POLITECNICO
DI TORINO

AI ML

Homework 1

Artificial Intelligence and Machine Learning

Teaching Assistant:
Silvia Bucci

Homework submission

- Individual work (**no groups**)
- **3** Homeworks
- Score → **[0,2]** for each one
- Total **score sufficient** for the admission to the final exam→ **[4,6]**

  4 → no extra point

  5 → +1 point
  6 → +2 points
  } This holds **ONLY** if you submit the Project!

- You can keep your homework till the last exam session before the next course beginning (**1 year**)

Homework submission

- **Deadline → 3 weeks** after the publication of each assignment.
  Homework due: November 18

  It is not a strict deadline, but it is highly recommended!

  Alternatively you can submit together with the project (one week before the written exam) - but this will be taken into consideration when assigning the score.

Homework submission

- **Uploading:** use the “Portale della didattica”

  Submit a zip file named ID_AIML_homework1.zip (ID = matricola)

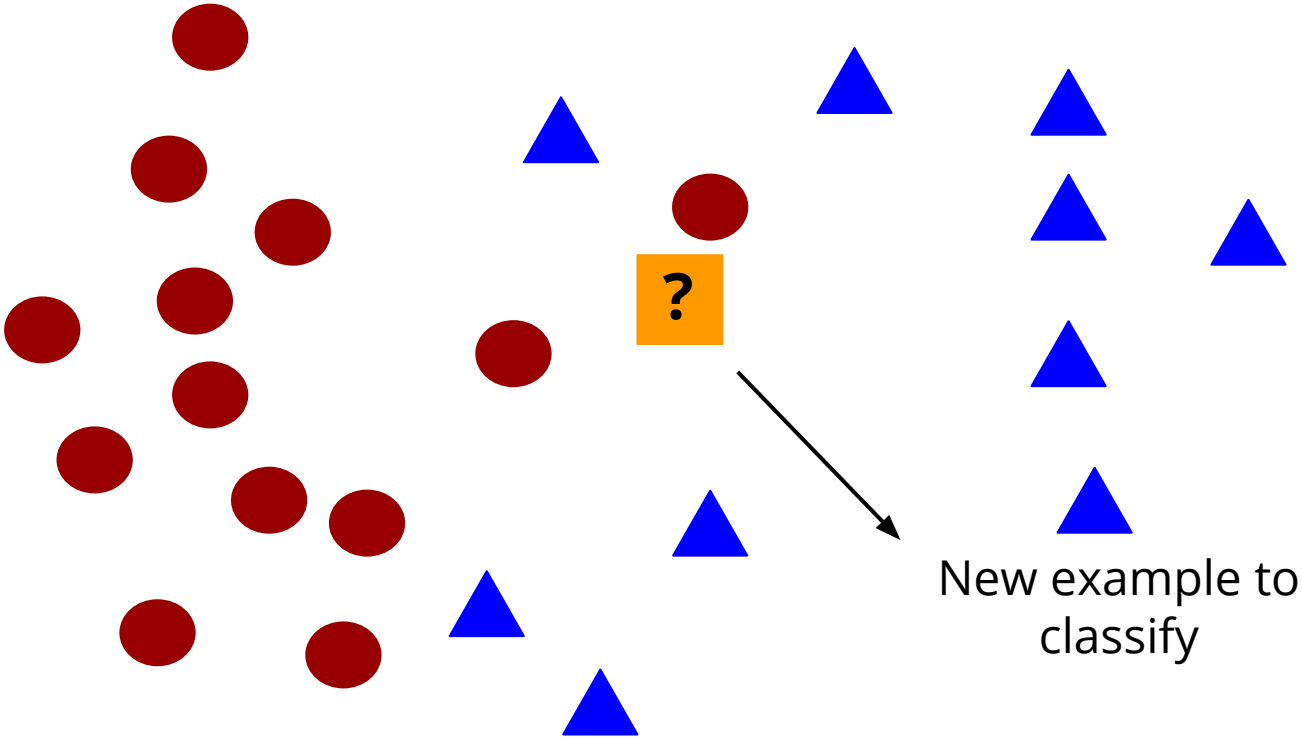
  It should contain
  - the report: a pdf file briefly describing the data + the logic of the methods + the results
  - the code

Note: **the report is *not* cut-and-paste the code in a pdf.**
You can copy some code lines in the report but only if they are presented together with the method explanation.

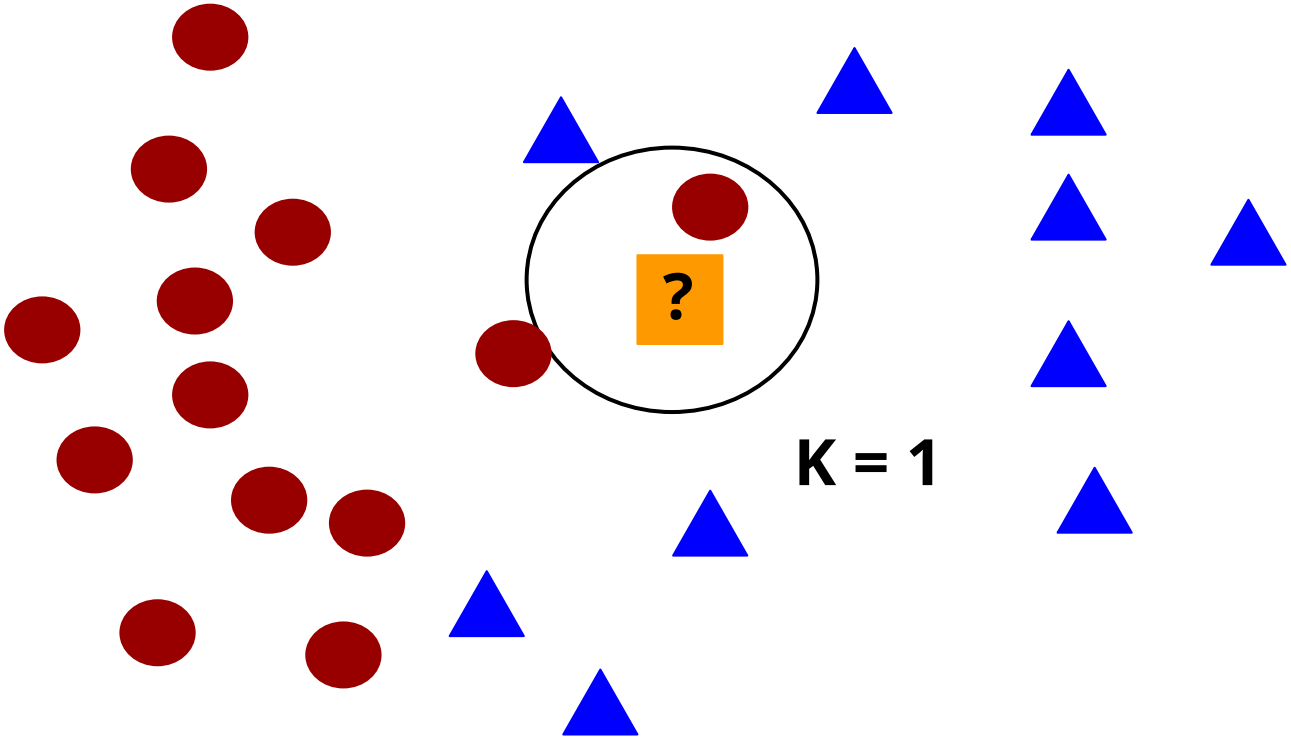
Homework 1: Nearest Neighbors / SVM

- Recap: NN/SVM
- Training procedure
- Presentation of the homework
- Try it yourself!

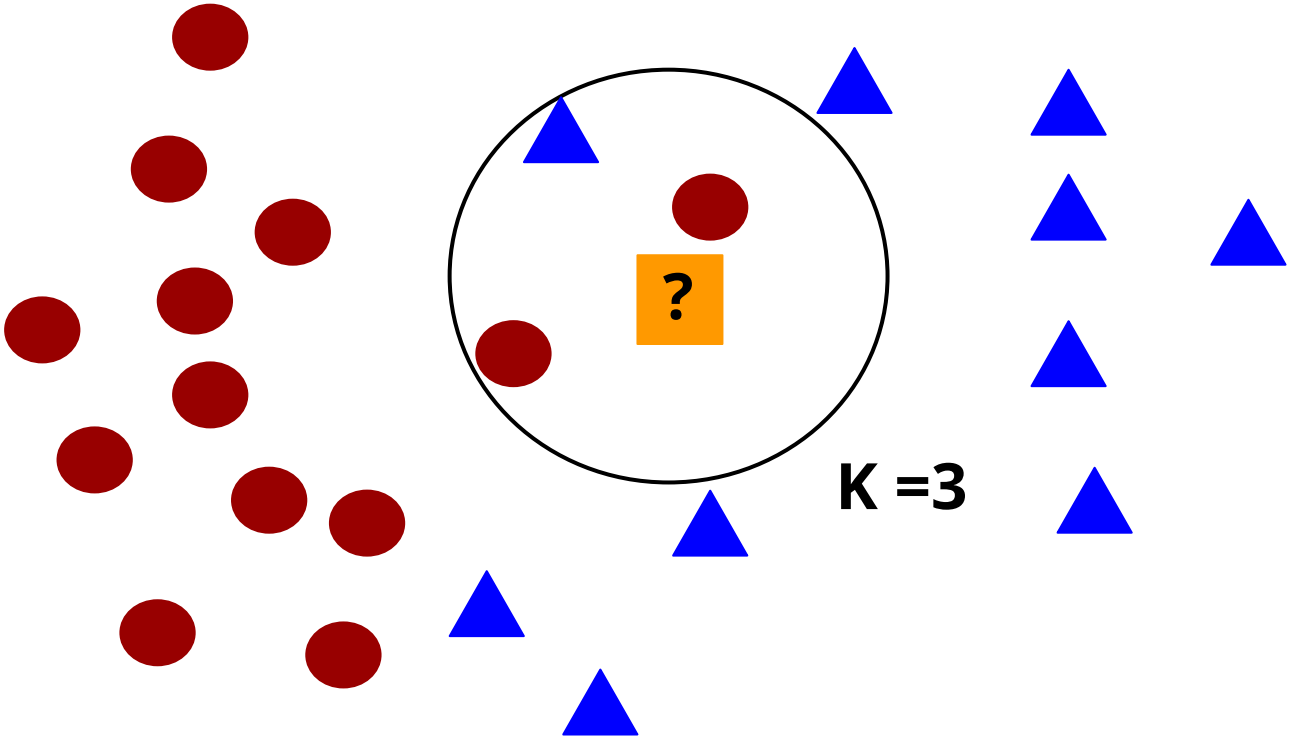
Recap: K-Nearest Neighbors



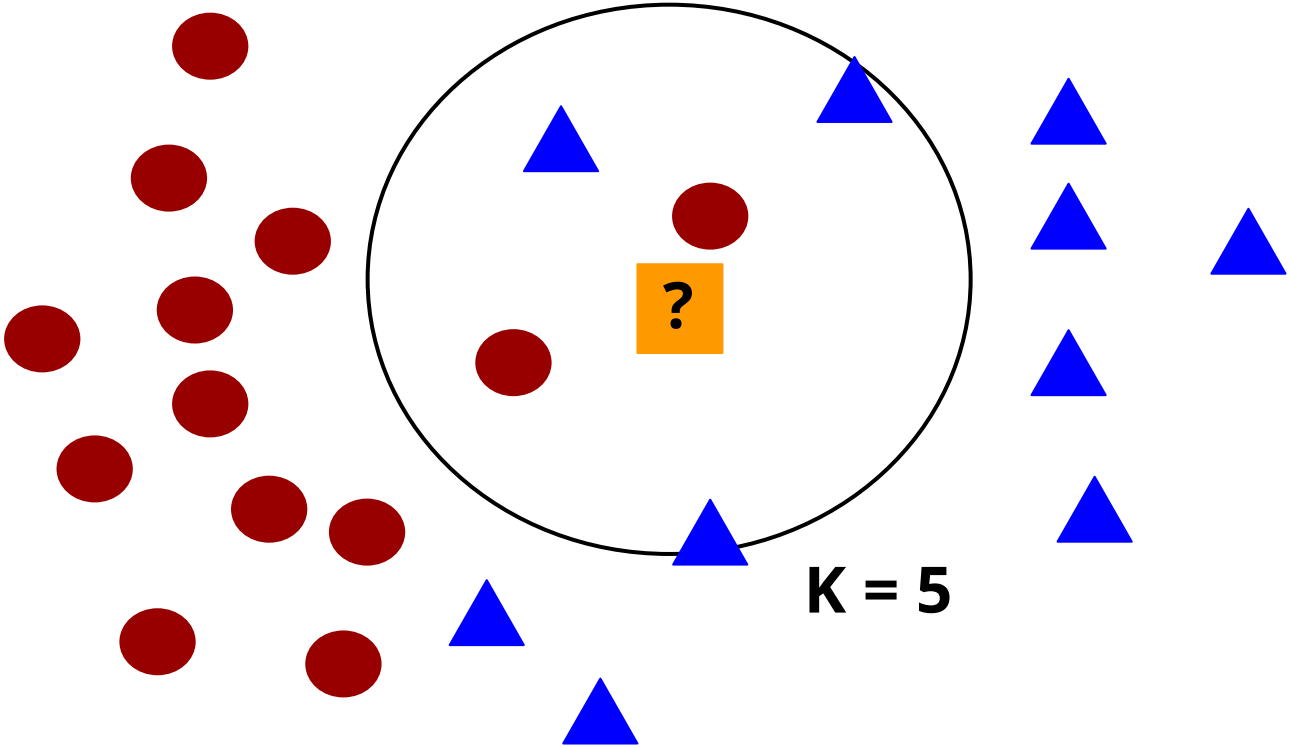
Recap: K-Nearest Neighbors



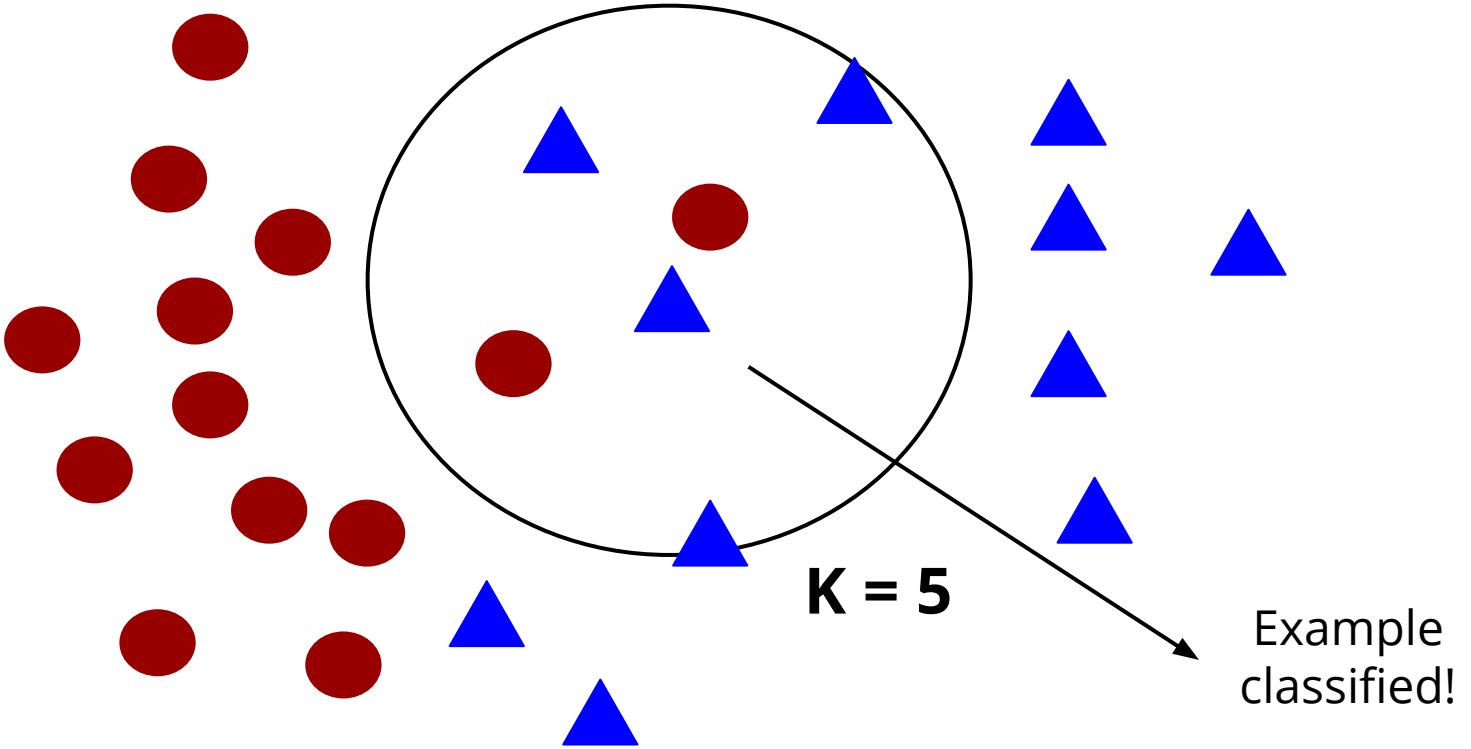
Recap: K-Nearest Neighbors



Recap: K-Nearest Neighbors



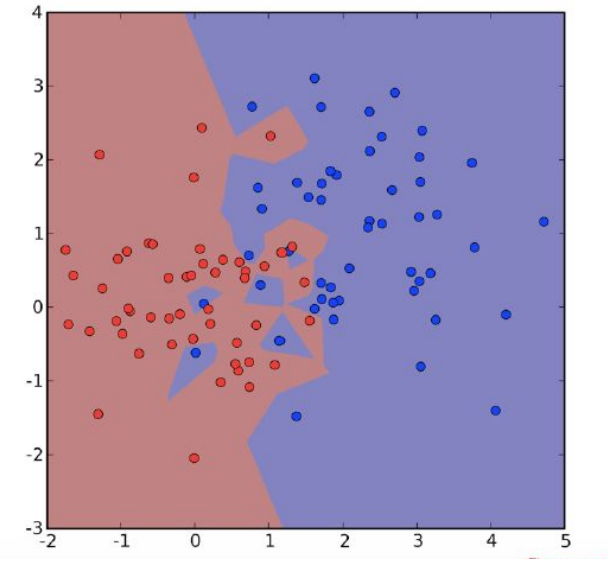
Recap: K-Nearest Neighbors



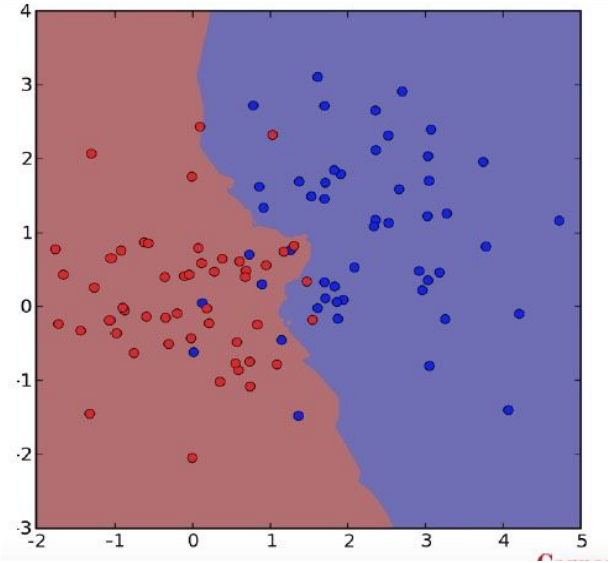
K-NN - Hyper Parameter

***K** is the number of neighbors used for the voting.*

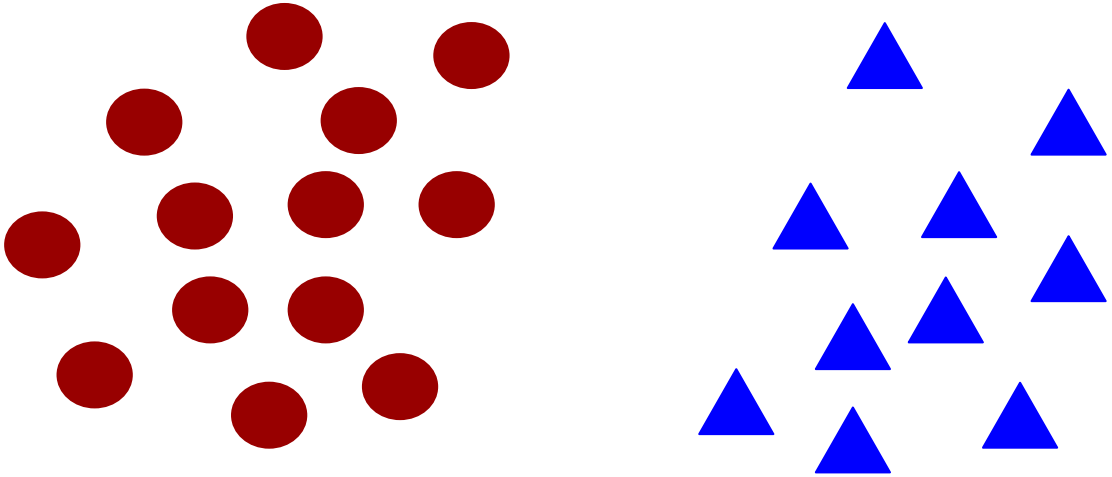
K = 1



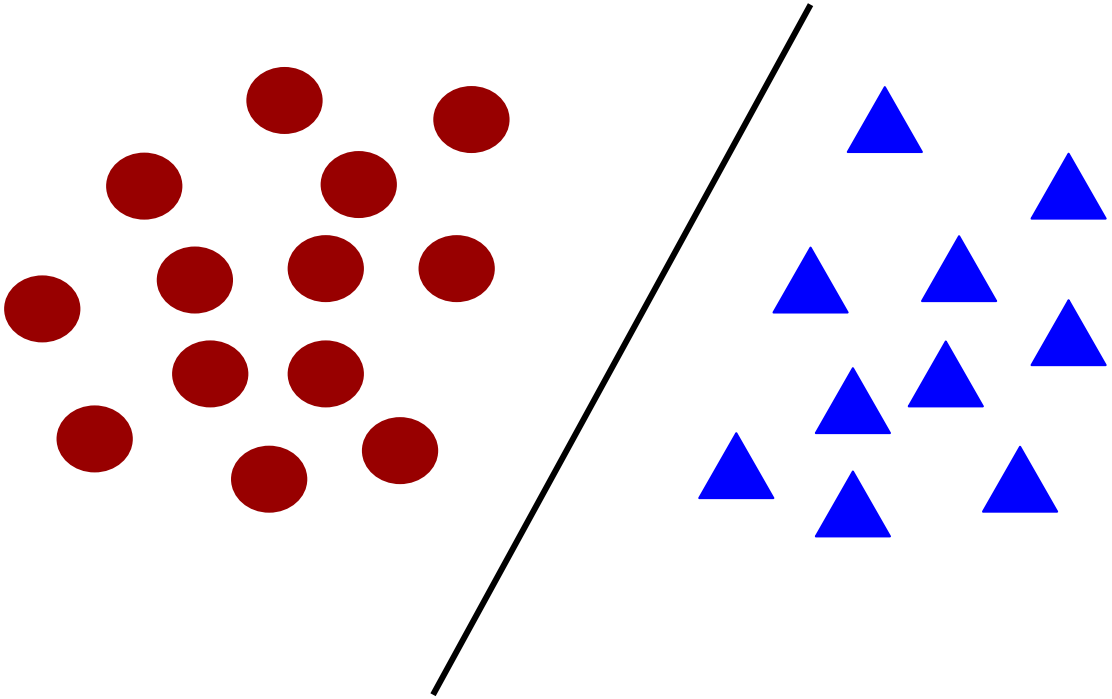
K = 4



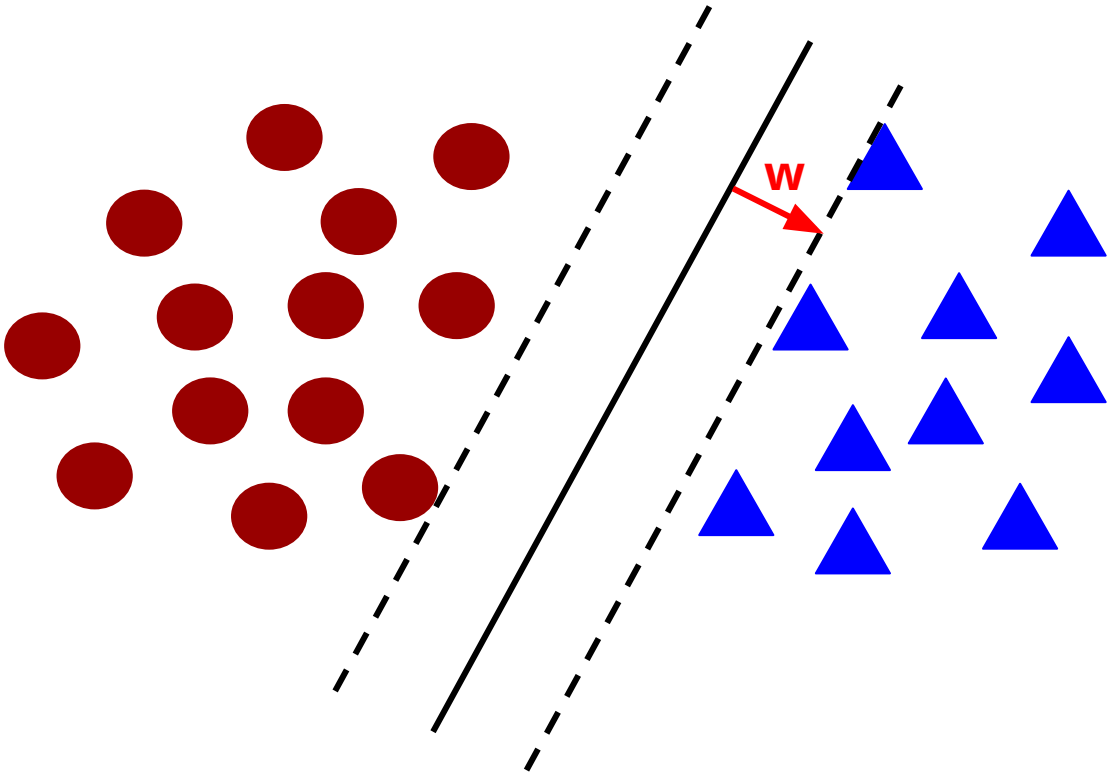
Recap: Linear SVM



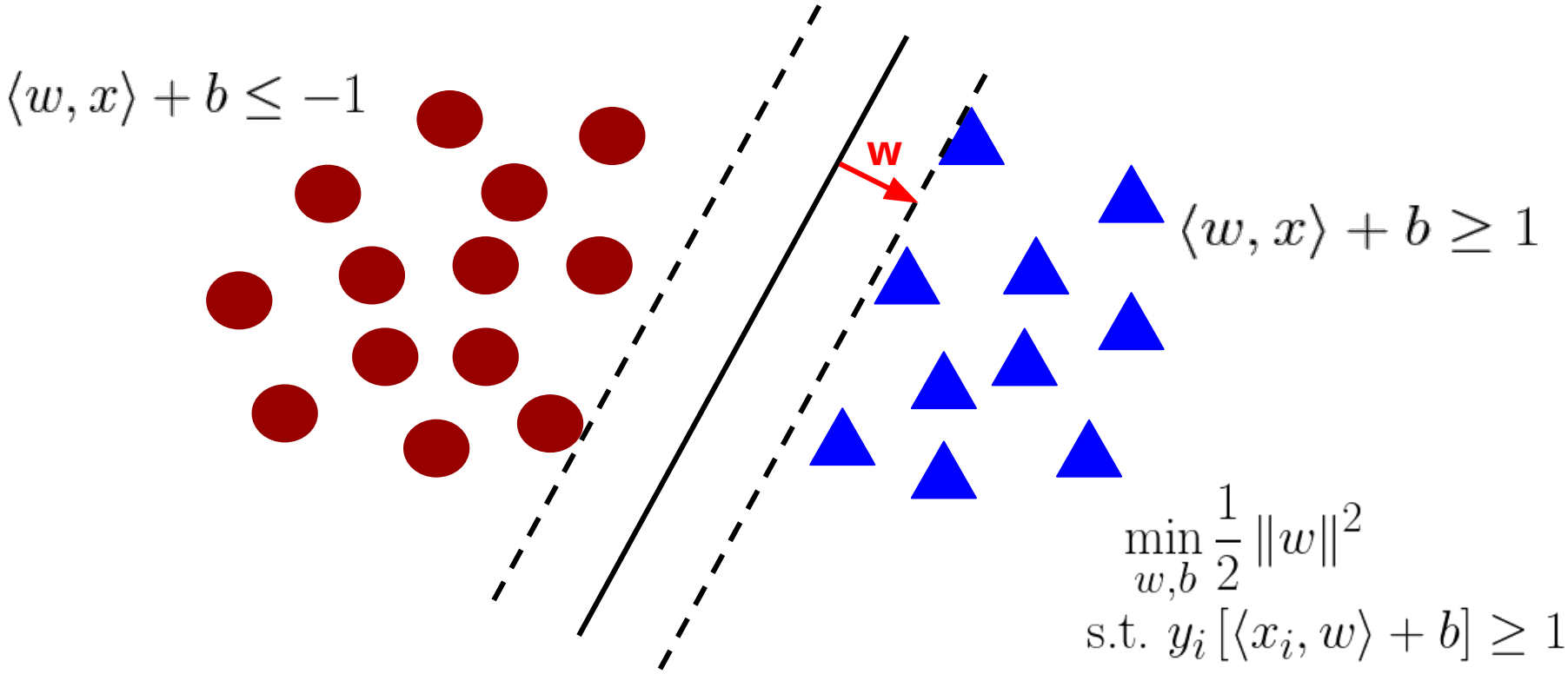
Recap: Linear SVM



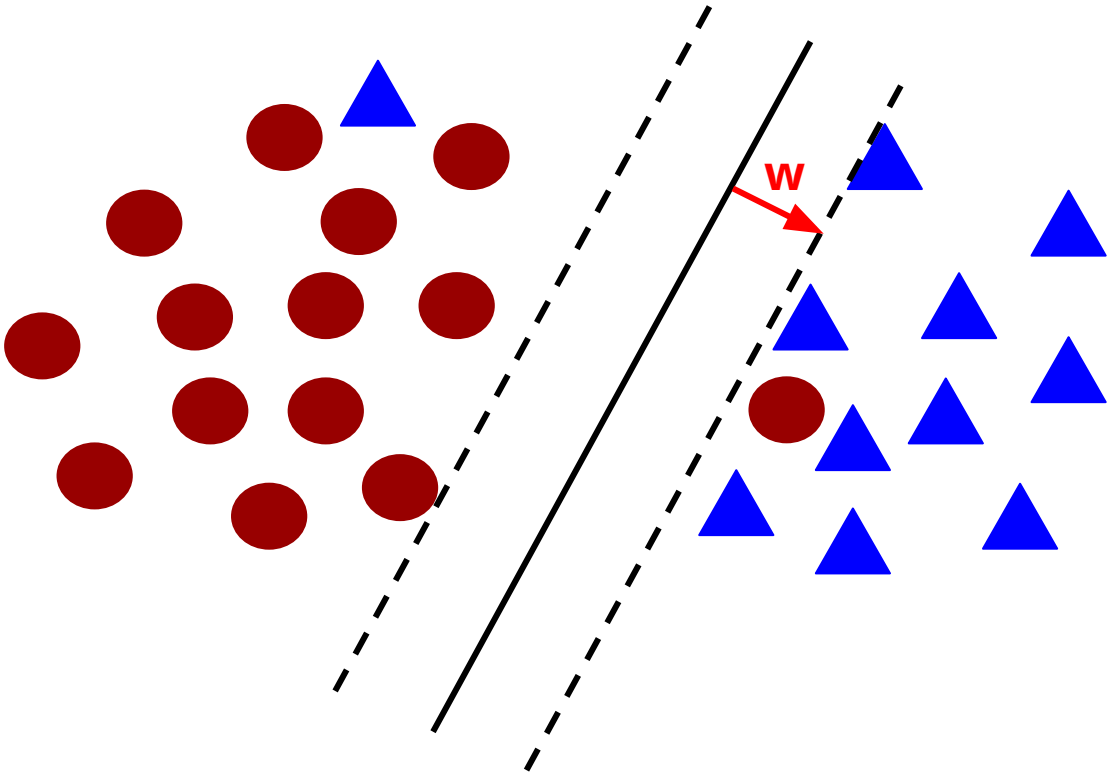
Recap: Linear SVM



Recap: Linear SVM - Hard margin problem

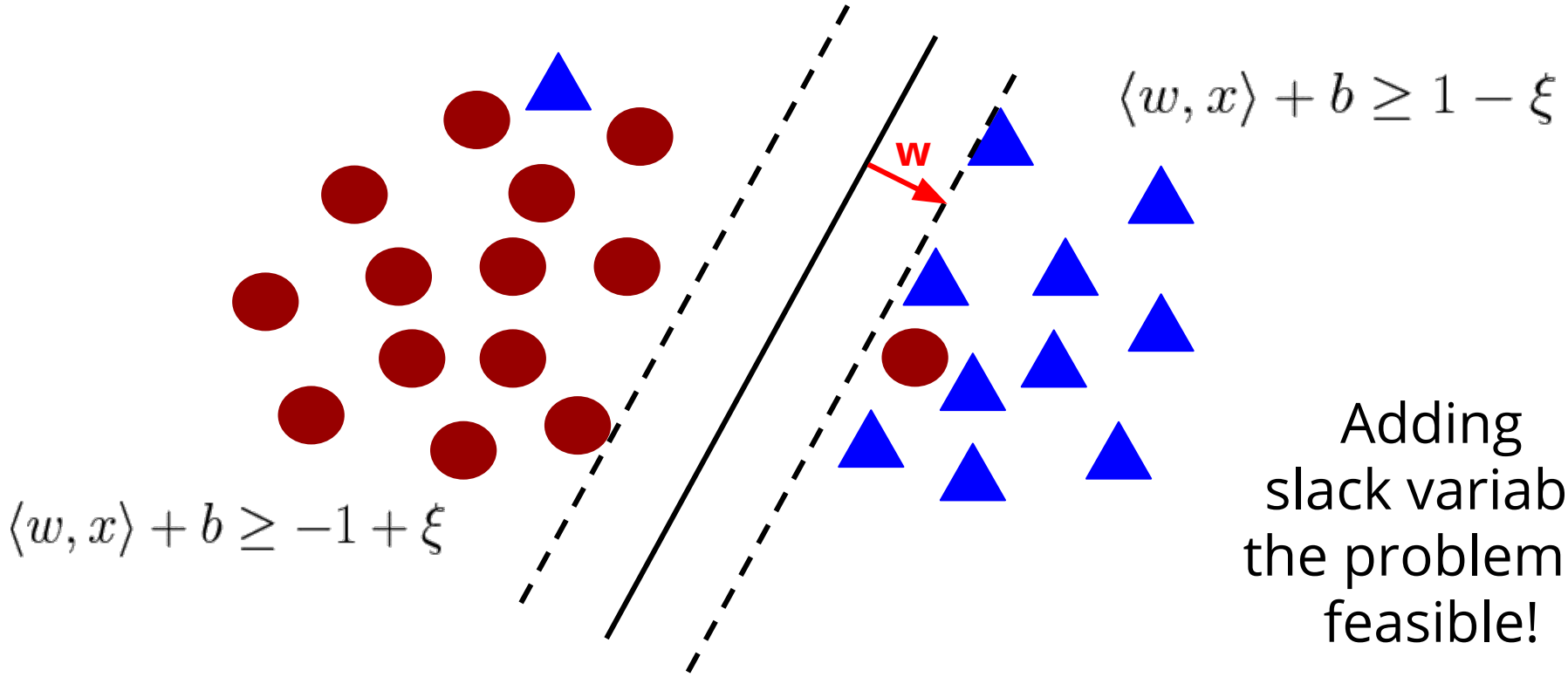


Recap: Linear SVM - Soft margin problem



Linear separator is impossible!

Recap: Linear SVM - Soft margin problem



Adding slack variable the problem is feasible!

Recap: Linear SVM

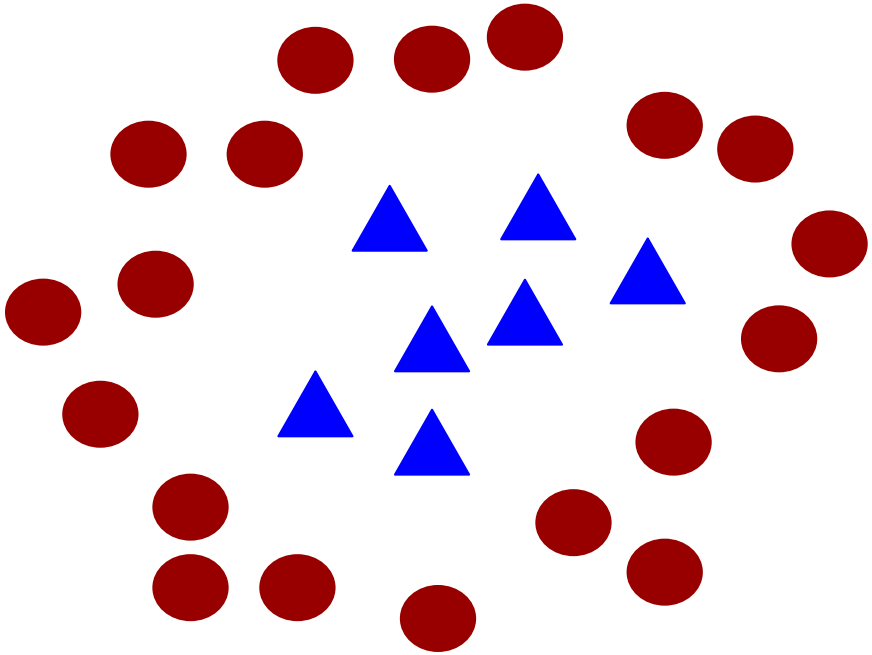
Hard margin problem

$$\min_{w,b} \frac{1}{2} \|w\|^2$$
$$\text{s.t. } y_i \left[\langle x_i, w\rangle + b\right] \geq 1$$

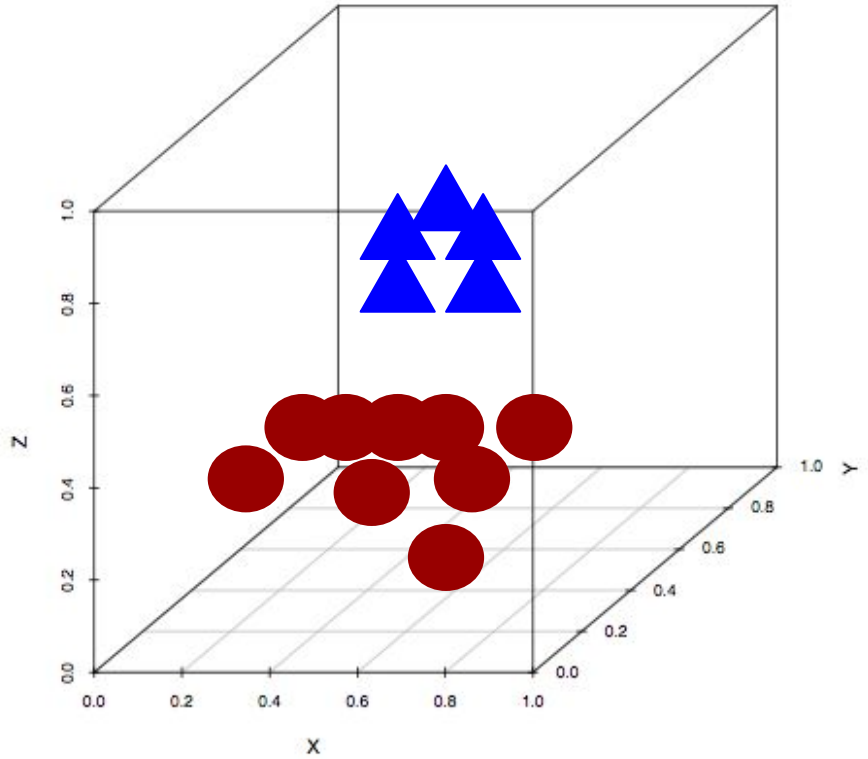
Soft margin problem

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i$$
$$s.t.\ y_i \left[\langle x_i, w\rangle + b\right] \geq 1 - \xi_i \ and \ \xi_i \geq 0$$

Recap: Kernel trick - SVM



Recap: Kernel trick - SVM



We can separate each data point by projecting it into an **higher dimension**

Recap: Kernel trick - SVM

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i$$

$$s.t.\ y_i \left[\langle x_i, w \rangle + b\right] \geq 1 - \xi_i \ and \ \xi_i \geq 0$$

Recap: Kernel trick - SVM

$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_i \xi_i$$

$$s.t.\ y_i [\langle x_i, w \rangle + b] \geq 1 - \xi_i \ and \ \xi_i \geq 0$$

$$\Phi(x_i)$$

Hard to compute!

Recap: Kernel trick - SVM

..so we use the **dual** soft margin problem

$$\max_{\alpha} -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle + \sum_i \alpha_i$$

$$s.t. \sum_i \alpha_i y_i = 0 \; and \; \alpha_i \in [0, C]$$

Recap: Kernel trick - SVM

..so we use the **dual** soft margin problem

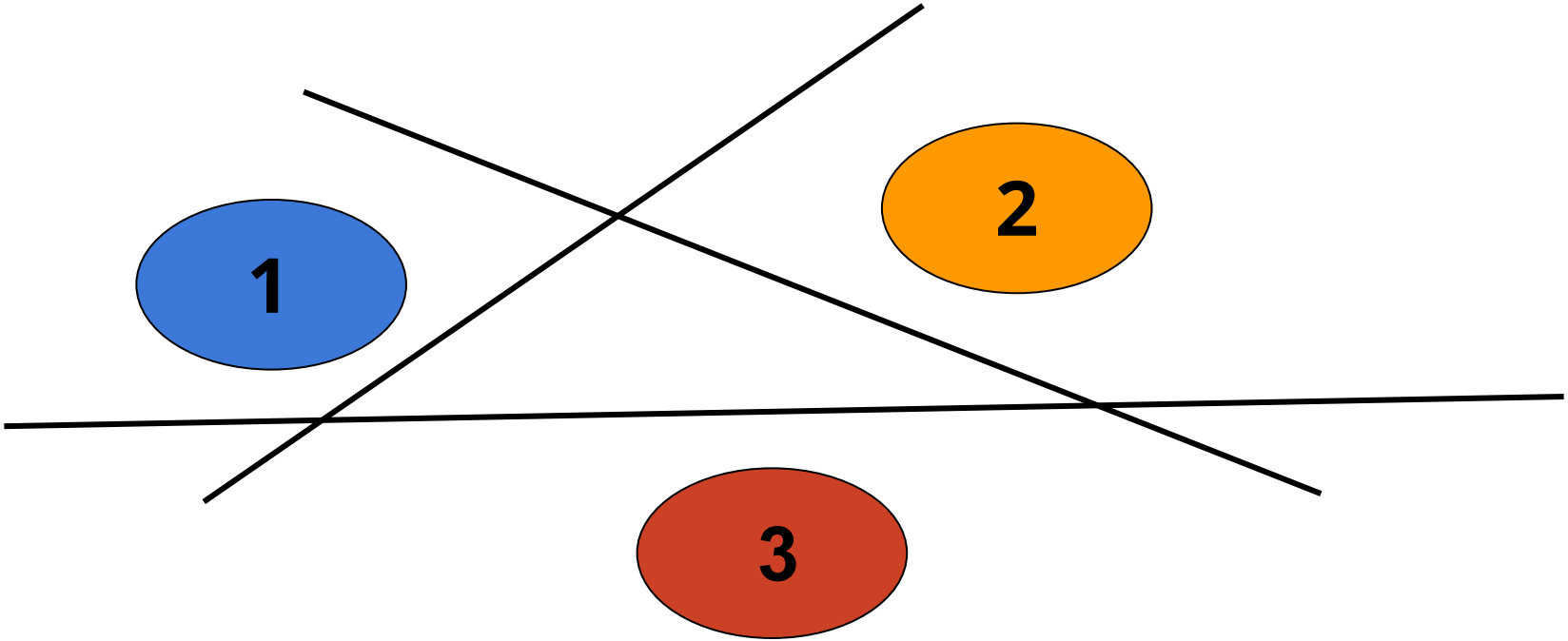
$$\max_{\alpha} -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j \langle x_i, x_j\rangle + \sum_i \alpha_i$$

$$s.t. \sum_i \alpha_i y_i = 0 \; and \; \alpha_i \in [0, C]$$

to be able to use the **KERNEL function**

$$k(x_i, x_j) = \langle \phi(x_i), \phi(x_j) \rangle$$

SVM - Multiclass: one vs all



SVM - Hyper Parameters

Soft margin problem:

- **C**

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$$
$$s.t.\ y_i\left[\langle x_i, w\rangle + b\right] \geq 1 - \xi_i \ and\ \xi_i \geq 0$$

SVM - Hyper Parameters

Soft margin problem:

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$$
$$s.t.\ y_i\left[\langle x_i, w\rangle + b\right] \geq 1 - \xi_i \ and \ \xi_i \geq 0$$

- **C**

***C** is the cost of misclassification*

SVM - Hyper Parameters

Kernel RBF:

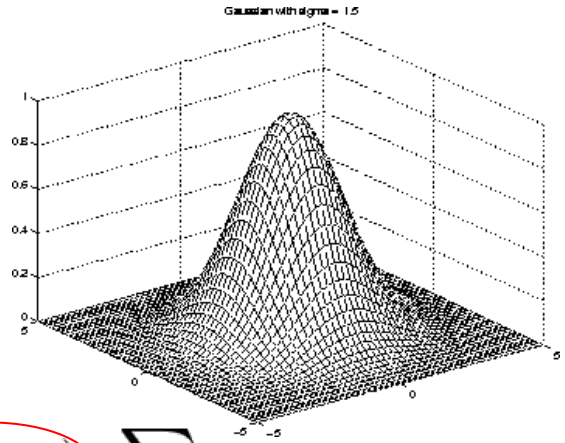
- **C**

$$\max_{\alpha} -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(x_i, x_j) + \sum_i \alpha_i$$

$$s.t.\ \sum_i \alpha_i y_i = 0\ and\ \alpha_i \in [0, C]$$

SVM - Hyper Parameters

ɣ is inversely proportional to the variance of the Gaussian



Kernel RBF:

- **C**
- **ɣ**

$$\max_{\alpha} -\frac{1}{2}\sum_{i,j}\alpha_i\alpha_j y_i y_j k(x_i,x_j) + \sum_i \alpha_i$$

$$s.t.\ \sum_i \alpha_i y_i = 0 \ and\ \alpha_i \in [0, C]$$

$$k(x_i,x_j) = e^{-\gamma(x_i - x_j)^2}$$

SVM - Hyper Parameters

Soft margin problem:

- **C**

$$\min_{w,b} \frac{1}{2}\|w\|^2 + C\sum_i \xi_i$$

$$s.t.\ y_i\left[\langle x_i, w\rangle + b\right] \geq 1 - \xi_i \ and \ \xi_i \geq 0$$

Kernel RBF:

- **C**
- **γ**

$$\max_{\alpha} -\frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j k(x_i, x_j) + \sum_i \alpha_i$$

$$s.t.\ \sum_i \alpha_i y_i = 0 \ and \ \alpha_i \in [0, C]$$

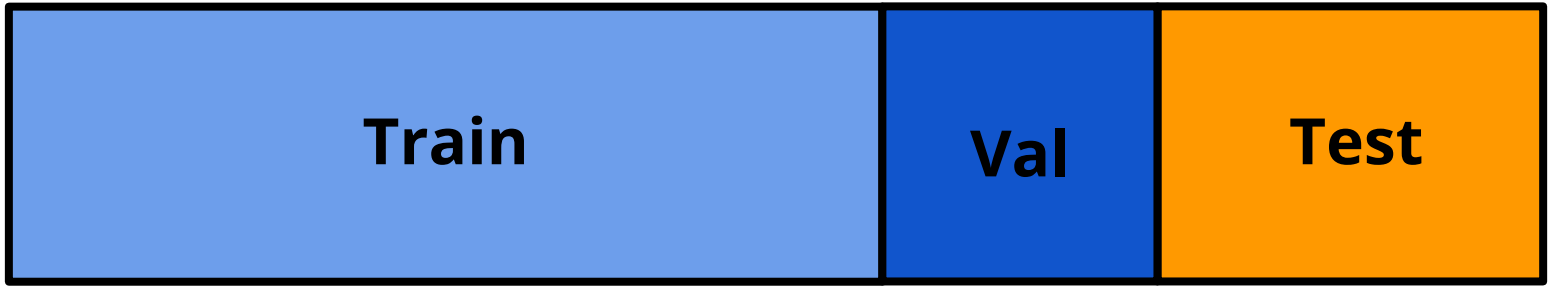
$$k(x_i, x_j) = e^{-\gamma(x_i - x_j)^2}$$

How to choose the Hyper Parameters?

Training procedure

To analyze the performance and evaluate a model you need to divide the dataset into **3 splits**

Training procedure

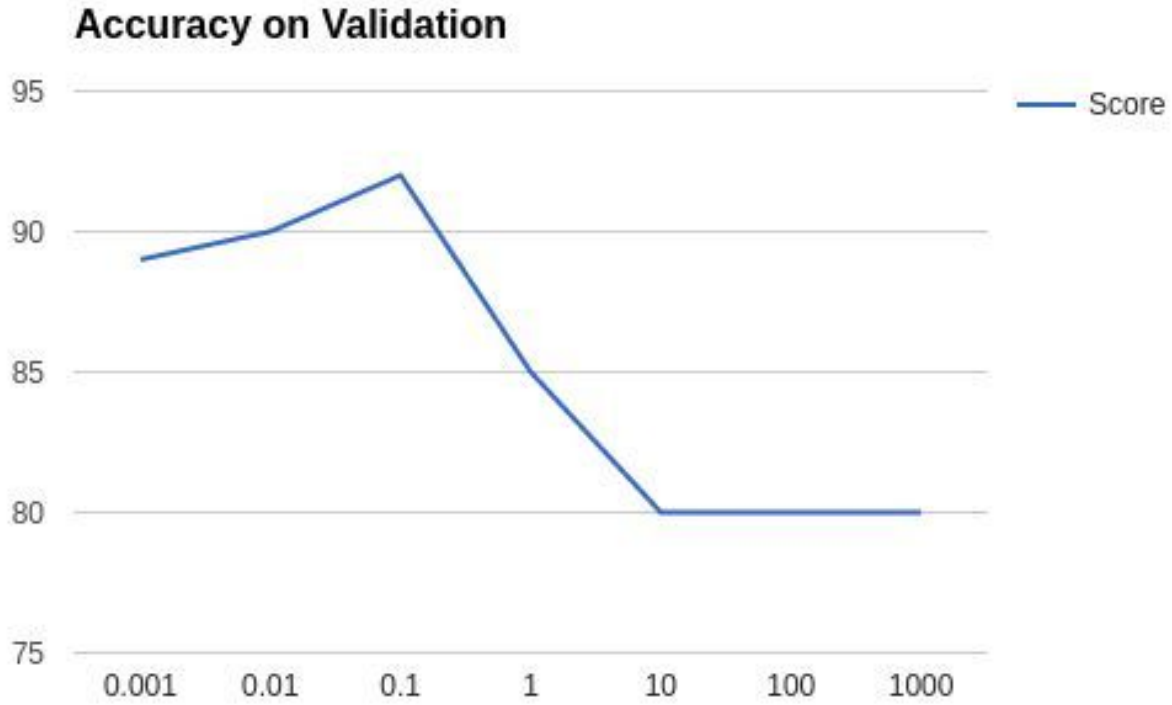


1. Train the model using only the **training set** with the **parameters chosen**
2. Evaluate the performance of the model on the **validation set**
3. Choose others parameters and repeat 1. and 2. until you find the parameters that work best on the validation set.

At the end you can evaluate the model on the **testing set** but you cannot change the parameters in function of the performance on the testing set!

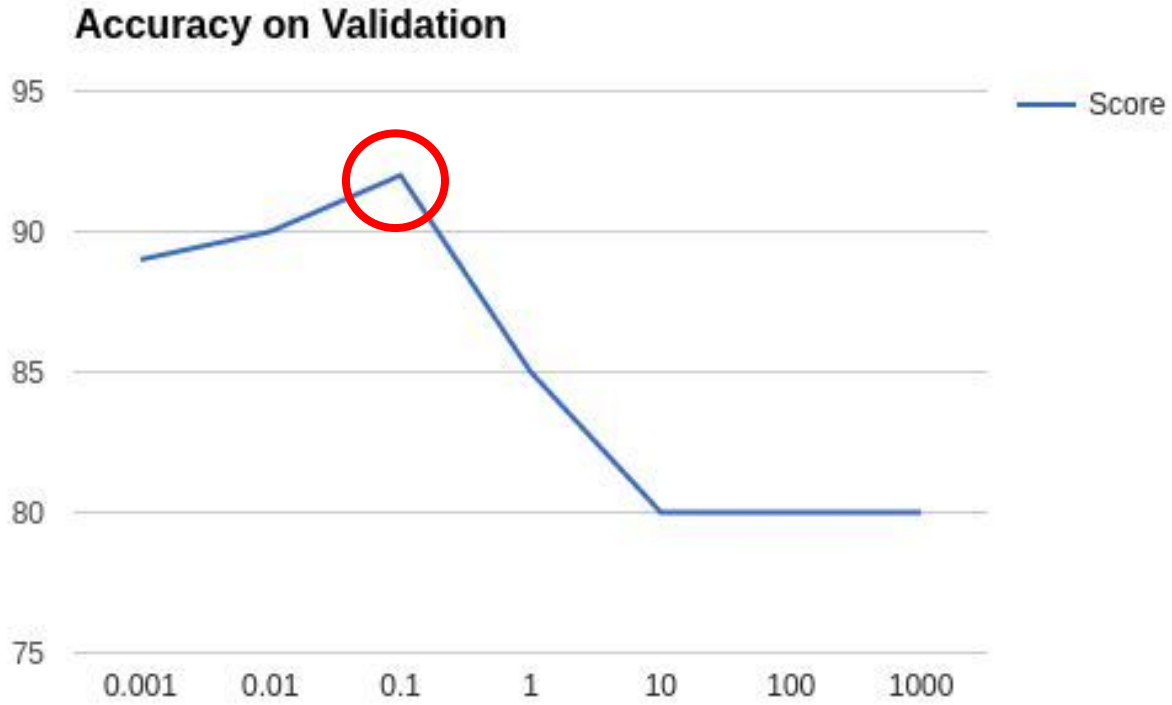
Training procedure - Example

Optimization of **C**



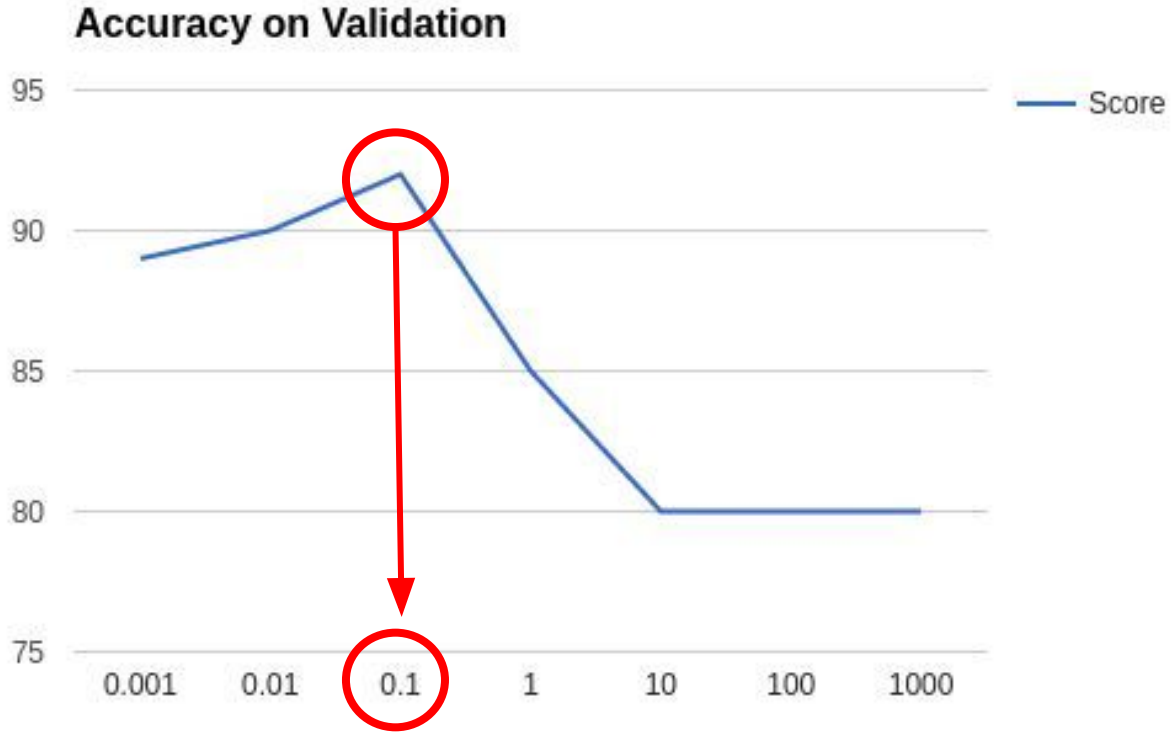
Training procedure - Example

Optimization of **C**



Training procedure - Example

Optimization of **C**



Best C found!

Training procedure - Example - Grid Search

Optimization of **C** and **ɣ**

Gamma \ C	10^{-2}	10^{-1}	**10^{0}**	10^{1}
10^{-9}	66%	75%	60%	60%
10^{-7}	48%	67%	78%	79%
10^{-5}	80%	82%	**90%**	85%
10^{-3}	45%	66%	78%	74%

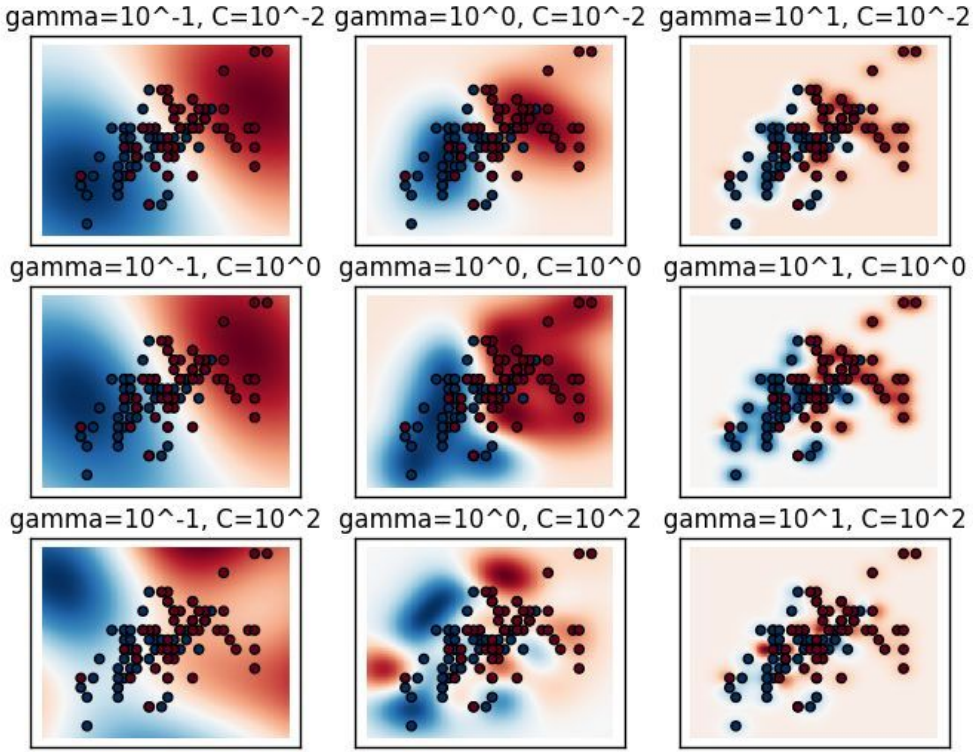
Training procedure - Example - Grid Search

Optimization of **C** and **ɣ**

Best C and ɣ found!

Gamma \ C	10^{-2}	10^{-1}	10^{0}	10^{1}
10^{-9}	66%	75%	60%	60%
10^{-7}	48%	67%	78%	79%
10^{-5}	80%	82%	**90%**	85%
10^{-3}	45%	66%	78%	74%

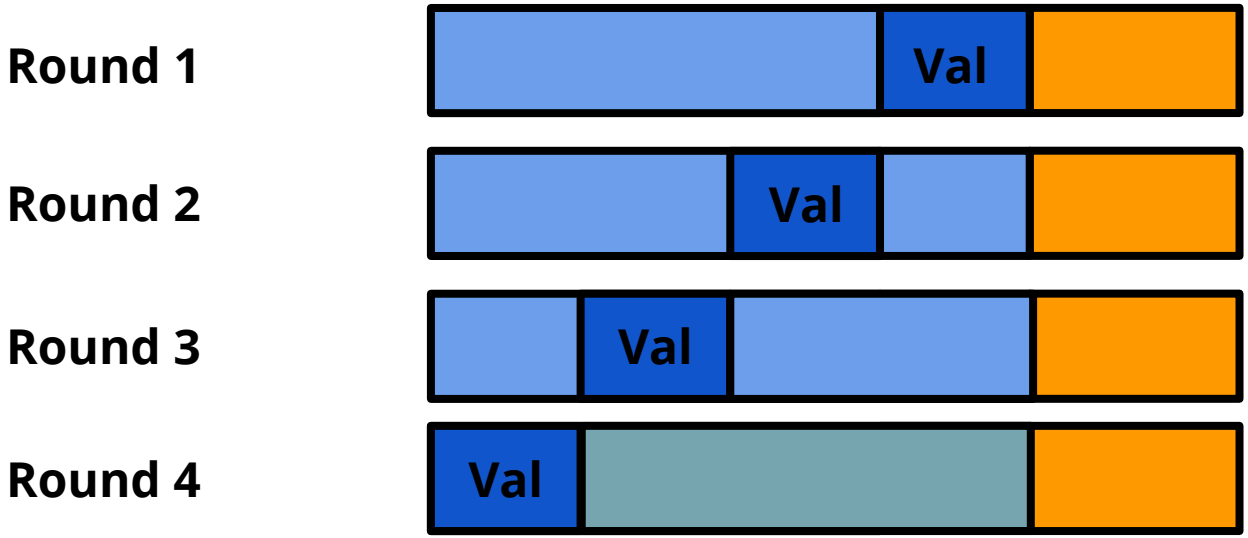
Training procedure - Example - Grid Search



Training procedure - K-Fold Cross Validation

..for a more precise evaluation of the model.

K = 4



Final Accuracy = Average(Round 1,Round 2, Round 3, Round 4)

Homework - K-Nearest Neighbors

1. Load **Wine** dataset
2. Select the first two attributes for a 2D representation of the image.
3. Randomly split data into train, validation and test sets in proportion 5:2:3
4. For **K = [1,3, 5,7]**:
    a. Apply **K-Nearest Neighbors**
    b. Plot the data and the decision boundaries
    c. Evaluate the method on the validation set
5. Plot a graph showing how the accuracy on the validation set varies when changing **K**
6. How the boundaries change? Why?
7. Use the best value of **K** and evaluate the model on the **test set**. How well does it works?

Homework - Linear SVM

8. For **C = [0.001, 0.01, 0.1, 1, 10, 100,1000]**:
    a. Train a **linear SVM** on the training set
    b. Plot the data and the decision boundaries
    c. Evaluate the method on the validation set
9. Plot a graph showing how the accuracy on the validation set varies when changing **C**
10. How the boundaries change? Why?
11. Use the best value of **C** and evaluate the model on the **test set**. How well does it works?

Homework - RBF Kernel

12. Repeat point 8. (train, plot, etc..), but this time use an **RBF kernel**
13. Evaluate the best **C** on the **test set**
14. Are there any differences compared to the linear kernel? How are the boundaries different?
15. Perform a grid search of the best parameters for an RBF kernel: we will now tune both **gamma** and **C** at the same time. Select an appropriate range for both parameters. Train the model and score it on the validation set. Evaluate the best parameters on the test set. Plot the decision boundaries.

Homework - K-Fold

16. Merge the training and validation split. You should now have 70% training and 30% test data.
17. Repeat the grid search for **gamma** and **C** but this time perform 5-fold validation.
18. Evaluate the parameters on the test set. Is the final score different? Why?

Extra : only once you finish the main requests

19. Discuss the difference between KNN and SVM
20. Try also with different pairs of attributes

In practice..

You should use the **scikit-learn** library:

https://scikit-learn.org/stable/

In particular, check these docs for some help:

- https://scikit-learn.org/stable/modules/neighbors.html#neighbors
- https://scikit-learn.org/stable/modules/svm.html
- https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html

Now it’s your turn, try!