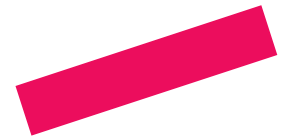


EVML3

# REGRESSION

Prediction from estimating relationships  
between variables and outcomes

JEROEN VEEN

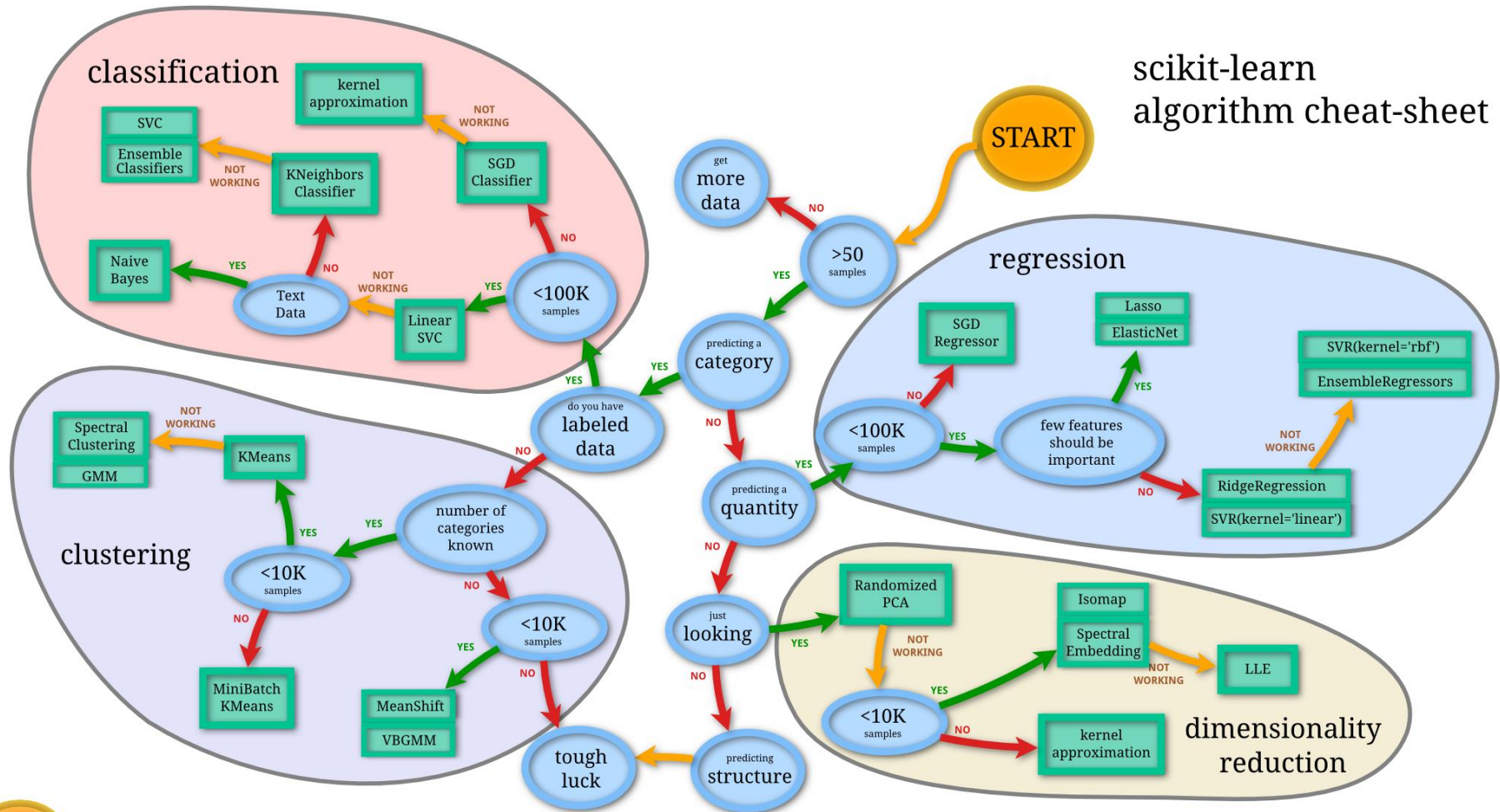


**HAN\_**UNIVERSITY  
OF APPLIED SCIENCES

# QUIZ TIME

- Individual, multiple-choice questions
- Online: <http://www.socrative.com> room **1PTGB6PY**
- Open book quiz, so books and slides can be consulted
- **HAN student number**, so NOT your name, nickname or anything else.
- Quiz starts exactly at class hour and takes 10 minutes.
- Be on time and have your equipment prepared.

# scikit-learn algorithm cheat-sheet



# CONTENTS

- Linear regression
- Polynomial regression
- Regularized linear models
- Logistic regression
- Classification and regression

# LINEAR REGRESSION

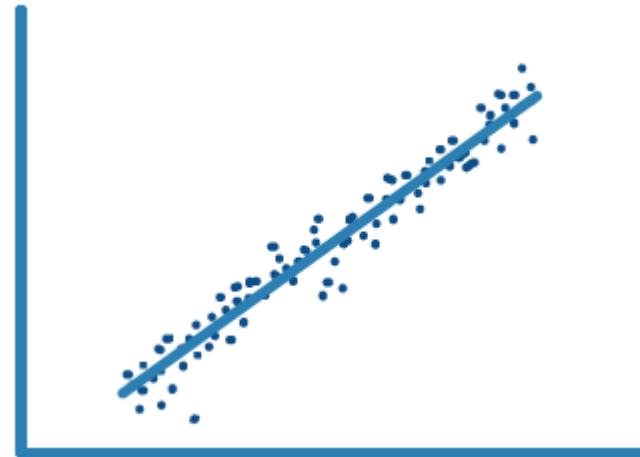
- Easy to interpret
- Fast to fit
- Benchmark

*Equation 4-1. Linear Regression model prediction*

$$\hat{y} = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

In this equation:

- $\hat{y}$  is the predicted value.
- $n$  is the number of features.
- $x_i$  is the  $i^{\text{th}}$  feature value.
- $\theta_j$  is the  $j^{\text{th}}$  model parameter (including the bias term  $\theta_0$  and the feature weights  $\theta_1, \theta_2, \dots, \theta_n$ ).

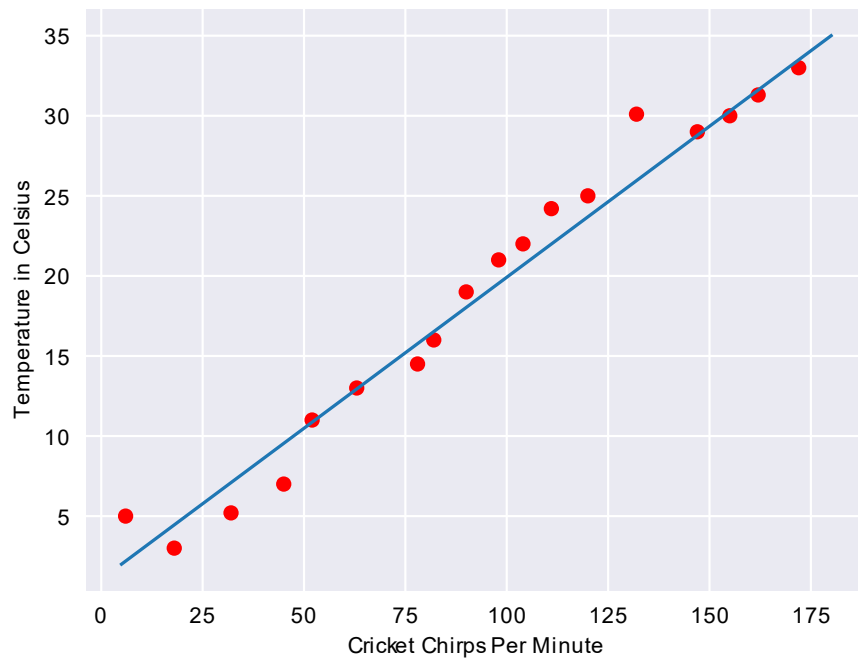


Source: Mathworks, Applying Supervised Learning

Source: Géron, ISBN: 9781492032632

# SIMPLE EXAMPLE

- Cricket chips vs temperature



$y$  is the temperature in Celsius—the value we're trying to predict.

$m$  is the slope of the line.

$x$  is the number of chirps per minute—the value of our input feature.

$b$  is the y-intercept.

# LINEAR REGRESSION MODEL FITTING

- Performance measure

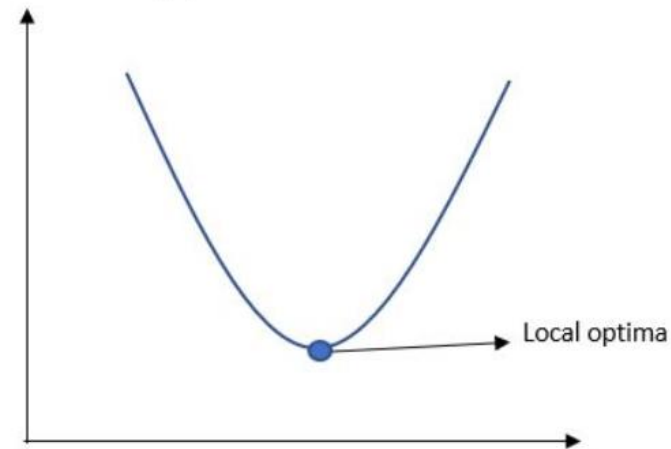
*Equation 4-3. MSE cost function for a Linear Regression model*

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

- Aka cost function (J)
- Note the vectorized form

- $\boldsymbol{\theta}$  is the model's *parameter vector*, containing the bias term  $\theta_0$  & weights  $\theta_1$  to  $\theta_n$ .
- $\mathbf{x}$  is the instance's *feature vector*, containing  $x_0$  to  $x_n$ , with  $x_0$  always equal to 1.
- $\boldsymbol{\theta} \cdot \mathbf{x}$  is the dot product of the vectors  $\boldsymbol{\theta}$  and  $\mathbf{x}$ , which is of course equal to  $\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$ .
- $h_{\boldsymbol{\theta}}$  is the hypothesis function, using the model parameters  $\boldsymbol{\theta}$ .

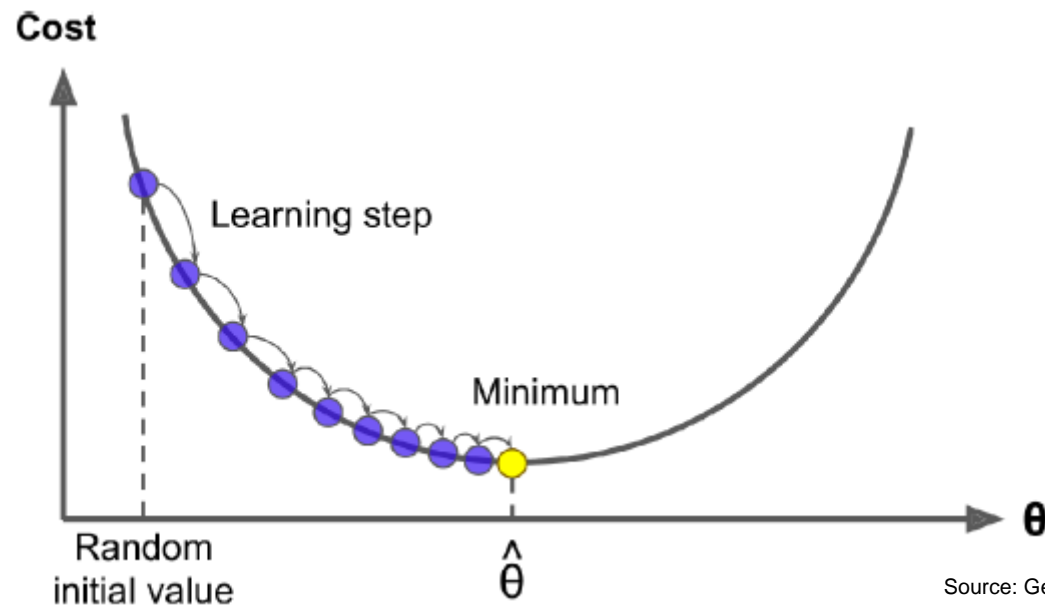
Cost Function (J)



Source: Géron, ISBN: 9781492032632

# GRADIENT DESCENT

- Tweak parameters iteratively in order to minimize a cost function



Source: Géron, ISBN: 9781492032632



# GRADIENT DESCENT

*Equation 4-5. Partial derivatives of the cost function*

$$\frac{\partial}{\partial \theta_j} \text{MSE}(\boldsymbol{\theta}) = \frac{2}{m} \sum_{i=1}^m (\boldsymbol{\theta}^\top \mathbf{x}^{(i)} - y^{(i)}) x_j^{(i)}$$

*Equation 4-7. Gradient Descent step*

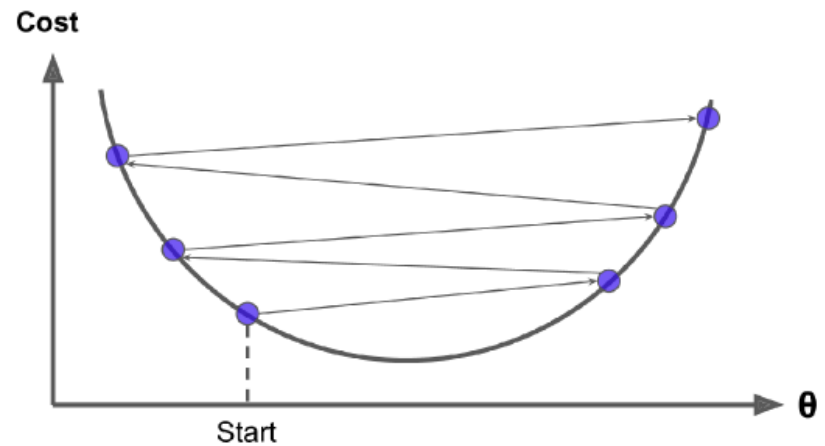
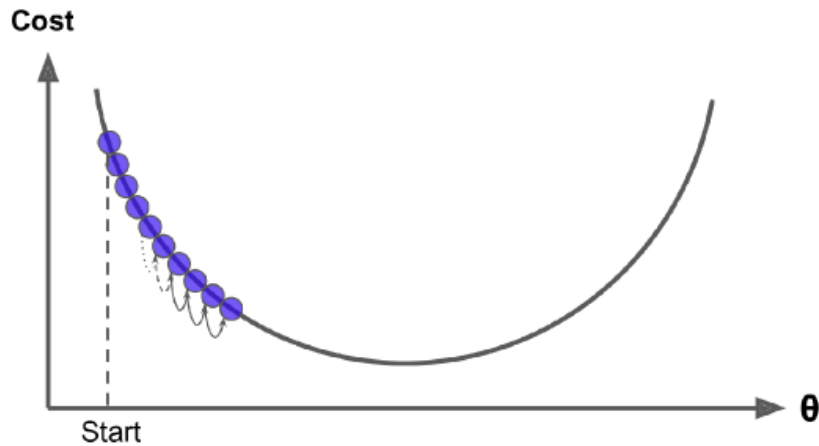
$$\boldsymbol{\theta}^{(\text{next step})} = \boldsymbol{\theta} - \boxed{\eta \nabla_{\boldsymbol{\theta}} \text{MSE}(\boldsymbol{\theta})}$$

Learning rate

$$\nabla_{\boldsymbol{\theta}} \text{MSE}(\boldsymbol{\theta}) = \begin{pmatrix} \frac{\partial}{\partial \theta_0} \text{MSE}(\boldsymbol{\theta}) \\ \frac{\partial}{\partial \theta_1} \text{MSE}(\boldsymbol{\theta}) \\ \vdots \\ \frac{\partial}{\partial \theta_n} \text{MSE}(\boldsymbol{\theta}) \end{pmatrix}$$

# LEARNING RATE

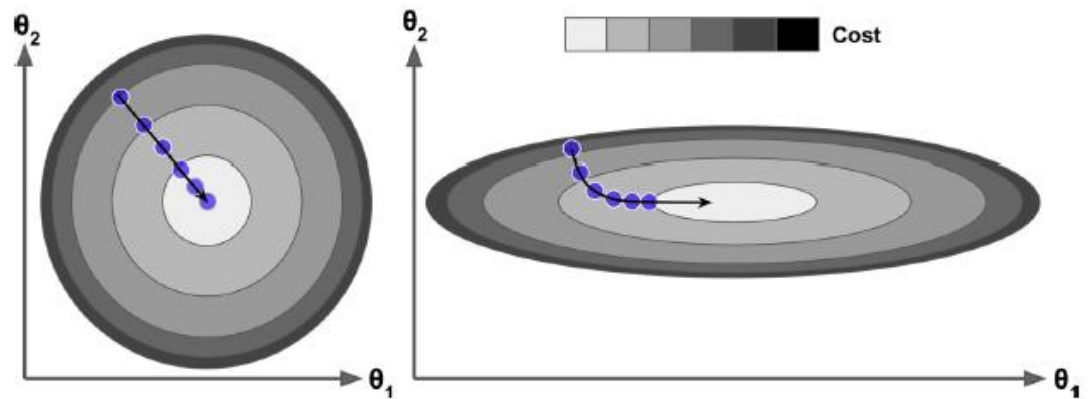
- Stepsize
- Convergence
- Stopping criterion



Source: Géron, ISBN: 9781492032632

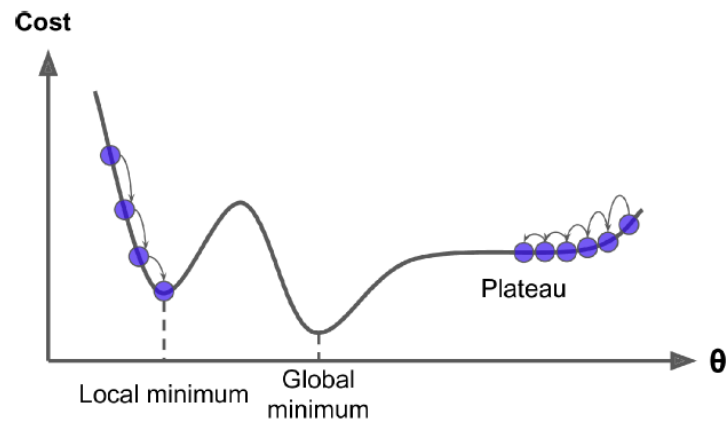
# PITFALLS

- Unbalanced features



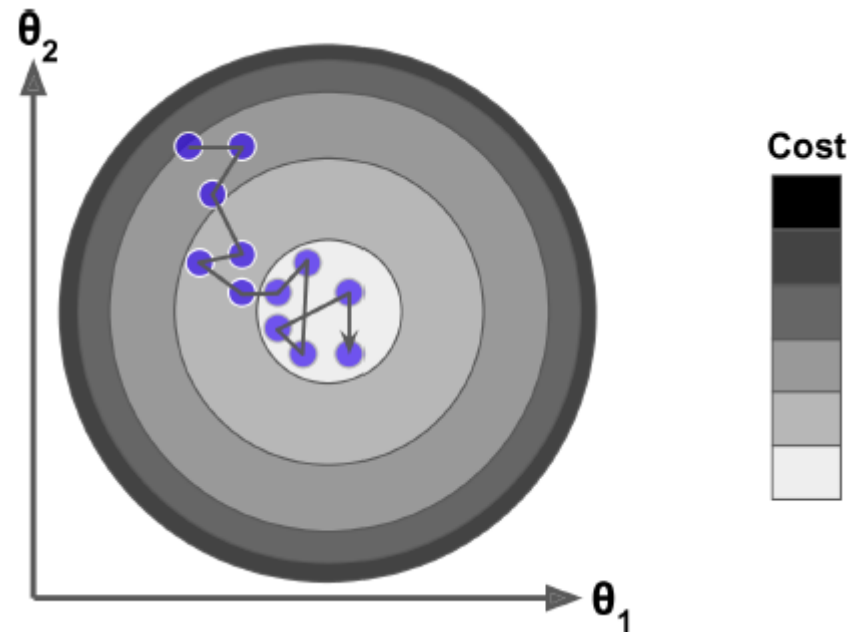
Source: Géron, ISBN: 9781492032632

- Local minima



# STOCHASTIC GRADIENT DESCENT

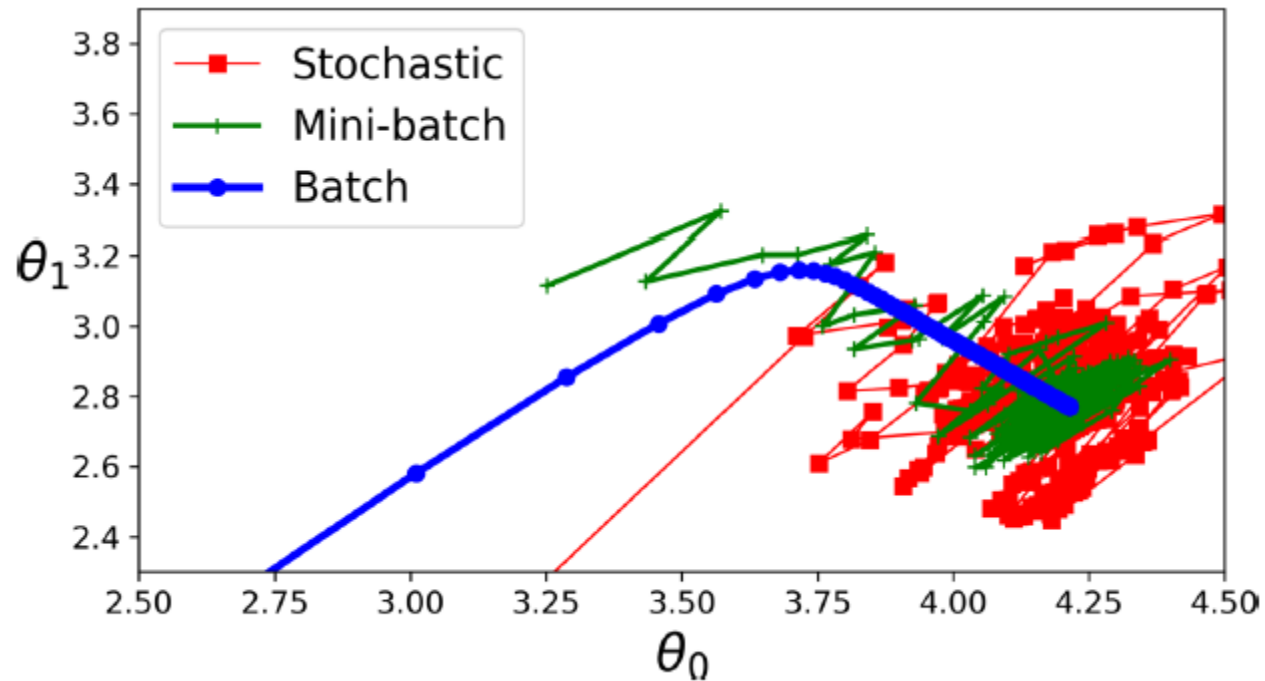
- Pick a random sample and compute gradient
- Start with flexible learning rate, and reducing it as the number of examples grows  
-> learning schedule



Source: Géron, ISBN: 9781492032632

# MINI-BATCH SGD

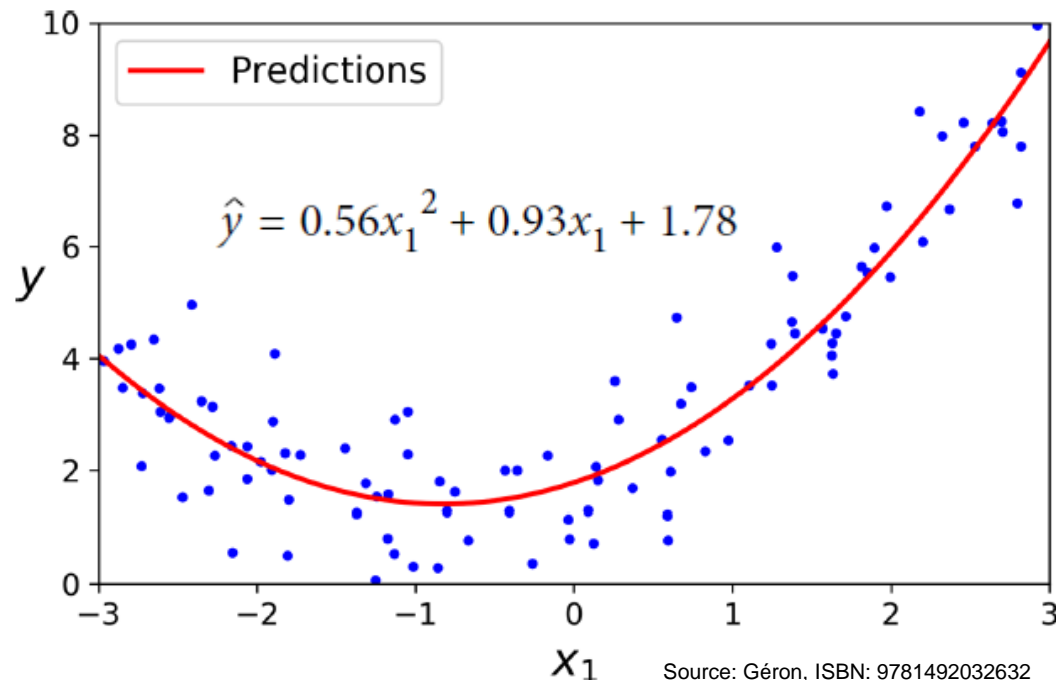
- Dealing with large datasets



Source: Géron, ISBN: 9781492032632

# POLYNOMIAL REGRESSION

- Add powers of each feature as new features

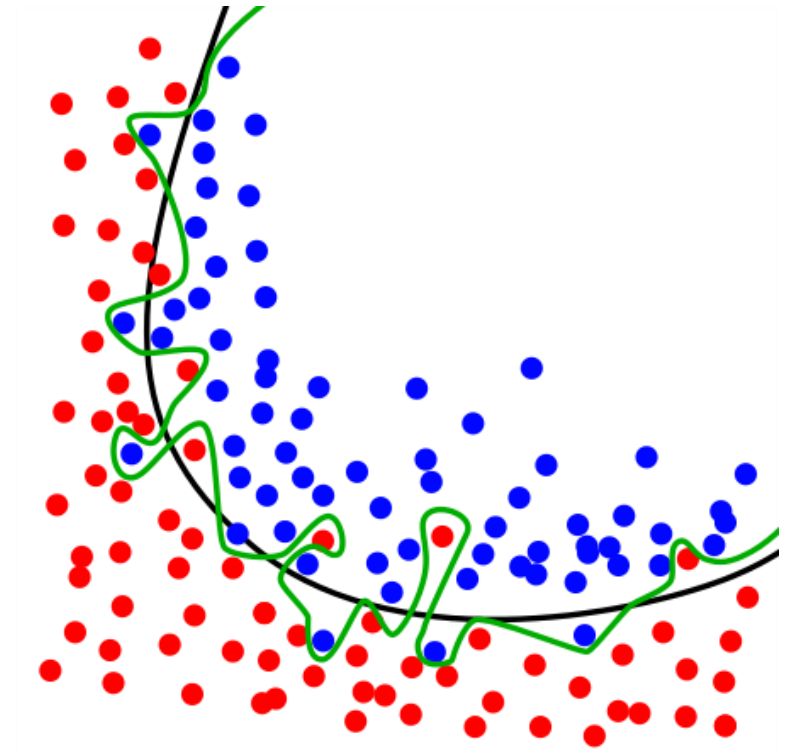


- Still a linear function of the fitting parameters

# PERIL OF OVERFITTING

- An overfit model has more parameters than can be justified by the data.
- Gets a low loss during training but does a poor job predicting new data.
- Results in poor generalization

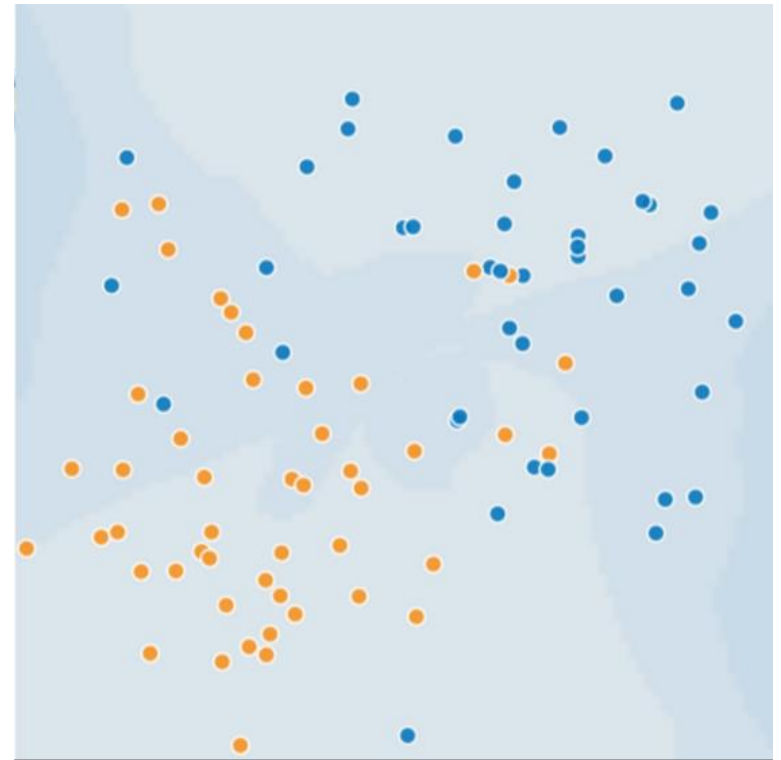
*With four parameters I can fit an elephant, and with five I can make him wiggle his trunk – John von Neumann*



Source: <https://en.wikipedia.org/wiki/Overfitting>

# SIMPLE EXAMPLE

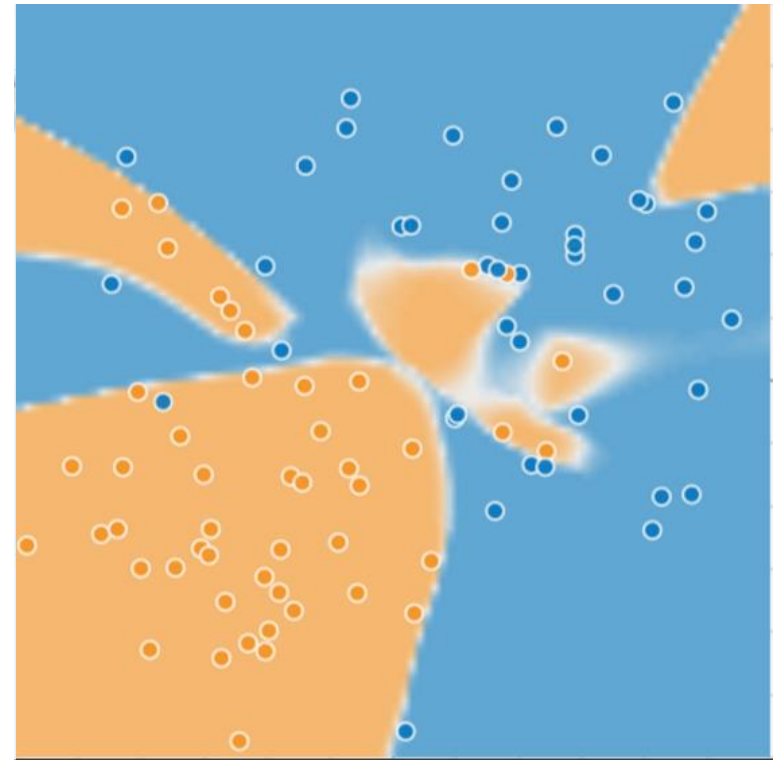
- Sick (blue) and healthy (orange) trees in a forest.





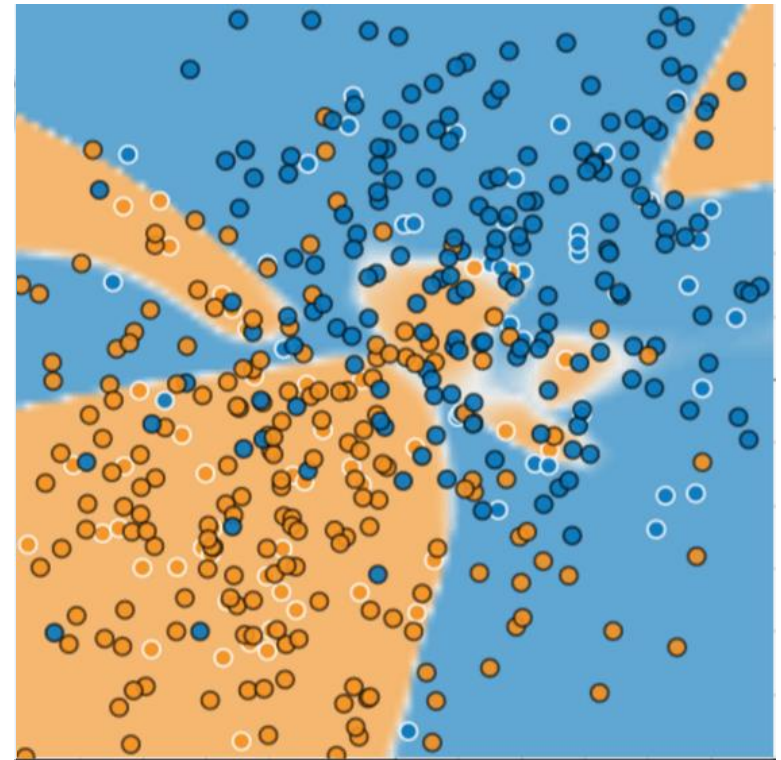
# SIMPLE EXAMPLE

- Sick (blue) and healthy (orange) trees in a forest.
- Low loss, excellent model?



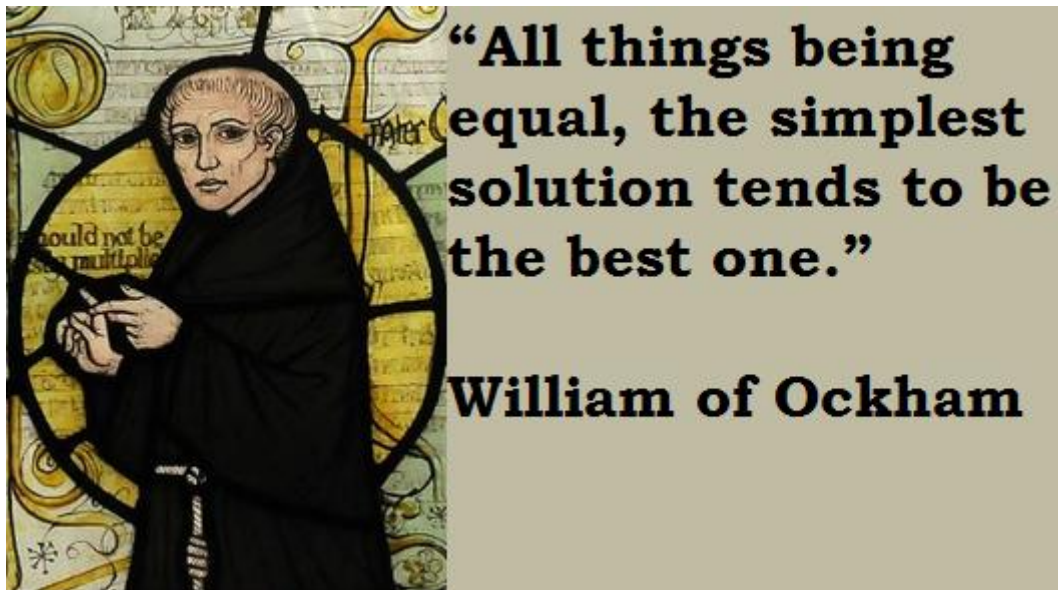
# SIMPLE EXAMPLE

- Sick (blue) and healthy (orange) trees in a forest.
- Low loss, excellent model?
- Poor generalization!
- Model is more complex than necessary



# OCKHAM'S RAZOR

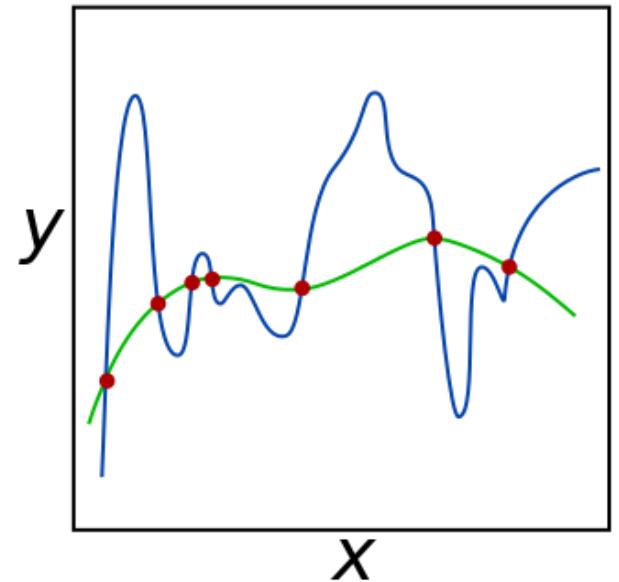
- William of Ockham, a 14th century friar and philosopher, loved simplicity. He believed that scientists should prefer simpler formulas or theories over more complex ones.



Source: <https://medium.com/neuralspace/how-bayesian-methods-embody-occams-razor-43f3d0253137>

# REGULARIZED LINEAR MODELS

- Reduce overfitting by constraining a model
- E.g. limiting the number of polynomial degrees
- Or adding a penalty to the cost function



Source: [https://en.wikipedia.org/wiki/Regularization\\_\(mathematics\)](https://en.wikipedia.org/wiki/Regularization_(mathematics))

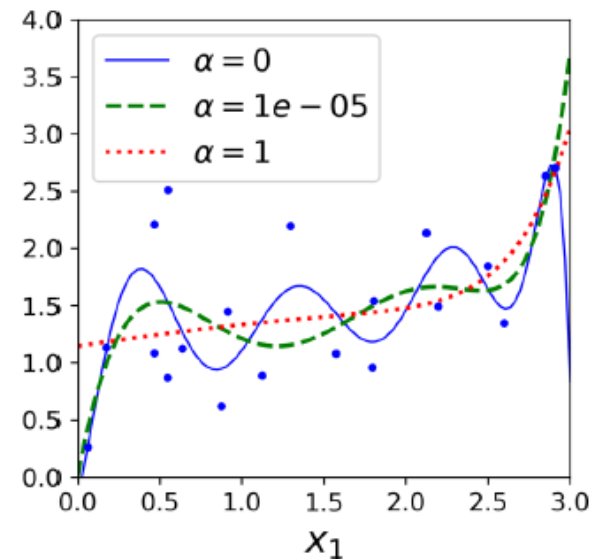
# RIDGE REGRESSION

- L2 regularization

*Equation 4-8. Ridge Regression cost function*

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

- Note that  $\theta_0$  is not in the penalization, since it is reflected in the MSE.
- Penalize sum of squared coefficients values
- Fit the data but also keep the model weights as small as possible
- Keep all features in the model, and balances the contribution



Source: Géron, ISBN: 9781492032632

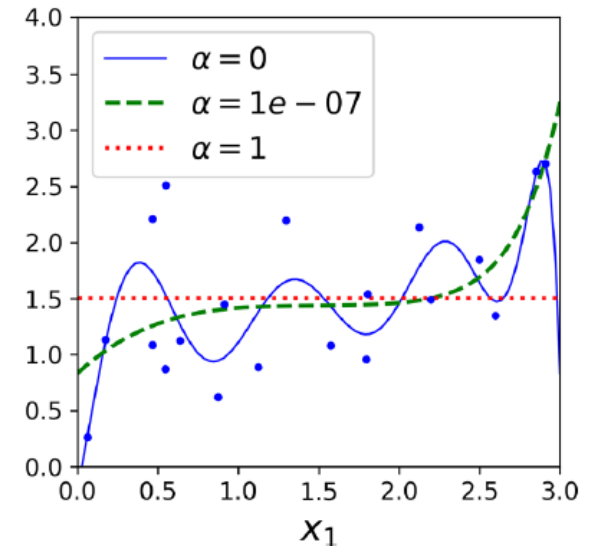
# LASSO REGRESSION

- L1 regularization

*Equation 4-10. Lasso Regression cost function*

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

- Penalize sum of absolute values
- Removes highly correlated features by nulling their coefficient
- Solution instability, impose learning schedule



Source: Géron, ISBN: 9781492032632

# ELASTIC NETS

- Middle ground between Ridge Regression and Lasso Regression

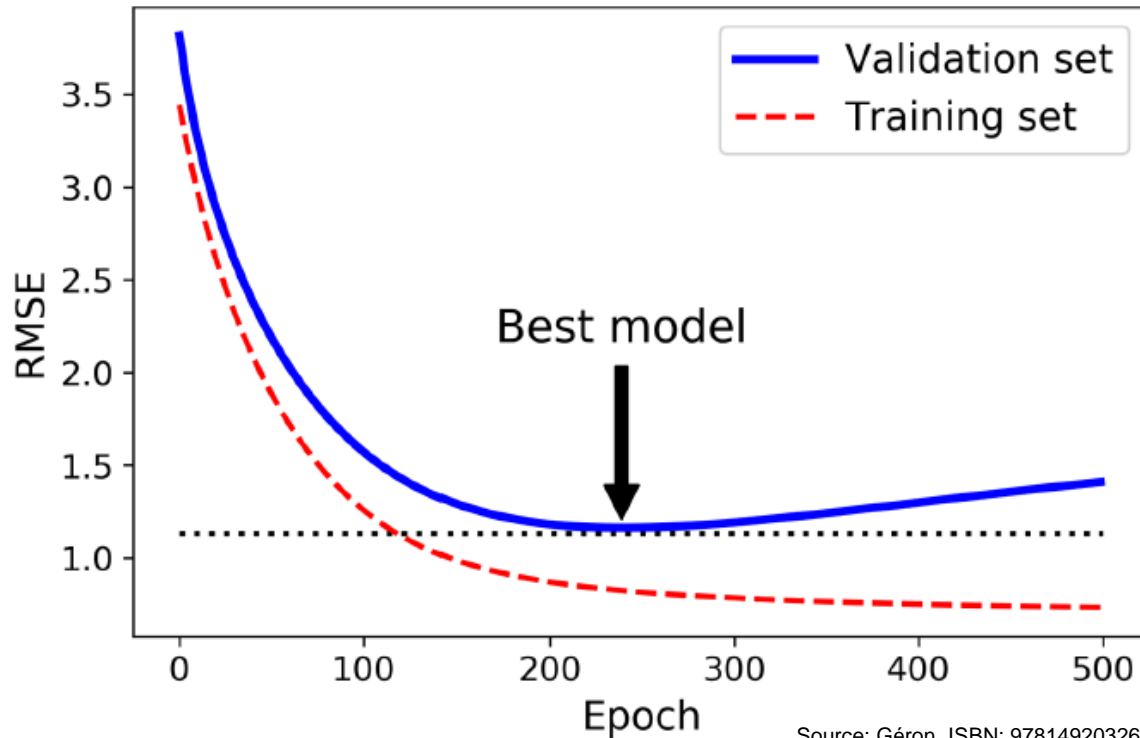
*Equation 4-12. Elastic Net cost function*

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

- Ridge is a good default, but if you suspect that only a few features are useful Elastic Net is preferred

# EARLY STOPPING

- Interpretation of learning curves



Source: Géron, ISBN: 9781492032632



# LOGISTIC REGRESSION

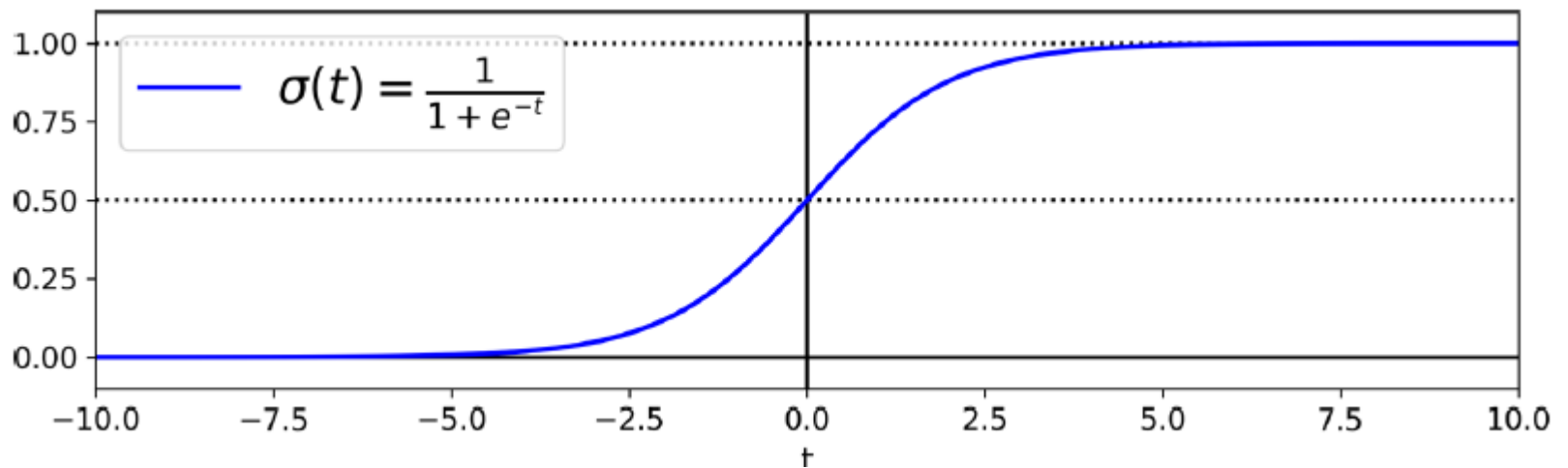
- Estimate the probability that an instance belongs to a particular class
- Binary classifier
- Baseline for evaluating more complex classification methods



Source: Mathworks, Applying Supervised Learning

# ESTIMATING PROBABILITY

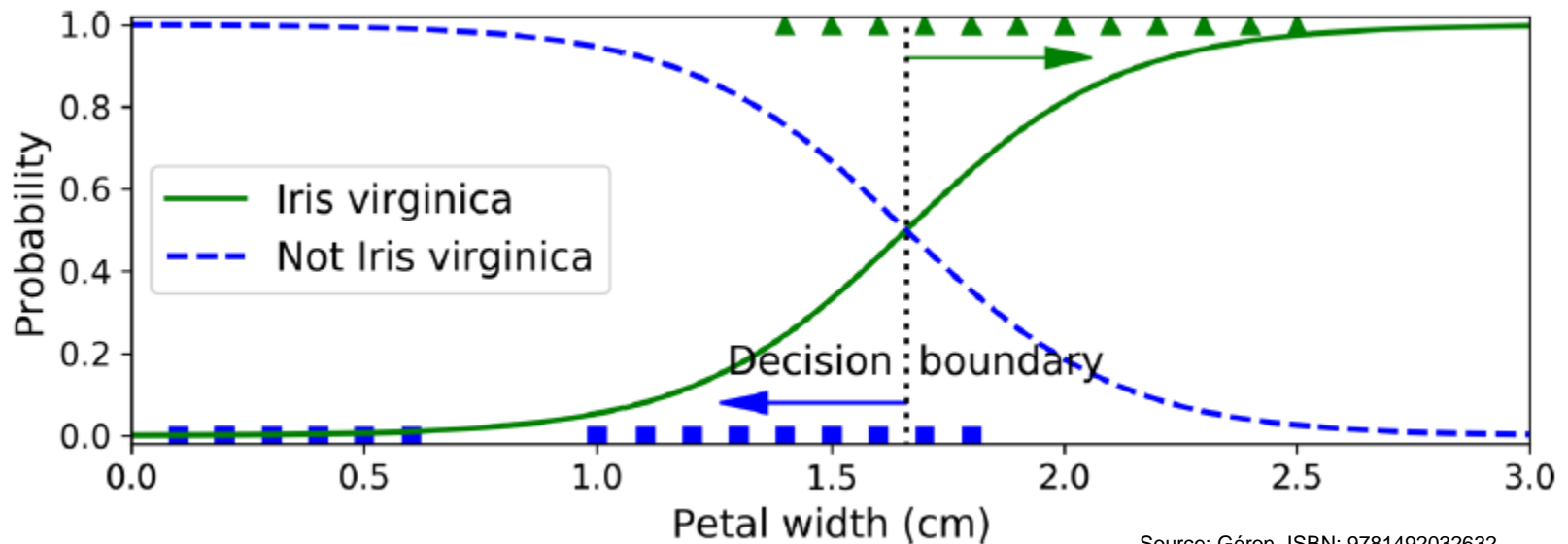
- Logistic functions maps prediction result to probability
- Sigmoid function



<https://developers.google.com/machine-learning/crash-course/logistic-regression/calculating-a-probability>

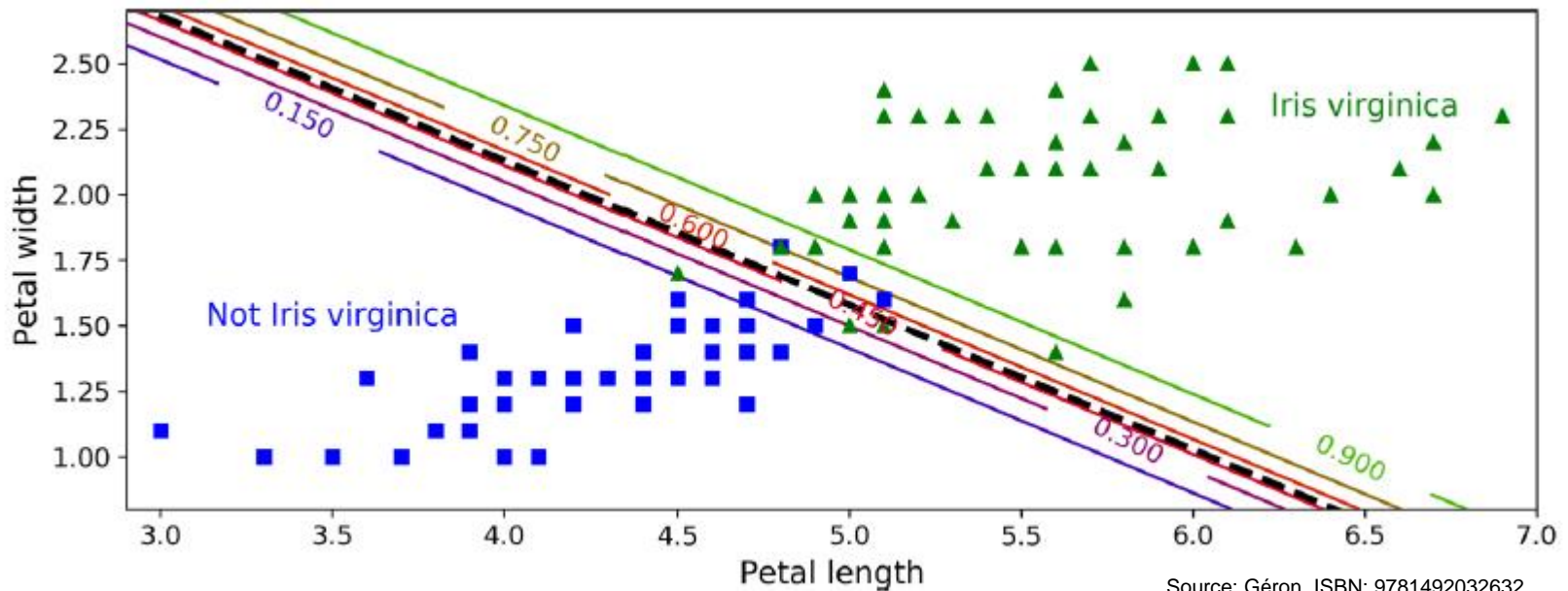
# DECISION BOUNDARY

- Aka classification threshold
- Both probabilities are equal to 50% ?



Source: Géron, ISBN: 9781492032632

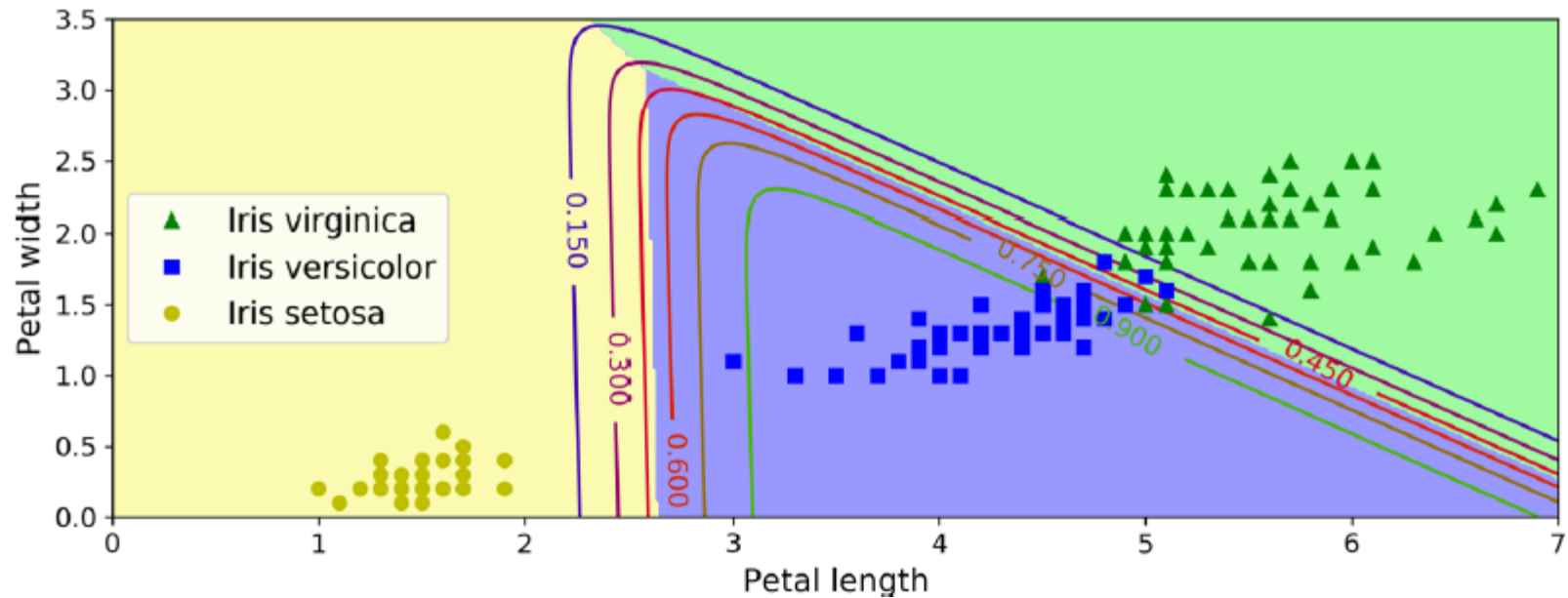
# LINEAR DECISION BOUNDARY



- Logistic Regression models can be regularized

# SOFTMAX REGRESSION

- Multinomial Logistic Regression
- Estimate probability of each class



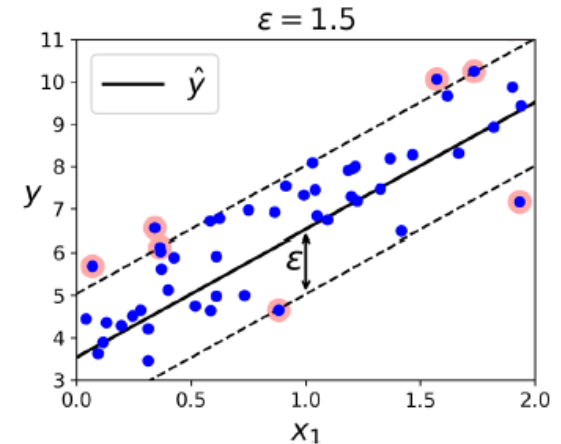
Equation 4-20. Softmax function

Source: Géron, ISBN: 9781492032632

$$\hat{p}_k = \sigma(s(\mathbf{x}))_k = \frac{\exp(s_k(\mathbf{x}))}{\sum_{j=1}^K \exp(s_j(\mathbf{x}))}$$

# CLASSIFICATION AND REGRESSION

- SVM  
Reverse the training objective



- Regression trees:  
Approximate value  
instead of predict a class

