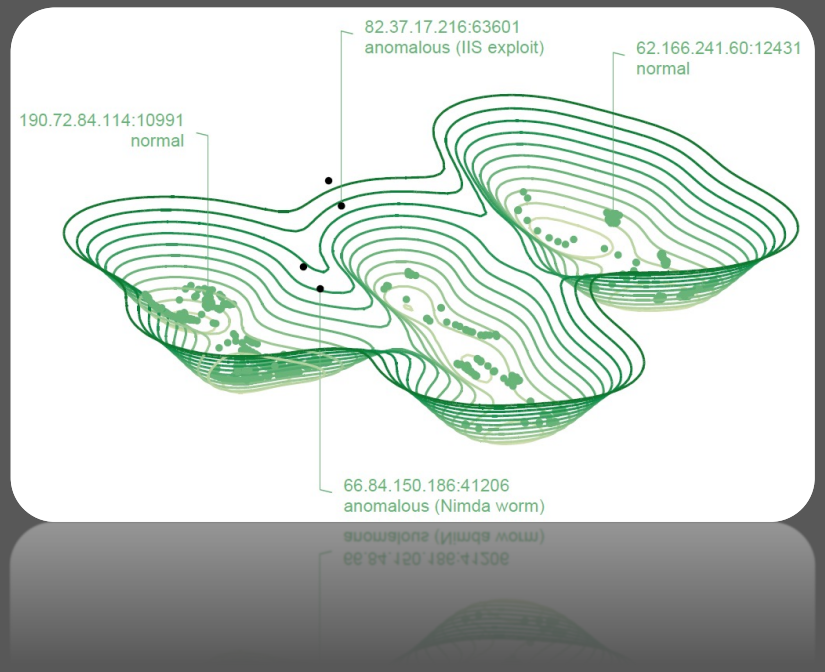


Machine Learning for Security

… a security AI component of module 6 “Intelligent Interaction Design”

Teaching staff

Dr. Andrea Continella
Assistant Professor in Cybersecurity

Office: ZI 2007
E-Mail: a.continella@utwente.nl

Services and Cybersecurity (SCS)

Services Cyber Security

first contact point

Thijs van Ede

Office: ZI 2042
E-Mail: t.s.vanede@utwente.nl

Services and Cybersecurity (SCS)

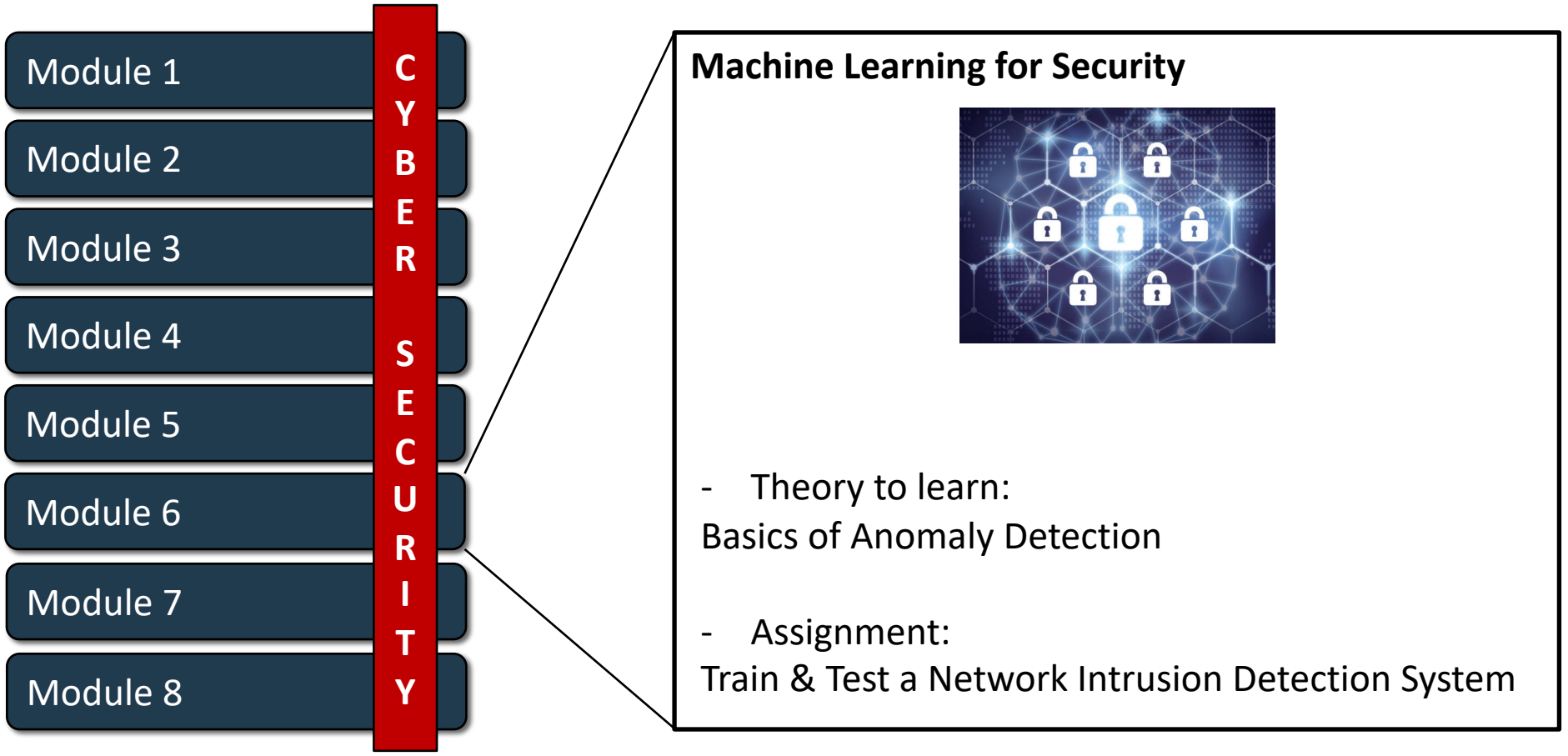
Mohammed El-hajj

Office: ZI 2031
E-Mail: m.elhajj@utwente.nl

Services and Cybersecurity (SCS)

Cross-Cutting Concern Security

The red Cyber Security line in the Bachelor computer science



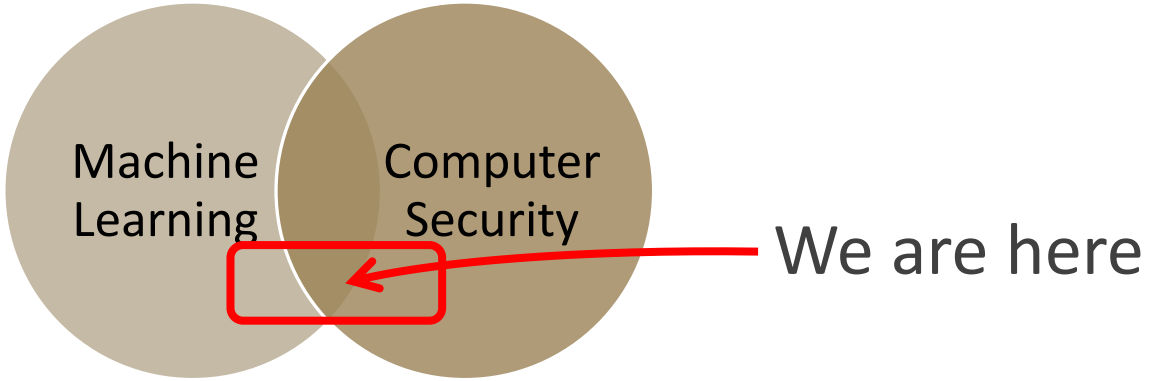
About this Lecture (1/2)

- Focus on the *application* of ML in security
- Brief intro to security
- Application of Machine Learning in Anomaly Detection to build a Network Intrusion Detection System

About this Lecture (2/2)

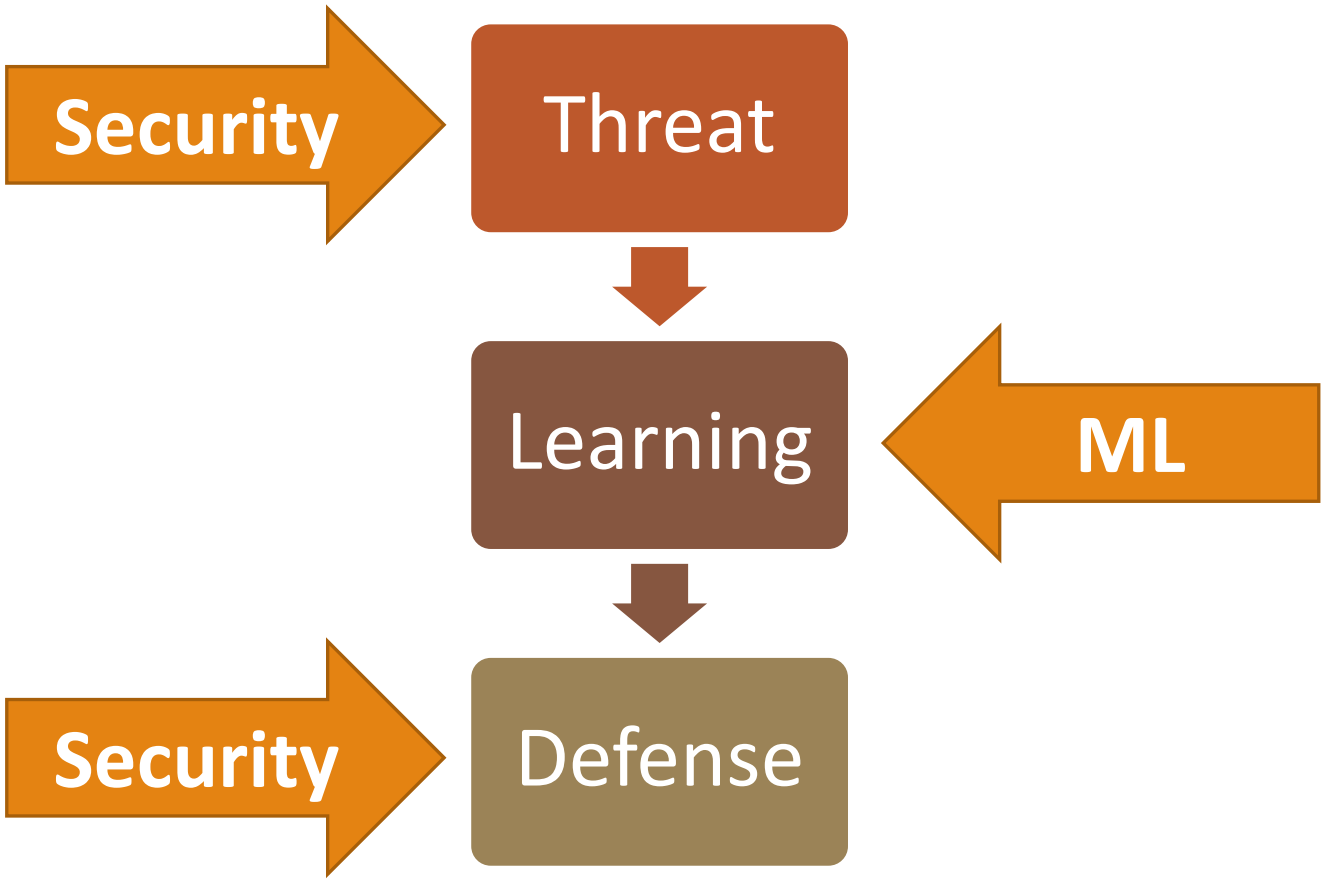
- Relevant for exam: questions on “ML for Security”
- Prepares for mandatory **hands-on assignment** (later)
- Thursday’s tutorial (10:45 – 12:30) is not a tutorial, but a **Q&A session**
- **Deadline for assignment: 11 January 2022, 23:59**

Machine Learning for Computer Security



Security & Machine Learning

Security & Machine Learning



Security

Security

Goals – CIA paradigm

- ***C****onfidentiality* of information and resources
- ***I****ntegrity* of information and resources
- ***A****vailability* of information and resources

Definitions

- **T**hreat = potential violation of a security goal
- **A**ttack = intentional violation of a security goal
- **S**ecurity = protection from intentional threats vs. cost

Security is an engineering problem!

Attackers, Hackers, …

Mass media created false myths and controversies around these and other words

Hacker: someone with an *advanced understanding* of computers and computer networks, and willingness to learn "everything"

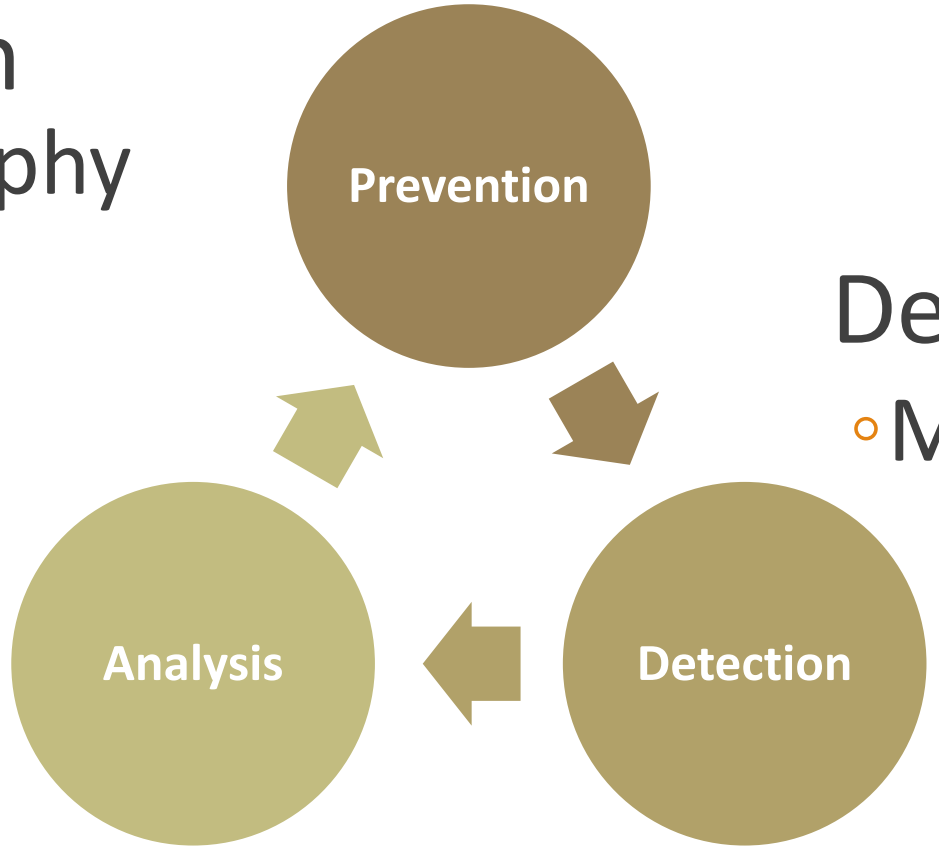
Black hats: malicious hackers

Attacker != hacker

Security

Prevention

- Cryptography



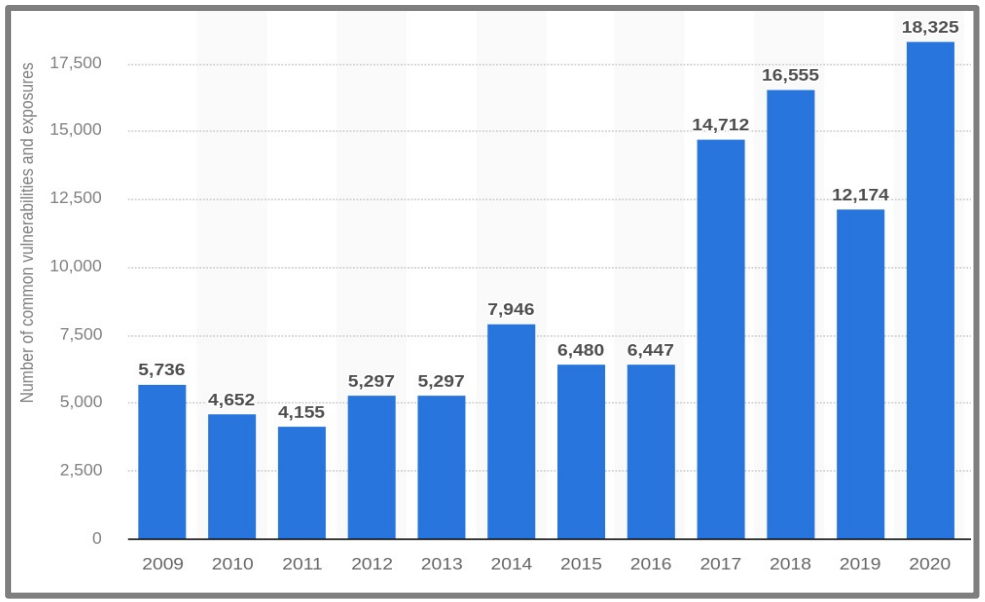
Detection

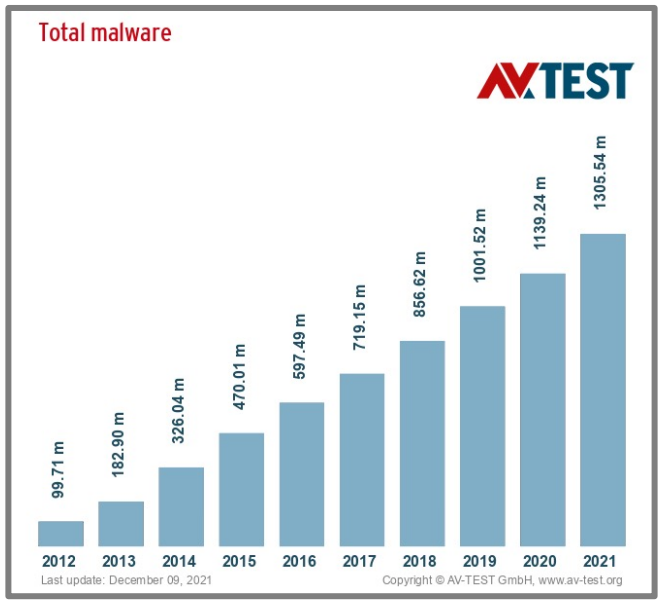
- Malware Detectors

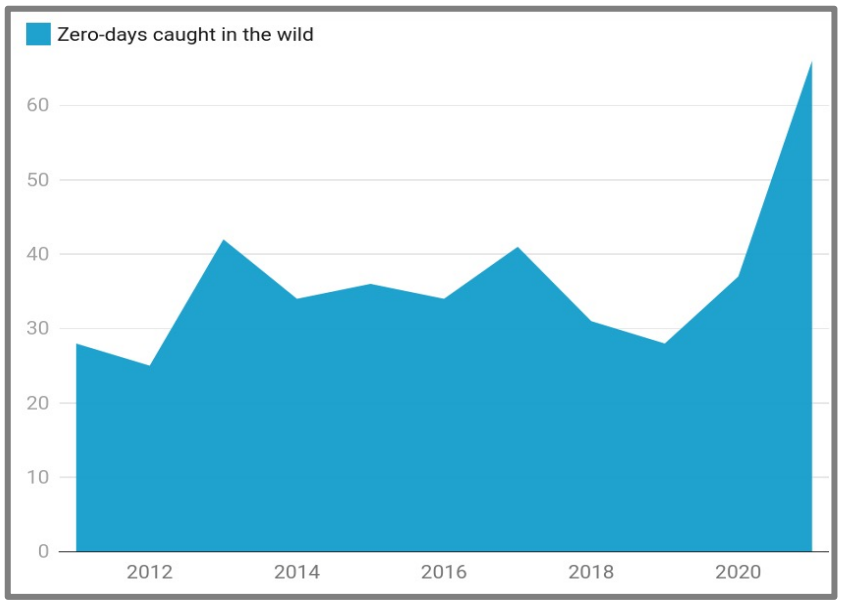
Analysis

- Forensics

Security


Number of common vulnerabilities and exposures
17,500
15,000
12,500
10,000
7,500
5,000
2,500
0
5,736
4,652
4,155
5,297
5,297
7,946
6,480
6,447
14,712
16,555
12,174
18,325
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020


Total malware
AV-TEST
99.71 m
182.90 m
326.04 m
470.01 m
597.49 m
719.15 m
856.62 m
1001.52 m
1139.24 m
1305.54 m
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
Last update: December 09, 2021
Copyright © AV-TEST GmbH, www.av-test.org


Zero-days caught in the wild
60
50
40
30
20
10
0
2012
2014
2016
2018
2020

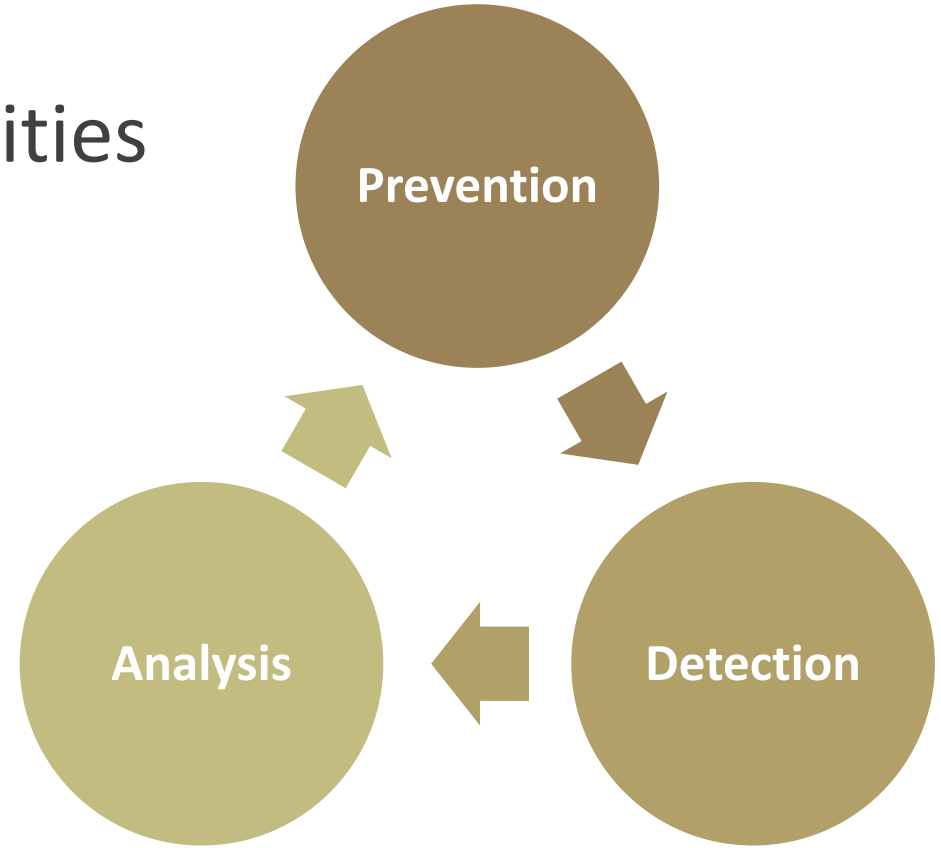
Security: Challenges

Imbalance

- Increasing amount of vulnerabilities
- Novel attack vectors
- High diversity of malware

Bottlenecks

- Discovery of vulnerabilities
- Generation of attack signatures
- Malware analysis



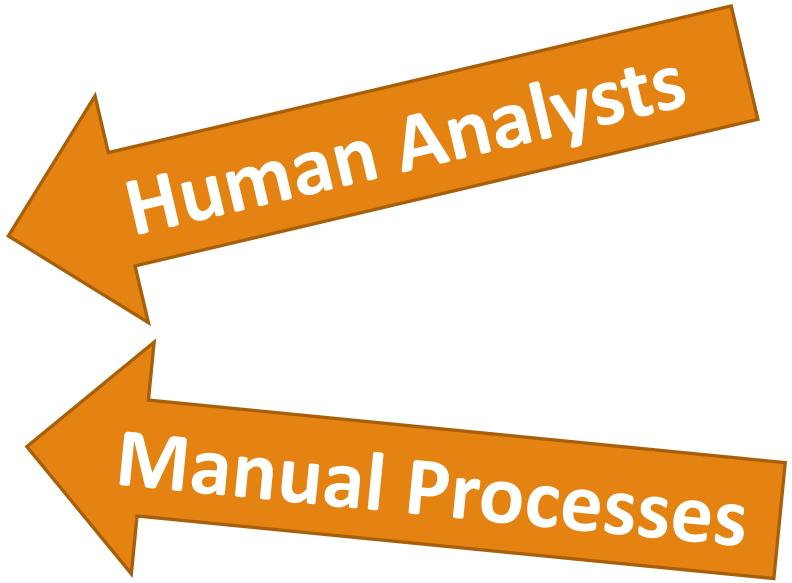
Security: Challenges

Imbalance

- Increasing amount of vulnerabilities
- Novel attack vectors
- High diversity of malware

Bottlenecks

- Discovery of vulnerabilities
- Generation of attack signatures
- Malware analysis



Security

- Increasing number of vulnerabilities and threats
- Bottleneck: Human Analyst
  - Manual prevention, detection and analysis
- Approach: "Smarter" security
  - Automate processes
  - Overcome shortcomings of manual actions

Machine Learning

SHORT REMINDER

Reminder on Machine Learning

- Automatic inference of dependencies from data
- Generalization of dependencies
- Abstraction, no memorization
- Application of learned dependencies to unseen data

Reminder on Machine Learning

More formally:

- Learning model θ
  - Encodes generalized dependencies
- Prediction function f_θ (parametrized by θ)
  - Outputs a prediction on given data
- Error function E
  - Assesses the progress of learning

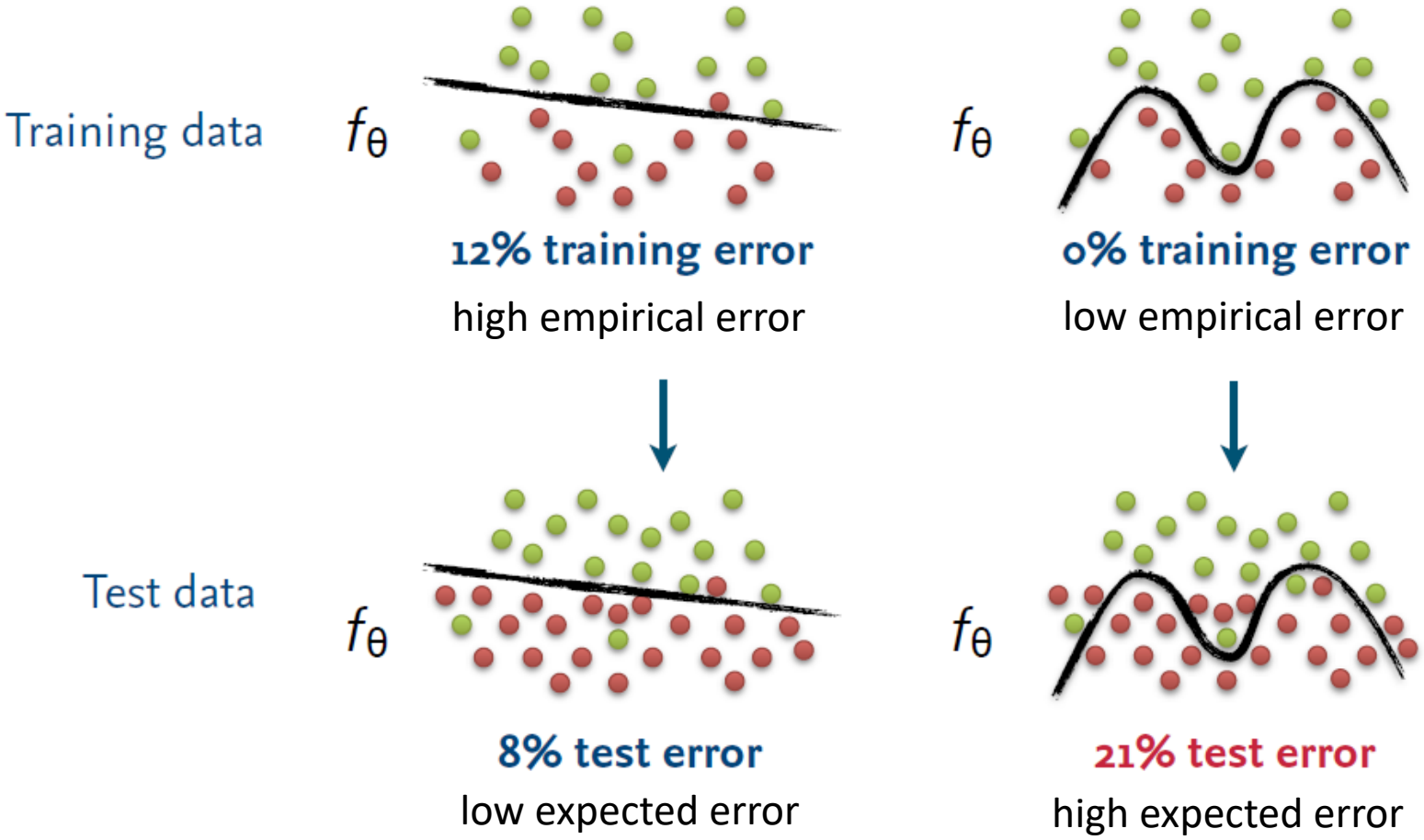
ML = Optimization problem seeking a learning model θ such that the expected error $E(f_\theta)$ of the prediction function f_θ is minimized.

Reminder on Machine Learning

ML = Optimization problem seeking a learning model θ such that the expected error $E(f_\theta)$ of the prediction function f_θ is minimized.

- In practice: we are given *n* samples of training data and can only determine the *empirical error* $E_n(f_\theta)$.
- Problem: minimizing the empirical error is not sufficient for learning accurate models

Overfitting Problem



Remedy: k-fold Cross Validation

- Estimating error on unseen data
- Testing parameters
- Splitting training data into training and test sets

k = 5	**Training Data Chunks**				
Run 1	Train	Train	Train	Train	Test
Run 2	Train	Train	Train	Test	Train
Run 3	Train	Train	Test	Train	Train
Run 4	Train	Test	Train	Train	Train
Run 5	Test	Train	Train	Train	Train

Network Intrusion Detection System (NIDS)

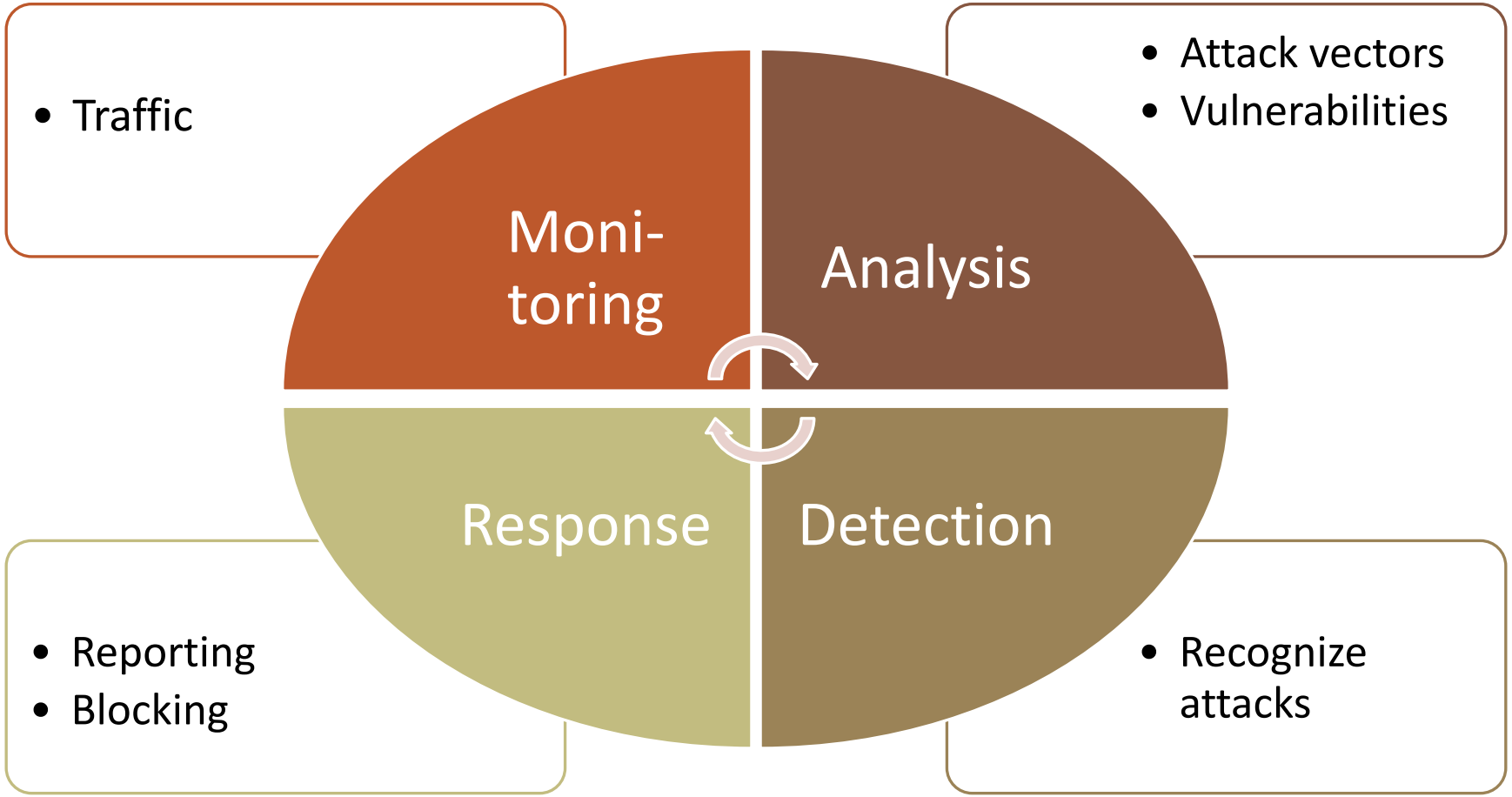
APPLYING MACHINE LEARNING IN SECURITY

Intrusion Detection

- **I**ntrusion **D**etection **S**ystem: Monitoring for attacks
- Attack: attempt to compromise
  confidentiality, integrity, or availability (CIA)
  of a resource or information
- Several types of IDS
  - Monitor source: network, application, server, ...
  - Analysis type: signatures, rules, machine learning, ...
  - Response type: report, block, sandbox, ...

Network Intrusion Detection

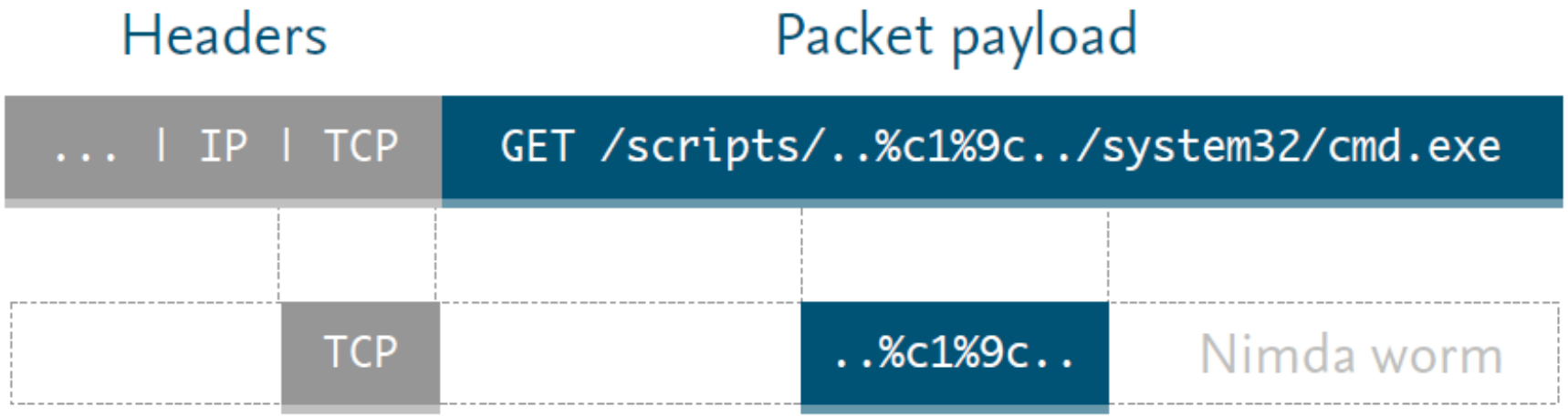
Network Intrusion Detection



Network Intrusion Detection

Signature-based NIDS

- Pattern detection (RegEx, rules, payload)



Network Intrusion Detection

Signature-based NIDS

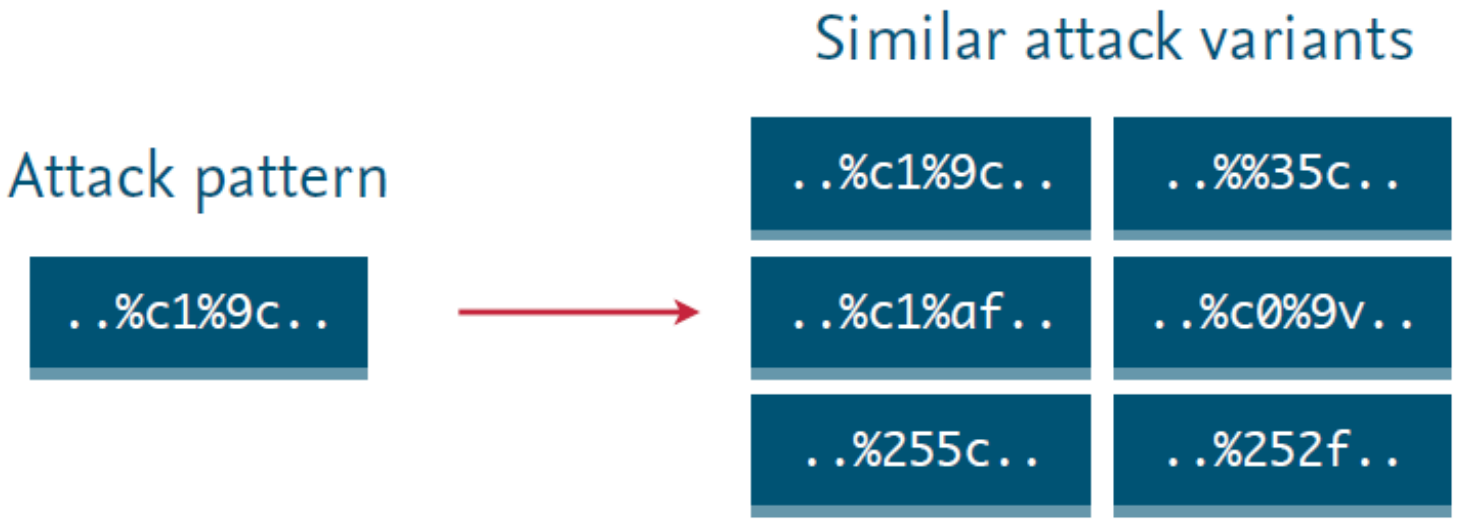
- Pattern detection (RegEx, rules, payload)

- Signatures from known attacks required
- Signature updates required
- Scalability issues, complexity and attack frequency
- Unable to detect novel or unknown attacks

Network Intrusion Detection

Signature-based NIDS

- Unable to detect novel or unknown attacks



Network Intrusion Detection

ML-based NIDS

- Automatic detection of attacks
- Can work without signatures (hybrid concepts exist)
- Different approaches, different requirements

Network Intrusion Detection with ML

Effectivity

- Detection rate vs. false alarms

Efficiency

- Processing speed (several MB per second)

Robustness

- Resistance against evasion attacks – adversarial scenario!
- Resistance against evolving attacks and aging

Network Intrusion Detection with ML

Approaches

- Focus on malicious activity only
  - Signatures
- Focus on benign activity only
  - Anomaly detection

our focus

- Focus on differences of malicious and benign activity
  - Classification

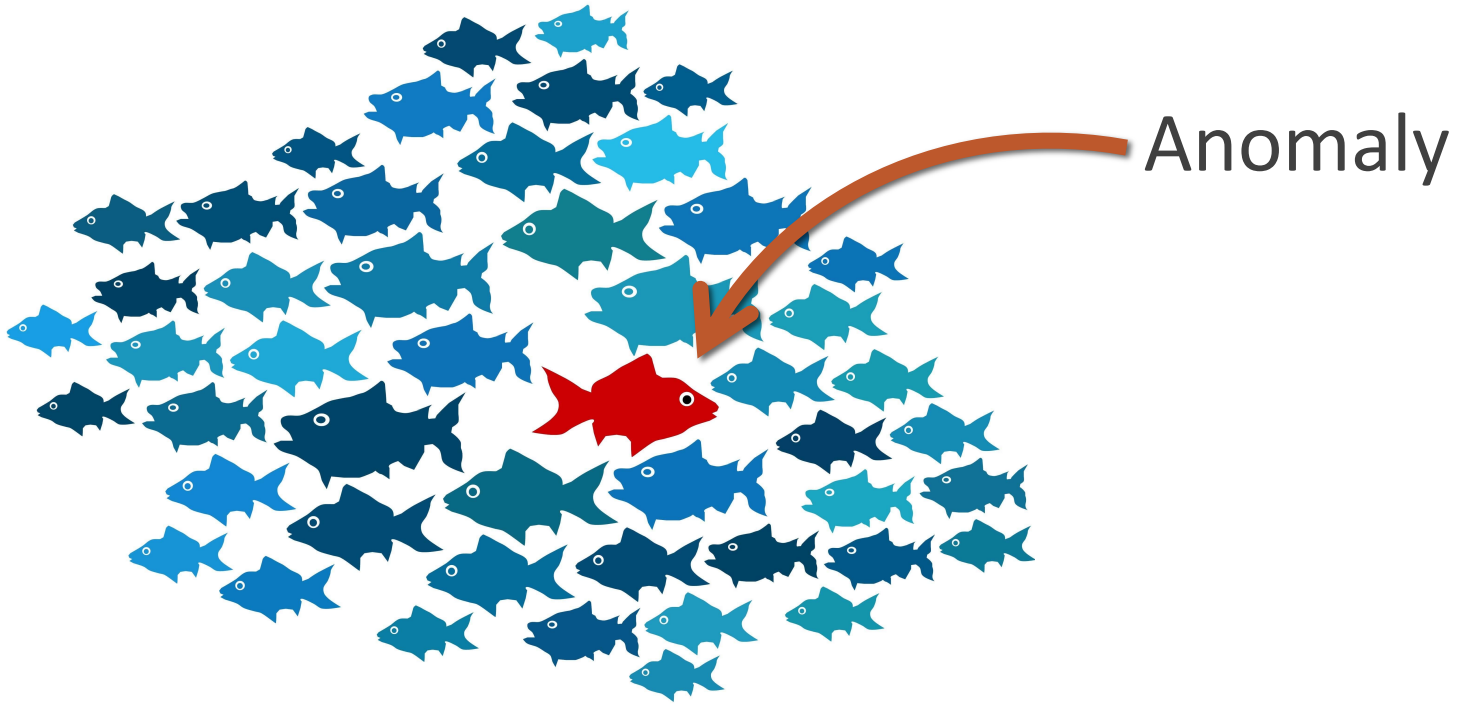
Anomaly Detection

- Learning of a model of normality
- Detection of deviations from normality

- Assumptions
  - Mainly benign training data
  - Unknown attacks differ from benign data

Anomaly Detection

- Easy example: spot the anomalous fish
  - Features
    - Color
    - Size
    - Direction



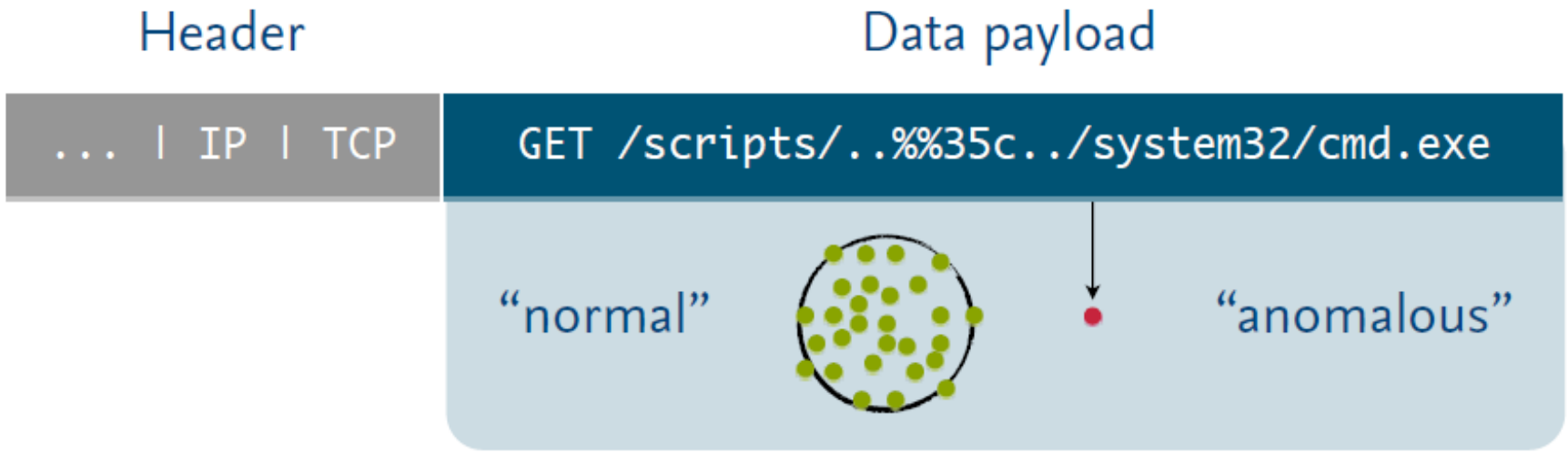
Anomaly Detection

- Another example: who is not Homer?
  - Features
    - Size
    - Accessories
    - ...

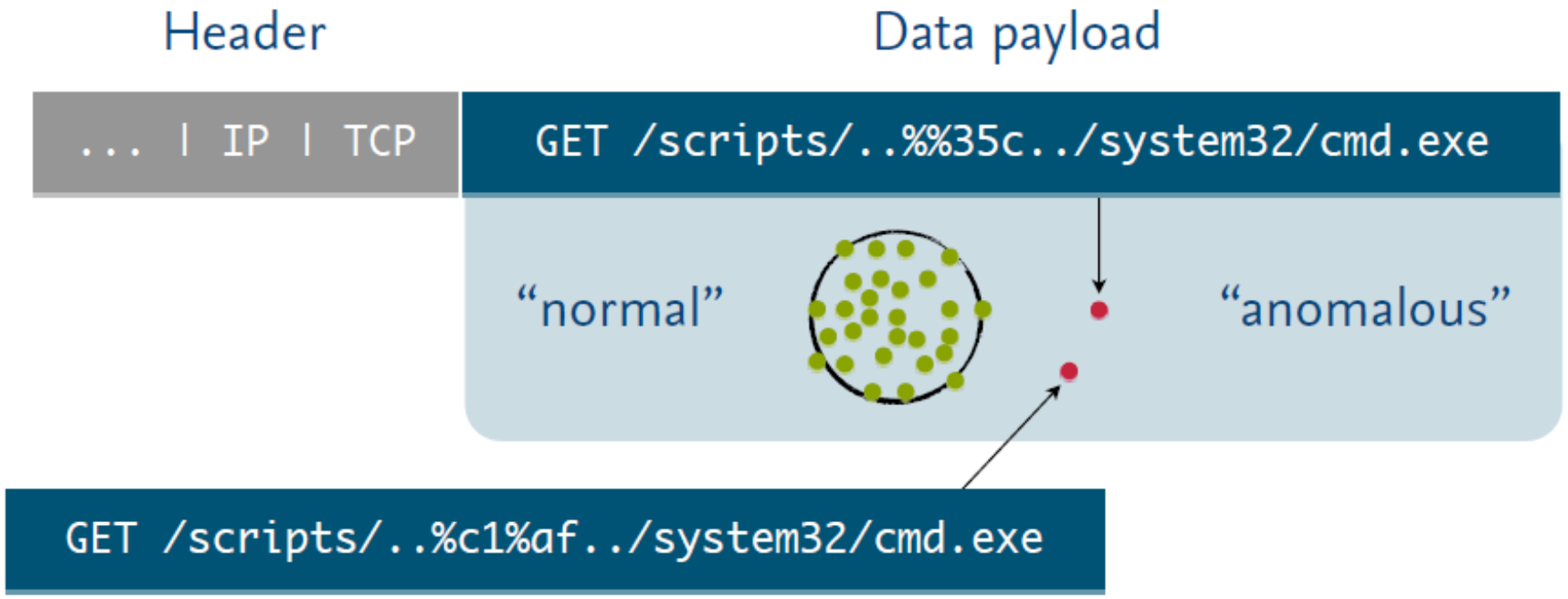
Anomaly Detection

- Risk: detection of irrelevant anomalies as attacks
- Choice of features is crucial
- Attacks are deviations from normality

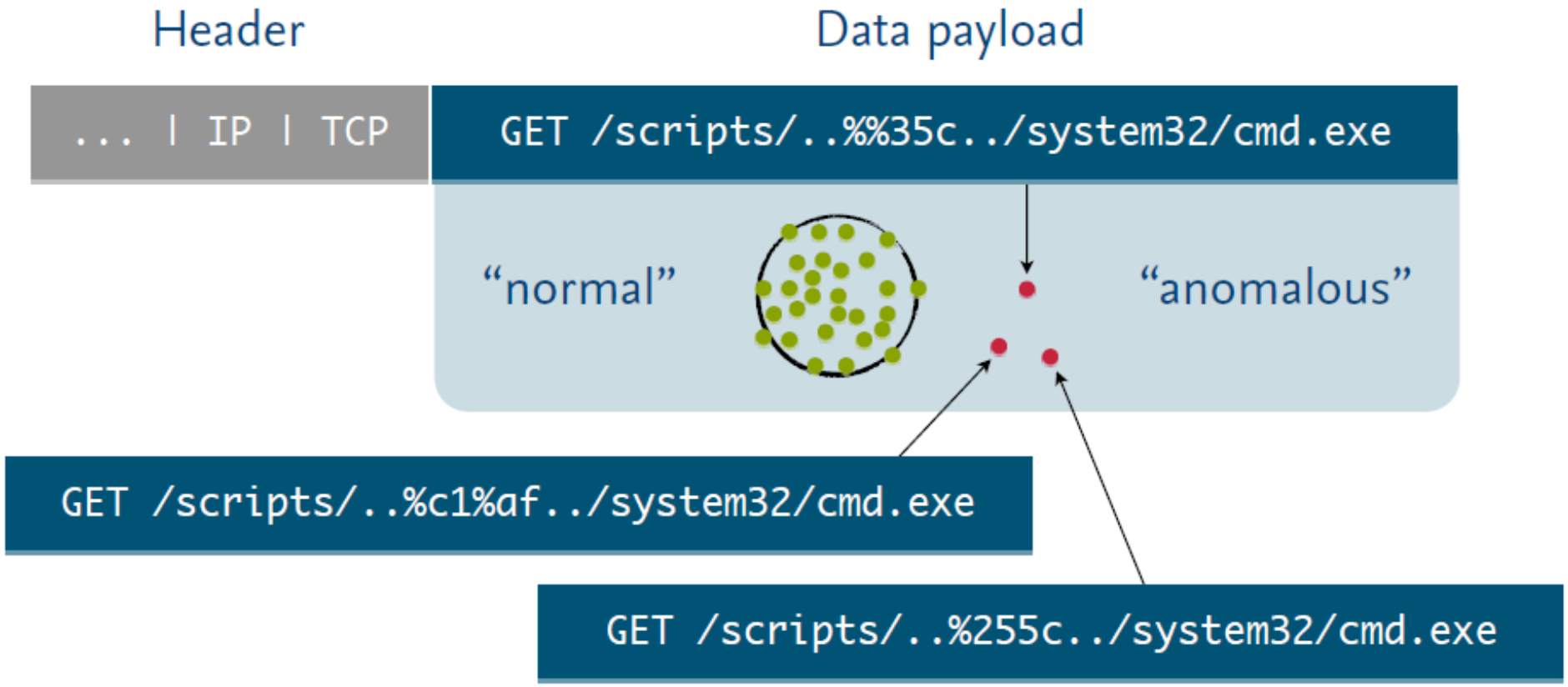
Anomaly Detection

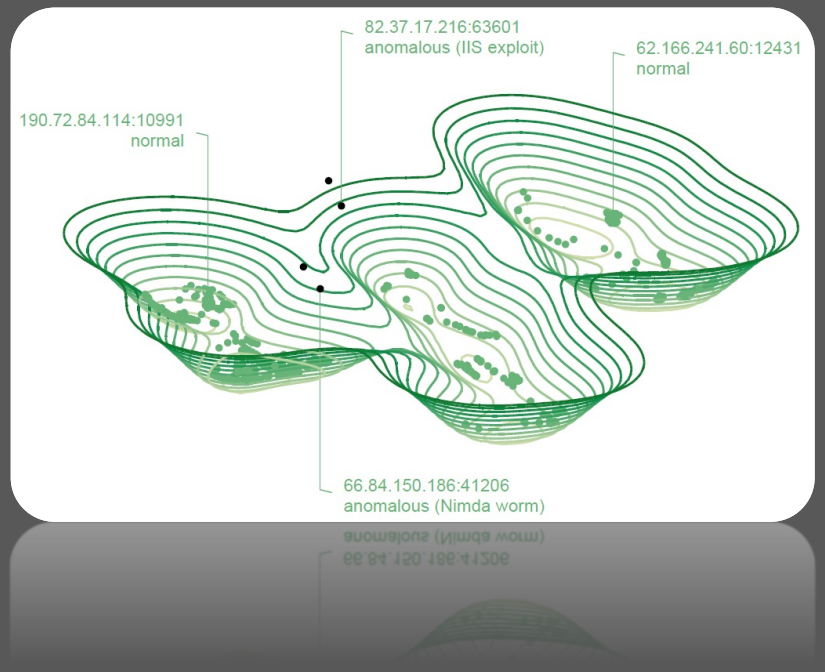


Anomaly Detection



Anomaly Detection





Machine Learning for Security

… a security AI component of module 6 “Intelligent Interaction Design”

Anomaly Detection One-Class SVM

APPLYING MACHINE LEARNING IN SECURITY

Anomaly Detection

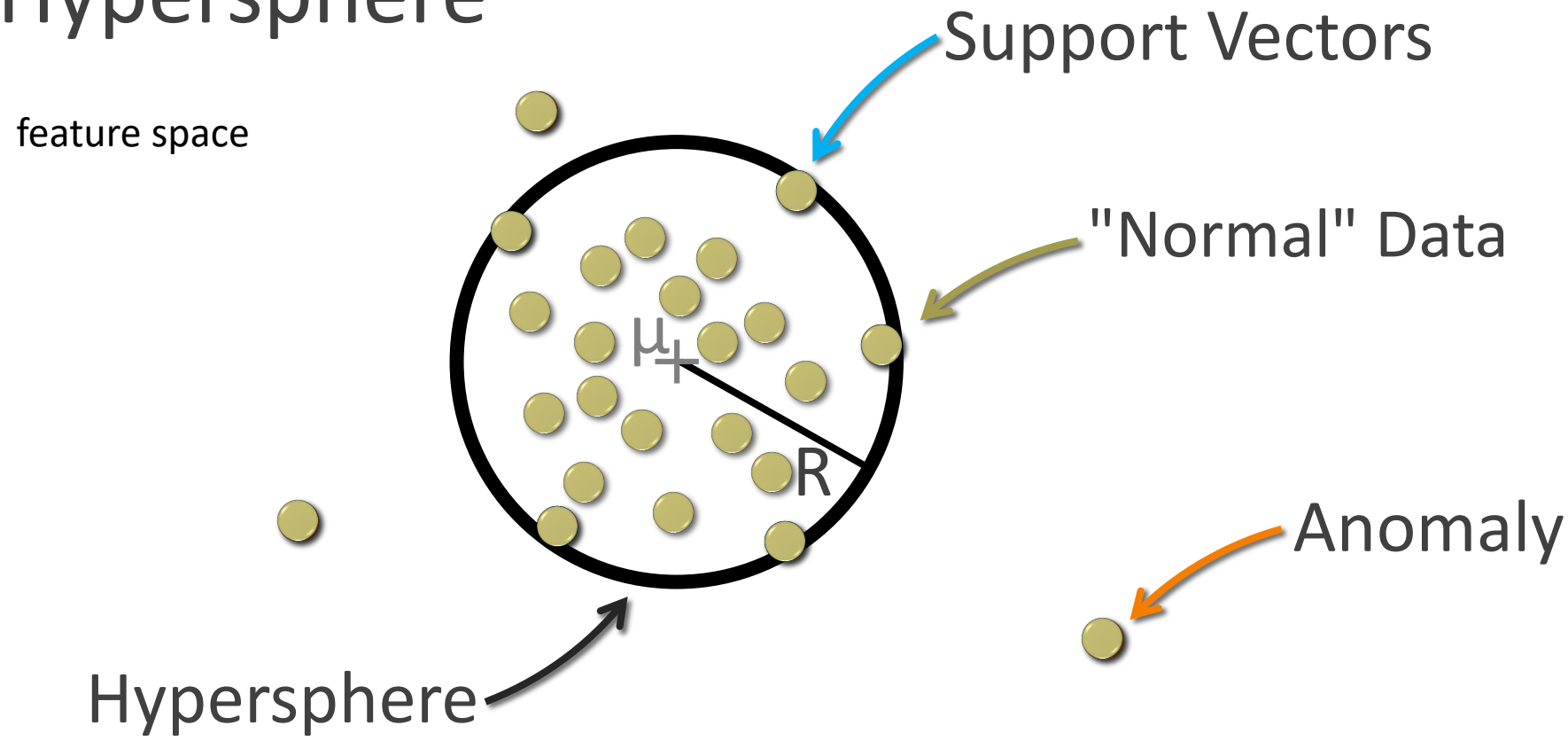
Algorithm: One-class Support-Vector-Machine

- Standard algorithm for anomaly detection
- Effective, efficient and robust

- Hypersphere enclosing data with minimum volume

One-Class SVM: Intuition

Hypersphere



One-Class SVM

Hypersphere

- Encloses data
- Minimum volume R^2 desired: $\min_{\mu,R} R^2$

- What does this really mean? More detailed look …

One-Class SVM: in more detail

- $D = \{d_1, d_2, \ldots\}$ input domain (e.g., TCP packets)
- **Feature map** $\phi: D \longrightarrow \mathbb{R}^N$ maps input items to a vector space $\mathbb{R}^N$ of real numbers:

  $$d \longmapsto \phi(d) = (\phi_1(d), \ldots, \phi_N(d))$$ for some N > 0

  (N = number of features; feature i is associtated with $\phi_i(d)$)
- Example features: packet size, timestamp, …

One-Class SVM: in more detail

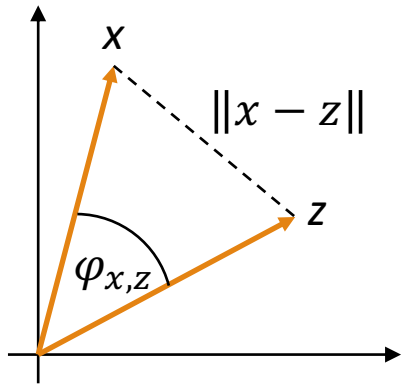
- Feature map $d \longmapsto \left(\phi_1(d), \ldots, \phi_N(d)\right)$ („**feature vector**“)
- Each $\phi_i(d)$ may exhibit a different numerical scale
- Solution: **Min-Max Normalization**

$$\overline{\phi_i}(d) = \frac{\phi_i(d) - min_i}{max_i - min_i} \in [0,1]$$

where min_i / max_i = min. / max. value of feature *i*

One-Class SVM: in more detail

- Dependencies not in plain feature vectors, but in the geometry and relationships induced by the vector space
- Solution: **Kernels**



- **Linear kernel** (inner product)

For two feature vectors *x*, *z*: $\kappa(x, z) = \langle x, z \rangle = \sum_{i=1}^{N} x_i z_i$

- Geometric relationships can be described via kernels, e.g.:
  - ✓ $\|x\| = \sqrt{\kappa(x, x)}$ *"Euclidean norm"*
  - ✓ $\|x - z\| = \sqrt{\kappa(x, x) - 2\kappa(x, z) + \kappa(z, z)}$
  - ✓ $\varphi_{x,z} = \cos^{-1} \frac{\kappa(x,z)}{\sqrt{\kappa(x,x) \cdot \kappa(z,z)}}$

One-Class SVM: in more detail

- There are different kernels (all based on linear kernel):

Kernel (function)	$\boldsymbol{\kappa(x, z)}$
Linear kernel	$\langle x, z \rangle$
Polynomial kernel	$(\langle x, z \rangle + \delta)^a$
RBF (Gaussian) kernel *[RBF = Radial Basis Function]*	$\exp\left(-\frac{\|x - z\|^2}{2\sigma^2}\right)$
Sigmoidal kernel	$\tanh(\langle x, z \rangle + \delta)$

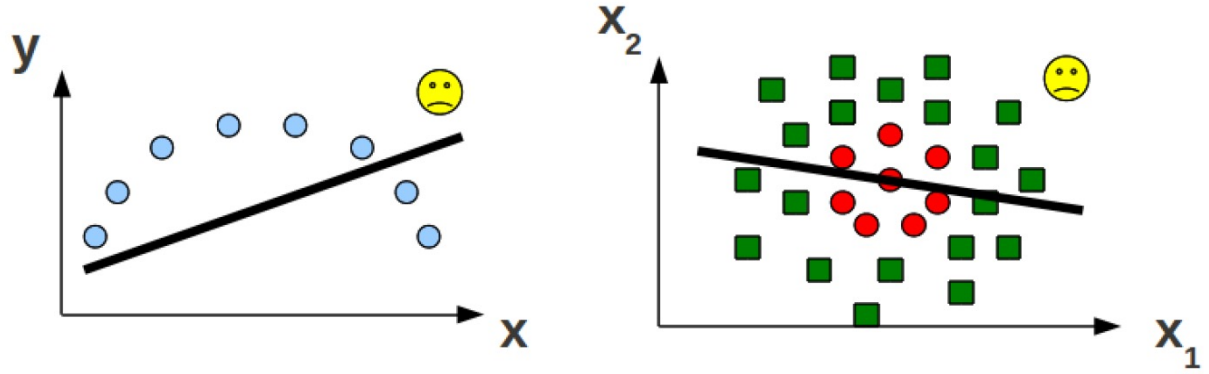
Parameters: a = polynomial degree; δ = offset term; σ = RBF kernel width

Interlude

INTUITION BEHIND KERNELS

Intuition behind Kernels

- Linear models are often not rich enough:



- **Kernels:** Make linear models work in non-linear settings
  - By mapping data to higher dimensions where it exhibits linear patterns
  - Mapping = changing the feature representation

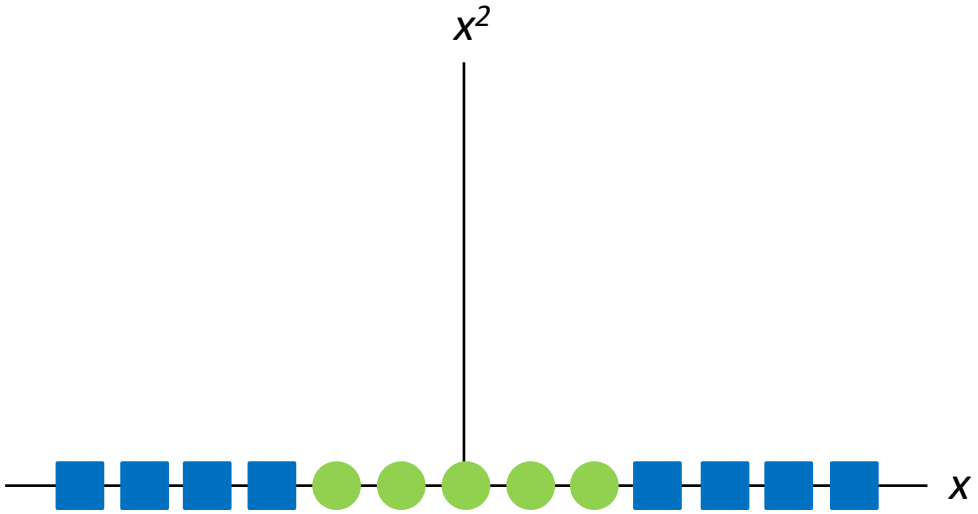
Intuition behind Kernels

- Example:
  - Consider the following binary separation problem:

  - Each sample represented by a *single feature x*
  - No linear separator exists for this data

Intuition behind Kernels

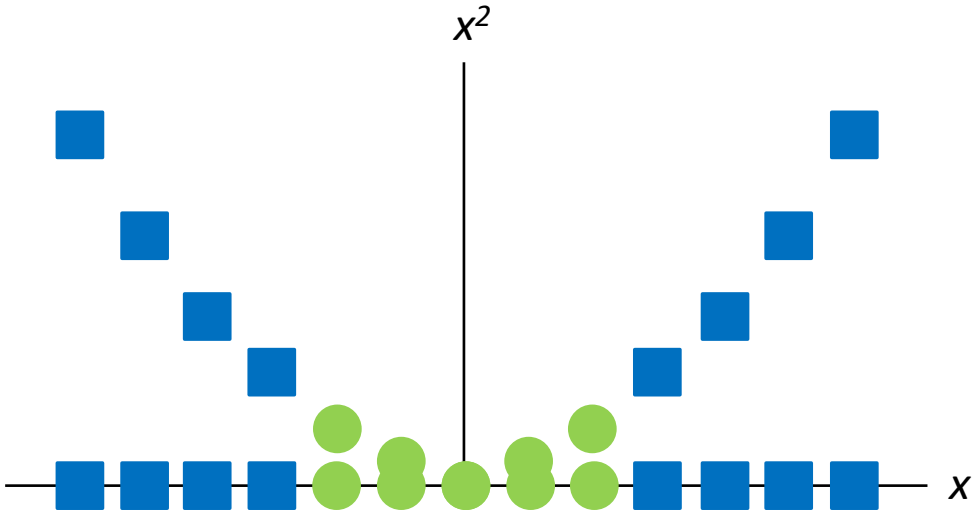
- Example:
  - Consider the following binary separation problem:



  - Each sample represented by a *single feature x*
  - No linear separator exists for this data
  - Map each sample on x^2 (→ two-dimensions)

Intuition behind Kernels

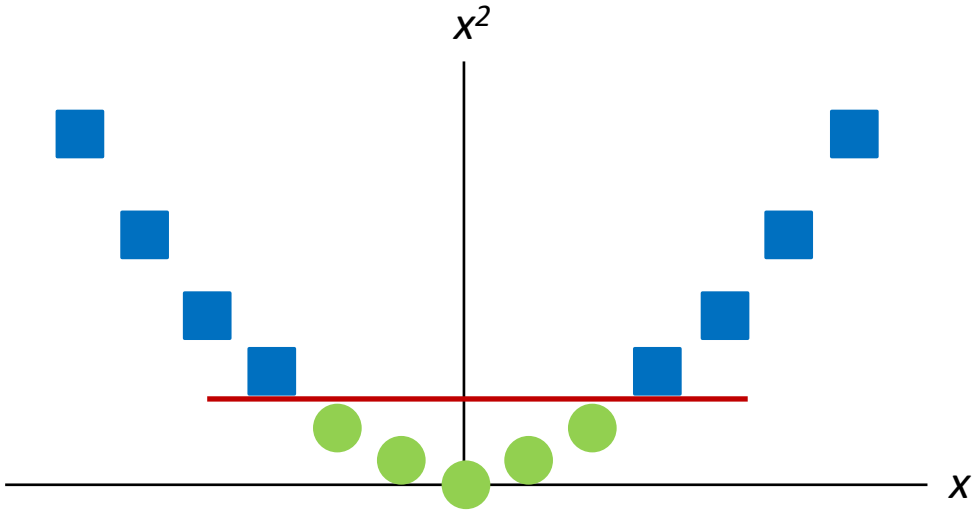
- Example:
  - Consider the following binary separation problem:



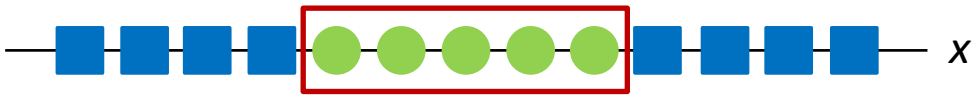
  - Each sample represented by a *single feature x*
  - No linear separator exists for this data
  - Map each sample on x^2 (→ two-dimensions)

Intuition behind Kernels

- Example:
  - Consider the following binary separation problem:



  - Linear in the new representation // non-linear in the old one:



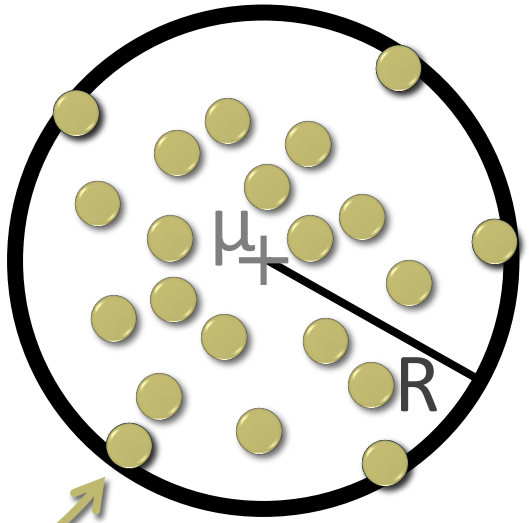
Back to One-Class SVM

One-Class SVM: in more detail

Hypersphere (linear kernel)

Optimization problem:

$$\mu_+ = \underset{\mu \in \mathbb{R}^N}{\operatorname{argmin}} \max_{1 \le i \le n} \|x_i - \mu\|^2$$

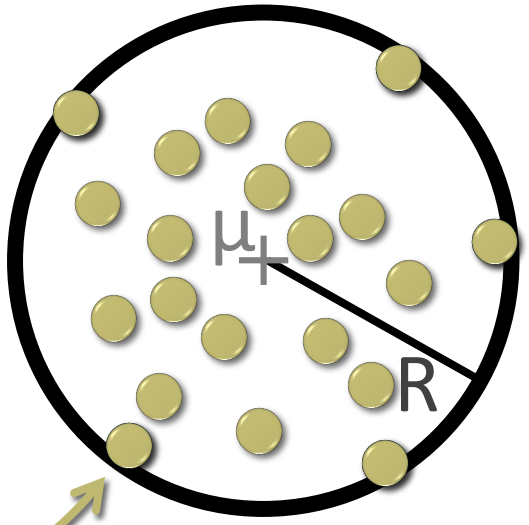


Feature vectors

$X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^N$

One-Class SVM: in more detail

Hypersphere (linear kernel)



Feature vectors

$X = \{x_1, \ldots, x_n\} \subset \mathbb{R}^N$

Optimization problem:

$$\mu_+ = \underset{\mu \in \mathbb{R}^N}{\operatorname{argmin}} \max_{1 \le i \le n} \|x_i - \mu\|^2$$

Solution: Method of "Lagrangian multipliers"

$$\mu_+ = \sum_{i=1}^{n} \alpha_i x_i$$

Lagrangian multiplier

S. Boyd and L. Vandenberghe (2004). Convex Optimization. CAM Press

One-Class SVM

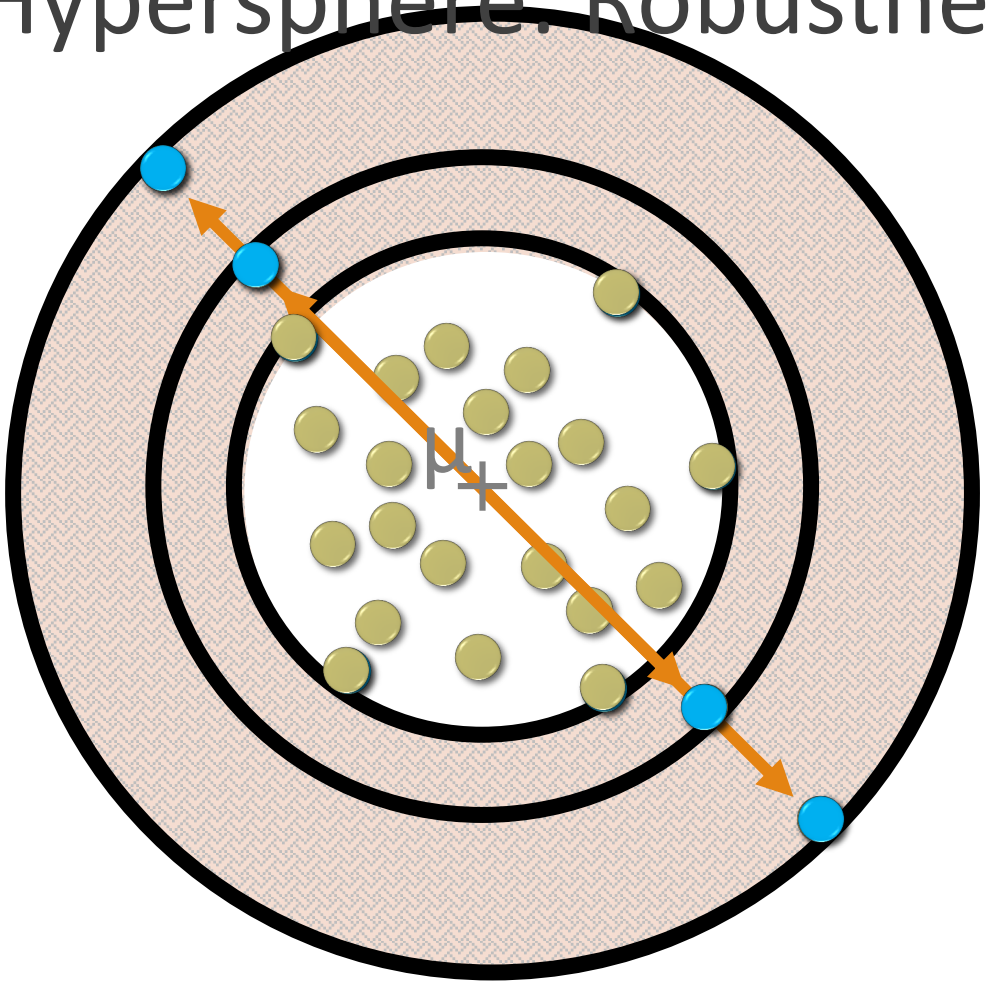
Hypersphere

- Encloses data
- Minimum volume R^2 desired: $\min_{\mu,R} R^2$
- Center = linear combination of training data

- Robustness?

One-Class SVM: linear kernel

Hypersphere: Robustness



Optimization problem:

$$\mu_+ = \underset{\mu \in \mathbb{R}^N}{\operatorname{argmin}} \max_{1 \le i \le n} \|x_i - \mu\|^2$$

One-Class SVM

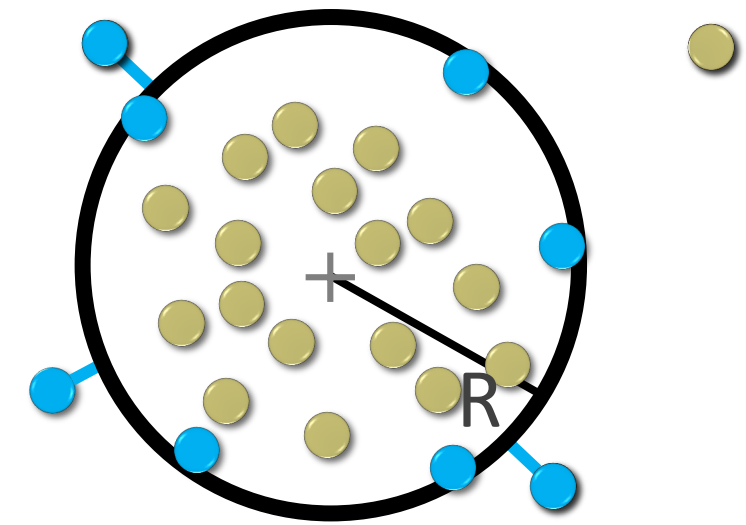
Hypersphere: Robustness

- Prone to outliers
- Ineffective on noisy data
- Inclusion of attacks in learning model possible

- Remedy: Regularization ("Softening")

One-Class SVM: linear kernel

Hypersphere: Regularization (Softening)



- Allow for some *local* "slack" distance ξ_i of each feature vector x_i to surface
- Introduce *global* "softening" parameter $v \in [0,1]$
- Hard surface ($v = 0$), soft surface ($v > 0$)

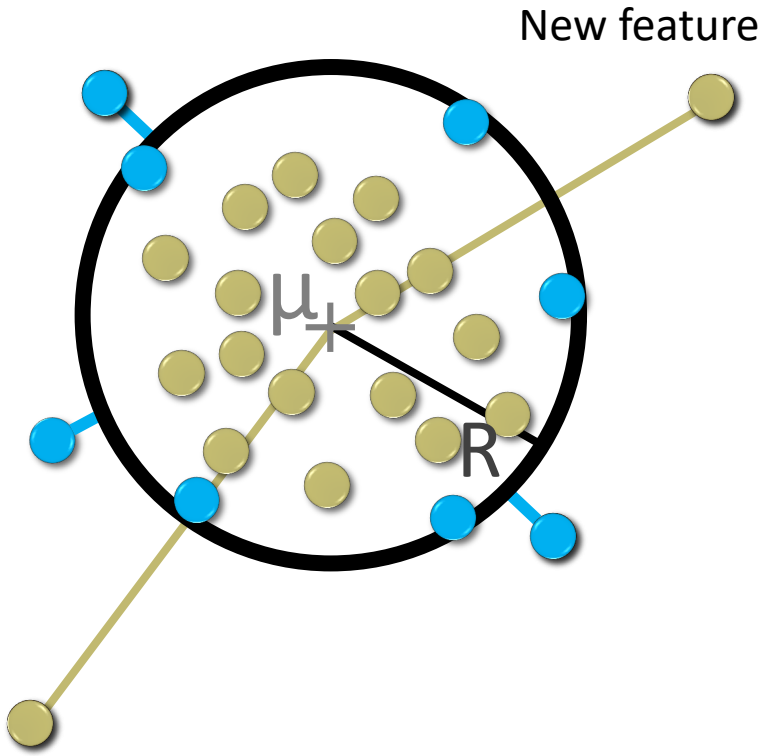
Constrained optimization problem:

$$\min_{R,\mu,\xi} R^2 + \frac{1}{vn}\sum_{i=1}^{n} \xi_i$$

subject to $\|x_i - \mu\|^2 \leq R^2 + \xi_i$ for all $i = 1, \ldots n$.

One-Class SVM: linear kernel

Hypersphere: Model and Prediction



- **Learning model** $\theta = (\mu_+)$
- A **prediction function** f_θ for assessing the deviation of a new feature vector *z* from the learned center μ_+ is:

$$f_\theta(z) = \|z - \mu_+\|^2$$

- Set a **threshold** τ. Predict *z* as "anomalous" if $f_\theta(z) \geq \tau$; otherwise predict as "normal".

One-Class SVM: other kernels

Optimization Problem *Let* $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ *be a training set,* κ *a kernel and* $\nu \in [0,1]$ *a regularization parameter. Then the one-class SVM is determined as follows*

$$\max_{\alpha} \quad \sum_{i=1}^{n} \alpha_i \kappa(\mathbf{x}_i, \mathbf{x}_i) - \sum_{i,j=1}^{n} \alpha_i \alpha_j \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

$$\text{subject to} \quad \sum_{i=1}^{n} \alpha_i = 1 \text{ and } 0 \leq \alpha_i \leq \frac{1}{\nu n} \text{ for all } i = 1, \ldots, n.$$

- This optimization problem returns the center μ_+ of the optimal hypersphere indirectly by determining the coefficients α.
- Solution via Lagrangian method

One-Class SVM: other kernels

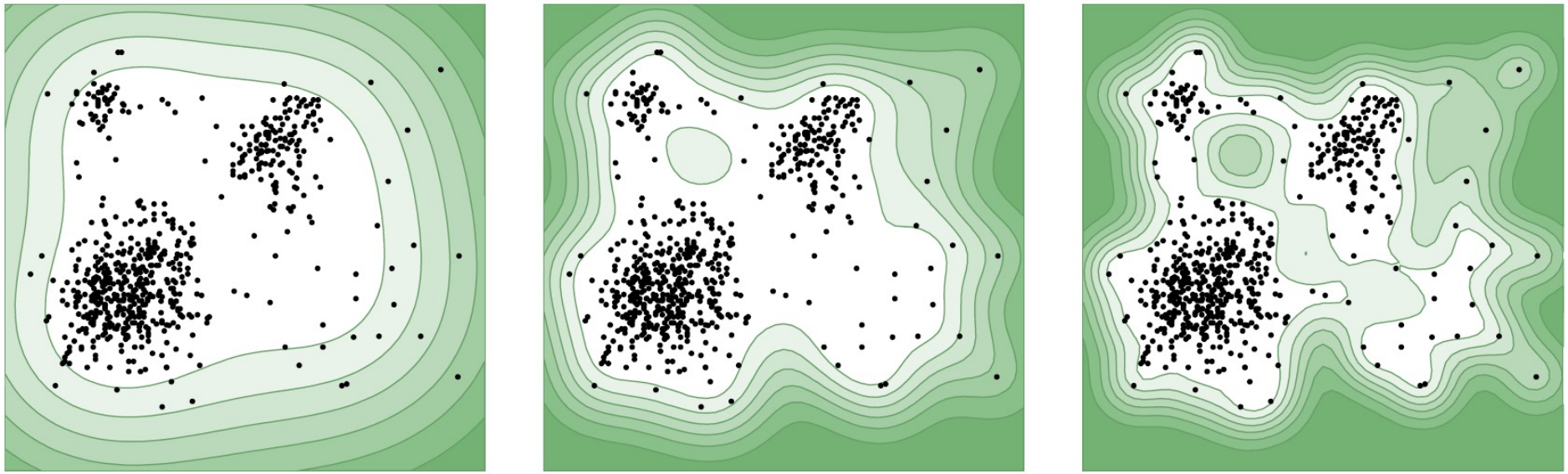
Learning Model and Prediction

- Training set $X = \{x_1, \ldots, x_n\}$ and kernel κ
- **Learning model** $\theta = (X, \alpha)$ (see optimization problem on previous slide)
- A **prediction function** f_θ for assessing the deviation of a new feature vector *z* from the learned model $\theta = (X, \alpha)$ is:

$$f_\theta(\mathbf{z}) = \kappa(\mathbf{z}, \mathbf{z}) - 2 \sum_{\substack{i=1 \\ \alpha_i > 0}}^{n} \alpha_i \kappa(\mathbf{z}, \mathbf{x}_i) + \sum_{\substack{i,j=1 \\ \alpha_i, \alpha_j > 0}}^{n} \alpha_i \alpha_j \kappa(\mathbf{x}_i, \mathbf{x}_j)$$

One-Class SVM: other kernels

- Using different kernels can provide better detection results
- Example with RBF kernel with varying kernel widths σ:



(a) $\sigma = 0.1$ (b) $\sigma = 0.01$ (c) $\sigma = 0.005$

Light shading indicates normal and dark shading anomalous regions.

One-Class SVM: Possible Outcomes

- Assume that we trained a 1-class SVM model θ
- Apply prediction function f_θ to some test data (i.e., we know which samples are “anomalous” vs. “normal”)
- There are 4 possible outcomes:

	Is actually “anomalous”	**Is actually “normal”**
Predicted “anomalous”	True Positive (TP)	False Positive (FP)
Predicted “normal”	False Negative (FN)	True Negative (TN)

Performance Metrics

- **True Positive Rate (TPR)**: proportion of actual positives that are correctly identified as such

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}$$

- **True Negative Rate (TNR)**: proportion of actual negatives that are correctly identified as such

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}}$$

- **False Positive Rate (FPR)**: proportion of actual negatives that are identified as positives

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{TN} + \mathrm{FP}}$$

If high, we flag many "normal" samples as "anomalous"

- **False Negative Rate (FNR)**: proportion of actual positives that are identified as negatives

$$\mathrm{FNR} = \frac{\mathrm{FN}}{\mathrm{TP} + \mathrm{FN}}$$

If high, we flag many "anomalous" samples as "normal"

- **Accuracy (ACC)**: proportion of true results (both TPs and TNs) among all cases examined

If = 0.0, nothing is classified correctly

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}$$

If = 1.0, everything is classified correctly

Study Material

Key reference:

K. Rieck, *Machine Learning for Application-Layer Intrusion Detection*. PhD thesis, TU Berlin, 2009. Available at: https://user.informatik.uni-goettingen.de/~krieck/docs/2009-diss.pdf

Sections	Content
1.1 – 1.2	Introduction
2.1 – 2.3	Network Layers and Numerical Features
3.1	Geometry in Feature Spaces and Kernels
4.1 – 4.2	Machine Learning for Network Intrusion Detection Systems
4.4.2	Calibration

Further reference:

R. Sommer and V. Paxson, *Outside the Closed World: On Using Machine Learning For Network Intrusion Detection*. S&P, IEEE, 2010. Available at: https://personal.utdallas.edu/~muratk/courses/dmsec_files/oakland10-ml.pdf

Assignment: [AI] Security

- Build your own NIDS (hands on!)
- Assignment guides you through the steps
- Data sets provided for training and testing
- Pass-criterion: Accuracy ≥ 0.9; FPR ≤ 0.05

Assignment: [AI] Security

- Available on Canvas under “[AI & CS] Practical assignments”
- Pass-Fail assignment
  - If Pass, your “score” will be listed on our “leaderboard”
  - Upload solution on canvas
- **Deadline: 11 January 2022, 23:59**
- Do not cheat!
- Discord channel (#qa-week6)