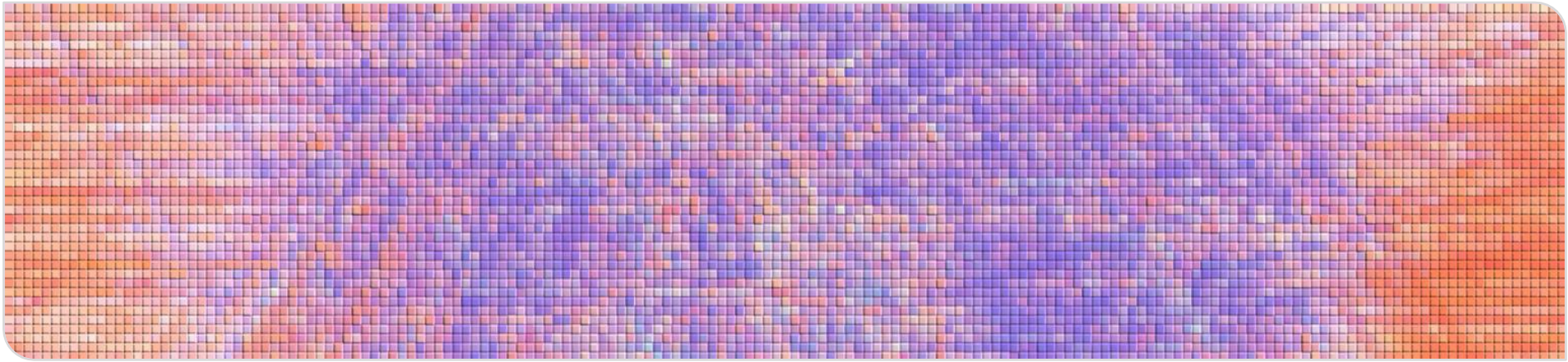
Karlsruher Institut für Technologie

AGD – WS 2020/21 – Übungssitzung 2

Jakob Bach (jakob.bach@kit.edu)



Zeitplan

Woche	Datum*	Übung
1	03.11.2020	
2	10.11.2020	
3	17.11.2020	Abgabe Blatt 1
4	24.11.2020	
5	01.12.2020	Abgabe Blatt 2
6	08.12.2020	Sitzung 1
7	15.12.2020	Abgabe Blatt 3

Woche	Datum*	Übung
8	22.12.2020	
9	12.01.2021	Abgabe Blatt 4
10	19.01.2021	Sitzung 2
11	26.01.2021	*Keine Abgabe*
12	02.02.2021	Sitzung 3
13	09.02.2021	Abgabe Blatt 5
14	16.02.2021	Sitzung 4

* regulärer Vorlesungstermin am Dienstag; Abgabetermin der Übungsblätter kann ein anderer Tag sein

Agenda

Inhalt: Übungsblätter 3/4, Vorlesungskapitel 4-8

- Umfrage: Vorbereitung auf Sitzung
- Eure Fragen: Code-Aufgaben, Online-Tests, Sonstiges
- Umfrage: R oder Python
- Theorie-Aufgaben
- Kurze Feedback-“Runde“

Ankündigungen:

- Weitere Übungssitzungen finden am 02.02. und 19.02. statt.
- Frist für Übungsblatt 5 um zwei Wochen verschoben.
- Blatt 6 wird nur Qualifikationsaufgabe für das Praktikum enthalten.

Fragen

- Qualifikationsaufgabe für Praktikum auch in Python lösen?
  - Ja, R und Python zulässig
- Prüfungsprotokolle vergangener mündlicher Prüfungen?
  - https://www.fsmi.uni-karlsruhe.de/odie/web/
- Wann ist Prüfung möglich?
  - Terminvergabe über Sekretariat: http://dbis.ipd.kit.edu/2795.php
  - Rund um das Jahr Prüfungstermine, wenn auch zu manchen Zeiten (z. B. kurz nach Vorlesungsende) mehr Termine als zu anderen Zeiten
  - Prüfung problemlos im Sommersemester möglich
- AGD-Praktikum auch im Bachelor?
  - im Modulhandbuch nachschauen
  - falls dort nicht enthalten, als Mastervorzugsleistung einbringen

Theorieaufgaben – Räumliche Indexstrukturen

- Wie unterscheiden sich R-Baum, kd-Baum und kdB-Baum?
  - *R-Baum*:
    - keine Partitionierung, d.h. nicht der komplette Raum ist aufgeteilt, sondern nur teilweise mit Rechtecken bedeckt, die sich evtl. überlappen (keine Überlappung bei R+-Bäumen)
    - balanciert
  - *kd-Baum*:
    - Partitionierung (gesamter Raum ist aufgeteilt)
    - unbalanciert
  - *kdB-Baum*:
    - Partitionierung (gesamter Raum ist aufgeteilt)
    - Balancierung auf physischer Ebene

Theorieaufgaben – Pruning von Bäumen

- Würden Sie Pre-Pruning oder Post-Pruning bevorzugen und warum?
  - *Post-Pruning*:
    - Zunächst rechenintensiv (da vollständiger Baum trainiert wird)
    - Danach beliebig starkes, sogar mehrfaches Pruning ohne neues Trainieren möglich
  - *Pre-Pruning*:
    - Schnelleres Training, da früher abgebrochen wird (z. B. wenn nicht ausreichende Verbesserung durch Split)
    - Stärke des Prunings muss vorher festgelegt werden (wann Training abbrechen?)
    - Sinnvolle Kombinationen von Attributen schwieriger erkannt durch frühzeitigen Abbruch (aber generelles Problem bei Entscheidungsbäumen, dass Attribute nur hintereinander betrachtet werden, d.h. immer nur ein Attribut gleichzeitig)

Theorieaufgaben – Wahl des Splits

- Wie wählt man am besten den Split eines Entscheidungsbaumes?
  - Idee: Reinheit/Unreinheit (des Klassenlabels in den Daten) anhand eines Kriteriums messen
  - Dadurch ein Attribut auswählen, welches gut splittet (möglichst reine Partitionen erstellt)
  - In der Vorlesung: Minimierung der Entropie des Splits
  - Andere Split-Kriterien wie z. B. Gini-Koeffizient existieren ebenfalls (in Software-Bibliotheken kann man unter Umständen das Kriterium wählen)

Theorieaufgaben – Overfitting

- Wie kann man Overfitting vermeiden?
  - Gewisses Risiko für Overfitting besteht immer
  - Risiko abhängig vom Klassifikator (komplexere Modelle anfälliger)
  - Risiko abhängig vom Datenbestand (wie hilfreich sind Attribute?)
  - Erkennung von Overfitting mit Holdout-Methode, k-fold cross-validation
  - Datenreduktionstechniken (PCA, Feature Selection etc.) helfen gegen Overfitting
  - Vereinfachen von Modellen (z. B. Pruning bei Entscheidungsbäumen, Regularisierung der Zielfunktion) hilft ebenfalls
  - Durch Bekämpfung von Overfitting könnte Vorhersage schlechter werden (Übergang zu Underfitting)

Theorieaufgaben – MDL und MML

- Wie unterscheiden sich MDL und MML, was ist ihr Anwendungsbereich?
  - Achtung: in Literatur wird Begriff „MDL“ teilweise auch für das verwendet, was in Vorlesung „MML“ heißt
  - MDL ist das Prinzip, eine möglichst kurze Beschreibung (Codierung) für Daten zu finden
  - Länge des Codes hängt mit der Verteilung der Daten zusammen
  - MML ist eine Anwendung von MDL
  - Um Daten mit einem Modell zu beschreiben, ist sowohl Modell als auch die Beschreibung der Daten, gegeben das Modell, relevant
  - Bei Auswahl geeigneter Modelle gemäß MML müssen diese beiden Komponenten berücksichtigt werden (Minimierung der Summe der beiden Codierungslängen)

Theorieaufgaben – Evaluationsmaße

- Welche Möglichkeiten gibt es, um Classifier zu vergleichen? bzw. Nennen Sie die möglichen Fehlerarten und erklären sie diese!

  (gute Prüfungsfrage, aber da ausführlich in den Folien, hier nur kurz)

  - Gesamterfolgsquote (Accuracy): für unbalancierte Klassenverteilung schlecht
  - Kappa-Koeffizient: Vergleich mit einfacher Baseline-Vorhersage
  - Precision
  - Recall: Recall von 1 einfach zu erreichen, indem immer „positiv“ vorhersagt
  - F1-Maß: vollständigeres Bild durch Kombination von Precision und Recall
  - Lift-Chart, ROC-Kurve für Vorhersage von Wahrscheinlichkeiten
  - Quadratic loss / informational loss auch für Wahrscheinlichkeiten
  - Numerische Vorhersagen: MSE, RMSE, MAE etc.

Theorieaufgaben – Informational Loss

- Warum wird die Logarithmusfunktion für die Informational-Loss-Funktion verwendet?
  - Informational loss: Logarithmus der vorhergesagten Wahrscheinlichkeit der tatsächlichen Klasse eines Datenobjektes (andere Klassen ignoriert)
  - Bei richtiger Vorhersage (Wahrscheinlichkeit = 1) ein Loss von 0
  - Abweichungen von korrekter Klasse werden logarithmisch stärker bestraft (unendlicher Loss bei vorhergesagter Wahrscheinlichkeit von 0)
  - Logarithmus ist auch informationstheoretisch begründet (MDL)

Theorieaufgaben – FP-Trees

- Wofür benutzen wir FP-Trees und warum?
  - FP = Frequent pattern
  - FP-Baum = Datenstruktur zum schnellen und Speicher-effizienten Finden von Frequent Itemsets
  - Bei Apriori: Generate&Test-Paradigma, iteratives Vereinigen von Itemsets; bei jeder Iteration wird Datenbank gescannt, um Häufigkeit der Itemsets zu zählen
  - beim FP-Tree nur 2 Datenbank-Scans, Zählen der Items und Aufbau des Baums, alle weiteren Analysen können dann ohne Zugriff auf den ursprünglichen Datenbestand durchgeführt werden