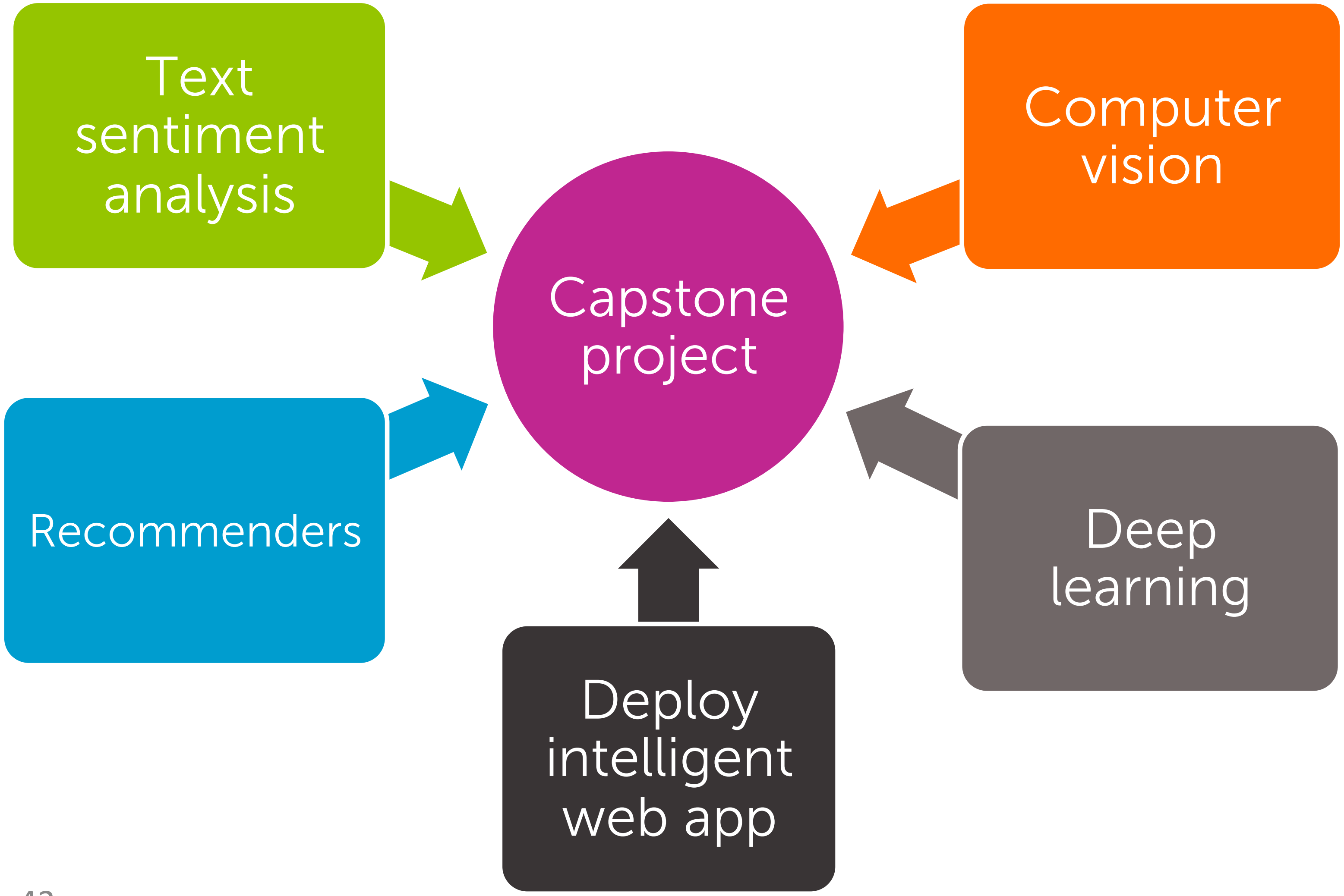
Algorithms
- Coordinate descent
- Eigen decomposition
- SVD

Concepts
- Matrix completion, eigenvalues, cold-start problem, diversity, scaling up



©2015 Emily Fox & Carlos Guestrin

6. Capstone: *Build and deploy an intelligent application with deep learning*



©2015 Emily Fox & Carlos Guestrin

Thank you...