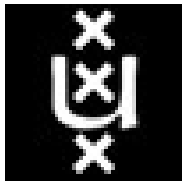


Applied Machine Learning
k-Nearest Neighbors Regression

BSc course Informatiekunde 2026
https://staff.fnwi.uva.nl/a.visser/education/AML

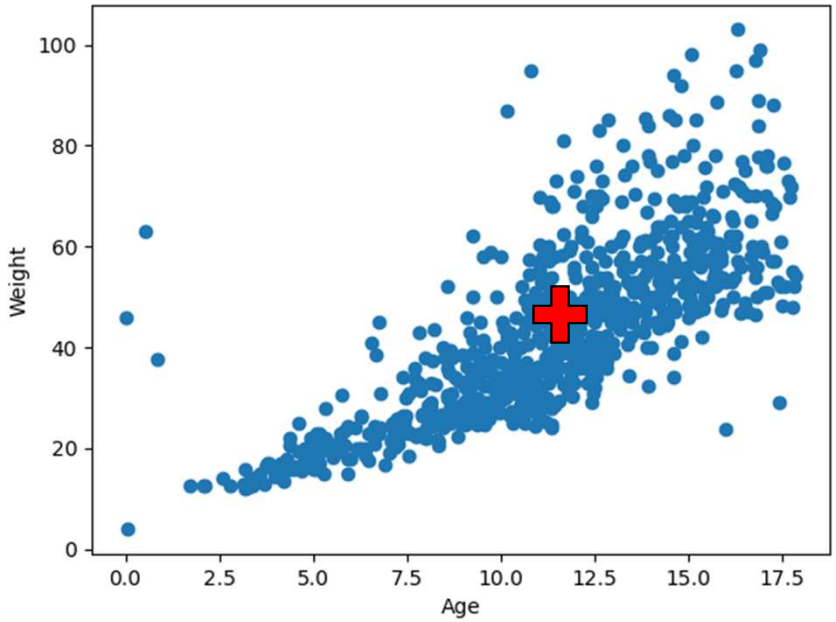
Arnoud Visser
Intelligent Robotics Lab & Computer Vision Lab
Informatics Institute
Universiteit van Amsterdam
A.Visser@uva.nl

Illustrations courtesy of Maarten Marx, Sarah Guido, Yolanda Hagar, and many others.

Simple model – version 0.1.1

$$weight = AverageWeight$$

regensburg_pediatric_appendicitis



Dataset

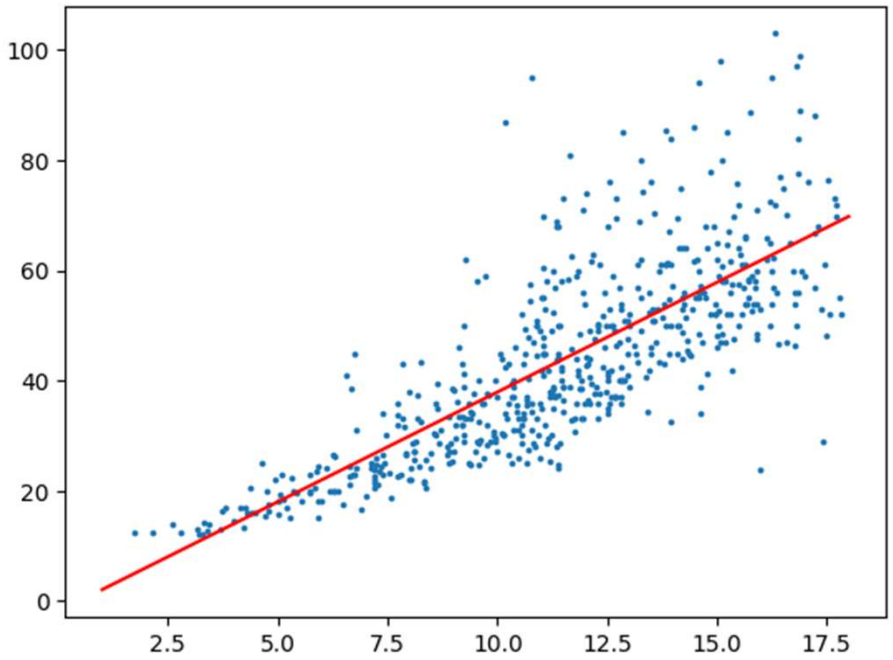
43.17 kg

- 775 datapoints → one value (mean())

Simple model – version 0.3

- So, $weight = BirthWeight + KiloPerYear * Age$

regensburg_pediatric_appendicitis



R2 score: 0.60 ± 0.06

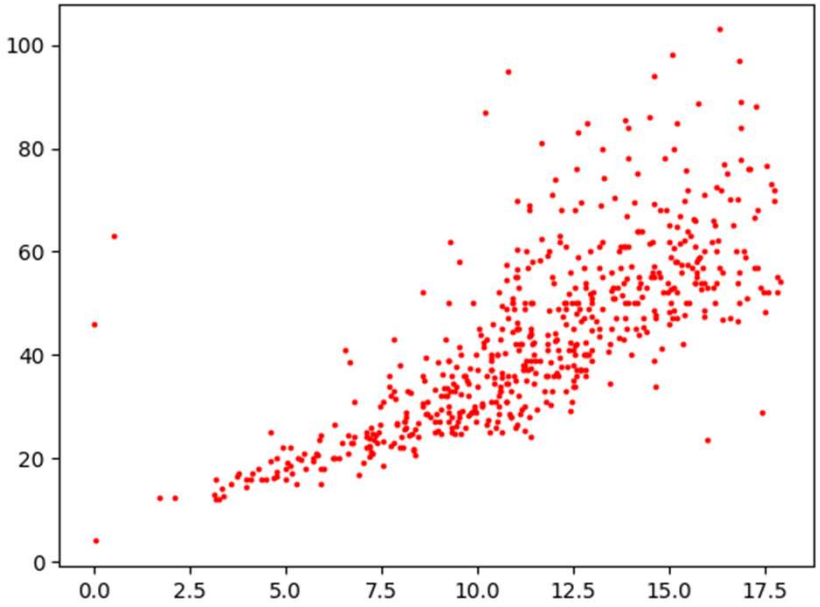
- 775 datapoints → two values (intercept , coeff)

Simple model – version 1-NN

- Let's try, $weight = neighbor\_weight()$

Model choice

regensburg_pediatric_appendicitis



- 775 datapoints → 775 values

Simple model – version 1-NN

$$weight = neighbor\_weight()$$

Model fit

regensburg_pediatric_appendicitis

```
from sklearn.neighbors import KNeighborsRegressor

nn1 = KNeighborsRegressor(n_neighbors=1)

nn1.fit(X_train, y_train)

print("Training score: {:.2f}".format(nn1.score(X_train, y_train)))
```

R2 score: 0.80

Simple model – version 1-NN

$$weight = neighbor\_weight()$$

Model check

regensburg_pediatric_appendicitis

```
from sklearn.neighbors import KNeighborsRegressor

nn1 = KNeighborsRegressor(n_neighbors=1)

nn1.fit(X_train, y_train)

print("Training score: {:.2f}".format(nn1.score(X_train, y_train)))
print("Test score: {:.2f}".format(nn1.score(X_test, y_test)))
```

Test R2 score: 0.39

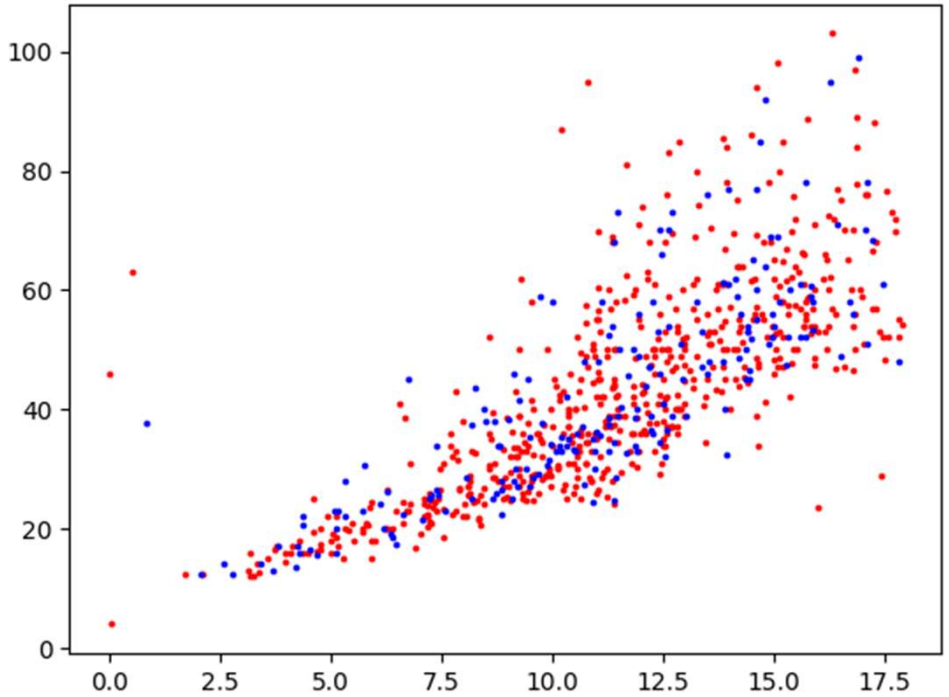
Cross-validated R2 score: 0.24 ± 0.17

Simple model – version 1-NN

$$weight = neighbor\_weight()$$

Model check

regensburg_pediatric_appendicitis



- 775 datapoints → 584 training values & 195 test values

Simple model – version 3-NN

$$weight = neighbor\_weight()$$

Hyperparameter choice

regensburg_pediatric_appendicitis

```
from sklearn.neighbors import KNeighborsRegressor

nn3 = KNeighborsRegressor(n_neighbors=3)

nn3.fit(X_train, y_train)

print("Training score: {:.2f}".format(nn3.score(X_train, y_train)))
```

R2 score: 0.73

Simple model – version 3-NN

$$weight = neighbor\_weight()$$

Model check

regensburg_pediatric_appendicitis

```
from sklearn.neighbors import KNeighborsRegressor

nn3 = KNeighborsRegressor(n_neighbors=3)

nn3.fit(X_train, y_train)

print("Training score: {:.2f}".format(nn3.score(X_train, y_train)))
print("Test score: {:.2f}".format(nn3.score(X_test, y_test)))
```

Test R2 score: 0.56

Cross-validated R2 score: 0.49 ± 0.08

Simple model – version 5-NN

$$weight = neighbor\_weight()$$

Hyperparameter choice

regensburg_pediatric_appendicitis

```
from sklearn.neighbors import KNeighborsRegressor

nn5 = KNeighborsRegressor(n_neighbors=5)

nn5.fit(X_train, y_train)

print("Training score: {:.2f}".format(nn5.score(X_train, y_train)))
```

R2 score: 0.69

Simple model – version 5-NN

$$weight = neighbor\_weight()$$

Model check

regensburg_pediatric_appendicitis

```
from sklearn.neighbors import KNeighborsRegressor

nn5 = KNeighborsRegressor(n_neighbors=5)

nn5.fit(X_train, y_train)

print("Training score: {:.2f}".format(nn5.score(X_train, y_train)))
print("Test score: {:.2f}".format(nn5.score(X_test, y_test)))
```

Test R2 score: 0.60

Cross-validated R2 score: 0.54 ± 0.05

Simple model – version 7-NN

$$weight = neighbor\_weight()$$

Hyperparameter choice

regensburg_pediatric_appendicitis

```
from sklearn.neighbors import KNeighborsRegressor

nn7 = KNeighborsRegressor(n_neighbors=7)

nn7.fit(X_train, y_train)

print("Training score: {:.2f}".format(nn7.score(X_train, y_train)))
```

R2 score: 0.66

Simple model – version 7-NN

$$weight = neighbor\_weight()$$

Model check

regensburg_pediatric_appendicitis

```
from sklearn.neighbors import KNeighborsRegressor

nn7 = KNeighborsRegressor(n_neighbors=7)

nn7.fit(X_train, y_train)

print("Training score: {:.2f}".format(nn7.score(X_train, y_train)))
print("Test score: {:.2f}".format(nn7.score(X_test, y_test)))
```

Test R2 score: 0.62

Cross-validated R2 score: 0.55 ± 0.06

Simple model – version 9-NN

$$weight = neighbor\_weight()$$

Hyperparameter choice

regensburg_pediatric_appendicitis

```
from sklearn.neighbors import KNeighborsRegressor

nn9 = KNeighborsRegressor(n_neighbors=9)

nn9.fit(X_train, y_train)

print("Training score: {:.2f}".format(nn9.score(X_train, y_train)))
```

R2 score: 0.64

Simple model – version 9-NN

$$weight = neighbor\_weight()$$

Model check

regensburg_pediatric_appendicitis

```
from sklearn.neighbors import KNeighborsRegressor

nn9 = KNeighborsRegressor(n_neighbors=9)

nn9.fit(X_train, y_train)

print("Training score: {:.2f}".format(nn9.score(X_train, y_train)))
print("Test score: {:.2f}".format(nn7.score(X_test, y_test)))
```

Test R2 score: 0.63

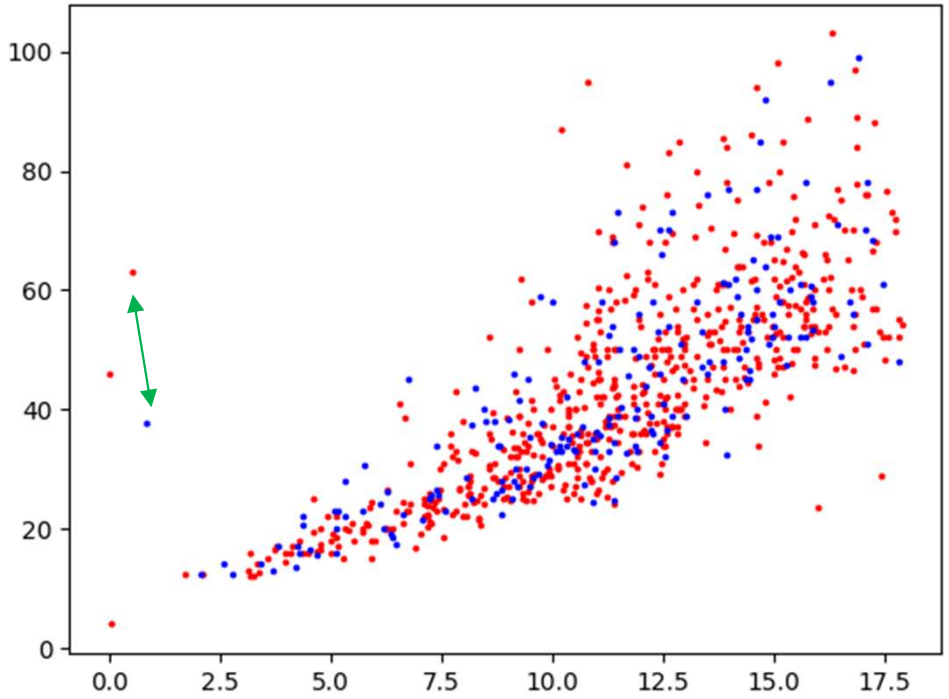
Cross-validated R2 score: 0.57 ± 0.06

Simple model – version 1-NN

$$weight = neighbor\_weight(age)$$

Model working

regensburg_pediatric_appendicitis



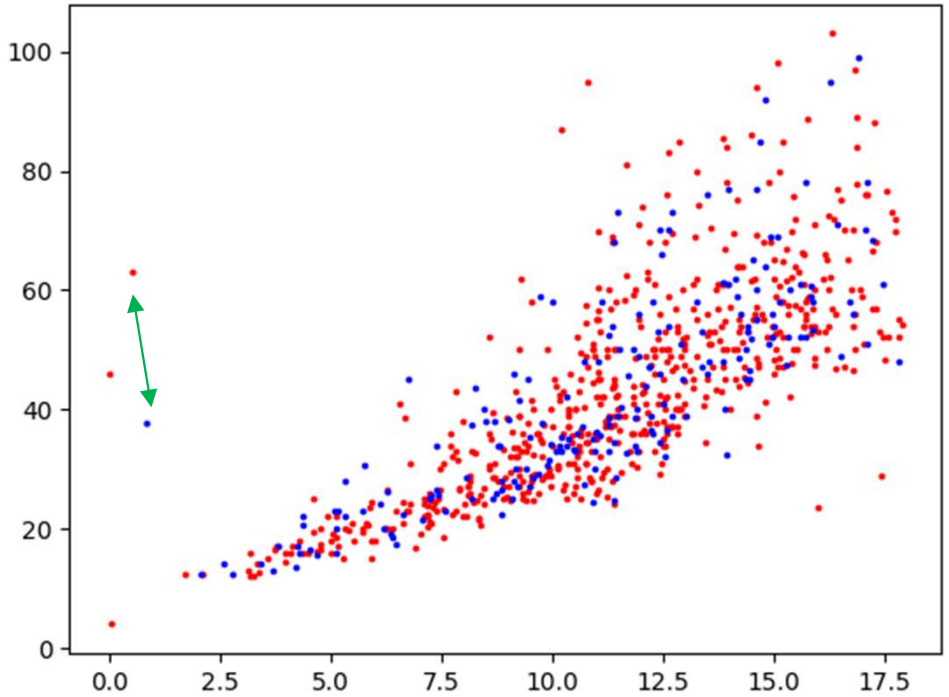
1 test value → searching for 1 training value

Simple model – version 1-NN

$$weight = neighbor\_weight(age)$$

Model working

regensburg_pediatric_appendicitis



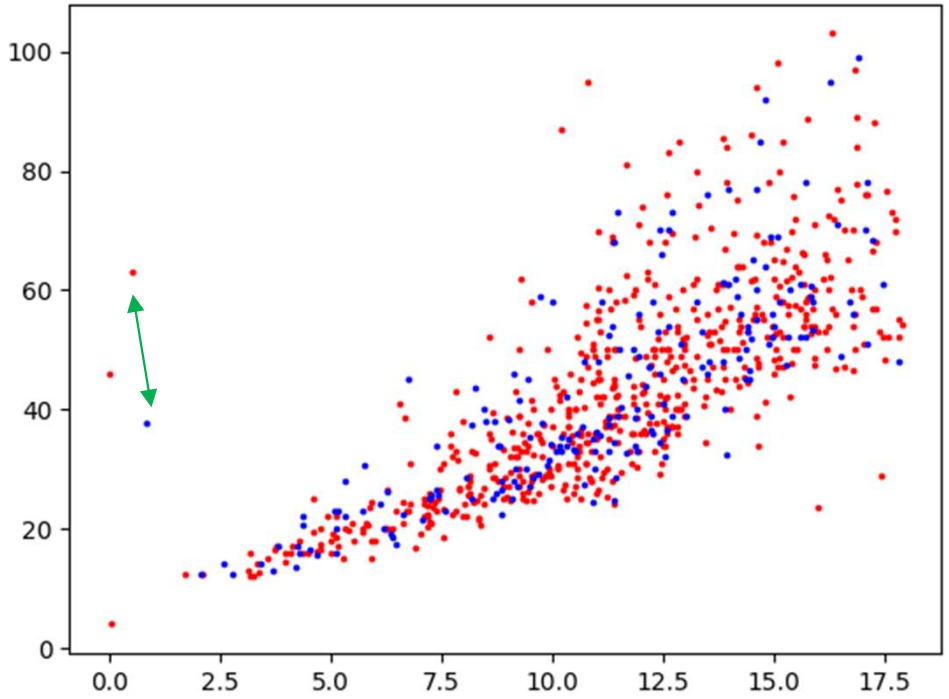
test age 0.85y → nearest training age 0.53y

Simple model – version 1-NN

$$weight = neighbor\_weight(age)$$

Model working

regensburg_pediatric_appendicitis



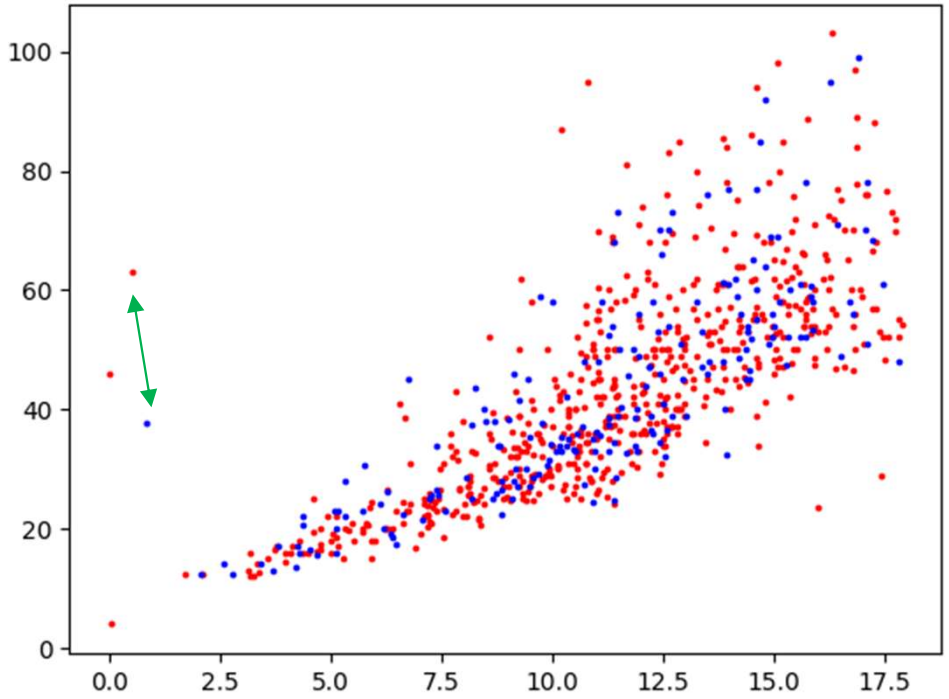
training age 0.53y → 63.1kg → test age 0.85y → 63.1kg

Simple model – version 1-NN

$$weight = neighbor\_weight(age)$$

Model working

regensburg_pediatric_appendicitis



ε → 25.2kg

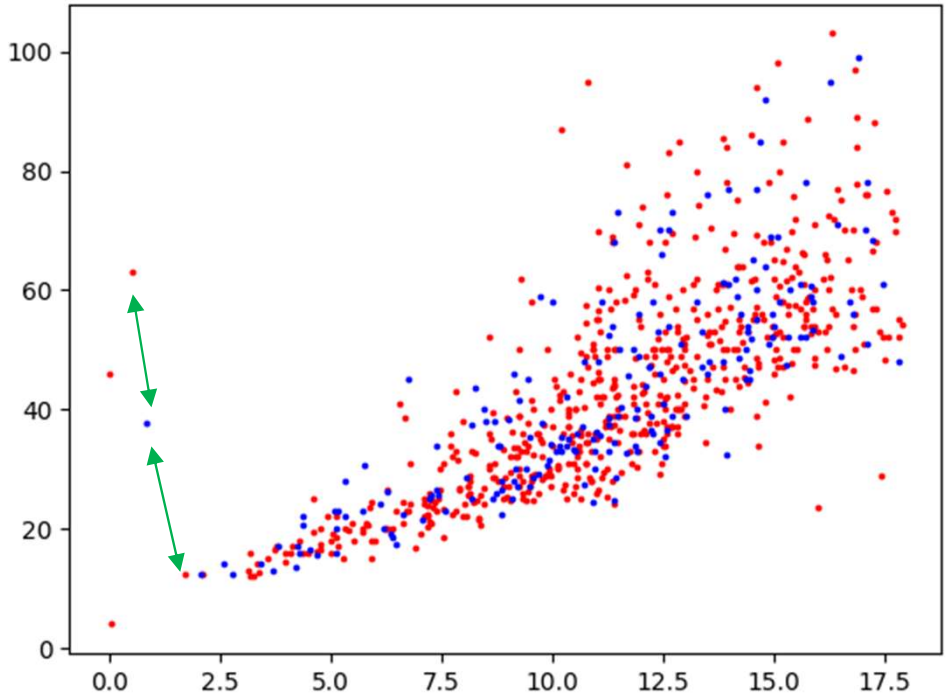
training age 0.53y → 63.1kg → test age 0.85y → 37.8kg

Simple model – version 2-NN

$$weight = neighbor\_weight(age)$$

Model working

regensburg_pediatric_appendicitis

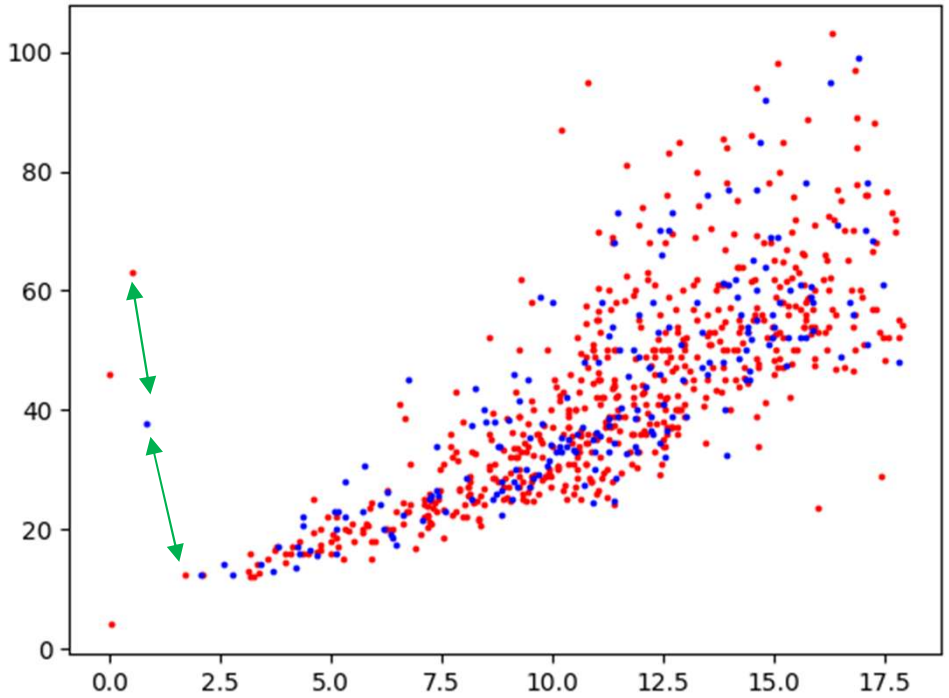


test age 0.85y → nearest training age 0.53y & 1.73y

Simple model – version 2-NN

$$weight = neighbor\_weight(age)$$

Model working



training age 0.53y → 63.1kg

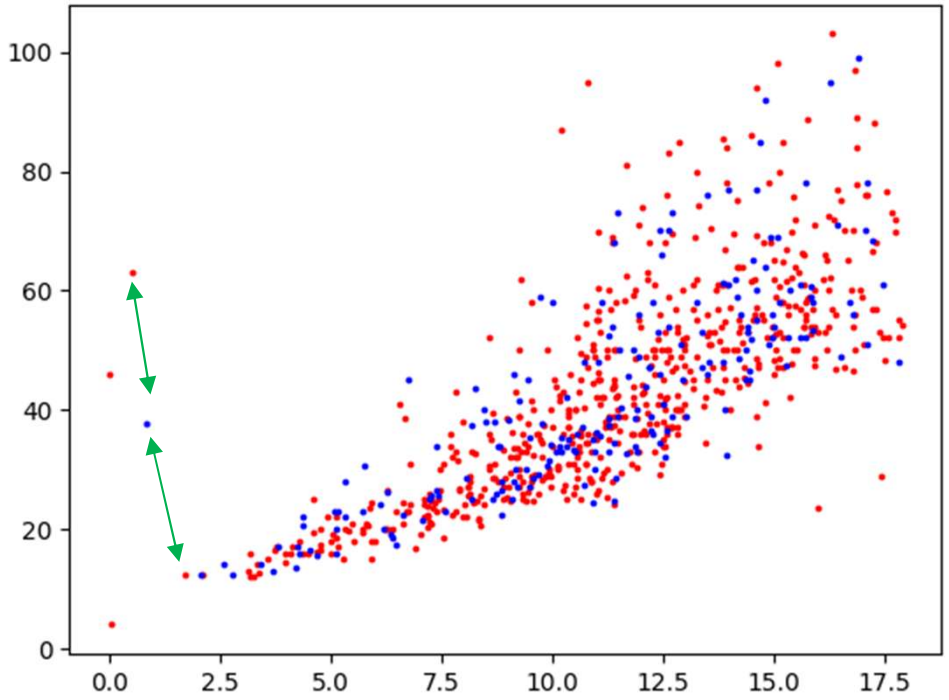
→ test age 0.85y → (63.1+12.5)/2 kg

training age 1.73y → 12.5kg

Simple model – version 2-NN

$$weight = neighbor\_weight(age)$$

Model working



training age 0.53y → 63.1kg

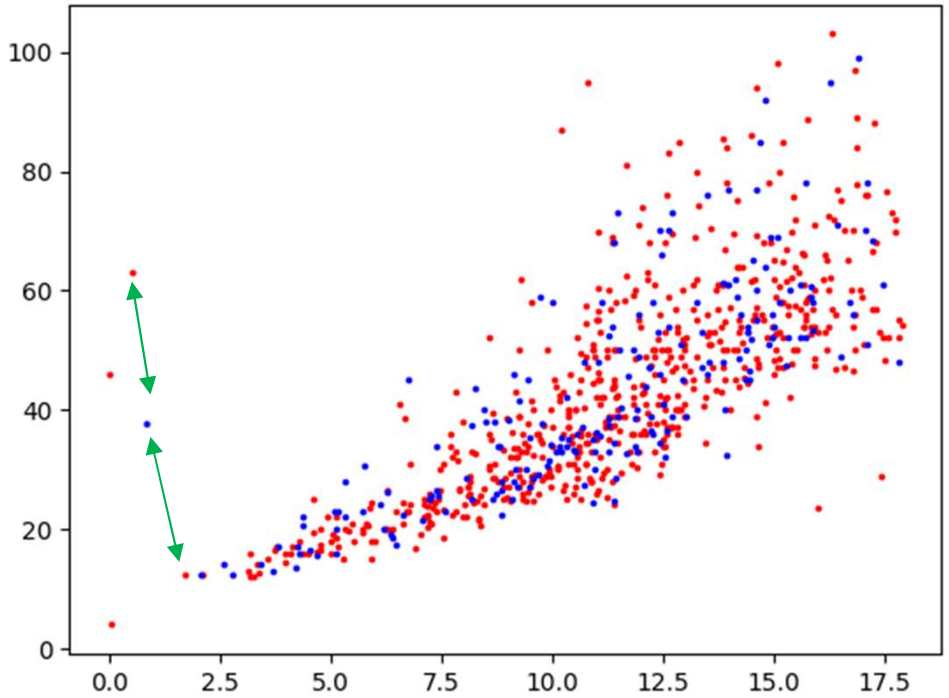
→ test age 0.85y → 37.8kg

training age 1.73y → 12.5kg

Simple model – version 2-NN

$$weight = neighbor\_weight(age)$$

Model working



training age 0.53y → 63.1kg

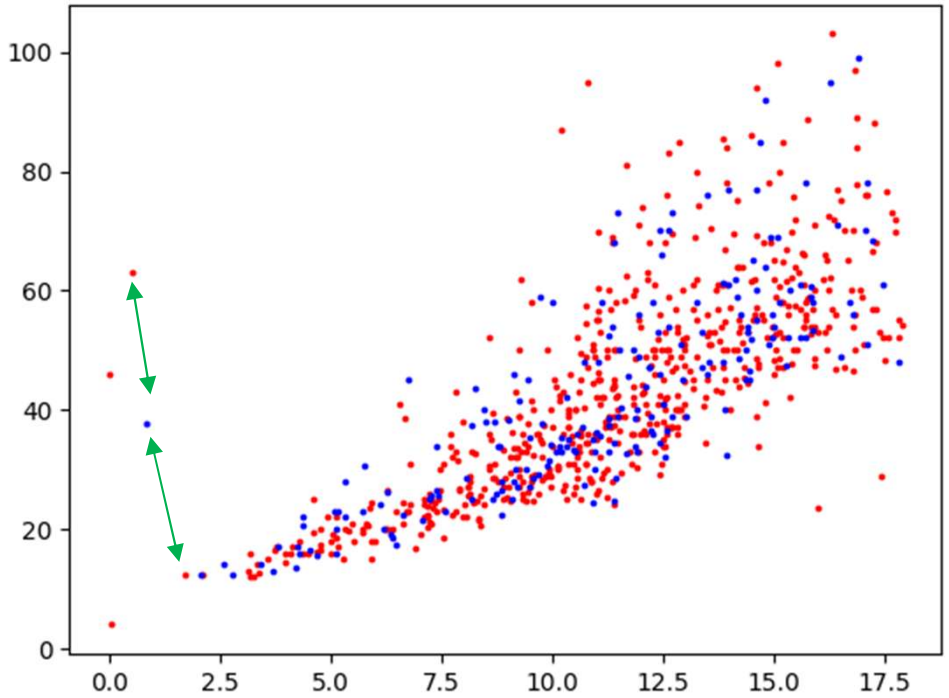
→ test age 0.85y → 37.8kg

training age 1.73y → 12.5kg

Simple model – version 2-NN

$$weight = neighbor\_weight(age)$$

Model working



training age 0.53y → 63.1kg

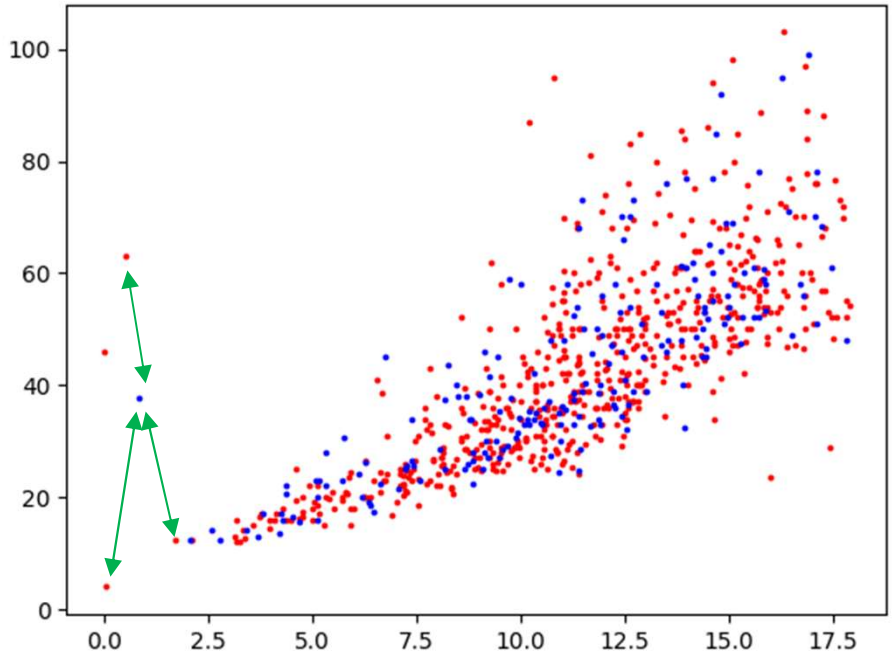
→ test age 0.85y → 37.8kg ε → 0.0kg

training age 1.73y → 12.5kg

Simple model – version 3-NN

$$weight = neighbor\_weight(age)$$

Model working



training age 0.53y → 63.1kg

training age 1.73y → 12.5kg

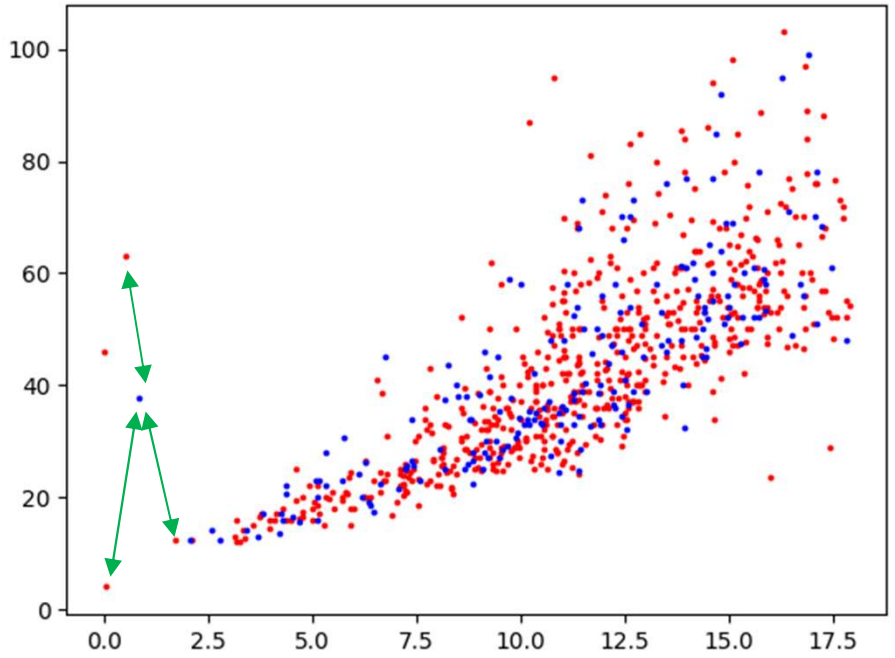
training age 0.04y → 3.96kg

→ test age 0.85y → (63.1 + 12.5 + 3.96) / 3 kg

Simple model – version 3-NN

$$weight = neighbor\_weight(age)$$

Model working



training age 0.53y → 63.1kg

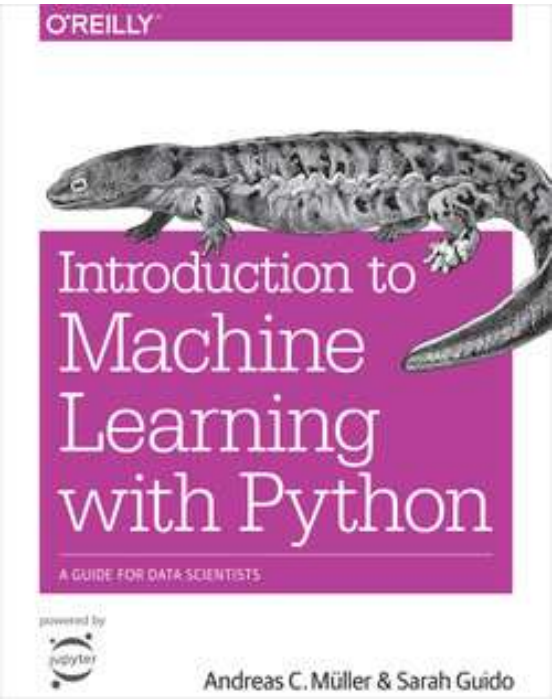
training age 1.73y → 12.5kg

training age 0.04y → 3.96kg

→ test age 0.85y → 26.5 kg ε → -11.3kg

Introduction to Machine Learning

- 2.3.2 k-neighbors regression

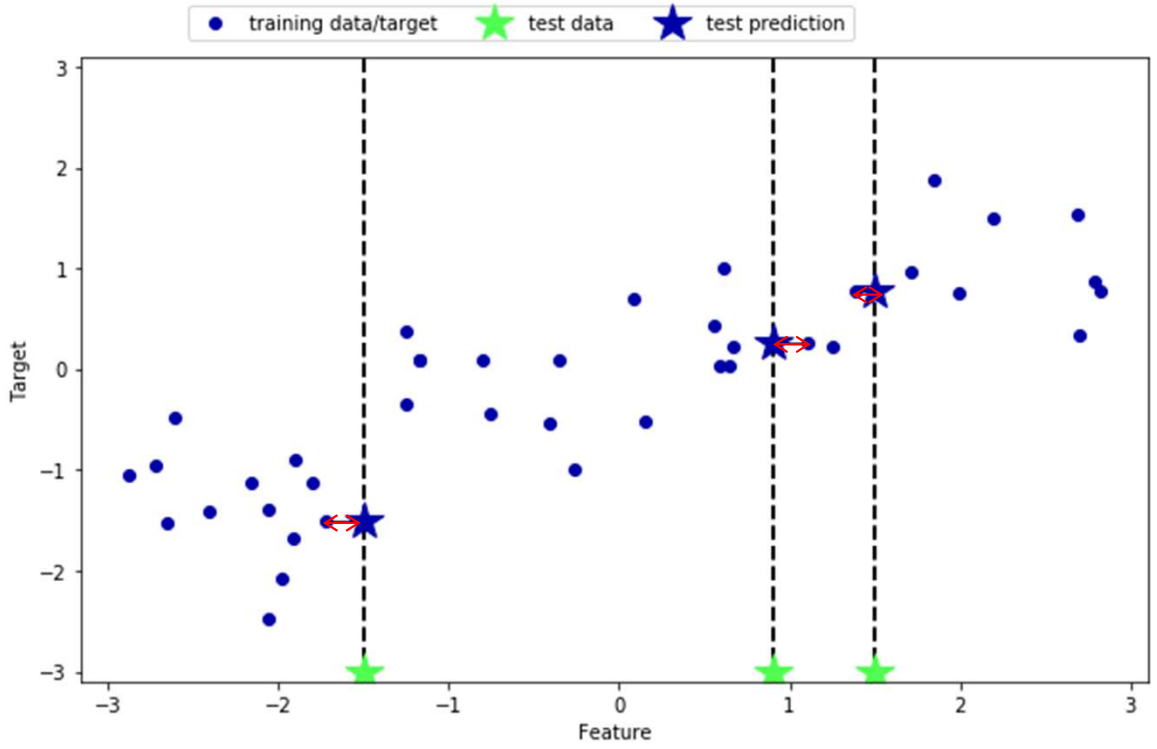


Andreas C. Müller, Sarah Guido, Introduction to Machine Learning with Python, O'Reilly Media, October 2016

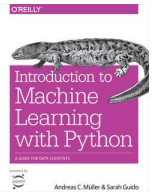
Page 42-46

Introduction to Machine Learning

- 1-neighbor example



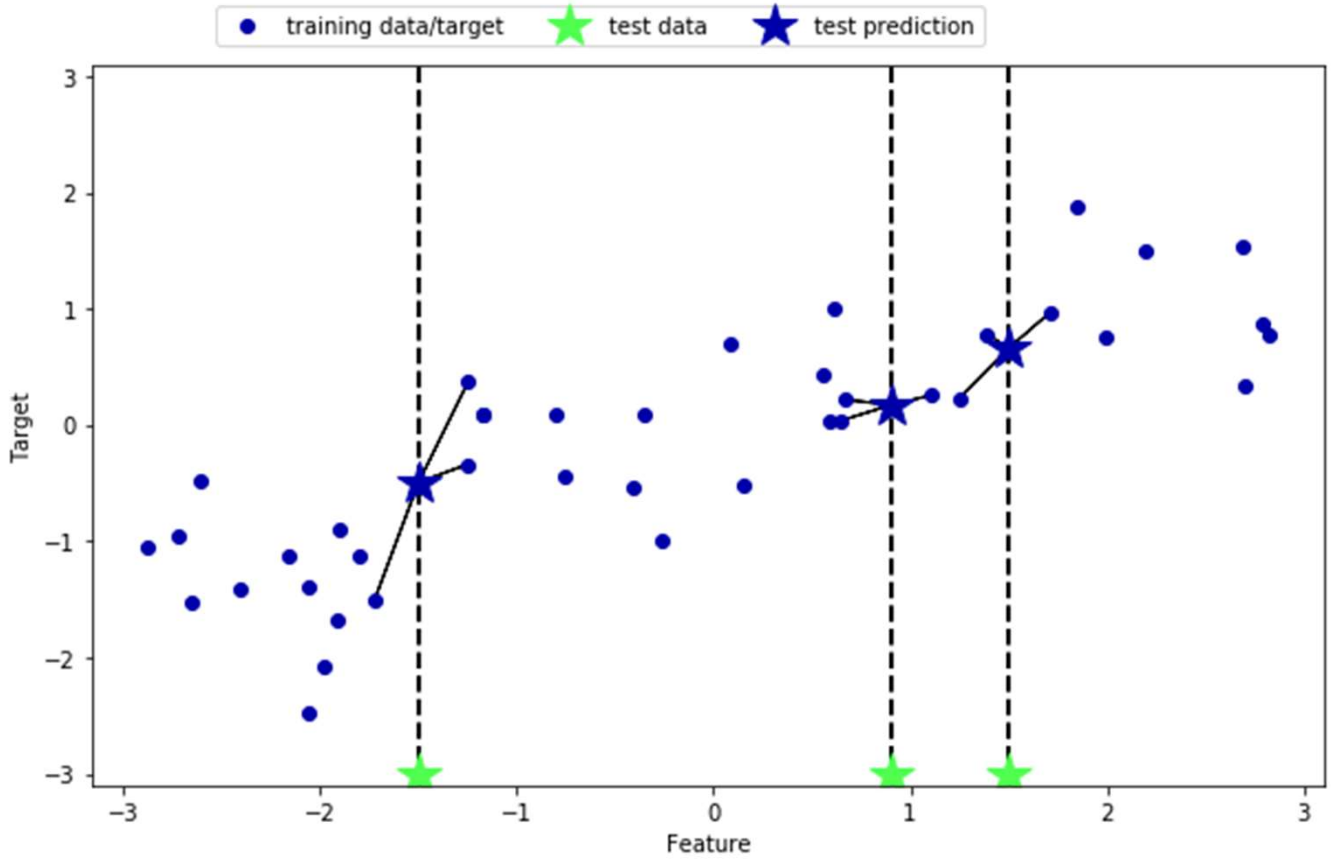
Andreas C. Müller, Sarah Guido, Introduction to Machine Learning with Python, O'Reilly Media, October 2016



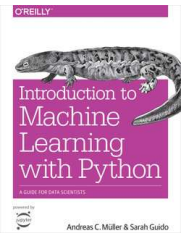
Page 42-46

Introduction to Machine Learning

- 3-neighbor example



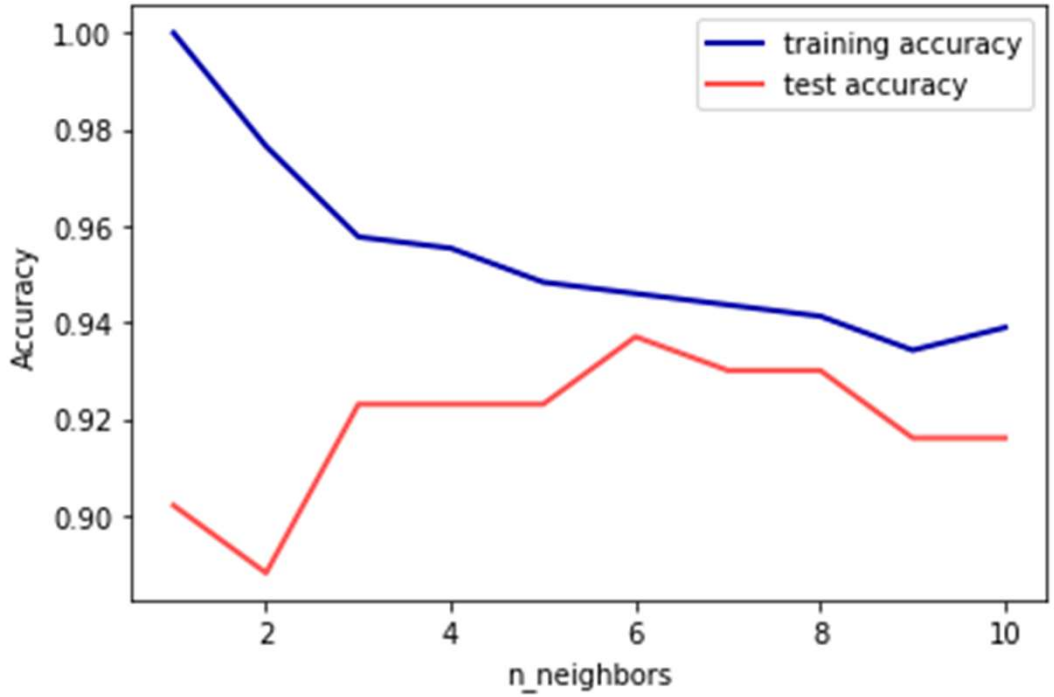
Andreas C. Müller, Sarah Guido, Introduction to Machine Learning with Python, O'Reilly Media, October 2016

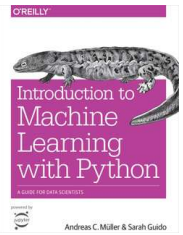


Page 42-46

Introduction to Machine Learning

- Hyperparameter tuning



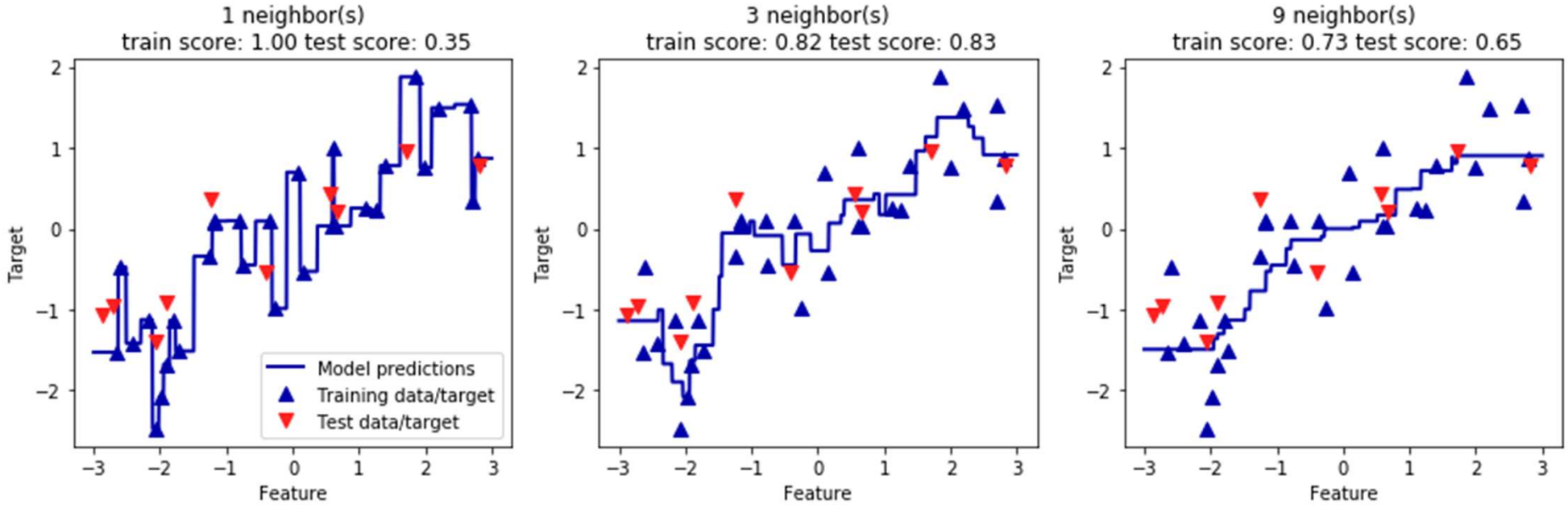


Andreas C. Müller, Sarah Guido, Introduction to Machine Learning with Python, O'Reilly Media, October 2016

Page 42-46

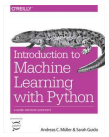
Introduction to Machine Learning

- Fitting a ‘line’



30 training points → averaging over 9 neighbors
→ three independent values for a line

Andreas C. Müller, Sarah Guido, Introduction to Machine Learning with Python, O’Reilly Media, October 2016

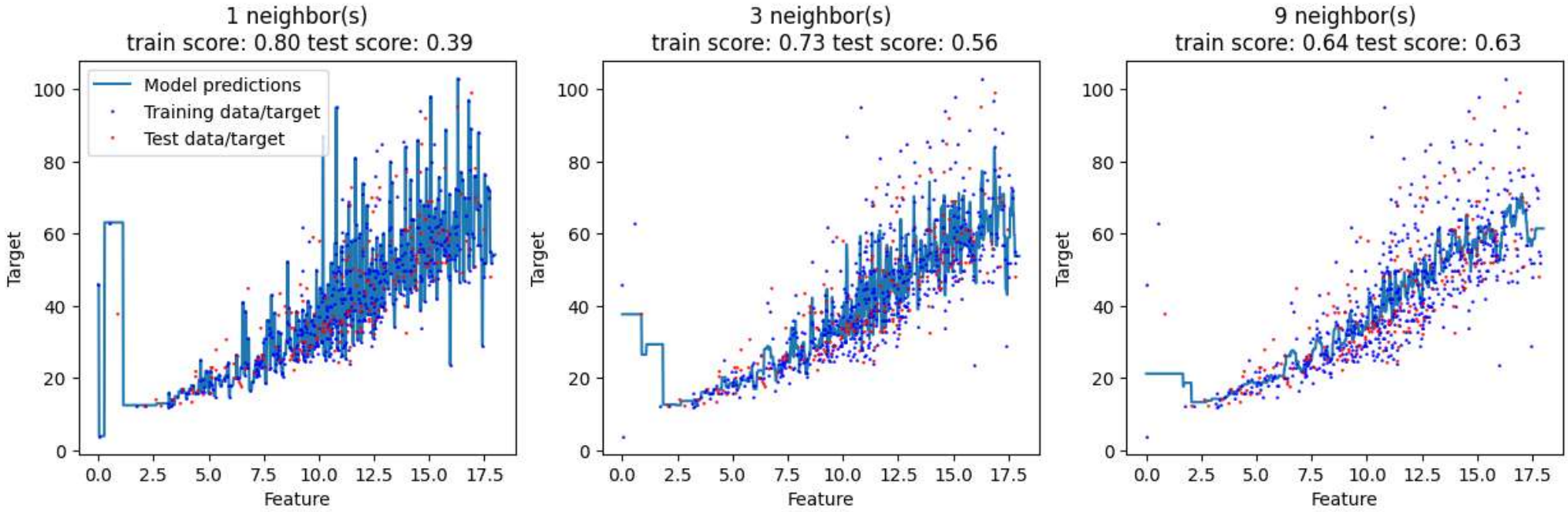


Page 42-46

‘Machine Learning’

regensburg_pediatric_appendicitis

- Fitting a ‘line’

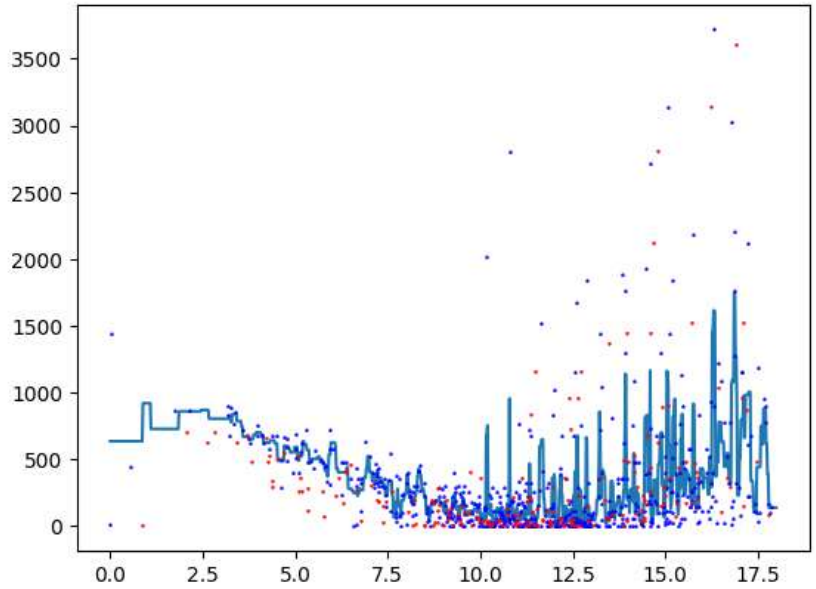


585 training points → averaging over 9 neighbors

→ 65 independent values for a line

K-Neighbors Regression

- The data is the model

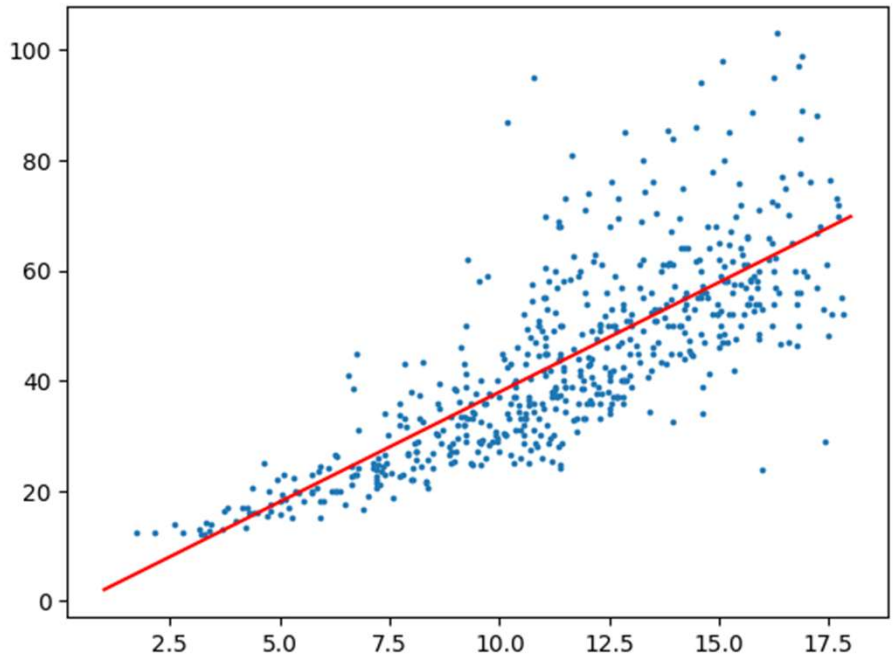


- Any function-shape can be learned
- The algorithm is non-parametric →
  no assumption on the shape of the function $y = f(X)$

Linear model

- So, $weight = BirthWeight + KiloPerYear * Age$

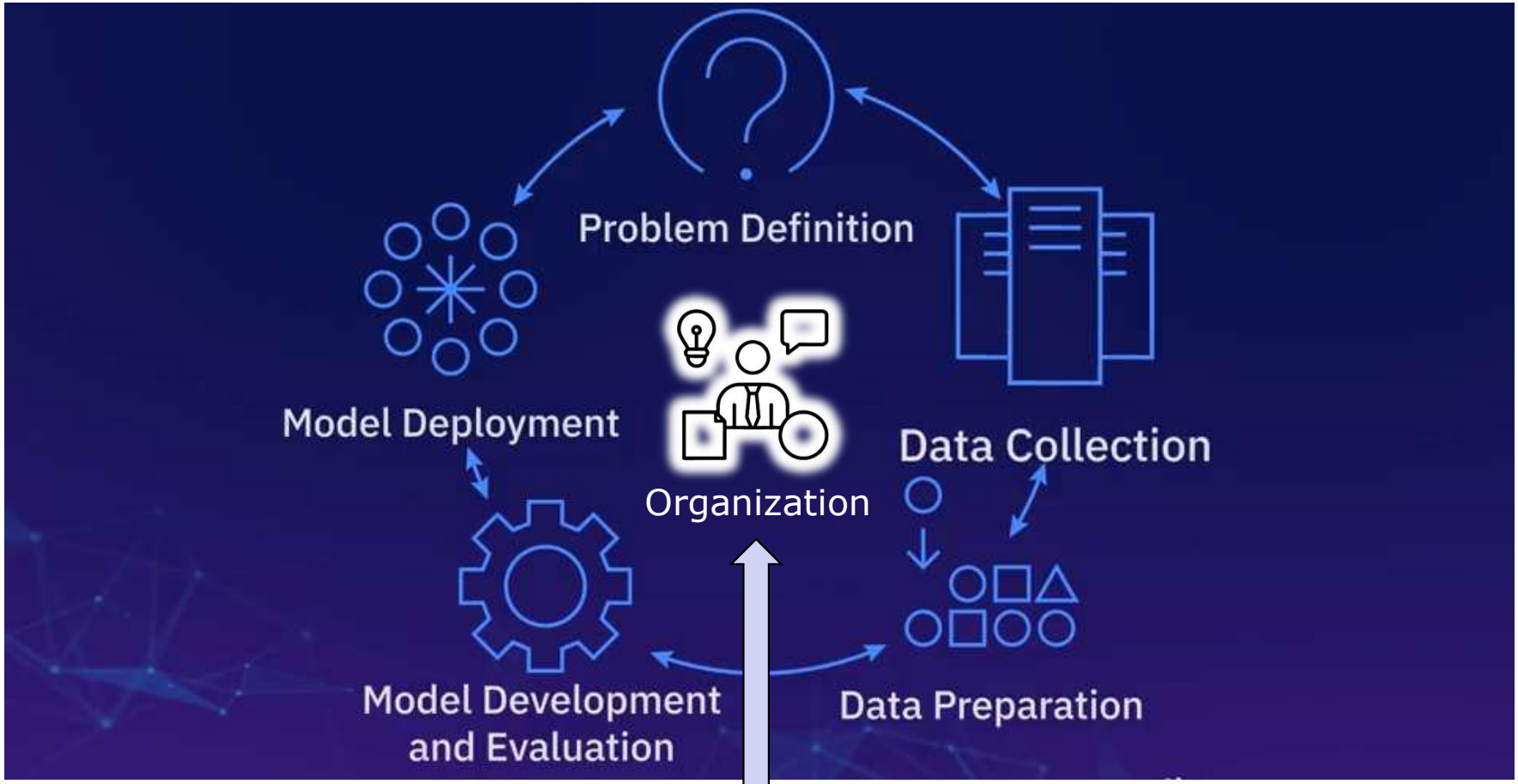
regensburg_pediatric_appendicitis



- There is 1-1 relationship between age ⇔ weight

Introduction to Machine Learning

Building a predictive model is a circular process.

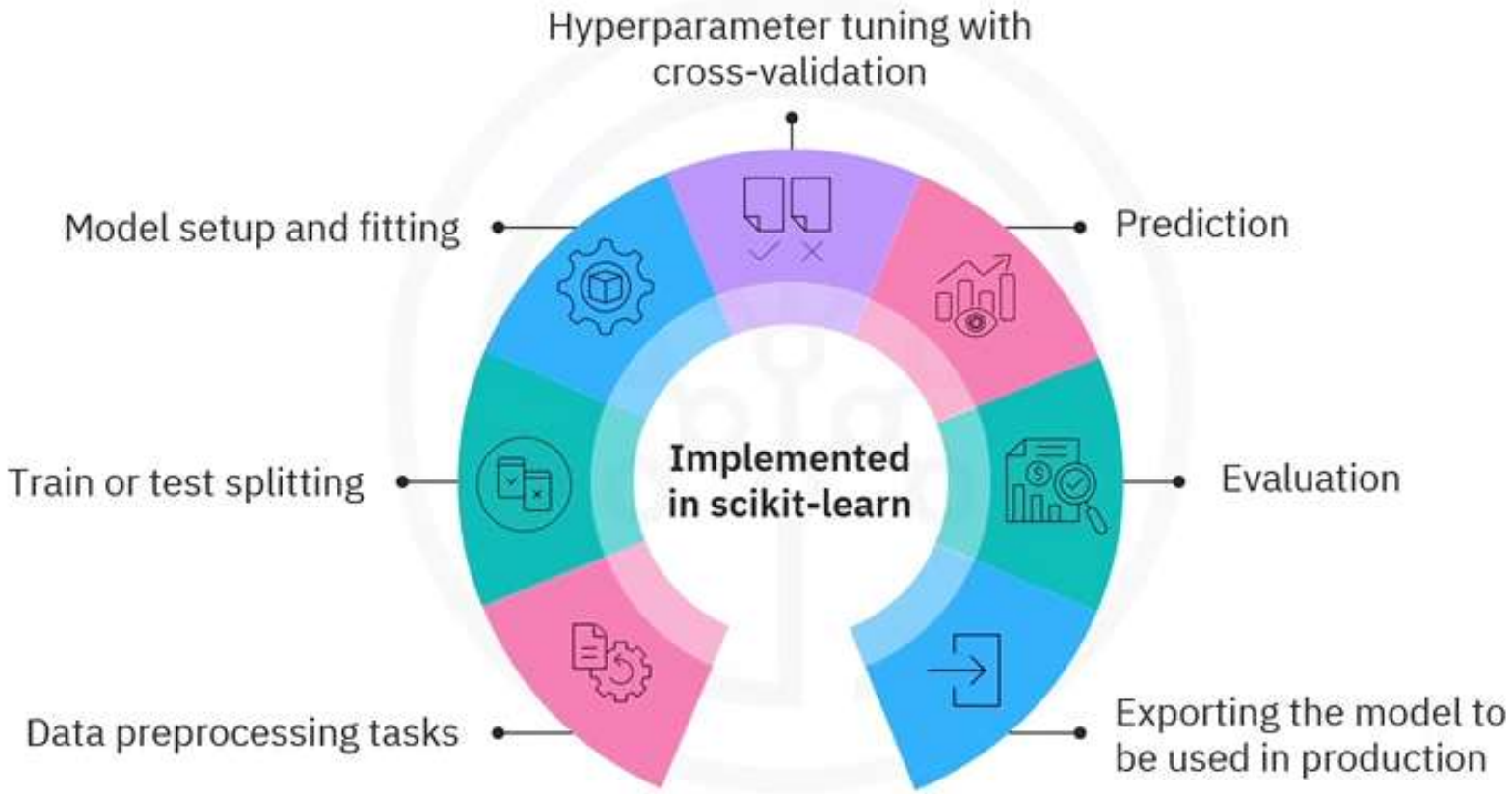


Expertise

- Depends on the organization
- Interpret the information in the right way

Introduction to Machine Learning

Building a predictive model is a circular process.

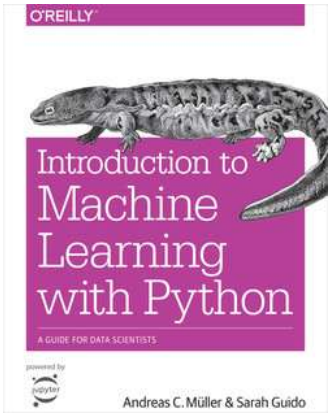


Joseph Santarcangelo & Joss Grossman, Machine Learning with Python, IBM Skills network, 2025

Introduction to Machine Learning

- 2.3.2 k-neighbors regression

- Strengths:
  - Easy to understand
  - Reasonable performance without many hyperparameters
- Weakness:
  - Prediction becomes slow for large training-sets
  - Doesn't scale for many features
  - Performs badly on sparse data



Andreas C. Müller, Sarah Guido, Introduction to Machine Learning with Python, O'Reilly Media, October 2016

Page 46

Conclusion

Learning outcomes of this course covered today

- What is a good fit of all datapoints of training set are used as ‘lookup table’