Handout (scikit-learn)

Vorhersagemodell für die Waldbrandgefahr in Deutschland

Sören Sparmann

2025-05-30

Lineare Regression mit Sklearn (scikit-learn)

Pakete importieren

Importiert das Paket zur Durchführung einer linearen Regression.

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
```

Modell instanziieren

Erzeugt ein leeres Regressionsmodell, das später mit Daten trainiert wird.

```
model = LinearRegression()
```

Datensatz einlesen

Liest die (historischen) Daten ein, die für das Training und die Evaluation des Modells verwendet werden.

```
# Pandas importieren
import pandas as pd

# CSV-Datei einlesen
df = pd.read_csv('data/train.csv')
df.head(5)
```

	x_1	x_2	y
0	1.0	1.9	4.1
1	2.1	1.2	4.0
2	3.0	4.0	9.2
3	3.9	3.0	9.1
4	5.0	5.1	14.0

x_1	x_2	y

Unabhängige und Abhängige Variablen festlegen

Definiert die Eingabedaten X und die Zielgröße y für das Modell.

```
# Unabhängigen Variablen
X = df[['x_1', 'x_2']].values

# Abhängige Variable
y = df['y'].values
```

Daten in Trainings- und Validierungsdaten aufteilen

Teilt den Datensatz in zwei Teile auf: **Trainingsdaten** und **Validierungsdaten**. Die **Trainingsdaten** werden verwendet, um das Modell zu „lernen“, also die optimalen Parameter zu finden. Die **Validierungsdaten** dienen anschließend zur **Bewertung**, wie gut das Modell auf **neue, bisher unbekannte Daten** generalisiert.

```
# Daten in Trainings- und Validierungsdaten aufteilen
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.25, random_sta
```

Modell an die Daten anpassen (trainieren)

Trainiert das Modell auf Basis der vorhandenen Datenpaare X und y.

```
model.fit(X_train, y_train)
```

```
LinearRegression()
```

Vorhersage machen

Berechnet auf Grundlage neuer Eingabewerte eine Vorhersage.

```
model.predict([
    [2, 3]
])
```

```
array([6.46585087])
```

Modell evaluieren

Bewertet die Güte der Modellanpassung anhand des Bestimmtheitsmaßes (R^2).

```
# Score (Bestimmtheitsmaß) berechnen
model.score(X_test, y_test)
```

```
0.9950829109284132
```

Vorhersagefehler visualisieren.

```
from sklearn.metrics import PredictionErrorDisplay

pred = model.predict(X_test)

ped = PredictionErrorDisplay(y_true=y_test, y_pred=pred)
ped.plot(kind="actual_vs_predicted")
```