LINEAR REGRESSION (PART III)

CS 334: Machine Learning

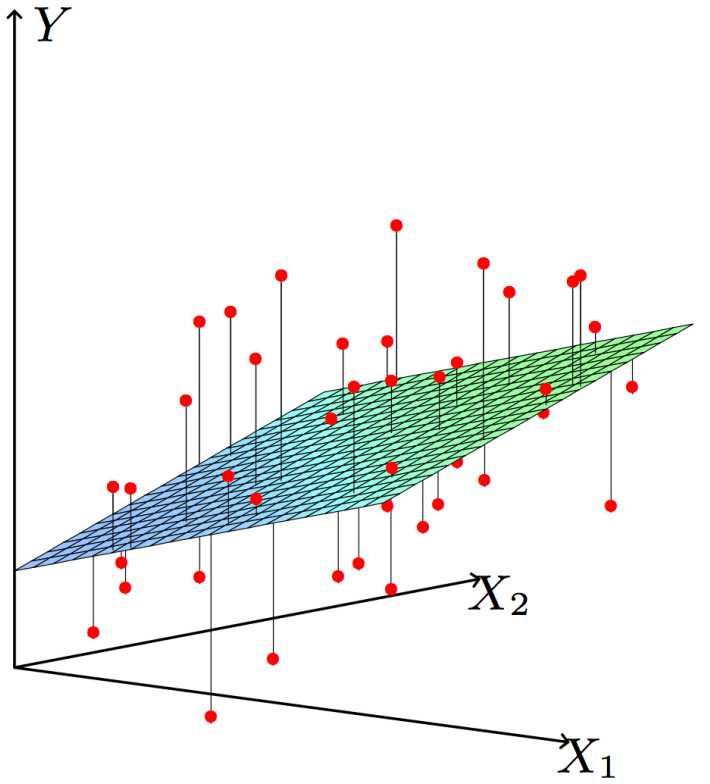
Slides adapted from Joyce Ho, Lee Cooper, Joydeep Ghosh, Carlos Carvalho, and Ryan Tibshirani

REVIEW: REGRESSION: LEAST SQUARES

- Find parameters that minimizes some cost function
- Residual: difference between actual Y and predicted Y
- Least squares: minimize residual sum of squares (RSS or SSR)

$$RSS(\boldsymbol{\beta}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i)^2)$$

$$= \frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$



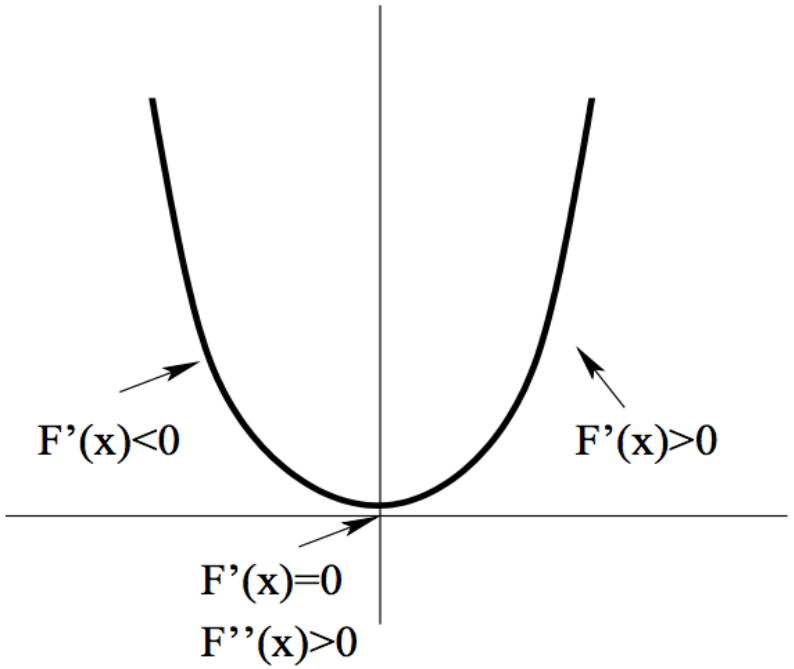
$$f(\mathbf{x}) = \beta_0 + \sum_{i=1}^{p} x_i\beta_i$$

How to find the solution?

Figure 3.1 (Hastie et al.)

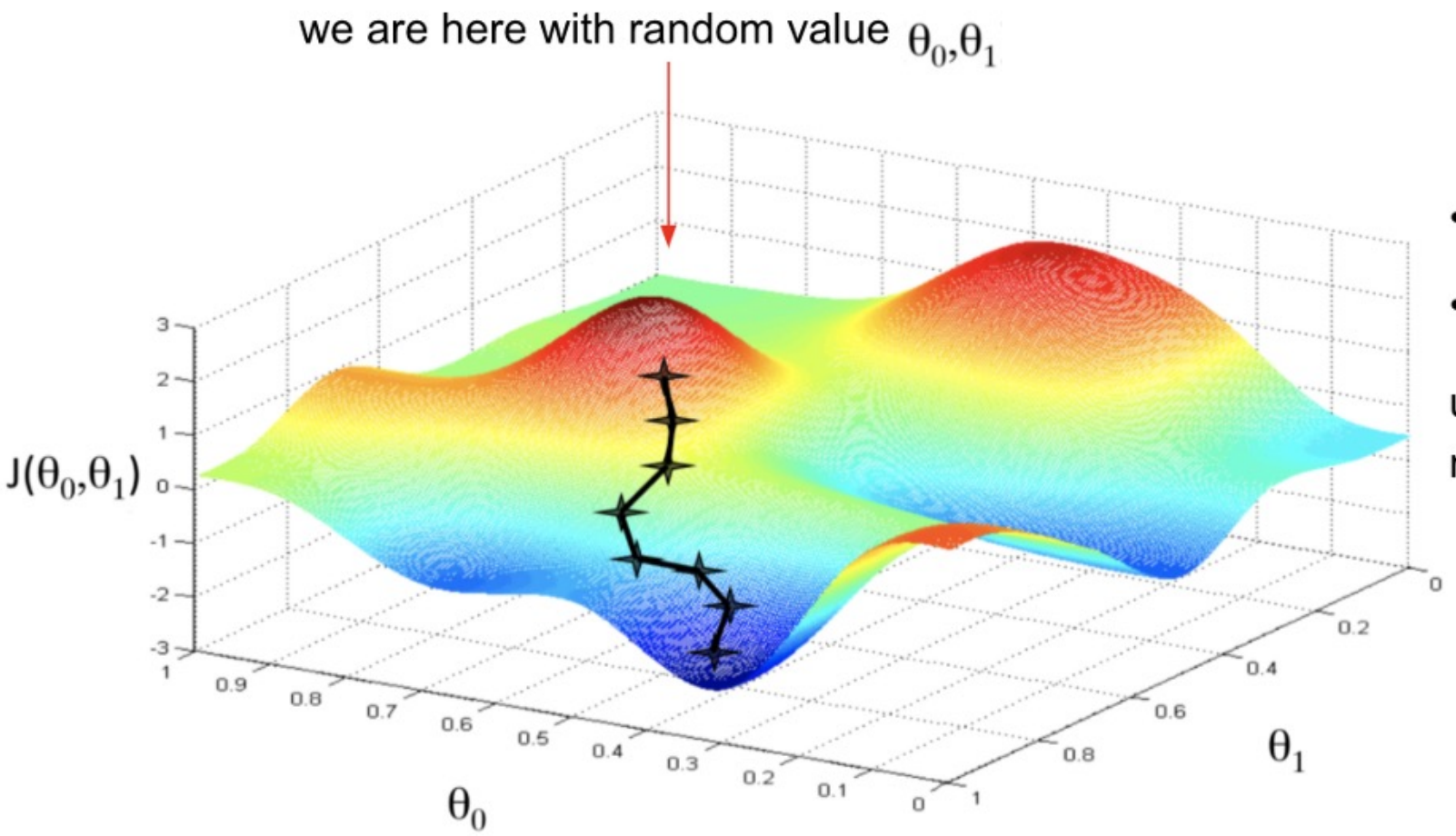
LEARNING THE PARAMETERS

- Closed form (direct solution):
  - set partial derivatives to zero and solve parameters, check that Hessian is greater than 0
- Iterative algorithms:
  - Gradient descent (GD)
  - Stochastic gradient descent (SGD)



Partial derivatives: https://www.khanacademy.org/math/multivariable-calculus

REVIEW: GRADIENT DESCENT (GD)



- Start with some θ_0,θ_1
- Keep changing θ_0,θ_1 to reduce $J(\theta_0,\theta_1)$

until we hopefully end up at a minimum

REVIEW: GD: ALGORITHM

Stopping criteria

Algorithm 1: Gradient Descent

while *Not Converged* **do**

$x^{(k+1)} = x^{(k)} - \eta^{(k)} \nabla f(x)$

end

return $x^{(k+1)}$

$$\nabla f(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x_1} \\ \frac{\partial f}{\partial x_2} \\ \cdots \\ \frac{\partial f}{\partial x_n} \end{bmatrix}$$

Step size
(learning rate)

GD: LINEAR REGRESSION

- What is the gradient for linear regression?

Algorithm 1: Gradient Descent

while *Not Converged* **do**

$$x^{(k+1)} = x^{(k)} - \eta^{(k)} \nabla f(x)$$

end

return $x^{(k+1)}$

REVIEW: LR MATRIX FORM

- Goal: find coefficient vector β : that minimizes MSE

$$
\begin{aligned}
MSE(\beta) &= \frac{1}{n}\mathbf{e}^T\mathbf{e} \\
&= \frac{1}{n}(\mathbf{y}-\mathbf{x}\beta)^T(\mathbf{y}-\mathbf{x}\beta) \\
&= \frac{1}{n}(\mathbf{y}^T-\beta^T\mathbf{x}^T)(\mathbf{y}-\mathbf{x}\beta) \\
&= \frac{1}{n}\left(\mathbf{y}^T\mathbf{y}-\mathbf{y}^T\mathbf{x}\beta-\beta^T\mathbf{x}^T\mathbf{y}+\beta^T\mathbf{x}^T\mathbf{x}\beta\right)
\end{aligned}
$$

- Computer the gradient of the MSE with respect to β :

$$
\begin{aligned}
\nabla MSE(\beta) &= \frac{1}{n}\left(\nabla\mathbf{y}^T\mathbf{y}-2\nabla\beta^T\mathbf{x}^T\mathbf{y}+\nabla\beta^T\mathbf{x}^T\mathbf{x}\beta\right) \\
&= \frac{1}{n}\left(0-2\mathbf{x}^T\mathbf{y}+2\mathbf{x}^T\mathbf{x}\beta\right) \\
&= \frac{2}{n}\left(\mathbf{x}^T\mathbf{x}\beta-\mathbf{x}^T\mathbf{y}\right)
\end{aligned}
$$

gradient

- Set the gradient to 0, solve β :

$$\mathbf{x}^T\mathbf{x}\widehat{\beta}-\mathbf{x}^T\mathbf{y}=0$$

$$\widehat{\beta}=(\mathbf{x}^T\mathbf{x})^{-1}\mathbf{x}^T\mathbf{y}$$

http://www.stat.cmu.edu/~cshalizi/mreg/15/lectures/13/lecture-13.pdf

GD: LINEAR REGRESSION

- Gradient update:

$$\beta^{+} = \beta + \frac{\eta}{N} \sum_{i} \mathbf{x}_i^{\top} (y_i - \mathbf{x}_i \beta)$$

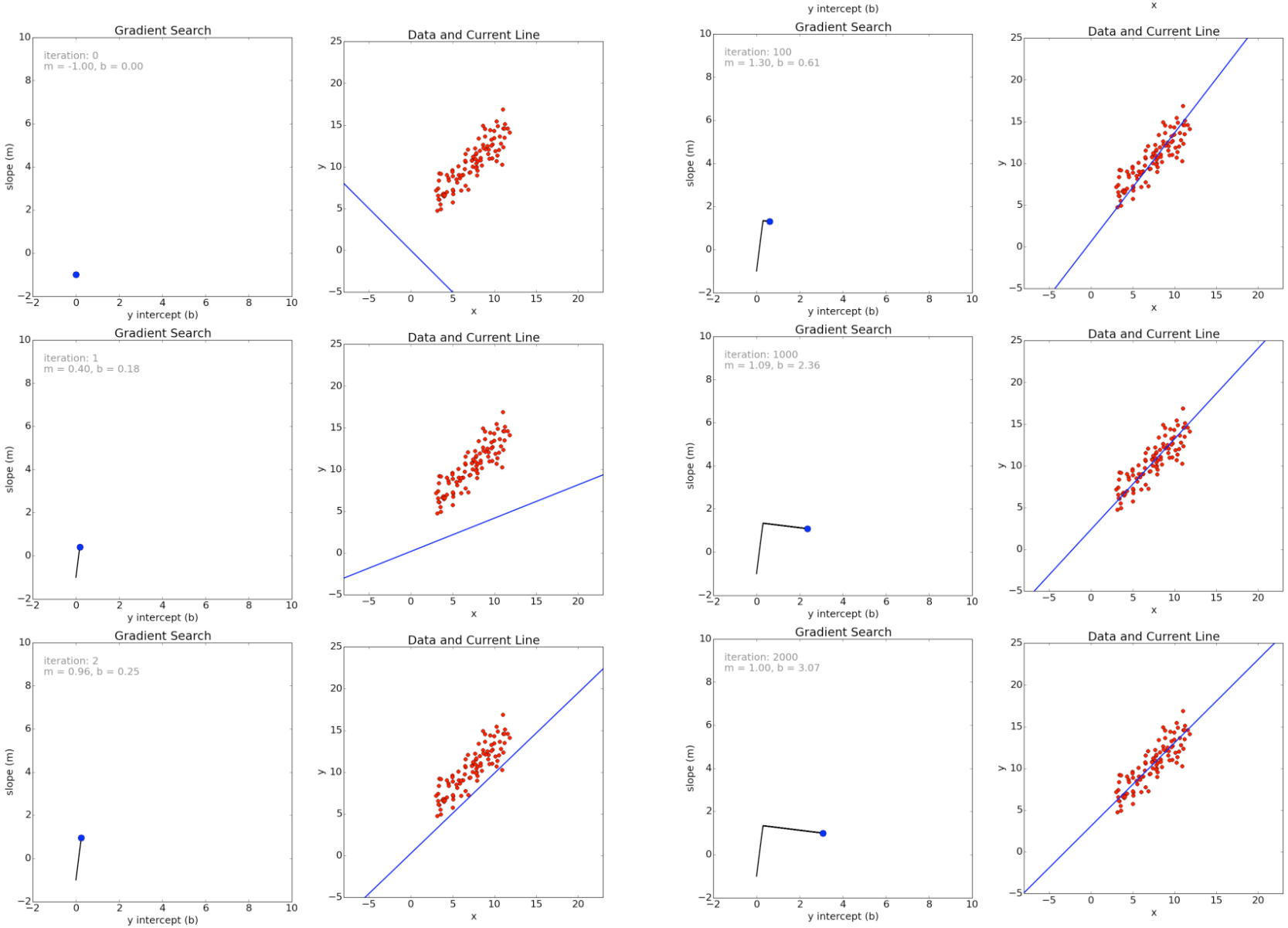
Algorithm 1: Gradient Descent

while *Not Converged* **do**
  $x^{(k+1)} = x^{(k)} - \eta^{(k)} \nabla f(x)$
end
return $x^{(k+1)}$

(Closed form solution)

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\top}\mathbf{X})^{-1}\mathbf{X}^{\top}\mathbf{y}$$

GD: LINEAR REGRESSION



http://spin.atomicobject.com/2014/06/24/gradient-descent-linear-regression/

GD & LINEAR REGRESSION

- Gradient update: $\beta^{+} = \beta + \frac{\eta}{N} \sum_{i} \mathbf{x}_i^{\top} (y_i - \mathbf{x}_i \beta)$

- Is this desirable for a large number of samples (e.g., 20 million)?

STOCHASTIC GRADIENT DESCENT (SGD)

- Estimate gradient from a single randomly picked sample

$$\nabla_\theta \ell(f(\mathbf{x}_i), y_i)$$

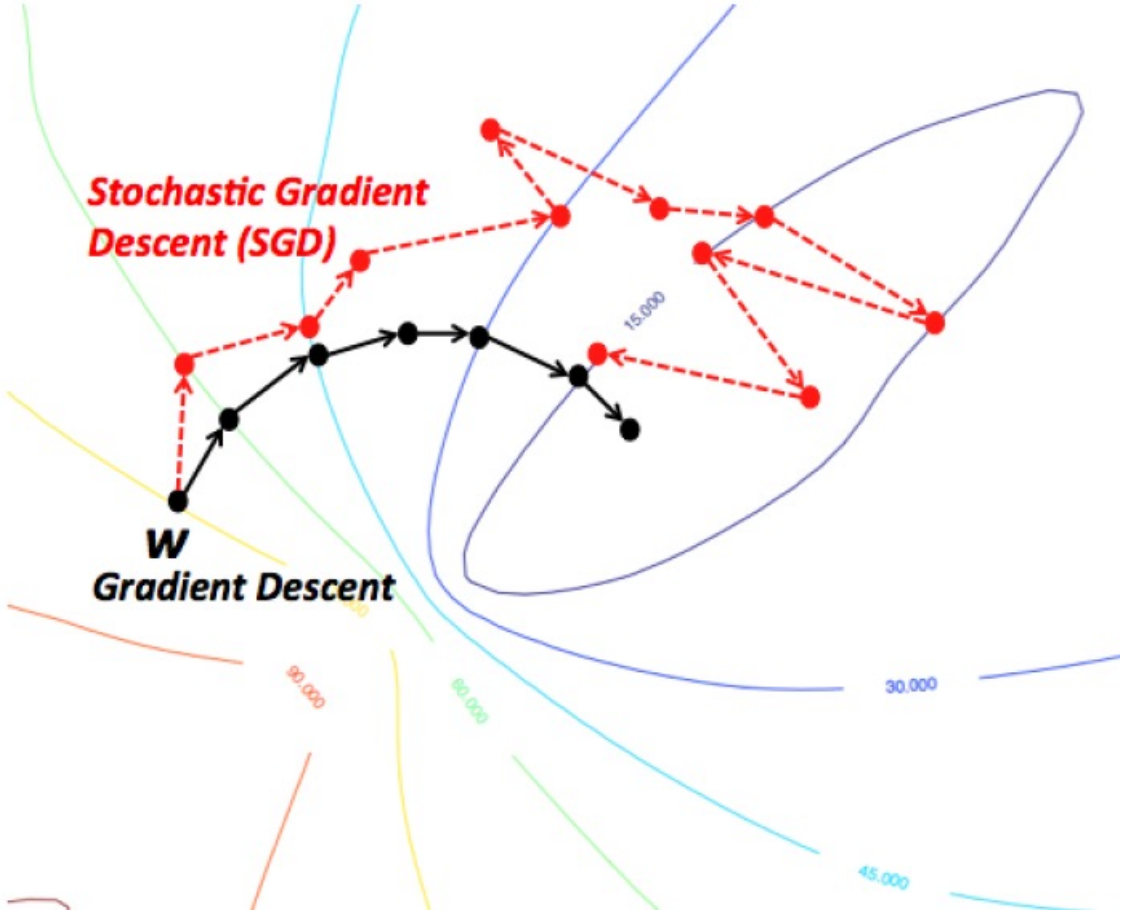
- The resulting update is of the form:

$$\theta^+ = \theta - \gamma \nabla_\theta \ell(f(\mathbf{x}_i), y_i)$$

- Although random noise is introduced, it behaves like gradient descent in its expectation

SGD ALGORITHM

Initialize parameter θ and learning rate η
while *not converged* **do**
  Randomly shuffle training data ← (Permutation based)
  for $i = 1, \cdot, N$ **do**
    $\theta^+ = \theta - \gamma \nabla_\theta \ell(f(\mathbf{x}_i), y_i)$
  end
end



Does SGD take more or fewer steps? Is each step faster or slower?

https://wikidocs.net/3413

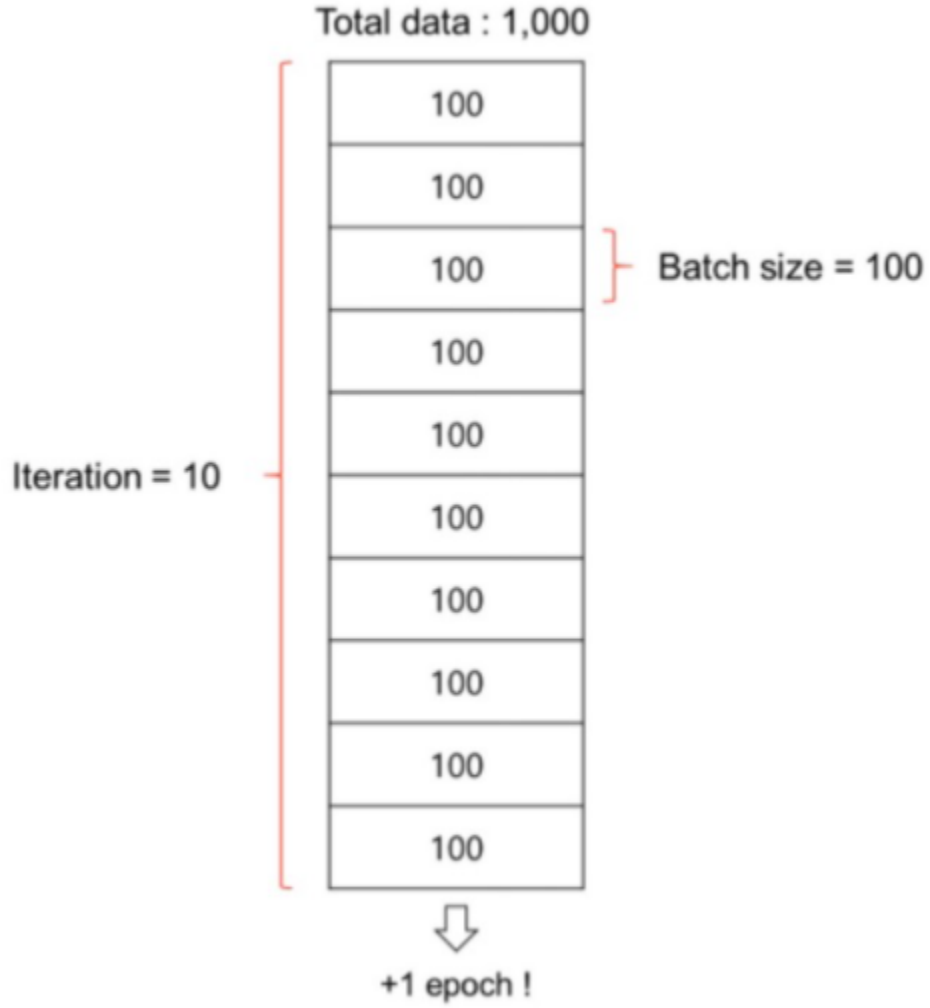
MINI-BATCH GRADIENT DESCENT

- (Batch, Full) Gradient descent: batch size = size of training set
  - Slow, more robust
- Stochastic gradient descent: batch size = 1
  - Fast, less stable
- Mini-batch gradient descent: 1 < batch size < size of training set (common values: 32, 64, 128)
  - Balance between robustness of gradient descent and efficiency of stochastic gradient descent

MINI-BATCH STOCHASTIC GRADIENT DESCENT

Algorithm 1 Stochastic Gradient Descent

```
for epoch = 1 : MAX EPOCH do
    Randomly shuffle training data and break into B = N/BatchSize batches
    for b = 1 : B do
        Compute the gradients associated with samples in batch b, ∇f
        Take the average of the gradients in the batch
        Update the parameters based on the gradient
    end for
end for
```



One **epoch** is one full pass of the dataset by the learning algorithms. An epoch is comprised of one or more batches

LINEAR REGRESSION (HINT FOR HW3)

- Negative gradient with respect to single sample:

$$\mathbf{x}_i^\top (y_i - \mathbf{x}_i \beta)$$

- Update cost (n records, d dimensions):
  - Batch/Full:
  - Stochastic:
  - Mini-batch:

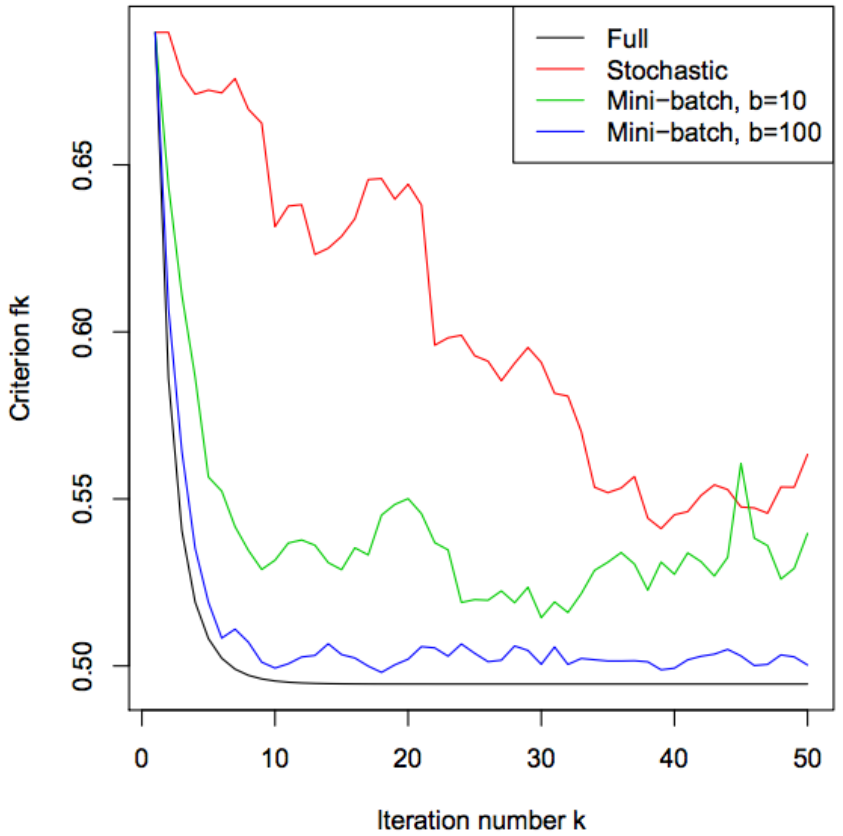
LINEAR REGRESSION (HINT FOR HW3)

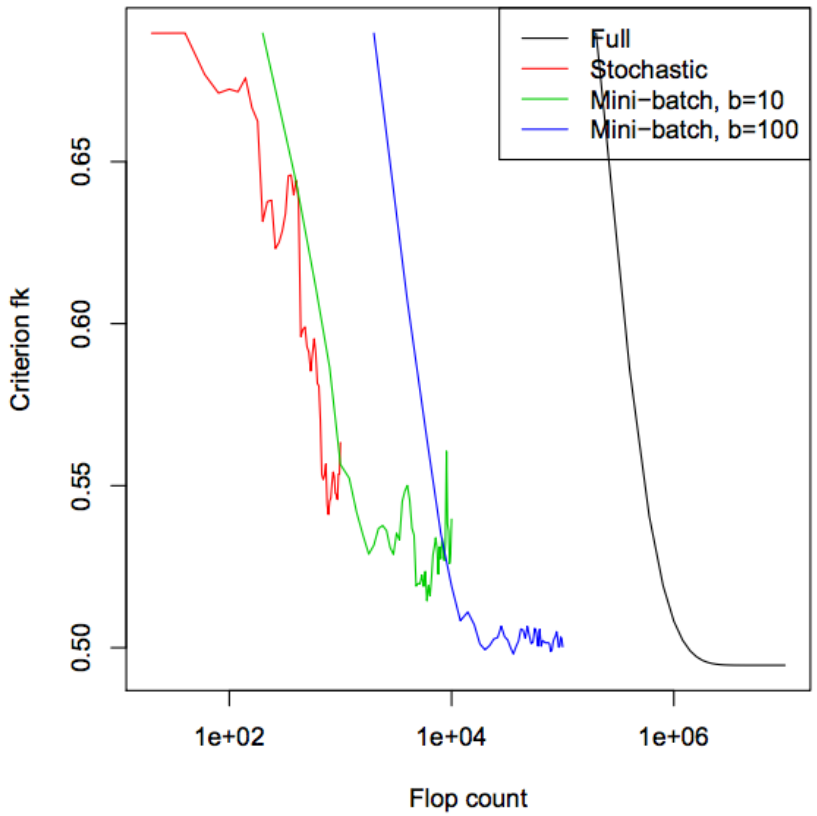
- Negative gradient with respect to single sample:

$$\mathbf{x}_i^\top (y_i - \mathbf{x}_i \beta)$$

- Update cost (n records, d dimensions):
  - Batch/Full: O(nd)
  - Stochastic: O(d)
  - Mini-batch: O(batch size * d)

SGD: IMPACT OF BATCH SIZE



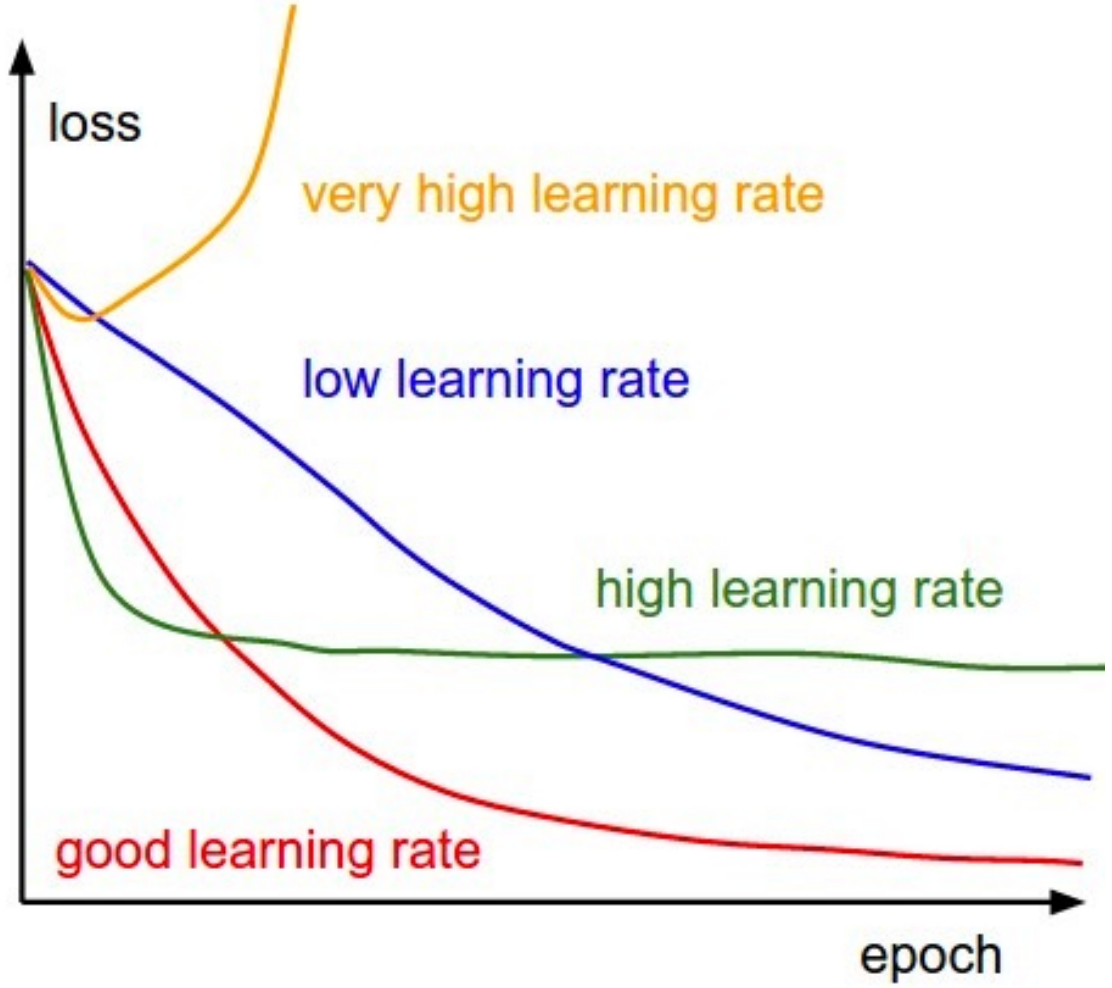


n=10,000
d=20

Iterations make better progress as mini-batch size b is larger but also takes more computation time

http://stat.cmu.edu/~ryantibs/convexopt/lectures/25-fast-stochastic.pdf

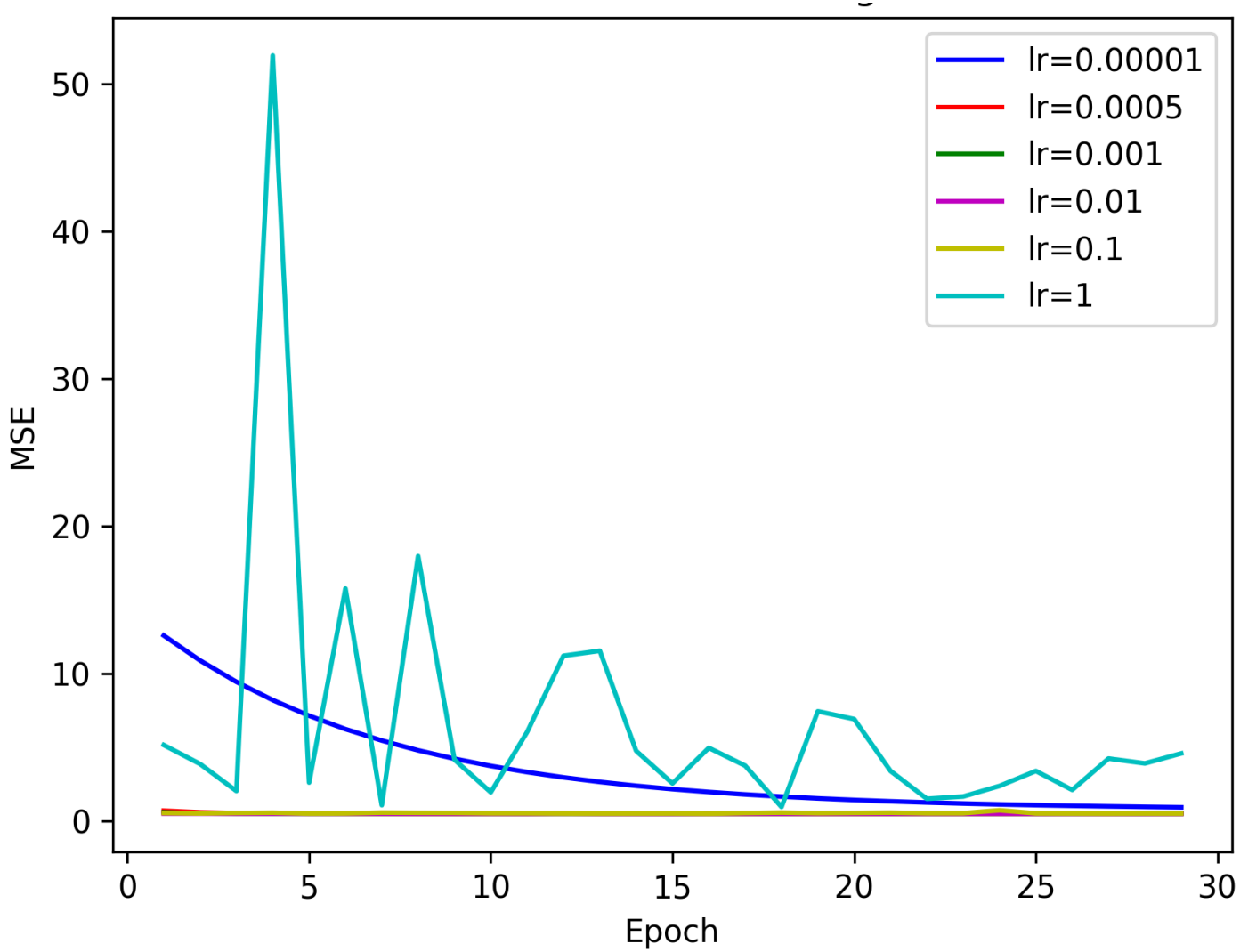
SGD: IMPACT OF LEARNING RATES (ILLUSTRATIVE)



SGD+GSD ☺

https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10

EXAMPLE: LEARNING CURVE

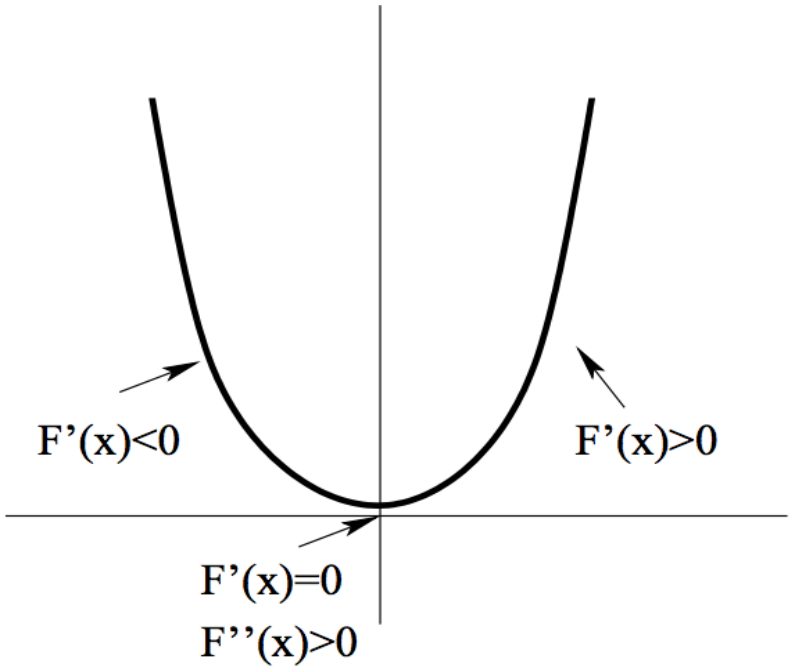


SGD: PRACTICAL TIPS

- Sensitive to feature scale:
  - Scale to [0, 1] or standardize
  - Randomly shuffle training examples before each epoch
  - Although theory says you should randomly pick examples, it is easier to make a pass through your training set sequentially (permutation based)
- Experiment with the learning rates using small sample of training set

LEARNING THE PARAMETERS

- Closed form (direct solution): set partial derivatives to zero and solve parameters, check that Hessian is great than 0
- Iterative algorithms: Gradient descent (GD) and Stochastic gradient descent (SGD)



Partial derivatives: https://www.khanacademy.org/math/multivariable-calculus

STANDARD LINEAR REGRESSION: LIMITATIONS

- Prediction accuracy:
  - perform well when n>>d
  - Otherwise no unique solutions and high variance (curse of dimensionality)
- Model interpretability: hard to interpret if too many features
- Irrelevant and redundant features (general problem for ML)

LEAST SQUARES AND BEST SUBSET

Term	LS	Best Subset
Intercept	2.465	2.477
lcavol	0.680	0.740
lweight	0.263	0.316
age	−0.141	
lbph	0.210	
svi	0.305	
lcp	−0.288	
gleason	−0.021	
pgg45	0.267	
Test Error	0.521	0.492
Std Error	0.179	0.143

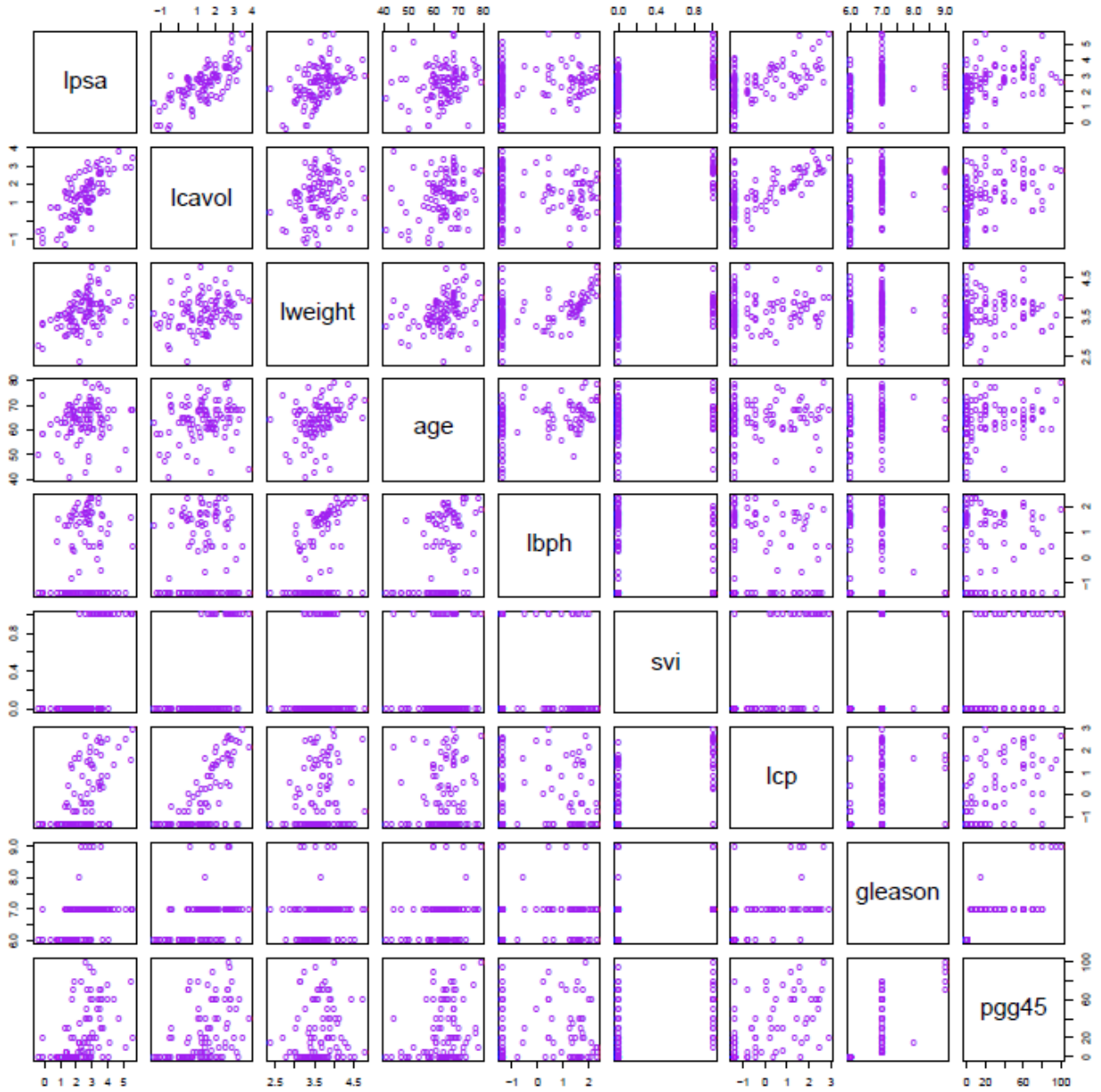


FIGURE 1.1. *Scatterplot matrix of the prostate cancer data. The first row shows the response against each of the predictors in turn. Two of the predictors,* `svi` *and* `gleason`*, are categorical.*

GROUP ACTIVITY

LINEAR REGRESSION – FEATURE SELECTION

- How can we modify least squares to select a subset of features, i.e. force some of the coefficients to 0?

$$\min_{\beta} L(\mathbf{X}\boldsymbol{\beta}, \mathbf{y})$$

$$RSS(\boldsymbol{\beta}) = \frac{1}{2}\sum_{i=1}^{N}(y_i - f(\mathbf{x}_i)^2)$$
$$= \frac{1}{2}(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})^\top(\mathbf{y} - \mathbf{X}\boldsymbol{\beta})$$

$$f(\mathbf{x}) = \beta_0 + \sum_{i=1}^{p} x_i\beta_i$$

MODEL REGULARIZATION

$$\min_{\beta} L(\mathbf{X}\boldsymbol{\beta}, \mathbf{y}) + \lambda \text{penalty}(\boldsymbol{\beta})$$

- Basic Idea: Add penalty term on model parameters to "shrink" the coefficients towards zero
- Called regularization
- Less prone to overfitting (prediction accuracy)
- Achieve a simpler model, get the "right" model complexity (interpretability)

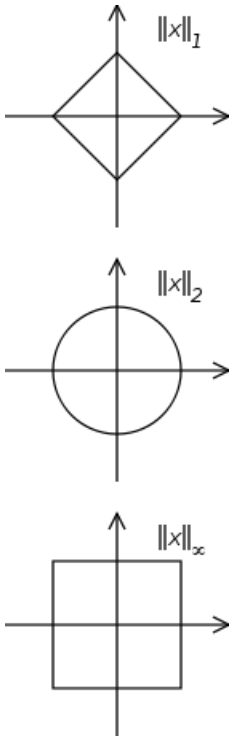
REGULARIZATION PARAMETER

$$\min_{\beta} L(\mathbf{X}\boldsymbol{\beta}, \mathbf{y}) + \lambda \text{penalty}(\boldsymbol{\beta})$$

- Tuning parameter (λ) controls the strength of the penalty term
- When 0, linear regression estimate
- When infinity, coefficients go to 0
- For in between, balance the fit of the model with shrinking coefficients

REVIEW: COMMON VECTOR NORMS

Norm	Formula
Taxicab (Manhattan)	$\|\|\mathbf{x}\|\|_1 = \sum_{i=1}^{n} \|x_i\|$
Euclidean	$\|\|\mathbf{x}\|\|_2 = \sqrt{\sum_{i=1}^{n} x_i^2}$
Maximum (infinity)	$\|\|\mathbf{x}\|\|_\infty = \max_{x_i} \|x_i\|$
p-norm	$\|\|\mathbf{x}\|\|_p = \left(\sum_{i=1}^{n} \|x_i\|^p \right)^{1/p}$



POPULAR PENALTIES

Name	Penalty function
Ridge	$\|\|\beta\|\|_2$
Lasso	$\|\|\beta\|\|_1$
L0 regularization	$\|\|\beta\|\|_0$
Elastic net	$\alpha\|\|\beta\|\|_1 + (1-\alpha)\|\|\beta\|\|_2$

RIDGE REGULARIZATION

- L2 norm discourages large values

$$\min_{\beta} L(\mathbf{X}\boldsymbol{\beta}, \mathbf{y}) + \lambda ||\boldsymbol{\beta}||_2$$

- Also known as shrinkage (statistics) or weight decay (neural nets)
- Closed form solution

$$\hat{\boldsymbol{\beta}} = (\lambda \mathbf{I} + \mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

- Better apply Ridge after standardization

LASSO REGULARIZATION

- L1 norm drives some coefficients to zero

$$\min_{\beta} L(\mathbf{X}\boldsymbol{\beta}, \mathbf{y}) + \lambda ||\boldsymbol{\beta}||_1$$

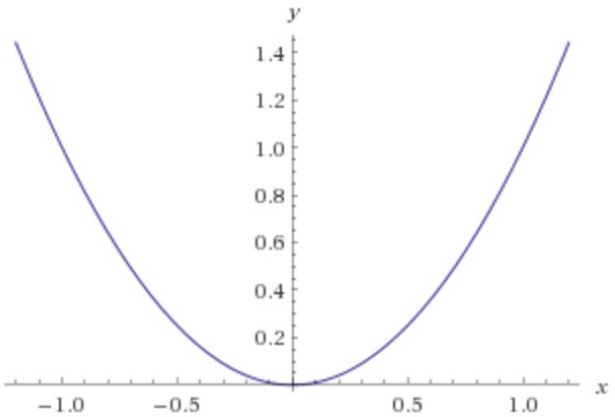
- Performs subset selection, yields sparse models

- No closed form solution but efficient algorithms exist (least angle regression LAR) with approximately same computational cost as ridge

Lasso = Least Absolute Selection and Shrinkage Operator

GD: RIDGE AND LASSO

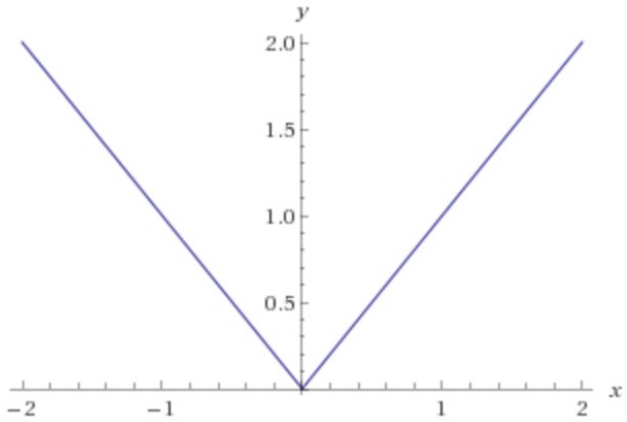
Ridge



$$\frac{\partial}{\partial x}\lambda||x||_2 = \pm\lambda x$$

Push towards 0 gets weaker as x gets smaller

Lasso



$$\frac{\partial}{\partial x}\lambda||x||_1 = \pm\lambda$$

Always pushes elements towards 0

EFFECT OF SELECTION ON COEFFICIENTS

Term	LS	Best Subset	Ridge	Lasso
Intercept	2.465	2.477	2.452	2.468
lcavol	0.680	0.740	0.420	0.533
lweight	0.263	0.316	0.238	0.169
age	−0.141		−0.046	
lbph	0.210		0.162	0.002
svi	0.305		0.227	0.094
lcp	−0.288		0.000	
gleason	−0.021		0.040	
pgg45	0.267		0.133	
Test Error	0.521	0.492	0.492	0.479
Std Error	0.179	0.143	0.165	0.164

How did this happen?

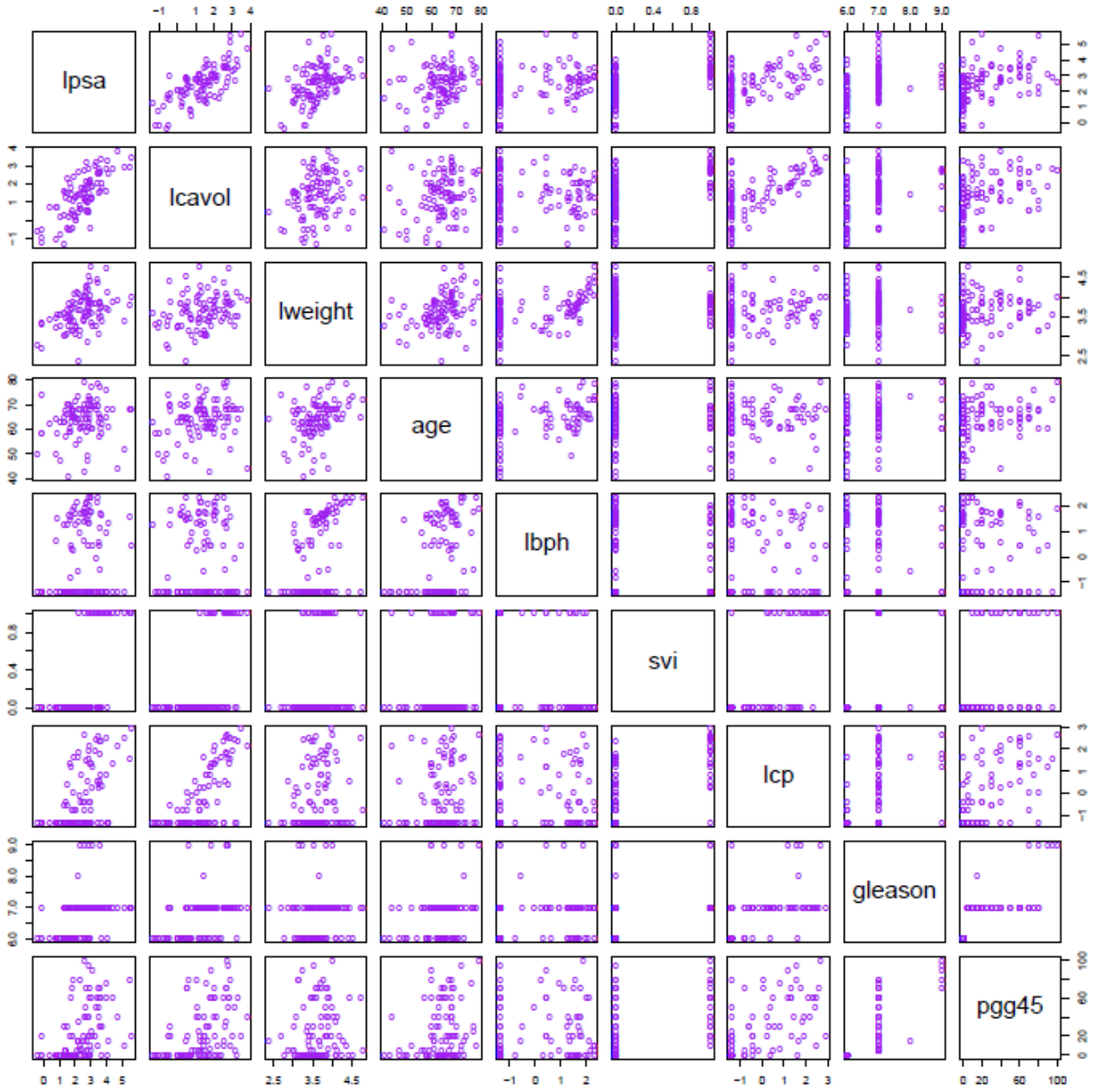
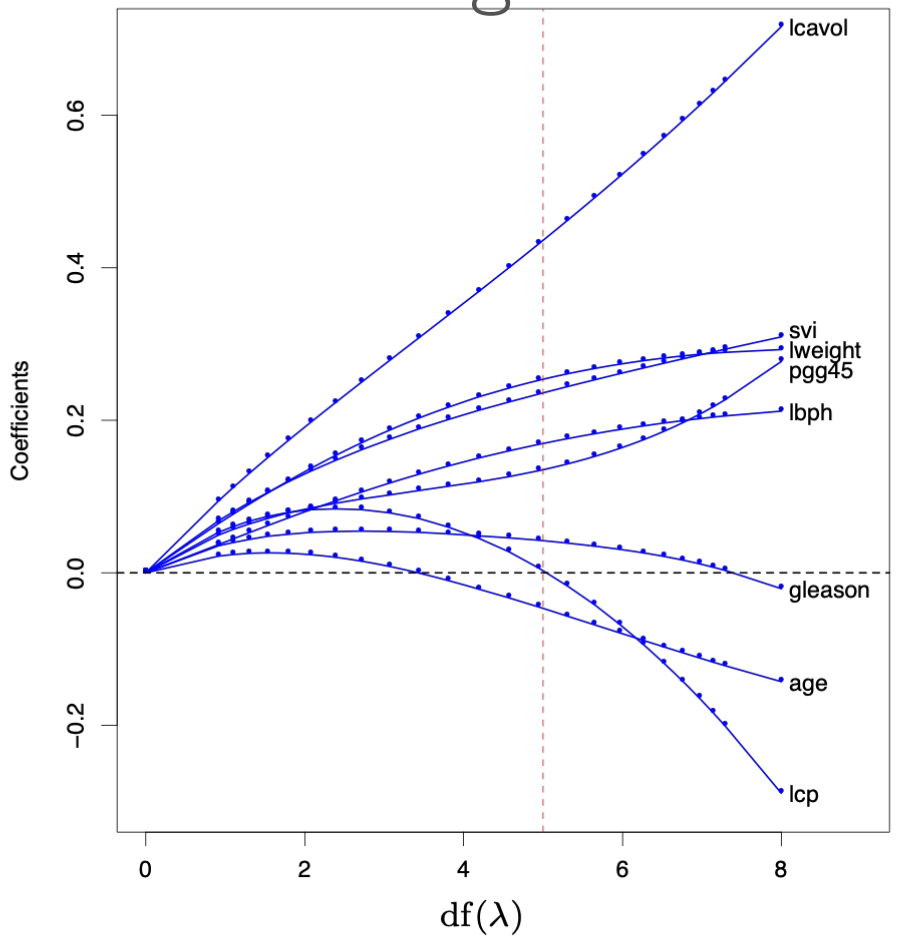


FIGURE 1.1. *Scatterplot matrix of the prostate cancer data. The first row shows the response against each of the predictors in turn. Two of the predictors,* `svi` *and* `gleason`*, are categorical.*

RIDGE VS. LASSO: COEFFICIENT PATHS

$$\min_{\beta} L(\mathbf{X}\boldsymbol{\beta}, \mathbf{y}) + \lambda||\boldsymbol{\beta}||_2$$

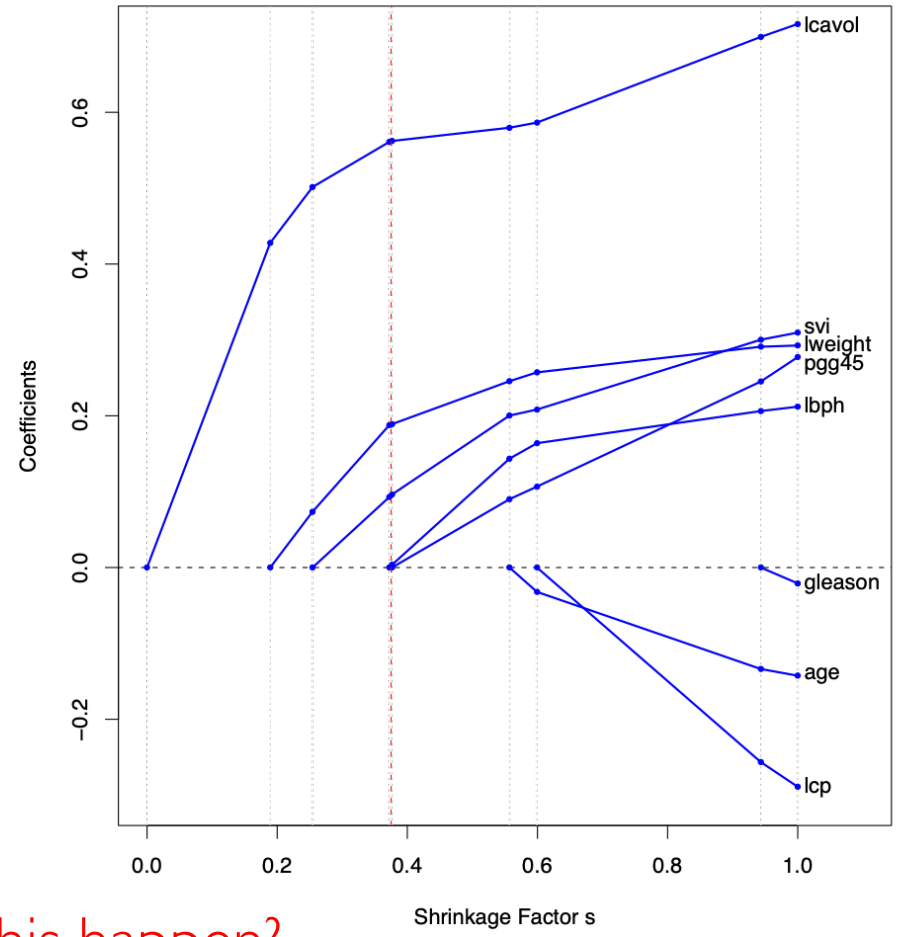
Ridge



effective degrees of freedom (inverse to lamda)

$$\min_{\beta} L(\mathbf{X}\boldsymbol{\beta}, \mathbf{y}) + \lambda||\boldsymbol{\beta}||_1$$

LASSO



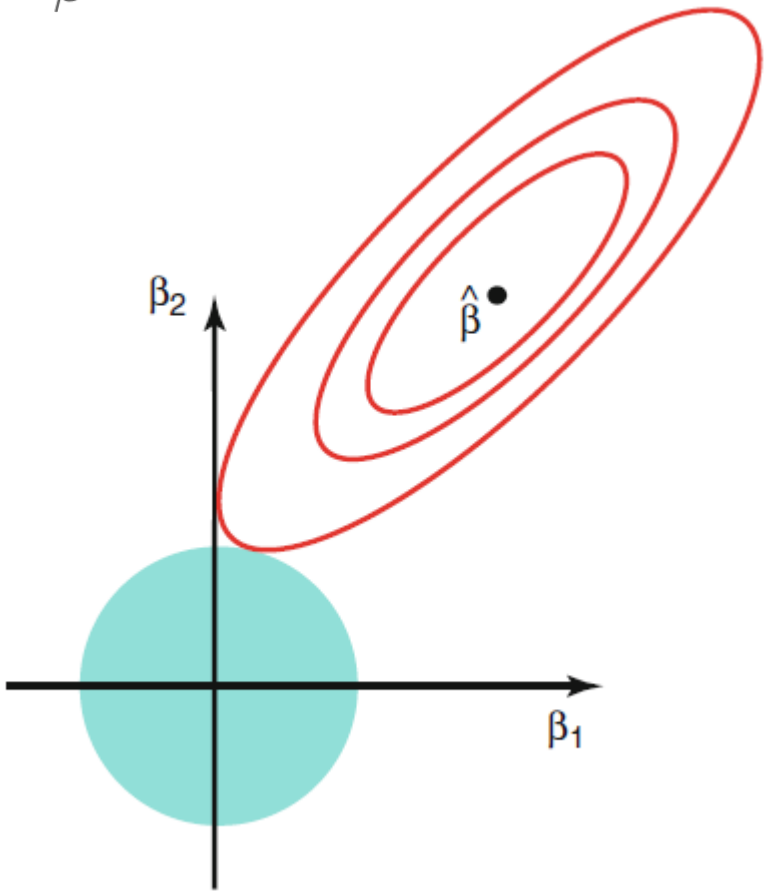
Why did this happen?

Figure 3.8 and 3.10 (Hastie et al.)

RIDGE VS. LASSO: OPTIMIZATION

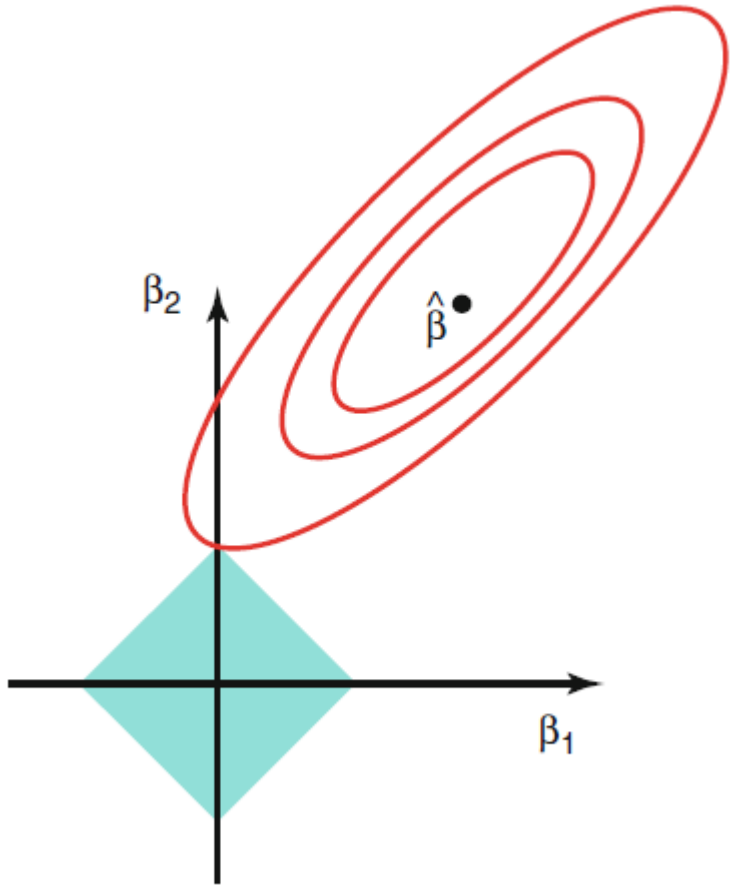
Ridge

$$\min_{\beta} L(\mathbf{X}\boldsymbol{\beta}, \mathbf{y}) + \lambda ||\boldsymbol{\beta}||_2$$



LASSO

$$\min_{\beta} L(\mathbf{X}\boldsymbol{\beta}, \mathbf{y}) + \lambda ||\boldsymbol{\beta}||_1$$



ure 6.7 (James et al.)

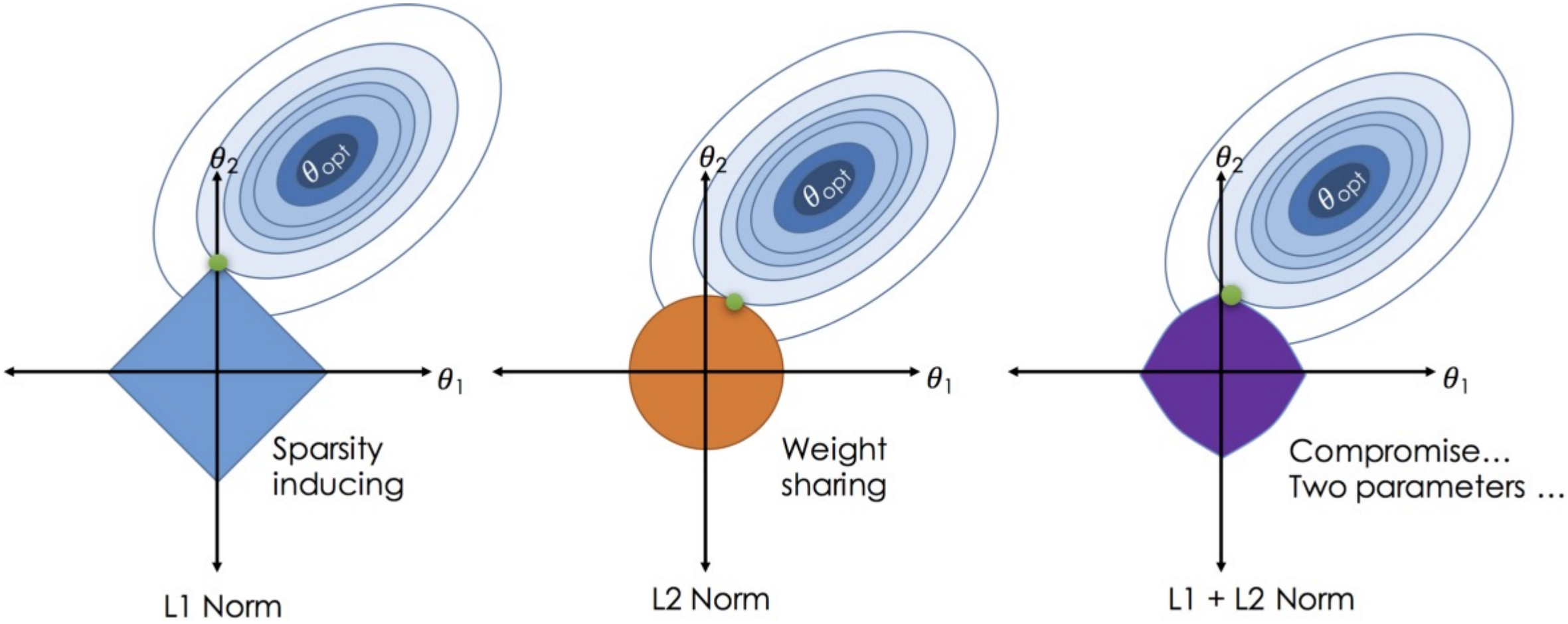
ELASTIC NET REGULARIZATION

- Compromise between ridge and lasso

$$\min_{\beta} L(\mathbf{X}\boldsymbol{\beta}, \mathbf{y}) + \lambda(\alpha||\boldsymbol{\beta}||_2 + (1-\alpha)||\boldsymbol{\beta}||_1)$$

- Selects variables like lasso
- Shrinks coefficients of correlated predictions like ridge
- Computational advantages over general L_q penalties

RIDGE VS LASSO VS ELASTIC NET



RIDGE & LASSO REGULARIZATION: NOTES

- If intercept term is included in regression, this coefficient is left unpenalized
- Can center the data, only perform regression on other coefficients
- Penalty term can be unfair if predictors are on different scales
  - Normalization

LINEAR REGRESSION

- Closed form (direct solution)
- Iterative algorithms: Gradient descent (GD) and Stochastic gradient descent (SGD)
- Regularization: Ridge and Lasso
- Assessment