Workshop 2 – Advanced Classification

Advanced Analytics and Applications [AAA]

I Revisiting Concepts from the Lecture

II Calculating Linear Decision Boundaries

III Programming

Question 1.1: Linear non-separable data

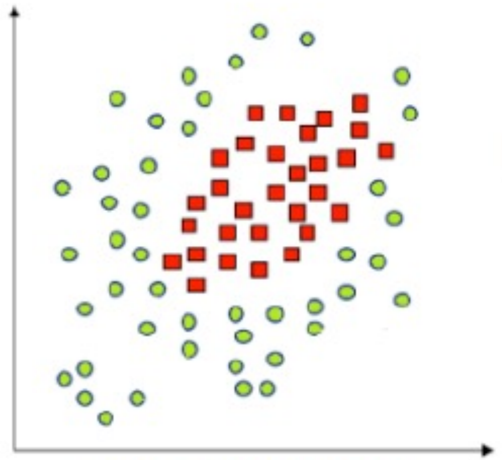


Figure 1 Random Data

Our objective is to find a classifier for the data (on the left).

Why is a linear classifier ineffective for such data?

Question 1.1: Linear non-separable data

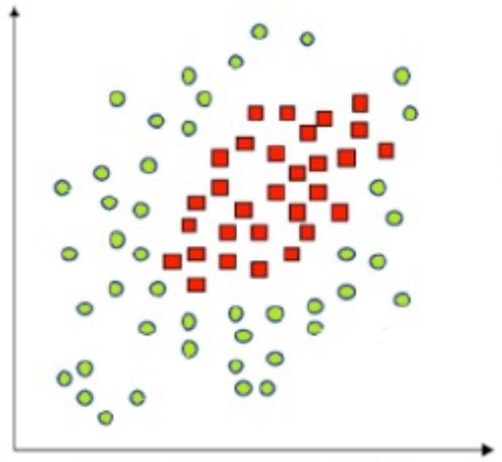


Figure 1 Random Data

Our objective is to find a classifier for the data (on the left).

Why is a linear classifier ineffective for such data?

Solution: Because a typical linear classifier can only separate data using a hyperplane (several solutions might exists). Hyperplanes are a **flat** subspace of the originating Euclidean space. Curved or any other subspaces cannot be calculated with classical linear classifiers.

For the data, there exists no hyperplane to separate the red from the green dots.

Question 1.2: Linear non-separable data

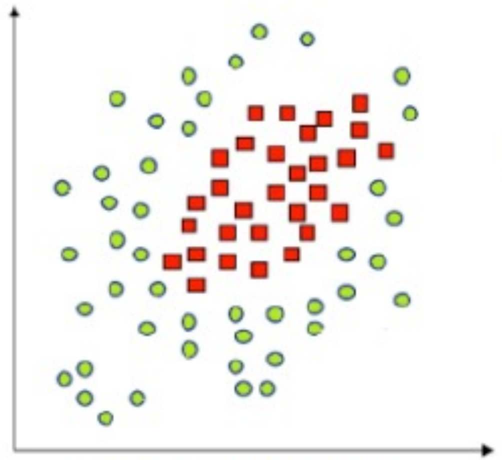


Figure 1 Random Data

Could we train and use a decision tree instead?

Question 1.2: Linear non-separable data

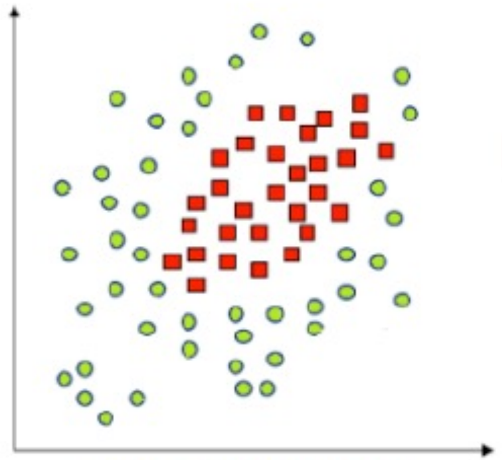


Figure 1 Random Data

Could we train and use a decision tree instead?

Solution: Technically, yes - of course. However, decision trees partition the space and the data into fine rectangles (i.e., decision boundaries). If our data is sampled from a circular distribution (e.g., 2D Gaussian), decision tree might struggle to find clear cut (or precise) decision boundaries.

Question 1.2: Linear non-separable data

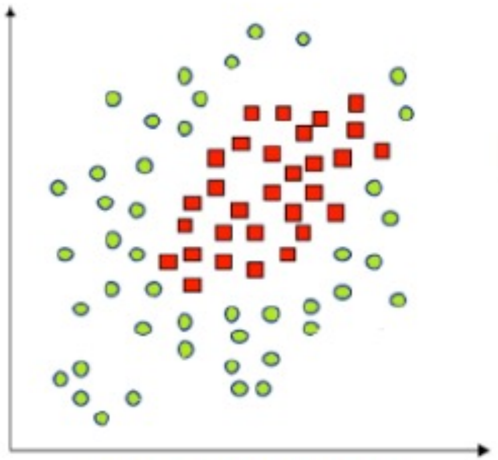


Figure 1 Random Data

How would a classifier using a RBF kernel approach this classification task?

Universität zu Köln

Question 1.2: Linear non-separable data

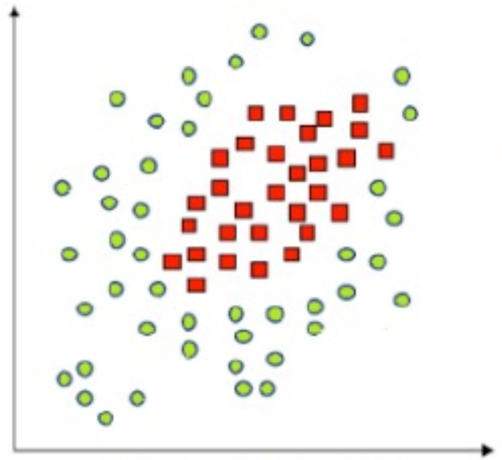


Figure 1 Random Data

How would a classifier using a RBF kernel approach this classification task?

Solution: We have tow data classes: red (-1) and green data (+1) points.

Assign each data point an gaussian-like **exponential function**. Then, we learn (i.e., interpolate) a high (theoretically infinite) dimensional compound function using the kernels assigned to the data points; red kernels are pointed towards negative values, green kernels towards positive values. Thus, our data becomes linearly separable in an infinite dimensional space.

Question 2.1: Support Vector Machines

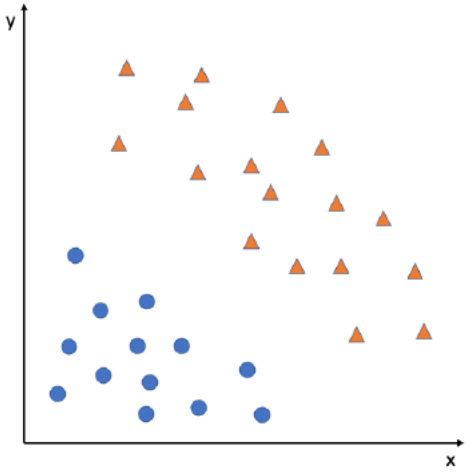


Figure 2 SVM

How would you intuitively separate the data?

Revisit Concepts

Question 2.1: Support Vector Machines

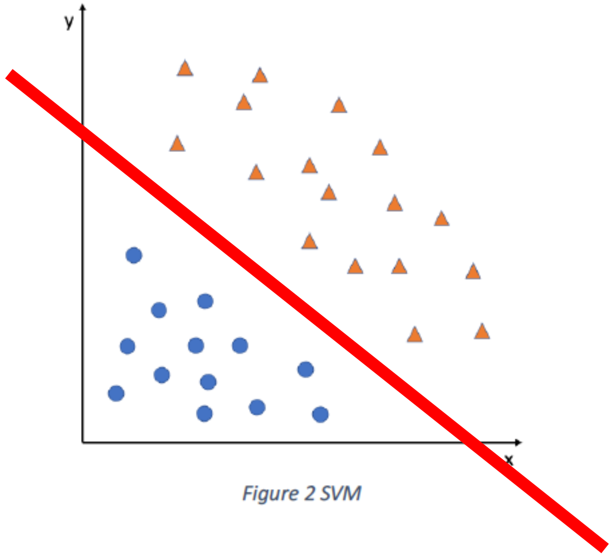


Figure 2 SVM

How would you intuitively separate the data?

Solution: Most humans intuitively place the separating line exactly between the blue and orange data points, with the distance of the line to both classes is typically more or less the same.

Question 2.2: Support Vector Machines

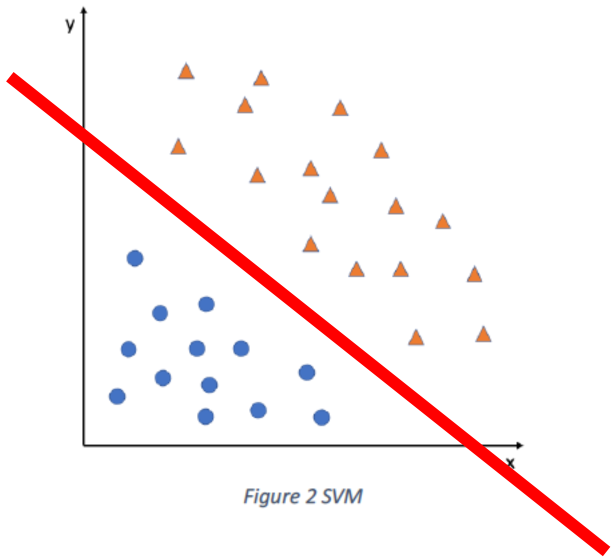


Figure 2 SVM

Draw several linear classifiers that separate the data correctly.

Question 2.2: Support Vector Machines

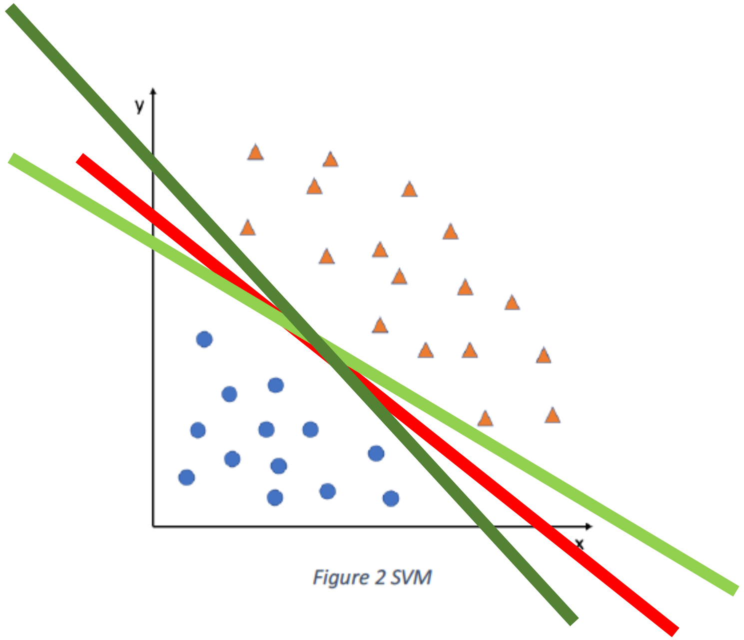


Figure 2 SVM

Draw several linear classifiers that separate the data correctly.

Solution: There are theoretically an infinite number of dividing lines.

Question 2.2: Support Vector Machines

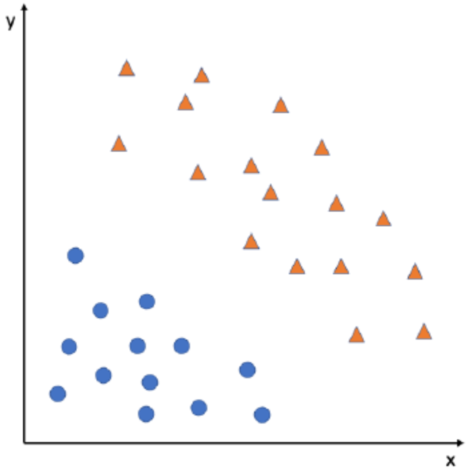


Figure 2 SVM

What would be the output of SVM?

Question 2.2: Support Vector Machines

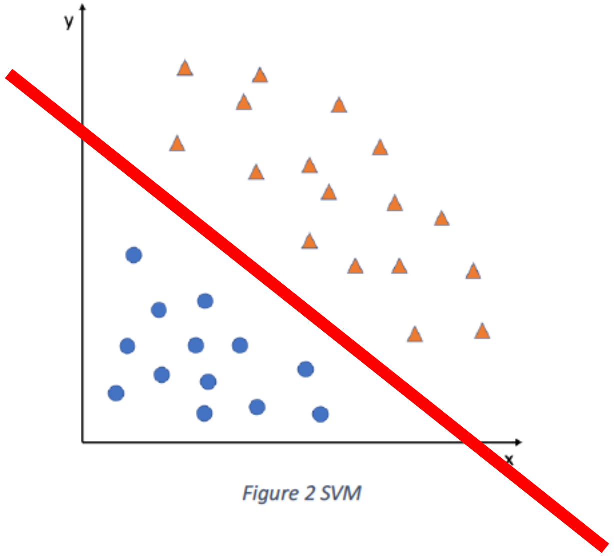


Figure 2 SVM

What would be the output of SVM?

Solution: The output of the SVM would resemble our intuition. A line that separate both data groups but also maximized the margin between the groups and the line.

Question 2.2: Support Vector Machines

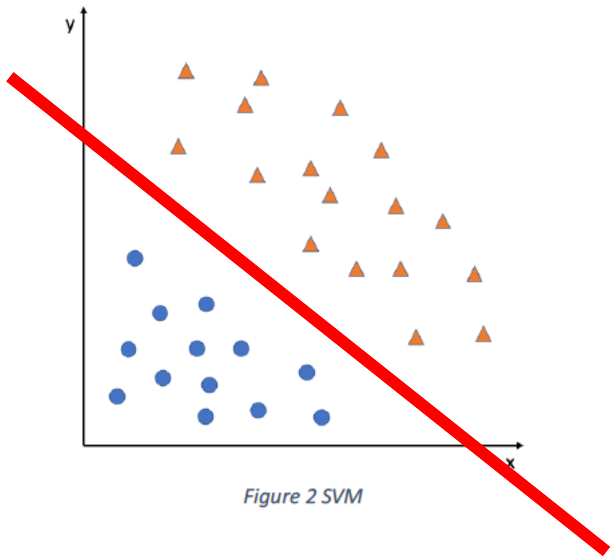


Figure 2 SVM

Briefly explain what happens when a support vector is moved?

Question 2.2: Support Vector Machines

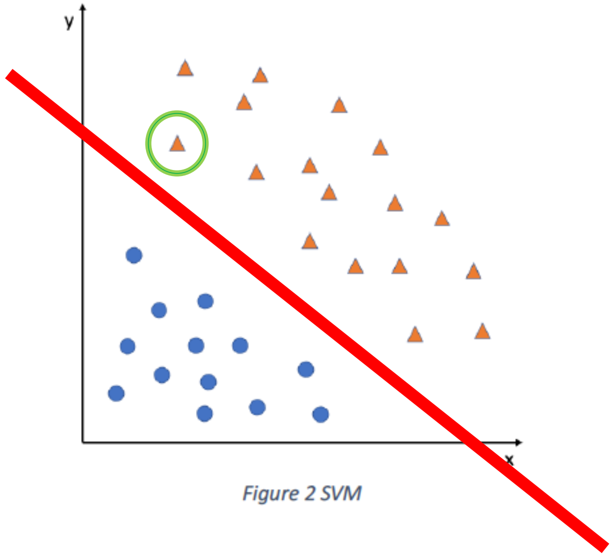


Figure 2 SVM

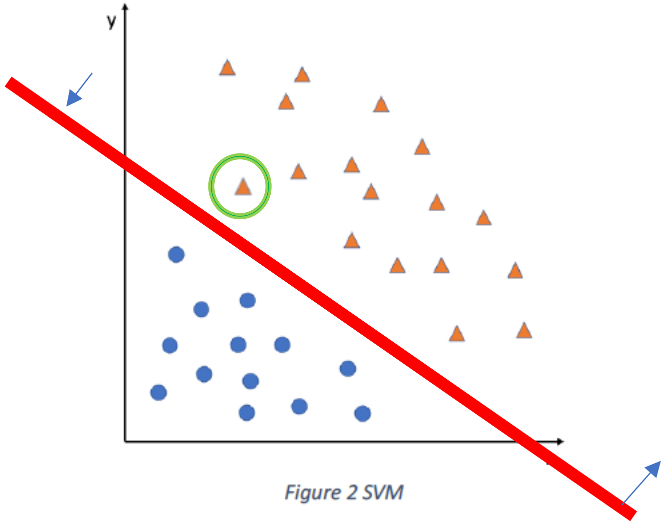


Figure 2 SVM

Briefly explain what happens when a support vector is moved?

Solution: The output of the SVM changes

Question 2.2: Support Vector Machines

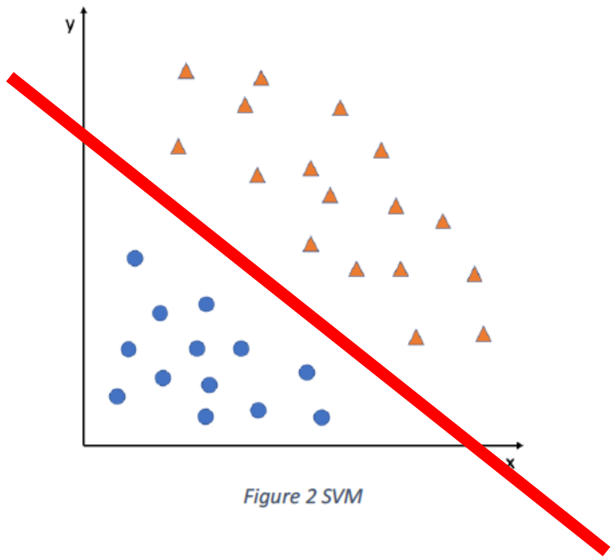


Figure 2 SVM

What happens when other than SV are moved?

Question 2.2: Support Vector Machines

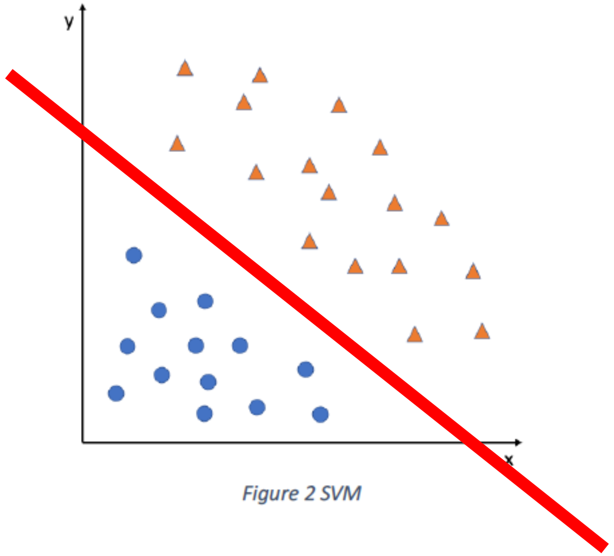


Figure 2 SVM

What happens when other than SV are moved?

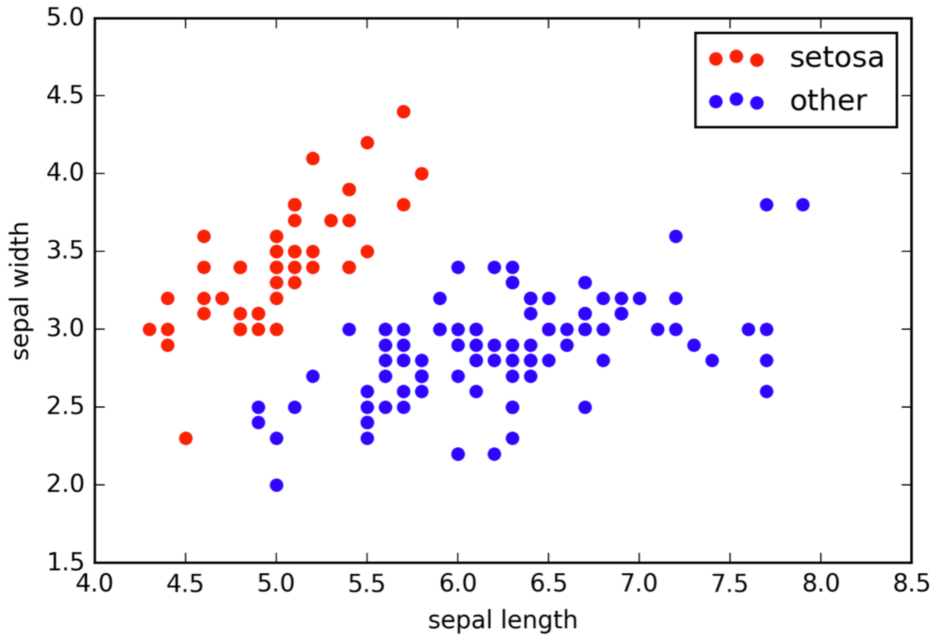
Solution: Nothing

I Revisiting Concepts from the Lecture

II Calculating Linear Decision Boundaries

III Programming

Question 3.2: Perceptron Algorithm



Explain how the perceptron algorithm works.

Prerequisites:

A sequence of N
data points $x^{(i)} \in \mathbb{R}^n, 1 \leq i \leq N$;
each having n characteristic features $x^{(i)} = (x_1^{(i)}, x_2^{(i)}, \ldots, x_n^{(i)})$;
and the task to assign to each element $x^{(i)}$ a label $y^{(i)} \in \{-1, +1\}$;
thereby dividing the data points into two classes labeled -1 and $+1$.

Question 3.2: Perceptron Algorithm

Objective:

The goal of the classification problem is, given some pre-labeled training data:

$$(x^{(i)}, y^{(i)})_{1 \leq i \leq M}, \qquad M < N$$

to make the machine find a function

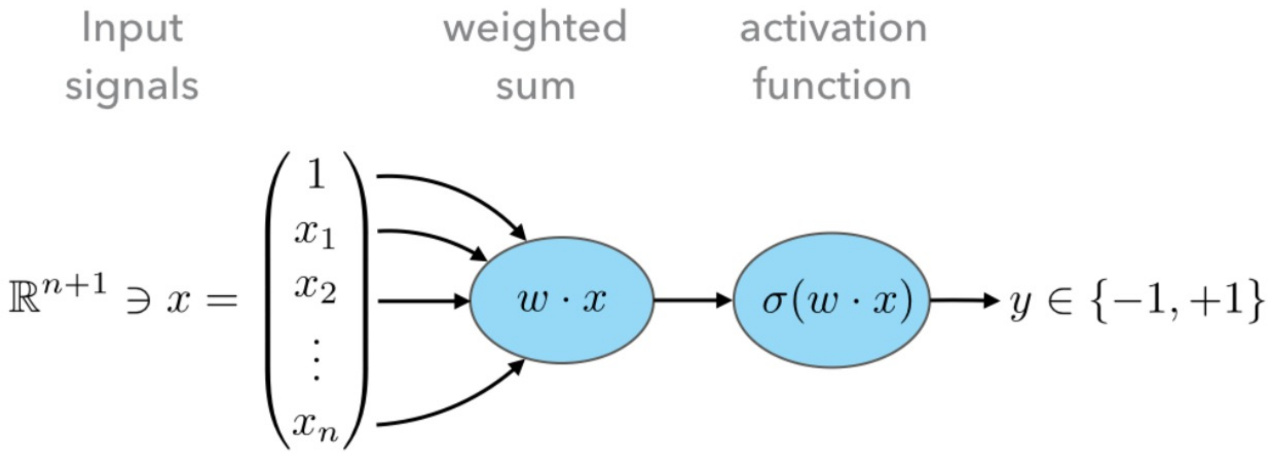
$$f : \mathbb{R}^n \to \{-1, +1\}$$

that:

- predicts *accurately* the labels of pre-labeled training data $(x^{(i)}, y^{(i)})_{1 \leq i \leq M}$, i.e., for *most* indices $1 \leq i \leq M$ it should hold $f(x^{(i)}) = y^{(i)}$;
- and *generalizes* well the remaining data points $x^{(i)}$ for $M > i \leq N$ or even completely unknown data.

Question 3.2: Perceptron Algorithm

Sketch of Perceptron Algorithm:



weight vector

$w \in \mathbb{R}^{n+1}$

signum function

$$\sigma(z) := \begin{cases} +1 & \text{for } z \geq 0 \\ -1 & \text{for } z < 0 \end{cases}$$

Question 3.2: Perceptron Algorithm

Perceptron Learning Rule

Algorithm: (Perceptron Learning Rule)

INPUT: Pre-labeled training data $(x^{(i)}, y^{(i)})_{1 \leq i \leq M \leq N}$

STEP 1: Initialize the weight vector w to zero or conveniently distributed random coefficients.

STEP 2: Pick a data point $(x^{(i)}, y^{(i)})$ in the training samples at random:

i. Compute the output

$$y = f(x^{(i)})$$

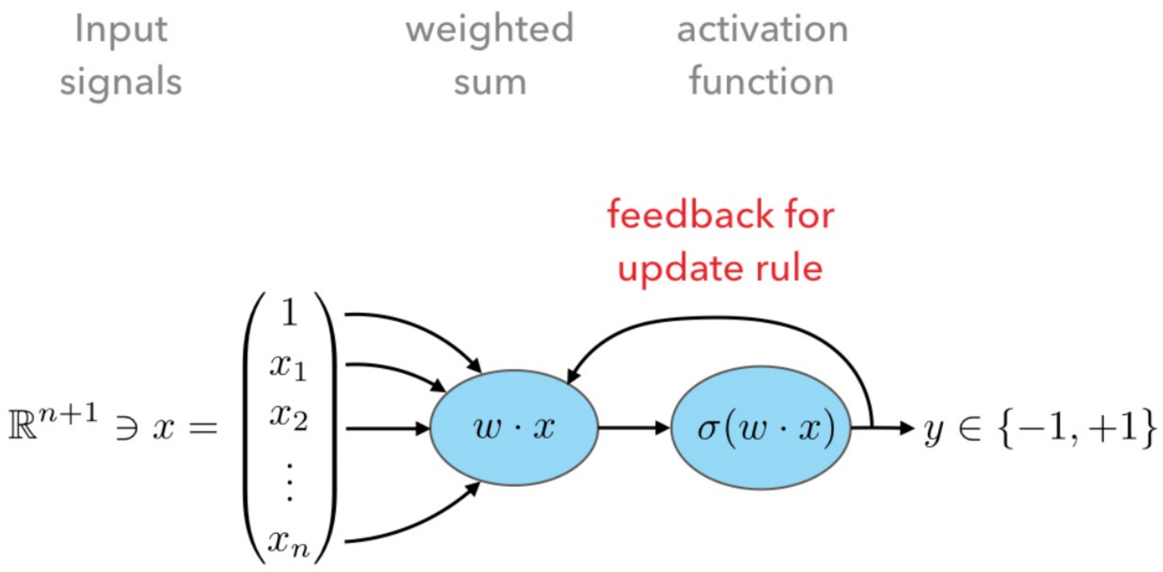
ii. Compare y with $y^{(i)}$:

If $y = y^{(i)}$, go back to **STEP 2**.

Else, update the weight vector w *appropriately* according to an *update rule*, and go back to **STEP 2**.

Question 3.2: Perceptron Algorithm

Revisit the sketch: Feedback Loop



weight vector
$w \in \mathbb{R}^{n+1}$

signum function

$$\sigma(z) := \begin{cases} +1 & \text{for } z \geq 0 \\ -1 & \text{for } z < 0 \end{cases}$$

Question 3.2: Perceptron Algorithm

The update rule

First, we compute the difference between the correct label $y^{(i)}$ given by the training data and the prediction $y = f(x^{(i)})$:

$$\Delta^{(i)} := y^{(i)} - y \tag{3}$$

Second, we perform an update of the weight vector as follows:

$$w \mapsto w^{\text{new}} := w + \delta w \tag{4}$$

where

$$\delta w := \eta \, \Delta^{(i)} \, x^{(i)}. \tag{5}$$

The parameter $\eta \in \mathbb{R}^+$ is called 'learning rate'.

Question 3.2: Perceptron Algorithm

The update rule: Case 1

$\Delta = 2$: This means that the model predicted $y = -1$ although the correct label is $y^{(i)} = 1$.

- Hence, by definition of f in (1) the value of $w \cdot x^{(i)}$ is too low;
- This can be fixed by adjusting the weights according to (4) and (5);
- Next time when this data point is examined one finds

$$\begin{aligned} w^{\text{new}} \cdot x^{(i)} &= (w + \delta w) \cdot x^{(i)} \\ &= w \cdot x^{(i)} + \eta\, \Delta\, (x^{(i)})^2 \\ &\geq w \cdot x^{(i)} \end{aligned}$$

  because, as $\Delta > 0$ and the square is non-negative, the last summand on the right is positive.

- Hence, the new weight vector is changed in such a way that, next time, it is more likely that f will predict the label of $x^{(i)}$ correctly.

Question 3.2: Perceptron Algorithm

The update rule: Case 2

$\Delta = -2$: This means that the model predicted $y = 1$ although the correct label is $y^{(i)} = -1$.

- By the same reasoning as in case 1. one finds:

$$\begin{aligned} w^{\text{new}} \cdot x^{(i)} &= (w + \delta w) \cdot x^{(i)} \\ &= w \cdot x^{(i)} + \eta \, \Delta \, (x^{(i)})^2 \\ &\leq w \cdot x^{(i)} \end{aligned}$$

because now we have $\Delta < 0$, and again, the correction works in the right direction.

Revisit Concepts

Question 3.3: Perceptron Algorithm

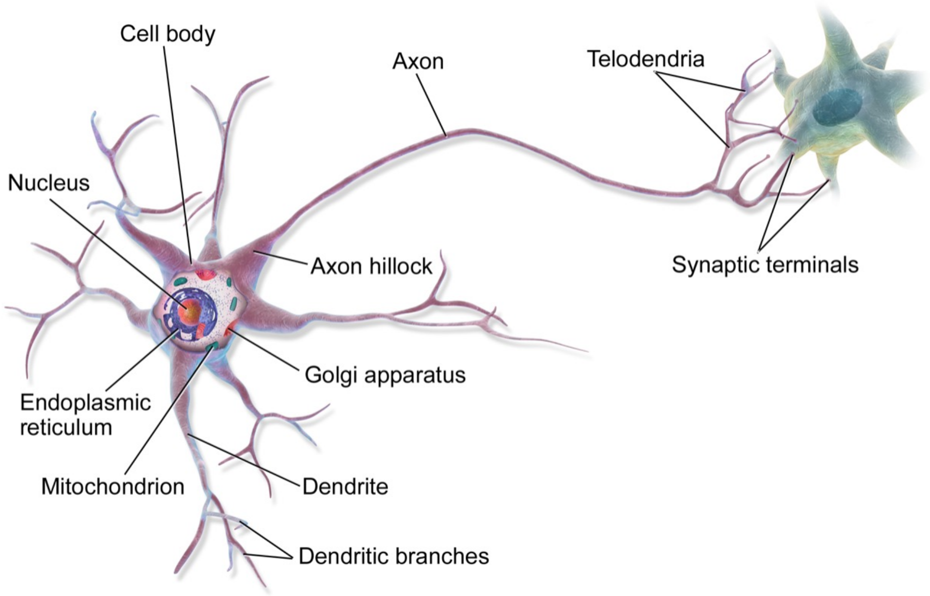


Fig. 4 A sketch of a neuron (source).

What is the key difference between the perceptron algorithm and a svm classifier?

Solution: -Optimality condition (i.e., maximum margin)
- Perceptron is an iterative algorithm which stops once one valid classifier has been found.

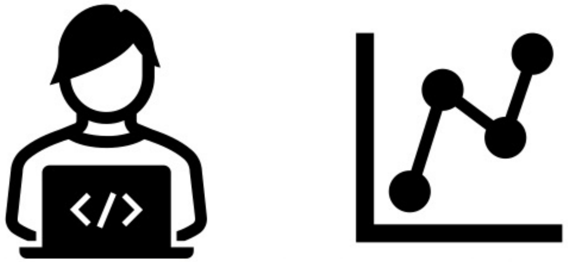
Universität zu Köln

I Revisiting Concepts from the Lecture

II Calculating Linear Decision Boundaries

III Programming

Call for Application for the role of **Development Student Assistant**



The Department for Information Systems for Sustainable Society (IS3) is inviting applications for the role of student assistant (8h / week), starting immediately.

We are looking for **advanced Bachelor's** or **early stage Master's** students with information systems, computer science, economics or business management backgrounds who are interested to join our team in the following role:

Development Student Assistant (SHK / WHB)
As a development student assistant, you will support our researchers in data acquisition, data base management, machine learning and analytics tasks and work hands-on with large-scale research simulation platforms. Occasionally there might be other, administrative tasks, or exam corrections you can help us with.

You should have a **strong technical background** (Python or Julia are a necessity, R is welcome), **analytical skills** and **readiness to acquire new technical skills**.

SHK (Bachelor): 12€/h
WHB (Master): 13,20€/h

Please send any inquiries and applications (CV and grades in one PDF) to is3-academic@uni-koeln.de.

If you have questions, reach out to Janik Muires (muires@wiso.uni-koeln.de) or Ramin Ahadi (ahadi@wiso.uni-koeln.de) in the workshops or via mail.

Universität zu Köln

67

Call for Application for the role of Student Assistant

The Department for Information Systems for Sustainable Society (IS3) is inviting applications for the role of student assistant (8h / week), starting immediately.

We are looking for **advanced Bachelor's** or **early stage Master's** students with information systems, computer science, economics or business management backgrounds who are interested to join our team in the following role:

Student Assistant (SHK / WHB)
As a student assistant, you will support our researchers in collecting documents, prepare reports, presentations and dissemination in various areas particularly energy markets, algorithmic trading, decentralized finance, and machine learning. Occasionally there might be some administrative tasks, or exam corrections, and analytics tasks you can help us with.

You should have a **strong writing and research skills. Technical background** (i.e. Python or Julia), and **analytical skills** are advantages.

SHK (Bachelor): 12€/h
WHB (Master): 13,20€/h

Please send any inquiries and applications (CV and grades in one PDF) to is3-academic@uni-koeln.de.

If you have questions, reach out to Saber Talari (talari@wiso.uni-koeln.de) or Julius Kuhmann(kuhmann@wiso.uni-koeln.de) in the workshops or via mail.

Contact

For general questions and enquiries on **research**, **teaching**, **job openings** and new **projects** refer to our website at www.is3.uni-koeln.de

For specific enquiries regarding this course contact us by sending an email to the **IS3 teaching** address at is3-teaching@wiso.uni-koeln.de