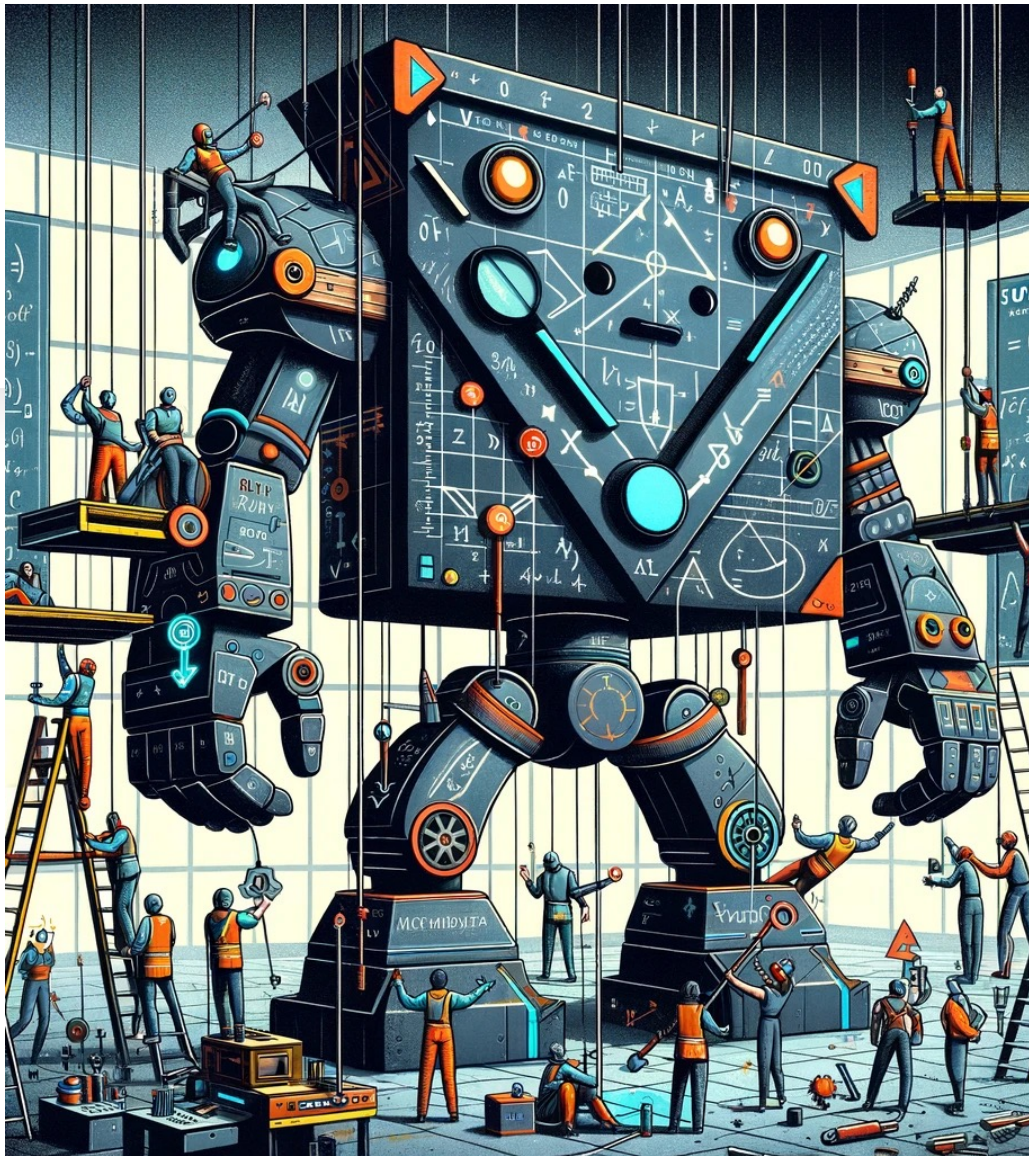
AI Study Group

Hands-on Machine Learning with Sci-kit-Learn, Keras & Tensorflow,
Geron, 3rd Edition, Chapter 5 Discussion

February 7, 2024

Chapter 5: Support Vector Machines

A. Support Vector Machines
B. Hard Margin Classification
C. Soft Margin Classification
D. Nonlinear SVM Classification
E. Polynomial Kernel – Kernel Trick
F. Gaussian RBF Kernel
G. Similarity Features
H. SVM Classes and Computational Complexity
I. SVM Regression
J. SVM Classifier Math



A. Support Vector Machines

Definition: Support Vector Machines (SVMs) are a type of machine learning algorithm that aims to find the optimal hyperplane in a high-dimensional feature space to separate data points into different classes while maximizing the margin between the classes. SVMs work by identifying support vectors, which are the data points closest to the decision boundary and use them to determine the best separating hyperplane.

SVMs are widely used in various applications, such as image classification, text classification, and anomaly detection, due to their ability to handle high-dimensional data and their effectiveness in finding robust decision boundaries.

A. Support Vector Machines

Key concepts in SVMs include:

- Hyperplane: A hyperplane is a decision boundary that separates data points of different classes. In a binary classification scenario, it is the line (in 2D), plane (in 3D), or higher-dimensional hyperplane that maximizes the margin between classes.

- Margin: The margin is the distance between the hyperplane and the nearest data points from each class. SVMs aim to find the hyperplane that maximizes this margin, as it is believed to lead to better generalization to unseen data.

- Support Vectors: Support vectors are the data points that lie closest to the decision boundary or hyperplane (i.e., the edge of the ”street”). These points have a significant influence on determining the position and orientation of the hyperplane.

- Kernel Trick: SVMs can handle complex and non-linear data by mapping the original feature space into a higher-dimensional space using a kernel function (e.g., polynomial or radial basis function). In this higher-dimensional space, SVMs find a hyperplane that separates the data effectively.

B. Hard Margin Classification

- Hard margin classification is a specific scenario within Support Vector Machines (SVMs) where the algorithm aims to find a hyperplane that perfectly separates two classes of data in a high-dimensional feature space while maximizing the margin between them.

- It seeks to create a decision boundary in such a way that all training data points are correctly classified, and there are no data points located within the margin or on the wrong side of the hyperplane (see Figure 5-1).

- SVMs can be sensitive to features' scales (see Figure 5-2).

B. Hard Margin Classification

Figure 5-1
(p. 176)

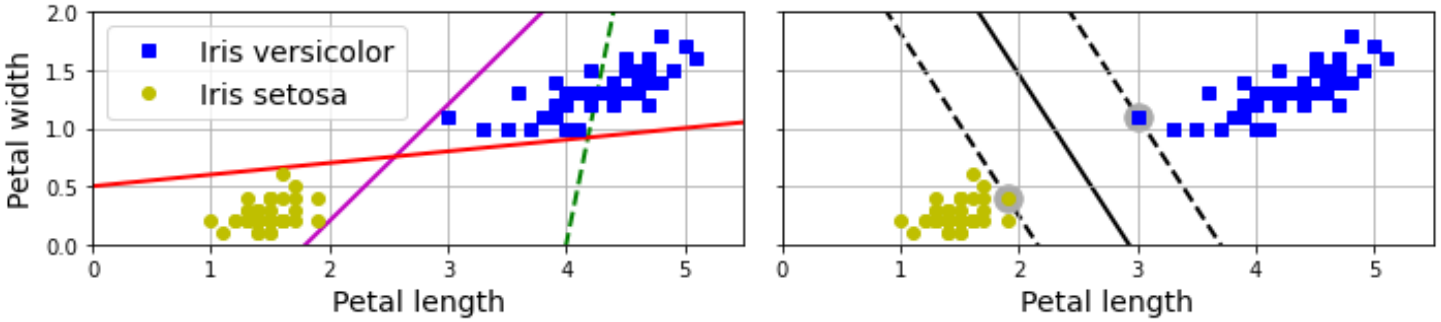
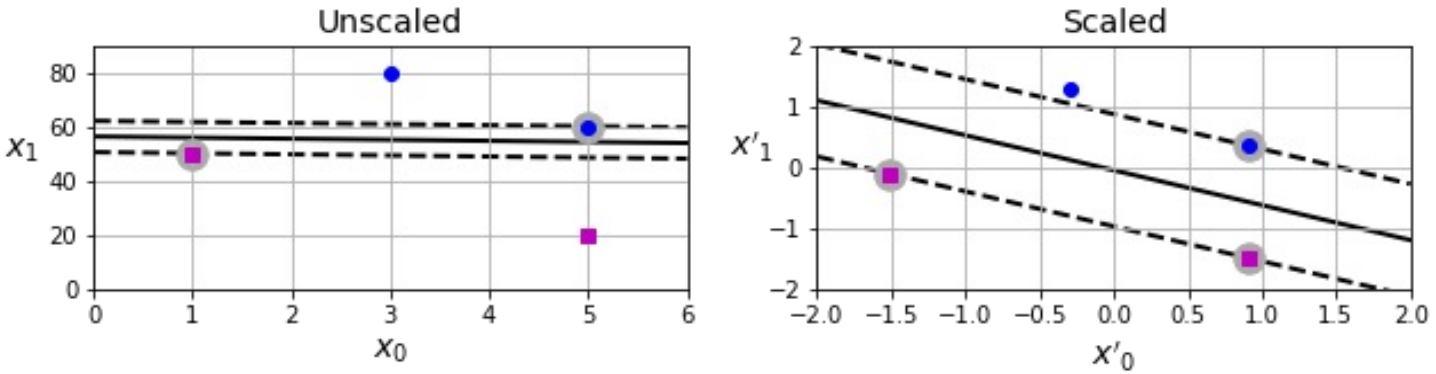


Figure 5-2
(p. 176)



C. Soft Margin Classification

- Soft margin classification is a variant of Support Vector Machines (SVMs) used for classification tasks, especially when the data is not perfectly linearly separable or contains outliers. Unlike hard margin classification, which seeks a hyperplane that perfectly separates the training data, soft margin classification allows for a certain degree of margin violations and misclassifications by introducing a regularization parameter (C). Specifically, if one detects overfitting in soft margin classification, one can regularize the model by reducing the hyperparameter C (and make the "street" wider).

- The objective is to find a hyperplane that maximizes the margin between classes while tolerating some classification errors and data points within the margin. It attempts also resolve the two issues with hard margin classification: linear separability and sensitivity to outliers.

- **Question #1**: wouldn't one opt for soft margin classification (versus hard margin classification) in most cases? In which cases would one opt for hard margin classification?
- **Question #2**: what is the intuition behind reducing the hyperparameter C, making the margin wider, and regularizing the model?

C. Soft Margin Classification

Figure 5-3
(p. 177)

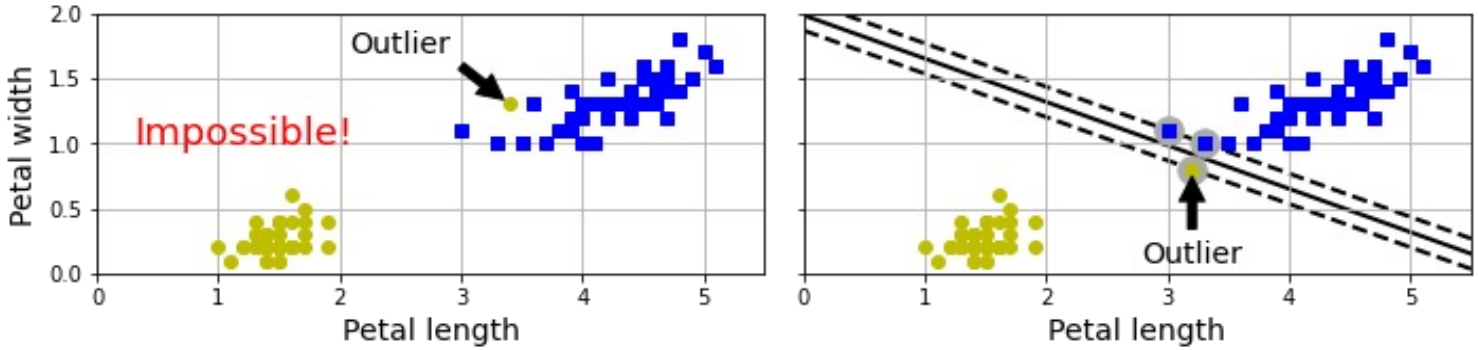
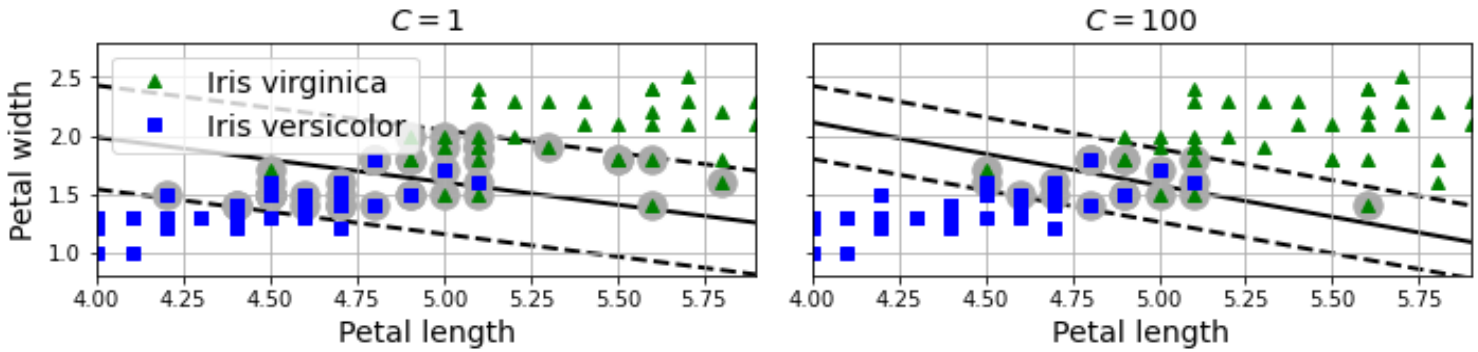


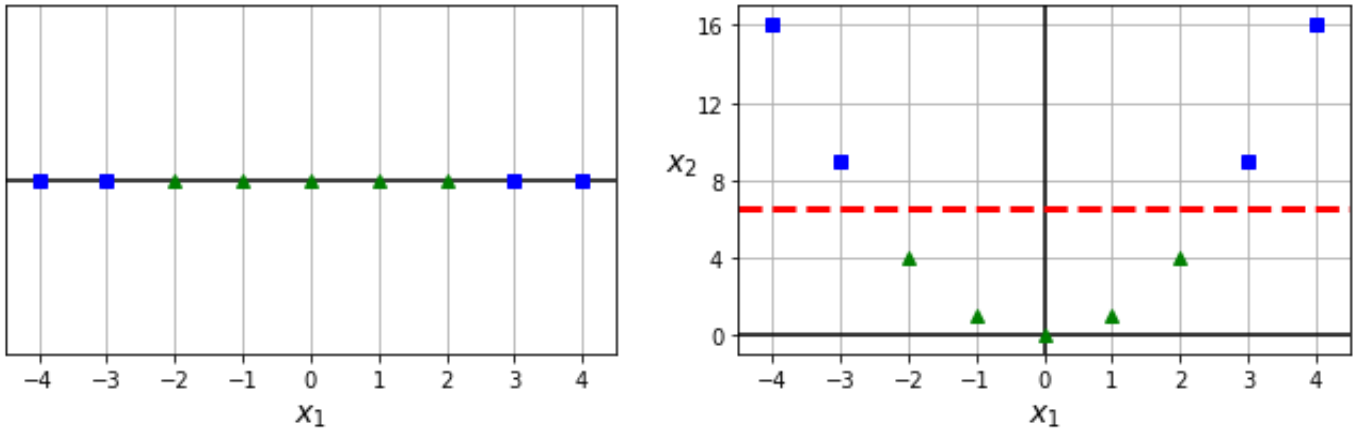
Figure 5-4
(p. 177)



D. Nonlinear SVM Classification

- Many data sets are not linearly separable. One approach to address this issue is to add polynomial features.

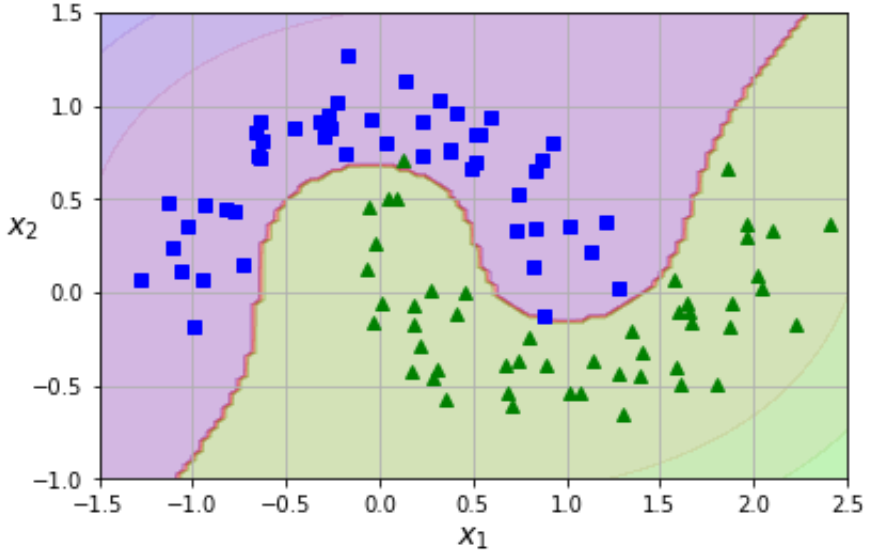
Figure 5-5
(p. 179)



D. Nonlinear SVM Classification

- To implement this approach, one can use the PolynomialFeatures transformer followed by a StandardScaler and a LinearSVC classifier to derive Figure 5-6 below.

Figure 5-6
(p. 179)



E. Polynomial Kernel - Kernel Trick

Kernels are a set of functions used to transform data from a lower dimension to a higher dimension and manipulate data using a dot product at a higher dimension. The kernel trick makes it possible to derive the same results for a polynomial SVM without having to add polynomial features, thus saving computation time.

The ”trick” is not transform every x into f(x), but rather use the dot product of x and x’:

$$K(x, x') = f(x)^T f(x')$$

Question #3: how was the kernel trick discovered?

E. Polynomial Kernel – Kernel Trick

Linear kernel:

$f(x) = x$

$K(x, x') = x^T x'$

Second-order polynomial kernel:

$f(x) = (x_1, x_2, x_1 x_2, x_1^2, x_2^2)$

$K(x, x') = (1 + x^T x')^2$

Gaussian RBF kernel:

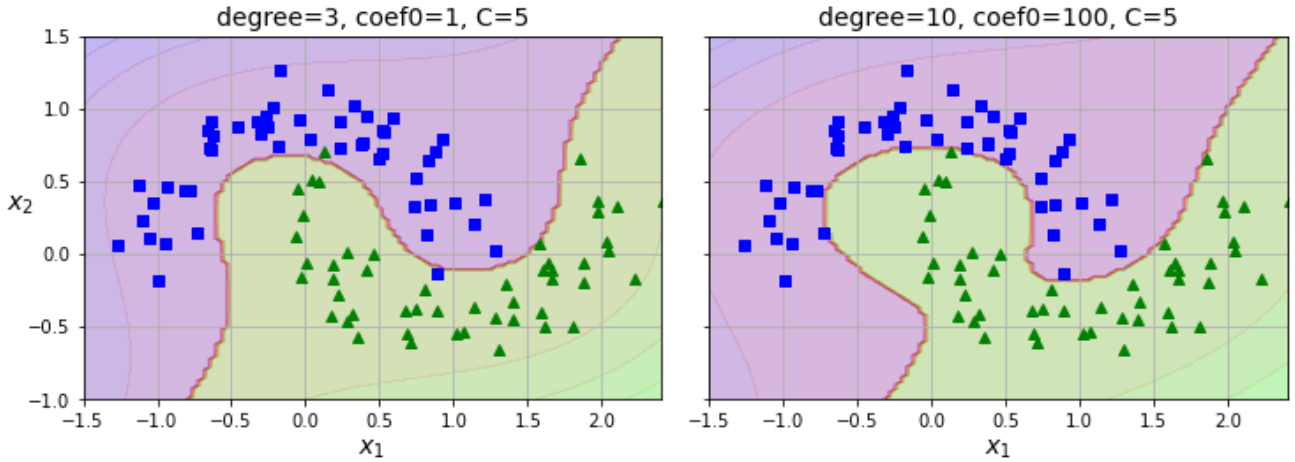
$f(x)$ = infinite dimensional (undefined)

$K(x, x') = \exp(-\gamma \|x - x'\|^2)$

E. Polynomial Kernel - Kernel Trick

In Figure 5-7 below, a polynomial kernel is being applied. Python has in its SVC class the "coef" hyperparameter to control how much a model is influenced by low-degree and high-degree terms.

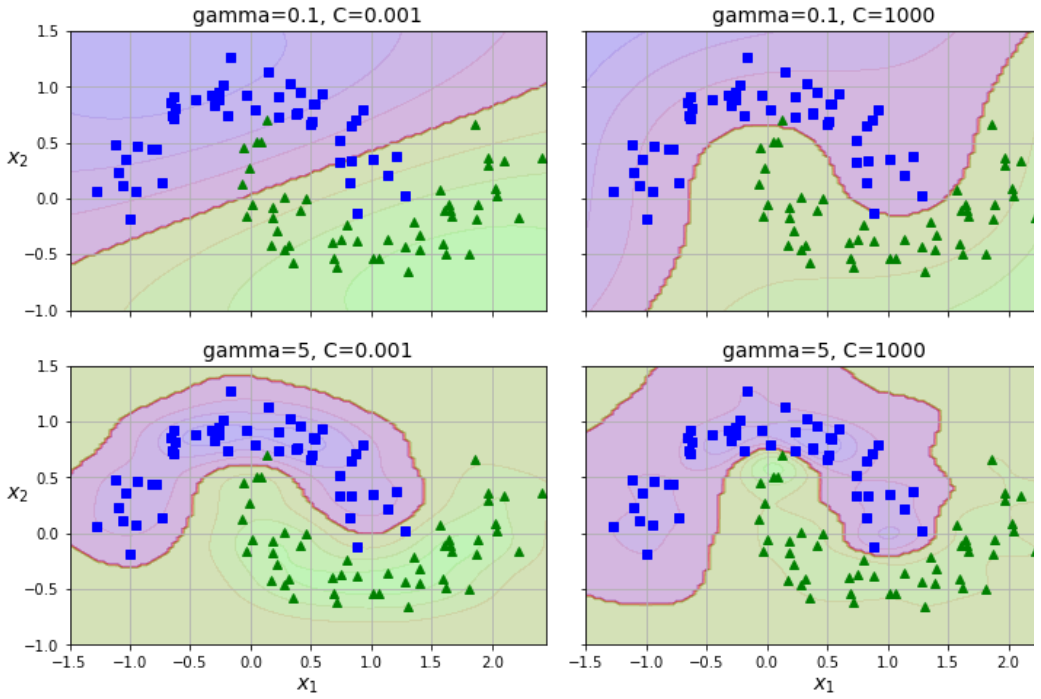
Figure 5-7
(p. 180)



F. Gaussian Radial Basis Function (RBF) Kernel

The Gaussian RBF kernel may be a useful kernel to apply. When invoking the Gaussian RBF kernel using the SVC class in Python, the gamma hyperparameter can be adjusted: if the model is overfitting, gamma should be reduced; when the model is underfitting, gamma should be increased (see Figure 5-9, p. 182).

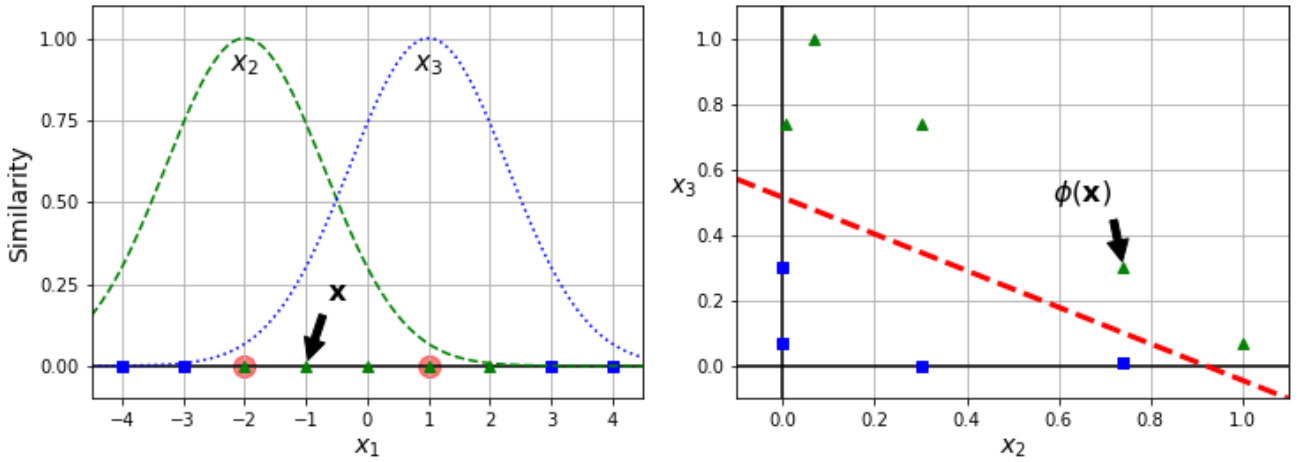
Question #4: do we have clear criteria for which cases to use the Gaussian RBF kernel versus other types of kernels (e.g., linear, polynomial)?



G. Similarity Features

A similarity function can add features to address nonlinear problems. A similarity function measures how much each instance resembles a particular landmark. The example below (Figure 5-8) depicts the landmarks in orange and uses similarity features using the Gaussian RBF. But similarity features may be computationally expensive to compute the additional features.

Figure 5-8
(p. 181)



Question #5: If landmarks are created for each instance for dataset to find the optimal landmarks, then this approach seems computationally intensive. What are advantages of this approach?

H. SVM Classes and Computational Complexity

Class	Time Complexity	Out-of-core Support	Scaling Required	Kernel Trick
LinearSVC	O(m x n)	No	Yes	No
SVC	$O(m^2 \times n)$ to $O(m^3 \times n)$	No	Yes	Yes
SGDClassifier	O(m x n)	Yes	Yes	No

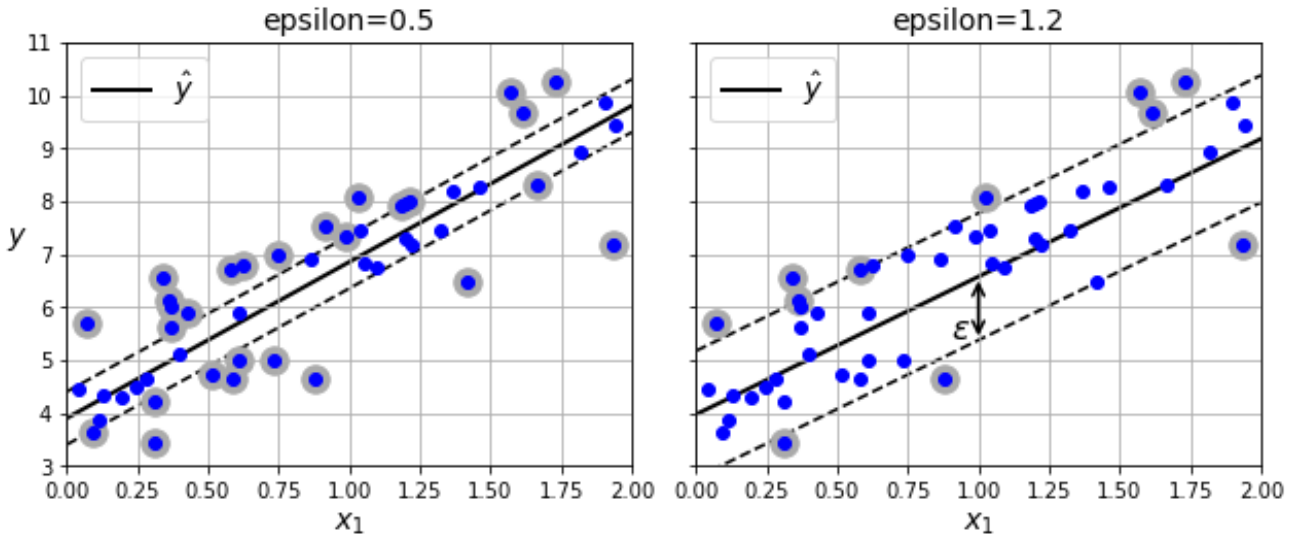
Question #6: What is "out-of-core support"?

Note: SGDClassifier was discussed in Chapter 3 ("Classification"). See page 106.

I. SVM Regression

SVMs can also be used for regression analysis. However, in this case, the SVM regression will attempt to fit as many instances as possible on the "street" while limiting margin violations. Note that reducing epsilon increases the number of support vectors, which regularizes the model.

Figure 5-10
(p. 184)



J. SVM Classifier Math

- Hard and soft margin problems of SVM are both convex quadratic optimization problems. They are also called quadratic programming (QP) problems.
- SVM can also be viewed as a dual problem.
- Kernelized SVMs:

Second-degree polynomial mapping

$$\varphi(x) = \varphi\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix}$$

J. SVM Classifier Math

Kernel trick for second-degree polynomial mapping

$$\varphi(a)^T\varphi(b)=\begin{pmatrix} a_1^2 \\ \sqrt{2}a_1a_2 \\ a_2^2 \end{pmatrix}\begin{pmatrix} b_1^2 \\ \sqrt{2}b_1b_2 \\ b_2^2 \end{pmatrix}=a_1^2b_1^2+2a_1b_1a_2b_2+ a_2^2b_2^2$$

$$= (a_1b_1 + a_2\, b_2)^2 = \left(\begin{pmatrix} a_1 \\ a_2 \end{pmatrix}^T \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}\right)^2 = (a^Tb)^2$$