LECTURER: TAI LE QUY

# MACHINE LEARNING

# SUPERVISED LEARNING

UNIT 4

# SUPPORT VECTOR MACHINES

## STUDY GOALS

- Explain the **concept** of **large margin classification**.
- Conceptualize a large margin classifier with **support vector machines**.
- Explain and make use of the **kernel trick**.
- Apply support vector machines with the use of **Python**.

**EXPLAIN SIMPLY**

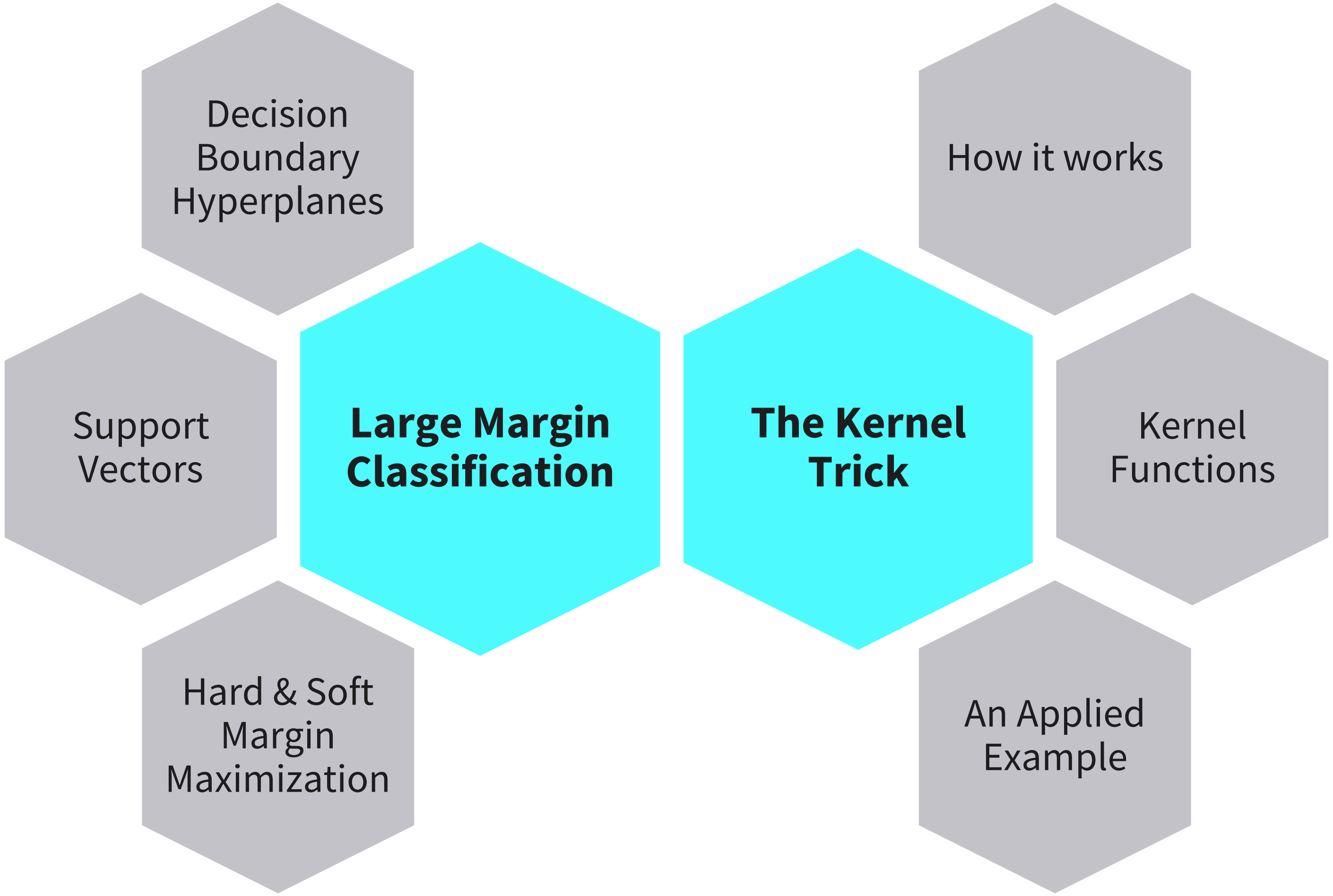
1. Imagine data with **10 features**. **How many dimensions** will the **decision boundary** have?

2. Explain in one sentence what the kernel trick is.

3. Explain if Support Vector Machines can be used to solve **classification or regression** tasks.

## UNIT CONTENT

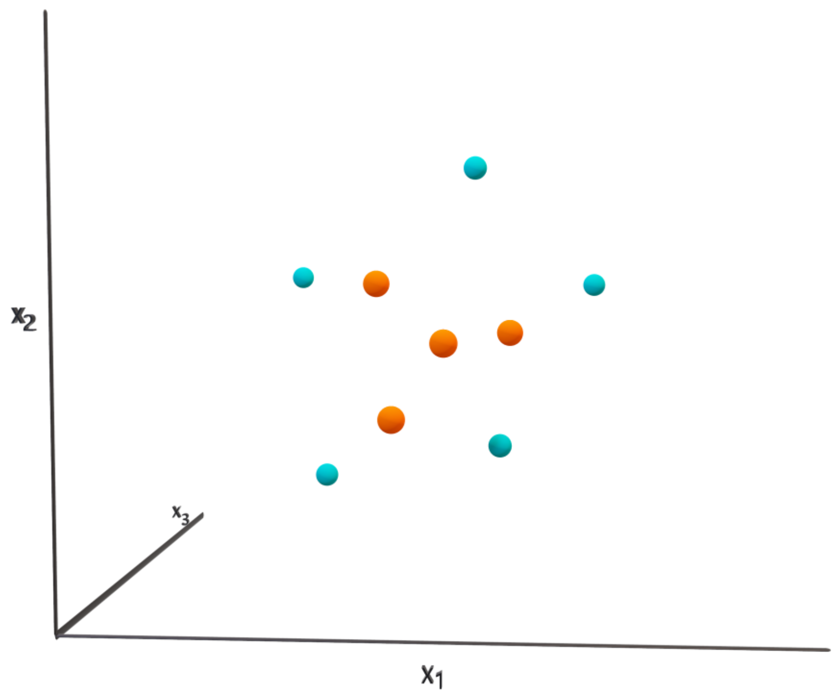
**Img. 1:** Support vector machines




Source of image 1: Christian Müller-Kett, 2021.

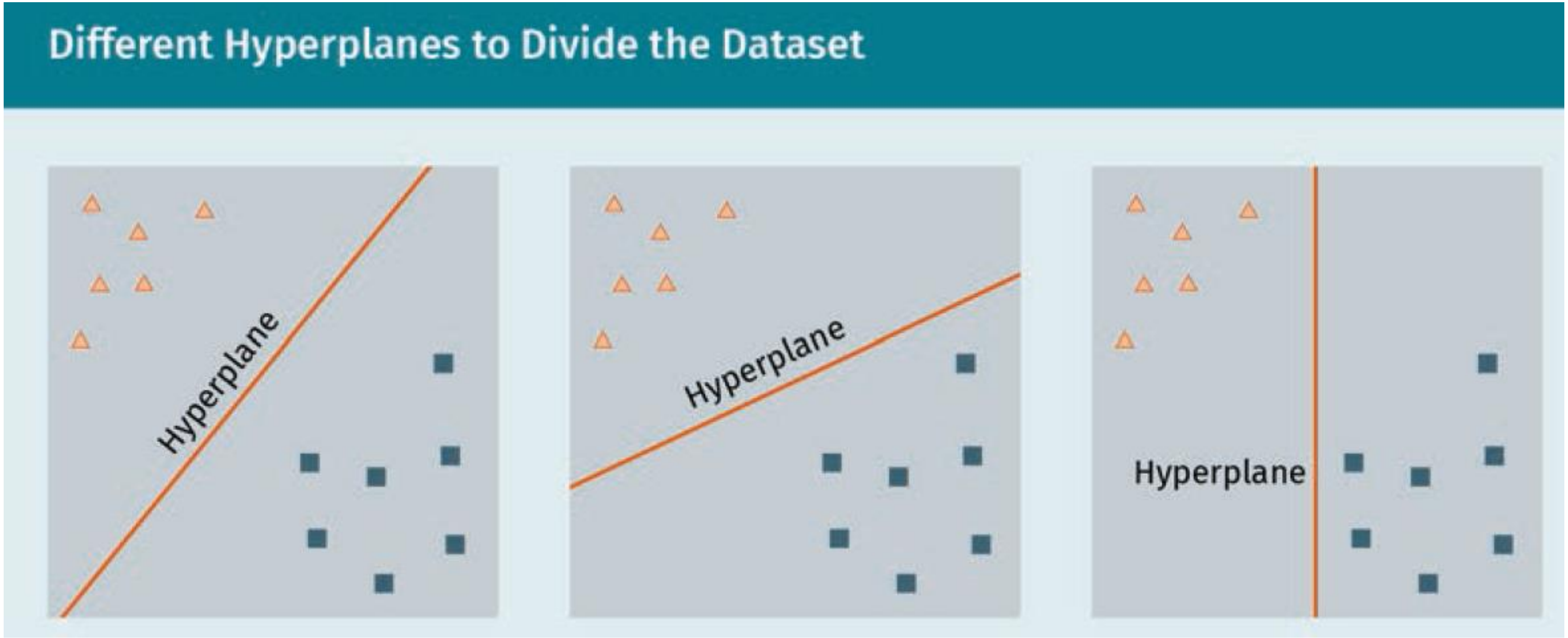
- **n-dimensional feature space**
- **n-1-dimensional decision boundary** hyperplane
- **Example**
  - 3 Features → 2-dimensional decision boundary plane
  - 2 Features → 1-dimensional decision boundary line
- Classes might **only be separable** in **higher dimensions**

**Img. 1:** Separation in higher dimensions



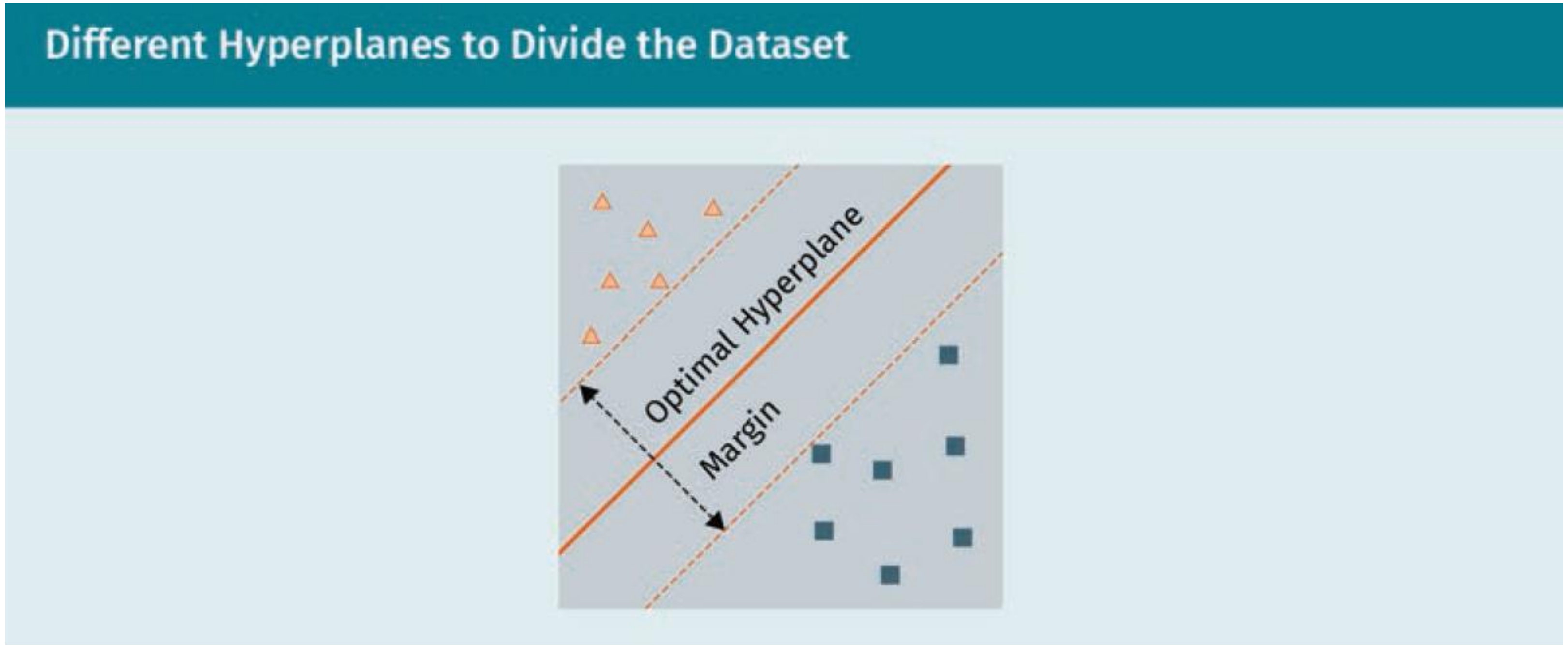

Source of text: Boehmke & Greenwell, 2019, Hastie et al., 2017
Source of image 1: Christian Müller-Kett, 2021

# SUPPORT VECTOR MACHINES (SVM)

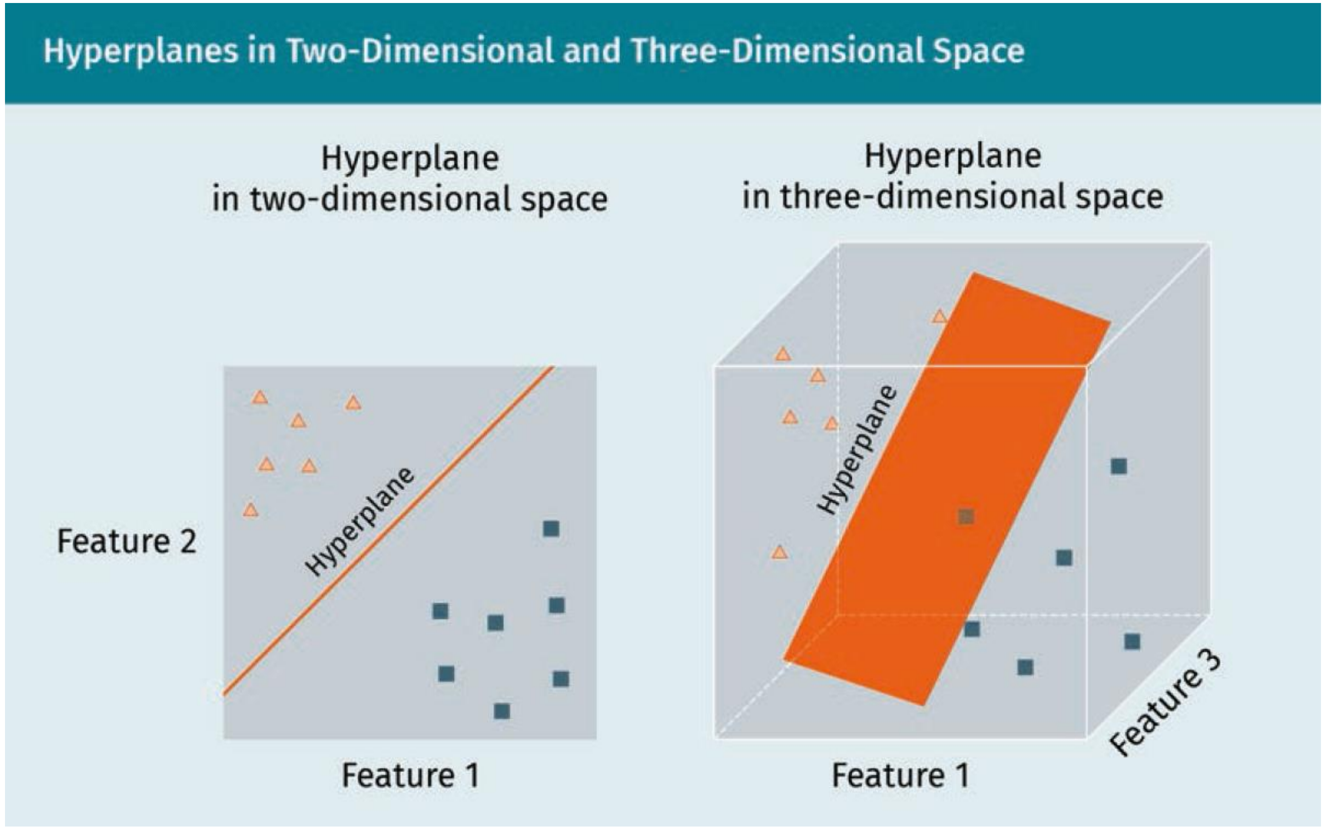



Source of image: Course book

- Choose hyperplane where the distance between the two classes is maximized
- SVMs: large margin classifiers

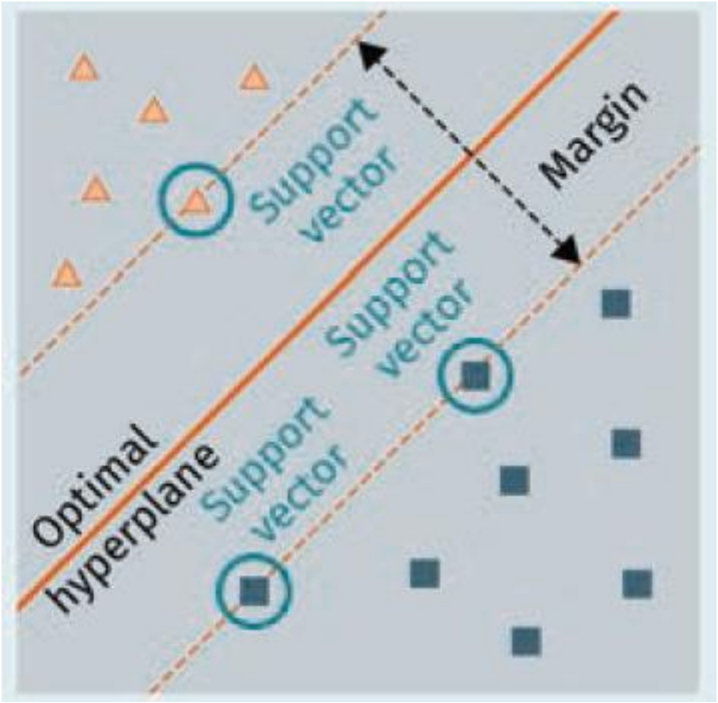


– Given n-dimensional vector space, a hyperplane is a subspace of the same with n-1 dimensions




Source of image: Course book

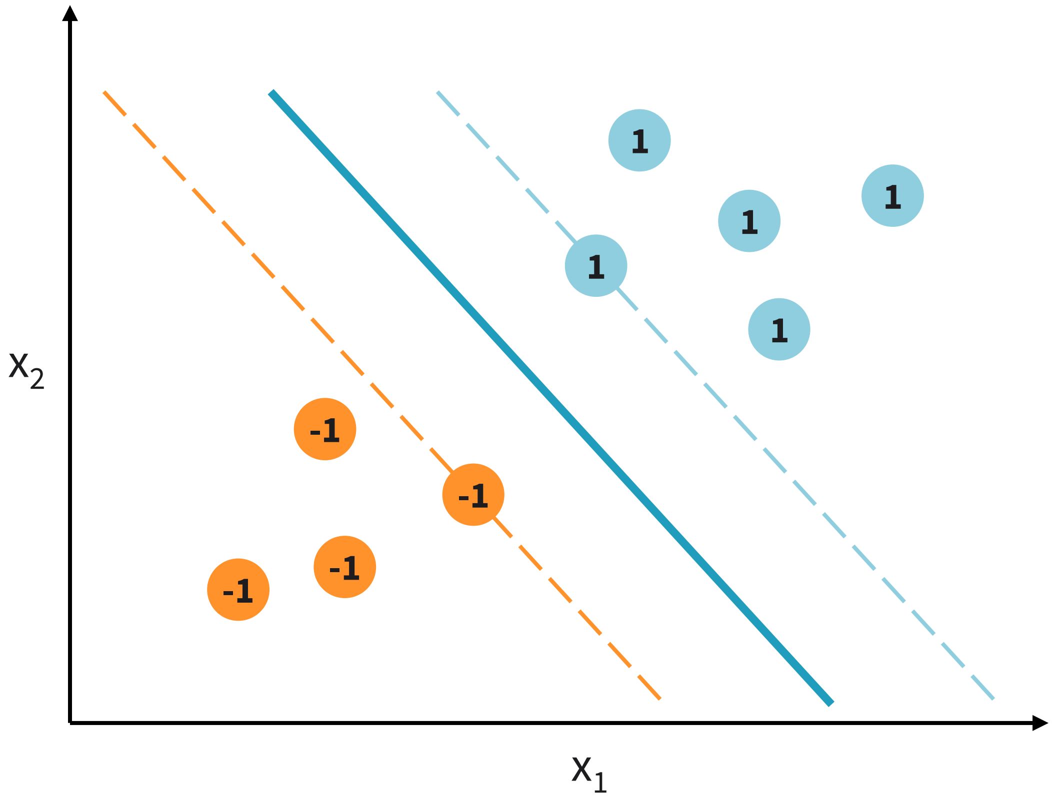
- Support vectors: data points that lie on the margin of a class and determine the location of the hyperplane
- Two types of SVMs:
  - Linear SVMs are used for data that are linearly separable
  - Nonlinear SVMs are used for data that are not linearly separable
    - The data are mapped into a higher dimensional space, where they are more easily separated using kernel trick




Source of image: Course book

- Samples **closest to the decision boundary**
- **Margin** (dotted lines) are described by $\boldsymbol{y_i(\vec{w} \cdot \vec{x} - b) = 1}$

**Img. 2:** Large margin classification

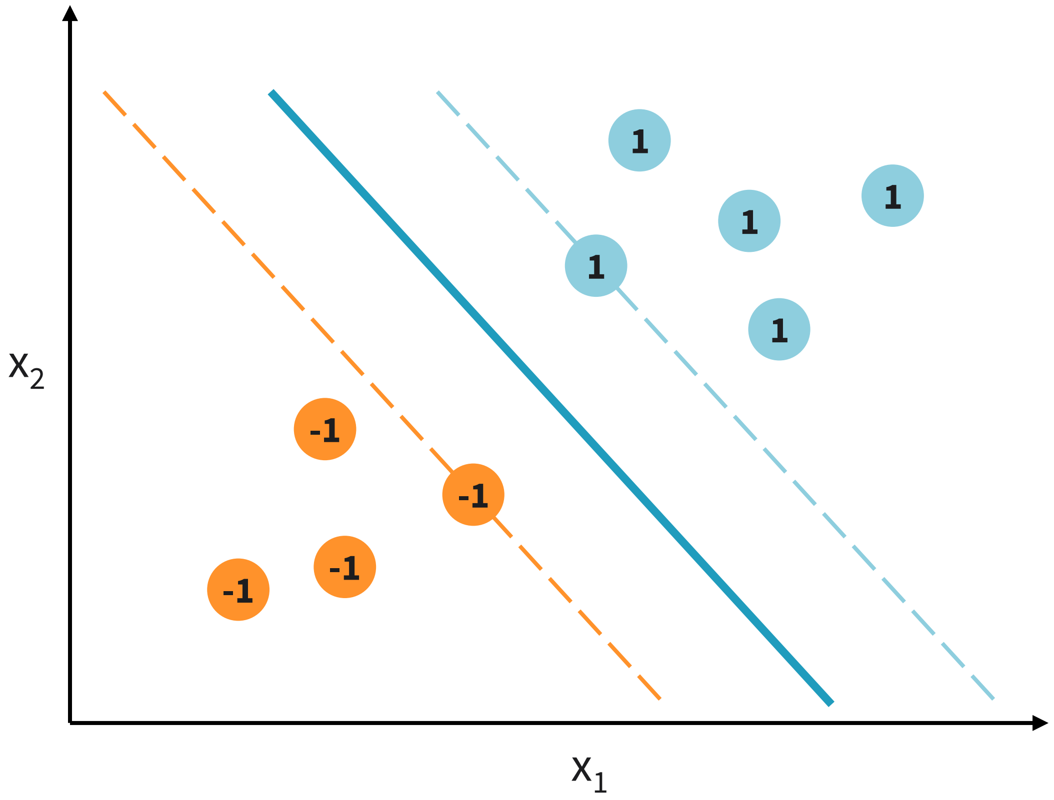



Source of text: Boehmke & Greenwell, 2019, Hastie et al., 2017
Source of image 1: Christian Müller-Kett, 2021

1. **Margin** (dotted lines) are described by $y_i(\vec{w} \cdot \vec{x} - b) = 1$
2. **Maximize** $\frac{2}{||w||}$

— Margin maximization depends on $x_i \cdot x_j$

**Img. 2:** Large margin classification

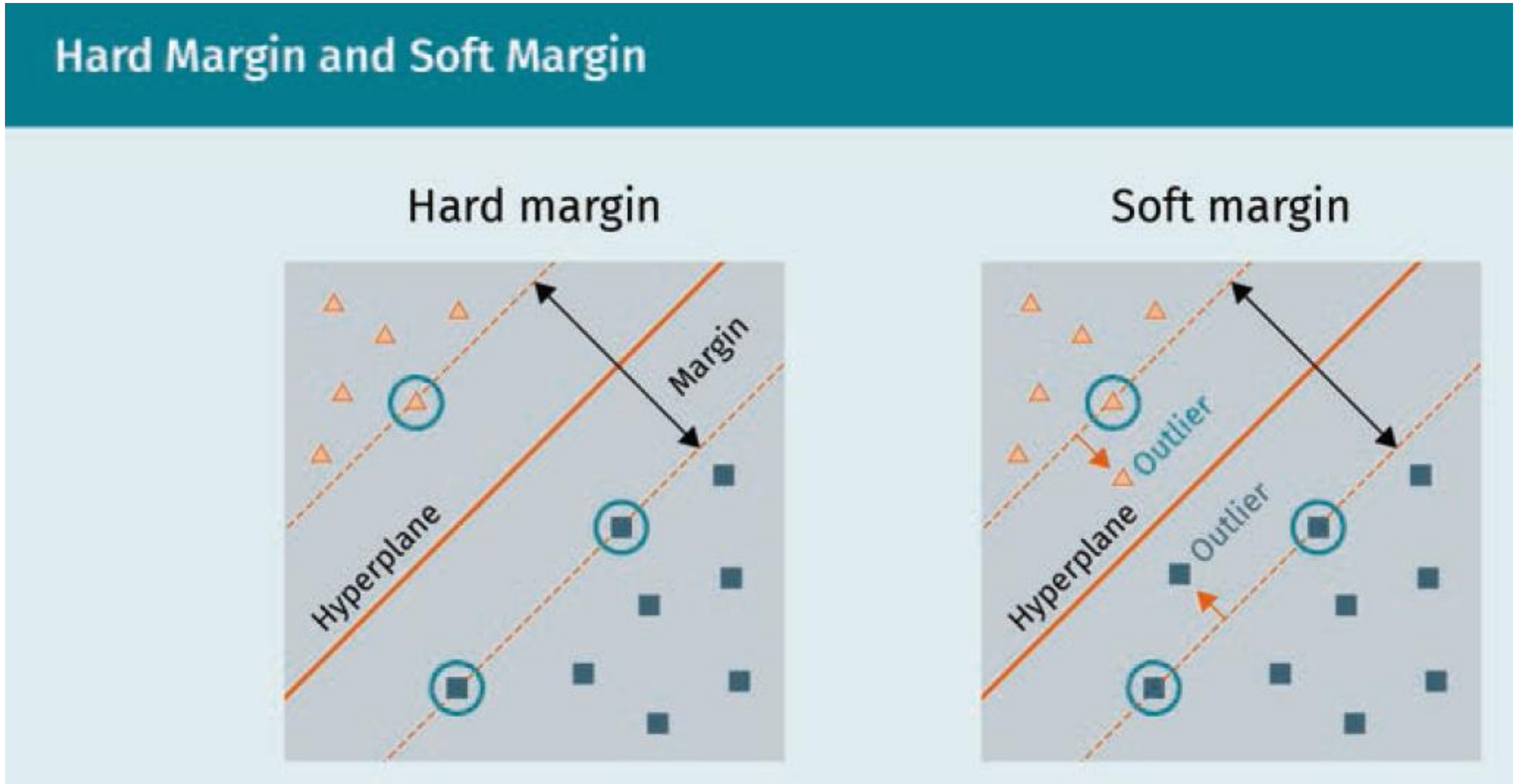



Source of text: Boehmke & Greenwell, 2019, Hastie et al., 2017
Source of image 1: Christian Müller-Kett, 2021


→ To make SVMs less sensitive to outliers we need to allow misclassifications (soft margin) → bias-variance-tradeoff

Source of image: Course book

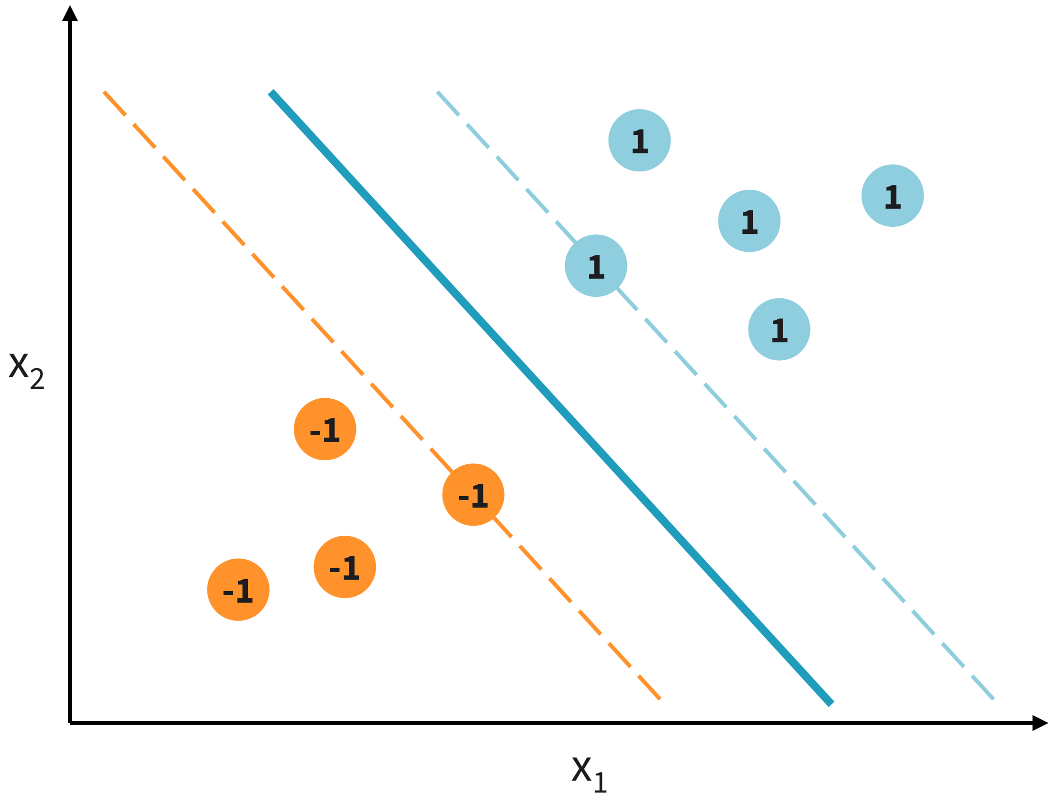
– With a soft margin, the threshold allows misclassification and thus leads to a higher bias on the training data




Source of image: Course book

- **Soft margin** allows samples to be misclassified
- **Lower C** = larger margin = **many misclassifications**
- **Larger C** = narrower margin = **few misclassifications**

**Img. 2:** Large margin classification



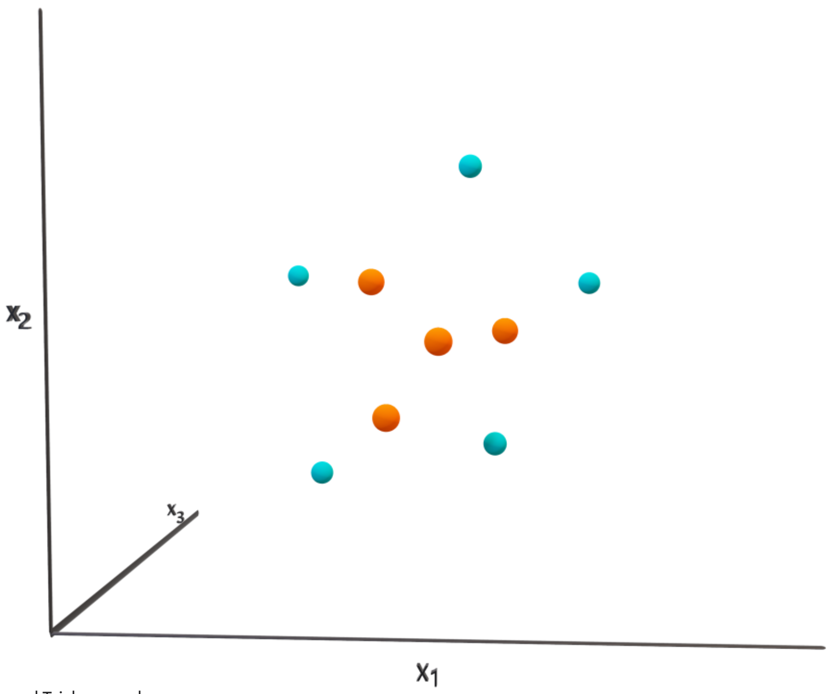

Source of text: Boehmke & Greenwell, 2019, Hastie et al., 2017
Source of image 1: Christian Müller-Kett, 2021

## THE KERNEL TRICK

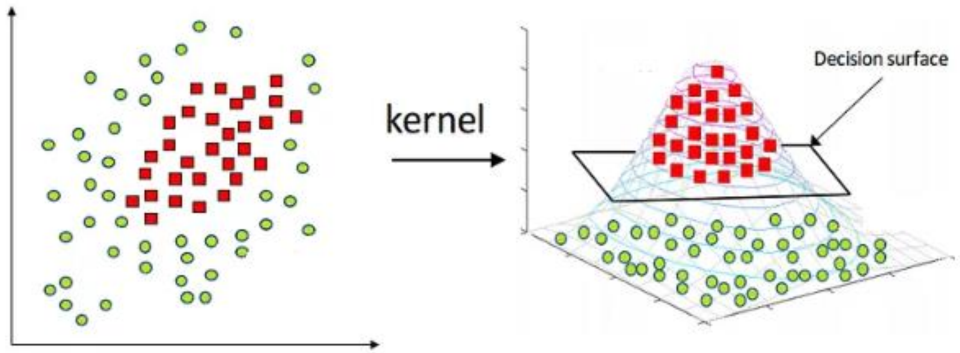
1. Margin maximization depends on $\boldsymbol{x_i} \cdot \boldsymbol{x_j}$
2. Classes might **only be separable** in **higher dimensions**

- Tranform the data to higher dimensions
- Calculate the **dot product** of the **transformed data**
- **Kernel functions** give the **same results** as the dot product of the transformed data
- **We do not have to transform** the data to higher dimensions

**Img. 1:** Separation in higher dimensions

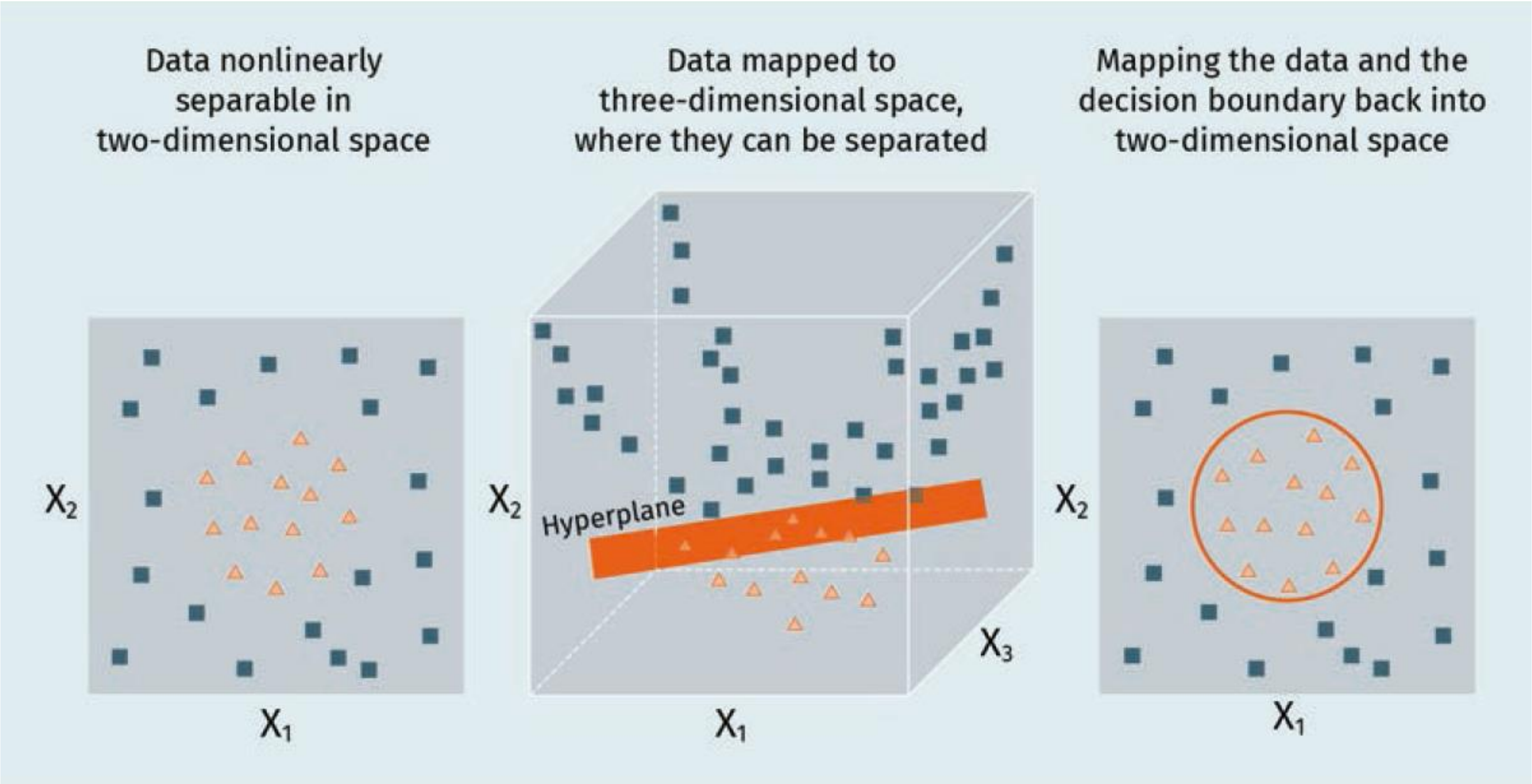



**Img. 2:** Kernel Trick example




Source of image 1: Christian Müller-Kett, 2021
Source of image 2: https://medium.com/@Suraj_Yadav/what-is-kernel-trick-in-svm-interview-questions-related-to-kernel-trick-97674401c48d

# THE KERNEL TRICK




$$X_1 = x_1^2, \quad X_2 = x_2^2, \quad X_3 = \sqrt{2}x_1x_2$$

- A transformation of the data into a higher-dimensional space is a computationally intensive task
- Solution: the kernel trick does not perform an actual transformation
  - Merely calculates the relation of the data points to each other as though they were in a higher-dimensional space.
  - This mapping of the data is done with the help of a kernel function

- Directly compute the distance, i.e., the scalar products of the data points for the expanded feature representation, without ever actually computing the expansion.
- Common methods
  - Polynomial kernel: computes all possible polynomials up to a certain degree
  - Radial basis function (RBF) (Gaussian kernel): corresponds to an infinite-dimensional feature space

- Polynomial kernel
  - Computes the decision boundary K via the dot product of the input features $X_1$ and $X_2$ by raising the power of the kernel to the degree d.

$$K(X_1, X_2) = (1 + \langle X_1, X_2 \rangle)^d$$

$$\langle X_1, X_2 \rangle = \sum_{i=1}^{n} x_{1i} x_{2i}$$

- Radial basis function kernel
  - Calculates the decision boundary K for the inputs $X_1$ and $X_2$ by taking the Euclidean distance between $X_1$ and $X_2$ ( $||X_1 - X_2||^2$ )and scaling it with the help of the hyperparameter determined by cross validation

$$K(X_1, X_2) = exp\left\{-\gamma \; ||X_1 - X_2||^2\right\}$$

  - The radial basis kernel is extremely flexible, and, as a rule of thumb, we generally start with this kernel when fitting SVMs in practice”
- More examples: https://scikit-learn.org/stable/auto_examples/svm/plot_svm_kernels.html#sphx-glr-auto-examples-svm-plot-svm-kernels-py

- **dth- degree Polynomial Kernel**
  - $K(x_i \cdot x_j) = (r + x_i \cdot x_j)^d$
- **Radial Basis Function (RBF)**
  - $K(x_i \cdot x_j) = e^{-\gamma \|x_i - x_j\|^2}$
- **...**

Source of text: Boehmke & Greenwell, 2019

## AN APPLIED EXAMPLE

```
…

from sklearn import svm

…

clf = svm.SVC(kernel='poly')

clf.fit(X_train, y_train)

…

y_pred = model.predict(testing_data)

…
```

**Img. 3:** Code


Source of image 5: Microsoft Archive

- Explain the **concept** of **large margin classification**.
- Conceptualize a large margin classifier with **support vector machines**.
- Explain and make use of the **kernel trick**.
- Apply support vector machines with the use of **Python**.

SESSION 4

# TRANSFER TASK

A start-up that sells **sustainable products in smaller stores** has been very successful in recent years. As a result, more stores are to be opened worldwide.

As a Data Scientist, you and your team are tasked with training a **machine learning model predicting product demand** one week ahead.

In the first attempt, you fit a **linear regression model**. Explain the drawback in comparison to Support Vector Machines.

For the next model, you train an SVM and observe nearly **perfect training accuracy**. Explain why this might be a problem and how you could solve this.

Please present your results.
The results will be discussed in plenary.

1. What should be maximized when finding the optimal hyperplane?
   a) kernels
   b) the number of dimensions
   c) the margin
   d) support vectors

2. How are the vectors that define the position of an SVM’s hyperplane called?

   a) support vectors
   b) Kernel vectors
   c) Feature vectors
   d) label vectors

3. Which one of the following SVM initialization statements written in Python is correct?

a) from scikit-learn import svm
   clf = VC(kernel='linear’)

b) from scikit-learn import svm
   clf = svm.SVC(kernel='linear_kernel’)

c) from sklearn import svm
   clf = svm.SVC(kernel='linear’)

d) from sklearn import svm
   clf = svm.SVC('linear_ kernel ')

# LIST OF SOURCES

Boehmke, B., & Greenwell, B. (2019). *Hands-on machine learning with R*. Chapman & Hall.
Hastie, T., Tibshirani, R., Friedman, J. H. (2017). *The elements of statistical learning. Data mining, inference, and prediction*. Second edition. New York, NY: Springer.