Berner Fachhochschule
Haute école spécialisée bernoise
Bern University of Applied Sciences

2024 FS CAS PML - Supervised Learning
4 Validierung (und mehr)
4.2 Validierungstechniken

Werner Dähler 2024

4 Validierung und mehr - AGENDA

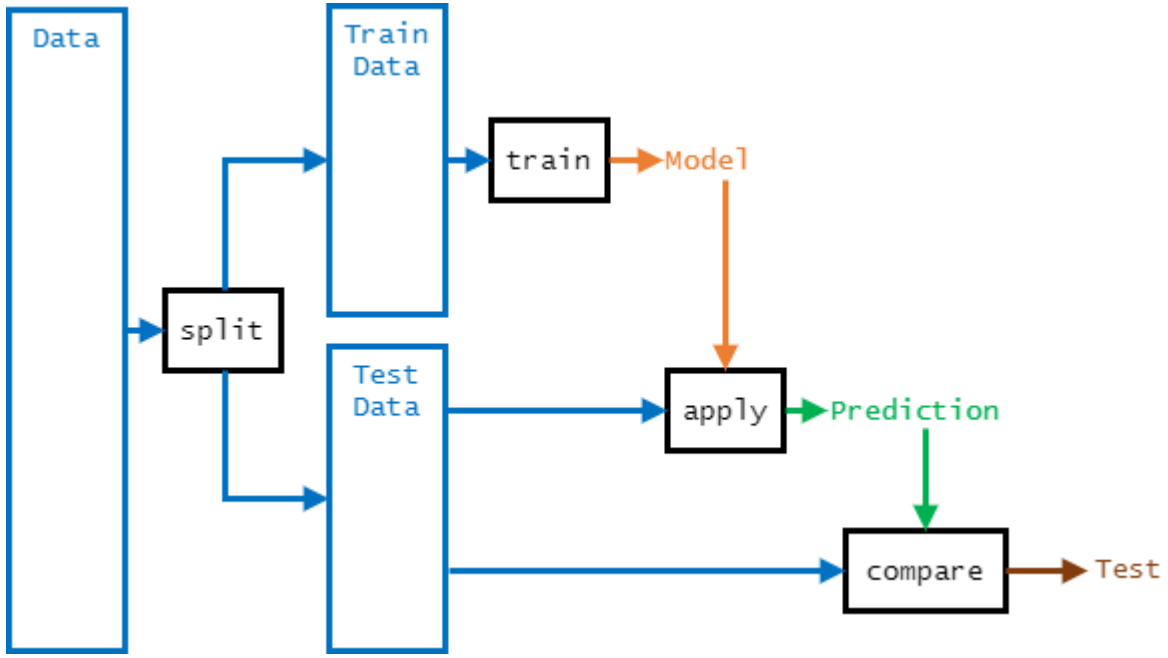
41. Sampling und Resampling
42. Validierungstechniken
43. Grid Search und Random Search
44. Performance Metriken
45. Unbalancierte Daten

4.2 Validierung und mehr - Validierungstechniken

4.2.1 Holdout Validierung

- Wiederholung der bisher meistens angewendeten Methode
  - trainieren des Modells auf den Trainingsdaten
  - anwenden des trainierten Modells auf die Testdaten
  - berechnen einer Performance Metrik

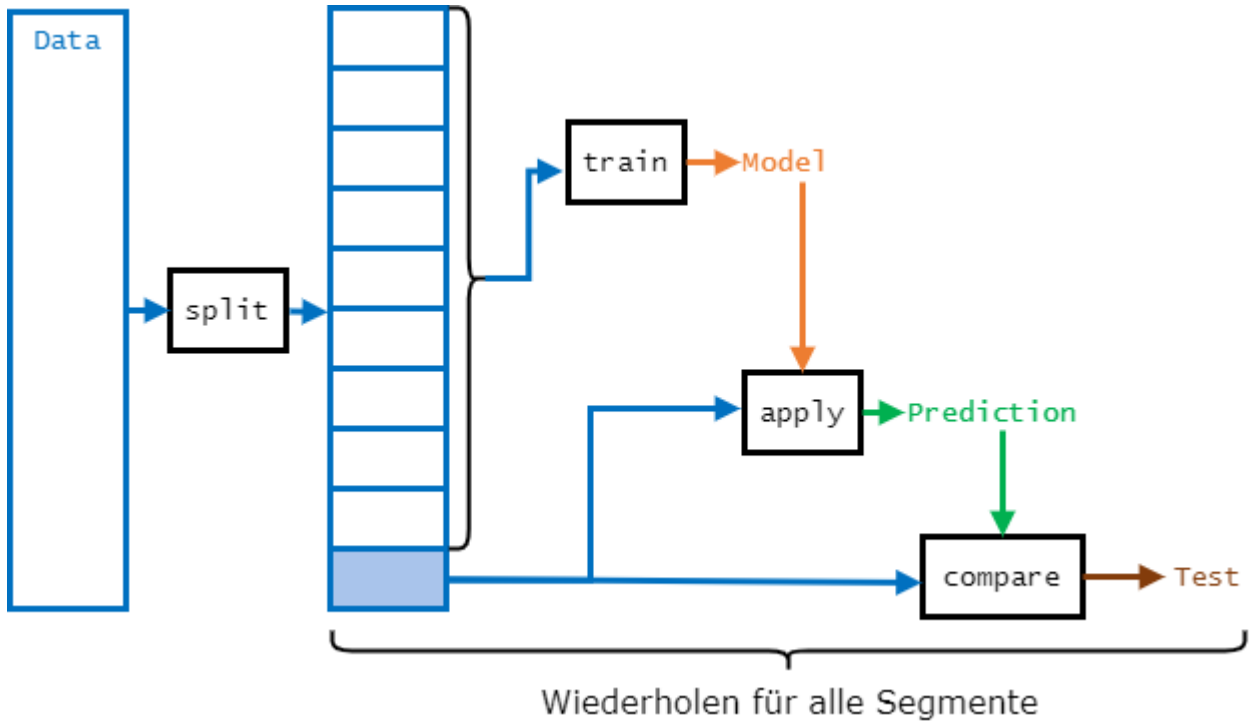
(vgl. prep_data in Modul bfh_cas_pml)



4.2 Validierung und mehr - Validierungstechniken

4.2.2 Kreuzvalidierung

- mögliche Schwäche von Holdout Validierung:
  je nach random_state können unterschiedliche Ergebnisse resultieren
- dieses Verfahren ermittelt einen *konsolidierten* Performance Wert nach systematischem Resampling
- wichtig zur Beurteilung der
  Stabilität eines Learners



4.2 Validierung und mehr - Validierungstechniken

4.2.2 Kreuzvalidierung

- Vorgehen generell
  1. festlegen der Anzahl Teilmengen k (normalerweise 5 oder10)
  2. erzeugen von k disjunkten (möglichst) gleich grossen Teilmengen
  3. für jede Teilmenge T_i:
     - T_i: Testset
     - alle anderen: Trainingsset
     - Modell trainieren auf Trainingsset
     - Prediction für Testset
     - Score berechnen und für jede Iteration speichern
  4. Analysieren der gesammelten Score-Werte
     - Kennzahlen (z.B. Mittelwert aller Score Werte)
     - visuell
- jede Instanz wird also k-1 mal zum Trainieren und einmal zum testen verwendet

4.2 Validierung und mehr - Validierungstechniken

4.2.2 Kreuzvalidierung

- Vorgehen mit scikit-learn (vgl. [ipynb])

```
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import cross_val_score

model = DecisionTreeClassifier(random_state=1234)

## cross validation
kfold = 10 ## default: 5
scores = cross_val_score(model, X, y, cv=kfold)
```

- cross_val_score gibt einen Array mit den Scores der einzelnen Iterationen zurück
- die beiden Anweisungen können auch kombiniert werden (vgl. [ipynb])

4.2 Validierung und mehr - Validierungstechniken

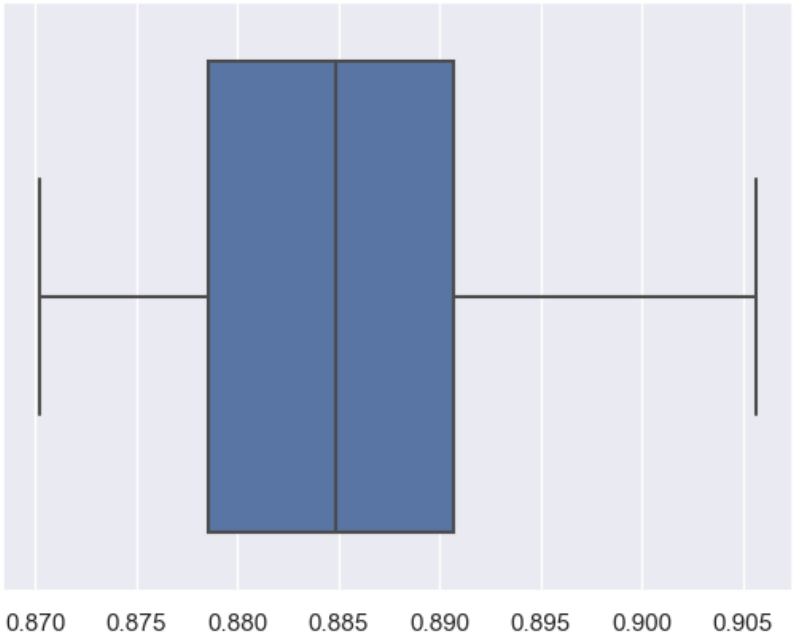
4.2.2 Kreuzvalidierung

- sichten der Ergebnisse

```
print('mean:', np.mean(scores))
print('std: ', np.std(scores))
sns.boxplot(x=scores);
```

```
mean: 0.885902636916835 8
std:  0.010953816382393995
```

- diese Untersuchung liefert uns ein weiteres wichtiges Ergebnis: die **Stabilität** des untersuchten Learners, d.h.:
  - die Abhängigkeit der Performance von random_state beim Train - Test - Split
- **je kleiner die Standardabweichung (std) umso stabiler das Resultat der jeweiligen Methode**



4.2 Validierung und mehr - Validierungstechniken

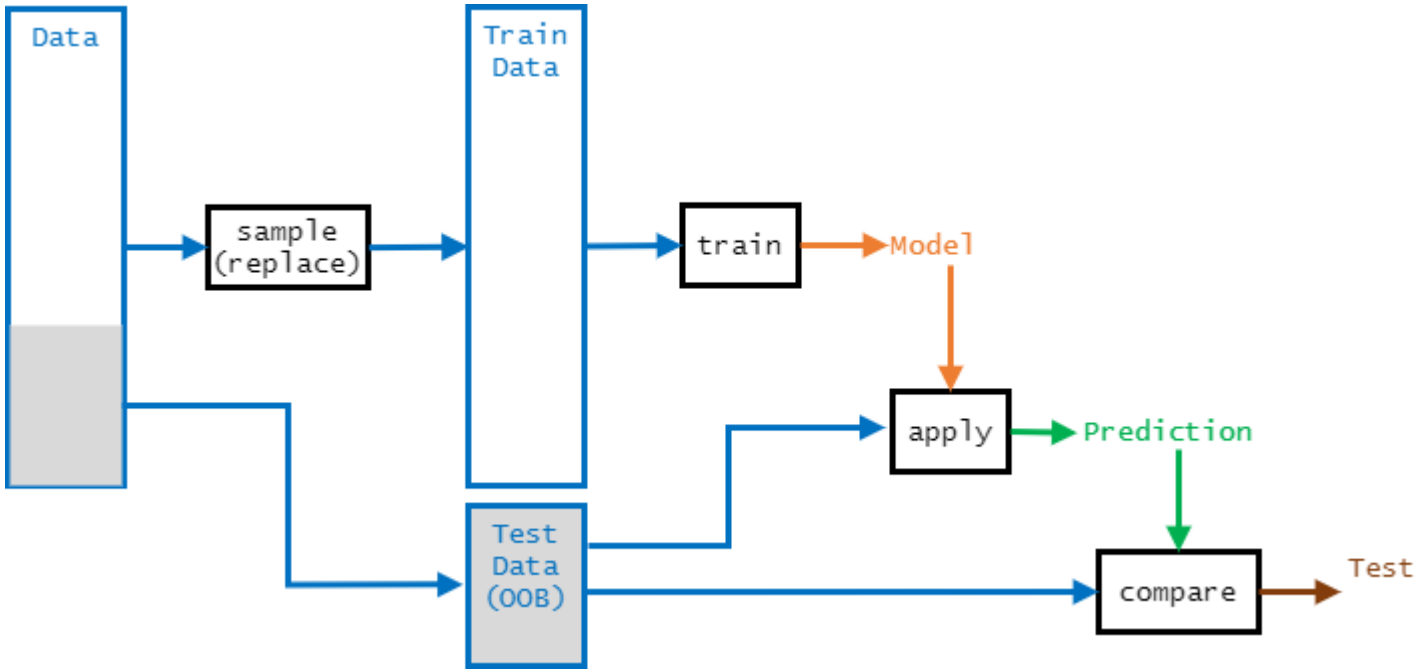
4.2.2 Kreuzvalidierung

- Leave-One-Out cross-validator (sklearn.model_selection.LeaveOneOut): eine spezielle Variante der Kreuzvalidierung
- es werden so viele Folds getestet, wie es Beobachtungen im Dataset hat
  - jede Beobachtung wird einmal zum Testen verwendet
  - alle jeweils übrigen zum trainieren
- ist nur geeignet bei kleineren Datasets!

4.2 Validierung und mehr - Validierungstechniken

4.2.3 Bootstrap Validierung

- Validierungsmethode, basierend auf Boostrap Resampling, d.h. auf wiederholtem Sampling (Resampling) unter speziellen Bedingungen
  - die Samples werden gleich gross wie die jeweilige Population
  - die Samples werden gezogen **mit Zurücklegen** (vgl. 4.1.1.2), da sonst Sample identisch wäre wie die zugrundeliegende Population



4.2 Validierung und mehr - Validierungstechniken

4.2.3 Bootstrap Validierung - EXTRA

- aus wahrscheinlichkeitstheoretischen Überlegungen hat eine Beobachtung (Instanz) bei Bootstrap Sampling eine Chance nicht ausgewählt zu werden von

$$1 - \frac{1}{n}$$

- bei einer Stichprobengrösse von n entspricht dies

$$\left(1 - \frac{1}{n}\right)^n$$

- und bei $n \to \infty$

$$\frac{1}{e} = 0.368$$

- damit wird der Anteil im Sample

$$1 - \frac{1}{e} = 0.632$$

4.2.3 Bootstrap Validierung - EXTRA

- Kontrollen
  - rechnerisch:

```
print(1 - 1 / np.exp(1))
```

```
0.6321205588285577
```

  - experimentell (mittels Simulation):

```
pop_size = 1000000
pop = range(pop_size)
smpl = np.random.choice(pop, pop_size, replace=True)
smpl_size = len(np.unique(smpl))
print(smpl_size / pop_size)
```

```
0.63227
```

4.2 Validierung und mehr - Validierungstechniken

4.2.3 Bootstrap Validierung - EXTRA

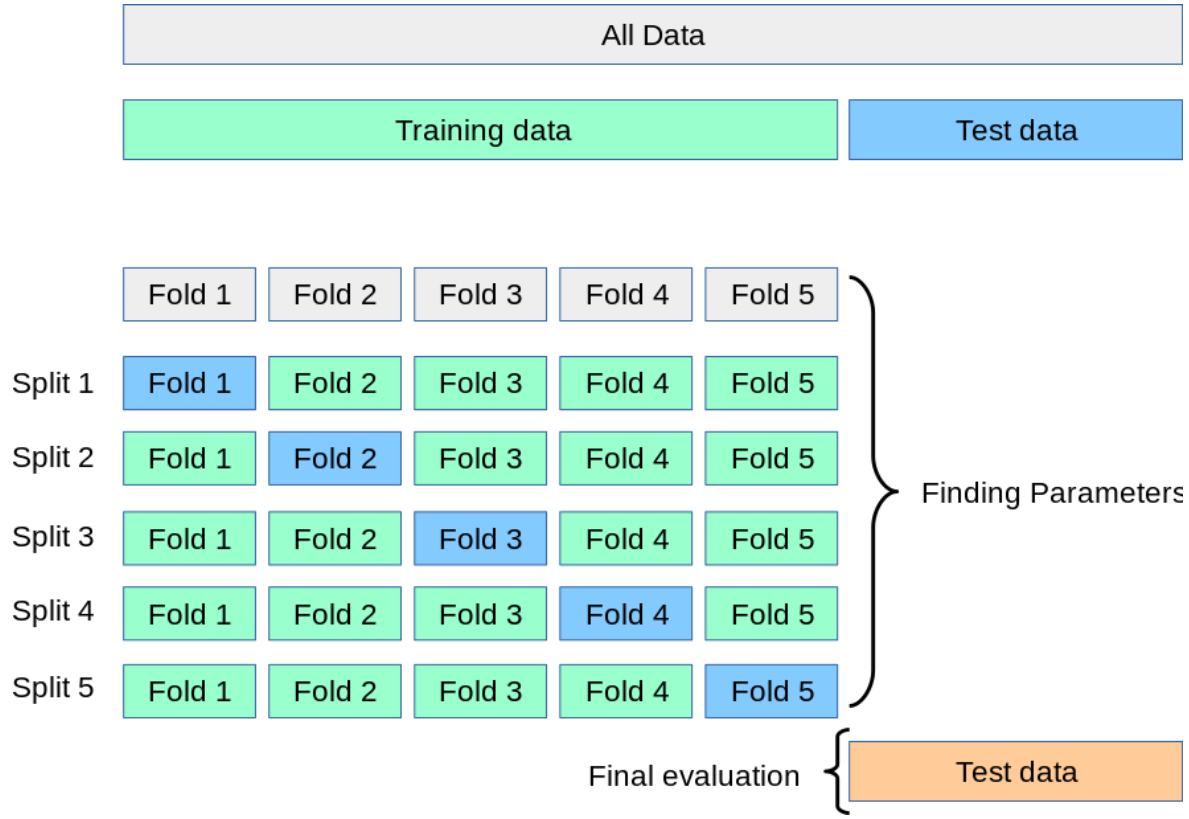
- ist nicht als eigenständige Methode implementiert wie z.B. cross_val_score()
- aber aus anderer Quelle: [ogrisel.github.io]

- ist aber eingebettet implementiert bei einigen Trainingsmethoden, z.B. RandomForestClassifier,
  vgl. die beiden Parameter:
  - **bootstrap : boolean, optional (default=True)**
    Whether bootstrap samples are used when building trees. If False, the whole datset is used to build each tree.
  - **oob_score : bool (default=False)**
    Whether to use out-of-bag samples to estimate the generalization accuracy.

4.2 Validierung und mehr - Validierungstechniken

4.2.4 Train - Eval - Test - Split

- in verschiedenen Publikationen findet man eine Erweiterung des Konzepts von Train - Test - Split, so unter
  - scikit-learn.org (nebenstehende Darstellung)
  - Wikipedia
  - towards data science
- die Idee dahinter ist, dass zum Beurteilen eines trainierten Modells andere Daten verwendet werden sollten als zum Tunen der Hyperparameter
- d.h. z.B. eine Kombination von
  - Holdout Validierung
  - Kreuzvalidierung nur mit Trainingsdaten



4.2 Validierung und mehr - Validierungstechniken

4.2.4 Train - Eval - Test - Split

- dies würde bedeuten, dass bei konsequentem Einsatz von Kreuzvalidierung
  - vorgängig ein expliziter Train - Test - Split durchgeführt werden muss
  - für Modell Selektion und Parameter Tuning ausschliesslich die Trainingsdaten verwendet werden
  - man sich nicht um den Train - Eval - Split kümmern muss, das übernimmt die Kreuzvalidierung selber
  - das finale Modell mit den Testdaten evaluiert wird

4.2 Validierung und mehr - Validierungstechniken

Workshop 13

Gruppen zu 2 bis 4, Zeit: 30'

- vergleichen Sie alle bisher bekannten Klassifikatoren (ausser SVC und MLPClassifier) in Bezug auf deren Stabilität unter Anwendung von Kreuzvalidierung
- verwenden Sie für die Klassifikatoren jeweils Default-Parametrisierung
- setzen Sie für die Kreuzvalidierung folgende Funktion ein: `sklearn.model_selection.cross_val_score`