

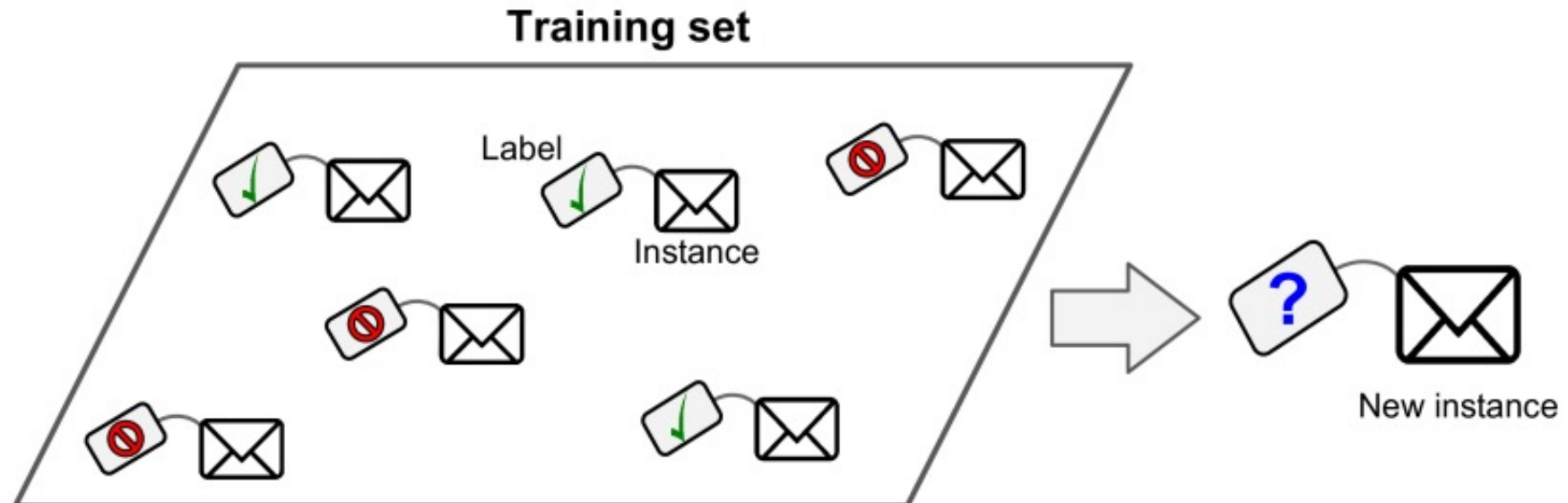
Supervised learning

DESU IA4Health

Info-GRAM

Supervised Learning

- In supervised learning, the training data you feed to the algorithm includes the desired solutions, called labels.
- Classification : Ex. Spam filter, it is trained with many example emails along with their class (spam or ham), and it must learn how to classify new emails



Supervised learning STEPS

1. We are going to split our data set in :
 - Training data set
 - Test data set
 - Validation data set
2. We are going to train our algorithm with the training data set
 - The algorithm will learn from the training data
3. We are going to test our algorithm with the test data
 - Model performance measures on test and validation datasets
4. We ensure that our algorithm generalizes well to other data set never seen before.

Supervised Learning

- Regression: To predict a target numeric value, such as the price of a car, given a set of features



Most important Supervised Learning algorithms

- Support Vector Machines (SVMs)
- Linear Regression
- Logistic Regression
- k-Nearest Neighbors
- Decision Trees and Random Forests
- Neural network

Linear Regression

- A linear model makes a prediction by simply computing a weighted sum of the input features, plus a constant called the bias term (also called the intercept term)

$$\hat{y} = h_{\theta}(\mathbf{x}) = \boldsymbol{\theta} \cdot \mathbf{x}$$

- $\boldsymbol{\theta}$ is the model's *parameter vector*, containing the bias term θ_0 and the feature weights θ_1 to θ_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 $\theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$.
- h_{θ} is the hypothesis function, using the model parameters $\boldsymbol{\theta}$.

Performance Linear regression

- Root Mean Square Error (RMSE)
- To train a Linear Regression model, you need to find the value of θ that minimizes the RMSE. In practice, it is simpler to minimize the Mean Square Error (MSE) than the RMSE, and it leads to the same result.

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

Optimisation algorithms for Linear Regression

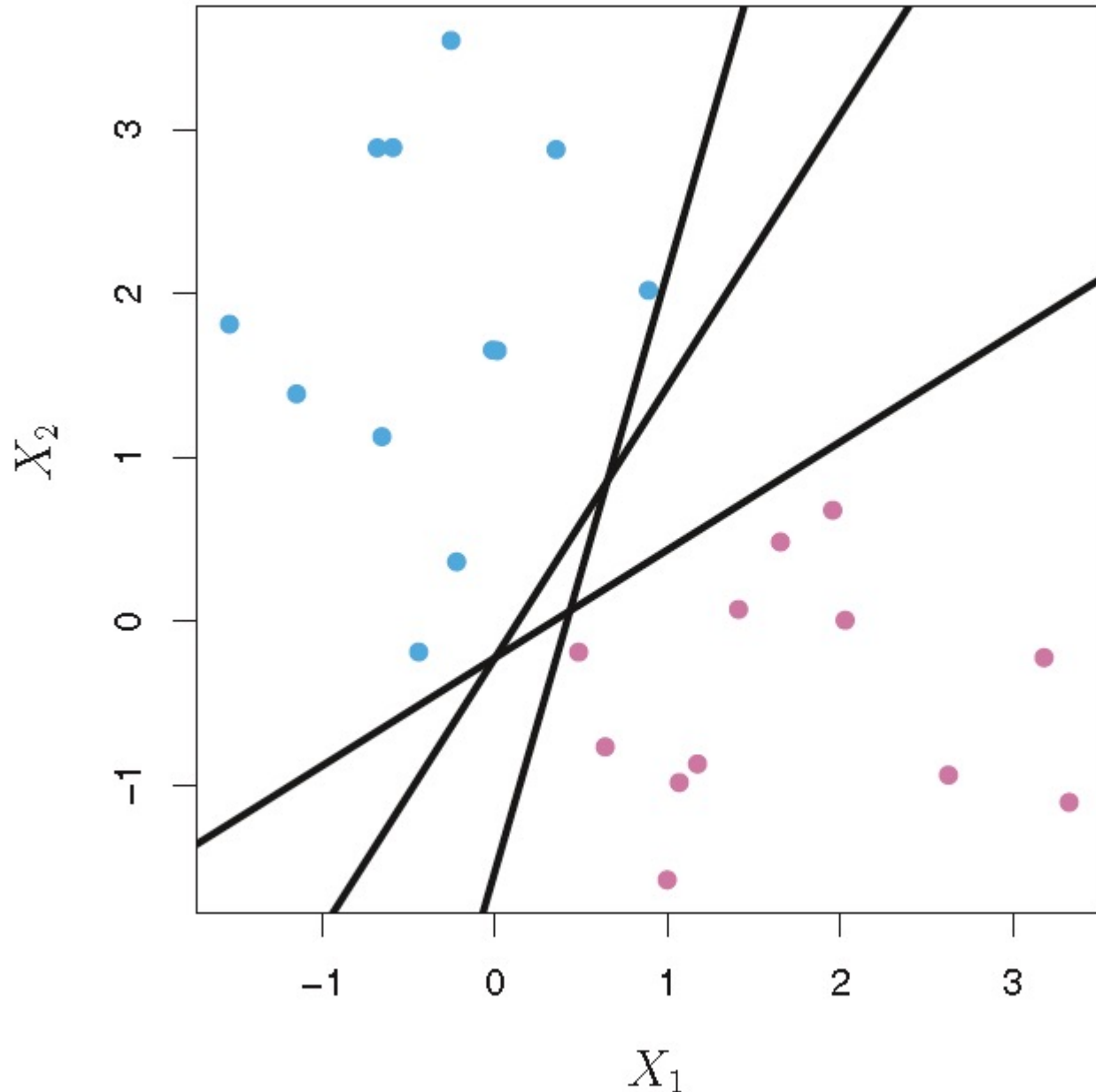
Algorithm	Large m	Out-of-core support	Large n	Hyperparams	Scaling required	Scikit-Learn
Normal Equation	Fast	No	Slow	0	No	n/a
SVD	Fast	No	Slow	0	No	LinearRegression
Batch GD	Slow	No	Fast	2	Yes	SGDRegressor
Stochastic GD	Fast	Yes	Fast	≥ 2	Yes	SGDRegressor
Mini-batch GD	Fast	Yes	Fast	≥ 2	Yes	SGDRegressor

Support Vector Machines

- A Support Vector Machine (SVM) is a versatile Machine Learning model, capable of performing linear or nonlinear classification, regression, and even outlier detection.
- SVMs are particularly well suited for classification of complex but small or medium-sized datasets.

SVM

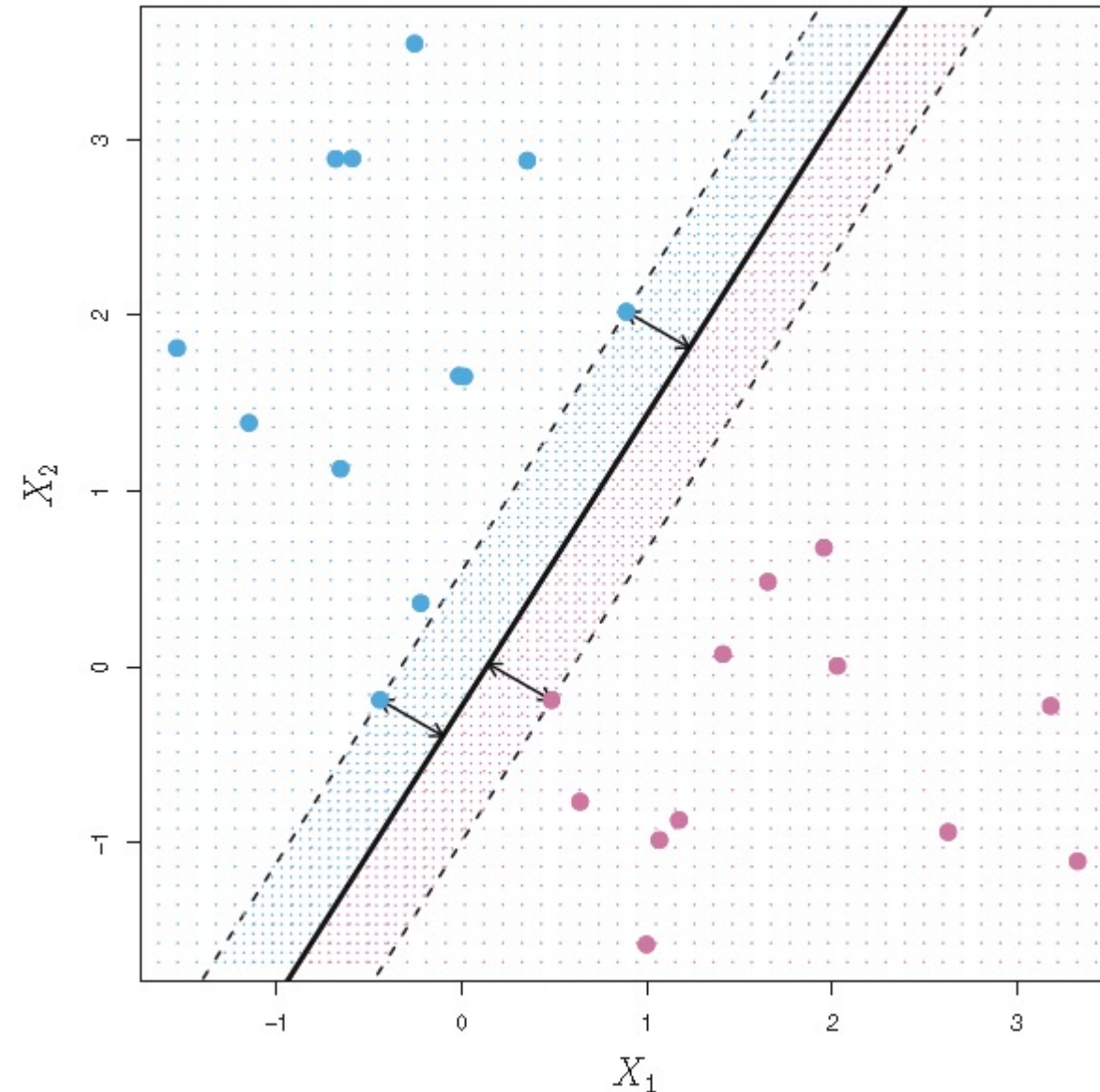
- Support vector classification aims to classify data points into various target classes. If the classes in our training data can be separated by a line or some boundary, then we can just classify the data depending on what side of this decision boundary the data lies on.
- In the following 2-d example, we can separate the data with any of the three lines, and then just assign classes based on whether the observation lies above or below the line. The data are 2-dimensional vectors specified by the features X_1 and X_2 with class labels as either $y = 1$ (blue) or $y = 0$ (red).



Training a SVM

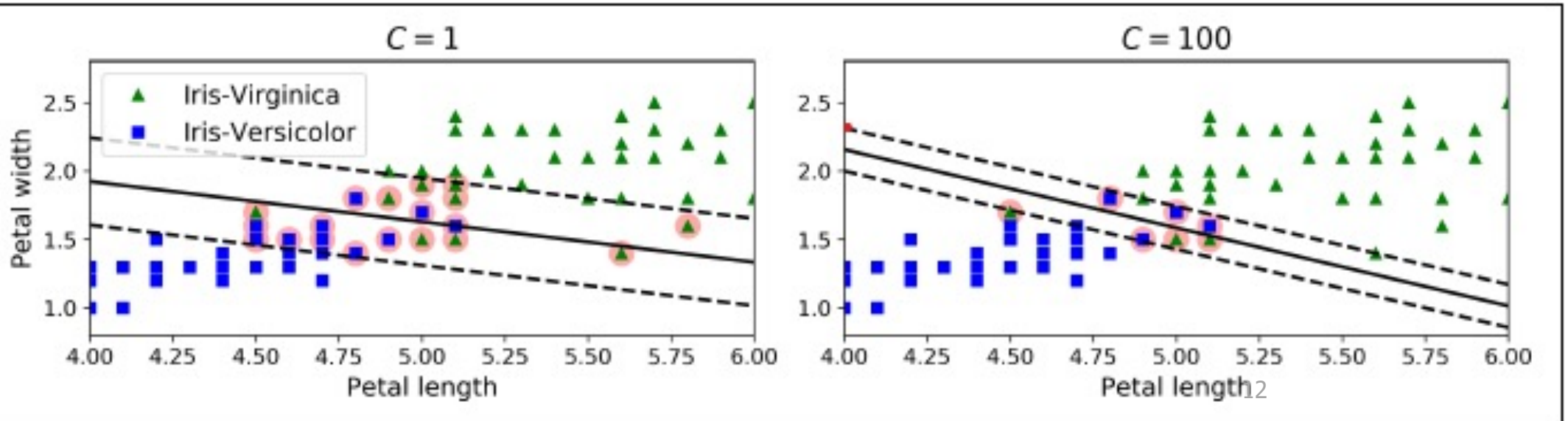
Training a linear support vector classifier is an optimization problem.

- We **maximize the *margin*** : the distance separating the closest pair of data points belonging to opposite classes.
- These points **are called the support vectors**, because they are the data observations that “support”, or determine, the decision boundary.
- To train a support vector classifier, we find the ***maximal margin hyperplane***, **or *optimal separating hyperplane***, which optimally separates the two classes in order to generalize to new data and make accurate classification predictions.



SVM Soft Margin Classification

- **Hard margin classification** : we strictly impose that all instances be off the street and on the right side.
 - Works only if data is learnly separable
 - Very sensitive to outliers
- **Soft margin classification**: The objective is to find a good balance between keeping the street as large as possible and limiting the *margin violations* .
- Margin width stablished with the C parameter : a smaller C value leads to a wider street but more margin violations

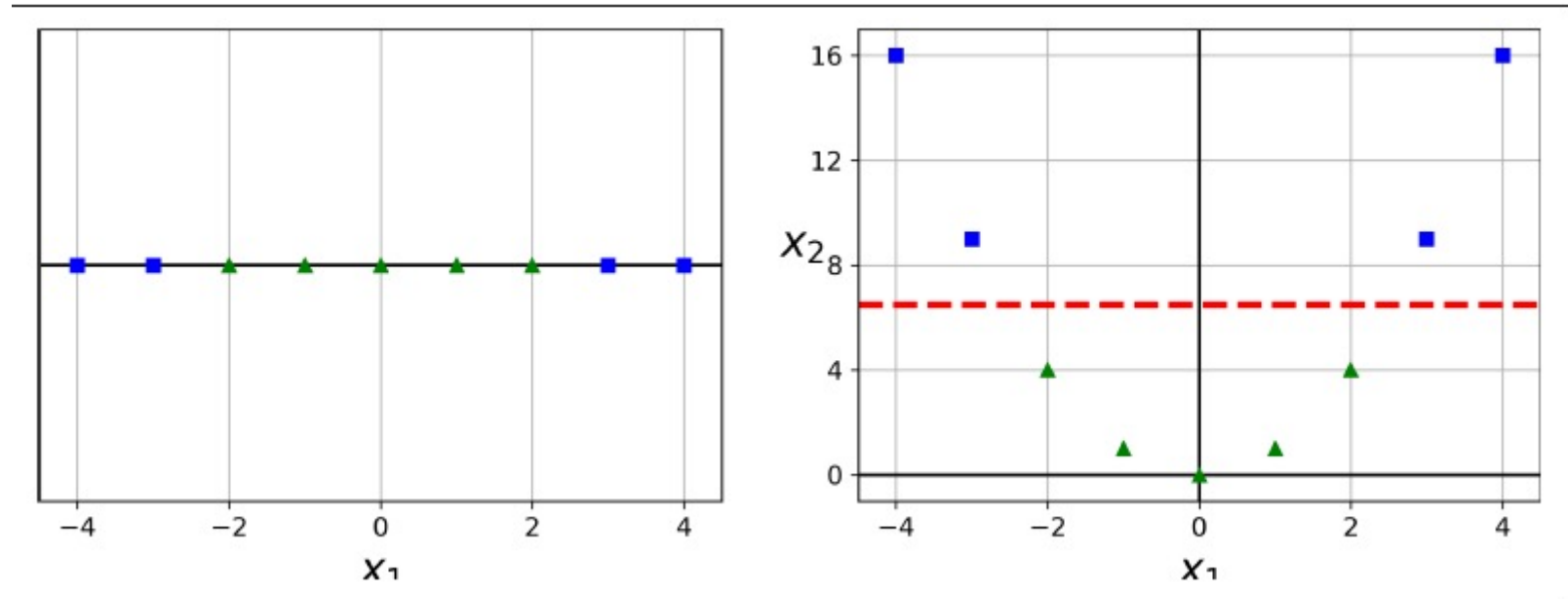


SVM

- SVM is sensitive to features scales so it is recommended to scale the data before training the model.
- The LinearSVC class regularizes the bias term, so you should center the training set first by subtracting its mean. This is automatic if you scale the data using the StandardScaler

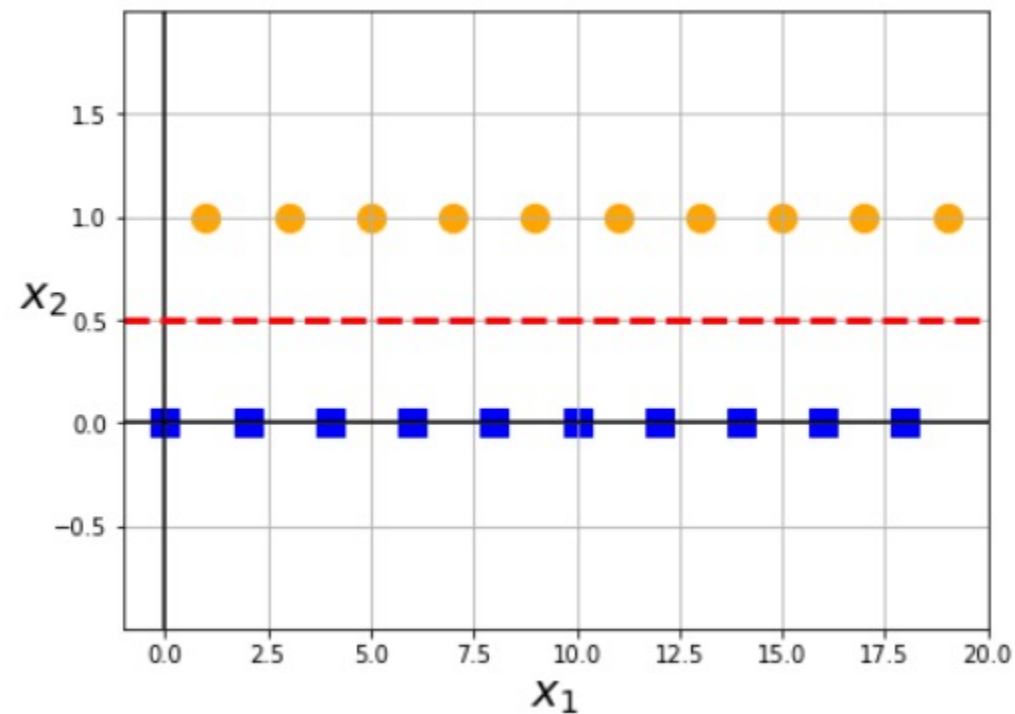
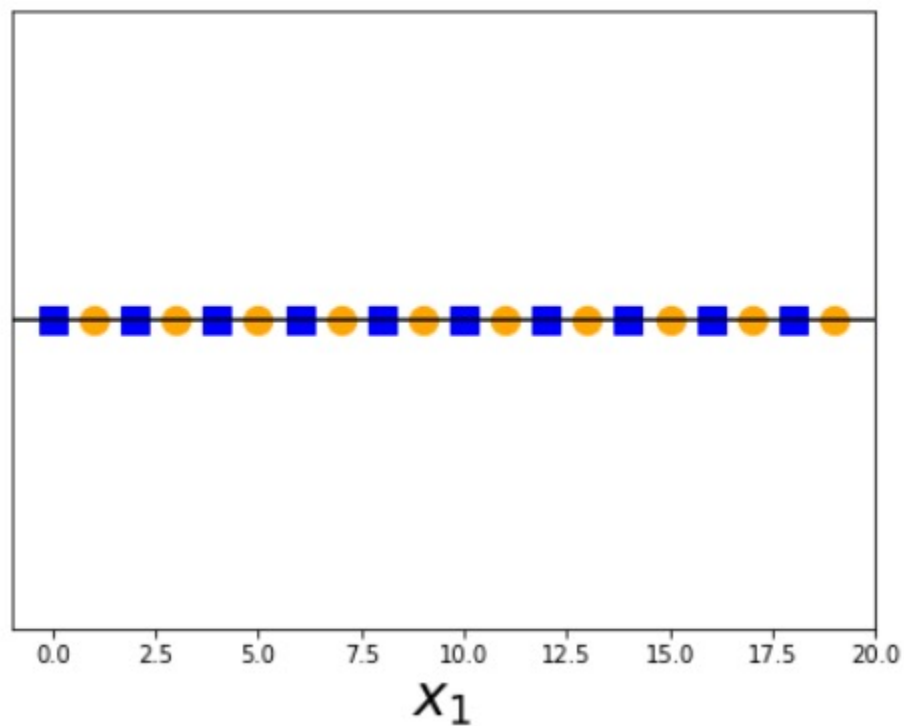
Nonlinear SVM Classification

- Although linear SVM classifiers are efficient and work surprisingly well in many cases, many datasets are not even close to being linearly separable.
- One approach to handling nonlinear datasets is to add more features, such as polynomial features

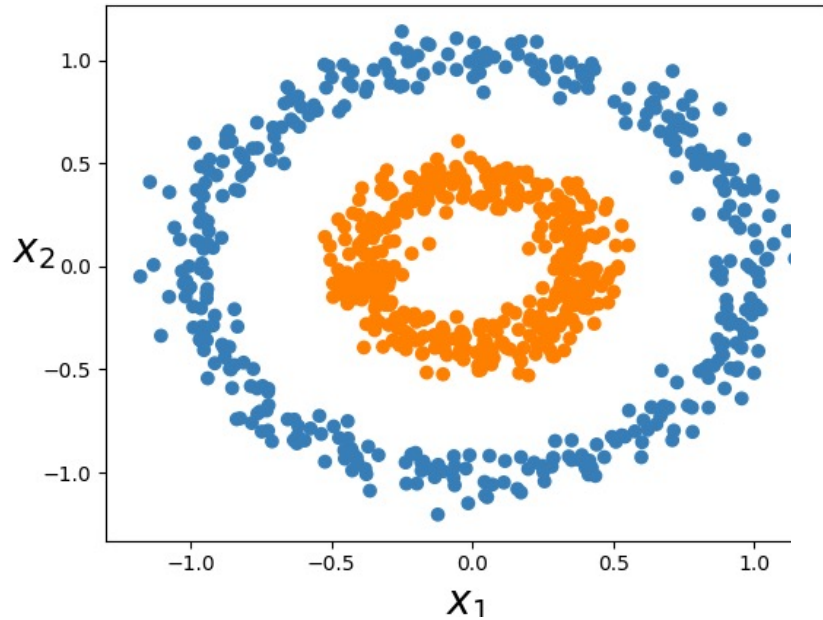


Data transformations in non linearly separable data

- Transformation $\phi(x) = x \bmod 2$

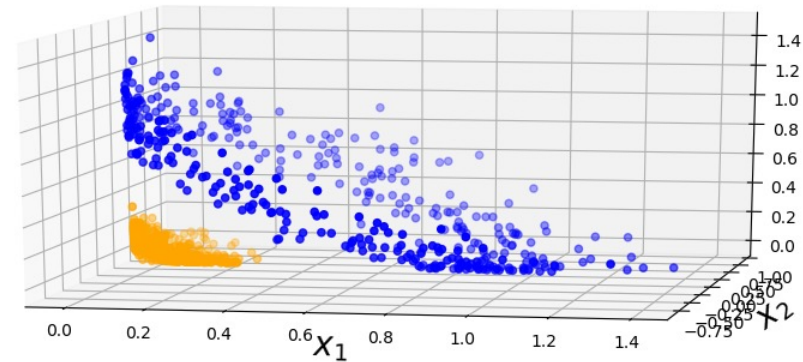


Data transformations in non linearly separable data



Equation 5-8. Second-degree polynomial mapping

$$\phi(\mathbf{x}) = \phi\left(\begin{pmatrix} x_1 \\ x_2 \end{pmatrix}\right) = \begin{pmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{pmatrix}$$



SVM Polynomial Kernel

- Adding polynomial features is simple to implement and can work great with all sorts of Machine Learning algorithms (not just SVMs).

A low polynomial degree cannot deal with very complex datasets,

A high polynomial degree creates a huge number of features, making the model too slow.

The kernel Trick

- Kernel methods represent the data only through a set of pairwise similarity comparisons between the original data observations \mathbf{x} (with the original coordinates in the lower dimensional space), instead of explicitly applying the transformations $\phi(\mathbf{x})$ and representing the data by these transformed coordinates in the higher dimensional feature space.
- They make possible to obtain a similar result as if many features have been added, without actually having to add them.

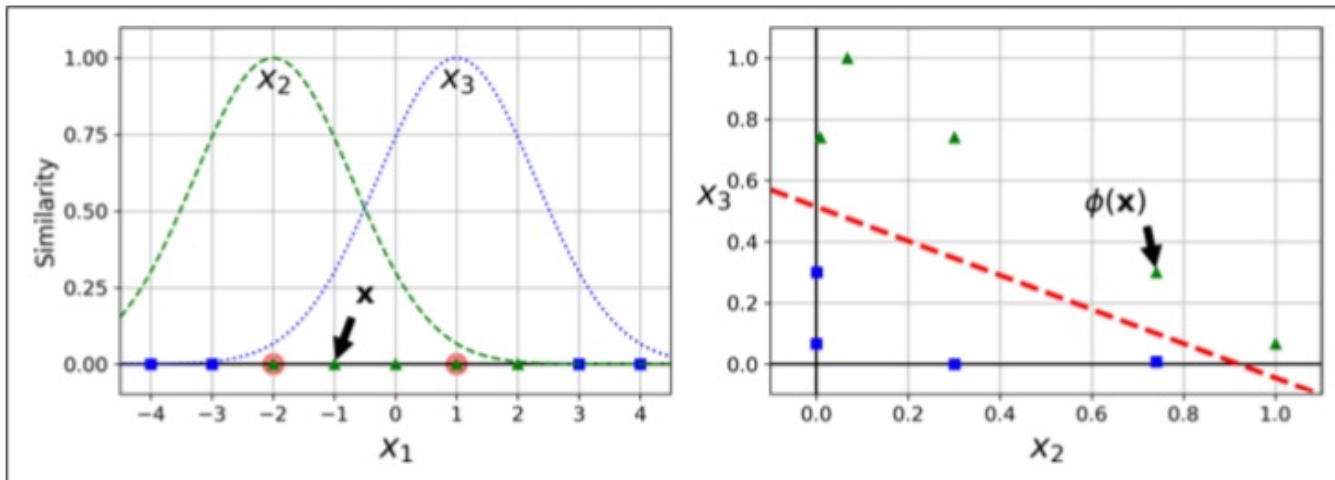
SVM Polynomial Kernel

Hyperparameters :

- Polynomial degree
- C
- The hyperparameter `coef0` controls how much the model is influenced by high-degree polynomials versus low-degree polynomials

Adding similarity features : RBF Kernel

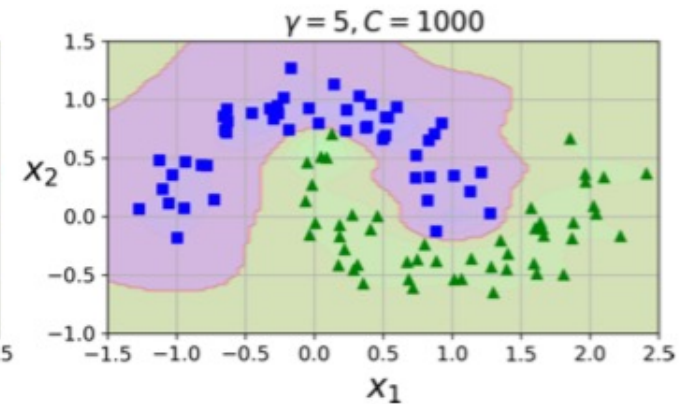
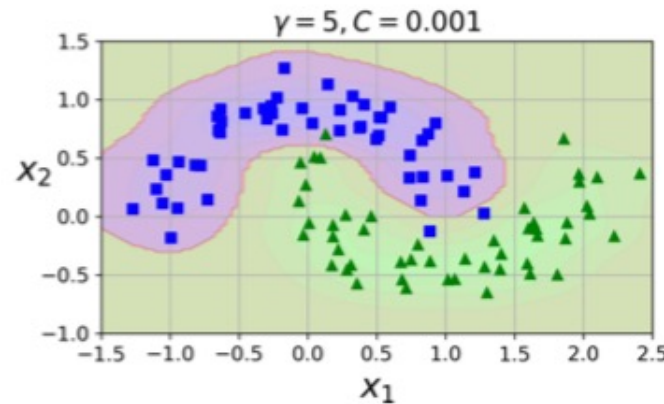
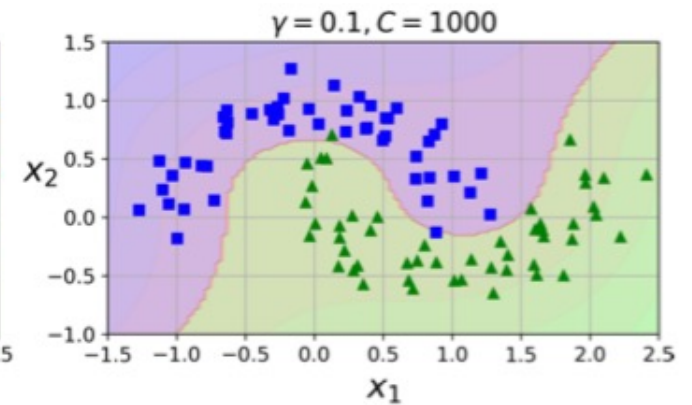
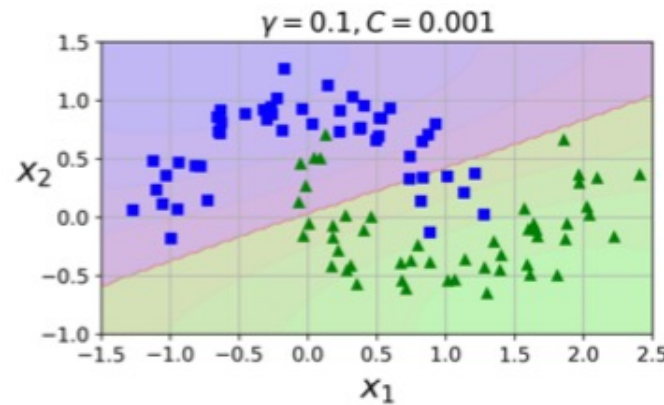
- Add features computed using a *similarity function* that measures how much each instance resembles a particular *landmark*.
- Similarity function : Gaussian *Radial Basis Function* (*RBF*)
 - bell-shaped function varying from 0 (very far away from the landmark) to 1 (at the landmark).



RBF: Hyperparameter tuning

Hyperparameters : γ and C

- Increasing γ makes the bell-shape curve narrower, and as a result each instance's range of influence is smaller. Result = Irregular decision boundary.
- A small γ value makes the bell-shaped curve wider, so instances have a larger range of influence. Result : smoother decision boundary.
- γ acts like a regularization hyperparameter:
 - if model is overfitting reduce it,
 - If it is underfitting, increase it.



Which kernel do I choose ?

- Rule of thumb :
 - Always try the linear kernel first, especially if the training set is very large or if it has plenty of features.
 - If the training set is not too large, you should try the Gaussian RBF kernel as well; it works well in most cases.
- Experiment with other kernels using cross-validation and grid search.