

## 8. Преобразуване на данните

### Data transformations (Трансформации на данни)

- **Attribute selection:** Независима от схемата и специфична за схемата
- **Attribute discretization:** unsupervised и supervised, базирана на грешка срещу базирана на ентропия, обратна на дискретизацията
- **Projections:** Principal Component Analysis (PCA), случайни проекции (random projections), Partial Least Squares, Independent Component Analysis (ICA), Linear Discriminant Analysis (LDA), текстови и времеви серии
- **Sampling:** Резервоарно извадково (Reservoir sampling)
- **Dirty data:** Пречистване на данни (data cleansing), устойчива регресия (robust regression), откриване на аномалии (anomaly detection)
- Преобразуване на многокласови в бинарни класове - кодове за корекция на грешки, ансамбли от вложени дихотомии
- Калибриране на вероятности за класове

### Просто да приложим обучаващ алгоритъм? НЕ!

- Избор на схема/параметри
- Третиране на избора като част от обучителния процес, за да се избегнат оптимистични оценки
- Модифициране на входа: **Data engineering**, за да стане обучението възможно или по-лесно

- Модифициране на изхода: Преобразуване на многокласови задачи в двукласови; Прекалибриране на вероятностите

### **Attribute selection (Избор на атрибути)**

- Много важен на практика ; Добавянето на случаен (т.е. нерелевантен) атрибут може значително да влоши представянето (напр. на C4.5); Алгоритмите на базата на примери (instance-based) са особено чувствителни; Изискванията към обема на данните нарастват експоненциално с броя нерелевантни атрибути; Изключение: **naïve Bayes** се справя добре; Дори релевантни атрибути може да са вредни, ако подвеждат алгоритъма

### **Scheme-independent attribute selection (Независим от схемата избор на атрибути)**

- **Filter approach**: оценка на атрибути на база общи характеристики
- Независим от конкретен ML алгоритъм
- Методи:
  - Минимален поднабор от атрибути, който разделя данните
  - Използване на различен, бърз алгоритъм за селекция (напр. C4.5, 1R, линейни модели)
- **Attribute weighting** чрез instance-based learning
- **Correlation-based Feature Selection (CFS)**: измерва взаимната несигурност между атрибути чрез ентропия

### **Scheme-specific selection (Специфичен за схемата избор) -**

начинът на избор на атрибути или примери зависи от конкретния алгоритъм (схема), който използваме-

- **Wrapper approach:** избира атрибути с включена обучаваща схема
- Оценка чрез **кръстосана валидация**
- Скъпо изчислително (време  $\sim k^2$  при алчни методи)
- Спиране чрез статистически тест (race search)
- Ефикасен за **decision tables** и **naïve Bayes**

**Attribute discretization (Дискретизация на атрибути)** - да превърнем непрекъснати (числови) стойности в групи (интервали) или категории.

- Полезна дори при алгоритми, работещи с числови атрибути
- Избягва нормално разпределение в **naïve Bayes**, clustering
- Видове: **1R**: проста локална дискретизация; **C4.5**: локална;  
**Глобална дискретизация**: използва повече данни
- Дискретизиране до:  $k$ -стойности или  $(k - 1)$  бинарни атрибута (по-добре за decision trees)

**Discretization: Unsupervised (Ненаблюдавана)**

- Определя интервали без информация за клас
- Методи: Equal-interval binning; Equal-frequency binning (хистограмно изравняване)
- Обикновено по-слабо от supervised
- Equal-frequency работи добре с **naïve Bayes**, ако броят интервали е  $\sqrt{N}$

**Discretization: Supervised (Наблюдавана)**

- Класическият метод е базиран на **ентропия**

- Създава decision tree с предварително спиране (pre-pruning)
- Използва принципа на **минимална дължина на описанието (MDL)**
- Оценка: Точка на разделяне + разпределение на класовете
- Сравнява дължините на описанието преди/след разделяне

**Error-based vs Entropy-based: Когато дискретизираме (групираме) числов атрибут в интервали, можем да изберем как точно да го направим – чрез:**

#### **Error-based дискретизация**

- Цел: да намали броя на грешките (misclassification error) след дискретизацията.
- Обикновено не обединява съседни интервали, ако те имат различни класове.

#### **Entropy-based дискретизация**

- Цел: да максимизира информацията, т.е. да избере разделения, които носят най-много яснота относно класа.
- Използва информационна ентропия (както при decision trees).

**Може ли два съседни интервала да имат един и същ клас?**

- Грешен отговор: Не. Това би било вярно, ако използваме само error-based – тя ще се опита да слее такива интервали.
- Правилен отговор: Да. При entropy-based, това е възможно, ако така се оптимизира ентропията. Алгоритъмът може да остави два съседни интервала с един и същи клас, ако разделението между тях помага за по-добро информационно разделяне на останалите.

**Дискретизация** -Превръщане на номинални стойности в числови

- **Indicator attributes, бинарни кодировки**
- Представяне на неравенства, напр.  $temperature < hot$
- По-добре от използване на цели числа (избягва грешна метрика)

**Projections (Проекции)**- вземаме оригиналните данни и ги променяме по определен начин, без да губим основната им същност.

- Обикновени трансформации → голям ефект
- Примери: Разлика между дати; Отношение на числови атрибути; Съединяване на номинални; Добавяне на шум; Случайно премахване на данни; Обфускиране/ скриваме или объркваме данните, без да губим смисъла./

**Principal Component Analysis (PCA)** - Анализ на главните компоненти - метод, който намалява броя на атрибутите (измеренията) в данни, като запазва най-важната информация: Без надзор, за откриване на важни посоки; Намалява размерността

Стъпки: 1/Посока на най-голяма вариация 2/ Следваща перпендикулярна посока и т.н.

- Използва **eigenvectors** на ковариационната матрица
- **Ковариационна матрица** - таблица, която показва как различните атрибути варират заедно. Пример: ако ръстът и теглото на хората се увеличават заедно → имат **висока ковариация**.
- **Eigenvectors (собствени вектори)**- показват **посоките**, в които данните се **разпръскват най-много**. Представи си, че

имаш облак от точки – **eigenvectors** ти казват в **коя посока има най-голямо разнообразие**. **РСА** използва тези посоки, за да проектира данните върху тях и намали размерността, като запази максимална информация

**Random projections** - намаляване на размерността на данни (т.е. намалява броя на атрибутите), като използва случайна матрица вместо изчисления като в **РСА**.

- **РСА** е скъпо (кубично); **Random projections** използват случайни посоки; Запазват разстоянията добре (в среден случай); Подходящи за **kD-trees**; Могат да се използват ансамбли от модели

### **Text to attribute vectors**

- Превръщане на текст в **bag of words**
- Стойности: **binary**,  $f_{ij}$ ,  $\log(1+f_{ij})$ , **TF**  $\times$  **IDF**
- Конфигурации: само букви, делимитри, малки букви, без **stopwords**, само **k** най-чести

**Time series (Времеви редове)** - Всеки запис = различен момент

- Трансформации: Изместване (**shifting**); Разлика (**делта**)
- Ако стойностите са неравномерни: нормализация спрямо времеви стъпки
- Ако атрибутите са различни времеви точки → специфични трансформации

**Automatic data cleansing** - алгоритъм самоизвършва корекция и подобряване на данните, без човешка намеса.

- Подобрене на **decision trees**: Премахване на грешни записи, след това ново обучение; По-добре: човешка проверка

- **Шум в атрибутите vs шум в класовете:** Шум в атрибути → оставя се; Систематичен шум в класове → оставя се; Несистематичен → премахване

**Robust regression** - регресия, която е по-устойчива на грешки и аутлайъри (екстремни стойности) в данните.

- Устойчива статистика = справяне с **отклонения**
- Подходи: Минимизиране на **абсолютна грешка**, не квадратична; Премахване на outliers

**Detecting anomalies** – Откриване на аномалии

- Чрез визуализация или автоматичен подход: използва се комитет от различни алгоритми (напр. decision tree, nearest-neighbor, LDA)
- Подход: консервативен консенсус – премахват се записи, които са неправилно класифицирани от всички модели
- Недостатък: може да засегне редки класове