Regularized Models

Lecture 4b

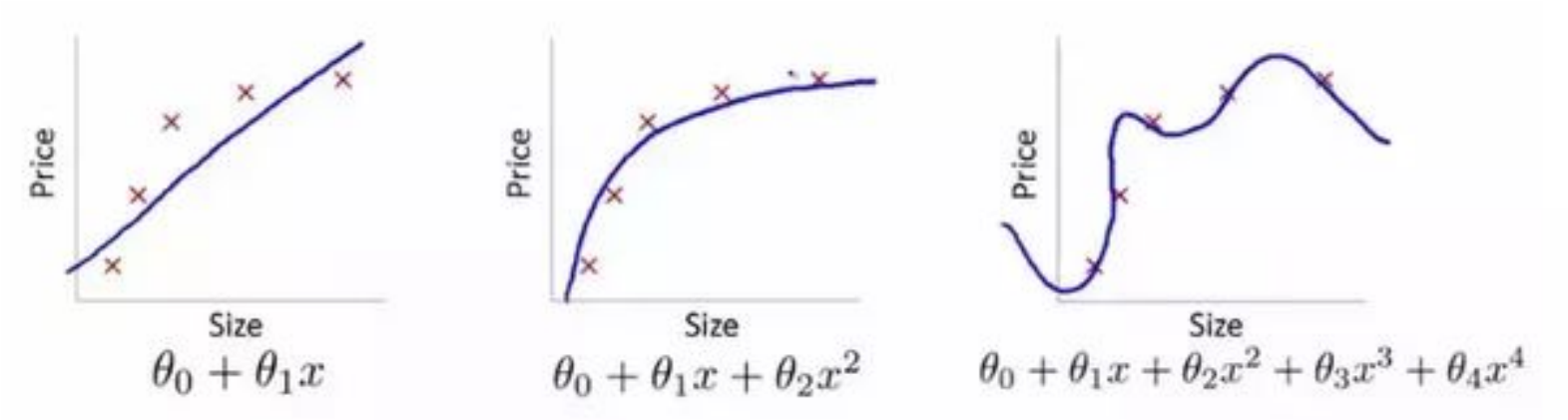
Today: Learning Objectives

- ❏ Know how to avoid overfitting with **regularization**
- ❏ Understand different regularized models including **Ridge Regression**, **Lasso Regression**, and **Elastic Net**
- ❏ Learn how to use **early stopping** to regulate an iterative learning algorithm


THE BEST WAY TO
EXPLAIN OVERFITTING

What's overfitting?

Overfitting: the learned model may fit the training set very well, but fail to generalize to new samples. An example of housing prices:



Underfit
(high **bias**)

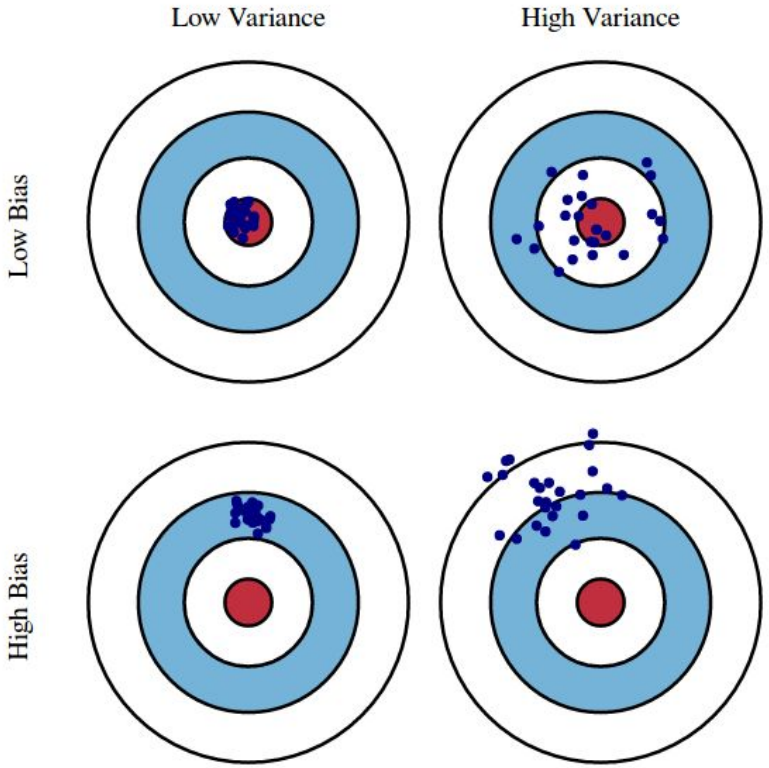
Just right

Overfit
(high **variance**)

*Figure from Andrew Ng, Stanford CS229

Model's Bias and Variance

Model's Error = **Bias** +
Variance +
Irreducible Error

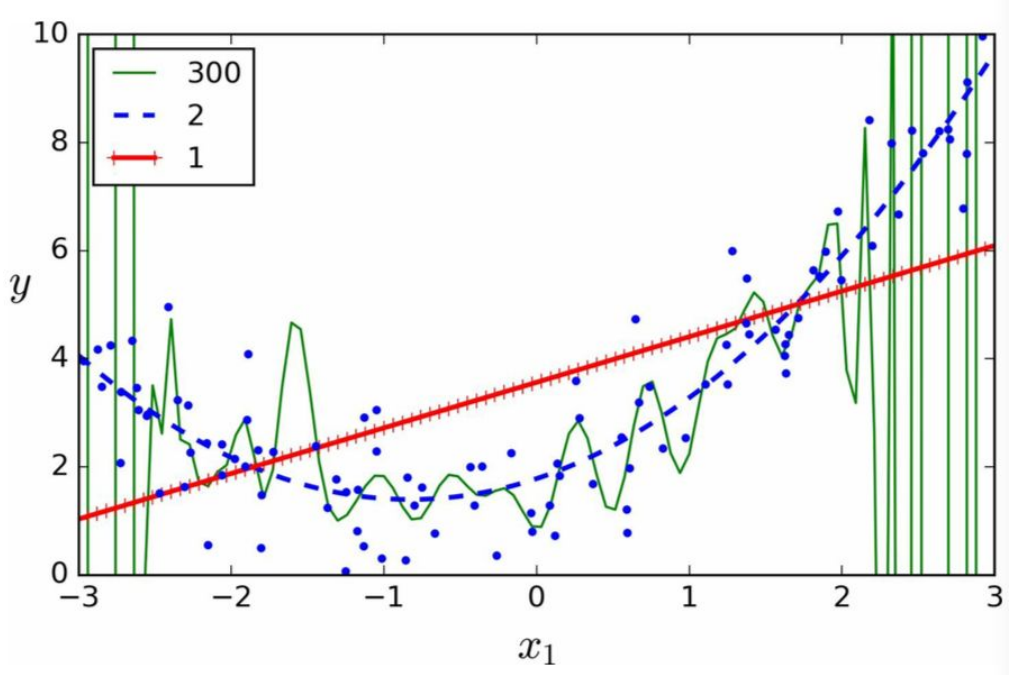


Bias: Bias refers to the error that is introduced by approximating a real-life problem, which may be extremely complicated, by a much simpler model.

Variance: Variance refers to the amount by which your prediction would change if we estimated it using a different training data set. Since the training data is used to fit the statistical learning method, different training data sets will result in a different estimation.

Higher degree Polynomial

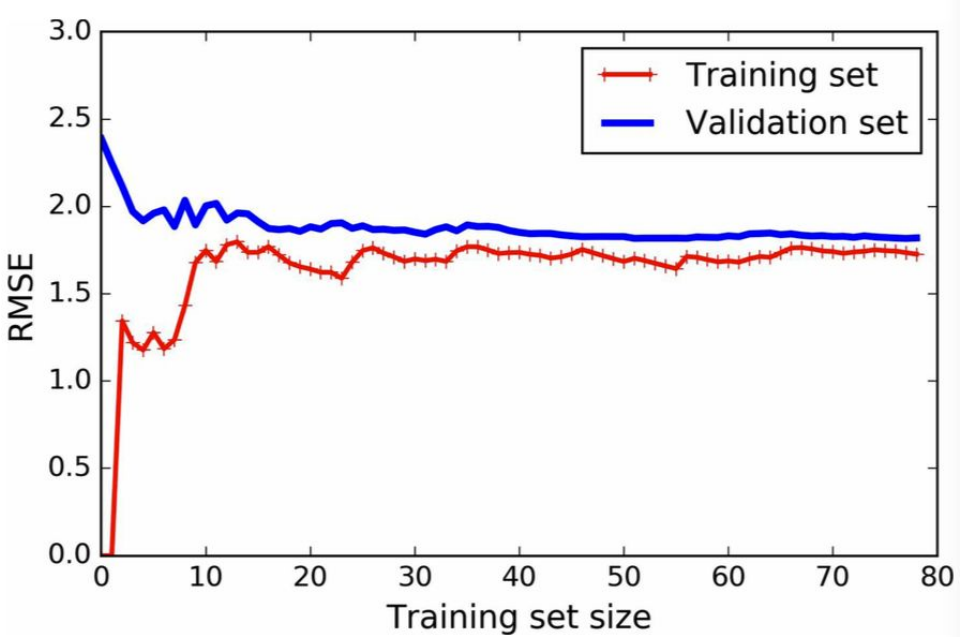
Does higher-degree polynomial regression always **fit** better to training data?



Underfitting vs. Overfitting

Check for under-fitting with learning curves

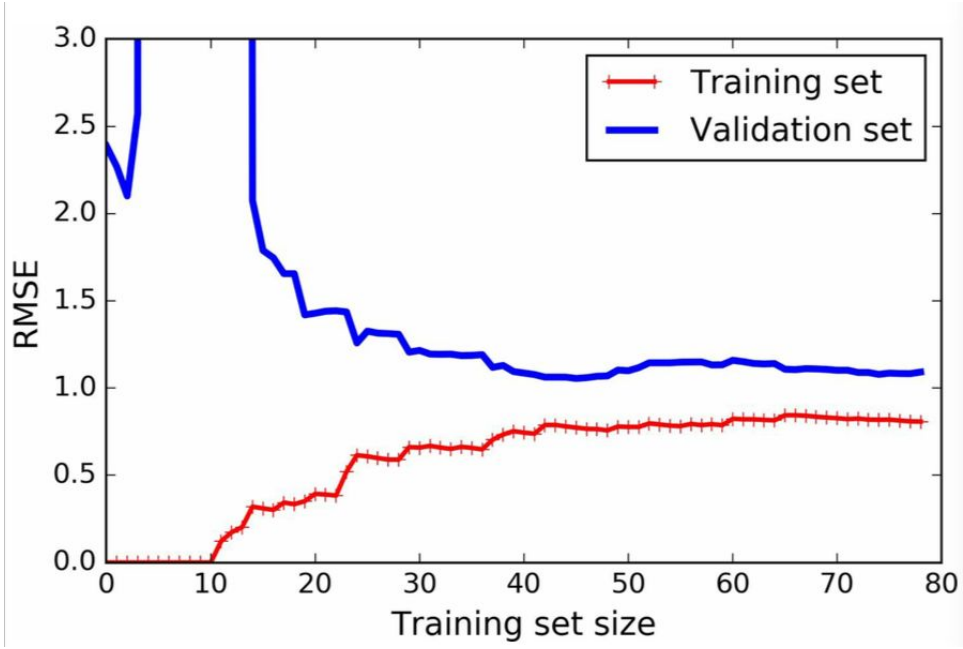
Plot of model's performance as a function of training set size.



<u>Underfitting</u>: Adding more training examples will not help

Learning curves of high-degree model

Plot of model's performance as a function of training set size.



<u>Overfitting:</u> Notice the gap between training and validating

Regularized Linear Models

One way to reduce **overfitting** is to regularize the model

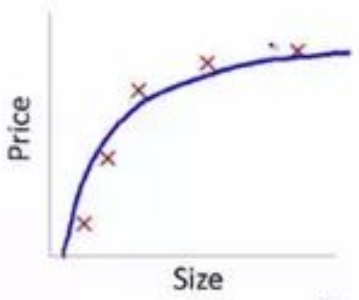
“The **fewer** degrees of freedom it has, the **harder** for it to overfit the data”

Regularization can be achieved by **constraining the weights** of the model:

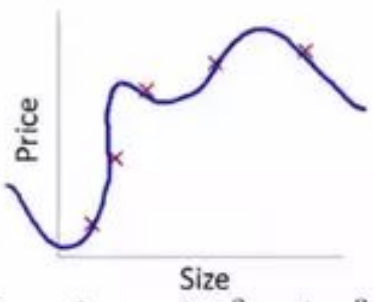
- Ridge Regression
- Lasso Regression
- Elastic Net

Regularization significantly reduces the variance of the model, **without** substantial increase in its bias

Regulating the weights



$\theta_0 + \theta_1 x + \theta_2 x^2$



$\theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4$

Suppose you want to make the below model “simpler”:

$$\hat{y} = \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3^3 + \theta_4 x_4^4$$

Instead of getting rid of the cubic term entirely, we regulate the cost function:

$$J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2 + 1000 \times \theta_3 + 1000 \times \theta_4$$

This extra term would assure that the value of θ_3, θ_4 will be near zero,

which greatly reduce the value of last two terms, so the model will be simpler..

Ridge Regression: encourage simplicity

Forces the learning algorithm to not only fit the data but also **keep the parameters as small as possible (near zero)** → discourage learning a complex model.

Add an **L2 norm** into the Linear Regression

$$J_{RR}(\theta) = J(\theta) + \lambda ||\theta||_2^2$$

$$J_{RR}(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} \theta_j^2$$

DATA LOSS (MSE)

REGULARIZATION

Regularization hyperparameter controls how much you want to regularize the model

Tuning Hyperparameter λ

$$J_{RR}(\theta) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2 + \lambda\sum_{j=1}^{n}\theta_j^2$$

What happen when hyperparameter is **near zero?**
or when it is **large?**

- When $\lambda = 0$, no parameters are regularized → linear regression
- As λ increases, more and more parameters are penalized (to near zero) → simpler model → more **bias**
- As λ decreases → more complex model → more **variance**.
- **Cross validation** comes in handy when you need to tune the value of λ

Ridge Gradient Descent

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta), (j = 1...n)$$

Plug in our new ridge regression cost function:

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{\partial}{\partial \theta_j} \left(\frac{1}{m} \sum_{i=1}^{m} (h_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} \theta_j^2 \right)$$

Take the partial derivative:

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{2}{m} \sum_{i=1}^{m} \left(h_\theta(\mathbf{x}^{(i)}) - y^{(i)} \right) x_j^{(i)} + 2\lambda\theta_j$$

Plug back into the definition:

$$\theta_j = \theta_j - \alpha \left(\frac{2}{m} \sum_{i=1}^{m} \left(h_\theta(\mathbf{x}^{(i)}) - y^{(i)} \right) x_j^{(i)} + 2\lambda\theta_j \right)$$

Ridge Gradient Descent

$$\theta_j = \theta_j - \alpha\left(\frac{2}{m}\sum_{i=1}^{m}\left(h_\theta(\mathbf{x}^{(i)}) - y^{(i)}\right)x_j^{(i)} + 2\lambda\theta_j\right)$$

$$\theta_j = \theta_j - \alpha 2\lambda\theta_j - \alpha\frac{2}{m}\sum_{i=1}^{m}\left(h_\theta(\mathbf{x}^{(i)}) - y^{(i)}\right)x_j^{(i)}$$

$$\theta_j = \theta_j(1 - 2\alpha\lambda) - \ldots$$

> 0

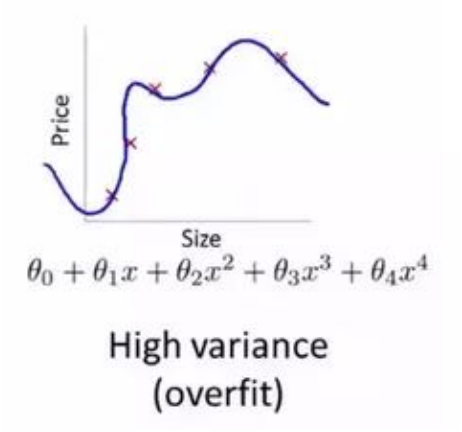
$\rightarrow$ shrinkage

Ridge Normal Equation (closed-form)

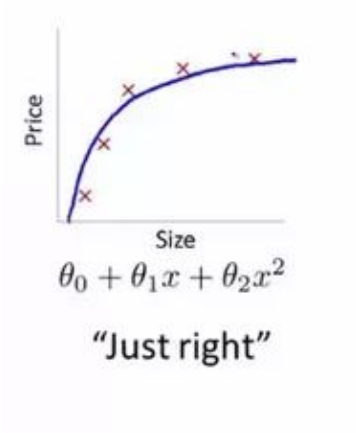
$$\theta = \left(X^T X + \lambda \begin{bmatrix} 0 & & & \\ & 1 & & \\ & & 1 & \\ & & & \ddots \\ & & & & 1 \end{bmatrix} \right)^{-1} X^T y$$

Extra Credit Opportunity: See if you can derive the cost function minimization into this closed-form!

Avoid overfitting with sparsity



$\theta_3 = 0; \theta_4 = 0;$



- Sparse model has a lot of zeros on its weights
- Many of its features are **forced** to be completely **useless** while other features become more **useful**
- It becomes a simpler model which reduces the chance of overfitting

LASSO Regression: encourage sparsity

Uses **L-1 norm** of the weight vector instead of **L-2 norm** like Ridge

L-1 norm tends to completely eliminate the weights on less important features

$$J_{LASSO}(\theta) = J(\theta) + \lambda ||\theta||_1$$

$$J_{LASSO}(\theta) = \frac{1}{m} \sum_{i=1}^{m} (h_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} |\theta_j|$$

→ performs **feature selection** and yields a **sparse** model

Equivalent form of cost function

$$J_{LASSO}(\theta) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} |\theta_j|$$

DATA LOSS (MSE)

REGULARIZATION

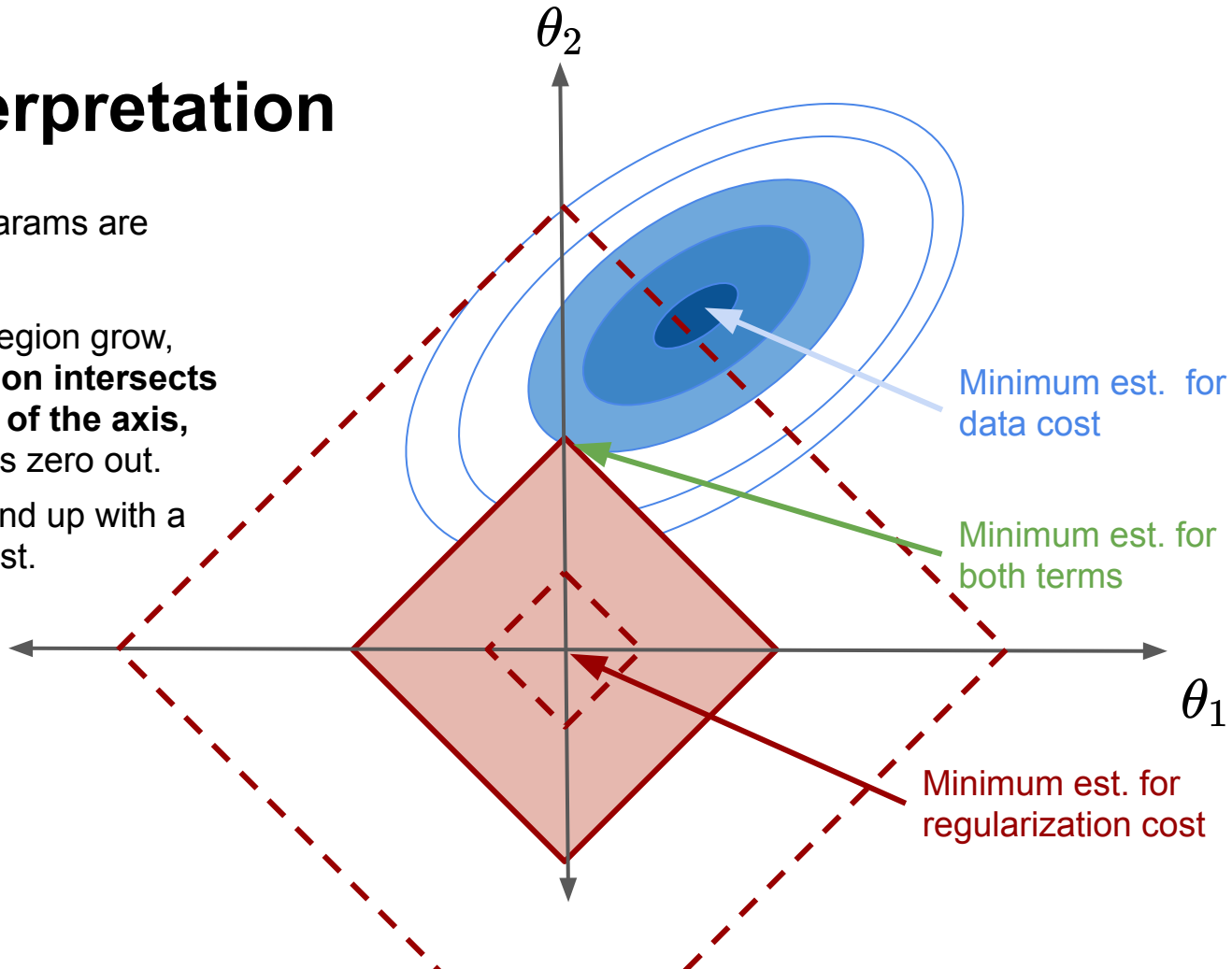
Rewrite to make the size constraints explicit (one-to-one correspondence between the hyperparameter `\lambda` and `t`)

$$\hat{\theta} = \arg\min_{\theta} \left(\frac{1}{m}\sum_{i=1}^{m}(h_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2 \right)$$

$$\text{subject to} \quad \sum_{j=1}^{n} |\theta_j| \leq t.$$

Geometry Interpretation

- When t is too small, the params are not useful to the model.
- When as the constraints region grow, **due to its shape the region intersects the data contour on one of the axis,** thus the other parameter is zero out.
- When t is too large, you end up with a similar params as data cost.



Can we utilize both Ridge and Lasso for regularization?

Elastic Net

A middle ground between Ridge and Lasso Regression with a mix ratio r

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2 + \underbrace{r\lambda\sum_{j=1}^{n}|\theta_j|}_{\text{LASSO}} + \underbrace{(1-r)\lambda\sum_{i=1}^{n}\theta_j^2}_{\text{RIDGE}}$$

- Preferable to have at least a little bit of regularization, **Ridge** if a good default
- If you suspect that only a few features are actually useful, use **Lasso**
- Lasso may behave erratically when #features > #samples, use **Elastic Net**

Geometry Interpretation

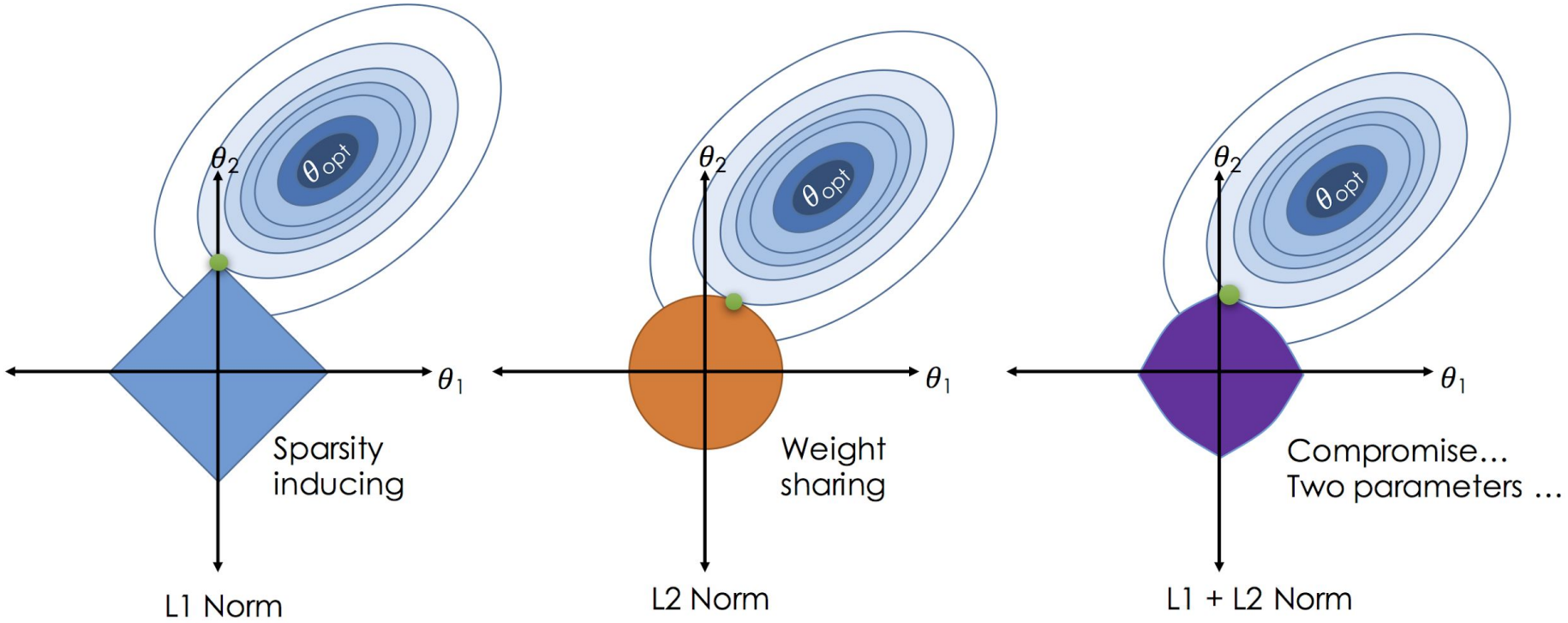
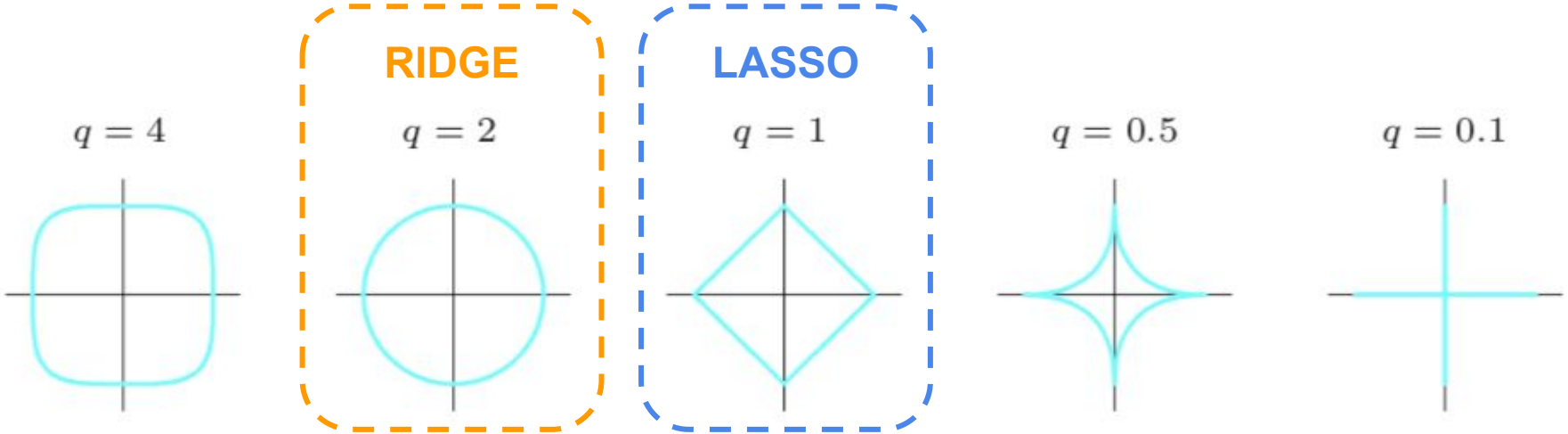


*Figure from Yin Hu, Medium

Family of Regularized Models

$$J(\theta) = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(\mathbf{x}^{(i)}) - y^{(i)})^2 + \lambda \sum_{j=1}^{n} |\theta_j|^q$$



Code Samples

```
from sklearn.linear_model import Ridge
ridge_reg = Ridge(alpha=1, solver="cholesky", random_state=42)
ridge_reg.fit(X, y)
ridge_reg.predict([[1.5]])
```

```
array([[1.55071465]])
```

```
from sklearn.linear_model import Lasso
lasso_reg = Lasso(alpha=0.1)
lasso_reg.fit(X, y)
lasso_reg.predict([[1.5]])
```

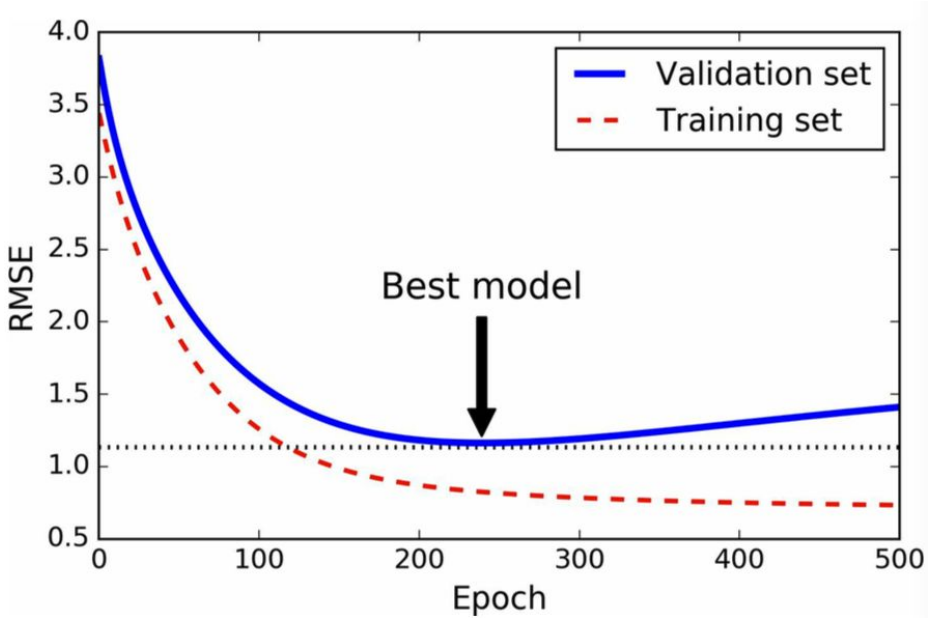
```
array([1.53788174])
```

```
from sklearn.linear_model import ElasticNet
elastic_net = ElasticNet(alpha=0.1, l1_ratio=0.5, random_state=42)
elastic_net.fit(X, y)
elastic_net.predict([[1.5]])
```

```
array([1.54333232])
```

Early Stopping

A different way to regularize **iterative** learning algorithms (ie. gradient descent) is to **stop training** as soon as the validation error reaches a minimum.



Implementation

```
from sklearn.base import clone
sgd_reg = SGDRegressor(n_iter=1, warm_start=True, penalty=None,
                       learning_rate="constant", eta0=0.0005, random_stat

minimum_val_error = float("inf")
best_epoch = None
best_model = None
for epoch in range(1000):
    sgd_reg.fit(X_train_poly_scaled, y_train)  # continues where it left
    y_val_predict = sgd_reg.predict(X_val_poly_scaled)
    val_error = mean_squared_error(y_val_predict, y_val)
    if val_error < minimum_val_error:
        minimum_val_error = val_error
        best_epoch = epoch
        best_model = clone(sgd_reg)
```

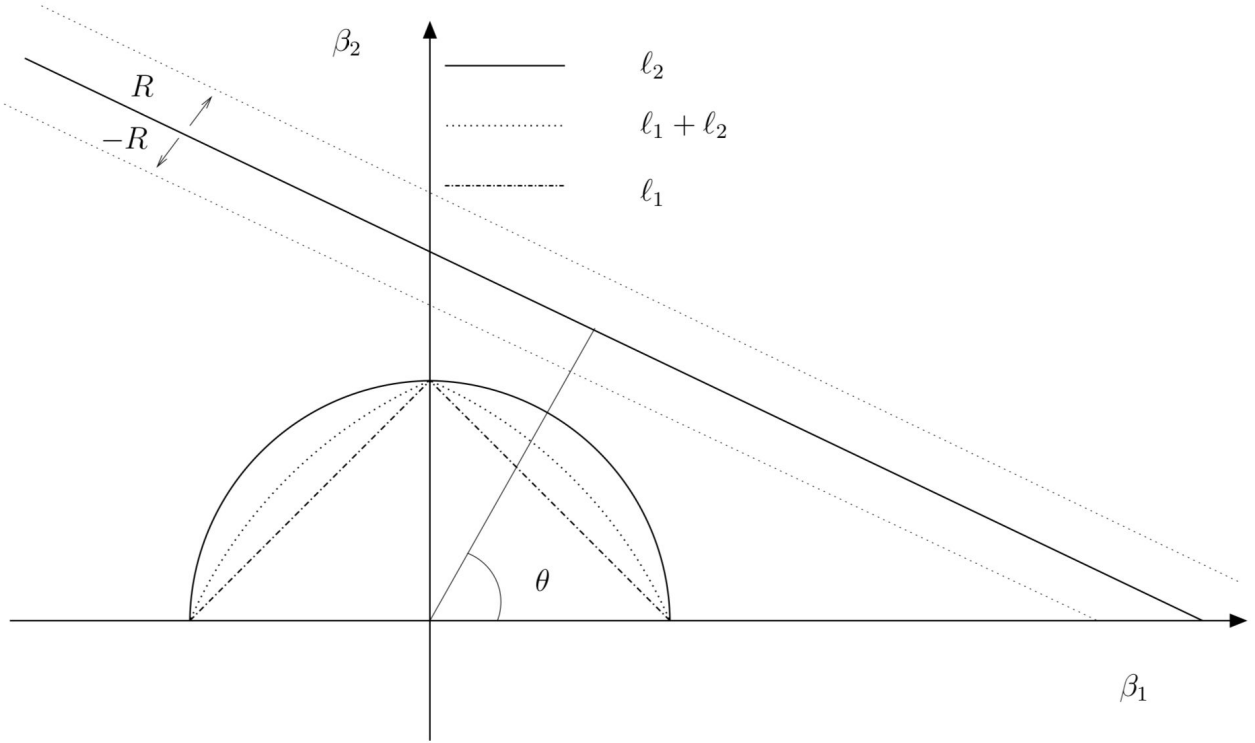
Today: Learning Objectives

- ✓ Know how to fight overfitting with **regularization**
- ✓ Understand different regularized models including **Ridge Regression**, **Lasso Regression**, and **Elastic Net**
- ✓ Learn how **early stopping** can regularize iterative learning algorithm

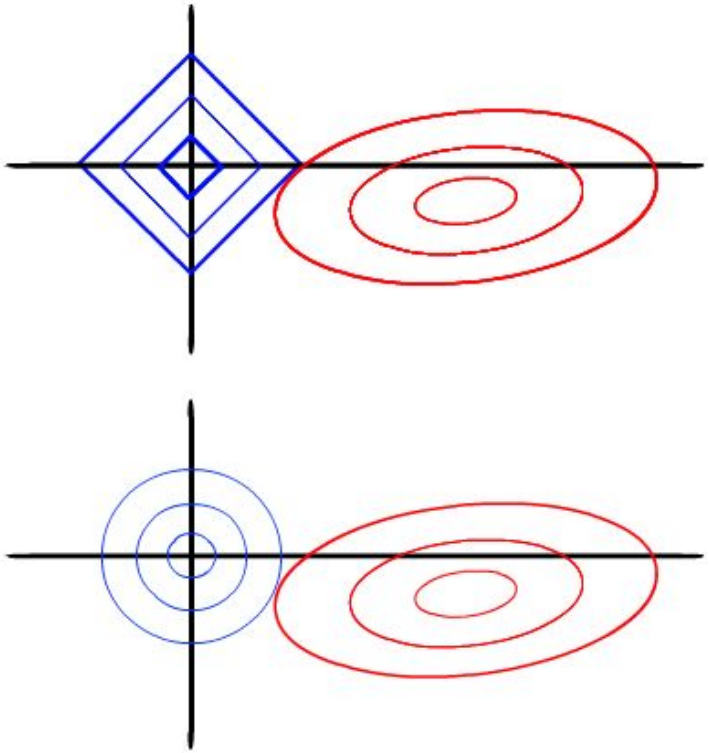
Coming up: CLASSIFICATION!

Unused Slides

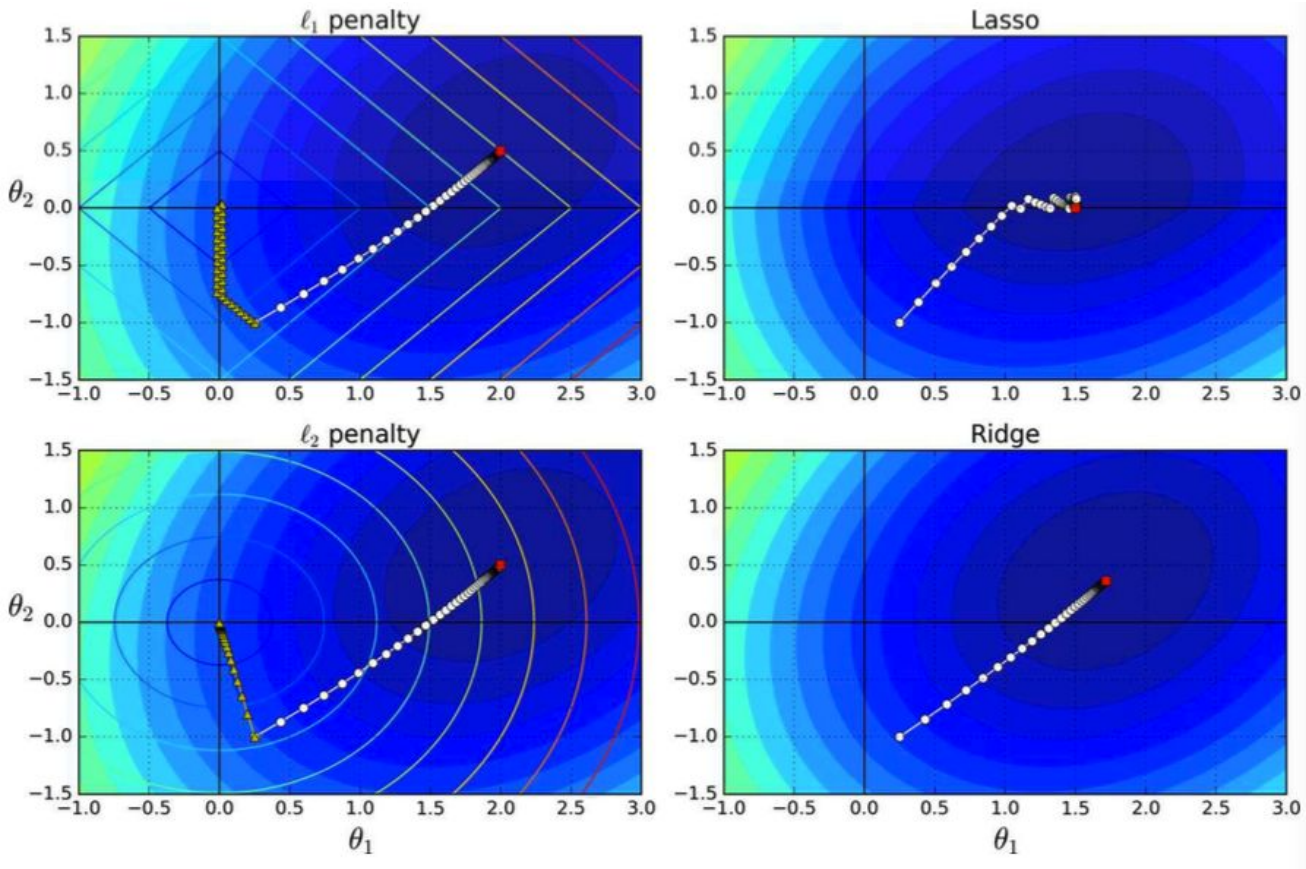
Geometry Interpretation



Geometry Interpretation (2)

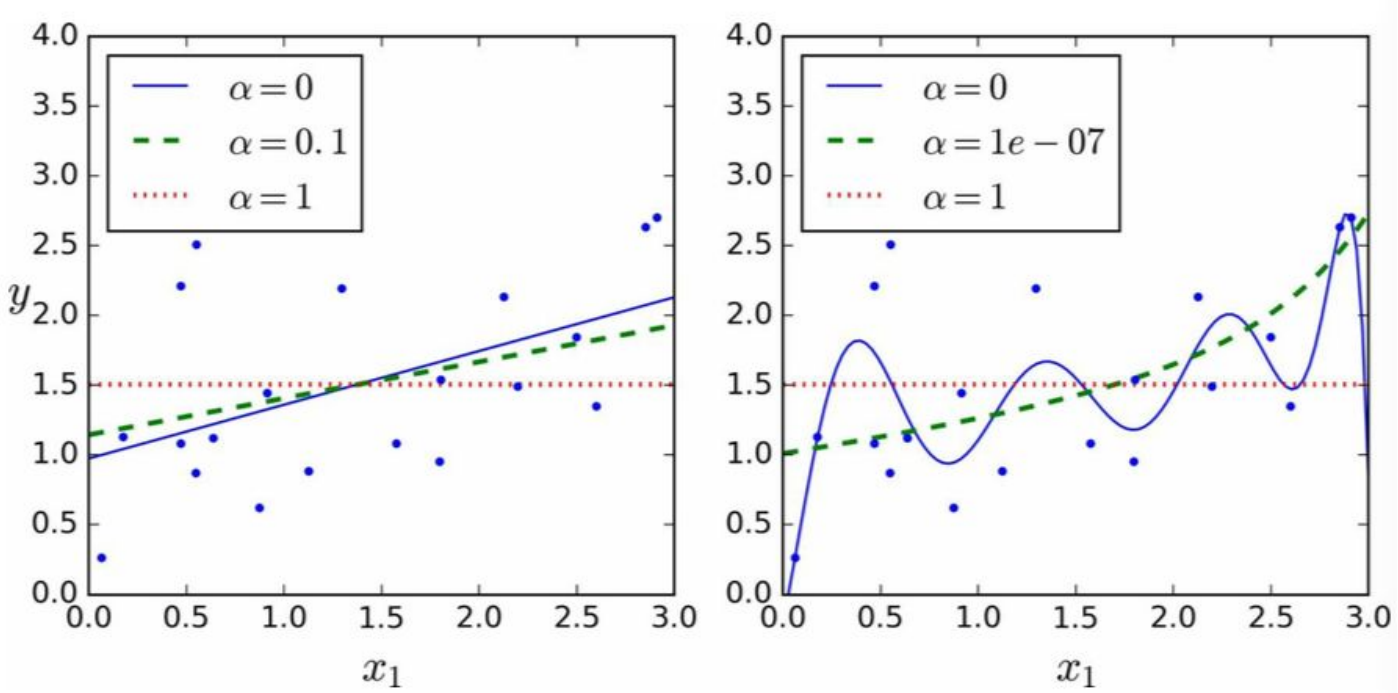


Geometry Interpretation (3)



Usage of hyperparameter

α should be λ



Can we utilize both Ridge and Lasso for regularization?

Lasso subgradient descent

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta), (j = 1...n)$$

$$\frac{\partial}{\partial \theta_j} J(\theta) = \frac{2}{m} \sum_{i=1}^{m} \left(h_\theta(\mathbf{x}^{(i)}) - y^{(i)} \right) x_j^{(i)} + \lambda \text{sign } (\theta_j)$$

$$g(\theta, J) = \nabla_\theta \text{MSE}(\theta) + \lambda \begin{pmatrix} \text{sign } (\theta_1) \\ \text{sign } (\theta_2) \\ \vdots \\ \text{sign } (\theta_n) \end{pmatrix} \quad \text{where sign } (\theta_i) = \begin{cases} -1 & \text{if } \theta_i < 0 \\ 0 & \text{if } \theta_i = 0 \\ +1 & \text{if } \theta_i > 0 \end{cases}$$