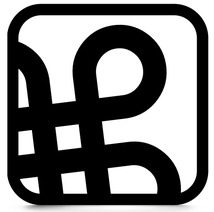


Macoun

SMART Phones - AI für iOS

Marco Köppel und Dominic Opitz

Wir sind keine Experten

Marco Köppel
Senior Developer
Emerging Experiences
Razorfish

Dominic Opitz
Developer
Emerging Experiences
Razorfish

Warum machen wir Machine Learning?

Wir sind auf Probleme gestoßen,
die wir durch programmieren
kaum lösen konnten.

Skateboard Push Tracker

Ziele

- Motiondaten mit iPhone aufzeichnen
- Device befindet sich dabei am Bein
- **GEHEN, STEHEN** und **PUSHEN** unterscheiden

First Try

```
CMAcceleration(x: -0.002198701766192913, y: 0.051184572279453278, z: 0.13377116620540619) CMAcceleration(x: 0.074266031384468079, y:
-0.79918569326400757, z: -0.59647870063781738) CMRotationRate(x: -0.72945111989974976, y: 0.99721711874008179, z: -1.1429635286331177)
CMAcceleration(x: 0.010604003444314003, y: 0.017799567431211472, z: 0.013400311581790447) CMAcceleration(x: 0.10299768298864365, y:
-0.78722399473190308, z: -0.60800480842590332) CMRotationRate(x: -0.9024428129196167, y: 0.77326393127441406, z: -1.134127140045166)
CMAcceleration(x: 0.0091210799291729927, y: 0.0030949583742767572, z: 0.0067848917096853256) CMAcceleration(x: 0.12591920793056488, y:
-0.77218371629714966, z: -0.62279748916625977) CMRotationRate(x: -1.1354789733886719, y: 0.32164925336837769, z: -1.0921088457107544)
CMAcceleration(x: 0.024776104837656021, y: -0.0077906004153192043, z: 0.041429042816162109) CMAcceleration(x: 0.14433705806732178, y:
-0.75316518545150757, z: -0.64180135726928711) CMRotationRate(x: -1.3546427488327026, y: 0.13530972599983215, z: -1.0637648105621338)
CMAcceleration(x: 0.030189665034413338, y: -0.020062947645783424, z: 0.087133824825286865) CMAcceleration(x: 0.16233047842979431, y:
-0.73311090469360352, z: -0.66045230627059937) CMRotationRate(x: -1.2761234045028687, y: 0.42047163844108582, z: -0.93186748027801514)
CMAcceleration(x: 0.082287967205047607, y: -0.045521333813667297, z: 0.061294469982385635) CMAcceleration(x: 0.18428307771682739, y:
-0.7154955267906189, z: -0.67387384176254272) CMRotationRate(x: -1.0489048957824707, y: 0.94382452964782715, z: -0.7290075421333313)
CMAcceleration(x: 0.020043021067976952, y: -0.0010269673075526953, z: 0.06962161511182785) CMAcceleration(x: 0.2063363790512085, y:
-0.69916832447052002, z: -0.68453556299209595) CMRotationRate(x: -1.0742801427841187, y: 1.0276204347610474, z: -0.56881415843963623)
CMAcceleration(x: 0.033384282141923904, y: 0.018697384744882584, z: 0.0011606441112235188) CMAcceleration(x: 0.22729687392711639, y:
-0.67903673648834229, z: -0.69802951812744141) CMRotationRate(x: -1.4101288318634033, y: 0.66383516788482666, z: -0.76786625385284424)
CMAcceleration(x: 0.090244121849536896, y: 0.034625429660081863, z: -0.013745960779488087) CMAcceleration(x: 0.2403222918510437, y:
-0.65172135829925537, z: -0.71937781572341919) CMRotationRate(x: -1.8331649303436279, y: -0.0057807518169283867, z: -0.84327793121337891)
CMAcceleration(x: -0.0069711022579818964, y: -0.02837078832089901, z: 0.012506791390478611) CMAcceleration(x: 0.24995197355747223, y:
-0.62031084299087524, z: -0.74346381425857544) CMRotationRate(x: -1.8873127698898315, y: -0.048122655600309372, z: -0.73739546537399292)
CMAcceleration(x: 0.048803605139255524, y: -0.0055771628394722939, z: 0.0084950784221291542) CMAcceleration(x: 0.25596019625663757, y:
-0.58922570943832397, z: -0.76635336875915527) CMRotationRate(x: -1.8339693546295166, y: -0.25578641891479492, z: -0.73997128009796143)
CMAcceleration(x: 0.12190590053796768, y: -0.038918342441320419, z: 0.0086646620184183121) CMAcceleration(x: 0.26028099656105042, y:
-0.55763930082321167, z: -0.78822094202041626) CMRotationRate(x: -1.8055182695388794, y: -0.09541301429271698, z: -0.55356806516647339)
CMAcceleration(x: 0.086791418492794037, y: -0.073909208178520203, z: 0.018561532720923424) CMAcceleration(x: 0.26600703597068787, y:
-0.5312696099281311, z: -0.80435866117477417) CMRotationRate(x: -1.4757566452026367, y: 0.39034834504127502, z: 0.008380768820643425)
CMAcceleration(x: -0.029533658176660538, y: -0.0017736318986862898, z: 0.0163121297955513) CMAcceleration(x: 0.27004268765449524, y:
-0.5115930438041687, z: -0.8156895637512207) CMRotationRate(x: -1.2554327249526978, y: 0.33240693807601929, z: 0.33968654274940491)
CMAcceleration(x: -0.073882840573787689, y: 0.0430105540306568146, z: 0.030234113335609436) CMAcceleration(x: 0.26953104138374329, y:
-0.49386197328567505, z: -0.82671236991882324) CMRotationRate(x: -1.2206333875656128, y: 0.27242001891136169, z: 0.64833784103393555)
CMAcceleration(x: -0.002352831419557333, y: -0.0069707366637885571, z: 0.0073112025856971741) CMAcceleration(x: 0.26848137378692627, y:
-0.48208871483802795, z: -0.83397138118743896) CMRotationRate(x: -0.88087606430053711, y: 0.616527259349823, z: 1.1275794506072998)
CMAcceleration(x: -0.041261706501245499, y: -0.053667884320020676, z: -0.15082763135433197) CMAcceleration(x: 0.2673664391040802, y:
-0.47976413369178772, z: -0.83566832542419434) CMRotationRate(x: -0.43149185180664062, y: 0.60652172565460205, z: 1.4823871850967407)
CMAcceleration(x: 0.061224699020385742, y: 0.049033477902412415, z: -0.05043792724609375) CMAcceleration(x: 0.26032376289367676, y:
-0.48322230577468872, z: -0.83589935302734375) CMRotationRate(x: -0.2216310054063797, y: 0.46699017286300659, z: 1.603003978729248)
```


●●●●● o2-de
14:58
95 %
Session
Acceleration
0
42
84
126
168
210
252
294
2,00
1,00
0,00
-1,00
-2,00
-3,00

Ergebnis ✗

- Kompliziert
- Zeitaufwändig
- Bewegungen werden bestimmt nicht präzise erkannt!

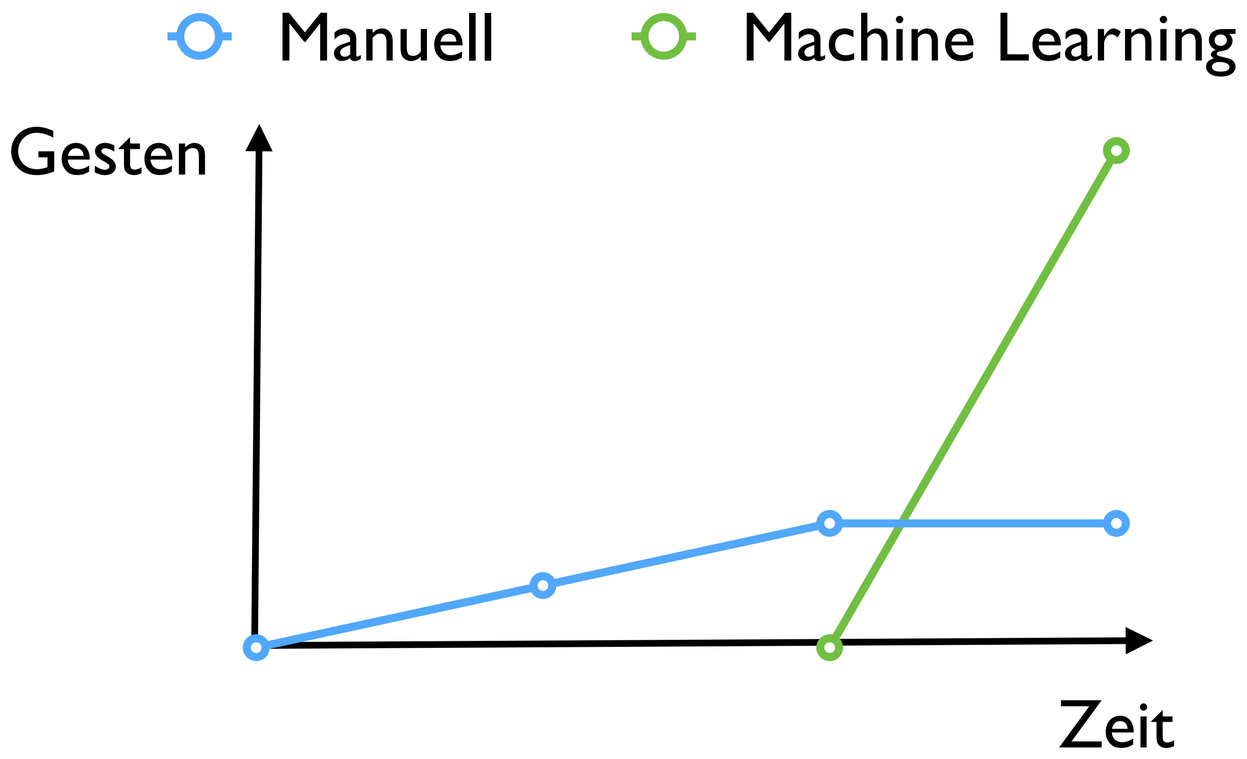
Photobooth

- Geste nachahmen
- Kamera auslösen
- Bild ausdrucken

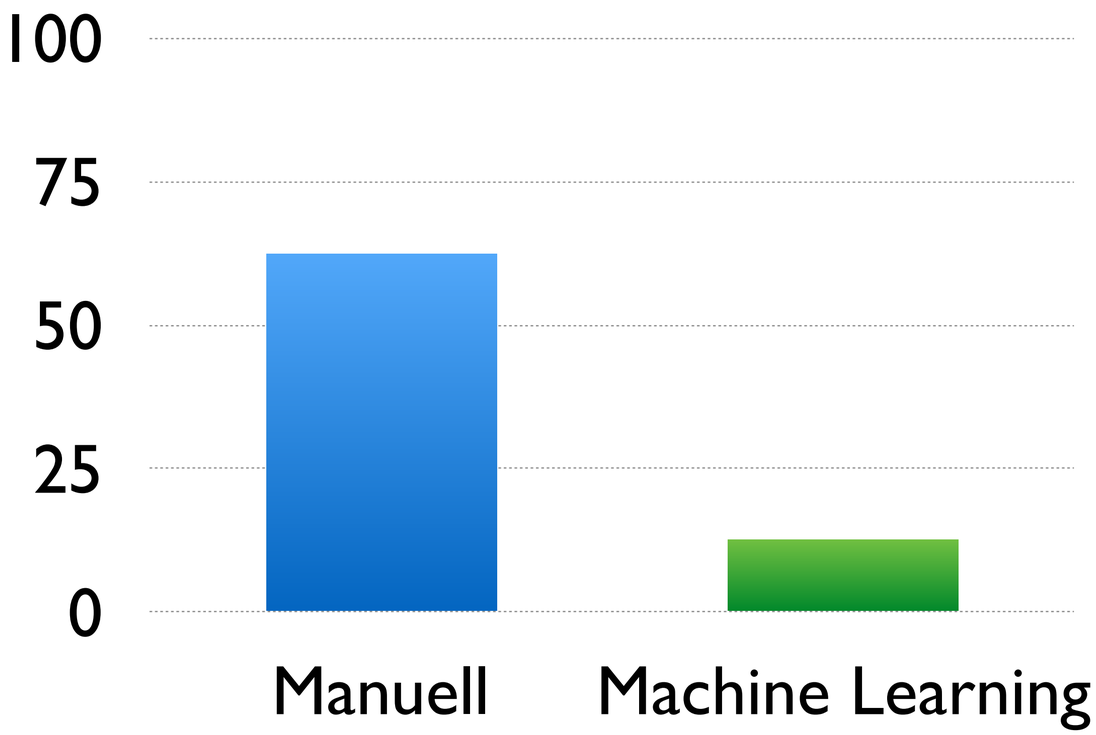
Gestenerkennung

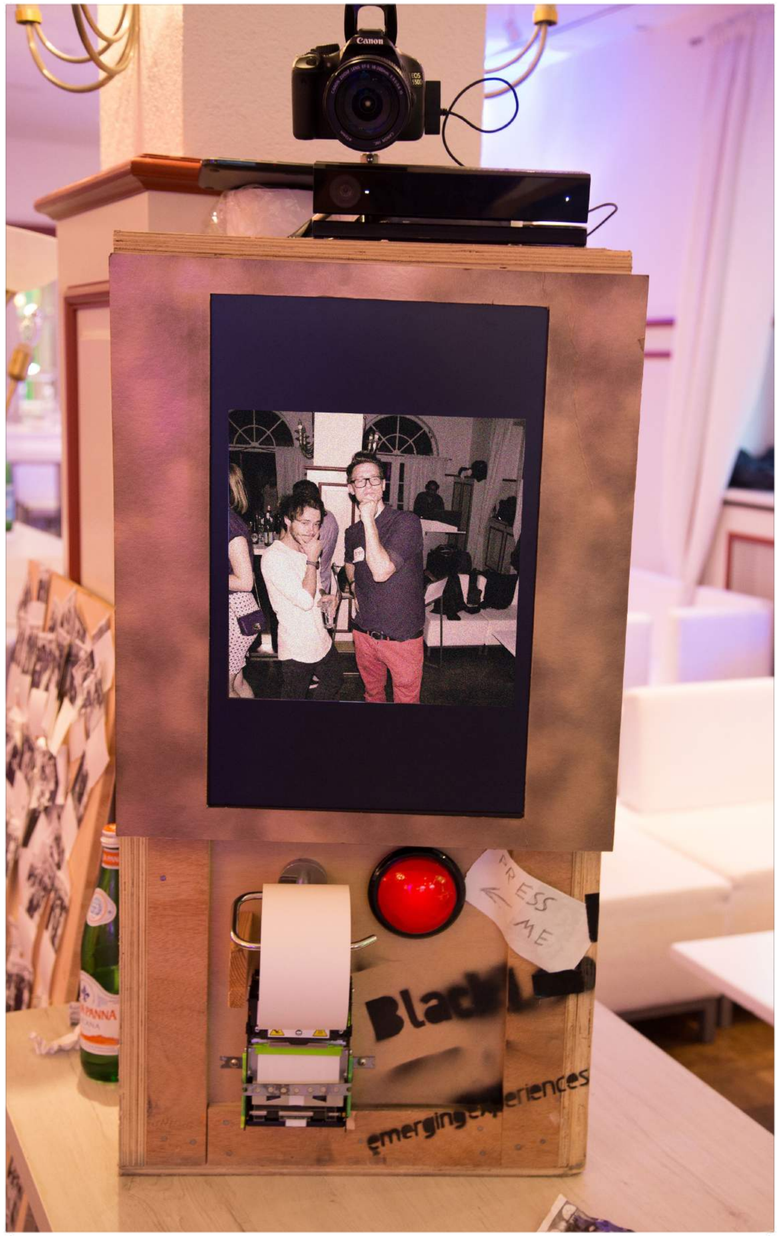
- Kinect body tracking
- Visual Gesture Builder
- Geste aufnehmen
- Algorithmus wird generiert

Zeitaufwand



Fehlerrate




Canon
PRESS ME
emerging experiences

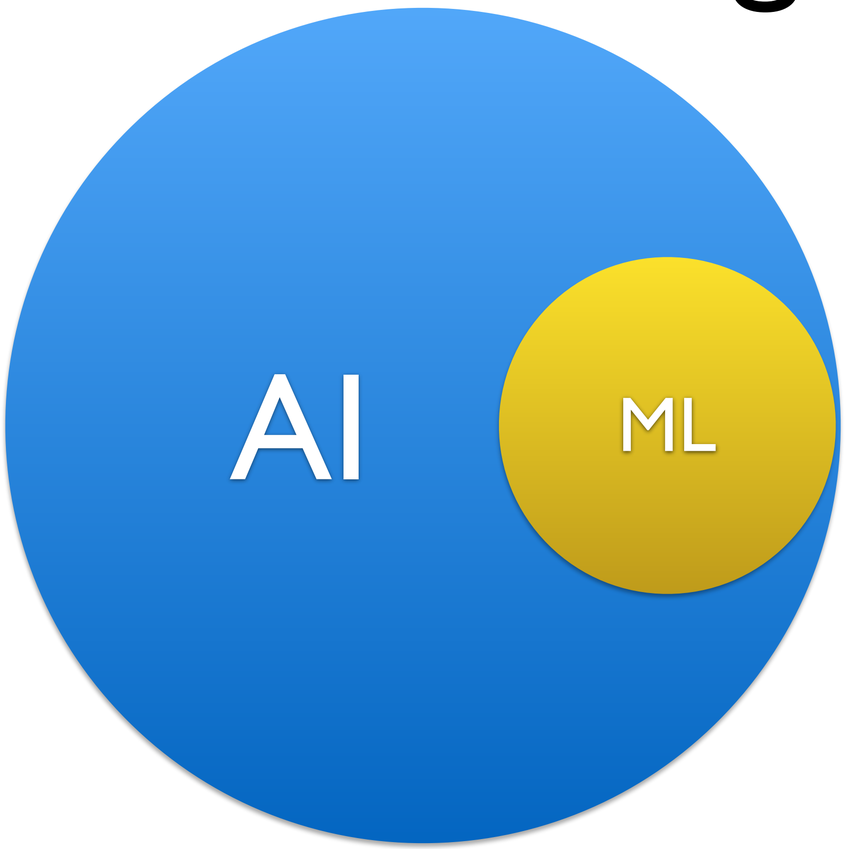
Was ist Machine Learning?

Field of study that gives computers the ability to learn without being explicitly programmed.

Arthur Samuel

Machine Learning ≠ AI

Machine Learning ≠ AI



Gute Gründe für ML

- Manche Funktionen kann man nicht genau genug beschreiben. (z.B. Gesten)
- Manche Funktionalitäten verhalten sich in realer Umgebung anders als erwartet. (z.B. Kühlanlagen)
- Umgebungen können sich ändern oder sind schlicht nicht bekannt. (z.B. Roboter und Recommendation Engines)
- …

Populäre Einsatzgebiete

- Virtuelle Assistenten a.k.a. Siri
  (Spracherkennung, App-Vorschläge, etc.)
- Roboter
  (Staubsauger und Rasenmäher, Automated Cars)
- Recommendation Engines
  (Amazon, Netflix)
- …

Ran an die Maschine

Supervised Classification

Classification - Regression

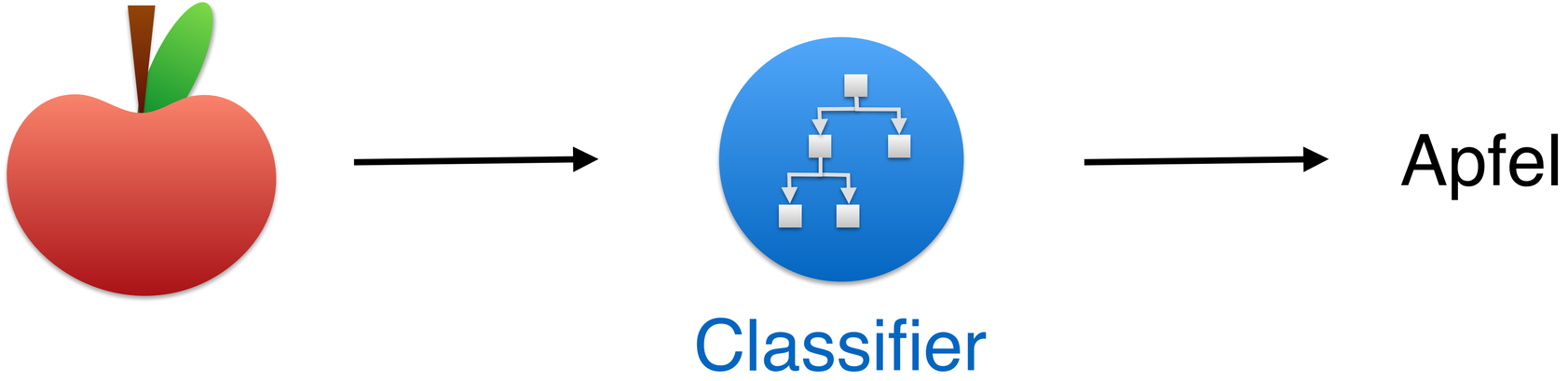
Classification: Output ist eine Klasse. A oder B oder C.

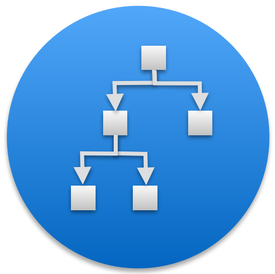
Regression: Output ist ein rationaler Wert.

Supervised - Unsupervised

Supervised: Trainingsdaten wurden eine Klasse bzw. Wert zugeordnet.

Unsupervised: Trainingsdaten werden gruppiert aber nicht beschrieben.

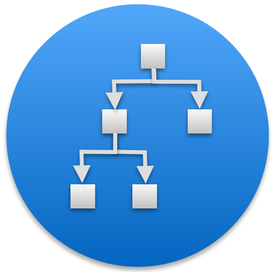

Classifier
Apfel



Classifier

Banane


Apfel
Banane
Classifier



Mathematische Funktionen, die die Zusammenhänge zwischen einem mehrdimensionalen Input und dem Output beschreiben.

Parameter für die Funktionen werden von der Maschine selbst gefunden.

Das Rezept

Schritt 1:
Daten
sammeln

Schritt 2:
Classifier
trainieren

Schritt 3:
Vorhersagen
machen

Das Rezept

Schritt 1: Daten sammeln

Schritt 2: Classifier trainieren

Schritt 3: Vorhersagen machen

Apfel oder Banane?

Farbe	Gewicht	Größe	Label
Rot	120	6,2	Apfel
Gelb	140	10,5	Banane
Grün	104	5,9	Apfel
Rot	118	6,3	Apfel
Gelb	148	11,8	Banane
Gelb	128	10,1	Banane
Grün	131	10,1	Banane

Apfel oder Banane?

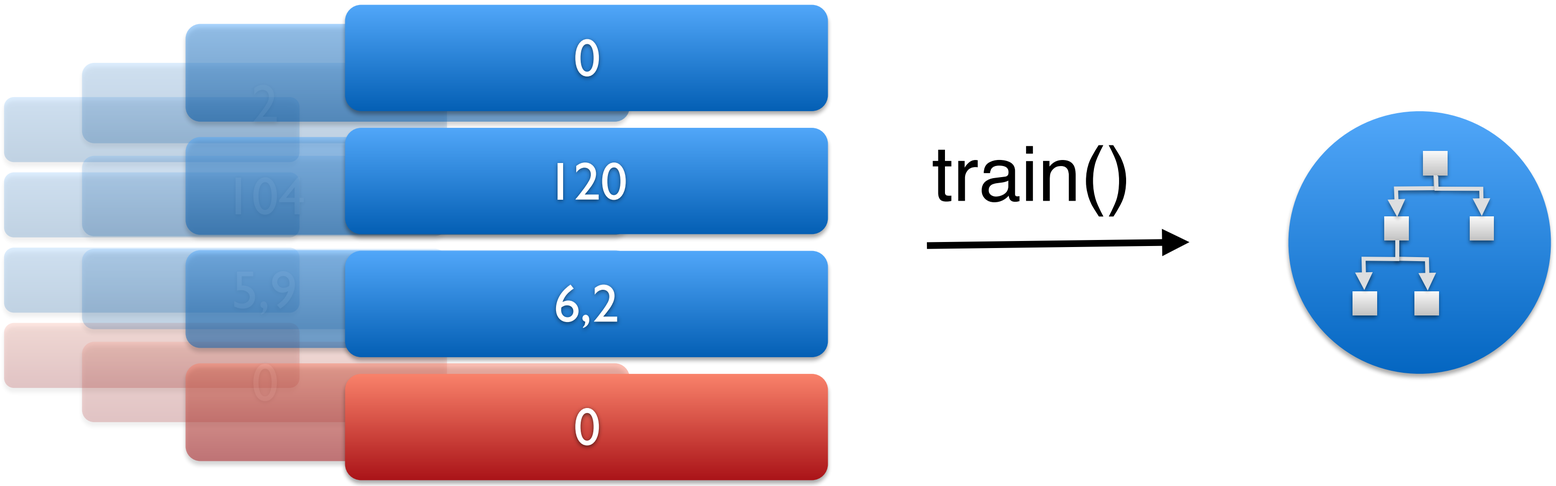
Farbe	Gewicht	Größe	Label
0	120	6,2	0
1	140	10,5	1
2	104	5,9	0
0	118	6,3	0
1	148	11,8	1
1	128	10,1	1
2	131	10,1	1

Das Rezept

Schritt 1:
Daten
sammeln

Schritt 2:
Classifier
trainieren

Schritt 3:
Vorhersagen
machen

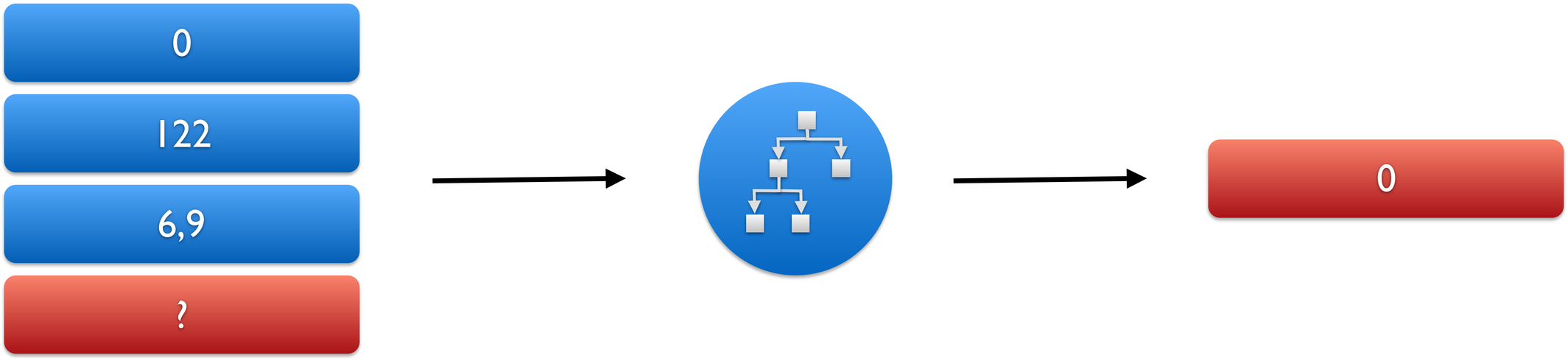

0
2
120
104
6,2
5,9
0
0
train()

Das Rezept

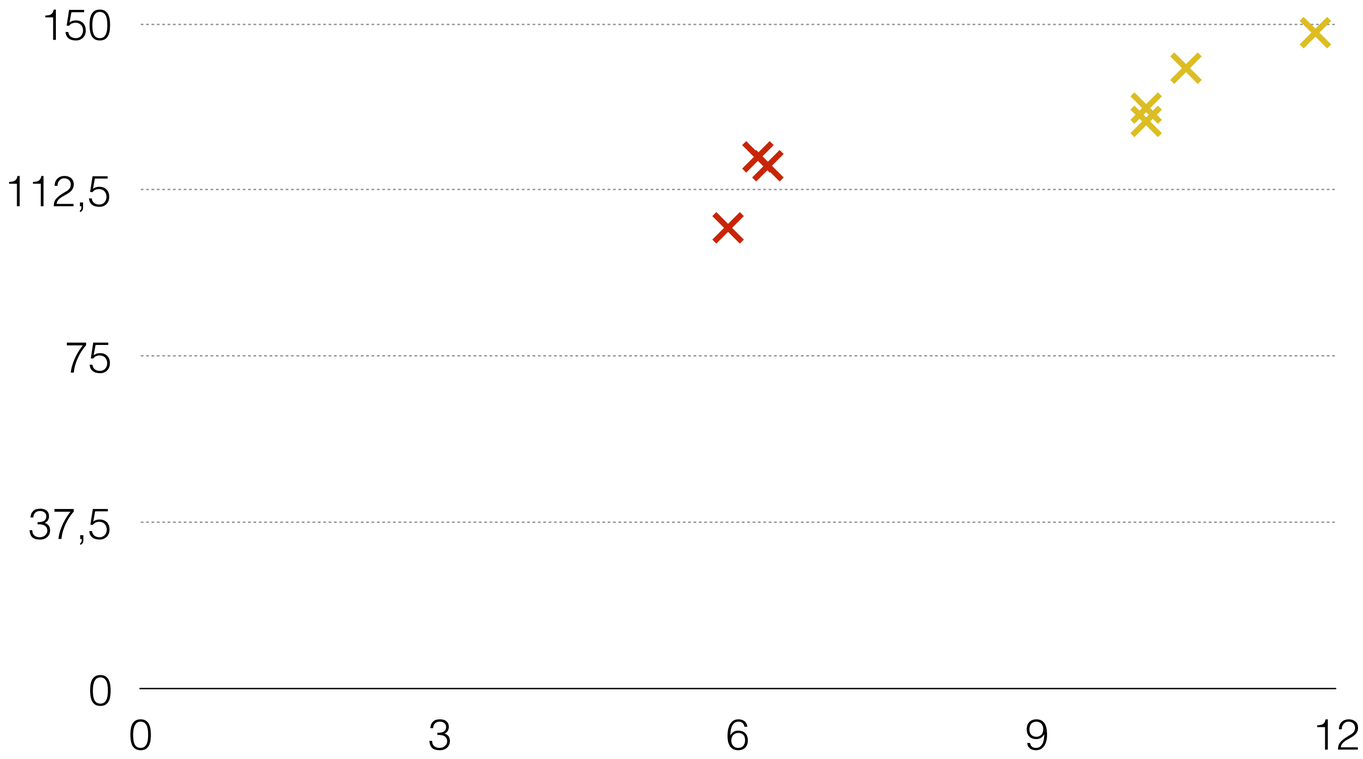
Schritt 1: Daten sammeln

Schritt 2: Classifier trainieren

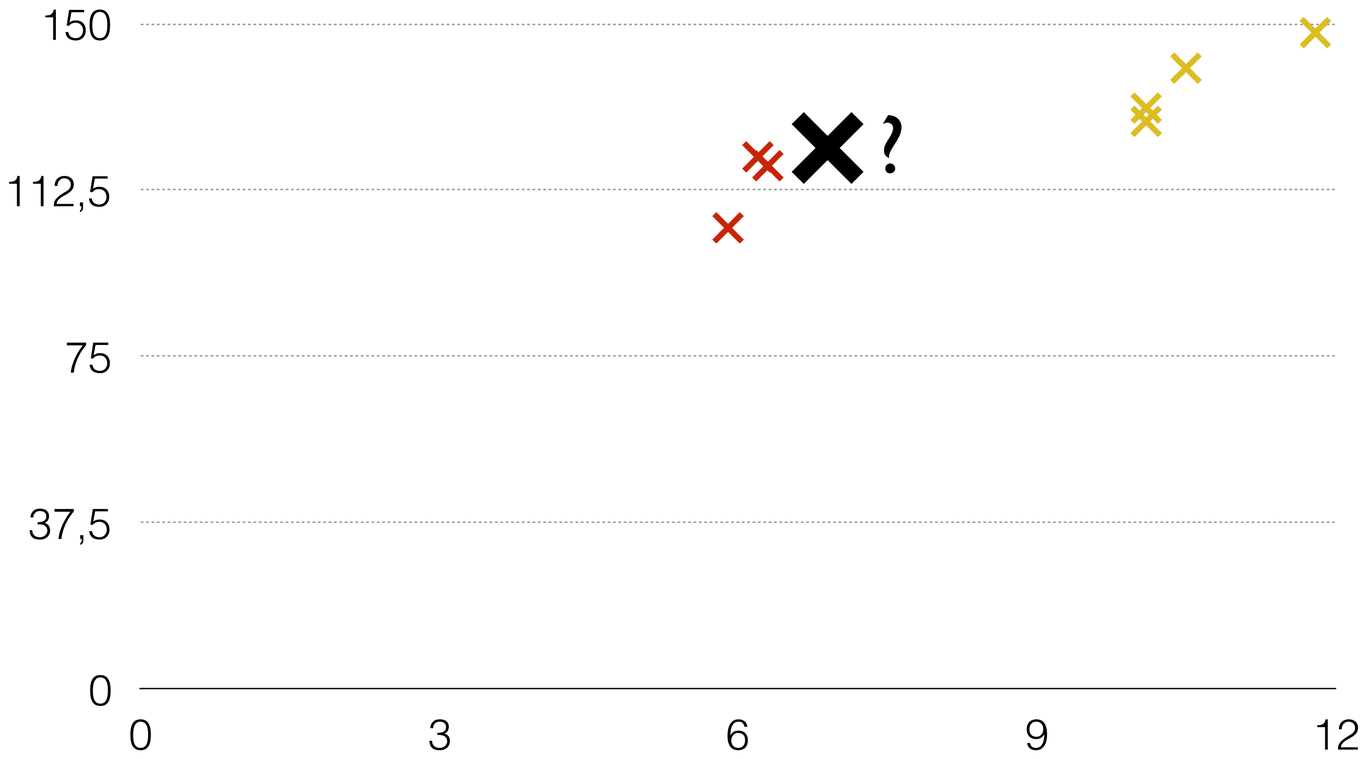
Schritt 3: Vorhersagen machen


0
122
6,9
?
0

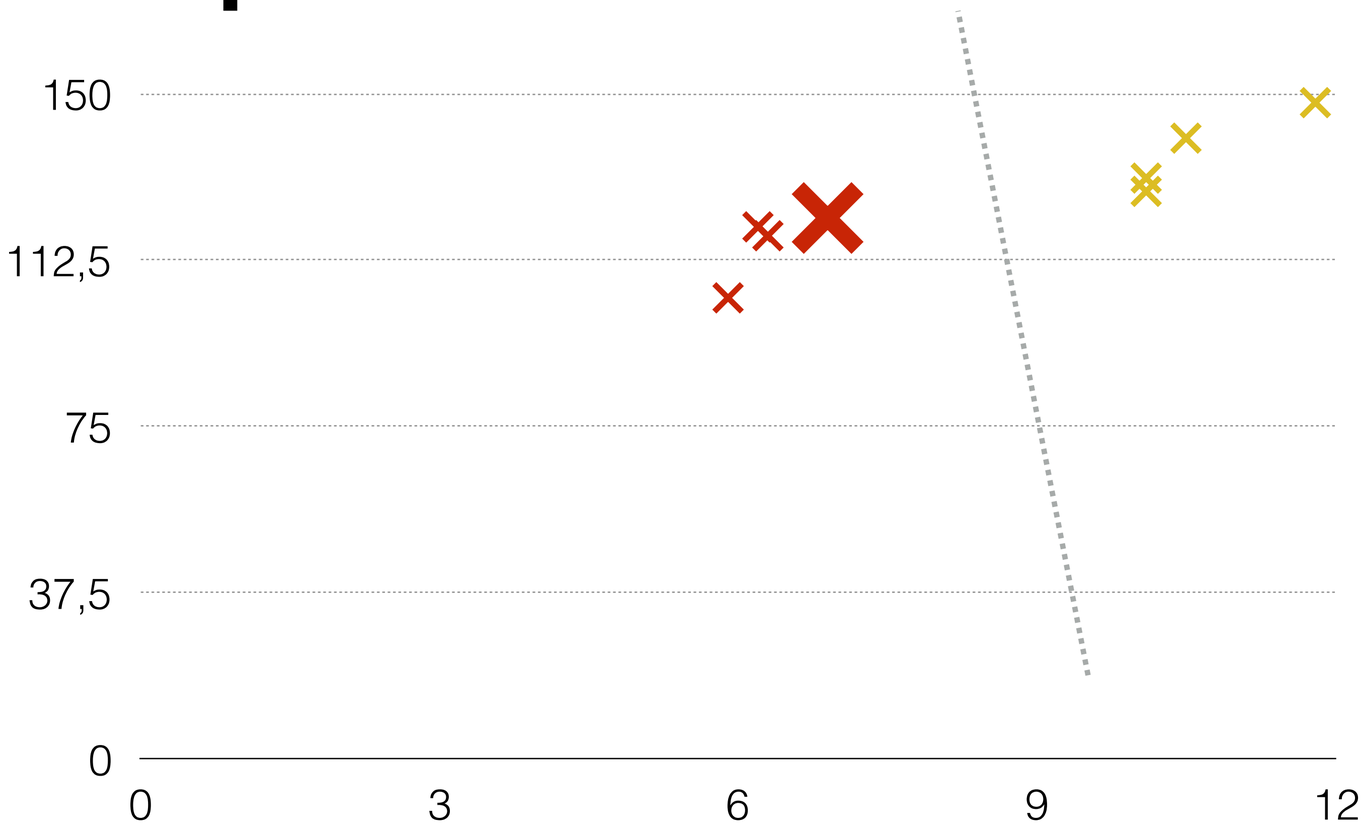
Apfel oder Banane?



Apfel oder Banane?



Apfel oder Banane?

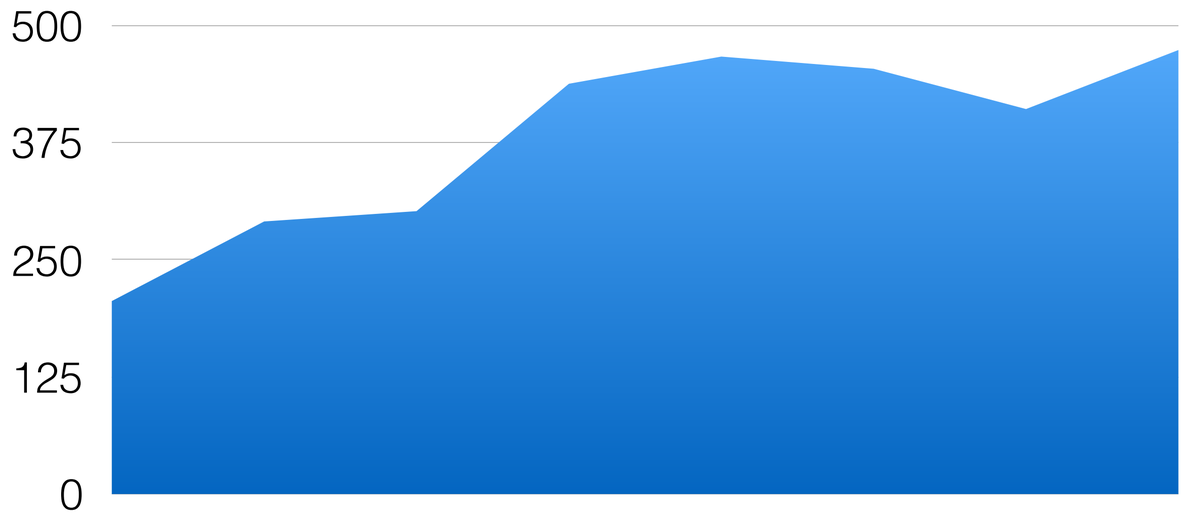


Ein weiteres Beispiel

Supervised Regression
mit der Macoun

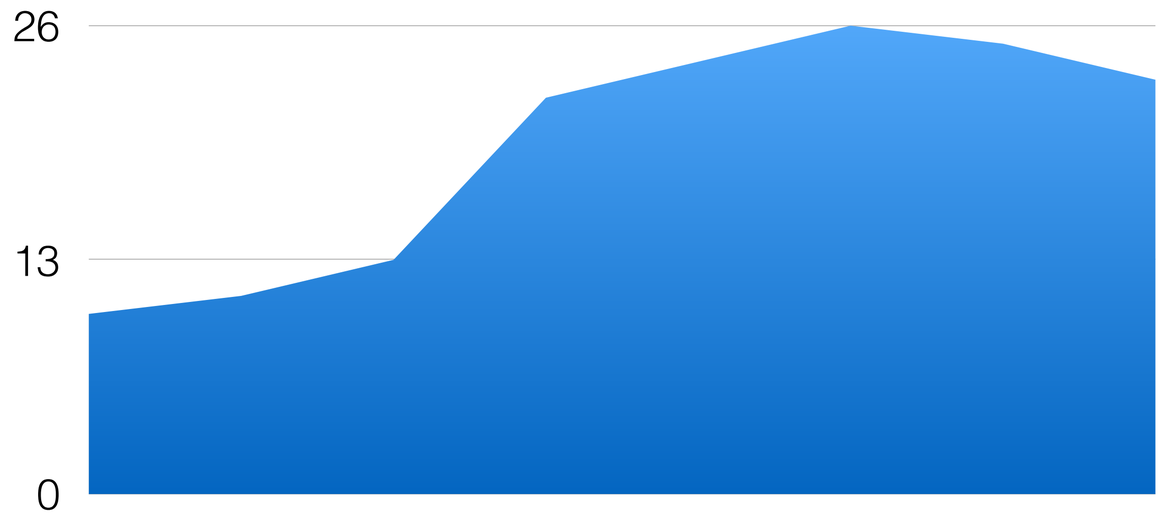
Macoun

- Besucherzahlen:



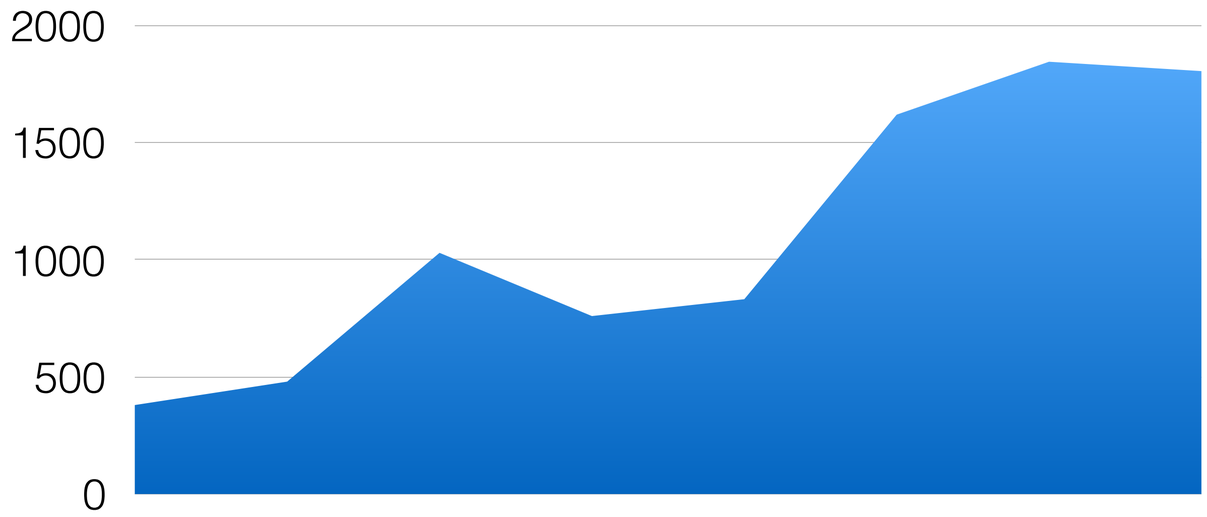
Macoun

- Sprecher:



Macoun

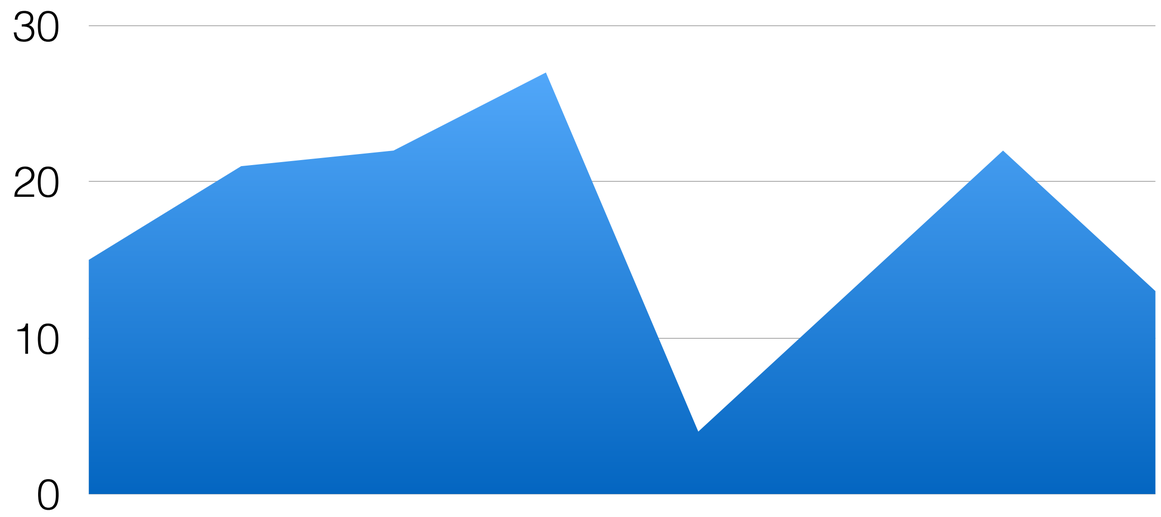
- Kaffeeverbrauch:



Keine direkten Korrelationen.

Macoun

- Temperatur:



Macoun

- Hat es geregnet:

Wird nicht wirklich besser

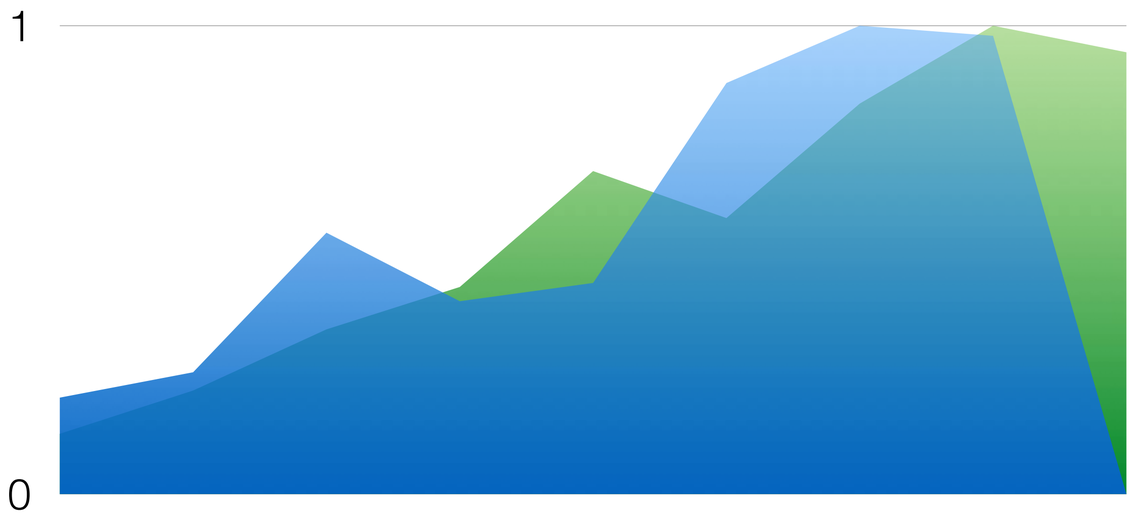
Macoun

- AAPL:

WTF?

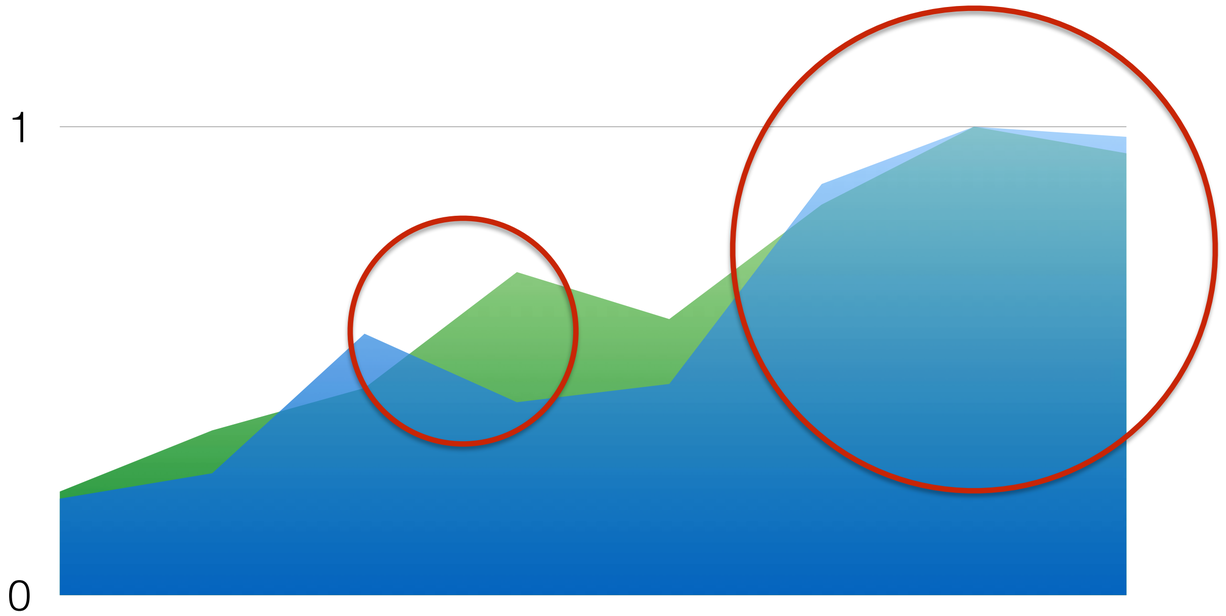
Macoun

- Kaffeeverbrauch:
- AAPL:



Macoun

- Kaffeeverbrauch:
- AAPL + 1 Jahre:



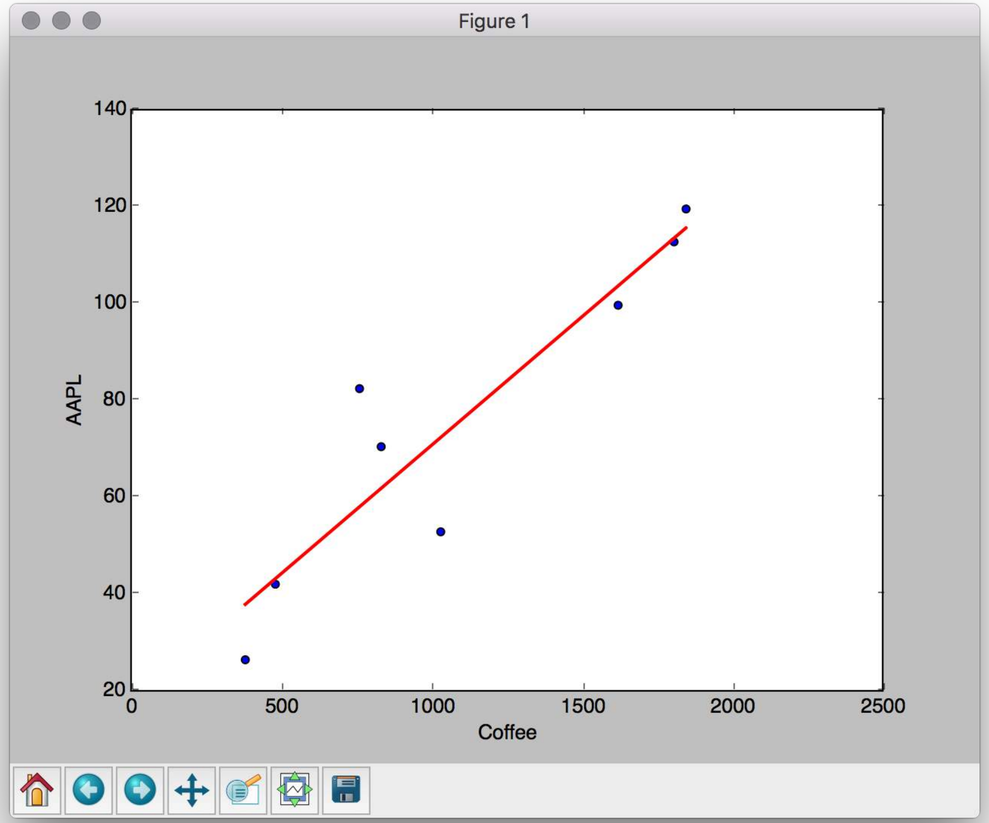
python - scikit learn

```
import numpy
from sklearn import linear_model

features = [[380], [480], [1030], [760], [832], [1620], [1846], [1806]]
target = [26.41, 42.01, 52.83, 82.4, 70.41, 99.62, 119.5, 112.71]

reg = linear_model.LinearRegression()
reg.fit(features, target)
```

python - scikit learn

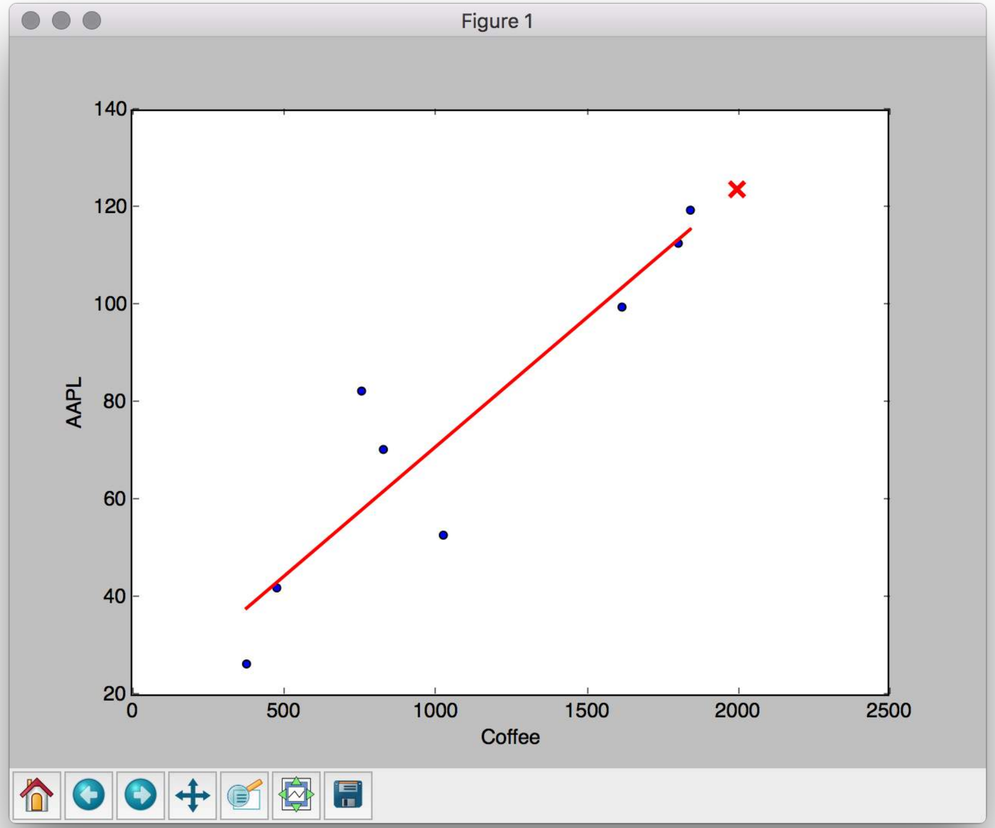


python - scikit learn

```
print reg.predict([2000])

>> [126.76258665]
```

python - scikit learn



Alle Angaben ohne Gewähr

Motion Learning

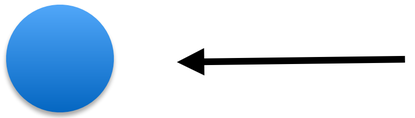
One more try

- Push Tracker mit Machine Learning umsetzen
- Offline
- Training und Analyse auf dem Device

Swift AI

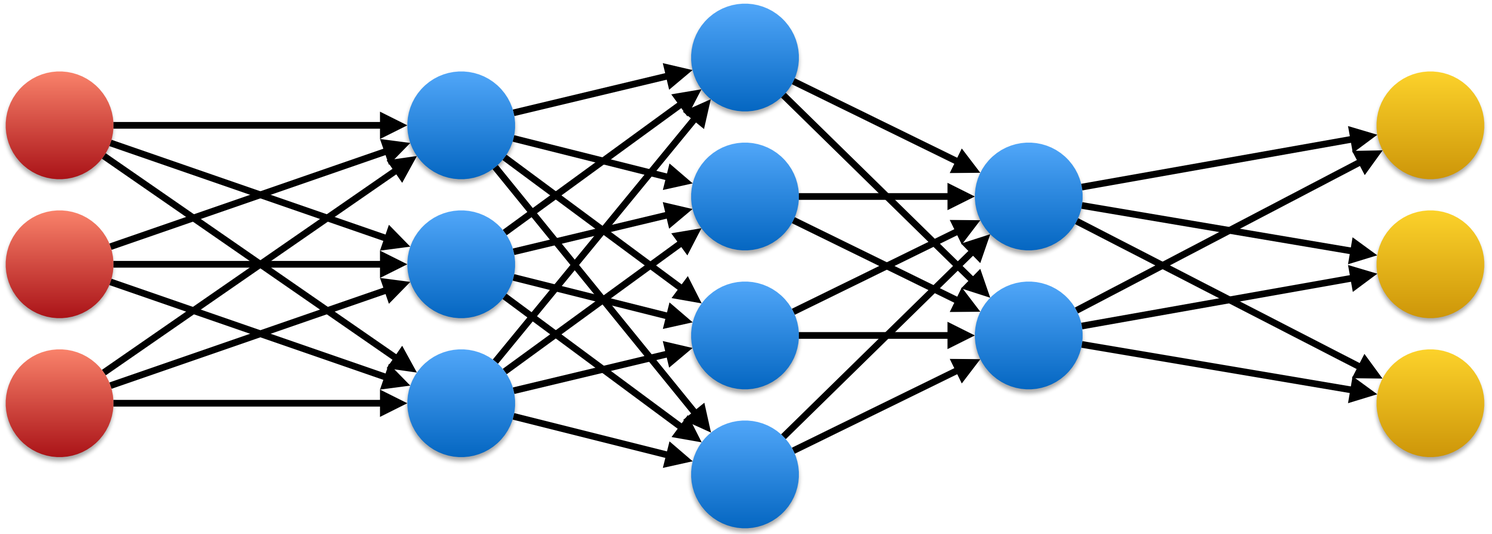
- Native Swift Library für iOS und macOS
- Accelerate Framework
- Feedforward Neural Network (FFNN)

Neural Network



Das ist ein Neuron

Neural Network

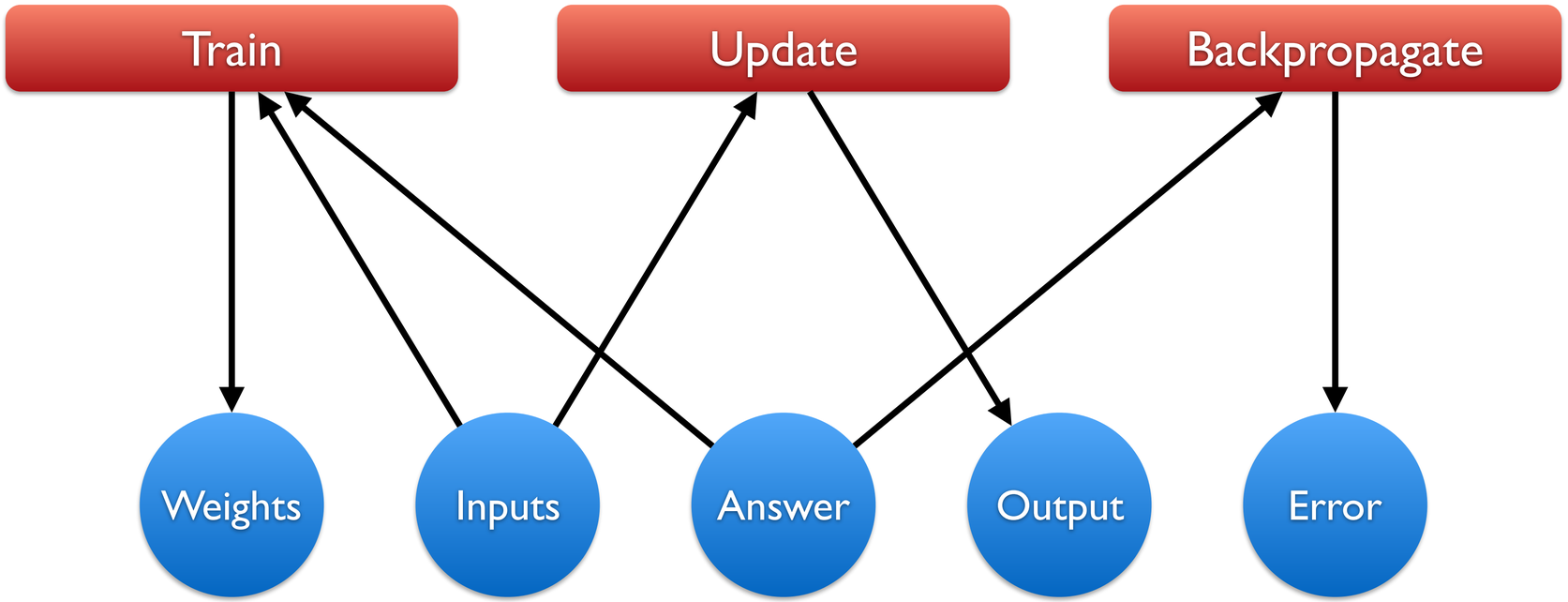


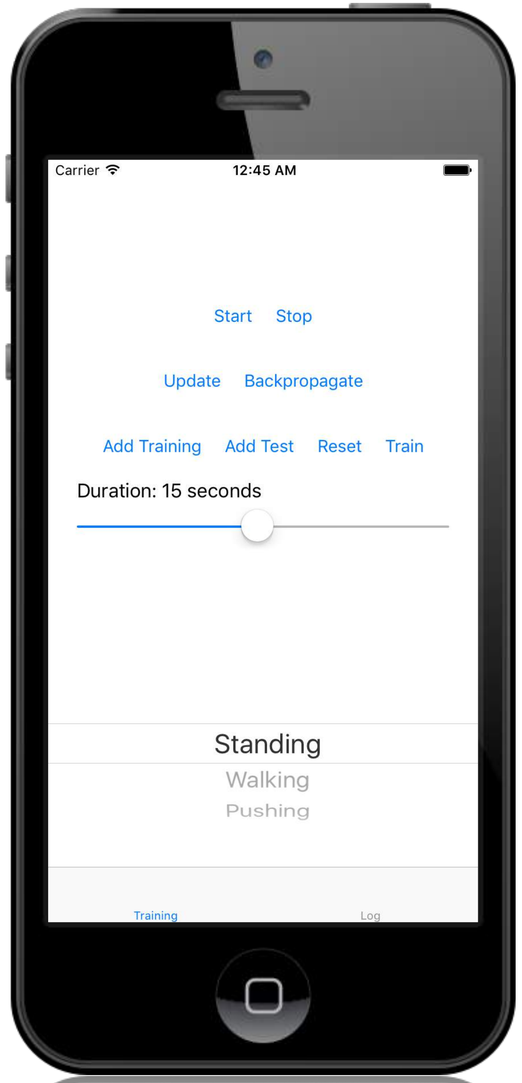
Input Layer

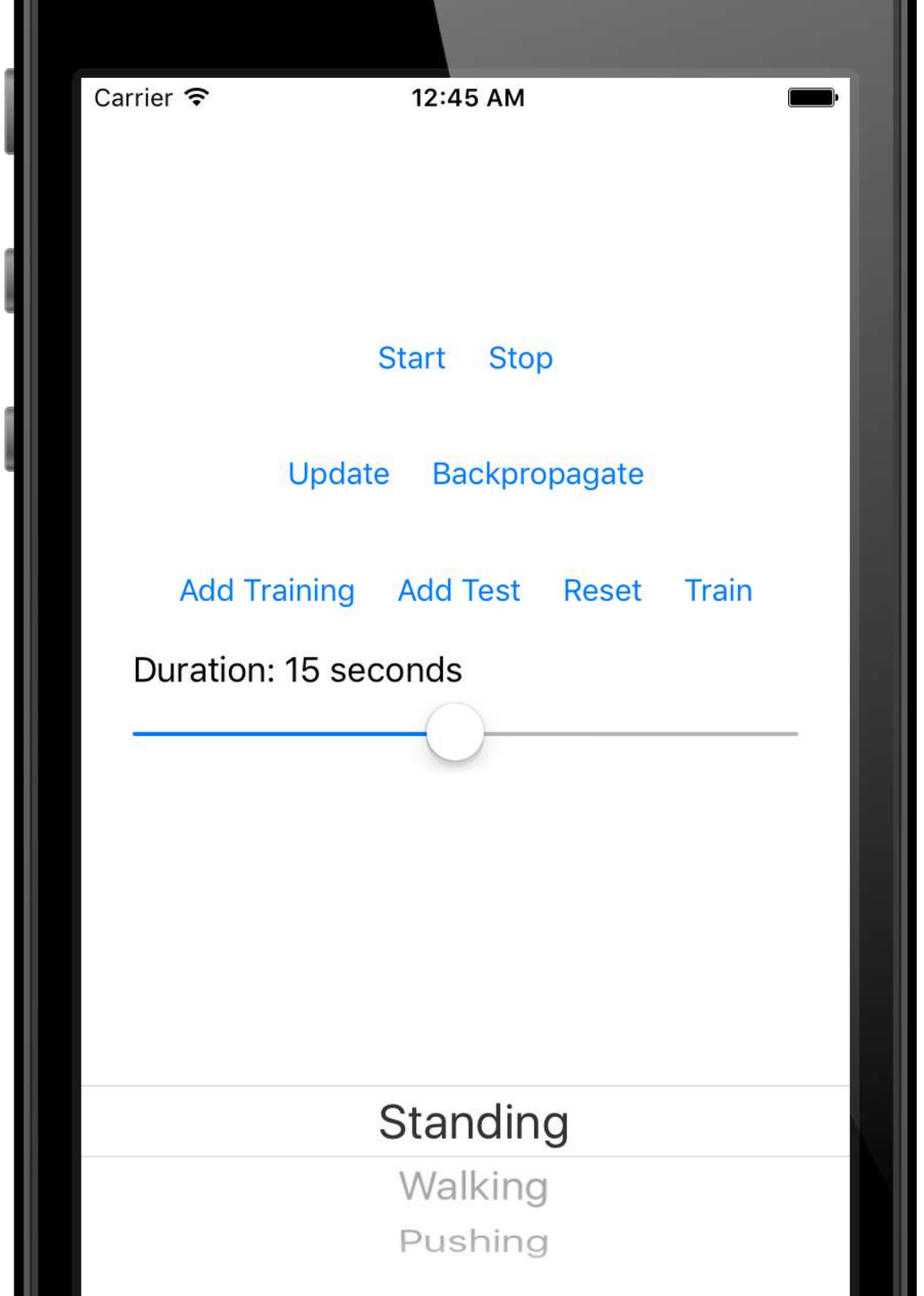
Hidden Layer

Output Layer

FFNN


Train
Update
Backpropagate
Weights
Inputs
Answer
Output
Error


Carrier
12:45 AM
Start
Stop
Update
Backpropagate
Add Training
Add Test
Reset
Train
Duration: 15 seconds
Standing
Walking
Pushing
Training
Log


Carrier
12:45 AM
Start
Stop
Update
Backpropagate
Add Training
Add Test
Reset
Train
Duration: 15 seconds
Standing
Walking
Pushing

Session

- 3 Sekunden Acceleration Recording
- 50 Updates pro Sekunde
- 450 Werte (x, y, z)

Inputs

- Keine Rohdaten verwenden
- Aussagekräftige Informationen liefern
- Unterschiede zwischen Bewegungen verdeutlichen
- Werte einzelner Achsen zusammenfassen

```
func inputs(for session: [CMAcceleration]) -> [Float]
  var inputs = [Float]()

  let accelerationsX = session.flatMap({ Float($0.x) })
  let accelerationsY = session.flatMap({ Float($0.y) })
  let accelerationsZ = session.flatMap({ Float($0.z) })

  inputs.append(average(of: accelerationsX))
  inputs.append(standardDeviation(of: accelerationsX))
  inputs.append(energy(of: accelerationsX))
  // Append y/z accelerations

  return inputs
}
```

```
var answer: [Float] {
  switch self {
  case .standing:
    return [1,0,0]
  case .walking:
    return [0,1,0]
  case .pushing:
    return [0,0,1]
  }
}
```

```
func train() -> [Float]? {
   let weights = try? network.train(
    inputs: trainingSessions.inputs,
    answers: trainingSessions.answers,
    testInputs: testSessions.inputs,
    testAnswers: testSessions.answers,
    errorThreshold: 0.3)

   return weights
}
```

```
func update(with session: [CMAcceleration]) -> [Float]? {
   let inputs = inputs(for: session)
   let output = try? network.update(inputs: inputs)

   return output
}
```

```
func backpropagate(with type: MotionType) -> Float? {
   let answer = type.answer
   let error = try? network.backpropagate(answer: answer)

   return error
}
```

Demo

Ergebnis ✓

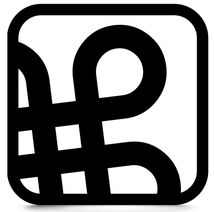
- Geringer Aufwand
- Schnelle Umsetzung
- Bewegungen werden präzise erkannt!

Und was jetzt?

- udacity.com - Videokurse
- http://ai.stanford.edu/~nilsson/MLBOOK.pdf - Introduction to ML
- http://scikit-learn.org - Python Library
- https://github.com/collinhundley/Swift-AI - FFNN für Swift

Fragen?

Vielen Dank



Macoun