ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΑΡΑΓΩΓΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ
ΤΟΜΕΑΣ ΟΡΓΑΝΩΣΗΣ ΚΑΙ ΔΙΟΙΚΗΣΗΣ

ΤΜΗΜΑΤΟΠΟΙΗΣΗ ΑΓΟΡΑΣ ΜΕ ΤΗ ΧΡΗΣΗ ΑΛΓΟΡΙΘΜΟΥ BIRCH

Ευαγγελία Ιατρού

Χανιά,2019

Περίληψη

Η διπλωματική αυτή εργασία πραγματεύεται το θέμα της τμηματοποίησης της αγοράς, ένα NP-hard πρόβλημα, με τη χρήση του αλγορίθμου Birch. Αρχικά, μελετάται η έννοια της αγοράς, τα επίπεδά της, η τμηματοποίηση της και τα κριτήρια με τα οποία αυτή πραγματοποιείται. Σημαντικό θέμα επίσης, το οποίο και αναλύεται, είναι οι αλγόριθμοι συσταδοποίησης και η κατηγοριοποίησή τους. Αφού γίνει η ομαδοποίηση των δεδομένων στην πορεία θα αξιολογηθεί και θα αποτιμηθεί. Ακόμη παρουσιάζεται ο αλγόριθμος Birch τον οποίο συγκρίναμε και με άλλους αλγόριθμους ομαδοποίησης. Τέλος, ακολουθεί περιγραφή του dataset Wholesale Customer και πειραματική ανάλυση με βάση αυτό.

Λέξεις κλειδιά: Τμηματοποίηση Αγοράς, Αλγόριθμος Birch, Wholesale Customer σετ δεδομένων

Abstract

This thesis deals with the segmentation of the market, an NP-hard problem, using the Birch algorithm. Initially, the concept of the market, its levels, its segmentation and the criteria by which it is carried out are studied. An important issue, also analysed, is clustering algorithms and their categorization. Once the data is aggregated along the path, it will be evaluated and evaluated. We also present the Birch algorithm, which we compared with other clustering algorithms. Finally, a description of the Wholesale Customer dataset follows and an experimental analysis based on this.

Keywords: Market Segmentation, Birch Algorithm, Wholesale Customer Data Set

Περιεχόμενα

Πίνακας Εικόνων

Εισαγωγή

Η τμηματοποίηση της αγοράς συνιστά ένα από τα πιο κεφαλαιώδη ζητήματα στη σύγχρονη βιβλιογραφία. Για αυτό το λόγο στην παρούσα ερευνητική εργασία επιλέχθηκε η επίλυση του εν λόγω προβλήματος με τη χρήση του αλγόριθμου Birch. Στόχος είναι να αναδειχθούν τα θεωρητικά ζητήματα που σχετίζονται με την τμηματοποίηση της αγοράς και στη συνέχεια καθώς και να εξεταστεί η αποδοτικότητα του προτεινόμενου αλγορίθμου.

Το πρώτο κεφάλαιο της εργασίας αφιερώνεται στην έννοια της αγοράς, στην τμηματοποίηση της, δηλαδή στα κριτήρια, τη διαδικασία και τα οφέλη αυτής, ενώ το δεύτερο κεφάλαιο αφορά τα κριτήρια βάσει των οποίων πραγματοποιείται η τμηματοποίηση. Το δεύτερο και τρίτο κεφάλαιο αναλύουν τα κριτήρια της τμηματοποίησης και την αξιολόγηση της ομαδοποίησης, αντίστοιχα. Εκτενής περιγραφή της ομαδοποίησης δεδομένων υπάρχει στο τέταρτο κεφάλαιο, όπου αναλύονται και συγκεκριμένοι αλγόριθμοι ευρέως διαδεδομένοι, όπως ο k-means.

Το πέμπτο κεφάλαιο είναι και το βασικό της εργασίας και παρουσιάζει το πρακτικό της μέρος, όπου περιγράφεται ο αλγόριθμος Birch και όλη η διαδικασία του μεθοδολογικού πλαισίου. Το τελευταίο κεφάλαιο περιέχει τη συζήτηση και τα συμπεράσματα τόσο της βιβλιογραφίας όσο και των αποτελεσμάτων του πρακτικού μέρους της παρούσας εργασίας.

1. Η τμηματοποίηση της αγοράς

Σε αυτό το κεφάλαιο περιγράφεται αναλυτικά η έννοια της αγοράς καθώς επίσης και της τμηματοποίησης της αγοράς. Δίνονται επίσης πληροφορίες σχετικά με τη διαδικασία και τα κριτήρια τμηματοποίησης της αγοράς καταναλωτών, τονίζοντας τα οφέλη αυτής της διαδικασίας.

1.1. Η έννοια της αγοράς

Η έννοια της αγοράς μπορεί να ενέχει διάφορους ορισμούς. Για παράδειγμα, μπορεί να ορίζεται ως η επιχείρηση ή το εμπόριο συγκεκριμένου προϊόντος, συμπεριλαμβανομένων των χρηματοπιστωτικών προϊόντων. Παράλληλα, ενδέχεται να εννοείται ως ένα μέρος συνάντησης ατόμων που αγοράζουν και πουλούν προϊόντα. Επιπρόσθετα, χρήσιμος ορισμός καθίσταται η διαδικασία με την οποία καθορίζονται οι τιμές των αγαθών και των υπηρεσιών. Αυτός ο ορισμός προκύπτει από το γεγονός ότι οι αγορές διευκολύνουν το εμπόριο και επιτρέπουν τη διανομή και την κατανομή πόρων σε μια κοινωνία. Επιπλέον, οι αγορές επιτρέπουν την αξιολόγηση και την τιμολόγηση οποιουδήποτε εμπορεύσιμου προϊόντος (Yankelovich & Meer, 2006). Από αυτά, συνεπάγεται ότι οι περισσότερες αγορές βασίζονται στους πωλητές που προσφέρουν τα αγαθά ή τις υπηρεσίες τους (συμπεριλαμβανομένης της εργασίας) έναντι χρημάτων από τους αγοραστές.

Μια αγορά μπορεί να προκύψει αυθόρμητα ή να κατασκευαστεί σκόπιμα, προκειμένου να καταστεί δυνατή η ανταλλαγή υπηρεσιών και αγαθών. Ωστόσο, οι αγορές διαφέρουν ανάλογα με τα προϊόντα (αγαθά, υπηρεσίες) ή τους παράγοντες (εργασία και κεφάλαιο), που εξυπηρετούν τη διαδικασία πώλησης. Επιπλέον διαφορές υπάρχουν στα γεωγραφικά όρια μιας αγοράς. Για παράδειγμα, η αγορά τροφίμων σε ένα μόνο κτίριο, η αγορά ακινήτων σε μια μικρή πόλη, η καταναλωτική αγορά σε ολόκληρη τη χώρα ή η οικονομία ενός διεθνούς εμπορικού μπλοκ, όπου

ισχύουν οι ίδιοι κανόνες σε όλα τα κράτη-μέλη. Φυσικά, υπάρχουν και παγκόσμιες αγορές, όπως είναι το εμπόριο διαμαντιών. Ταυτόχρονα, οι αγορές που χαρακτηρίζονται ως αναπτυγμένες ή αναπτυσσόμενες αφορούν τις εθνικές οικονομίες.

Ειδικότερα, στις βασικές οικονομικές αρχές, η έννοια της αγοράς σχετίζεται με οποιαδήποτε δομή, που επιτρέπει στους αγοραστές και τους πωλητές να ανταλλάσσουν κάθε είδους αγαθά, υπηρεσίες και πληροφορίες. Αυτό αποδεικνύεται από την άποψη ότι η οποιαδήποτε ανταλλαγή αγαθών ή υπηρεσιών, με ή χωρίς χρήματα, αποτελεί μια συναλλαγή. Συνεπώς, ένα σημαντικό θέμα της μελέτης των οικονομικών αφορά τους συμμετέχοντες στην αγορά, που αποτελούνται από όλους τους αγοραστές και τους πωλητές ενός αγαθού, οι οποίοι επηρεάζουν την τιμή του. Αυτή η διαπίστωση, έχει οδηγήσει στη δημιουργία αρκετών θεωριών και μοντέλων, σχετικών με τις βασικές δυνάμεις της αγοράς της προσφοράς και της ζήτησης (Yankelovich & Meer, 2006).

Από τα παραπάνω προκύπτει η συζήτηση για το πόσο μια συγκεκριμένη αγορά μπορεί να θεωρηθεί ως μια "ελεύθερη αγορά", δηλαδή, μια αγορά ελεύθερη από την κυβερνητική παρέμβαση. Έτσι, η μικροοικονομία επικεντρώνεται, παραδοσιακά, στη μελέτη της δομής της αγοράς και στην αποτελεσματικότητα της ισορροπίας της αγοράς. Όταν η τελευταία -εφόσον υπάρξει- δεν είναι αποτελεσματική, τότε οι οικονομολόγοι κάνουν λόγο για ανεπάρκεια αγοράς. Ωστόσο, δεν είναι πάντοτε σαφής ο τρόπος βελτίωσης κατανομής των πόρων, λόγω της πιθανότητας κυβερνητικής αποτυχίας.

1.2. Τμηματοποίηση και τα επίπεδα της αγοράς

Ο κατακερματισμός της αγοράς είναι ένας όρος μάρκετινγκ, ο οποίος αφορά την ομαδοποίηση των υποψήφιων αγοραστών ανά τμήματα. Αυτά

τα τμήματα αφορούν τις κοινές ανάγκες, ενώ παράλληλα ανταποκρίνονται παρόμοια σε μια ενέργεια μάρκετινγκ. Συνεπώς, η τμηματοποίηση της αγοράς επιτρέπει στις εταιρείες να στοχεύουν σε διαφορετικές κατηγορίες καταναλωτών, οι οποίοι αντιλαμβάνονται με διάφορους τρόπους την πλήρη αξία ορισμένων προϊόντων και υπηρεσιών. Επιπλέον, η διαδικασία του κατακερματισμού αποτελεί μια επέκταση της έρευνας αγοράς με στόχο να εντοπίσει συγκεκριμένες ομάδες-στόχους καταναλωτών, ούτως ώστε να σχεδιαστούν τα προϊόντα και η επωνυμία με ελκυστικό τρόπο για την ομάδα (Rokach & Maimon, 2008).

Πιο συγκεκριμένα, ο βασικός στόχος της κατάτμησης της αγοράς είναι η ελαχιστοποίηση του κινδύνου για την εταιρεία. Αυτό εξασφαλίζεται μέσω του καθορισμού των προϊόντων, που έχουν τις καλύτερες πιθανότητες να αποκτήσουν ένα μερίδιο μιας συγκεκριμένης αγοράς-στόχου. Παράλληλα, καθορίζεται ο καλύτερος τρόπος παράδοσης προϊόντων στην αγορά. Αυτό επιτρέπει την αύξηση της συνολικής αποτελεσματικότητάς της εταιρείας, χρησιμοποιώντας τους περιορισμένους πόρους της στις προσπάθειες, που αποφέρουν την καλύτερη απόδοση των επενδύσεων.

Για την κατανόηση των παραπάνω, καθίσταται χρήσιμη η αναφορά κάποιων παραδειγμάτων τμηματοποίησης της αγοράς. Αρχικά, τέτοια παραδείγματα εντοπίζονται στα προϊόντα, στο μάρκετινγκ και στη διαφήμιση που χρησιμοποιούν οι άνθρωποι καθημερινά. Πιο συγκεκριμένα, οι κατασκευαστές αυτοκινήτων βασίζονται στην ικανότητά τους να εντοπίζουν σωστά τα τμήματα της αγοράς. Εν συνεχεία, δημιουργούν προϊόντα και διαφημιστικές καμπάνιες που απευθύνονται σε αυτά τα τμήματα (Yankelovich & Meer, 2006).

Άλλο ένα παράδειγμα αποτελούν οι παραγωγοί δημητριακών, οι οποίοι ασχολούνται ενεργά με τρία ή τέσσερα τμήματα της αγοράς. Για παράδειγμα, προωθούν τις παραδοσιακές μάρκες και τα υγιή εμπορικά

τους σήματα σε καταναλωτές, που αντιλαμβάνονται την αξία τους αναλογικά με τους ηλικιωμένους. Παράλληλα, ενισχύουν την αφοσίωση των νεότερων καταναλωτών, συνδέοντας τα προϊόντα τους, με δημοφιλή κινηματογραφικά θέματα.

Από την άλλη, σχετικά με τις εταιρείες αθλητικών παπουτσιών αναφέρεται πως διαθέτουν τμήματα αγοράς για τους παίκτες καλαθοσφαίρισης και τους δρομείς μεγάλων αποστάσεων. Είναι διαφανές ότι πρόκειται για ξεχωριστές ομάδες. Συνεπώς, οι παίκτες καλαθοσφαίρισης και οι δρομείς μεγάλων αποστάσεων ανταποκρίνονται σε πολύ διαφορετικές διαφημίσεις.

Κάθε αγορά χωρίζεται σε τμήματα για να μπορέσει μια εταιρεία να αναπτύξει το καταλληλότερο σχέδιο μάρκετινγκ. Ειδικότερα, κάθε επίπεδο ενός τμήματος της αγοράς απαιτεί διαφορετικές πληροφορίες και διαφορετική προσέγγιση μάρκετινγκ. Τα επίπεδα ενός τμήματος της αγοράς μπορούν να αναλυθούν στα επίπεδα παγκόσμιας, εξειδικευμένης, τοπικής και ατομικής αγοράς. Πιο συγκεκριμένα, σύμφωνα με τους Yankelovich & Meer (2006) τα επίπεδα ενός τμήματος της αγοράς χωρίζονται ως εξής:

i. <u>Παγκόσμια αγορά:</u> Ένα παγκόσμιο τμήμα της αγοράς είναι εκείνο το τμήμα του πληθυσμού, που ταιριάζει σε ένα γενικό δημογραφικό προφίλ του κοινού-στόχου. Πρόκειται για ένα ολοκληρωμένο επίπεδο τμηματοποίησης που περιέχει γενικές πληροφορίες σχετικά με τις ιδιαιτερότητες του κοινού, όπως η ηλικία, το μέσο εισόδημα, η γεωγραφική κατανομή και τα πρότυπα αγοράς. Το τμήμα της παγκόσμιας αγοράς δεν είναι διαχωρισμένο ή κατακερματισμένο με κάποιο τρόπο. Αντίθετα, αναλύεται ως ομάδα με γενικευμένες συμπεριφορές, ώστε να ταιριάζουν σε ένα προφίλ μάρκετινγκ.

ii. Εξειδικευμένη αγορά: Μια εξειδικευμένη ομάδα είναι μια ομάδα καταναλωτών που έχουν καθορισμένες προτιμήσεις προϊόντων. Για παράδειγμα, αν πωλούνται αθλητικά αυτοκίνητα, τότε μπορεί να διαπιστωθεί ότι η παγκόσμια αγορά-στόχος είναι οι άνδρες ηλικίας 18 έως 55 ετών.

iii. Τοπική αγορά: Η βαθύτερη παρατήρηση στα επίπεδα ενός τμήματος της αγοράς, οδηγεί στον εντοπισμό του εξειδικευμένου κοινού ενός προϊόντος. Έτσι, τα τοπικά τμήματα της αγοράς χρησιμοποιούνται για να καθορίσουν την κατεύθυνση του στόχου μιας στρατηγικής μάρκετινγκ. Επίσης, καθορίζουν και τα σημεία, όπου οι καταναλωτικές ανάγκες δύνανται να είναι οι μεγαλύτερες. Για παράδειγμα, εάν διαπιστωθεί ότι η πλειοψηφία των καταναλωτών που θέλουν κόκκινο αυτοκίνητο, όπως αναφέρθηκε παραπάνω, βρίσκονται στις νότιες Ηνωμένες Πολιτείες, τότε οι διαφημιστικές καμπάνιες θα περιλαμβάνουν κόκκινα σπορ αυτοκίνητα. Επιπλέον, θα προωθηθούν όσο γίνεται περισσότερα κόκκινα σπορ αυτοκίνητα στη συγκεκριμένη γεωγραφική περιοχή.

iv. Ατομική αγορά: Το τελικό επίπεδο της τμηματοποίησης της αγοράς ασχολείται με τις καταναλωτικές συνήθειες των μεμονωμένων ατόμων. Αυτό το επίπεδο αφορά κυρίως τη συλλογή δεδομένων από ιδιώτες. Αυτό στοχεύει στη συλλογή απαραίτητων δεδομένων για την καλύτερη κατανόηση της γενικής σύνθεσης του παγκόσμιου τμήματος. Λόγω αυτού, οι πωλητές και οι εκπρόσωποι εξυπηρέτησης πελατών διατηρούν επαφή με τους πελάτες σε ατομικό επίπεδο. Έτσι, εξασφαλίζεται και διατηρείται η εμπιστοσύνη του εμπορικού σήματος και των επαναλαμβανόμενων δραστηριοτήτων (Choi, Cha & Tappert, 2010).

1.3. Κριτήρια τμηματοποίησης της αγοράς καταναλωτών

Κατανοώντας ότι η αγορά δεν είναι ομοιογενής και χωρίζεται σε τμήματα, θεωρείται ότι, ακόμα και με έναν απλό διαχωρισμό της εταιρείας, θα προκύψουν διαφορετικές χρήσεις των προϊόντων ή των υπηρεσιών. Από την άλλη, η τμηματοποίηση της εταιρείας προϋποθέτει ότι όλα τα τμήματα θα χρησιμοποιούν πανομοιότυπους ανθρώπους με πανομοιότυπες αξίες. Είναι ιδιαίτερα χρήσιμο να κατηγοριοποιηθούν τα κριτήρια τμηματοποίησης της αγοράς καταναλωτών, τα οποία σύμφωνα με τους Rokach & Maimon (2008), είναι τα εξής:

i. *Δημογραφική τμηματοποίηση*

Αν και τα δημογραφικά κριτήρια δεν μπορούν να θεωρηθούν ως τμήμα, διαδραματίζουν πολύ σημαντικό ρόλο στην τμηματοποίηση. Αυτά αποτελούν τις γενικές πληροφορίες για τους πελάτες, οι οποίες χρησιμοποιούνται για να καθορίσουν το ιδιαίτερο προφίλ χαρακτηριστικών των πελατών, που αντιστοιχούν σε κάθε τμήμα. Με άλλα λόγια, τα δημογραφικά κριτήρια βοηθούν στην κατάλληλη αναγνώριση τμήματος, όπου ανήκει ο κάθε πελάτης. Αυτή η αναγνώριση εξυπηρετεί με τη σειρά της τον ορθό τρόπο προσέγγισης των πελατών. Στην ομάδα των δημογραφικών κριτηρίων περιλαμβάνονται ειδικότερα τα εξής: 1. Ηλικία 2. Φύλο 3. Εισόδημα 4. Επάγγελμα 5. Επίπεδο μόρφωσης 6. Καταγωγή (ή εθνικότητα, ή φυλή) 7. Θρησκεία 8. Οικογενειακή κατάσταση (έγγαμος, άγαμος, αριθμός παιδιών) 9. Κοινωνική τάξη 10. Τόπος κύριας κατοικίας

ii. *Γεωγραφική τμηματοποίηση*

Αυτό το κριτήριο κατακερματισμού της αγοράς ομαδοποιεί τους ανθρώπους με βάση την περιοχή διαμονής. Δηλαδή, οι δυνητικοί πελάτες θα έχουν διαφορετικές ανάγκες βάσει της περιοχής στην οποία βρίσκονται. Οι άνθρωποι, που βρίσκονται σε μη αστικές περιοχές,

ενδέχεται να χρειάζονται ένα άλλο μοντέλο από το προϊόν σε σχέση με εκείνους που βρίσκονται σε αστικές περιοχές. Έτσι, προκύπτουν διαφορετικές ανάγκες ανάλογα με την περιοχή. Επίσης στις κρύες χώρες, μια εταιρεία μπορεί να προωθήσει θερμοσίφωνες, ενώ στις θερμές χώρες, η ίδια εταιρεία μπορεί να προωθήσει κλιματιστικά. Κατ' επέκταση, πολλές εταιρείες χρησιμοποιούν τη γεωγραφική κατάτμηση ως βάση για την κατάτμηση της αγοράς.

Η γεωγραφική κατάτμηση είναι ο ευκολότερος τύπος τμηματοποίησης, αλλά στην πραγματικότητα χρησιμοποιήθηκε την τελευταία δεκαετία, όπου οι βιομηχανίες ήταν νέες και η εμβέλεια ήταν μικρότερη. Σήμερα, αν και η εμβέλεια είναι υψηλή, εξακολουθούν να χρησιμοποιούνται γεωγραφικές αρχές κατακερματισμού. Αυτό συμβαίνει σε περιπτώσεις επέκτασης μιας επιχείρησης σε περισσότερες τοπικές περιοχές, καθώς και σε διεθνείς περιοχές (Choi, Cha & Tappert, 2010).

iii. *Συμπεριφοριστική τμηματοποίηση*

Ένα άλλο σημαντικό κριτήριο κατακερματισμού της αγοράς χωρίζει τον πληθυσμό βάσει συμπεριφοράς, χρήσης και λήψης αποφάσεων. Ένα παράδειγμα τμηματοποίησης, που βασίζεται στη συμπεριφορά είναι ότι οι νέοι θα προτιμήσουν διαφορετική μάρκα σαπούνι σε σχέση με έναν αθλητή. Με βάση τη συμπεριφορά ενός ατόμου, το προϊόν διατίθεται στο εμπόριο. Ένα άλλο παράδειγμα τμηματοποίησης της συμπεριφοράς είναι το μάρκετινγκ κατά τη διάρκεια των γιορτών. Την περίοδο εκείνη, τα μοντέλα αγορών είναι εντελώς διαφορετικά σε σχέση με τα πρότυπα αγοράς των υπόλοιπων ημερών.

Αυτός ο τύπος κατακερματισμού της αγοράς βρίσκεται σε εξέλιξη, ιδιαίτερα στην αγορά έξυπνων τηλεφώνων. Για παράδειγμα, το Blackberry αφορούσε χρήστες που ήταν επιχειρηματίες, η Samsung απευθυνόταν σε

χρήστες που τους αρέσει το Android, ενώ η Apple απευθύνθηκε σε premium πελάτες, που επιθυμούν να νιώθουν μοναδικοί και δημοφιλείς.

iv. Ψυχογραφική τμηματοποίηση

Η ψυχογραφική κατάτμηση είναι αυτή που χρησιμοποιεί τον τρόπο ζωής των ανθρώπων, τις δραστηριότητές τους, τα ενδιαφέροντά τους καθώς και τις απόψεις τους, για τον ορισμό ενός τμήματος της αγοράς. Ο ψυχογραφικός κατακερματισμός είναι αρκετά παρόμοιος με τον κατακερματισμό της συμπεριφοράς. Ωστόσο, η πρώτη λαμβάνει επίσης υπόψη τις ψυχολογικές πτυχές της καταναλωτικής συμπεριφοράς αγορών. Αυτές οι ψυχολογικές πτυχές μπορεί να είναι ο τρόπος ζωής του καταναλωτή ή και η κοινωνική του θέση. Για παράδειγμα, τα καταστήματα Zara ασχολούνται αποκλειστικά με τον τρόπο ζωής. Δηλαδή, οι πελάτες, οι οποίοι επιθυμούν τα πιο πρόσφατα και ξεχωριστά είδη ένδυσης, μπορούν να επισκεφθούν τα καταστήματα του (Rokach & Maimon, 2008).

1.4. Διαδικασία τμηματοποίησης της αγοράς

Όπως έχει ήδη αναφερθεί, ο κατακερματισμός της αγοράς είναι σημαντικός για κάθε επιχείρηση. Ωστόσο, τα στάδια του κατακερματισμού της αγοράς είναι εξίσου σημαντικά, ώστε να καθοριστεί η τελική αγορά-στόχος (Yankelovich & Meer, 2006). Αυτά τα στάδια αφορούν τον καθορισμό της στρατηγικής της εταιρείας, τον εντοπισμό πηγών εσόδων ή κερδών, τον προσανατολισμό στην πραγματική συμπεριφορά των καταναλωτών, την αξιολόγηση ή πρόβλεψη αλλαγών στην αγορά ή στη συμπεριφορά των καταναλωτών και την επεκτασιμότητα των τμημάτων. Πιο συγκεκριμένα σύμφωνα με τους Yankelovich & Meer (2006), είναι τα εξής:

1. *Καθορισμός της στρατηγικής της εταιρείας*

Μόλις βρεθεί ένας τομέας, που είναι κερδοφόρος και επεκτάσιμος, θα πρέπει να ενσωματωθεί στη στρατηγική μάρκετινγκ. Αν αναλογιστεί κάποιος τον τρόπο, που οι εταιρείες McDonalds ή KFC έγιναν τόσο μεγάλες αλυσίδες γρήγορου φαγητού, θα αντιληφθεί ότι είχαν μια πολύ ξεκάθαρη διαδικασία κατακερματισμού. Αυτή οδήγησε στην ευκολότερη εύρεση περιοχών προς στόχευση.

Με τα βήματα της κατάτμησης της αγοράς, τα τμήματα γίνονται σαφή και στη συνέχεια ενσωματώνονται άλλες μεταβλητές της στρατηγικής μάρκετινγκ, σύμφωνα με το στοχευόμενο τμήμα. Επίσης, υπάρχει η δυνατότητα τροποποίησης των προϊόντων, όπως να διατηρηθεί η βέλτιστη τιμή, να ενισχυθεί η διανομή και ο τόπος, ώστε να προωθηθεί με σαφήνεια το κοινό-στόχος. Έτσι, η λειτουργία της επιχείρησης γίνεται πιο απλή λόγω της διαδικασίας κατακερματισμού της αγοράς.

2. *Εντοπισμός των πηγών εσόδων ή κερδών*

Είναι απαραίτητο να αναγνωριστεί ποιο από τα τμήματα είναι πιο κερδοφόρο. Αυτό είναι, επίσης, ένα ακόμα βήμα στόχευσης στη διαδικασία της τμηματοποίησης. Για παράδειγμα, ο Ιταλός ιδιοκτήτης ενός εστιατορίου αποφασίζει ότι βγάζει μεγάλο κέρδος από τους μεσήλικες, αλλά μικρό από τους νέους. Οι νέοι προτιμούν το γρήγορο φαγητό και τους αρέσει η κοινωνικοποίηση. Έτσι, παραγγέλνουν λιγότερα, ξοδεύοντας περισσότερο χρόνο στο τραπέζι, μειώνοντας έτσι την κερδοφορία (Yankelovich & Meer, 2006). Για να τροποποιηθεί αυτή η νοοτροπία και το τμήμα να γίνει περισσότερο κερδοφόρο, ο εκάστοτε ιδιοκτήτης πρέπει να αναλογιστεί κάποια βασικά κριτήρια.

Συνεπώς, καθίσταται απαραίτητος ο προσδιορισμός των αξιών, των στάσεων και των πεποιθήσεων των καταναλωτών, αφού συνδέονται συγκεκριμένα με το προϊόν ή τις υπηρεσίες, που τους προσφέρονται.

Επιπλέον, πρέπει να εντοπιστούν οι ανάγκες των πελατών και ο τρόπος ομαδοποίησής τους ανάλογα με τις ανάγκες τους. Αυτό καθίσταται εφικτό αν η εταιρεία αναλογιστεί την κατανάλωση από την οπτική των πελατών, ούτως ώστε να αντιληφθεί την επιθυμία του καθενός (Yankelovich & Meer, 2006). Για παράδειγμα, αν σε μια περιοχή υπάρχουν πολλά εστιατόρια, αλλά δεν υπάρχει ιταλικό εστιατόριο ή δεν υπάρχει αλυσίδα γρήγορου φαγητού, πρέπει να εντοπιστεί η ανάγκη δημιουργίας τέτοιων αν το επιθυμούν οι καταναλωτές.

Επιπλέον, η ελκυστικότητα της επιχείρησης εξαρτάται από τον ανταγωνισμό, που διατίθεται στον εκάστοτε τομέα. Δηλαδή, εάν ο ανταγωνισμός είναι υψηλός σε ένα δεδομένο τμήμα, τότε δεν έχει νόημα να ληφθεί υπόψη αυτό το τμήμα. Λαμβάνοντας το παραπάνω παράδειγμα του ιταλικού εστιατορίου, αν ο ιδιοκτήτης αντιληφθεί ότι έχει περισσότερους μεσήλικες και νέους στην περιοχή του, είναι προτιμότερο να προωθεί το κατάστημα τα Σαββατοκύριακα. Το ίδιο συμβαίνει και με τα εμπορικά κέντρα, όπου οι νέοι άνθρωποι τα προτιμούν περισσότερο τα Σαββατοκύριακα. Από την άλλη, οι μεσήλικες μπορούν να φέρουν να επισκεφτούν καταστήματα με τα παιδιά τους – ή χωρίς αυτά- οποιαδήποτε ημέρα. Συνεπώς, ο πρώτος στόχος είναι η μεσήλικα ομάδα και ο δεύτερος στόχος είναι οι νέοι (Yankelovich & Meer, 2006).

3. *Εστίαση στην πραγματική συμπεριφορά των πελατών*

Μετά την αναγνώριση των αναγκών των πελατών, πρέπει να προσδιοριστεί ποιοι θα είναι οι πελάτες, οι οποίοι θα επιλέξουν ένα προϊόν- στόχο σε σχέση με κάποιο άλλο αντίστοιχο. Απλά, πρέπει να καθοριστεί το είδος τμηματοποίησης, που επρόκειτο να χρησιμοποιηθεί σε αυτήν την περίπτωση. Λαμβάνοντας υπόψη το παραπάνω παράδειγμα του ιταλικού εστιατορίου, ο στόχος θα είναι παιδιά, νέοι και μεσήλικες. Το ιταλικό φαγητό, γενικά, δεν προτιμάται από τους ηλικιωμένους

ανθρώπους, που προτιμούν τρόφιμα τα οποία μπορούν να μασήσουν ευκολότερα.

4. *Αξιολόγηση ή πρόβλεψη αλλαγών στην αγορά ή στη συμπεριφορά των καταναλωτών*

Εν συνεχεία, μόλις εντοπιστούν τα πιο κερδοφόρα τμήματα μέσω των βημάτων της κατάτμησης της αγοράς, τότε πρέπει να τοποθετηθεί το προϊόν-στόχος στο μυαλό των καταναλωτών. Αυτό εκκινεί από τη βασική ιδέα ότι η επιχείρηση πρέπει να δώσει αξία στα προϊόντα της. Πρέπει να καθοριστεί η ακριβής αξία του προϊόντος για τον πελάτη και το ποσοστό αξιοπιστίας του εμπορικού σήματος.

Για παράδειγμα, ο Ιταλός ιδιοκτήτης εστιατορίου εντόπισε ότι οι νέοι δεν του αποφέρουν κέρδος. Σε αυτήν την περίπτωση πρέπει να δημιουργήσει και να ξεκινήσει μια αλυσίδα γρήγορου φαγητού δίπλα στο ιταλικό εστιατόριο. Αυτό που συμβαίνει είναι ότι, αν και η περιοχή διαθέτει άλλα εστιατόρια γρήγορου φαγητού, το εστιατόριο του είναι το μόνο που προσφέρει καλή ιταλική κουζίνα και ένα καλό εστιατόριο γρήγορου φαγητού δίπλα της. Έτσι, τόσο η ομάδα-στόχος από τους μεσήλικες όσο και οι νέοι μπορούν να απολαύσουν νόστιμο φαγητό.

5. *Επεκτασιμότητα των τμημάτων*

Επομένως, αν εντοπιστεί ένα τμήμα με ελλείψεις, τότε θα πρέπει να διαμορφωθεί κατάλληλα, ούτως ώστε η επιχείρηση να μπορεί να επεκταθεί με τον επιλεγμένο τύπο τμηματοποίησης. Στις περιπτώσεις, που ένα τμήμα είναι πολύ εξειδικευμένο, τότε η επιχείρηση θα τελειώσει τη πορεία της σε εύθετο χρόνο. Έτσι, σύμφωνα με το παραπάνω παράδειγμα, ο ιδιοκτήτης του ιταλικού εστιατορίου θα κατευθυνθεί και σε άλλους γεωγραφικούς τομείς σε άλλες περιοχές, όπου μπορεί να δημιουργήσει την ίδια ιδέα και να επεκτείνει την επιχείρησή του

(Yankelovich & Meer, 2006). Φυσικά, η μεγαλύτερη επέκταση αποφέρει περισσότερα κέρδη.

1.5. Οφέλη τμηματοποίησης

Σύμφωνα με τα παραπάνω, προκύπτουν ποικίλα οφέλη από την τμηματοποίηση της αγοράς. Περιληπτικά, μπορούν να αναφερθούν με το τρόπο που τα ορίζουν οι Hofstede et al. (1999), δηλαδή:

1. Κατανόηση των επιλογών των καταναλωτών
2. Εντοπισμός των τμημάτων των καταναλωτών
3. Αύξηση των ανταγωνιστικών επιλογών (πχ προσφορά περισσότερων προϊόντων)
4. Αποφυγή του πολέμου τιμών
5. Βελτίωση της ποιότητας των υπηρεσιών
6. Ενδυνάμωση της επικοινωνίας
7. Εστίαση σε ότι είναι σημαντικό για τους καταναλωτές
8. Χτίσιμο της αφοσίωσης των πελατών
9. Βελτίωση της επιτυχίας της φίρμας

2. Κριτήρια τμηματοποίησης

Δεδομένου ότι η τμηματοποίηση είναι η ομαδοποίηση παρόμοιων περιπτώσεων / αντικειμένων, απαιτείται κάποιο μέτρο, το οποίο μπορεί να καθορίζει εάν δύο αντικείμενα είναι όμοια ή ανόμοια. Για να καθοριστεί αυτό, υπάρχουν δύο βασικοί τύποι μέτρων διαθέσιμοι προς χρήση. Ο πρώτος αφορά τα μέτρα απόστασης και ο δεύτερος τα μέτρα ομοιότητας. Ένα χαρακτηριστικό μπορεί να είναι γραμμικό ή ονομαστικό και ένα γραμμικό χαρακτηριστικό μπορεί να είναι συνεχές ή διακριτό.

2.1. Μέτρα Απόστασης

Είναι χρήσιμο να δηλώνεται η απόσταση μεταξύ δύο περιπτώσεων x_i και x_jως: d (x_i, x_j). Ένα έγκυρο μέτρο απόστασης πρέπει να είναι συμμετρικό και να αποκτά την ελάχιστη τιμή (συνήθως μηδέν) στην περίπτωση πανομοιότυπων φορέων. Το μέτρο απόστασης ονομάζεται μέτρο μέτρησης απόστασης, εάν ικανοποιεί επίσης τις ακόλουθες παραμέτρους:
Ανισότητα τριγώνου

$$d\ (x_i, x_k) \leq d\ (x_i, x_j) + (x_j, x_k)\ \forall x_i x_j x_k \in S \qquad [2.1]$$

$$d\ (x_i, x_j) = 0 \Rightarrow x_i = x_j\ \forall x_i x_j \in S \qquad [2.2]$$

i. Minkowski: ***Μέτρα απόστασης για αριθμητικά χαρακτηριστικά*** (Oded Maimon, Lior Rokach, 2008).

Ορίζονται δύο αντικείμενα p-διαστάσεων

$$x_i = (x_{i1}, x_{i2}, x_{i3}, \ldots, x_{ip}) \text{ και } x_j = (x_{j1}, x_{j2}, x_{j3}, \ldots, x_{jp}) \qquad [2.3]$$

Η απόσταση μεταξύ τους μπορεί να υπολογιστεί από την παρακάτω σχέση:

$$\boldsymbol{d_{g,ij} = (|x_{i1} - x_{j1}|^g + |x_{i2} - x_{j2}|^g + \ldots + |x_{ip} - x_{jp}|^g)^{\frac{1}{g}}} \qquad [2.4]$$

Η συνήθης ευκλείδεια απόσταση μεταξύ δύο αντικειμένων επιτυγχάνεται για g = 2. Αν θεωρηθεί g= 1, το άθροισμα των απόλυτων ευκλείδειων αποστάσεων (μετρικό του Μανχάταν) λαμβάνεται και με $g = \infty$, παίρνοντας τη μεγαλύτερη από τις παραστάσιες αποστάσεις (Απόσταση του Chebychev).

Η μονάδα μέτρησης, που χρησιμοποιείται, μπορεί να επηρεάσει την ανάλυση ομαδοποίησης. Για να αποφευχθεί η εξάρτηση από την επιλογή των μονάδων μέτρησης, τα δεδομένα πρέπει να τυποποιηθούν. Αυτό συμβαίνει, γιατί τυποποιώντας τις μετρήσεις επιχειρείται να δοθεί σε όλες τις μεταβλητές ίσο βάρος. Ωστόσο, αν σε κάθε μεταβλητή έχει εκχωρηθεί βάρος με βάση τη σημασία της, τότε η σταθμισμένη απόσταση μπορεί να υπολογιστεί ως εξής:

$d_{g,ij}= (w_1|x_{i1} - x_{j1}|^g + w_2 |x_{i2} - x_{j2}|^g + \dots + w_p |x_{ip} - x_{jp}|^g)^{\frac{1}{g}}$ όπου $w_i \in [0,\infty)$ [2.5]

ii. ***Μέτρα απόστασης για δυαδικά χαρακτηριστικά*** (Choi Seung-Seok, Cha, Sung-Hyuk, Tappert Charles, 2010)

Τα μέτρα δυαδικής ομοιότητας και ανομοιότητας διαδραματίζουν κρίσιμο ρόλο σε προβλήματα ανάλυσης προτύπων όπως ταξινόμηση, ομαδοποίηση κλπ. Δεδομένου ότι η απόδοση εξαρτάται από την επιλογή ενός κατάλληλου μέτρου, πολλοί ερευνητές έχουν καταβάλει πολύπλοκες προσπάθειες για να εντοπίσουν τις πιο σημαντικές δυαδικές ομοιότητες και μέτρα απόστασης. Έτσι, έχουν προταθεί πολυάριθμα μέτρα δυαδικής ομοιότητας και μέτρα απόστασης σε διάφορα πεδία.

Στην περίπτωση δυαδικών χαρακτηριστικών, η απόσταση μεταξύ αντικειμένων μπορεί να υπολογιστεί με βάση από τον πίνακα συνάφειας (contingency table). Στην περίπτωση αυτή, χρησιμοποιώντας τον απλό

συντελεστή αντιστοίχισης μπορεί να εκτιμηθεί η ανομοιότητα μεταξύ δύο αντικειμένων:

$$\mathbf{d}\ (\boldsymbol{x_i}, \boldsymbol{x_j}) = \frac{r+s}{q+r+s+t}, \qquad [2.6]$$

όπου q είναι ο αριθμός των χαρακτηριστικών, ο οποίος ισούται με 1. Τα δύο αντικείμενα i και j, t είναι ο αριθμός των χαρακτηριστικών, που ισούται με 0 και για τα δύο αντικείμενα i και j, s είναι ο αριθμός των χαρακτηριστικών, που ισούται με 0 για το αντικείμενο i, αλλά ίσο 1 για το αντικείμενο j, ενώ r είναι ο αριθμός των χαρακτηριστικών, που ισούται με 1 για το αντικείμενο i, αλλά ίσο με 0 για το αντικείμενο j.

Ένα δυαδικό χαρακτηριστικό είναι ασύμμετρο, εάν τα χαρακτηριστικά του δεν είναι εξίσου σημαντικά (συνήθως το θετικό αποτέλεσμα θεωρείται πιο σημαντικό). Στην περίπτωση αυτή, ο παρονομαστής αγνοεί τις ασήμαντες αρνητικές αντιστοιχίες (t). Αυτό ονομάζεται συντελεστής Jaccard και υπολογίζεται από τη σχέση:

$$\mathbf{d}\ (\boldsymbol{x_i}, \boldsymbol{x_j}) = \frac{r+s}{q+r+s} \qquad [2.7]$$

iii. ***Μέτρα απόστασης για ονομαστικά χαρακτηριστικά*** (Choi Seung-Seok, Cha, Sung-Hyuk, Tappert Charles, 2010)

Ένα ονομαστικό ή συμβολικό χαρακτηριστικό είναι ένα διακριτό γνώρισμα, του οποίου οι τιμές δεν βρίσκονται απαραιτήτως σε οποιαδήποτε γραμμική σειρά. Όταν τα χαρακτηριστικά είναι ονομαστικά, μπορούν να χρησιμοποιηθούν δύο κύριες προσεγγίσεις:

Απλή αντιστοίχιση:

$$\boldsymbol{d}\ (\boldsymbol{x_i}, \boldsymbol{x_j}) = \frac{p-m}{m}, \qquad [2.8]$$

όπου m είναι ο αριθμός των αντιστοιχιών (δηλ. ο αριθμός των χαρακτηριστικών για τα οποία i και j είναι στην ίδια κατάσταση), και p είναι ο συνολικός αριθμός των χαρακτηριστικών που περιγράφουν τα

αντικείμενα. Υπάρχει η δυνατότητα να δοθεί μεγαλύτερη βαρύτητα στην αύξηση των αποτελεσμάτων του m ή στις αντιστοιχίσεις των χαρακτηριστικών, που έχουν μεγάλο αριθμό καταστάσεων. Η δημιουργία της δυαδικής αναπαράστασης και ο υπολογισμός της ανομοιογένειας τους, περιγράφεται στον παραπάνω τύπο.

v. Μέτρα απόστασης για τα χαρακτηριστικά γραμμής

Όταν τα χαρακτηριστικά είναι κανονικά, η ακολουθία των τιμών είναι σημαντική. Σε αυτές τις περιπτώσεις, τα χαρακτηριστικά μπορούν να αντιμετωπιστούν ως αριθμητικά, αφού χαρτογραφηθεί η περιοχή τους σε [0,1]. Η χαρτογράφηση αυτή μπορεί να πραγματοποιηθεί ως εξής:

$$\boldsymbol{z_{i,n} = \frac{r_{i,n} - 1}{M_n - 1}}, \qquad [2.9]$$

όπου zi, είναι η κανονικοποιημένη τιμή του χαρακτηριστικού an του αντικειμένου i, ri,n, που είναι η τιμή πριν την κανονικοποίηση και Mn είναι το άνω όριο του χαρακτηριστικού an με δεδομένο ότι το κάτω όριο είναι η τιμή 1.

vi. Μετρήσεις απόστασης για χαρακτηριστικά μεικτού τύπου

Σε πολλές εφαρμογές, κάθε απόσταση σε ένα σύνολο δεδομένων περιγράφεται από περισσότερους από έναν τύπους χαρακτηριστικών. Στην περίπτωση αυτή, τα μέτρα ομοιότητας και ανομοιότητας, που περιεγράφηκαν παραπάνω, δεν μπορούν να εφαρμοστούν απευθείας σε αυτό το είδος δεδομένων.

Συνεπώς, η ανομοιογένεια d (x_i, x_j) μεταξύ δύο περιπτώσεων, που περιέχουν p χαρακτηριστικά μικτού τύποι, ορίζεται ως εξής:

$$\mathbf{d}\ (\boldsymbol{x_i, x_j}) = \frac{\sum_{n=1}^{p} \boldsymbol{\delta_{ij}^{(n)} d_{ij}^{(n)}}}{\sum_{n=1}^{p} \boldsymbol{\delta_{ij}^{(n)}}}, \qquad [2.10]$$

όπου ο δείκτης $\delta_{i\ j}^{(n)} = 0$, αν μια από τις τιμές λείπει.

Η συνεισφορά του χαρακτηριστικού n στην απόσταση μεταξύ των δύο αντικειμένων είναι $d^{(n)}$ (x_i, x_j). Εάν το χαρακτηριστικό είναι δυαδικό ή κατηγορικό, τότε $d^{(n)}(x_i, x_j) = 0$, εάν $x_{in} = x_{ij}$, αλλιώς $d^{(n)}$ $(x_i, x_j) = 1$. Αν το χαρακτηριστικό έχει συνεχόμενη αξία, $d_{ij}^{(n)} = \left|\frac{x_{in} - x_{jn}}{max_h x_{hn} - min_h x_{hn}}\right|$, όπου h τρέχει πάνω από όλα τα αντικείμενα, που δεν λείπουν για το χαρακτηριστικό n. Εάν το χαρακτηριστικό είναι κανονικό, υπολογίζονται οι τυποποιημένες τιμές του χαρακτηριστικού πρώτα και στη συνέχεια ο $z_{i,n}$ θεωρείται συνεχής.

2.2. Συναρτήσεις Ομοιότητας

Ένα εναλλακτικό κριτήριο τμηματοποίησης, είναι η χρήση της συνάρτησης ομοιότητας ***s(xi,xj)***, η οποία συγκρίνει τα διανύσματα xi και xj. Η συγκεκριμένη συνάρτηση πρέπει να έχει μία μεγάλη τιμή, όταν τα δύο διανύσματα είναι όμοια και τη μέγιστη τιμή, όταν τα δύο αυτά διανύσματα είναι πανομοιότυπα.

i. *Μέτρο Συνημίτονου*

Όταν η γωνία μεταξύ των δύο διανυσμάτων είναι ένα σημαντικό μέτρο της ομοιότητάς τους, το κανονικοποιημένο εσωτερικό τους γινόμενο μπορεί να είναι ένα κατάλληλο μέτρο ομοιότητας:

$$\mathbf{sim}(x,y) = \frac{\boldsymbol{x}\cdot\boldsymbol{y}}{\|\boldsymbol{x}\|\cdot\|\boldsymbol{y}\|}, \qquad [2.11]$$

όπου $\|x\|$ είναι η Ευκλείδεια νόρμα του διανύσματος $x=(x_1, x_2,\ldots, x_p)$.

ii. *Μέτρο συσχέτισης Pearson*

$$s\,(x_i, x_j) = \frac{(x_i-\bar{x}_i)^T\cdot(x_j-\bar{x}_j)}{\|x_i-\bar{x}_i\|\cdot\|x_j-\bar{x}_j\|} \qquad [2.12]$$

iii. *Εκτεταμένο μέτρο Jaccard*

$$s\,(x_i, x_j) = \frac{x_i^T\cdot x_j}{\|x_i\|^2+\|x_j\|^2 - x_i^T\cdot x_j} \qquad [2.13]$$

iv. Dice Coefficient Measure

$$s(x_i, x_j) = \frac{2x_i^T \cdot x_j}{\|x_i\|^2 + \|x_j\|^2} \quad [2.14]$$

3. Αξιολόγηση και αποτίμηση Ομαδοποίησης

Η αξιολόγηση (ή η "επικύρωση") των αποτελεσμάτων της συσταδοποίησης είναι τόσο δύσκολη όσο και η ίδια η συσταδοποίηση. Δημοφιλείς προσεγγίσεις περιλαμβάνουν την «εσωτερική» αξιολόγηση, όπου η συσταδοποίηση συνοψίζεται σε έναν ενιαίο βαθμό ποιότητας, την «εξωτερική» αξιολόγηση, όπου η συσταδοποίηση συγκρίνεται με μία υπάρχουσα ταξινόμηση «βασικής αλήθειας» (ground truth), την «χειροκίνητη» αξιολόγηση από έναν εμπειρογνώμονα και την «έμμεση» αξιολόγηση όπου αξιολογείται η χρησιμότητα της συσταδοποίησης στην εφαρμογή για την οποία προορίζεται.

Τα μέτρα εσωτερικής αξιολόγησης υποφέρουν από το πρόβλημα του ότι αναπαριστούν συναρτήσεις που μπορούν να θεωρηθούν ως στόχος συσταδοποίησης. Για παράδειγμα, το σύνολο δεδομένων θα μπορούσε να συσταδοποιηθεί με τον συντελεστή Silhouette, ωστόσο δεν υπάρχει γνωστός αποτελεσματικός αλγόριθμος για αυτό. Χρησιμοποιώντας ένα τέτοιο εσωτερικό μέτρο αξιολόγησης, μάλλον συγκρίνεται η ομοιότητα των προβλημάτων βελτιστοποίησης και όχι απαραίτητα το πόσο χρήσιμη είναι η συσταδοποίηση (Petrovic, 2006).

Η εξωτερική αξιολόγηση παρουσιάζει παρόμοια προβλήματα: εάν έχουμε τέτοιες ετικέτες "βασικής αλήθειας", τότε δεν θα χρειαζόταν να συσταδοποιήσουμε· και σε πρακτικές εφαρμογές συνήθως δεν έχουμε

τέτοιες ετικέτες. Από την άλλη πλευρά, οι ετικέτες αντικατοπτρίζουν μόνο ένα πιθανό διαμερισμό του συνόλου δεδομένων, το οποίο δεν σημαίνει ότι δεν υπάρχει διαφορετική, και ίσως ακόμα καλύτερη, συσταδοποίηση.

Επομένως, καμία από αυτές τις προσεγγίσεις δεν μπορεί εν τέλει να κρίνει την πραγματική ποιότητα μιας συσταδοποίησης, καθώς χρειάζεται η ανθρώπινη αξιολόγηση, η οποία είναι εξαιρετικά υποκειμενική. Παρ' όλα αυτά, τέτοιες στατιστικές μπορούν να είναι αρκετά πληροφοριακές όσον αφορά τον εντοπισμό κακών συσταδοποιήσεων, αλλά δεν πρέπει να απορρίπτεται η υποκειμενική ανθρώπινη αξιολόγηση (Petrovic, 2006).

3.1. Εσωτερική αξιολόγηση

Όταν ένα αποτέλεσμα συσταδοποίησης αξιολογείται με βάση τα δεδομένα της ίδιας συσταδοποίησης, αυτό ονομάζεται εσωτερική αξιολόγηση. Αυτές οι μέθοδοι συνήθως εκχωρούν την καλύτερη βαθμολογία στον αλγόριθμο που παράγει συστάδες με υψηλή ομοιότητα εντός μίας συστάδας και χαμηλή ομοιότητα μεταξύ συστάδων. Ένα μειονέκτημα της χρήσης εσωτερικών κριτηρίων στην αξιολόγηση συστάδων είναι ότι οι υψηλές βαθμολογίες σε ένα εσωτερικό μέτρο δεν οδηγούν αναγκαστικά σε αποτελεσματικές εφαρμογές ανάκτησης πληροφοριών. Επιπλέον, αυτή η αξιολόγηση είναι προκατειλημμένη ως προς αλγορίθμους που χρησιμοποιούν το ίδιο μοντέλο συστάδων (Liu et al., 2010). Για παράδειγμα, η συσταδοποίηση *KMeans* φυσικά βελτιστοποιεί τις αποστάσεις αντικειμένων και ένα εσωτερικό κριτήριο που βασίζεται στην απόσταση θα υπερεκτιμήσει πιθανώς την προκύπτουσα συσταδοποίηση.

Επομένως, τα μέτρα εσωτερικής αξιολόγησης είναι καλύτερα προσαρμοσμένα για να παρέχουν ένα είδος κατανόησης σε καταστάσεις όπου ένας αλγόριθμος αποδίδει καλύτερα από έναν άλλο, αλλά αυτό δεν σημαίνει ότι ένας αλγόριθμος παράγει πιο έγκυρα αποτελέσματα από έναν άλλον. Η εγκυρότητα, όπως μετράται από ένα τέτοιο δείκτη, εξαρτάται από τον ισχυρισμό ότι αυτό το είδος δομής υπάρχει στο σύνολο δεδομένων. Ένας αλγόριθμος που έχει σχεδιαστεί για ένα είδος μοντέλων δεν έχει καμία πιθανότητα εάν το σύνολο δεδομένων περιέχει ένα ριζικά διαφορετικό σύνολο μοντέλων ή εάν η αξιολόγηση χρησιμοποιεί ένα ριζικά διαφορετικό κριτήριο. Για παράδειγμα, η συσταδοποίηση *KMeans* μπορεί να βρει μόνο κυρτές συστάδες οι οποίες προϋποτίθενται από πολλούς δείκτες αξιολόγησης. Σε ένα σύνολο δεδομένων με μη κυρτές συστάδες, η χρήση τόσο *KMeans* όσο και ενός κριτηρίου αξιολόγησης που προϋποθέτει κυρτότητα, δεν είναι ασφαλής (Liu et al., 2010).

Υπάρχουν αρκετά μέτρα εσωτερικής αξιολόγησης που συνήθως βασίζονται στη διαίσθηση ότι τα στοιχεία που ανήκουν στην ίδια συστάδα θα πρέπει να είναι πιο όμοια συγκριτικά με τα στοιχεία των διαφορετικών συστάδων. Για παράδειγμα, οι ακόλουθες μέθοδοι μπορούν να χρησιμοποιηθούν για να εκτιμηθεί η ποιότητα αλγορίθμων συσταδοποίησης βάσει εσωτερικού κριτηρίου:

- **Συντελεστής σιλουέτας**

Ο συντελεστής σιλουέτας αντιπαραθέτει τη μέση απόσταση των στοιχείων της ίδιας συστάδας με τη μέση απόσταση των στοιχείων άλλων συστάδων. Τα αντικείμενα με υψηλή τιμή σιλουέτας θεωρούνται καλά συσταδοποιημένα ενώ αντικείμενα με χαμηλή τιμή μπορεί να είναι υπερβολικά μεγάλα. Αυτός ο δείκτης λειτουργεί καλά με την

συσταδοποίηση *KMeans* και χρησιμοποιείται επίσης για τον προσδιορισμό του βέλτιστου αριθμού συστάδων.

- **Bic index**

Το κριτήριο πληροφοριών Bayesian (BIC) έχει σχεδιαστεί για την αποφυγή υπερφόρτωσης και ορίζεται ως:

$$\boldsymbol{BIC = ln(L) + v\, ln(n)}, \qquad [3.1]$$

όπου n είναι ο αριθμός των αντικειμένων, L είναι η πιθανότητα των παραμέτρων να παράγουν δεδομένα στο μοντέλο, και v είναι ο αριθμός ελεύθερων παραμέτρων στο μοντέλο Gauss. Ο BIC δείκτης λαμβάνει υπόψη τόσο την προσαρμογή του μοντέλου στα δεδομένα όσο και την πολυπλοκότητά του. Η απόδοσή του καθορίζεται από το πόσο μικρό BIC έχει, γιατί αυτό θα είναι το καλύτερο.

- **Calinski-Harabasz index**

Αυτός ο δείκτης υπολογίζεται από τον παρακάτω τύπο:

$$\mathbf{CH} = \frac{\boldsymbol{traceS_B}}{\boldsymbol{traceS_w}} \cdot \frac{\boldsymbol{n_p - 1}}{\boldsymbol{n_p - k}}, \qquad [3.2]$$

όπου (S_B) είναι η μήτρα διασποράς μεταξύ των συστάδων, (S_w) η εσωτερική μήτρα διασποράς, n_p ο αριθμός των συγκεντρωμένων δειγμάτων και k ο αριθμός των συστάδων.

- **Dunn index**

$$\mathbf{Dunn} = \boldsymbol{min_{1 \le i \le c}} \left\{ \boldsymbol{min} \left\{ \frac{\boldsymbol{d(c_i, c_j)}}{\boldsymbol{max_{1 \le k \le c}(d(x_k))}} \right\} \right\}, \qquad [3.3]$$

όπου ως $d(c_i, c_j)$ ορίζουμε την απόσταση μεταξύ των δύο συστάδων x_i και x_j. Το $d(X_k)$ αντιπροσωπεύει την εσωτερική απόσταση της συστάδας (X_k)

και *c* είναι ο αριθμός του συνόλου των δεδομένων. Το μοντέλο με μεγάλο δείκτη Dunn, είναι καλύτερο.

3.2. Εξωτερική αξιολόγηση

Στην εξωτερική αξιολόγηση, τα αποτελέσματα της συσταδοποίησης αξιολογούνται βάσει δεδομένων που δεν χρησιμοποιήθηκαν για συσταδοποίηση, όπως οι γνωστές ετικέτες κλάσης και τα εξωτερικά «σημεία αναφοράς» (benchmarks). Αυτά τα σημεία αναφοράς αποτελούνται από ένα σύνολο προ-ταξινομημένων αντικειμένων και αυτά τα σύνολα δημιουργούνται συχνά από ανθρώπους (ειδικούς). Έτσι, τα σύνολα αναφοράς μπορούν να θεωρηθούν ως ένας χρυσός κανόνας για αξιολόγηση. Αυτοί οι τύποι μεθόδων αξιολόγησης υπολογίζουν πόσο κοντά είναι η συσταδοποίηση στις προκαθορισμένες κλάσεις αναφοράς. Ωστόσο, πρόσφατα συζητήθηκε αν αυτό είναι κατάλληλο για πραγματικά δεδομένα ή μόνο για συνθετικά σύνολα δεδομένων με πραγματική βασική αλήθεια, αφού οι κλάσεις μπορούν να περιέχουν εσωτερική δομή, τα υπάρχοντα χαρακτηριστικά μπορεί να μην επιτρέπουν τον διαχωρισμό των συστάδων ή οι κλάσεις ενδέχεται να περιέχουν ανωμαλίες. Επιπλέον, από την άποψη της ανακάλυψης γνώσης, η αναπαραγωγή γνωστής γνώσης δεν είναι απαραίτητα το επιδιωκόμενο αποτέλεσμα. Στο ειδικό σενάριο περιορισμένης συσταδοποίησης, όπου η μετα-πληροφορία (όπως οι ετικέτες κλάσης) χρησιμοποιείται ήδη στη διαδικασία συσταδοποίησης, η διατήρηση πληροφοριών για σκοπούς αξιολόγησης είναι μη τετριμμένη (Petrovic, 2006).

Ορισμένα μέτρα προσαρμόζονται από παραλλαγές που χρησιμοποιούνται για την αξιολόγηση έργων ταξινόμησης. Αντί για τον υπολογισμό του αριθμού των φορών που η κλάση έχει αντιστοιχιστεί

σωστά σε ένα μοναδικό σημείο δεδομένων (γνωστό ως πραγματικά θετικά), τέτοιες μετρικές μέτρησης ζευγών αξιολογούν εάν κάθε ζεύγος σημείων δεδομένων που είναι πραγματικά στην ίδια συστάδα προβλέπεται να είναι στην ίδια συστάδα. Όπως και με την εσωτερική αξιολόγηση, υπάρχουν αρκετά εξωτερικά μέτρα αξιολόγησης, για παράδειγμα (Liu et al., 2010):

Καθαρότητα: Η καθαρότητα είναι ένα μέτρο του βαθμού στον οποίο οι συστάδες περιέχουν μία μόνο κλάση. Ο υπολογισμός του μπορεί να θεωρηθεί ως εξής: Για κάθε συστάδα, μετρήστε τον αριθμό των σημείων δεδομένων από την πιο κοινή κλάση στην εν λόγω ομάδα. Τώρα πάρτε το άθροισμα επί όλων των συστάδων και διαιρέστε με το συνολικό αριθμό των σημείων δεδομένων. Τυπικά, δεδομένων ορισμένων συνόλων συστάδων M και κάποιου συνόλου κλάσεων D, που και τα δύο διαμερίζουν N σημεία δεδομένων, η καθαρότητα μπορεί να οριστεί ως:

$$\frac{1}{N}\sum_{m\in M}\max_{d\in D}|m\cap d| \qquad [3.4]$$

Σημειώστε ότι αυτό το μέτρο δεν 'ποινικοποιεί' την ύπαρξη πολλών συστάδων. Έτσι, για παράδειγμα, μία βαθμολογία καθαρότητας 1 είναι δυνατή με την τοποθέτηση κάθε σημείου δεδομένων στη δική του συστάδα. Επίσης, η καθαρότητα δεν λειτουργεί καλά στην περίπτωση μη ισορροπημένων δεδομένων: εάν ένα σύνολο δεδομένων μεγέθους 1000 αποτελείται από δύο κλάσεις, μία κλάση περιέχει 999 σημεία και η άλλη μόνο ένα σημείο (Amigó, 2009). Ανεξάρτητα από το πόσο κακά ένας αλγόριθμος συσταδοποίησης αποδίδει, θα δίνει πάντα μία πολύ υψηλή τιμή καθαρότητας.

• **Μέτρο Rand (William M. Rand)**

Ο δείκτης Rand υπολογίζει πόσο παρόμοιες είναι οι συστάδες (που επιστρέφονται από τον αλγόριθμο συσταδοποίησης) ως προς τις «ταξινομήσεις αναφοράς» (benchmark classifications). Μπορούμε επίσης να δούμε τον δείκτη Rand ως μέτρο του ποσοστού των σωστών αποφάσεων που λαμβάνονται από τον αλγόριθμο. Μπορεί να υπολογιστεί χρησιμοποιώντας τον ακόλουθο τύπο:

$$RI = \frac{TP + TN}{TP + FP + FN + TN} \qquad [3.5]$$

όπου TP είναι ο αριθμός των πραγματικών θετικών, TN είναι ο αριθμός των πραγματικών αρνητικών, FP είναι ο αριθμός των ψευδώς θετικών, και FN είναι ο αριθμός των ψευδώς αρνητικών. Ένα ζήτημα με τον δείκτη Rand είναι ότι τα ψευδώς θετικά και τα ψευδώς αρνητικά είναι εξίσου σταθμισμένα. Αυτό μπορεί να είναι ένα ανεπιθύμητο χαρακτηριστικό για ορισμένες εφαρμογές συσταδοποίησης. Το μέτρο F αντιμετωπίζει την ανησυχία αυτή, όπως και ο διορθωμένος από τυχαιότητα, προσαρμοσμένος δείκτης Rand.

• **Δείκτης Jaccard**

Ο δείκτης Jaccard χρησιμοποιείται για να ποσοτικοποιήσει την ομοιότητα μεταξύ δύο συνόλων δεδομένων και παίρνει μία τιμή μεταξύ 0 και 1. Ένας δείκτης 1 σημαίνει ότι τα δύο σύνολα δεδομένων είναι πανομοιότυπα και ένας δείκτης από 0 δηλώνει ότι τα σύνολα δεδομένων δεν έχουν κοινά στοιχεία. Ο εν λόγω δείκτης ορίζεται από τον ακόλουθο τύπο:

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{TP}{TP + FP + FN} \qquad [3.6]$$

Αυτός είναι απλά ο αριθμός των μοναδικών στοιχείων που είναι κοινά και στα δύο σύνολα διαιρούμενος με τον συνολικό αριθμό των μοναδικών στοιχείων και στα δύο σύνολα.

Επίσης, σημειώνεται ότι ο TN δεν λαμβάνεται υπόψη και μπορεί να ποικίλει από 0 και πάνω χωρίς περιορισμό.

4. Ομαδοποίηση Δεδομένων

4.1. Ερμηνεία Ομαδοποίησης

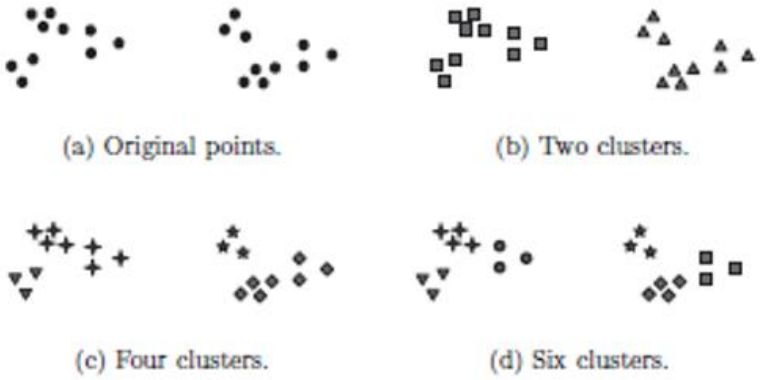
Η ανάλυση συστάδων χωρίζει τα δεδομένα σε ομάδες (συστάδες) που έχουν νόημα, είναι χρήσιμες ή και τα δύο. Εάν ο στόχος είναι οι ομάδες που έχουν νόημα, τότε οι συστάδες θα πρέπει να συλλάβουν τη φυσική δομή των δεδομένων. Σε ορισμένες περιπτώσεις, ωστόσο, η ανάλυση συστάδων είναι μόνο ένα χρήσιμο σημείο εκκίνησης για άλλους σκοπούς, όπως η σύνοψη δεδομένων. Είτε για κατανόηση είτε για χρησιμότητα, η ανάλυση συστάδων έχει παίξει πολύ σημαντικό ρόλο σε ένα μεγάλο εύρος πεδίων: στην ψυχολογία και σε άλλες κοινωνικές επιστήμες, τη βιολογία, τη στατιστική, την αναγνώριση προτύπων, την ανάκτηση πληροφοριών, τη μηχανική μάθηση και την εξόρυξη δεδομένων (Gan et al., 2007).

Η συσταδοποίηση για την κατανόηση κλάσεων, ή εννοιολογικά σημαντικών ομάδων αντικειμένων που έχουν κοινά χαρακτηριστικά, διαδραματίζει σημαντικό ρόλο στον τρόπο με τον οποίο οι άνθρωποι αναλύουν και περιγράφουν τον κόσμο. Πράγματι, τα ανθρώπινα όντα είναι εξειδικευμένα στη διαίρεση αντικειμένων σε ομάδες

(συσταδοποίηση) και στην κατανομή συγκεκριμένων αντικειμένων σε αυτές τις ομάδες (ταξινόμηση). Για παράδειγμα, ακόμα και σχετικά μικρά παιδιά μπορούν γρήγορα να επισημάνουν τα αντικείμενα σε μια φωτογραφία όπως τα κτίρια, τα οχήματα, οι άνθρωποι, τα ζώα, τα φυτά κλπ. Στο πλαίσιο της κατανόησης δεδομένων, οι συστάδες είναι πιθανές κλάσεις και η ανάλυση συστάδων είναι η μελέτη των τεχνικών αυτόματης εύρεσης κλάσεων (Aggarwal et al., 2013).

Η ανάλυση συστάδων ομαδοποιεί αντικείμενα δεδομένων βασιζόμενη μόνο σε πληροφορίες ευρισκόμενες στα δεδομένα που περιγράφουν τα αντικείμενα και τις σχέσεις τους. Ο στόχος είναι τα αντικείμενα μιας ομάδας να είναι παρόμοια (ή να συσχετίζονται) το ένα με το άλλο και να διαφέρουν από (ή να μην σχετίζονται με) τα αντικείμενα σε άλλες ομάδες. Όσο μεγαλύτερη είναι η ομοιότητα (ή ομοιογένεια) μέσα σε μια ομάδα και η διαφορά μεταξύ των ομάδων, τόσο καλύτερη ή πιο διακριτή είναι η συσταδοποίηση. Σε πολλές εφαρμογές, η έννοια μιας συστάδας δεν είναι καλά ορισμένη. Για την καλύτερη κατανόηση της δυσκολίας του να αποφασίσει κανείς το τι συνιστά μία συστάδα, εξετάστε το παρακάτω Σχήμα, το οποίο δείχνει είκοσι σημεία και τρεις διαφορετικούς τρόπους διαίρεσης τους σε συστάδες. Τα σχήματα των δεικτών υποδεικνύουν την ένταξη σε συστάδες. Τα Σχήματα (β) και (δ) χωρίζουν τα δεδομένα σε δύο και έξι μέρη, αντίστοιχα. Ωστόσο, η φαινομενική κατανομή των δύο μεγαλύτερων ομάδων σε τρεις υποσυστάδες μπορεί απλά να είναι ένα επίπλαστο αποτέλεσμα της λειτουργίας του ανθρώπινου οπτικού συστήματος. Επίσης, μπορεί να μην είναι παράλογο να πούμε ότι τα σημεία σχηματίζουν τέσσερις συστάδες, όπως φαίνεται στο Σχήμα (γ). Αυτό το σχήμα δείχνει ότι ο ορισμός μιας συστάδας είναι ασαφής και ότι ο καλύτερος ορισμός εξαρτάται από τη φύση των δεδομένων και τα επιθυμητά αποτελέσματα (Abbas, 2008). Η ανάλυση συστάδων σχετίζεται

με άλλες τεχνικές που χρησιμοποιούνται για τη διαίρεση αντικειμένων δεδομένων σε ομάδες. Για παράδειγμα, η συσταδοποίηση μπορεί να θεωρηθεί μια μορφή ταξινόμησης καθώς δημιουργεί μία επισήμανση αντικειμένων με ετικέτες κλάσης (συστάδας).



Εικόνα 1 – Συσταδοποίηση

4.2. Αλγόριθμοι Συσταδοποίησης

Η συσταδοποίηση είναι ένα έργο για το οποίο έχουν προταθεί πολλοί αλγόριθμοι. Καμία τεχνική συσταδοποίησης δεν είναι καθολικά εφαρμόσιμη και διαφορετικές τεχνικές είναι προτιμώμενες για διαφορετικούς σκοπούς συσταδοποίησης. Έτσι, απαιτείται η κατανόηση τόσο του προβλήματος συσταδοποίησης όσο και της τεχνικής συσταδοποίησης για την εφαρμογή μίας κατάλληλης μεθόδου σε ένα δεδομένο πρόβλημα (Abbas, 2008).

Παραμετρικός σχεδιασμός: Μπορούν (αλλά δεν χρειάζεται) να γίνουν υποθέσεις σχετικά με τη μορφή της κατανομής που χρησιμοποιήθηκε για τη μοντελοποίηση των δεδομένων από την ανάλυση συστάδων. Ο παραμετρικός σχεδιασμός θα πρέπει να επιλέγεται ανάλογα με τη φύση των δεδομένων. Είναι συχνά βολικό να υποθέσουμε, για παράδειγμα, ότι

τα δεδομένα μπορούν να μοντελοποιηθούν από μία πολυπαραγοντική «γκαουσιανή» (Gaussian) κατανομή.

Θέση, μέγεθος, σχήμα και πυκνότητα των συστάδων: Ο πειραματιστής μπορεί να έχει μία ιδέα για τα επιθυμητά αποτελέσματα συσταδοποίησης σε σχέση με τη θέση, το μέγεθος, το σχήμα και την πυκνότητα των συστάδων. Διαφορετικοί αλγόριθμοι συσταδοποίησης έχουν διαφορετικό αντίκτυπο σε αυτές τις παραμέτρους, όπως θα δείξει η περιγραφή των αλγορίθμων. Επομένως, η μεταβολή του αλγόριθμου συσταδοποίησης επηρεάζει τις παραμέτρους σχεδιασμού.

Αριθμός συστάδων: Ο αριθμός των συστάδων μπορεί να καθοριστεί εάν ο επιθυμητός αριθμός είναι γνωστός εκ των προτέρων (π.χ. λόγω αναφοράς σε έναν «χρυσό κανόνα» (gold standard) ή μπορεί να μεταβληθεί για να βρεθεί η βέλτιστη ανάλυση συστάδων. Θεωρητικά, το πρόβλημα της συσταδοποίησης μπορεί να λυθεί με εξαντλητική απαρίθμηση, δεδομένου ότι το σετ δειγμάτων είναι πεπερασμένο, οπότε υπάρχει μόνο ένας πεπερασμένος αριθμός πιθανών διαμερίσεων· στην πράξη, μία τέτοια προσέγγιση είναι αδιανόητη για όλα εκτός από τα απλούστερα προβλήματα.

Αμφισημία: Τα ρήματα μπορούν να έχουν πολλαπλές σημασίες, που απαιτούν την ανάθεση σε πολλαπλές τάξεις. Αυτό είναι δυνατό μόνο με τη χρήση ενός αλγόριθμου χαλαρής συσταδοποίησης, ο οποίος καθορίζει τις πιθανότητες συμμετοχής στη συστάδα για τα αντικείμενα που συσταδοποιούνται. Ένας αλγόριθμος σκληρής συσταδοποίησης εκτελεί μία απόφαση ναι / όχι σχετικά με το αν ένα αντικείμενο ανήκει σε μία συστάδα και δεν μπορεί να μοντελοποιήσει την αμφισημία ρήματος, αλλά είναι ευκολότερο να χρησιμοποιηθεί και να ερμηνευθεί.

Η επιλογή ενός αλγόριθμου συσταδοποίησης καθορίζει τη ρύθμιση των παραμέτρων. Στις επόμενες παραγράφους, περιγράφεται μία σειρά αλγορίθμων συσταδοποίησης και οι παράμετροί τους. Οι αλγόριθμοι διαιρούνται σε (A) αλγόριθμους ιεραρχικής συσταδοποίησης και (B) αλγόριθμους συσταδοποίησης διαμέρισης (Amigó et al., 2009).

Οι αλγόριθμοι συσταδοποίησης μπορούν να κατηγοριοποιηθούν με βάση το μοντέλο των συστάδων τους. Η ακόλουθη επισκόπηση θα απαριθμήσει μόνο τα πιο σημαντικά παραδείγματα αλγορίθμων συσταδοποίησης, καθώς υπάρχουν πιθανόν πάνω από 100 δημοσιευμένοι αλγόριθμοι συσταδοποίησης. Δεν προσφέρουν όλοι μοντέλα για τις συστάδες τους και επομένως δεν μπορούν εύκολα να κατηγοριοποιηθούν.

Δεν υπάρχει κανένας αντικειμενικά "σωστός" αλγόριθμος συσταδοποίησης, αλλά όπως σημειώνεται, η "συσταδοποίηση είναι στο μάτι του θεατή". Ο πλέον κατάλληλος αλγόριθμος συσταδοποίησης για ένα συγκεκριμένο πρόβλημα πρέπει συχνά να επιλεγεί πειραματικά, εκτός αν υπάρχει μαθηματικός λόγος να προτιμάται ένα μοντέλο συστάδων έναντι άλλου. Θα πρέπει να σημειωθεί ότι ένας αλγόριθμος που έχει σχεδιαστεί για ένα είδος μοντέλου θα αποτύχει γενικά σε ένα σύνολο δεδομένων που περιέχει ένα ριζικά διαφορετικό είδος μοντέλου. Για παράδειγμα, οι «k-μέσοι» (k-means) δεν μπορούν να βρουν μη κυρτές συστάδες.

4.3. Διαχωριστικές Μέθοδοι (Partitioning methods)

Έχοντας ως δεδομένο ένα σύνολο n αντικειμένων, μια διαχωριστική μέθοδος δημιουργεί k χωρίσματα των δεδομένων, όπου κάθε χώρισμα αντιπροσωπεύει ένα σύμπλεγμα και $k \leq n$. Δηλαδή, χωρίζει τα δεδομένα

σε k ομάδες, ώστε κάθε ομάδα να περιέχει τουλάχιστον ένα αντικείμενο. Με άλλα λόγια, οι διαχωριστικοί μέθοδοι διεξάγουν χωρίσματα ενός επιπέδου στα σύνολα δεδομένων. Οι βασικοί διαχωριστικοί μέθοδοι τυπικά υιοθετούν τον αποκλειστικό διαχωρισμό της συστάδας. Αυτό σημαίνει ότι κάθε αντικείμενο πρέπει να ανήκει ακριβώς σε μία ομάδα. Γενικά, οι περισσότερες διαχωριστικές μέθοδοι βασίζονται στην απόσταση. Έχοντας ως δεδομένο το k, όσον αφορά τον αριθμό των χωρισμάτων για να κατασκευαστούν, η διαχωριστική μέθοδος δημιουργεί έναν αρχικό διαχωρισμό. Στη συνέχεια, χρησιμοποιεί μια επαναληπτική τεχνική μετεγκατάστασης, με σκοπό τη βελτίωση της κατανομής, μετακινώντας αντικείμενα από μια ομάδα σε μια άλλη.

Το γενικό κριτήριο της καλής κατανομής είναι ότι τα αντικείμενα στην ίδια συστάδα είναι "κοντά" ή συσχετίζονται, ενώ τα αντικείμενα σε διαφορετικές συστάδες είναι "πολύ απομακρυσμένα" ή πολύ διαφορετικά. Ωστόσο, υπάρχουν διάφορα είδη άλλων κριτηρίων για την αξιολόγηση της ποιότητας των χωρισμάτων. Οι παραδοσιακές μέθοδοι τμηματοποίησης μπορούν να επεκταθούν στην ομαδοποίηση υποσυνόλων, αντί να κάνουν αναζήτηση στο πλήρες χώρο των δεδομένων. Αυτό είναι χρήσιμο όταν υπάρχουν πολλά χαρακτηριστικά και τα δεδομένα είναι αραιά. Χαρακτηριστικός αλγόριθμος διαχωριστικός είναι ο k-means.

4.4. Αλγόριθμος K-means

Υποθέτουμε ότι ένα σύνολο δεδομένων D περιέχει n αντικείμενα στον Ευκλείδειο χώρο. Οι Διαχωριστικές Μέθοδοι κατανέμουν τα αντικείμενα σε k συστάδες, $C_1,\ldots,C_k$, όπου $C_i \subset D$ και $C_i \cap C_j$ για $(1 \leq I,j \leq k)$. Έτσι, χρησιμοποιείται μια αντικειμενική λειτουργία για να εκτιμηθεί η ποιότητα κατανομής, ούτως ώστε τα αντικείμενα μέσα σε μια συστάδα να

είναι παρόμοια μεταξύ τους, αλλά αντίθετα με τα αντικείμενα σε άλλες συστάδες. Από αυτό προκύπτει ότι η αντικειμενική λειτουργία στοχεύει στην υψηλή ομοιομορφία στο εσωτερικό και στο χαμηλό επίπεδο ομοιότητας (Tran et al., 2013).

Η διαφορά μεταξύ ενός αντικειμένου $p \in C_i$ και c_i, που αντιπροσωπεύει τη συστάδα, μετριέται με $dist(p,c_i)$, όπου $dist(x,y)$ είναι η ευκλείδεια απόσταση μεταξύ δύο σημείων x και y. Η ποιότητα της συστάδας C_i μπορεί να υπολογιστεί με τη διακύμανση εντός του χώρου. Η διακύμανση είναι το άθροισμα του τετραγωνικού σφάλματος μεταξύ όλων των αντικειμένων του C_i και του κέντρου βάρους c_i, που ορίζεται ως εξής: $\mathbf{E}=\sum_{i=1}^{k}\sum_{p\in C_i} dist(p,c_i)^2$, όπου E είναι το άθροισμα του τετραγωνικού σφάλματος για όλα τα αντικείμενα του συνόλου δεδομένων, p είναι το σημείο στο χώρο, που αντιπροσωπεύει ένα δεδομένο αντικείμενο, και c_i είναι το κέντρο βάρους της συστάδας C_i (τα p και c_i είναι πολυδιάστατα).

Συγκεκριμένα, ο αλγόριθμος K-means ορίζει το κέντρο μιας συστάδας ως τη μέση τιμή των σημείων μέσα στη συστάδα. Η διαδικασία, που ακολουθείται ενέχει κάποια στάδια. Πρώτα, πρέπει να επιλεχθεί τυχαία k αριθμός αντικειμένων από το D, καθένα από τα οποία αντιπροσωπεύει αρχικά ένα μέσο ή ένα κέντρο της συστάδας. Εν συνεχεία, για κάθε ένα από τα υπόλοιπα αντικείμενα, ένα αντικείμενο αντιστοιχεί στη συστάδα στην οποία είναι το πιο παρόμοιο, με βάση την ευκλείδεια απόσταση μεταξύ του αντικειμένου και της μέσης συστάδας (Tran et al., 2013).

Από τα παραπάνω προκύπτει ότι ο αλγόριθμος K-means βελτιώνει διαδοχικά τη διακύμανση μεταξύ των ομάδων. Έτσι, για κάθε συστάδα, υπολογίζει το νέο μέσο, χρησιμοποιώντας τα αντικείμενα που έχουν αντιστοιχιστεί στη συστάδα στην προηγούμενη επανάληψη. Στη συνέχεια, όλα τα αντικείμενα ανακατανέμονται χρησιμοποιώντας τα ενημερωμένα μέσα ως τα νέα κέντρα των συστάδων. Οι επαναλήψεις

συνεχίζονται έως ότου η αντιστοίχιση καταστεί σταθερή. Δηλαδή, οι συστάδες που σχηματίζονται στον τρέχοντα γύρο είναι ίδιες με αυτές που σχηματίστηκαν στον προηγούμενο γύρο.

Ένας απλός ψευδοκώδικας του αλγορίθμου παρουσιάζεται και εξηγείται στον παρακάτω πίνακα, ενώ παράλληλα ακολουθεί ο σχηματισμός συμπλέγματος των συστάδων:

Αλγόριθμος K-means: Στον αλγόριθμο K-means κάθε κέντρο της συστάδας αντιπροσωπεύεται από τη μέση τιμή των αντικειμένων της συστάδας. **Εισαγωγή:** k: ο αριθμός των συστάδων D: ένα σύνολο δεδομένων που περιέχει n αντικείμενα **Εξαγωγή:** Ένα σύνολο από k συστάδες **Μέθοδος** Επιλέγεται αυθαίρετα ένας k αριθμός αντικειμένων από το D, ως αρχικό κέντρο της συστάδας. Εν συνεχεία επαναλαμβάνεται ή τοποθετείται ξανά κάθε αντικείμενο στη συστάδα, στην οποία το αντικείμενο είναι το πιο παρόμοιο, με βάση τη μέση τιμή των αντικειμένων της συστάδας. Έπειτα, ενημερώνεται η μέση συστάδα μέσω του υπολογισμού της μέσης τιμής των αντικειμένων για κάθε συστάδα, μέχρι να σταματήσουν να εμφανίζονται αλλαγές.

Πίνακας 1 - Ψευδοκώδικας αλγορίθμου

Η απόκτηση έγκυρων αποτελεσμάτων στην πράξη, προκύπτει από την εφαρμογή του αλγορίθμου K-means, πολλές φορές, με διαφορετικά

αρχικά κέντρα συστάδων. Η K-means μέθοδος μπορεί να εφαρμοστεί μόνο όταν ορίζεται ο μέσος όρος ενός συνόλου αντικειμένων. Για να καταστεί η μέθοδος K-means πιο αποτελεσματική σε μεγάλα σύνολα δεδομένων, χρησιμοποιείται μια πρώτη προσέγγιση μιας σειράς μεγάλου μεγέθους δειγμάτων στη συσσωμάτωση. Επιπρόσθετα, ένας άλλος τρόπος είναι μέσω μιας προσέγγισης φιλτραρίσματος, η οποία χρησιμοποιεί ένα δείκτη χωρικών ιεραρχικών δεδομένων για εξοικονόμηση κόστους, όταν πρόκειται για υπολογιστικά μέσα. Μια τρίτη προσέγγιση διερευνά την ιδέα της μικροκλίμακας, η οποία ομαδοποιεί πρώτα τα κοντινά αντικείμενα σε "μικροομαδοποιήσεις" και στη συνέχεια εκτελεί K-means ομαδοποίηση στις "μικροομαδοποιήσεις" (Tran et al., 2013).

4.5. Συσταδοποίηση με βάση τη συνδεσιμότητα (ιεραρχική συσταδοποίηση)

Η συσταδοποίηση που βασίζεται στη συνδεσιμότητα, επίσης γνωστή ως ιεραρχική συσταδοποίηση, βασίζεται στη βασική ιδέα των αντικειμένων που σχετίζονται περισσότερο με κοντινά αντικείμενα παρά με αντικείμενα που βρίσκονται πιο μακριά. Αυτοί οι αλγόριθμοι συνδέουν "αντικείμενα" για να σχηματίσουν "συστάδες" με βάση την απόστασή τους. Μία συστάδα μπορεί να περιγραφεί σε μεγάλο βαθμό από τη μέγιστη απόσταση που απαιτείται για τη σύνδεση μερών της συστάδας. Σε διαφορετικές αποστάσεις, θα σχηματιστούν διαφορετικές συστάδες, οι οποίες μπορούν να αναπαρίστανται από ένα δενδρόγραμμα, το οποίο εξηγεί από πού προέρχεται το κοινό όνομα "ιεραρχική συσταδοποίηση": αυτοί οι αλγόριθμοι δεν παρέχουν απλώς μία διαμέριση του συνόλου των δεδομένων, αλλά αντ' αυτού μία εκτεταμένη ιεραρχία συστάδων που συγχωνεύονται μεταξύ τους σε συγκεκριμένες αποστάσεις (Zhao et al., 2005). Σε ένα δενδρόγραμμα, ο άξονας y σηματοδοτεί την απόσταση στην

οποία οι συστάδες συγχωνεύονται, ενώ τα αντικείμενα τοποθετούνται κατά μήκος του άξονα x έτσι ώστε να μην αναμειγνύονται οι συστάδες.

Η συσταδοποίηση που βασίζεται στη συνδεσιμότητα είναι μια ολόκληρη οικογένεια μεθόδων που διαφέρουν βάσει του τρόπου που υπολογίζονται οι αποστάσεις. Εκτός από τη συνήθη επιλογή των συναρτήσεων απόστασης, ο χρήστης πρέπει επίσης να αποφασίσει ποιο κριτήριο σύνδεσης θα χρησιμοποιήσει (δεδομένου ότι μία συστάδα αποτελείται από πολλαπλά αντικείμενα, υπάρχουν πολλοί υποψήφιοι για τον υπολογισμό της απόστασης). Οι δημοφιλείς επιλογές είναι γνωστές ως συσταδοποίηση μονής σύνδεσης (η ελάχιστη απόσταση αντικειμένων) και πλήρους σύνδεσης (η μέγιστη απόσταση αντικειμένων), επίσης γνωστή ως συσταδοποίηση μέσης σύνδεσης) (Stuetzle et al., 2010). Επιπλέον, η ιεραρχική συσταδοποίηση μπορεί να είναι συσσωρευτική (ξεκινώντας με τα μεμονωμένα στοιχεία και συγκεντρώνοντάς τα σε συστάδες) ή διαιρετική (ξεκινώντας με το πλήρες σύνολο δεδομένων και χωρίζοντάς το σε διαμερίσεις).

4.6. Συσταδοποίηση βάσει κέντρου βάρους (centroid-based clustering)

Σε συσταδοποιήσεις που βασίζονται στο κέντρο βάρους, οι συστάδες αναπαρίστανται από ένα κεντρικό διάνυσμα, το οποίο μπορεί να μην είναι αναγκαστικά μέλος του συνόλου δεδομένων. Όταν ο αριθμός των συστάδων είναι σταθεροποιημένος στο k, η συσταδοποίηση *KMeans* δίνει έναν τυπικό ορισμό ως ένα πρόβλημα βελτιστοποίησης: βρείτε τα k κέντρα συστάδων και αναθέστε τα αντικείμενα στο πλησιέστερο κέντρο συστάδων, έτσι ώστε να ελαχιστοποιούνται οι τετραγωνισμένες αποστάσεις από την συστάδα.

Το ίδιο το πρόβλημα βελτιστοποίησης είναι γνωστό ότι ανήκει στην κατηγορία των δυσεπίλυτων NP-hard προβλημάτων και επομένως κανείς αλγόριθμος δεν μπορεί να πιστοποιήσει ότι το βέλτιστο που προσδιορίζει είναι το ολικό βέλτιστο του προβλήματος. Μία ιδιαίτερα γνωστή προσεγγιστική μέθοδος είναι ο αλγόριθμος Lloyd, που συχνά αναφέρεται μόνο ως "*αλγόριθμος KMeans*" (αν και άλλος αλγόριθμος εισήγαγε αυτό το όνομα). Εντούτοις, βρίσκει μόνο ένα «τοπικό βέλτιστο» (local optimum) και συνήθως εκτελείται πολλές φορές με διαφορετικές τυχαίες αρχικοποιήσεις (Aiyer et al., 2005). Οι παραλλαγές των *KMeans* περιλαμβάνουν συχνά τέτοιες βελτιστοποιήσεις, όπως την επιλογή της καλύτερης εκτέλεσης από πολλαπλές εκτελέσεις, αλλά και τον περιορισμό των «κεντροειδών» (centroids) σε μέλη του συνόλου των δεδομένων (*k*-medoids), την επιλογή διαμέσων (συσταδοποίηση *KMeans*), την επιλογή των αρχικών κέντρων λιγότερο τυχαία (*KMeans* ++) ή επιτρέποντας μία εκχώρηση «ασαφούς συστάδας» (fuzzy c-means).

Οι περισσότεροι αλγόριθμοι τύπου *KMeans* απαιτούν ο αριθμός των συστάδων - *k* - να έχει καθοριστεί εκ των προτέρων, κάτι το οποίο θεωρείται ένα από τα μεγαλύτερα μειονεκτήματα αυτών των αλγορίθμων. Επιπλέον, οι αλγόριθμοι προτιμούν συστάδες περίπου παρόμοιου μεγέθους, καθώς πάντοτε θα αναθέτουν ένα αντικείμενο στο πλησιέστερο κέντρο βάρους. Αυτό συχνά οδηγεί σε λανθασμένη χάραξη συνόρων των συστάδων (κάτι που δεν προκαλεί έκπληξη διότι ο αλγόριθμος βελτιστοποιεί τα κέντρα συστάδων, όχι τα σύνορα συστάδων).

Η ανάλυση *KMeans* έχει μία σειρά από ενδιαφέρουσες θεωρητικές ιδιότητες. Πρώτον, διαχωρίζει τον χώρο δεδομένων σε μία δομή γνωστή ως διάγραμμα Voronoi. Δεύτερον, είναι εννοιολογικά κοντά στην ταξινόμηση πλησιέστερων γειτόνων, και ως εκ τούτου είναι δημοφιλής

στη μηχανική μάθηση. Τρίτον, μπορεί να θεωρηθεί ως μία παραλλαγή της συσταδοποίησης που βασίζεται σε μοντέλο και ο αλγόριθμος Lloyd ως παραλλαγή του αλγορίθμου μεγιστοποίησης προσδοκιών (Jain, 2010).

4.7. Συσταδοποίηση με βάση την κατανομή

Το μοντέλο συσταδοποίησης που σχετίζεται περισσότερο με τη στατιστική βασίζεται σε μοντέλα κατανομής. Οι συστάδες σε αυτήν την περίπτωση μπορούν εύκολα να οριστούν ως αντικείμενα που ανήκουν κατά πάσα πιθανότητα στην ίδια κατανομή. Μια βολική ιδιότητα αυτής της προσέγγισης είναι ότι αυτό μοιάζει πολύ με τον τρόπο που παράγονται τεχνητά σύνολα δεδομένων: με δειγματοληψία τυχαίων αντικειμένων από μια κατανομή.

Ενώ η θεωρητική θεμελίωση αυτών των μεθόδων είναι εξαιρετική, υποφέρουν από ένα βασικό πρόβλημα που είναι γνωστό ως «υπεραρμογή» (overfitting), εκτός εάν θέτονται περιορισμοί στην πολυπλοκότητα του μοντέλου. Ένα πιο σύνθετο μοντέλο θα είναι συνήθως σε θέση να εξηγήσει καλύτερα τα δεδομένα, πράγμα που καθιστά εγγενώς δύσκολη την επιλογή της κατάλληλης πολυπλοκότητας του μοντέλου.

Μια εξέχουσα μέθοδος είναι γνωστή ως «γκαουσιανά μοντέλα μείγματος» (χρησιμοποιώντας τον αλγόριθμο μεγιστοποίησης προσδοκίας). Εδώ, το σύνολο δεδομένων συνήθως μοντελοποιείται με έναν σταθερό (για να αποφευχθεί η υπεραρμογή) αριθμό γκαουσιανών κατανομών που αρχικοποιούνται τυχαία και των οποίων οι παράμετροι βελτιστοποιούνται επαναλαμβανόμενα ώστε να ταιριάζουν καλύτερα στο σύνολο δεδομένων. Αυτό θα συγκλίνει σε ένα τοπικό βέλτιστο, έτσι ώστε

πολλαπλές εκτελέσεις να παραγάγουν ίσως διαφορετικά αποτελέσματα. Προκειμένου να επιτευχθεί μία σκληρή συσταδοποίηση, τα αντικείμενα στη συνέχεια συχνά ανατίθενται στη γκαουσιανή κατανομή στην οποία πιθανότατα ανήκουν· για χαλαρές συσταδοποιήσεις, αυτό δεν είναι απαραίτητο (Tran et al., 2013).

Η συσταδοποίηση με βάση την κατανομή παράγει σύνθετα μοντέλα για συστάδες που μπορούν να συλλάβουν συσχετισμό και εξάρτηση μεταξύ χαρακτηριστικών. Ωστόσο, αυτοί οι αλγόριθμοι θέτουν ένα επιπλέον φορτίο στον χρήστη: για πολλά σύνολα πραγματικών δεδομένων, μπορεί να μην υπάρχει ένα συνοπτικά ορισμένο μαθηματικό μοντέλο (π.χ. υποθέτοντας ότι οι κατανομές Gauss είναι μια μάλλον ισχυρή παραδοχή στα δεδομένα).

4.8. Συσταδοποίηση με βάση την πυκνότητα

Στη συσταδοποίηση με βάση την πυκνότητα, οι συστάδες ορίζονται ως περιοχές υψηλότερης πυκνότητας από το υπόλοιπο του συνόλου δεδομένων. Τα αντικείμενα σε αυτές τις αραιοκατοικημένες περιοχές - που απαιτούνται για τη διαίρεση των συστάδων - θεωρούνται συνήθως θόρυβος και σημεία συνόρων (Rodriguez et al., 2014).

Η πιο δημοφιλής μέθοδος συσταδοποίησης με βάση την πυκνότητα είναι η DBSCAN. Σε αντίθεση με πολλές νεότερες μεθόδους, διαθέτει ένα καλά ορισμένο μοντέλο συστάδων που ονομάζεται "πυκνότητα-προσβασιμότητα". Παρόμοια με τη συσταδοποίηση που βασίζεται σε συνδέσεις, βασίζεται στην σύνδεση σημείων εντός ορισμένων κατωφλίων απόστασης (Tran et al., 2013). Ωστόσο, συνδέει μόνο σημεία που ικανοποιούν ένα κριτήριο πυκνότητας, που στην αρχική εκδοχή ορίζεται ως ένας ελάχιστος αριθμός άλλων αντικειμένων εντός αυτής της ακτίνας.

Μία συστάδα αποτελείται από όλα τα αντικείμενα που συνδέονται με την πυκνότητα (τα οποία μπορούν να σχηματίσουν μία συστάδα αυθαίρετου σχήματος, σε αντίθεση με πολλές άλλες μεθόδους) συν όλα τα αντικείμενα που βρίσκονται εντός του εύρους αυτών των αντικειμένων. Μία άλλη ενδιαφέρουσα ιδιότητα του DBSCAN είναι ότι η πολυπλοκότητά του είναι αρκετά χαμηλή και ότι θα βρει ουσιαστικά τα ίδια αποτελέσματα (είναι καθοριστικό για τα σημεία πυρήνα και θορύβου, αλλά όχι για τα συνοριακά σημεία) σε κάθε εκτέλεση, επομένως δεν χρειάζεται να εκτελεσθεί πολλές φορές. Το OPTICS είναι μια γενίκευση του DBSCAN που καταργεί την ανάγκη επιλογής μίας κατάλληλης τιμής για την παράμετρο εύρους ε, και παράγει ένα ιεραρχικό αποτέλεσμα σχετικό με αυτό της συσταδοποίησης συνδέσεων. Η συσταδοποίηση σύνδεσης πυκνότητας (Density-Link-Clustering, DeLi-Clu) συνδυάζει ιδέες από την συσταδοποίηση μονής σύνδεσης και το OPTICS, εξαλείφοντας εξ ολοκλήρου την παράμετρο ε και προσφέροντας βελτιώσεις επιδόσεων έναντι του OPTICS χρησιμοποιώντας ένα δείκτη R-δένδρων (Ansari et al., 2013).

Το βασικό μειονέκτημα των DBSCAN και OPTICS είναι ότι αναμένουν κάποιου είδους πτώση της πυκνότητας για να ανιχνεύσουν τα σύνορα των συστάδων. Σε σύνολα δεδομένων, για παράδειγμα, με αλληλεπικαλυπτόμενες γκαουσιανές κατανομές- μια κοινή περίπτωση χρήσης σε τεχνητά δεδομένα - τα σύνορα συστάδων που παράγονται από αυτούς τους αλγορίθμους συχνά φαίνονται αυθαίρετα, επειδή η πυκνότητα των συστάδων μειώνεται συνεχώς. Σε ένα σύνολο δεδομένων που αποτελείται από μίγματα Gaussians, αυτοί οι αλγόριθμοι σχεδόν πάντα υπολείπονται, από άποψη επίδοσης, μεθόδων, όπως η συσταδοποίηση EM, που είναι σε θέση να μοντελοποιήσουν με ακρίβεια αυτό το είδος δεδομένων.

Η μετατόπιση μέσου (mean-shift) είναι ένας τρόπος συσταδοποίησης όπου κάθε αντικείμενο μετακινείται στην πυκνότερη περιοχή του γείτονα χώρου του, με βάση την εκτίμηση της πυκνότητας πυρήνα. Τελικά, τα αντικείμενα συγκλίνουν σε τοπικά μέγιστα πυκνότητας. Παρόμοια με τη συσταδοποίηση *KMeans*, αυτοί οι "ελκυστήρες πυκνότητας" (density attractors) μπορούν να χρησιμεύσουν ως εκπρόσωποι για το σύνολο των δεδομένων, αλλά η μέση μετατόπιση μπορεί να ανιχνεύσει ομάδες αυθαίρετου σχήματος παρόμοιες με το DBSCAN. Λόγω της δαπανηρής επαναληπτικής διαδικασίας και της εκτίμησης της πυκνότητας, η μέση μετατόπιση είναι συνήθως βραδύτερη από την DBSCAN ή τους *k*-μέσους. Εκτός από αυτό, η εφαρμοσιμότητα του αλγόριθμου μέσης μετατόπισης σε πολυδιάστατα δεδομένα εμποδίζεται από την μη ομαλή συμπεριφορά της εκτίμησης της πυκνότητας του πυρήνα, η οποία οδηγεί σε υπερκατακερματισμό των ουρών των συστάδων (Ansari et al., 2013).

Ακολουθεί επισκόπηση των μεθόδων ομαδοποίησης:

Πίνακας 2- Μέθοδοι ομαδοποίησης

Μέθοδοι	**Γενικά χαρακτηριστικά**
Διαχωριστικές Μέθοδοι	- Βρίσκουν αμοιβαίες αποκλειστικές συστάδες σφαιρικού σχήματος - Βασίζονται στην απόσταση - Αποτελεσματικές για σύνολα δεδομένων μικρού και μεσαίου μεγέθους
Ιεραρχικές Μέθοδοι	- Η ομαδοποίηση είναι μια ιεραρχική αποσύνθεση - Αδύνατη η διόρθωση εσφαλμένων

	συγχωνεύσεων ή διαχωρισμών
Μέθοδοι που βασίζονται στην πυκνότητα	- Εντοπίζουν αυθαίρετα διαμορφωμένες συστάδες - Οι συστάδες είναι πυκνές περιοχές αντικειμένων στο χώρο, που διαχωρίζονται από περιοχές χαμηλής πυκνότητας - Πυκνότητα συστάδων: Κάθε σημείο πρέπει να έχει έναν ελάχιστο αριθμό γειτονικών σημείων - Δυνατότητα φιλτραρίσματος των ακραίων τιμών
Μέθοδοι που βασίζονται σε πλέγμα	- Πολλαπλή ανάλυση πλέγματος στη δομή των δεδομένων - Γρήγορος χρόνος επεξεργασίας (συνήθως ανεξάρτητος από τον αριθμό των αντικειμένων των δεδομένων, αλλά εξαρτάται από το μέγεθος του πλέγματος)

4.9. Προσδιορισμός του αριθμού των συστάδων

Ο προσδιορισμός του "σωστού" αριθμού συστάδων σε ένα σύνολο δεδομένων είναι σημαντικός, επειδή ορισμένοι αλγόριθμοι ομαδοποίησης, όπως ο K-means., απαιτούν μια τέτοια παράμετρο. Μπορεί να θεωρηθεί ότι βρίσκει μια καλή ισορροπία μεταξύ της συμπιεστότητας και της ακρίβειας στην ανάλυση των συστάδων.

Είναι απαραίτητη η εξέταση δύο ακραίων παραδειγμάτων. Στις περιπτώσεις που ολόκληρο το σύνολο δεδομένων θεωρείται μια συστάδα, τότε μεγιστοποιείται η συμπίεση των δεδομένων. Ωστόσο, μια τέτοια ανάλυση συστάδων δεν έχει αξία. Από την άλλη μεριά, η επεξεργασία

κάθε αντικειμένου σε ένα σύνολο δεδομένων ως συστάδα δίνει την καλύτερη ανάλυση συγκέντρωσης (δηλαδή υψηλής ακρίβειας, λόγω της μηδενικής απόστασης μεταξύ ενός αντικειμένου και του αντίστοιχου κέντρου συστάδας). Σε ορισμένες μεθόδους, όπως το k-means, επιτυγχάνεται ακόμη και η καλύτερη τιμή. Ωστόσο, η ύπαρξη ενός αντικειμένου ανά συστάδα δεν επιτρέπει καμία σύνοψη δεδομένων (Jain, 2010).

Επιπρόσθετα, ο προσδιορισμός του αριθμού των συστάδων δεν είναι εύκολος, επειδή, συχνά, ο "σωστός" αριθμός είναι διφορούμενος. Στις περισσότερες περιπτώσεις, η καταγραφή του σωστού αριθμού συστάδων εξαρτάται από το σχήμα της διανομής και την κλίμακα στο σύνολο των δεδομένων, καθώς και από την ανάλυση των συστάδων, που απαιτείται από τον χρήστη. Υπάρχουν πολλοί τρόποι για να εκτιμηθεί ο αριθμός των συστάδων. Παρακάτω παρουσιάζονται εν συντομία μερικές απλές αλλά δημοφιλείς και αποτελεσματικές μέθοδοι.

Η μέθοδος ***elbowmethod*** βασίζεται στην παρατήρηση ότι η αύξηση του αριθμού των συστάδων μπορεί να βοηθήσει στη μείωση του αθροίσματος της διακύμανσης εντός των συστάδων σε κάθε συστάδα. Αυτό συμβαίνει επειδή η ύπαρξη περισσότερων συστάδων επιτρέπει να συλληφθούν λεπτότερες ομάδες με δεδομένα, που είναι πιο παρόμοια μεταξύ τους. Ωστόσο, το οριακό αποτέλεσμα της μείωσης του αθροίσματος των διακυμάνσεων των συστάδων μπορεί να μειωθεί εάν σχηματιστούν πάρα πολλές συστάδες. Αυτό συμβαίνει επειδή η διάσπαση μια συνεκτικής συστάδας σε δύο, δίνει μόνο μια μικρή μείωση. Συνεπώς, ένας τρόπος για την επιλογή του σωστού αριθμού συστάδων είναι να χρησιμοποιηθεί το σημείο καμπής στην καμπύλη του αθροίσματος των διακυμάνσεων εντός των συστάδων, σε σχέση με τον αριθμό των συστάδων (Jain, 2010).

Από τεχνική άποψη, έχοντας ως δεδομένο έναν αριθμό, k> 0, μπορούν να σχηματιστούν k συστάδες στο εν λόγω σύνολο δεδομένων, αν χρησιμοποιηθεί ένας αλγόριθμος ομαδοποίησης, όπως το k-means. Έτσι, υπολογίζεται το άθροισμα των διακυμάνσεων εντός των συστάδων, var(k). Συνεπώς, υπάρχει η δυνατότητα σχηματισμού της καμπύλης του var σε σχέση με το k. Το πρώτο (ή το πιο βαρυσήμαντο) σημείο καμπής της καμπύλης υποδεικνύει τον "σωστό" αριθμό.

Οι πιο προηγμένες μέθοδοι μπορούν να καθορίσουν τον αριθμό των συστάδων, χρησιμοποιώντας κριτήρια πληροφόρησης ή θεωρητικές προσεγγίσεις πληροφοριών. Επίσης, ο "σωστός" αριθμός συστάδων σε ένα σύνολο δεδομένων μπορεί να προσδιοριστεί με διασταυρούμενη πιστοποίηση, μια τεχνική που χρησιμοποιείται συχνά στην ταξινόμηση. Έτσι, διαιρείται, αρχικά, το σύνολο δεδομένων, D, σε m τμήματα. Στη συνέχεια, χρησιμοποιούνται τα m-1 τμήματα για να δημιουργηθεί ένα μοντέλο συστάδων και τα εναπομείναντα τμήματα χρησιμεύουν για τον έλεγχο της ποιότητας των συστάδων (Jain, 2010).

Πιο συγκεκριμένα, για κάθε σημείο του συνόλου δοκιμών, πρέπει να εντοπιστεί το πλησιέστερο κέντρο. Συνεπώς, μπορεί να χρησιμοποιηθεί το άθροισμα των τετραγωνικών αποστάσεων μεταξύ όλων των σημείων στο σετ δοκιμών και το πλησιέστερο κέντρο για να υπολογιστεί και αξιολογηθεί το ποσοστό που ένα μοντέλο συστάδων ταιριάζει με το σύνολο δοκιμών. Επιπλέον, για οποιονδήποτε ακέραιο αριθμό k> 0, επαναλαμβάνεται αυτή η διαδικασία m φορές για να εξαχθούν ομάδες με k αριθμό συστάδων, χρησιμοποιώντας κάθε τμήμα με τη σειρά του ως σύνολο δοκιμών. Ο μέσος όρος του μέτρου ποιότητας λαμβάνεται ως το συνολικό μέτρο ποιότητας (Jain, 2010). Εν συνεχεία, υπάρχει η δυνατότητα σύγκρισης του συνολικού μέτρου ποιότητας σε σχέση με τις διαφορετικές τιμές του k και έτσι εντοπίζεται ο αριθμός των συστάδων, που ταιριάζουν καλύτερα στα δεδομένα.

5. Ο Αλγόριθμος Birch

5.1. Παρουσίαση του αλγορίθμου

Ο BIRCH (Balanced Iterative Reducing and Clustering using Hierarchies) είναι μια αποτελεσματική μέθοδος ομαδοποίησης μεγάλων βάσεων δεδομένων (Zhang et al., 1996). Η εύρεση χρήσιμων προτύπων σε μεγάλες βάσεις δεδομένων έχει προσελκύσει τα τελευταία χρόνια το ενδιαφέρον της επιστημονικής κοινότητας. Αυτό συμβαίνει λόγω του ότι η ομαδοποίηση των δεδομένων σε ένα πολυδιάστατο σύστημα αποτελεί ένα από τα πιο ευρέως διαδεδομένα προβλήματα σε αυτόν τον τομέα. Οι προηγούμενες εργασίες δεν αντιμετωπίζουν επαρκώς το πρόβλημα των μεγάλων συνόλων με δεδομένα και ελαχιστοποίηση του I/O κόστους.

Ειδικότερα, ο BIRCH ομαδοποιεί διαδοχικά και δυναμικά τα εισερχόμενα σημεία του πολυδιάστατου μετρικού στοιχείου, προσπαθώντας να παράγει την καλύτερη δυνατή ποιοτική ομαδοποίηση σύμφωνα με τους διαθέσιμους πόρους. Δηλαδή τη διαθέσιμη χωρητικότητα και τους χρονικούς περιορισμούς. Αυτός ο αλγόριθμος είναι ο πρώτος στην ομαδοποίηση δεδομένων, που πετυχαίνει την αποτελεσματική διαχείριση του "θορύβου" (σημεία δεδομένων, που δεν αποτελούν μέρος του υποκείμενου σχεδίου). Το I/O κόστος του είναι γραμμικό στο μέγεθος του συνόλου δεδομένων, όπου μια ενιαία σάρωση του συνόλου δεδομένων αποδίδει μια καλή συσσώρευση και ένα ή περισσότερα πρόσθετα περάσματα μπορούν (προαιρετικά) να χρησιμοποιηθούν περαιτέρω για τη βελτίωση της ποιότητας (Zhang, 1996). Πρόκειται για έναν μη επιβλεπόμενο αλγόριθμο εξόρυξης δεδομένων, ο οποίος πραγματοποιεί Ιεραρχική Ομαδοποίηση (Hierarchical Clustering)

στα δεδομένα. Ιδιαίτερη αξία έχει η χρήση του αλγορίθμου σε αρκετά μεγάλα σύνολο δεδομένων.

Σε πρώτη φάση είναι χρήσιμο να μελετηθεί το τρόπος λειτουργίας του αλγορίθμου. Ο αλγόριθμος λειτουργεί σε στάδια. Το πρώτο περιλαμβάνει την κατασκευή ενός δέντρου χαρακτηριστικών ομαδοποίησης (CF Tree), το οποίο είναι ζυγισμένο ως προς το ύψος δέντρο που ορίζεται με χρήση δύο βασικών παραμέτρων. Ένα CF-tree είναι ένα δέντρο ισορροπημένο σε ύψος με δύο παραμέτρους. Αυτές αφορούν τον παράγοντα διακλάδωσης B για τον χωρίς φύλλα κόμβο (nonleaf node) και L για κόμβο με φύλλα (leaf node) και το όριο T.

Κάθε κόμβος χωρίς φύλλα περιέχει στο μέγιστο B καταχωρήσεις της μορφής $[CF_i, child_i]$, όπου i = 1, 2, …, B, το «$child_i$» είναι ένας δείκτης στον i-th child node και το CF_i είναι η εγγραφή CF της υποομάδας, που αντιπροσωπεύεται από αυτό το «παιδί». Έτσι, ένας κόμβος χωρίς φύλλα αντιπροσωπεύει μια υποσυστάδα, η οποία αποτελείται από όλες τις υποσυστάδες, που αντιπροσωπεύονται από τις καταχωρήσεις τους. Από την άλλη, ένας κόμβος με φύλλα περιέχει καταχωρήσεις στο μέγιστο L, και κάθε είσοδος είναι CF. Επιπλέον, κάθε τέτοιος κόμβος έχει δύο δείκτες, τους «prev» και «next», οι οποίοι χρησιμοποιούνται για να αλυσοδέσουν όλων των κόμβων φύλλων μαζί για αποτελεσματική σάρωση. Εν συνεχεία, ένας κόμβος φύλλων αντιπροσωπεύει, επίσης, μια υποσυστάδα, η οποία εμπεριέχει όλες τις υποσυστάδες, που αντιπροσωπεύονται από τις καταχωρήσεις τους. Όμως, όλες οι καταχωρήσεις σε κόμβο φύλλων πρέπει να πληρούν ένα όριο για το T, το οποίο καθορίζεται ως: η διάμετρος (εναλλακτικά, η ακτίνα) κάθε καταχώρησης σε φύλλα πρέπει να είναι μικρότερη από T (Zhang, 1996).

Ειδικότερα, το μέγεθος του δένδρου είναι συνάρτηση του T. Όσο μεγαλύτερο είναι το T, τόσο μικρότερο είναι το δέντρο. Σε αυτό το σημείο,

απαιτείται ένας κόμβος για να χωρέσει σε μια σελίδα μεγέθους P, όπου P είναι μια παράμετρος του BIRCH. Μόλις δοθεί η διάσταση d του χώρου δεδομένων, είναι γνωστά τα μεγέθη των καταχωρήσεων των κόμβων με φύλλα και των κόμβων χωρίς φύλλων και στη συνέχεια τα B και L καθορίζονται από το P. Κατά συνέπεια, ο P μπορεί να ποικίλει για τον συντονισμό απόδοσης.

Ένα τέτοιο CF-tree θα χτιστεί δυναμικά, καθώς εισάγονται νέα αντικείμενα δεδομένων. Συνήθως, χρησιμοποιείται για την καθοδήγηση μιας νέας εισαγωγής στη σωστή συστάδα για λόγους ομαδοποίησης. Με τον ίδιο ακριβώς τρόπο χρησιμοποιείται και για την εισαγωγή μιας νέας εισαγωγής στη σωστή θέση για λόγους διαλογής. Ωστόσο, το CF-tree είναι μια πολύ συμπαγής αναπαράσταση του συνόλου δεδομένων, επειδή κάθε είσοδος, σε έναν κόμβο φύλλων, δεν είναι ένα μοναδικό σημείο δεδομένων, αλλά μια υποσυστάδα. Αυτή η υποσυστάδα απορροφά τόσα σημεία δεδομένων όσα του επιτρέπει η οριακή τιμή.

Ο τρόπος εισαγωγής σε ένα δέντρο ενός διανύσματος χαρακτηριστικών έχει ως εξής (Zhang, 1996):

- Ξεκινώντας από τη ρίζα του δέντρου κατεβαίνουμε μέχρι να φτάσουμε σε φύλλο επιλέγοντας σε κάθε κόμβο το παιδί στο οποίο βρίσκεται πιο κοντά το δεδομένο διάνυσμα με χρήση κάποιας απόστασης, που συνήθως επιλέγεται να είναι η Ευκλείδεια.
- Εάν το φύλλο μπορεί να απορροφήσει το δεδομένο, είτε επειδή περιέχει λιγότερα στοιχεία από L, είτε επειδή η προσθήκη του σε αυτό εξακολουθεί να διατηρεί την ακτίνα L μικρότερη από την τιμή του κατωφλίου T, τότε στο φύλλο αυτό εισάγεται το νέο δεδομένο και αντίστοιχα ενημερώνονται όλοι οι πρόγονοι του στο δέντρο για αυτήν την εισαγωγή. Σε αντίθετη περίπτωση γίνεται διάσπαση του φύλλου

σε 2, η οποία ενδέχεται να οδηγήσει και σε διάσπαση των ενδιάμεσων κόμβων του δέντρου, εάν αυτοί δεν μπορούν να χωρέσουν το νέο φύλλο που δημιουργήθηκε. Για την διάσπαση ενός κόμβου σε δύο νέους, χρησιμοποιούμε ως πόλους των δύο νέων κόμβων τα δύο πιο απομακρυσμένα διανύσματα χαρακτηριστικών και τα υπόλοιπα τοποθετούνται στον κόμβο που βρίσκονται πιο κοντά.

Από τη στιγμή που θα έχει κατασκευαστεί το δέντρο, εφαρμόζεται ένας από τους κλασσικούς αλγόριθμους ομαδοποίησης στα φύλλα του δέντρου. Έτσι προκύπτουν οι τελικές συστάδες, σύμφωνα με τις οποίες πλέον μπορούμε να κατατάσσουμε τα στοιχεία.

Μία σημαντική συμβολή αυτού του αλγορίθμου είναι η διαμόρφωση του προβλήματος της ομαδοποίησης, κατά τρόπο που να είναι κατάλληλος για πολύ μεγάλα σύνολα δεδομένων, καθιστώντας σαφείς τους περιορισμούς χρόνου και μνήμης. Επιπλέον, ο BIRCH έχει κάποια πλεονεκτήματα έναντι προηγούμενων προσεγγίσεων που βασίζονται στην απόσταση.

Αρχικά, είναι τοπικός (σε αντίθεση με τον παγκόσμιο). Δηλαδή, κάθε απόφαση ομαδοποίησης γίνεται χωρίς σάρωση όλων των σημείων δεδομένων ή όλων των υφιστάμενων συστάδων. Επίσης, χρησιμοποιεί μετρήσεις, που αντικατοπτρίζουν τη φυσική εγγύτητα των σημείων και ταυτόχρονα μπορούν να διατηρηθούν διαδοχικά κατά τη διάρκεια της διαδικασίας ομαδοποίησης. Επιπλέον, ο BIRCH εκμεταλλεύεται το γεγονός ότι ο χώρος δεδομένων, συνήθως, δεν είναι ομοιόμορφα κατειλημμένος και επομένως όλα τα σημεία δεδομένων δεν είναι εξίσου σημαντικά για τους σκοπούς της ομαδοποίησης. Μια πυκνή περιοχή σημείων αντιμετωπίζεται συλλογικά ως ένα ενιαίο σύνολο. Αντίθετα, τα σημεία σε αραιοκατοικημένες περιοχές αντιμετωπίζονται ως ακραίες τιμές και αφαιρούνται προαιρετικά.

Επίσης, πρέπει να αναφερθεί ότι αυτός ο αλγόριθμος αξιοποιεί πλήρως τη διαθέσιμη μνήμη για να αποκομίσει τις καλύτερες δυνατές υποομάδες (καλύτερη ακρίβεια), ενώ ελαχιστοποιεί το I/O κόστος (καλύτερη αποτελεσματικότητα). Έτσι, η διαδικασία συσσωμάτωσης και αναγωγής οργανώνεται και χαρακτηρίζεται από τη χρήση μιας δομής δέντρου μέσα στη μνήμη, που είναι ισορροπημένη και ιδιαίτερα δομημένη. Λόγω αυτών των χαρακτηριστικών, ο χρόνος λειτουργίας του είναι γραμμικώς κλιμακωτός. Παράλληλα, δεν απαιτεί ολόκληρο το σύνολο δεδομένων εκ των προτέρων και σαρώνει το σύνολο των δεδομένων μόνο μία φορά.

Ακολουθεί η επισκόπηση του BIRCH, ανελυμένη σε φάσεις. . Η κύρια λειτουργία της Φάσης 1 είναι η σάρωση όλων των δεδομένων και η δημιουργία ενός αρχικού δέντρου εντοπισμού CF, χρησιμοποιώντας τη δεδομένη ποσότητα μνήμης και ανακυκλώνοντας χώρο στο δίσκο. Αυτό το δέντρο CF προσπαθεί να αντικατοπτρίζει τις πληροφορίες συσσωμάτωσης του συνόλου δεδομένων όσο το δυνατόν λεπτομερέστερα κάτω από τα όρια της μνήμης (Zhang, 1996). Επιπλέον, με τα πολυπληθή σημεία δεδομένων οργανωμένα σε υποομάδες και τα αραιά σημεία δεδομένων να έχουν αφαιρεθεί ως ακραίες τιμές, αυτή η φάση δημιουργεί μια περίληψη των δεδομένων αυτών.

Μετά τη φάση 1, οι μεταγενέστεροι υπολογισμοί για τις επόμενες φάσεις είναι:

1. γρήγοροι, επειδή (α) δεν χρειάζονται I/O λειτουργίες και (β) το πρόβλημα της ομαδοποίησης των αρχικών δεδομένων μειώνεται σε ένα μικρότερο πρόβλημα συγκέντρωσης των υποομάδων στις καταχωρήσεις των φύλλων.
2. Ακριβείς, επειδή (α) εξαλείφονται πολλές ακραίες τιμές και (β) τα εναπομείναντα δεδομένα αντανακλώνται με την καλύτερη

λεπτομέρεια, που μπορεί να επιτευχθεί, λαμβάνοντας υπόψη τα διαθέσιμα στοιχεία.

3. με λιγότερη ευαισθησία, επειδή οι καταχωρήσεις φύλλων του αρχικού δέντρου σχηματίζουν μια εντολή εισόδου, που περιέχει καλύτερη τοποθεσία δεδομένων σε σύγκριση με την αυθαίρετη αρχική εισαγωγή δεδομένων.

Η φάση 2 είναι προαιρετική. Παρατηρείται ότι οι υπάρχουσες παγκόσμιες ή ημι-παγκόσμιες μέθοδοι ομαδοποίησης, που εφαρμόζονται στη φάση 3, έχουν διαφορετικές κλίμακες μεγέθους εισροών εντός των οποίων έχουν καλές επιδόσεις όσον αφορά την ταχύτητα και την ποιότητα. Έτσι, υπάρχει πιθανότητα να υπάρχει ένα κενό μεταξύ του μεγέθους των αποτελεσμάτων της Φάσης 1 και της εμβέλειας εισόδου της Φάσης 3. Η Φάση 2 χρησιμεύει ως 'μαξιλάρι' και γεφυρώνει αυτό το χάσμα. Συνεπώς, παρόμοια με τη Φάση 1, ανιχνεύει τις καταχωρήσεις φύλλων στο αρχικό δέντρο CF για να ανοικοδομήσει μικρότερο δέντρο CF, ενώ παράλληλα απομακρύνει τις ακραίες τιμές και ομαδοποιεί συνωστισμένες υποομάδες σε μεγαλύτερες. Επίσης, παρατηρείται ότι οι υπάρχοντες αλγόριθμοι ομαδοποίησης για ένα σύνολο σημείων δεδομένων μπορούν εύκολα να προσαρμοστούν. Αυτό συμβάλλει στη λειτουργία τους με ένα σύνολο υποομάδων, το καθένα από τα οποία περιγράφεται από το CF του.

Μετά τη Φάση 3, αποκτάμε ένα σύνολο ομάδων που καταγράφει το κύριο μοτίβο διανομής στα δεδομένα. Ωστόσο, ενδέχεται να υπάρχουν μικρές και εντοπισμένες ανακρίβειες λόγω του σπάνιου προβλήματος της κακής τοποθέτησης και το γεγονός ότι η Φάση 3 εφαρμόζεται σε μια χονδροειδή περίληψη των δεδομένων. Η φάση 4 είναι προαιρετική και συνεπάγεται το κόστος των πρόσθετων περασμάτων πέρα από τα δεδομένα για να διορθωθούν αυτές οι ανακρίβειες και να βελτιωθούν περαιτέρω οι συστάδες. Μέχρι τώρα τα αρχικά δεδομένα έχουν σαρωθεί

μόνο μία φορά, αν και οι πληροφορίες των δέντρων και των εξωστρεφών ενδέχεται να έχουν σαρωθεί πολλές φορές (Zhang, 1996).

Η Φάση 4 χρησιμοποιεί τα κεντροειδή των ομάδων, που παράγονται από τη φάση 3 και αναδιανέμει τα σημεία δεδομένων για να αποκτήσει ένα σύνολο νέων ομάδων. Αυτό, όχι μόνο επιτρέπει σε σημεία, που ανήκουν σε μια συστάδα, να μετεγκατασταθούν, αλλά επίσης εξασφαλίζει ότι όλα τα αντίγραφα ενός γνωστού σημείου δεδομένων πηγαίνουν στην ίδια συστάδα. Η φάση 4 μπορεί να επεκταθεί με επιπλέον περάσματα, εφόσον το επιθυμεί ο χρήστης. Ως μπόνους, κατά τη διάρκεια αυτού του περάσματος, κάθε σημείο δεδομένων μπορεί να επισημανθεί με τη συστάδα στην οποία ανήκει, αν ο χρήστης επιθυμεί να προσδιορίσει τα σημεία δεδομένων σε κάθε συστάδα. Αυτό το στάδιο επιτρέπει την επιλογή απόρριψης των ακραίων τιμών (Zhang, 1996). Δηλαδή, ένα σημείο που είναι πολύ μακριά από το πλησιέστερο μπορεί να αντιμετωπιστεί ως μια απόκλιση και δεν περιλαμβάνεται στο αποτέλεσμα.

5.2. Εφαρμογές του αλγορίθμου

Εκτός του προβλήματος κατάτμησης της αγοράς που επιχειρεί να επιλύσει ο αλγόριθμος Birch στην παρούσα εργασία, από την βιβλιογραφία έχει βρεθεί ότι έχει εφαρμοστεί και σε άλλα ποικίλα προβλήματα. Συγκεκριμένα, έχει χρησιμοποιηθεί για κατηγοριοποίηση pixel σε εικόνες, οι οποίες καταφθάνουν δυναμικά και για συμπίεση εικόνων (Zhang, 1996). Επιπλέον έχει εφαρμοστεί σε προβλήματα ομαδοποίησης κειμένου σε ερευνητικές δουλειές, όπου και έχει συγκριθεί με αλγόριθμο KMeans (Karpov and Goroslavsky, 2012). Τέλος, διαφορετικές εκδοχές του αλγορίθμου έχουν εφαρμοστεί σε προβλήματα ανάλυσης μονάδων θερμότητας (Du and Li, 2010).

6. Σκοπός – Στόχοι – Ερευνητικά ερωτήματα

Όπως αναλύθηκε παραπάνω, η εργασία καλείται να απαντήσει σε κάποια ερευνητικά ερωτήματα που ανακύπτουν από την πειραματική αξιολόγηση και σύγκριση του αλγορίθμου Birch. Συγκεκριμένα, θα σχολιαστεί η τόσο η επίδοση (η ταχύτητα) του αλγορίθμου σε σχέση με άλλους όσο και η απόδοση του (το ποσό καλά πραγματοποιεί την ομαδοποίηση των δεδομένων). Τα ερωτήματα αυτά θα απαντηθούν τόσο σε πραγματικά δεδομένα αγοράς που αναφέρθηκαν στην προηγούμενη ενότητα όσο και σε τεχνητά δεδομένα που ξέρουμε εξαρχής σε πόσες ομάδες είναι χωρισμένα.

7. Μεθοδολογία και προσαρμογή του Αλγορίθμου στο πρόβλημα.

Η μεθοδολογία που ακολουθείται στην δεδομένη εργασία περιλαμβάνει την υλοποίηση του αλγορίθμου Birch σε γλώσσα Python και την εκτέλεση τόσο αυτού όσο και άλλων κλασσικών machine learning αλγορίθμων ομαδοποίησης για την απάντηση των ερωτημάτων της προηγούμενης ενότητας με τα σύνολα δεδομένων που παρουσιάστηκαν.

Για την υλοποίηση του αλγορίθμου Birch είναι απαραίτητο να κατασκευαστεί μία κλάση για καθένα από τα βασικά αντικείμενα που αναφέρθηκαν παραπάνω. Αρχικά, είναι απαραίτητο να ορίσουμε την κλάση CF η οποία θα περιγράφει ένα χαρακτηριστικό ομαδοποίησης και θα δίνει τη δυνατότητα πραγματοποίησης αναγκαίων πράξεων με άλλα. Η κλάση περιέχει συναρτήσεις όπως η άθροιση η δημιουργία του αρνητικού χαρακτηριστικού ομαδοποίησης. Η άθροιση δίνει το νέο CF που θα προκύψει η ομάδα που περιγράφει το δεδομένο χαρακτηριστικό ενωθεί με μία άλλη. Η δημιουργία του αρνητικού χαρακτηριστικού θα φανεί χρήσιμη στην συνέχεια, όταν αφαιρείται ένα στοιχείο από μία ομάδα,

πράγμα χρήσιμο στην υλοποίηση του Birch όταν γίνεται η διάσπαση ενός κόμβου σε δύο. Στις επόμενες υποενότητες, δίνονται με τη μορφή ψευδοκώδικα οι λειτουργίες των συναρτήσεων αυτών.

Τόσο για την υλοποίηση του αλγορίθμου Birch όσο και για την εκτέλεση του πειραματικού μέρους χρησιμοποιείται η γλώσσα Python καθώς και διάφορες βιβλιοθήκες που είναι ιδιαίτερα χρήσιμες στον τομέα της ανάλυσης δεδομένων. Αρχικά, αναφέρουμε ότι θα χρησιμοποιηθούν οι βιβλιοθήκες *Numpy* και *Pandas*. Η βιβλιοθήκη *Numpy* είναι ιδιαίτερα χρήσιμη για επιστημονικούς υπολογισμούς και πράξεις γραμμικής άλγεβρας που θέλουμε να πραγματοποιήσουμε στη γλώσσα Python. Η βιβλιοθήκη *Pandas* είναι μία βιβλιοθήκη εύχρηστες και υψηλής επίδοσης δομές δεδομένων και εργαλεία ανάλυσης δεδομένων. Ένα βασικό εργαλείο που παρέχει είναι το dataframe, το οποίο το οποίο μπορεί να χρησιμοποιηθεί για την ανάγνωση συνόλου δεδομένων από το δίσκο σε ιδιαίτερα χρήσιμη μορφή. Επιπλέον, χρησιμοποιείται η matplotlib η οποία είναι μία βιβλιοθήκη απεικόνισης διαγραμμάτων σε ιδιαίτερα εύχρηστη μορφή.

Για λόγους καλύτερης κατανόησης του κειμένου, ο σχετικός κώδικας θα παρουσιαστεί με την μορφή ψευδοκώδικα και ο ακριβής κώδικας σε Python παρατίθεται στο παράρτημα.

```
Συνάρτηση add(CF1, CF2):
      Πρόσθεσε τα επιμέρους πεδία των διανυσμάτων χαρακτηριστικών CF1
      και CF2, ένα προς ένα.
```

Ψευδοκώδικας 1 – Ψευδοκώδικας άθροισης δύο χαρακτηριστικών ομαδοποίησης

```
Συνάρτηση neg(CF):
      Φτιάξε ένα νέο διάνυσμα χαρακτηριστικών
      Σε κάθε πεδίο του βάλε το αντίθετο του πεδίου του διανύσματος
      χαρακτηριστικού εισόδου.
```

Ψευδοκώδικας 2 – Ψευδοκώδικας αρνητικού χαρακτηριστικού ομαδοποίησης

Επόμενο απαραίτητο βήμα είναι μία κλάση η οποία θα περιγράφει ένα κόμβο του δέντρου. Ένα αντικείμενο της κλάσης αυτής θα περιέχει τόσο το χαρακτηριστικό ομαδοποίησης του κόμβου, ως αντικείμενο της κλάσης CF που κατασκευάστηκε νωρίτερα, όσο και δείκτες προς τα παιδιά και τον πατέρα του κόμβου. Η κλάση θα πρέπει να παρέχει επίσης τις εξής μεθόδους με τις ακόλουθες λειτουργίες:

- *get_initialized_root()* : Όταν καλεστεί, επιστρέφει ένα αρχικοποιημένο CF δέντρο με βάση ένα νέο διάνυσμα που δίνεται από το χρήστη. Στη συνάρτηση αυτή ορίζονται και τα επιθυμητά χαρακτηριστικά του δέντρου όπως το μέγεθος του φύλλου L, ο παράγοντας διάσπασης B και το κατώφλι T.
- *get_new_root()* : Συνάρτηση που καλείται, εάν κατά την εισαγωγή ενός νέου διανύσματος στο δέντρο πραγματοποιηθεί διάσπαση της ρίζας. Στην περίπτωση αυτή, η συνάρτηση δέχεται ως όρισμα τους δύο νέους κόμβους που αποτελούν τη διάσπαση της ρίζας και επιστρέφει τη νέα ρίζα του δέντρου.
- *centroid()* : Μέθοδος που επιστρέφει το κέντρο της ομάδας που αναπαρίσταται από το δεδομένο κόμβο.
- *get_radius()* : Μέθοδος που επιστρέφει την ακτίνα της ομάδας που αναπαρίσταται από το δεδομένο κόμβο.

- *get_distance()* : Μέθοδος που επιστρέφει την απόσταση ενός νέου διανύσματος από την ομάδα που αναπαρίσταται από το δεδομένο κόμβο.
- *update_cf()* : Μέθοδος που χρησιμοποιείται για την αριθμητική ενημέρωση του χαρακτηριστικού ομαδοποίησης που περιλαμβάνεται σε αυτήν την ομάδα.
- *add_new_child()* : Εισάγει ένα νέο κόμβο - ομάδα ως παιδί σε αυτό τον κόμβο και ενημερώνει κατάλληλα την τιμή του κόμβου.
- *get_closer_child()* : Εύρεση του πιο κοντινού παιδιού του κόμβου σε ένα νέο διάνυσμά.
- *get_furthest_pair()* : Εύρεση των παιδιών του κόμβου που βρίσκονται πιο μακριά μεταξύ τους.
- *needs_split()* : Συνάρτηση που ελέγχει κατά πόσο ένας κόμβος πρέπει να διασπαστεί η όχι ένας ενδιάμεσος κόμβος με βάση το εάν περιέχει παραπάνω στοιχεία από όσα ορίζει η παράμετρος διάσπασης.
- *split_node()* : Συνάρτηση που διασπάει έναν ενδιάμεσο κόμβο σε δύο νέους. Ένας πρόκειται για τη ρίζα του δέντρου, τότε επιστρέφει τη νέα ρίζα, αλλιώς επιστρέφει τους κόμβους που προέκυψαν από τη διάσπαση.

- *insert()* : Μέθοδος για την εισαγωγή ενός νέου δεδομένου σε αυτόν τον κόμβο και αναδρομικά στο κατάλληλο παιδί του.

Στη συνέχεια, παρατίθεται ο ψευκώδικας που περιγράφει την λειτουργία των κάποιων από τις παραπάνω συναρτήσεις που είναι πιο περίπλοκη η υλοποίηση τους. Οι υπόλοιπες εκτέλουν στοιχειώδεις διαδικασίες που προκύπτουν άμεσα από την περιγραφή τους.

```
Συνάρτηση centroid (node):

      Εύρεση του κεντρικού σημείου σύμφωνα με τον τύπο node.LS/node.N
      Επέστρεψε το αποτέλεσμα
```

Ψευδοκώδικας 3 – Ψευδοκώδικας υπολογισμού κέντρου ενός κόμβου.

```
Συνάρτηση needs_split (node, branching_factor):

      Εάν ο κόμβος που βρίσκεσαι έχει περισσότερα παιδιά από το
      branching factor
            Επέστρεψε ότι ο κόμβος χρειάζεται διάσπαση
      Διαφορετικά
            Ενημέρωσε ότι η διάσπαση δεν είναι απαραίτητη.
```

Ψευδοκώδικας 4 – Ψευδοκώδικας ελέγχου διάσπασης κόμβου.

```
Συνάρτηση split_node (node):

      Απόσταση = 0
      Ορισμός κενής μεταβλητής με όνομα ζεύγος, που θα περιέχει το
      ζεύγος παιδιών με την μεγαλύτερη απόσταση
      Για κάθε ζεύγος παιδιών (Παιδί1, Παιδί2) που ανήκει στον κόμβο
      node επανέλαβε:
            Εάν Αποσταση(Παιδι1, Παιδί2) είναι μεγαλύτερη του
      Απόσταση:
                  Απόσταση = Απόσταση του ζεύγους που εξετάζεται
                  ζεύγος = το ζεύγος παιδιών που εξετάζεται
      Παιδί 1 = ζεύγος.Παιδί 1
      Παιδί 2 = ζεύγος.Παιδί 2
      Ομάδα παιδιού 1 = Κενό σύνολο
      Ομάδα παιδιού 2 = Κενό σύνολο
      Για κάθε ένα παιδί Χ του κόμβου node:
            Εάν Απόσταση(Χ, Παιδί1) <= Απόσταση(Χ, Παιδί2):
                  Προσθήκη του Χ στην Ομάδα_παιδιού_1
            Αλλιώς:
                  Προσθήκη του Χ στην Ομάδα_παιδιού_2
      Παιδί1.παιδιά = Ομάδα_Παιδιού_1
      Παιδί2.παιδιά = Ομάδα_Παιδιού_2
      Εάν υπάρχει γονικός κόμβος για τα Παιδί1, Παιδί2
```

```
            Επέστρεψε τον γονικό κόμβο
      Αλλιώς
            Επέστρεψε Παιδί1, Παιδί2
```

Ψευδοκώδικας 5 – Ψευδοκώδικας διάσπασης κόμβου

```
Συνάρτηση insert (node, datapoint, branching_factor):
      Βρες τον πιο κοντινό  παιδί στον κόμβο που επιχειρείς να
      εισάγεις στο δέντρο.
      Εισήγαγε τον κόμβο σε αυτό το παιδί επαναλαμβάνοντας την
      διαδικασία αυτή από την αρχή.
      Αφού ολοκληρωθεί η εισαγωγή, έλεγξε εάν ο κόμβος απαιτεί
      διάσπαση, και εάν ναι πραγματοποίησε την με την διαδικασία
      split_node και ενημέρωσε κατάλληλα.
```

Ψευδοκώδικας 6 – Ψευδοκώδικας εισαγωγής στοιχείου σε κόμβο

Ωστόσο, τα φύλλα του δέντρου είναι υπεύθυνα για παραπάνω λειτουργίες και για αυτό επεκτείνουμε την κλάση για να καλυφθούν αυτές οι νέες λειτουργίες. Ανάμεσα στις νέες λειτουργείες που υποστηρίζουν τα φύλλα είναι η αποθήκευση και η διαχείριση των νέων δεδομένων που εισάγονται στο δέντρο. Επίσης, τα φύλλα πρέπει να είναι όλα συνδεδεμένα μεταξύ τους. Έτσι, αυτές οι διαφορές που παρουσιάζουν από έναν οποιονδήποτε κόμβο του δέντρου οδηγούν στην τροποποίηση των υπαρχόντων μεθόδων της πατρικής κλάσης CFNode. Τελικά, η νέα κλάση CFLeaf, περιλαμβάνει ότι και η πατρική, παρέχοντας όμως τροποποιημένες κάποιες λειτουργίες και κάποιες επιπλέον για την επιθυμητή λειτουργία του εκτεταμένου κόμβου ως φύλλο. Οι νέες και οι τροποποιημένες μέθοδοι παρουσιάζονται στη συνέχεια.

- *_compute_distance_from_datapoint()* : Υπολογίζει την απόσταση ενός δεδομένου από κάποιο δεδομένο από αυτά που έχουν αποθηκευτεί στη λίστα στο δεδομένο φύλλο.

- *_add_new_datapoint()* : Συνάρτηση που προσθέτει ένα νέο δεδομένο στη λίστα και ενημερώνει αντίστοιχα το χαρακτηριστικό ομαδοποίησης του δέντρου.
- *_needs_split()* : Τροποποιημένη εκδοχή της αντίστοιχης του ενδιάμεσου κόμβου η οποία ελέγχει αν το φύλλο έχει στοιχεία περισσότερα από το μέγεθός του (L) ή η ακτίνα του είναι μεγαλύτερη από το κατώφλι T.
- *_split_node()* : Τροποποιημένη εκδοχή της αντίστοιχης του ενδιάμεσου κόμβου, η οποία διασπάει το φύλλο σε 2, φροντίζοντας τα δύο νέα φύλλα που θα εισαχθούν να πάρουν τη θέση του προηγούμενου στην αλυσίδα που συνδέει τα φύλλα.
- *insert()* : Τροποποιημένη εκδοχή της αντίστοιχης του ενδιάμεσου κόμβου η οποία εισάγει στο δεδομένο φύλλο το σημείο. Είναι η τελική κλήση της αναδρομικής εκτέλεσης της αντίστοιχης του ενδιάμεσου κόμβου.

Ο ψευδοκώδικας των παραπάνω διαφέρει ελάχιστα από αυτόν τον ενδιάμεσων κόμβων όπως φαίνεται από τις περιγραφές και για αυτό δεν παρατίθεται.

Τέλος, κατασκευάζουμε την κλάση Birch που από ένα σύνολο δεδομένων και με δοσμένες παραμέτρους μπορεί να κατασκευάσει το αντίστοιχο μοντέλο εισάγοντας τα δεδομένα στο δέντρο και εξάγοντας τις κλάσεις που θα προκύψουν. Στην python, η κλάση είναι υλοποιημένη με βάση το πρότυπο του *scikit-learn*. Το *scikit-learn* είναι μία βιβλιοθήκη έτοιμων machine learning αλγορίθμων και εργαλείων κατάλληλων για προεπεξεργασία δεδομένων και αξιολόγηση των μοντέλων που προκύπτουν. Δεδομένου ότι ο Birch υπάγεται στους αλγόριθμους ομαδοποίησης, κατασκευάζεται ως επέκταση των κλάσεων BaseEstimator,

ClusterMixin, TransformerMixin που είναι το θεμέλιο για την κατασκευή τέτοιου τύπου αλγορίθμων στο scikit. Αυτό οδηγεί στην υλοποίηση των βασικών μεθόδων *fit()* (που εκπαιδεύει το μοντέλο) και *predict()* που κατατάσσει σε μια κλάση τα δεδομένα από το μοντέλο που εκπαιδεύτηκε. Η υλοποίηση της μεθόδου fit του Birch στηρίζεται στην κατασκευή ενός CF δέντρου με χρήση των προηγούμενων κλάσεων και τελικά της ομαδοποίησης των κέντρων των φύλλων με χρήση του αλγόριθμου ομαδοποίησης Agglomerative Hierarchical Clustering, όπως προτείνεται στο paper παρουσίασης του Birch. Στη συνέχεια, δίνεται ο ψευδοκώδικας για την εκπαίδευση ενός μοντέλου με τον αλγόριθμο Birch και για την κατηγοριοποίηση ενός νέου στοιχείου σε μια ομάδα.

```
Συνάρτηση Birch_fit (datapoints, branching_factor, leaf_size, \
                threshold, num_clusters):

      Επέλεξε το πρώτο διάνυσμα (έστω X) του συνόλου εκπαίδευσης
      datapoints.
      Αρχικοποίησε ένα κενό CFNode με το X ως εξής:
            CFNode(πεδίο LS) = αθροισμα των συντεταγμένων του X
            CFNode(πεδίο SS) = άθροισμα του τετραγώνου των
      συντεταγμένων του X
            CFNode(πεδίο N) = 1
      Για κάθε ένα σημείο από το σύνολο εκπαίδευσης datapoints,
      ξεκινώντας από το δεύτερο:
            Εισήγαγε το στον αρχικό κόμβο σύμφωνα με την μέθοδο
            insert, όπως περιγράφεται στον Ψευδοκώδικα 6.

      Πάρε από κάθε φύλλο του δέντρου που κατασκευάστηκε τα κεντρικά
      σημεία.
      Εφάρμοσε στο σύνολο των κεντρικών σημείων κάποιον έτοιμο
      αλγόριθμο ομαδοποίησης, π.χ. Agglomerative Clustering.
      Βρες τις ομάδες που σχηματίζονται.
      .
      Ομαδοποιημένο Σύνολο = Κένο Σύνολο
      Για κάθε σημείο X του συνόλου εκπαίδευσης:
            Βρες την ομάδα με κεντρικό σημείο Y που ελαχιστοποιεί την
            Απόσταση(X, Y)
            Πρόσθεση στο κένο σύνολο το ζεύγος (X, Y) που έδωσε την
            ελαχιστοποίηση της απόστασης
      Επέστρεψε το Ομαδοποιημένο Σύνολο
```

Ψευδοκώδικας 7 – Ψευδοκώδικας εκπαίδευσης μοντέλου με μέθοδο Birch

```
Συνάρτηση predict (model, datapoint):
      Βρες την ομάδα στης οποίας βρίσκεται πιο κοντά στο κεντρικό
      σημείο το στοιχείο που θες να κατηγοριοποιήσεις.
      Επέστρεψε την ομάδα που βρήκες.
```

Ψευδοκώδικας 8 – Ψευδοκώδικας κατηγοριοποίησης ενός στοιχείου.

8. Ανάλυση δεδομένων και παρουσίαση αποτελεσμάτων

8.1. Ανάλυση Wholesale Dataset

Για την εκτέλεση του αλγορίθμου Birch και τη σύγκριση του με άλλους αλγορίθμους χρησιμοποιήθηκε το Wholesale Customer Dataset της αποθήκης συνόλων δεδομένων του UCI . Το σετ δεδομένων αναφέρεται σε πελάτες ενός χονδρικού διανομέα. Περιλαμβάνει τα ετήσια έξοδα σε χρηματικές μονάδες διαφόρων ειδών προϊόντων. Όσον αφορά τα χαρακτηριστικά του, το dataset αποτελείται από 440 εγγραφές και 8 στήλες χαρακτηριστικών ακέραιου τύπου. Το dataset διαβάζεται με χρήση του *Pandas* library.

Ένα δείγμα του παραπάνω σετ δεδομένων φαίνεται στη συνέχεια.

	Channel	Region	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
0	2	3	12669	9656	7561	214	2674	1338
1	2	3	7057	9810	9568	1762	3293	1776
2	2	3	6353	8808	7684	2405	3516	7844
3	1	3	13265	1196	4221	6404	507	1788
4	2	3	22615	5410	7198	3915	1777	5185

Πίνακας 3 – Δείγμα του σετ δεδομένων

Παρατηρούμε ότι το dataset πέραν από τις χρηματικές μονάδες ανά προϊόν, έχει και δύο επιπλέον στήλες *Channel* και *Region*, οι οποίες δεν αποτελούν χρήσιμη πληροφορία για την διεξαγωγή της ομαδοποίησης.

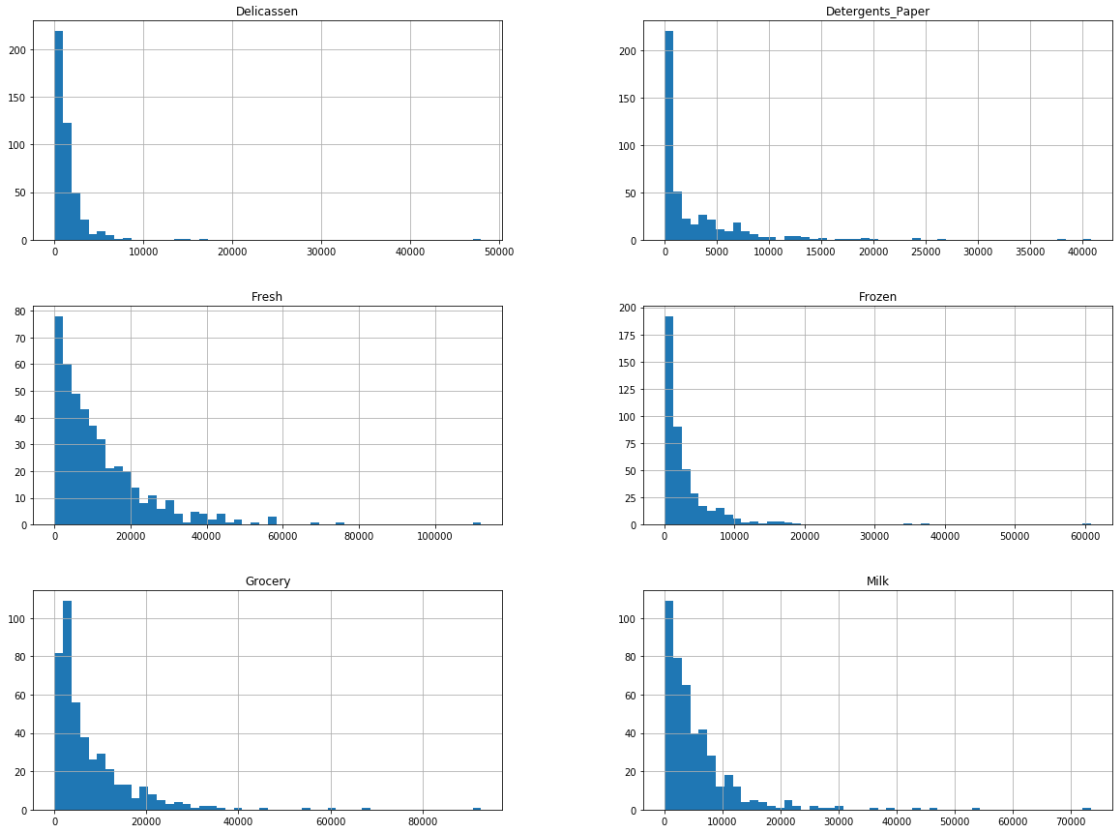
Συνεπώς, θα κρατήσουμε σαν δεδομένα εκπαίδευσης μόνο τις στήλες που περιγράφουν χρηματική μονάδα ανά προϊόν. Το αποτέλεσμα είναι το ακόλουθο.

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
0	12669	9656	7561	214	2674	1338
1	7057	9810	9568	1762	3293	1776
2	6353	8808	7684	2405	3516	7844
3	13265	1196	4221	6404	507	1788
4	22615	5410	7198	3915	1777	5185

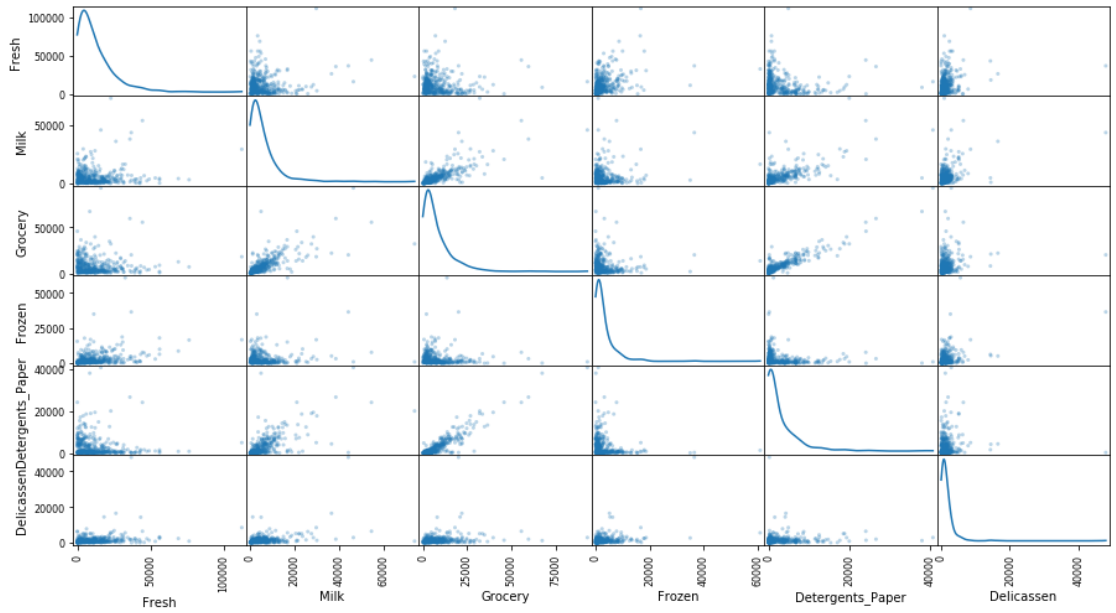
Πίνακας 4 – Δείγμα του σετ δεδομένων μετά την αφαίρεση των άχρηστων στηλών

Αρχικά είναι χρήσιμο να ληφθεί μία εποπτική εικόνα της κατανομής των χαρακτηριστικών αλλά και των βασικών μέτρων θέσης και διασποράς τους. Για τα λόγο αυτό γίνεται κατασκευή ιστογραμμάτων για τα χαρακτηριστικά εισόδου, διαγράμματα διασποράς και επιπλέον μια προσέγγιση της κατανομής του κάθε χαρακτηριστικού.

- **Ιστογράμματα χαρακτηριστικών**



Εικόνα 2 - Διαγράμματα διασποράς ανάμεσα σε χαρακτηριστικά με προσέγγιση κατανομής στην ενδιάμεση διαγώνιο



Σχεδιάγραμμα 1 - Μέτρα θέσης και διασποράς

	Fresh	Milk	Grocery	Frozen	Detergents_Paper	Delicassen
count	440.000000	440.000000	440.000000	440.000000	440.000000	440.000000
mean	12000.297727	5796.265909	7951.277273	3071.931818	2881.493182	1524.870455
std	12647.328865	7380.377175	9503.162829	4854.673333	4767.854448	2820.105937
min	3.000000	55.000000	3.000000	25.000000	3.000000	3.000000
25%	3127.750000	1533.000000	2153.000000	742.250000	256.750000	408.250000
50%	8504.000000	3627.000000	4755.500000	1526.000000	816.500000	965.500000
75%	16933.750000	7190.250000	10655.750000	3554.250000	3922.000000	1820.250000
max	112151.000000	73498.000000	92780.000000	60869.000000	40827.000000	47943.000000

Πίνακας 5 - Σετ δεδομένων

Από τα παραπάνω που περιγράφουν το σετ των δεδομένων γίνεται φανερό ότι χρειάζεται καθαρισμό από δεδομένα που εμφανίζονται σποραδικά χωρίς να εκφράζουν κάτι το συγκεκριμένο (ακραίες τιμές - outliers). Αυτό μπορεί να προκύψει εύκολα από τα μέτρα θέσης, αφού παρατηρούμε ότι η τυπική απόκλιση κάθε στήλης είναι μεγαλύτερη από τη μέση τιμή της και επιπλέον το μέγιστο κάθε χαρακτηριστικού είναι κατά τάξεις μεγέθους μεγαλύτερο από το 75% τεταρτημόριο. Αντίστοιχα η πληροφορία αυτή προκύπτει και από τα ιστογράμματα, αφού παρατηρούμε μεγαλύτερη συχνότητα εμφάνισης δεδομένων μέχρι ένα σημείο και από κάποιο χρηματικό ποσό και πάνω η συχνότητα προσεγγίζει συνεχώς το μηδέν. Για το την αφαίρεση των ακραίων τιμών από το set δεδομένων χρησιμοποιείται ο κανόνας του Tukey, σύμφωνα με τον οποίο γνωρίζοντας τα 25% και 75% τεταρτημόρια μπορώ να κρατήσω ως ωφέλιμη πληροφορία τις γραμμές που ικανοποιούν για κάθε στήλη c τη συνθήκη

$$c \in (Q1 - 1.5(Q3 - Q1), Q3 + 1.5(Q3 - Q1)) \quad [5.2]$$

όπου Q1 και Q3 τα σημεία που ορίζουν τα 25% και 75% τεταρτημόρια αντίστοιχα.

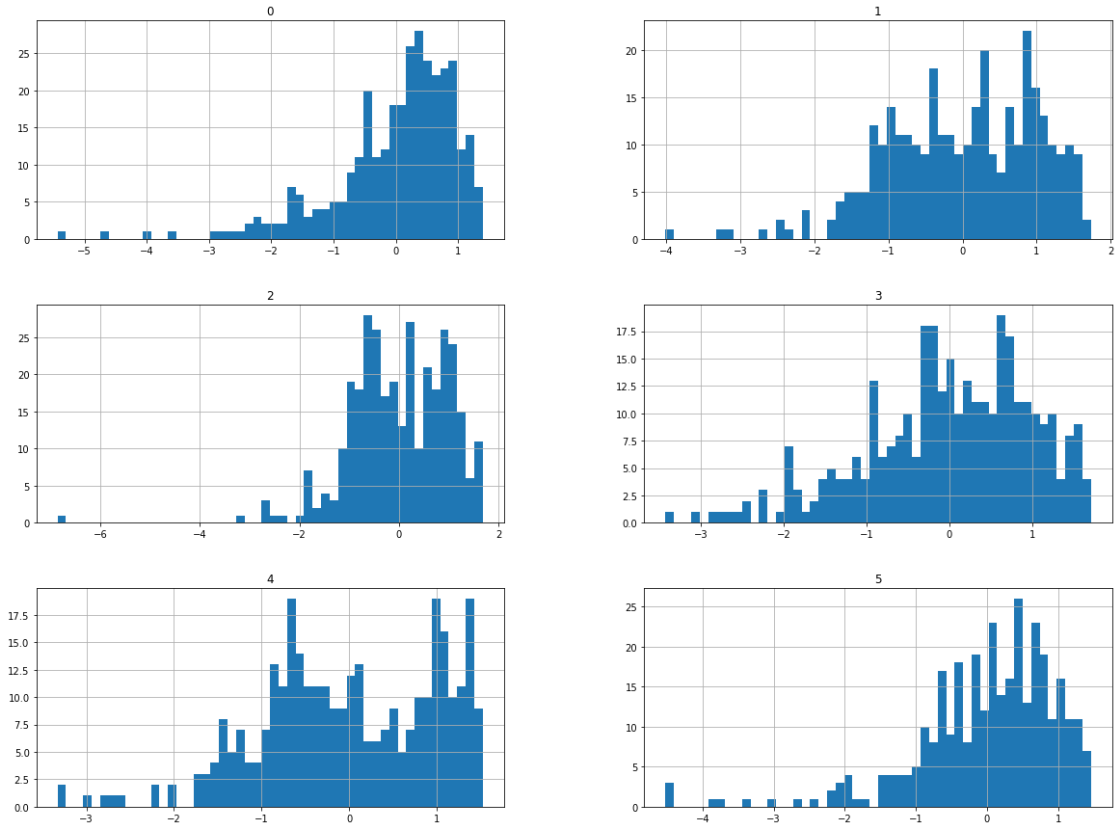
Με την παραπάνω μέθοδο πετάμε όσα δεδομένα θεωρούνται ακραίες τιμές. Ωστόσο, παρατηρώντας τις κατανομές βλέπουμε ότι τα δεδομένα μας έχουν όλα μια κατανομή με μορφή καμπάνα η οποία είναι τραβηγμένη προς τα αριστερά. Για να μεταφέρουμε την καμπάνα προς το κέντρο μπορούμε να επιχειρήσουμε τη λογαρίθμιση αφού έχουν κανονικοποιηθεί τα δεδομένα. Αυτή η κανονικοποίηση επιτυγχάνεται μέσω κατάλληλων πακέτων του scikit που εκτελούν αυτόματα την παραπάνω διεργασία.

	0	1	2	3	4	5
0	0.615334	1.280144	0.644267	-1.500906	0.791760	0.617847
1	0.191409	1.296337	0.869898	0.389301	0.918108	0.856454
2	0.648639	-0.857297	0.085574	1.546320	-0.217220	0.862128
3	0.400113	1.120214	0.271751	-0.483000	0.549912	0.686160
4	0.583597	0.149567	0.566950	-0.776635	0.889239	-0.138908

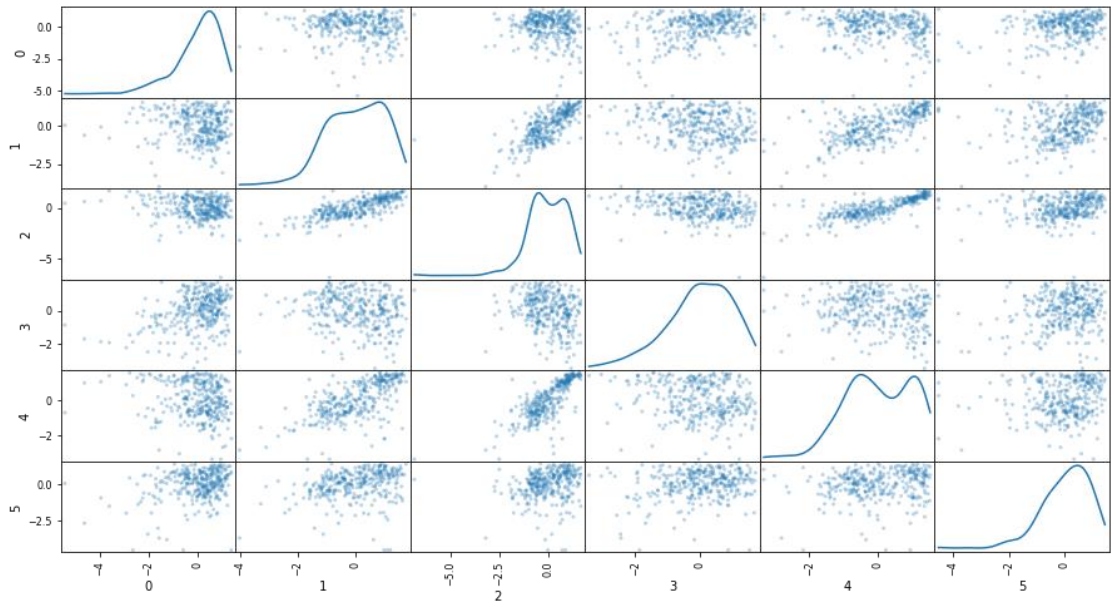
Πίνακας 6 – Κανονικοποιημένα Δεδομένα

Ύστερα από αυτήν την προεπεξεργασία των δεδομένων, γίνεται ξανά απεικόνιση του συνόλου δεδομένων ώστε να διαπιστωθεί ότι τα χαρακτηριστικά ακολουθούν προσεγγιστικά την κανονική κατανομή και παρουσιάζουν ομοιογένεια (έχουν αγνοηθεί οι ακραίες τιμές).

- **Ιστογράμματα χαρακτηριστικών**



Εικόνα 3 – Διαγράμματα διασποράς ανάμεσα σε χαρακτηριστικά με προσέγγιση κατανομής στην ενδιάμεση διαγώνιο



Σχεδιάγραμμα 2 - Μέτρα θέσης και διασποράς

	0	1	2	3	4	5
count	3.320000e+02	3.320000e+02	3.320000e+02	3.320000e+02	3.320000e+02	3.320000e+02
mean	-3.897819e-15	3.103274e-15	-1.107548e-15	-1.781707e-15	-1.514184e-15	-3.547363e-15
std	1.001509e+00	1.001509e+00	1.001509e+00	1.001509e+00	1.001509e+00	1.001509e+00
min	-5.432896e+00	-4.008717e+00	-6.862210e+00	-3.425964e+00	-3.329983e+00	-4.522132e+00
25%	-4.308469e-01	-7.315610e-01	-6.247975e-01	-6.038933e-01	-6.922460e-01	-4.865560e-01
50%	2.338636e-01	1.130955e-01	-6.215825e-03	9.602421e-02	-8.216637e-03	1.566526e-01
75%	6.870477e-01	8.352597e-01	8.035682e-01	7.338032e-01	9.483444e-01	6.890494e-01
max	1.392514e+00	1.729687e+00	1.679672e+00	1.709577e+00	1.526095e+00	1.460408e+00

Εικόνα 4 – Στατιστικά μέτρα κανονικοποιημένων δεδομένων

Επιβεβαιώνεται από τα παραπάνω διαγράμματα ότι τα δεδομένα μας ικανοποιούν τις προϋποθέσεις που αναφέρθηκε πριν από την προεπεξεργασία τους. Όσον αφορά τις στήλες που απορρίψαμε στην αρχή (Channel και Region) ενδέχεται να περιέχουν την πληροφορία που να δείχνει πόσες ομάδες (clusters) ορίζουν τα δεδομένα μας. Για το λόγο αυτό επιλέγουμε να δούμε πόσες είναι οι διαφορετικές τιμές της στήλης *Region* και της *Channel*, αλλά και πόσες διαφορετικές δυάδες *(Region, Channel)* έχουμε.

```
Βρέθηκαν
        2 διαφορετικά κανάλια
        3 διαφορετικές περιοχές
        6 διαφορετικοί συνδυασμοί τους
```

Εικόνα 5 – Αποτελέσματα πιθανών ομάδων

Επομένως, είναι πιθανό να προκύψουν 2, 3 ή 6 διαφορετικές ομάδες από τα δεδομένα μας και αυτές είναι οι τιμές που θα εξετάσουμε κατά την

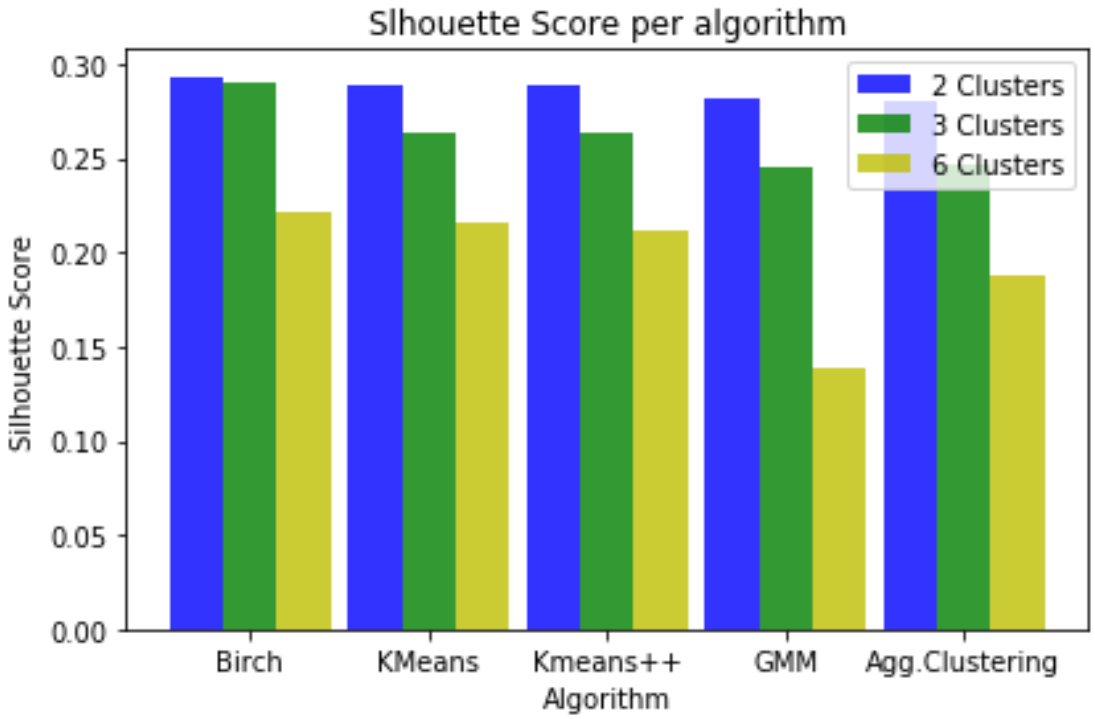
εκτέλεση των αλγορίθμων. Θα εφαρμόσουμε επομένως διάφορους αλγόριθμους ομαδοποίησης ζητώντας 2, 3 ή 6 ομάδες για να συγκρίνουμε τις επιδόσεις τους.

8.2. Παρουσίαση Αποτελεσμάτων

Σε αυτό το τμήμα της εργασίας θα γίνει σύγκριση του αλγόριθμου Birch σε σχέση με άλλους αλγορίθμους ομαδοποίησης. Η ενότητα αποτελείται από 2 υποενότητες. Στην πρώτη γίνεται σύγκριση και ανάλυση για την ομαδοποίηση των δεδομένων του Wholesale Customer Dataset, το οποίο αναλύθηκε παραπάνω. Στην δεύτερη παράγουμε συνθετικά dataset διάφορων μορφών και αξιολογούμε την ικανότητα του κάθε αλγορίθμου για την ομαδοποίηση των δεδομένων. Οι αλγόριθμοι που θα χρησιμοποιηθούν για τη συγκριτική αξιολόγηση του Birch είναι ο KMeans (με τυχαία αρχικοποίηση και με αρχικοποίηση του τύπου KMeans++), ο Gaussian Mixture και ο Agglomerative Hierarchical Clustering. Σημειώνεται ότι από όλους, μόνο ο αλγόριθμος KMeans δίνει δυνατότητα επιλογής αρχικοποίησης. Ο αλγόριθμος Birch, που υλοποιήθηκε και στα πλαίσια της εργασίας, εκτελείται απευθείας πάνω στα δεδομένα, χρησιμοποιώντας αυτά για αρχικοποίηση, όπως παρουσιάζεται στον Ψευδοκώδικα 7.

8.3. Ανάλυση με βάση το Wholesale customer dataset.

Όπως αναφέρθηκε παραπάνω για να εξετάσουμε την απόδοση των αλγορίθμων θα επιχειρήσουμε τον διαχωρισμό των δεδομένων σε 2, 3 ή 6 ομάδες. Για το λόγο αυτό τρέχουμε όλους τους αλγόριθμους ομαδοποίησης για το κάθε μέγεθος κλάσης ώστε να αξιολογήσουμε πόσο καλά γίνεται η κατάτμηση των δεδομένων της αγοράς και κατά πόσο έχει νόημα ο διαχωρισμός τους σε ομάδες.



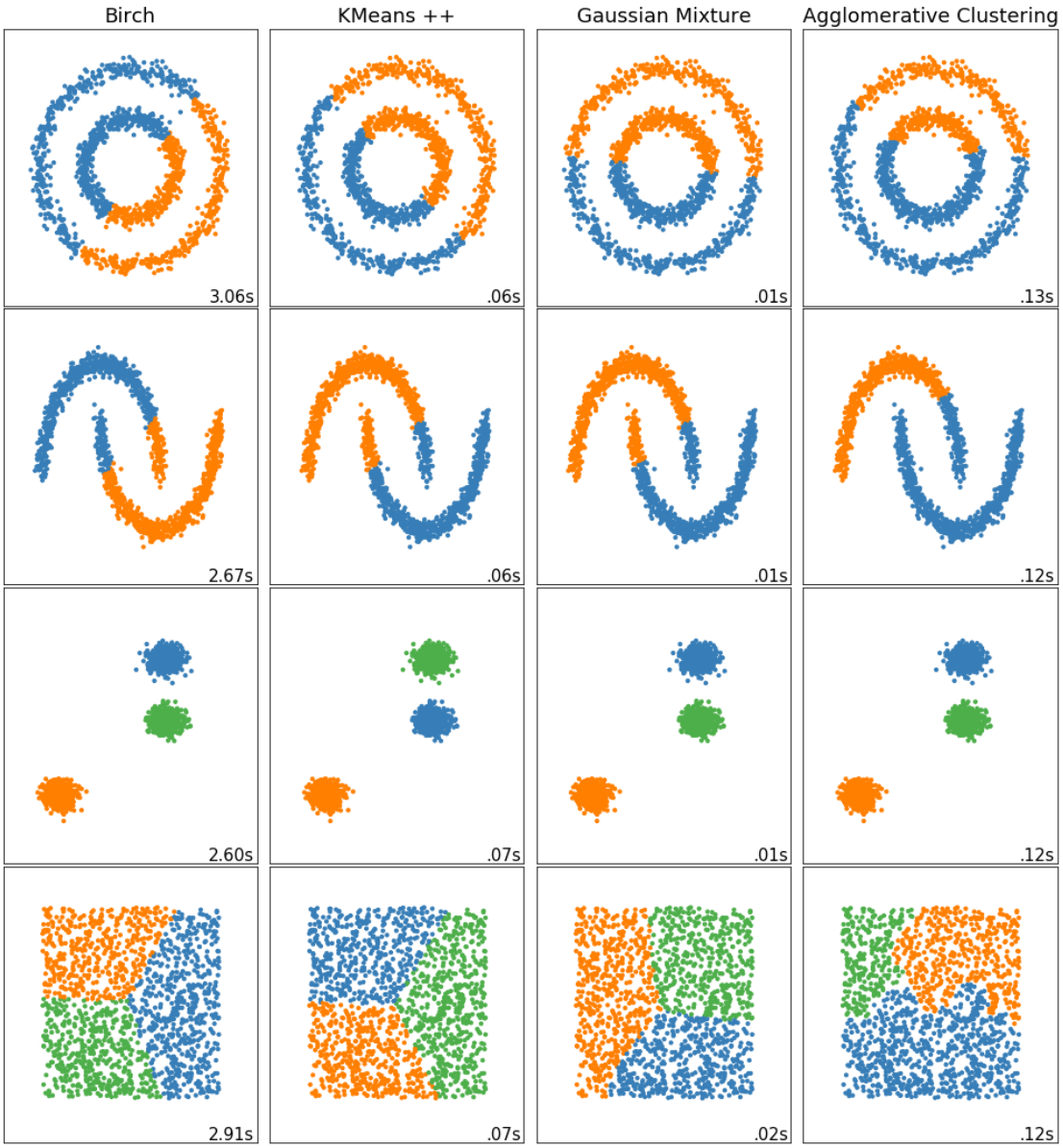
Εικόνα 6 - Μετρική Silhouette

Στην παραπάνω εικόνα φαίνεται η μετρική Silhouette που χρησιμοποιείται για να αξιολογήσει μη επιβλεπόμενους αλγόριθμους ομαδοποίησης για τα ζητούμενη πλήθη ομάδων και μια πληθώρα αλγορίθμων. Η μετρική εκφράζει το πόσο ομοιογενές είναι ένα στοιχείο εντός της ομάδας που ταξινομήθηκε και πόσο διαφέρει από τις υπόλοιπες ομάδες. Όσο πιο κοντά είναι η τιμή της στο 1 τόσο πιο καλή είναι η ομαδοποίηση που έχει επιτευχθεί. Στο διάγραμμα δίνεται η μέση τιμή της μετρικής για όλα τα στοιχεία του σετ δεδομένων. Αρχικά, παρατηρούμε ότι για την περίπτωση των 6 ομάδων η τιμή της μετρικής είναι αρκετά χαμηλή σε όλους τους αλγορίθμους (~20%) με αποτέλεσμα να καταλαβαίνουμε ότι δεν έχει ιδιαίτερη σημασία ο διαχωρισμός των δεδομένων σε 6 ομάδες. Στις περιπτώσεις των 2 και 3 ομάδων η μέση τιμή της μετρικής είναι κοντά στο 30% που είναι η ψηλότερη τιμή αναλογικά με των άλλων αλγορίθμων. Συμπεραίνουμε ότι ο Birch είναι ιδιαίτερα αποδοτικός αλγόριθμος που ανταποκρίνεται καλύτερα σε πραγματικά

δεδομένα. Ειδικά στην περίπτωση των 3 ομάδων είναι αισθητά φανερό ότι ο αλγόριθμος παρουσιάζει καλύτερη επίδοση.

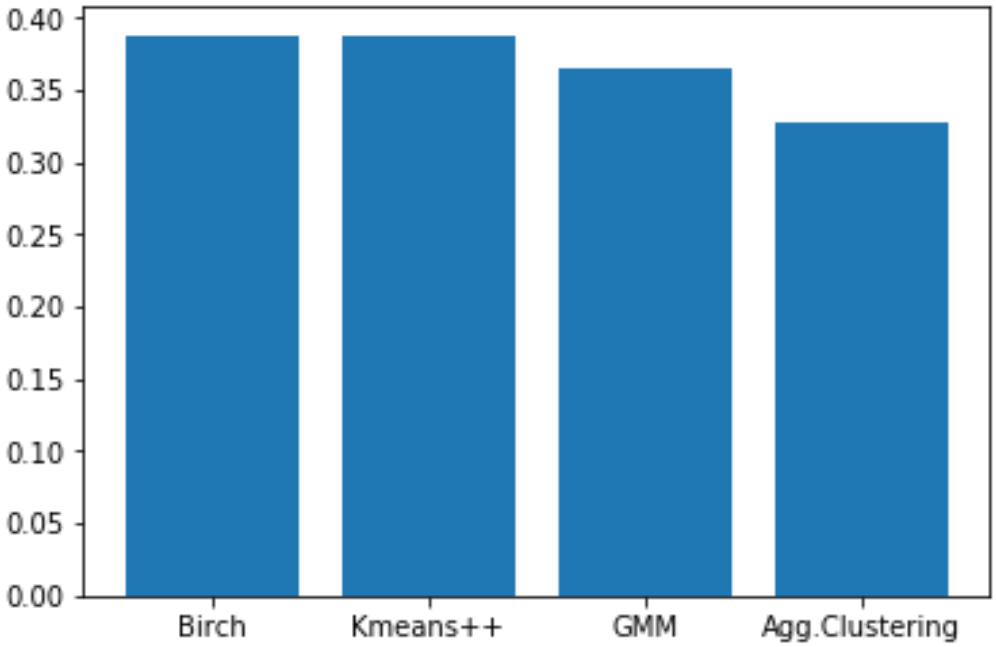
8.4. Ανάλυση με παραγόμενα δεδομένα διαφορετικών μορφών

Σε αυτήν την ενότητα παράγονται τυχαία dataset με ομαδοποιημένα δεδομένα όπου οι ομάδες σχηματίζουν διάφορα σχήματα. Εξετάζεται η ικανότητα των αλγορίθμων να ομαδοποιήσει και να αναγνωρίσει διάφορες μορφές ομάδων. Στο επόμενο σχήμα απεικονίζονται οι ομάδες με διαφορετικό χρώμα.



Εικόνα 7 - Ομαδοποίηση

Παρατηρούμε ότι για όλες τις μορφές ομαδοποιεί εξίσου καλά τα δεδομένα με τους άλλους αλγορίθμους. Για την περίπτωση των τυχαίων δεδομένων που έχουν ακανόνιστη μορφή και από το σχήμα τους δεν μπορούμε να αξιολογήσουμε την επίδοση μπορούμε να βγάλουμε συμπεράσματα και πάλι με την μετρική της σιλουέτας. Από το επόμενο διάγραμμα βλέπουμε ότι μέσω του Birch και του KMeans επιτυγχάνουμε την καλύτερη ομαδοποίηση.



Εικόνα 8 - Διάγραμμα ομαδοποίησης

9. Συμπεράσματα

Σκοπός της εργασίας ήταν η εφαρμογή του αλγορίθμου Birch στο πρόβλημα τμηματοποίησης αγοράς, χρησιμοποιώντας πραγματικά και τεχνητά δεδομένα. Τόσο στα πραγματικά όσο και στα τεχνητά δεδομένα παρατηρούμε ότι ο αλγόριθμος Birch μπορεί να ομαδοποιήσει τα δεδομένα είτε με τον ίδιο είτε με καλύτερο τρόπο σε σχέση με τους κλασσικούς αλγορίθμους ομαδοποίησης. Ειδικότερα αυτό έγινε εμφανές στο σετ δεδομένων που έχει προκύψει από πραγματικά δεδομένα. Σημειώνεται επίσης ότι ο αλγόριθμος δεν υστερεί σε σχέση με τους υπόλοιπους ούτε στο θέμα της επίδοσης, αφού παρατηρούμε ότι ο χρόνος εκτέλεσης του είναι παρόμοιας τάξης μεγέθους. Πιο αναλυτικά βλέπουμε κάποιες μικρές διαφορές σχετικά με τον χρόνο των άλλων αλγορίθμων, ειδικά στα τεχνητά κατασκευασμένα δεδομένα. Ωστόσο, αυτό οφείλεται στην επιβάρυνση του αλγορίθμου λόγω της εισαγωγής των στοιχείων στο δέντρο. Παρότι η διαδικασία δεν είναι χρονοβόρα, η επιβάρυνση που

προσθέτει είναι πιο έντονη σε σύνολα δεδομένων μικρού μεγέθους. Σε μεγαλύτερα σύνολα δεδομένων, στα οποία ενδείκνυται να τρέχει ο αλγόριθμος Birch, αυτή η επιβάρυνση είναι αμελητέα. Επομένως για αρκετά μικρά σύνολα δεδομένων ίσως να είναι προτιμότερο να τρέξει κάποιος από τους κλασσικούς αλγορίθμους ομαδοποίησης. Ωστόσο, ενδεχόμενες βελτιστοποίησης από μεριάς υλοποίησης να μπορούν να τον καθιστούν πιο γρήγορο, όπως παραλληλοποίηση του κώδικα, η οποία υφίσταται σε έτοιμες υλοποίησης βιβλιοθηκών με τις οποίες έχει συγκριθεί ο Birch στην παρούσα εργασία. Όλα τα παραπάνω τον καθιστούν ένα ικανότατο αλγόριθμο για προβλήματα που αφορούν την κατάτμηση της αγοράς εργασίας.

Βιβλιογραφία

Abbas, O. A. (2008). Comparisons Between Data Clustering Algorithms. Int. Arab J. Inf. Technol., 5(3), 320-325.

Aggarwal, C. C., & Reddy, C. K. (Eds.). (2013). Data clustering: algorithms and applications. CRC press.

Aiyer, A., Pyun, K. P., Huang, Y. Z., O'Brien, D. B., & Gray, R. M. (2005). Lloyd clustering of Gauss mixture models for image compression and classification. Signal Processing: Image Communication, 20(5), 459-485.

Amigó, E., Gonzalo, J., Artiles, J., & Verdejo, F. (2009). A comparison of extrinsic clustering evaluation metrics based on formal constraints. Information retrieval, 12(4), 461-486.

Ansari, S., Chetlur, S., Prabhu, S., Kini, G. N., Hegde, G., & Hyder, Y. (2013). An overview of clustering analysis techniques used in data mining. International Journal of Emerging Technology and Advanced Engineering, 3(12), 284-286.

Choi, S. S., Cha, S. H., & Tappert, C. C. (2010). A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics*, *8*(1), 43-48.

Du, HaiZhou and Li, YongBin (2010). An Improved BIRCH Clustering Algorithm and Application in Thermal Power. *IEEE 2010 International Conference on Web Information Systems and Mining*.

Gan, G., Ma, C., & Wu, J. (2007). Data clustering: theory, algorithms, and applications (Vol. 20). Siam.

Hofstede, F. T., Steenkamp, J. B. E., & Wedel, M. (1999). International market segmentation based on consumer-product relations. Journal of Marketing Research, 1-17.

Jain, A. K. (2010). Data clustering: 50 years beyond K-means. Pattern recognition letters, 31(8), 651-666.

Karpov, Ilya and Goroslavski, Alexandr (2012). Application of BIRCH to text clustering. *Proceedings of the 14th All-Russian Conference "Digital Libraries: Advanced Methods and Technologies, Digital Collections" — RCDL-2012*80

Liu, Y., Li, Z., Xiong, H., Gao, X., & Wu, J. (2010). Understanding of internal clustering validation measures. In Data Mining (ICDM), 2010 IEEE 10th International Conference on (pp. 911-916). IEEE.

McCallum, A., Nigam, K., & Ungar, L. H. (2000). Efficient clustering of high-dimensional data sets with application to reference matching. In Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 169-178). ACM.

Petrovic, S. (2006, October). A comparison between the silhouette index and the davies-bouldin index in labelling ids clusters. In Proceedings of the 11th Nordic Workshop of Secure IT Systems (pp. 53-64). sn.

Rendón, E., Abundez, I., Arizmendi, A., & Quiroz, E. M. (2011). Internal versus external cluster validation indexes. International Journal of computers and communications, 5(1), 27-34.

Rodriguez, A., & Laio, A. (2014). Clustering by fast search and find of density peaks. Science, 344(6191), 1492-1496.

Rokach, L., & Maimon, O. Z. (2008). *Data mining with decision trees: theory and applications* (Vol. 69). World Scientific.

Stuetzle, W., & Nugent, R. (2010). A generalized single linkage method for estimating the cluster tree of a density. Journal of Computational and Graphical Statistics, 19(2), 397-418.

Tran, T. N., Drab, K., & Daszykowski, M. (2013). Revised DBSCAN algorithm to cluster data with dense adjacent clusters. Chemometrics and Intelligent Laboratory Systems, 120, 92-96.

Xu, R., & Wunsch, D. (2005). Survey of clustering algorithms. IEEE Transactions on neural networks, 16(3), 645-678.

Yankelovich, D., & Meer, D. (2006). Rediscovering market segmentation. *Harvard business review*, *84*(2), 122.

Zhang, T., Ramakrishnan, R., & Livny, M. (1997). BIRCH: A new data clustering algorithm and its applications. Data Mining and Knowledge Discovery, 1(2), 141-182.

Zhang, Tian, Raghu Ramakrishan, and Miron Livny. 1996. "BIRCH: An Efficient Data Clustering Method for Very Large Databases." In Management of Data, 103–14. doi:10.1145/233269.233324.

Zhao, Y., Karypis, G., & Fayyad, U. (2005). Hierarchical clustering algorithms for document datasets. Data mining and knowledge discovery, 10(2), 141-168.

Παράρτημα: Κώδικας Python που εκτελέστηκε για τις ανάγκες της εργασίας

```
#!/usr/bin/env python

import math
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

class CF(object):
    # Constructor κλάσης CF
    def __init__(self, N, linear_sum, squared_sum):
        self._N = N
        self._LS = linear_sum
        self._SS = squared_sum

    # Μέθοδος που επιστρέφει ένα άδειο CF, για προκαθορισμένης
διάστασης διανυσματικό χώρο
    @classmethod
    def empty_cf(cls, shape):
        return cls(0, np.zeros(shape), 0)

    @property
    def N(self):
        return self._N

    @property
    def LS(self):
        return self._LS

    @property
    def SS(self):
        return self._SS

    # Πρόσθεση του δεδομένου CF με ένα νέο που δίνεται ως όρισμα
    def add(self, newCF):
        return CF(self._N + newCF.N, self._LS + newCF.LS, self._SS +
newCF.SS)

    # Δημιουργία "αρνητικού" CF
    def neg(self):
        return CF(-self._N, -self._LS, -self._SS)

    # Συνάρτηση απεικόνισης του CF ως συμβολοσειρά
    def toString(self):
        return '(%d, %s, %s)'% (self._N,
         np.array2string(self._LS, separator=',', prefix='['),
         np.array2string(self._SS, separator=',', prefix='['))


class CFNode(object):
    def __init__(self, cf, parent, branching_factor, leaf_size,
threshold):
        self._cf = cf
        self._children = []
```

```
        self._parent = parent
        self._branching_factor = branching_factor
        self._leaf_size = leaf_size
        self._threshold = threshold

    @property
    def CF(self):
        return self._cf

    @property
    def parent(self):
        return self._parent

    @parent.setter
    def parent(self, parent):
        self._parent = parent

    @property
    def children(self):
        return self._children

    @property
    def branching_factor(self):
        return self._branching_factor

    @property
    def leaf_size(self):
        return self._leaf_size

    @property
    def threshold(self):
        return self._threshold

    @classmethod
    def get_initialized_root(cls, datapoint, branching_factor,
leaf_size, threshold):
        cf = CF(1, datapoint, np.dot(datapoint, datapoint))
        dummy_leaf = CFLeaf(CF.empty_cf(cf.LS.shape), None,
branching_factor, leaf_size, threshold)
        root = cls(cf, None, branching_factor, leaf_size, threshold)
        root.children.append(CFLeaf(cf, root, branching_factor,
leaf_size, threshold))
        root.children[0].data_points.append(datapoint)
        dummy_leaf.next_leaf = root.children[0]
        root.children[0].prev_leaf = dummy_leaf
        return root, dummy_leaf

    @classmethod
    def get_new_root(cls, splitted_root):
        new_root = cls(CF.empty_cf(splitted_root[0].CF.LS.shape),
None, splitted_root[0].branching_factor,
                       splitted_root[0].leaf_size,
splitted_root[0].threshold)
        splitted_root[0].parent = new_root
        splitted_root[1].parent = new_root
        new_root._add_new_child(splitted_root[0])
        new_root._add_new_child(splitted_root[1])
        return new_root

    def toString(self, level):
        retString = str(level) + ""
```

```
        for i in range(0, level):
            retString = retString + "\t"
        retString = retString + "CF : " + self._cf.toString() + "\n"
        for child in self._children:
            retString = retString + child.toString(level + 1)
        return retString

    def centroid(self):
        return (self._cf.LS/self._cf.N)

    def get_radius(self):
        return (np.sqrt(np.divide(self._cf.SS -
np.divide(np.dot(self._cf.LS, self._cf.LS), self._cf.N),
self._cf.N)))

    # Εύρεση απόστασης ενός σημείου από το κέντρο της δεδομένης
κλάσης
    def _get_distance(self, datapoint):
        return np.linalg.norm(self.centroid() - datapoint)

    # Ενημέρωση της τιμής του CF του κόμβου με βάση ένα νέο
χαρακτηριστικό.
    def _update_cf(self, newCF):
        self._cf = self._cf.add(newCF)
        return self

    def _add_new_child(self, CFnode):
        self._children.append(CFnode)
        self._update_cf(CFnode.CF)
        return self

    def _get_closer_child (self, datapoint):
        distances = list(map(lambda x :
self._get_distance(datapoint), self._children))
        return np.argmin(distances)

    def _get_furthest_pair(self, cf_points):
        def _get_distance(point):
            return np.sqrt(np.sum(np.square(cf_points - point),
axis=1))

        distances = np.apply_along_axis(_get_distance, 1, cf_points)
        return np.unravel_index(np.argmax(distances),
distances.shape)

    def _needs_split(self):
        return (len(self._children) > self._branching_factor)

    def _split_node(self):
        cf_array = []
        for child in self._children:
            cf_array.append(child.CF.LS/child.CF.N)
        cf_array = np.asarray(cf_array)
        id1, id2 = self._get_furthest_pair(cf_array)
        emptyCF = CF.empty_cf(self._cf.LS.shape)
        newNode1 = CFNode(cf=emptyCF, parent=self._parent,
branching_factor=self._branching_factor,
                          leaf_size=self._leaf_size,
threshold=self._threshold)
        newNode2 = CFNode(cf=emptyCF, parent=self._parent,
branching_factor=self._branching_factor,
```

```
                            leaf_size=self._leaf_size,
threshold=self._threshold)

        for child in self._children:
            if self._children[id1]._get_distance(child.centroid()) <=
self._children[id2]._get_distance(child.centroid()):
                newNode1._add_new_child(child)
            else:
                newNode2._add_new_child(child)

        if self._parent is None:
            return CFNode.get_new_root([newNode1, newNode2])
        else:
            return [newNode1, newNode2]


    def insert(self, datapoint):
        idx = self._get_closer_child(datapoint)
        self._update_cf(CF(1, datapoint, np.dot(datapoint,
datapoint)))
        check_split, returner = self.children[idx].insert(datapoint)
        if check_split:
            self._update_cf(self.children[idx].CF.neg())
            del self._children[idx]
            self._add_new_child(returner[0])
            self._add_new_child(returner[1])
            if self._needs_split():
                return True, self._split_node()
        return False, self


class CFLeaf(CFNode):
    def __init__(self, cf, parent, branching_factor, leaf_size,
threshold):
        super().__init__(cf, parent, branching_factor, leaf_size,
threshold)
        self._data_points = []
        self._next_leaf = None
        self._prev_leaf = None

    @property
    def next_leaf(self):
        return self._next_leaf

    @next_leaf.setter
    def next_leaf(self, next_leaf):
        self._next_leaf = next_leaf

    @property
    def prev_leaf(self):
        return self._prev_leaf

    @prev_leaf.setter
    def prev_leaf(self, prev_leaf):
        self._prev_leaf = prev_leaf


    @property
    def data_points(self):
        return self._data_points
```

```
    def toString(self, level):
        retString = str(level) + ""
        tabs = ""
        for i in range(0, level):
            tabs = tabs + "\t"
        retString = retString + tabs + "CF : " + self._cf.toString()
+ "\n"
        for point in self._data_points:
            retString = retString + "" + tabs + "Inner Point : " +
np.array2string(point, separator=',', prefix='[') + "\n"

        return retString

    def _compute_distance_from_datapoint(self, idx, point):
        return np.linalg.norm(self._data_points[idx] - point)

    def _add_new_datapoint(self, point):
        self._data_points.append(point)
        cf = CF(1, point, np.dot(point, point))
        self._update_cf(cf)
        return self

    def _needs_split(self):
        return (self.get_radius() > self._threshold or
len(self._data_points) > self._leaf_size)

    def _split_node(self):
        cf_array = []
        for point in self._data_points:
            cf_array.append(point)
        cf_array = np.asarray(cf_array)

        id1, id2 = self._get_furthest_pair(cf_array)
        emptyCF = CF.empty_cf(self._cf.LS.shape)
        newNode1 = CFLeaf(cf=emptyCF, parent=self._parent,
branching_factor=self._branching_factor,
                          leaf_size=self._leaf_size,
threshold=self._threshold)
        newNode2 = CFLeaf(cf=emptyCF, parent=self._parent,
branching_factor=self._branching_factor,
                          leaf_size=self._leaf_size,
threshold=self._threshold)

        for point in self._data_points:
            if self._compute_distance_from_datapoint(id1, point) <=
self._compute_distance_from_datapoint(id2, point):
                newNode1._add_new_datapoint(point)
            else:
                newNode2._add_new_datapoint(point)
        newNode1.next_leaf = newNode2
        newNode2.prev_leaf = newNode1
        newNode1.prev_leaf = self._prev_leaf
        newNode2.next_leaf = self._next_leaf
        if self._next_leaf is not None:
            self._next_leaf.prev_leaf = newNode2
        self._prev_leaf.next_leaf = newNode1
        self._next_leaf = None
        self._prev_leaf = None
        return [newNode1, newNode2]

    def insert(self, datapoint):
```

```
        self._add_new_datapoint(datapoint)
        if self._needs_split():
            return True, self._split_node()
        return False, self


from sklearn.base import BaseEstimator, ClusterMixin,
TransformerMixin
from sklearn.cluster import AgglomerativeClustering

class Birch(BaseEstimator, ClusterMixin, TransformerMixin):
    def __init__(self, branching_factor, leaf_size, threshold,
n_clusters):
        self._B = branching_factor
        self._L = leaf_size
        self._T = threshold
        self._n_clusters = n_clusters
        self._centroids = None
        self._labels_ = None

    @property
    def labels_(self):
        return self._labels_

    def _assign_to_cluster(self, point):
        point = np.tile(point, (len(self._centroids),1))
        distances = np.linalg.norm(point - self._centroids, axis=1)
        return np.argmin(distances)

    def _get_cf_tree(self, X):
        rootNode, dummy = CFNode.get_initialized_root(X[0, :],
self._B, self._L, self._T)
        for index in range(1, len(X)):
            _, rootNode = rootNode.insert(X[index, :])
        return rootNode, dummy

    def _get_final_centroid(self, X, y, i):
        return np.mean(X[np.where(y == i)[0], :], axis=0)

    def fit(self, X, y=None):
        _, dummy = self._get_cf_tree(X)
        temp = dummy
        centroids = []
        while temp.next_leaf is not None:
            temp = temp.next_leaf
            centroids.append(temp.centroid())
        centroids = np.asarray(centroids)
        agglomerative_clustering =
AgglomerativeClustering(n_clusters=self._n_clusters).fit(centroids)
        assignments = agglomerative_clustering.labels_
        clusters = np.unique(assignments)
        self._centroids = np.array(list(map(lambda x :
self._get_final_centroid(centroids, assignments, x),
clusters)))
        self._labels_ = self.predict(X)
        return self

    def predict(self, X):
        return np.apply_along_axis(self._assign_to_cluster, 1, X)

    def fit_predict(self, X, y=None):
```

```
        self.fit(X, y)
        return self._labels_

wholesale_cust_data = pd.read_csv("./data/Wholesale customers
data.csv")
wholesale_cust_data.head()


trainset = wholesale_cust_data.iloc[: ,2:len(wholesale_cust_data)]
trainset.head()


trainset.hist(bins=50, figsize=(20, 15))
plt.show()


pd.plotting.scatter_matrix(trainset,
alpha=0.3,figsize=(15,8),diagonal='kde')
plt.show()


stats = trainset.describe()
stats

def remove_outliers_tukey(df, stats, colname):
    q1 = stats[colname]["25%"]
    q3 = stats[colname]["75%"]
    step = q3-q1
    keep_rows = df.where((df[colname] > q1 - 1.5* step) &
(df[colname] < q3 + 1.5* step))
    keep_rows = keep_rows.dropna()
    return keep_rows

cleaned_trainset = trainset
for colname in trainset.columns.values:
    cleaned_trainset = remove_outliers_tukey(cleaned_trainset, stats,
colname)


from sklearn.preprocessing import StandardScaler

cleaned_trainset_normalized =
StandardScaler().fit_transform(np.log(cleaned_trainset))
cleaned_trainset_normalized =
pd.DataFrame(cleaned_trainset_normalized)
cleaned_trainset_normalized.head()

cleaned_trainset_normalized.hist(bins=50, figsize=(20, 15))
plt.show()


pd.plotting.scatter_matrix(cleaned_trainset_normalized,
alpha=0.3,figsize=(15,8),diagonal='kde')
plt.show()
stats = cleaned_trainset_normalized.describe()
stats


indexing_columns = wholesale_cust_data.iloc[: ,0:2]
unique_channels = len(indexing_columns.Channel.unique())
unique_regions = len(indexing_columns.Region.unique())
```

```
unique_pairs = len(indexing_columns.drop_duplicates())
print("Βρέθηκαν\n\t%d διαφορετικά κανάλια\n\t%d διαφορετικές
περιοχές\n\t%d διαφορετικοί συνδυασμοί τους"
      %(unique_channels, unique_regions, unique_pairs))

from sklearn.cluster import KMeans
from sklearn.cluster import AgglomerativeClustering
from sklearn.mixture import GaussianMixture
from sklearn.cluster import DBSCAN
from sklearn import metrics
import time

def silhouette_score(model, data):
    labels = model.fit_predict(data)
    return metrics.silhouette_score(data, labels)


results = []

models = []
models.append(Birch(branching_factor=4, leaf_size=10,  threshold=0.7,
n_clusters=2))
models.append(KMeans(n_clusters=2, init="random"))
models.append(KMeans(n_clusters=2, init="k-means++"))
models.append(GaussianMixture(n_components=2))
models.append(AgglomerativeClustering(n_clusters=2))

silhouette_scores = list(map(lambda x : silhouette_score(x,
cleaned_trainset_normalized.values), models))
results.append(silhouette_scores)


models = []
models.append(Birch(branching_factor=7, leaf_size=30, threshold=0.8,
n_clusters=3))
models.append(KMeans(n_clusters=3, init="random"))
models.append(KMeans(n_clusters=3, init="k-means++"))
models.append(GaussianMixture(n_components=3))
models.append(AgglomerativeClustering(n_clusters=3))

silhouette_scores = list(map(lambda x : silhouette_score(x,
cleaned_trainset_normalized.values), models))
results.append(silhouette_scores)


models = []
models.append(Birch(branching_factor=2, leaf_size=30, threshold=0.6,
n_clusters=6))
models.append(KMeans(n_clusters=6, init="random"))
models.append(KMeans(n_clusters=6, init="k-means++"))
models.append(GaussianMixture(n_components=6))
models.append(AgglomerativeClustering(n_clusters=6))

silhouette_scores = list(map(lambda x : silhouette_score(x,
cleaned_trainset_normalized.values), models))
results.append(silhouette_scores)


results = np.array(results)

# data to plot
```

```
n_groups = 5

# create plot
fig, ax = plt.subplots()
index = np.array([1,2,3,4,5])
bar_width = 0.3
opacity = 0.8

Rects2Cluster = plt.bar(index, results[0, :], bar_width,
                 alpha=opacity,
                 color='b',
                 label='2 Clusters')

Rects3Cluster = plt.bar(index + bar_width, results[1, :], bar_width,
                 alpha=opacity,
                 color='g',
                 label='3 Clusters')

Rects4Cluster = plt.bar(index + 2*bar_width, results[2, :],
bar_width,
                 alpha=opacity,
                 color='y',
                 label='6 Clusters')

plt.xlabel('Algorithm')
plt.ylabel('Silhouette Score')
plt.title('Slhouette Score per algorithm')
plt.xticks(index + bar_width, (["Birch", "KMeans", "Kmeans++", "GMM",
"Agg.Clustering"]))
plt.legend()

plt.tight_layout()
plt.show()

from sklearn import datasets
from itertools import cycle, islice

np.random.seed(0)

from sklearn.datasets import make_blobs
n_samples = 1500
noisy_circles = datasets.make_circles(n_samples=n_samples, factor=.5,
                                      noise=.05)
noisy_moons = datasets.make_moons(n_samples=n_samples, noise=.05)
blobs = datasets.make_blobs(n_samples=n_samples, random_state=8)
no_structure = np.random.rand(n_samples, 2), None


# In[23]:


import warnings
# Set up cluster parameters
plt.figure(figsize=(9 * 1.3 + 2, 14.5))
plt.subplots_adjust(left=.02, right=.98, bottom=.001, top=.96,
wspace=.05,
                    hspace=.01)

plot_num = 1

default_base = {'n_neighbors': 10,
```

```
                'n_clusters': 3}

datasets = [
    (noisy_circles, {'n_clusters': 2}),
    (noisy_moons, {'n_clusters': 2}),
    (blobs, {}),
    (no_structure, {})]

for i_dataset, (dataset, algo_params) in enumerate(datasets):
    # update parameters with dataset-specific values
    params = default_base.copy()
    params.update(algo_params)

    X, y = dataset

    # normalize dataset for easier parameter selection
    X = StandardScaler().fit_transform(X)

    birch = Birch(branching_factor=50, leaf_size=100, threshold=0.85,
n_clusters=params['n_clusters'])
    kmeans = KMeans(n_clusters=params['n_clusters'], init="k-
means++")
    gaussian_mix = GaussianMixture(n_components=params['n_clusters'])
    agg_cluster =
AgglomerativeClustering(n_clusters=params['n_clusters'])

    clustering_algorithms = (
        ('Birch', birch),
        ('KMeans ++', kmeans),
        ('Gaussian Mixture', gaussian_mix),
        ('Agglomerative Clustering', agg_cluster),
    )

    for name, algorithm in clustering_algorithms:
        t0 = time.time()

        # catch warnings related to kneighbors_graph
        with warnings.catch_warnings():
            warnings.filterwarnings(
                "ignore",
                message="the number of connected components of the "
+
                "connectivity matrix is [0-9]{1,2}" +
                " > 1. Completing it to avoid stopping the tree
early.",
                category=UserWarning)
            algorithm.fit(X)

        t1 = time.time()
        if hasattr(algorithm, 'labels_'):
            y_pred = algorithm.labels_.astype(np.int)
        else:
            y_pred = algorithm.predict(X)

        plt.subplot(len(datasets), len(clustering_algorithms),
plot_num)
        if i_dataset == 0:
            plt.title(name, size=18)

        colors = np.array(list(islice(cycle(['#377eb8', '#ff7f00',
'#4daf4a',
```

```
                                            '#f781bf', '#a65628',
'#984ea3',
                                            '#999999', '#e41a1c',
'#dede00']),
                                     int(max(y_pred) + 1))))
        plt.scatter(X[:, 0], X[:, 1], s=10, color=colors[y_pred])

        plt.xlim(-2.5, 2.5)
        plt.ylim(-2.5, 2.5)
        plt.xticks(())
        plt.yticks(())
        plt.text(.99, .01, ('%.2fs' % (t1 - t0)).lstrip('0'),
                 transform=plt.gca().transAxes, size=15,
                 horizontalalignment='right')
        plot_num += 1

plt.show()


models = []
models.append(Birch(branching_factor=50, leaf_size=100,
threshold=0.85, n_clusters=3))
models.append(KMeans(n_clusters=3, init="k-means++"))
models.append(GaussianMixture(n_components=3))
models.append(AgglomerativeClustering(n_clusters=3))

silhouette_scores = list(map(lambda x : silhouette_score(x, X),
models))
index = [1,2,3,4]
plt.bar(index, silhouette_scores)
plt.xticks(index, (["Birch", "Kmeans++", "GMM", "Agg.Clustering"]))
```