Data Science 1

Vorlesung 4 - Intro

Prof. Dr. Christian Bockermann

Hochschule Bochum

Wintersemester 2021/2022

Test: Hybrid-Vorlesung

- Geteilter BBB-Bildschirm auf Vorlesungsbeamer
- Audio-Signal via Headset an BBB
- **Kein** Audio aus dem Raum in BBB Raum

Test: Hybrid-Vorlesung

- Geteilter BBB-Bildschirm auf Vorlesungsbeamer
- Audio-Signal via Headset an BBB
- **Kein** Audio aus dem Raum in BBB Raum
- **Private Chats sind nicht mehr privat!!**

Test: Hybrid-Vorlesung

- Geteilter BBB-Bildschirm auf Vorlesungsbeamer
- Audio-Signal via Headset an BBB
- **Kein** Audio aus dem Raum in BBB Raum

- **Private Chats sind nicht mehr privat!!**
- **Umfragen sind ebenfalls nicht mehr geheim!**

Test: Hybrid-Vorlesung

- Geteilter BBB-Bildschirm auf Vorlesungsbeamer
- Audio-Signal via Headset an BBB
- **Kein** Audio aus dem Raum in BBB Raum

- **Private Chats sind nicht mehr privat!!**
- **Umfragen sind ebenfalls nicht mehr geheim!**

Wünschenswert:

- Kann jemand vor Ort den Chat im Blick haben?
- Feedback von Online-Teilnehmer wichtig!!

Was geschah zuletzt?

Was geschah zuletzt?

Wir sprachen über das Pandas Modul (Vorlesung 3)!

- Modul zum Laden + Vorverarbeiten von Daten
- Prototyping: CSV Daten einlesen, Daten Filtern,...
- Indizierung von Tabellen mit `.loc[..]`, `.iloc[..]` usw.

Was geschah zuletzt?

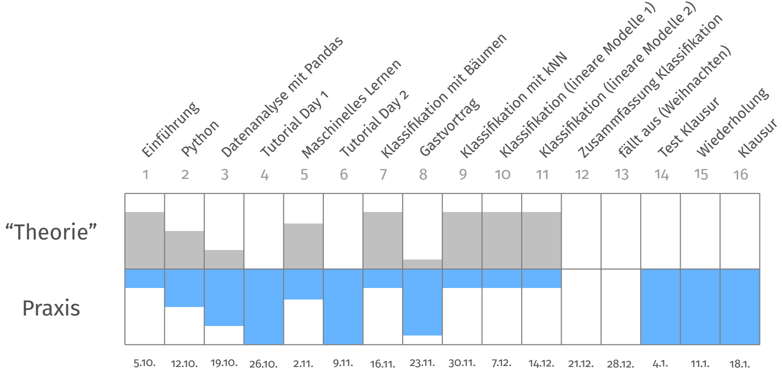
Wir sprachen über das Pandas Modul (Vorlesung 3)!

- Modul zum Laden + Vorverarbeiten von Daten
- Prototyping: CSV Daten einlesen, Daten Filtern,...
- Indizierung von Tabellen mit `.loc[..]`, `.iloc[..]` usw.

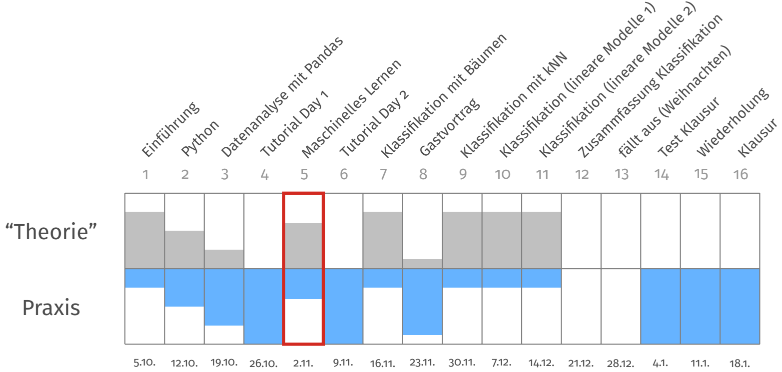
Tutorial Day 1

- Intensivierung von Python
- Wiederholung von Listen, Schleifen

Wo sind wir heute (Vorlesung 4) ?



Wo sind wir heute (Vorlesung 4) ?



Inhalt Vorlesung 4 - Worum geht's?

- Definition der Lernaufgaben des Maschinellen Lernens
- Modell-Training als Optimierungsproblem
- Modell-Validierung durch Train-/Test-Daten
- (Einfaches Python Modell `Zufall`)

Klassifikation ordnet Beispielen diskreten Klassen zu

- Vorgegebene Klassen $\mathcal{Y} = \{C_1, \ldots, C_k\}$
- Gegeben Menge $\mathbf{X} \times \mathbf{y} \subset \mathcal{X} \times \mathcal{Y}$ bei der jedem Beispiel x_i die zugehörige Klasse zugeordnet ist: (x_i, y_i)
- Qualitätsfunktion $q : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \to \mathcal{Y}) \to \mathbb{R}$

Ziel:

- Finde Modell

$$f : \mathcal{X} \to \mathcal{Y},$$

das die Qualitätsfunktion optimiert.

Klassifikation ordnet Beispielen diskreten Klassen zu

- Vorgegebene Klassen $\mathcal{Y} = \{C_1, \ldots, C_k\}$
- Gegeben Menge $\mathbf{X} \times \mathbf{y} \subset \mathcal{X} \times \mathcal{Y}$ bei der jedem Beispiel x_i die zugehörige Klasse zugeordnet ist: (x_i, y_i)
- Qualitätsfunktion $q : (\mathcal{X} \times \mathcal{Y}) \times (\mathcal{X} \to \mathcal{Y}) \to \mathbb{R}$

Ziel:

- Finde Modell

$$f : \mathcal{X} \to \mathcal{Y},$$

das die Qualitätsfunktion optimiert.

Lernen als Optimierungsproblem!

Beispiel: Klassifikation von Schwertlilien

- Klassen: $\mathcal{Y} = \{\text{setosa}, \text{versicolor}, \text{virginica}\}$
- Menge $\mathbf{X} \times \mathbf{y}$ mit 150 Beispiele mit Spalte “`species`”
- Qualitätsfunktion

$$q(\mathbf{X}\times\mathbf{y}, f) = \sum_{(x,y)\in\mathbf{X}\times\mathbf{y}} err(y, \underbrace{f(x)}_{=\hat{y}}), \quad err(y,\hat{y}) = \begin{cases} 0, & \text{falls } y = \hat{y} \\ 1, & \text{sonst.} \end{cases}$$

Beispiel: Klassifikation von Schwertlilien

- Klassen: $\mathcal{Y} = \{\text{setosa}, \text{versicolor}, \text{virginica}\}$
- Menge $\mathbf{X} \times \mathbf{y}$ mit 150 Beispiele mit Spalte “`species`”
- Qualitätsfunktion

$$q(\mathbf{X}\times\mathbf{y}, f) = \sum_{(x,y)\in\mathbf{X}\times\mathbf{y}} err(y, \underbrace{f(x)}_{=\hat{y}}), \;\; err(y, \hat{y}) = \begin{cases} 0, & \text{falls } y = \hat{y} \\ 1, & \text{sonst.} \end{cases}$$

Funktion q zählt die Anzahl der Vorhersagefehler des Modells f auf der Menge X

Beispiel: Klassifikation von Schwertlilien

- Klassen: $\mathcal{Y} = \{\text{setosa}, \text{versicolor}, \text{virginica}\}$
- Menge $\mathbf{X} \times \mathbf{y}$ mit 150 Beispiele mit Spalte “species”
- Qualitätsfunktion

$$q(\mathbf{X}\times\mathbf{y}, f) = \sum_{(x,y)\in\mathbf{X}\times\mathbf{y}} err(y, \underbrace{f(x)}_{=\hat{y}}), \quad err(y,\hat{y}) = \begin{cases} 0, & \text{falls } y = \hat{y} \\ 1, & \text{sonst.} \end{cases}$$

Funktion q zählt die Anzahl der Vorhersagefehler des Modells f auf der Menge X

Ziel: Finde f^* mit minimalem $q(X, f)$

Beispiel: **Klassifikation von Schwertlilien**

- Klassen: $\mathcal{Y} = \{\text{setosa}, \text{versicolor}, \text{virginica}\}$
- Menge $\mathbf{X} \times \mathbf{y}$ mit 150 Beispiele mit Spalte "`species`"
- Qualitätsfunktion

$$q(\mathbf{X}\times\mathbf{y}, f) = \sum_{(x,y)\in\mathbf{X}\times\mathbf{y}} err(y, \underbrace{f(x)}_{=\hat{y}}), \quad err(y, \hat{y}) = \begin{cases} 0, & \text{falls } y = \hat{y} \\ 1, & \text{sonst.} \end{cases}$$

Funktion q zählt die Anzahl der Vorhersagefehler des Modells f auf der Menge X

Ziel: Finde f^* mit minimalem $q(X, f)$ $\longrightarrow$ Optimierungsproblem

Wozu brauchen wir die Lernaufgaben?

- Fokussierung von ML-Ansätzen auf gezielte Aufgaben
- Durchaus Zusammenspiel verschiedener Lernaufgaben in einer Anwendung

Wozu brauchen wir die Lernaufgaben?

- Fokussierung von ML-Ansätzen auf gezielte Aufgaben
- Durchaus Zusammenspiel verschiedener Lernaufgaben in einer Anwendung

Beispiel: Microsoft Kinect / XBox360

Idee: Spiel-Steuerung durch Gesten/Bewegungen

Hochschule Bochum
Bochum University
of Applied Sciences

Wie kommen wir von der Kamera zur Gestensteuerung?

“Foto” in Graustufen

Bild mit Pixelraster

Bill Mockridge

Wie kommen wir von der Kamera zur Gestensteuerung?

"Foto" in Graustufen

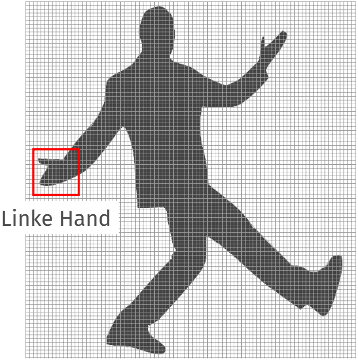


Bild mit Pixelraster

Bill Mockridge

Wie kommen wir von der Kamera zur Gestensteuerung?

“Foto” in Graustufen

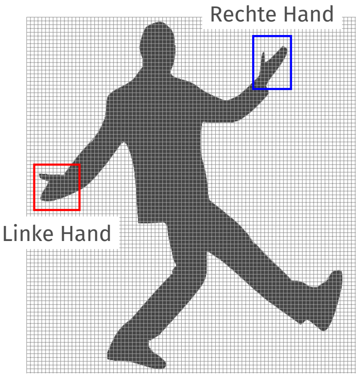


Bild mit Pixelraster

Bill Mockridge

Hochschule Bochum
Bochum University
of Applied Sciences

Idee 1: Klassifiziere jedes Pixel nach Körperteil

Bild mit Pixelraster

- Klassifikation: $f : \mathcal{X} \to \mathcal{Y}$
- $\mathcal{X} = \{pixel(x, y, color)\}$
- $\mathcal{Y} = \{\text{HandLi}, \text{HandRe}, \ldots\}$

Hochschule Bochum
Bochum University
of Applied Sciences

Idee 1: Klassifiziere jedes Pixel nach Körperteil

Bild mit Pixelraster

- Klassifikation: $f : \mathcal{X} \rightarrow \mathcal{Y}$
- $\mathcal{X} = \{pixel(x, y, color)\}$
- $\mathcal{Y} = \{\text{HandLi}, \text{HandRe}, \ldots\}$
- Farbwert *color* entspricht Tiefenwert im 3D (duale Kamera)

Idee 1: **Klassifiziere jedes Pixel nach Körperteil**

- Modell f trainieren, dass für jedes Pixel die Körperregion vorhersagt

Idee 1: **Klassifiziere jedes Pixel nach Körperteil**

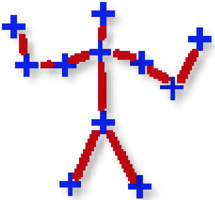
- Modell f trainieren, dass für jedes Pixel die Körperregion vorhersagt

Woher kommen die Trainingsdaten?

Idee 2: Clustering der klassifizierten Körperpixel

- Cluster-Mittelpunkt als Referenzpunkte für Körperteile

Idee 3: Referenzpunkte als Darstellung zur Gestenerkennung



- Auf vereinfachtem Körpermodell: Tracking von Hand/Fuß/...
- u.U Mustererkennung in Körperteil-Bewegungen

Beispiel: XBox 360/Kinect

- Eingabedaten: Kamera-Bilder mit Tiefen-Information
- Pixel-Klassifikation mit Entscheidungsbäumen (*Random Forest*)
- Klassifikation in Echtzeit (200 fps auf XBox GPU)

Literatur:

- *Real-Time Human Pose Recognition in Parts from Single Depth Images*, 2011
  J. Shotton, et.al.
  Microsoft Research Cambridge & Xbox Incubation

Modell-Validierung (Überwachtes Lernen)

Charakterisierung des Überwachten Lernens

- Lernen auf Daten $\mathbf{X}$ mit zugeordnetem Label $\mathbf{y}$ (="Wahrheit")
- Label oft manuell vergeben oder Messwerte (Regression)
- Validierung von Modell f durch Vergleich mit $\mathbf{y}$ möglich

Charakterisierung des Überwachten Lernens

- Lernen auf Daten **X** mit zugeordnetem Label **y** (="Wahrheit")
- Label oft manuell vergeben oder Messwerte (Regression)
- Validierung von Modell f durch Vergleich mit **y** möglich

Beispiel: MNIST-Datensatz - Ziffernerkennung

Für Trainingsdaten: Manuelle Zuordnung der Ziffernbilder zum richtigen Label (2, 9, 6,...)

Lernen auf Daten

a1	a2	a3
4	1	2
5	1	3
3	8	7

$\mathbf{X}$

y
1
-1
1

$\mathbf{y}$

$\hat{y}$
-1
-1
1

$f(\mathbf{X})$

- Lernalgorithmus sucht bestes Modell f^* für Daten $\mathbf{X}, \mathbf{y}$
- Ziel des Trainings: Fehler auf $\mathbf{X}, \mathbf{y}$ minimieren:

$$f^* = \arg\min_f \sum_{y \in \mathbf{y}} err(y, f(y)) \quad \text{(Trainingsfehler)}$$

Zentrale Frage: Wie gut ist das gelernte Modell f^* ?

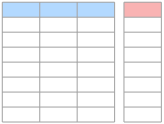
- Trainingsfehler gibt nur Auskunft über f^* auf *bekannten Daten* $\mathbf{X} \times \mathbf{y}$

Zentrale Frage: Wie gut ist das gelernte Modell f^* ?

- Trainingsfehler gibt nur Auskunft über f^* auf *bekannten Daten* $\mathbf{X} \times \mathbf{y}$

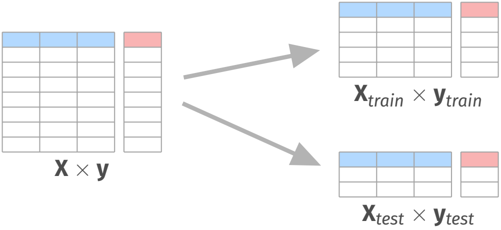
Wie gut ist f^* auf unbekannten Daten?

Ansatz: Aufteilung in Trainings- und Test-Daten



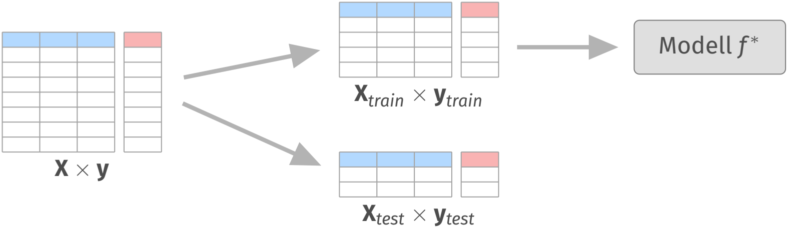
$\mathbf{X} \times \mathbf{y}$

Ansatz: Aufteilung in Trainings- und Test-Daten



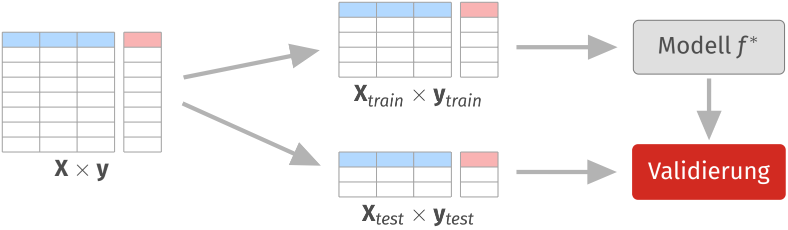
- Nutze *unabhängige* Test-Daten um f^* zu validieren!
- Oft 80% Trainingsdaten, 20% zum Testen (auch 70/30)

Ansatz: Aufteilung in Trainings- und Test-Daten



- Nutze *unabhängige* Test-Daten um f^* zu validieren!
- Oft 80% Trainingsdaten, 20% zum Testen (auch 70/30)

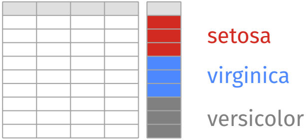
Ansatz: Aufteilung in Trainings- und Test-Daten



- Nutze *unabhängige* Test-Daten um f^* zu validieren!
- Oft 80% Trainingsdaten, 20% zum Testen (auch 70/30)

Aufteilung in Train/Test Daten

- Ok, nehmen wir 80:20 - was müssen wir beachten?
- Denken Sie an den Iris Datensatz (Übungsblatt 2, Aufgabe 2)!



Was passiert bei folgender Aufteilung von 60:40?

```
n = iris.shape[0]         # n Beispiele
splitAt = int(0.6 * n)  # 60% zum Training

X_train = iris[0:splitAt]
X_test = iris[splitAt:]
```

Wie ähnlich sollten sich $\mathbf{X}_{train} \times \mathbf{y}_{train}$ und *Xytest* sein?

Klassenverhältnis im Iris Datensatz:

- Gleichverteilt: setosa / virginica / versicolor jeweils 1/3
- Bei *linearem Splitting* im Verhältnis 60:40 ergibt sich:

$\mathbf{X}_{train} \times \mathbf{y}_{train}$	$\mathbf{X}_{test} \times \mathbf{y}_{test}$
50 × setosa 40 × virginica	10 × virginica 50 × versicolor

Wie ähnlich sollten sich $\mathbf{X}_{train} \times \mathbf{y}_{train}$ und *Xytest* sein?

Klassenverhältnis im Iris Datensatz:

- Gleichverteilt: setosa / virginica / versicolor jeweils 1/3
- Bei *linearem Splitting* im Verhältnis 60:40 ergibt sich:

$\mathbf{X}_{train} \times \mathbf{y}_{train}$	$\mathbf{X}_{test} \times \mathbf{y}_{test}$
50 × setosa 40 × virginica	10 × virginica 50 × versicolor

Klasse versicolor in Trainingsdaten <u>nicht enthalten</u>!

Klasse setosa in Testdaten <u>nicht enthalten</u>!

Split gemäß der Klassenverteilung: Stratified Sampling

- Stratified Sampling erhält die Klassenverhältnisse
- Beispiel für 60:40 Split:

$n = 150$

setosa	$50 = \frac{1}{3} \cdot n$
virginica	$50 = \frac{1}{3} \cdot n$
versicolor	$50 = \frac{1}{3} \cdot n$

$\mathbf{X} \times \mathbf{y}$

setosa	$\frac{1}{3} \cdot \frac{3}{5} \cdot n = 30$
virginica	$\frac{1}{3} \cdot \frac{3}{5} \cdot n = 30$
versicolor	$\frac{1}{3} \cdot \frac{3}{5} \cdot n = 30$

$\mathbf{X}_{train} \times \mathbf{y}_{train}$

setosa	$\frac{1}{3} \cdot \frac{2}{5} \cdot n = 20$
virginica	$\frac{1}{3} \cdot \frac{2}{5} \cdot n = 20$
versicolor	$\frac{1}{3} \cdot \frac{2}{5} \cdot n = 20$

$\mathbf{X}_{test} \times \mathbf{y}_{test}$

Split gemäß der Klassenverteilung: Stratified Sampling

- Stratified Sampling erhält die Klassenverhältnisse
- Beispiel für 60:40 Split:

$n = 150$

setosa	$50 = \frac{1}{3} \cdot n$
virginica	$50 = \frac{1}{3} \cdot n$
versicolor	$50 = \frac{1}{3} \cdot n$

$\mathbf{X} \times \mathbf{y}$

setosa	$\frac{1}{3} \cdot \frac{3}{5} \cdot n = 30$
virginica	$\frac{1}{3} \cdot \frac{3}{5} \cdot n = 30$
versicolor	$\frac{1}{3} \cdot \frac{3}{5} \cdot n = 30$

$\mathbf{X}_{train} \times \mathbf{y}_{train}$

setosa	$\frac{1}{3} \cdot \frac{2}{5} \cdot n = 20$
virginica	$\frac{1}{3} \cdot \frac{2}{5} \cdot n = 20$
versicolor	$\frac{1}{3} \cdot \frac{2}{5} \cdot n = 20$

$\mathbf{X}_{test} \times \mathbf{y}_{test}$

Aber: Was ist mit den Verteilungen der anderen Attribute?
Zum Beispiel `sepal_length`?

Tutorial Day 2

- Kommenden **Dienstag**, um **8:30 Uhr**
- Kurze Wiederholungsvorlesung + Übungen

- Weiterführung von Tutorial Day 1
- Keine Folien zum Vorbereiten
- Mehr Übungen zu Python und Pandas