Lecture 6
Support Vector Machines

GEOL 4397: Data analytics and machine learning for geoscientists

Jiajia Sun, Ph.D.

Feb. 12th, 2019

UNIVERSITY of HOUSTON
YOU ARE THE PRIDE
EARTH AND ATMOSPHERIC SCIENCES

Agenda

- Regularization
- Support Vector Machine
- Nonlinear SVM
- Implementation in Scikit-Learn

Training data

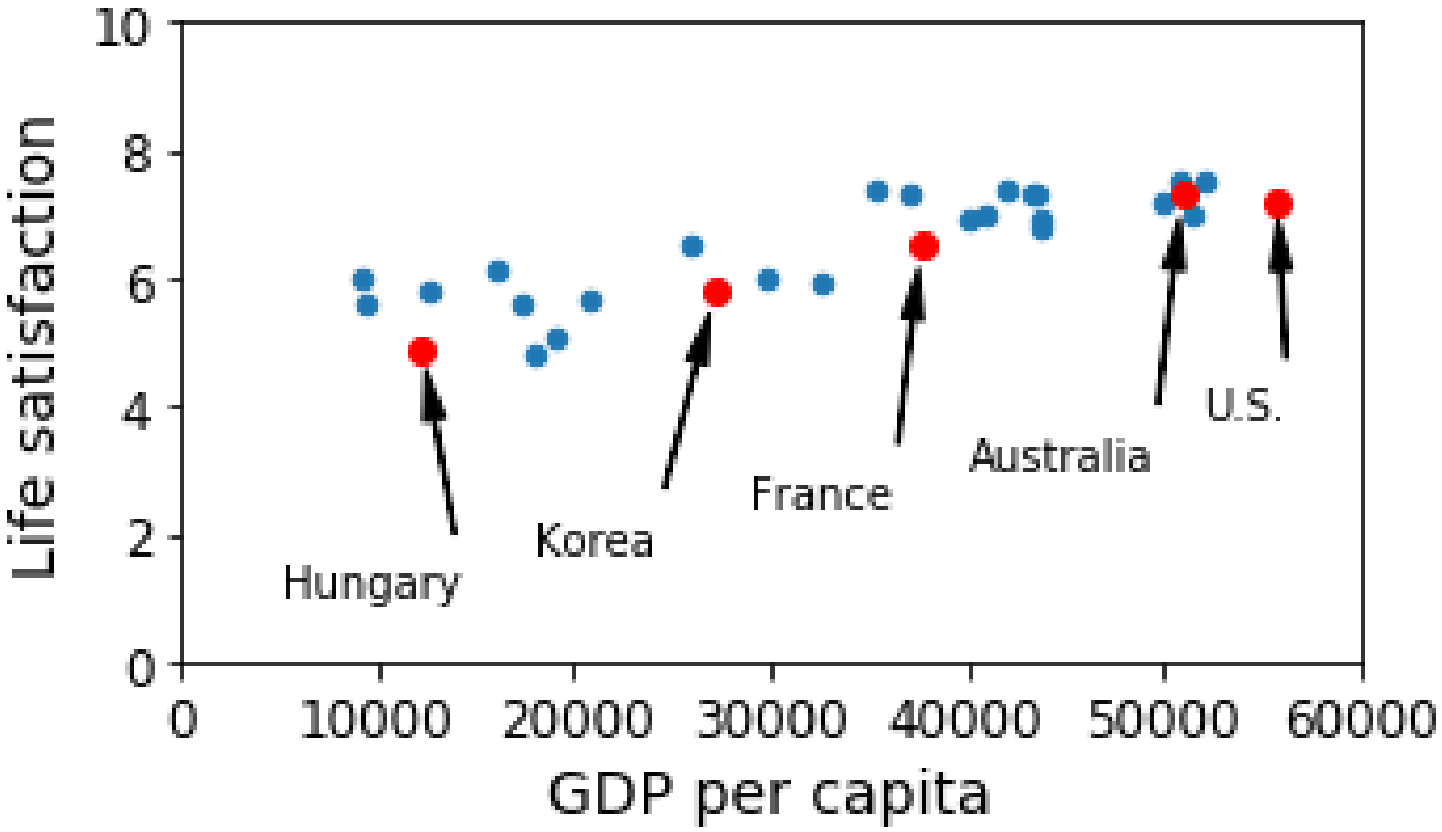


Figure from Aurelien Geron's ML book, page 19

Good fit

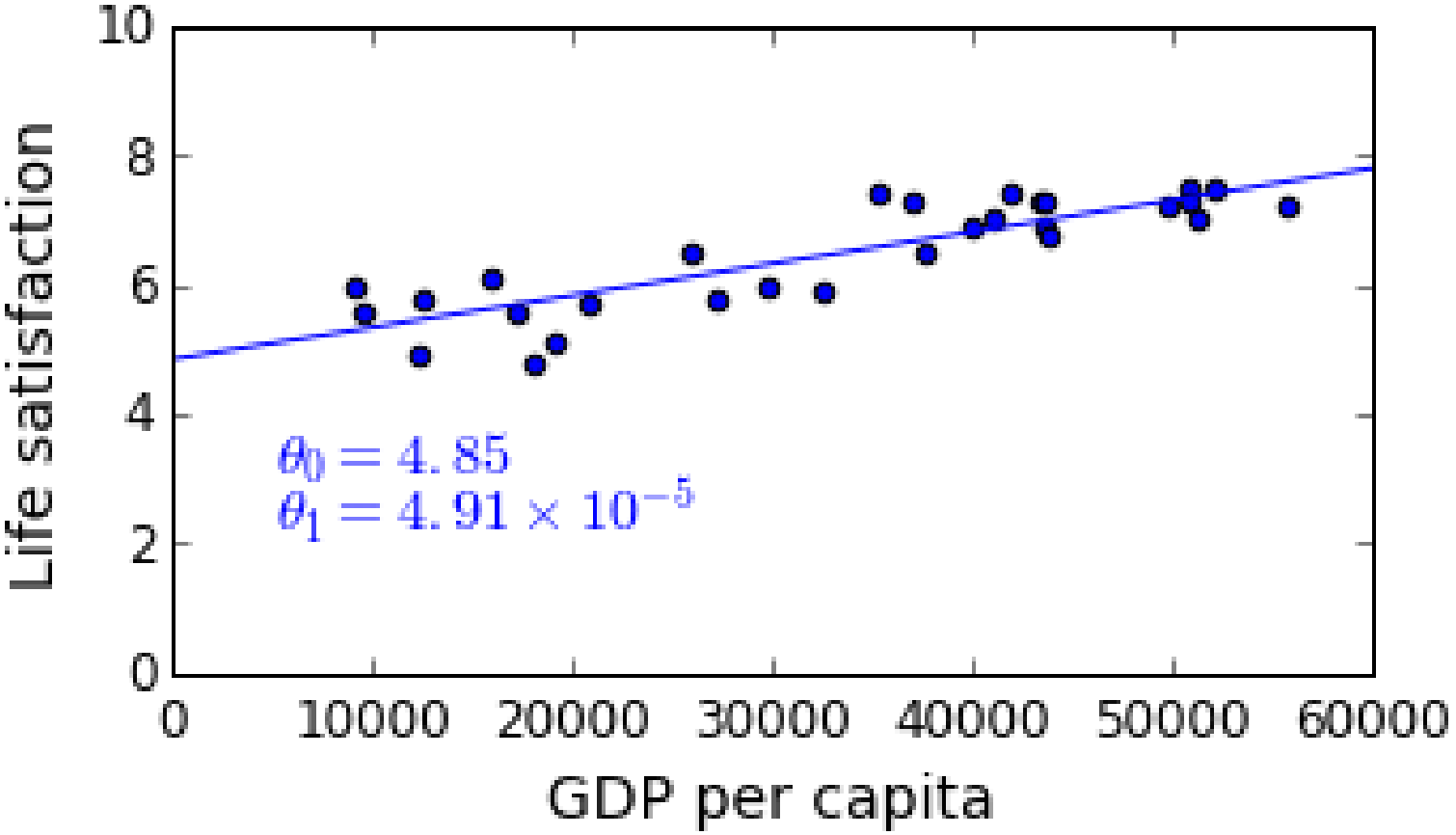


Figure from Aurelien Geron's ML book, page 20

Overfit (polynomial degree = 60)

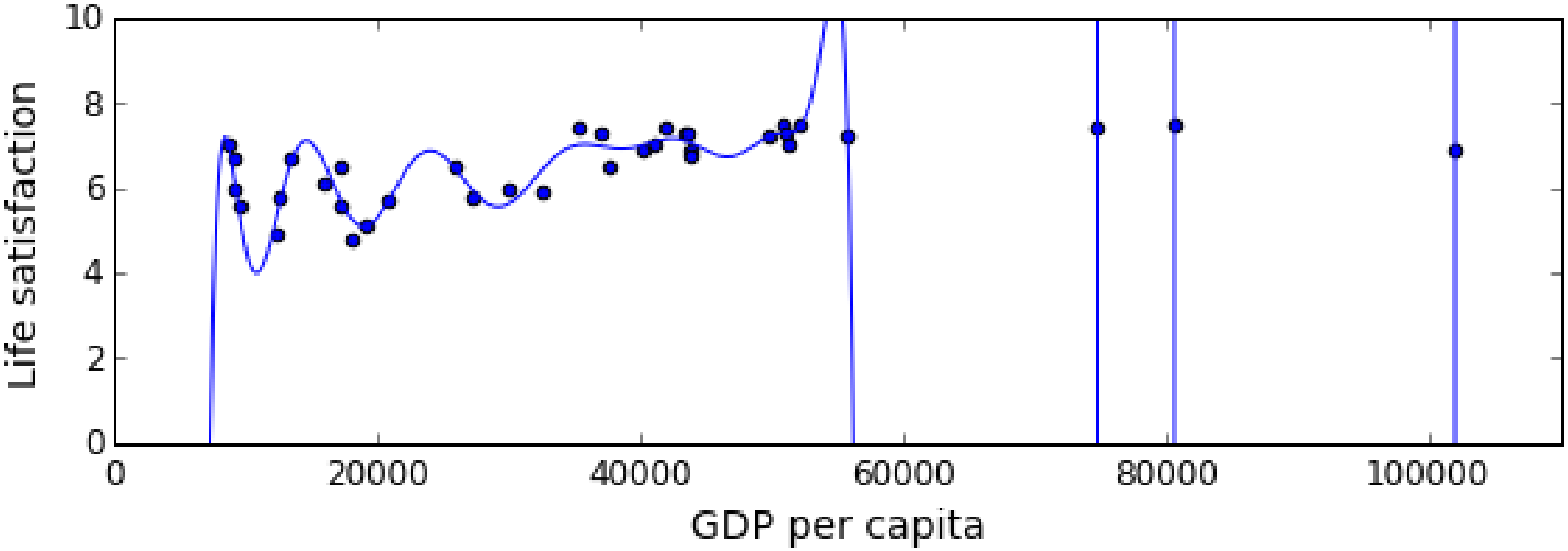


Figure from Aurelien Geron's ML book, page 26

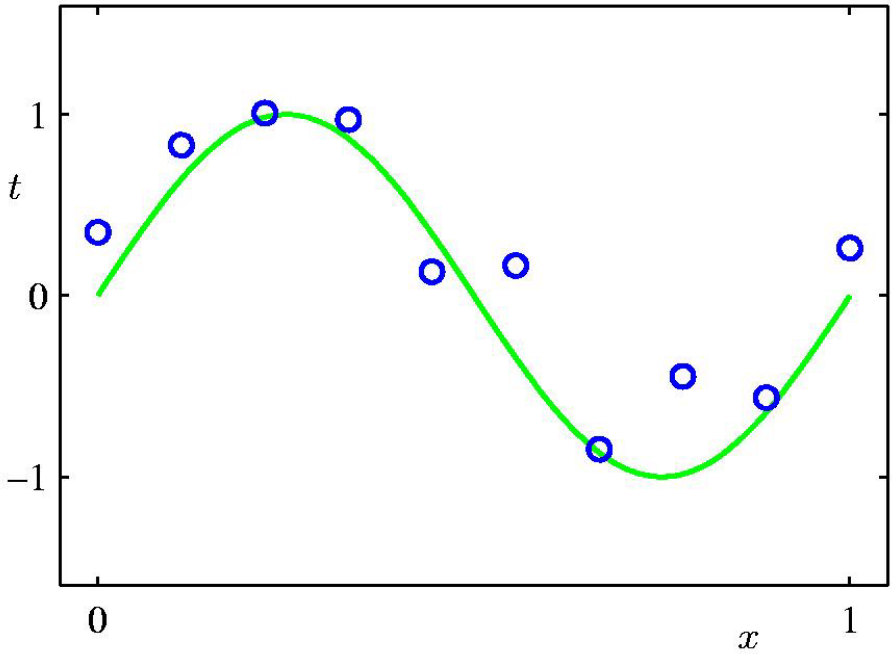
Remedy for overfitting

- Overfitting happens when your ML model is overly complex
- Therefore, possible solutions are:
  1. Collect more training data
  2. Reduce data noise
  3. Simplify model
     - using linear model rather than a high-degree polynomial model
     - using regularization
     - …

Remedy for overfitting

- Overfitting happens when your ML model is overly complex
- Therefore, possible solutions are:
  1. Collect more training data
  2. Reduce data noise
  3. Simplify model
     - using linear model rather than a high-degree polynomial model
     - using regularization
     - …

Training data set of M = 10 points, each comprising an observation of input variable x and the corresponding target variable t.



The green curve shows the function $\sin(2\pi x)$ used to generate the data.

Goal: predict the value of t for some new value of x, based on the model learned from training data.

Polynomial curve fitting

Fit the data using a polynomial function of the form:

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_N x^N$$

where N is the degree of the polynomial

Polynomial curve fitting

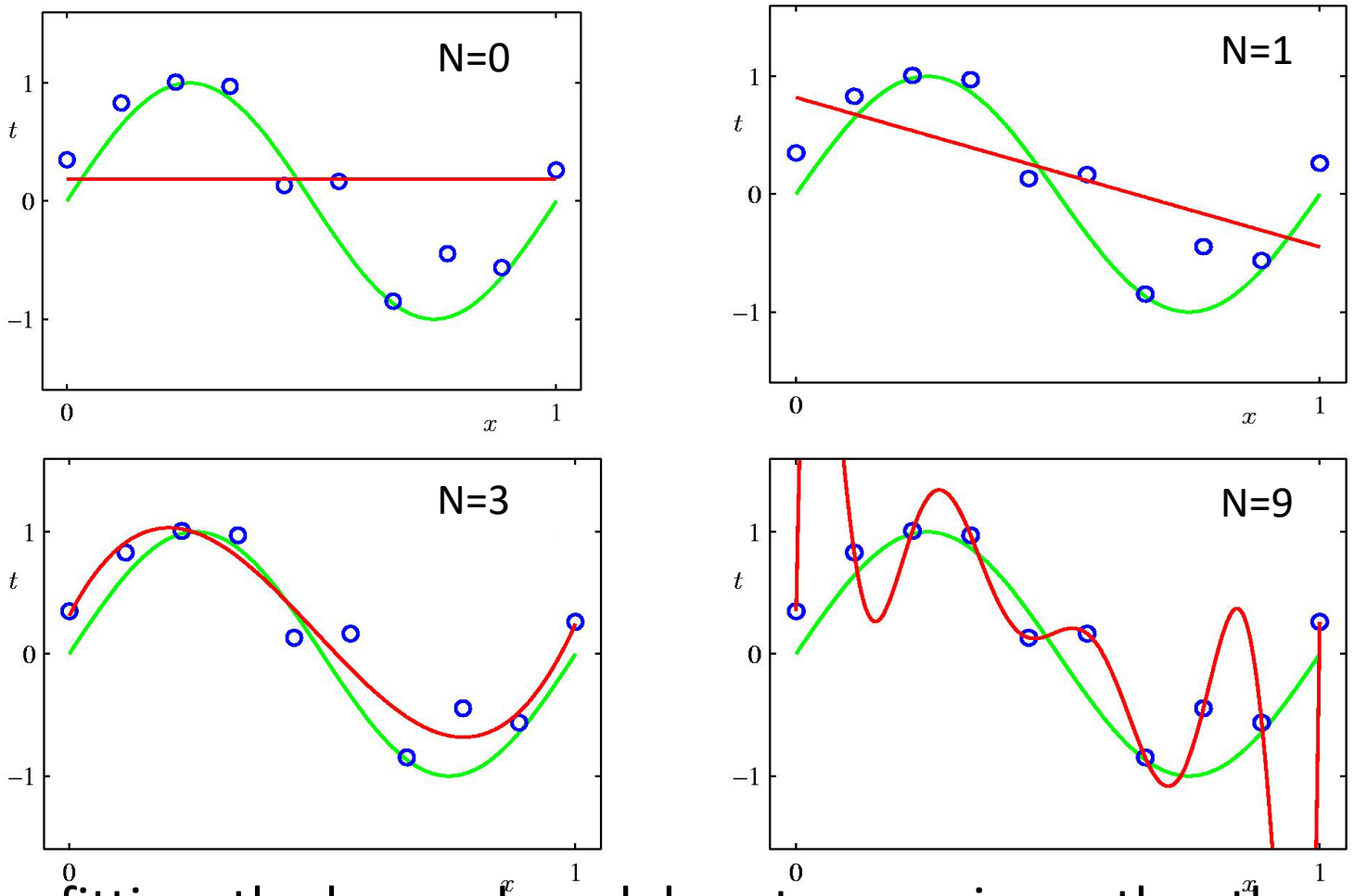
Fit the data using a polynomial function of the form:

$$h_\theta(x) = \theta_0 + \theta_1 x + \theta_2 x^2 + \cdots + \theta_N x^N$$

where N is the degree of the polynomial

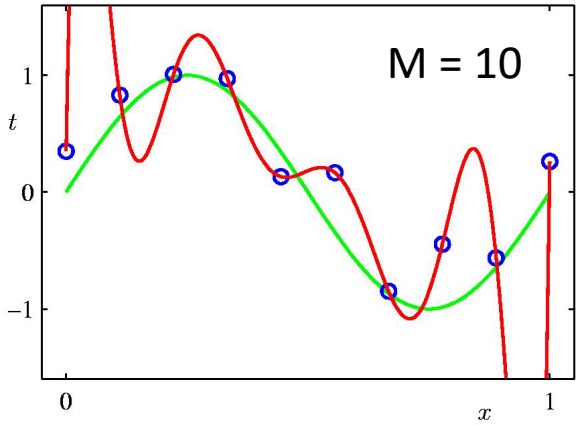
Question: how to select N?

Training models with N = 0, 1, 3, 9



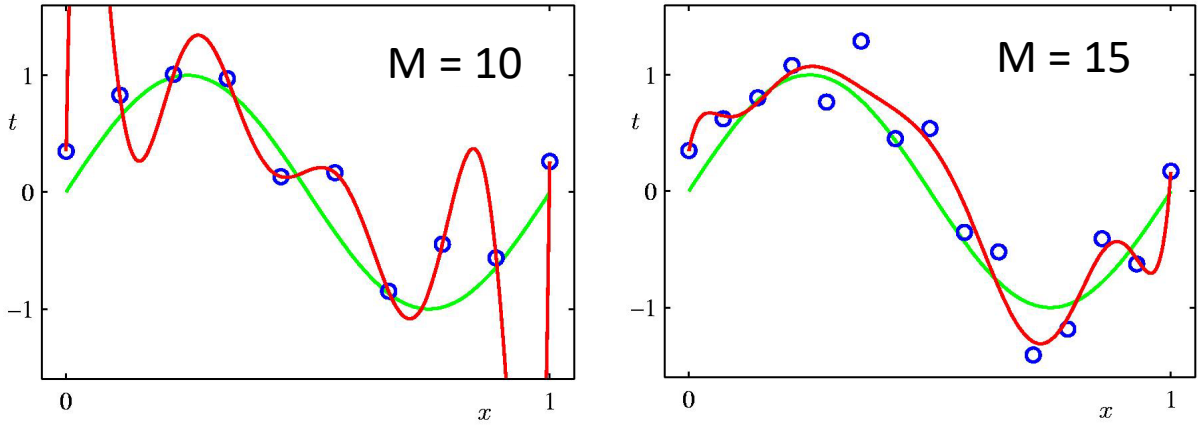
Overfitting: the learned model captures noise, rather than the true and meaningful features/trends among the training data.

More training data



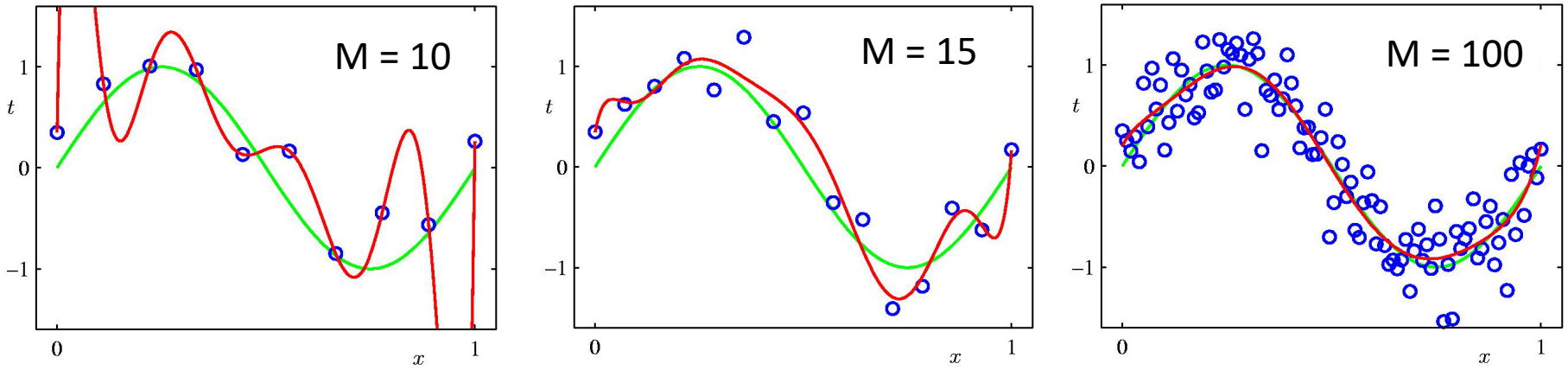
Learned models with N = 9 for M = 10

More training data



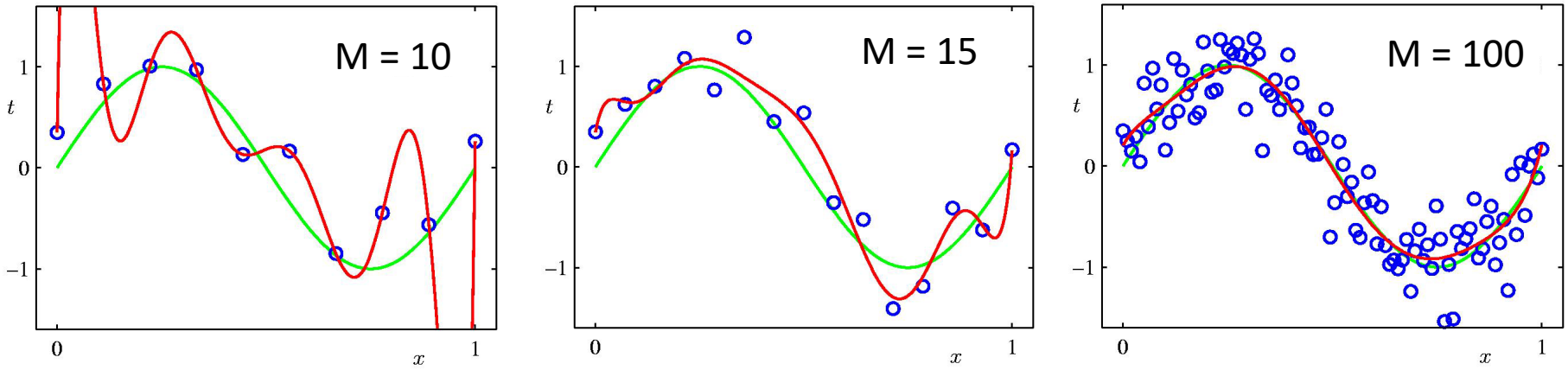
Learned models with N = 9 for M = 10, 15

More training data



Learned models with N = 9 for M = 10, 15, and 100.

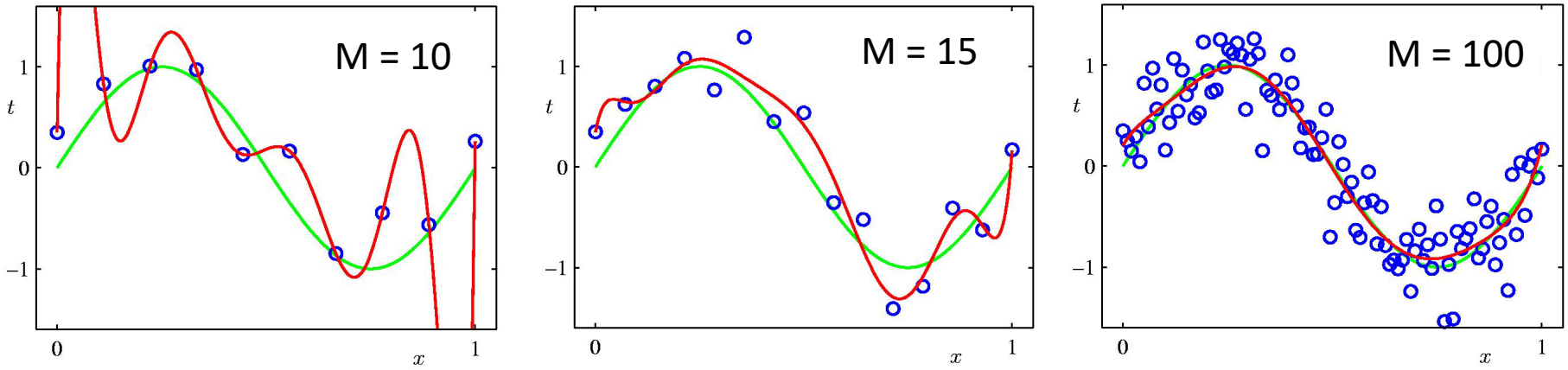
More training data



Learned models with N = 9 for M = 10, 15, and 100.

Increasing the size of training data set helps decrease the overfitting.

More training data



Learned models with N = 9 for M = 10, 15, and 100.

Increasing the size of training data set helps decrease the overfitting.

However, obtaining more training data is not always doable in reality.

Remedy for overfitting

- Overfitting happens when your ML model is overly complex
- Therefore, possible solutions are:
  1. Collect more training data
  2. Reduce data noise
  3. Simplify model
     - using linear model rather than a high-degree polynomial model
     - using regularization
     - …

Remedy for overfitting

- Overfitting happens when your ML model is overly complex
- Therefore, possible solutions are:
  1. Collect more training data
  2. Reduce data noise — Not always feasible either!
  3. Simplify model
     - using linear model rather than a high-degree polynomial model
     - using regularization
     - …

Remedy for overfitting

- Overfitting happens when your ML model is overly complex
- Therefore, possible solutions are:
  1. Collect more training data
  2. Reduce data noise — Not always feasible either!
  3. Simplify model
     - using linear model rather than a high-degree polynomial model
     - using regularization
     - …

	N = 0	N = 1	N = 3	N = 9
θ_0	0.19	0.82	0.31	0.35
θ_1		-1.27	7.99	232.37
θ_2			-25.43	-5321.83
θ_3			17.37	48568.31
θ_4				-231639.30
θ_5				640042.26
θ_6				-1061800.52
θ_7				1042400.18
θ_8				-557682.99
θ_9				125201.43

Table of the coefficients $\boldsymbol{\theta}$ learned from training data.

Note how the magnitudes of coefficients increase dramatically as the order of polynomial increases.

Regularization

- Discourage the learned model parameters (i.e., coefficients) from being too large.

Regularization

- Discourage the learned model parameters (i.e., coefficients) from being too large.
- Keep the model simple.
- Keep the model from being unnecessarily complicated.

Regularization

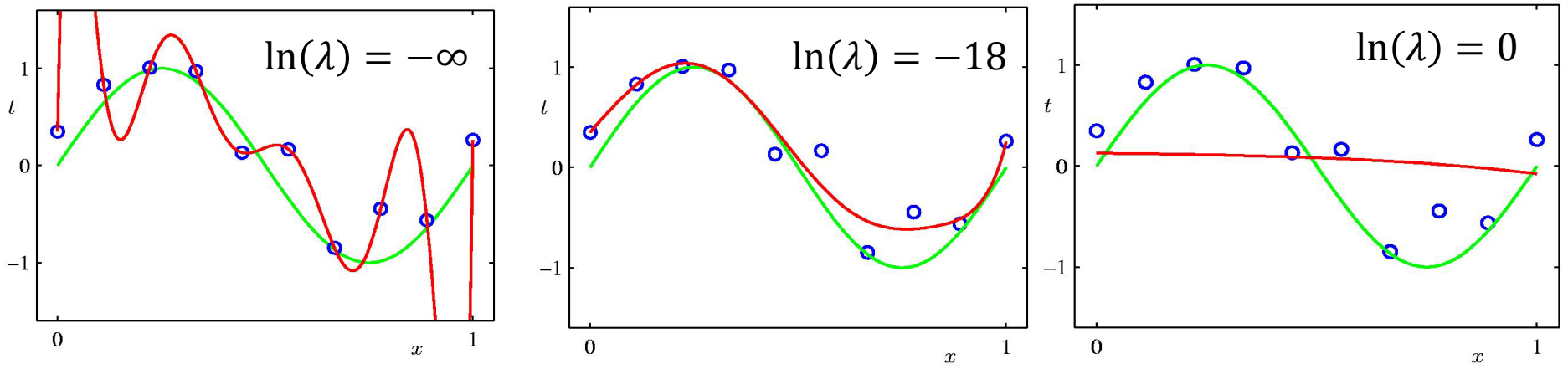
$$J(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{M}\left(h_\theta\left(x^{(i)}\right) - t^{(i)}\right)^2$$

Regularization

$$J(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{M}\left(h_\theta\left(x^{(i)}\right) - t^{(i)}\right)^2 + \frac{1}{2}\lambda\sum_{j=1}^{N}\theta_j^2$$

Also known as shrinkage because it shrinks/reduces the values of the model parameters

Regularized curve fitting



N = 9 M = 10 with three different regularization parameters

	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
θ_0	0.35	0.35	0.13
θ_1	232.37	4.74	-0.05
θ_2	-5321.83	-0.77	-0.06
θ_3	48568.31	-31.97	-0.05
θ_4	-231639.30	-3.89	-0.03
θ_5	640042.26	55.28	-0.02
θ_6	-1061800.52	41.32	-0.01
θ_7	1042400.18	-45.95	-0.00
θ_8	-557682.99	-91.53	0.00
θ_9	125201.43	72.68	0.01

As regularization parameter increases, the magnitudes of the learned model parameters decrease.

How to select regularization parameter?

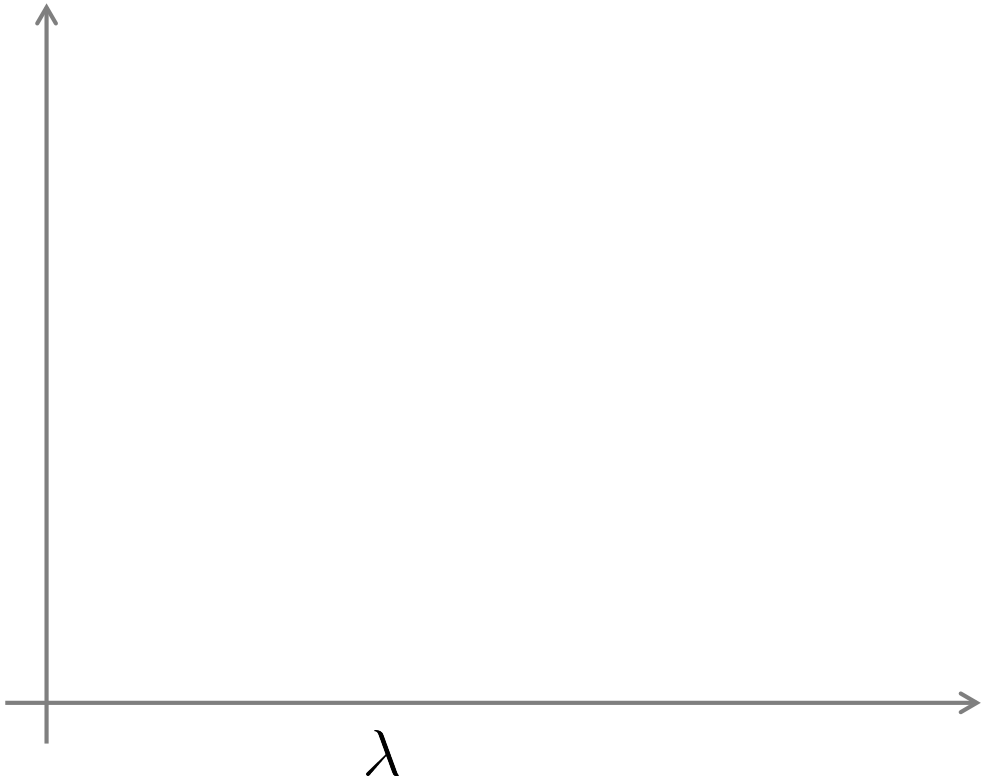
How to select regularization parameter?

- Split the whole data set into training and cross-validation data sets

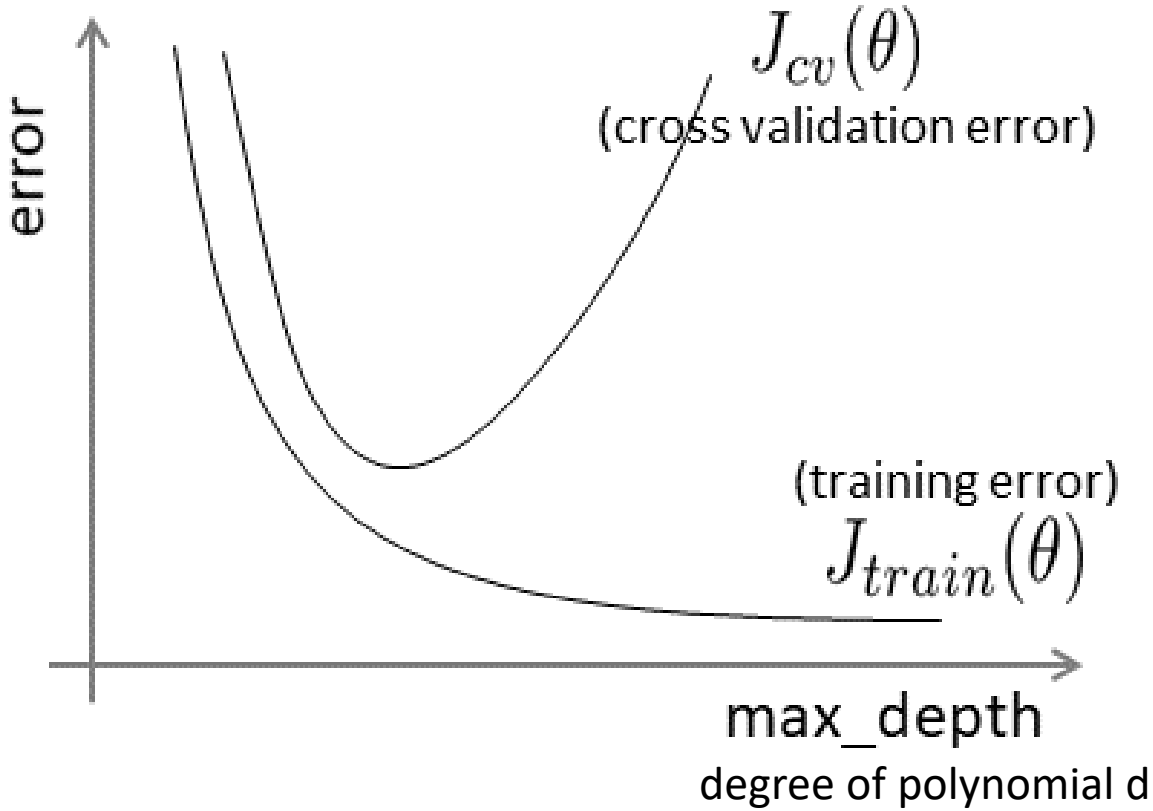
How to select regularization parameter?

- Split the whole data set into training and cross-validation data sets
- Carry out machine learning with different values for regularization parameters
- Calculate the errors for both training and CV data sets for each regularization parameter

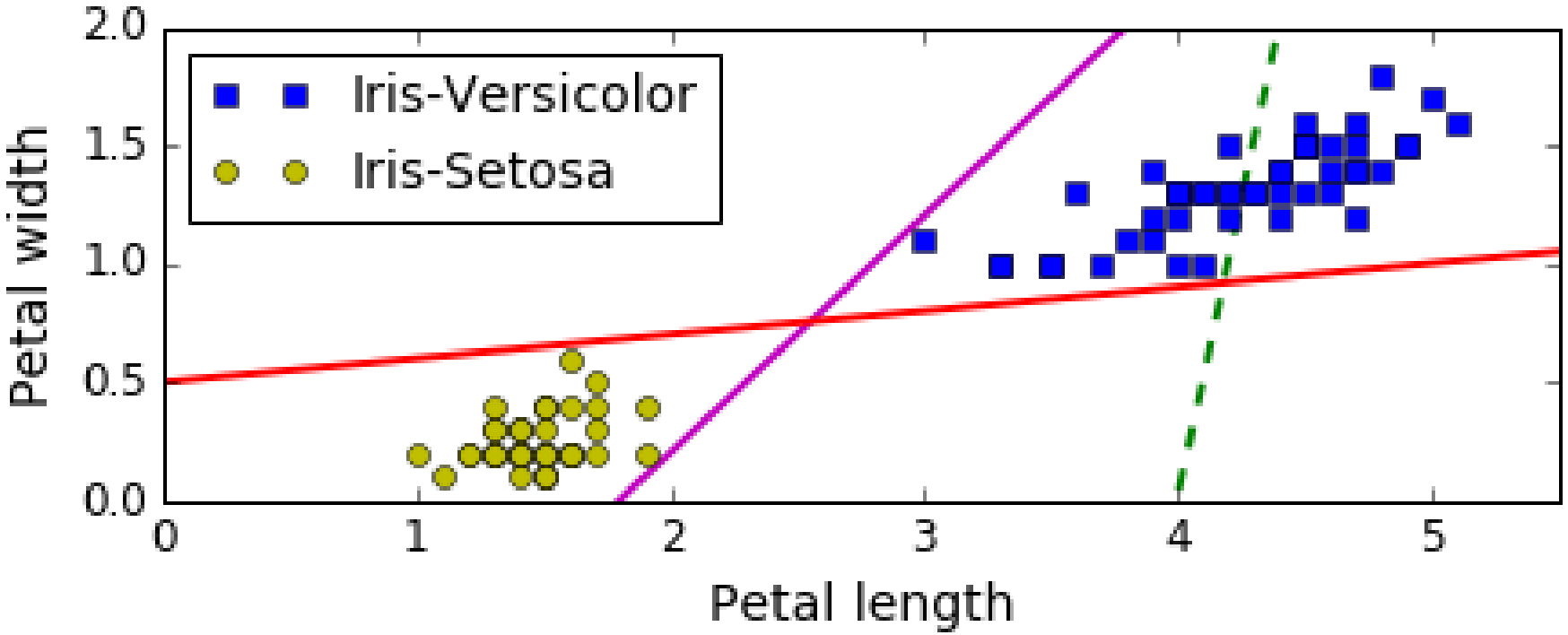
Bias/variance as a function of regularization parameter

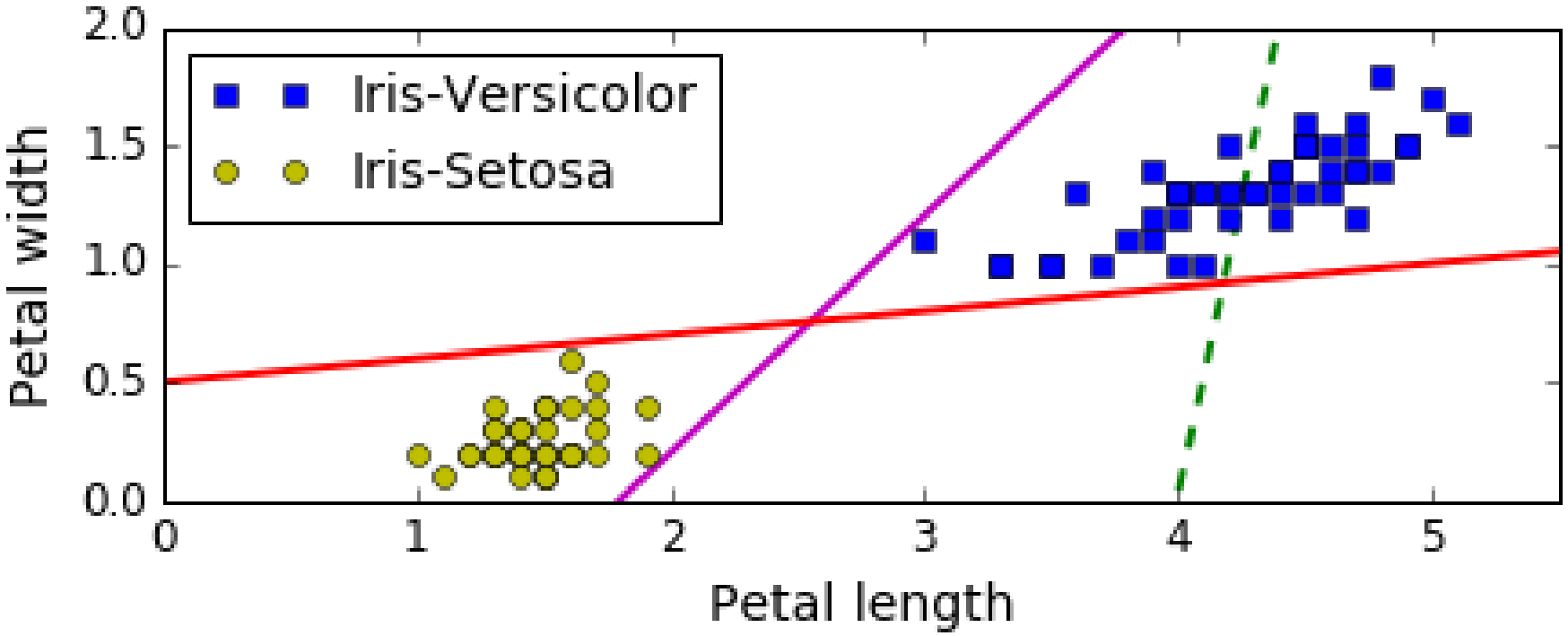


Diagnosing bias vs. variance

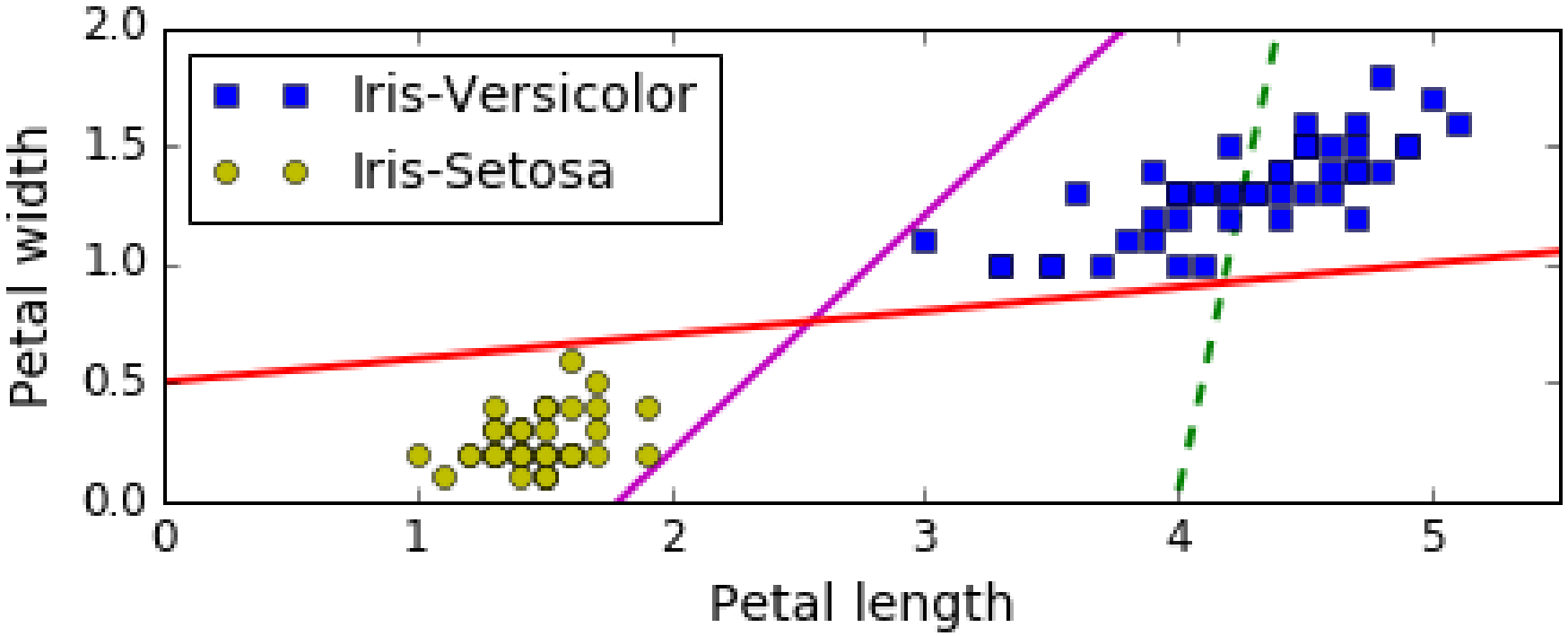


Support vector machine

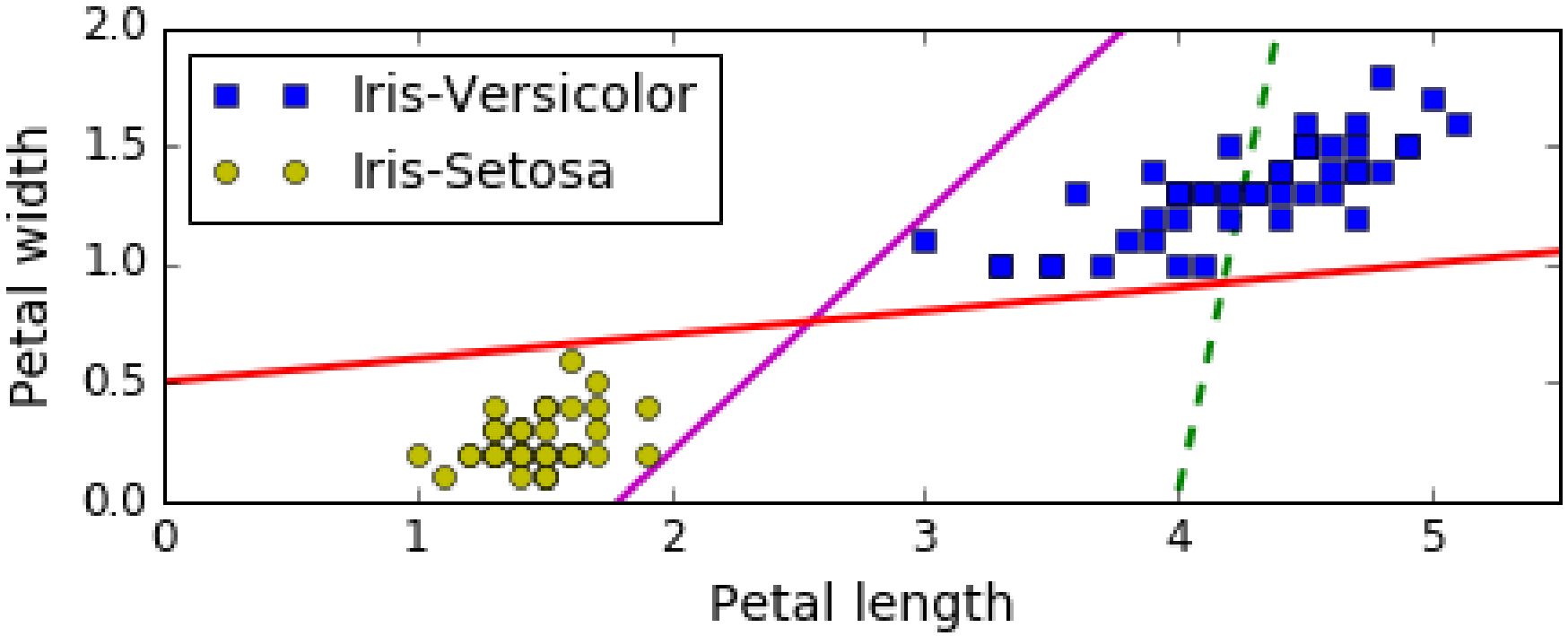

Iris-Versicolor
Iris-Setosa
Petal width
Petal length
0.0
0.5
1.0
1.5
2.0
0
1
2
3
4
5



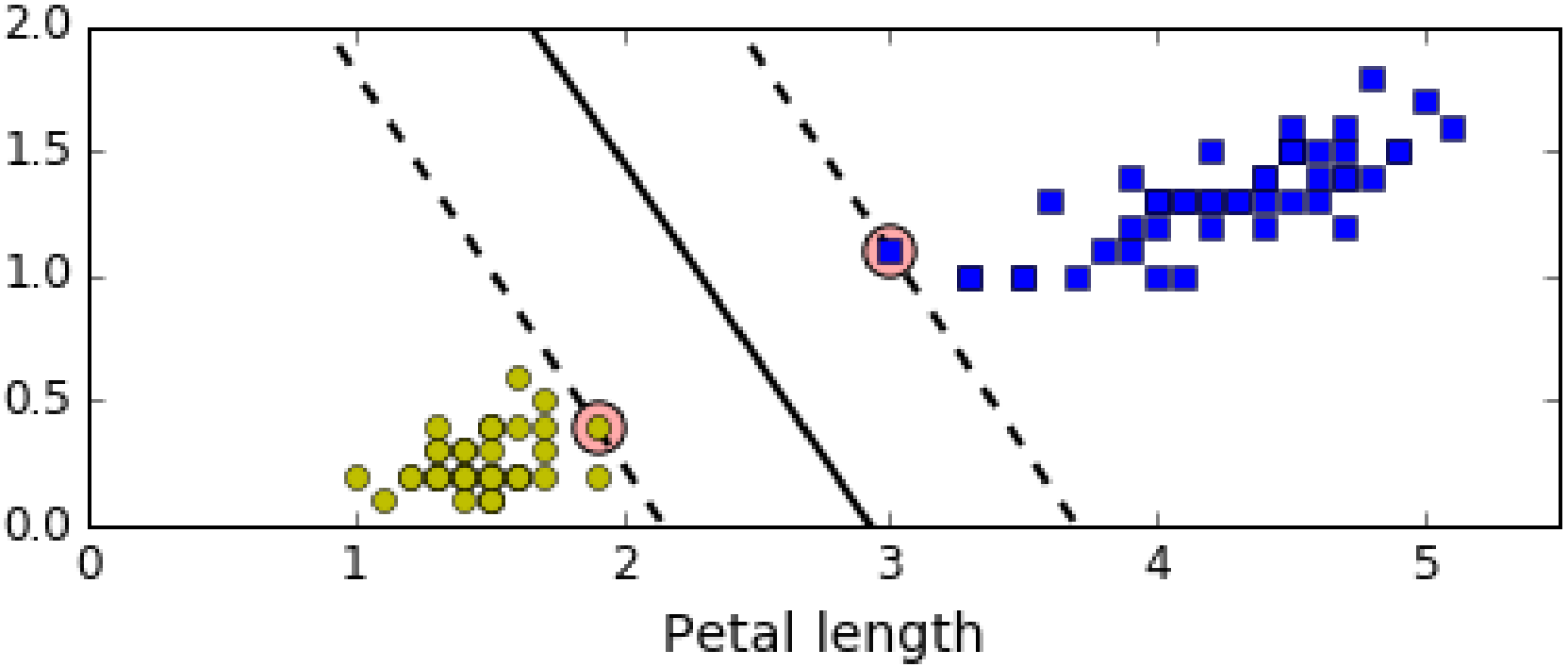
- The two classes are linearly separable



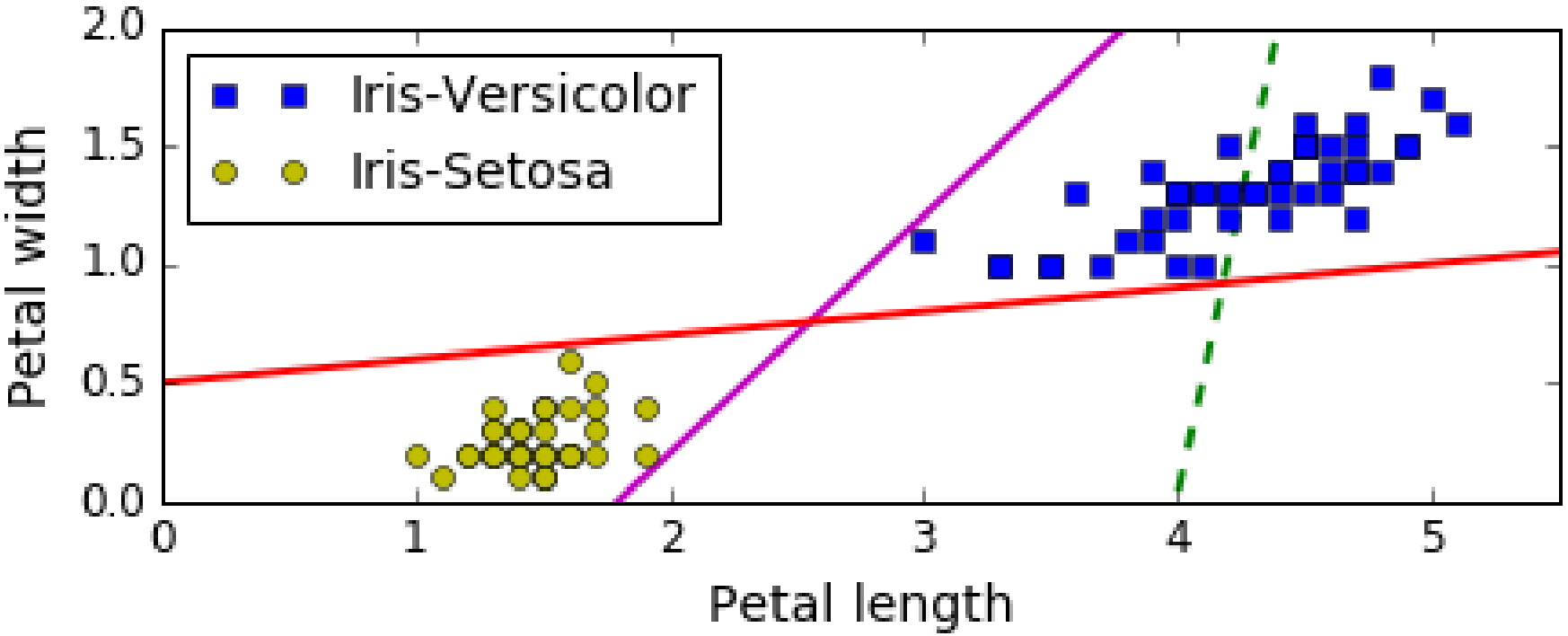
- The two classes are linearly separable
- Decision boundary in green dashed line is so bad that it does not even separate the classes properly.



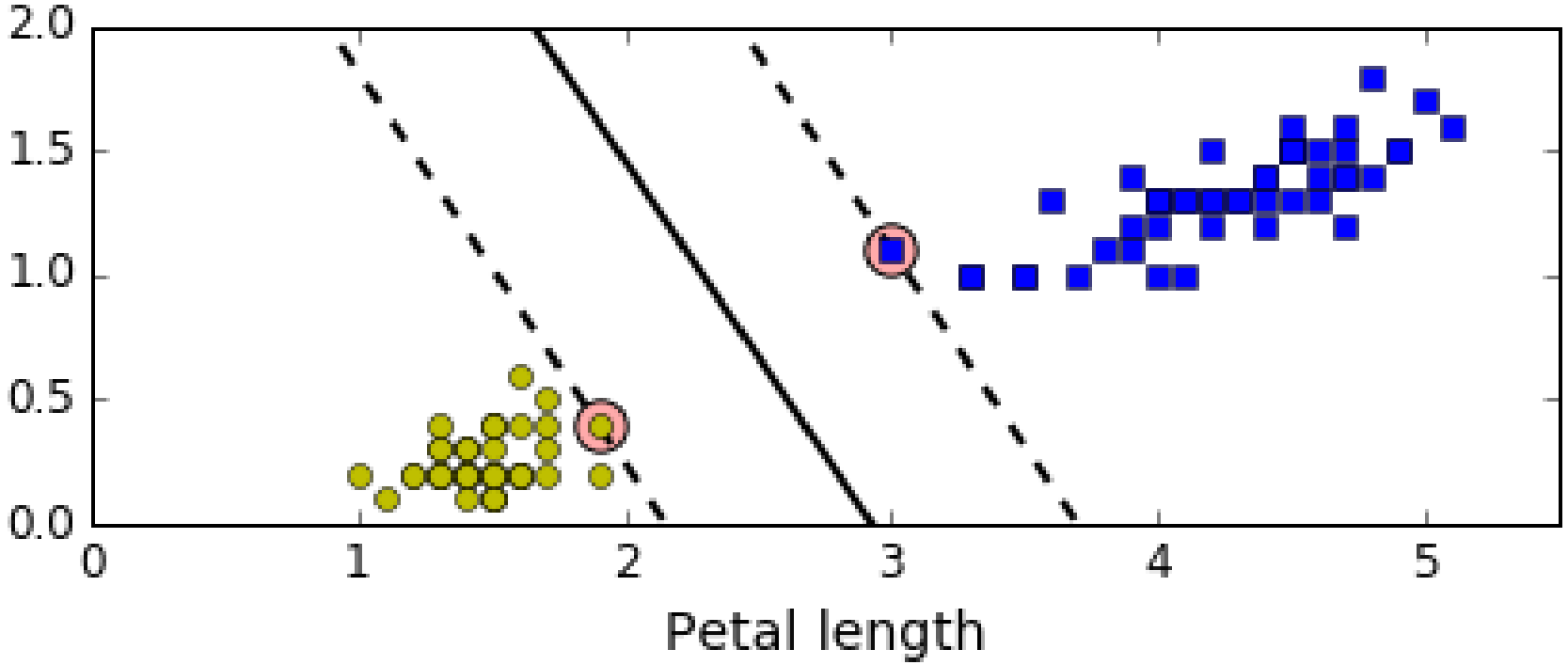
- The two classes are linearly separable
- Decision boundary in green dashed line is so bad that it does not even separate the classes properly.
- The decision boundaries in red and purple are good.



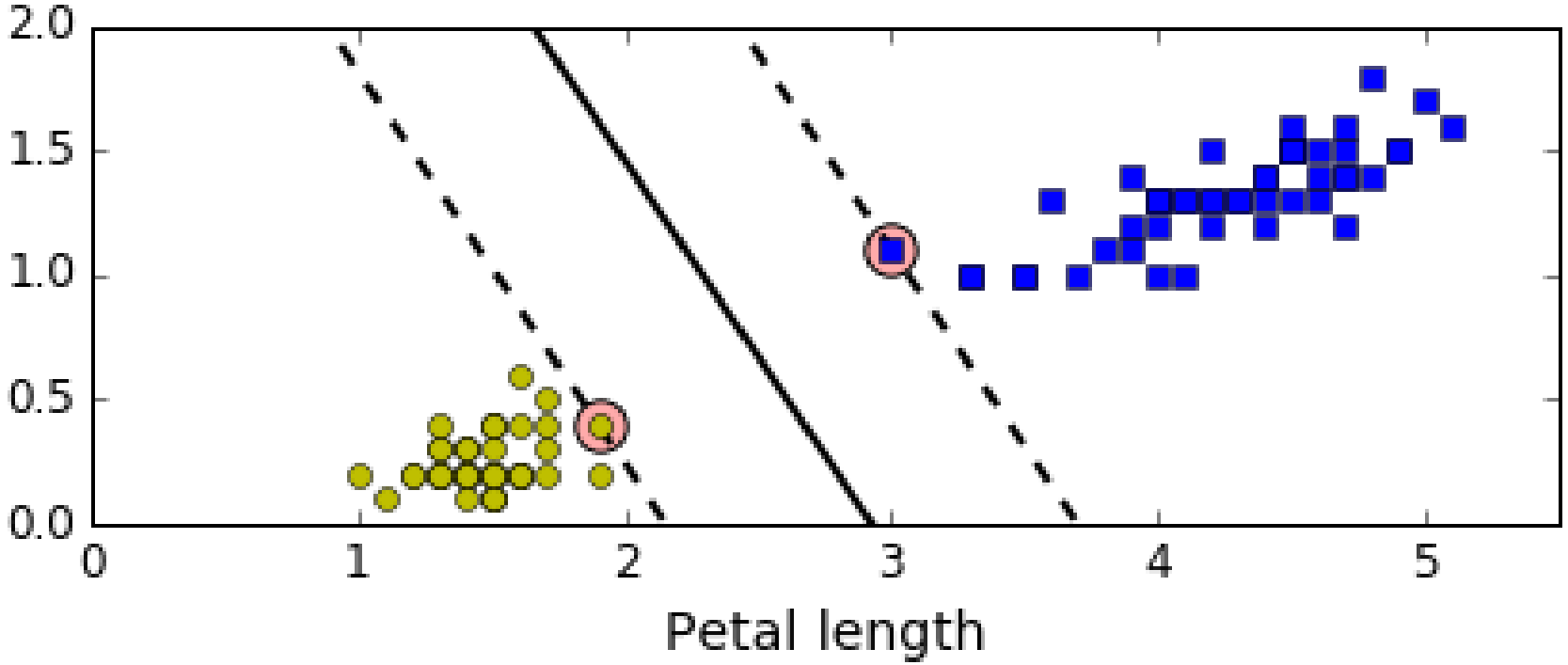
How about the decision boundary in solid black?



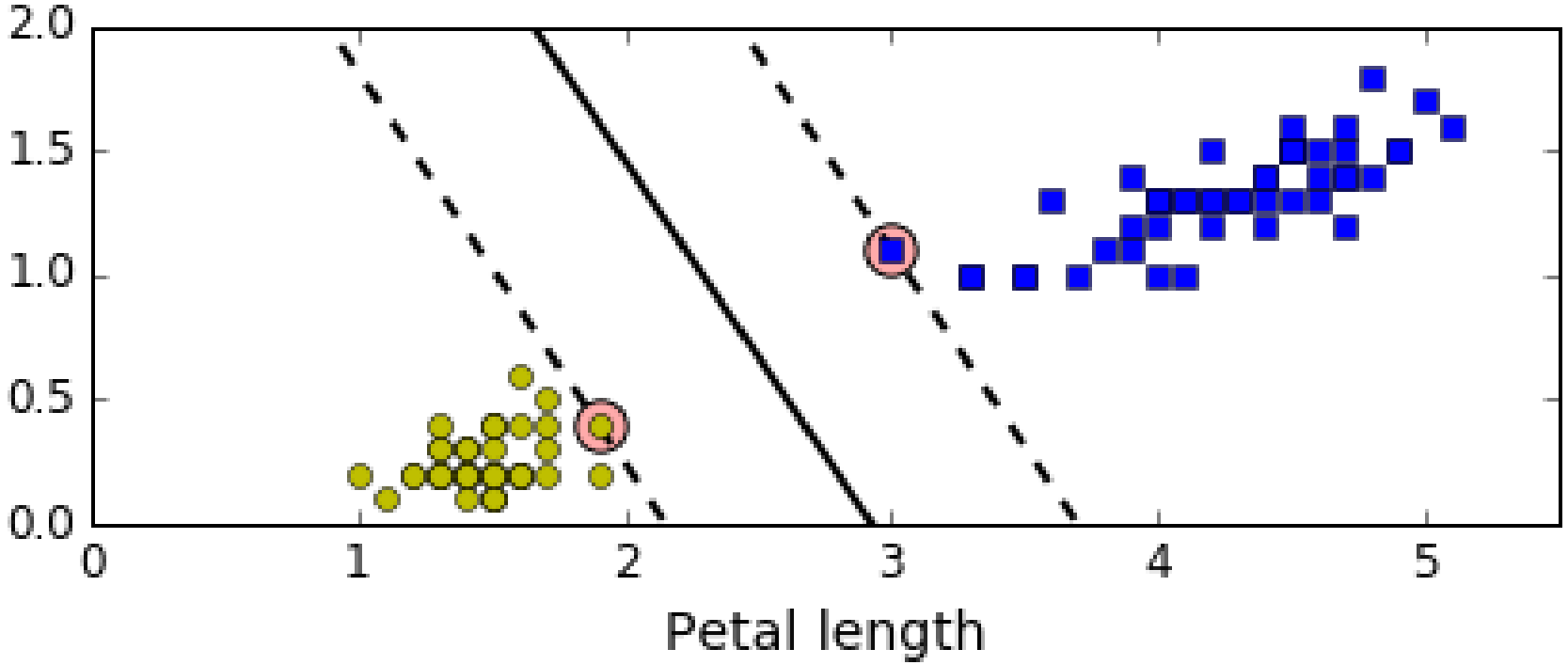
The decision boundaries in red and purple are so close to the training data example (a.k.a., instances) that they probably will not perform as well on new instances.



The decision boundary in solid black not only separates the two classes but also stays as far away from the closest training instances as possible.



- This is what SVM does. That is, large (or, max) margin classification.
- You can think of SVM classifier as fitting the widest possible street (represented by the parallel dashed lines) between the classes.

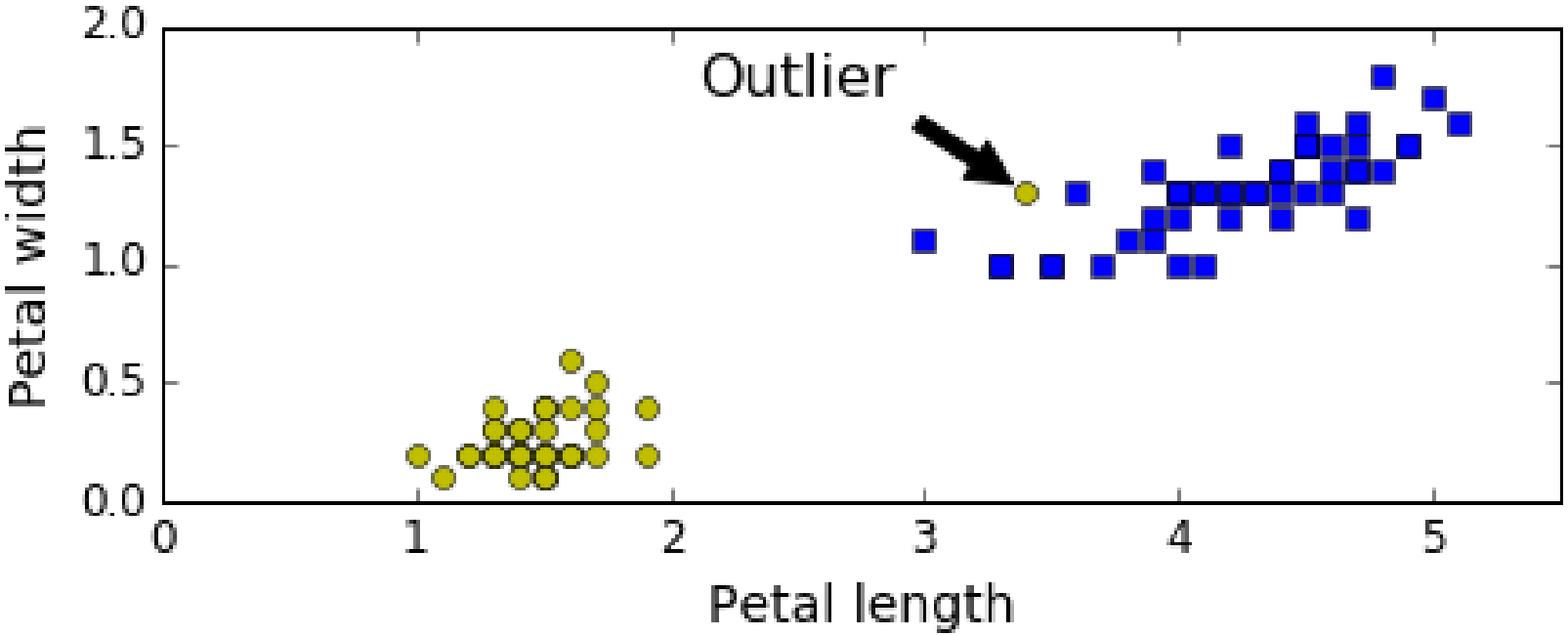


- Notice that, adding more training instances ‘off the street’ will not affect the decision boundary at all.
- The decision boundary is fully determined (or ‘supported’) by the instances on the edge of the street.
- These instances are called ‘support vectors’.

Hard margin classification

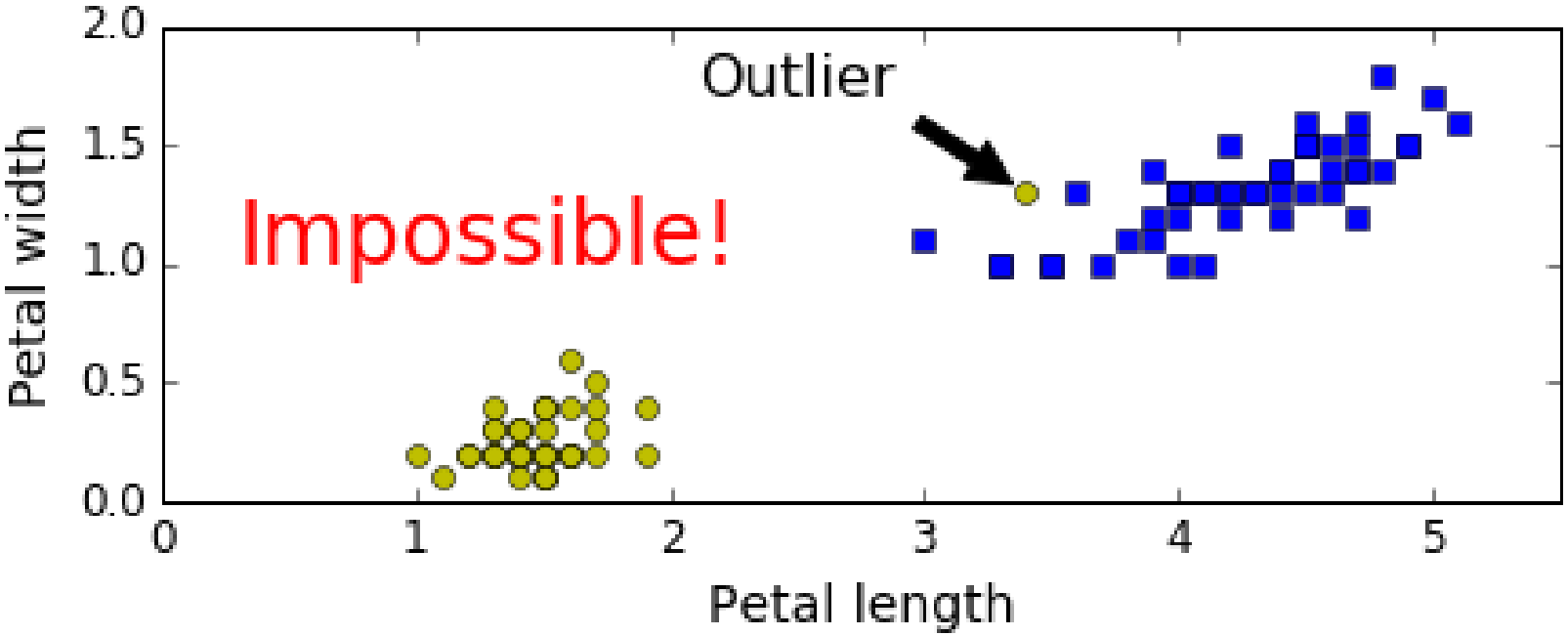
- If we strictly impose that all instances be off the street and on the right side, this is called hard margin classification.

- Problem with hard margin classification?



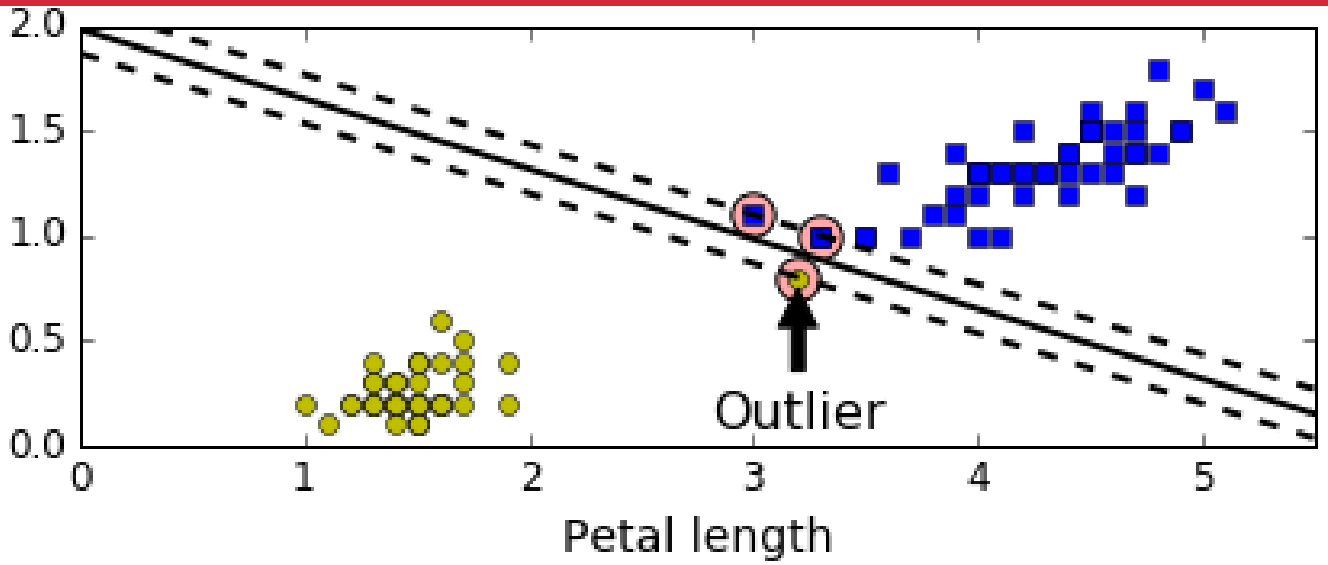
Add just one outlier

Is hard margin classification still possible?

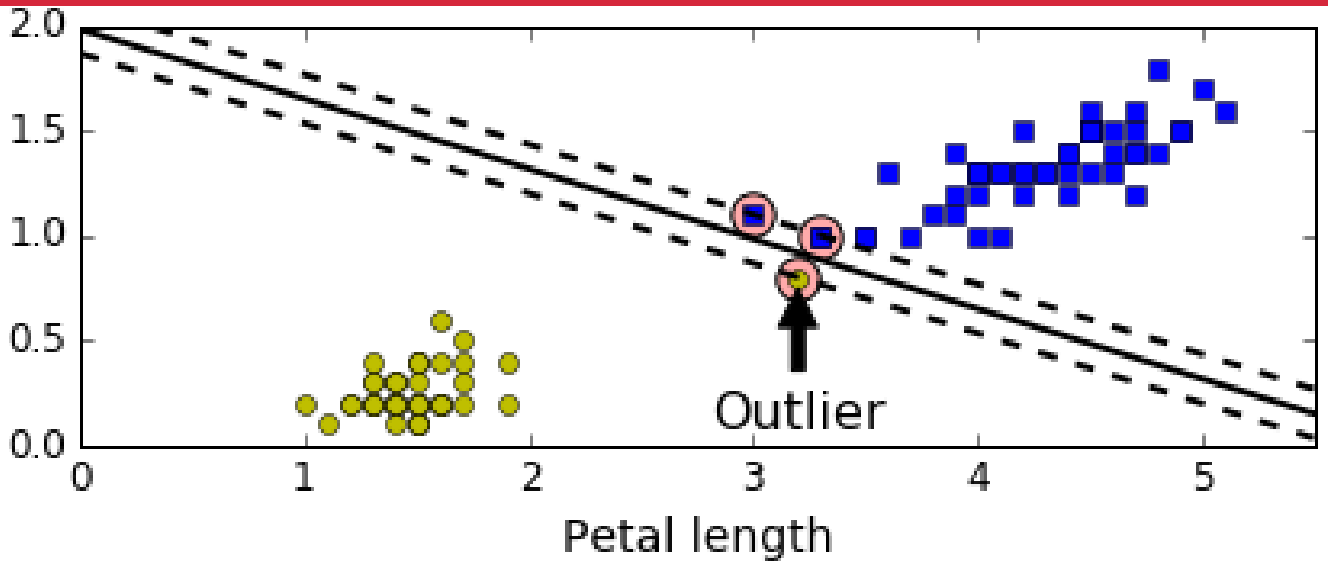


Add just one outlier

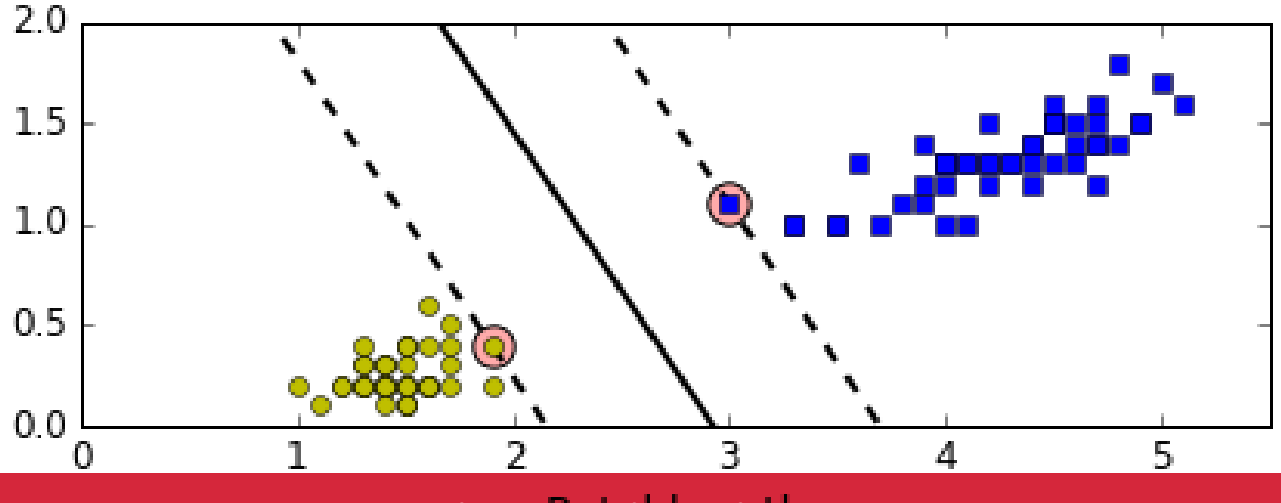
Is hard margin classification still possible?



With just one outlier, the decision boundary becomes very different.



With just one outlier, the decision boundary becomes very different.



Hard margin classification

- Too sensitive to outliers (or, errors, noise in the data)

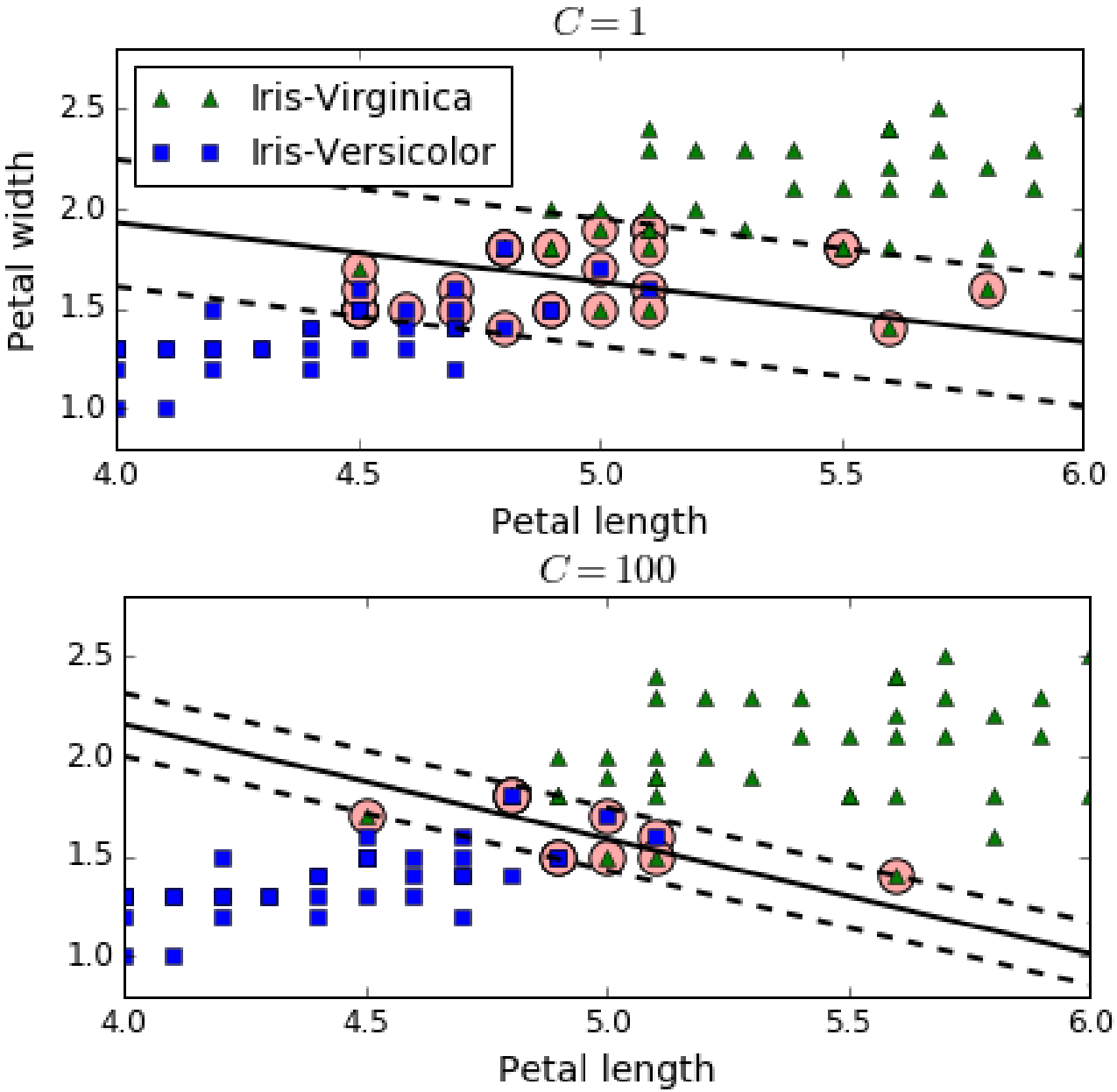
Soft margin classification

- Allow margin violations (i.e., instances that end up in the middle of the street, or even on the wrong side)
- Keep the street as wide as possible

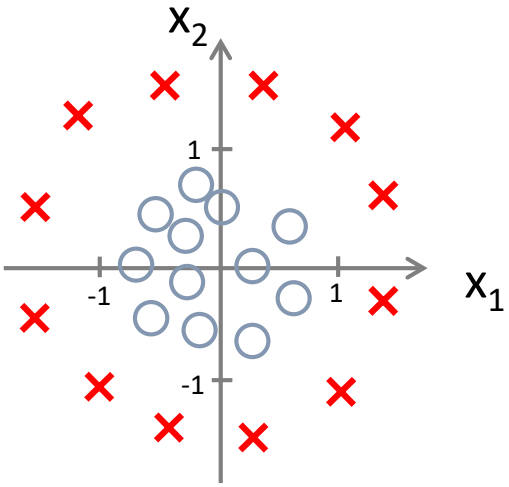
Soft margin classification

- Allow margin violations (i.e., instances that end up in the middle of the street, or even on the wrong side)
- Keep the street as wide as possible

- In Scikit-learn, you can control margin violations and street width using the C hyperparameter.
- A smaller C value leads to a wider street and more margin violations.


$C = 1$
Iris-Virginica
Iris-Versicolor
Petal width
2.5
2.0
1.5
1.0
4.0
4.5
5.0
5.5
6.0
Petal length
$C = 100$
2.5
2.0
1.5
1.0
4.0
4.5
5.0
5.5
6.0
Petal length

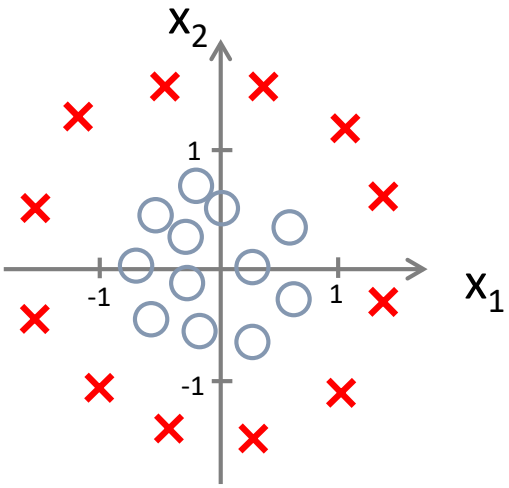
Nonlinear decision boundaries (logistic regression)



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2$$

This slide is taken from Andrew Ng's ML class on coursera

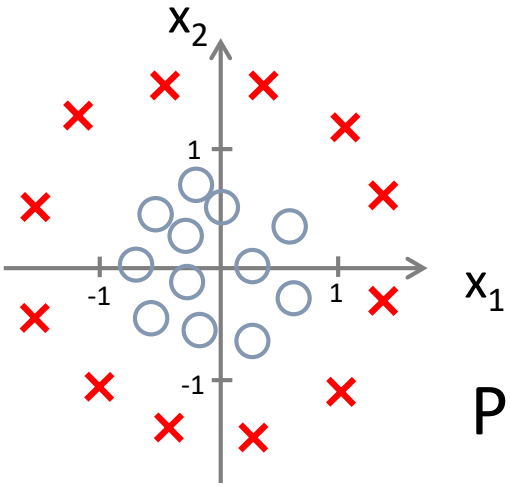
Nonlinear decision boundaries (logistic regression)



$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

This slide is taken from Andrew Ng's ML class on coursera

Nonlinear decision boundaries (logistic regression)



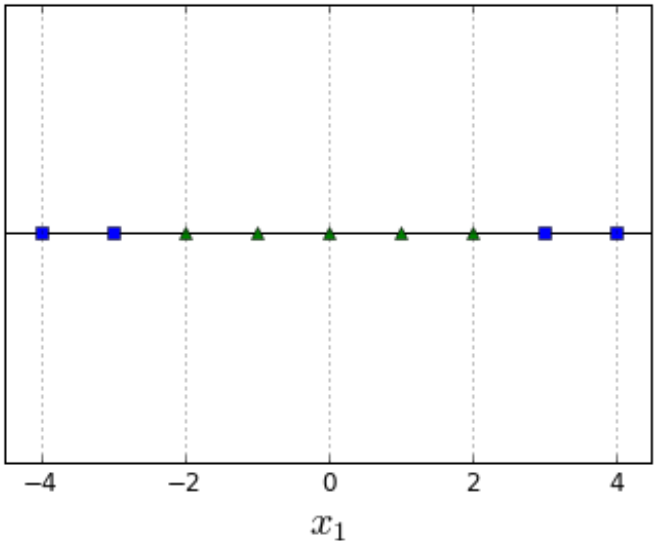
$$h_\theta(x) = g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1^2 + \theta_4 x_2^2)$$

Predict "$y = 1$" if $-1 + x_1^2 + x_2^2 \geq 0$

This slide is taken from Andrew Ng's ML class on coursera

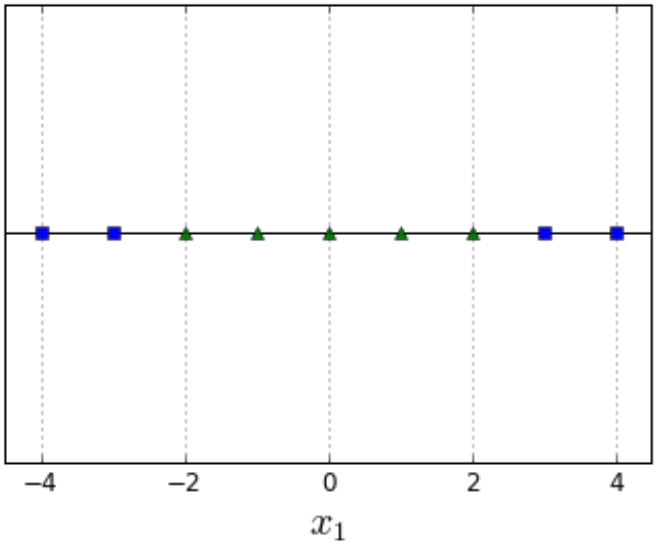
Nonlinear SVM decision boundary

- One approach to handling nonlinear datasets is to add more features, such as polynomial features.



Nonlinear SVM decision boundary

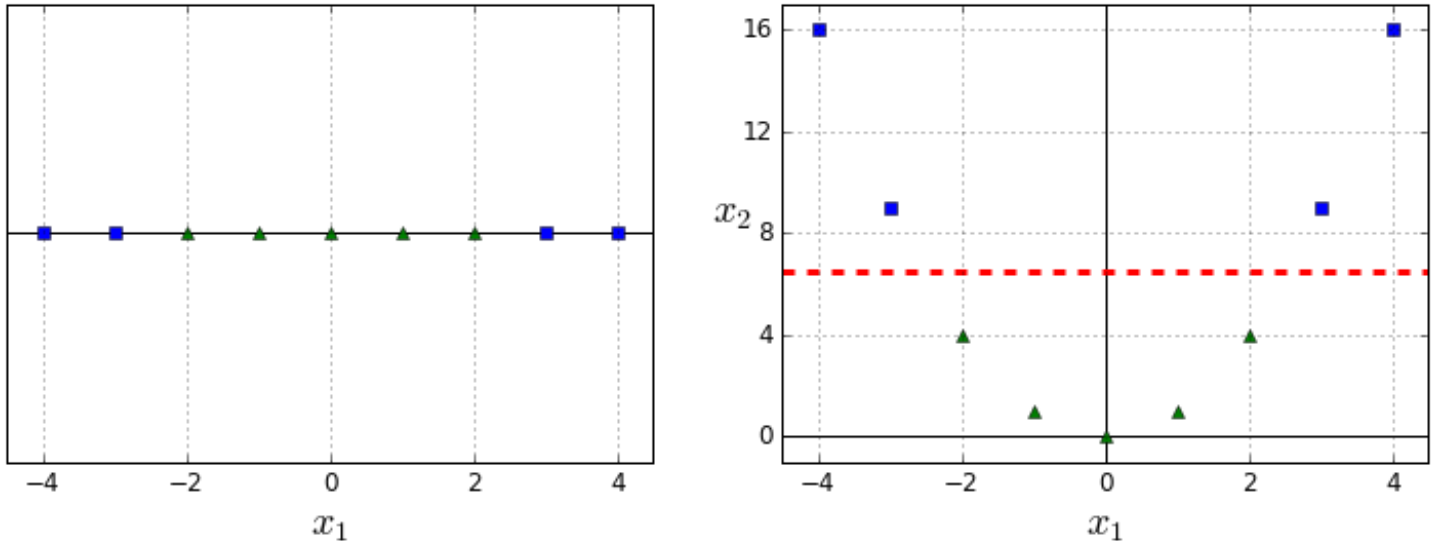
- One approach to handling nonlinear datasets is to add more features, such as polynomial features.



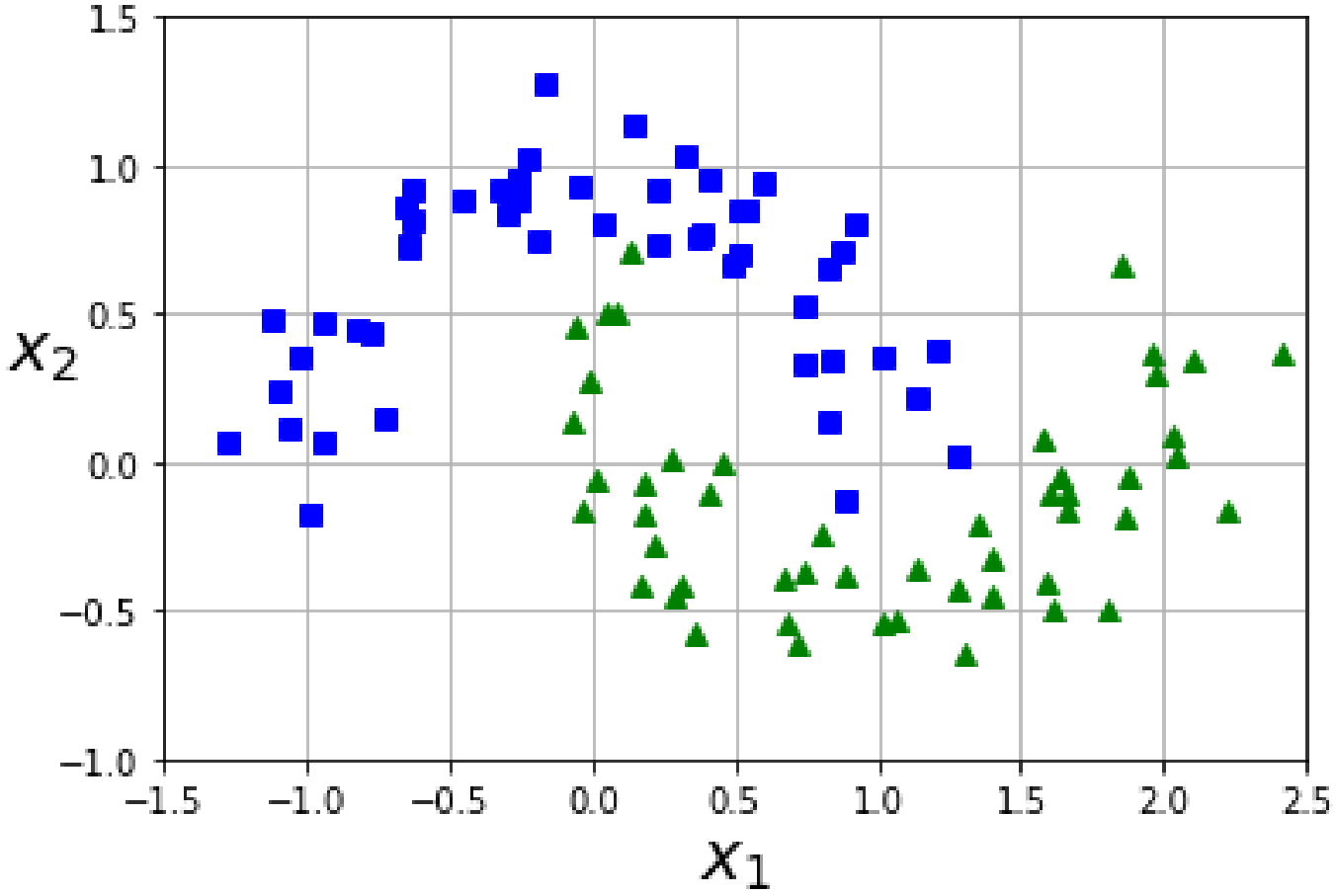
Add a second feature $x_2 = (x_1\ )^2$

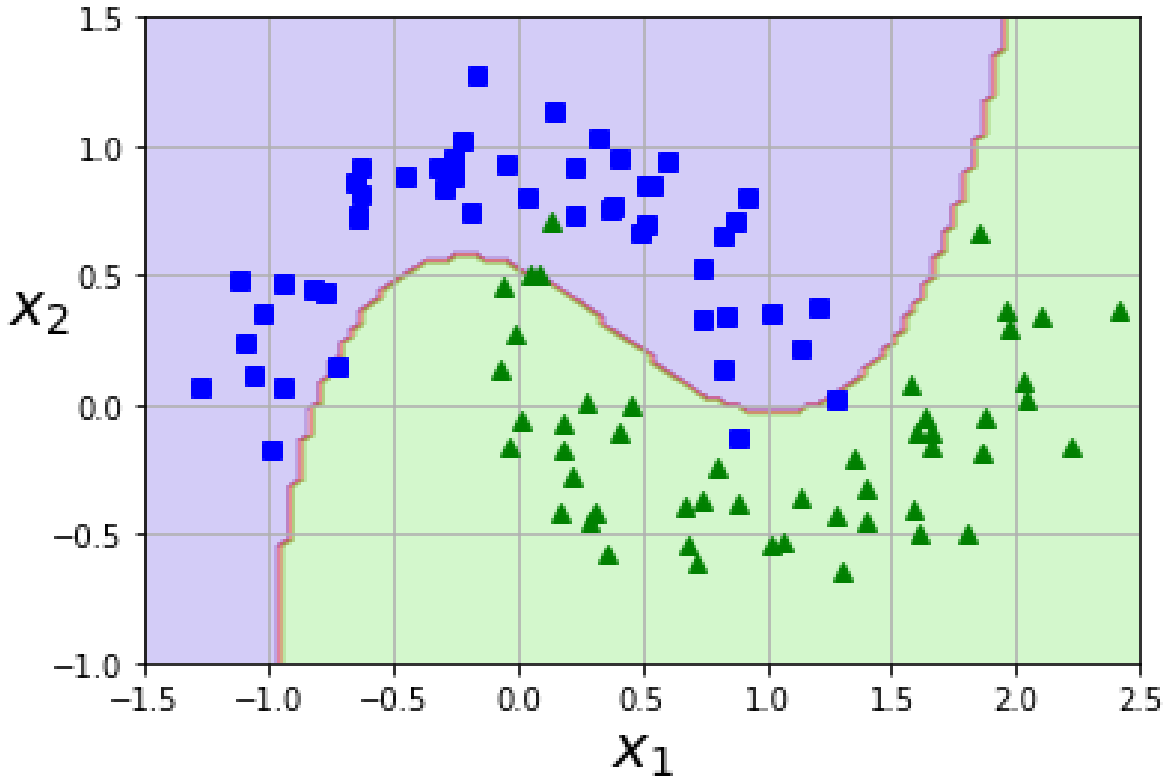
Nonlinear SVM decision boundary

- One approach to handling nonlinear datasets is to add more features, such as polynomial features.



Add a second feature $x_2 = (x_1)^2$


1.5
1.0
0.5
0.0
−0.5
−1.0
x_2
−1.5
−1.0
−0.5
0.0
0.5
1.0
1.5
2.0
2.5
x_1




```
from sklearn.svm import SVC
svm_clf = SVC(kernel = 'poly', degree = 3, coef0 = 1, C = 5)
svm_clf.fit(X,y)
```