

Data Mining und Mustererkennung

Einführung

Prof. Dr. Johannes Maucher

HdM MIB

Version 1.4
25.09.2013

Document History

Version Nr.	Date	Changes
1.0	01.10.2009	Initial Version
1.1	15.03.2010	Anpassungen für SS 10
1.3	02.10.2012	Anpassungen für WS 12/13
1.4	25.09.2013	Anpassungen für WS 13/14

1 Einführung Data Mining

- Definitionen
- Web Mining
- Data Mining Prozess
- Kategorisierung der Anwendungen

2 Organisation der Veranstaltung

- Ablauf
- Bewertung

3 Beschreibung der Versuche

- Versuch 1 und 2
- Versuch 3
- Versuch 4
- Versuch 5
- Versuch 6

4 Literatur

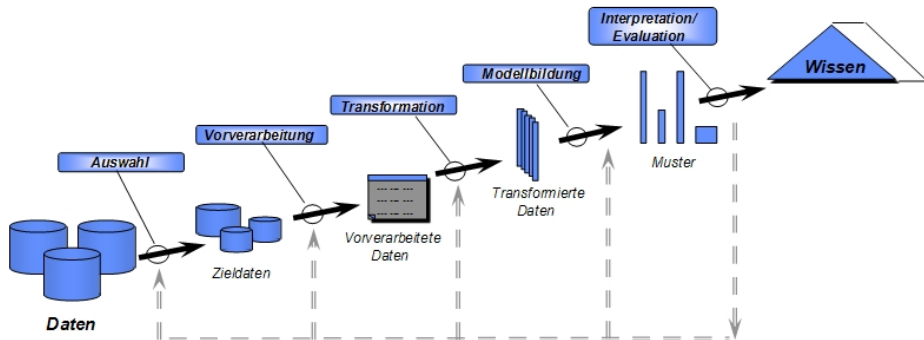
Was ist Data Mining?

- Unter Data Mining versteht man die automatische (nicht-triviale) **Extraktion von Wissen** aus meist sehr großen Datenbeständen.
- Im Data Mining werden systematisch meist statistisch-mathematisch fundierte Methoden angewandt mit dem Ziel der **Mustererkennung**.
- **Wissen** entsteht durch die Interpretation der Muster und wird meist für **prädiktive Zwecke** eingesetzt.

Was ist Web Mining?

- Web Mining ist die Anwendung von (angepassten) Data Mining Methoden auf das Web
- Web stellt einen riesigen Datenpool dar.
- Neue Möglichkeiten bieten insbesondere die User generierten Inhalte seit Web 2.0
- Besonderheiten:
 - Webseiten enthalten vornehmlich **unstrukturierten Text** ⇒ **Content Mining**
 - Webseiten sind untereinander verlinkt ⇒ **Structure Mining**
 - Server Logs enthalten Information über Nutzerverhalten ⇒ **Usage Mining**

Data Mining



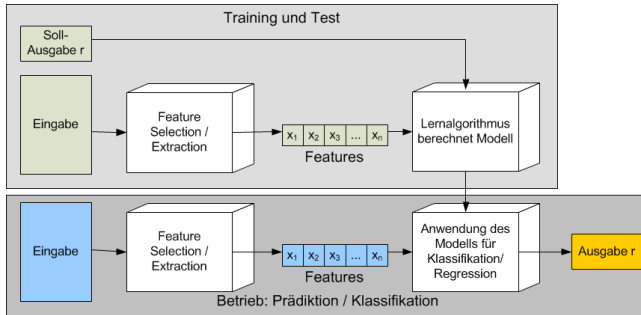
Maschinelles Lernen: Übersicht Kategorien

Wie wird gelernt? ↓

Was wird gelernt? →

	Klassifikation	Regression	Assoziation zwischen		Strategie
			Instanzen	Merkmale	
Überwacht	X	X			
Unüberwacht			X	X	
Bestärkend					X

Allgemeines Schema für Ablauf lernfähiger Systeme



Überwachtes Lernen Trainingsdaten enthalten Eingabe und zugehörige Sollausgabe

Unüberwachtes Lernen Trainingsdaten enthalten nur Eingabe

Klassifikation Ausgabe r ist diskreter (Klassenindex) Skalar oder Vektor

Regression Ausgabe r ist numerischer Skalar oder Vektor

Datenstrukturen

Struktur der Eingabedaten und der Trainingsdaten für unüberwachtes Lernen:

	Merkmal 1	Merkmal 2	...	Merkmal K
Instanz 1	$x_{1,1}$	$x_{1,2}$	\dots	$x_{1,K}$
Instanz 2	$x_{2,1}$	$x_{2,2}$	\dots	$x_{2,K}$
\vdots	\vdots	\vdots	\vdots	\vdots
Instanz N	$x_{N,1}$	$x_{N,2}$	\dots	$x_{N,K}$

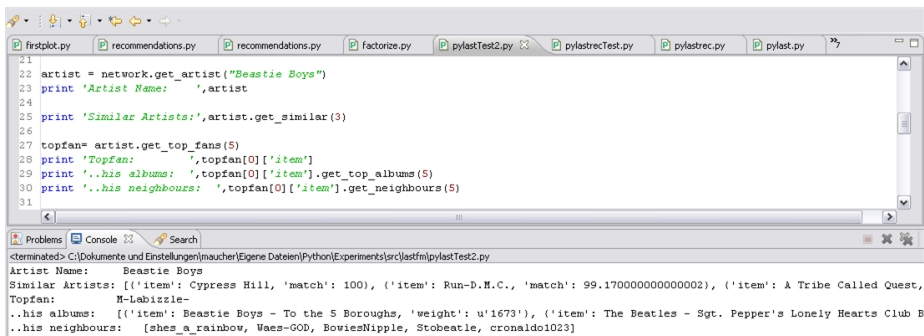
Struktur Trainingsdaten für überwachtes Lernen:

	Merkmal 1	Merkmal 2	...	Merkmal K	Sollausgabe
Instanz 1	$x_{1,1}$	$x_{1,2}$	\dots	$x_{1,K}$	r_1
Instanz 2	$x_{2,1}$	$x_{2,2}$	\dots	$x_{2,K}$	r_2
\vdots	\vdots	\vdots	\vdots	\vdots	\vdots
Instanz N	$x_{N,1}$	$x_{N,2}$	\dots	$x_{N,K}$	r_N

Ähnlichkeiten zwischen Personen oder Dingen finden

- In der **Kundenanalyse**: Welche Kunden zeigen ähnliches Kaufverhalten?
- In der **Warenkorbanalyse**: Welche Waren sind ähnlich in dem Sinne, dass sie häufig gemeinsam gekauft werden?
- **Recommender Systeme** berechnen auf der Basis der Ähnlichkeiten Empfehlungen.
- Automatische Bestimmung von und Einteilung in geographische Regionen.
- **Datenquellen**: Kundenkarten, Kundendatenbank, Web
- **Verfahren**: Clustering, Self-Organizing Maps, Collaborative Filtering

Ähnlichkeiten zwischen Personen oder Dingen finden: Last.fm



```
21
22 artist = network.get_artist("Beastie Boys")
23 print 'Artist Name: ',artist
24
25 print 'Similar Artists:',artist.get_similar(3)
26
27 topfan= artist.get_top_fans(5)
28 print 'Topfan: ',topfan[0]['item']
29 print '..his albums: ',topfan[0]['item'].get_top_albums(5)
30 print '..his neighbours: ',topfan[0]['item'].get_neighbours(5)
31
```

<terminated> C:\Dokumente und Einstellungen\maucher\Eigene Dateien\Python\Experiments\src\lastfm\pylastTest2.py

Artist Name: Beastie Boys

Similar Artists: [{'item': Cypress Hill, 'match': 100}, {'item': Run-D.M.C., 'match': 99.170000000000002}, {'item': A Tribe Called Quest,

Topfan: M-Labizzle-

..his albums: [{'item': Beastie Boys - To the 5 Boroughs, 'weight': u'1673'}, {'item': The Beatles - Sgt. Pepper's Lonely Hearts Club B

..his neighbours: [shes_a_rainbow, Waes-GOD, BowiesNipple, Stobeatie, cronaldo1023]

Ähnlichkeiten zwischen Personen oder Dingen finden: Amazon

```

experimentsSunivers. | recommendations.py | factorize.py | pylastTest2.py | pylastrecTest.py | pylastrec.py | pylast.py | »
5 #ecs.setLocale('de')
6 notebooks = ecs.ItemSearch(Keywords='Laptop', SearchIndex='PCHardware', ResponseGroup='Large', Sort='salesrank', MinimumPrice='2
7 print len(notebooks)
8 #print [bt.Title for bt in books] # books is a paged iterator
9
10 ## Display some attributes of the first item from ItemSearch
11 prodlist=[]
12 pricekat=[500,1200]
13 for i in range(100):
14     try:
15         prodgroup=notebooks[i].ProductGroup
16         price =float(notebooks[i].ListPrice.Amount)/100.0
17         title=notebooks[i].Title[0:30]
18         disk=notebooks[i].HardDiskSize
19         cpuspeed=notebooks[i].CPUSpeed
20         cputype=notebooks[i].CPUType
21         system=notebooks[i].SystemMemorySize
22         if int(system)<10:
23             system=10
24     except:
25         continue
26     prodlist.append([prodgroup, price, title, disk, cpuspeed, cputype, system])
27
28 # Print the first 10 items
29 for i in range(10):
30     print prodlist[i]
31
32 # Save the list to a file
33 f = open('prodlist.txt', 'w')
34 for i in range(len(prodlist)):
35     f.write(str(prodlist[i]) + '\n')
36 f.close()

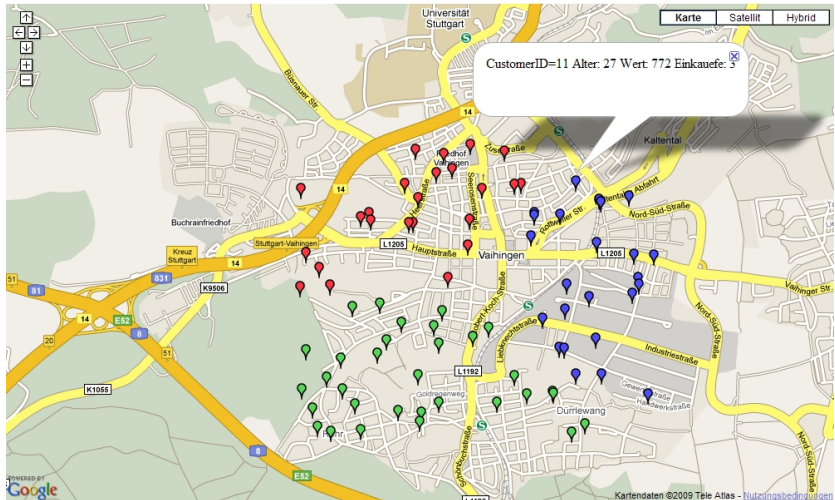
```

Problems Console Search

<terminated> C:\Dokumente und Einstellungen\maucher\Eigene Dateien\Python\Experiments\src\amazon\experimentsSunivers.py

11508							
ASUS Eee PC 1005HA-PU1X-BK 10.	389.99	1.66	Intel Atom N280	1000	SODIMM	10.1	4.0
Samsung NC10-14GB 10.2-Inch B1	379.0	1.6	Intel Core Solo	1000	DDR2 SDRAM	10.2	4.5
Toshiba Satellite L505-S6959 1	699.99	2.1	Intel Core Duo	4000	DDR2 SDRAM	16	5.0
Samsung NC10-11GP 10.2-Inch Pi	389.0	1.6	Intel Core Solo	1000	DDR2 SDRAM	10.2	4.5
Acer Aspire AS1410-8414 11.6-I	449.99	1.4	Intel Core Solo	2000	SODIMM	11.6	4.5
ASUS Eee PC 1000HE 10.1-Inch B	399.0	1.66	Intel Core Solo	1000	DDR2 SDRAM	10	4.5
HP Pavilion DV6-1230US 15.6-In	1022.0	2.1	Intel Core Duo	4000	SODIMM	15.6	3.5
Apple MacBook Pro MC118LL/A 15	1699.0	2.53	Intel Core Duo	4000	DDR2 SDRAM	15.4	4.5
HP TouchSmart TX2-1270US 12.1-	1187.0	2.2	AMD Turion 64	4000	DDR2 SDRAM	12.1	4.0
Apple MacBook MB466LL/A 13.3-I	1099.0	2	Intel Core Duo	2000	DDR3 SDRAM	13.3	4.5
Toshiba Satellite L305-S5961 1	479.99	2	PowerPC G4	2000	SDRAM	15.4	4.0
Toshiba Satellite L515-S4925 1	649.99	2.1	Intel Pentium M	4000	DDR2 SDRAM	14	4.0
HP Pavilion DV6-1260SE 15.6-In	1187.0	2.3	AMD Turion 64	4000	DDR2 SDRAM	16	4.5
Acer Aspire Timeline A53810T2-	599.99	1.3	Pentium	4000	DDR2 SDRAM	13.3	3.5

Ähnlichkeiten zwischen Personen oder Dingen finden: Geografische Cluster



Assoziationen zwischen Attributen finden

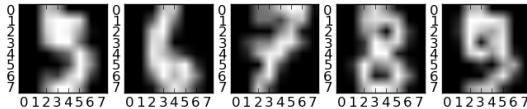
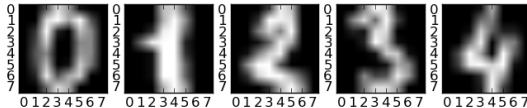
- Datensätze werden durch eine Menge von Attributen beschrieben
- In großen Mengen von Datensätzen der gleichen Art werden Regeln der Form
Wenn Attribut A1 den Wert a1 annimmt, dann nimmt in x% aller Fälle Attribut B1 den Wert b1 an.

gesucht.

- Beispiel: Frühere Masterarbeit bei Bosch:
 - Datenbank enthält 17.000 Kraftstoffproben
 - Die Daten werden 2 mal jährlich neu erhoben
 - Jeder Datensatz wird mit 141 Attributen beschrieben, z.B. Land, Stadt, Schwefel-Gehalt, Schmierfähigkeit, usw.
 - Ziel ist es Assoziationen (Regeln) zwischen bestimmten Attributwerten, z.B. niedrigem Schwefelgehalt, und überdurchschnittlichen Verschleißerscheinungen zu finden.

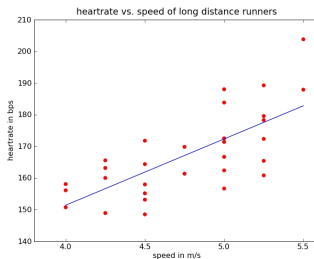
Klassifikation

- Datensätze werden abhängig von ihren Attributwerten in Klassen eingeteilt.
- Das Modell welches die Zuordnung *Eingangsattribute* \Rightarrow *Klasse* definiert wird aus Trainingsdaten gelernt.
- Nachdem das Modell gelernt ist, können damit Vorhersagen der Form
*Wenn ein Datensatz die Eingangsattribute **A** hat, dann fällt er mit einer Wahrscheinlichkeit von x% in Klasse C*
vorgenommen werden.
- Beispiel: Optical Character Recognition (OCR)



Regression

- Aus einer Vielzahl von Datensätzen eine **kontinuierliche Funktion** erlernen
- Im Gegensatz zur Klassifikation ist der Ausgabewert im Fall der Regression **nicht diskret**, sondern **numerisch**.
- Die Datensätze enthalten häufig Messreihen in welchen mehrere Parameter gemessen wurden.
- Ziel ist dann das Auffinden eines funktionalen Zusammenhangs zwischen den Parametern.
- Für die Regression können meist die gleichen Verfahren wie für die Klassifikation eingesetzt werden, z.B. Neuronale Netze, Entscheidungsbäume, Support Vector Machines, ...



Die Termine

- 1.Termin: Information zur Veranstaltung und kurze Einführung Data Mining
- 2.Termin: Gruppeneinteilung, [Python Einführung](#)
- 3.Termin: Selbständige Vorbereitung und Einarbeitung in benötigte Tools
- 4.Termin: Versuch 1 und 2: Data Mining Prozess: Energie Ressourcen
- 5.Termin: Versuch 1 und 2: Data Mining Prozess: Aktienkursvorhersage
- 6.Termin: Versuch 3: Recommender Systeme
- 7.Termin: Versuch 4: Dokument Klassifikation / Spam Filter
- 8.Termin: Versuch 5: Merkmalsextraktion und Ähnlichkeit von RSS News Feeds
- 9.Termin: Versuch 6: Face Recognition
- 7.Termin: Versuch 6: Face Recognition

Bewertung

Bewertung: Jeder Versuch und der Vortrag ergeben zusammen 5 Teilnoten¹ aus denen das arithmetische Mittel berechnet wird. In die Versuchsbewertung (individuelle Noten) fließt mit ein:

- Vorbereitung: Wird durch etwa 10-minütiges Kolloq am Versuchsbeginn abgefragt
- Qualität des abzugebenden Protokolls: Antworten auf die Fragen, Qualität der Ergebnisse, Diskussion und Analyse der Ergebnisse.
- Funktion der abzugebenden Programme
- Interesse und Experimentierfreudigkeit
- Effiziente Durchführung

Achtung: Unentschuldigte Abwesenheit wird mit der Note 4.7 für den jeweiligen Teilversuch bewertet.

¹Versuch 1 und 2 werden zusammen bearbeitet und bewertet

Data Mining Prozess am Beispiel von Energie- und Finanzdaten

Ziele: Data Mining Prozesskette verstehen

- Welche Schritte werden in welcher Reihenfolge durchgeführt?
- Persistente Datenhaltung mit HDF5
- Anwendung von unüberwachtem und überwachtem maschinellen Lernen
- Anwendung von Machine Learning Modulen des Python Pakets *scikit-learn*[SKL].

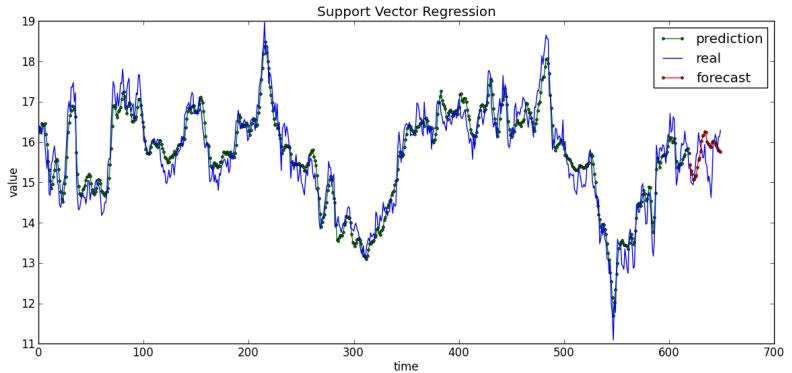
Daten: BP Energieieverbrauchsstatistik 2010, Aktienkurse von Yahoo!Finance

- Inhalte:**
- Datenauswahl und Transformation
 - Clustering von Ländern mit ähnlichen Energieverbrauchswerten
 - Vorhersage der CO_2 -Emmissionen aus Energieverbrauchsdaten
 - Geographisches Clustering und Darstellung des Ergebnis in Google Maps
 - Zugriff auf die Yahoo!-Finance API
 - Aktienkursvorhersage
 - Clustering von Unternehmen hinsichtlich ihres Kursverlaufs

Clustering nach Energieverbrauch und Darstellung in Google Maps



Aktienkursvorhersage



Recommender Systeme

Ziele: Wie können aus dem bisherigen Kaufverhalten von Kunden Produktempfehlungen berechnet werden?

- Wie können Ähnlichkeitsmaße zwischen Usern bzw. zwischen Produkten berechnet werden?
- Was ist Item- bzw. User- basiertes Collaboratives Filtering?
- Zugriff auf `lastFm` Daten über Python API?
- Wie lassen sich aus den `lastFm` Daten Empfehlungen berechnen?

Daten: • Fiktive Daten von Kunden, die Filme bewertet haben
• Über einen Python Wrapper zugänglich gemachte `lastFm` Daten

Inhalte: • Ähnlichkeiten zwischen Personen, die Filme bewertet haben, berechnen
• Ähnlichkeiten zwischen Filmen berechnen
• Empfehlungsberechnung mit User- und Item-basierten Collaborativem Filtering
• Zugriff auf `lastFm` Daten
• Ähnlichkeit zwischen `lastFm` Usern und Linkempfehlungen berechnen.

Dokumentklassifikation

Ziele: Wie können durch Textanalyse Dokumente klassifiziert werden?

- Was ist ein Naive Bayes Classifier?
- Wie kann ein Naive Bayes Classifier für die Dokumentanalyse allgemein und die Spamfilterung speziell eingesetzt werden?
- Wie kann Text in Python verarbeitet und mit Methoden der Statistik analysiert werden?

Daten: Fiktive, selbst erzeugte E-Mail Betreff-Zeilen. Danach RSS-Meldungen von verschiedenen Nachrichtenservern.

Inhalte:

- Ermittlung notwendiger Statistiken aus einer Sammlung von Dokumenten (Betreffzeilen) und Training eines Naive Bayes Classifiers
- Anwendung des trainierten Classifiers auf eigens erzeugte Betreffzeilen.
- Zugriff auf RSS-Feeds beliebiger Server
- Klassifizierung der RSS-Feeds, hier in die Kategorien, *Technik* und *Allgemein*

Merkmalsextraktion in Dokumenten

Ziele: Wie können aus großen Dokumentmengen Merkmale extrahiert werden, anhand derer eine sinnvolle und effiziente Klassifikation möglich ist.

- Wie können RSS-Feeds in Python Programme eingebunden und dort verarbeitet werden?
- Transformation der Dokumentinhalte in Formen, welche eine gute mathematische Verarbeitung zulassen
- Wie kann Ähnlichkeit zwischen Dokumenten ermittelt werden?
- Wie kann die Nicht-negative Matrix Faktorisierung für die Merkmalsextraktion angewandt werden?

Daten: RSS-Feeds einer Menge internationaler News-Server

Inhalte:

- RSS-Feeds in Pythonprogramm laden
- Aufbereitung und Analyse der RSS Feeds in Python.
- Ähnlichkeit zwischen RSS-Feeds bestimmen.
- Nicht-negative Matrix Faktorisierung implementieren und auf die RSS-Feeds anwenden.
- Beschreibung der Feed-Elemente anhand automatisch extrahierter Merkmale.

Gesichtserkennung

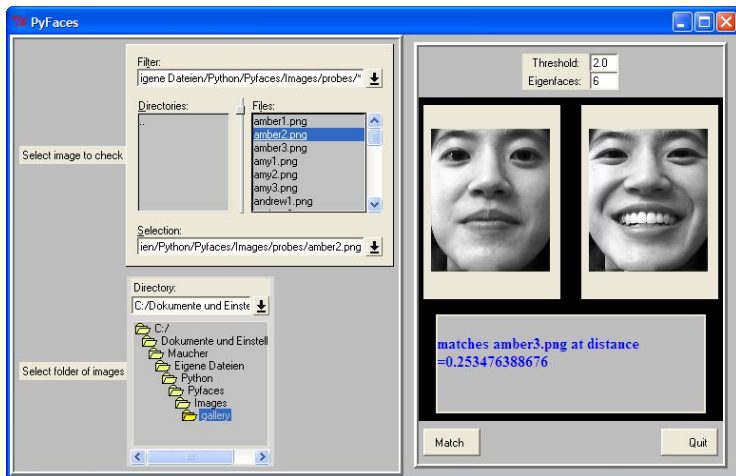
Ziele: Wie können biometrisch aufgenommene Gesichtsbilder erkannt werden?

- Transformation der Referenzbilder mit der PCA in den Eigenface-Raum
- Erkennen neu aufgenommener Gesichtsbilder im Eigenface-Raum

Daten: Fotos der Kursteilnehmer

- Inhalte:**
- Gesichtsbilder in Python einlesen und verarbeiten
 - Principal Component Analysis (PCA) in Python implementieren
 - Algorithmus zum Auffinden des nächsten Bildes implementieren

Gesichtserkennung



Referenzen



Johannes Maucher

Einführung in die Künstliche Intelligenz;
Vorlesung MIB, HdM Stuttgart, 2009



I.A. Witten; E. Frank

Data Mining; Practical Machine Learning Tools and Techniques
Elsevier Verlag, 2nd Edition, 2005



Scikit-Learn; Machine Learning Library in Python; <http://scikit-learn.org/stable//>



T. Segaran

Collective Intelligence
O'Reilly Verlag 2007



Ethem Alpaydin;

Maschinelles Lernen;
Deutsche Übersetzung im Oldenbourg Verlag; München, 2008



W. Ertel

Grundkurs Künstliche Intelligenz; Eine praxisorientierte Einführung
Vieweg Verlag 2008