

Experimente gestalten fürs Maschinelle Lernen

Übung 3: Entscheidungsbäume in max depth

16. November 2021

Zeit für die Bearbeitung: zwei Wochen, bis zu dem <i>30.11., 9 Uhr</i> . Abgabe bei GitLab

Hinweise zur Gruppenarbeit (falls zutreffend)

Aufgabenteile gekennzeichnet mit **(A)** und **(B)**, müssen von unterschiedlichen Gruppenmitgliedern commitet werden.

Übung

Ziel dieser Aufgabe ist, eigene Experimente zu dem Einfluss des `max_depth` Parameters auf Entscheidungsbäume durchzuführen. Dafür schreiben wir entweder unseren eigenen Code oder verwenden große Teile des Codes aus dem letzten Blatt.¹

1. Arbeite weiterhin in Deinem GitLab Portfolio. Erstelle darin einen neuen Ordner `03_übung` und bearbeite die folgenden Punkte ausschließlich in diesem Ordner.
2. **Nicht vergessen:**
 - (a) die Code Guidelines aus der Vorlesung vom 4. Okt. (siehe Slides mit Mindmap) zu beachten.
 - (b) für diesen Ordner ein `README.md` zu schreiben! Hinweis: um den Schreibprozess zu optimieren, kann man sich auch `README` templates erstellen und immer wieder wenn passend benutzen (auch außerhalb des Seminars).
 - (c) im `README.md` die Antworten auf die in der Aufgabe gestellten Fragen festzuhalten. Es gibt keine rhetorischen Fragen.
3. **(A)** Generiere die 3 Datensätze aus dem letzten Übungsblatt wieder und „bezeichne“ sie mit `X_train`. Die Labels dazu (in rot und blau kodiert) sollten unter `y_train` zu finden sein.

Hinweise:

- Der Code vom letzten Mal sollte hierfür sehr hilfreich sein.
 - Was bedeutet „bezeichne“? Die Aufgabe spezifiziert nicht, wie oder in welcher Datenstruktur die 3 Datensätze zusammengefasst werden sollten. Wichtig ist nur, dass man für die Trainingsdaten in `X_train` suchen muss, und für die dazugehörigen Labels in `y_train`.
4. **(B)** Ändere und erweitere den Code so, dass Du damit auch einen Testdatensatz `X_test` mit `y_test` generierst. Achte dabei, dass die Testdaten aus der gleichen Distribution kommen (also auf gleicher Art generiert werden), aber nicht wirklich die gleichen Daten wie `X_train` sind (also mit einem anderen random seed, oder im Code vom letzten Mal: `random_state=...`).

¹was wir vor einer Woche inspiziert, interpretiert und verstanden haben

5. **(B) Für jeden Datensatz:**

- (a) Fitte/trainiere den Datensatz mit einem Entscheidungsbaum aus scikit-learn mit den default Parametern: `cls = DecisionTreeClassifier().fit(X_train, y_train)`

Hinweise:

- `cls` enthält den gelernten Classifier (das Modell).
- Trainiere einen Classifier pro Datensatz.

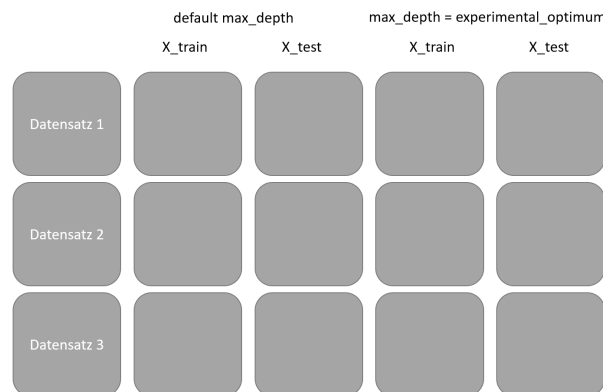
- (b) Welche Vorhersagen macht der Classifier auf **X_test**? EN: predict on the test set.
- (c) Berechne die Genauigkeit (accuracy) des Classifiers auf dem **X_test**: Wie viel Prozent der Vorhersagen auf dem Test Set sind richtig? Hinweis: Dafür darfst Du gerne Deine eigene Funktion schreiben oder Python Pakete dazu benutzen.
- (d) Berechne die Genauigkeit (accuracy) des Classifiers auf dem **X_train**.
- (e) Experimentiere mit dem Parameter **max_depth** des Entscheidungsbaums und bestimme anhand der Genauigkeit welcher **max_depth** optimal ist (pro Datensatz). Was bedeutet optimal? Bezieht sich das auf das Train Set oder auf das Test Set oder auf deren Verhältnis? Warum?

6. **(A) Visualisiere das Ergebnis. Dafür:**

- (a) Fasse die drei Datensätze in einem Plot zusammen, z.B. untereinander, wie im Plot im letzten Blatt.
- (b) Stelle für jeden Datensatz den Plot des Classifiers mit dem default **max_depth** Parameter neben dem Classifier mit dem oben gefundenen Optimalen **max_depth**.

Hinweise:

- Siehe Mockup/Skizze unten.
 - Der plotting Code vom letzten Mal sollte nützlich sein.
- (c) Wie unterscheiden sich visuell die Ergebnisse? Achte dabei auf die decision boundary (die Grenze, die rot und blau trennt) und beschreibe basierend auf dem Plot wie man die Genauigkeit (accuracy) Werte zwischen den unterschiedlichen experimentellen Einstellungen erklären kann.



7. Hierfür sollen die Punkte 5. und 6. nochmal, aber mit einem Random Forest durchgeführt werden. Wie unterscheiden sich die Ergebnisse des Random Forests zu dem Decision Tree (Diskussion)? Achte dabei unbedingt auch auf die decision boundary, **max_depth**, Train und Test Genauigkeit.
8. Für einen der Modelle/Bäume/`clf` Deiner Wahl: `sklearn.tree.plot_tree(clf, filled=True)`
9. Im letzten Blatt gab es die Aufforderung Dir schon Gedanken zu machen zu Daten und möglichen Projektaufgaben. Schreibe deine Idee kurz im README der 3. Übung auf und schreibe Pro- und Gegenargumente dafür, das Problem mit einem Decision Tree (nicht) zu lösen. Die folgenden Fragen sollten hilfreich sein:

1. Was ist das Problem was man lösen möchte?

2. Aus welchen Daten kann/könnte man lernen?
3. Was sind die Features/Attribute?
4. Wie kann man die Performanz messen?