Zusammenfassung des Subscale-Algorithmus

Jonas Kienzle[1], Pascal Kunkel[2], Pius Horn[3]

Abstract: Die LaTeX-Klasse `lni` setzt die Layout-Vorgaben für Beiträge in LNI Konferenzbänden um. Dieses Dokument beschreibt ihre Verwendung und ist ein Beispiel für die entsprechende Darstellung. Der Abstract ist ein kurzer Überblick über die Arbeit der zwischen 70 und 150 Wörtern lang sein und das Wichtigste enthalten sollte. Die Formatierung erfolgt automatisch innerhalb des abstract-Bereichs.

Keywords: Subscale; DBSCAN;

1 Einleitung

Eine wichtige Herausforderung in Datascience ist das Erkennen der Zusammenhänge von Daten in großen Datenmengen. Diese Erkennung kann in Form von Anhäufungen ähnlicher Daten, in sogenannten Clustern, erfolgen. Das Auffinden dieser Cluster findet unter anderem in den Domänen Biologie, maschinelles Sehen und Astronomie Anwendung. Insbesondere in der Auswertung von personenbezogenen Daten ist Clustering aufgrund der großen Datenmengen, die durch Big Data entstehen, nicht mehr wegzudenken. Das Suchen und Finden von Clustern kann unter anderem in der Medizintechnik genutzt werden, um beispielsweise Risikogruppen zu identifizieren. Hierbei stellt ein Patient, beziehungsweise eine Patientin, einen Datensatz dar. Die jeweiligen Gesundheitsdaten, wie das Blutbild sowie Alter und Geschlecht, definieren die entsprechenden Attribute. Jeder Datensatz ist somit ein Vektor in einem multidimensionalen Raum, wobei jedes Attribut einer Dimension entspricht. Das Ziel ist es, in diesem großen und hochdimensionalen Datensatz, dichte Regionen zu erkennen, die als Cluster definiert werden können. Bei dem hierfür verwendeten Clustering handelt es sich um einen sogenannten unüberwachten Datamining-Prozess, da initial keine Kenntnis über die Datenverteilung bekannt ist. Mit steigender Anzahl von Dimensionen wird es schwierig zusammenhängende Cluster zu ermitteln. Grund hierfür ist, dass bei einer großen Menge von Attributen die jeweiligen einzelnen Attribute geringer gewichtet werden. Infolgedessen versuchen unterschiedliche Algorithmen zum Erkennen von Clustern unwichtige Attribute auszublenden, um diese finden zu können. Anlässlich der wachsenden Anzahl der Dimensionen, wächst auch die Anzahl der möglichen Subspaces exponentiell, wodurch die Suche nach relevanten Subspaces sehr rechenintensiv wird. Hier setzt der Subscale-Algorithmus an, da dieser mit hochdimensionalen Datensets besser wie vergleichbare Algorithmen wie CLIQUE, SUBCLU und INSCY skaliert.

[1] Hochschule Offenburg, EMI, Heimgasse 23a, 77797, Ohlsbach, Deutschland jkienzle@stud.hs-offenburg.de
[2] Hochschule Offenburg, EMI, Im Langenacker 7, 76534, Baden-Baden, Deutschland pkunkel1@stud.hs-offenburg.de
[3] Hochschule Offenburg, EMI, Schubertstraße 5, 77948, Friesenheim, Deutschland phorn2@stud.hs-offenburg.de

2 Subscale Clustering

Dem Subscale Algorithmus liegt ein einfaches geometrisches Prinzip zu Grunde. Dies besteht darin, dass die Projektion eines Clusters in einem höherdimensionalen Raum auf einen Raum mit niederer Dimension ebenfalls Cluster bildet. Der Umkehrschluss gilt hier allerdings nicht. Es kann also nicht gesagt werden, dass in einer gemeinsamen höheren Dimension Clsuter entstehen, weil Cluster in den Teildimensionen entstehen. Dies lässt sich anschaulich an einem Beispiel zeigen. Man stelle sich zwei Punkte vor. Der erste befindet sich im Ursprung. Der zweite Punkt ist in jeder Dimension um die maximal erlaubte Distanz vom Ursprung verschieden. Betrachtet man die Punkte nun als Projektion auf einer Achse stellt man fest, dass die Punkte um die maximal erlaubte Distanz verschieden sind. Sie bilden also gerade noch eine Dense Unit. Sobald allerdings mehrere Dimensionen betrachtet werden ist die Distanz zwischen den Punkte größer, als der zuvor festgelegte Grenzwert. Die Folge davon ist dann, dass die beiden Punkte nicht mehr als Teil der selben Dense Unit angesehen werden dürfen.

Um solche Fälle auszuschließen wird nach dem Suchen der Teilräume, in denen Cluster entstehen könnten, noch der DBScan Algorithmus ausgeführt. DBScan wird durch Subscale einmal für jeder gefundenen Subspace aufgerufen. DBScan beschreibt einen Algorithmus zum Finden von Clustern in einem vorgegebenen Raum. Ein Cluster wird von DBScan durch zwei wesentliche Punkte beschrieben. Der erste ist eine Angabe zu der maximalen Distanz zwischen zwei Datenpunkten. Diese Angabe wird Epsilon genannt und muss von dem Anwender des Algorithmus definiert werden. Sollte die Distanz zweier Datenpunkte kleiner sein, als Epsilon, gelten die Datenpunkt als Nachbarn. Unterscheiden sich die Datenpunkte um mehr als Epsilon gelten die Punkte nicht Nachbarn. Der zweite wichtige Parameter muss auch durch den Nutzer festgelegt werden und stellt eine Mindestgröße für die Anzahl an Nachbarn dar, welche ein Datenpunkt haben muss, um als Kernpunkt zu gelten. Ein Clsuter im DBScan Algorithmus baut sich aus zwei Arten von Punkten auf. Diese sind Kernpunkte und Randpunkte. Kernpunkte sind alle Punkte, welche die entsprechende Mindestanzahl an Nachbarn erfüllen. Randpunkte haben einen Kernpunkt als Nachbarn, allerdings selbst nicht genug Nachbarn, um als Kernpunkt zu zählen. Ein Cluster besteht also aus Kernpunkten und wird durch die Randpunkte begrenzt.

Ein Problem, welches zu hohen Laufzeiten führen kann ist der Vergleich zwischen zwei Dense Units in einem Subspace. Vergleicht man die vorkommenden Punkte einzeln miteinander ist die ein rechenaufwendiger Prozess. Daher wird beim Subscale Algorithmus ein anderer Ansatz gewählt. Dieser Ansatz bedient sich dem Prinzip einer Hash Tabelle, wonach aus einem einzigartigen Schlüssel eine Speicheraddresse errechnet wird, an welcher dann die Daten liegen. Wenn die Art der Schlüsselgenerierung von der Zusammensetzung der entsprechenden Dense Units abhängt, gilt, dass ein Konflikt beim Speichern einer Dense Unit bedeutet, dass diese Kombination von Punkten bereits in der Hash Tabelle gespeichert wurde. Es existiert also eventuell ein Subspace Cluster in einem Raum, welcher aus der entsprechenden gefundenen Dimensionen zusammengesetzt ist. Nun muss noch ein gutes Verfahren gefunden werden, welches die Datenpunkte zu einem eindeutigen

Schlüssel kombiniert. Beim Subscale Algorithmus wird hier auf die Forschung von Lehner und Erdös zurückgegriffen. Im Rahmen deren Arbeit wurde festgestellt, dass die Summe größer Integer mit sehr hoher Wahrscheinlichkeit bei unterschiedlichen Summanten zu einer unterschiedlichen Summe führt. Um dies innerhalb des Subscale Algorithmus zu verwenden wird nun zu Beginn jeder Datenpunkt mit einem möglichst größen Integer Wert als ID gekennzeichnet. Wenn nun ein Cluster in einem Unterraum zu der entsprechenden Hash Tabelle hinzugefügt werden soll, kann die Summe der IDs der entsprechenden Datenpunkte als Schlüssel verwendet werden. Damit hat man nun eine schnelle Möglichkeit, um zu verifizieren, ob kleinere Cluster in mehreren Dimensionen vorkommen.

Bei der Durchführung des Subscale Algorithmus wird nun, wie bereits erwähnt, jedem Datenpunkt eine eindeutige ID in Form einer möglichst großen Ganzzahl zugewiesen. Im Anschluss daran wird eine Hash Tabelle generiert, welche als Schlüssel die Summe der Punkte, welche einer Dense Unit angehören, enthält. Die Werte an jedem Eintrag der Hash Tabelle sind eine Liste an Dimensionen, welche gemeinsam der Subraum bilden und eine Auflistung der Datenpunkte. Die Tabelle sollte anfangs noch leer sein. Nun wird in einer ersten Schleife über alle Dimensionen iteriert. Für jede Dimension wird dann eine Liste der in dem entsprechenden eindimensionalen Raum vorkommenden Dense Units erstellt. –einfügen sub dense unit Sobald die Dense Units generiert sind wird für jede Dense Unit geprüft, ob bereits ein Eintrag in der Hash Tabelle vorhanden ist. Der Schlüssel wird als Summe der IDs der Datenpunkte der Dense Unit berechnet. Wenn es zu einer Kollision beim Zugriff auf die entsprechende Stelle in der Hash Tabelle kommt bedeutet das, dass selben Punkte bereits in einer anderen Dimension eine Dense Unit gebildet haben. Damit wird es wahrscheinlich, dass es in dem gemeinsamen Raum auch ein Cluster gibt. Um dies als Daten zu reprensentieren wird die Liste an Dimensionen in dem aktuellen Eintrag angepasst, indem die aktuelle Dimension hinzugefügt wird. Wenn es zu keiner Kollision kommt bedeutet dies, dass diese Kombination aus Datenpunkten bisher in keiner anderen Dimension eine Dense Unit gebildet haben. In diesem Fall wird ein neuer Eintrag in der Hast Tabelle angelegt. Sobald dies in allen Dimensionen erfolgt ist, werden die Daten umgruppiert, sodass sie über den Subraum zu erreichen sind. Der DBScan Algorithmus wird dann für jeden Subraum erneut ausgeführt, um zu verifizieren, dass alle Cluster valide sind.

2.1 Funktionsweise

3 Bestimmung der Inputparameter

Typischer Weise ist die Dichteverteilung der zu untersuchenden Daten initial unbekannt. Die Qualität der Ergebnisse des Subscale-Algorithmus hängen jedoch stark von den gewählten Input-Parametern ϵ und τ ab, welche in der direkten Abhängigkeit zu der Dichteverteilung stehen. Diese Parameter zu finden stellt eine herausfordernde Aufgabe dar. Bei der Anwendung des Subscale Algorithmus sind die Benutzervariablen ϵ und τ in zwei Arbeitsschritten relevant:

Arbeitsschritt 1

Die Inputparameter werden von Subscale initial zur Ermittlung der 1-D Dichtepunkte benötigt. Mit diesen Dichtepunkten können durch den Subscale Algorithmus die maximalen Subspaces identifiziert werden.

Arbeitsschritt 2

Diese gefundenen maximalen Subspaces sollen anschließend jeweils auf Cluster untersucht werden. Die Analyse dieser Subspaces nach Cluster erfolgt durch den DBSCAN Algorithmus, welcher ebenfalls ϵ und τ als Input Parameter benötigt.

Der *Arbeitsschritt 1* benötigt gegenüber *Arbeitsschritt 2* mehr als 95 % der Rechenleistung [**JournalArticel**]. Grund dafür ist, dass die von DBSCAN zu durchsuchenden Subspaces schon auf diejenigen Punkte reduziert sind, welche mit einer hohen Wahrscheinlichkeit über Cluster verfügen. Die Durchführung von DBSCAN ist somit relativ effizient möglich. In der Literatur finden sich bis dato nach unserem Wissensstand keine Informationen darüber, wie die initialen Inputparameter von Subscale für die Ermittlung der 1-D Dichtepunkte effizient ermittelt werden können.

Für die Wahl der Inputparametern für DBSCAN gibt es in der Literatur hingegen verschiedene heuristische Ansätze. Der naive Ansatz wäre es, bei DBSCAN dieselben Werte für ϵ und τ wie auch bei Subscale zu verwenden. Es ist jedoch besser, wenn die Inputwerte den zu untersuchenden Subspaces individuell angepasst werden. Da die Distanz zwischen den Punkten mit der wachsenden Anzahl an Dimensionen größer wird, ist ϵ in hochdimensionalen Räumen zum Beispiel eher größer zu wählen. Je höher die Dimensionalität, desto schwieriger ist jedoch das Auffinden eines geeigneten ϵ-Wertes, da mit der Dimensionalität gleichzeitig der Kontrast zwischen Distanzen und somit die Aussagekraft der Distanzen abnimmt [**TODS4203-19**]. In der Thesis von Amardeep Kaur über den Subscale Algorithmus [**thesis**] wird daher vorgeschlagen den kleinstmöglichen Abstand zwischen zwei beliebigen 1-D Projektionen von Punkten im gegebenen Datensatz als Wert für ϵ zu wählen. Prinzipiell ist ϵ möglichst klein zu wählen, um lediglich jene Cluster zu finden, welche tatsächlich auf einen Zusammenhang zwischen Daten zurückzuführen sind. Zudem hängt ϵ von der verwendeten Distanzfunktion ab [**TODS4203-19**]. In manchen Fällen kann außerdem Domänenwissen für die Wahl von ϵ von Vorteil sein.

Der Parameter τ ist hingegen einfacher zu bestimmen. Seine Aufgabe ist es die Dichteabschätzung der Cluster zu "glätten". Es wird empfohlen τ auf den doppelten Wert von der Anzahl Dimensionen festzulegen (bei zwei-dimensionalen Daten wäre $\tau = 4$ ein vielversprechender Input). Interessanterweise scheinen die Ergebniscluster eher von ϵ als von τ abhängig zu sein [**TODS4203-19**].

3.1 Metriken zur Erkennung suboptimaler Inputparameter