Machine Learning and Pervasive Computing

Stephan Sigg

Georg-August-University Goettingen, Computer Networks

04.02.2015

Overview and Structure

22.10.2014	Organisation
22.10.3014	Introduction (Def.: Machine learning, Supervised/Unsupervised, Examples)
29.10.2014	Machine Learning Basics (Toolchain, Features, Metrics, Rule-based)
05.11.2014	A simple Supervised learning algorithm
12.11.2014	Excursion: Avoiding local optima with random search
19.11.2014	–
26.11.2014	Bayesian learner
03.12.2014	–
10.12.2014	Decision tree learner
17.12.2014	k-nearest neighbour
07.01.2015	Support Vector Machines
14.01.2015	Artificial Neural Networks and Self Organizing Maps
21.01.2015	Hidden Markov models and Conditional random fields
28.01.2015	High dimensional data, Unsupervised learning
04.02.2015	Anomaly detection, Online learning, Recom. systems

Outline

Anomaly detection

Recommender systems

Stochastic Classification

Online learning

Anomaly detection

Problem statement

Anomaly detection

Anomaly detection is the task to identify anomalous behaviour with respect to behaviour trained on a data set of prior samples

Anomaly detection

Problem statement

Anomaly detection

Anomaly detection is the task to identify anomalous behaviour with respect to behaviour trained on a data set of prior samples

Examples: **Fraud detection** Identify users with unusual behaviour (e.g. non-human profiles in a web-application)

Anomaly detection

Problem statement

Anomaly detection

Anomaly detection is the task to identify anomalous behaviour with respect to behaviour trained on a data set of prior samples

Examples:

- **Fraud detection**: Identify users with unusual behaviour (e.g. non-human profiles in a web-application)
- **Manufacturing**: Identify faulty engines (e.g. too much vibration, too loud, ...)

Anomaly detection

Problem statement

Anomaly detection

Anomaly detection is the task to identify anomalous behaviour with respect to behaviour trained on a data set of prior samples

Examples:

- Fraud detection: Identify users with unusual behaviour (e.g. non-human profiles in a web-application)
- Manufacturing: Identify faulty engines (e.g. too much vibration, too loud, ...)
- Computers in a datacenter: Identify machines that do not work properly (e.g. processing load over network traffic)

Anomaly detection

Problem statement

Anomaly detection

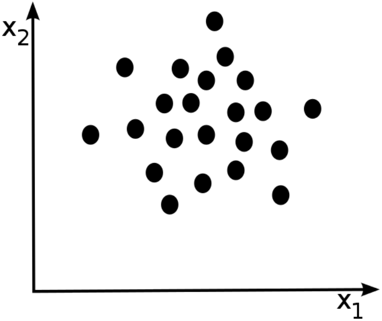
Anomaly detection is the task to identify anomalous behaviour with respect to behaviour trained on a data set of prior samples

Examples:

- **Fraud detection** Identify users with unusual behaviour (e.g. non-human profiles in a web-application)
- **Manufacturing** Identify faulty engines (e.g. too much vibration, too loud, ...)
- **Computers in a datacenter** Identify machines that do not work properly (e.g. processing load over network traffic)
- **Network traffic** Identify unusual traffic patterns (e.g. traffic amount or origin)

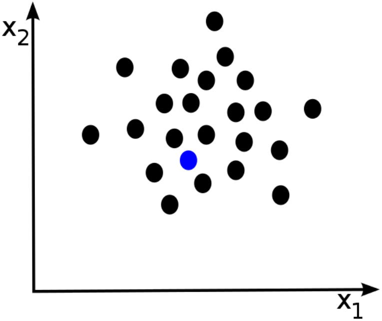
Anomaly detection

Problem statement



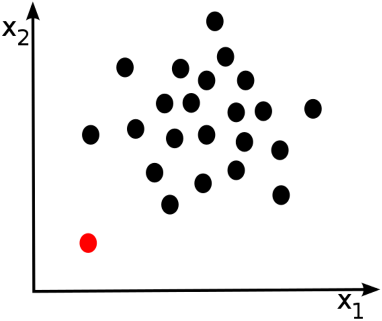
Anomaly detection

Problem statement



Anomaly detection

Problem statement



Anomaly detection

Anomaly detection algorithm

1. Choose features that are indicative of anomalous examples

Anomaly detection

Anomaly detection algorithm

1. Choose features that are indicative of anomalous examples
2. Conditioning on a Gaussian distribution, fit the parameters

$$\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_j^{(i)} \text{(mean of a Gaussian distribution)}$$

$$\sigma_j^2 = \frac{1}{n}\sum_{i=1}^{n} \left(x_j^{(i)} - \mu_j\right)^2 \text{(standard deviation of a Gaussian distribution)}$$

Anomaly detection

Anomaly detection algorithm

1. Choose features that are indicative of anomalous examples
2. Conditioning on a Gaussian distribution, fit the parameters

$$\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_j^{(i)} \text{ (mean of a Gaussian distribution)}$$

$$\sigma_j^2 = \frac{1}{n}\sum_{i=1}^{n} \left(x_j^{(i)} - \mu_j\right)^2 \text{ (standard deviation of a Gaussian distribution)}$$

3. For a new sample x, compute

$$\mathcal{P}[x] = \prod_{j=1}^{m} P[x_j; \mu_j, \sigma_j^2] = \prod_{j=1}^{m} \underbrace{\frac{1}{\sqrt{2\pi}\sigma_j} e^{-\frac{(x_j-\mu_j)^2}{2\sigma_j^2}}}_{\text{Gaussian distribution}}$$

Anomaly detection

Anomaly detection algorithm

1. Choose features that are indicative of anomalous examples
2. Conditioning on a Gaussian distribution, fit the parameters

$$\mu_j = \frac{1}{n}\sum_{i=1}^{n} x_j^{(i)} \text{(mean of a Gaussian distribution)}$$

$$\sigma_j^2 = \frac{1}{n}\sum_{i=1}^{n} \left(x_j^{(i)} - \mu_j\right)^2 \text{(standard deviation of a Gaussian distribution)}$$

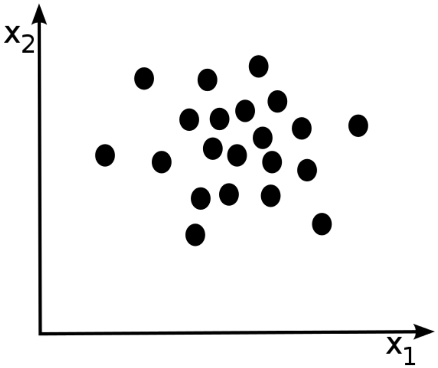
3. For a new sample x, compute

$$\mathcal{P}[x] = \prod_{j=1}^{m} P[x_j; \mu_j, \sigma_j^2] = \prod_{j=1}^{m} \underbrace{\frac{1}{\sqrt{2\pi}\sigma_j} e^{-\frac{(x_j-\mu_j)^2}{2\sigma_j^2}}}_{\text{Gaussian distribution}}$$

4. anomaly if $\mathcal{P}[x] < \varepsilon$

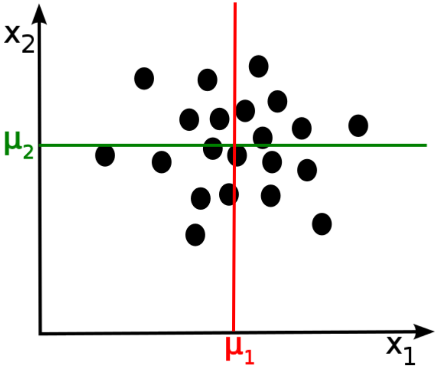
Anomaly detection

Example



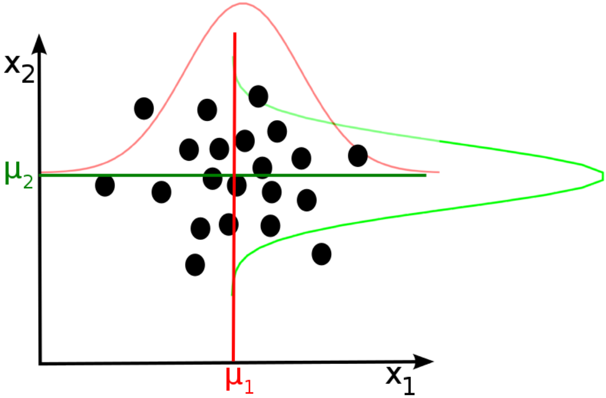
Anomaly detection

Example



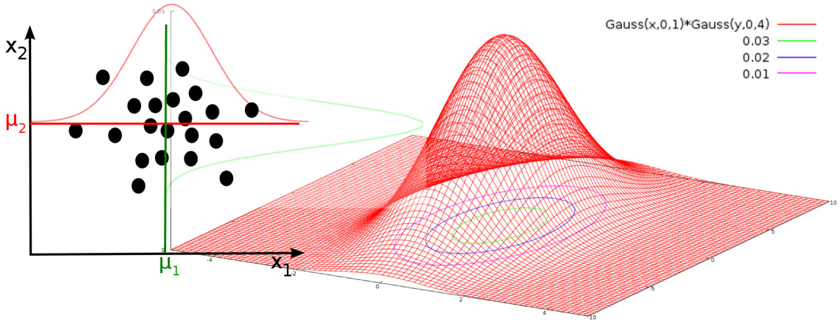
Anomaly detection

Example



Anomaly detection

Example



Anomaly detection

Problem statement

Choice of good values for ε

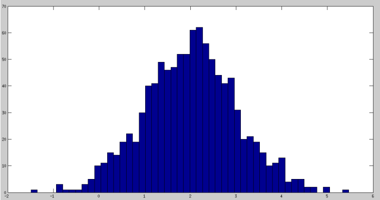
Using crossvalidation and testing sets, calculate

- Precision/Recall
- F_1-score
- ...

Anomaly detection

Non-Gaussian features

In anomaly detection, we have so far assumed Gaussian distributed features.

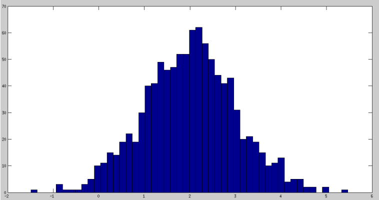


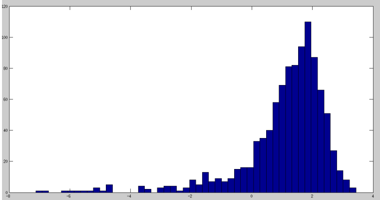
Anomaly detection

Non-Gaussian features

In anomaly detection, we have so far assumed Gaussian distributed features.

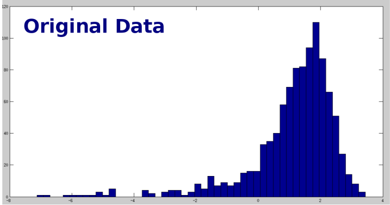
→ What if the feature distribution is not Gaussian ?

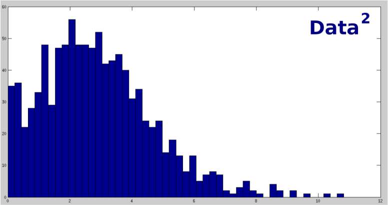


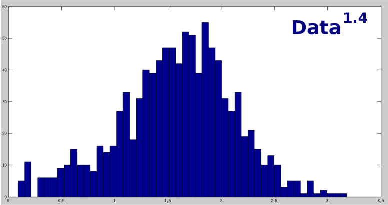


Anomaly detection

Generate new features with a more Gaussian-like distribution





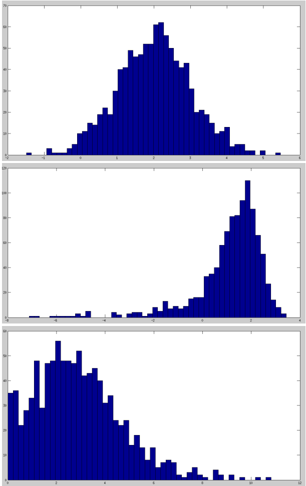


Anomaly detection

Non-Gaussian features

Possible operations on features

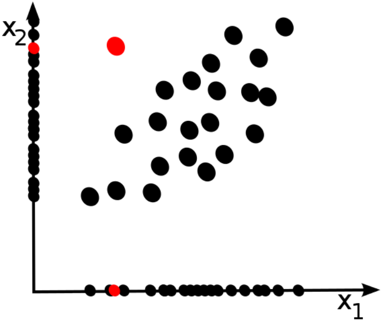
$$x_{\text{new}} = \log(x)$$
$$x_{\text{new}} = \sqrt{x}$$
$$x_{\text{new}} = x^{\frac{1}{3}}$$
$$x_{\text{new}} = \log(x + k)$$
$$\vdots$$



Anomaly detection

Multivariate Gaussian Distribution

- Note that there are cases in which the anomaly looks perfectly normal when considering each dimension separately

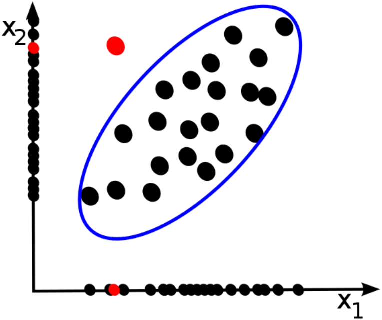


Anomaly detection

Multivariate Gaussian Distribution

- Note that there are cases in which the anomaly looks perfectly normal when considering each dimension separately

→ The consideration of multivariate Gaussian distributions might be suggestive in order to detect such anomalies.



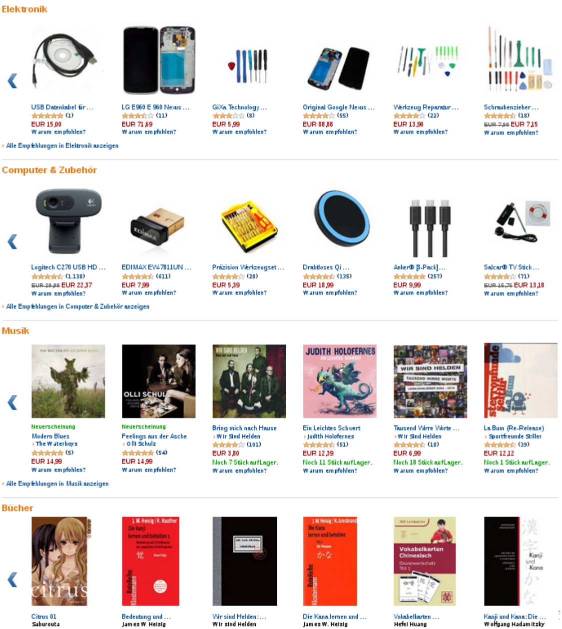
Outline

Anomaly detection

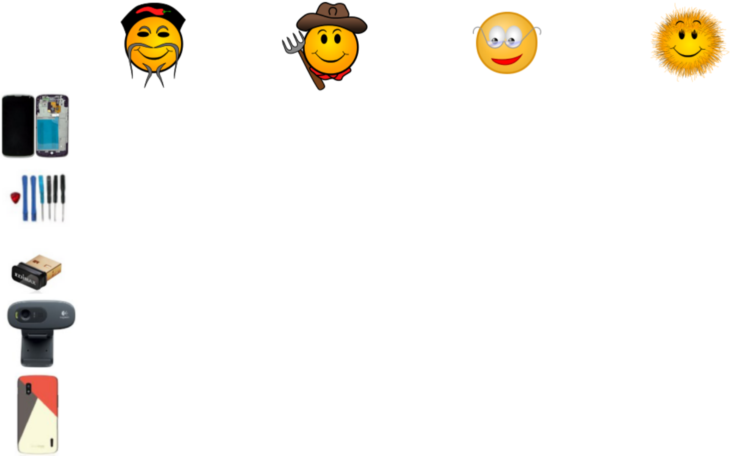
Recommender systems

Stochastic Classification

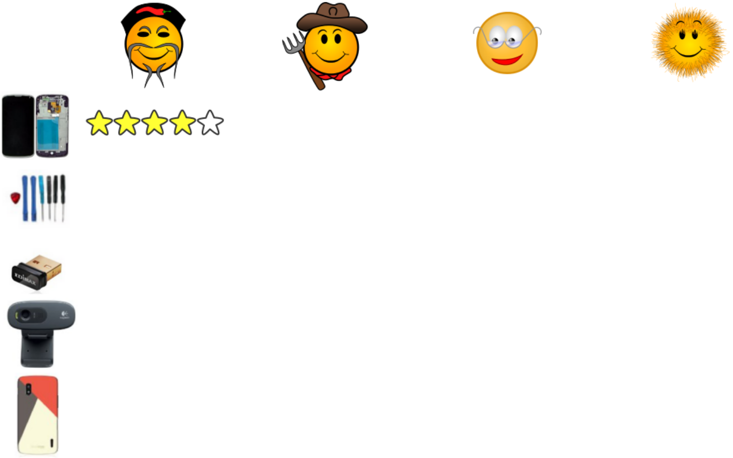
Online learning


Elektronik
USB Datenkabel für ...
EUR 15,00
LG E960 E 960 Nexus ...
EUR 71,69
GiXa Technology ...
EUR 5,99
Original Google Nexus ...
EUR 88,88
Werkzeug Reparatur ...
EUR 13,90
Schraubenzieher ...
EUR 7,99 EUR 7,15
Warum empfohlen?
Alle Empfehlungen in Elektronik anzeigen
Computer & Zubehör
Logitech C270 USB HD ...
EUR 29,99 EUR 22,37
EDIMAX EW-7811UN ...
EUR 7,99
Präzision Werkzeugset ...
EUR 5,39
Drahtloses Qi ...
EUR 18,99
Anker® [3-Pack] ...
EUR 9,99
Salcar® TV Stick ...
EUR 15,76 EUR 13,18
Alle Empfehlungen in Computer & Zubehör anzeigen
Musik
Neuerscheinung
Modern Blues
The Waterboys
EUR 14,99
Neuerscheinung
Feelings aus der Asche
Olli Schulz
EUR 14,99
Bring mich nach Hause
Wir Sind Helden
EUR 3,80
Noch 7 Stück auf Lager.
Ein Leichtes Schwert
Judith Holofernes
EUR 12,39
Noch 11 Stück auf Lager.
Tausend Wirre Worte ...
Wir Sind Helden
EUR 6,99
Noch 18 Stück auf Lager.
La Bum (Re-Release)
Sportfreunde Stiller
EUR 12,12
Noch 1 Stück auf Lager.
Alle Empfehlungen in Musik anzeigen
Bücher
Citrus 01
Saburouta
Bedeutung und ...
James W. Heisig
Wir sind Helden: ...
Wir sind Helden
Die Kana lernen und ...
James W. Heisig
Vokabelkarten ...
Hefei Huang
Kanji und Kana: Die ...
Wolfgang Hadamitzky

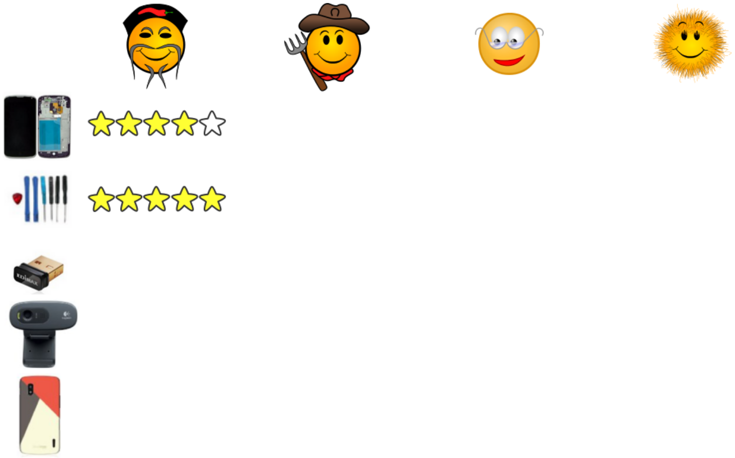
Recommender systems



Recommender systems



Recommender systems

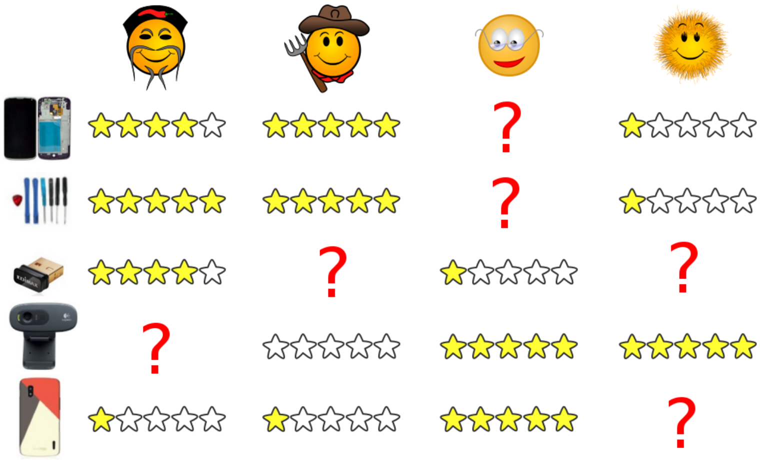


Recommender systems

Recommender systems

Recommender systems

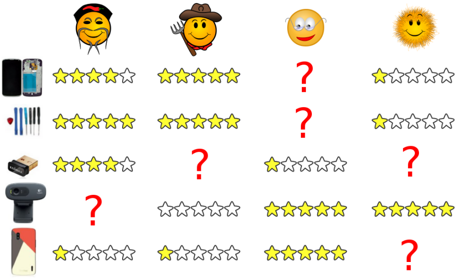
Recommender systems



Recommender systems

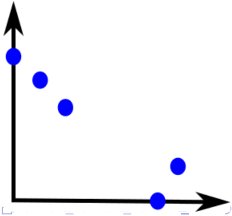
Task/Problem of Recommender systems

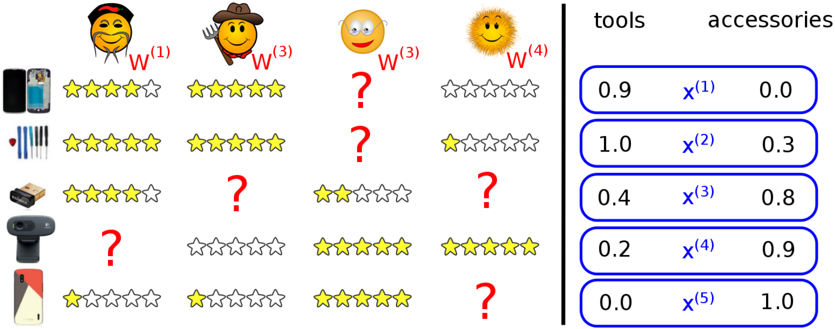
Given these ratings for a number of products, predict how the user-ratings for products that have not yet been rated

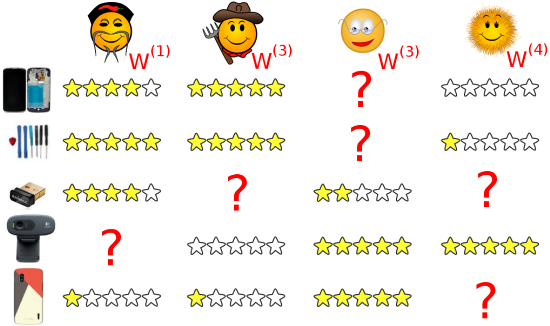


					tools	accessories
	★★★★☆	★★★★★	?	☆☆☆☆☆	0.9	0.0
	★★★★★	★★★★★	?	★☆☆☆☆	1.0	0.3
	★★★★☆	?	★★☆☆☆	?	0.4	0.8
	?	☆☆☆☆☆	★★★★★	★★★★★	0.2	0.9
	★☆☆☆☆	★☆☆☆☆	★★★★★	?	0.0	1.0

					tools	accessories
	★★★★☆	★★★★★	?	☆☆☆☆☆	0.9	0.0
	★★★★★	★★★★★	?	★☆☆☆☆	1.0	0.3
	★★★★☆	?	★★☆☆☆	?	0.4	0.8
	?	☆☆☆☆☆	★★★★★	★★★★★	0.2	0.9
	★☆☆☆☆	★☆☆☆☆	★★★★★	?	0.0	1.0




$w^{(1)}$
$w^{(3)}$
$w^{(3)}$
$w^{(4)}$
?
?
?
?
?
?
tools
accessories
0.9 $x^{(1)}$ 0.0
1.0 $x^{(2)}$ 0.3
0.4 $x^{(3)}$ 0.8
0.2 $x^{(4)}$ 0.9
0.0 $x^{(5)}$ 1.0

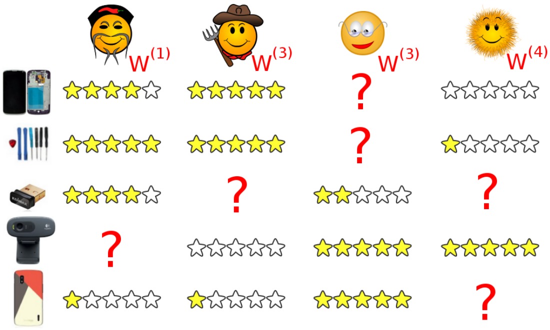


	tools	accessories
$x^{(1)}$	0.9	0.0
$x^{(2)}$	1.0	0.3
$x^{(3)}$	0.4	0.8
$x^{(4)}$	0.2	0.9
$x^{(5)}$	0.0	1.0

Represent items as feature vectors

$$x^{(4)} = \begin{bmatrix} 0.2 \\ 0.9 \end{bmatrix} \quad W^{(1)} = \begin{bmatrix} 5 \\ 1 \end{bmatrix}$$

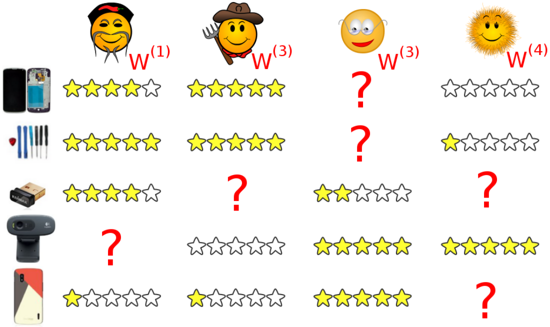
$$\Rightarrow \left(W^{(1)}\right)^T x^{(4)} = 0.2 \cdot 5 + 0.9 \cdot 1 = 1,9$$



tools		accessories
0.9	$\mathbf{x}^{(1)}$	0.0
1.0	$\mathbf{x}^{(2)}$	0.3
0.4	$\mathbf{x}^{(3)}$	0.8
0.2	$\mathbf{x}^{(4)}$	0.9
0.0	$\mathbf{x}^{(5)}$	1.0

Learn weights from provided ratings for single user j (Linear regression):

$$\min_{W^{(j)}} \frac{1}{2} \sum_{i:y^{(i,j)} \neq ?} \left(\left(W^{(j)} \right)^T x^{(i)} - y^{(i,j)} \right) + \frac{\lambda}{2} \sum_{k=1}^{F} \left(W_k^{(j)} \right)^2$$



	tools		accessories
	0.9	$\mathbf{x}^{(1)}$	0.0
	1.0	$\mathbf{x}^{(2)}$	0.3
	0.4	$\mathbf{x}^{(3)}$	0.8
	0.2	$\mathbf{x}^{(4)}$	0.9
	0.0	$\mathbf{x}^{(5)}$	1.0

Learn weights from provided ratings for single all users $1, \ldots, N$:

$$\min_{W^{(1)},\ldots,W^{(N)}} \frac{1}{2}\sum_{j=1}^{N}\sum_{i:y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right) + \frac{\lambda}{2}\sum_{j=1}^{N}\sum_{k=1}^{F}\left(W_k^{(j)}\right)^2$$

Recommender systems

Optimisation algorithm

$$\min_{W^{(1)},\ldots,W^{(N)}} \frac{1}{2}\sum_{j=1}^{N}\sum_{i:y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^{T}x^{(i)}-y^{(i,j)}\right)+\frac{\lambda}{2}\sum_{j=1}^{N}\sum_{k=1}^{F}\left(W_{k}^{(j)}\right)^{2}$$

Recommender systems

Optimisation algorithm

$$\min_{W^{(1)},\ldots,W^{(N)}} \frac{1}{2}\sum_{j=1}^{N}\sum_{i:y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^{T}x^{(i)} - y^{(i,j)}\right) + \frac{\lambda}{2}\sum_{j=1}^{N}\sum_{k=1}^{F}\left(W_k^{(j)}\right)^2$$

Gradient descent update:

$$W_k^{(j)} = W_k^{(j)} - \alpha\left(\sum_{i=y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^{T}x^{(i)} - y^{(i,j)}\right)x_k^{(i)} + \lambda W_k^{(j)}\right)$$

Recommender systems

Optimisation algorithm

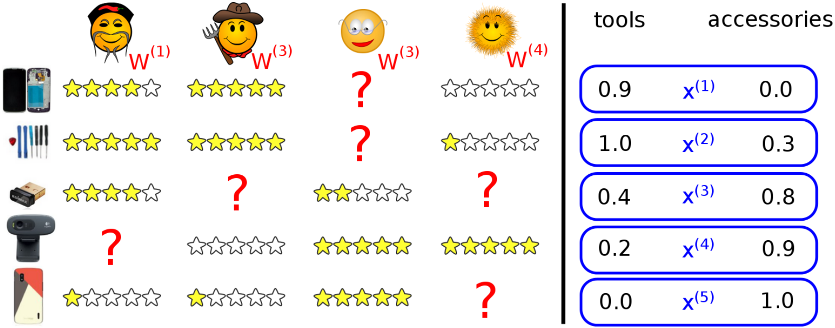
$$\min_{W^{(1)},\ldots,W^{(N)}} \frac{1}{2} \sum_{j=1}^{N} \sum_{i:y^{(i,j)} \neq ?} \left(\left(W^{(j)} \right)^T x^{(i)} - y^{(i,j)} \right) + \frac{\lambda}{2} \sum_{j=1}^{N} \sum_{k=1}^{F} \left(W_k^{(j)} \right)^2$$

Gradient descent update:

$$W_k^{(j)} = W_k^{(j)} - \alpha \left(\underbrace{\sum_{i=y^{(i,j)} \neq ?} \left(\left(W^{(j)} \right)^T x^{(i)} - y^{(i,j)} \right) x_k^{(i)} + \lambda W_k^{(j)}}_{\text{partial derivative}} \right)$$

Recommender systems

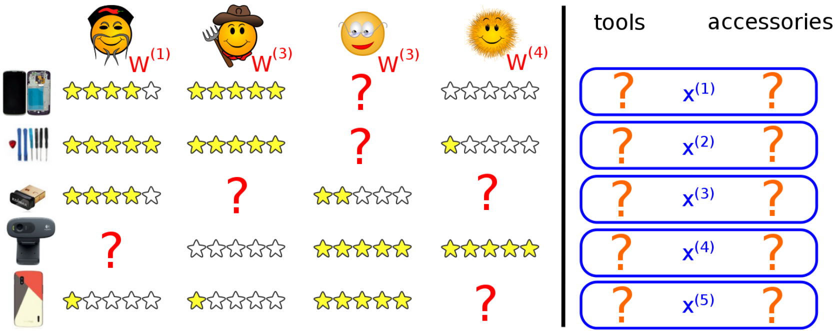
Collaborative filtering



We are able to calculate the weights given the feature vectors

Recommender systems

Collaborative filtering



We are able to calculate the weights given the feature vectors

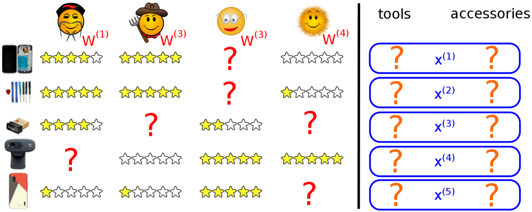
→ But how do we obtain these feature vectors?

Recommender systems

Collaborative filtering

Users might tell their preferences

e.g. more interested in tools or accessories



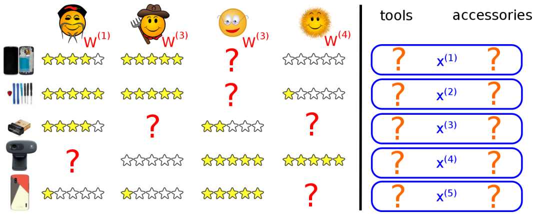
$$W^{(1)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix} \quad W^{(2)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix} \quad W^{(3)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix} \quad W^{(4)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

Recommender systems

Collaborative filtering

Users might tell their preferences

e.g. more interested in tools or accessories



$$W^{(1)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix} \quad W^{(2)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix} \quad W^{(3)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix} \quad W^{(4)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

$$\left(W^{(1)}\right)^T x^{(1)} \approx 4; \; \left(W^{(2)}\right)^T x^{(1)} \approx 5;$$

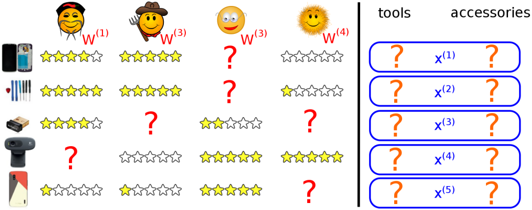
$$\left(W^{(3)}\right)^T x^{(1)} \approx ?; \; \left(W^{(4)}\right)^T x^{(1)} \approx 0$$

Recommender systems

Collaborative filtering

Users might tell their preferences

e.g. more interested in tools or accessories



$$W^{(1)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix} \quad W^{(2)} = \begin{bmatrix} 5 \\ 0 \end{bmatrix} \quad W^{(3)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix} \quad W^{(4)} = \begin{bmatrix} 0 \\ 5 \end{bmatrix}$$

$$\left(W^{(1)}\right)^T x^{(1)} \approx 4; \quad \left(W^{(2)}\right)^T x^{(1)} \approx 5;$$

$$\left(W^{(3)}\right)^T x^{(1)} \approx ?; \quad \left(W^{(4)}\right)^T x^{(1)} \approx 0$$

From these weights we can estimate the feature values

Recommender systems

Collaborative filtering

Optimisation algorithm

Given the weights/preferences $W^{(1)}, \ldots, W^{(N)}$, we are able to infer a feature $x^{(i)}$

$$\min_{x^{(i)}} \frac{1}{2} \sum_{j:y^{(i,j)} \neq ?} \left(\left(W^{(j)} \right)^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \sum_{k=1}^{F} \left(x_k^{(i)} \right)^2$$

Recommender systems

Collaborative filtering

Optimisation algorithm

Given the weights/preferences $W^{(1)}, \ldots, W^{(N)}$, we are able to infer a feature $x^{(i)}$

$$\min_{x^{(i)}} \frac{1}{2} \sum_{j: y^{(i,j)} \neq ?} \left(\left(W^{(j)} \right)^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \sum_{k=1}^{F} \left(x_k^{(j)} \right)^2$$

Given the weights/preferences $W^{(1)}, \ldots, W^{(N)}$, we are able to infer $x^{(1)}, \ldots, x^{(n)}$

$$\min_{x^{(1)}, \ldots, x^{(n)}} \frac{1}{2} \sum_{i=1}^{n} \sum_{j: y^{(i,j)} \neq ?} \left(\left(W^{(j)} \right)^T x^{(i)} - y^{(i,j)} \right)^2 + \frac{\lambda}{2} \sum_{i=1}^{n} \sum_{k=1}^{F} \left(x_k^{(i)} \right)^2$$

Recommender systems

Collaborative filtering

Naive (iterative) Collaborative filtering algorithm

Given $x^{(1)}, \dots, x^{(n)}$, we are able to estimate $W^{(1)}, \dots, W^{(N)}$

Given $W^{(1)}, \dots, W^{(N)}$, we are able to estimate $x^{(1)}, \dots, x^{(n)}$

Recommender systems

Collaborative filtering

Naive (iterative) Collaborative filtering algorithm

Given $x^{(1)}, \dots, x^{(n)}$, we are able to estimate $W^{(1)}, \dots, W^{(N)}$

Given $W^{(1)}, \dots, W^{(N)}$, we are able to estimate $x^{(1)}, \dots, x^{(n)}$

Collaborative filtering (naive)

Init: Randomly initialise the $W^{(i)}$

Repeat:
- Estimate the $x^{(i)}$ from the $W^{(i)}$
- Estimate the $W^{(i)}$ from the $x^{(i)}$

Recommender systems

Collaborative filtering

Naive (iterative) Collaborative filtering algorithm

Given $x^{(1)}, \ldots, x^{(n)}$, we are able to estimate $W^{(1)}, \ldots, W^{(N)}$
Given $W^{(1)}, \ldots, W^{(N)}$, we are able to estimate $x^{(1)}, \ldots, x^{(n)}$

Collaborative filtering (naive)

Init: Randomly initialise the $W^{(i)}$

Repeat:
- Estimate the $x^{(i)}$ from the $W^{(i)}$
- Estimate the $W^{(i)}$ from the $x^{(i)}$

→ CF iteratively improves the estimates for $x^{(i)}$ and $W^{(i)}$

→ Algorithm collaborates with users: by providing some information about their preferences, it computes and improves the features

Recommender systems

Collaborative filtering

$$\min_{W^{(1)},\ldots,W^{(N)}} \quad \frac{1}{2}\sum_{j=1}^{N}\sum_{i:y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^{T}x^{(i)} - y^{(i,j)}\right)^{2} + \frac{\lambda}{2}\sum_{j=1}^{N}\sum_{k=1}^{F}\left(W_k^{(j)}\right)^{2}$$

$$\min_{x^{(1)},\ldots,x^{(n)}} \quad \frac{1}{2}\sum_{i=1}^{n}\sum_{j:y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^{T}x^{(i)} - y^{(i,j)}\right)^{2} + \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{F}\left(x_k^{(i)}\right)^{2}$$

Recommender systems

Collaborative filtering

$$\min_{W^{(1)},\ldots,W^{(N)}} \quad \frac{1}{2}\sum_{j=1}^{N}\sum_{i:y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{j=1}^{N}\sum_{k=1}^{F}\left(W_k^{(j)}\right)^2$$

$$\min_{x^{(1)},\ldots,x^{(n)}} \quad \frac{1}{2}\sum_{i=1}^{n}\sum_{j:y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^T x^{(i)} - y^{(i,j)}\right)^2 + \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{F}\left(x_k^{(i)}\right)^2$$

Recommender systems

Collaborative filtering

$$\min_{W^{(1)},\ldots,W^{(N)}} \quad \frac{1}{2}\sum_{j=1}^{N}\sum_{i:y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^{T}x^{(i)} - y^{(i,j)}\right)^{2} + \frac{\lambda}{2}\sum_{j=1}^{N}\sum_{k=1}^{F}\left(W_k^{(j)}\right)^{2}$$

$$\min_{x^{(1)},\ldots,x^{(n)}} \quad \frac{1}{2}\sum_{i=1}^{n}\sum_{j:y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^{T}x^{(i)} - y^{(i,j)}\right)^{2} + \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{F}\left(x_k^{(i)}\right)^{2}$$

Minimize $W^{(i)}$ and $x^{(i)}$ simultaneously:

$$\min_{x^{(1)},\ldots,x^{(n)},W^{(1)},\ldots,W^{(N)}} \quad \frac{1}{2}\sum_{i,j:y^{(i,j)}\neq ?}\left(\left(W^{(j)}\right)^{T}x^{(i)} - y^{(i,j)}\right)^{2} + \frac{\lambda}{2}\sum_{i=1}^{n}\sum_{k=1}^{F}\left(x_k^{(i)}\right)^{2} + \frac{\lambda}{2}\sum_{j=1}^{N}\sum_{k=1}^{F}\left(W_k^{(j)}\right)^{2}$$

Recommender systems

Collaborative filtering

Collaborative filtering algorithm

Collaborative filtering

Init: Randomly initialise the $W^{(j)}$ and $x^{(i)}$

Optimisation: Simultaneously minimise the above function for $W^{(j)}$ and $x^{(i)}$

Gradient descent:

$$x_k^{(i)} = x_k^{(i)} - \alpha \left(\sum_{j=y^{(i,j)}\neq ?} \left(\left(W^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right) W_k^{(j)} + \lambda x_k^{(i)} \right)$$

$$W_k^{(j)} = W_k^{(j)} - \alpha \left(\sum_{i=y^{(i,j)}\neq ?} \left(\left(W^{(j)}\right)^T x^{(i)} - y^{(i,j)} \right) x_k^{(i)} + \lambda W_k^{(j)} \right)$$

Prediction: For a user i with parameters $W^{(j)}$ and an item with learned featues x, estimate a rating of $\left(W^{(j)}\right)^T x$

Outline

Anomaly detection

Recommender systems

Stochastic Classification

Online learning

Stochastic Classification

Classification algorithm typically become very slow when data size increases

Stochastic Classification

Classification algorithm typically become very slow when data size increases

→ This is because they loop repeatedly over the complete data set until convergence

Stochastic Classification

Classification algorithm typically become very slow when data size increases

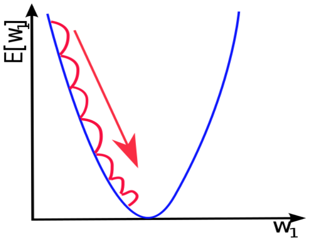
→ This is because they loop repeatedly over the complete data set until convergence

Solution

- Randomly iterate the update only over individual items instead of repeatedly considering the whole data set.

Stochastic Classification

Example: Gradient descent



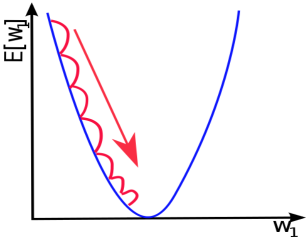
$$\text{minimize } E[W] = \frac{1}{2n}\sum_{i=1}^{n}\left(W^T x^{(i)} - y^{(i)}\right)^2$$

$$\text{Repeat}\forall j : W_j = W_j - \delta \cdot \frac{\partial}{\partial W_j} E[W_j]$$

$$\rightarrow \forall j : W_j = W_j - \delta \cdot \frac{1}{n}\sum_{i=1}^{n}\left(W_j x_j^{(i)} - y^{(i)}\right) \cdot x_j^{(i)}$$

Stochastic Classification

Example: Gradient descent



→ For single gradient descent-step, algorithms loops over all samples!

$$\text{minimize } E[W] = \frac{1}{2n}\sum_{i=1}^{n}\left(W^T x^{(i)} - y^{(i)}\right)^2$$

$$\text{Repeat}\forall j : W_j = W_j - \delta \cdot \frac{\partial}{\partial W_j} E[W_j]$$

$$\rightarrow \forall j : W_j = W_j - \delta \cdot \frac{1}{n}\sum_{i=1}^{n}\left(W_j x_j^{(i)} - y^{(i)}\right) \cdot x_j^{(i)}$$

Stochastic Classification

Example: Gradient descent

$\rightarrow$ For a single gradient descent-step, the algorithms loops over all samples!

$$\begin{aligned} \text{minimize } E[W] &= \frac{1}{2n}\sum_{i=1}^{n}\left(W^T x^{(i)} - y^{(i)}\right)^2 \\ \text{Repeat}\forall j : W_j &= W_j - \delta \cdot \frac{\partial}{\partial W_j} E[W_j] \\ \rightarrow \forall j : W_j &= W_j - \delta \cdot \frac{1}{n}\sum_{i=1}^{n}\left(W_j x_j^{(i)} - y^{(i)}\right) \cdot x_j^{(i)} \end{aligned}$$

Stochastic Classification

Example: Gradient descent

→ For a single gradient descent-step, the algorithms loops over all samples!

To speed up the algorithm, compute gradient descent updates from individual training samples (randomly ordered)

$$\text{minimize } E[W] = \frac{1}{2n}\sum_{i=1}^{n}\left(W^T x^{(i)} - y^{(i)}\right)^2$$

$$\text{Repeat}\forall j : W_j = W_j - \delta \cdot \frac{\partial}{\partial W_j} E[W_j]$$

$$\rightarrow \forall j : W_j = W_j - \delta \cdot \frac{1}{n}\sum_{i=1}^{n}\left(W_j x_j^{(i)} - y^{(i)}\right) \cdot x_j^{(i)}$$

Stochastic Classification

Example: Gradient descent

Standard:

$$
\begin{aligned}
\text{minimize } E[W] &= \frac{1}{2n}\sum_{i=1}^{n}\left(W^T x^{(i)} - y^{(i)}\right)^2 \\
\text{Repeat}\forall j : W_j &= W_j - \delta \cdot \frac{\partial}{\partial W_j} E[W_j] \\
\rightarrow \forall j : W_j &= W_j - \delta \cdot \frac{1}{n}\sum_{i=1}^{n}\left(W_j x_j^{(i)} - y^{(i)}\right) \cdot x_j^{(i)}
\end{aligned}
$$

Stochastic Classification

Example: Gradient descent

Standard:

$$\text{minimize } E[W] = \frac{1}{2n}\sum_{i=1}^{n}\left(W^T x^{(i)} - y^{(i)}\right)^2$$

$$\text{Repeat}\forall j : W_j = W_j - \delta \cdot \frac{\partial}{\partial W_j} E[W_j]$$

$$\rightarrow \forall j : W_j = W_j - \delta \cdot \frac{1}{n}\sum_{i=1}^{n}\left(W_j x_j^{(i)} - y^{(i)}\right) \cdot x_j^{(i)}$$

Stochastic:

Repeat over all training examples i (random order):

$$\Rightarrow \forall j : \quad W_j = W_j - \delta \cdot \left(W_j x_j^{(i)} - y^{(i)}\right) \cdot x_j^{(i)}$$

Stochastic Classification

The stochastic implementation will generally move the parameters towards the global minimum (...but not always!)

Stochastic Classification

The stochastic implementation will generally move the parameters towards the global minimum (...but not always!)

- Greatly speeds up the algorithm as it does not loop over all samples in each single iteration

Stochastic Classification

The stochastic implementation will generally move the parameters towards the global minimum (...but not always!)

Greatly speeds up the algorithm as it does not loop over all samples in each single iteration

Tradeoff use $1 \leq k \leq n$ random examples for each gradient descent update

$l = 1, 1 + k, 1 + 2k, \ldots$

$$\Rightarrow \forall j : W_j = W_j - \delta \cdot \frac{1}{l} \sum_{l=i}^{i+k} \left(W_j x_j^{(i)} - y^{(i)} \right) \cdot x_j^{(i)}$$

Stochastic Classification

The stochastic implementation will generally move the parameters towards the global minimum (...but not always!)

Greatly speeds up the algorithm as it does not loop over all samples in each single iteration

Tradeoff use $1 \leq k \leq n$ random examples for each gradient descent update

$l = 1, 1+k, 1+2k, \ldots$

$$\Rightarrow \forall j : W_j = W_j - \delta \cdot \frac{1}{l} \sum_{l=i}^{i+k} \left(W_j x_j^{(i)} - y^{(i)} \right) \cdot x_j^{(i)}$$

$k > 1$ might be faster than $k = 1$ for parallelized code

Outline

Anomaly detection

Recommender systems

Stochastic Classification

Online learning

Online learning

Problem formulation

Online learning

Given a continuous stream of input data, update the parameters of your algorithm on-the-fly

Example: learn user behaviour from website users

Online learning

Problem formulation

Online learning

Given a continuous stream of input data, update the parameters of your algorithm on-the-fly

Example: learn user behaviour from website users

Similar to stochastic classification: Update the parameters based on individual training examples

$$\Rightarrow \forall j : \quad W_j = W_j - \delta \cdot \left(h(x^{(i)}) - y^{(i)}\right) \cdot x_j^{(i)}$$

Online learning

Problem formulation

Online learning

Given a continuous stream of input data, update the parameters of your algorithm on-the-fly

Example: learn user behaviour from website users

Similar to stochastic classification: Update the parameters based on individual training examples

$$\Rightarrow \forall j: \quad W_j = W_j - \delta \cdot \left(h(x^{(i)}) - y^{(i)}\right) \cdot x_j^{(i)}$$

$\Rightarrow$ Able to adapt to changing user behaviour over time

Outline

Anomaly detection

Recommender systems

Stochastic Classification

Online learning

Questions?

Stephan Sigg
stephan.sigg@cs.uni-goettingen.de

Literature

- C.M. Bishop: Pattern recognition and machine learning, Springer, 2007.
- R.O. Duda, P.E. Hart, D.G. Stork: Pattern Classification, Wiley, 2001.