ALD – Projektarbeit SS 2021

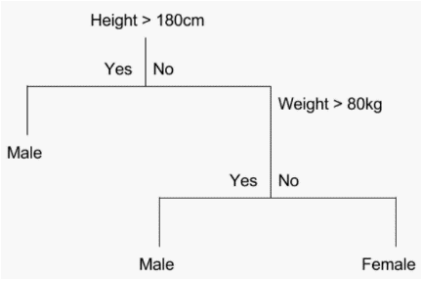
Betreuung: K. Horneck

Entscheidungsbaum

Entscheidungsbäume sind eine einfache, und doch sehr erfolgreiche Form des maschinellen Lernens. Ein Entscheidungsbaum ist ein Baum, in dem jeder Verzweigungsknoten eine Auswahl zwischen einer Reihe von Alternativen und jeder Blattknoten eine Klassifizierung oder Entscheidung darstellt.

Einfaches Beispiel:
Bei einem Datensatz mit zwei Attributen (Input x besteht aus x_1 (Größe) und x_2 (Gewicht) ist die Ausgabe y (Geschlecht als Mann oder Frau) ein einfaches Beispiel für einen binären Entscheidungsbaum.

x_1 Size	x_2 Weight	y Sex
160	60	Female
170	55	Female
180	85	Male
181	79	Female
…		



<u>Entscheidungsbaum für Klassifizierung mit CART Algorithmus:</u>

Implementierung eines Entscheidungsbaumes für **die Klassifikation** => **Zielvariable (Output) ist kategorial** (lt obigen Beispiel Sex: Female/Male

Verwende für die Implementierung **den CART Algorithmus** mit dem Maß der Gini Impurity. Im CART Algorithmus sind die Splits immer binär (**Binärbaum**).
Der Schlüssel zum CART-Algorithmus besteht darin, das optimale Merkmal und den optimalen Schwellenwert so zu finden, dass die Gini-Impurity (Verunreinigung) mit dem Gini-Index minimiert wird.
Gini Index = $1 - \Sigma (P_i)^2$ für alle unterschiedlichen Klassen (Kategorien) in der Outputvariable (lt obigen Bsp gibt es zwei Klassen => Klasse 1: Sex = Female, Klasse 2: Sex = Male)

Hilfreiche Informationen im Internet

- https://en.wikipedia.org/wiki/Decision_tree_learning
- https://medium.com/geekculture/decision-trees-with-cart-algorithm-7e179acee8ff
- https://www.youtube.com/watch?v=7VeUPuFGJHk&t=255s

Aufgabenstellung

1. Pseudo code
2. **Implementierung** des Entscheidungsbaumes (decision tree) in Python als Binärbaum **ohne** Verwendung von machine learning package (pandas und numpy können verwendet werden, Scikit-learn (Sklearn) darf **nur** zum Vergleich bzw Test verwendet werden)

   Verwende einen Binärbaum als Datenstruktur. Jeder Knoten repräsentiert ein Attribut (Eingabevariable (x)) und einen Split-Point auf dieser Variablen. Die Blattknoten des Baums enthalten eine Ausgabevariable (y), die zur Vorhersage verwendet wird.

   Verwende mind. zwei Python-Klassen (**class node** für Repräsentation der Knoten im Entscheidungsbaum und **class DecisionTreeClassifier** für den Entscheidungsbaum).

```
class Node:
class DecisionTreeClassifier:
```

3. Datensatz
   Es steht der Datensatz titanic.csv zur Verfügung (Trainingsdaten, siehe auch Zusatzinfos). Getestet wird der Baum an Hand der Passagiere „Rose“ (female, Ticketklasse 1 und Alter 17 Jahre) und „Jack“ (male, Ticketklasse 3 und Alter 19 Jahre).
4. Graphische oder textuelle Darstellung des Entscheidungsbaumes
5. Beschreibung des Algorithmus
6. Beschreibung des verwendeten ADT und der verwendeten Datenstruktur
7. Beschreibung der Laufzeit- und Speicherplatzkomplexität

Zusatzinfos:

- Der Datensatz enthält keine fehlenden Werte
- Es muss keine Unterscheidung in Train und Testdaten gemacht werden. Der Baum wird mit dem gesamten Datensatz erstellt (= Trainingsdaten)
- Nur die Attribute “**Pclass”, “Sex”, “Age”** – siehe Datensatz – sind für die Ermittlung relevant (es muss kein Feature Selection durchgeführt werden)
- Es sind keine Fehleranalysen zu erstellen
- Es ist kein „Pruning“ notwenig. Maximale Tiefe muss nicht beschränkt werden. Abbruchkriterium an Hand der Parameter (min_samples_leaf = 2 und in_sample_node = 2)
- Anzahl der Kategorien (Klassen) der Outputvariable (hier Überlebt Ja = 1 oder Nein = 0) kann mit 2 angenommen werden

Datensatz

Schiffsunglück der Titanic 1909 - Verwenden Sie den Datensatz: titanic.csv

- Von diesen Datensatz sind **für die Verarbeitung nur X = „Pclass“, „Sex“, „Age“ und y = „Survived“** relevant**.**
- Die Daten enthalten keine fehlenden Werte
- Wie bewertet Euer Modell die Passagiere „Rose“ (female, Ticketklasse 1 und Alter 17 Jahre) und „Jack“ (male, Ticketklasse 3 und Alter 19 Jahre)?

Input Variablen

- **Pclass** - Ticketkategorie (Info: Klasse 1, 2 oder 3)
- Name

- **Sex**
- **Age**
- Siblings.Spouses.Aboard (Info: Geschwister bzw Ehepartner an Board)
- Parents.Children.Aboard (Info: Kinder bzw. Eltern an Board)
- Fare (Info: Ticketpreis)

Output Variable

- **Survived**: Schiffsunglück überlebt - JA (1) bzw Nein (0)

Zusätzliche Python-Informationen (kann verwendet werden)**:**

Verwende nur diese Attribute (Features, Variablen) `'Survived', 'Pclass', 'Sex', 'Age'`:

```
df_a = df.loc[:,['Survived', 'Pclass', 'Sex', 'Age']]
```

Codieren der Variablen in numerische Daten, zB male = 0 (female = 1)

```
df_a.loc[df_a["Sex"] == "male", "Sex"] = 0
```

Vor Übergabe an den DecisionTreeClassifier trenne den Dansatz in X (Input) und y (Output):

```
X = df_a[['Pclass', 'Sex', 'Age_band']]
y = df_a[['Survived']]
```