Kohonenkarte (Vergleich Euklidischer Abstand und KNN-Modell)

Master Elektrotechnik Vertiefung Informationstechnik

Wahlmodul: Künstliche Intelligenz

Lehrbeauftragter: Gregor Braun

Vorgelegt von: Lynne Merveille Nono Makuate

Am 17.06.2025

Inhaltsverzeichnis

1 Einleitung

In der heutigen datengetriebenen Welt ist die Klassifikation ein zentrales Thema im Bereich des maschinellen Lernens. Verschiedene Verfahren ermöglichen es, Muster und Strukturen in Daten zu erkennen und daraus Vorhersagen zu treffen. Dieses Projekt befasst sich mit zwei solchen Verfahren: dem k-nächste-Nachbarn-Algorithmus (KNN), einem klassischen Beispiel für überwachtes Lernen, und der selbstorganisierenden Karte (SOM), einem bekannten Verfahren des unüberwachten Lernens, das von Teuvo Kohonen entwickelt wurde.

Das Ziel dieser Arbeit war es, beide Ansätze anhand desselben Datensatzes, dem berühmten Iris-Datensatz von Fisher, in C# zu implementieren, miteinander zu vergleichen und sowohl numerisch als auch visuell zu bewerten. Dabei wurde ein besonderer Fokus auf die Lernmechanismen, die Genauigkeit sowie die interne Strukturierung der Daten gelegt.

2 Zielsetzung

Die zentrale Fragestellung des Projekts lautete: *Wie effektiv kann eine Kohonenkarte zur Klassifikation verwendet werden, im Vergleich zu einem etablierten überwachten Verfahren wie dem KNN-Algorithmus?*

Um diese Frage zu beantworten, wurden beide Modelle implementiert und auf denselben Datensatz angewendet. Darüber hinaus sollte das Projekt folgende Ziele erfüllen:

- Entwicklung einer modularen C#-Anwendung zur flexiblen Nutzung beider Klassifikatoren.
- Vergleich der Klassifikationsergebnisse anhand von Konfusionsmatrizen und Genauigkeit.
- Visuelle Darstellung der Kohonenkarte zur Analyse der topologischen Organisation.
- Förderung des Verständnisses für die Unterschiede zwischen überwachtem und unüberwachtem Lernen.

3 Methodik

Die Software ist vollständig in der Programmiersprache C# geschrieben und strukturiert in mehrere Klassen unterteilt worden, um eine klare Trennung der Verantwortlichkeiten zu gewährleisten. Im Zentrum stand eine Hauptklasse, **Program.cs**, die den Datenimport, das Training der Modelle und die Ausgabe der Ergebnisse koordinierte.

Der Iris-Datensatz wurde in der Datei IrisData.cs eingebettet, um eine externe Abhängigkeit zu vermeiden. Jeder Eintrag besteht aus vier numerischen Merkmalen und einer Klassenbezeichnung (Setosa, Versicolor oder Virginica).

Das KNN-Modell basiert auf der Idee, dass ein Punkt anhand der Klassen seiner k-nächsten Nachbarn klassifiziert wird. In unserem Fall wurde $k = 3$ gewählt. Der euklidische Abstand wurde als Distanzmaß verwendet.

Die selbstorganisierende Karte (SOM) wurde als zweidimensionales Gitter von Neuronen realisiert. Jedes Neuron trägt ein Gewichtungsvektor, der im Laufe des Trainings an die Eingabemuster angepasst wird. Die SOM arbeitet dabei ohne explizite Klassenlabels, sie organisiert die Daten allein anhand ihrer Merkmalsähnlichkeiten.

4 Entwicklung

4.1 Trainingsprozess

Der Trainingsprozess stellt das Herzstück der selbstorganisierenden Karte (SOM) dar und unterscheidet sich grundlegend von dem des k-nächste-Nachbarn-Algorithmus (KNN). Während KNN kein eigentliches „Training“ benötigt, sondern rein instanzbasiert arbeitet, muss die SOM ihre interne Struktur erst durch wiederholtes Lernen aufbauen.

Im Fall der Kohonenkarte beginnt das Training mit einer zufälligen Initialisierung der Gewichtsvektoren jedes Neurons im zweidimensionalen Gitter. Jeder Vektor hat dieselbe Dimension wie die Eingabedaten (in unserem Fall vier Merkmale pro Iris-Blume).

In jeder Iteration wird ein zufälliges Datenmuster ausgewählt und mit allen Neuronen verglichen, um das sogenannte „Best Matching Unit“ (BMU) zu finden, welches das Neuron mit der geringsten euklidischen Distanz zur Eingabe darstellt. Dieser Schritt ist entscheidend, da er bestimmt, welche Region der Karte durch das aktuelle Beispiel repräsentiert wird. Anschließend werden nicht nur die Gewichte des BMU, sondern auch die der umliegenden Neuronen angepasst. Diese Anpassung erfolgt nach folgender Regel:

```
neuron.Gewichte[i] += lernrate * (eingabe[i] - neuron.Gewichte[i]);
```

Die Lernrate sowie der Radius des Nachbarschaftseinflusses verringern sich mit zunehmender Iterationszahl. Dadurch wird zu Beginn des Trainings ein grobes topologisches Mapping erstellt, welches im Verlauf feiner verfeinert wird. Dieser Mechanismus erlaubt es der SOM, sowohl globale Strukturen als auch lokale Feinheiten der Daten zu erfassen.

Im Gegensatz dazu erfolgt beim KNN keine Modellbildung. Stattdessen wird bei jeder Vorhersage die Distanz zu allen bekannten Trainingspunkten berechnet, und die Mehrheit der k-nächsten Nachbarn entscheidet über die zugewiesene Klasse.

Dieser fundamentale Unterschied im Lernansatz macht den Vergleich zwischen SOM und KNN besonders interessant.

4.2 Ergebnisse und Visualisierung

Nach dem Training der SOM und der Durchführung der KNN-Klassifikation sind beide Modelle auf denselben Datensatz, dem Iris-Datensatz mit 150 Datenpunkten, aufgeteilt in drei gleich große Klassen, getestet worden.

Die erzielten Genauigkeiten waren:

- **KNN**: 95,97 %
- **SOM**: 97,32 %

Diese Werte zeigen eine bemerkenswerte Leistung beider Modelle. Insbesondere fällt auf, dass die Kohonenkarte, obwohl sie ursprünglich nicht zur Klassifikation, sondern zur Datenvisualisierung entwickelt worden ist, eine höhere Genauigkeit als das überwachte KNN-Verfahren erzielen konnte. Dies lässt vermuten, dass die SOM die inhärente Struktur des Iris-Datensatzes sehr gut erlernt hat.

Zusätzlich zur numerischen Analyse wurde eine einfache visuelle Darstellung der Kohonenkarte in der Konsole implementiert. Dabei ist jeder Neuronposition eine abgeleitete Klasse zugewiesen worden(basierend auf den Datenpunkten, für die es das BMU war). Die Karte wird als 2D-Gitter mit kurzen Klassenkürzeln angezeigt:

```
Kohonen-Karte (2D Visualisierung):

I I I . . . S S S S
. . . V V . . S S S
I I . V V . . S S S
I I . . V . . . . S
I I . . . V V . . .
I . I . V V . V V V
I . I . V . . V V V
I . . V . V . V . V
. V I . . V V V . .
I I . . . V V . . V
```

Diese Darstellung verdeutlicht die Clusterbildung und Topologieerhaltung der SOM. So sind beispielsweise „Setosa“-Neuronengruppen deutlich getrennt von „Virginica“-Gruppen sichtbar. Diese Trennung unterstützt die Interpretation und Analyse der Datenstruktur auf eine Art, die KNN allein nicht leisten kann.

Zusätzlich sind Konfusionsmatrizen generiert, die detailliert zeigen, wie viele Datenpunkte korrekt oder falsch klassifiziert wurden, getrennt nach Klassen. Auch hier konnte die SOM in allen drei Klassen sehr stabile Vorhersagen treffen.

5 Diskussion

Die Ergebnisse des Projekts werfen interessante Fragen zum Einsatz und zur Leistungsfähigkeit unterschiedlicher Lernmethoden auf. Trotz der Einfachheit des KNN-Modells und seiner typischerweise hohen Genauigkeit bei kleinen Datensätzen konnte die SOM in diesem Fall bessere Resultate erzielen und das, obwohl sie unüberwacht lernt.

Ein wesentlicher Vorteil der SOM liegt in ihrer Fähigkeit, nicht nur zu klassifizieren, sondern gleichzeitig eine topologische Repräsentation der Daten zu erzeugen. Sie lernt die Ähnlichkeiten zwischen den Eingaben und bewahrt diese in der Anordnung der Neuronen auf der Karte. Dies macht sie besonders wertvoll in Anwendungen, bei denen nicht nur das Ergebnis zählt, sondern auch das Verständnis der Datenstruktur wichtig ist.

Im Gegensatz dazu betrachtet KNN jeden neuen Datenpunkt isoliert. Es gibt keine Lernphase, kein internes Modell. Dies macht KNN besonders anfällig gegenüber Rauschen und Ausreißern, während die SOM durch ihr kontinuierliches Training eine robustere Darstellung aufbauen kann.

Ein möglicher Nachteil der SOM ist ihre Komplexität und Trainingsdauer, insbesondere bei größeren Datensätzen. Auch kann die Wahl der Kartenparameter (Größe, Lernrate, Iterationen) das Ergebnis stark beeinflussen. Diese Hyperparameter müssen oft empirisch bestimmt werden.

Nicht zuletzt zeigt sich auch, dass die Kombination von SOM und KNN denkbar wäre: Man könnte zum Beispiel eine SOM als Vorverarbeitungsschritt verwenden, um die Daten zu clustern und anschließend innerhalb der Cluster einen lokalen KNN anzuwenden.

6 Fazit

Das Projekt hat gezeigt, dass die Kohonenkarte ein leistungsfähiges Werkzeug zur unüberwachten Klassifikation darstellt. In Verbindung mit einer geeigneten Visualisierung liefert sie nicht nur gute Vorhersagen, sondern auch ein besseres Verständnis der inneren Datenstruktur. Der Vergleich mit dem KNN-Algorithmus hat verdeutlicht, dass beide Verfahren ihre Berechtigung haben. KNN als einfaches, schnelles Modell, SOM als lernfähiges, interpretierbares System.

Ausblick

Eine mögliche Erweiterung dieses Projekts könnte darin bestehen, die SOM mit anderen Visualisierungen zu kombinieren (z. B. als Bitmap oder mit Farben), oder sie mit modernen Clustering-Methoden wie t-SNE oder UMAP zu vergleichen. Auch der Einsatz auf größeren, komplexeren Datensätzen wäre denkbar, ebenso wie eine Kombination von SOM und KNN in einem hybriden System.