ML Konzepte: Teil 4 (Unbearbeitete Version)

Hayrettin Acar

December 14, 2023

1 Clustering

Ziel beim Clustering ist es, ähnliche Datenpunkte auf der Grundlage bestimmter Merkmale oder Eigenschaften zu gruppieren. Ziel ist es, inhärente Strukturen oder Muster in den Daten zu erkennen, ohne dass zuvor bestimmte Kategorien bekannt sind. Die Idee ist dabei, dass Punkte innerhalb desselben Clusters einander ähnlicher sind als Punkte in anderen Clustern.

1.1 K-means Verfahren

Gegeben sei ein Datensatz mit $x^{(i)} \in \mathbb{R}^d$, allerdings **ohne** entsprechende Zielvariablen $y^{(i)}$. Die Eingabedaten sollen nun in k verschiedene Gruppen $c_1, ...c_k$ aufgeteilt werden. Das Ziel ist hierbei, die Clusterzentren $\mu_1, ..., \mu_k \in \mathbb{R}^d$ anhand des definierten Kriteriums auf die Eingabedaten zu verteilen und iterativ anzupassen, meist räumlich innerhalb der Datengruppen zu zentrieren. Zunächst wird die Lage aller Cluster initialisiert (z.B. zufällig). Das Verfahren läuft dann in zwei Schritten ab:

1. Ordne den i-ten Datensatz einem entsprechenden Cluster zu, wobei gelten soll:

$$c^{(i)} := \arg\min_j ||x^{(i)} - \mu_j||^2$$

   Jeder Datenpunkt $x^{(i)}$ wird also demjenigen Cluster j zugeordnet, welcher räumlich am nächsten liegt.

2. Verschiebe jedes j-te Clusterzentrum anhand des berechneten Mittelwertes aller zugeordneten Daten,

$$\mu_j := \frac{\sum_{i=1}^m 1\{c^{(i)} = j\} x^{(i)}}{\sum_{i=1}^m 1\{c^{(i)} = j\}}$$

Diese Schritte werden nun solange wiederholt, bis sich die Lage der Zentren stabilisiert hat. Die Kostenfunktion über alle Zentren ergibt sich zu

$$J(c, k) = \sum_{i=1}^{k} ||x^{(i)} - \mu_{c^{(i)}}||^2 \tag{1}$$

wobei $\mu_{c^{(i)}}$ das Zentrum desjenigen Clusters beschreibt, zu welchem das i-te Beispiel zugeordnet wurde. Es kann darüber hinaus gezeigt werden, dass diese (nicht-konvexe) Kostenfunktion immer zu einem lokalen Minimum konvergiert.

1.2 Gaussian Mixture Model (GMM)

Der Grundgedanke eines Gauß'schen Mischungsmodells besteht darin, die Datenverteilung als eine Kombination mehrerer Gauß'scher Verteilungen zu modellieren. Das Modell geht davon aus, dass die Datenpunkte durch eine Mischung dieser Gaußverteilungen erzeugt werden, (1). Neben dem Clustering, werden GMMs auch in der Anomaliedetektion verwendet. Hierbei werden Daten als Ausreißer erkannt, wenn beispielsweise die kombinierte Wahrscheinlichkeit für das Auftreten bestimmter Merkmale relativ gering ist.

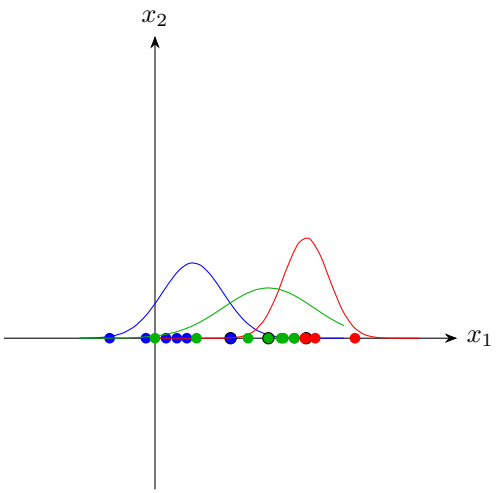

Figure 1: Beispiel für das GMM Clustering. Die Datenpunkte werden jeweils einer Gauss-Verteilung zugeordnet.

Gegeben sei ein Datensatz $\{x^{(1)}, x^{(2)}, .., x^{(m)}\}$, welcher sich auf k unterschiedliche Cluster aufteilt. Es sei die Annahme getroffen, dass sich alle Cluster mithilfe einer Gauß'schen Verteilung modellieren lassen, wobei die Wahrscheinlichkeit, dass der Datenpunkt dem z-ten Cluster entstammt mit π_z beschrieben wird. Die kombinierte Wahrscheinlichkeit für das Auftreten eines Datenpunktes für das z-te Cluster kann angegeben werden mit

$$p(x^{(i)}, z^{(i)}) = p(z^{(i)})p(x^{(i)}|z^{(i)}) = \pi_{z^{(i)}}\mathcal{N}(x|\mu_{z^{(i)}}, \Sigma_{z^{(i)}}).$$

Hierbei beschreibt μ_z den Erwartungswert-Vektor und Σ die Kovarianzmatrix.

Typischerweise ist nun $z^{(i)}$ nicht bekannt (sog. **latente/versteckte** Variable), d.h. die Clusterzuweisung ist anhand der gegebenen Daten nicht observierbar. Das Ziel ist nun die Parameter für jedes Cluster zu bestimmen,

$$\pi = (\pi_1, ..., \pi_k)$$
$$\mu = \mu_1, ..., \mu_k$$
$$\Sigma = \Sigma_1, ..., \Sigma_k$$

Die Likelihood-Funktion lässt sich zunächst bestimmen zu

$$L(\pi, \mu, \Sigma) = \prod_{i=1}^{m} \sum_{z^{(i)}=1}^{k} \pi_{z^{(i)}}\mathcal{N}(x|\mu_{z^{(i)}}, \Sigma_{z^{(i)}}) \tag{2}$$

$$l(\pi, \mu, \Sigma) = \log(L) = \sum_{i=1}^{m} \log\left(\sum_{z^{(i)}=1}^{k} \pi_{z^{(i)}}\mathcal{N}(x|\mu_{z^{(i)}}, \Sigma_{z^{(i)}})\right) \tag{3}$$

Sie beschreibt die kombinierte Wahrscheinlichkeit, dass der gesamte Datensatz mit den angenommenen Parametern auftritt. Wenn nun $z^{(i)}$ gegeben ist, vereinfacht sich die likelihood Funktion erheblich, da nicht mehr die Summenbetrachtung für jeden einzelnen Datensatz notwendig ist,

$$l(\pi, \mu, \Sigma) = \sum_{i=1}^{m} \log \mathcal{N}(x|\mu_{z^{(i)}}, \Sigma_{z^{(i)}}) + \log(\pi_{z^{(i)}}) \tag{4}$$

Die übrigen Parameter lassen sich dann mithilfe einer maximum-likelihood Abschätzung berechnen zu

$$\pi_j = \frac{1}{m}\sum_{i=1}^{m} 1\{z^{(i)} = j\} \tag{5}$$

$$\mu_j = \frac{\sum_{i=1}^{m} 1\{z^{(i)} = j\}x^{(i)}}{\sum_{i=1}^{m} 1\{z^{(i)} = j\}} \tag{6}$$

$$\Sigma_j = \frac{\sum_{i=1}^{m} 1\{z^{(i)} = j\}(x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^T}{\sum_{i=1}^{m} 1\{z^{(i)} = j\}} \tag{7}$$

Wie kann nun eine Parameterabschätzung erfolgen, wenn die latente Variable nicht bekannt ist?

Ein Ansatz ist der sog. EM (**Expectation Maximization**) Algorithmus. Die Idee dahinter ist, die latente Variable zunächst abzuschätzen und daraufhin die Wahrscheinlichkeit der übrigen Variablen zu maximieren. Die Grundlage hierfür ist es, die direkte Zuweisung zwischen Cluster und Datensatz aufzuweichen und durch eine **Gewichtung** zu ersetzen. Hierfür wird eine neue Variable $\gamma_j^{(i)}$ eingeführt, welche für jeden Datensatz i und einem Cluster j angibt, wie hoch die Wahrscheinlichkeit der Zugehörigkeit ist,

$$\gamma_j^{(i)} = p(z = j | x = x^{(i)}; \pi, \mu, \Sigma) \tag{8}$$

$$= \frac{\pi_j \mathcal{N}(x^{(i)} | \mu_j, \Sigma_j)}{\sum_{c=1}^{k} \pi_c \mathcal{N}(x^{(i)} | \mu_c, \Sigma_c)} \tag{9}$$

Der Algorithmus besteht nun aus zwei Schritten. Zunächst werden die Parameter μ, Σ, π initialisiert, anschließend werden die folgenden Schritte durchgeführt:

1. Schritt: Bestimme die Gewichtungsvariable $\gamma_j^{(i)}$
2. Schritt: Schätze die übrigen Parameter neu ab,

$$\pi_j := \frac{1}{m} \sum_{i=1}^{m} \gamma_j^{(i)}$$

$$\mu_j := \frac{\sum_{i=1}^{m} \gamma_j^{(i)} x^{(i)}}{\sum_{i=1}^{m} \gamma_j^{(i)}}$$

$$\Sigma_j := \frac{\sum_{i=1}^{m} \gamma_j^{(i)} (x^{(i)} - \mu_j)(x^{(i)} - \mu_j)^T}{\sum_{i=1}^{m} \gamma_j^{(i)}}$$

Die Parameter werden nun also unter Berücksichtigung der Gewichtungsvariablen γ berechnet. Es kann gezeigt werden, dass der 1. Schritt eine untere Abschätzung für die likelihood-Funktion darstellt, welcher mit dem 2. Schritt iterativ optimiert wird. Mithilfe der **Jensen-Ungleichheit** kann darnn darüber hinaus gezeigt werden, dass die Optimierung der unteren Abschätzung gleichzeitig die Optimierung der likelihood-Funktion darstellt. Daher ist der EM-Algorithmus geeignet, um ein lokale Optimierung durchzuführen.