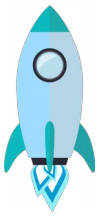

CHEAT-SHEET
ML-Methoden

Bundeswettbewerb Künstliche Intelligenz

K-Nächste-Nachbarn

Art des Problems Klassifizierung/Regression

Grundidee
Um das Label eines unbekannten Datenpunktes vorherzusagen, werden die k nächsten Datenpunkte in Betracht gezogen. Der neue Datenpunkt gehört zu der Klasse welche am häufigsten unter den k Punkten vertreten ist. Alternativ kann man KNN auch für Regression verwenden. Dabei wird dem neuen Datenpunkt der Durchschnitt der k umliegenden Punkte zugeordnet.

Entscheidungsbäume

Art des Problems
Klassifizierung/Regression

Grundidee
Datenpunkte werden in Klassen eingeteilt (oder ihnen wird ein Wert zugewiesen), aufgrund von kategorischen Entscheidungen die aufeinander Folgen bis ein Blatt des Baums erreicht wird.

Kriterium 1
Die Blattknoten sollten idealerweise nur Datenpunkte einer Klasse enthalten.
-Gini impurity:

$$G = \sum_{i=1}^{m} p_i \cdot (1 - p_i)$$

Kriterium 2
Der Entscheidungsbaum sollte möglichst klein sein, also möglichst wenige Entscheidungsknoten enthalten.
-Information gain (IG(Parent)):

$$G(parent) - (\frac{n_{child1}}{n_{parent}} \cdot G(child1) - \frac{n_{child2}}{n_{parent}} \cdot G(child2))$$

Ein geeigneter Baum kann bspw. mit Hilfe des CART Algorithmus bestimmt werden.

Entscheidungsbäume können auch für Regressionsprobleme genutzt werden. Kriterium 1 ist dann der quadratische Fehler und Kriterium 2 ist in diesem Fall die Varianzreduktion.

Lineare Regression

Art des Problems Regression

Grundidee
Zeichnen einer Gerade f(x) = mx+b durch die Datenpunkte, sodass die Lossfunktion (vereinfacht: die Summe der Abstände zwischen der Geraden und der Datenpunkte) möglichst gering ist.

Gängige Lossfunktion MSE

$$\frac{1}{n}\sum_{i=1}^{n}(f(x_i) - y(x_i))^2$$

Nach Ableiten und Umformen sind die optimalen Parameter:

$$\hat{\theta} = \begin{bmatrix} m \\ b \end{bmatrix} = (x_b^T \cdot x_b)^{-1} \cdot x_b^T \cdot y \qquad x_b = \begin{bmatrix} 1 \\ x \end{bmatrix}$$

Naiver Bayes

Art des Problems Klassifizierung

Grundidee
Satz von Bayes:

$$P(X|Y) = \frac{P(Y|X) \cdot P(X))}{P(Y)}$$

Naiver Teil
Wenn Y ein Schnitt von Variablen ist, kann es schnell passieren, dass es keine Beobachtung Y im Datensatz gibt und wir müssen “naiv“ davon ausgehen, dass die einzelnen Variablen aus Y unabhängig sind.

$$P(A \cap B) = P(A) \cdot P(B)$$

Damit lässt sich die ursprüngliche Gleichung umformen:

$$P(X|A \cap B) = \frac{P(X|A) \cdot P(X|B)}{(P(A) \cdot P(B)}$$

CHEAT-SHEET
ML-Bibliotheken

Bundeswettbewerb Künstliche Intelligenz

Numpy

Numerische Berechnungen mit großen oder mehrdimensionalen Matrizen/ Arrays

```
import numpy as np
```

Erstellen von Arrays
```
a = np.array([1,2,3,4,5,6,7,8])
b = np.array([[1,2,3,4],[5.0,6,7,8]])
c = np.zeros((2,3))
e = np.random.rand(3)
f = np.arange(1,4)
```

Umformen von Arrays
```
a.reshape(2, 4) → [[1 2 3 4][5 6 7 8]]
np.concatenate([f,f]) → [1 2 3 1 2 3]
np.stack([f, f]) → [[1 2 3] [1 2 3]]
```

(Lineare) Algebra
```
f*3 → [3 6 9]
f < 4 →  [True, False, False]
np.sum(a) → 36
np.max(a) → 8
np.argmax(a) → 7
np.mean(a) → 4.5
```

Indexing
```
b[0] → [1. 2. 3. 4.]
b[:,0] → [1. 5.]
b[-1] → [5. 6. 7. 8.]
b[:,-1] → [4. 8.]
```

Pandas

Bearbeitung versch. Datenformate

```
import pandas as pd
```

Erstellen eines pandas DataFrames
```
df = pd.DataFrame(dictionary,
columns=["x", "y"], index=["a", "b"])
spalten_namen = list(df.columns)
anzahl_reihen = len(df)
```

Operationen auf pandas DataFrames
```
df = df.T #transponieren des DataFrame
df["neue spalte"] = [Wer1, Wert2]
df.append(pd.Series([3, 3], name="n"))

spalte_x = df["x"]
wert_x_b = spalte_x["b"]

df.dropna()
df.drop_duplicates()
daten_summen = df.sum()
daten_mittelwerte = df.mean()
```

Sklearn

Stellt Estimator zur Verfügung, welche direkt trainieren und vorhersagen können.

```
from sklearn import ...

classifier = neighbors.KNeighborsClassifier(k)
classifier.fit(X_train, y_train)
predictions = classfier.predict(X_test)
```

liste von Estimators
```
"KNN", neighbors.KNeighborsClassifier(5)
"SVC", svm.SVC(random_state=42)
"DTC", tree.DecisionTreeClassifier(42)
"RFC", ensemble.RandomForestClassifier(42)
"ABC", ensemble.AdaBoostClassifier(69)
"NBC", naive_bayes.GaussianNB()
"LRC", linear_model.LogisticRegression(1e5)
"MLP", neural_network.MLPClassifier()
```

Matplotlib

Visualisierung von Daten

```
import matplotlib.pyplot as plt
```

Erstellen von Plots:
```
fig, ax = plt.subplots()

# Basic Plot
ax.plot(x_werte, y_werte,
        label="l", color="red")

# Scatterplot
ax.scatter(x_werte, y_werte,
        s=metric, c=color)

# Balkendiagramm
ax.bar(namen, werte)

ax.set_title("Title")
ax.set_xlabel("$x$-Achse")
ax.set_ylabel("$y$-Achse")

ax.legend()

ax.set_axisbelow(True)
ax.grid()

plt.show()
```