

# **Advanced Data Analysis**

**DATA 71200**

Class 8: Linear Models

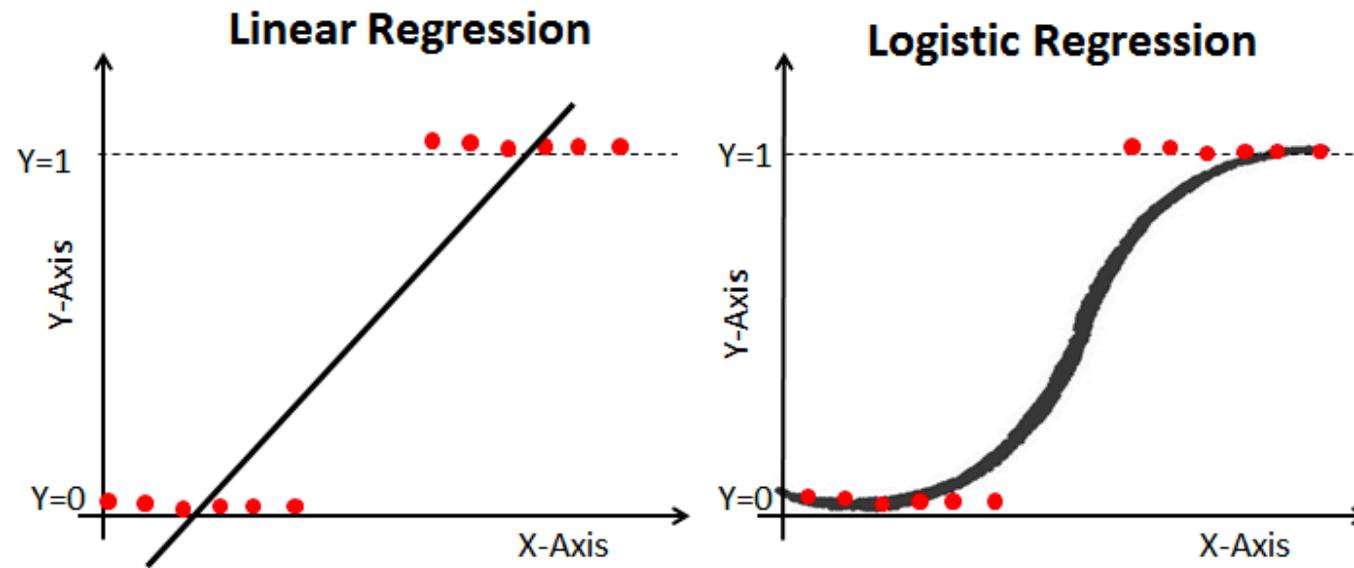
# Review from Class 5

## ► (Linear) Regression

- Continuous predictive model created by estimating a linear relationship between features and response

## ► Logistic Regression

- Predictive model of the probability of a certain class



# Linear Regression

## ► Dimensionality

- Line - 1 feature
- Plane - 2 features
- Hyperplane - more than 2 features

# Linear Regression

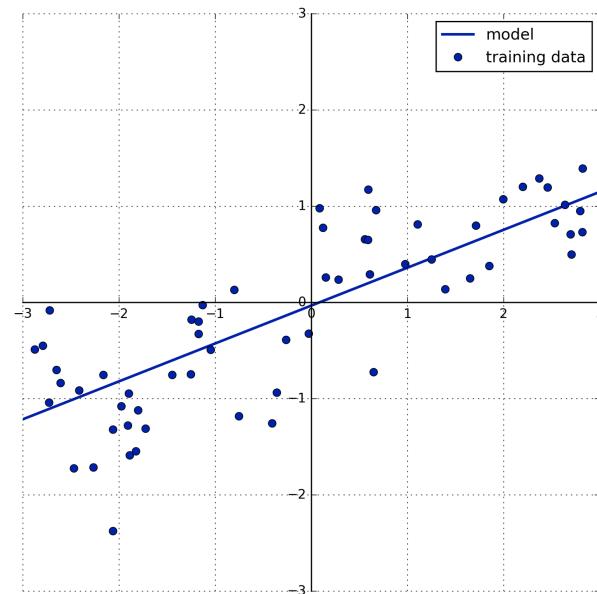
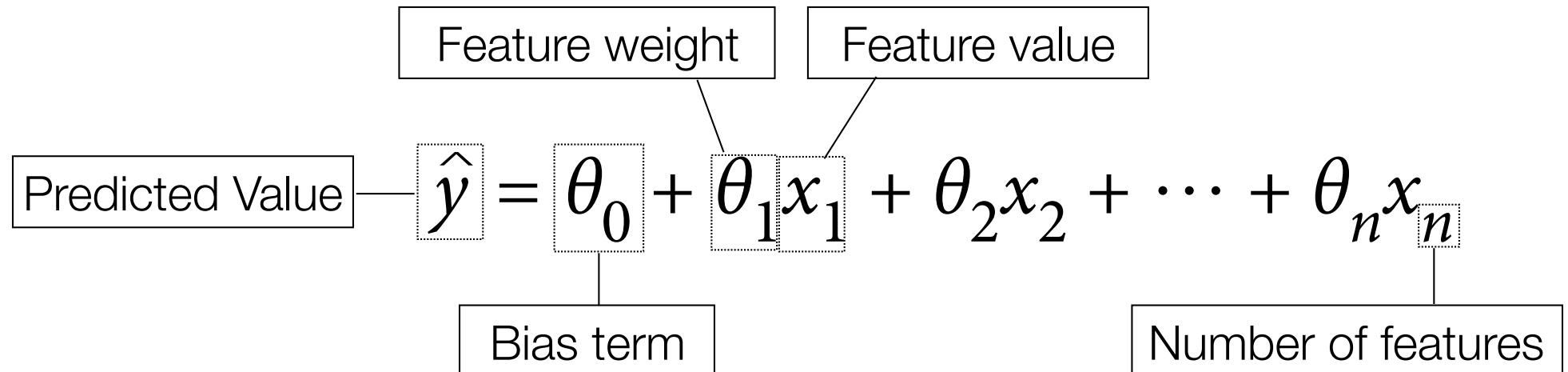
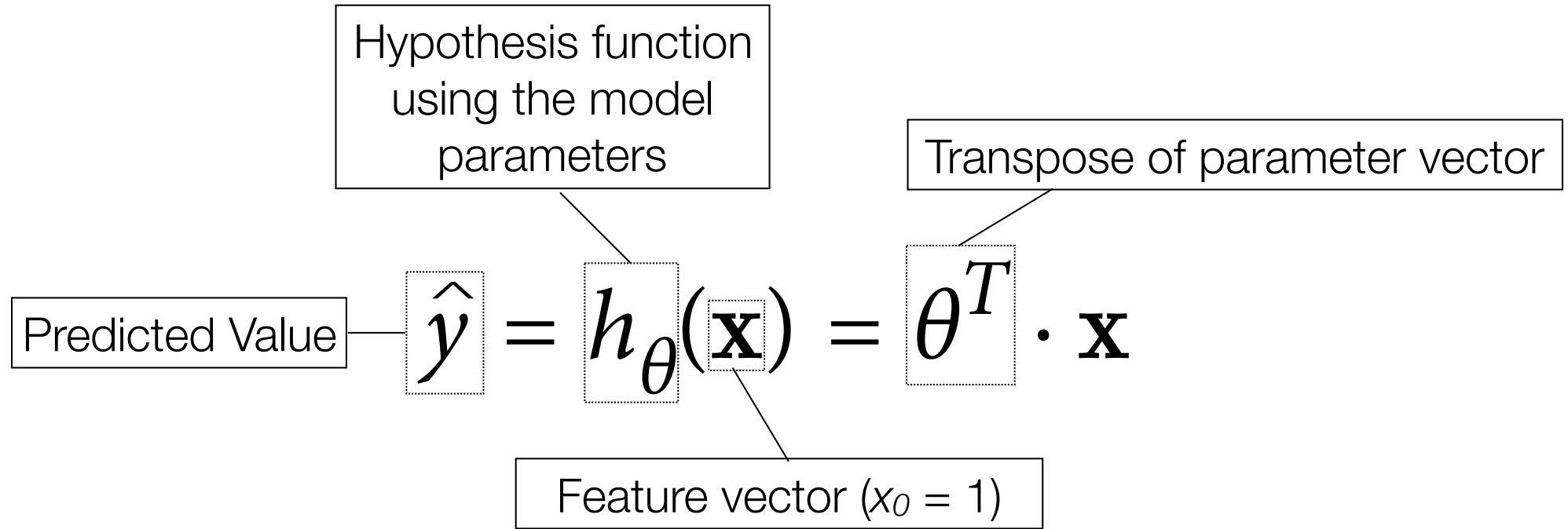


Figure 2-11. Predictions of a linear model on the wave dataset

# Linear Regression (Vectorized)



# The Bias/Variance Tradeoff

An important theoretical result of statistics and Machine Learning is the fact that a model's generalization error can be expressed as the sum of three very different errors:

## *Bias*

This part of the generalization error is due to wrong assumptions, such as assuming that the data is linear when it is actually quadratic. A high-bias model is most likely to underfit the training data.<sup>10</sup>

## *Variance*

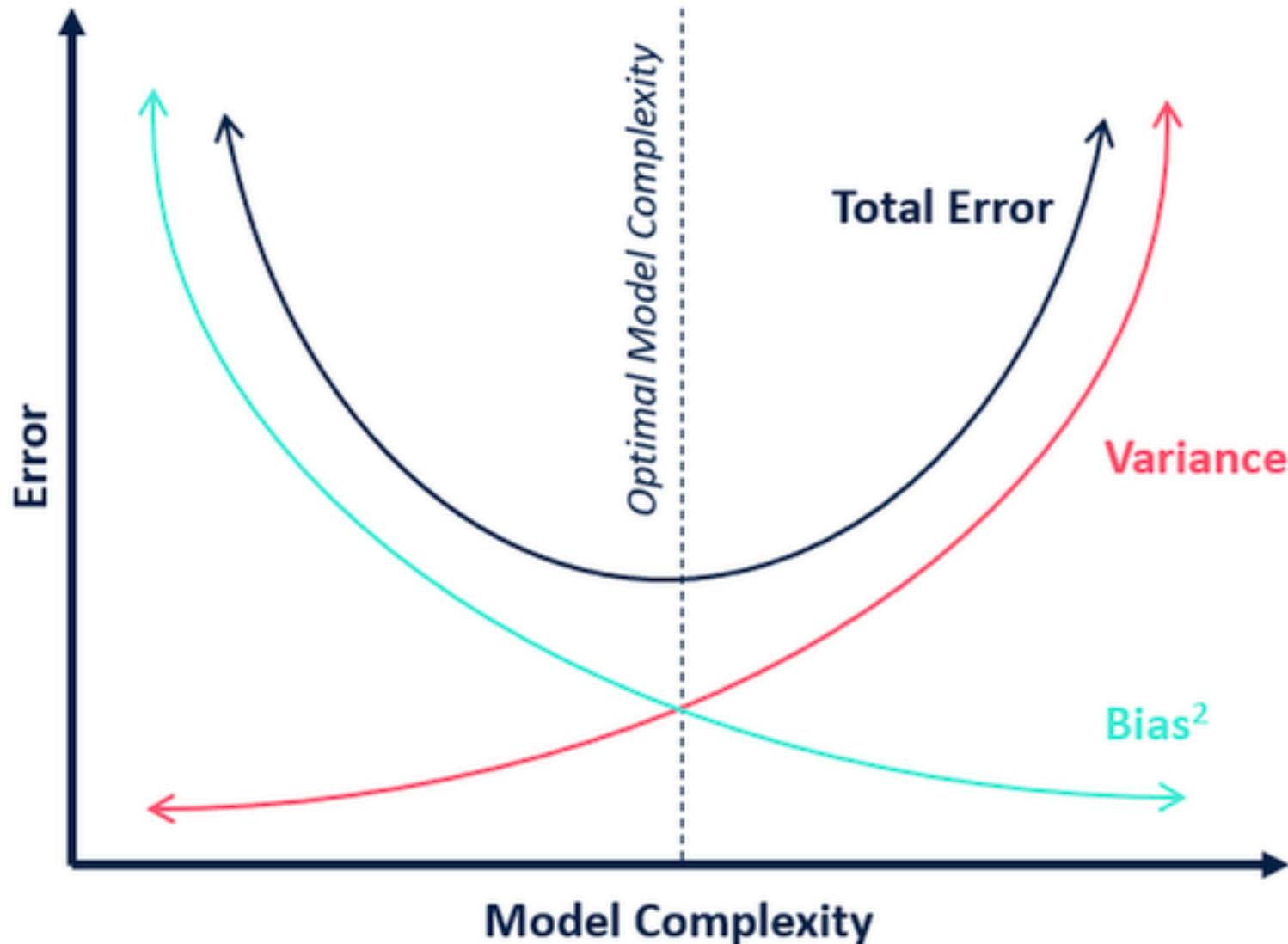
This part is due to the model's excessive sensitivity to small variations in the training data. A model with many degrees of freedom (such as a high-degree polynomial model) is likely to have high variance, and thus to overfit the training data.

## *Irreducible error*

This part is due to the noisiness of the data itself. The only way to reduce this part of the error is to clean up the data (e.g., fix the data sources, such as broken sensors, or detect and remove outliers).

Increasing a model's complexity will typically increase its variance and reduce its bias. Conversely, reducing a model's complexity increases its bias and reduces its variance. This is why it is called a tradeoff.

# Bias/Variance Tradeoff



# MSE Cost Function

*Equation 4-3. MSE cost function for a Linear Regression model*

$$\text{MSE}(\mathbf{X}, h_{\theta}) = \frac{1}{m} \sum_{i=1}^m (\theta^T \cdot \mathbf{x}^{(i)} - y^{(i)})^2$$

**X**

- $\mathbf{X}$  is a matrix containing all the feature values (excluding labels) of all instances in the dataset. There is one row per instance and the  $i^{th}$  row is equal to the transpose of  $\mathbf{x}^{(i)}$ , noted  $(\mathbf{x}^{(i)})^T$ .<sup>6</sup>
  - For example, if the first district is as just described, then the matrix  $\mathbf{X}$  looks like this:

$$\mathbf{X} = \begin{pmatrix} (\mathbf{x}^{(1)})^T \\ (\mathbf{x}^{(2)})^T \\ \vdots \\ (\mathbf{x}^{(1999)})^T \\ (\mathbf{x}^{(2000)})^T \end{pmatrix} = \begin{pmatrix} -118.29 & 33.91 & 1,416 & 38,372 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

**$h_{\theta}$**

- $h_{\theta}$  is the hypothesis function, using the model parameters  $\theta$ .

# MSE Cost Function

$$\text{MSE}(\mathbf{X}, h_{\theta}) = \frac{1}{m} \sum_{i=1}^m \left( \theta^T \cdot \mathbf{x}^{(i)} - y^{(i)} \right)^2$$

Number of data instances

Transpose of parameter vector

$i^{th}$  instance in vector of labels

$i^{th}$  instance in vector of feature

Also referred to as  
Ordinary Least Squares

Jupyter Notebook  
02-supervised-learning  
.ipynb [25-29]

# Ridge Regression Cost Function

$$J(\theta) = \text{MSE}(\theta) + \alpha \frac{1}{2} \sum_{i=1}^n \theta_i^2$$

Cost function

Tunable parameter

Number of features (dimensions)

$$\text{MSE}(\mathbf{X}, h_\theta) = \frac{1}{m} \sum_{i=1}^m (\theta^T \cdot \mathbf{x}^{(i)} - y^{(i)})^2$$

Jupyter Notebook  
02-supervised-learning  
.ipynb [30-32]

# Ridge Regression

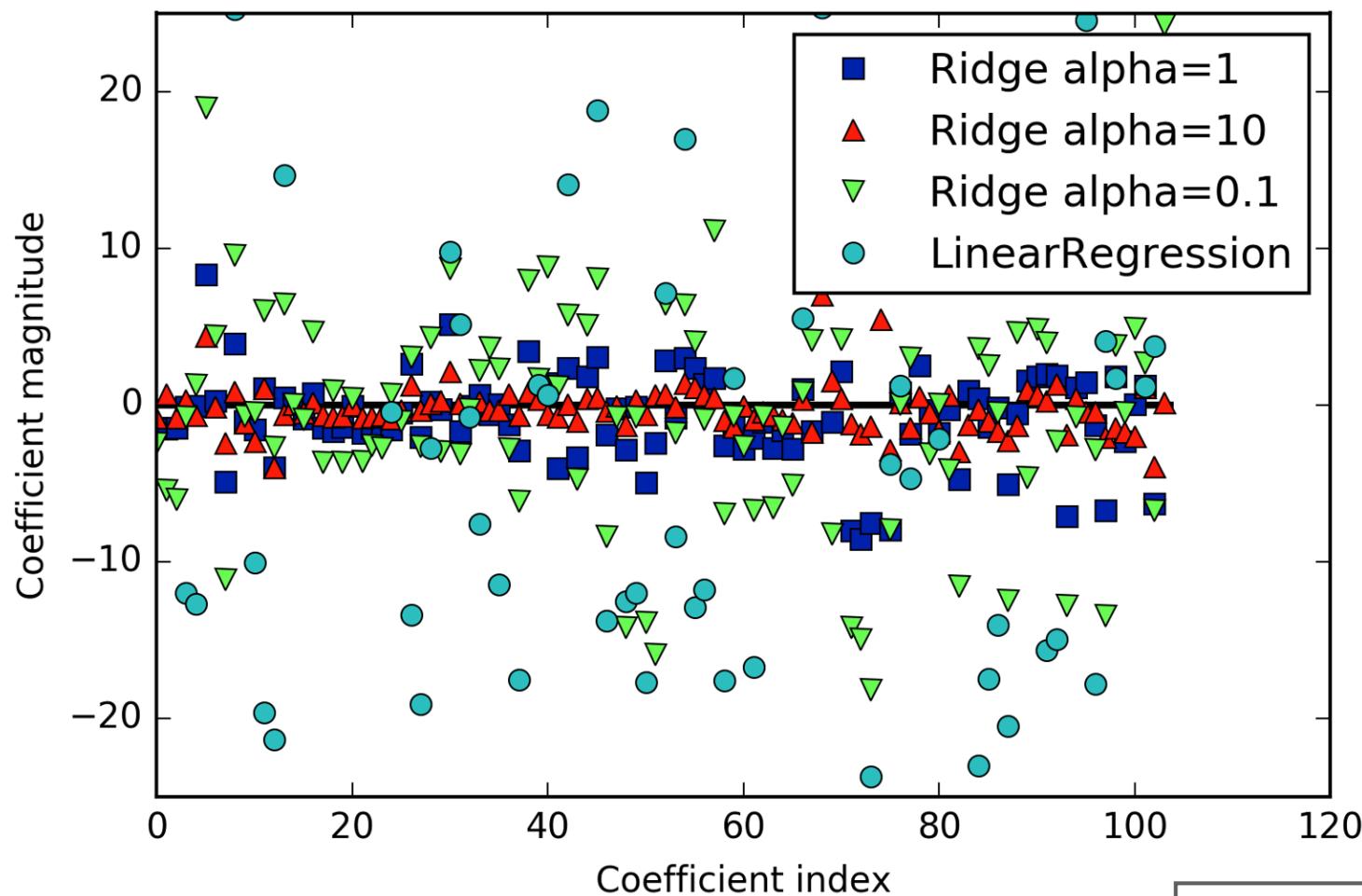


Figure 2-12. Comparing coefficient magnitudes for ridge regression with different values of alpha and linear regression

Jupyter Notebook  
02-supervised-learning  
.ipynb [33]

# Ridge Regression

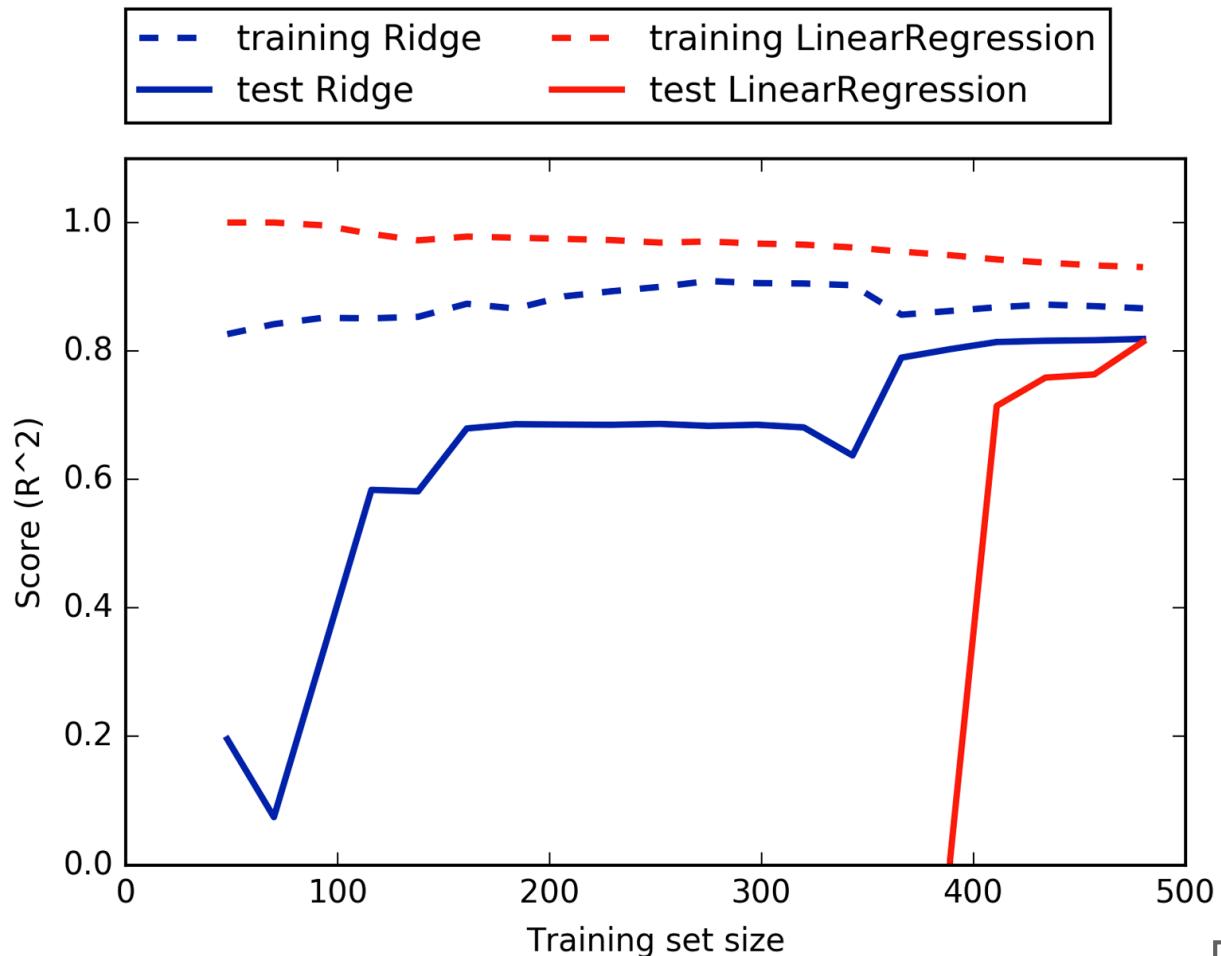


Figure 2-13. Learning curves for ridge regression and linear regression on the Boston Housing dataset

Jupyter Notebook  
02-supervised-learning  
.ipynb [34]

# Lasso Regression Cost Function

$$J(\theta) = \text{MSE}(\theta) + \alpha \sum_{i=1}^n |\theta_i|$$

MSE( $\mathbf{X}, h_\theta$ ) =  $\frac{1}{m} \sum_{i=1}^m (\theta^T \cdot \mathbf{x}^{(i)} - y^{(i)})^2$

Tunable parameter

Number of features (dimensions)

Recall the Ridge Equation

$$J(\theta) = \text{MSE}(\theta) + \alpha \frac{1}{2} \sum_{i=1}^n \theta_i^2$$

Jupyter Notebook  
02-supervised-learning  
.ipynb [35-37]

# Lasso versus Ridge Regression

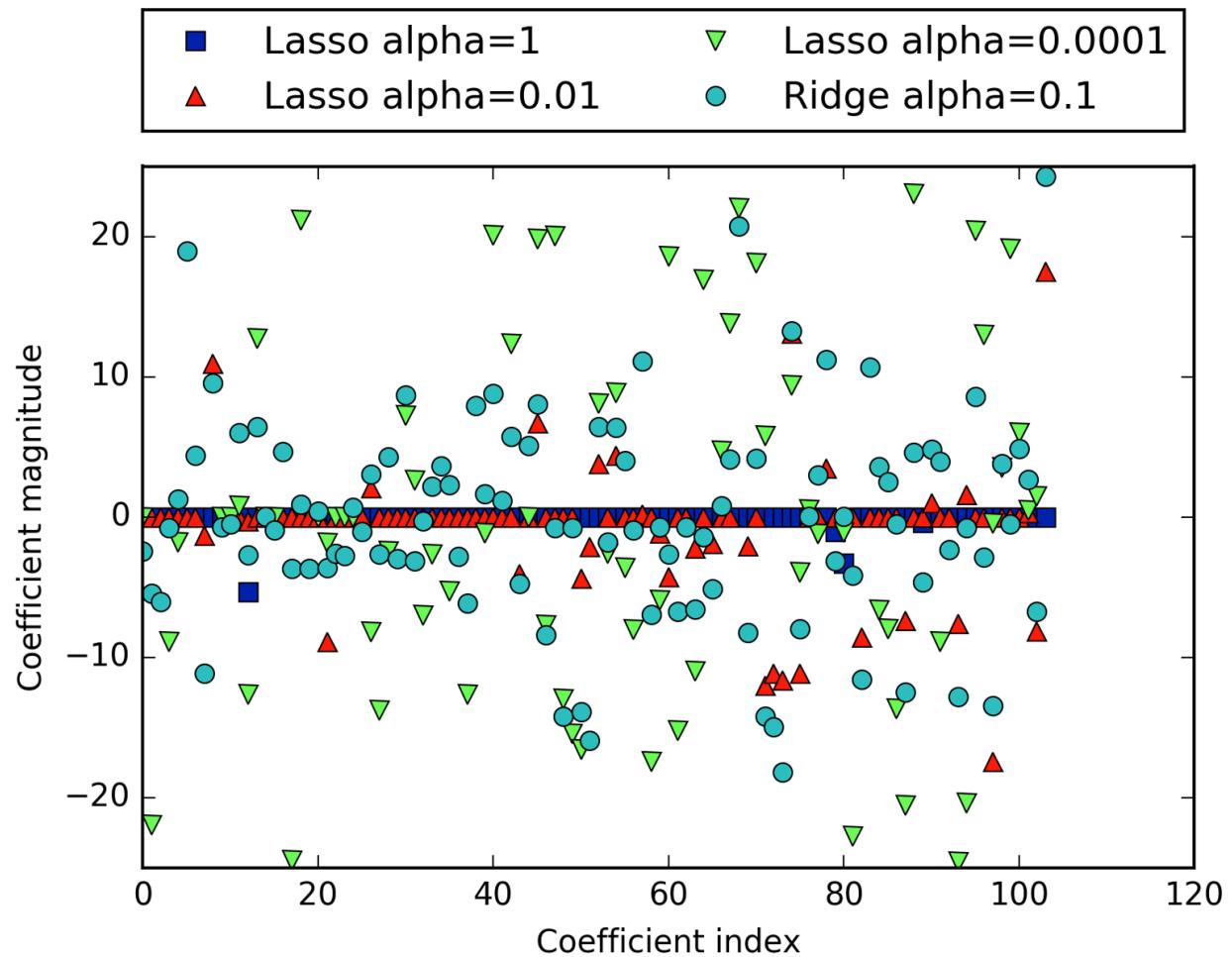
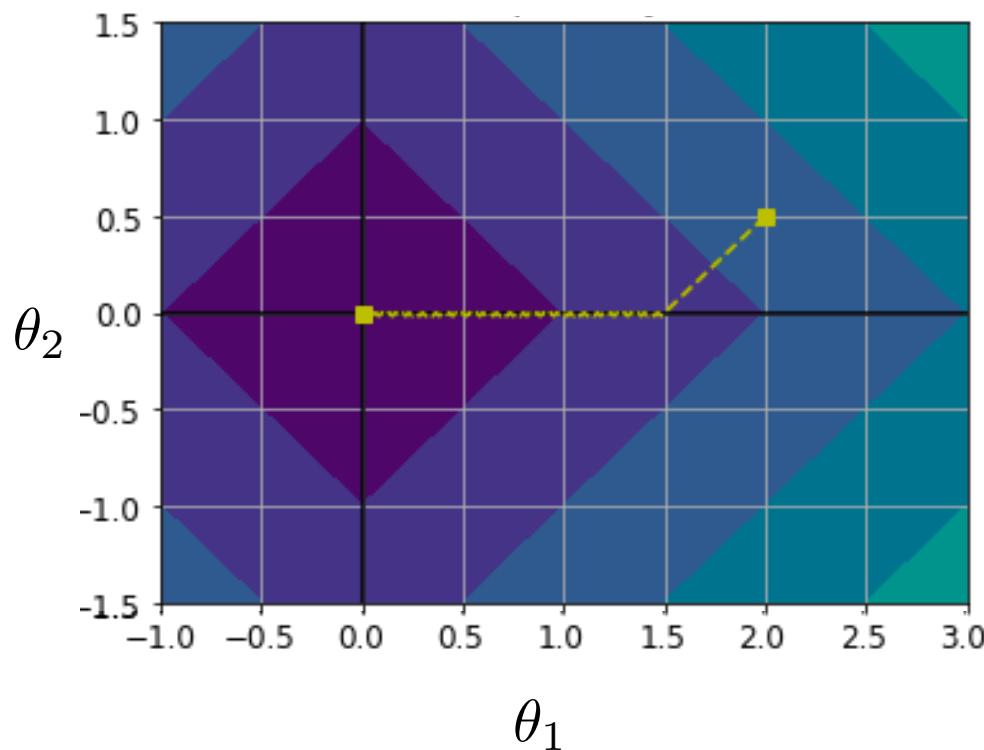


Figure 2-14. Comparing coefficient magnitudes for lasso regression with different values of alpha and ridge regression

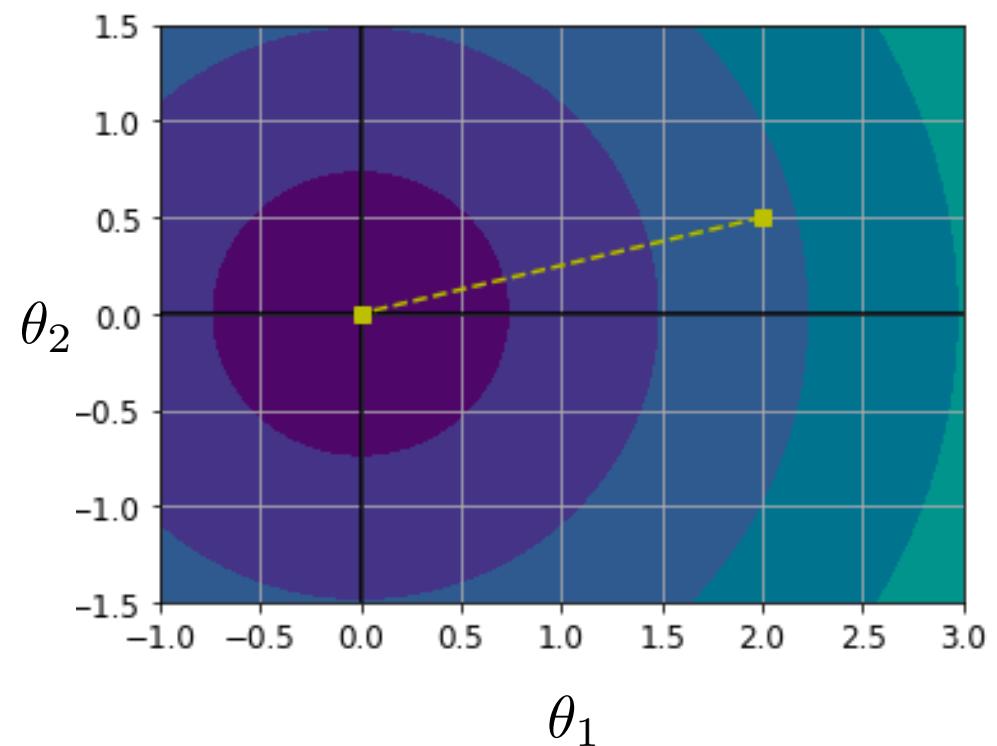
Jupyter Notebook  
02-supervised-learning  
.ipynb [38]

# Lasso versus Ridge Regression

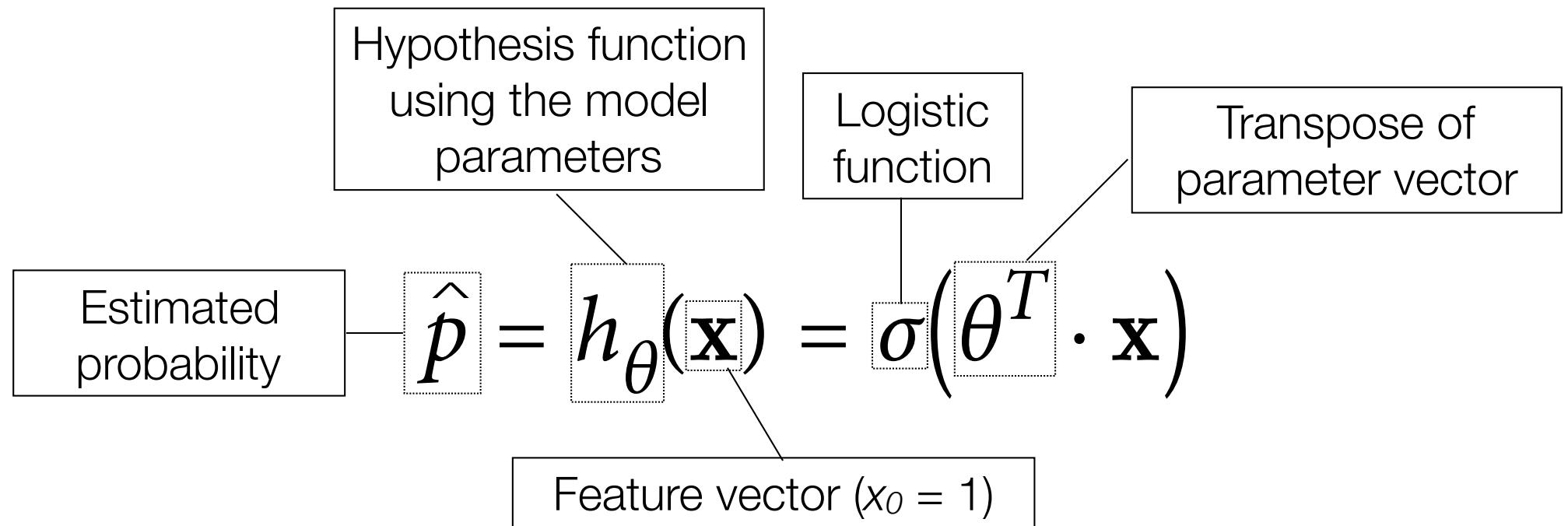
$\ell_1$  penalty (Lasso)



$\ell_2$  penalty (Ridge)



# Logistic Regression (Vectorized)



## Logistic Function

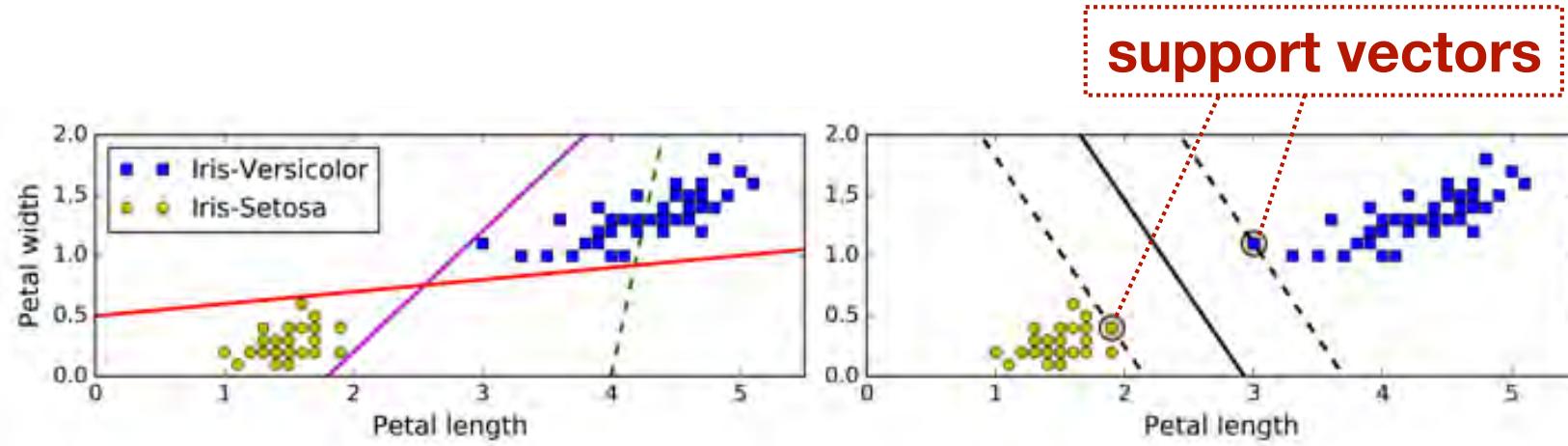
$$\sigma(t) = \frac{1}{1 + \exp(-t)}$$

## Predicted Class (binary)

$$\hat{y} = \begin{cases} 0 & \text{if } \hat{p} < 0.5, \\ 1 & \text{if } \hat{p} \geq 0.5. \end{cases}$$

# Linear Support Vector Classifier

- ▶ Find the linear classifier with the best separation (margin) between the two classes
  - Operationally the data points used to make this calculation are the support vectors



Three possible linear classifiers

Linear classifier with the widest margin

# Logistic Reg versus Linear SVC

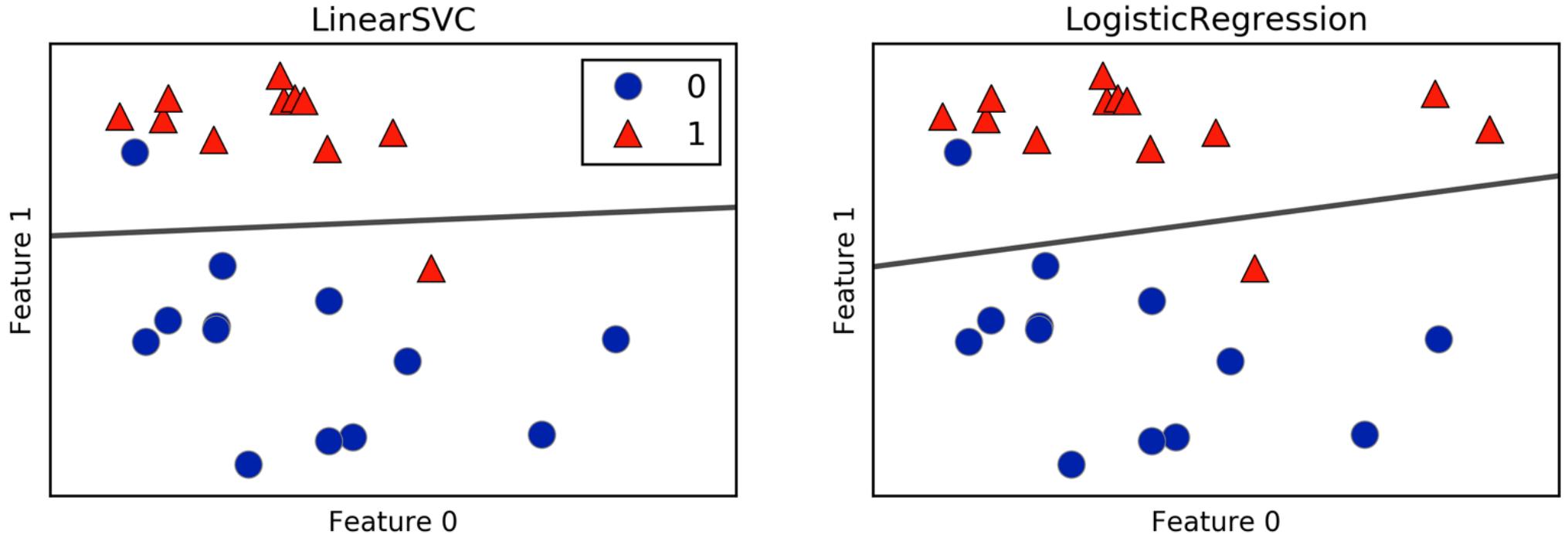


Figure 2-15. Decision boundaries of a linear SVM and logistic regression on the forge dataset with the default parameters

Jupyter Notebook  
02-supervised-learning  
.ipynb [39]

# Coefficient Magnitude

- ▶ inverse of regularization strength

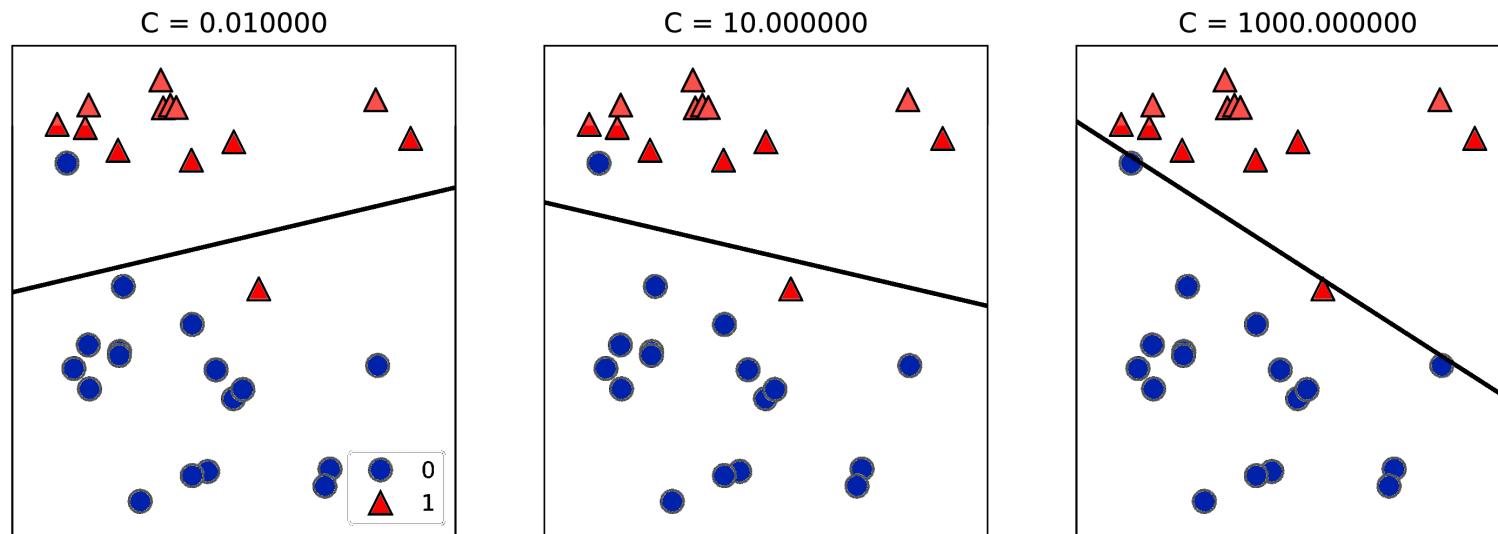


Figure 2-16. Decision boundaries of a linear SVM on the forge dataset for different values of  $C$

Jupyter Notebook  
02-supervised-learning  
.ipynb [40]

# Coefficient Magnitude

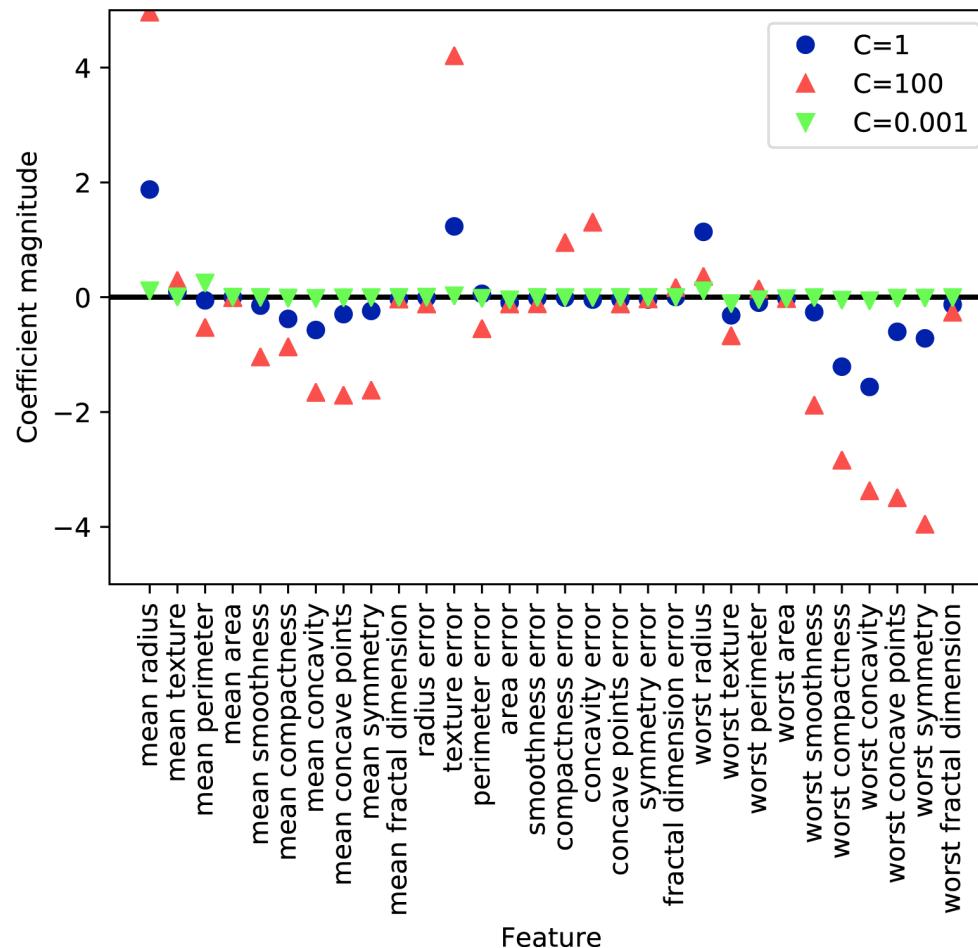


Figure 2-17. Coefficients learned by logistic regression on the Breast Cancer dataset for different values of  $C$

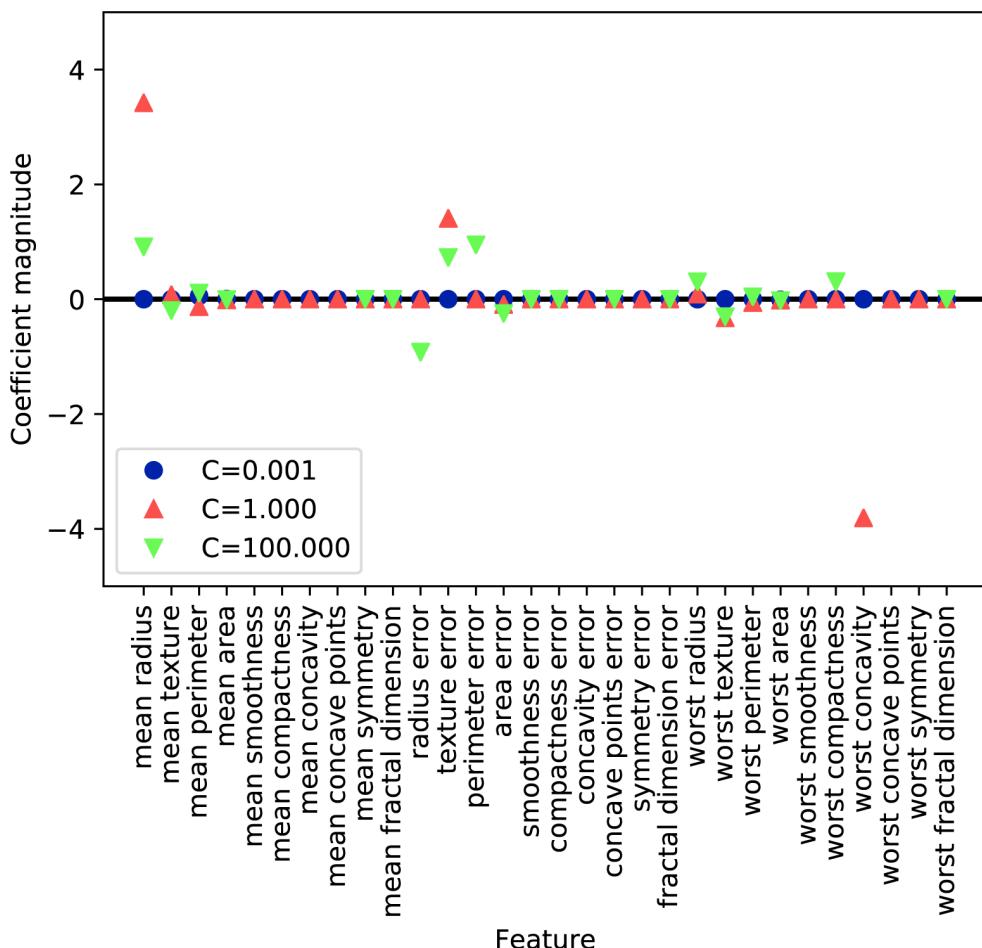


Figure 2-18. Coefficients learned by logistic regression with  $L1$  penalty on the Breast Cancer dataset for different values of  $C$

Jupyter Notebook 02-supervised-learning.ipynb [41-45]

# One versus Rest Classification

- ▶ Find the linear classifier that facilitates the best separation (margin) between the two classes

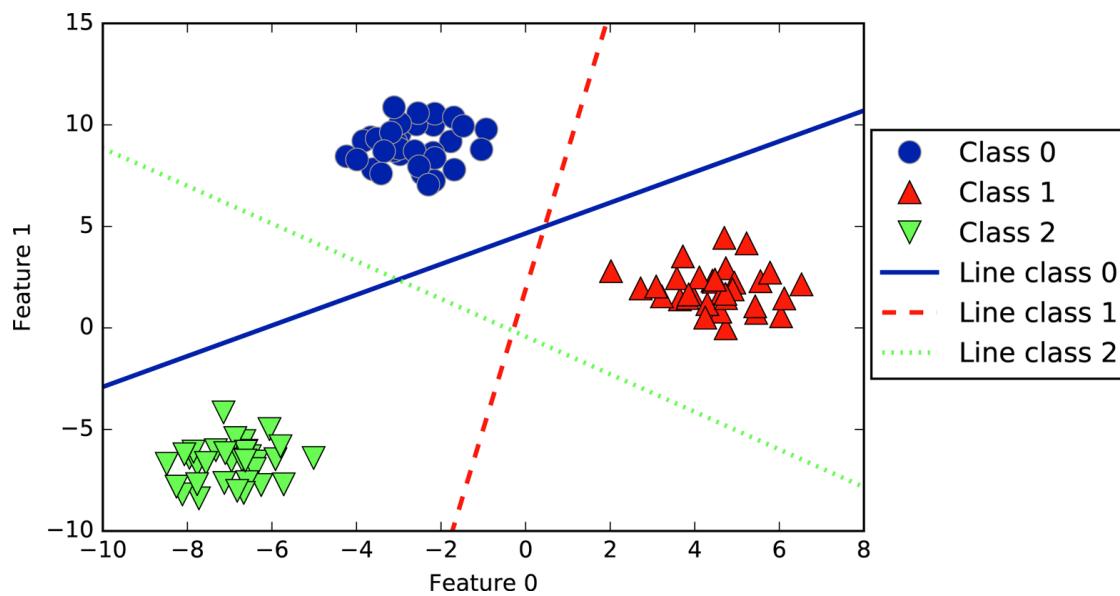


Figure 2-20. Decision boundaries learned by the three one-vs.-rest classifiers

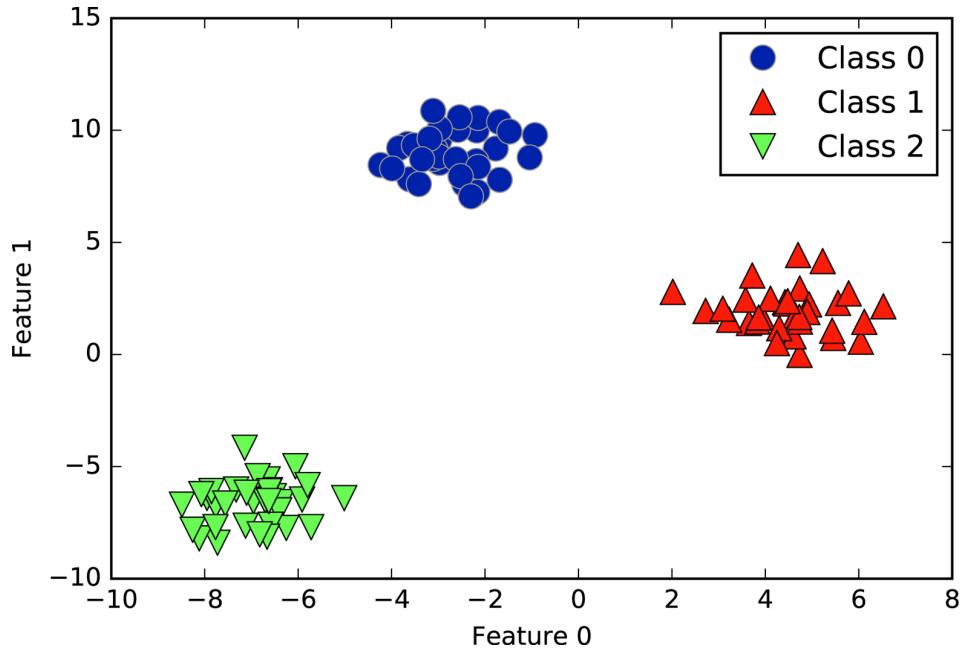


Figure 2-19. Two-dimensional toy dataset containing three classes

Jupyter Notebook  
02-supervised-learning  
.ipynb [46-48]

# Logistic Reg versus Linear SVC

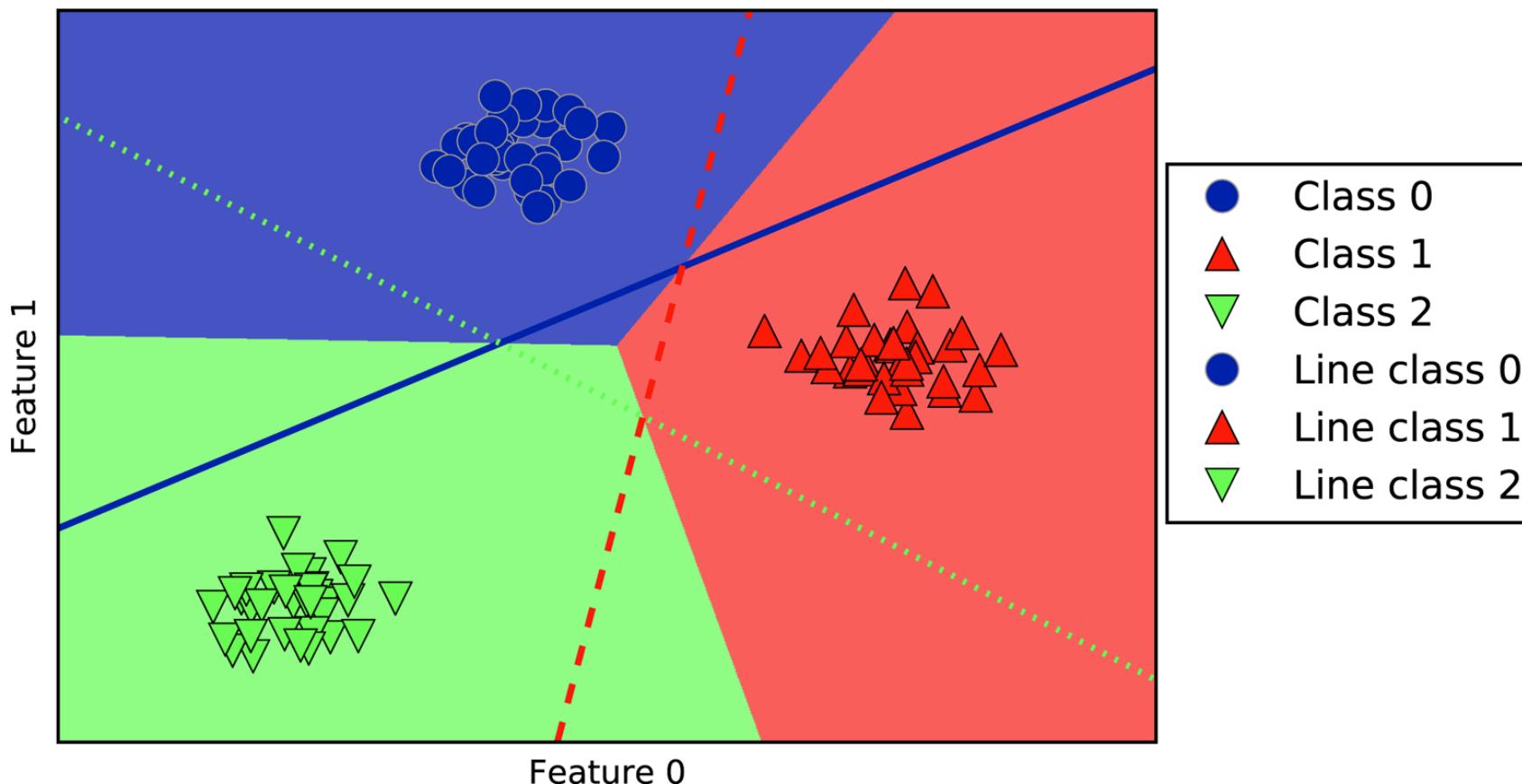


Figure 2-21. Multiclass decision boundaries derived from the three one-vs.-rest classifiers

Jupyter Notebook  
02-supervised-learning  
.ipynb [49]

# Linear Models Overview

## ▶ Assumptions

- Linear relationship between input and output
- No noise in input or output
- Independence (lack of correlation in input)
- Gaussian (normally) distributed data

## ▶ Best Practices

- Scale inputs

# Linear Models Overview

## ► **Parameters**

- Regularization parameter: alpha (regression) or C (classification)
- Type of regularization (L1 or L2)

## ► **Strengths**

- Fast to train (good for large datasets)
- Fast at prediction
- Easy to understand the algorithm
- Work well even when there are a large number of features compared to the number of samples

## ► **Weaknesses**

- Not so easy to interpret a model's coefficients
- Generalization may be poor with small number of features