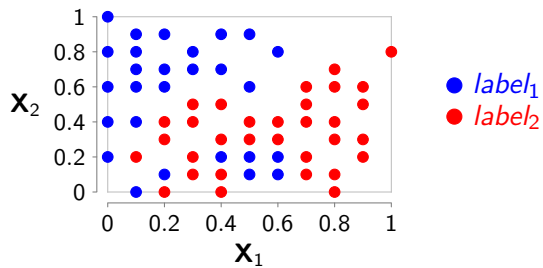


libconform v0.1.0: a Python library for conformal prediction

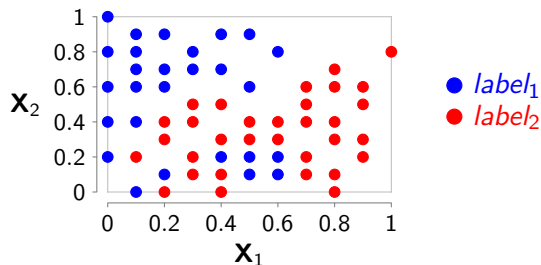
Jonas Faßbender

Conformal prediction

Conformal prediction

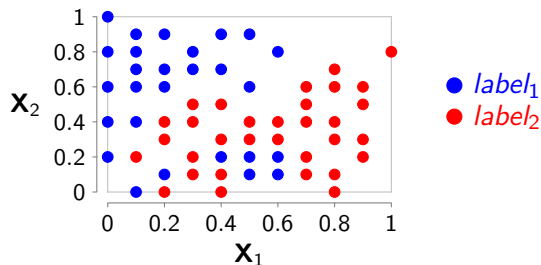


Conformal prediction



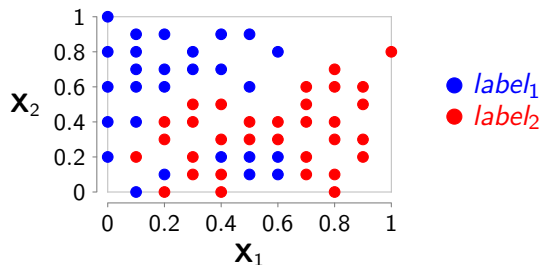
- Feature space $\mathbf{X} := \mathbb{R}^2$

Conformal prediction



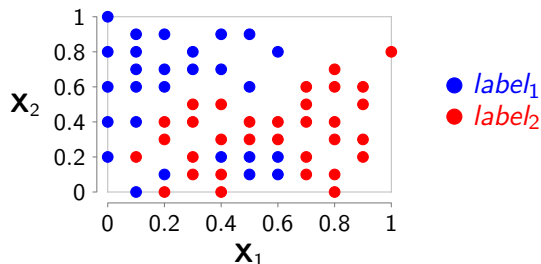
- ▶ Feature space $\mathbf{X} := \mathbb{R}^2$
- ▶ Label space $\mathbf{Y} := \{label_1, label_2\}$

Conformal prediction



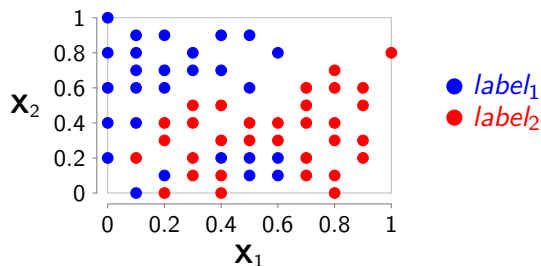
- ▶ Feature space $\mathbf{X} := \mathbb{R}^2$
- ▶ Label space $\mathbf{Y} := \{label_1, label_2\}$
- ▶ Example space $\mathbf{Z} := \mathbf{X} \times \mathbf{Y}$

Conformal prediction



- ▶ Feature space $\mathbf{X} := \mathbb{R}^2$
- ▶ Label space $\mathbf{Y} := \{label_1, label_2\}$
- ▶ Example space $\mathbf{Z} := \mathbf{X} \times \mathbf{Y}$
- ▶ Example $z_i := (x_i, y_i); x_i \in \mathbf{X}, y_i \in \mathbf{Y}$

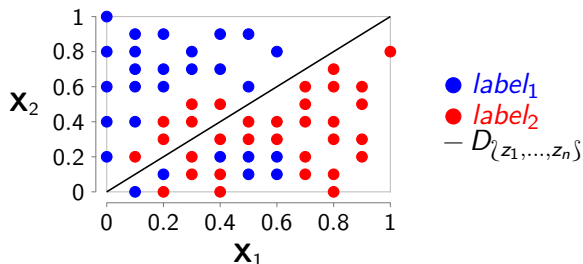
Conformal prediction



- ▶ Feature space $\mathbf{X} := \mathbb{R}^2$
- ▶ Label space $\mathbf{Y} := \{label_1, label_2\}$
- ▶ Example space $\mathbf{Z} := \mathbf{X} \times \mathbf{Y}$
- ▶ Example $z_i := (x_i, y_i); x_i \in \mathbf{X}, y_i \in \mathbf{Y}$
- ▶ Datensatz $\{z_1, \dots, z_n\}$

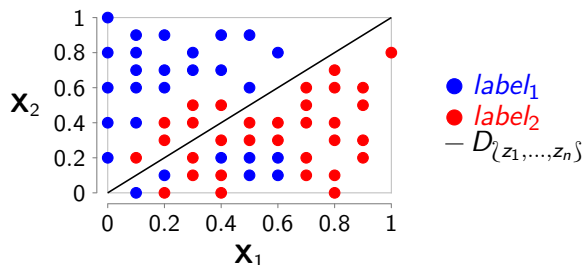
Conformal prediction

Conformal prediction



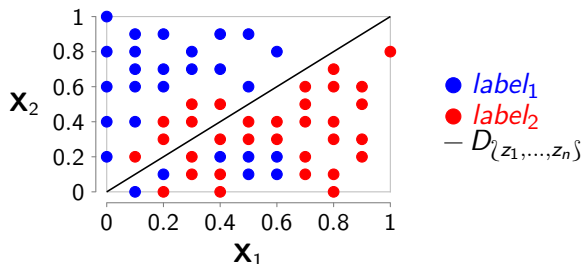
- Klassischer Machine Learning Predictor $D_{\{z_1, \dots, z_n\}}$

Conformal prediction



- ▶ Klassischer Machine Learning Predictor $D_{(z_1, \dots, z_n)}$
- ▶ Bare predictions, kein confidence Wert in prediction

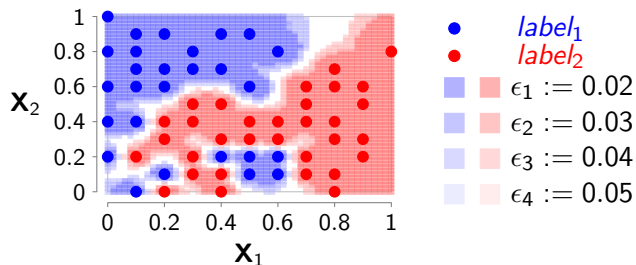
Conformal prediction



- ▶ Klassischer Machine Learning Predictor $D_{\{z_1, \dots, z_n\}}$
- ▶ Bare predictions, kein confidence Wert in prediction
- ▶ Kann zu sog. nonconformity measure umgewandelt werden (Basis von CP)

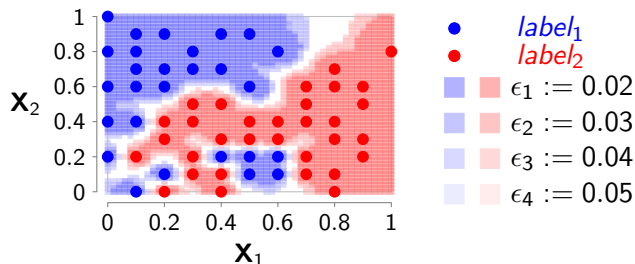
Conformal prediction

Conformal prediction



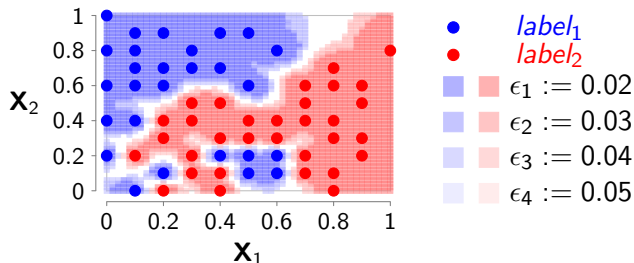
► Conformal Predictor $\Gamma_{\{z_1, \dots, z_n\}}^\epsilon$

Conformal prediction



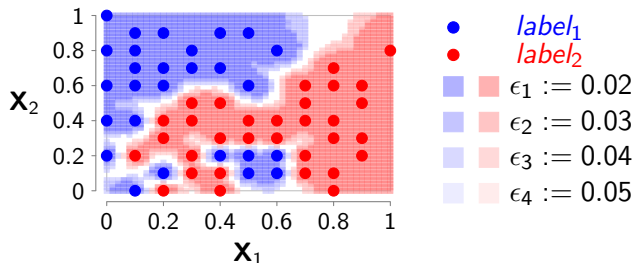
- ▶ Conformal Predictor $\Gamma_{\{z_1, \dots, z_n\}}^\epsilon$
- ▶ Wichtigste Eigenschaft: validity under exchangeability

Conformal prediction



- ▶ Conformal Predictor $\Gamma_{\{z_1, \dots, z_n\}}^\epsilon$
- ▶ Wichtigste Eigenschaft: validity under exchangeability
- ▶ $\Gamma_{\{z_1, \dots, z_n\}}^\epsilon$ hat Genauigkeit von mindestens $1 - \epsilon$ (wenn z_1, \dots, z_n exchangeable)

Conformal prediction



- ▶ Conformal Predictor $\Gamma_{\{z_1, \dots, z_n\}}^\epsilon$
- ▶ Wichtigste Eigenschaft: validity under exchangeability
- ▶ $\Gamma_{\{z_1, \dots, z_n\}}^\epsilon$ hat Genauigkeit von mindestens $1 - \epsilon$ (wenn z_1, \dots, z_n exchangeable)
- ▶ In Realität: wahre exchangeability selten, aber meistens nah genug dran

libconform

libconform

```
import numpy as np
from libconform import CP
from libconform.ncs import
    NCSKNearestNeighbors

from sklearn.datasets import load_iris

X, y = load_iris(True)

# randomly permute X, y
indices = np.arange(len(X))
np.random.shuffle(indices)

X, y = X[indices], y[indices]

# split in train and test data set
X_train, y_train = X[:-20], y[:-20]
X_test, y_test = X[-20:], y[-20:]

ncs = NCSKNearestNeighbors(n_neighbors=1)
epsilons = [0.01, 0.02, 0.03, 0.04, 0.05]
cp = CP(ncs, epsilons)

cp.train(X_train, y_train)

res = cp.score(X_test, y_test)
print(res)
```

libconform

- ▶ Python: lingua franca für Machine Learning

libconform

- ▶ Python: lingua franca für Machine Learning
- ▶ MIT-licensed

libconform

- ▶ Python: lingua franca für Machine Learning
- ▶ MIT-licensed
- ▶ **Unstable**

libconform

- ▶ Python: lingua franca für Machine Learning
- ▶ MIT-licensed
- ▶ **Unstable**
- ▶ Fokus: extensibility

libconform

- ▶ Python: lingua franca für Machine Learning
- ▶ MIT-licensed
- ▶ **Unstable**
- ▶ Fokus: extensibility
- ▶ Grundlegendsten Algorithmen der CP-Familie implementiert (CP, smoothed CP, inductive CP, mondrian CP, RRCM, Venn prediction,...)

libconform – TODO

libconform – TODO

- ▶ Test-dichte zu gering

libconform – TODO

- ▶ Test-dichte zu gering
- ▶ Dokumentation

libconform – TODO

- ▶ Test-dichte zu gering
- ▶ Dokumentation
- ▶ Unausgereifte interne APIs

libconform – TODO

- ▶ Test-dichte zu gering
- ▶ Dokumentation
- ▶ Unausgereifte interne APIs
- ▶ Single-threaded und langsam

libconform – TODO

- ▶ Test-dichte zu gering
- ▶ Dokumentation
- ▶ Unausgereifte interne APIs
- ▶ Single-threaded und langsam
- ▶ Weitere Algorithmen der CP-Familie implementieren (aggregated CP, cross-conformal prediction, Venn-Abers,...)

libconform – TODO

- ▶ Test-dichte zu gering
- ▶ Dokumentation
- ▶ Unausgereifte interne APIs
- ▶ Single-threaded und langsam
- ▶ Weitere Algorithmen der CP-Familie implementieren (aggregated CP, cross-conformal prediction, Venn-Abers,...)
- ▶ Mehr nonconformity scores

Vielen Dank

Bei Interesse:

- ▶ <https://github.com/jofas/conform/>
- ▶ jonas@fassbender.dev